Web-aplikacija za razvoj podatkovne pismenosti djece mlađe školske dobi

Šantek, Marija

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 726

WEB-APLIKACIJA ZA RAZVOJ PODATKOVNE PISMENOSTI DJECE MLAĐE ŠKOLSKE DOBI

Marija Šantek

Zagreb, veljača 2025.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 726

WEB-APLIKACIJA ZA RAZVOJ PODATKOVNE PISMENOSTI DJECE MLAĐE ŠKOLSKE DOBI

Marija Šantek

Zagreb, veljača 2025.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 30. rujna 2024.

DIPLOMSKI ZADATAK br. 726

Pristupnica:	**Marija Šantek (0036517738)**
Studij:	Računarstvo
Profil:	Računalno inženjerstvo
Mentorica:	izv. prof. dr. sc. Ivana Bosnić

Zadatak: **Web-aplikacija za razvoj podatkovne pismenosti djece mlađe školske dobi**

Opis zadatka:

Istražiti principe razvoja korisničkih sučelja namijenjenih djeci mlađe školske dobi te metode poučavanja podatkovne pismenosti (data literacy) za djecu navedenog uzrasta. Pažnju posvetiti svim koracima životnog ciklusa podataka, od prikupljanja podataka preko analize i vizualizacije do donošenja zaključaka i korištenja. Za djecu mlađe školske dobi razviti web-aplikaciju koja omogućuje stvaranje skupova podataka jednostavnim unosom brojčanih i tekstualnih vrijednosti, geografskih lokacija ili GPX-traka kretanja. Djeca podatke trebaju unositi u dijeljeni skup podataka. Unesene podatke aplikacija treba vizualizirati korištenjem djeci razumljivih i privlačnih grafičkih prikaza (oblaci riječi, grafovi i karte). Aplikacija također treba omogućiti prikaz jednostavne analize svojstava skupova podataka (najmanja i najveća vrijednost, prosjek, filtriranje po stupcima, usporedba grafova pojedinačnih korisnika u grupnom radu i slično). Web-aplikacija treba podržavati rad sa skupovima podataka u "virtualnim sobama" kojima se može pristupiti unosom lozinke, a korisnike treba razlikovati nadimcima, bez klasične autentifikacije. Dizajn aplikacije treba biti responzivan radi korištenja na raznorodnim uređajima, višejezičan, a korisničko sučelje i iskustvo trebaju biti prilagođeni djeci mlađe školske dobi. Razvijeno rješenje potrebno je testirati i diskutirati.

Rok za predaju rada: 14. veljače 2025.

Hvala mom Matiji

Sadržaj

1. Uvod

U digitalnom dobu u kojem živimo, podatkovna pismenost (engl. data literacy) postaje ključna vještina, čak i za najmlađe članove društva. Podatkovna pismenost podrazumijeva sposobnost razumijevanja, analize i interpretacije podataka, što omogućuje donošenje odluka temeljenih na informacijama. Dok se važnost ovih vještina često naglašava u kontekstu odraslih i profesionalaca, postoji rastuća potreba za njihovim razvojem već u ranom djetinjstvu. Djeca mlađe školske dobi, koja su prirodno radoznala i spremna na učenje kroz igru i interakciju, predstavljaju savršenu populaciju za rano uvođenje koncepta rada s podacima.

Razvoj web-aplikacija prilagođenih djeci mlađe školske dobi otvara mogućnosti za kreativno i intuitivno poučavanje podataka. Uzimajući u obzir kognitivne, vizualne i motoričke sposobnosti djece, potrebno je primijeniti specifične principe dizajna korisničkog sučelja, kao i metode poučavanja koje potiču igru, suradnju i praktično učenje. Cilj je omogućiti djeci jednostavan unos podataka, njihovu vizualizaciju kroz zanimljive grafičke prikaze i provođenje osnovne analize, čime se potiče razumijevanje svih koraka životnog ciklusa podataka – od prikupljanja, preko analize i vizualizacije, do donošenja zaključaka.

