Igrifikacija u učenju programiranja kroz online kvizove

Antunović, Renato

SVEUČILIŠTE U SPLITU

PRIRODOSLOVNO-MATEMATIČKI FAKULTET

DIPLOMSKI RAD

IGRIFIKACIJA U UČENJU PROGRAMIRANJA KROZ ONLINE KVIZOVE

Mentor: prof. dr. sc. Ani Grubišić

Student: Renato Antunović

Split, rujan 2024.

Temeljna dokumentacijska kartica

Diplomski rad

Sveučilište u Splitu
Prirodoslovno-matematički fakultet
Odjel za Informatiku
Ruđera Boškovića 33, 21000 Split, Hrvatska

IGRIFIKACIJA U UČENJU PROGRAMIRANJA KROZ ONLINE KVIZOVE

Renato Antunović

SAŽETAK

Cilj ovog diplomskog rada bio je istražiti koncept igrifikacije i njezinu primjenu u obrazovanju, kao i u drugim područjima. Rad obuhvaća pregled i usporedbu nekih od najpopularnijih alata za izradu online kvizova, ističući njihove karakteristike i funkcionalnosti. Ključni dio rada odnosi se na razvoj i izradu web aplikacije namijenjene učenju programiranja putem online kvizova, koju mogu koristiti i nastavnici i učenici. U radu su detaljno opisane korištene tehnologije te postupak implementacije aplikacije, s naglaskom na njezinu funkcionalnost i prilagodljivost obrazovnim potrebama.

Ključne riječi: igrifikacija, e-učenje, kvizovi

Rad je pohranjen u knjižnici Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Rad sadrži: 50 stranica, 28 grafičkih prikaza, 1 tablicu i 14 literaturnih navoda.

Izvornik je na hrvatskom jeziku.

Mentor: **prof. dr. sc. Ani Grubišić**, *redoviti profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Ocjenjivači: **prof. dr. sc. Ani Grubišić**, *redoviti profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

prof. dr. sc. Branko Žitko, *redoviti profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Ines Šarić-Grgić, dipl. ing., *asistent Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Rad prihvaćen: **rujan 2024.**

Basic documentation card

Master thesis

University of Split
Faculty of Science
Department of Computer Science
Ruđera Boškovića 33, 21000 Split, Croatia

GAMIFICATION IN LEARNING PROGRAMMING THROUGH ONLINE QUIZZES

Renato Antunović

ABSTRACT

The objective of this thesis was to explore the concept of gamification and its application in education, as well as in other fields. The paper provides a review and comparison of some of the most popular tools for creating online quizzes, emphasizing their features and functionalities. The core part of the thesis focuses on the development and design of a web application intended for programming education through online quizzes, suitable for both teachers and students. The thesis thoroughly describes the technologies used and the process of implementing the application, with a focus on its functionality and adaptability to educational needs.

Keywords: gamification, e-learning, quizzes

Thesis consists of:	50 pages, 28 figures, 1 table and 14 references. Original language: Croatian
Mentor:	**Ani Grubišić, PhD**, *Full Professor, Faculty of Science, University of Split*
Reviewers:	**Ani Grubišić, PhD**, *Full Professor, Faculty of Science, University of Split*
	Branko Žitko, PhD, *Full Professor, Faculty of Science, University of Split*
	Ines Šarić-Grgić, MSc, *Assistant, Faculty of Science, University of Split*

Thesis accepted: **September 2024.**

Ovim putem želim izraziti iskrenu zahvalnost svojoj mentorici, prof. dr. sc. Ani Grubišić, na njenim stručnim savjetima, pomoći i podršci tijekom cijelog procesa izrade ovog rada.

Također zahvaljujem svojoj obitelji i prijateljima koji su mi pružali neizmjernu potporu i motivaciju i bez kojih bi ovo sve bilo mnogo teže.

Posebnu zahvalnost dugujem svojoj djevojci, Katarini, koja je bila uz mene tijekom svakog trenutka pisanja ovog rada. Ti najbolje znaš kroz što sam sve prolazio kako bih ga dovršio, i tvoje strpljenje, razumijevanje i podrška bili su mi neprocjenjivi.

Na kraju, ovaj rad posvećujem svojim roditeljima, koji su mi oduvijek pružali podršku, ne samo u obrazovanju, već i kroz sve životne izazove. Vaša vjera u mene i vaše neprestano ohrabrenje omogućili su mi da postignem ono što sam danas, i za to vam neizmjerno hvala.

SADRŽAJ

Uvod 1

1. Teorijski okvir 2

1.1 Igrifikacija u obrazovanju 3

1.2 Igrifikacija u poučavanju programiranja 4

2. Online alati za izradu kvizova za učenje 7

2.1 Kahoot! 7

2.2 Quizlet 9

2.3 Socrative 10

2.4 Top Hat 12

2.5 Sažetak alata za izradu online kvizova 14

3. Oblikovanje aplikacije 16

3.1 Odabir tehnologija za razvoj aplikacije 16

3.1.1 Flask 16

3.1.2 SocketIO 16

3.1.3 TailwindCSS 17

3.1.4 HTML, CSS, JavaScript 17

3.1.5 Jinja2 17

3.1.6 SQLAlchemy 18

3.1.7 Alembic 18

3.1.8 WTForms 18

3.2 Modeliranje zahtjeva aplikacije 19

3.3 Modeliranje tijeka aplikacije 21

4. Implementacija aplikacije 22

4.1 Struktura projekta 22

4.2 Baza podataka 25

4.2.1 Model korisnika ... 25
4.2.2 Model ispita ... 26
4.2.3 Model pitanja ... 27
4.2.4 Model bodovanja ... 28
4.3 Registracija korisnika ... 29
4.4 Prijava korisnika ... 31
4.5 Nadzorna ploča ... 32
4.5.1 Prikaz ispita ... 32
4.5.2 Stvaranje novog ispita i unos pitanja ... 33
4.5.3 Uređivanje ispita ... 35
4.5.4 Brisanje ispita ... 36
4.6 Ispitna čekaonica ... 37
4.7 Pristupanje ispitu ... 38
4.8 Pokretanje ispita ... 40
4.9 Rješavanje ispita ... 41
4.10 Kraj ispita ... 43
4.11 Prikaz konačnih rezultata ispita ... 44
4.12 Dvojezična podrška ... 45
4.13 Odjava korisnika ... 47
Zaključak ... 48
Literatura ... 49

Uvod

Obrazovni sustavi oduvijek su nastojali pronaći inovativne načine kako bi unaprijedili proces poučavanja i motivirali učenike. S razvojem tehnologije, suvremeni pristupi obrazovanju uključuju upotrebu digitalnih alata koji omogućuju veću interaktivnost, fleksibilnost i prilagodljivost nastavnim procesima. U posljednjih nekoliko godina, sve veći broj nastavnika počinje koristiti takve digitalne alate koji omogućuju provođenje kvizova i testova na online platformama. Ovi alati prilagođeni su različitim obrazovnim okruženjima te omogućuju nastavnicima kreiranje dinamičnih i prilagodljivih zadataka. Time se učenicima omogućuje aktivno sudjelovanje u procesu učenja, dok nastavnici dobivaju brze povratne informacije o njihovom napretku. Unatoč postojanju brojnih alata koji pružaju ove mogućnosti, svaka obrazovna institucija i nastavnik suočavaju se s izazovom pronalaska rješenja koje najbolje odgovara njihovim potrebama.

Glavni fokus ovog diplomskog rada bio je osmisliti, isplanirati i izraditi web aplikaciju koja podržava učenje programiranja putem online kvizova. Cilj aplikacije je omogućiti nastavnicima kreiranje i upravljanje ispitima te pružiti učenicima jednostavan način za sudjelovanje u kvizovima. Aplikacija je dizajnirana tako da bude fleksibilna i jednostavna za korištenje, omogućujući korisnicima prilagodbu sučelja na hrvatski ili engleski jezik, te nudi interaktivno iskustvo kroz ažuriranje rezultata u stvarnom vremenu.

Struktura rada podijeljena je u nekoliko poglavlja. Prvo poglavlje bavi se teorijskim okvirom i definicijom pojma igrifikacije, uključujući njezinu primjenu u obrazovanju. Zatim slijedi pregled postojećih online alata za izradu kvizova, gdje su prikazane njihove ključne karakteristike i načini upotrebe. Nakon toga, detaljno je opisan proces razvoja aplikacije, uključujući odabir tehnologija, modeliranje zahtjeva te implementaciju ključnih funkcionalnosti. Na kraju rada, dani su zaključci o korištenju igrifikacije u obrazovanju i o mogućnostima daljnjeg razvoja i primjene aplikacije.

1. Teorijski okvir

Igrifikacija (engl. *gamification*), kao pristup koji koristi elemente igara u neigračkim kontekstima, posljednjih je godina postala iznimno popularan alat za poboljšanje angažmana korisnika. Temelji ovog koncepta leže u mogućnosti da se izazovi, nagrade i mehanizmi povratnih informacija iz igara primijene na sustave koji nisu izvorno razvijeni kao igre, s ciljem povećanja interakcije i sudjelovanja korisnika. Igrifikacija omogućuje strukturiranje svakodnevnih aktivnosti na način da postanu zabavnije, motivirajuće i nagrađujuće, često se oslanjajući na psihološke principe povezane s igračkim iskustvima [1][2]. S vremenom, igrifikacija se razvila u moćan alat ne samo u kontekstu zabave, već i u mnogim industrijama, uključujući zdravstvo, obrazovanje, marketing i radna okruženja.

Povijest igrifikacije seže u šezdesete godine prošlog stoljeća, kada su teorije ponašanja kao što je token-ekonomija prvi put primijenjene u zatvorenim sustavima, poput psihijatrijskih bolnica, kako bi se promijenilo ponašanje pacijenata [1]. Ti rani eksperimenti pokazali su da sustavi bodova i nagrada mogu snažno utjecati na ponašanje pojedinaca, stvarajući temelj za kasniju primjenu tih principa u digitalnim igrama i drugim kontekstima. Međutim, jedan od glavnih izazova u razvoju igrifikacije bio je njeno preveliko oslanjanje na sustave bodovanja, što je dovelo do kritike te pojave, u engleskom jeziku poznatog kao *pointsification* – trenda koji pretvara svaki aspekt korisničkog iskustva u sistem nagrađivanja bodovima bez istinskog osjećaja angažmana ili igre [1].

S vremenom, primjena igrifikacije proširila se na različite sektore, gdje je postala alat za rješavanje problema vezanih uz motivaciju i angažman korisnika. U zdravstvu, na primjer, igrifikacija se koristi za promicanje zdravih navika, kao što su tjelovježba, pravilna prehrana ili higijena ruku [3]. Slično tome, u radnim okruženjima, igrifikacija je korištena kako bi se povećala produktivnost i zadovoljstvo zaposlenika kroz nagrade, sustave postignuća i društvenu povezanost [4]. Ova šira primjena igrifikacije pokazala je da se elementi igara mogu prilagoditi gotovo bilo kojem kontekstu u kojem postoji potreba za povećanjem motivacije i sudjelovanja, bilo da se radi o profesionalnim ili osobnim ciljevima.

Unatoč uspjesima, dizajn učinkovitih sustava igrifikacije suočava se s brojnim izazovima. Jedan od ključnih problema je dugoročno održavanje angažmana korisnika. Iako igrifikacija može potaknuti početni interes kroz nagrade i sustave napredovanja, često se suočava s problemima poput ovisnosti korisnika o nagradama, varanja u sustavima bodovanja ili zamora korisnika zbog stalnog sudjelovanja [1][3].

1.1 Igrifikacija u obrazovanju

Igrifikacija se sve više koristi u obrazovnim sustavima, osobito u visokoškolskom obrazovanju, kako bi se potaknula motivacija, angažman i uspješnost učenika. Različite studije potvrđuju da primjena igrifikacijskih elemenata u obrazovanju može značajno poboljšati iskustvo učenja, osiguravajući dinamično okruženje koje olakšava usvajanje gradiva [5][6][7]. Uvođenje elemenata poput bodova, znački i rang lista omogućuje učenicima trenutačne povratne informacije o njihovom napretku, čime se dodatno povećava njihova motivacija i natjecateljski duh [6][7].

Posebno je korisno primijeniti fleksibilne sustave bodova, gdje učenici mogu prikupljati bodove kroz različite aktivnosti, pružajući im izbor između više načina za ostvarenje postavljenih ciljeva(Waterloo). Značke, kao digitalno priznanje za postignuća, dodatno potiču konkurenciju među učenicima i pomažu u stvaranju osjećaja postignuća, dok rang liste omogućuju praćenje napretka u odnosu na druge učenike, ali se moraju pažljivo konstruirati kako ne bi obeshrabrile one na dnu ljestvice [6][7].

Jedan od ključnih utjecaja igrifikacije u obrazovanju je poboljšanje dinamike učionice. Korištenjem alata kao što su Kahoot! i Jeopardy, nastavnici mogu potaknuti interakciju među učenicima, omogućujući aktivno sudjelovanje kroz natjecanje i igru. Ovi alati ne samo da stvaraju zabavno okruženje, već omogućuju učenicima da uče kroz međusobnu suradnju i natjecanje, čime se poboljšava zadržavanje znanja i angažman učenika [7].

Iako igrifikacija donosi mnoge prednosti, njezina primjena u obrazovnom okruženju može donijeti i određene izazove. Na primjer, prekomjerna uporaba natjecateljskih elemenata može stvoriti stres kod određenih učenika, a tehničke poteškoće poput problema s pristupom platformama mogu ometati proces učenja [6]. Ipak, pažljivo planirana igrifikacija može značajno povećati angažman i dugoročnu motivaciju učenika, čineći učenje učinkovitijim i zabavnijim [6][7].

Korištenje alata kao što je Kahoot! u obrazovanju dovelo je do značajnog povećanja angažmana i motivacije učenika. Istraživanja su pokazala da učenici koji koriste Kahoot! postižu bolje rezultate u usporedbi s onima koji sudjeluju u tradicionalnim metodama učenja. Na primjer, studija provedena u kontekstu učenja PHP-a pokazala je da više od 90% učenika smatra da je igrifikacija pomoću Kahoot!-a povećala njihovu motivaciju i interes za predmet, čime se poboljšao njihov ukupni akademski uspjeh [8]. Osim toga, nastavnici su izvijestili o poboljšanoj

dinamici učionice, uz veću interakciju i sudjelovanje učenika tijekom predavanja, što je rezultiralo pozitivnim stavom prema primjeni Kahoot!-a u nastavnom procesu [9][10].

Međutim, primjena ovakvih alata nije bez izazova. Neki učenici su izvijestili o osjećaju stresa zbog natjecateljske prirode Kahoot!-a, osobito kada su pitanja bila vremenski ograničena. Unatoč tome, velika većina sudionika u istraživanju izrazila je pozitivne stavove prema korištenju Kahoot!-a, navodeći da im anonimnost koju nudi alat omogućuje da slobodnije sudjeluju u kvizovima, smanjujući anksioznost koja je često prisutna u tradicionalnijim oblicima učenja. Takav anonimni pristup, u kojem se natjecanje usredotočuje na postignuća bez pritiska javnog izlaganja, posebno je učinkovit u smanjenju anksioznosti kod učenika [10].

Nastavnici su također istaknuli nekoliko prednosti korištenja Kahoot!-a. Alat im omogućuje da u stvarnom vremenu dobiju povratne informacije o znanju učenika, što olakšava prilagodbu nastavnog sadržaja prema potrebama grupe. Ova vrsta interakcije rezultira boljim zadržavanjem informacija, što potvrđuju i statistički podaci koji pokazuju da učenici koji sudjeluju u igrifikaciji imaju bolje rezultate na ispitima nego oni koji uče tradicionalnim metodama [8][9]. U kontekstu učenja programiranja, Kahoot! se pokazao kao iznimno koristan alat, potičući učenike da aktivno sudjeluju i brzo primjenjuju teorijska znanja na praktične probleme [8].

Ipak, neki su nastavnici izrazili zabrinutost zbog tehničkih izazova s kojima su se susreli prilikom korištenja Kahoot!-a, uključujući probleme s internetskom povezanošću i projekcijom sadržaja. Unatoč tome, ukupni učinci igrifikacije na motivaciju, angažman i akademski uspjeh prepoznati su kao značajni, a ovakvi alati se sve više integriraju u moderne obrazovne sustave [8][10].

