Osiguranje kvalitete grafičkih korisničkih sučelja

Beljan, Ivan

Master's thesis / Diplomski rad

2022

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Economics and Business / Sveučilište u Zagrebu, Ekonomski fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:148:730052

Rights / Prava: Attribution-NonCommercial-ShareAlike 3.0 Unported/Imenovanje-Nekomercijalno-Dijeli pod istim uvjetima 3.0

Download date / Datum preuzimanja: **2025-02-19**

Repository / Repozitorij:

REPEFZG - Digital Repository - Faculty of Economcs & Business Zagreb

Sveučilište u Zagrebu

Ekonomski fakultet

Integrirani preddiplomski i diplomski sveučilišni studij

Poslovna ekonomija - smjer Menadžerska informatika

OSIGURANJE KVALITETE GRAFIČKIH KORISNIČKIH SUČELJA

Diplomski rad

Ivan Beljan

Zagreb, rujan 2022.

Sveučilište u Zagrebu

Ekonomski fakultet

Integrirani preddiplomski i diplomski sveučilišni studij

Poslovna ekonomija - smjer Menadžerska informatika

OSIGURANJE KVALITETE GRAFIČKIH KORISNIČKIH SUČELJA

QUALITY ASSURANCE IN GRAPHICAL USER INTERFACES

Diplomski rad

Student: Ivan Beljan

JMBAG studenta: 0067541546

Mentor: izv. prof., dr. sc. Nikola Vlahović

Zagreb, rujan 2022.

SAŽETAK

Razvoj digitalne tehnologije uvjetuje promjenu načina interakcije čovjeka i računala. Suvremena komunikacija s računalnim sustavima ostvaruje se putem korisničkih sučelja. Grafička korisnička sučelja (engl. *graphical user interface - GUI)* su najzastupljeniji oblik korisničkih sučelja. Prilagođena su računalnim mogućnostima prikaza svih poznatih vrsta uređaja. Upotrebljivost grafičkih korisničkih sučelja zasniva se na preglednom prikazu informacija integracijom teksta, vizualnih, interaktivnih te stiliziranih elemenata.

Sukladno rastu važnosti digitalne tehnologije u ljudskom životu, mijenja se i percepcija kvalitete digitalnih dobara. Osiguranje zadovoljavajuće razine kvalitete digitalnih proizvoda, provodi se sistematiziranim skupom aktivnosti - testiranjem. Primjenom odgovarajućih metodologija testiranja grafičkih korisničkih sučelja, ovjerava se stupanj zadovoljenja postavljenih zahtjeva. Postavljeni korisnički i sistemski zahtjevi definiraju referentni okvir željenog ponašanja proizvoda. Upotrebljivost grafičkih korisničkih sučelja testira se pripadajućom metodom testiranja (engl. *usability testing).* Ključni elementi *usability testing* metode su provjera lakoće upravljanja, intuitivnosti i efikasnosti izvođenja operacija sustava.

Ciljevi rada se očituju u postavljanju teorijskih pretpostavki *GUI-a,* percepcije kvalitete, metodologija i pristupa osiguranja kvalitete te primjeni teorijskih načela na odabranim primjerima. Očitovanje svrhe rada postiže se ispunjavanjem postavljenih ciljeva i donošenjem zaključka o idealnom pristupu testiranja *GUI-a*. Koristeći vlastiti prototip aplikacije, *OneKey*, prikazane su mogućnosti manualnog testiranja aplikacije. Na primjeru pojedinih usluga *Google* servisa, demonstrirane su sposobnosti automatskog testiranja softvera.

Objedinjenjem postignutih ciljeva, izvodi se konačni sud o mogućnostima testiranja grafičkih korisničkih sučelja i pripadajućih elemenata. Na temelju provedenih testiranja, stvoren je zaključak o primjenjivosti pojedine metodologije testiranja i testnih tehnika za ovjeru grafičkih korisničkih sučelja.

Ključne riječi: interakcija čovjeka i računala, grafička korisnička sučelja, osiguranje kvalitete, manualno testiranje, automatsko testiranje

SUMMARY

The development of digital technology modified human-computer interaction throughout history. Modern computer-based communication relies on user interfaces, *UIs*. Graphical user interfaces, *GUIs*, are the most popular form of user interfaces. They are adapted to various device display specifications. The usability of graphical user interfaces is based on displaying information transparently, by integrating text with visual, interactive and stylized elements.

The soaring importance of digital technology, adjusted the perceived quality requirements of digital goods. As a systematized set of activities, software testing ensures impeccable quality levels of digital products. Verification of settled product requirements is attained by applying appropriate *GUI* testing methodologies. Concluded user and system requirements define the desired product specifications and behavior. Usability of graphical user interfaces is tested with a proper testing approach - usability testing. The key elements of the usability testing methodology are management simplicity, intuitiveness and system operation performance.

The objectives of this thesis manifest by setting theoretical starting points for *GUIs*, quality perception, quality assurance methodologies and their demonstration by using specific examples. Fulfilled objectives, and concluding the ideal *GUI* testing approach, reveal the main purpose of the thesis. The capacity of the manual testing approach is displayed by testing a self-built application prototype, *OneKey*. Using specific *Google* services, enabled the demonstration of software testing automation capabilities.

Combining the met objectives with the testing process, draws a conclusion about the possibilities of *GUI* testing methodologies. The conclusion refers to the applicability of certain testing methodologies and test techniques for graphical user interface verification.

Keywords: human-computer interaction, graphical user interface, quality assurance, manual testing, test automation

IZJAVA O AKADEMSKOJ ČESTITOSTI

Izjavljujem i svojim potpisom potvrđujem da je diplomski rad isključivo rezultat mog vlastitog rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu, a što pokazuju korištenje bilješke i bibliografija.

Izjavljujem da nijedan dio rada nije napisan na nedozvoljen način, odnosno da je prepisan iz necitiranog izvora te da nijedan dio rada ne krši bilo čija autorska prava.

Izjavljujem, također, da nijedan dio rada nije iskorišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.

_____

(vlastoručni potpis studenta)

_____

(mjesto i datum)

STATEMENT OF THE ACADEMIC INTEGRITY

I hereby declare and confirm by my signature that the final thesis is the sole result of my own work based on my research and relies on the published literature, as shown in the listed notes and bibliography.

I declare that no part of the thesis has been written in an unauthorized manner, i.e., it is not transcribed from the non-cited work, and that no part of the thesis infringes any of the copyrights.

I also declare that no part of the thesis has been used for any other work in any other higher education, scientific or educational institution.

_____

(personal signature of the student)

_____

(place and date)

SADRŽAJ

1. UVOD

Interakcija računala i čovjeka postala je uobičajena pojava. Digitalna transformacija je doprinijela integraciji računalnih sustava u sve sfere ljudskog života, olakšavajući privatne i poslovne procese. Inovacije i sveopći napredak mijenjaju način na koji ljudi upravljaju računalima. Iako se načini upravljanja uređajima mijenjaju, obilježeni su jednim zajedničkim pojmom - korisničkim sučeljem. Korisničko sučelje je medijator ljudsko-računalne interakcije. Ono omogućuje prijenos ljudskog stvaralaštva u digitalnu sferu. Uspješnost informacijske razmjene računala i čovjeka, počiva na kvaliteti koja se takvom interakcijom može postići.

Osiguranje kvalitete digitalnih dobara ostvaruje se provođenjem specifičnog skupa aktivnosti - testiranjem. Da bi se osigurala što veća razina kvalitete interakcije, potrebno je provesti ciljano testiranje korisničkog sučelja kao ključnog čimbenika komunikacije. Fokus ovog rada je u prikazu efikasnog procesa testiranja grafičkih korisničkih sučelja na različitim primjerima. Ključne aktivnosti testiranja obuhvaćaju razradu objekta testiranja, kvalitetnu pripremu pretpostavki te provedbu testnog procesa.

Ovjera kvalitete korisničkih sučelja bit će provedena odgovarajućim testnim tehnikama uz pomoć manualnog i automatskog testiranja. Oba pristupa su u određenoj mjeri primjenjiva za testiranje korisničkih sučelja. Ovisno o provedenim testnim aktivnostima, primjenom obiju metodologija, izvest će konačni sud o mogućnostima i specifikacijama pojedinog pristupa za testiranje grafičkih korisničkih sučelja.

1.1. Predmet i cilj rada

Predmet rada očituje se u provedbi metodologija manualnog i automatskog testiranja na specifičnim primjerima. Cilj provođenja testnih aktivnosti je donošenje krajnjeg osvrta o efikasnosti, ulozi i utjecaju pojedine metodologije prilikom testiranja grafičkih korisničkih sučelja. Konačni cilj je moguće raščlaniti na manje korake:

1. Raščlamba pojma ljudsko-računalne interakcije, temeljene na primjeni raznovrsnih korisničkih sučelja.
2. Prikaz povijesnog razvoja percepcije kvalitete i njen utjecaj na tržište digitalnih dobara.
3. Predstavljanje metodologija osiguranja kvalitete softvera temeljenih na skupu normi i standarda, okupljenih oko pojma informatičke sigurnosti.

4. Usporedna raščlamba temeljnih pristupa testiranja softvera, navodeći slabosti i snage pojedinog pristupa.
5. Demonstracija provedbe testnog procesa ovjere grafičkog korisničkog sučelja, primjenom manualnog i automatskog testiranja.

Prolaskom navedenih koraka, dolazi se do cjelovitog prikaza testiranja grafičkih korisničkih sučelja. Provođenje testnih koraka omogućit će filtraciju najboljih praksi testiranja. Cjelokupni proces će rezultirati donošenjem zaključka o korištenju odgovarajuće metodologije i pripadajućih testova, radi efikasnog osiguranja kvalitete grafičkih korisničkih sučelja.

1.2. Metode istraživanja

Izvor podataka teorijskog dijela rada počiva na relevantnoj literaturi. Ovisno o obrađenom poglavlju, koristi se pripadajuća literatura u obliku knjiga, članaka, ali i informacija prikupljenih pohađanjem tečajeva.

Istraživanje se temelji na primjeni metodologija manualnog i automatskog testiranja na odabranim primjerima. Manualno testiranje će se provesti korištenjem prototipa aplikacije stvorenog za potrebe demonstracije navedene metodologije. Automatizacija procesa testiranja bit će prikazana na primjerima usluga *Google* servisa. Navedeni servis je odabran kao testni objekt zbog njegove dostupnosti i popularnosti.

2. INTERAKCIJA ČOVJEKA I RAČUNALA

Ljudska svakodnevica neizbježno uključuje interakciju s raznim uređajima. Od buđenja uz pomoć naprednih budilica do života u pametnim kućama - čovjek konstantno prilagođava vlastitu okolinu trendovima digitalne tehnologije (Spremić, 2017). Smislenu integraciju digitalnih pomagala u ljudske živote omogućuje kvalitetna interakcija čovjeka i stroja. Brojni stručnjaci polja interakcijskog dizajna, usmjerenih na razvoj, oblikovanje i unapređenje korisničkog iskustva (*user experience - UX)*, omogućili su da jednostavno primjenjivi proizvodi svakodnevno doprinose kvaliteti života (Sharp et al., 2019). Sukladno važnosti koju nosi ljudsko-strojna interakcija, u nastavku će biti prikazan povijesni razvoj, vrste i elementi sučelja koja omogućuju simbiozu čovjeka i tehnologije.

2.1. Povijesni razvoj upravljanja računalima

Mogućnosti upravljanja računalima se intenzivno mijenjaju kroz povijest. Iako računalno provedene operacije postaju složenije, upravljanje njima postaje jednostavnije. Abakus je primjer preteče pomagala za obavljanje temeljnih računskih operacija. Koncept abakusa primjenjuje se za razvoj prvih kalkulatora i računala. Prvo programabilno računalo temeljilo se na bušenim karticama, po uzoru na tkalačke stanove iz 19. stoljeća. Interakcija s računalom odvijala se izmjenom kartica, kojima se prilagođavaju željene strojne operacije (Vlahović & Pivar, 2020).

Sredinom 20. stoljeća ostvaruju se brojne inovacije koje unapređuju način na koji čovjek upravlja računalom. Pojavljuju se strojevi poput MARK-I i Turingova Colossusa, čiju primjenu olakšavaju integrirani prekidači za upravljanje sustavom. Pravi proboj na komercijalno tržište ostvario se pojavom računala s integriranim zaslonom i skupom gumba, ENIAC i UNIVAC. Napredak računala postaje nezaustavljiv, dijelovi sve manji, a broj korisnika sve veći. Unapređenjem mikroprocesora omogućeno je okupljanje svih važnih komponenti računala na izrazito malim plohama (Vlahović & Pivar, 2020). Manji procesori omogućili su proizvodnju manjih uređaja za koje se oblikuju nove metode interakcije. Tipkovnica i miš su mjerilo efikasnog unosa podataka u računalo no zbog uzleta mobilne tehnologije dodirnik postaje primarno sredstvo pokretanja računalnih naredbi. Važan je osvrt na predviđanja tehnološkog razvoja, obilježen korištenjem pokreta, govora pa čak i misli kako bi se upravljalo računalnim sustavom (Sharp et al., 2019).

Usporedno s razvojem materijalnih računalnih komponenti, razvija se i popratna programska oprema. Pretpostavka softvera je mogućnost digitalizacije podataka iz stvarnog svijeta. Pojam digitalizacije nastaje zbog znamenki 0 i 1, (engl. *binary digit)*. Radi se o sustavu računalno čitljivih izmjena dvaju stanja. Na primjeru bušenih kartica, znamenka 1 označava rupicu, dok 0 predočava njen izostanak. Sustav temeljen na binarnosti dokazao se iznimno lako upotrebljivim i programabilnim. Logika binarnih sustava je jednostavna i jasna no u početku nije bila upotrebljiva širem spektru korisnika. Prikaz podataka pomoću zaslona utječe na potrebu za unapređenjem mogućnosti stvaranja vizualnih elemenata softvera. Uvođenjem kodnih kartica te rasterskih i vektorskih grafika, omogućen je prikaz složenih oblika podataka na jednostavan način. Poboljšanje programske opreme za razvoj elemenata korisničkih sučelja značajno olakšava mogućnost rada na računalu za sve veći broj korisnika (Vlahović, 2020).

2.2. Korisničko sučelje

Očitovanje moderne interakcije čovjeka i uređaja omogućeno je korisničkim sučeljima različitih pojavnih oblika. Ona služe kao medijator transfera ljudske ideje u elektroničku sferu. Postižu potenciranje ljudskog stvaralaštva korištenjem informacijske tehnologije na specifičan način. Razvoj ove interakcije spomenut je i ranije, kada je čovjek računalom upravljao pomoću bušenih kartica, pa do danas, kada je dovoljan dodir prstom na plohi dodirnika. Napredak tehnologije je neupitan, a popraćen je ljudskom željom za učenjem i inovacijama. Sukladno razvoju informacijskog sustava, razvile su se i metode upravljanja tim sustavima kroz interakcijski dizajn. Pri provedbi interakcijskog dizajna važno je razumjeti dvije uključene komponente - mentalni model korisnika i funkcioniranje tehnologije (Tidwell, 2011).

Strelovit razvoj informacijskih sustava potaknuo je sukladan razvoj mehanizama njihovog upravljanja. Korisnička sučelja su pomagala koja omogućuju komunikaciju s uređajima raznih vrsta i mogućnosti, što opravdava njihovu raznolikost. Sučelja se opisuju različitim pridjevima, a grupiraju se s obzirom na funkciju, interakcijske stilove, način unosa i platformu za koju su namijenjena. U nastavku slijedi pregled nekih od najznačajnijih skupina i primjera sučelja, razvijenih u proteklih četrdeset godina (Sharp et al., 2019).

Pametna korisnička sučelja su primjer funkcijski orijentiranih sučelja. Stvaranje pametnog proizvoda je cilj svake moderne tehnologije, a ono se ostvaruje mrežnom povezanosti

proizvoda. Povezanost osposobljava komunikaciju jednog uređaja s korisnicima ili drugim uređajima. Premisa ove tehnologije je stvaranje kontekstualne svijesti stroja, ponajviše uz pomoć umjetne inteligencije. Integracijom sučelja u većinu elemenata ljudske okoline dovest će do računalnog upravljanja okoline. Važno je osigurati ulogu čovjeka u digitaliziranom okruženju kako ne bi došlo do otuđenja. Postoji odgovarajući pristup kojim se zagovara oblikovanje ljudske okoline na temelju čovjekovog iskustva, vrijednosti, potreba i prioriteta. Naziv spomenutog pristupa je ljudsko-građevinska interakcija (engl. *human-building interaction - HBI),* a primjenjuje se u pametnim građevinama čije su interne funkcije optimizirane uporabom računala i popratnih pomagala (Sharp et al., 2019).

Ovisno o interakcijskom stilu, učestala su naredbena, grafička i multimedijska korisnička sučelja. Temeljno svojstvo naredbenih sučelja je unos podataka ostvaren kraticama naredbi i pritiskom kombinacija tipki. Prednost ovakvih sučelja je u generalizaciji funkcija koje omogućuju efikasnije ispunjenje određenih zadataka ili skupine zadataka. Grafička korisnička sučelja integriraju izbornike, ikone, prečace, skočne prozore za prikaz informacija. Korisnici prioritiziraju ovu vrstu sučelja zbog svojstva upotrebljivosti i intuitivnosti. Pregled informacija grafičkim sučeljem je nezamjenjiv, a mogućnosti oblikovanja prikaza beskrajne (Sharp et al., 2019). Multimedijska sučelja kombiniraju različite vrste medija, kao što su slike, video, grafike, tekstovi, zvuk i animacije, unutar istog sučelja. Pretpostavka popularnosti multimedijskog sučelja je pravilo asocijativnosti sadržaja. Tim pravilom se olakšava mogućnost pamćenja sadržaja i navigiranja uz pomoć vizualnih pomagala (Johnson, 2010).

Raznolikost mogućnosti unosa i prikaza sadržaja su temelj grupiranja sljedeće skupine korisničkih sučelja. Sadržaj se može unositi računalnom periferijom, govorom, dodirom, gestama i moždanim valovima. U slučaju virtualne stvarnosti, popratna periferna oprema i glavni računalni sustav stvaraju iluziju interaktivnosti čovjeka u digitalnom okruženju. Uporabom naočala za virtualnu stvarnost i pripadajućih kontrola pokreta i govora, korisnik može navigirati raznim mogućnostima stroja. Jednakim metodama korisnik može upravljati i proširenom stvarnosti. U odnosu na virtualnu, proširena stvarnost superponira digitalne elemente na fizičkim uređajima i objektima, kako bi se stvorila iluzija spajanja digitalnog i stvarnog okruženja. Tehnologija upravljanja računalom pomoću moždanih valova, percipira se kao korisničko sučelje budućnosti. Elektrode spojene na glave korisnika su sposobne registrirati električne signale među neuronima i preoblikovati ih u računalne naredbe.

Izučavanje mogućnosti upravljanja računalom pomoću misli obuhvaća korištenje strojnog učenja za pretvorbu moždanih valova u naredbe (Sharp et al., 2019).

Tehnološki razvoj doveo je do diverzifikacije dostupnih uređaja. Korisnička sučelja moraju biti prilagođena svakoj vrsti uređaja kako bi se efikasno ispunjavale željene aktivnosti. Nisu jednake mogućnosti prikaza sadržaja na tabletu, računalu ili nosivoj tehnologiji. Mobilna tehnologija je postala temelj najpopularnije vrste korisničkih uređaja. Cilj mobilnih uređaja je olakšati brojne procese i postupke, pružajući veliku količinu podataka u stvarnom vremenu. Nosive tehnologije služe za prikaz fizičke aktivnosti i obavještavaju korisnika o novostima povezanim s drugim uređajima i programima. Roboti i dronovi se uvelike koriste u proizvodnim lancima i djelatnostima s visokim stupnjem rizika za čovjeka. Pametni kućanski uređaji optimiziraju postojeće procese, izvještavajući korisnika o stanju procesa ili omogućuju upravljanje na daljinu (Sharp et al., 2019).

Razvoj mogućnosti korisničkih sučelja doveo je do brojnih inovacija. Smjer u kojem se daljnji razvoj kreće, odnosi se na prirodna korisnička sučelja (engl. *natural user interface - NUI).* Kombinacijom gotovo svih spomenutih sučelja nastoji se stvoriti međuljudsko korisničko iskustvo u interakciji čovjeka i računala. *NUI* koncept predstavlja računalo kao stroj sposoban prepoznati ljudsko ponašanje i reagirati sukladno na različita raspoloženja. Nije poznato koliko će razvoja i vremena biti potrebno za stvaranje *NUI* sustava, ali do tog trenutka ljudi će se i dalje oslanjati na prevladavajuća grafička korisnička sučelja (Sharp et al., 2019). Važnost i uloga grafičkih korisničkih sučelja u današnjim uvjetima slijedi u nastavku.

2.3. Grafička korisnička sučelja

Revolucionarni početak grafičkih korisničkih sučelja, (engl. *graphical user interface - GUI),* dogodio se 1981. godine, kada tvrtka *Xerox* razvija *Star interface* - sučelje za osobna računala. Temeljilo se na konceptualnom modelu uredskog poslovanja, putem kojeg bi korisnici oponašali radnje ureda. Premještanje papira, odlazak do printera, organizacija dokumenata i slične radnje prenesene su u digitalno okruženje, gdje se takve aktivnosti obavljaju pomicanjem pokazivača miša. Time se otvaraju vrata novih mogućnosti za korisnike jer informacije postaju lakše dostupne i preglednije unutar grafičkog sučelja (Sharp et al., 2019).

Preteča ove vrste sučelja je *WIMP* (*windows, icons, menus, pointer*). Sučelje kvadratnog oblika, čija se interakcija temelji na prozorima, kliznim trakama, panelima i dijaloškim okvirima. Takva sučelja su ograničavala mogućnosti rada programera, zbog čega su u većoj mjeri bila namijenjena manje iskusnim korisnicima (Sharp et al., 2019).

Sučelja su u današnje vrijeme prilagođena svojstvima mobilnih uređaja i dodirnika, a temelj interakcije postaje dodir. Iako su neka svojstva zastarjela, temeljne pretpostavke *WIMP* pristupa su se uz pomodne modifikacije zadržale do danas (Sharp et al., 2019). Ikone, izbornici te prozori dolaze u različitim pojavnim oblicima, a više o njima bit će spomenuto u nastavku.

2.3.1. Prozor korisničkog sučelja

Prema navođenju Sharp et al. (2019.) prikaz informacija na zaslonu omogućen je takozvanim prozorima korisničkih sučelja. Nadilazeći granice mogućnosti zaslona uređaja, osposobljeni su pristup i pregled veće količine informacija radi povećanja efikasnosti. Dodatno obilježje ovog pomagala jesu klizne trake, kojima se dodatno zaobilaze ograničenja veličine ekrana i prozora, omogućujući vertikalno ili horizontalno navigiranje.

Uz sav napredak, pojavljuju se i nedostaci. Otvaranje prevelikog broja programa, može dovesti do konfuzije i problema sa snalaženjem. Moderniji uređaji nalaze rješenja problema u smanjenju prikaza otvorenih prozora, prikazujući ih na zajedničkom, preglednom mjestu. Dodatno pomagalo karakteristično za računalne prozore su dijaloški okviri - prozori manjih dimenzija koji daju povratnu informaciju, zatražuju potvrdu ili donose upozorenja i potvrde, kako bi usmjerili korisničku interakciju (Sharp et al., 2019).

2.3.2. Primjena izbornika

Izbornici su unutar sučelja najčešće smješteni na vrhu ili na stranama prikaza, podijeljeni na manje cjeline. Redoslijed postavljanja izbornika ovisi o procijenjenoj važnosti pojedinog izbornika, obraćajući pri tome pažnju na njihovo grupiranje i funkcije. Tako se korisnicima olakšava pristup važnom sadržaju, kojeg uistinu žele istražiti (Cooper et al., 2014).

Jedna od temeljnih vrsta izbornika je *flat list.* Ona omogućuju prikaz malog broja opcija istovremeno, što je pogodno za uređaje malih zaslona. Zbog tog ograničenja, koriste se ugniježđene liste te tako zahtijevaju provedbu u nekoliko koraka. Proširenje *flat* izbornika ostvareno je pomoću proširenog, *expanding,* izbornika koji dozvoljava prikaz veće količine

podataka. Važno pomagalo su pritom kaskadni izbornici, koji omogućuju proširenje glavnog izbornika do drugog ili trećeg stupnja radi specifikacije željene aktivnosti. Dodatni oblik *flat* sučelja je *mega* izbornik - prikaz velike količine dostupnih opcija koristeći *drop-down* metodu. Ovakav pristup dizajna sučelja naziva se "*recognition rather than recall*", jer se korisnicima prezentiraju dostupne mogućnosti umjesto da ih oni svojevoljno pretražuju (Johnson, 2010). Kao alternativa proširenoj vrsti, javljaju se *collapsible* sučelja, čija je temeljna uloga prikazati raznovrsnost sadržaja grupiranog pod krovni naslov. Odabirom naslova, otkriva se cijeli niz pripadajućih mogućnosti. Posljednji oblik izbornika pojavljuje se samo u kontekstualnom slučaju. Pojava je uvjetovana trenutnim zadatkom i aktivira se specifičnom naredbom. Aktivacijom *contextual* izbornika pojavljuju se mogućnosti odabira radnje povezane s trenutnom aktivnosti (Sharp et al., 2019).

2.3.3. Pojavni oblici ikona

Prve reprezentativne ikone računalnog sustava korištene su u *Star* sučelju. Njihova se uloga očituje u reprezentaciji objekata na zaslonu. Početne inačice ikona služile su prikazu datoteka, dokumenata, otpada, ulazne i izlazne pošte. Ikona u digitalnoj tehnologiji zauzima značajnu ulogu, prožimajući se u svim operacijskim sustavima, uređajima i aplikacijama. Osim prikaza objekata zaslona, ikone su počele predočavati stanja, alate pa čak i kompleksne operacije (Sharp et al., 2019).

Učvršćivanje značaja ikona u svijesti korisnika igra važnu ulogu u razvoju sučelja. Osim navedenoga, mapiranje je drugo poželjno svojstvo ikona kojim se osigurava postojanje poveznice ikone. Postoje generalno prihvaćene ikone s prigodnim značenjem, a najuspješnije među njima su izomorfne, zbog direktne reprezentacije. Dizajn ikona prolazi kroz složen razvoj. Od pikseliziranih, jednostavnih prikaza u samim počecima, preko fotorealističnih stilova, pa natrag do jednostavnog dvodimenzionalnog dizajna snažnih boja. Prihvaća se kako jednostavnost ikona ima svojstva prepoznatljivosti i karakterističnosti pa je lakše primjenjiva za prilagodbu manjem zaslonu (Sharp et al., 2019).

3. OSIGURANJE KVALITETE DIGITALNIH DOBARA

Razvoj tehnologije usko je popraćen promjenom ljudskih potreba i načina njihovog vrednovanja. Odgovori na pitanja o kvaliteti i načinima njenog ocjenjivanja, donesena su u nastavku poglavlja. Važno je definirati pojam i porijeklo kvalitete te na koji se način to svojstvo integriralo u informacijsku tehnologiju. Nakon inicijalnih postavki, slijedi detaljnija analiza postupka vrednovanja kvalitete softvera.

3.1. Percepcija kvalitete

Kvalitetu nije moguće svrstati unutar određenog vremenskog okvira. Njena je pojava usko vezana uz svaki oblik objektivne realnosti. Sve što postoji moguće je kvalitativno ocijeniti. Bilo da se radi o imanentnom ili transcendentalnom, čovjek je tomu sposoban pridružiti vrijednost na temelju postavljenih kriterija. Čak je i *homo sapiens* nesvjesno vodio računa o kvaliteti prilikom izbora namirnica i skloništa. Ovakva analogija olakšava pokušaj definiranja pojma kvalitete. Svojstveno joj je povezivanje s vrijednošću, korisnošću ili potrebom (Lazibat, 2009). Postavljajući definicijski okvir kvalitete, moguće se složiti oko toga kako je ona "mjera ili pokazatelj obujma, odnosno iznosa uporabne vrijednosti nekog proizvoda ili usluge, za zadovoljenje točno određene potrebe, na određenom mjestu, u određenom trenutku" (Injac, 1998).

Evolucija, napredak i globalni razvoj dovode do izmjene ljudskih potreba. Sukladno tomu, s ciljem otkrivanja noviteta, koristeći prethodno stečena znanja, vještine i iskustvo, mijenja se i pojam kvalitete te se njena primjena sve više širi i u poslovnu sferu ljudskog djelovanja (Lazibat, 2009). U nepreglednom moru poduzeća i proizvoda koja se neprestano bore za tržišnu izvrsnost, postavlja se važno pitanje; kako uopće postići kvalitetu? Čini se kao izrazito složen pothvat jer apetiti tržišta rastu, a životni se ciklus proizvoda skraćuje. U silnoj borbi za konkurentnost, isplivat će vrlo povoljan odgovor - "kvaliteta je besplatna". Dostupna je i ostvariva svima, a profit donosi na pošten način. Pretpostavka za njeno postizanje je udovoljavanje rastućim zahtjevima modernog svijeta, bolje nego konkurencija (Crosby, 1989).

Jednom kada poduzeće dostigne ultimativni cilj i ostvari kvalitetu svojih dobara u očima potrošača, potrebno je taj dojam njegovati i unapređivati. Za sve ovo je, između ostalog, zaslužno osiguranje kvalitete (engl. *quality assurance*). Osiguranje kvalitete obuhvaća skup planiranih i sistemičnih aktivnosti unutar sustava kvalitete, kako bi se ispunilo potrebne

zahtjeve. Pogrešna je pretpostavka da bi za kvalitetu trebao biti zadužen samo istoimeni odjel. Osiguranje kvalitete trebalo bi biti prožeto cijelim poduzećem, manifestirano svakom aktivnosti, svakog zaposlenika i tek naposljetku sustavno validirano od strane odjela za osiguranje kvalitete, koristeći za to predviđene alate i metode (Lazibat, 2009).