Ovaj rad istražuje osnovne principe dizajna korisničkih sučelja za djecu mlađe školske dobi, metode poučavanja podatkovne pismenosti i korake potrebne za razvoj web-aplikacije koja ispunjava navedene zahtjeve. Razvijena aplikacija omogućava stvaranje i vizualizaciju skupova podataka, suradnju u "virtualnim sobama" i jednostavnu analizu podataka, a sve uz jednostavno, prilagođeno i intuitivno korisničko sučelje.

2. Teorijska osnova i istraživanje

Tijekom vremena digitalna uključenost i podatkovna pismenost kontinuirano su se mijenjale kao odgovor na razvoj digitalnih tehnologija i medija. Ovaj dinamičan odnos između društva i tehnologije oblikovao je način na koji različite društvene skupine pristupaju, koriste i razumiju digitalne resurse. S napretkom tehnologije, pitanja digitalne pismenosti i uključenosti postala su sve važnija u kontekstu društvene jednakosti i pristupa mogućnostima koje pruža digitalno doba[1]. Podatkovne tehnologije sve se više integriraju u različite sektore društva, uključujući obrazovanje. Iako se pojam "podataka" može tumačiti na različite načine, u kontekstu podatkovne pismenosti on se najčešće odnosi na kvantificirani digitalni materijal koji služi za prikazivanje stvarnih pojava putem matematičkih reprezentacija[2]. Takvi podaci omogućuju analizu, vizualizaciju i donošenje odluka temeljenih na informacijama, što ih čini ključnim elementom suvremenog obrazovanja i istraživanja. Razvoj alata koji omogućuju razumijevanje i obradu podataka postaje sve važniji, osobito u školskom okruženju, gdje učenici trebaju usvojiti vještine rada s podacima od najranije dobi.

2.1. Poučavanje podatkovne pismenosti

Podatkovna pismenost odnosi se na znanje i vještine koje omogućuju pristupanje, interpretiranje, kritičko obrađivanje, upravljanje i etičko korištenje istraživačkih podataka [3]. U suvremenom društvu, u kojem tehnologija omogućuje pristup velikoj količini podataka, sposobnost učinkovite obrade i primjene tih informacija postaje ključna. Podatkovna pismenost uključuje razumijevanje kako pronaći relevantne podatke, procijeniti njihovu točnost i pouzdanost te ih koristiti za donošenje informiranih odluka. Ove vještine su posebno važne u poslovnom svijetu, gdje podatkovna pismenost omogućuje učinkovitije donošenje odluka i postizanje uspjeha. Razvoj podatkovne pismenosti zah-

tijeva sustavan pristup, uključujući edukaciju i praksu u analizi podataka, te pridržavanje etičkih standarda u njihovom korištenju[4].

Kod djece mlađe školske dobi, razvoj podatkovne pismenosti uključuje učenje osnovnih koncepata rada s podacima na način koji je prilagođen njihovim kognitivnim sposobnostima. Vrlo jednostavni skupovi podataka mogu pružiti dovoljnu kompleksnost za poučavanje koncepata rada s podacima[5]. Aktivnosti poput prikupljanja informacija o vremenskim prilikama, vođenja evidencije o različitim potrošnjama ili analiziranja rezultata školskih anketa samo su neki od primjera prikupljanja, organiziranja i vizualiziranja podataka.