1.2 Igrifikacija u poučavanju programiranja

Igrifikacija u kontekstu poučavanja programiranja fokusira se na specifične izazove koje ova vještina nosi, kao što su apstraktne koncepcije, tehničke vještine i praktična primjena teorije. Za razliku od klasičnih metoda poučavanja, igrifikacija omogućava učenicima da razvijaju svoje znanje programiranja kroz postavljanje ciljeva i kontinuiranu povratnu informaciju tijekom rješavanja zadataka, čime se poboljšava proces učenja. Na primjer, alati kao što su *CodeCombat* i *Codecademy* koriste elemente igrifikacije kao što su bodovi, razine i izazovi koji simuliraju stvarne probleme, omogućujući učenicima da se postupno suoče s kompleksnijim zadacima i istovremeno prate svoj napredak [11][12].

Ono što razlikuje igrifikaciju u poučavanju programiranja od tradicionalnijih metoda jest prilagodljivost i mogućnost personalizacije učenja. Učenici u igrificiranim okruženjima mogu birati brzinu i način učenja koji im najviše odgovara, što im omogućuje da postignu bolje rezultate u vlastitom tempu. Ovaj aspekt posebno je koristan u učenju programiranja jer omogućuje učenicima da razviju dublje razumijevanje koncepata prije nego što prijeđu na naprednije zadatke. Na taj način, igrifikacija ne samo da potiče motivaciju, već i osigurava da učenici temeljitije usvoje gradivo, što je od ključne važnosti za uspješno savladavanje programiranja [11][12].

Također, igrifikacija omogućuje kontinuiranu evaluaciju znanja programiranja, što olakšava nastavnicima prilagodbu sadržaja i pristupa ovisno o napretku učenika. Nastavnici koji koriste alate poput *Kahoot!* i *Codecademy* izvještavaju o povećanoj suradnji među učenicima i boljoj komunikaciji tijekom predavanja. Igrifikacija potiče učenike na aktivno sudjelovanje i suradnju u rješavanju zadataka, što tradicionalne metode često ne uspijevaju postići [12][13].

I kod igrifikacije učenja programiranja postoje neki izazovi. Učenička iskustva u igrificiranim okruženjima ponekad su opterećena natjecateljskim duhom, što može izazvati anksioznost kod onih koji se teže nose s pritiskom. Iako većina učenika prepoznaje prednosti igrifikacije, pojedinci izražavaju zabrinutost da ponekad fokus na bodove i postignuća može odvlačiti pažnju od stvarnog učenja [12][13]. Ipak, većina nastavnika smatra da je igrifikacija korisna u razvoju ključnih vještina programiranja jer omogućuje učenicima da kroz igru primijene i razumiju ključne teorijske koncepte [11][13].

Igrifikacija u obrazovanju nudi različite metode za povećanje angažmana i motivacije učenika, a među najučinkovitijima su interaktivni kvizovi. Korištenjem interaktivnih kvizova, učitelji mogu stvoriti dinamično okruženje u kojem učenici uče kroz igru, dok istovremeno ocjenjuju svoje znanje. Njihova prednost je u trenutačnim povratnim informacijama, koje učenicima omogućuju da odmah vide rezultate svog truda, a nastavnicima daju uvid u napredovanje učenika.. Na taj način kvizovi postaju više od običnog alata za evaluaciju – oni motiviraju učenike na aktivno sudjelovanje u procesu učenja. Drugi pristup igrifikaciji u obrazovanju je primjena sustava nagrađivanja. Učenici mogu dobivati bodove za različita postignuća, poput pravovremenog izvršavanja zadataka na vrijeme ili sudjelovanja u raspravama, što povećava njihovu motivaciju i osjećaj postignuća. Ovi bodovi mogu se zamijeniti za nagrade ili privilegije, čime se potiče njihova motivacija za postizanje boljih rezultata. Igranje uloga također je učinkovita metoda igrifikacije jer omogućuje učenicima da se uključe u simulacije stvarnih situacija. Ova metoda ne samo da potiče kritičko razmišljanje i rješavanje problema,

već učenicima omogućuje da razviju empatiju stavljajući se u tuđe uloge i razumijevajući različite perspektive. Još jedan koristan alat je pripovijedanje, koje uvodi učenike u obrazovne avanture kroz zanimljive narative. Kroz ovu metodu učenici se emocionalno povezuju s likovima i događajima, što olakšava razumijevanje i pamćenje gradiva. Na kraju, natjecanja u učionici potiču timski rad i suradnju među učenicima. Ova natjecanja ne samo da stvaraju pozitivno okruženje za učenje, već također razvijaju vještine potrebne za buduće grupne projekte i profesionalne situacije. [14]

U ovom diplomskom radu fokus će biti upravo na kvizovima kao ključnom alatu za primjenu igrifikacije u obrazovanju, s posebnim naglaskom na povećanje angažmana učenika putem ovih interaktivnih alata.

2. Online alati za izradu kvizova za učenje

Uvođenje modernih tehnologija u obrazovanje donijelo je brojne digitalne alate osmišljene za poticanje interaktivnog učenja. Ovi alati omogućuju stvaranje dinamičnih i prilagođenih nastavnih aktivnosti koje su lako dostupne učenicima putem računala ili mobilnih uređaja, čineći ih izuzetno prikladnima za suvremene učionice. Njihova fleksibilnost omogućuje nastavnicima da ih biraju u skladu s vlastitim preferencijama, razinom tehničkog znanja te uzrastom i potrebama učenika. Pravilnim odabirom, nastavnici mogu osigurati visoku razinu angažmana i motivacije učenika, što u konačnici doprinosi boljim obrazovnim rezultatima.

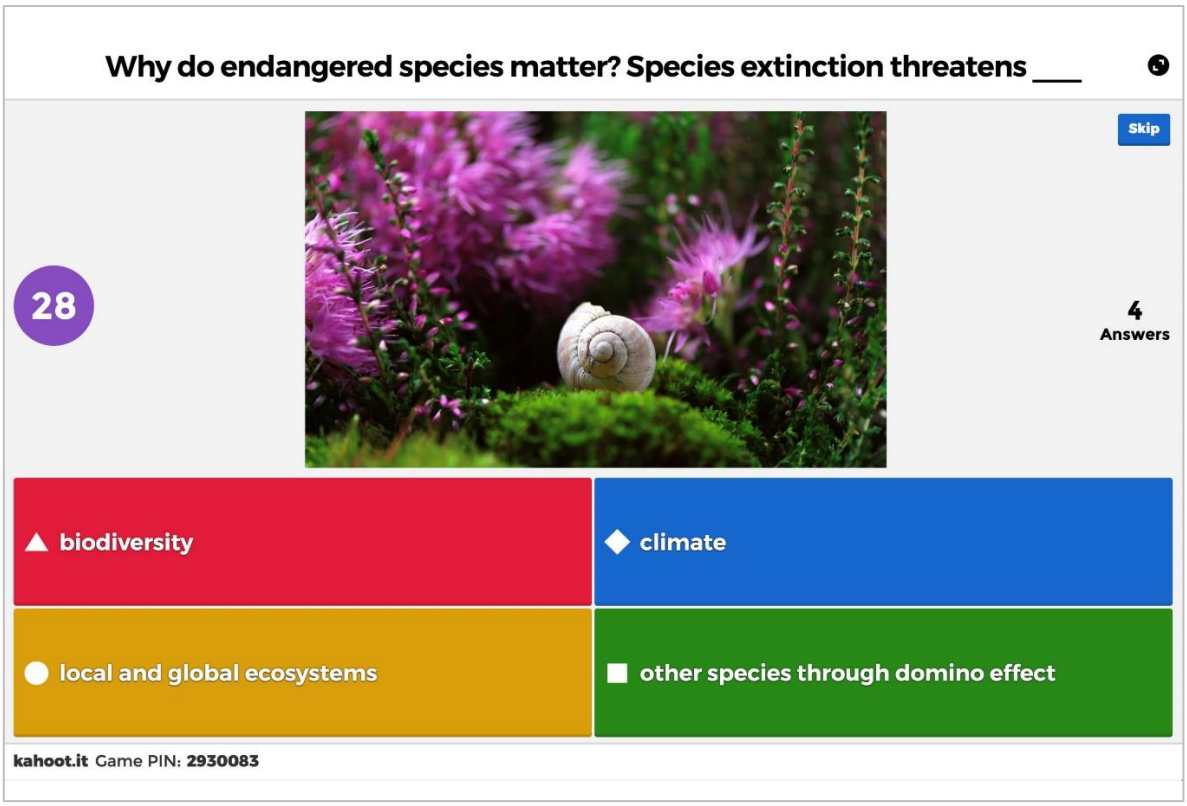
2.1 Kahoot!

Kahoot! je popularan online alat za izradu kvizova koji omogućava profesorima i nastavnicima kreiranje interaktivnih ispita za učenike. Dostupan je putem web stranice https://kahoot.com, a može se koristiti i na računalima i mobilnim uređajima. Aplikacija je dostupna za preuzimanje na platformama Android i iOS, a osnovna verzija Kahoot!-a besplatna je za sve korisnike. Za napredne funkcionalnosti, poput personalizacije kvizova i analitike, dostupni su plaćeni planovi koji nude dodatne alate za obrazovanje i poslovne korisnike.

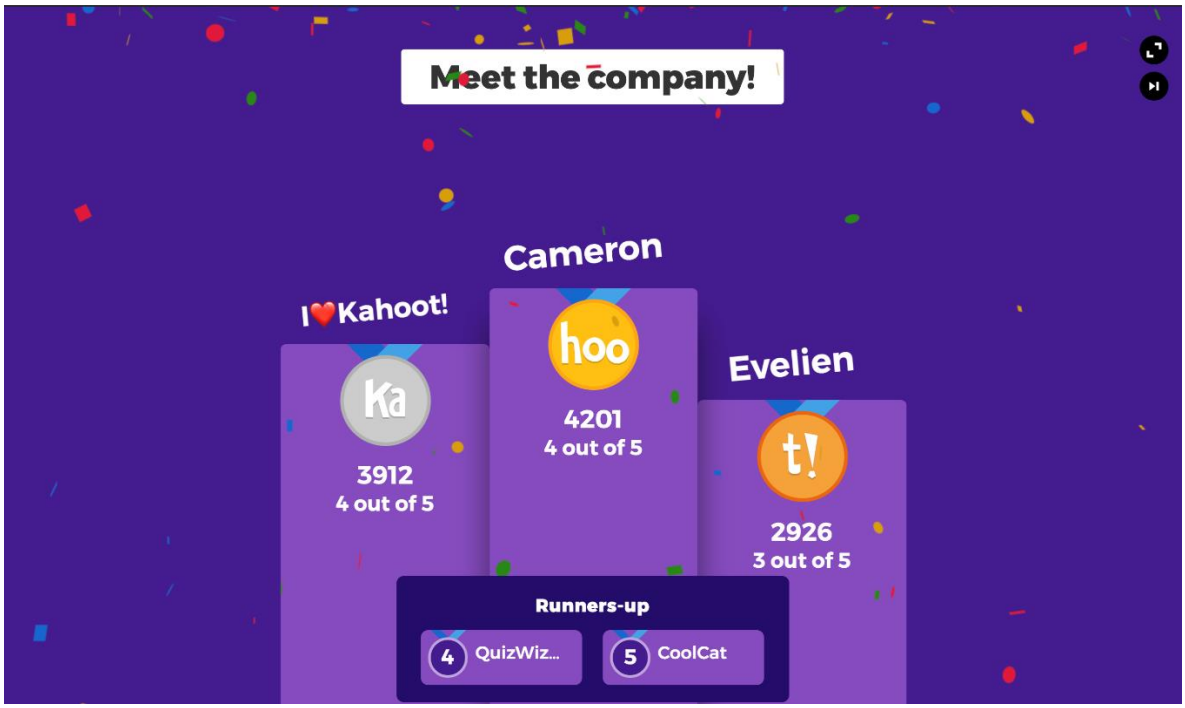
Kahoot! nudi jednostavno i intuitivno sučelje koje omogućuje brzo kreiranje kvizova s različitim vrstama pitanja, poput višestrukog izbora, točno/netočno pitanja i pitanja otvorenog tipa. Korisnici mogu dodavati slike i videozapise za bolju interakciju, a kvizovi se mogu igrati u stvarnom vremenu putem zajedničkog kôda koji učenici unose na svojim uređajima. Sve što je potrebno je internetska veza i uređaj (računalo, tablet ili mobitel), a učenici sudjeluju u kvizovima putem preglednika ili mobilne aplikacije.

Jedna od njegovih ključnih prednosti je natjecateljski element – bodovanje se temelji na brzini i točnosti odgovora, a rezultati se prikazuju u stvarnom vremenu na rang listi, što motivira učenike da sudjeluju aktivno i natječu se međusobno. Za razliku od nekih drugih alata za e-učenje, Kahoot! se izdvaja po svojoj fleksibilnosti i zabavnom pristupu. Koristi se u raznim obrazovnim kontekstima – od osnovnih škola do sveučilišta – i posebno je popularan zbog svoje jednostavnosti i mogućnosti uključivanja velikog broja sudionika u istom trenutku. O njegovoj popularnosti dovoljno govori podatak da su krajem 2023. godine dosegli broj od 10 milijardi zabilježenih sudionika od svog osnivanja.

Na slici 1. prikazan je primjer učeničkog ekrana za vrijeme riješavanja kviza, dok je na slici 2. prikazan primjer konačnih rezultata kviza.



***Slika* 1.** Prikaz učeničkog ekrana za vrijeme riješavanja kviza, koji uključuje pitanje i dostupne mogućosti odgovora .



***Slika* 2.** Prikaz konačnih rezltata kviza, s detaljima o osvojenim bodovima i rangiranju učenika.

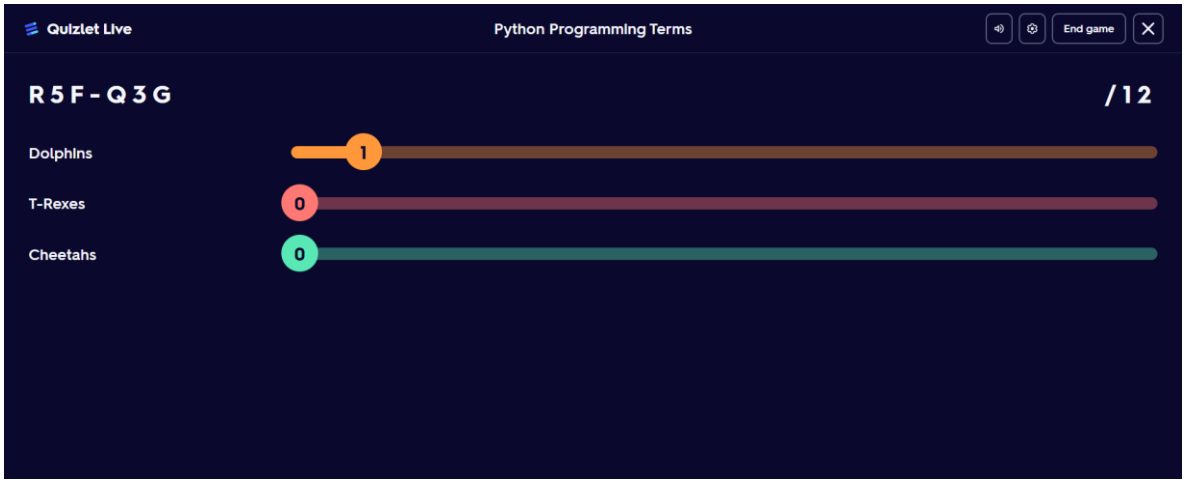
2.2 Quizlet

Quizlet je online platforma namijenjena za učenje i ponavljanje gradiva kroz različite oblike interaktivnih kartica (*engl.* flashcards) i kvizova. Platformi se može pristupiti putem web stranice https://quizlet.com, a mobilna aplikacija je dostupna za Android i iOS platforme. Nudi kvizove koji su jednostavni za korištenje, a mogu se kreirati na temelju postojećih skupova kartica ili izrađivati od nule. Kvizovi uključuju različite vrste pitanja, poput pitanja s višestrukim izborom, podudaranja pojmova, kratkih odgovora ili pitanja tipa "ispuni praznine." Ova fleksibilnost omogućuje nastavnicima da prilagode sadržaj potrebama svojih učenika, bilo da se radi o jezicima, informatici, biologiji ili matematici.

