Taikomieji moksliniai tyrimai / Applied Scientific Research ISSN 2783-6290
2023, 2(2), 20-27. DOI: https://doi.org/10.56131/tmt.2023.2.2.160 Gauta 2023-05-23. Publikuota 2023-10-18

Kompiuteryje atliekamų pasikartojančių veiksmų automatizavimo sistemos kūrimas

Gytis Medelis
Šiaulių valstybinė kolegija, studijų programos „Programų sistemos“ absolventas
Šiaulių valstybinė kolegija / Higher Education Institution, Lithuania; Graduate of the study program “Program Systems”

Jovita Urnikienė
Šiaulių valstybinė kolegija, Informatikos mokslų katedros lektorė
Šiaulių valstybinė kolegija / Higher Education Institution, Lithuania; Lecturer of the Department of Informatics Sciences
j.urnikiene@svako.lt

Anotacija

Tobulėjant technologijoms bei taupant laiką vis daugiau darbų automatizuojama, vis mažiau jų atliekama rankomis. Automatizavimo procesas naudingas ir dirbant kompiuteriu.

Straipsnyje nagrinėjama, kaip sukurta pelės paspaudimų, braukimų ir teksto rašymo imitavimo sistema, suteikianti galimybę atlikti pasikartojančius veiksmus, susikurti ir išsisaugoti veiksmų vykdymo algoritmą.

Teoriniame straipsnio skyriuje aptariamos panašios pelės paspaudimų automatizuotos sistemos, jų nauda ir pritaikymas įvairiose srityse, jų privalumai ir trūkumai, teksto ir grafinių elementų atpažinimo algoritmai, populiariausios bibliotekos. Analizuojamas sistemų atitikimas ISO/IEC 9126 standartui. Sistemos programavimui pasirinkta C++ programavimo kalba, Qt karkasas, teksto atpažinimui – tesseract biblioteka, grafiniams objektams – OpenCV biblioteka.

Praktiniame straipsnio skyriuje pateiktas kuriamos automatizuotų veiksmų sistemos projektas, aptartas kūrimo procesas, pati sistema, pateikti testavimo rezultatai. Sistema turi 10 mygtukų valdymo įrankių: teksto, spalvos ir nuotraukų atpažinimų sistemos, pelės paspaudimo, teksto rašymo ir braukimo imitavimo sistemos, automatizuoto veiksmo trynimas, algoritmo vykdymas arba stabdymas, nustatymų parinkimas, meniu sutraukimas arba išplėtimas, sistemos išjungimas. Naudojant sistemą yra numatyta galimybė kurti pelės paspaudimų, braukimų ir teksto įvedimo algoritmą, nustatyti sąlygas, kurti veiksmų instrukcijas, dalintis instrukcijomis forume. Sistema atpažįsta tekstus, spalvas ir nuotraukas.

Reikšminiai žodžiai: automatizuota veiksmų sistema, paspaudimų, braukimų ir teksto rašymo imitavimas, pasikartojančių veiksmų automatizavimas.

Creation of a system for automating repetitive actions performed on a computer

Summary

As technology advances and time saves time, more and more tasks are being automated, reducing the need for manual work. The process of automation is beneficial in computer-based work as well. This article analyzes automation systems for mouse clicks, their benefits, and their applications in various fields.

Project problem. The automated click systems do not provide flexible, reliable, and convenient algorithmic functions for the use of complex marketing systems, mobile games, or a broader market segment.

The objective of this work is to create a system that simulates mouse clicks, swipes, and text input, allowing for repetitive actions, creating and saving action execution algorithms.

The theoretical part discusses similar automated systems, their advantages and disadvantages, text and image recognition algorithms, and popular libraries. The compliance of the systems with the ISO/IEC 9126 standard is analyzed. The programming language chosen for development is C++, Qt framework, the Tesseract library is used for text recognition, while the OpenCV library is used for graphic objects.

The practical part presents the project of the created automated action system, the development process, the system itself, and the system's application possibilities. The system includes 10 control tools: text, color, and image recognition systems, mouse click, text input, and swipe simulation systems, automated action deletion, algorithm execution/stop, settings selection, menu collapse/expand, and system shutdown. The system allows for the creation of mouse click, swipe, and text input algorithms, defining conditions, creating action instructions, and sharing instructions on a forum. The system recognizes texts, colors, and images.

Keywords: automating click operations, automatic actions system, automated optical character recognition, automatic clicker.