Primjenom korištene analogije u informatičkom okruženju, dolazi se do pojašnjenja uloge osiguranja kvalitete unutar procesa razvoja softvera te njen utjecaj na konačni proizvod. Izravno rečeno, osiguranje kvalitete u informatičkom sektoru označava metode vrednovanja softverskog proizvoda u ispunjenju očekivanih korisničkih zahtjeva. Zaposlenici odjela osiguranja kvalitete - testeri, su svjetlo projektnog puta koje svim ostalim uključenim stranama omogućuje manevriranje u pravom smjeru. Od njih potječu indikacije o tome kako projekt napreduje, u kojem se smjeru razvija te otkrivaju nedostatke koji ugrožavaju vrijednost proizvoda (Kaner et al., 2002).

3.2. Prva prijavljena greška u radu računala

Prvi prijavljeni nedostatak računalnog sustava dogodio se 9. rujna 1947. u 15:45. Navedeni trenutak obilježava događaj slijetanja jednog kukca na relej 70, panela F, na harvardskom *MARK II Aiken Relay* kalkulatoru. Pronalaskom uzroka greške u radu računala, i pripadajućim izvještavanjem, stvorena je prva instanca prijavljenog i dokumentarnog nedostatak u radu računala (engl. *bug)*. Čak je i kukac pronašao svoje mjesto na izvještaju, zalijepljen na komadu prozirne ljepljive trake. U spomen tom događaju, svako se buduće manifestiranje nedostataka, grešaka i kvarova u informatičkom svijetu naziva *bug*. Sukladno tomu, sustavi obilježenim većim brojem nedostataka, okarakterizirani su kao *buggy* (McFadden, 2020). Iako se ne radi o pravim kukcima, *bug* je i dalje nepoželjna pojava računalnih sustava, a više o tome bit će obrađeno u nastavku.

3.3. Testiranje kao potproces razvoja softvera

Razvoj softvera se odnosi na razvoj programa ili skupine programa, uz osiguranje ispravnosti, kvalitete, efikasnosti i dugovječnosti programskog rješenja. Uzimajući u obzir važnost koju softver danas ima u vidu financijskih i idejnih ostvarenja, neophodna je primjena odgovarajućih modela razvoja usmjerenih ka zadovoljenju vremenskih, funkcionalnih i drugih kriterija koji omeđuju proces razvoja. Iako zastario, vodopadni model kroz slijedni pristup razvoja softvera omogućuje predodžbu procesa i temeljnih faza. Iz ovog se modela dalje razvijaju novi, primjenjiviji i prilagodljiviji pristupi, kao što su; iterativni, spiralni,

prototipni model te agilne metodologije koje lakše odgovaraju na potrebe dinamičnog okruženja. Glavno obilježje vodopadnog modela je nužnost završetka svake pojedine razvojne faze, prije početka iduće. Tipične faze ovog modela su 1) definiranje i analiza zahtjeva, 2) modeliranje aplikacije, 3) razvoj aplikacije, 4) testiranje, 5) verifikacija i validacija, 6) implementacija i 7) održavanje (Vlahović, 2020).

Definiranjem i analizom zahtjeva, počinje se proces razvoja. Ovisno postavlja li zahtjeve razvojni inženjer ili korisnik, zahtjevi se mogu odnositi ili na programske karakteristike rješenja ili na primjenu softvera. Uspješnost ove faze ovisi o mogućnosti dogovora i kompromisa na koje su obje strane spremne, pogotovo kada se radi o nerealnim i skupim očekivanjima. Modeliranje aplikacije očituje se vizualizacijom definiranih zahtjeva. Cilj ovog koraka je vizualni prikaz temeljne problematike aplikacije koristeći predviđene alate, kao što je to *UML (Unified Modelling Language).* Osim problematike, prikazuju se aktivnosti, stanja i projekcije s dinamičkog i statičkog aspekta. Razvojni inženjeri različitih područja specijalizacije su zaduženi za izradu softverske aplikacije (Vlahović, 2020).

Razvoju programskog rješenja se pristupa koristeći programski jezik, softverski alat ili kombiniranim pristupom. Dok je korištenje programskog jezika i raspisivanje programskog koda najintenzivniji pristup, on omogućuje najveću razinu individualnosti. Manje slobode, ali pojednostavljenje i automatizaciju dijela razvojnog procesa nude razna softverska pomagala. Kombinacija obaju pristupa nudi rješenje koje se u današnje vrijeme učestalo koristi zbog financijske i vremenske uštede. Da bi se uvjerilo u ispravnost izrađenog programa, potrebno je provoditi aktivnosti kompilacije i testiranja (Vlahović, 2020).

Kompilacijom programa otkrivaju se formalne, semantičke i sintaktičke pogreške, dok se testiranjem otkrivaju logički nedostaci; ponašanja koja odstupaju od zamišljenih. Glavno obilježje ove faze jest dubinsko testiranje svih dijelova programa, od najjednostavnijih do izrazito složenih funkcija (Vlahović, 2020). Prema navođenju Martina i et al. (2018.), testabilnost je vrsna odlika kvalitetnog programskog koda. Nužno je integrirati testove prilikom dizajna sustava, kako bi se očuvalo njegovu stabilnost. Jednaka razdioba kao i u prvoj fazi razvoja softvera, odvija se i u fazi verifikacije i validacije. Dok se verifikacijom utvrđuje izgrađenost softvera po pravilima struke, uz odgovarajuću dokumentaciju i specifikacije poslova, validacija se bavi pitanjem uporabljivosti i ispunjenja korisničkih zahtjeva (Vlahović, 2020).

Primjena proizvoda i faza implementacije započinju instalacijom programa. Korisnika je potrebno osposobiti za rad s novim softverom kako bi se pokrenula faza "uhodavanja", unutar koje se ispravljaju novootkrivene greške i dodatno educiraju zaposlenici. Konačna faza ne označava kraj rada na aplikaciji, nego se odnosi na kontinuirano održavanje i prilagođavanjem novim zahtjevima. Promjene zahtjeva dolaze od strane korisnika, informatičara ili okoline, do onog trenutka kada aplikacija zastari, njeno održavanje postane složeno i neisplativo (Vlahović, 2020).

3.3.1. Razine testiranja softvera

Uspješnost softverskog projekta ovisi o kvaliteti koju on pruža. Da bi se osigurala kvaliteta na ispravan način, potrebno je razgraničiti cilj testnog procesa koji se može odvijati na različitim razinama (Naik & Tripathy, 2008).

Kako bi se vrednovala funkcionalnost čitavog sustava, nužno je provjeriti funkcionalnost svake pojedinačne komponente. Upravo se taj proces odvija unutar testiranja cjeline (engl. *unit testing*). Ovim se pristupom olakšava integracija novih funkcionalnosti u već razrađen sustav, a pronađeni nedostaci mogu se pridružiti specifičnoj jedinici sustava (Naik & Tripathy, 2008).

Integracijsko testiranje (engl. *integration testing)* omogućuje slijednu provjeru pojedinačnih komponenti i sklada kojeg te komponente ostvaruju. Osim svojstava jedinice, tester dobiva uvid u grupne performanse komponenti sustava. Dok se ne istraže svojstva međuodnosa komponenti, nije moguće znati kako se sustav ponaša kao cjelina (Naik & Tripathy, 2008).

Posljednji korak u procesu verifikacije zauzima testiranje sustava (engl. *system testing)*. Njegova složenost se očituje u zadovoljenju postavljenih rokova projekta. Najveća prepreka dolazi do izražaja prilikom ovjere ispravaka svih prijavljenih nedostataka, u iznimno kratkom vremenu. Osim nedostataka, potrebno je utvrditi valjanost provedbe ostalih testnih razina i faza. Ključna aktivnost testiranja sustava odnosi se na ovjeru specificiranih zahtjeva i otklonjenih nedostataka (Naik & Tripathy, 2008).

Uz pomoć korisnika ili predstavnika korisnika aplikacije, provodi se testiranje spremnosti aplikacije da bude korištena (engl. *acceptance testing)*. Ovom se konačnom provjerom mogu utvrditi ispravnost provedenih korisničkih zahtjeva i kvaliteta softvera. Sud o tome donose

izvršitelji testiranja, temeljeno na kriterijima oblikovanim prema vlastitim očekivanjima (Naik & Tripathy, 2008).

3.3.2. Životni ciklus sistemskog testiranja softvera

Izolirajući sistemsko testiranje kao zasebni proces, potrebno je predstaviti faze kroz koje se ono odvija. Svaka komponenta testiranja mora zadovoljiti postavljena mjerila kvalitete, kako bi test u cjelini bio uspješan. Uvijek primjenjivi slijed odvija se u sljedećim fazama (Naik & Tripathy, 2008):

1. Analizom zahtjeva vrednuju se okviri postavljenih zahtjeva, unutar kojih se odvija testiranje. Prikupljaju se podaci o zahtjevima kako bi se moglo planirati odgovarajuće testove. Pritom se vodi računa funkcionalnosti zahtjeva - utječu li izravno na rad softvera ili doprinose korisničkom iskustvu.

2. Planiranje testiranja omogućuje stvaranje testnog plana unutar kojeg se definiraju uloge pojedinog testera te projekcije troškova, intenzivnosti rada, resursa i vremena.

3. Stvaranjem testnih slučajeva na temelju razrađenih zahtjeva, oblikuju se pristupi testiranja softvera. Ako se radi o automatizaciji testiranja, izrađuju se i pripadajuće skripte s naredbama za automatizaciju.

4. Priprema okruženja obilježava stvaranje testnih uvjeta. Prije svega, to su nužne specifikacije računala i programske opreme potrebne za simulaciju oblikovanih testnih slučajeva.

5. Provedba testiranja označava pristup radu u kojem testeri, na temelju postavljenih zahtjeva i oblikovanih testnih slučajeva, provode testiranje. Cilj ovog korak je usporediti stvarno i očekivano stanje postavljenih zahtjeva. U slučaju nepodudarnosti, stvaraju se izvještaji koji se prosljeđuju programerima radi popravljanja programa. Svi se prijavljeni nedostaci pažljivo prate i uzimaju u obzir tijekom trajanja ove faze. Njihovi ispravci se ponovno testiraju kako bi se uvjerilo u otklanjanje nedostatka.

6. Zatvaranje životnog ciklusa posljednji je korak procesa testiranja. Karakteristično je pritom okupljanje svih testera kako bi se vrednovao slijed ciklusa, izvještaji i kurentno stanje. Korektno pristupanje ovoj fazi omogućuje otklanjanje svih

nedostataka samog procesa testiranja, kako bi pokretanje novog životnog ciklusa bilo još uspješnije.

Razumijevanje pojedine faze životnog ciklusa testiranja omogućuje jasniju razdiobu postavljenih zadataka. Strukturiranje procesa testiranja je važno kako bi se moglo valjano provesti odgovarajuću aktivnost pojedine faze. Ovisno o primijenjenoj metodologiji testiranja, razlikovat će se pristup za svaku fazu ciklusa (Naik & Tripathy, 2008). Razrada raznih metodologija testiranja softvera slijedi u nastavku.

3.4. Metodologije testiranja softvera

Najvjerniji oblik testiranja svakako je primjena aplikacije u stvarnom vremenu i okruženju. Cilj odjela osiguranja kvalitete je što bliže replicirati korisničke zahtjeve i uvjete, kako bi se postigli reprezentativni rezultati. Ovisno o načinu i metodama provedbe testiranja, taj dio razvojnog procesa može biti izrazito složen. Unapređenje tehnologije i rastuća potražnja za raznim programskim rješenjima, popraćen složenošću tih programskih rješenja, sve više zahtjeva specijalizaciju timova za osiguranje kvalitete - testiranje (Vlahović, 2020).

Prema navođenju Kanera et al., (2002), svaki oblik testiranja može biti opisan kroz pet dimenzija. Prva se dimenzija odnosi na testere, osobe koje provode testiranje. Najčešći provoditelji testiranja su stručni zaposlenici odjela osiguranje kvalitete, korisnici ili njihovi predstavnici te stručnjaci polja za koje je program razvijen. Dimenzija pokrivenosti predstavlja ono što biva testirano, a odnosi se na širinu testiranja. Širina testiranja definira u kojoj se mjeri testiraju pojedini elementi, funkcije sustava ili sustav u cjelini. Potencijalni problemi su također polazišna točka testiranja. Pristup radu u ovoj dimenziji, prema Amlandu (1999.), obuhvaća prioritizaciju zadataka baziranu na vjerojatnosti pojave greške i pripadajućeg troška. Raznoliki načini i pristupi testiranju definirani su dimenzijom aktivnosti, a generiraju raznovrsne rezultate. Posljednja je dimenzija evaluacija, kao konačni sud uspjeha ili neuspjeha testiranja. Svaki oblik testiranja zapravo do neke mjere obuhvaća i druge dimenzije no one nisu podjednako izražene u svakom pristupu. Testiranje se može vršiti s naglaskom na specifičnu dimenziju, ali se sukladno vrsti i važnosti testiranja može proširiti u multidimenzionalnu sferu.

3.4.1. Metodologije testiranja grafičkih korisničkih sučelja

Aplikacijama današnjeg doba korisnici najčešće pristupaju pomoću grafičkih korisničkih sučelja, ponajviše zbog pogodnosti *WIMP* sustava, koji olakšava snalaženje i prijenos informacija. Da bi se osiguralo besprijekorno korisničko iskustvo, provode se odgovarajući testovi. Objekti koji se u tim sučeljima testiraju jesu prozori, izbornici, ikone, klizne trake, gumbi i ostale komponente sučelja - svi elementi koji se nalaze na sučelju.

Metodologija koja okuplja više dimenzija testiranja, a pogodna je za testiranje grafičkih korisničkih sučelja je testiranje uporabljivosti (engl. *usability testing).* Radi se o metodologiji testiranja koja navodi testera na vrednovanje (1) lakoće učenja pristupa i upravljanja sustavom, (2) reakcije sustava na unos korisničkih podataka, (3) efikasnosti operacija i razumljivosti sustava. Ovisno o tome mogu li korisnici pristupiti, navigirati i napustiti softver na jednostavan način, govori se o njegovoj pristupačnosti. Reakcije sustava na korisničke aktivnosti donose pregled o tome može li korisnik jasno izvršiti željene radnje u željeno vrijeme, uz prisutnost obilježja prihvatljivih boja, oblika, zvuka i veličine slova. Efikasnost sustava se mjeri brojem koraka koje korisnik mora provesti kako bi izvršio željenu radnju, u što kraćem vremenu. Posljednja je stavka razumljivost sustava, kojom se vrednuje lakoća kojom korisnik može razumjeti strukturu sustava (Naik & Tripathy, 2008).

Usability testing kao metodologija korisna je za razumijevanje stvarnog funkcioniranja sustava. Prolaskom kroz sustav iz perspektive korisnika, omogućuje se lakše predviđanje neplaniranih radnji i njihov utjecaj na rezultat. Vrijeme je ključni element ispitivanja uporabljivosti. Za ovu vrstu testiranja nužno je izdvojiti dovoljno vremena, a preporuča se kontinuirana provedba tijekom čitavog procesa testiranja u raznim fazama. Preporučljivo je i korištenje specifičnih vrsta testiranja unutar provjere uporabljivosti. Neprestana primjena specifičnih metoda testiranja može dovesti uklanjanja inicijalnih dvojbi oko određenog problemskog područja, i usmjerenja pažnje na novi, značajniji propust (Krug, 2014). Najznačajnije vrste testova, koje će biti korištene prilikom provedbe testiranja u završnom dijelu rada, definirane su kroz ranije spomenute dimenzije.

Testiranje temeljeno na izvršitelju testiranja definira odgovornu osobu ili tim za provedbu testiranja. Za potrebe demonstracije testiranja u poglavlju 5., bit će primijenjeno testiranje softvera od strane internog odjela za osiguranje kvalitete (engl. *alpha testing)*. Prednost *alpha*

testiranja proizlazi iz poznavanja programske strukture koda i oblikovanja testova sukladno prepoznatim mogućnostima (Kaner et al., 2002).

Pokrivenost testiranja postavlja obujam i doseg provođenja testnih aktivnosti. U slučaju slijednog testiranja zasebnih komponenti (engl. *function testing)*, pažnja se usmjerava na okvire svake pojedine funkcije. Nakon ovjere valjanosti pojedinih komponenti, provodi se skupno testiranje ostvarenog sklada pojedinih funkcionalnosti (engl. *feature of function integration testing)*. Proširenje testiranja očituje se detaljnom ovjerom svakog elementa, unutar svakog korisničkog sučelja programa (engl. *menu tour)*. Srodno pristupu *menu tour,* koristi se testiranje toka (engl. *path testing).* Ovim se pristupom istražuje ishod poduzimanja specifičnog seta aktivnosti. Pretpostavka je kako ciljane radnje unutar programa, korisnika dovode do željenog stanja ili prikaza. Za kvalitetnu provedbu *path* testiranja, potrebno je provesti i nelogične korake do kojih bi prosječni korisnik došao slučajno (Kaner et al., 2002).

Prema Whittakeru i Jorgensenu (1999., 2000.), uzimajući problem kao inicijalnu pretpostavku testiranja, moguće je izvesti razne zaključke o uporabljivosti programa. Osnovi problem mnogih sustava očituje se u postavljenim ograničenjima. Sustav je ograničen okvirom unutar kojeg mu je definirano djelovanje. Testiranjem ograničenja unosa (engl. *input constraints),* potrebno je unositi varijable čije vrijednosti odstupaju od postavljenih okvira. Testiranje ograničenja prikaza (engl. *output constraints)* jednako je važno koliko i testiranje unosa. Iako se unose ispravni podaci, sustav može prikazivati neispravne rezultate. Prikaz netočnih informacija ukazuje na dodatne nedostatke sustava.

Testiranje temeljeno na aktivnosti pristupa omogućuje ovjeru softvera iz različitih perspektiva. Generalno funkcioniranje programa ovjerava se standardiziranim testovima (engl. *smoke testing*). Predmeti *smoke* testiranja su najbitnije funkcionalnosti za koje se očekuje ispravno ponašanje. U protivnom slučaju utvrđuje se postojanje osnovne pogreške u kreiranju programa koje bi trebale biti lako uklonjene. Metodom scenarija (engl. *scenario testing)* bitno je provoditi testne aktivnosti onako kako se pretpostavlja da bi to učinio prosječni korisnik. Stoga, ovakvi testovi zahtijevaju realističnost, korištenje kombiniranih značajki programa, jednostavnu ovjeru uspješnosti te uvjerljivost za dioničare. U slučaju pojave nedostatka, važno je da dioničari shvaćaju utjecaj i opseg nastalog problema. Nadalje, kvaliteta programskog koda obilježena je performansama programa. Program koji reagira brzo i glatko zadržava korisnike. Testiranje učinkovitosti programa (engl. *performance testing*) omogućuje uočavanje nedostataka temeljnih postavki programskog koda. Ako se u

različitim testnim okolnostima za istu aktivnost prikazuju značajna odstupanja u ponašanju, moguće je izvesti zaključak o utvrđenom nedostatku (Kaner et al., 2002).

Nakon provedbe definiranih testnih slučajeva, potrebno je zaključiti proces testiranja. Posljednja faza je evaluacija uspješnosti testiranja. Željeno ponašanje programa definirano je u dokumentu značajki programa ili sličnom autoritativnom dokumentu. U slučaju otkrivanja odstupanja ponašanja programa od značajki postavljenih u odgovarajućem dokumentu, izvodi se zaključak o uspješnom testnom pothvatu, a uočeni nedostaci se prijavljuju i prosljeđuju na doradu (Kaner et al., 2002).

3.5. Osiguranje kvalitete kao komponenta informacijske sigurnosti

Brojni moderni poslovni modeli svoje poslovanje temelje na informacijskim sustavima - nizu komplementarnih komponenti koje međuzavisnim djelovanjem omogućuju temeljne operacije nad informacijama. Skup komponenti; hardver, softver, podaci, korisnici, mreža i organizacija, svojim usporednim i sinkroniziranim djelovanjem omogućuju prikupljanje, obradu, pohranu i distribuciju informacija. Cilj ovog djelovanja je provedba poslovnih transakcija, njihovo dokumentiranje, pohrana, izvještavanje te efikasno upravljanje poslovnim subjektom (Spremić, 2017).

Intenzivnom integracijom informacijskih, digitaliziranih sustava u poslovne modele organizacija, raste i prijetnja izlaganju kibernetičkim rizicima. Opasnost ovih rizika očituje se u njihovoj dvojnoj naravi; neizbježni su i sveprisutni, a njihovo manifestiranje može uzorkovati signifikantne probleme u poslovanju. Upravljanje ovakvim rizicima pomaže očuvanju vrijednosti informatičkih ulaganja. Uvode se mjere sigurnosti informacijskog sustava, kojima se osmišljavaju i primjenjuju metode i kontrolni mehanizmi za zaštitu od sigurnosnih prijetnji. Primjeri sigurnosnih mjera su metode; identifikacije, provjere ovlaštenja, zaštite u prijenosu te kriptografske metode zaštite. Način na koji se korisnik predstavlja informacijskom sustavu, te sukladno tomu, ovlaštenja koja pripadajuća identifikacija korisnika omogućuje, igra veliku ulogu u zaštiti informacijskih sustava (Spremić, 2017).

Spajajući tako znanja o osiguranju kvalitete i sigurnosti informacijskih sustava, moguće je izvesti jasan zaključak - obje djelatnosti usmjerene su na upravljanje određenim rizicima. Dok se osiguranje kvalitete bori s rizicima koji utječu na kvalitetu proizvoda i korisničko iskustvo, sigurnost informacijskih sustava nastoji spriječiti sigurnosne propuste programa

koji bi nanijeli štetu, i korisnicima i proizvođaču. Integracijom pojedinih metoda sigurnosnih kontrola u potproces testiranja softvera, odnosno u proces razvoja softvera, suzio bi se broj propusta i mogućih ulaza prijetnji u okolinu. Prvi korak u tom smjeru jest implementacija osnovnih sigurnosnih zahtjeva u podobne testne scenarije. Kontinuiranom provjerom sigurnosti mreže, sustava te serverske strane programa, osiguralo bi se nepostojanje propusta u navedenim dijelovima sustava. Pošto je važno u obzir uzeti poslovnu dimenziju ovog procesa, u korist svakako ide troškovna prednost koju ima pronalazak i popravak (sigurnosnih) nedostataka programa prije nego dosegne krajnje korisnike (Mahfuz 2016).

3.6. Norme kvalitete testiranja

Poduzeća su dužna ispuniti regulativne okvire o efikasnosti vlastitih procesa, proizvoda i usluga. Okviri se definiraju normama i standardima kvalitete, a cilj im je usmjeriti poduzeća na ispunjenje korisničkih zahtjeva i očekivanja. Serija normi ISTQB, ISO/IEC/IEEE 29119 i ISO 9126, odnose se na standarde testiranja softvera čiji je cilj međunarodna primjena na razini organizacija, prilikom provođenja bilo koje vrste testiranja (ISO/IEC/IEEE 29119-1:2022, 2022).

Vodeća i globalna shema certifikacije ISTQB, primjenjuje se u polju testiranja softvera putem koje se unapređuju znanja i sposobnosti izvršitelja testiranja. Ovim standardom moguće je kroz velik broj edukacija napredovati od certificiranog testera osnovne razine do specijalista pojedine metode testiranja. Takozvana jezgrena, *core,* razina modula, primjenjiva je za svaki oblik tehnologije, metodologije ili aplikacije. Agilni tok modula, *agile*, usmjeren je na prakse testiranja korištene u agilnoj metodologiji razvoja softvera. Specijalizantski tok edukacijskog modula omogućuje dalekosežni i dubinski razvoj sposobnosti za specifične pristupe testiranju i aktivnosti. Ono što ovu certifikaciju čini vodećom, jest održivo unapređenje poslovne prakse kroz: osiguranje visoke razine kvalitete, konstantnu prilagodbu relevantnosti, održavanje valjanosti već dodijeljenih certifikata i globalno ujednačavanje standarda kako bi bili primjenjivi na međunarodnoj razini (ISTQB, 2022).

Sustavi osiguranja kvalitete, kao što je to serija normi ISO/IEC/IEEE 29119, nastali su radi želje za unapređenjem i olakšanjem međunarodne trgovine. Temeljni ciljevi takvih normi su kontinuirano unapređivanje kvalitete, poboljšanje operacija koje to omogućuju te pružanje povjerenja sudionicima procesa (Lazibat, 2009). Set standarda obuhvaćenih serijom normi ISO/IEC/IEEE 29119 numeriran je, a obuhvaća 1) koncepte i definicije, 2) procese testiranja,

3) testnu dokumentaciju, 4) tehnike testiranja i 5) testiranje na temelju ključnih riječi (Software Testing Standards, 2014).

Svaka od komponenti ove norme, usmjerena je na specijalizaciju pojedinog područja testiranja. Prolazeći tako slijedno kroz skupinu standarda, prvo s čim se korisnik susreće jest uvod u standarde, koji sadrži objašnjenja potrebna za razumijevanje i primjenu preostalih komponenti seta. Procesi testiranja, kroz tri razine (organizacijska, upravljačka i dinamička), olakšavaju korisniku cijelu metodologiju testiranja, zadajući okvire djelovanja koji u postavljenim uvjetima završavaju uspješnim ishodom. Nadovezujući se na testnu proceduru druge stavke, testna dokumentacija služi kao svojevrsni predložak evidencije kakvu bi izvršitelj testiranja trebao voditi tijekom svoje aktivnosti. Prilikom oblikovanja testnih scenarija, svakako je bitno obratiti pažnju na tehnike testiranja. Tom normom su dane preporuke za dizajn testova koji rezultiraju dokazivanjem prisustva nedostataka ili ispunjavanja željenih programskih zahtjeva. Posljednja od stavki, testiranje na temelju ključnih riječi, podupire oblikovanje testnih slučajeva na temelju predodređenih ključnih riječi, povezanih s aktivnostima koje izvršitelj testa čini prilikom testiranja (TestBytes, 2017).

Sposobnost sustava da omogući funkcionalnosti koje zadovoljavaju postavljene zahtjeve i potrebe, vrednuje se standardom ISO 9126. Prema toj se odredbi mogu vrednovati očekivanja od implementiranog sustava. Osnovne su značajke pritom; sigurnost, usklađenost, prikladnost, korektnost i interoperabilnost (Spillner, et al. 2014).

4. RAŠČLAMBA MANUALNOG I AUTOMATSKOG TESTIRANJA

Svaki od pristupa testiranju ima svoje jače strane i propuste. U nastavku slijedi razrada temeljnih značajki pojedinog pristupa. Nakon zasebnog uvoda i upoznavanja s testnim vrstama, slijedi njihov usporedni prikaz.

4.1. Manualno testiranje

Manualno testiranje softvera uključuje provedbu testnog procesa bez uporabe pomagala i alata. Ručno testiranje se koristi za pronalazak kritičnih nedostataka sustava, ali i kao inicijalno testiranje kojim se ovjerava podobnost primjene automatskog testiranja. Temeljna zadaća manualnog pristupa testiranju je osiguranje funkcioniranja sustava u skladu s postavljenim funkcionalnim zahtjevima (Kuchana, 2016).

Gotovo svaka vrsta testiranja može biti provedena manualno, ali je pri tome upitna efikasnost pojedinog testa. Manualno testiranje se oslanja na vještine izvršitelja testiranja, a najvažnije će biti opisane u nastavku. Tehničko razmišljanje je korisna sposobnost jer omogućava testeru razumijevanje uzroka i posljedica aktivnosti na logičkoj razini koda te predviđanje ponašanja sustava. Kreativnim razmišljanjem provoditelj testiranja može jednostavno generirati ideje i scenarije za pronalazak potencijalnih problema i nedostataka. Osim ideje, bitna je i provedba. Praktični tok misli je odlika sposobnih testera da konceptualne pretpostavke provedu u djelo, primjenom testnih slučajeva i tehnika. Prilikom testiranja je bitno zadržati objektivno stajalište kako bi se projekt mogao povezati s objektivnim mjerilima kvalitete. Sukladno objektivnom pristupu, kritičko razmišljanje usmjerava testera na vrednovanje ideja i zaključaka, ali i prihvaćanje vlastitih pogrešaka. Kriticizam pomaže zaduženoj osobi povezati projekt s objektivnim mjerilima kvalitete programa (Kaner, et al. 2002.).

4.1.1. Prednosti manualnog testiranja

Prednosti ovog pristupa testiranju temelje se na činjenici kako automatizacija niti u jednom sustavu ne može biti provedena u potpunosti. Jedan od razloga je što se manualnim testiranjem vrši inicijalna provjera karakteristika sustava, vrednujući pritom mogućnost testiranja automatskim testovima. Drugi i važniji razlog je ljudska komponenta. Softver je namijenjen krajnjem korisniku, a računala zasad nisu sposobna replicirati ljudsko ponašanje na način na koji to može odraditi drugi čovjek (Hamilton, 2022).

4.1.2. Nedostaci manualnog testiranja

Pošto se radi o radno intenzivnom pristupu, jasno je kako će se glavni nedostaci očitovati u toj sferi. Ponavljajući zadaci velik su izazov ručnom testiranju. Dovode do stanja monotonije, gubitka koncentracije te su zbog visoke vremenske zahtjevnosti i skupi. Neefikasnost ručnog testiranja očituje se; u vrednovanju testnih slučajeva, pri kalkulaciji pokrića testa, pri analiziranju velike količine dobivenih podataka ili simulacijama ponašanja programa. Osim navedenih dijelova procesa testiranja, sve je više oblika testova koje nije moguće provesti manualnim testiranjem, pogotovo u slučajevima kada testni objekti nemaju korisničko sučelje, što ovaj pristup čini zavisnim (Naik & Tripathy, 2008).