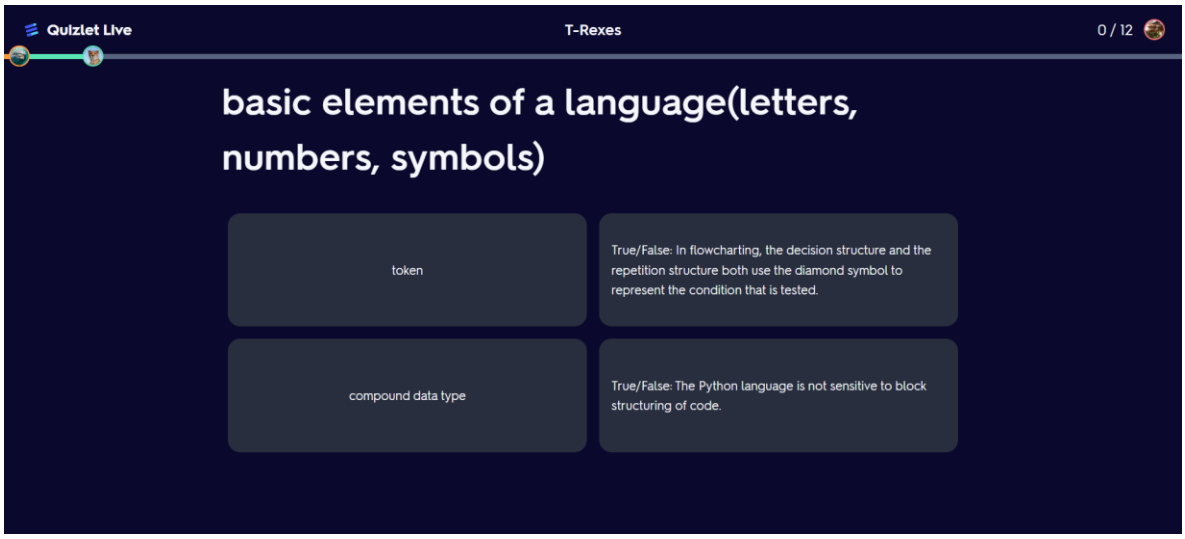
Kvizovi se mogu igrati individualno ili u sklopu modula Quizlet Live, gdje se učenici natječu u timovima. Svaki tim dobiva jedinstvenu kombinaciju odgovora na pitanja, što potiče timsku suradnju jer učenici moraju zajedno pronaći točan odgovor. Quizlet Live nema vremensko ograničenje za odgovore, ali se boduje brzina odgovaranja – tim koji prvi odgovori točno na sva pitanja osvaja pobjedu. Također, nema klasičnog odbrojavanja vremena (*engl.* timer), što učenicima omogućuje da se fokusiraju na točnost i suradnju umjesto na brzinu.

Sam proces korištenja kvizova je vrlo jednostavan: nastavnici kreiraju set pitanja, biraju vrstu kviza te ga podijele s učenicima putem poveznice ili kôda. Quizlet također nudi analitiku za nastavnike, koja im omogućuje praćenje napretka učenika i pregled točnih i netočnih odgovora.

Slika 3. prikazuje kako izgleda profesorovo sučelje u Quizletu za vrijeme trajanja kviza, dok se na slici 4. nalazi prikaz pitanja iz kviza učenicima.



***Slika* 3.** Izgled profesorovog sučelja u Quizletu za vrijeme trajanja kviza.



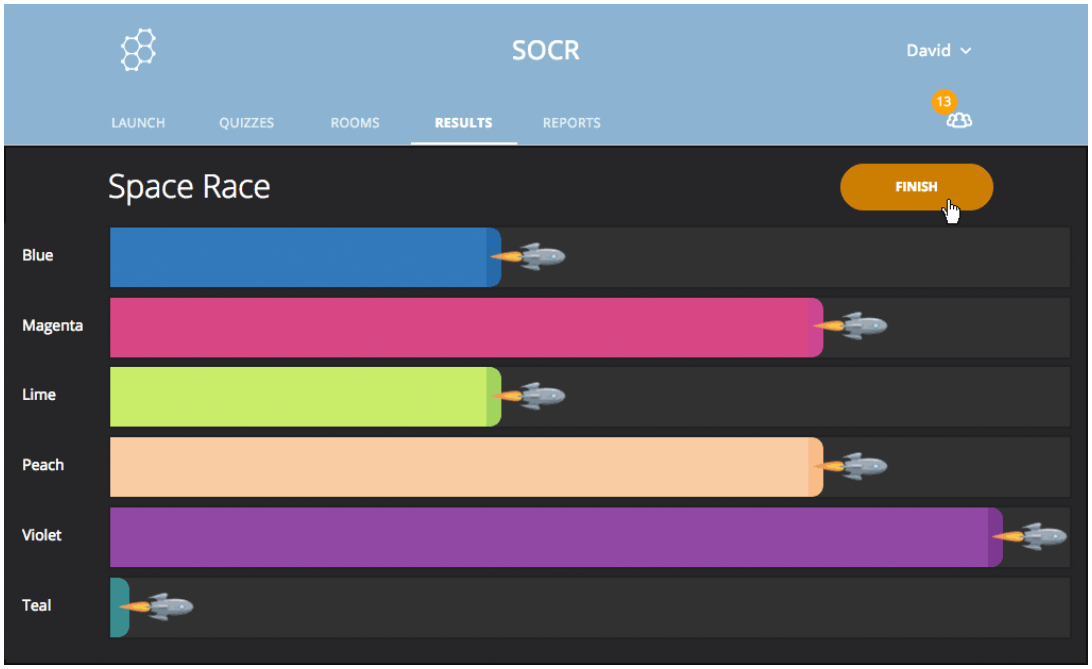
***Slika* 4.** Prikaz pitanja koja učenici riješavaju tijekom kviza.

2.3 Socrative

Socrative je interaktivni alat dizajniran za provođenje kvizova i formativnih procjena u učionici i online okruženju. Koristi se kako bi nastavnici u stvarnom vremenu pratili napredak učenika i pružali povratne informacije. Dostupan je na svim uređajima, uključujući iOS, Android i web preglednike, čime se osigurava jednostavan pristup bilo gdje i bilo kada. Učenici pristupaju kvizovima putem jedinstvenog kôda sobe, a rezultati se prikazuju nastavnicima u stvarnom vremenu, omogućujući trenutačnu analizu i prilagodbu nastave. Podržava različite vrste pitanja, uključujući višestruki izbor, pitanja točno/netočno te kratke odgovore. Ove značajke omogućuju nastavnicima da prilagode kvizove različitim temama i razinama učenja. Alat također omogućuje upotrebu funkcije *Space Race*, u kojoj se učenici natječu u timovima odgovarajući na pitanja unutar vremenskog okvira (slika 5.)

Velika prednost ovog alata je mogućnost trenutne analize rezultata (slika 6.). Nakon svakog odgovora, rezultati se automatski prikazuju nastavnicima, što nastavniku nudi smjernice za prilagodbu sadržaja prema potrebama učenika. Alat je posebno koristan za identificiranje praznina u znanju i omogućuje nastavnicima da ciljano rješavaju te probleme tijekom predavanja. Socrative je također poznat po jednostavnoj integraciji s različitim obrazovnim standardima, poput *Common Core* standarda u SAD-u, što omogućuje nastavnicima da precizno procijene napredak učenika u skladu s obrazovnim ciljevima. Osim toga, funkcija Exit Ticket omogućuje nastavnicima brzu procjenu razumijevanja gradiva na kraju predavanja, što pomaže u održavanju visokog fokusa učenika tijekom cijelog predavanja.

Socrative dolazi u različitim cjenovnim planovima. Besplatni plan omogućuje osnovne funkcionalnosti poput jednog razreda s do 50 učenika, dok plaćeni planovi nude proširene mogućnosti, uključujući do 200 učenika po razredu, više soba, detaljniju analitiku, mogućnost dijeljenja kvizova, te napredne alate za organizaciju sadržaja.

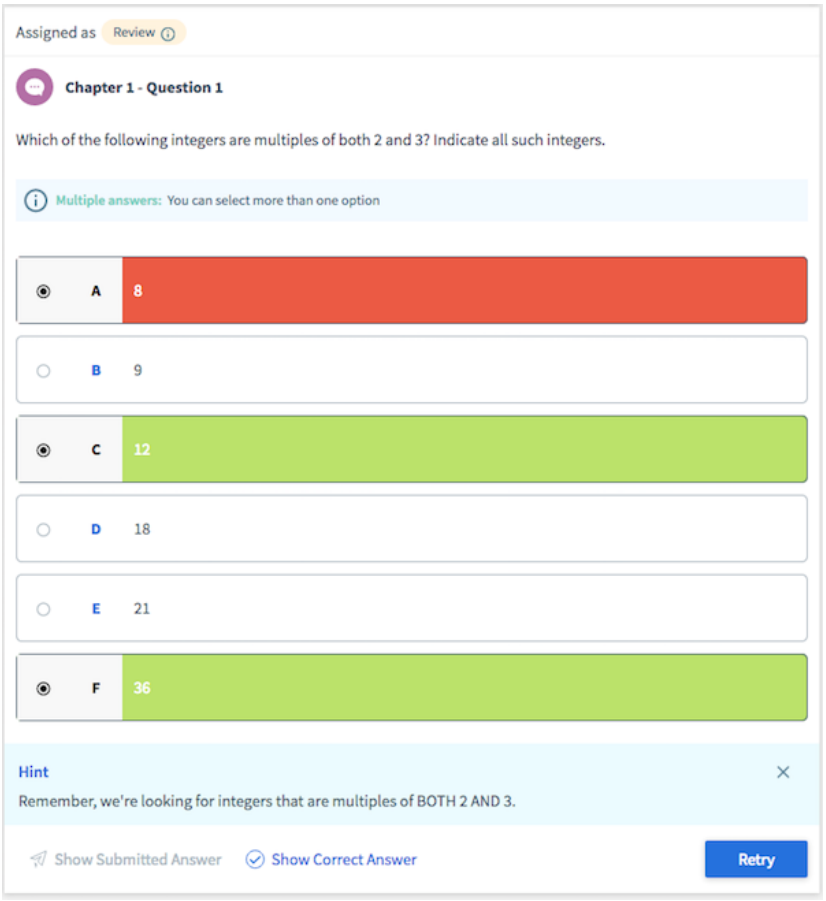


***Slika* 5.** Prikaz funkcije *Space Race* u alatu Socrative.

***Slika* 6.** Prikaz rezultata kviza u stvarnom vremenu u alatu Socrative.

2.4 Top Hat

Top Hat je napredni obrazovni alat osmišljen za poboljšanje interaktivnog učenja, omogućujući nastavnicima kreiranje dinamičnih i zanimljivih ispita i kvizova za studente. Jedna od glavnih prednosti alata je korisničko sučelje koje nastavnicima olakšava izradu i organizaciju testova. I ovaj alat pruža različite tipove pitanja poput višestrukog izbora, točno/netočno i kratkih odgovora, čineći ga prilagodljivim za različite ishode učenja. Podržava multimedijske sadržaje, omogućujući dodavanje slika, videozapisa i grafikona za veći angažman studenata. Slika 7. prikazuje sučelje ispita iz perspektive učenika.

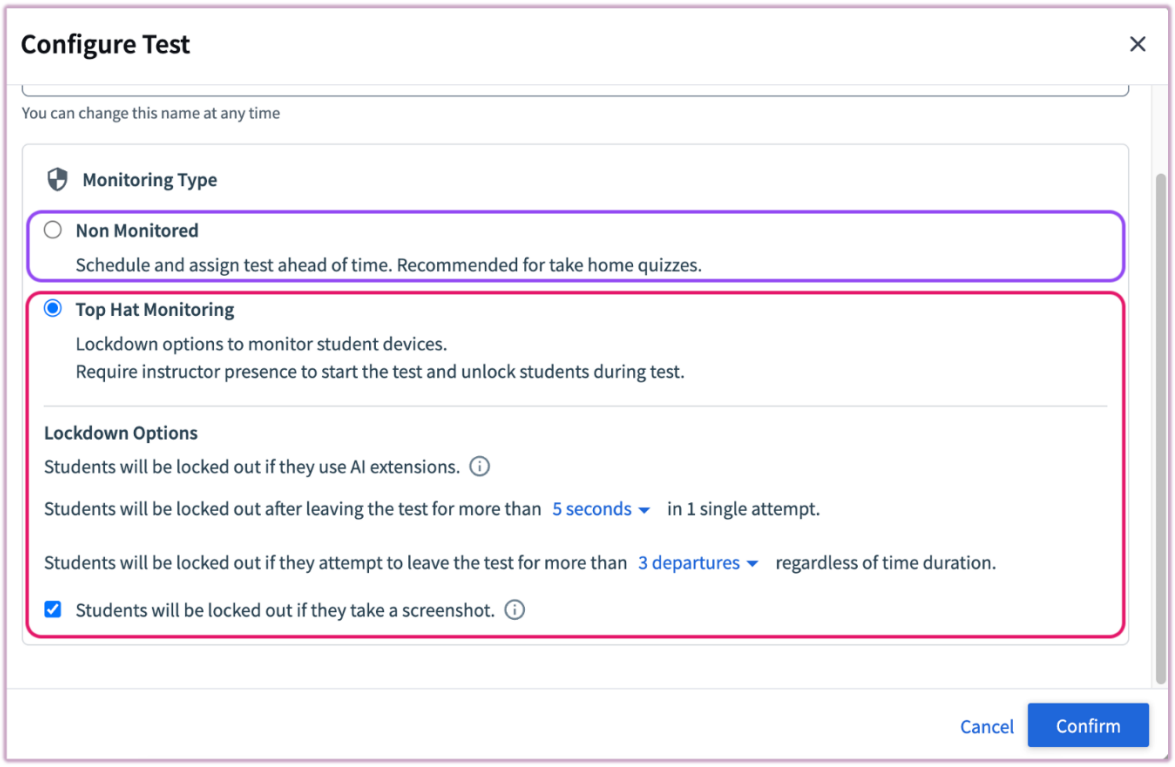


***Slika* 7.** Izgled sučelja ispita u alatu Top Hat iz perspektive učenika.

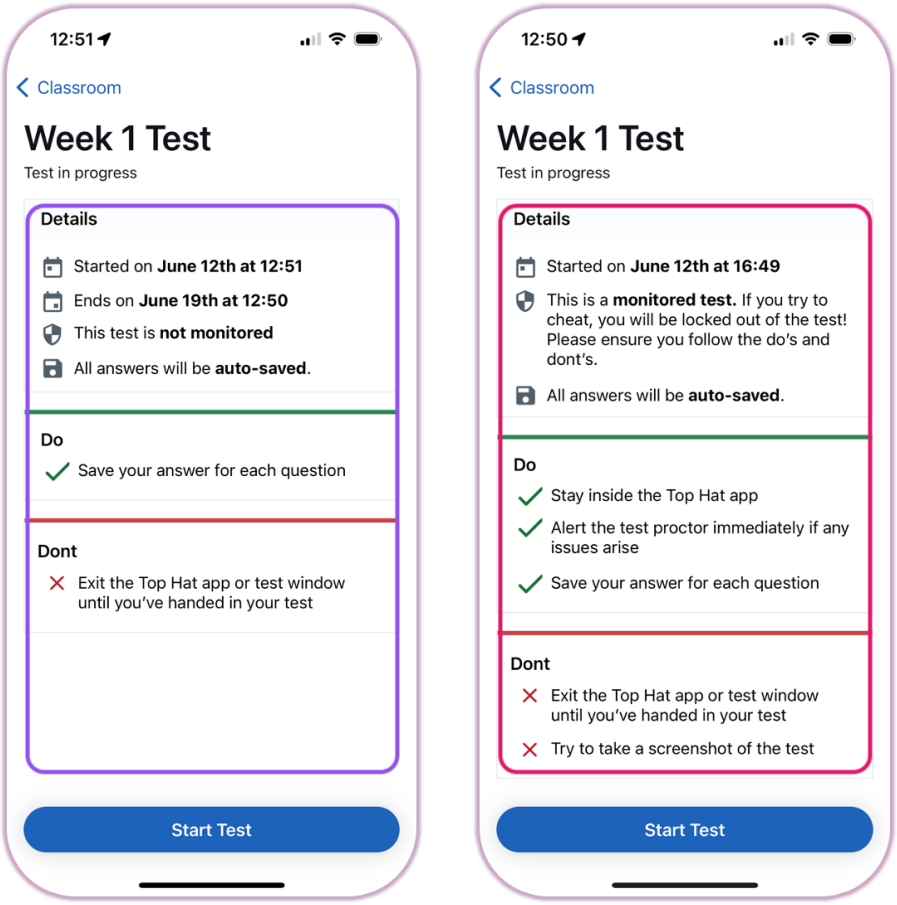
Proces izrade testa je jednostavan, a nastavnici mogu kreirati testove unutar same platforme koristeći prilagodljive predloške. Mogu postaviti parametre poput vremenskih ograničenja, redoslijeda pitanja i nasumičnog odabira kako bi spriječili varanje. Ova fleksibilnost omogućuje prilagođeno ocjenjivanje, prema potrebama i ciljevima kolegija. Također, Top Hat omogućuje testiranje u stvarnom vremenu, gdje nastavnici mogu postaviti ispite tijekom predavanja uživo

i pratiti odgovore učenika. Nudi i napredne opcije za zaključavanje testa kako bi se osigurala akademska poštenost. Nastavnici mogu konfigurirati različite razine zaštite, uključujući automatsko zaključavanje testa nakon što učenik napusti prozor na određeni broj sekundi. Dodatno, test se može zaključati nakon određenog broja izlazaka iz prozora, neovisno o vremenu, te se može uključiti opcija zaključavanja ako učenik napravi snimku zaslona. Sve ove postavke prikazane su na slici 8., a pomažu spriječiti prepisivanje i povećavaju sigurnost tijekom online evaluacije učenika. Slika 9. prikazuje način na koji će sva ova pravila i upute biti prikazani učenicima.