Įvadas

Temos aktualumas ir naujumas. Žmogaus darbo automatizavimas yra neišvengiamas procesas, kuris vis spartėja. Remiantis emailmonday [1] duomenimis, pastebima, kad marketingo automatizavimo sistemų pajamos nuo 2013 metų iki 2019 metų išaugo nuo 750 milijonų iki 6 milijardų JAV dolerių. Vis daugiau marketingo įmonių pradeda kurti aplikacijų programavimo sąsajas, skirtas rinkti duomenis ir analizuoti marketingą (pvz., Binance API [2], Polygon API [3]). Marketingo įmonės suteikia galimybę naudoti jų sukurtas aplikacijų programavimo sąsajas, skirtas automatizuoti pirkimo ir pardavimo funkcijas. Tačiau ne visos marketingo įmonės turi šią aplikacijų programavimo sąsają. Marketingas gali būti tik smulki programos sistemos dalis. Hong Kongo universiteto tyrėjai Shing Ki Wong ir Siu Ming Yiu atliko mokslinį tyrimą [4], kuriame aptarė mobilių žaidimų išpopuliarėjimą ir staigų pajamų kilimą. Remiantis Statista [5] duomenimis, žaidimų pajamos sparčiai augo nuo 68,3 milijardų 2019 m. iki 92,2 milijardų 2022 m. JAV dolerių.

Šio straipsnio autoriai aptarė, kaip populiarūs žaidimai naudoja marketingo funkcijas ir kaip sukčiai išnaudoja automatizuotų paspaudimų sistemas pirkimų ir statymų atlikimui. Autorių atliktame tyrime aprašomas automatizuotų paspaudimų sistemų efektyvumas ir jų trūkumai marketinge. Sistemos efektyvumas pasižymi galimybe priimant sprendimus reaguoti greičiau nei žmogus ir galimybe sprendimus priiminėti visą parą. Kadangi šis tyrimas analizuoja, kaip užtikrinti saugą mobiliuosiuose žaidimuose, aptariamas tik automatizuotų paspaudimų sistemų atpažinimas ir apsauga nuo trečiųjų šalių programų. Apsauga veikia, patikrindama kiekvieno paspaudimo laiko intervalą ir tikslumą.

Projekto problema. Automatizuotų paspaudimų sistemos neturi algoritmo sudarymo funkcijų lanksčiai, patikimai ir patogiai organizuoti pasikartojantiems veiksmams. Šis funkcionalumas reikalingas tam, kad sistema būtų naudojama sudėtingoms marketingo sistemoms, mobiliesiems žaidimams ar net platesnių organizacijos procesų automatizavimui.

Projekto tikslas: sukurti paspaudimų, braukimų ir teksto rašymo imitavimo sistemą, suteikiančią galimybę atlikti pasikartojančius veiksmus, susikurti ir išsisaugoti veiksmų vykdymo algoritmą.

Projekto uždaviniai: atlikti sistemos poreikio analizę; atlikti rinkoje esančių panašių sistemų analizę; parinkti projektui realizuoti reikalingus programinius įrankius; suprojektuoti ir realizuoti automatizuotų veiksmų sistemą.

Projekto reikšmingumas ir sklaida. Automatizuotų veiksmų sistema suteikia galimybę asmeniui, nemokančiam programuoti, automatizuoti savo darbą. Tokio tipo sistema suteikia tikslumo ir pranašumo marketinge, įgalindama stebėti pokyčius kompiuteriu visą parą ir priimdama greitą sprendimą pagal sukurtas instrukcijas. Kadangi automatizuotų veiksmų sistema priklauso trečiųjų šalių programų kategorijai, todėl, plečiant sistemos funkcijas, daugėja pritaikymo sričių. Automatizuotų veiksmų sistemos ypatingumas pasireiškia universalumu dirbant su skirtingomis sistemomis, todėl nėra galimybės įvardinti praplėstos sistemos paskirties variantų skaičiaus.

Kuriamos sistemos poreikis ir nauda

Automatizuotos veiksmų sistemos yra plataus spektro programinės įrangos ir technologijų rinkinys, kuriuo siekiama automatizuoti pasikartojančius procesus, kuriuos atlieka darbuotojai. Šios sistemos gali apimti skirtingas veiklos sritis, pvz., gamybą, sandėliavimą, transportavimą, finansus ir kt., dažnai tokios sistemos pritaikomos žaidimuose.

Skyriuje nagrinėjamos pelės paspaudimų, braukimų ir teksto rašymo imitavimo sistemos.

Vienas iš svarbiausių priežasčių, kodėl kyla automatizuotų sistemų poreikis, yra noras padidinti darbo našumą. Skaitmeninėje aplinkoje daugybė užduočių gali būti monotoniškos, pvz., paspaudimų arba pakeitimų atlikimas įvairiose svetainėse, reklaminių kampanijų valdymas, sąskaitų pajamavimas, informacijos kopijavimas ir kt. Pelės paspaudimų automatizavimo sistemos leidžia

efektyviai atlikti tokius uždavinius, užtikrinant, kad proceso vykdymas būtų greitas, tikslus ir nuoseklus.

Minėtame Shing Ki ir Siu Min Yiu tyrime [4] išskiriama sistemų galimybė reaguoti greičiau nei žmogus, priimant sprendimus, ir galimybė sprendimus priiminėti visą parą. Kita automatizuotų veiksmų sistemų nauda gali būti siejama su darbo efektyvumo pagerinimu. Tokių sistemų įdiegimas leidžia optimizuoti veiklą, t. y. pasikartojančius veiksmus atlikti greičiau.