4.2. Automatsko testiranje

Suprotno manualnom pristupu testiranju softvera, u kojemu čovjek obavlja sve aktivnosti, automatsko testiranje zagovara korištenje pomagala, alata i skripti prilikom provedbe testiranja. Na ovaj se način ostvaruje izvršenje testnih scenarija bez čovjekove intervencije. Čovjek je i dalje prisutan kako bi inicijalizirao proces. Zadužen je za pripremu željenog pomagala za izvršenje planiranog testa, umećući odgovarajuće informacije i postavljajući definirane parametre (Kuchana, 2016). Stručnjaci ovog područja opskrbljeni su temeljnim znanjem o testiranju, iskustvom u programiranju te znanjem o dostupnim alatima i skriptnim jezicima. Nakon provedbe testnog slučaja, generira se izvještaj o procesu testiranja iz kojeg se iščitavaju rezultati. U slučaju naprednijih alata, usporedba ostvarenih i očekivanih rezultata bit će također prikazana prilikom ispisa izvještaja. Za svaku pojavu nedostataka, bilo u sustavu ili izvještaju, izvršitelj testiranja će vrednovati radi li se o grešci u postavljanju testnog slučaja, alata za automatizaciju ili se radi o legitimnom kvaru (Spillner et al., 2014).

Poduzeća koja odluče prihvatiti automatizaciju, moraju zadovoljiti određene pretpostavke kako bi uvođenje automatskog sustava testiranja bilo uspješno. Stabilnost sustava je prva pretpostavka kojom se osigurava značajnost testiranja i niski troškovi održavanja. Sustav koji primjenjuje automatizaciju testiranja mora imati jasno definirane procese. Oblikovanje testnih slučajeva je izvedivo kada za očekivane rezultate i postupke postoji pravilo jednoznačnosti. Automatizacijsko testiranje počiva na premisi uporabe adekvatnih alata i infrastrukture, kojima moraju upravljati za to osposobljene osobe. Poduzeće bi trebalo omogućiti dostupnost adekvatnih pomagala za kvalificirane testere, uzimajući u obzir potreban budžet (Naik & Tripathy, 2008).

4.2.1. Prednosti automatizacije testiranja

Generalne prednosti automatske provedbe testiranja su efikasnost, vremenska nezahtjevnost te pomoć izvršitelju pri unapređenju kvalitete testiranja i vlastitih sposobnosti. Alati omogućuju rasterećenje izvršitelja testiranja, oslobađajući ga provedbe repetitivnih zadataka. Tim više, pomagala su sposobna u kratkim vremenskim intervalima provesti izuzetan broj iteracija takvih testova uz veće pokriće. Automatizacija testiranja se provodi za testiranje manualno nedostupnih funkcija kao što su testiranje performansi koda, njegovog opterećenja te druge sustavne provjere (Naik & Tripathy, 2008).

Testiranje dinamičnih komponenti programa se efikasnije provodi automatizacijom jer se u kratkom vremenu može zabilježiti velik broj instanci različitih vrsta unesenih podataka, prateći pritom reakciju programa. Značajna ušteda vremena očituje se u prilagodljivosti testiranja izmjenom skriptnih parametara, bez potrebe za kompletnom promjenom skripte da bi se provelo novo testiranje. Navigacija pomagala je od velike prednosti pošto omogućuje pokretanje testa iz proizvoljne točke sekvence testiranja. U vidu testnih izvještaja, od velikog je značaja automatsko generiranje izvještaja, a u nekim slučajevima i usporedba očekivanih i ostvarenih rezultata testiranja (Naik & Tripathy, 2008).

4.2.2. Nedostaci automatizacije testiranja

Iako je prilagodljivost automatskih testova prednost, ona nažalost nije primjenjiva u svim situacijama. Jedna od njih je pri testiranju grafičkih korisničkih sučelja, kada izmjena elemenata čini stvorenu skriptu neuporabljivom. Dok se referentne oznake promijenjenih elemenata ne ažuriraju, provođenje testa će dovoditi do kvara. Da bi automatizacija bila primjenjiva, poduzeće mora uložiti značajne financijske i organizacijske napore jer testiranje kaotičnih sustava rezultira kaotičnim rezultatima (Naik & Tripathy, 2008).

Iako je provođenje automatskog testiranja izrazito brzo, njegovo učenje nije. Unapređenje programa poduzeća obvezuje i kontinuirano ažuriranje alata za automatizaciju, a time i vještine izvršitelja testa. Prilagodba novim alatima zahtijeva određeno vrijeme koje utječe na produktivnost. Osim vremenskih ograničenja, važno je u obzir uzeti isplativost troškova. Automatizacija testiranja tek postaje isplativa nakon što se provede velik broj testiranja (Naik & Tripathy, 2008).

4.3. Usporedni prikaz značajki manualnog i automatskog testiranja

Na temelju već obrađenih činjenica, sastavljen je usporedni prikaz temeljnih obilježja obaju pristupa testiranju, prikazan tablicom 1.

Tablica 1: Usporedni prikaz značajki manualnog i automatskog testiranja

Parametar	Manualno testiranje	Automatsko testiranje
Provedba	Zahtijeva ljudsku intervenciju	Koristi alate i skripte
Vremenska zahtjevnost planiranja testiranja	Relativno niža	Relativno viša - zbog oblikovanja programskog koda automatizacije
Vremenska zahtjevnost provedbe testa	Visoka	Niska
Pokrivenost testiranja	Zahtjevno osigurati dostatnu pokrivenost	Pokrivenost se jednostavno osigurava u parametrima programskog koda
Repetitivni zadaci	Neefikasno	Brzo i učinkovito
Pouzdanost	Mogućnost ljudske greške	Nema zamora pri testiranju
Testiranje testnih objekata bez korisničkog sučelja	Nije moguće	Moguće isključivo automatizacijom testa
Testiranje kaotičnih sustava	Moguće	Nemoguće
Troškovi	Rastu proporcionalno s brojem testova	Opadaju proporcionalno s brojem testova
Izvještaji	U djelokrugu izvršitelja testiranja	Lako dostupni većem broju korisnika
Ljudski faktor	Testiranje iz korisničke perspektive	Ne uzima u obzir ljudsko ponašanje
Vještine testera	Relativno manje	Relativno veće
Podobni testovi	Testiranje iskoristivosti, korisničkih sučelja i istraživačko testiranje	Regresijsko testiranje, testiranje performansi, opterećenja te repetitivnih zadataka

Iz podataka prikazanih tablicom 1, može se izvesti zaključak o efikasnosti pojedine metode testiranja. Automatizacija testova uz pomoć alata i skripti efikasno izvršava veliki broj instanci u kratkom vremenu. Iako je planiranje testnog procesa zahtjevno zbog oblikovanja

programskog koda, vrijeme se nadoknađuje kontinuiranim provjerama repetitivnih zadataka i jednostavnom prilagodbom parametara testiranja. Troškovi opadaju učestalom primjenom, a isključuje ljudsku pogrešku prilikom provedbe. Prednost automatskog testiranja očituje se u testiranju performansi, regresijskog testiranja i sustava koji nemaju korisničko sučelje.

Manualno testiranje se koristi za provjeru korisničkog iskustva i vizualnih elementa sustava. Priprema testiranja je vremenski manje zahtjevna, a potrebna je i manja razina vještina testera za provedbu željenog testa. Osigurati dostatnu pokrivenost testiranja je zahtjevno i utječe na vrijeme provedbe testiranja većeg broja instanci. Troškovi testnih aktivnosti rastu povećanjem broja provjera, a izvještaje stvara čovjek.

5. PRIKAZ MANUALNOG I AUTOMATSKOG TESTIRANJA NA ODABRANIM PRIMJERIMA

Odabirom grafičkog korisničkog sučelja kao predmeta testiranja, potrebno je stvoriti kratki pregled pristupa tom zadatku. Imajući u vidu sve dosad spomenute činjenice o procesu testiranja softvera, bit će provedeno testiranje grafičkih korisničkih sučelja manualnim i automatskim metodama. Važno je naglasiti kako će za potrebe manualnog testiranja biti korišteno korisničko sučelje prototipa aplikacije vlastite izrade, dok se za automatizaciju testiranja koriste servisi *Google* platforme.

5.1. Proces manualnog testiranja na primjeru vlastitog prototipa aplikacije

Koncept prototipa aplikacije *OneKey (1Key),* temelji se na potrebi za educiranjem zaposlenika o informacijskoj sigurnosti i primjeni alata koji omogućuju zaštitu korisničkih podataka u poslovnom okruženju. Prototip je izrađen kao oblik generičke poslovne aplikacije čije su funkcionalnosti primjenjive za veći broj korisnika. Segmentacija potrošača obuhvaća srednja i mala poduzeća na globalnoj razini. Temeljni korisnički zahtjevi vezani su uz a) sustav registracije i prijave u aplikaciju, b) unos i pohranu korisničkih pristupnih podataka, c) kopiranje i skrivanje, odnosno prikaz lozinki, d) generiranje kvalitetnih lozinki, e) ocjena vlastitih lozinki prema S.M.A.R.T. metodologiji, f) kviz o informacijskoj sigurnosti i g) tematsku igricu.

5.1.1. Planiranje manualnog testnog procesa

Integracijskim testiranjem prototipa aplikacije utvrđuje se pravilno funkcioniranje pojedinih komponenti, ali i njihovog sinergijskog učinka unutar cjeline. Cilj testiranja je potvrditi odgovara li ponašanje navedenih funkcionalnosti i pripadajućih grafičkih elemenata, specificiranim tehničkim pretpostavkama. Imajući na umu kako se radi o manualnom testiranju integracijske razine i grafičkom korisničkom sučelju kao predmetu testiranja, bit će korištene kombinacije adekvatnih vrsta testiranja i pripadajućih tehnika.

Provoditelj aktivnosti testiranja je ujedno i autor prototipa aplikacije. *Alpha testing* je odgovarajuća metoda pristupa testiranju s obzirom na dimenziju izvršitelja. Internalizacija procesa testiranja olakšava planiranje testiranja zbog poznavanja strukture programa. Adekvatna pokrivenost testiranja bit će postignuta detaljnim testiranjem svih dostupnih korisničkih sučelja i koraka koji se unutar svakog sučelja mogu poduzeti, koristeći *menu tour* i *path testing* metodologije. Dodatak spomenutim metodologijama je *feature of function*

integration testing. Istraživanjem sklada integriranih komponenti prototipa moguće je ovjeriti kvalitetu programskog koda i osjećaja koji pruža korištenje aplikacije. Unutar prototipa aplikacije je dominantna primjena polja unosa za provođenje temeljnih aktivnosti. Neizbježno je usmjeriti pažnju na testiranje potencijalnih nedostataka polja unosa pomoću dimenzije problema kao pretpostavke testiranja. Koristeći pripadajuće tehnike *input* i *output constraints* testiranja, unosit će se varijable čije vrijednosti odstupaju i ne odstupaju od postavljenih mogućnosti sustava. Ovim pristupom će se proučiti reakcije sustava i mogućnost prikaza rezultata unosom ispravnih i neispravnih podataka. Dimenzija aktivnosti definira način pristupa testiranju. Primjenom metoda aktivnosti, *scenario i smoke testing,* testiranje će biti usmjereno na simulaciju korisničkih aktivnosti unutar aplikacije. Važno je osigurati da proizvod zadovoljava temeljne potrebe korisnika i pruži sigurno korisničko iskustvo. Uspješnost navedenog procesa testiranja će se vrednovati usporedbom stvarnog i očekivanog ponašanja, za svaki pronađeni nedostatak pojedinačno.

5.1.2. Provedba manualnog testiranja

Provedba testiranja obuhvaća veći broj aktivnosti. Inicijalna stavka je postavljanje testnog okruženja. Potrebno je osigurati opremu kao što su računalo i stabilna mrežna veza, revidirati testne metode i korisničke zahtjeve te započeti provedbu.

Nakon zaključene provedbe testiranja, tester sastavlja izvještaj o testnom procesu. Izvještaj se sastoji o manjih cjelina, od kojih svaka prikazuje sučelje sa identificiranim nedostacima. Grupiranje nedostataka prema lokaciji omogućuje lakše snalaženje u fazi ispravljanja od strane razvojnog inženjera. Univerzalni podaci izvještaja su 1) podaci o izvršitelju testiranja, 2) podaci o aplikaciji i testnom okruženju te 3) datum provedbe testiranja. Pojedinačni dijelovi izvještaja o pronađenim nedostacima obuhvaćaju opis nedostatka, korake reprodukcije (engl. *way to reproduce - WTR)*, usporedbu stvarnog i očekivanog ponašanja, procjenu rizika, trajanje jedne instance reprodukcije te sugestiju za razvojnog inženjera.

Testno okruženje:

Izvršitelj testiranja: Ivan Beljan, *QA associate*

Aplikacija: 1Key, v1.0.0 *(programski kod u prilogu 1)*

Okruženje: `Intel Core i7−47l0MQ CPU @ 2.50GHz / RAM @ l6 GB / Al v ≈ l00 Mbit/s`

Datum: 01.09.2022.

Problem tekstualne kutije sustava prijave

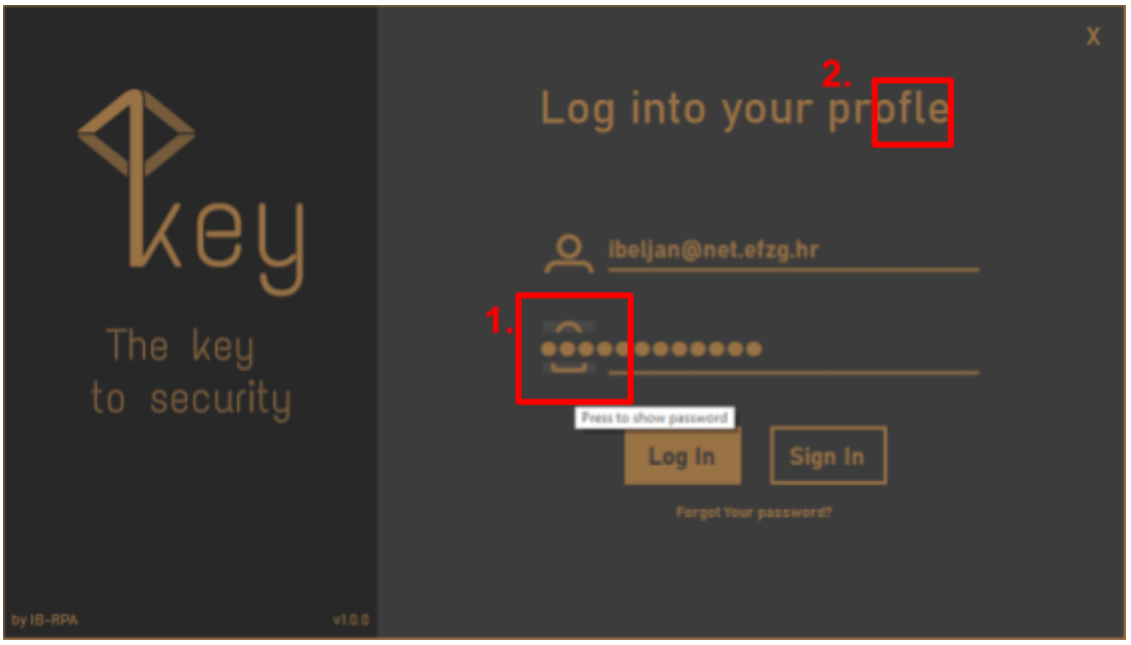
BUG: Preklapanje slojeva tekstualne kutije i piktograma polja unosa lozinke.

WTR:

1. Pokretanje aplikacije 1Key
2. Unos sadržaja u tekstualnu kutiju predviđenu za unos lozinke
3. Skeniranje ostalih elemenata sučelja

Stvarno stanje: Pri pokušaju unosa korisničke lozinke na za to predviđeno mjesto, manifestira se problem preklapanja slojeva; tekstualne kutije i piktograma polja unosa lozinke (1.). Prepoznata je i pravopisna greška riječi *"profile"* iz naslova sustava prijave (2.).

Slika 1: Problem tekstualne kutije sustava prijave



Očekivano stanje: Vertikalno poravnanje tekstualne kutije za unos korisničke lozinke u odnosu na tekstualnu kutiju za unos korisničkog imena. Pravopisna ispravnost svih natpisa aplikacije.

Procjena rizika: Vizualni nedostatak visoke razine zbog mogućnosti utjecaja na lozinku pristupa sustavu, ali i zbog stvaranja neadekvatnog inicijalnog dojma aplikacije.

Trajanje instance: 15 sekundi

Sugestija: Vertikalno poravnanje korespondentnih tekstualnih kutija radi stvaranja sigurnijeg inicijalnog dojma. Provođenje detaljnije provjere pravopisa.

Nedostaci upravitelja lozinki

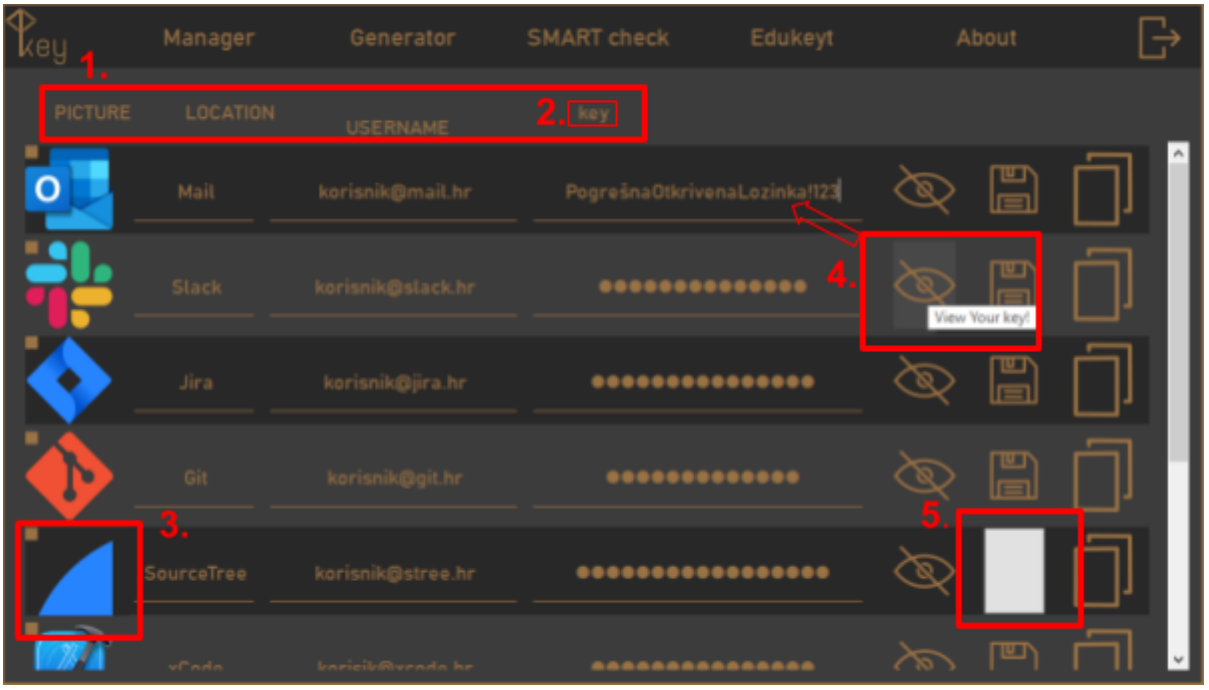
BUG: Razni vizualni i funkcionalni nedostaci sučelja upravitelja lozinki.

WTR:

1. Pokretanje aplikacije 1Key
2. Unos korisničkih podataka i pritiskanje gumba "LOG IN"
3. Popunjavanje svih dostupnih tekstualnih polja
4. Pritiskanje svakog dostupnog gumba za prikaz/skrivanje lozinke, pohranu korisničkih podataka i kopiranje lozinke u međuspremnik

Stvarno stanje: Inicijalnim skeniranjem aplikacije, utvrđuju se problemi; poravnanja naslova tekstualnih polja, nekonzistentnost veličine fontova (1. i 2.) te problem učitavanja piktograma pohrane korisničkih podataka jednog od redaka (5.). Nakon ispunjavanja dostupnih polja, prikazan je problem učitavanja vizualnog prikaza odredišta korisničkih podataka (3.). Funkcionalni se problem očituje prilikom pokušaja prikaza lozinke, otkrivajući lozinku krivog retka (4.).

Slika 2: Razni vizualni i funkcionalni nedostaci sučelja upravitelja lozinki



Očekivano stanje: Adekvatno poravnanje naslova tekstualnih polja, korektno učitavanje reprezentativnih fotografija odredišta i piktograma te ispravan prikaz ili skrivanje pripadajuće lozinke.

Procjena rizika: Vizualni su nedostaci niske razine rizika (1., 2., 3., 5.), zbog manjka utjecaja na rad sustava, dok funkcionalni problem prikaza/skrivanja lozinke (4.) nosi visok stupanj rizika jer znatno utječe na očekivani rad funkcionalnosti.

Trajanje instance: 5 minuta

Sugestija: Vertikalno i horizontalno poravnanje korespondentnih naslova tekstualnih kutija. Promjena načina prilagodbe učitane slike, pripadajućem okviru. Ponovno učitavanje piktograma pohrane podataka i spajanje funkcije prikaza/skrivanja lozinke s korespondentnim retkom.

Nekorektno generiranje lozinki

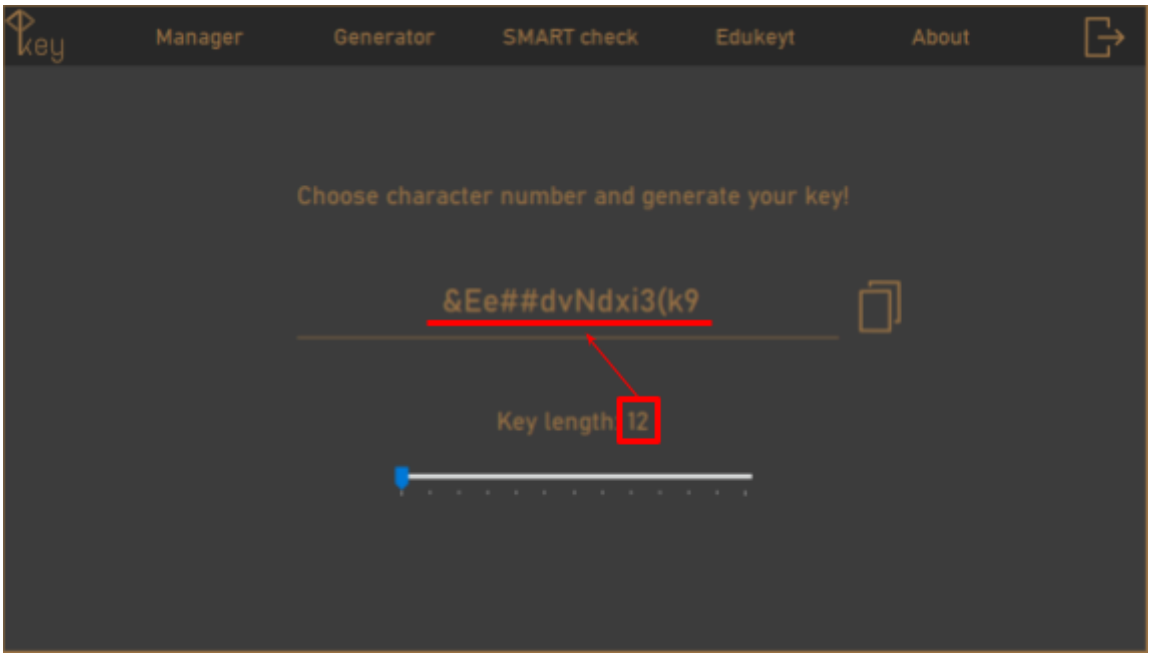
BUG: Nepodudaranje broja znakova generiranih lozinki.

WTR:

1. Pokretanje aplikacije 1Key
2. Unos korisničkih podataka i pritiskanje gumba “LOG IN”
3. Pristup generatoru lozinki, pritiskom na gumb “Generator”
4. Skeniranje stanja sučelja generatora lozinki

Stvarno stanje: Iako se na prvi pogled doima kao korektno stanje, prebrojavanjem znakova generiranih lozinki duljine 12 znakova, utvrđeno je odstupanje očekivanog (12) i stvarnog (15) broja znakova.

Slika 3: Nekorektno generiranje lozinki



Očekivano stanje: Generiranje lozinki podudarnog broja znamenki postavljenih kliznom trakom.

Procjena rizika: Funkcionalni nedostatak rizika visoke razine koji može uzrokovati problem izvan sustava aplikacije, primjenom od strane korisnika.

Trajanje instance: 30 sekundi

Sugestija: Usklađivanje postavki klizne trake s parametrima željene duljine lozinke, iz pripadajućeg tekstualnog polja.

Nepotpuni prikaz povratnih informacija sustava

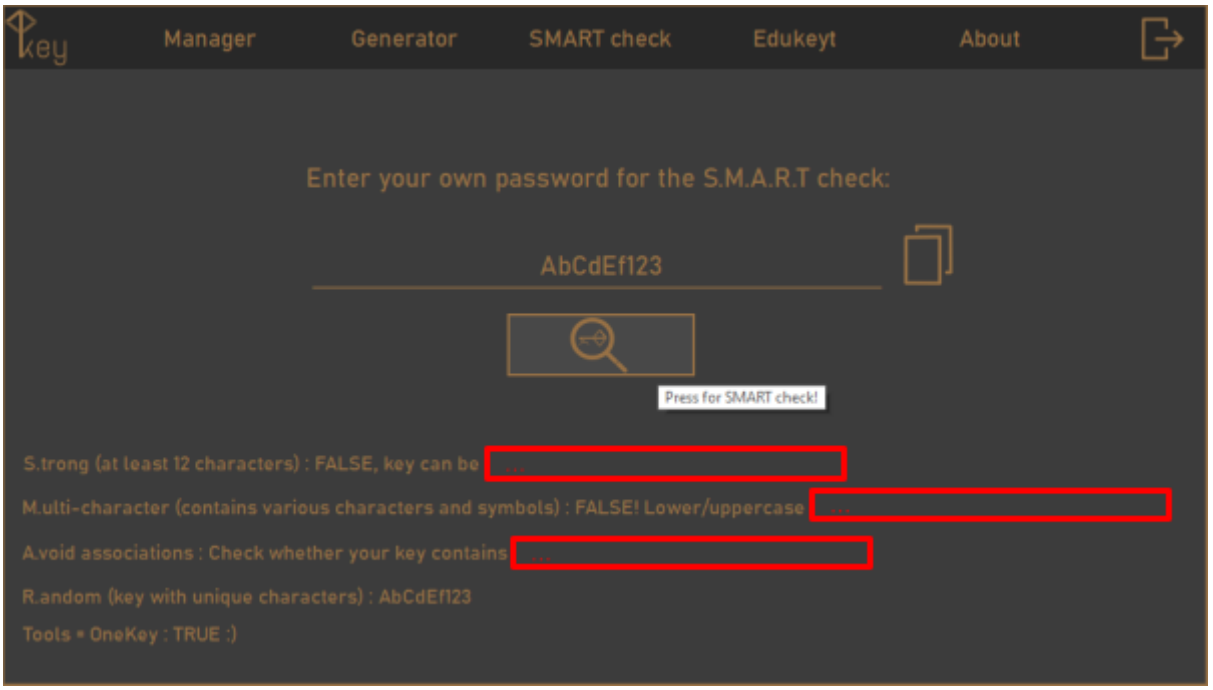
BUG: Nedostatan prikaz ocjene valjanosti i sugestija validirane lozinke.

WTR:

1. Pokretanje aplikacije 1Key
2. Unos korisničkih podataka i pritiskanje gumba "LOG IN"
3. Pristup verifikatoru lozinki, pritiskom na gumb "SMART check"
4. Unos nasumične lozinke
5. Pokretanje ovjere lozinke pritiskom na gumb povećala

Stvarno stanje: Prva tri polja, koja prikazuju povratnu informaciju sustava o ispravnosti korisničke lozinke, prikazuju nepotpune činjenice.

Slika 4: Nepotpuni prikaz povratnih informacija sustava



Očekivano stanje: Potpuni prikaz činjenica o ispitanoj lozinki.

Procjena rizika: Rizik visoke razine koji onemogućuje korisniku ispravan pregled suda provjerene lozinke. Radi se o izravnom narušavanju planirane funkcionalnosti sustava.

Trajanje instance: 25 sekundi

Sugestija: Proširivanje nedostatnih tekstualnih polja do maksimalne moguće razine, kako bi se omogućio potpuni prikaz činjenica ispitanih lozinki.

Neispravan prikaz odgovora unutar kviza

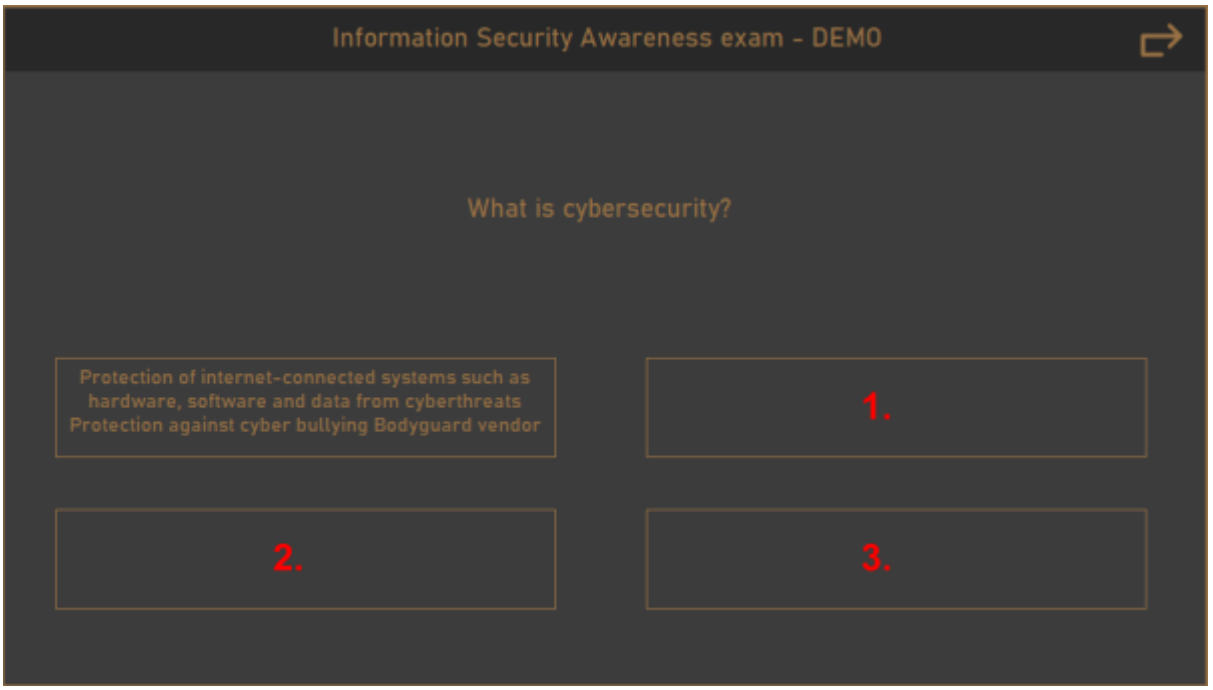
BUG: Prikaz svih dostupnih odgovara unutar jednog gumba.