Top Hat je dostupan na svim uređajima, što osigurava neometanu komunikaciju između nastavnika i studenata, bilo u učionici ili na daljinu.



***Slika* 8.** Postavke sigurnosnih opcija u alatu Top Hat, uključujući zaključavanje ispita nakon izlaska iz prozora i detekciju snimanja zaslona.



***Slika* 9.** Prikaz pravila i uputa učenicima prije početka ispita u alatu Top Hat.

2.5 Sažetak alata za izradu online kvizova

Dostupni alati za izradu kvizova razlikuju se prema razini složenosti i funkcionalnosti, pružajući različite mogućnosti prilagodbe za različite obrazovne kontekste. Dok su neki osmišljeni za niže razrede s naglaskom na jednostavno sučelje i elemente zabave, drugi nude napredne alate za analizu rezultata, čime su prikladni za starije učenike i visoko obrazovanje. Ovi alati omogućuju nastavnicima da preciznije prate napredak učenika te prilagode aktivnosti prema njihovim potrebama i željama.

Tablica 1. daje pregled ključnih karakteristika alata koji su spomenuti u prethodnim dijelovima poglavlja.

***Tablica* 1.** Usporedba alata na temelju određenih značajki

	Kahoot!	Quizlet	Socrative	Top Hat
Vrste pitanja	višestruki izbor, točno/netočno	višestruki izbor, točno/netočno, povezivanje, unos odgovora	višestruki izbor, točno/netočno, kratki odgovor	višestruki izbor, točno/netočno, povezivanje, unos odgovora, Likertova skala, sortiranje…
Multimedija u pitanjima	slike, videozapisi	slike, videozapisi	slike, videozapisi, audiozapisi	slike, videozapisi, tablice, bilješke
Rang liste učenika	DA	DA	DA	NE
Natjecanje timova	DA	DA	DA	NE
Vremenski ograničen odgovor	DA	NE	NE	DA
Zaključavanje testa	DA	DA	DA	DA
Cijena	Besplatno Kahoot+ od $4,99/mj.	Besplatno Quizlet Plus: $35.99/god.	Besplatno Essentials: $9.99/mj. Pro: $16.99/mj.	Basic: besplatno $53.00/god.
Dostupne platforme	desktop, web, iOS/Android	desktop, web, iOS/Android	desktop, web, iOS/Android	desktop, web
Broj mjesečnih korisnika	50 milijuna	60 milijuna	3 milijuna	750 000
Lakoća korištenja*	4.7/5	4.6/5	4.4/5	3.9/5

** temeljeno na ocjenama korisnika za svaku aplikaciju, sa web lokacije*

https://www.capterra.com/

3. Oblikovanje aplikacije

U ovom dijelu rada bit će predstavljena web aplikacija za igrifikaciju online učenja koja je razvijena kao dodatak nastavi. Aplikacija je namijenjena i nastavnicima i učenicima. Nastavnici mogu kreirati i upravljati ispitima kojima će učenici pristupati. S druge strane, učenici mogu sudjelovati u ispitima, rješavati zadatke i uspoređivati svoje rezultate sa svojim vršnjacima, s ciljem povećavanja motivacije i angažmana u učenju.

3.1 Odabir tehnologija za razvoj aplikacije

Tehnološka rješenja korištena u razvoju aplikacije odabrana su na temelju nekoliko ključnih čimbenika, uključujući fleksibilnost, lakoću upotrebe i sposobnost integracije različitih komponenti sustava. U nastavku se detaljno opisuju korištene tehnologije, zajedno s njihovom specifičnom ulogom u razvoju aplikacije.

3.1.1 Flask

Flask je mikro-okvir za izgradnju web aplikacija u programskom jeziku Python. Zbog svoje minimalističke arhitekture, Flask omogućuje fleksibilnost i modularnost u razvoju aplikacija, što ga čini idealnim izborom za projekte koji ne zahtijevaju složene funkcionalnosti koje pružaju mnogo veći okviri poput Django-a. U sklopu ove aplikacije, Flask je korišten kao osnovni okvir za implementaciju poslužiteljske strane. Njegove ključne karakteristike, poput jednostavnog routing sustava i podrške za WSGI aplikacije, omogućile su učinkovito upravljanje HTTP zahtjevima i razmjenu podataka između klijenta i poslužitelja. Također, Flask podržava jednostavno proširenje dodatnim komponentama, poput SocketIO i Jinja2, čime je omogućeno proširenje osnovnih funkcionalnosti aplikacije.

3.1.2 SocketIO

SocketIO je biblioteka koja omogućuje dvosmjernu komunikaciju brzog odziva (*engl.* low-latency) između klijenta i poslužitelja u stvarnom vremenu. U kontekstu ove aplikacije, SocketIO je odigrao ključnu ulogu u omogućavanju automatizacije korisničkih akcija i tijeka aplikacije. Kroz SocketIO, aplikacija je u mogućnosti emitirati događaje u stvarnom vremenu svim sudionicima ispita, omogućujući trenutno ažuriranje rezultata na rang listi, promjene u statusu ispita i interaktivno sudjelovanje korisnika. Ova tehnologija omogućuje skalabilnu i

pouzdanu komunikaciju u stvarnom vremenu, što je ključno za ovakve aplikacije koje zahtijevaju konstantnu sinkronizaciju podataka između više klijenata.

3.1.3 TailwindCSS

TailwindCSS je praktičan CSS okvir koji omogućuje brzu i jednostavnu stilizaciju korisničkog sučelja putem unaprijed definiranih klasa. Za razliku od tradicionalnih CSS okvira, poput Bootstrap-a, koji nude gotove komponente, TailwindCSS pruža veću kontrolu nad dizajnom omogućujući direktno prilagođavanje elemenata unutar HTML-a. U razvoju ove aplikacije, TailwindCSS je korišten za kreiranje intuitivnog i responzivnog korisničkog sučelja. Njegova fleksibilnost omogućila je jednostavno prilagođavanje sučelja različitim veličinama ekrana, čime je osigurano optimalno korisničko iskustvo na različitim uređajima.

3.1.4 HTML, CSS, JavaScript

Osnovne web tehnologije korištene za izradu frontend sučelja uključuju HTML, CSS i JavaScript. HTML (*engl.* Hypertext Markup Language) koristi se za strukturu i semantiku web stranice, dok je CSS (*engl.* Cascading Style Sheets) odgovoran za izgled i oblikovanje elemenata sučelja. JavaScript, kao najvažniji jezik za razvoj interaktivnosti na webu, korišten je za manipulaciju DOM-om (*engl.* Document Object Model), obradu korisničkih akcija te povezivanje s backend funkcionalnostima. U aplikaciji su HTML i CSS korišteni za definiranje osnovnog izgleda i strukture korisničkog sučelja, uključujući stranice za prijavu, registraciju, nadzornu ploču nastavnika te stranice za rješavanje ispita. JavaScript, u kombinaciji s Flaskom i SocketIO-om, omogućuje neprekidnu dvosmjernu komunikaciju između klijenta i poslužitelja, osiguravajući interaktivnost i ažuriranje u stvarnom vremenu. JavaScript upravlja događajima koji se odvijaju na klijentskoj strani, kao što su prijenos podataka o odgovorima učenika i prikaz ažuriranih rezultata na rang listi, dok SocketIO prenosi te podatke u stvarnom vremenu, omogućujući sinkronizaciju svih korisničkih akcija unutar ispita.

3.1.5 Jinja2

Jinja2 je Pythonov templating engine koji se koristi u Flasku za dinamičko generiranje HTML stranica. Kroz Jinja2, moguće je unijeti promjenjive podatke unutar statičkih HTML predložaka, omogućujući poslužitelju da šalje personalizirane stranice korisnicima na temelju njihovih podataka ili akcija. U sklopu ove aplikacije, Jinja2 je korišten za kreiranje dinamičkih stranica ispita, rezultata te korisničkog sučelja, čime se omogućilo jednostavno i učinkovito

upravljanje prikazom informacija na klijentskoj strani, ovisno o stanju ispita i interakcijama korisnika.

3.1.6 SQLAlchemy

SQLAlchemy je moćan ORM (*engl.* Object-Relational Mapper) alat koji omogućuje interakciju s relacijskim bazama podataka putem Python kôda, umjesto direktnim SQL upitima. Njegova upotreba omogućuje razvoj objektno-orijentiranih aplikacija uz istovremeno učinkovito upravljanje složenim bazama podataka. SQLAlchemy je korišten u ovoj aplikaciji za definiranje strukture baze podataka i upravljanje podacima o korisnicima, ispitima, pitanjima i rezultatima. Korištenje ovog ORM alata omogućilo je jednostavno mapiranje podataka iz baze u Python objekte, čime je značajno olakšana manipulacija podacima i smanjena potreba za ručnim pisanjem SQL upita.

3.1.7 Alembic

Alembic je alat za migraciju baze podataka, izgrađen na temelju SQLAlchemy-a, koji omogućuje verzioniranje promjena u strukturi baze podataka. Budući da aplikacije evoluiraju kroz vrijeme, promjene u bazi podataka su neizbježne, a Alembic omogućuje upravljanje tim promjenama na kontroliran i dokumentiran način. U ovoj aplikaciji, Alembic je korišten za migraciju promjena poput dodavanja novih tablica i atributa, čime je osigurano da je struktura baze podataka uvijek u skladu s novim zahtjevima i proširenjima aplikacije.

3.1.8 WTForms

WTForms je Python biblioteka koja olakšava rad s HTML obrascima, omogućujući jednostavno kreiranje i validaciju korisničkih unosa. U ovoj aplikaciji, WTForms je korišten u gotovo svim dijelovima gdje je potrebna interakcija s korisničkim unosima, prije svega prijava i registracija korisnika, kreiranje novih ispita, dodavanje pitanja u ispite te upravljanje podacima vezanim uz kreirane ispite. WTForms omogućuje sigurnu validaciju svih korisničkih unosa prije obrade na poslužiteljskoj strani, čime se osigurava stabilnost aplikacije i sprječava unošenje nevažećih ili malicioznih podataka.

3.2 Modeliranje zahtjeva aplikacije

Aplikacija omogućuje dvjema glavnim skupinama korisnika – nastavnicima i učenicima – sudjelovanje u ispitima. Svaka grupa korisnika ima specifične funkcionalnosti prilagođene njihovim ulogama.

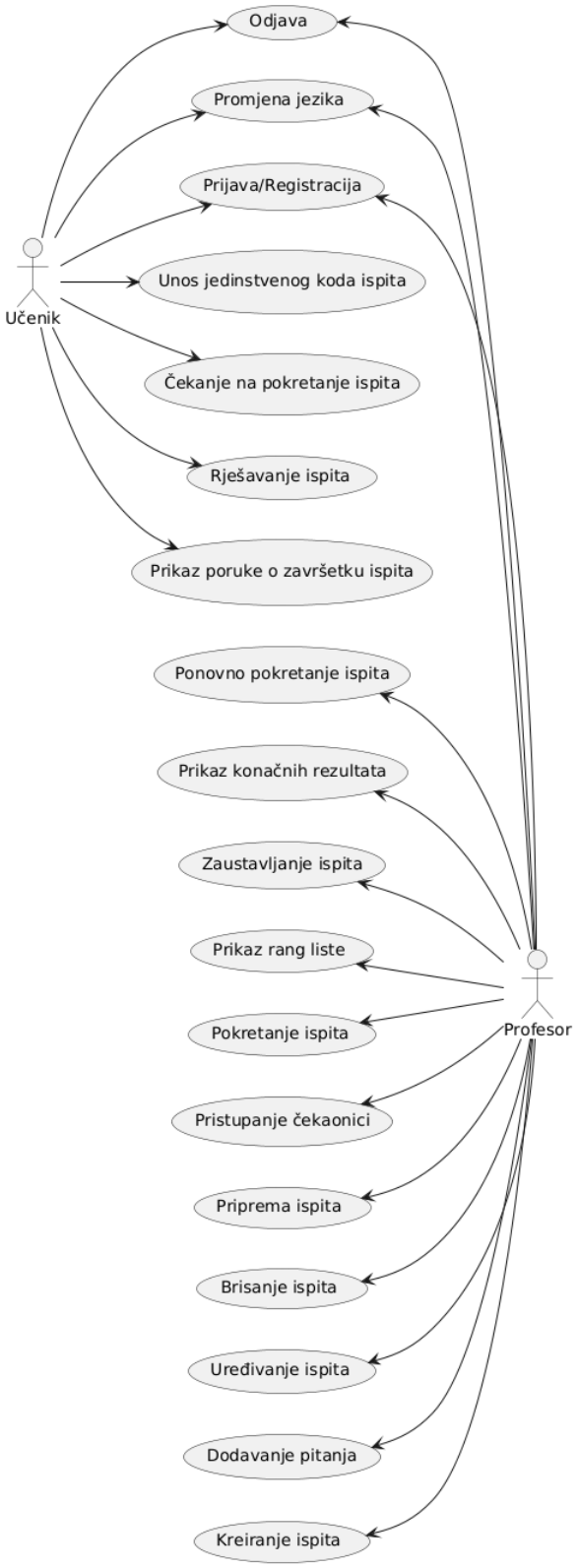
Nakon prijave ili registracije u sustav, nastavnici se preusmjeravaju na nadzornu ploču, gdje im je prikazan popis svih ispita koje su prethodno kreirali. Unutar ove nadzorne ploče nastavnici imaju mogućnost kreiranja novih ispita, uređivanja postojećih te dodavanja ili uklanjanja pitanja. Osim toga, ispiti se mogu pripremiti za izvođenje, brisati ili ponovno pokretati ako su već rješavani. Ponovno pokretanje ispita briše prethodne rezultate, čime se osigurava mogućnost višekratnog korištenja ispita s različitim (ali i istim) grupama učenika. Nakon završetka ispita, nastavnicima ostaju trajno dostupni rezultati, koji im omogućuju analizu uspjeha učenika.