Dar vienas svarbus automatizuotų veiksmų sistemos poreikio aspektas yra tikslumas ir patikimumas. Rankiniu būdu atliekant pelės paspaudimus gali kilti žmogiškojo veiksnio klaidų. Automatizuotos sistemos yra programuojamos taip, kad veiktų pagal konkretų scenarijų ar algoritmą, užtikrinant, kad paspaudimai būtų atliekami tiksliai ir pasikartojamai. Tai leidžia išvengti nereikalingų klaidų ir sumažinti galimus rizikos veiksnius, padidinant veiklos patikimumą ir kokybę.

Apibendrinant, pelės paspaudimų automatizavimo sistemų poreikio analizę, galima išskirti tokias automatizuotų veiksmų sistemų naudas:

- Darbo efektyvumo bei darbo našumo padidėjimas;
- Mažesnis klaidų kiekis (kalbant apie žmogiškojo veiksnio klaidas);
- Pasikartojančių veiksmų optimizavimas, veiklos patikimumo ir kokybės užtikrinimas.

Automatinių pelės paspaudimų sistemų rinkos apžvalga

Straipsnyje analizuojamos populiariausios rinkoje sistemos: „Auto Clicker app for games“ [6], „QuickTouch - Automatic Clicker“ [7], „Auto Clicker - Automatic tap“ [8], „Click Assistant - Auto Clicker“ [9]. Populiarumas nustatomas naudojant Google Play platformos atsisiuntimo duomenis. Alternatyvi sukurta sistema „Auto Clicker - Automatic tap“ [8] turi daugiau nei 50 mln. atsisiuntimų ir yra pati populiariausia Google Play platformoje, straipsnyje vertinama versija 1.6.3. Konkuruojanti analoginė sistema „Click Assistant - Auto Clicker“ [9] turi daugiau nei 10 mln. atsisiuntimų, čia vertinama sistemos versija 1.16.9. Analoginė sistema „QuickTouch - Automatic Clicker“ [7] turi daugiau nei 10 mln. atsisiuntimų Google Play platformoje, čia vertinama versija 4.8.11. Sistema „Auto Clicker app for games“ [6] turi daugiau nei 1 mln. atsisiuntimų, čia vertinama versija 2.4.4.

Išvardintos sistemos vertinamos pagal ISO-9126 [10] kokybės standartą, kuriame pateiktos 6 pagrindinės kokybės charakteristikos: patikimumas, funkcionalumas, naudojamumas, palaikomumas, našumas, perkeliamumas. Daugiausia dėmesio skiriama šioms charakteristikoms: naudojamumas, našumas ir funkcionalumas.

Automatizuotų paspaudimų sistemos našumas svarbus, siekiant neapkrauti operacinės sistemos. Kadangi sistema skirta sąveikauti su kitomis sistemomis, sunku nustatyti tinkamus konkrečius minimalius spartos limitus. Kuo trečiųjų šalių programos mažiau apkrauna prietaisą, tuo geriau.

Naudojamumas vertinamas, naudojant kuo mažiau paspaudimų iki pagrindinio meniu. Svarbu, kad pagrindinis meniu turėtų mygtukus su neklaidinančiomis piktogramomis, kurios atitiktų atliekamus veiksmus. Jei mygtukas naudos tekstą, svarbu, kad būtų kuo trumpiau aprašytas veiksmas.

Funkcionalumas yra išskirtinai svarbus kriterijus, nes sistema priskiriama trečiųjų šalių programoms. Trečiųjų šalių programos atlieka veiksmus su kita programa, todėl nuo funkcionalumo galimybių priklausys, kokius darbus naudotojas turės galimybę įvykdyti su analizuojama sistema.

Atliktas sistemų vertinimas pagal tokius kriterijus:

- ar sistema suteikia galimybę redaguoti automatizuoto paspaudimo poziciją, tikslumą, paklaidos dydį ir vykdymo laiką;
- ar yra atpažinimo sistemos: optinio simbolio atpažinimo, spalvos atpažinimo, nuotraukų atpažinimo sistemos;
- ar suteikiama galimybė sukurti automatizuotą teksto rašymą, jo redagavimą, keičiant aktyvavimo laiką ir rašomą tekstą;
- ar yra sukurtas automatizuoto braukimo veiksmas ir ar galima jį redaguoti, keičiant vykdymo laiką, braukimo taškų skaičių ir kiekvieno taško braukimo laiką, tikslumą, paklaidos dydį;
- algoritmavimo funkcionalumas bei sistemos galimybė keisti pasikartojamų veiksmų skaičių, ar yra galimybė išsaugoti visus veiksmus, jungti skirtingus algoritmus.