WTR:

1. Pokretanje aplikacije 1Key
2. Unos korisničkih podataka i pritiskanje gumba “LOG IN”
3. Pristup edukacijskom sučelju pritiskom na gumb “Edukeyt”
4. Pokretanje kviza pritiskom na gumb “Take the quiz!”
5. Skeniranje stanja prikaza kviza

Stvarno stanje: Svi odgovori na postavljeno pitanje kviza su, umjesto na zasebnim gumbima, prikazani unutar jednog gumba.

Slika 5: Neispravan prikaz odgovora unutar kviza



Očekivano stanje: Zasebni prikaz odgovora unutar korespondentnih ćelija.

Procjena rizika: Rizik srednje visoke razine pošto se radi o nepotpunom demo kvizu, čiji utjecaj nije povezan s temeljnim funkcioniranjem sustava.

Trajanje instance: 30 sekundi

Sugestija: Provjera mehanike pridruživanja pojedinačnog odgovora sa zasebnim gumbom.

Problem prikaza protagonista igre

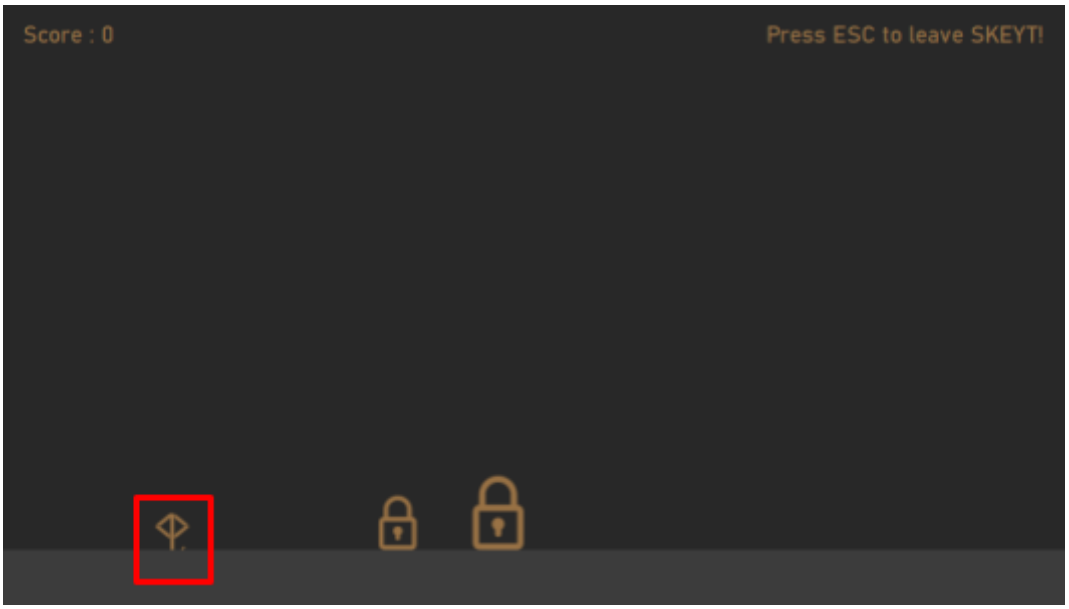
BUG: Narušeni prikaz protagonista igre uzrokovan vertikalnim poravnanjem.

WTR:

1. Pokretanje aplikacije 1Key
2. Unos korisničkih podataka i pritiskanje gumba "LOG IN"
3. Pristup sučelju igrice pritiskom na piktogram aplikacije u gornjem lijevom kutu
4. Skeniranje stanja sučelja

Stvarno stanje: Protagonist igre, element ključa iz loga aplikacije, nedostatno je poravnat. Prikazan je u relativnoj pozadini sučelja, na niskoj poziciji, onemogućujući preskakanje prepreka.

Slika 6: Problem prikaza protagonista igre



Očekivano stanje: Ispravno vertikalno poravnanje na temelju kojeg je donji rub protagonista neposredno iznad postavljenog tla unutar igre.

Procjena rizika: Uzimajući u obzir prirodu igrice i njene uloge u sustavu, radi se o niskoj razini rizika. Igrica služi kao svojevrsna zabavna dosjetka, radi stvaranja pozitivne slike u svijesti korisnika. Njeno (ne)funkcioniranje se ne kosi s ciljevima aplikacije.

Trajanje instance: 30 sekundi

Sugestija: Prilagodba dinamičkog sustava poravnanja elemenata igre.

5.1.3. Analiza rezultata manualnog procesa testiranja

Pristup manualnog testiranja omogućuje provjeru dojma kojeg aplikacija ostavlja korištenjem. Testiranjem prototipa aplikacije *OneKey,* utvrđeni su brojni vizualni nedostaci koji narušavaju opće korisničko iskustvo. Pronalazak nedostataka je ponajviše ostvaren primjenom kombinacija metode *menu tour* i testiranja pomoću *path, feature of function integration* i *scenario testing* tehnika. Pronađeni nedostaci su grupirani po sučeljima umjesto po vrsti kako bi navigiranje i lociranje bilo jednostavnije prilikom provedbe popravka.

Većina utvrđenih nedostataka očituje se u vizualnim komponentama pojedinih dijelova sučelja, narušavajući pritom ukupni dojam. U određenim instancama se vizualni nedostaci prelijevaju i na funkcionalne aspekte aplikacije. Takvim nedostacima se dodjeljuje veća razina rizika i viši stupanj prioritizacije prilikom procesa popravljanja.

Najviši prioritet se dodjeljuje pri ispravku nedostataka 1) sučelja za pristup aplikaciji unutar kojeg je narušen inicijalni dojam i mogućnost unošenja lozinke, 2) sučelja upravljanja lozinkama, gdje se pritiskom na gumb otkrivanja lozinke prikaže lozinka krivog retka, 3) sučelja generatora lozinki u kojem broj prikazanih znakova odstupa od postavljenih. Ostalim prijavljenim nedostacima je dodijeljena niža razina prioriteta te će biti ispravljeni nakon što se osigura ispravnost grešaka višeg stupnja rizika. Prikaz ispravljenih korisničkih sučelja moguće je istražiti unutar priloga 2.

5.2. Proces automatskog testiranja na primjeru Google servisa

Iako su zahtjevnost i vremenska intenzivnost pripreme ove vrste testa visoki, njezin je učinak teško zamjenjiv kada se radi o provedbi velikog broja, repetitivnih instanci testiranja. Sukladno tomu, na primjerima određenih usluga *Google* servisa; *Google* tražilice i *Google* elektroničke pošte, *Gmail*, bit će prikazana provedba testiranje komponenti sučelja automatskom metodom.

5.2.1. Primijenjeni alati i testni objekti

Izvršenje automatskog testiranja provedeno je kombinacijom raznovrsnih alata i pomagala. Proces učenja, i ispravljanja nastalih grešaka tijekom oblikovanja programskog koda, temelji se na edukativnom tečaju dostupnom na video servisu *YouTube,* pod nazivom *Selenium Course for Beginners - Web Scraping Bots, Browser Automation, Testing (Tutorial)*. Uz tečaj, korištena je i web stranica temeljena na principu pitanja i odgovora, namijenjena za

rješavanje lokalnih razvojnih problema uz pomoć iskustava drugih korisnika - *Stack overflow - PyCharm topics* (Chandrasekar, 2008). Alat koji je omogućio provedbu programske logike je *Python 3.9* - programski jezik opće primjene, korišten pri izradi softverskih rješenja raznih domena. Da bi se navedeni programski jezik mogao efikasno koristiti, nužno je postaviti integrirano razvojno okruženje koje omogućuje primjenu raznolikih alata baziranih na odabranom programskom jeziku. Skupina alata, obuhvaćenih okruženjem *Pycharm* i osposobljenih korištenjem okvira za testiranje *Selenium,* idealni su za primjenu ove vrste automatskog testiranja mrežnih aplikacija. Najbitnija svojstva navedenog okruženja su mogućnost identifikacije i upravljanje *html* elementima testiranih komponenti sustava. Uz navedene alate, važna je i primjena terminala dostupnog na računalu. Proširujući mogućnosti programskog jezika, integrirano je pokretanje testiranja i generiranje izvještaja, upravo kroz naredbeni redak (engl. *command prompt).*

Objekti čije će se funkcionalnosti testirati alatom za automatizaciju jesu; tražilica *Google* servisa te sustavi prijave i odjave elektroničke pošte, *Gmail.* Komponenta tražilice koja će biti testirana je tekstualna kutija za unos željenog pojma pretrage, i pripadajuće pokretanje procesa pretrage. Testiranjem ovog elementa utvrđuju se ograničenja unosa pojmova i mogućnost obrade raznolikih, unesenih varijabli. Prilikom testiranja sustava prijave i odjave, osim testiranja mogućnosti unosa, bit će istražena logika validacije lozinki, kao kritične točke ovog sustava.

5.2.2. Planiranje automatizacije testiranja

Korištenjem *unit testing* razine testiranja provjerava se pojedina ili specifična funkcionalnost sustava. Tako je moguće izolirati dijelove sustava radi provjere njihove integriranosti. Unutar grafičkog sučelja navedenih usluga *Google* servisa, izoliraju se sustavi pretrage i prijave da bi se provelo testiranje tih zasebnih komponenti.

Pošto se testiranje provodi nad grafičkim korisničkim sučeljima, odnosno njihovim dijelovima, jasno je kako će pri tome biti korištene već spomenute metode testiranja. Pri automatizaciji testiranja će ipak biti značajniji naglasak testiranja toka podataka i polja unosa unutar pojedinih sučelja. Temeljna razlika u odnosu na manualno testiranje, ostvaruje se proširenjem dimenzija pokrivenosti i aktivnosti testiranja. Uvođenjem testiranja izolirane funkcionalnosti sučelja (engl. *function testing),* proširena je pokrivenost provedenog testiranja. Dimenzija aktivnosti se obogaćuje mogućnošću primjene testiranja performansi

aplikacije (engl. *performance testing)*. Temeljem reakcija testiranih komponenti sustava na ponavljajuće instance automatiziranih aktivnosti, moguće je izvesti zaključak o sposobnosti i kvaliteti programskog koda testiranih funkcionalnosti.

Nakon uspješne provedbe planiranja testnih aktivnosti, slijedi opis testnog okruženja. Opis uključuje 1) izvršitelja testiranja, 2) objekte testiranja, 3) strojnu i programsku opremu, 4) korištene alate i 5) podatke datuma provedbe testiranja.

Testno okruženje:

Izvršitelj testiranja: Ivan Beljan, *QA associate*

Objekt 1: *Google pretraživač*

Objekt 2: *Sustav prijave i odjave servisa Gmail*

Okruženje: `Intel Core i7−4710MQ CPU @ 2.50GHz / RAM @ 16 GB / Al v ≈ 100 Mbit/s`

Alat 1: *Python 3.9*

Alat 2: *Selenium*

Razvojno okruženje: *PyCharm - Python IDE*

Datum: 04.09.2022.

5.2.3. Interpretacija provedenog testiranja na primjeru tražilice

Priprema testiranja započinje istraživanjem slijeda kojim podaci prolaze kroz proces pretrage. Proces se odvija koracima 1) učitavanjem tražilice, 2) unosom željenog pojma pretrage, 3) pokretanjem pretrage i 4) prikazom rezultata pretrage.

Početni korak primjene automatizacije testiranja je odabir mrežnog preglednika za pokretanje *Google* tražilice. Po učitavanju preglednika potrebno je identificirati *html* elemente tražilice - tekstualno polje unosa. Prilikom identifikacije, važno je obratiti pažnju na unikatnu nomenklaturu unutar *html* sustava, kako bi se programskom alatu naredilo lociranje komponente na temelju jedinstvene oznake - *id.* Da bi se instanca testa provela u potpunosti, potrebno je unijeti željenu varijablu u locirano tekstualno polje te pokrenuti pretragu. Ovisno o tome hoće li sustav uspjeti pokrenuti pretragu ili ne, moguće je razlučiti kakvoću testiranog elementa.

Slika 7: Isječak programskog koda automatizacije tražilice

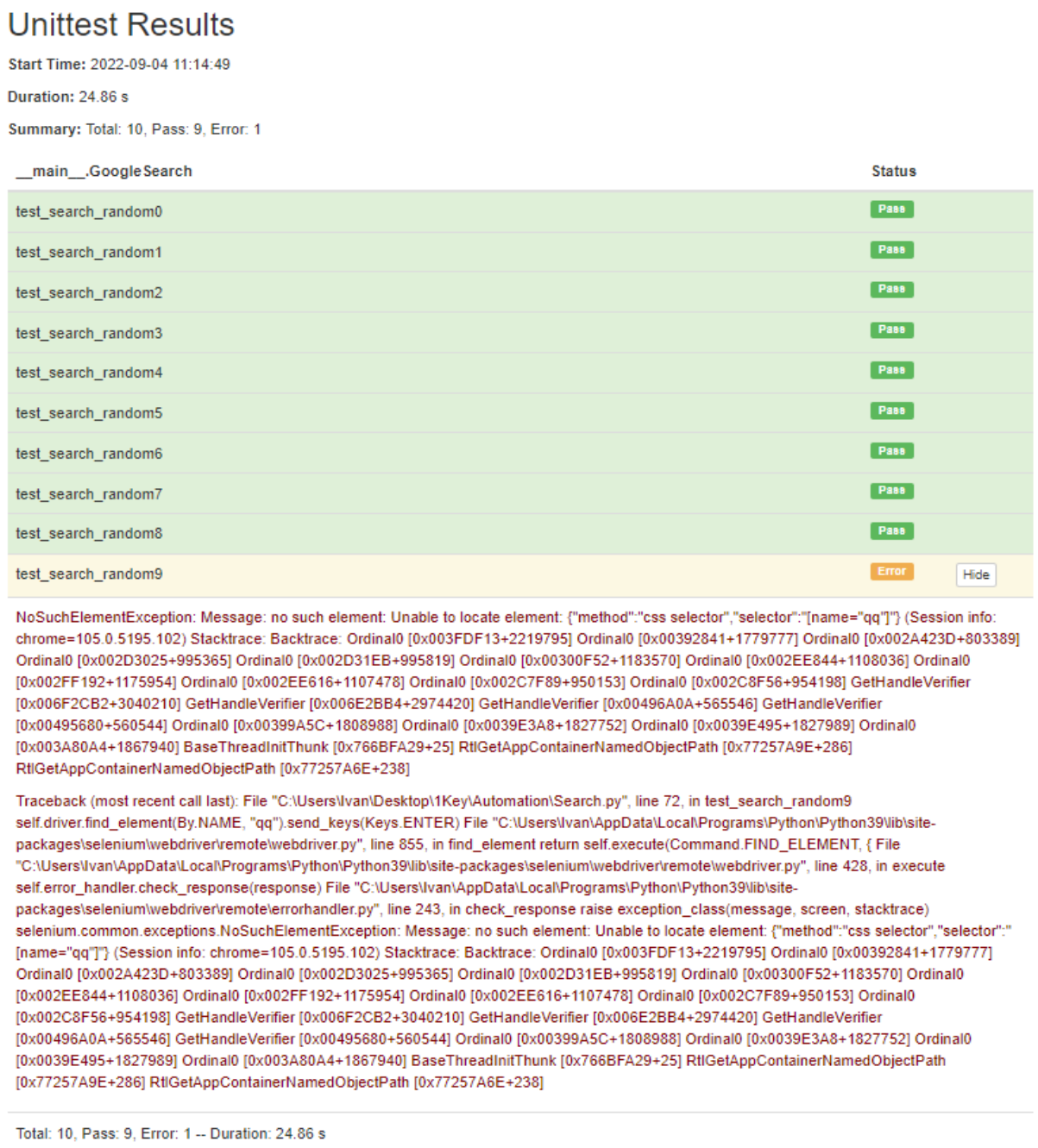
```
import os
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
import unittest
import HtmlTestRunner
import random

random_search = ['aaaaaaaHJGFJVKUZFTDzuftftzfd65R&778o8709gbz7h798h87gftff', 'abc',
                 'aa bb  cc  dd  ee  ff  gg  hh  II  JJ  11  22  33  -- 55  ++', 'ABC',
                 '12Cdefg', 'Qaqaqaqaqaqa', 'prQAefgh', 'QA test automation job', 'GUI',
                 'User experience', '     ', '', 'efzgggggggg', 'Human-computercommunication']

class GoogleSearch(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        os.environ['PATH'] += r"C:/Users/Ivan/Desktop/1Key/SeleniumDrivers"
        cls.driver = webdriver.Chrome()
        cls.driver.implicitly_wait(10)
        cls.driver.maximize_window()

    def test_search_random0(self):
        self.driver.get("https://google.com")
        self.driver.find_element(By.ID, "W0wltc").click()
        self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
        self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)
```

Iz priloženog isječka programskog koda slike 7, moguće je prepoznati pojedine komponente strukture koda. Prvi odjeljak koda se odnosi na preuzimanje alata i funkcija te postavljanje dijela okruženja testiranja. Posebno je važan alat za generiranje izvještaja *unittest,* funkcija identifikacije komponenti sučelja *By* i pomagalo za automatski unos pojmova, *keys.*

Nakon toga je varijabli *random_search* pridružen skup pojmova koji će biti nasumično uneseni u tražilicu. Skup pojmova je reprezentativan jer se sastoji od kombinacija dostupnih znakova, simbola, velikih i malih slova, brojeva i praznina. Cjelokupni programski kod ovog dijela testiranja, vidljiv je u prilogu 3.

Svih deset automatiziranih testnih slučajeva prolaze identičan ciklus; pokretanje tražilice, unos nasumičnog pojma i pokretanje pretrage. Pošto je sustav za to osposobljen, test se pokreće unosom odgovarajuće naredbe u naredbenom retku, nakon čega se u kratkom vremenu provodi testni proces i generiranje izvještaja, prikazanog slikom 8.

Slika 8: Prikaz generiranog izvještaja testiranja tražilice



Unittest Results

Start Time: 2022-09-04 11:14:49

Duration: 24.86 s

Summary: Total: 10, Pass: 9, Error: 1

__main__.GoogleSearch	Status
test_search_random0	Pass
test_search_random1	Pass
test_search_random2	Pass
test_search_random3	Pass
test_search_random4	Pass
test_search_random5	Pass
test_search_random6	Pass
test_search_random7	Pass
test_search_random8	Pass
test_search_random9	Error Hide

NoSuchElementException: Message: no such element: Unable to locate element: {"method":"css selector","selector":"[name="qq"]"} (Session info: chrome=105.0.5195.102) Stacktrace: Backtrace: Ordinal0 [0x003FDF13+2219795] Ordinal0 [0x00392841+1779777] Ordinal0 [0x002A423D+803389] Ordinal0 [0x002D3025+995365] Ordinal0 [0x002D31EB+995819] Ordinal0 [0x00300F52+1183570] Ordinal0 [0x002EE844+1108036] Ordinal0 [0x002FF192+1175954] Ordinal0 [0x002EE616+1107478] Ordinal0 [0x002C7F89+950153] Ordinal0 [0x002C8F56+954198] GetHandleVerifier [0x006F2CB2+3040210] GetHandleVerifier [0x006E2BB4+2974420] GetHandleVerifier [0x00496A0A+565546] GetHandleVerifier [0x00495680+560544] Ordinal0 [0x00399A5C+1808988] Ordinal0 [0x0039E3A8+1827752] Ordinal0 [0x0039E495+1827989] Ordinal0 [0x003A80A4+1867940] BaseThreadInitThunk [0x766BFA29+25] RtlGetAppContainerNamedObjectPath [0x77257A9E+286] RtlGetAppContainerNamedObjectPath [0x77257A6E+238]

Traceback (most recent call last): File "C:\Users\Ivan\Desktop\1Key\Automation\Search.py", line 72, in test_search_random9 self.driver.find_element(By.NAME, "qq").send_keys(Keys.ENTER) File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\webdriver.py", line 855, in find_element return self.execute(Command.FIND_ELEMENT, { File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\webdriver.py", line 428, in execute self.error_handler.check_response(response) File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\errorhandler.py", line 243, in check_response raise exception_class(message, screen, stacktrace) selenium.common.exceptions.NoSuchElementException: Message: no such element: Unable to locate element: {"method":"css selector","selector":"[name="qq"]"} (Session info: chrome=105.0.5195.102) Stacktrace: Backtrace: Ordinal0 [0x003FDF13+2219795] Ordinal0 [0x00392841+1779777] Ordinal0 [0x002A423D+803389] Ordinal0 [0x002D3025+995365] Ordinal0 [0x002D31EB+995819] Ordinal0 [0x00300F52+1183570] Ordinal0 [0x002EE844+1108036] Ordinal0 [0x002FF192+1175954] Ordinal0 [0x002EE616+1107478] Ordinal0 [0x002C7F89+950153] Ordinal0 [0x002C8F56+954198] GetHandleVerifier [0x006F2CB2+3040210] GetHandleVerifier [0x006E2BB4+2974420] GetHandleVerifier [0x00496A0A+565546] GetHandleVerifier [0x00495680+560544] Ordinal0 [0x00399A5C+1808988] Ordinal0 [0x0039E3A8+1827752] Ordinal0 [0x0039E495+1827989] Ordinal0 [0x003A80A4+1867940] BaseThreadInitThunk [0x766BFA29+25] RtlGetAppContainerNamedObjectPath [0x77257A9E+286] RtlGetAppContainerNamedObjectPath [0x77257A6E+238]

Total: 10, Pass: 9, Error: 1 -- Duration: 24.86 s

Po provedbi testiranja, generirane su dvije instance izvještaja. Prva instanca je odmah vidljiva u naredbenom retku, dostupna u prilogu 4. Radi se o nestrukturiranom ispisu rezultata testiranja. Preglednija verzija izvještaja generirana je u programu *Pycharm*, a pokreće se u mrežnom pregledniku. Navedeni izvještaj sa slike 8, prikazuje datum testiranja, trajanje i sažetak rezultata. Za svaku instancu testiranja prikazan je podataka o uspješnosti. Iz

prikazanog primjera, vidljiva je 90%-tna uspješnost provedenog testiranja, za 10 testnih slučajeva, u svega 24.88 sekundi. U slučaju neuspješnog testnog slučaja, moguće je proširiti ispis i provjeriti stanje. Unutar prvog retka krije se uzrok problema: sustav ne uspijeva identificirati element naziva "*qq"*. Provjerom programskog koda utvrđuje se kako je prijavljena greška primjer *false negative* situacije u kojoj nije došlo do pogreške sustava nego ljudske pogreške. Tipfelerom je umjesto identifikacijske oznake tražilice *q* pretražena komponenta naziva *qq* što dovodi do greške.

Zaključak testiranja se izvodi nakon ispravka tipfelera i ponovnog pokretanja testa. Komponenta tekstualne kutije unosa, unutar sučelja *Google* tražilice, funkcionira uz 100%-tni učinak. To pak znači da je navedena komponentna izrazito fleksibilna, sposobna obraditi podatke različitih veličina i duljina te prepoznati pojmove raznovrsnih simbola i znakova.

5.2.4. Interpretacija provedenog testiranja na primjeru sustava prijave

Nešto kompleksnijih ciklus podataka, odvija unutar sustava prijave i odjave elektroničke pošte, *Gmail.* Ponovo je prvi korak učitavanje sučelja sustava prijave. Potom je potrebno identificirati i pravilno razlikovati polje unosa korisničkog imena i tipke za nastavak procesa. Idućim se prikazom omogućuje unos lozinke i prijave u sustav. Nakon što se učita elektronička pošta, pokreće se proces odjave i zaključuje kružni tok podataka. U ovom se toku očituje važnost pravilne identifikacije elemenata zbog postojanja više različitih polja unosa i tipki različitih funkcija. Proces automatizacije navedene komponente, izražen programskim kodom, dostupan je u prilogu 5.

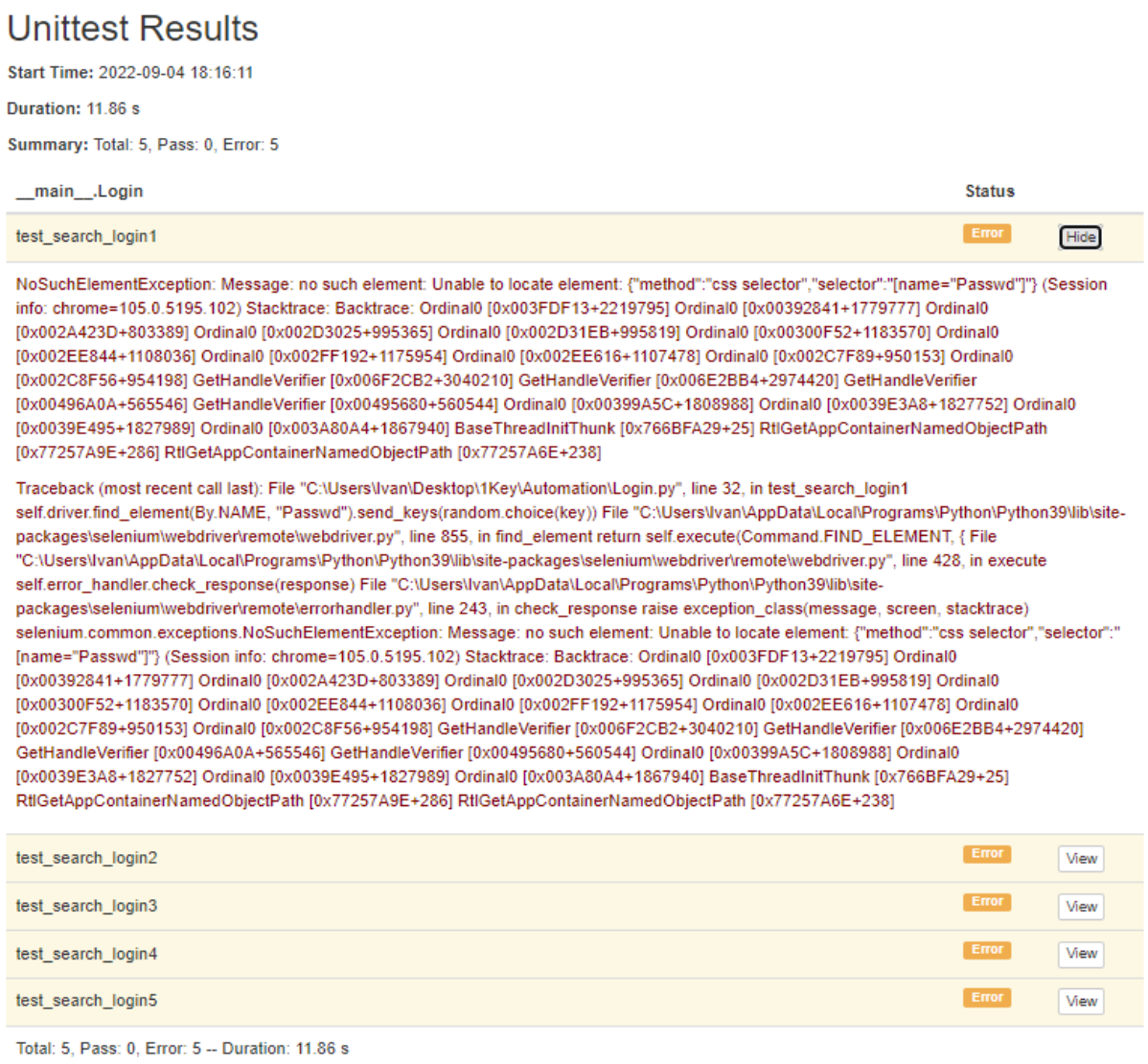
Ponovno je moguće prepoznati kvalitetnu strukturu koda. Od postavljanja specifikacija okruženja i dodjele skupine reprezentativnih pojmova željenim varijablama, do jedne od instanci testiranja. Osim što se testira tok podataka sustava prijave, naglasak je na verifikatoru lozinki sustava. Navedena se specijalizacija provodi korištenjem ispravnog korisničkog imena i jedne od nasumičnih lozinki. Među lozinkama se nalazi ispravna varijanta i veći broj neispravnih inačica, s manjim ili većim promjenama.