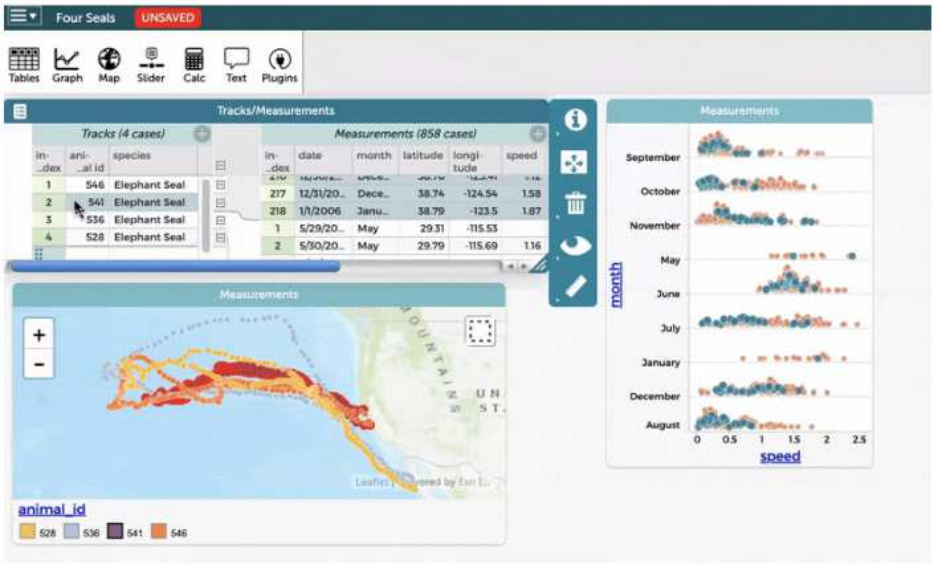
2.2. Korisnička sučelja za djecu mlađe školske dobi

U tipičnom školskom sustavu, skupovi podataka obično se ili posuđuju iz konteksta koji su daleko izvan iskustava učenika ili se izmišljaju u obrazovne svrhe[6]. Međutim, učenicima je puno lakše razumjeti obradu podataka kada rade s informacijama iz vlastite okoline. Takve podatke mogu bilježiti ručno ili unositi u digitalne edukacijske alate, što im omogućuje praktično učenje kroz stvarne primjere. Moderni edukacijski alati osmišljeni su kako bi djeci približili svijet podataka na interaktivan i intuitivan način, potičući ih ne samo na razvoj tehničkih vještina, već i na istraživanje, igru i kritičko razmišljanje. Pristup učenju kroz igru pokazao se posebno učinkovit za ovu dobnu skupinu, jer kombinira elemente zabave i obrazovanja, čime se stvara pozitivno iskustvo učenja. Djeca kroz ovakve alate mogu steći osnovna znanja o prikupljanju podataka, njihovoj analizi i vizualizaciji, ali i razumjeti kako se ti procesi mogu primijeniti u svakodnevnim situacijama.

Sučelja ovih alata često koriste jednostavne grafičke prikaze, intuitivne kontrole i aktivnosti prilagođene uzrastu korisnika. Ovakav način učenja osnažuje učenike za interakciju i razumijevanje podataka koji su relevantni za njihova osobna iskustva[7]. Kroz interaktivne vizualizacije i praktične primjere, djeca imaju priliku vidjeti kako podaci mogu biti korišteni za otkrivanje obrazaca, usporedbu informacija ili rješavanje problema. Ovakvi alati otvaraju vrata novim mogućnostima u obrazovanju, potičući djecu na istraživanje i kreativno razmišljanje, dok istovremeno jačaju temeljne vještine potrebne za rad s podacima.

Primjeri nekih sučelja koja imaju navedene karakteristike su:

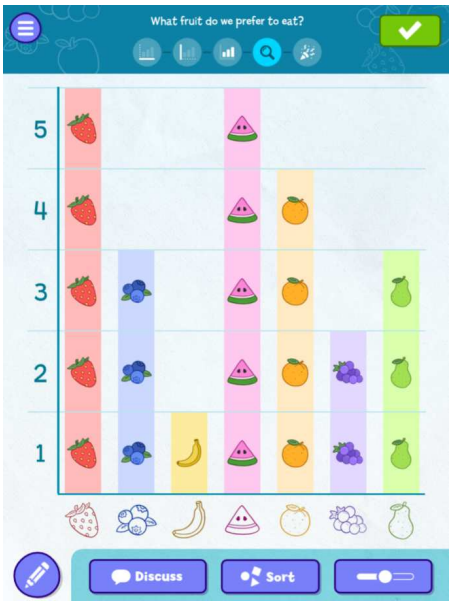
1. CODAP[1] (*Common Online Data Analysis Platform*) je besplatni alat koji se često koristi u obrazovnim kontekstima za osnovnu analizu podataka. Iako nije specifično osmišljen za mlađu djecu, njegova jednostavna vizualizacija i mogućnost prilagodbe čine ga korisnim za uvod u podatkovnu pismenost. Djeca mogu učitavati skupove podataka (npr. podaci o vremenu, broj zabilježenih ptica u parku), a sučelje im omogućuje da povlačenjem podataka stvaraju osnovne grafove i tablice.