1 lentelė

Analoginių sistemų bendras vertinimas

Sistema / **Kriterijus**	Auto Clicker app for games [6]	QuickTouch - Automatic Clicker [7]	Auto Clicker - Automatic tap [8]	Click Assistant - Auto Clicker [9]
Funkcionalumas				
Yra galimybė redaguoti automatizuoto paspaudimo paklaidą	-	+	-	+
Yra galimybė kurti automatizuotą teksto rašymą	-	-	-	-
Yra galimybė kurti atpažinimo sistemas	+	-	-	-
Yra galimybė visapusiškai redaguoti automatizuotą braukimą	-	-	-	-
Yra galimybė dalintis sukurtomis automatizuotų veiksmų instrukcijomis	-	-	-	-
Naudojamumas				
Naudojamas neklaidinantis dizainas	+	+	+	+
Našumas				
Sistema naši (2 sparčiausios sistemos)	-	-	+	+
Iš viso:	**2**	**2**	**2**	**3**

Sudaryta straipsnio autorių

Nėra duomenų, kad būtų galima įvertinti nagrinėjamų sistemų patikimumo, palaikomumo ir perkeliamumo charakteristikų, kurios pateiktos kaip svarbios ISO/IEC 9126 [10] kokybės standarte.

Pateiktoje 1 lentelėje matoma, kad nei viena iš nagrinėtų sistemų neturi galimybės kurti automatizuotą teksto rašymą, galimybės visapusiškai redaguoti automatizuotą braukimą ir galimybės dalintis sukurtomis automatizuotų veiksmų instrukcijomis. Nagrinėjamos sistemos tenkina ne daugiau kaip 2 - 3 kriterijus (iš išvardintų 7). Pagal gautus rezultatus galima teigti, kad sistema „Click Assistant - Auto Clicker“ yra pranašiausia.

Įrankiai projekto realizavimui

freeCodeCamp straipsnyje [11] teigiama, kad kompiliuojamos kalbos yra spartesnės už interpretuojamas. Norint atitikti ISO/IEC 9126 [10] standarto našumo charakteristikos modelio reikalavimus, svarbu kurti sistemą su programavimo kalba, kurios kodas bus kompiliuojamas, o ne interpretuojamas. Todėl programavimo kalbos, tokios kaip PHP, Ruby, Python ir JavaScript, netiks sistemos kūrimui.

Norint suteikti sistemai funkcionalumą, reikia turėti prieigą prie atviro kodo teksto atpažinimo bibliotekos. Atviro kodo ir nemokamų teksto atpažinimo bibliotekų pasirinkimas yra ribotas. Biblioteka Tesseract yra sukurta Google naudojant C++ programavimo kalbą. Programavimo kalbos C#, Go ir Rust turi galimybę naudotis tik savanorių perkurta Tesseract biblioteka, o tai neužtikrina patikimumo.

Vertinama ir programuotojo galimybė valdyti įrenginio atmintį. Kuriant sistemą, manipuliuojant duomenimis, svarbu kopijuoti kuo mažiau informacijos. Visos analizuojamos kalbos suteikia galimybę perduoti ir manipuliuoti duomenis nuorodomis.

Įvertinus išvardintus privalumus, sistemai kurti pasirinkta *C++* programavimo kalba, bei *Qt* karkasas.

Dažnai reikalinga atpažinti ne tik spausdintą tekstą, bet ir grafinę informaciją. Tačiau rinkoje yra mažas pasirinkimas kompiuterinio matymo realiuoju laiku bibliotekų, kurios suteiktų platų nemokamų technologijų pasirinkimą nuotraukų lyginimui. Tinkamiausia ir patikimiausia nuotraukų lyginimui biblioteka yra OpenCV. Kadangi biblioteka OpenCV suteikia galimybę integruoti perceptual hash, histogramų lyginimo, Structural Similarity Index, Mean Squared Error, Feature-based comparison, normalized cross correlation nuotraukų lyginimo metodus, todėl kuriamoje sistemoje realizuojamas įvairių metodų pasirinkimas lyginant nuotraukas. Kuriama sistema,

naudodama OpenCV biblioteką, integruoja perceptual hash algoritmą, skirtą nuotraukų lyginimui, kuris išgautas hash unikalias skaičių sekas palygina tarpusavyje, siekiant nustatyti jų panašumus.