Slika 9: Isječak programskog koda automatizacije sustava prijave

```
import os
from selenium import webdriver
from selenium.webdriver.common.by import By
import unittest
import HtmlTestRunner
from time import sleep
import random

chrome_options = webdriver.ChromeOptions()
chrome_options.add_argument('--incognito')

username = ['test.automation.korisnik@gmail.com']

key = ['TestAutomation2022!', 'Testautomation2022!', 'testAutomation2022!',
       'testautomation2022!', 'TESTAUTOMATION2022!', 'tESTaUTOMATION2022!',
       'Testautonation2022!', 'testaautomation2022!', 'TestAutomation2220222!',
       'tesAutomation2022!', 'Test Automation2022!', '!TestAutomation2022']

class Login(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        os.environ['PATH'] += r"C:/Users/Ivan/Desktop/1Key/SeleniumDrivers"
        cls.driver = webdriver.Chrome(options=chrome_options)
        cls.driver.maximize_window()

    def test_search_login1(self):
        self.driver.get("https://gmail.google.com")
        self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
        self.driver.find_element(By.ID, "identifierNext").click()
        sleep(3)
        self.driver.find_element(By.NAME, "Passwd").send_keys(random.choice(key))
        self.driver.find_element(By.ID, "passwordNext").click()
        sleep(3)
        self.driver.get("https://accounts.google.com/Logout?hl=en&continue=https:"
                        "//www.google.com/search%3Fq%3Dgmail%26oq%3Dgmail%26aqs%3Dchrome"
                        "..69i57j0i271l2j69i65l2.670j0j1%26sourceid%3Dchrome%26ie%3DUTF-"
                        "8&timeStmp=1662305131&secTok=.AG5fkS_BFBrvaFMtIHLOoXLSG7k5ArRVvw&ec=GAdAAQ")
        sleep(3)
        self.driver.find_element(By.ID, "W0wltc").click()
```

Za potrebe testiranja, stvoren je testni korisnički račun sa stvarnim podacima. Testni proces validacije se sastoji od pet instanci testiranja sustava prijave. Na temelju rezultata moguće je iznijeti jasne zaključke o njegovoj kvaliteti. Unutar 11.86 sekundi, sustav je pet puta neuspješno pokušao izvršiti proces prijave i odjave. Iako rezultati testiranja prikazuju neuspješne instance, interpretacija izvještaja daje pozitivne rezultate. Izvještaj naredbenog retka dostupan je u prilogu 6, dok je pregledniji izvještaj prikazan u nastavku, slikom 10.

Slika 10: Generirani izvještaj testiranja sustava prijave



Unittest Results

Start Time: 2022-09-04 18:16:11

Duration: 11.86 s

Summary: Total: 5, Pass: 0, Error: 5

__main__.Login	Status	
test_search_login1	Error	Hide

NoSuchElementException: Message: no such element: Unable to locate element: {"method":"css selector","selector":"[name="Passwd"]"} (Session info: chrome=105.0.5195.102) Stacktrace: Backtrace: Ordinal0 [0x003FDF13+2219795] Ordinal0 [0x00392841+1779777] Ordinal0 [0x002A423D+803389] Ordinal0 [0x002D3025+995365] Ordinal0 [0x002D31EB+995819] Ordinal0 [0x00300F52+1183570] Ordinal0 [0x002EE844+1108036] Ordinal0 [0x002FF192+1175954] Ordinal0 [0x002EE616+1107478] Ordinal0 [0x002C7F89+950153] Ordinal0 [0x002C8F56+954198] GetHandleVerifier [0x006F2CB2+3040210] GetHandleVerifier [0x006E2BB4+2974420] GetHandleVerifier [0x00496A0A+565546] GetHandleVerifier [0x00495680+560544] Ordinal0 [0x00399A5C+1808988] Ordinal0 [0x0039E3A8+1827752] Ordinal0 [0x0039E495+1827989] Ordinal0 [0x003A80A4+1867940] BaseThreadInitThunk [0x766BFA29+25] RtlGetAppContainerNamedObjectPath [0x77257A9E+286] RtlGetAppContainerNamedObjectPath [0x77257A6E+238]

Traceback (most recent call last): File "C:\Users\Ivan\Desktop\1Key\Automation\Login.py", line 32, in test_search_login1 self.driver.find_element(By.NAME, "Passwd").send_keys(random.choice(key)) File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\webdriver.py", line 855, in find_element return self.execute(Command.FIND_ELEMENT, { File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\webdriver.py", line 428, in execute self.error_handler.check_response(response) File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\errorhandler.py", line 243, in check_response raise exception_class(message, screen, stacktrace) selenium.common.exceptions.NoSuchElementException: Message: no such element: Unable to locate element: {"method":"css selector","selector":"[name="Passwd"]"} (Session info: chrome=105.0.5195.102) Stacktrace: Backtrace: Ordinal0 [0x003FDF13+2219795] Ordinal0 [0x00392841+1779777] Ordinal0 [0x002A423D+803389] Ordinal0 [0x002D3025+995365] Ordinal0 [0x002D31EB+995819] Ordinal0 [0x00300F52+1183570] Ordinal0 [0x002EE844+1108036] Ordinal0 [0x002FF192+1175954] Ordinal0 [0x002EE616+1107478] Ordinal0 [0x002C7F89+950153] Ordinal0 [0x002C8F56+954198] GetHandleVerifier [0x006F2CB2+3040210] GetHandleVerifier [0x006E2BB4+2974420] GetHandleVerifier [0x00496A0A+565546] GetHandleVerifier [0x00495680+560544] Ordinal0 [0x00399A5C+1808988] Ordinal0 [0x0039E3A8+1827752] Ordinal0 [0x0039E495+1827989] Ordinal0 [0x003A80A4+1867940] BaseThreadInitThunk [0x766BFA29+25] RtlGetAppContainerNamedObjectPath [0x77257A9E+286] RtlGetAppContainerNamedObjectPath [0x77257A6E+238]

test_search_login2	Error	View
test_search_login3	Error	View
test_search_login4	Error	View
test_search_login5	Error	View

Total: 5, Pass: 0, Error: 5 -- Duration: 11.86 s

Pošto je temeljni cilj testiranja verifikacija validatora lozinki, može se zaključiti kako je navedeni sustav izrazito osjetljiv na odstupanja znakova. Unatoč tome što se u skupu nasumičnih lozinki nalaze lozinke s minimalnim odstupanjima od stvarne vrijednosti, sustav niti u jednoj instanci testiranja nije uspio pokrenuti prikaz elektroničke pošte. Greška koja se u izvještaju ističe također je interpretirana kao nemogućnost pronalaska navedene komponente *"Passwd"*. Uzrok tomu je logički slijed automatizacije. Nakon unosa lozinke i aktivacije gumba prijave, sustav očekuje prikaz elektroničke pošte i iniciranje odjave. Unosom krive lozinke i aktivacijom gumba prijave, ponovno će se prikazati sučelje unosa lozinke, što će dovesti do javljanja greške.

5.3. Osvrt provedenog procesa manualnog i automatskog testiranja

Provedba manualnog i automatskog testiranja dovodi do izvođenja očekivanog zaključka. Raspisivanjem teorijskih pretpostavki testiranja grafičkih korisničkih sučelja i usporednom raščlambom manualnog i automatskog testiranja, postavljene su pretpostavke o efikasnosti pojedine metodologije testiranja. Pretpostavke se odnose na testiranje grafičkih korisničkih sučelja, a potvrđene su primjenom navedenih testnih metodologija. U kojoj se mjeri ostvaruju teorijska načela, bit će opisano u nastavku.

Prilikom manualnog pristupa testiranju, uspješno su provjerene sve dostupne komponente pojedinih korisničkih sučelja aplikacije *OneKey*. Detaljan prolazak svake komponente sučelja i poduzimanje slijednih koraka kako bi se došlo do pojedinog stanja, iznjedrilo je nedostatke koji nisu bili uočljivi na prvi pogled. Osim uočavanja nedostataka, primjena spomenutih aktivnosti omogućuje upoznavanje funkcionalnosti aplikacije. Stjecanje navike pri provedbi testiranja omogućuje percipiranje drugih nedostataka. Ako se aplikacija učestalo koristi prilikom testiranja, provoditelj testiranja oblikuje referentnu točku funkcioniranja aplikacije. Svakim daljnjim testiranjem se lako uoče odstupanja od referentne točke. Ostvarena nesmetanost korištenja aplikacije *OneKey,* dokazuje se brzom provedbom svih iniciranih aktivnosti prilikom testiranja.

Automatizacija testiranja počiva na sličnim pretpostavkama procesa testiranja kao i manualno testiranje. Krajnji cilj obiju metodologija se očituje u ovjeri kvalitete grafičkog korisničkog sučelja no do ostvarenja tog cilja dolazi se drukčijim putem. Automatsko testiranje manje je pogodno za testiranje grafičkih elemenata sučelja. Njihova se prednost očituje prilikom testiranja funkcionalnosti koje se kriju u pozadini pojedine komponente. U slučaju testiranja dijelova *Google* servisa, lako je uočiti djelotvornost primjene ove metodologije. Provedba cijelog testnog slučaja većim brojem instanci i generiranje pripadajućeg izvještaja na zadanom primjeru, omogućeno je unutar jedne minute.

Temeljna razlika automatiziranog i manualnog testiranja očituje se u mogućnostima pojedine metodologije. Automatizacija većinom počiva na mogućnostima programskog koda, dok je okvir manualnog testiranja zadan vještinama provoditelja testa. Programski kod nije sposoban percipirati dojam i korisničko iskustvo kao tester, stoga se ova metodologija primjenjuje za testiranje sustavnih postavki, kao što su performanse i opterećenje programa. Provedba velikog broja instanci testiranja u kratkom vremenu, stvara dodatno opterećenje na

testno okruženje. Uspješna provedba testiranja unatoč otežanim uvjetima, dovodi do pozitivnih zaključaka o stanju sustava.

Osim funkcionalnih razlika, važne su i korporativne razlike u primjeni dviju metodologija testiranja. Troškovi i vrijeme zauzimaju vrhovno mjesto pri donošenju odluka o primjeni pristupa testiranju. Za potrebe demonstracije pojedinih testnih metoda, korišteni su besplatni alati i neplaćeno vrijeme provoditelja. U slučaju stvarne primjene, pokazatelji troška i vremena igraju ključnu ulogu. Pri manualnom testiranju, troškovi testiranja rastu učestalim provođenjem aktivnosti zbog vremenske zahtjevnosti provedbe testiranja. Inicijalno skuplji pristup je automatizacija testiranja. Zahtjevi primjene automatskog testiranja su odgovarajuća oprema, alati, edukacije i veće vještine testera. Svi navedeni zahtjevi odražavaju se na rast ukupnog troška primjene automatizacije. Troškovi automatizacije opadaju proporcionalno s rastom broja provedenih testiranja.

Prilikom planiranja testnog procesa potrebno je uzeti u obzir brojne elemente. Vrsta testnog objekta, provoditelj testiranja, budžet i vremenski okviri znatno utječu na konačnu odluku. Iako je u određenim instancama moguće koristiti oba pristupa, kao u slučaju testiranja polja unosa, potrebno je odabrati onu metodu koja će zadatak ispuniti efikasnije na svim razinama. U idealnim uvjetima, testiranje se provodi kombinacijom automatskog i manualnog testiranja kako bi se osigurala potpuna pokrivenost testiranja.

6. ZAKLJUČAK

Globalni napredak i želja za inovacijama potiče ljude na stvaranje nevjerojatnih dostignuća u području informacijske tehnologije. Integracija raznolikih uređaja u sve sfere ljudskog života dovodi do optimizacije načina interakcije koja se ostvaruje tehnologijom. Iako je krajnji cilj razvoja korisničkih sučelja imitirati međuljudski odnos, važno je zadržati se na postojećem stanju. Grafička korisnička sučelja, stvorena kombinacijom elemenata različitih korisničkih sučelja, su najzastupljeniji način razmjene informacija između računala i čovjeka.

Uspješnost razmjene informacija temelji se na kvalitetnoj interakciji. Da bi se ostvarila zadovoljavajuća razina kvalitete grafičkog korisničkog sučelja, provode se aktivnosti osiguranja kvalitete - testiranja. Ovjera kvalitete grafičkih korisničkih sučelja odvija se kombinacijom raznih testnih metodologija i pristupa, pri čemu su najznačajniji manualno i automatsko testiranje. Osim navedenih pristupa, važno je odabrati relevantne tehnike testiranja - *alpha, function, path testing,* ali sve i ostale korištene vrste. Cilj rada je prikaz mogućnosti osiguranja kvalitete grafičkih korisničkih sučelja i izvođenje pripadajućeg zaključka o najefikasnijem pristupu za ostvarenje tog cilja.

Primjena manualnog testiranja za ovjeru valjanosti grafičkog korisničkog sučelja omogućuje duboku analizu od strane provoditelja testiranja. Neovisno o vrsti provoditelja, čovjek je taj koji trenutno može na najbolji način simulirati aktivnosti drugog čovjeka. *GUI* omogućuje prikaz informacija koristeći razne vizualne elemente. Vizualni dojam se vrednuje intuicijom, osjećajima i razinom svijesti kakvu računala zasad ne mogu doseći. Važnost manualnog testiranja, prilikom osiguranja kvalitete grafičkih korisničkih sučelja, proizlazi iz sposobnosti čovjeka da skupu provedenih aktivnosti dodijeli osvrt temeljen na sentimentalnom dojmu.

Raznolikost elemenata grafičkih korisničkih sučelja omogućuje primjenu raznih testnih pristupa prilikom osiguranja kvalitete. Iako se *GUI* temelji na vizualnom dojmu, brojne se komponente unutar njega efikasnije testiraju uz pomoć automatiziranog alata. Računalo ne može vrednovati atraktivnost izgleda određenog elementa, ali moguće je oblikovati naredbe za testiranje funkcionalnosti tih komponenti. Automatizacija testiranja *GUI* komponentni ostvaruje se simulacijom raznovrsnih aktivnosti dodira, miša ili tipkovnice. Testiranje uz pomoć automatskog alata omogućuje brzo repliciranje velikog broja instanci u kratkom vremenskom intervalu.

Osiguranje kvalitete grafičkih korisničkih sučelja moguće je u potpunosti provesti isključivo primjenom obiju testnih metodologija. Dok se manualnim testiranjem verificiraju atraktivnost, dojam i korisničko iskustvo svih *GUI* elemenata, automatskim testiranjem je moguće provjeriti temeljne funkcionalnosti i performanse sučelja na efikasniji način. Zaključak o idealnom testnom pristupu ne proizlazi iz pojedinačnih nedostataka nego iz zajedničkih snaga. Automatsko testiranje nikada neće u potpunosti supstituirati metodologiju manualnog testiranja. Fokus treba biti na mogućnosti efikasne integracije automatskog testiranja u manualni testni proces. Nadopunjujući se međusobnim prednostima, ostvarit će se potpuna pokrivenost testiranja te kvalitetna validacija vizualnih elemenata i programske logike čitavog sustava.

POPIS LITERATURE

1. Amland, S. (1999.) *Risk Based Testing and Metrics: Risk Analysis Fundamentals and Metrics for testing including a Financial Application case study.* 5th International Conference EuroSTAR ‘99. [online]. Studeni: Barcelona, Španjolska. Dostupno na: https://sceweb.sce.uhcl.edu/helm/ROLE-Tester/myfiles/Module10/18_TST170_AppendixF.pdf. [27. kolovoza 2022.]

2. Cooper, A. et al. (2014.) *About Face: The Essentials of Interaction Design.* Četvrto izdanje. Indianapolis: John Wiley & Sons, Inc.

3. Crosby, P. B. (1989.) *Kvaliteta je besplatna: Umijeće osiguravanja kvalitete*, Zagreb: Privredni vjesnik.

4. Hamilton, T. (2022.) *Difference Between Manual and Automation Testing* [online]. Ahmedabad: guru99.com. Dostupno na: https://www.guru99.com/difference-automated-vs-manual-testing.html [20. kolovoza 2022.]

5. Injac, N. (1998.) *Mala enciklopedija kvalitete.* Prvi dio. Zagreb: Oskar

6. ISO/IEC/IEEE 29119-1:2022 (2022.) *Software and systems engineering — Software testing — Part 1: General concepts* [online] Dostupno na: https://www.iso.org/obp/ui/#iso:std:iso-iec-ieee:29119:-1:ed-2:v1:en [20. kolovoza 2022.]

7. ISTQB (2002.) *About Us: What We Do* [online] Dostupno na: https://www.istqb.org/about-us/what-we-do [20. kolovoza 2022.]

8. Johnson, J. (2010.) *Designing with the Mind in Mind: A Simple Guide to Understanding User Interface Design Rules.* Burlington: Elsevier, Inc.

9. Kuchana, K. K. (2016.) *Basic Manual Software testing+Agile+Bugzilla for beginners* [online] Dostupno na: https://www.udemy.com/course/manual-software-testing-course-best -for-beginners/ [26. kolovoza 2022.]

10. Kaner, C., Bach, J. & Pettichord, B. (2002.) *Lessons Learned in Software Testing: A Context-Driven Approach*. New Jersey: John Wiley and Sons, Inc.

11. Krug, S. (2014.) *Don’t make me think: A Common Sense Approach to Web and Mobile Usability.* 3. izd. London: Pearson Education, Inc. (New Riders / Peachpit)

12. Lazibat, T. (2009.) *Upravljanje kvalitetom*. Zagreb, Znanstvena knjiga: M.E.P.

13. Mahfuz, A. S. (2016.) *Software Quality Assurance: Integrating Testing, Security, and Audit.* Boca Raton: CRC Press Taylor & Francis Group

14. Martin, R. C., Grenning, J. & Brown, S. (2018.) *Clean Architecture: A Craftsman's Guide to Software Structure and Design.* London: Pearson Education, Inc. (Prentice Hall)

15. McFadden, C. (2020.) *The Origin of the Term 'Computer Bug'* [online]. New York: Interesting Engineering, Inc. Dostupno na: https://interestingengineering.com/innovation/the-origin-of-the-term-computer-bug [20. kolovoza 2022.]

16. Naik, K. & Tripathy, P. (2008.) *Software testing and quality assurance: Theory and practice*, New Jersey: John Wiley & Sons, Inc.

17. Chandrasekar, P. (2008.) *Stack overflow - PyCharm topics* [online] New York: Stack Exchange, Inc. Dostupno na: https://stackoverflow.com/search?q=pycharm&s=c5e5b38e-5773-4a3e-bc87-4a40b92ca7ef [05. rujna 2022.]

18. *Selenium Course for Beginners - Web Scraping Bots, Browser Automation, Testing (Tutorial) //* YouTube 31.08.2021. Dostupno na: https://www.youtube.com/watch?v=j7VZsCCnptM [05. rujna 2022.] .

19. Sharp, H., Rogers, Y. & Preece, J. (2019.) *Interaction Design: beyond human-computer interaction.* 5. izd. Indianapolis, Indiana: John Wiley and Sons, Inc.

20. Software Testing Standards (2014.) *The International Software Testing Standard: ISO/IEC/IEEE 29119 Software Testing* [online]. Dostupno: http://www.softwaretestingstandard.org/ [22. kolovoza 2022.]

21. Spillner, A., Linz, T. & Shaefer, H. (2014.) *Software testing foundations*. 4. izd. Santa Barbara: Rocky Nook Inc.

22. Spremić, M. (2017.) *Digitalna transformacija poslovanja*. Zagreb: Sveučilište u Zagrebu, Ekonomski fakultet.

23. Spremić, M. (2017.) *Sigurnost i revizija informacijskih sustava u okruženju digitalne ekonomije*. Zagreb: Sveučilište u Zagrebu, Ekonomski fakultet.

24. TestBytes (2017.) *5 Latest Software Testing Standards* [online] Dostupno na: https://www.testbytes.net/blog/software-testing-standards/ [20. kolovoza 2022.]

25. Tidwell, J. (2011.) *Designing Interfaces: Patterns for Interactive Interactions Design.* 2. izd. Sebastopol: O'Rilley Media, Inc.

26. Vlahović, N. & Pivar, J. (2020.) *Sklopovlje računala.* U: Pejić Bach, M. i Spremić, M., ur., *Osnove poslovne informatike.* Zagreb: Sveučilište u Zagrebu, Ekonomski fakultet, str. 1-31.

27. Vlahović, N. (2020.) *Računalni softver.* U: Pejić Bach, M. i Spremić, M., ur., *Osnove poslovne informatike.* Zagreb: Sveučilište u Zagrebu, Ekonomski fakultet, str. 33-65.

28. Whittaker, J. A. & Jorgensen, A. A. (1999.) *Why Software Fails*. ACM Software Engineering Notes. [online]. Dostupno na: https://dl.acm.org/doi/pdf/10.1145 /329155.329168 [20. kolovoza 2022.]

29. Whittaker, J. A. & Jorgensen, A. A. (2000.) *How to Break Software.* International Conference on Software Testing, Analysis & Review Conference. [online] Prosinac: Kopenhagen, Danska. Dostupno na: https://www.researchgate.net/publication /315700027_How_to_Break_ Software_with_examples [20. kolovoza 2022.]

POPIS SLIKA

POPIS TABLICA

ŽIVOTOPIS STUDENTA

Ime i prezime studenta: Ivan Beljan

Datum i mjesto rođenja: 11.11.1996., Samedan (Švicarska)

OBRAZOVANJE

Ekonomski fakultet Sveučilišta u Zagrebu **2015. - 2022.**
Integrirani preddiplomski i diplomski studij Poslovne Ekonomije
- *Smjer: Menadžerska informatika*

RADNO ISKUSTVO

NANOBIT d.o.o.. Zagreb **siječanj 2021. - Danas**
Quality Assurance Associate

Transcom Worldwide, Zagreb **listopad 2020. - siječanj 2021.**
Customer Support

KING ICT Akademija, Zagreb **srpanj 2020.**
Quality Assurance Intern

Ministarstvo znanosti i obrazovanja, Zagreb **srpanj 2018. - listopad 2018.**
Maintenance Assistant

NAGRADE I PRIZNANJA

General Data Protection Regulation for Agents and TLs (Croatian) **2020.**
- *Certificate of completion*

2020-2021 Information Security Awareness (Brick & Mortar Croatian) **2020.**
- *Certificate of completion*

ZNANJA I VJEŠTINE

Digitalne vještine: Xcode, Android Studio, SourceTree, PyCharm, Selenium, Visual Studio, Photoshop, Microsoft Office, Google Tools
Strani jezici: engleski jezik (napredno), njemački jezik (napredno)

PRILOZI

Prilog 1.: Ispis programskog koda prototipa aplikacije OneKey

Prilog 2.: Ekranski prikazi ispravljenih korisničkih sučelja prototipa aplikacije OneKey

Prilog 3.: Ispis programskog koda automatizacije testiranja Google pretraživača

Prilog 4.: Generirani izvještaj naredbenog retka prilikom testiranja tražilice

Prilog 5.: Ispis programskog koda automatizacije testiranja sustava prijave i odjave elektroničke pošte - Gmail

Prilog 6.: Generirani izvještaj naredbenog retka prilikom testiranja sustava prijave

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\login.cs 1

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.ComponentModel;
 4  using System.Data;
 5  using System.Drawing;
 6  using System.Linq;
 7  using System.Text;
 8  using System.Threading.Tasks;
 9  using System.Windows.Forms;
10  using System.Data.OleDb;
11
12  namespace WindowsFormsApp2
13  {
14      public partial class loginFrm : Form
15      {
16          public loginFrm()
17          {
18              InitializeComponent();
19
20          }
21
22          OleDbConnection con = new OleDbConnection
              ("Provider=Microsoft.Jet.OLEDB.4.0;Data
              source=1key_users.mdb");
23          OleDbCommand cmd = new OleDbCommand();
24          OleDbDataAdapter da = new OleDbDataAdapter();
25
26          private void exitBtn_Click(object sender, EventArgs e)
27          {
28              Application.Exit();
29          }
30
31          private void signinBtn_Click(object sender, EventArgs e)
32          {
33              var newform = new signinFrm();
34              newform.Show();
35              this.Hide();
36          }
37
38          private void forgotPassBtn_Click(object sender, EventArgs e)
39          {
40              var newform = new sysAdminFrm();
41              newform.Show();
42          }
43
44          private void passwordBtn_MouseDown(object sender, MouseEventArgs
                e)
45          {
46              passTxtBox.UseSystemPasswordChar = false;
47          }
48
49          private void passwordBtn_MouseUp(object sender, MouseEventArgs
              e)
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\login.cs 2

```
50          {
51              passTxtBox.UseSystemPasswordChar = true;
52          }
53
54          private void loginBtn_Click(object sender, EventArgs e)
55          {
56              con.Open();
57              string login = "SELECT * FROM tbl_users WHERE username= '" + ↵
                   userTxtBox.Text + "' and password= '" + passTxtBox.Text + ↵
                  "'";
58              cmd = new OleDbCommand(login, con);
59              OleDbDataReader dr = cmd.ExecuteReader();
60
61              if (dr.Read() == true)
62              {
63                  var newform = new splashScreenFrm();
64                  newform.Show();
65                  this.Hide();
66              }
67              else
68              {
69                  MessageBox.Show("Invalid username or key, please try     ↵
                      again", "Login error!");
70                  userTxtBox.Text = "";
71                  passTxtBox.Text = "";
72                  userTxtBox.Focus();
73              }
74          }
75      }
76  }
77
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\popupSysAdmin.cs 1

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace WindowsFormsApp2
12 {
13     public partial class sysAdminFrm : Form
14     {
15         public sysAdminFrm()
16         {
17             InitializeComponent();
18         }
19
20         private void sysAdminExitBtn_Click(object sender, EventArgs e)
21         {
22             this.Close();
23             var newform = new loginFrm();
24             newform.Show();
25         }
26     }
27 }
28
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\splashScreen.cs 1

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace WindowsFormsApp2
12 {
13     public partial class splashScreenFrm : Form
14     {
15         public splashScreenFrm()
16         {
17             InitializeComponent();
18             splashScreenLoadingBar.Value = 0;
19         }
20
21         private void label1_Click(object sender, EventArgs e)
22         {
23
24         }
25
26         private void timer1_Tick(object sender, EventArgs e)
27         {
28             splashScreenLoadingBar.Value += 1;
29             splashScreenLoadingBar.Text =                                    ↵
                 splashScreenLoadingBar.Value.ToString() + "%";
30
31             if(splashScreenLoadingBar.Value == 40)
32             {
33                 timer1.Enabled = false;
34                 managerFrm se_form = new managerFrm();
35                 se_form.Show();
36                 this.Hide();
37             }
38         }
39     }
40 }
41
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\signin.cs 1

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.ComponentModel;
 4  using System.Data;
 5  using System.Drawing;
 6  using System.Linq;
 7  using System.Text;
 8  using System.Threading.Tasks;
 9  using System.Windows.Forms;
10  using System.Data.OleDb;
11
12  namespace WindowsFormsApp2
13  {
14      public partial class signinFrm : Form
15      {
16          public signinFrm()
17          {
18              InitializeComponent();
19          }
20
21          OleDbConnection con = new OleDbConnection
              ("Provider=Microsoft.Jet.OLEDB.4.0;Data
              source=1key_users.mdb");
22          OleDbCommand cmd = new OleDbCommand();
23          OleDbDataAdapter da = new OleDbDataAdapter();
24
25          private void SignInExitBtn_Click(object sender, EventArgs e)
26          {
27              this.Close();
28              var newform = new loginFrm();
29              newform.Show();
30          }
31
32          private void signInPasswordBtn_MouseDown(object sender,
              MouseEventArgs e)
33          {
34              signInKeyTextBox.UseSystemPasswordChar = false;
35              signInKey2TextBox.UseSystemPasswordChar = false;
36          }
37
38          private void signInPasswordBtn_MouseUp(object sender,
              MouseEventArgs e)
39          {
40              signInKeyTextBox.UseSystemPasswordChar = true;
41              signInKey2TextBox.UseSystemPasswordChar = true;
42          }
43
44          private void signInPassword2Btn_MouseDown(object sender,
              MouseEventArgs e)
45          {
46              signInKeyTextBox.UseSystemPasswordChar = false;
47              signInKey2TextBox.UseSystemPasswordChar = false;
48          }
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\signin.cs 2

```
49
50          private void signInPassword2Btn_MouseUp(object sender,
              MouseEventArgs e)
51          {
52              signInKeyTextBox.UseSystemPasswordChar = true;
53              signInKey2TextBox.UseSystemPasswordChar = true;
54          }
55
56          private void registerBtn_Click(object sender, EventArgs e)
57          {
58              if (signInUserTextBox.Text == "" && signInKeyTextBox.Text
                  == "" && signInKey2TextBox.Text == "")
59              {
60                  MessageBox.Show("Enter username and key!");
61              }
62              else if (signInKeyTextBox.Text == signInKey2TextBox.Text)
63              {
64                  con.Open();
65                  string register = "INSERT INTO tbl_users VALUES ('" +
                      signInUserTextBox.Text + "', '" +
                      signInKeyTextBox.Text + "')";
66                  cmd = new OleDbCommand(register, con);
67                  cmd.ExecuteNonQuery();
68                  con.Close();
69
70                  signInUserTextBox.Text = "";
71                  signInKeyTextBox.Text = "";
72                  signInKey2TextBox.Text = "";
73
74                  MessageBox.Show("Account has been created!", "Nice");
75
76              }
77              else
78              {
79                  MessageBox.Show("Entered keys are not matching, please
                      try again", "Invalid key");
80                  signInKeyTextBox.Text = "";
81                  signInKey2TextBox.Text = "";
82                  signInKeyTextBox.Focus();
83              }
84
85
86
87          }
88
89          private void signInClearBtn_Click(object sender, EventArgs e)
90          {
91              signInUserTextBox.Text = "";
92              signInKeyTextBox.Text = "";
93              signInKey2TextBox.Text = "";
94              signInUserTextBox.Focus();
95          }
96
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\signin.cs 3

```
 97          private void signInBackToLoginBtn_Click(object sender,
               EventArgs e)
 98          {
 99              this.Close();
100              var newform = new loginFrm();
101              newform.Show();
102          }
103      }
104  }
105
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainManager.cs 1

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.ComponentModel;
 4  using System.Data;
 5  using System.Drawing;
 6  using System.Linq;
 7  using System.Text;
 8  using System.Threading.Tasks;
 9  using System.Windows.Forms;
10  using System.Data.OleDb;
11
12  namespace WindowsFormsApp2
13  {
14      public partial class managerFrm : Form
15      {
16          public managerFrm()
17          {
18              InitializeComponent();
19
20
21          }
22
23          OleDbConnection con = new OleDbConnection                    ↵
              ("Provider=Microsoft.Jet.OLEDB.4.0;Data                    ↵
              source=1key_profiles.mdb");
24          OleDbCommand cmnd = new OleDbCommand();
25          OleDbDataAdapter da = new OleDbDataAdapter();
26
27
28          private void managerBtn_Click(object sender, EventArgs e)
29          {
30              this.Close();
31              var newform = new managerFrm();
32              newform.Show();
33          }
34
35          private void mainLogOutBtn_Click(object sender, EventArgs e)
36          {
37              this.Close();
38              var newform = new loginFrm();
39              newform.Show();
40          }
41
42          private void generatorBtn_Click_1(object sender, EventArgs e)
43          {
44              this.Close();
45              var newform = new generatorFrm();
46              newform.Show();
47          }
48
49          private void smartCheckBtn_Click(object sender, EventArgs e)
50          {
51              this.Close();
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainManager.cs 2