Slika 2.1. CODAP

2. Data Toolbox[2] je aplikacija gdje, kroz jednostavne primjere i interaktivne aktivnosti, djeca uče kako prikupljati i analizirati podatke. Sučelje je strukturirano oko jednostavnih zadataka poput prebrojavanja objekata, stvaranja osnovnih grafova i donošenja zaključaka. Primjerice, djeca mogu unositi podatke o vremenu i gledati kako se stvaraju stupčasti grafovi ili linijski prikazi.

[1] Common Online Data Analysis Platform, https://codap.concord.org/
[2] Preschool Data Toolbox, https://oceansofdata.org/our-work/preschool-data-toolbox



Slika 2.2. Data Toolbox

3. Kids in Data[3] je interaktivna platforma još uvijek u razvoju osmišljena kako bi djeci na zabavan i razumljiv način približila svijet podataka. Kroz jednostavne i prilagođene aktivnosti, djeca mogu učiti kako prikupljati, analizirati i vizualizirati podatke koristeći stvarne primjere iz svakodnevnog života. Platforma koristi privlačne grafičke prikaze, igre i praktične zadatke kako bi potaknula djecu na istraživanje i razumijevanje podataka, razvijajući pritom njihovu radoznalost i temeljne analitičke vještine.

Slika 2.3. Kids in Data

[3]Kids in data, https://kidsindata.com/

3. Specifikacija web-aplikacije

3.1. Specifikacija aplikacije

Specifikacija aplikacije definira funkcionalnosti i tehničke zahtjeve koji su ključni za ostvarivanje ciljeva projekta. Glavni cilj aplikacije je pružiti djeci mlađe školske dobi interaktivan alat za razvoj podatkovne pismenosti, fokusirajući se na prikupljanje, analizu i vizualizaciju podataka na pristupačan i intuitivan način.

3.1.1. Osnovne funkcionalnosti aplikacije

Prikupljanje podataka

Aplikacija omogućuje djeci unos različitih vrsta podataka, uključujući:

- **Brojčane vrijednosti (npr. mjerenja ili rezultati)**: Djeca mogu unositi brojčane podatke koji mogu predstavljati različite vrste mjerenja, poput visine, temperature, vremena ili broja predmeta. Ove brojčane vrijednosti mogu se unositi putem jednostavnih obrazaca koji sadrže polja za unos broja, što omogućuje djeci da se upoznaju s osnovnim konceptima brojanja, mjerenja i usporedbe podataka. Na primjer, dijete može unijeti svoje rezultate u različitim zadacima (broj skokova, brzinu trčanja, broj koraka i sl.), a aplikacija će omogućiti usporedbu tih podataka s drugim korisnicima.

- **Tekstualne vrijednosti (npr. imena, naslovi ili opisi)**: Ovaj tip unosa omogućuje djeci unos informacija u tekstualnom obliku, kao što su njihova imena, nazivi mjesta ili jednostavni opisi. Djeca mogu upisivati podatke u obrasce koji im omogućuju unos teksta, kao što su nazivi svojih omiljenih predmeta, imena svojih prijatelja ili članova obitelji. Takve vrijednosti prikazuju se u oblaku riječi.

- **Geografske lokacije (npr. označavanje mjesta na karti)**: Unos geografskih lokacija omogućuje djeci da označe mjesta na karti, što može uključivati gradove, škole, znamenitosti ili vlastite točke interesa. Ovaj tip unosa pruža djeci priliku da uče o geografiji i kartama.