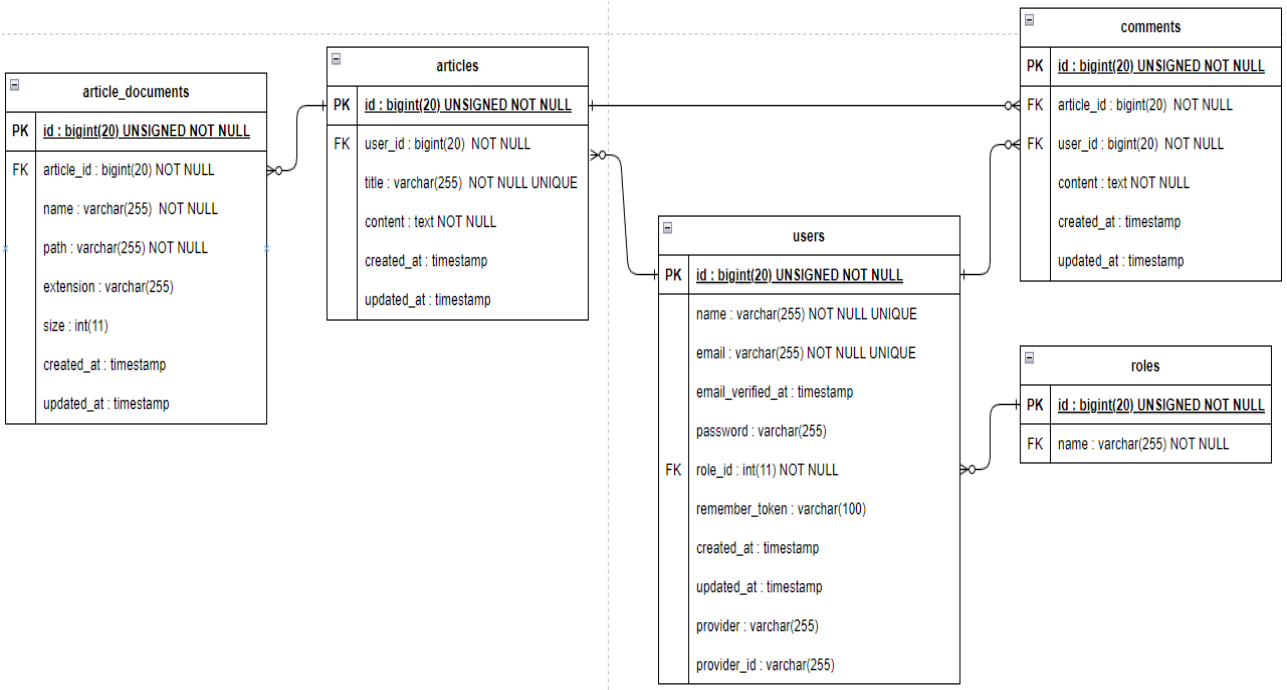
Automatizuotų veiksmų objektų dokumentai, suprogramuoti automatizuotų veiksmų sistemos, bus saugomi JSON formatu, remiantis ISO-9126 [10] standarto charakteristikos modeliais. QT karkasas turi integruotą C++ programavimo kalbos biblioteką, suteikiančią galimybę saugoti programavimo objektus JSON formatu. Nuotraukos bus paverčiamos BASE64 formatu, suteikiančiu galimybę baitus saugoti kaip tekstą, prieš išsaugant į JSON dokumentą.

Sistemos realizacija

Kuriamas projektas turės skirtingus vartotojus, su priskirtomis skirtingomis teisėmis. Kuriamas forumas, kuriame vartotojai galės kurti, redaguoti ir trinti straipsnius, komentarus, straipsnio dokumentus, dalintis sukurtomis automatizuotų veiksmų instrukcijomis. Projekto duomenys bus pasiekiami naudojant *HTTP* užklausas. Visiems minėtiems duomenims saugoti reikalinga duomenų bazė. Bus naudojama MySQL duomenų bazių valdymo sistema.

Duomenims saugoti sukurtos 5 lentelės: naudotojai (*users*), komentarai (*comments*), straipsniai (*articles*), straipsnių dokumentai (*article_documents*), rolės (*roles*). Duomenų bazės schema pateikta 1 pav.

Duomenų bazėje saugomas slaptažodis turi galimybę turėti ir null reikšmę, nes ateityje bus planuojama integruoti galimybę prisijungti su tiekėjo (Google) duomenimis, todėl kuriami *provider* ir *provider_id* laukeliai. Naudotojų (*users*) lentelėje saugomas *remember_token* laukelis skirtas išsaugoti naudotojo prieigos raktui ir suteikti galimybę prisijungti prie sistemos, nesuvedus naudotojo slapyvardžio/elektroninio pašto ir slaptažodžio. Duomenų bazės lentelėse *users*, *comments*, *articles* ir *article_documents* bus saugomi įrašų sukūrimo (*created_at*) ir atnaujinimo (*updated_at*) duomenys. Lentelės *roles* sukūrimo ir atnaujinimo duomenys nesaugomi, nes duomenų sukūrimas vyks tik serveryje.



1 pav. Duomenų bazės struktūra

Sudaryta straipsnio autorių

Projektui realizuoti reikalingos sistemos: „NodeJS“, „NPM“, „Composer“, kurios suteikia galimybę kurti Laravel karkaso projektus ir paketus naujausia versija. Pradėjus kurti sistemą, sukuriama „MySQL“ duomenų bazė. Naudojant Laravel karkaso dokumentaciją, įrašomas passport paketas, suteikiantis galimybę limituoti kreipimąsi į kuriamą API ir atšaukti naudotojo suteiktą prieigos raktą prieš pasibaigiant galioti.