S druge strane, učenici se, nakon prijave ili registracije, preusmjeravaju na stranicu koja zahtijeva unos jedinstvenog kôda ispita. Ovaj kôd, automatski generiran prilikom kreiranja ispita, omogućuje učenicima pristup specifičnom ispitu. Nakon unosa kôda, učenici su preusmjereni na stranicu za učitavanje ispita, gdje ostaju dok ga nastavnik ne pokrene. U stvarnom vremenu, nastavniku se na zaslonu prikazuju svi učenici koji su pristupili ispitu, omogućujući mu praćenje spremnosti grupe. Kada nastavnik odluči pokrenuti ispit, učenici prelaze na stranicu za odgovaranje na pitanja, gdje se ona prikazuju jedno po jedno. Nastavnici u stvarnom vremenu mogu pratiti napredak učenika putem rang liste koja se ažurira pri svakom točnom odgovoru. Rang lista nudi transparentan prikaz bodova učenika i omogućuje natjecateljsku komponentu učenja.

Nastavnici u bilo kojem trenutku imaju mogućnost zaustaviti ispit, čime se učenicima onemogućava daljnje rješavanje i pregledavanje pitanja, a oni se vraćaju na početnu stranicu s mogućnošću pristupa drugim ispitima. Ako učenik završi sva pitanja prije nego što nastavnik zaustavi ispit, aplikacija automatski prikazuje obavijest i preusmjerava učenika na njegovu početnu stranicu. Po završetku ispita, nastavnici mogu pregledati i analizirati konačne rezultate, a učenici se mogu ponovno prijaviti na drugi ispit putem novog kôda.

Aplikacija također podržava dvojezično sučelje, omogućujući korisnicima odabir između hrvatskog i engleskog jezika. Jezične postavke, kao i mogućnost odjave iz sustava, dostupne su svim korisnicima u bilo kojem trenutku kroz navigacijsku traku prisutnu na svim stranicama osim onih za prijavu i registraciju. Navedene funkcionalnosti aplikacije bit će prikazane putem

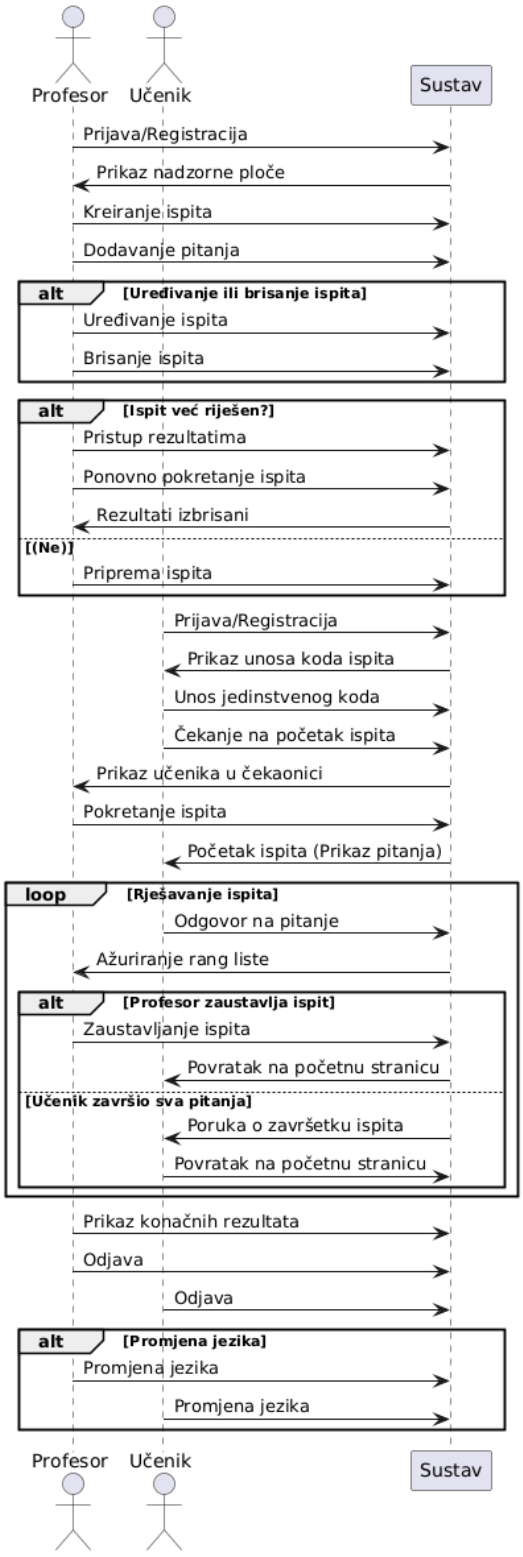
dijagrama slučajeva korištenja (slika 10.), kojeg je potrebno izraditi prije samog početka razvoja aplikacije.



***Slika* 10.** Dijagram slučajeva korištenja aplikacije s prikazom ključnih funkcionalnosti.

3.3 Modeliranje tijeka aplikacije

Na slici 11. prikazan je dijagram slijeda koji pobliže opisuje kako se navedene funkcionalnosti izvode.



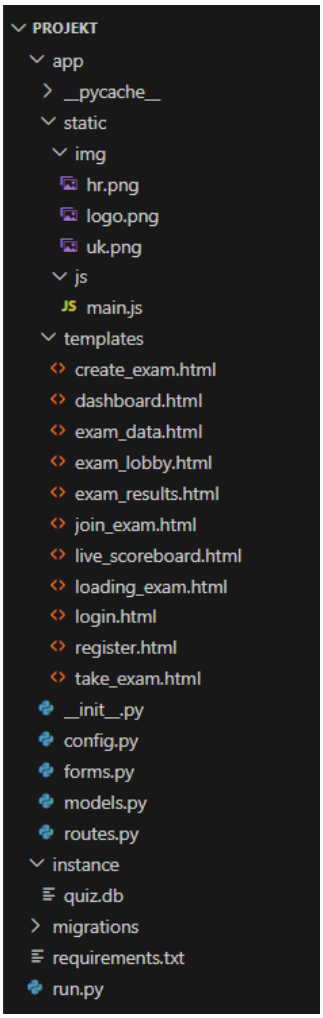
***Slika* 11.** Dijagram slijeda koji prikazuje izvedbu ključnih funkcionalnosti aplikacije.

4. Implementacija aplikacije

U ovom poglavlju analizirat će se tehnički aspekti izrade aplikacije, s naglaskom na način kako su osnovni elementi povezani i organizirani u funkcionalnu cjelinu. Prikazat će se struktura projekta, ključne datoteke i implementirana rješenja, kroz koja se ostvaruje složenost sustava iza korisničkog sučelja te način na koji tehničke komponente podržavaju sve funkcionalnosti ove aplikacije.

4.1 Struktura projekta

Projekt je organiziran tako da se jasno odvajaju statički resursi (slike, JavaScript datoteke), predlošci sučelja (engl. *templates*), konfiguracijske datoteke i funkcionalni moduli aplikacije, čime je osigurana čitljivost i održivost koda. Slika 12. prikazuje organizaciju projektnih datoteka unutar odgovarajućih mapa, a odmah zatim opis nekih datoteka i mapa koje je potrebno pobliže objasniti.



***Slika* 12.** Organizacija projektnih datoteka unutar odgovarajućih mapa.

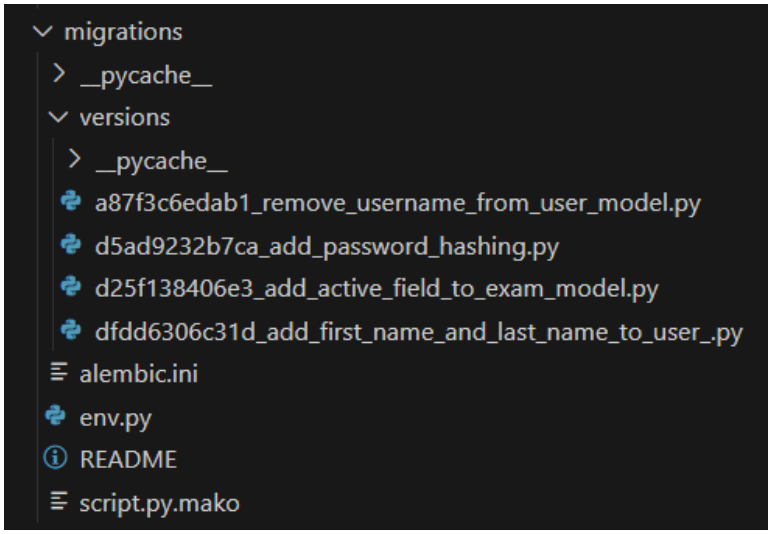
- **`__init__.py`:** Služi za inicijalizaciju aplikacije. U ovoj datoteci postavlja se konfiguracija aplikacije, inicijalizira baza podataka te pokreću ekstenzije kao što su Flask-SocketIO za komunikaciju u stvarnom vremenu i Flask-Login za autentifikaciju korisnika. Dio kôda zaslužan za inicijalizaciju aplikacije:

```
from flask import Flask, request, session
from flask_sqlalchemy import SQLAlchemy
from flask_login import LoginManager
from flask_socketio import SocketIO
from flask_migrate import Migrate
import os

db = SQLAlchemy()
login_manager = LoginManager()
socketio = SocketIO()
migrate = Migrate()

def create_app():
    app = Flask(__name__)
    app.config.from_object('app.config.Config')

    db.init_app(app)
    login_manager.init_app(app)
    socketio.init_app(app)
    migrate.init_app(app, db)

    from .routes import bp as main_bp
    app.register_blueprint(main_bp)

    return app
```

- **`config.py`:** Ova datoteka sadrži konfiguracijske postavke aplikacije, što uključuje postavke za bazu podataka, tajni ključ za sesije i CSRF zaštitu. Kôd ove datoteke:

```
class Config:
    SECRET_KEY = os.environ.get('SECRET_KEY', '8vC3BqPJueHIXmZC0JYe')
    SQLALCHEMY_DATABASE_URI = os.environ.get('DATABASE_URL',
'sqlite:///quiz.db')
    SQLALCHEMY_TRACK_MODIFICATIONS = False
    WTF_CSRF_ENABLED = True
    WTF_CSRF_SECRET_KEY = os.environ.get('SECRET_KEY',
'8vC3BqPJueHIXmZC0JYe')
```

- **`migrations:`** Ova mapa sadrži migracijske skripte koje se generiraju pomoću Alembic alata kako bi se omogućile promjene u strukturi baze podataka. Svaka migracija prilagođava bazu prema novim definicijama modela. Migracije se generiraju automatski na temelju promjena u modelima. Sadržaj ove mape prikazan je na slici 13.



***Slika* 13.** Sadržaj mape migrations.

- **`templates:`** Mapa templates sadrži HTML datoteke koje definiraju izgled korisničkog sučelja. Flask koristi Jinja2 kao svoj tzv. (engl.) *template engine*, koji omogućuje dinamičko generiranje HTML-a utemeljenog na Python varijablama i logici. Datoteke unutar mape templates automatski prepoznaje Flask, koji ih pronalazi i koristi za renderiranje odgovarajućih stranica na zahtjev korisnika.

  Svaki put kada se pozove funkcija **render_template**, Flask traži odgovarajući HTML predložak unutar ove mape. Na primjer:

```
@bp.route('/login', methods=['GET', 'POST'])
def login():
    return render_template('login.html')
```

Flask automatski pretražuje mapu templates i prikazuje sadržaj datoteke **login.html**, čime se omogućuje jednostavno upravljanje korisničkim sučeljem.

Jinja2 omogućuje korištenje naprednih funkcija unutar HTML datoteka, kao što su:

- petlje – omogućuju iteriranje kroz kolekcije podataka unutar HTML-a
- uvjetne naredbe – omogućuju prikazivanje različitih dijelova stranice ovisno o stanju varijabli
- varijable – podaci iz backenda mogu se lako prikazati na HTML stranici koristeći jednostavnu sintaksu.

4.2 Baza podataka

U ovoj sekciji analizirat će se modeli podataka korišteni u aplikaciji, koji predstavljaju temelj za upravljanje korisnicima, ispitima, pitanjima i rezultatima. Modeli su definirani korištenjem SQLAlchemy ORM-a, što omogućuje rad s bazom podataka na objektno orijentiran način, bez potrebe za pisanjem izravnih SQL naredbi.

4.2.1 Model korisnika

User model koristi se za pohranu podataka o korisnicima aplikacije, uključujući nastavnike i učenike. Svaki korisnik ima jedinstveni ID, ime, prezime, email adresu, ulogu (nastavnik ili učenik) te hashiranu lozinku za sigurnu autentifikaciju. Uz to, model implementira metode za postavljanje i provjeru lozinke, koristeći Werkzeug sigurnosne funkcije za generiranje i provjeru hashiranih lozinki.

```
class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    first_name = db.Column(db.String(50), nullable=False)
    last_name = db.Column(db.String(50), nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)
    password_hash = db.Column(db.String(128), nullable=False)
    role = db.Column(db.String(10), nullable=False)

    def set_password(self, password):
        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        return check_password_hash(self.password_hash, password)

    @staticmethod
    def get(user_id):
        return User.query.get(int(user_id))
```

4.2.2 Model ispita

Model *Exam* koristi se za pohranu podataka o ispitima koje kreiraju nastavnici. Svaki ispit ima jedinstveni ID, naziv ispita, jedinstveni kôd koji se automatski generira pri kreiranju te status (npr. "not_started", "started", "stopped"). Model sadrži odnose prema modelima *Question* i *Score*, što omogućuje povezivanje ispita s pitanjima i rezultatima.

```
class Exam(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100), nullable=False)
    code = db.Column(db.String(6), unique=True, nullable=False)
    created_by = db.Column(db.Integer, db.ForeignKey('user.id'),
nullable=False)
    active = db.Column(db.String(20), nullable=False,
default='not_started')
    questions = db.relationship('Question', backref='exam', lazy=True)
    scores = db.relationship('Score', backref='exam', lazy=True)

    def __init__(self, name, created_by):
        self.name = name
        self.created_by = created_by
        self.code = self.generate_unique_code()

    def generate_unique_code(self):
        import random
        import string
        code = ''.join(random.choices(string.digits, k=6))
        while Exam.query.filter_by(code=code).first():
            code = ''.join(random.choices(string.digits, k=6))
        return code
```

4.2.3 Model pitanja

Model *Question* definira strukturu pitanja unutar ispita. Svako pitanje povezano je s određenim ispitom putem stranog ključa, a sadrži tekst pitanja, četiri ponuđene opcije i točan odgovor. Također, svaki odgovor ima pridruženi broj bodova koji će učenik dobiti ako odgovori točno.

```
class Question(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    exam_id = db.Column(db.Integer, db.ForeignKey('exam.id'),
nullable=False)
    question_text = db.Column(db.String(200), nullable=False)
    option_a = db.Column(db.String(100), nullable=False)
    option_b = db.Column(db.String(100), nullable=False)
    option_c = db.Column(db.String(100), nullable=False)
    option_d = db.Column(db.String(100), nullable=False)
    correct_answer = db.Column(db.String(1), nullable=False)
    points = db.Column(db.Integer, nullable=False)
```

4.2.4 Model bodovanja

Model *Score* koristi se za praćenje rezultata učenika. Povezan je s ispitom i učenikom putem stranih ključeva te bilježi broj bodova koje je učenik ostvario tijekom ispita. Ovaj model također implementira metodu **update_score** koja se koristi za ažuriranje bodova učenika nakon svakog točnog odgovora.

```
class Score(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    exam_id = db.Column(db.Integer, db.ForeignKey('exam.id'),
nullable=False)
    student_id = db.Column(db.Integer, db.ForeignKey('user.id'),
nullable=False)
    score = db.Column(db.Integer, nullable=False, default=0)

    @staticmethod
    def update_score(exam_id, student_id, points):
        score_entry = Score.query.filter_by(exam_id=exam_id,
student_id=student_id).first()
        if score_entry:
            score_entry.score += points
        else:
            score_entry = Score(exam_id=exam_id, student_id=student_id,
score=points)
            db.session.add(score_entry)
        db.session.commit()
```