- ***GPX*-trake kretanja (npr. za prikaz ruta ili staza)**: *GPX* (*GPS Exchange Format*) trake omogućuju djeci unos i vizualizaciju ruta ili staza kretanja koje su prešle, bilo da se radi o šetnji, biciklističkoj ruti ili putovanju. To su datoteke koje sadrže informacije o geografskoj lokaciji, brzini, visini i vremenskim oznakama koje učenici mogu jednostavno prenijeti u aplikaciju. Ove trake kretanja mogu se prikazivati na kartama, omogućujući djeci da vizualiziraju i analiziraju svoje aktivnosti, kao i da uspoređuju rute s drugim korisnicima na karti ili analiziraju podatke o svojoj vlastitoj aktivnosti.

Dijeljeni skupovi podataka

Svi uneseni podaci pohranjuju se u zajednički skup podataka kojem može pristupiti profesor koji je administrator same *"virtualne sobe"*. Pristup sobama omogućen je unosom lozinke, čime se osigurava privatnost i sigurnost rada.

Vizualizacija podataka

Aplikacija nudi raznovrsne načine vizualizacije podataka prilagođene djeci mlađe školske dobi:

- **Grafički prikazi (stupčasti i kružni grafovi)**: Stupčasti grafovi omogućuju usporedbu brojčanih podataka između različitih kategorija, dok kružni grafovi prikazuju udjele unutar cjeline. Djeca mogu jednostavno vizualizirati podatke, poput omiljenih sportova ili broja učenika koji preferiraju određene boje.

- **Oblaci riječi za analizu učestalosti pojmova**: Ova vizualizacija prikazuje najčešće korištene riječi tako da se one s većom učestalošću prikazuju većim fontom. Djeca mogu brzo prepoznati ključne pojmove iz unesenih podataka, primjerice najpopularnije hobije u razredu.

- **Interaktivne karte za prikaz geografskih informacija**: Djeca mogu označa-

vati lokacije, unositi podatke o mjestima ili učitavati GPS rute, što omogućuje vizualizaciju geografskih podataka. To može uključivati označavanje mjesta koja su posjetili ili usporedbu svojih kretanja na karti.

Analiza podataka

Aplikacija pruža jednostavne alate za analizu podataka, uključujući:

- Izračune osnovnih statističkih mjera (najmanja i najveća vrijednost, prosjek).
- Filtriranje i sortiranje podataka po različitim kategorijama.
- Usporedbu grafova i podataka pojedinačnih korisnika unutar grupe.

Anonimizacija odgovora učenika

Kako bi se zaštitila privatnost djece, aplikacija omogućuje uključivanje ili isključivanje anonimiziranih odgovora prilikom prikaza rezultata. Kada je anonimizacija uključena, odgovori učenika prikazuju se bez osobnih oznaka, čime se osigurava sigurno dijeljenje podataka u učionici ili tijekom grupnih analiza. Ova funkcionalnost posebno je korisna u edukacijskom okruženju gdje se želi potaknuti sudjelovanje bez straha od izloženosti ili usporedbe s drugima. Nastavnici imaju kontrolu nad postavkama prikaza, prilagođavajući ih specifičnim potrebama i osjetljivosti učenika.

Responzivnost i višejezičnost

Aplikacija je dizajnirana tako da pruža jednako korisničko iskustvo na različitim uređajima (računala, tableti, pametni telefoni). Uz to, podržava više jezika, omogućujući prilagodbu različitim jezičnim skupinama.