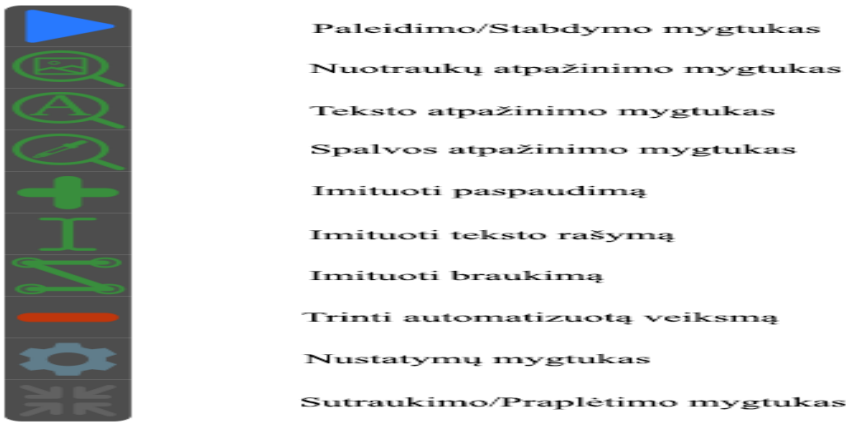
Įrašius reikalingą įrangą serverio sistemos kūrimui, sukuriamos migracijos, skirtos įrašyti lenteles į duomenų bazę. Sistemai valdyti duomenis sukuriami duomenų modeliai ir kontroleriai, skirti valdyti veiksmus su duomenimis. Sukuriami duomenų ryšiai duomenų modeliuose, kuriamos duomenų bazių rolės, naudojant Laravel seeder ir factory klases. Pripildžius duomenų roles ir sukūrus ryšius, kuriami laikini testavimo duomenys. Sukurtos duomenų validavimo klasės ir atsakymai skirti API užklausoms.

Realizavus API užklausų atsakymus, kuriama kliento pusės dalis. Įdiegiamas Qt karkasas, vcpkg paketų įrašymo ir CMake įdiegimo įrankiai. Parsisiunčiama tesseract optinio teksto atpažinimo biblioteka bei OpenCV biblioteka, sugeneruojami bibliotekos dokumentai.

Sukuriamas Qt projektas, naudojantis Qt karkaso įrankiais, kuriami skirtingi rolių meniu. Naudojantis Qt karkaso qss kodavimu, kuriamas programos dizainas.

Pagrindinio meniu spalvos parenkamos, naudojantis colormind svetainės palečių generatoriumi.

Sukurtoje sistemoje numatytos trys skirtingos naudotojų rolės: svečias (*Guest*), narys (*Member*), vip narys (*Vip*). Svečias neturi atpažinimo sistemų kūrimo mygtukų, paprastas narys turi svečio mygtukus ir teksto bei spalvos atpažinimo kūrimo mygtukus, o vip narys turi galimybę kurti visus automatizuotus veiksmus. Nagrinėjamas tik vip teisėmis sukurtas automatizuotų veiksmų meniu, nes tik vip nariams suteikiamos visos teises. Valdymo meniu su mygtukų paaiškinimu pateiktas 2 pav.



2 pav. Automatizuotų veiksmų meniu mygtukai

Sudaryta straipsnio autorių

Automatizuotų veiksmų sistemoje yra sukurta: nuotraukų, teksto ir spalvos atpažinimų sistemos, braukimo, teksto rašymo ir paspaudimo imitavimas, automatizuoto veiksmo trynimas, detalūs sistemos nustatymai. Naudojant sukurtas sistemos funkcijas yra galimybė kurti detalų veiksmų algoritmą, jį išsaugoti ir dalintis su kitais json formatu. Automatizuotų veiksmų sistemoje yra galimybė nupiešti nuotraukų ir teksto atpažinimo lauką. Paspaudus nuotraukų ar teksto atpažinimo mygtuką, kompiuterio ekranuose pateikiamas vaizdas naudoja alpha kanalą, kuris atvaizduoja vaizdą pilkai. Pilkas ekrano vaizdas indikuoja, kad atsirado galimybė piešti atpažinimo lauką. Nupiešus norimo dydžio stačiakampį, sukuriamas pasirinktos atpažinimo sistemos objektas. Nuotraukų atpažinimo laukas suteikia galimybę pasirinkti nuotraukų lyginimo metodą. Sistemoje yra sukurta 6 nuotraukų lyginimo metodai: perceptual hash, histogramų lyginimas, Structural Similarity Index, Mean Squared Error, Feature-based comparison, normalized cross correlation.

Norint pritaikyti automatizuotų veiksmų sistemą tam tikrose srityse, pirmiausia reikia sukurti veiksmų vykdymo algoritmą pagal tai, ką norima atlikti, t. y. įrašyti kiekvieną pelės paspaudimą: reikiamo failo atidarymą, reikalingo mygtuko paspaudimą, pažymėti reikalingą vietą, reikalingo teksto nuskaitymą, tikrinti jo atitikimą nurodytiems kriterijams, jo perkėlimą į kitą vietą (programą), paveikslėlio atpažinimą, nurodyti, kaip elgtis, radus vienokį ar kitokį atitikimą (rašyti sąlygos sakinius). Paleidus sistemą, ji vykdys iš anksto nurodyto algoritmo žingsnių seką. Pasikeitus bent vienam žingsniui, visą algoritmą teks susikurti iš naujo.