```
52              var newform = new smartCheckFrm();
53              newform.Show();
54          }
55
56          private void eduBtn_Click(object sender, EventArgs e)
57          {
58              this.Close();
59              var newform = new eduFrm();
60              newform.Show();
61          }
62
63          private void aboutBtn_Click(object sender, EventArgs e)
64          {
65              this.Close();
66              var newform = new aboutFrm();
67              newform.Show();
68          }
69
70          private void miniGameBtn_Click(object sender, EventArgs e)
71          {
72              this.Close();
73              var newform = new miniGameFrm();
74              newform.Show();
75          }
76
77          private void managerSetPic1Btn_click(object sender, EventArgs
              e)
78          {
79
80              String imageLocation = "";
81              try
82              {
83                  OpenFileDialog dialog = new OpenFileDialog();
84                  dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files
                      (*.png)|*.png| All files (*.*)|*.*";
85
86                  if (dialog.ShowDialog() ==
                      System.Windows.Forms.DialogResult.OK)
87                  {
88                      imageLocation = dialog.FileName;
89                      picManagerLocation1.ImageLocation = imageLocation;
90                  }
91              }
92              catch (Exception)
93              {
94                  MessageBox.Show("An Error Ocurred");
95              }
96          }
97
98          private void managerSetPic2Btn_click(object sender, EventArgs
              e)
99          {
100
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainManager.cs 3

```
101             String imageLocation = "";
102             try
103             {
104                 OpenFileDialog dialog = new OpenFileDialog();
105                 dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files
                      (*.png)|*.png| All files (*.*)|*.*";
106
107                 if (dialog.ShowDialog() ==
                      System.Windows.Forms.DialogResult.OK)
108                 {
109                     imageLocation = dialog.FileName;
110                     picManagerLocation2.ImageLocation = imageLocation;
111                 }
112             }
113             catch (Exception)
114             {
115                 MessageBox.Show("An Error Ocurred");
116             }
117         }
118
119         private void managerSetPic3Btn_click(object sender, EventArgs
              e)
120         {
121             String imageLocation = "";
122             try
123             {
124                 OpenFileDialog dialog = new OpenFileDialog();
125                 dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files
                      (*.png)|*.png| All files (*.*)|*.*";
126
127                 if (dialog.ShowDialog() ==
                      System.Windows.Forms.DialogResult.OK)
128                 {
129                     imageLocation = dialog.FileName;
130                     picManagerLocation3.ImageLocation = imageLocation;
131                 }
132             }
133             catch (Exception)
134             {
135                 MessageBox.Show("An Error Ocurred");
136             }
137         }
138
139         private void managerSetPic4Btn_click(object sender, EventArgs
              e)
140         {
141
142             String imageLocation = "";
143             try
144             {
145                 OpenFileDialog dialog = new OpenFileDialog();
146                 dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files
                      (*.png)|*.png| All files (*.*)|*.*";
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainManager.cs 4

```
147
148                 if (dialog.ShowDialog() ==
                      System.Windows.Forms.DialogResult.OK)
149                 {
150                     imageLocation = dialog.FileName;
151                     picManagerLocation4.ImageLocation = imageLocation;
152                 }
153             }
154             catch (Exception)
155             {
156                 MessageBox.Show("An Error Ocurred");
157             }
158         }
159
160         private void managerSetPic5Btn_click(object sender, EventArgs
              e)
161         {
162
163             String imageLocation = "";
164             try
165             {
166                 OpenFileDialog dialog = new OpenFileDialog();
167                 dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files
                    (*.png)|*.png| All files (*.*)|*.*";
168
169                 if (dialog.ShowDialog() ==
                      System.Windows.Forms.DialogResult.OK)
170                 {
171                     imageLocation = dialog.FileName;
172                     picManagerLocation5.ImageLocation = imageLocation;
173                 }
174             }
175             catch (Exception)
176             {
177                 MessageBox.Show("An Error Ocurred");
178             }
179         }
180
181         private void managerSetPic6Btn_click(object sender, EventArgs
              e)
182         {
183
184             String imageLocation = "";
185             try
186             {
187                 OpenFileDialog dialog = new OpenFileDialog();
188                 dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files
                    (*.png)|*.png| All files (*.*)|*.*";
189
190                 if (dialog.ShowDialog() ==
                      System.Windows.Forms.DialogResult.OK)
191                 {
192                     imageLocation = dialog.FileName;
```

```
193                     picManagerLocation6.ImageLocation = imageLocation;
194                 }
195             }
196             catch (Exception)
197             {
198                 MessageBox.Show("An Error Ocurred");
199             }
200         }
201 
202         private void managerSetPic7Btn_click(object sender, EventArgs
              e)
203         {
204 
205             String imageLocation = "";
206             try
207             {
208                 OpenFileDialog dialog = new OpenFileDialog();
209                 dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files
                    (*.png)|*.png| All files (*.*)|*.*";
210 
211                 if (dialog.ShowDialog() ==
                    System.Windows.Forms.DialogResult.OK)
212                 {
213                     imageLocation = dialog.FileName;
214                     picManagerLocation7.ImageLocation = imageLocation;
215                 }
216             }
217             catch (Exception)
218             {
219                 MessageBox.Show("An Error Ocurred");
220             }
221         }
222 
223         private void managerSetPic8Btn_click(object sender, EventArgs
              e)
224         {
225 
226             String imageLocation = "";
227             try
228             {
229                 OpenFileDialog dialog = new OpenFileDialog();
230                 dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files
                    (*.png)|*.png| All files (*.*)|*.*";
231 
232                 if (dialog.ShowDialog() ==
                    System.Windows.Forms.DialogResult.OK)
233                 {
234                     imageLocation = dialog.FileName;
235                     picManagerLocation8.ImageLocation = imageLocation;
236                 }
237             }
238             catch (Exception)
239             {
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainManager.cs 6

```
240                 MessageBox.Show("An Error Ocurred");
241             }
242         }
243 
244         private void managerSetPic9Btn_click(object sender, EventArgs e)
245         {
246 
247             String imageLocation = "";
248             try
249             {
250                 OpenFileDialog dialog = new OpenFileDialog();
251                 dialog.Filter = "jpg files (*.jpg)|*.jpg| PNG files (*.png)|*.png| All files (*.*)|*.*";
252 
253                 if (dialog.ShowDialog() == System.Windows.Forms.DialogResult.OK)
254                 {
255                     imageLocation = dialog.FileName;
256                     picManagerLocation9.ImageLocation = imageLocation;
257                 }
258             }
259             catch (Exception)
260             {
261                 MessageBox.Show("An Error Ocurred");
262             }
263         }
264 
265 
266 
267         private void managerViewKey1Btn_Click(object sender, EventArgs e)
268         {
269             if (managerKey1TextBox.UseSystemPasswordChar == true)
270             {
271                 managerKey1TextBox.UseSystemPasswordChar = false;
272             }
273             else
274             {
275                 managerKey1TextBox.UseSystemPasswordChar = true;
276             }
277 
278         }
279 
280         private void managerViewKey2Btn_Click(object sender, EventArgs e)
281         {
282             if (managerKey2TextBox.UseSystemPasswordChar == true)
283             {
284                 managerKey2TextBox.UseSystemPasswordChar = false;
285             }
286             else
287             {
```

```
288                 managerKey2TextBox.UseSystemPasswordChar = true;
289             }
290 
291         }
292 
293         private void managerViewKey3Btn_Click(object sender, EventArgs ↪
              e)
294         {
295             if (managerKey3TextBox.UseSystemPasswordChar == true)
296             {
297                 managerKey3TextBox.UseSystemPasswordChar = false;
298             }
299             else
300             {
301                 managerKey3TextBox.UseSystemPasswordChar = true;
302             }
303 
304         }
305 
306         private void managerViewKey4Btn_Click(object sender, EventArgs ↪
              e)
307         {
308             if (managerKey4TextBox.UseSystemPasswordChar == true)
309             {
310                 managerKey4TextBox.UseSystemPasswordChar = false;
311             }
312             else
313             {
314                 managerKey4TextBox.UseSystemPasswordChar = true;
315             }
316 
317         }
318 
319         private void managerViewKey5Btn_Click(object sender, EventArgs ↪
              e)
320         {
321             if (managerKey5TextBox.UseSystemPasswordChar == true)
322             {
323                 managerKey5TextBox.UseSystemPasswordChar = false;
324             }
325             else
326             {
327                 managerKey5TextBox.UseSystemPasswordChar = true;
328             }
329 
330         }
331 
332         private void managerViewKey6Btn_Click(object sender, EventArgs ↪
              e)
333         {
334             if (managerKey6TextBox.UseSystemPasswordChar == true)
335             {
336                 managerKey6TextBox.UseSystemPasswordChar = false;
```

```
337             }
338             else
339             {
340                 managerKey6TextBox.UseSystemPasswordChar = true;
341             }
342 
343         }
344 
345         private void managerViewKey7Btn_Click(object sender, EventArgs ➲
              e)
346         {
347             if (managerKey7TextBox.UseSystemPasswordChar == true)
348             {
349                 managerKey7TextBox.UseSystemPasswordChar = false;
350             }
351             else
352             {
353                 managerKey7TextBox.UseSystemPasswordChar = true;
354             }
355 
356         }
357 
358         private void managerViewKey8Btn_Click(object sender, EventArgs ➲
              e)
359         {
360             if (managerKey8TextBox.UseSystemPasswordChar == true)
361             {
362                 managerKey8TextBox.UseSystemPasswordChar = false;
363             }
364             else
365             {
366                 managerKey8TextBox.UseSystemPasswordChar = true;
367             }
368 
369         }
370 
371         private void managerViewKey9Btn_Click(object sender, EventArgs ➲
              e)
372         {
373             if (managerKey9TextBox.UseSystemPasswordChar == true)
374             {
375                 managerKey9TextBox.UseSystemPasswordChar = false;
376             }
377             else
378             {
379                 managerKey9TextBox.UseSystemPasswordChar = true;
380             }
381 
382         }
383 
384 
385         private void managerSaveKey1Btn_Click(object sender, EventArgs ➲
              e)
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainManager.cs 9

```
386         {
387             con.Open();
388             string register = "INSERT INTO tbl_profiles VALUES ('" +
                  managerLocation1TextBox.Text + "', '" +
                  managerUser1TextBox.Text + "', '" +
                  managerKey1TextBox.Text + "')";
389             cmnd = new OleDbCommand(register, con);
390             cmnd.ExecuteNonQuery();
391             con.Close();
392
393
394             MessageBox.Show("Account data saved", "Nice");
395         }
396
397         private void managerSaveKey2Btn_Click(object sender, EventArgs
              e)
398         {
399             con.Open();
400             string register = "INSERT INTO tbl_profiles VALUES ('" +
                  managerLocation2TextBox.Text + "', '" +
                  managerUser2TextBox.Text + "', '" +
                  managerKey2TextBox.Text + "')";
401             cmnd = new OleDbCommand(register, con);
402             cmnd.ExecuteNonQuery();
403             con.Close();
404
405
406             MessageBox.Show("Account data saved", "Nice");
407         }
408
409         private void managerSaveKey3Btn_Click(object sender, EventArgs
              e)
410         {
411             con.Open();
412             string register = "INSERT INTO tbl_profiles VALUES ('" +
                  managerLocation3TextBox.Text + "', '" +
                  managerUser3TextBox.Text + "', '" +
                  managerKey3TextBox.Text + "')";
413             cmnd = new OleDbCommand(register, con);
414             cmnd.ExecuteNonQuery();
415             con.Close();
416
417
418             MessageBox.Show("Account data saved", "Nice");
419         }
420
421         private void managerSaveKey4Btn_Click(object sender, EventArgs
              e)
422         {
423             con.Open();
424             string register = "INSERT INTO tbl_profiles VALUES ('" +
                  managerLocation4TextBox.Text + "', '" +
                  managerUser4TextBox.Text + "', '" +
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainManager.cs 10

```
                  managerKey4TextBox.Text + "')";
425               cmnd = new OleDbCommand(register, con);
426               cmnd.ExecuteNonQuery();
427               con.Close();
428
429
430               MessageBox.Show("Account data saved", "Nice");
431           }
432
433           private void managerSaveKey5Btn_Click(object sender, EventArgs ➲
                e)
434           {
435               con.Open();
436               string register = "INSERT INTO tbl_profiles VALUES ('" +    ➲
                  managerLocation5TextBox.Text + "', '" +                     ➲
                  managerUser5TextBox.Text + "', '" +                         ➲
                  managerKey5TextBox.Text + "')";
437               cmnd = new OleDbCommand(register, con);
438               cmnd.ExecuteNonQuery();
439               con.Close();
440
441
442               MessageBox.Show("Account data saved", "Nice");
443           }
444
445           private void managerSaveKey6Btn_Click(object sender, EventArgs ➲
                e)
446           {
447               con.Open();
448               string register = "INSERT INTO tbl_profiles VALUES ('" +    ➲
                  managerLocation6TextBox.Text + "', '" +                     ➲
                  managerUser6TextBox.Text + "', '" +                         ➲
                  managerKey6TextBox.Text + "')";
449               cmnd = new OleDbCommand(register, con);
450               cmnd.ExecuteNonQuery();
451               con.Close();
452
453
454               MessageBox.Show("Account data saved", "Nice");
455           }
456
457           private void managerSaveKey7Btn_Click(object sender, EventArgs ➲
                e)
458           {
459               con.Open();
460               string register = "INSERT INTO tbl_profiles VALUES ('" +    ➲
                  managerLocation7TextBox.Text + "', '" +                     ➲
                  managerUser7TextBox.Text + "', '" +                         ➲
                  managerKey7TextBox.Text + "')";
461               cmnd = new OleDbCommand(register, con);
462               cmnd.ExecuteNonQuery();
463               con.Close();
464
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainManager.cs 11

```
465
466             MessageBox.Show("Account data saved", "Nice");
467         }
468
469         private void managerSaveKey8Btn_Click(object sender, EventArgs
              e)
470         {
471             con.Open();
472             string register = "INSERT INTO tbl_profiles VALUES ('" +
                  managerLocation8TextBox.Text + "', '" +
                  managerUser8TextBox.Text + "', '" +
                  managerKey8TextBox.Text + "')";
473             cmnd = new OleDbCommand(register, con);
474             cmnd.ExecuteNonQuery();
475             con.Close();
476
477
478             MessageBox.Show("Account data saved", "Nice");
479         }
480
481         private void managerSaveKey9Btn_Click(object sender, EventArgs
              e)
482         {
483             con.Open();
484             string register = "INSERT INTO tbl_profiles VALUES ('" +
                  managerLocation9TextBox.Text + "', '" +
                  managerUser9TextBox.Text + "', '" +
                  managerKey9TextBox.Text + "')";
485             cmnd = new OleDbCommand(register, con);
486             cmnd.ExecuteNonQuery();
487             con.Close();
488
489
490             MessageBox.Show("Account data saved", "Nice");
491         }
492
493         private void managerCopyKey1Btn_Click(object sender, EventArgs
              e)
494         {
495             Clipboard.SetText(managerKey1TextBox.Text);
496         }
497
498         private void managerCopyKey2Btn_Click(object sender, EventArgs
              e)
499         {
500             Clipboard.SetText(managerKey2TextBox.Text);
501         }
502
503         private void managerCopyKey3Btn_Click(object sender, EventArgs
              e)
504         {
505             Clipboard.SetText(managerKey3TextBox.Text);
506         }
```

```
507
508         private void managerCopyKey4Btn_Click(object sender, EventArgs ➥
              e)
509         {
510             Clipboard.SetText(managerKey4TextBox.Text);
511         }
512
513         private void managerCopyKey5Btn_Click(object sender, EventArgs ➥
              e)
514         {
515             Clipboard.SetText(managerKey5TextBox.Text);
516         }
517
518         private void managerCopyKey6Btn_Click(object sender, EventArgs ➥
              e)
519         {
520             Clipboard.SetText(managerKey6TextBox.Text);
521         }
522
523         private void managerCopyKey7Btn_Click(object sender, EventArgs ➥
              e)
524         {
525             Clipboard.SetText(managerKey7TextBox.Text);
526         }
527
528         private void managerCopyKey8Btn_Click(object sender, EventArgs ➥
              e)
529         {
530             Clipboard.SetText(managerKey8TextBox.Text);
531         }
532
533         private void managerCopyKey9Btn_Click(object sender, EventArgs ➥
              e)
534         {
535             Clipboard.SetText(managerKey9TextBox.Text);
536         }
537     }
538 }
539
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainGenerator.cs 1

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.ComponentModel;
 4  using System.Data;
 5  using System.Drawing;
 6  using System.Linq;
 7  using System.Text;
 8  using System.Threading.Tasks;
 9  using System.Windows.Forms;
10
11  namespace WindowsFormsApp2
12  {
13      public partial class generatorFrm : Form
14      {
15          int currentPasswordLenght = 0;
16          Random character = new Random();
17
18          public generatorFrm()
19          {
20              InitializeComponent();
21              generatorTrackBar.Minimum = 12;
22              generatorTrackBar.Maximum = 24;
23              passwordGenerator(12);
24          }
25
26          private void mainLogOutBtn_Click(object sender, EventArgs e)
27          {
28              this.Close();
29              var newform = new loginFrm();
30              newform.Show();
31          }
32
33          private void managerBtn_Click(object sender, EventArgs e)
34          {
35              this.Close();
36              var newform = new managerFrm();
37              newform.Show();
38          }
39
40          private void generatorBtn_Click(object sender, EventArgs e)
41          {
42              this.Close();
43              var newform = new generatorFrm();
44              newform.Show();
45          }
46
47          private void smartCheckBtn_Click(object sender, EventArgs e)
48          {
49              this.Close();
50              var newform = new smartCheckFrm();
51              newform.Show();
52          }
53
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainGenerator.cs 2

```
54          private void eduBtn_Click(object sender, EventArgs e)
55          {
56              this.Close();
57              var newform = new eduFrm();
58              newform.Show();
59          }
60
61          private void aboutBtn_Click(object sender, EventArgs e)
62          {
63              this.Close();
64              var newform = new aboutFrm();
65              newform.Show();
66          }
67
68          private void generatorCopyBtn_Click(object sender, EventArgs e)
69          {
70              Clipboard.SetText(generatorTextBox.Text);
71          }
72
73          private void passwordGenerator(int passwordLength)
74          {
75              String allCharacters =                                    ↵
                "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstu↵
                vwxyz_-><!$%&/=)(?*+ #";
76              String randomPassword = "";
77
78              for (int i = 0; i < passwordLength; i++)
79              {
80                  int randomNum = character.Next(0,                     ↵
                      allCharacters.Length);
81                  randomPassword += allCharacters[randomNum];
82
83
84              }
85              generatorTextBox.Text = randomPassword;
86          }
87
88          private void generatorTrackBar_Scroll(object sender, EventArgs  ↵
              e)
89          {
90              generatorPasswordLenghtLbl.Text = "Key length: " +        ↵
                  generatorTrackBar.Value.ToString();
91              currentPasswordLenght = generatorTrackBar.Value;
92              passwordGenerator(currentPasswordLenght);
93          }
94
95          private void generatorMiniGameBtn_Click(object sender,        ↵
              EventArgs e)
96          {
97              this.Close();
98              var newform = new miniGameFrm();
99              newform.Show();
100         }
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainGenerator.cs 3

```
101         }
102     }
103
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainSmartCheck.cs 1

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.ComponentModel;
 4  using System.Data;
 5  using System.Drawing;
 6  using System.Linq;
 7  using System.Text;
 8  using System.Threading.Tasks;
 9  using System.Windows.Forms;
10
11  namespace WindowsFormsApp2
12  {
13
14
15      public partial class smartCheckFrm : Form
16      {
17
18          int charCount = 0;
19          int charMissing = 0;
20
21          public smartCheckFrm()
22          {
23              InitializeComponent();
24          }
25
26          private void eduBtn_Click(object sender, EventArgs e)
27          {
28              this.Close();
29              var newform = new eduFrm();
30              newform.Show();
31          }
32
33          private void aboutBtn_Click(object sender, EventArgs e)
34          {
35              this.Close();
36              var newform = new aboutFrm();
37              newform.Show();
38          }
39
40          private void mainLogOutBtn_Click(object sender, EventArgs e)
41          {
42              this.Close();
43              var newform = new loginFrm();
44              newform.Show();
45          }
46
47          private void generatorBtn_Click(object sender, EventArgs e)
48          {
49              this.Close();
50              var newform = new generatorFrm();
51              newform.Show();
52          }
53
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainSmartCheck.cs 2

```
54          private void smartCheckBtn_Click(object sender, EventArgs e)
55          {
56              this.Close();
57              var newform = new smartCheckFrm();
58              newform.Show();
59          }
60
61          private void managerBtn_Click(object sender, EventArgs e)
62          {
63              this.Close();
64              var newform = new managerFrm();
65              newform.Show();
66          }
67
68          private void checkMiniGameBtn_Click(object sender, EventArgs e)
69          {
70              this.Close();
71              var newform = new miniGameFrm();
72              newform.Show();
73          }
74
75          private void smartCheckCopyBtn_Click(object sender, EventArgs
              e)
76          {
77                  Clipboard.SetText(smartCheckTextBox.Text);
78          }
79
80          private void smartCheckPasswordBtn_Click(object sender,
              EventArgs e)
81          {
82              charCount = smartCheckTextBox.Text.Length;
83              charMissing = 12 - charCount;
84              String str = smartCheckTextBox.Text;
85              String result = String.Empty;
86
87
88              if (charCount == 0)
89              {
90                  MessageBox.Show("Enter a key!");
91              }
92              else
93              {
94                  if (charCount >= 12)
95                  {
96                      smartCheckSLbl.Text = "S.trong (at least 12
                      characters) : TRUE, " + charCount.ToString() + "
                      characters";
97                  }
98                  else
99                  {
100                     smartCheckSLbl.Text = "S.trong (at least 12
                      characters) : FALSE, key can be improved! Add at
                      least " + charMissing.ToString() + " characters";
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainSmartCheck.cs 3

```
101                 }
102
103                 if (
104                    (smartCheckTextBox.Text.Contains("A") ||
                    smartCheckTextBox.Text.Contains("B") ||
                    smartCheckTextBox.Text.Contains("C") ||
                    smartCheckTextBox.Text.Contains("D") ||
105                   smartCheckTextBox.Text.Contains("E") ||
                    smartCheckTextBox.Text.Contains("F") ||
                    smartCheckTextBox.Text.Contains("G") ||
                    smartCheckTextBox.Text.Contains("H") ||
106                   smartCheckTextBox.Text.Contains("I") ||
                    smartCheckTextBox.Text.Contains("J") ||
                    smartCheckTextBox.Text.Contains("K") ||
                    smartCheckTextBox.Text.Contains("L") ||
107                   smartCheckTextBox.Text.Contains("M") ||
                    smartCheckTextBox.Text.Contains("N") ||
                    smartCheckTextBox.Text.Contains("O") ||
                    smartCheckTextBox.Text.Contains("P") ||
108                   smartCheckTextBox.Text.Contains("Q") ||
                    smartCheckTextBox.Text.Contains("R") ||
                    smartCheckTextBox.Text.Contains("S") ||
                    smartCheckTextBox.Text.Contains("T") ||
109                   smartCheckTextBox.Text.Contains("U") ||
                    smartCheckTextBox.Text.Contains("V") ||
                    smartCheckTextBox.Text.Contains("W") ||
                    smartCheckTextBox.Text.Contains("X") ||
110                   smartCheckTextBox.Text.Contains("Y") ||
                    smartCheckTextBox.Text.Contains("Z")
111                   )
112                   &&
113                   (smartCheckTextBox.Text.Contains("a") ||
                    smartCheckTextBox.Text.Contains("b") ||
                    smartCheckTextBox.Text.Contains("c") ||
114                   smartCheckTextBox.Text.Contains("d") ||
                    smartCheckTextBox.Text.Contains("e") ||
                    smartCheckTextBox.Text.Contains("f") ||
                    smartCheckTextBox.Text.Contains("g") ||
115                   smartCheckTextBox.Text.Contains("h") ||
                    smartCheckTextBox.Text.Contains("i") ||
                    smartCheckTextBox.Text.Contains("j") ||
                    smartCheckTextBox.Text.Contains("k") ||
116                   smartCheckTextBox.Text.Contains("l") ||
                    smartCheckTextBox.Text.Contains("m") ||
                    smartCheckTextBox.Text.Contains("n") ||
                    smartCheckTextBox.Text.Contains("o") ||
117                   smartCheckTextBox.Text.Contains("p") ||
                    smartCheckTextBox.Text.Contains("q") ||
                    smartCheckTextBox.Text.Contains("r") ||
                    smartCheckTextBox.Text.Contains("s") ||
118                   smartCheckTextBox.Text.Contains("t") ||
                    smartCheckTextBox.Text.Contains("u") ||
                    smartCheckTextBox.Text.Contains("v") ||
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainSmartCheck.cs 4

```
                    smartCheckTextBox.Text.Contains("w") ||
119                   smartCheckTextBox.Text.Contains("x") ||
                    smartCheckTextBox.Text.Contains("y") ||
                    smartCheckTextBox.Text.Contains("z"))
120                   &&
121                   (smartCheckTextBox.Text.Contains("0") ||
                    smartCheckTextBox.Text.Contains("1") ||
122                   smartCheckTextBox.Text.Contains("2") ||
                    smartCheckTextBox.Text.Contains("3") ||
                    smartCheckTextBox.Text.Contains("4") ||
                    smartCheckTextBox.Text.Contains("5") ||
123                   smartCheckTextBox.Text.Contains("6") ||
                    smartCheckTextBox.Text.Contains("7") ||
                    smartCheckTextBox.Text.Contains("8") ||
                    smartCheckTextBox.Text.Contains("9") ||
124                   smartCheckTextBox.Text.Contains("0")
125                   )
126                   &&
127                   (smartCheckTextBox.Text.Contains("_") ||
                    smartCheckTextBox.Text.Contains(" ") ||
                    smartCheckTextBox.Text.Contains("#") ||
                    smartCheckTextBox.Text.Contains("(") ||
128                   smartCheckTextBox.Text.Contains("-") ||
                    smartCheckTextBox.Text.Contains(">") ||
                    smartCheckTextBox.Text.Contains("<") ||
                    smartCheckTextBox.Text.Contains("$") ||
129                   smartCheckTextBox.Text.Contains("!") ||
                    smartCheckTextBox.Text.Contains("%") ||
                    smartCheckTextBox.Text.Contains("&") ||
                    smartCheckTextBox.Text.Contains("/") ||
130                   smartCheckTextBox.Text.Contains("=") ||
                    smartCheckTextBox.Text.Contains("?") ||
                    smartCheckTextBox.Text.Contains("*") ||
                    smartCheckTextBox.Text.Contains("+") ||
131                   smartCheckTextBox.Text.Contains(")"))
132                   )
133                 {
134                     smartCheckMLbl.Text = "M.ulti-character (contains
                    various characters and symbols) : TRUE";
135                 }
136                 else
137                 {
138                     smartCheckMLbl.Text = "M.ulti-character (contains
                    various characters and symbols) : FALSE! Lower/
                    uppercase character, number or symbol missing";
139                 }
140
141                 smartCheckALbl.Text = "A.void associations : Check
                    whether your key contains words, names, dates or
                    similar. If so, adjust your key!";
142
143                 for (int i = 0; i < str.Length; i++)
144                 {
```

```
145                     if (!result.Contains(str[i]))
146                         result += str[i];
147                     smartCheckRLbl.Text = "R.andom (key with unique
                    characters) : " + result;
148
149                 }
150                 smartCheckTLbl.Text = "Tools = OneKey : TRUE :)";
151             }
152         }
153     }
154 }
155
156
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainEdu.cs 1

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace WindowsFormsApp2
12 {
13     public partial class eduFrm : Form
14     {
15         public eduFrm()
16         {
17             InitializeComponent();
18         }
19
20         private void mainLogOutBtn_Click(object sender, EventArgs e)
21         {
22             this.Close();
23             var newform = new loginFrm();
24             newform.Show();
25         }
26
27         private void managerBtn_Click(object sender, EventArgs e)
28         {
29             this.Close();
30             var newform = new managerFrm();
31             newform.Show();
32         }
33
34         private void generatorBtn_Click(object sender, EventArgs e)
35         {
36             this.Close();
37             var newform = new generatorFrm();
38             newform.Show();
39         }
40
41         private void smartCheckBtn_Click(object sender, EventArgs e)
42         {
43             this.Close();
44             var newform = new smartCheckFrm();
45             newform.Show();
46         }
47
48         private void eduBtn_Click(object sender, EventArgs e)
49         {
50             this.Close();
51             var newform = new eduFrm();
52             newform.Show();
53         }
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainEdu.cs 2

```
54
55          private void aboutBtn_Click(object sender, EventArgs e)
56          {
57              this.Close();
58              var newform = new aboutFrm();
59              newform.Show();
60          }
61
62          private void eduTakeQuizBtn_Click(object sender, EventArgs e)
63          {
64              this.Close();
65              var newform = new eduQuizFrm();
66              newform.Show();
67          }
68
69          private void eduMiniGameBtn_Click(object sender, EventArgs e)
70          {
71              this.Close();
72              var newform = new miniGameFrm();
73              newform.Show();
74          }
75      }
76  }
77
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainEduQuiz.cs 1