3.2. Razrada dizajna korisničkog sučelja

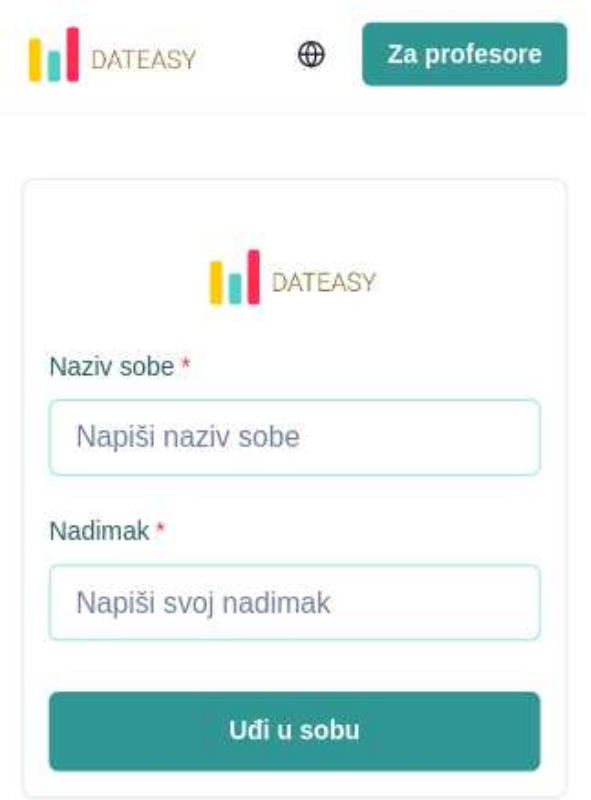
Dizajn korisničkog sučelja ključan je za osiguravanje intuitivne i privlačne interakcije s aplikacijom. S obzirom na ciljanog korisnika, djecu mlađe školske dobi, dizajn se temelji na načelima jednostavnosti, vizualne privlačnosti i prilagodbe uzrastu.

3.2.1. Glavne komponente sučelja

Dio komponenti sučelja prikazat će se primjerima za mobilne uređaje zbog bolje preglednosti. Bitno je naglasiti da se prikaz proporcionalno prilagođava svim rezolucijama ekrana. Aplikacija se sastoji od nekoliko glavnih komponenti koje definiraju korisničko sučelje.

Početni zaslon

Početni zaslon pruža opcije za odabir uloge korisnika (profesor ili učenik) te pristup virtualnim sobama putem unosa lozinke. Cilj je brzo uvesti korisnike u aplikaciju uz minimalne komplikacije.

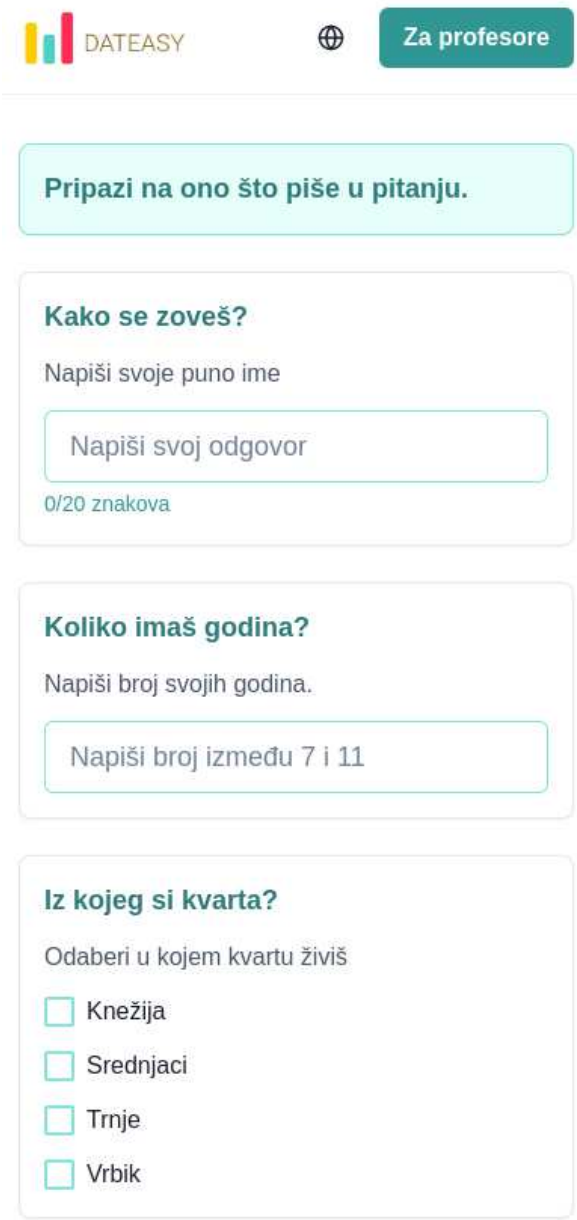


Slika 3.1. Početni prikaz za učenike

Unos i validacija podataka

Korisnici imaju priliku unositi podatke kroz jednostavne obrasce. Unutar obrazaca je implementirana i validacija. Primjeri mogućih tipova obrazaca su:

- Tekstualna polja za unos naziva ili opisa.
- Numerička polja za unos brojeva.
- Sučelje za unos geografskih lokacija putem interaktivne karte.
- Podršku za učitavanje *GPX*-datoteka s podacima o rutama.