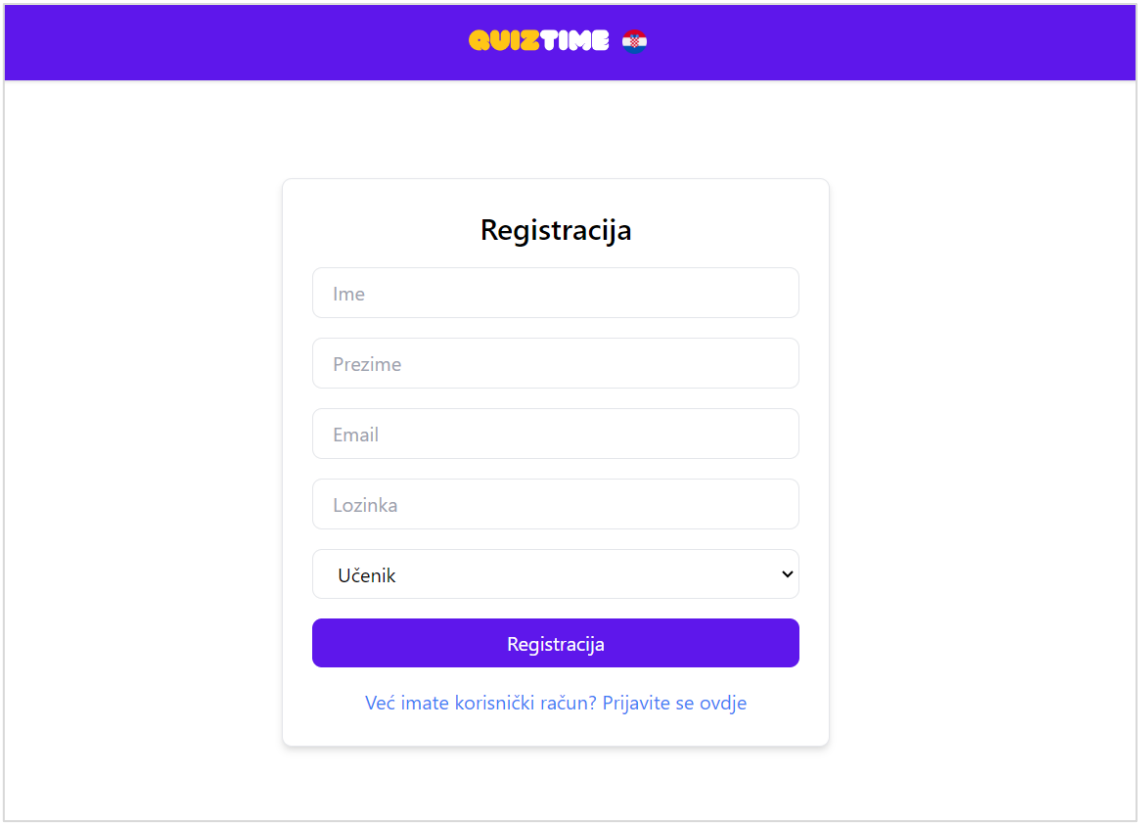
4.3 Registracija korisnika

Korisnici mogu kreirati nove račune putem registracijskog obrasca. Prilikom registracije, korisnici moraju unijeti osnovne podatke poput imena, prezimena, email adrese, lozinke te odabrati ulogu (nastavnik ili učenik). Lozinka se nikada ne pohranjuje u svom izvornom obliku; umjesto toga, koristi se Werkzeug sigurnosna funkcija koja generira hash lozinke, čime se osigurava sigurnost pohranjenih podataka. Npr. lozinka „o5tdQ347“ će u bazi biti spremljena kao:

```
„pbkdf2:sha256:260000$f8fQDPFdcqoHEWN6$7986b69b44822c643499615ae64788c33b4
e08b891cd260d07649d3d150ae1f6“
```

Registracijski obrazac provodi nekoliko razina validacije. Uz osnovnu provjeru ispravnosti email adrese, lozinka mora ispunjavati sigurnosne kriterije: mora sadržavati najmanje jedno veliko i jedno malo slovo, broj te biti dulja od osam znakova. Također, aplikacija provjerava je li email adresa već registrirana u sustavu. Ruta (*engl.* route) za registraciju prikazana je u nastavku, a izgled sučelja na slici 14.

```python
@bp.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        first_name = request.form['first_name']
        last_name = request.form['last_name']
        email = request.form['email']
        password = request.form['password']
        role = request.form['role']

        if not re.match(r'^[A-Za-z]+$', first_name):
            flash('First name can only contain letters.', 'danger')
            return redirect(url_for('main.register'))
        if not re.match(r'^[A-Za-z]+$', last_name):
            flash('Last name can only contain letters.', 'danger')
            return redirect(url_for('main.register'))
        if not re.match(r'^[^\s@]+@[^\s@]+\.[^\s@]+$', email):
            flash('Please enter a valid email address.', 'danger')
            return redirect(url_for('main.register'))
        if not re.match(r'^(?=.*[a-z])(?=.*[A-Z])(?=.*\d).{8,}$',
password):
            flash('Password must be at least 8 characters long and contain
at least one uppercase letter, one lowercase letter, and one number.',
'danger')
            return redirect(url_for('main.register'))

        if User.query.filter_by(email=email).first():
            flash('Email is already registered.', 'danger')
            return redirect(url_for('main.register'))

        user = User(first_name=first_name, last_name=last_name,
email=email, role=role)
        user.set_password(password)
        db.session.add(user)
        db.session.commit()
        flash('Registration successful! Please log in.', 'success')
        return redirect(url_for('main.login'))
    return render_template('register.html')
```



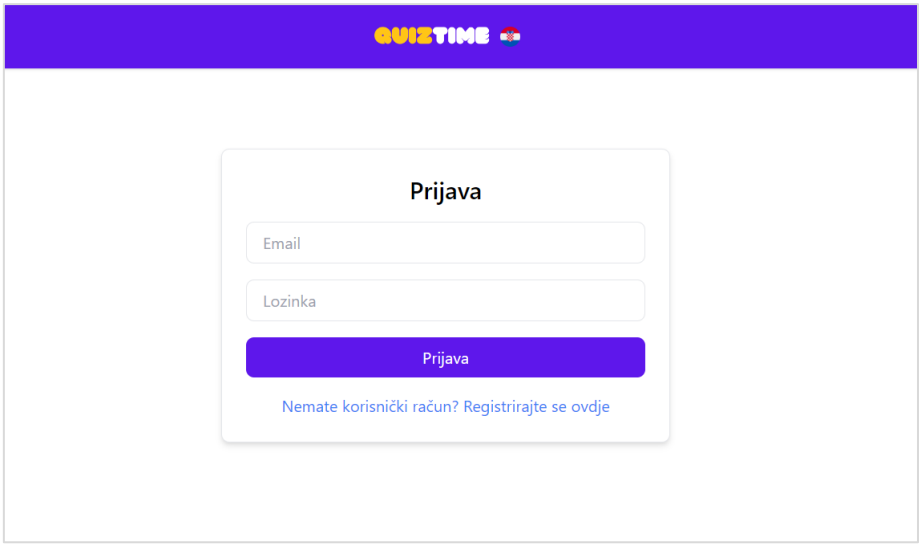
***Slika* 14.** Izgled sučelja registracijskog obrasca.

4.4 Prijava korisnika

Nakon što je korisnik uspješno registriran, može se prijaviti u aplikaciju unosom email adrese i lozinke. Tijekom procesa prijave, aplikacija provjerava je li korisnička lozinka točna koristeći prethodno hashiranu verziju pohranjene lozinke. Ako su uneseni podaci ispravni, korisnik se prijavljuje u sustav, a njegova sesija postaje aktivna. U nastavku se nalazi ruta ove funkcionalnosti:

```
@bp.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']
        user = User.query.filter_by(email=email).first()
        if user and user.check_password(password):
            login_user(user)
            return redirect(url_for('main.index'))
        flash('Invalid email or password.', 'danger')
    return render_template('login.html')
```

Aplikacija koristi Flask-Login ekstenziju kako bi upravljala sesijama i autentifikacijom korisnika. Ova ekstenzija omogućuje jednostavno postavljanje funkcija za učitavanje korisnika (*engl.* user loader), čime se osigurava identifikacija korisnika u svakom zahtjevu unutar sesije. Na slici 15. prikazan je izgled sučelja za prijavu korisnika.



***Slika* 15.** Izgled sučelja za prijavu korisnika.

4.5 Nadzorna ploča

Nadzorna ploča je centralno mjesto za nastavnike na koje su automatski preusmjereni nakon što se prijave u aplikaciju. Ona omogućuje nastavnicima pregled svih ispita koje su kreirali, kao i opcije za upravljanje tim ispitima. U ovoj sekciji bit će prikazane sve ključne funkcionalnosti dostupne na nadzornoj ploči, uključujući kreiranje novih ispita, uređivanje postojećih, brisanje ispita i pregled rezultata ispita.

4.5.1 Prikaz ispita

Nakon prijave, nastavniku se prikazuju svi ispiti koje je do sada kreirao. Svaki ispit prikazan je s osnovnim informacijama, uključujući naziv ispita i dinamičnu tipku (*engl.* button) koja ovisi o statusu ispita. Nastavnik može odabrati ispit za daljnje upravljanje, uključujući uređivanje, brisanje ili pregled rezultata ispita. Sučelje je prikazano na slici 16.

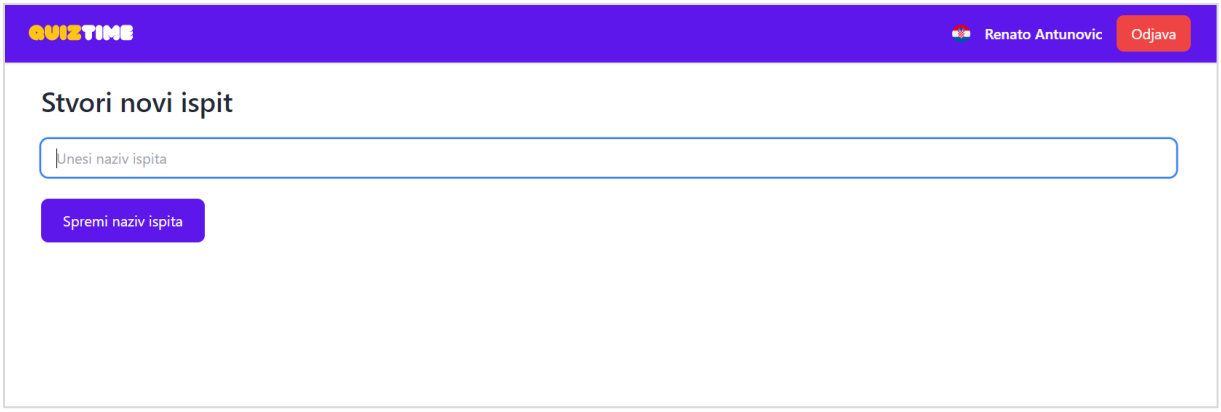
Tipke koje će se prikazati u ovisnosti o statusu ispita:

- **Čekaonica:** otvaranje ispitne čekaonice ukoliko ispit još nije pokrenut ("not_started")
- **Zaustavi ispit:** zaustavljanje ispita ukoliko je ispit pokrenut ("started")
- **Rezultati:** prikaz konačnih rezultata ispita ukoliko je nastavnik zaustavio ispit ("stopped")

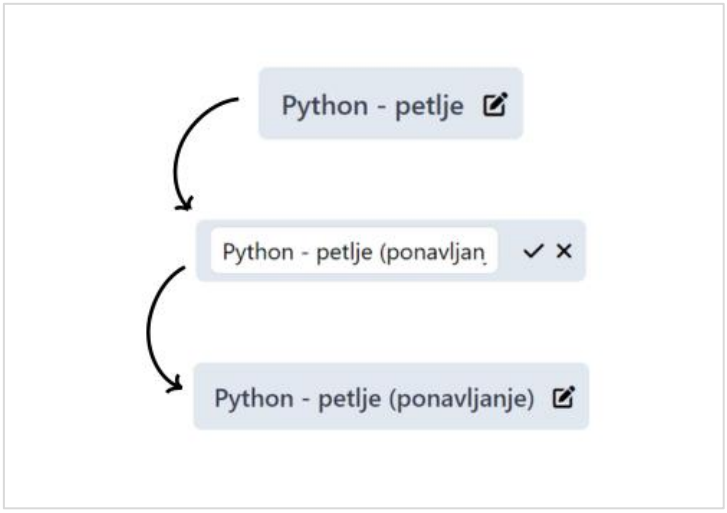
***Slika* 16.** Primjer izgleda korisničkog sučelja nadzorne ploče za nastavnika.

4.5.2 Stvaranje novog ispita i unos pitanja

Na nadzornoj ploči nastavnici imaju mogućnost stvaranja novog ispita. Klikom na tipku *Stvori novi ispit*, otvara se stranica **create_exam.html** gdje nastavnik može unijeti naziv ispita (slika 17). Nakon unosa naziva ispita, sustav automatski preusmjerava korisnika na stranicu **exam_data.html**, gdje nastavnik može upravljati pitanjima vezanim uz ispit. Na ovoj stranici u gornjem lijevom kutu prikazuje se naziv ispita, koji se može urediti putem opcije za uređivanje (slika 18).

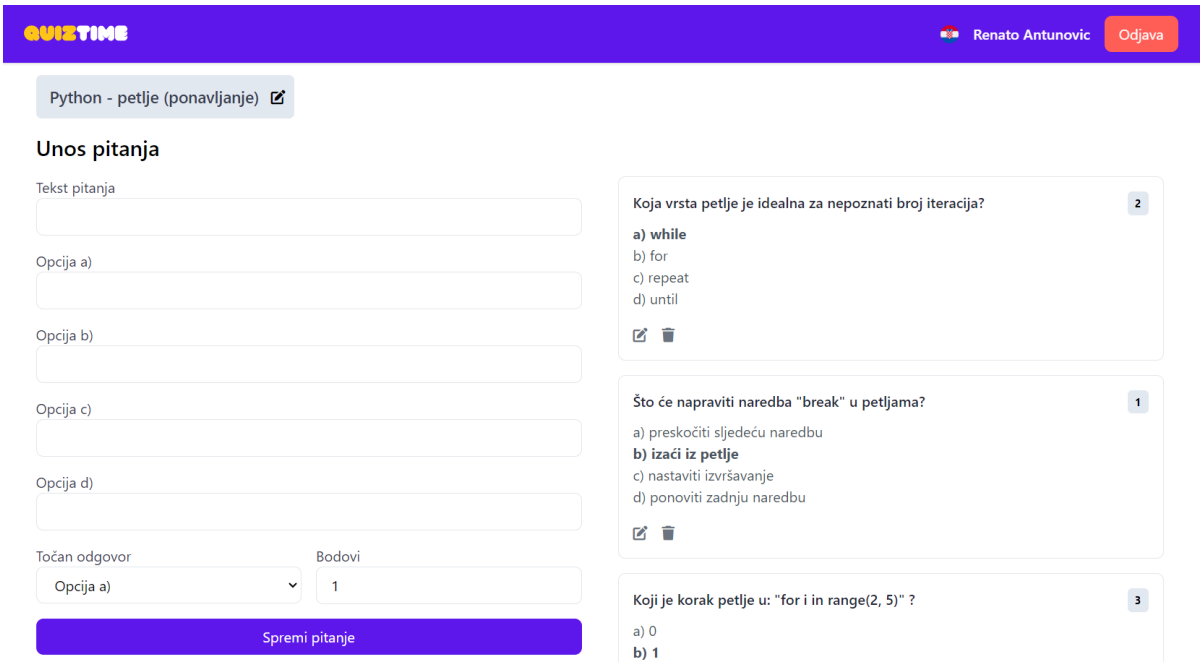


***Slika* 17.** Sučelje za kreiranje novog ispita s poljem za unos naziva ispita.

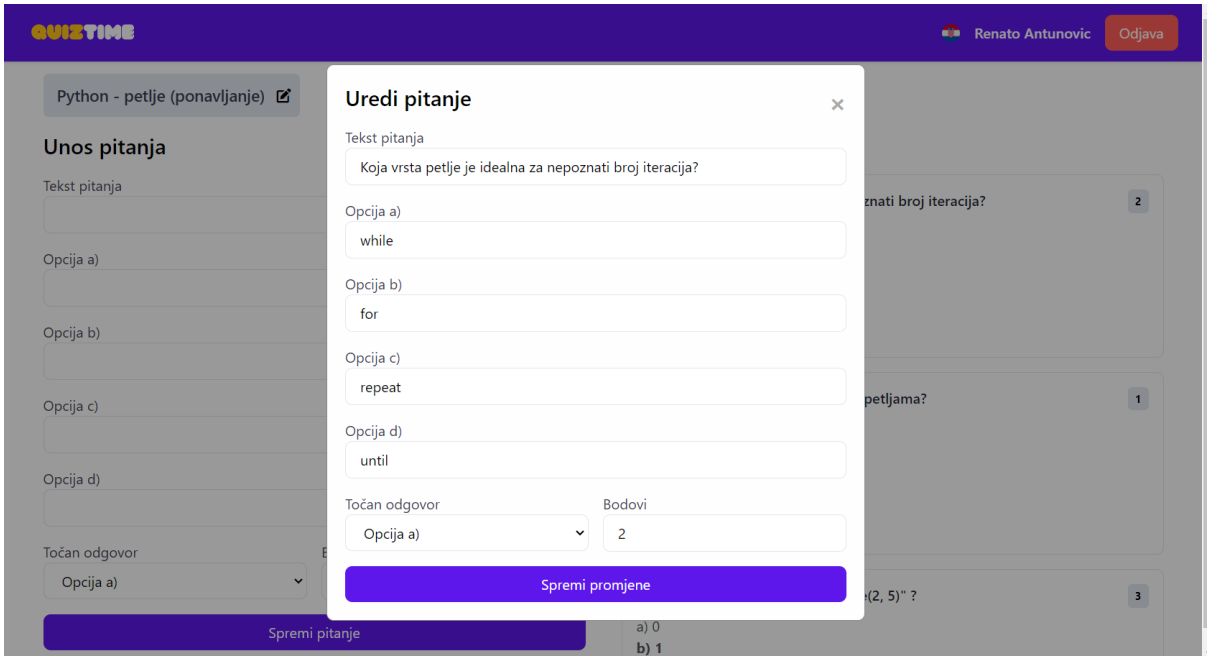


***Slika* 18.** Prikaz naziva ispita, uz mogućnost uređivanja putem opcije za uređivanje.

S lijeve strane ekrana nalazi se obrazac za unos pitanja, dok se s desne strane prikazuje lista svih dosad dodanih pitanja (slika 19). Svako pitanje prikazano je s opcijama za uređivanje i brisanje. Klikom na opciju za uređivanje pitanja otvara se modal s obrascem za uređivanje (slika 20), dok opcija za brisanje trajno uklanja pitanje iz ispita.