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.ComponentModel;
 4  using System.Data;
 5  using System.Drawing;
 6  using System.Linq;
 7  using System.Text;
 8  using System.Threading.Tasks;
 9  using System.Windows.Forms;
10
11  namespace WindowsFormsApp2
12  {
13      public partial class eduQuizFrm : Form
14      {
15          //quiz variables
16          int correctAnswer;
17          int questionNumber = 1;
18          int score;
19          int percentage;
20          int totalQuestions;
21
22          public eduQuizFrm()
23          {
24              InitializeComponent();
25              askQuestion(questionNumber);
26              totalQuestions = 5;
27          }
28
29          private void eduQuizExitBtn_Click(object sender, EventArgs e)
30          {
31              this.Close();
32              var newform = new eduFrm();
33              newform.Show();
34          }
35
36          private void checkAnswerEvent(object sender, EventArgs e)
37          {
38              var senderObject = (Button)sender;
39              int buttonTag = Convert.ToInt32(senderObject.Tag);
40
41              if (buttonTag == correctAnswer)
42              {
43                  score++;
44              }
45
46              if (questionNumber == totalQuestions)
47              {
48                  percentage = (int)Math.Round((double)(score * 100) /
                      totalQuestions);
49
50                  MessageBox.Show(
51                      "Quiz ended!" + Environment.NewLine +
52                      "You have answered " + score + " questions
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainEduQuiz.cs 2

```
                    correctly." + Environment.NewLine +
53                    "Your total percentage: " + percentage + "%" +
                    Environment.NewLine +
54                    "Thank you for participating in our DEMO course -
                    the certified course will be available soon!"
55                     );
56
57                  score = 0;
58                  questionNumber = 0;
59                  askQuestion(questionNumber);
60
61              }
62              questionNumber++;
63              askQuestion(questionNumber);
64          }
65
66          private void askQuestion(int qnum)
67          {
68              switch (qnum)
69              {
70                  case 1:
71                      quizQuestionLbl1.Text = "What is cybersecurity?";
72
73                      button1.Text = "Protection of internet-connected
                    systems such as hardware, software and data from
                    cyberthreats";
74                      button2.Text = "Protection against cyber bullying";
75                      button3.Text = "Bodyguard vendor";
76                      button4.Text = "Unnecessary business expense";
77
78                      correctAnswer = 1;
79                      break;
80
81                  case 2:
82                      quizQuestionLbl1.Text = "Spear phishing is...";
83
84                      button1.Text = "a form of malicious software in
                    which any file or program can be used to harm a
                    computer user";
85                      button2.Text = "an attack that relies on human
                    interaction to trick users into breaking security
                    procedures to gain sensitive information that is
                    typically protected.";
86                      button3.Text = "a type of phishing attack that has
                    an intended target user, organization or business.";
87                      button4.Text = "a form of social engineering where
                    fraudulent email or text messages that resemble those
                     from reputable or known sources are sent.";
88
89                      correctAnswer = 3;
90                      break;
91
92                  case 3:
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainEduQuiz.cs 3

```
 93                      quizQuestionLbl1.Text = "The most efficient way to
                     deliver malware is/are";
 94
 95                      button1.Text = "USB stick";
 96                      button2.Text = "E-mails";
 97                      button3.Text = "UPS delivery";
 98                      button4.Text = "Social media";
 99
100                      correctAnswer = 2;
101                      break;
102
103                  case 4:
104                      quizQuestionLbl1.Text = "What is office hygiene in
                     cybersecurity?";
105
106                      button1.Text = "Using disinfectants in the office";
107                      button2.Text = "Cleaning the office after work";
108                      button3.Text = "Wearing a medical mask while
                     communicating with colleagues";
109                      button4.Text = "Method to understand the best way
                     to protect paper, desks, screens and buildings";
110
111                      correctAnswer = 4;
112                      break;
113
114                  case 5:
115                      quizQuestionLbl1.Text = "Cybersecurity is truly
                     effective when:";
116
117                      button1.Text = "it is used during the work shift";
118                      button2.Text = "employees willingly embrace and
                     proactively use cyber-secure practices both
                     professionally and personally";
119                      button3.Text = "employers only, willingly embrace
                     and proactively use cyber-secure practices both
                     professionally and personally ";
120                      button4.Text = "you install a cybersecurity tool";
121
122                      correctAnswer = 2;
123                      break;
124              }
125          }
126      }
127  }
128
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainAbout.cs 1

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace WindowsFormsApp2
12 {
13     public partial class aboutFrm : Form
14     {
15         public aboutFrm()
16         {
17             InitializeComponent();
18         }
19
20         private void mainLogOutBtn_Click(object sender, EventArgs e)
21         {
22             this.Close();
23             var newform = new loginFrm();
24             newform.Show();
25         }
26
27         private void managerBtn_Click(object sender, EventArgs e)
28         {
29             this.Close();
30             var newform = new managerFrm();
31             newform.Show();
32         }
33
34         private void generatorBtn_Click_1(object sender, EventArgs e)
35         {
36             this.Close();
37             var newform = new generatorFrm();
38             newform.Show();
39         }
40
41         private void smartCheckBtn_Click_1(object sender, EventArgs e)
42         {
43             this.Close();
44             var newform = new smartCheckFrm();
45             newform.Show();
46         }
47
48         private void eduBtn_Click(object sender, EventArgs e)
49         {
50             this.Close();
51             var newform = new eduFrm();
52             newform.Show();
53         }
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\mainAbout.cs 2

```
54
55          private void aboutBtn_Click_1(object sender, EventArgs e)
56          {
57              this.Close();
58              var newform = new aboutFrm();
59              newform.Show();
60          }
61
62          private void aboutMiniGameBtn_Click(object sender, EventArgs e)
63          {
64              this.Close();
65              var newform = new miniGameFrm();
66              newform.Show();
67          }
68      }
69  }
70
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\miniGame.cs 1

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.ComponentModel;
 4  using System.Data;
 5  using System.Drawing;
 6  using System.Linq;
 7  using System.Text;
 8  using System.Threading.Tasks;
 9  using System.Windows.Forms;
10
11  namespace WindowsFormsApp2
12  {
13      public partial class miniGameFrm : Form
14      {
15
16          bool jumping = false;
17          int jumpSpeed;
18          int force = 17;
19          int score = 0;
20          int obstacleSpeed = 10;
21          Random rand = new Random();
22          int position = 0;
23          bool isGameOver = false;
24
25          public miniGameFrm()
26          {
27              InitializeComponent();
28
29              GameReset();
30          }
31
32          private void MainGameTimerEvent(object sender, EventArgs e)
33          {
34              picGameHero.Top += jumpSpeed;
35
36              txtScore.Text = "Score : " + score;
37
38              if (jumping == true && force < 0)
39              {
40                  jumping = false;
41              }
42
43              if (jumping == true)
44              {
45                  jumpSpeed = -12;
46                  force -= 1;
47              }
48              else
49              {
50                  jumpSpeed = 15;
51              }
52
53              if (picGameHero.Top > 430 && jumping == false)
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\miniGame.cs 2

```
54              {
55                  force = 17;
56                  picGameHero.Top = 431;
57                  jumpSpeed = 0;
58              }
59
60              foreach (Control x in this.Controls)
61              {
62                  if (x is PictureBox && (string)x.Tag == "obstacle")
63                  {
64                      x.Left -= obstacleSpeed;
65
66                      if (x.Left < -100)
67                      {
68                          x.Left = this.ClientSize.Width + rand.Next(200,
                       500) + (x.Width * 15);
69                          score++;
70                      }
71
72                      if (picGameHero.Bounds.IntersectsWith(x.Bounds))
73                      {
74                          gameTimer.Stop();
75                          picGameHero.Image =
                    Properties.Resources.gameHeroDead;
76                          txtScore.Text += ", Press K to restart the
                    game!";
77                          isGameOver = true;
78                      }
79                  }
80              }
81
82              if (score > 5)
83              {
84                  obstacleSpeed = 15;
85              }
86
87              if (score > 10)
88              {
89                  obstacleSpeed = 20;
90              }
91
92              if (score > 15)
93              {
94                  obstacleSpeed = 25;
95              }
96
97              if (score > 20)
98              {
99                  obstacleSpeed = 30;
100             }
101
102             if (score > 25)
103             {
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\miniGame.cs 3

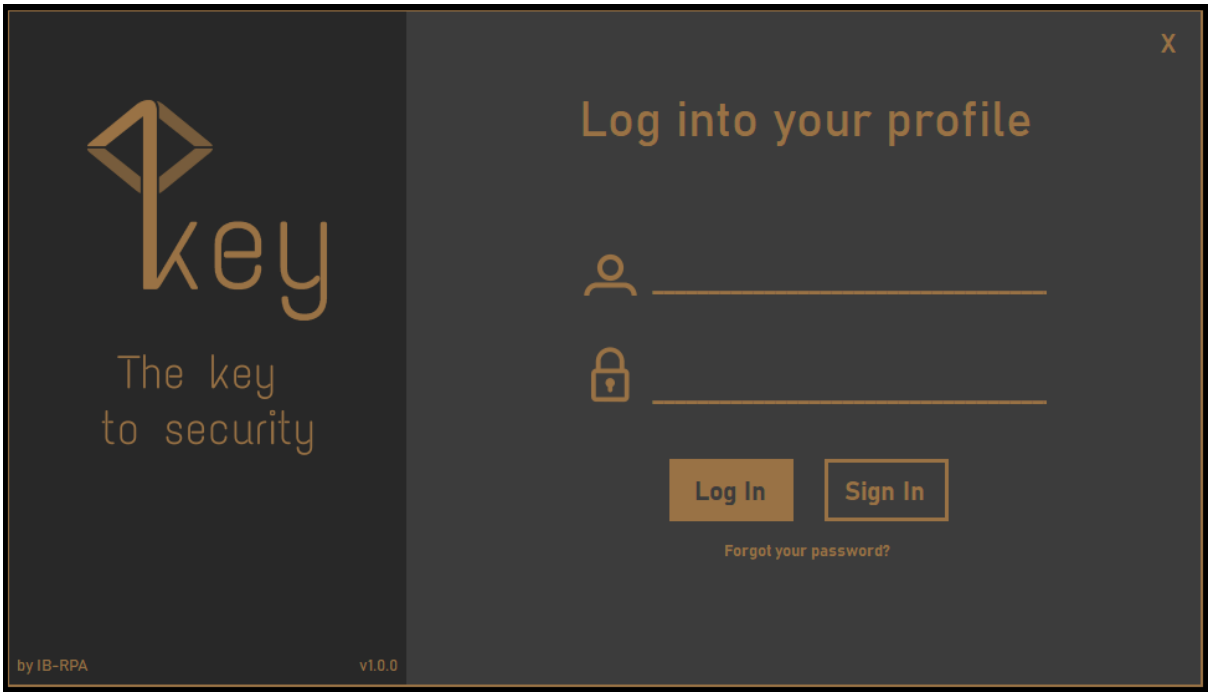
```
104                 obstacleSpeed = 35;
105             }
106
107             if (score > 30)
108             {
109                 obstacleSpeed = 50;
110             }
111         }
112
113         private void KeyIsDown(object sender, KeyEventArgs e)
114         {
115             if (e.KeyCode == Keys.Space && jumping == false)
116             {
117                 jumping = true;
118             }
119         }
120
121         private void KeyIsUp(object sender, PreviewKeyDownEventArgs e)
122         {
123             if (jumping == true)
124             {
125                 jumping = false;
126             }
127
128             if (e.KeyCode == Keys.K && isGameOver == true)
129             {
130                 GameReset();
131             }
132
133             if (e.KeyCode == Keys.Escape)
134             {
135                 this.Close();
136                 var newform = new managerFrm();
137                 newform.Show();
138             }
139         }
140
141         private void GameReset()
142         {
143             force = 17;
144             jumpSpeed = 0;
145             jumping = false;
146             score = 0;
147             obstacleSpeed = 10;
148             txtScore.Text = "Score : " + score;
149             picGameHero.Image = Properties.Resources.gameHeroSkeyt;
150             isGameOver = false;
151             picGameHero.Top = 431;
152
153
154             foreach (Control x in this.Controls)
155             {
156                 if (x is PictureBox && (string)x.Tag == "obstacle")
```

C:\Users\Ivan\Desktop\1Key\1Keyapp\1Key\miniGame.cs 4

```
157                 {
158                     position = this.ClientSize.Width + rand.Next(500,
                      800) + (x.Width * 10);
159
160                     x.Left = position;
161                 }
162             }
163             gameTimer.Start();
164         }
165     }
166 }
167
```

Prilog 2.: Ekranski prikazi ispravljenih korisničkih sučelja prototipa aplikacije OneKey

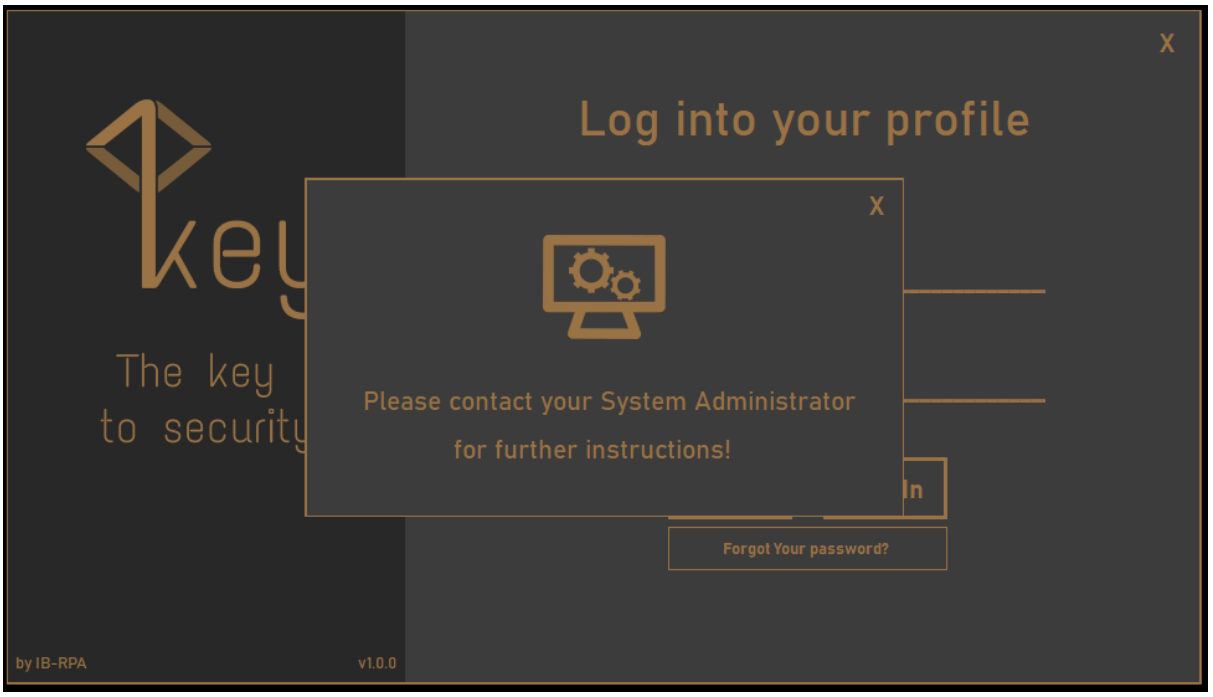
Sučelje prijave u aplikaciju:



Sučelje registracije:

Prilog 2.: Ekranski prikazi ispravljenih korisničkih sučelja prototipa aplikacije OneKey

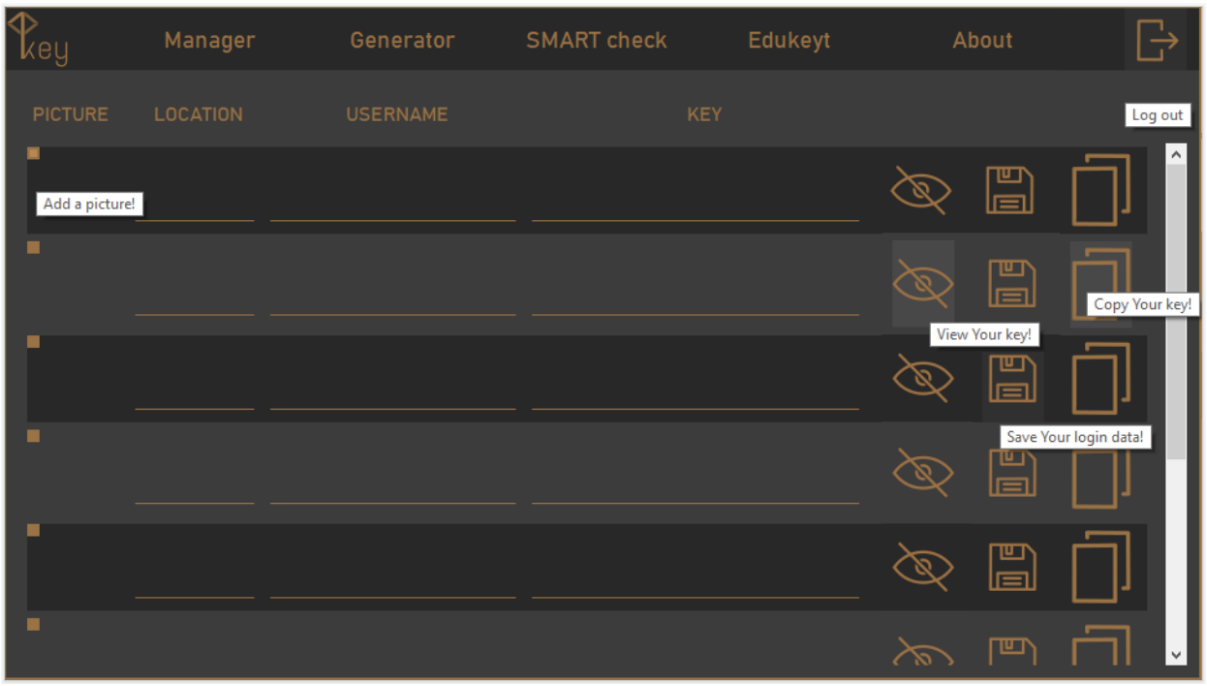
Dijaloški okvir administratora:



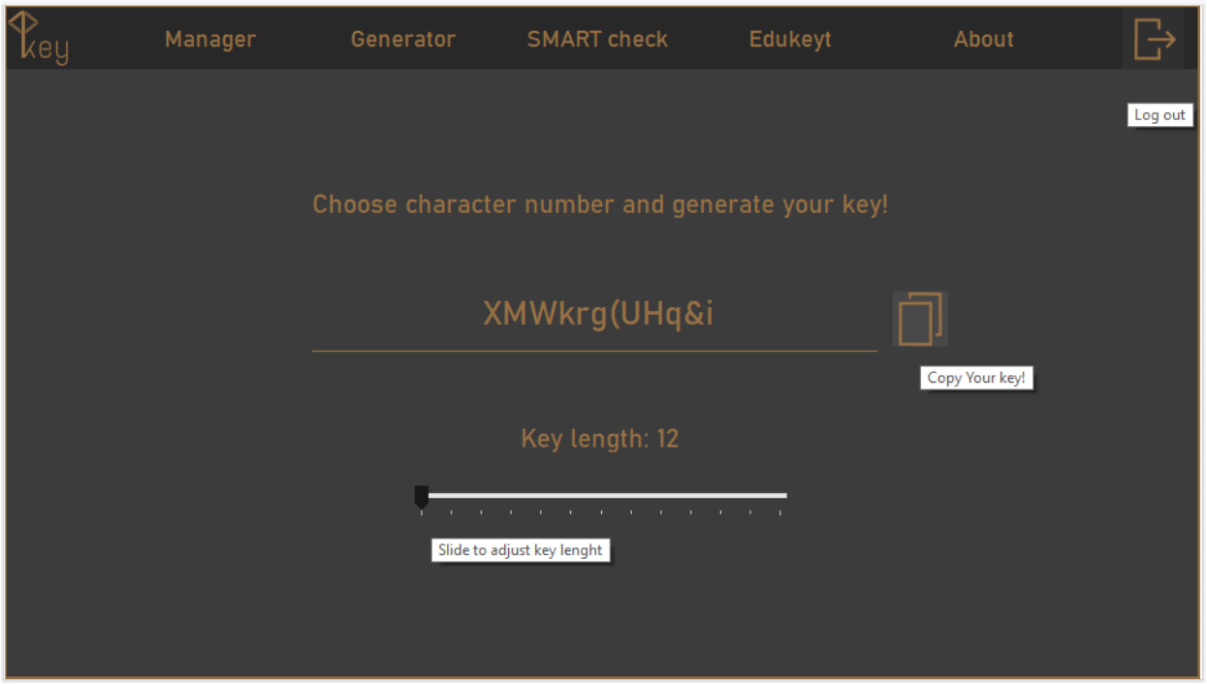
Pozdravni zaslon:

Prilog 2.: Ekranski prikazi ispravljenih korisničkih sučelja prototipa aplikacije OneKey

Sučelje upravitelja lozinki:

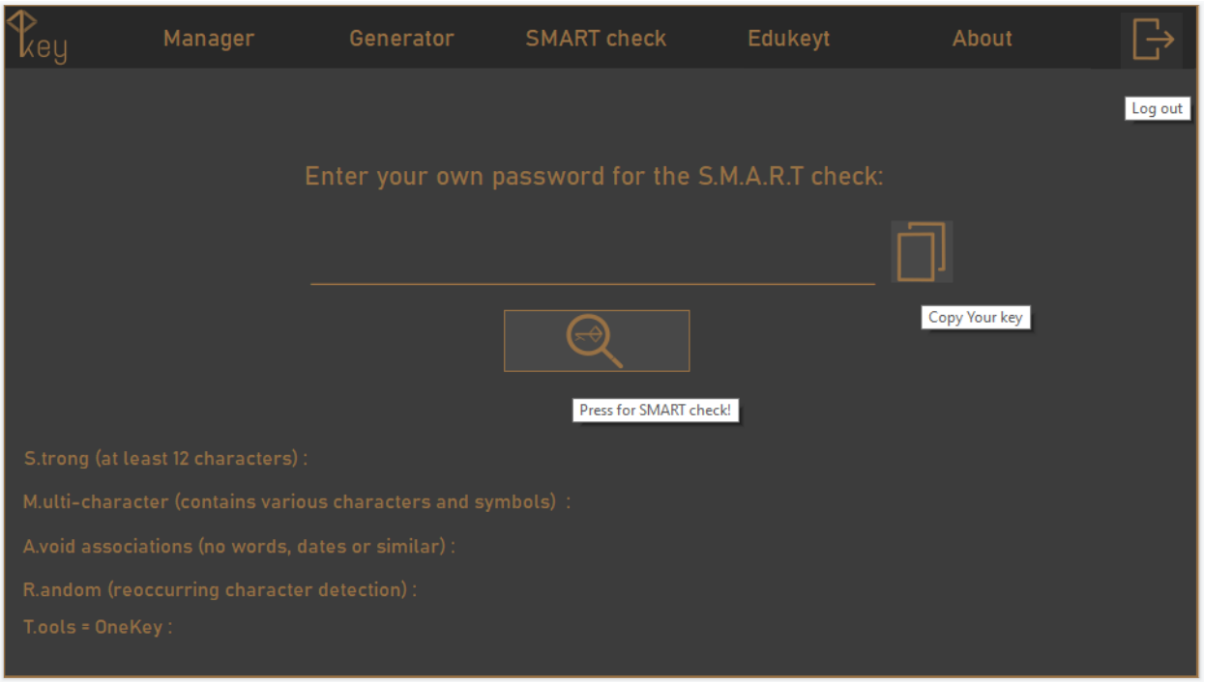


Sučelje generatora lozinki:

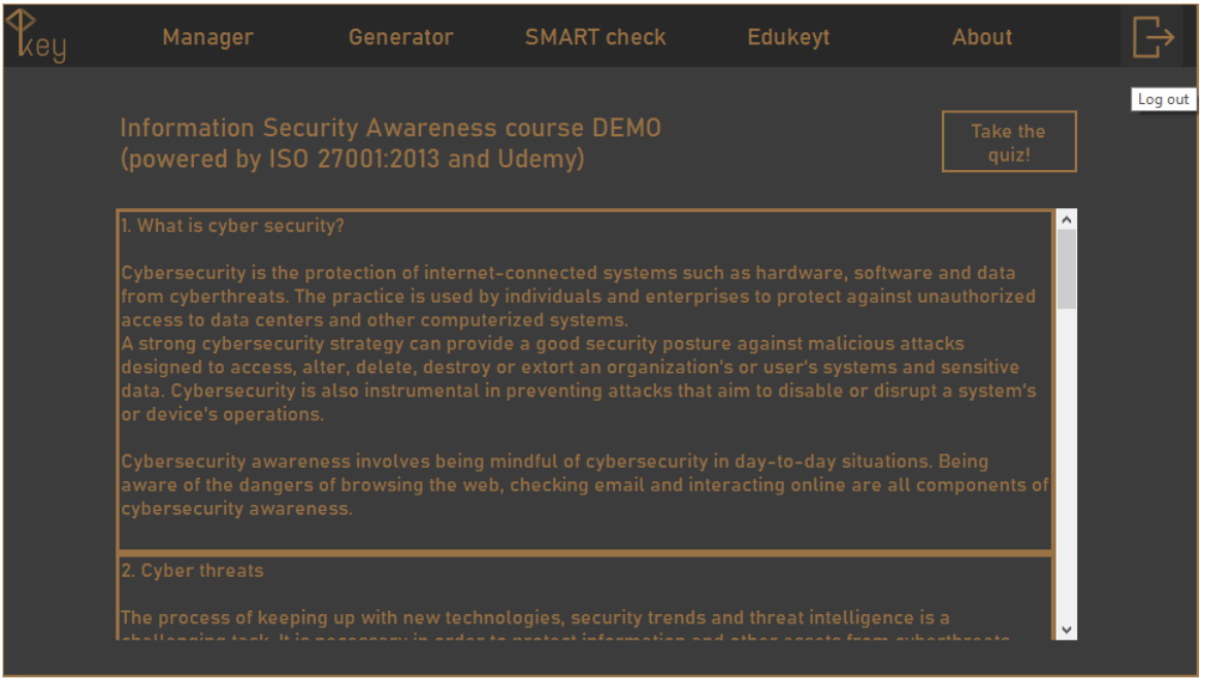


Prilog 2.: Ekranski prikazi ispravljenih korisničkih sučelja prototipa aplikacije OneKey

Sučelje ovjere lozinki SMART metodologijom:

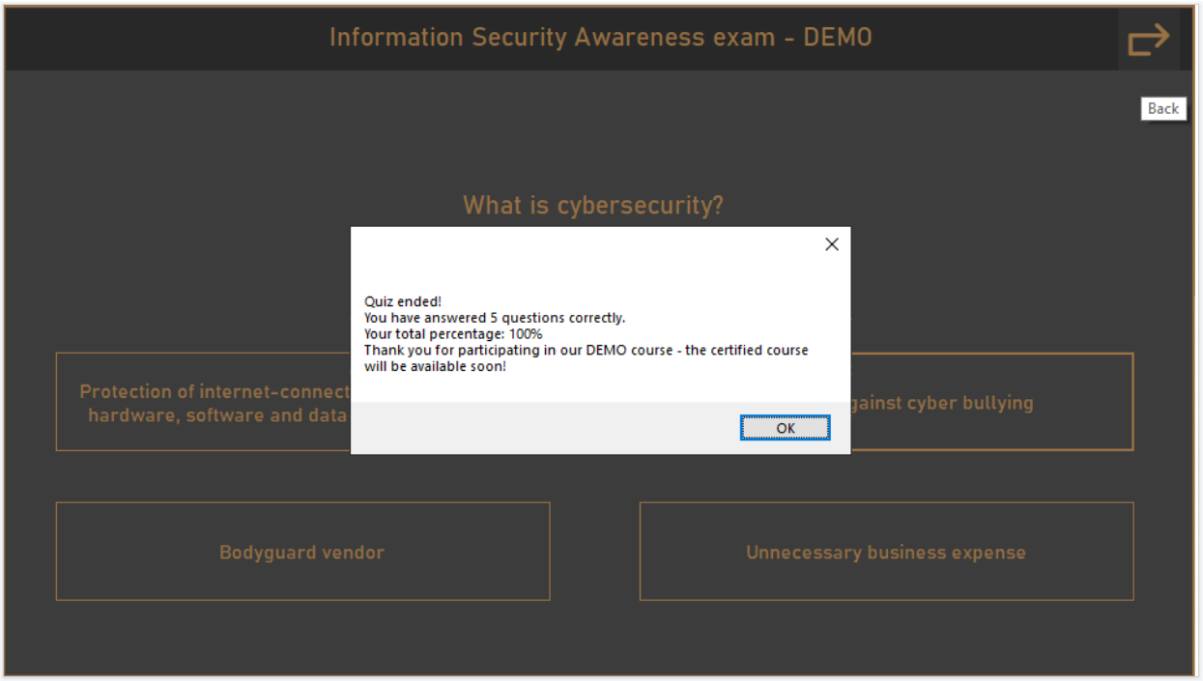


Edukacijsko sučelje:

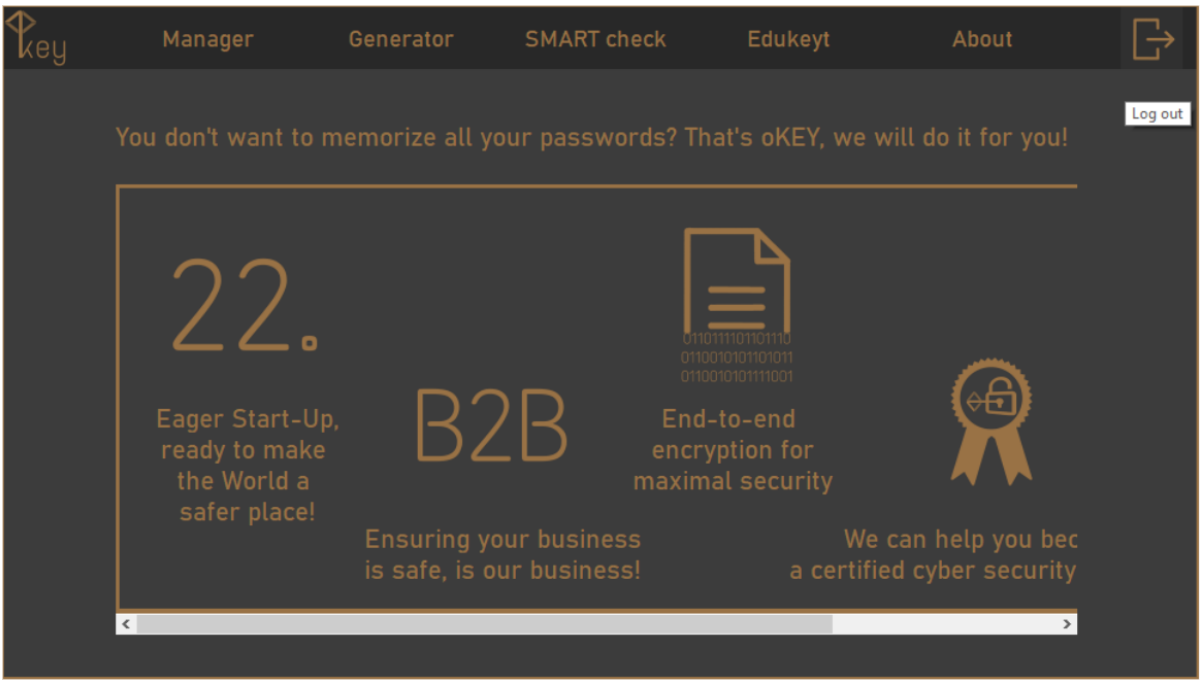


Prilog 2.: Ekranski prikazi ispravljenih korisničkih sučelja prototipa aplikacije OneKey