Automatizuotų veiksmų sistemos darbo metu vykdomų atpažinimo sistemų rezultatai atvaizduojami atpažinimo sistemos nustatymuose. Naudotojas turi galimybę testuoti parašytą algoritmą realiu laiku.

Sistemoje yra galimybė įkelti sukurtas automatizuotų veiksmų instrukcijas į sukurtą forumo straipsnį json formatu. Skaitytojai turi galimybę parsisiųsti sukurtas automatizuotų veiksmų instrukcijas ir naudoti su automatizuotų veiksmų sistema.

Sistemos sauga

Sistema naudoja Qt karkaso QSslConfiguration klasę, kuri suteikia galimybę siųsti paketus į serverį, naudojant SSL/TLS protokolą. Pasirinktas serverio tiekėjas suteikia SSL sertifikatus nemokamai. Saugumui pasiekti siunčiami ir grįžtami paketai perduodami per SSL/TLS protokolą.

Serverio aplikacijai kurti naudojamas Laravel karkasas, kuris naudoja Eloquent (ORM) duomenų valdymą. Eloquent objekto-reliacijos atvaizdavimo modelis apsaugo nuo SQL injection puolimų. Naudojama Laravel karkaso duomenų validacija, kuri patikrina gaunamus naudotojo duomenis. Registracijos metu, naudojant Laravel validaciją, tikrinama, ar įvestas elektroninis paštas turi taisyklingą pašto adresą.

Į serverį patenkantys duomenys yra validuojami naudotojo sistemoje ir serveryje. Tokiu būdu yra sumažinamas užklausų kiekis serveriui. Mažesnė serverio apkrova spartina sistemos darbą.

Serveryje saugomi naudotojų slaptažodžiai yra koduojami, naudojant Laravel karkaso bcrypt funkciją, kuri sukuria unikalią skaičių koduotę slaptažodžiui. Nėra galimybės atkoduoti unikalią skaičių koduotę, todėl saugomas slaptažodis yra neatpažįstamas, nors ir neteisėtai gavus naudotojų duomenis.

Siekiant užtikrinti serverio duomenų apsaugą, naudojama Node.js platformos „npm audit“ komanda, kuri patikrina sistemos paketų suderinamumus ir pateikia paketų atrastus pažeidžiamumus. Komanda vykdoma kas mėnesį, siekiant užtikrinti, kad neatsirastų naujų pažeidimų senuose paketuose.

Siekiant apsaugoti naudotojų paskyras nuo brute force puolimo, reikalaujama, kad naudotojų slaptažodis turėtų bent vieną didžiąją, mažąją raidę, skaičių ir specialų simbolį, slaptažodis turi būti ne trumpesnis nei 6 simboliai.

Qt karkasas suteikia galimybę skaityti ir rašyti json dokumentus taisyklingai, todėl saugomi automatizuoti objektai yra taisyklingos struktūros. Qt karkaso json dokumentų skaitymo įrankis, skaitydamas klaidingai suformuotą dokumentą, pateikia klaidą (exception).

Programuojant ir projektuojant automatizuotų veiksmų sistemą, yra laikomasi D.R.Y (Don‘t repeat yourself) principo, kuris užtikrina, kad rašomas kodas yra nesikartojantis. Nenaudojant D.R.Y principo, atsiranda galimybė, kad sistemoje bus pataisomas kodas ne visose vietose, kur tikimasi tokio pat funkcionalumo.

Valdant atmintį su C++ programavimo kalba, svarbu įsitikinti, kad programa neturi atminties nutekėjimo (memory leak) pažeidžiamumo. Prižiūrėti kompiuterio atmintį aplikacijoje naudojamos shared_ptr ir unique_ptr klasės, skirtos valdyti atminties prieigą ir išvengti atminties nutekėjimo pažeidžiamumo.

Sistemos taikymo pavyzdžiai

Sukurta sistema suteikia galimybę atpažinti nuskanuotą sąskaitų informaciją, naudojant automatizuotą teksto atpažinimą, ir atpažintą informaciją perkelti į kitas sistemas, naudojant paspaudimų ir teksto rašymo sukurtas imitacijas. Tokiu būdu yra automatizuojamas kas mėnesį sugeneruojamų sąskaitų surašymas.