***Slika* 19.** Sučelje s obrascem za unos pitanja i pregled dodanih pitanja, s opcijama za uređivanje i brisanje.



***Slika* 20.** Modal za uređivanje pitanja.

4.5.3 Uređivanje ispita

Klikom na opciju za uređivanje ispita s nadzorne ploče, nastavnik također dolazi na stranicu **exam_data.html**. Ova stranica nudi iste mogućnosti kao i prilikom inicijalnog unosa pitanja u novi ispit. Nastavnik može uređivati naziv ispita, dodavati nova pitanja, te uređivati ili brisati već unesena pitanja, koristeći iste funkcionalnosti kao prilikom kreiranja ispita.

4.5.4 Brisanje ispita

Nastavnik ima i opciju brisanja ispita, što omogućuje trajno uklanjanje ispita i svih povezanih podataka, uključujući pitanja i rezultate učenika:

```
@bp.route('/delete_exam/<int:exam_id>', methods=['POST'])
@login_required
def delete_exam(exam_id):
    exam = Exam.query.get_or_404(exam_id)
    try:
        Question.query.filter_by(exam_id=exam.id).delete()
        db.session.delete(exam)
        db.session.commit()
    except Exception as e:
        db.session.rollback()
    return redirect(url_for('main.professor_dashboard'))
```

4.6 Ispitna čekaonica

Ispitna čekaonica predstavlja "prostor" gdje se učenici mogu pridružiti ispitu prije nego što on započne. Nastavnik započinje otvaranjem čekaonice nakon što je kreirao ispit. Nakon otvaranja čekaonice, na nastavnikovom ekranu se prikazuje jedinstveni kôd ispita koji učenici koriste kako bi pristupili ispitu.

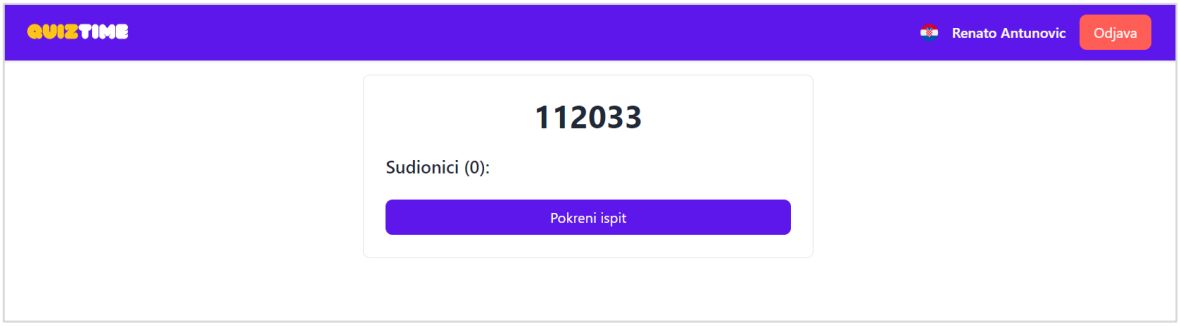
Tijekom ovog procesa, nastavnik na svom ekranu može vidjeti popis učenika koji su se pridružili ispitu. Taj popis se dinamično ažurira u stvarnom vremenu kako učenici pristupaju čekaonici. Koristi se već spomenuti SocketIO za komunikaciju između poslužitelja i klijenata:

```
socket.on('student_joined', function (data) {
        var studentsList = document.getElementById('students_list');
        if (studentsList) {
            var newStudent = document.createElement('li');
            newStudent.innerText = `${data.first_name} ${data.last_name}`;
            studentsList.appendChild(newStudent);

            // Update the student count
            var studentCountElement = document.querySelector('h2 span');
            if (studentCountElement) {
                studentCountElement.innerText =
studentsList.children.length;
            }
        }
    });
```

Ruta za otvaranje ispitne čekaonice:

```
@bp.route('/exam_lobby/<int:exam_id>')
@login_required
def exam_lobby(exam_id):
    exam = Exam.query.get_or_404(exam_id)
    students =
User.query.filter(User.id.in_(exam_participants.get(exam_id, []))).all()
    return render_template('exam_lobby.html', students=students,
exam=exam)
```

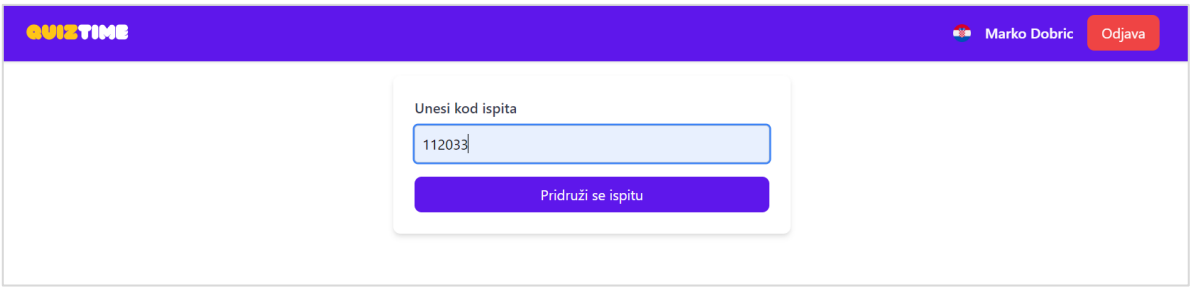
Nakon otvaranja ispitne čekaonice, nastavnik može vidjeti sve učenike koji su pristupili ispitu (slika 21). Učenici ne vide taj popis na svojim ekranima, ali se on prikazuje nastavniku kako bi mogao imati pregled svih sudionika (kojeg on može prikazati svima dijeljenjem svog ekrana putem projektora i sl.):



***Slika* 21.** Izgled sučelja ispitne čekaonice nakon što ju je nastavnik otvorio.

4.7 Pristupanje ispitu

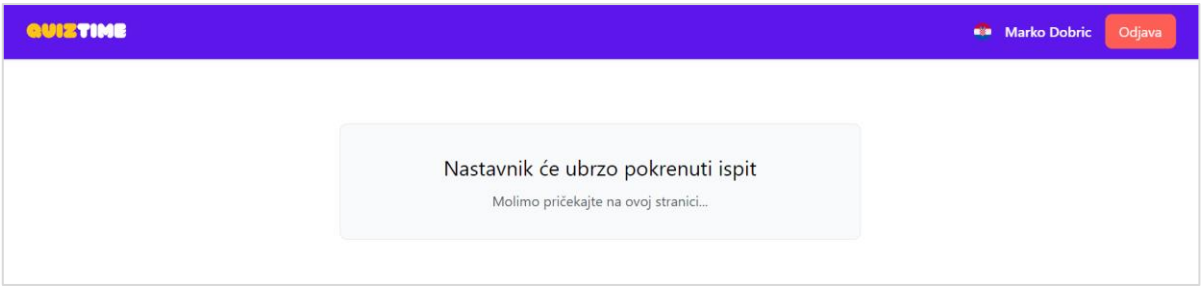
Nakon što nastavnik otvori čekaonicu, učenici mogu pristupiti ispitu unosom jedinstvenog kôda koji dobiju od nastavnika (slika 22). Kada učenik unese kôd, njegov ekran se preusmjerava na stranicu **loading_exam.html**, koja prikazuje poruku da će ispit uskoro započeti, što je prikazano na slici 23.



***Slika* 22.** Izgled sučelja za unos jedinstvenog kôda ispita.

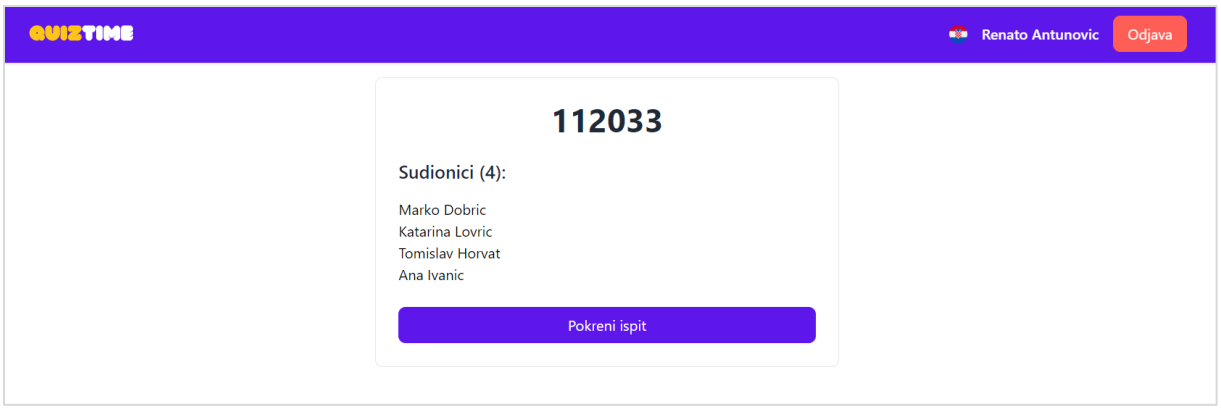
Ruta za pristupanje ispitu:

```
@bp.route('/join_exam', methods=['GET', 'POST'])
@login_required
def join_exam():
    if request.method == 'POST':
        exam_code = request.form['exam_code']
        exam = Exam.query.filter_by(code=exam_code).first()
        if exam:
            session['exam_id'] = exam.id
            exam_participants.setdefault(exam.id,
[]).append(current_user.id)

            # Emit the event to update other participants on the server-
side
            socketio.emit('student_joined', {
                'first_name': current_user.first_name,
                'last_name': current_user.last_name,
                'student_count': len(exam_participants[exam.id])
            }, room=exam.code)

            return render_template('loading_exam.html', exam=exam)
        else:
            flash('Invalid exam code.', 'danger')
            return redirect(url_for('main.join_exam'))
    return render_template('join_exam.html')
```



***Slika* 23.** Izgled sučelja stranice loading_exam.html

U međuvremenu, nastavnikov ekran prikazuje imena i prezimena učenika koji su se pridružili ispitu u stvarnom vremenu (slika 24). I ova funkcionalnost je podržana SocketIO tehnologijom, koja omogućava dinamično ažuriranje podataka o sudionicima ispita.



***Slika* 24.** Izgled sučelja ispitne čekaonice nakon što se nekoliko učenika pridružilo.

4.8 Pokretanje ispita

Kada se nastavnik uvjeri da su svi učenici pristupili ispitu, on može pokrenuti. Nakon što ispit započne, učenici su preusmjereni na stranicu za polaganje ispita (**take_exam.html**), dok je nastavnik preusmjeren na stranicu za prikaz rang liste (**live_scoreboard.html**).

Pokretanje ispita se odvija preko POST zahtjeva koji obavještava poslužitelj da je ispit započeo. Koristi se SocketIO za obavještavanje svih sudionika ispita da je ispit započeo, čime se automatski preusmjeravaju na odgovarajuće stranice:

```
@bp.route('/start_exam/<int:exam_id>', methods=['POST'])
@login_required
def start_exam(exam_id):
    exam = Exam.query.get_or_404(exam_id)
    exam.active = 'started'
    db.session.commit()

    socketio.emit('exam_started', {
        'role': 'student',
        'redirect_url': url_for('main.take_exam', exam_id=exam.id,
_external=True),
        'exam_id': exam.id
    }, room=exam.code)

    return jsonify(success=True,
redirect_url=url_for('main.live_scoreboard', exam_id=exam.id,
_external=True))
```

4.9 Rješavanje ispita

Učenici pristupaju stranici za rješavanje ispita gdje im se pitanja prikazuju jedno po jedno (slika 25). Klikom na jedan od četiri ponuđena odgovora, učitava se sljedeće pitanje:

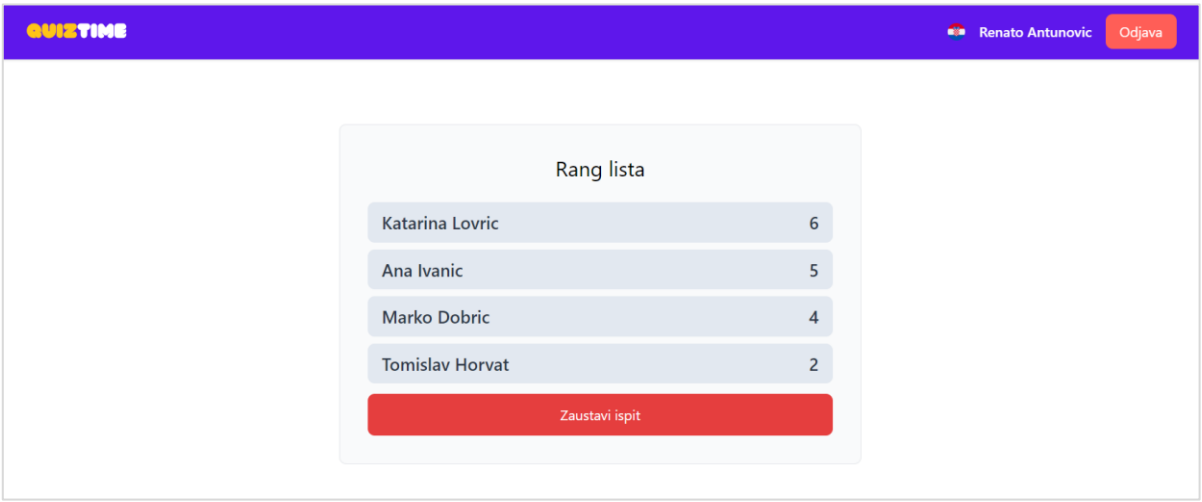
```
function loadQuestion(index) {
            var question = questions[index];
            document.getElementById('question_text').innerText =
question.question_text;
            var optionsContainer =
document.getElementById('options_container');
            optionsContainer.innerHTML = '';

            var options = ['a', 'b', 'c', 'd'];
            options.forEach(function (option) {
                var optionElement = document.createElement('div');
                optionElement.className = 'option-box';
                optionElement.innerText = question['option_' + option];
                optionElement.onclick = function () {
                    if (!examCompleted) {
                        submitAnswer(question.id, option);
                    }
                };
                optionsContainer.appendChild(optionElement);
            });
        }
```

***Slika* 25.** Izgled sučelja stranice take_exam.html

Nakon što učenik odgovori na pitanje, njegov rezultat se šalje poslužitelju, gdje se ažurira rezultat:

```
@bp.route('/submit_answer/<int:question_id>', methods=['POST'])
@login_required
def submit_answer(question_id):
    question = Question.query.get_or_404(question_id)
    answer = request.json['answer']
    correct = (answer == question.correct_answer)

    exam_code = question.exam.code

    if correct:
        points = question.points
        Score.update_score(exam_id=question.exam_id,
student_id=current_user.id, points=points)

        # Emit the score update event
        socketio.emit('score_update', {
            'student_id': current_user.id,
            'exam_code': exam_code
        }, room=exam_code)

    return jsonify(success=True)
```

Na slici 26. prikazan je izgled sučelja stranice **live_scoreboard.html**, odnosno prikaz rang liste uživo.