Kviz certifikacije o svijesti o informacijskoj sigurnosti:

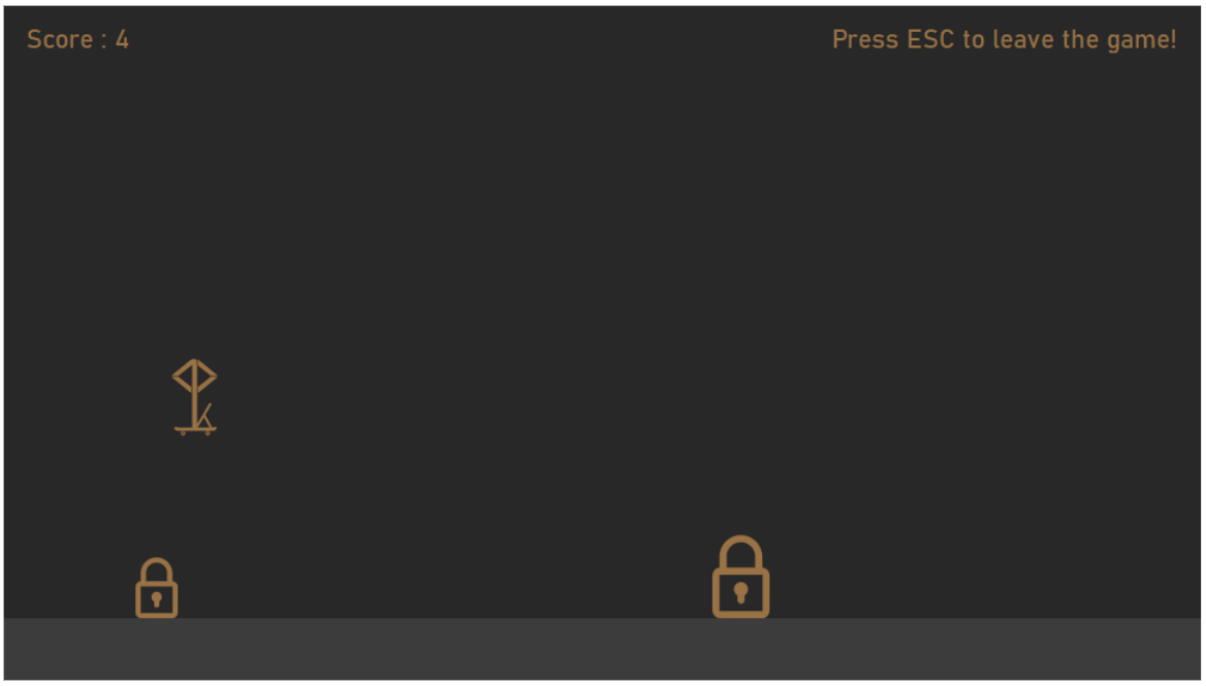


Pregled osnovnih informacija aplikacije:



Prilog 2.: Ekranski prikazi ispravljenih korisničkih sučelja prototipa aplikacije OneKey

Sučelje igrice:



File - C:\Users\Ivan\Desktop\1Key\Automation\Search.py

```
import os
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
import unittest
import HtmlTestRunner
import random

random_search = ['aaaaaaaHJGFJVKUZFTDzuftftzfd65R&778o8709gbz7h798h87gftff', 'abc',
                 'aa bb  cc  dd  ee  ff  gg  hh  II  JJ  11  22  33  -- 55  ++', 'ABC',
                 '12Cdefg', 'Qaqaqaqaqaqa', 'prQAefgh', 'QA test automation job', 'GUI',
                 'User experience', '     ', '', 'efzgggggggg', 'Human-computercommunication']

class GoogleSearch(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        os.environ['PATH'] += r"C:/Users/Ivan/Desktop/1Key/SeleniumDrivers"
        cls.driver = webdriver.Chrome()
        cls.driver.implicitly_wait(10)
        cls.driver.maximize_window()

    def test_search_random0(self):
        self.driver.get("https://google.com")
        self.driver.find_element(By.ID, "W0wltc").click()
        self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
        self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)

    def test_search_random1(self):
        self.driver.get("https://google.com")
        self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
        self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)

    def test_search_random2(self):
        self.driver.get("https://google.com")
        self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
        self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)

    def test_search_random3(self):
        self.driver.get("https://google.com")
        self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
        self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)

    def test_search_random4(self):
        self.driver.get("https://google.com")
        self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
        self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)

    def test_search_random5(self):
        self.driver.get("https://google.com")
        self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
        self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)

    def test_search_random6(self):
        self.driver.get("https://google.com")
        self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
```

File - C:\Users\Ivan\Desktop\1Key\Automation\Search.py

```
57          self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)
58
59      def test_search_random7(self):
60          self.driver.get("https://google.com")
61          self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
62          self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)
63
64      def test_search_random8(self):
65          self.driver.get("https://google.com")
66          self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
67          self.driver.find_element(By.NAME, "q").send_keys(Keys.ENTER)
68
69      def test_search_random9(self):
70          self.driver.get("https://google.com")
71          self.driver.find_element(By.NAME, "q").send_keys(random.choice(random_search))
72          self.driver.find_element(By.NAME, "qq").send_keys(Keys.ENTER)
73
74
75      @classmethod
76      def tearDownClass(cls):
77          cls.driver.close()
78          cls.driver.quit()
79          print("Test Completed")
80
81  if __name__ == '__main__':
82      unittest.main(
83          testRunner=HtmlTestRunner.HTMLTestRunner(
84              output='C:/Users/Ivan/Desktop/1Key/Automation/AutomationReporting'
85                                                  )
86                  )
```

```
C:\Users\Ivan>C:\Users\Ivan\Desktop\1Key\Automation\Search.py

Running tests...
----------------------------------------------------------------------

DevTools listening on ws://127.0.0.1:53195/devtools/browser/48c081a8-3786-482b-adf8-9f7cf114cc2a
 test_search_random0 (__main__.GoogleSearch) ... OK (2.721913)s
 test_search_random1 (__main__.GoogleSearch) ... OK (2.712526)s
 test_search_random2 (__main__.GoogleSearch) ... OK (1.422986)s
 test_search_random3 (__main__.GoogleSearch) ... OK (0.445530)s
 test_search_random4 (__main__.GoogleSearch) ... OK (0.883214)s
 test_search_random5 (__main__.GoogleSearch) ... OK (1.073209)s
 test_search_random6 (__main__.GoogleSearch) ... OK (0.889211)s
 test_search_random7 (__main__.GoogleSearch) ... OK (2.502367)s
 test_search_random8 (__main__.GoogleSearch) ... OK (1.671758)s
[3296:1176:0904/111508.057:ERROR:device_event_log_impl.cc(214)] [11:15:08.056] USB: usb_device_handle_win.cc:1048 Failed to read descriptor from node connection: A device attached to the system is not functioning. (0x1F)
[3296:1176:0904/111508.057:ERROR:device_event_log_impl.cc(214)] [11:15:08.057] USB: usb_device_handle_win.cc:1048 Failed to read descriptor from node connection: A device attached to the system is not functioning. (0x1F)
 test_search_random9 (__main__.GoogleSearch) ... ERROR (10.539199)s

======================================================================
ERROR [10.539199s]: __main__.GoogleSearch.test_search_random9
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Ivan\Desktop\1Key\Automation\Search.py", line 72, in test_search_random9
    self.driver.find_element(By.NAME, "qq").send_keys(Keys.ENTER)
  File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\webdriver.py", line 855, in find_element
    return self.execute(Command.FIND_ELEMENT, {
  File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\webdriver.py", line 428, in execute
    self.error_handler.check_response(response)
  File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\errorhandler.py", line 243, in check_response
    raise exception_class(message, screen, stacktrace)
selenium.common.exceptions.NoSuchElementException: Message: no such element: Unable to locate element: {"method":"css selector","selector":"[name="qq"]"}
  (Session info: chrome=105.0.5195.102)
Stacktrace:
Backtrace:
        Ordinal0 [0x003FDF13+2219795]
        Ordinal0 [0x00392841+1779777]
        Ordinal0 [0x002A423D+803389]
```

```
    Ordinal0 [0x002D3025+995365]
    Ordinal0 [0x002D31EB+995819]
    Ordinal0 [0x00300F52+1183570]
    Ordinal0 [0x002EE844+1108036]
    Ordinal0 [0x002FF192+1175954]
    Ordinal0 [0x002EE616+1107478]
    Ordinal0 [0x002C7F89+950153]
    Ordinal0 [0x002C8F56+954198]
    GetHandleVerifier [0x006F2CB2+3040210]
    GetHandleVerifier [0x006E2BB4+2974420]
    GetHandleVerifier [0x00496A0A+565546]
    GetHandleVerifier [0x00495680+560544]
    Ordinal0 [0x00399A5C+1808988]
    Ordinal0 [0x0039E3A8+1827752]
    Ordinal0 [0x0039E495+1827989]
    Ordinal0 [0x003A80A4+1867940]
    BaseThreadInitThunk [0x766BFA29+25]
    RtlGetAppContainerNamedObjectPath [0x77257A9E+286]
    RtlGetAppContainerNamedObjectPath [0x77257A6E+238]


----------------------------------------------------------------------
Ran 10 tests in 0:00:30

FAILED
 (Errors=1)

Generating HTML reports...
Desktop\1Key\Automation\AutomationReporting\TestResults___main__.GoogleSearch_202
2-09-04_11-14-49.html
```

File - C:\Users\Ivan\Desktop\1Key\Automation\Login.py

```
import os
from selenium import webdriver
from selenium.webdriver.common.by import By
import unittest
import HtmlTestRunner
from time import sleep
import random

chrome_options = webdriver.ChromeOptions()
chrome_options.add_argument('--incognito')

username = ['test.automation.korisnik@gmail.com']

key = ['TestAutomation2022!', 'Testautomation2022!', 'testAutomation2022!',
       'testautomation2022!', 'TESTAUTOMATION2022!', 'tESTaUTOMATION2022!',
       'Testautonation2022!', 'testaautomation2022!', 'TestAutomation2220222!',
       'tesAutomation2022!', 'Test Automation2022!', '!TestAutomation2022']

class Login(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        os.environ['PATH'] += r"C:/Users/Ivan/Desktop/1Key/SeleniumDrivers"
        cls.driver = webdriver.Chrome(options=chrome_options)
        cls.driver.maximize_window()

    def test_search_login1(self):
        self.driver.get("https://gmail.google.com")
        self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
        self.driver.find_element(By.ID, "identifierNext").click()
        sleep(3)
        self.driver.find_element(By.NAME, "Passwd").send_keys(random.choice(key))
        self.driver.find_element(By.ID, "passwordNext").click()
        sleep(3)
        self.driver.get("https://accounts.google.com/Logout?hl=en&continue=https:"
                        "//www.google.com/search%3Fq%3Dgmail%26oq%3Dgmail%26aqs%3Dchrome"
                        "..69i57j0i271l2j69i65l2.670j0j1%26sourceid%3Dchrome%26ie%3DUTF-"
                        "8&timeStmp=1662305131&secTok=.AG5fkS_BFBrvaFMtIHLOoXLSG7k5ArRVvw&ec=GAdAAQ")
        sleep(3)
        self.driver.find_element(By.ID, "W0wltc").click()

    def test_search_login2(self):
        self.driver.get("https://gmail.google.com")
        self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/"
                                           "div/div[2]/div/div[1]/div/form/span/section/div/"
                                           "div/div/div/ul/li[3]").click()
        self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/"
                                           "div/div[2]/div/div[1]/div/form/span/section/div/"
                                           "div/div/div[1]/ul/li[1]/div").click()
        self.driver.find_element(By.XPATH, "/html/body/div[5]/div/div[2]/div[3]/div[1]").click()
        self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
        self.driver.find_element(By.ID, "identifierNext").click()
        sleep(3)
        self.driver.find_element(By.NAME, "Passwd").send_keys(random.choice(key))
        self.driver.find_element(By.ID, "passwordNext").click()
        sleep(3)
        self.driver.get("https://accounts.google.com/Logout?hl=en&continue=https://www.google.com/"
                        "search%3Fq%3Dgmail%26oq%3Dgmail%26aqs%3Dchrome..69i57j0i271l2j69i65l2.670j0j"
                        "1%26sourceid%3Dchrome%26ie%3DUTF-8&timeStmp=1662305131&secTok=.AG5fkS_"
                        "BFBrvaFMtIHLOoXLSG7k5ArRVvw&ec=GAdAAQ")
        sleep(3)
        self.driver.find_element(By.ID, "W0wltc").click()
```

File - C:\Users\Ivan\Desktop\1Key\Automation\Login.py

```
63          self.driver.get("https://gmail.google.com")
64          sleep(3)
65
66      def test_search_login3(self):
67          self.driver.get("https://gmail.google.com")
68          self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
69          self.driver.find_element(By.ID, "identifierNext").click()
70          sleep(3)
71          self.driver.find_element(By.NAME, "Passwd").send_keys(random.choice(key))
72          self.driver.find_element(By.ID, "passwordNext").click()
73          sleep(3)
74          self.driver.get("https://accounts.google.com/Logout?hl=en&continue=https://www.google.com/"
75                          "search%3Fq%3Dgmail%26oq%3Dgmail%26aqs%3Dchrome..69i57j0i271l2j69i65l2.670"
76                          "j0j1%26sourceid%3Dchrome%26ie%3DUTF-8&timeStmp=1662305131&secTok=.AG5fkS_"
77                          "BFBrvaFMtIHLOoXLSG7k5ArRVvw&ec=GAdAAQ")
78          sleep(3)
79          self.driver.get("https://gmail.google.com")
80          sleep(3)
81          self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/div/"
82                                             "div[2]/div/div[1]/div/form/span/section/div/div/div/"
83                                             "div/ul/li[3]").click()
84          self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/div/"
85                                             "div[2]/div/div[1]/div/form/span/section/div/div/div/"
86                                             "div[1]/ul/li[1]/div").click()
87          self.driver.find_element(By.XPATH, "/html/body/div[5]/div/div[2]/div[3]/div[1]").click()
88          sleep(5)
89
90      def test_search_login4(self):
91          self.driver.get("https://gmail.google.com")
92          self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
93          self.driver.find_element(By.ID, "identifierNext").click()
94          sleep(3)
95          self.driver.find_element(By.NAME, "Passwd").send_keys(random.choice(key))
96          self.driver.find_element(By.ID, "passwordNext").click()
97          sleep(3)
98          self.driver.get("https://accounts.google.com/Logout?hl=en&continue=https://www.google.com/"
99                          "search%3Fq%3Dgmail%26oq%3Dgmail%26aqs%3Dchrome..69i57j0i271l2j69i65l2.670"
100                         "j0j1%26sourceid%3Dchrome%26ie%3DUTF-8&timeStmp=1662305131&secTok=.AG5fkS_"
101                         "BFBrvaFMtIHLOoXLSG7k5ArRVvw&ec=GAdAAQ")
102         sleep(3)
103         self.driver.get("https://gmail.google.com")
104         sleep(3)
105         self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/div/"
106                                            "div[2]/div/div[1]/div/form/span/section/div/div/div/"
107                                            "div/ul/li[3]").click()
108         self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/div/"
109                                            "div[2]/div/div[1]/div/form/span/section/div/div/div/"
110                                            "div[1]/ul/li[1]/div").click()
111         self.driver.find_element(By.XPATH, "/html/body/div[5]/div/div[2]/div[3]/div[1]").click()
112         sleep(5)
113
114     def test_search_login5(self):
115         self.driver.get("https://gmail.google.com")
116         self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
117         self.driver.find_element(By.ID, "identifierNext").click()
118         sleep(3)
119         self.driver.find_element(By.NAME, "Passwd").send_keys(random.choice(key))
120         self.driver.find_element(By.ID, "passwordNext").click()
121         sleep(3)
122         self.driver.get("https://accounts.google.com/Logout?hl=en&continue=https://www.google.com/"
123                         "search%3Fq%3Dgmail%26oq%3Dgmail%26aqs%3Dchrome..69i57j0i271l2j69i65l2.670"
124                         "j0j1%26sourceid%3Dchrome%26ie%3DUTF-8&timeStmp=1662305131&secTok=.AG5fkS_"
```

File - C:\Users\Ivan\Desktop\1Key\Automation\Login.py

```
                        "BFBrvaFMtIHLOoXLSG7k5ArRVvw&ec=GAdAAQ")
        sleep(3)
        self.driver.get("https://gmail.google.com")
        sleep(3)
        self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/div/"
                                           "div[2]/div/div[1]/div/form/span/section/div/div/div/"
                                           "div/ul/li[3]").click()
        self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/div/"
                                           "div[2]/div/div[1]/div/form/span/section/div/div/div/"
                                           "div[1]/ul/li[1]/div").click()
        self.driver.find_element(By.XPATH, "/html/body/div[5]/div/div[2]/div[3]/div[1]").click()
        sleep(5)

    @classmethod
    def tearDownClass(cls):
        cls.driver.close()
        cls.driver.quit()
        print("Test Completed")

if __name__ == '__main__':
    unittest.main(
        testRunner=HtmlTestRunner.HTMLTestRunner(
            output='C:/Users/Ivan/Desktop/1Key/Automation/AutomationReporting'
                                                )
            )
```

```
C:\Users\Ivan>C:\Users\Ivan\Desktop\1Key\Automation\Login.py

Running tests...
----------------------------------------------------------------------

DevTools listening on ws://127.0.0.1:54809/devtools/browser/eaee2568-7b12-4b5c-9844-f6b53325d3be
[2676:16000:0904/181619.977:ERROR:device_event_log_impl.cc(214)] [18:16:19.976] USB: usb_device_handle_win.cc:1048 Failed to read descriptor from node connection: A device attached to the system is not functioning. (0x1F)
[2676:16000:0904/181619.978:ERROR:device_event_log_impl.cc(214)] [18:16:19.977] USB: usb_device_handle_win.cc:1048 Failed to read descriptor from node connection: A device attached to the system is not functioning. (0x1F)
 test_search_login1 (__main__.Login) ... ERROR (7.476660)s
 test_search_login2 (__main__.Login) ... ERROR (2.662869)s
 test_search_login3 (__main__.Login) ... ERROR (0.417825)s
 test_search_login4 (__main__.Login) ... ERROR (0.481796)s
 test_search_login5 (__main__.Login) ... ERROR (0.820653)s

======================================================================
ERROR [7.476660s]: __main__.Login.test_search_login1
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Ivan\Desktop\1Key\Automation\Login.py", line 32, in test_search_login1
    self.driver.find_element(By.NAME, "Passwd").send_keys(random.choice(key))
                                                                      File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\webdriver.py", line 855, in find_element
    return self.execute(Command.FIND_ELEMENT, {
                                                                      File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\webdriver.py", line 428, in execute
    self.error_handler.check_response(response)
                                                                      File "C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdriver\remote\errorhandler.py", line 243, in check_response
    raise exception_class(message, screen, stacktrace)
selenium.common.exceptions.NoSuchElementException: Message: no such element: Unable to locate element: {"method":"css selector","selector":"[name="Passwd"]"}
  (Session info: chrome=105.0.5195.102)
Stacktrace:
Backtrace:
        Ordinal0 [0x003FDF13+2219795]
        Ordinal0 [0x00392841+1779777]
        Ordinal0 [0x002A423D+803389]
        Ordinal0 [0x002D3025+995365]
        Ordinal0 [0x002D31EB+995819]
        Ordinal0 [0x00300F52+1183570]
        Ordinal0 [0x002EE844+1108036]
        Ordinal0 [0x002FF192+1175954]
```

```
        Ordinal0 [0x002EE616+1107478]
        Ordinal0 [0x002C7F89+950153]
        Ordinal0 [0x002C8F56+954198]
        GetHandleVerifier [0x006F2CB2+3040210]
        GetHandleVerifier [0x006E2BB4+2974420]
        GetHandleVerifier [0x00496A0A+565546]
        GetHandleVerifier [0x00495680+560544]
        Ordinal0 [0x00399A5C+1808988]
        Ordinal0 [0x0039E3A8+1827752]
        Ordinal0 [0x0039E495+1827989]
        Ordinal0 [0x003A80A4+1867940]
        BaseThreadInitThunk [0x766BFA29+25]
        RtlGetAppContainerNamedObjectPath [0x77257A9E+286]
        RtlGetAppContainerNamedObjectPath [0x77257A6E+238]


======================================================================
ERROR [2.662869s]: __main__.Login.test_search_login2
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Ivan\Desktop\1Key\Automation\Login.py", line 44, in test_search_login2
    self.driver.find_element(By.XPATH, "/html/body/div[1]/div[1]/div[2]/div/div[2]/div/"
                                                                                  File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\webdriver.py", line 855, in find_element
    return self.execute(Command.FIND_ELEMENT, {
                                                                                  File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\webdriver.py", line 428, in execute
    self.error_handler.check_response(response)
                                                                                  File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\errorhandler.py", line 243, in check_response
    raise exception_class(message, screen, stacktrace)
selenium.common.exceptions.NoSuchElementException: Message: no such element: Unable
to                                locate                                element:
{"method":"xpath","selector":"/html/body/div[1]/div[1]/div[2]/div/div[2]/div/div/div[2]/div/d
iv[1]/div/form/span/section/div/div/div/div/ul/li[3]"}
  (Session info: chrome=105.0.5195.102)
Stacktrace:
Backtrace:
        Ordinal0 [0x003FDF13+2219795]
        Ordinal0 [0x00392841+1779777]
        Ordinal0 [0x002A423D+803389]
        Ordinal0 [0x002D3025+995365]
        Ordinal0 [0x002D31EB+995819]
        Ordinal0 [0x00300F52+1183570]
        Ordinal0 [0x002EE844+1108036]
        Ordinal0 [0x002FF192+1175954]
        Ordinal0 [0x002EE616+1107478]
```

```
        Ordinal0 [0x002C7F89+950153]
        Ordinal0 [0x002C8F56+954198]
        GetHandleVerifier [0x006F2CB2+3040210]
        GetHandleVerifier [0x006E2BB4+2974420]
        GetHandleVerifier [0x00496A0A+565546]
        GetHandleVerifier [0x00495680+560544]
        Ordinal0 [0x00399A5C+1808988]
        Ordinal0 [0x0039E3A8+1827752]
        Ordinal0 [0x0039E495+1827989]
        Ordinal0 [0x003A80A4+1867940]
        BaseThreadInitThunk [0x766BFA29+25]
        RtlGetAppContainerNamedObjectPath [0x77257A9E+286]
        RtlGetAppContainerNamedObjectPath [0x77257A6E+238]


======================================================================
ERROR [0.417825s]: __main__.Login.test_search_login3
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Ivan\Desktop\1Key\Automation\Login.py", line 68, in test_search_login3
    self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
                                                                                  File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\webdriver.py", line 855, in find_element
    return self.execute(Command.FIND_ELEMENT, {
                                                                                  File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\webdriver.py", line 428, in execute
    self.error_handler.check_response(response)
                                                                                  File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\errorhandler.py", line 243, in check_response
    raise exception_class(message, screen, stacktrace)
selenium.common.exceptions.NoSuchElementException: Message: no such element: Unable
to locate element: {"method":"css selector","selector":"[id="identifierId"]"}
  (Session info: chrome=105.0.5195.102)
Stacktrace:
Backtrace:
        Ordinal0 [0x003FDF13+2219795]
        Ordinal0 [0x00392841+1779777]
        Ordinal0 [0x002A423D+803389]
        Ordinal0 [0x002D3025+995365]
        Ordinal0 [0x002D31EB+995819]
        Ordinal0 [0x00300F52+1183570]
        Ordinal0 [0x002EE844+1108036]
        Ordinal0 [0x002FF192+1175954]
        Ordinal0 [0x002EE616+1107478]
        Ordinal0 [0x002C7F89+950153]
        Ordinal0 [0x002C8F56+954198]
        GetHandleVerifier [0x006F2CB2+3040210]
```

```
        GetHandleVerifier [0x006E2BB4+2974420]
        GetHandleVerifier [0x00496A0A+565546]
        GetHandleVerifier [0x00495680+560544]
        Ordinal0 [0x00399A5C+1808988]
        Ordinal0 [0x0039E3A8+1827752]
        Ordinal0 [0x0039E495+1827989]
        Ordinal0 [0x003A80A4+1867940]
        BaseThreadInitThunk [0x766BFA29+25]
        RtlGetAppContainerNamedObjectPath [0x77257A9E+286]
        RtlGetAppContainerNamedObjectPath [0x77257A6E+238]


======================================================================
ERROR [0.481796s]: __main__.Login.test_search_login4
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Ivan\Desktop\1Key\Automation\Login.py", line 92, in test_search_login4
    self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
                                                                                File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\webdriver.py", line 855, in find_element
    return self.execute(Command.FIND_ELEMENT, {
                                                                                File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\webdriver.py", line 428, in execute
    self.error_handler.check_response(response)
                                                                                File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\errorhandler.py", line 243, in check_response
    raise exception_class(message, screen, stacktrace)
selenium.common.exceptions.NoSuchElementException: Message: no such element: Unable
to locate element: {"method":"css selector","selector":"[id="identifierId"]"}
  (Session info: chrome=105.0.5195.102)
Stacktrace:
Backtrace:
        Ordinal0 [0x003FDF13+2219795]
        Ordinal0 [0x00392841+1779777]
        Ordinal0 [0x002A423D+803389]
        Ordinal0 [0x002D3025+995365]
        Ordinal0 [0x002D31EB+995819]
        Ordinal0 [0x00300F52+1183570]
        Ordinal0 [0x002EE844+1108036]
        Ordinal0 [0x002FF192+1175954]
        Ordinal0 [0x002EE616+1107478]
        Ordinal0 [0x002C7F89+950153]
        Ordinal0 [0x002C8F56+954198]
        GetHandleVerifier [0x006F2CB2+3040210]
        GetHandleVerifier [0x006E2BB4+2974420]
        GetHandleVerifier [0x00496A0A+565546]
        GetHandleVerifier [0x00495680+560544]
```

```
        Ordinal0 [0x00399A5C+1808988]
        Ordinal0 [0x0039E3A8+1827752]
        Ordinal0 [0x0039E495+1827989]
        Ordinal0 [0x003A80A4+1867940]
        BaseThreadInitThunk [0x766BFA29+25]
        RtlGetAppContainerNamedObjectPath [0x77257A9E+286]
        RtlGetAppContainerNamedObjectPath [0x77257A6E+238]


======================================================================
ERROR [0.820653s]: __main__.Login.test_search_login5
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Ivan\Desktop\1Key\Automation\Login.py", line 116, in test_search_login5
    self.driver.find_element(By.ID, "identifierId").send_keys(random.choice(username))
                                                                                File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\webdriver.py", line 855, in find_element
    return self.execute(Command.FIND_ELEMENT, {
                                                                                File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\webdriver.py", line 428, in execute
    self.error_handler.check_response(response)
                                                                                File
"C:\Users\Ivan\AppData\Local\Programs\Python\Python39\lib\site-packages\selenium\webdr
iver\remote\errorhandler.py", line 243, in check_response
    raise exception_class(message, screen, stacktrace)
selenium.common.exceptions.NoSuchElementException: Message: no such element: Unable
to locate element: {"method":"css selector","selector":"[id="identifierId"]"}
  (Session info: chrome=105.0.5195.102)
Stacktrace:
Backtrace:
        Ordinal0 [0x003FDF13+2219795]
        Ordinal0 [0x00392841+1779777]
        Ordinal0 [0x002A423D+803389]
        Ordinal0 [0x002D3025+995365]
        Ordinal0 [0x002D31EB+995819]
        Ordinal0 [0x00300F52+1183570]
        Ordinal0 [0x002EE844+1108036]
        Ordinal0 [0x002FF192+1175954]
        Ordinal0 [0x002EE616+1107478]
        Ordinal0 [0x002C7F89+950153]
        Ordinal0 [0x002C8F56+954198]
        GetHandleVerifier [0x006F2CB2+3040210]
        GetHandleVerifier [0x006E2BB4+2974420]
        GetHandleVerifier [0x00496A0A+565546]
        GetHandleVerifier [0x00495680+560544]
        Ordinal0 [0x00399A5C+1808988]
        Ordinal0 [0x0039E3A8+1827752]
        Ordinal0 [0x0039E495+1827989]
```

```
    Ordinal0 [0x003A80A4+1867940]
    BaseThreadInitThunk [0x766BFA29+25]
    RtlGetAppContainerNamedObjectPath [0x77257A9E+286]
    RtlGetAppContainerNamedObjectPath [0x77257A6E+238]


----------------------------------------------------------------------
Ran 5 tests in 0:00:15

FAILED
 (Errors=5)

Generating HTML reports...
Desktop\1Key\Automation\AutomationReporting\TestResults___main__.Login_2022-09-04_
18-16-11.html
```