Internetinis žaidimas gladiatus.lt turi aukcioną, kuriame yra išrašomas aukciono laikas žodžiais. Pasibaigus aukciono statymo laikui, paskutinis žaidėjas, pasiūlęs didžiausią kainą daiktui, laimi jį. Aukciono pabaigai indikuoti parašomas laikas „labai trumpai“, tačiau šis terminas gali kisti nuo 30 min. iki 2 val. Norint laimėti, žaidėjas turi kuo dažniau perkrauti puslapį ir tikrinti, ar kitas žaidėjas nepasiūlė didesnės kainos. Automatizuotų veiksmų sistema suteikia galimybę sukurti algoritmą, kuris perkrautų puslapį paspaudimais ir, naudojant teksto atpažinimą, tikrintų, ar pasiūlyta žaidėjo kaina buvo padidinta. Yra galimybė sukurti sąlygą atpažintam tekstui ir aptiktam kainos padidinimui pasiūlyti dar didesnę kainą.

Automatizuotų veiksmų sistema suteikia galimybę ištestuoti kitas sistemas, sukuriant paspaudimų ir teksto įvedimo instrukcijas. Naudojant pasirinktą atpažinimo sistemą, aptinkamas teksto rezultatas, kurį galima laikinai išsaugoti kompiuterio atmintyje. Naudojant paspaudimų ir teksto rašymo imitacijas, galima patalpinti gautą rezultatą į atskirą dokumentą.

Išvados

Straipsnyje išnagrinėta automatizuotų veiksmų sistema, kurios tikslas yra suteikti galimybę naudotojams kurti ir vykdyti pasikartojančius veiksmus kompiuteryje. Analizė atskleidė, kad yra didelė tokių sistemų paklausa, nes jos gali žymiai padidinti darbo našumą, sumažinti klaidų kiekį ir suteikti galimybę automatizuoti įvairias užduotis. Automatizuotų veiksmų sistemos yra aktualios ir naudingos įvairiose veiklos srityse, įskaitant marketingą, žaidimus, ir kitas kompiuterinio darbo sritis.

Nagrinėtos rinkoje siūlomų automatizuotų sistemų galimybės ir trūkumai, pateikta išsami jų analizė. Atskleista, kad dauguma esamų sistemų turi ribotas galimybes, ypač jei kalbama apie tekstų rašymo ar grafinio atpažinimo funkcijas.

Projekto realizavimo metu pasirinktos technologijos, tokios kaip *C++* programavimo kalba ir *Qt* karkasas, tekstų ir grafikos elementų nuskaitymui naudojamos *Tesseract* ir *OpenCV* bibliotekos suteikia galimybę sukurti lanksčią ir efektyvią automatizuotų veiksmų sistemą.

Sukurtoje sistemoje yra galimybė kurti atpažinimo, braukimo, teksto rašymo ir paspaudimo imitavimo funkcijas, sudaryti detalų veiksmų algoritmą. Tai suteikia naudotojams galimybę sukurti sudėtingus veiksmų algoritmus pagal savo poreikius.

Sistemoje taip pat yra galimybė dalintis sukurtomis automatizuotų veiksmų instrukcijomis su kitais naudotojais, tai leidžia bendruomenei taupyti laiką, keičiantis jau sukurtomis instrukcijomis ir tobulinti automatizuotų veiksmų algoritmus.

Sistemos sauga užtikrinama naudojant SSL/TLS protokolą, užtikrinant saugaus slaptažodžio reikalavimo laikymąsi bei jį koduojant.

Literatūra

1. RIJN, J. *The Ultimate Marketing Automation statistics overview*. 2023. https://www.emailmonday.com/marketing-automation-statistics-overview/#ftoc-heading-1
2. Binance API. Unlimited Opportunities with One Key. 2023. https://www.binance.com/en/binance-api
3. Developers love Polygon. Intuitive & Easy to Use APIs. 2023. https://polygon.io/stocks
4. WONG, Shing Ki; YIU, Siu-Ming. Detection on auto clickers in mobile games. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 2019, 10(3): 65-80. DOI:10.22667/JOWUA.2019.09.30.045
5. CLEMENT, J. Mobile gaming app revenue worldwide from 2019 to 2022. *Statista*, 2023. https://www.statista.com/statistics/511639/global-mobile-game-app-revenue/
6. Auto Clicker app for games. *Huau Apps*, 2023. https://play.google.com/store/apps/details?id=com.ksxkq.autoclick
7. QuickTouch – Automatic Clicker. *SimpleHat Software, LLC*. 2023. https://play.google.com/store/apps/details?id=simplehat.clicker
8. Auto Clicker – Automatic tap. *True Developers Studio*, 2023. https://play.google.com/store/apps/details?id=com.truedevelopersstudio.automatictap.autoclicker&hl=en&gl=US
9. Click Assistant - Auto Clicker. *Y.C. Studio*, 2023. https://play.google.com/store/apps/details?id=com.rise.automatic.autoclicker.clicker
10. ISO/IEC 9126 in Software Engineering. *itskawal2000*, 2023. https://www.geeksforgeeks.org/iso-iec-9126-in-software-engineering/
11. Interpreted vs Compiled Programming Languages: What's the Difference? *freeCodeCamp*, 2023. https://www.freecodecamp.org/news/compiled-versus-interpreted-languages/
12. ISO/IEC 21778:2017. *ISO*, 2023. https://www.iso.org/standard/71616.html