***Slika* 26.** Prikaz rang liste na stranici live_scoreboard.html

4.10 Kraj ispita

Nastavnik može prekinuti ispit klikom na opciju *Zaustavi ispit*. Ova funkcionalnost prekida ispit za sve učenike, a pitanja im više nisu vidljiva. Učenici su preusmjereni natrag na stranicu za unos kôda ispita (**join_exam.html**), dok nastavnik može pristupiti konačnim rezultatima:

```
@bp.route('/stop_exam/<int:exam_id>', methods=['POST'])
@login_required
def stop_exam(exam_id):
    exam = Exam.query.get_or_404(exam_id)
    exam.active = 'stopped'
    db.session.commit()
    socketio.emit('exam_stopped', room=exam.code)
    return redirect(url_for('main.exam_results', exam_id=exam_id))
```

Na klijentskoj strani, učenici su obaviješteni o završetku ispita i preusmjereni na početnu stranicu:

```
socket.on('exam_stopped', function () {
        document.getElementById("question_container").style.display =
'none';
        setTimeout(function () {
            alert('Nastavnik je zaustavio ispit. Molimo pričekajte konačne
rezultate.');
            window.location.href = '/join_exam';
        }, 100);
    });
```

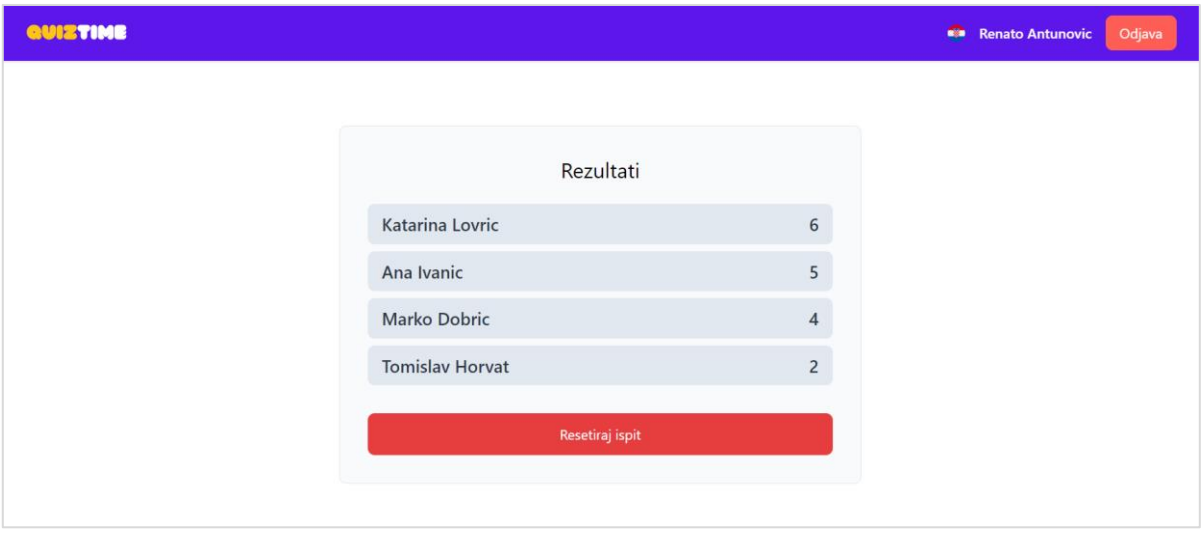
U slučaju da učenik odgovori na sva pitanja prije nego što nastavnik prekine ispit, sustav automatski bilježi završetak ispita i učeniku prikazuje poruku o završetku. Učenik više ne vidi pitanja i preusmjeren je natrag na početnu stranicu (join_exam.html):

```
if (currentQuestionIndex < questions.length) {
    loadQuestion(currentQuestionIndex);
} else {
    document.getElementById("question_container").style.display = 'none';
    setTimeout(function () {
        alert('Odgovorili ste na sva pitanja! Molimo pričekajte konačne
rezultate.');
        window.location.href = '/join_exam';}, 100);
}
```

4.11 Prikaz konačnih rezultata ispita

Nakon završetka ispita, na ekranu nastavnika se prikazuju konačni rezultati (slika 27). Stranica s rezultatima prikazuje rang listu učenika prema broju osvojenih bodova. Ova stranica se također može dohvatiti putem nadzorne ploče klikom na opciju *Rezultati*. U nastavku je prikazana ruta za prikaz konačnih rezultata ispita:

```
@bp.route('/exam_results/<int:exam_id>', methods=['GET'])
@login_required
def exam_results(exam_id):
    exam = Exam.query.get_or_404(exam_id)

    final_scores = (db.session.query(Score, User)
                    .join(User, Score.student_id == User.id)
                    .filter(Score.exam_id == exam_id)
                    .order_by(Score.score.desc())
                    .all())

    return render_template('exam_results.html', exam=exam,
final_scores=final_scores)
```



***Slika* 27.** Prikaz konačnih rezultata ispita na stranici exam_results.html

Podaci o rezultatima trajno su pohranjeni u bazi podataka i mogu se dohvatiti kasnije. Nastavnik ima opciju i resetirati ispit, čime se brišu rezultati i omogućava ponovno pokretanje ispita.

Isto tako je potrebno ukloniti učenike iz ispitne čekaonice koji su možda unijeli kôd ispita nakon što je ispit već pokrenut ili čak završen.

Ruta za funkcionalnost ponavljanja ispita nalazi se u nastavku:

```
@bp.route('/reset_exam/<int:exam_id>', methods=['POST'])
@login_required
def reset_exam(exam_id):
    exam = Exam.query.get_or_404(exam_id)
    Score.query.filter_by(exam_id=exam_id).delete()
    exam.active = 'not_started'

    if exam_id in exam_participants:
        del exam_participants[exam_id]

    db.session.commit()
    socketio.emit('exam_reset', room=exam.code)

    return redirect(url_for('main.professor_dashboard'))
```

4.12 Dvojezična podrška

Sučelje aplikacije može biti prikazano na dva različita jezika (hrvatski i engleski), čime omogućava prilagođavanje većem broju korisnika. Ova funkcionalnost je implementirana pomoću *session* varijabli koje pohranjuju odabir jezika svakog korisnika tijekom trajanja sesije.

Korisnici mogu promijeniti jezik sučelja odabirom između dvije opcije, koje su predstavljene u obliku britanske ili hrvatske zastave, a implementirane su kao tipka na desnoj strani navigacijske trake. Nakon što se jezik odabere, aplikacija koristi funkciju za prevođenje koja dohvaća odgovarajući prijevod za ključne riječi i fraze unutar aplikacije.

U nastavku je primjer definiranja prijevoda za nadzornu ploču unutar datoteke __init__.py:

```
translations = {
    'en': {
        'dashboard': 'Dashboard',
        'logout': 'Logout',
        'exam_lobby': 'Exam lobby',
        'stop_exam': 'Stop',
        'exam_results': 'Results',
        'edit_exam': 'Edit',
        'delete_exam': 'Delete',
        'create_exam': 'Create new exam'
    },
```

```
    'hr': {
        'dashboard': 'Nadzorna ploča',
        'logout': 'Odjava',
        'exam_lobby': 'Čekaonica',
        'stop_exam': 'Zaustavi',
        'exam_results': 'Rezultati',
        'edit_exam': 'Uredi',
        'delete_exam': 'Izbriši',
        'create_exam': 'Stvori novi ispit'
    }
}

def get_translation(key):
    lang = session.get('lang', 'en') #default: english
    return translations.get(lang, {}).get(key, key)
```

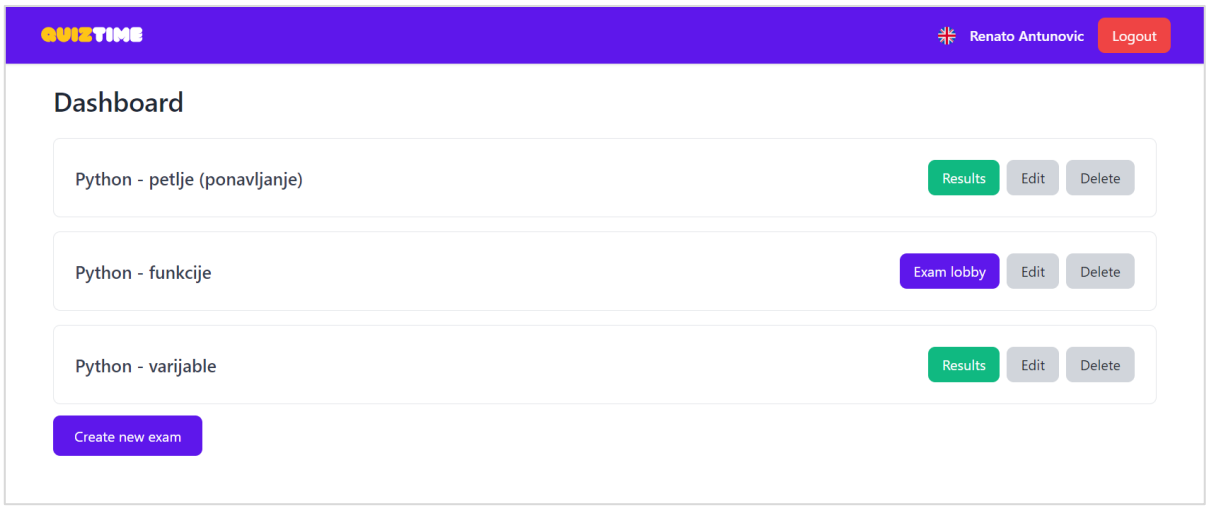
Nakon toga se može prikazati i kako se fraze „označavaju“ za prijevod unutar HTML datoteka:

```
<h2 class="text-3xl font-semibold text-gray-800 mb-6">{{
get_translation('dashboard') }}</h2>
```

Svaka ključna fraza u aplikaciji prolazi kroz ovu funkciju kako bi se prikazao odgovarajući prijevod na temelju odabranog jezika. Za odabir jezika zaslužna je sljedeća ruta koja ažurira *session* varijablu i preusmjerava korisnika natrag na prethodnu stranicu:

```
LANGUAGES = {
    'en': 'English',
    'hr': 'Croatian'
}

@bp.route('/set_language/<language>')
def set_language(language):
    if language not in LANGUAGES.keys():
        language = 'en'
    session['lang'] = language
    return redirect(request.referrer)
```

Slika 28. prikazuje kako nadzorna ploča izgleda nakon promjene jezika.



***Slika* 28.** Prikaz nadzorne ploče na engleskom jeziku

4.13 Odjava korisnika

Funkcionalnost odjave omogućava korisnicima da se u bilo kojem trenutku odjave iz sustava. Nakon što se korisnik odjavi, njegova *session* varijabla se briše, a korisnik je preusmjeren na stranicu za prijavu.

Odjava je implementirana pomoću Flask-Login ekstenzije, koja upravlja autentifikacijom i sesijama korisnika. Korisnici mogu pristupiti ruti za odjavu, koja pokreće funkciju za uklanjanje korisnika iz sesije i preusmjerava ga na početnu stranicu:

```
@bp.route('/logout')
@login_required
def logout():
    logout_user()
    session.clear()
    flash('You have been logged out successfully.', 'info')
    return redirect(url_for('main.login'))
```

Zaključak

Ovaj diplomski rad predstavlja spoj teorijske primjene igrifikacije u obrazovanju i tehničkog razvoja web aplikacije koja podržava online kvizove. Kroz proces razvoja aplikacije, tehnologije Flask i SocketIO pokazale su se ključnima za ostvarivanje glavnih funkcionalnosti, uključujući dinamičnu komunikaciju između klijenta i poslužitelja te rad u stvarnom vremenu. Flask je omogućio jednostavnu organizaciju i upravljanje aplikacijom, dok je SocketIO osigurao ažuriranje rezultata, automatizirano preusmjeravanje i prikaz rang liste u stvarnom vremenu.

I ostale tehnologije korištene u ovom radu doprinijele su fleksibilnosti i prilagodljivosti, omogućujući aplikaciji da zadovolji različite obrazovne potrebe. TailwindCSS je bio zaslužan za moderan i pregledan dizajn sučelja, SQLAlchemy je olakšao rad s bazom podataka, osiguravajući stabilnu pohranu i upravljanje informacijama o ispitima i rezultatima učenika, dok je WTForms pojednostavio kreiranje i validaciju obrazaca, što je olakšalo upravljanje unosom podataka u aplikaciju.

Daljnji razvoj aplikacije mogao bi uključivati nove funkcionalnosti koje bi dodatno unaprijedile njezinu prilagodljivost različitim obrazovnim potrebama, uključujući mogućnost integracije multimedijskih sadržaja ili detaljnije povratne informacije. Zaključno, primjena igrifikacije i digitalnih alata u obrazovanju će zasigurno nastaviti s razvojom, otvarajući nove mogućnosti za budućnost obrazovanja.

Literatura

[1] Raczkowski, F. (2014). It's all fun and games… A history of ideas concerning gamification. In *Proceedings of DiGRA 2013 Conference*. DiGRA.

[2] Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining gamification. In *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments* (pp. 9-15). ACM.

[3] Duarte, E., Pereira, P., Rebelo, F., & Noriega, P. (2014). A review of gamification for health-related contexts. In *Advances in Human Factors and Ergonomics* (pp. 742-753). Springer.

[4] Singh, H. K., & Verma, S. (2020). Gamification at workplace: Theories, constructs and conceptual frameworks. *ACIS 2020 Proceedings*, 24.

[5] Montenegro-Rueda, M., Fernández-Cerero, J., Mena-Guacas, A. F., & Reyes-Rebollo, M. M. (2023). Impact of gamified teaching on university student learning. *Education Sciences, 13*(5), 470.

[6] Gamification for learning: Strategies and examples. (2023). eLearning Industry. https://elearningindustry.com/gamification-for-learning-strategies-and-examples

[7] Gamification and game-based learning. (2022). Centre for Teaching Excellence. https://uwaterloo.ca/centre-for-teaching-excellence/catalogs/tip-sheets/gamification-and-game-based-learning

[8] Ouahbi, I., Darhmaoui, H., & Kaddari, F. (2021). Gamification approach in teaching web programming courses in PHP: Use of KAHOOT application. *International Journal of Modern Education and Computer Science, 13*(2), 33-39.

[9] Wang, A., & Tahir, R. (2020). The effect of using Kahoot! for learning: A literature review. *Computers & Education, 149*, 103818.

[10] Licorish, S. A., Owen, H. E., Daniel, B., & others. (2018). Students' perception of Kahoot!'s influence on teaching and learning. *Research and Practice in Technology Enhanced Learning, 13*(9).

[11] Zhan, Z., He, L., Tong, Y., Liang, X., Guo, S., & Lan, X. (2022). The effectiveness of gamification in programming education: Evidence from a meta-analysis. *Computers and Education: Artificial Intelligence, 3*, 100096.

[12] Villegas, C., & Lemos Agüero, N. (2023). The gamification of e-learning environments for learning programming. *JOIV: International Journal on Informatics Visualization*, *7*(2), 455-462.

[13] Fotaris, P., Mastoras, T., Leinfellner, R., & Rosunally, Y. (2016). Climbing up the leaderboard: An empirical study of applying gamification techniques to a computer programming class. *Electronic Journal of e-Learning*, *14*(2), 95-110.

[14] Acadecraft. (2024). Gamification strategies: 5 effective ways to use gamification as a teaching strategy. https://www.acadecraft.com/blog/gamification-strategies-5-effective-ways-to-use-gamification-as-a-teaching-strategy/