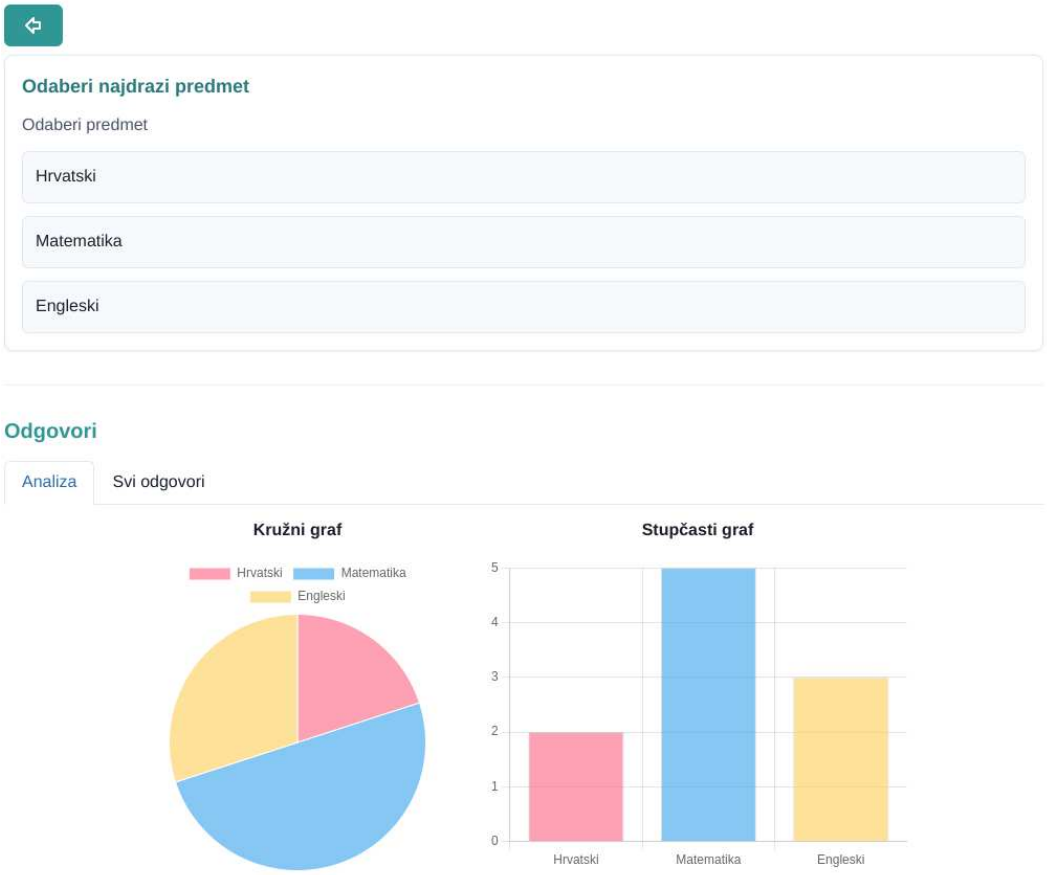


Slika 3.2. Prikaz zadataka za učenike

Vizualizacija podataka

Aplikacija pruža razne metode za vizualizaciju podataka:

- **Grafovi** – Jednostavni stupčasti i linijski grafovi za prikaz brojčanih podataka.
- **Oblaci riječi** – Prikaz učestalosti pojmova kroz oblake riječi.
- **Interaktivne karte** – Prikaz geografskih podataka i ruta.



Slika 3.3. Primjer prikaza odgovora u zadatku tipa višestruki odabir

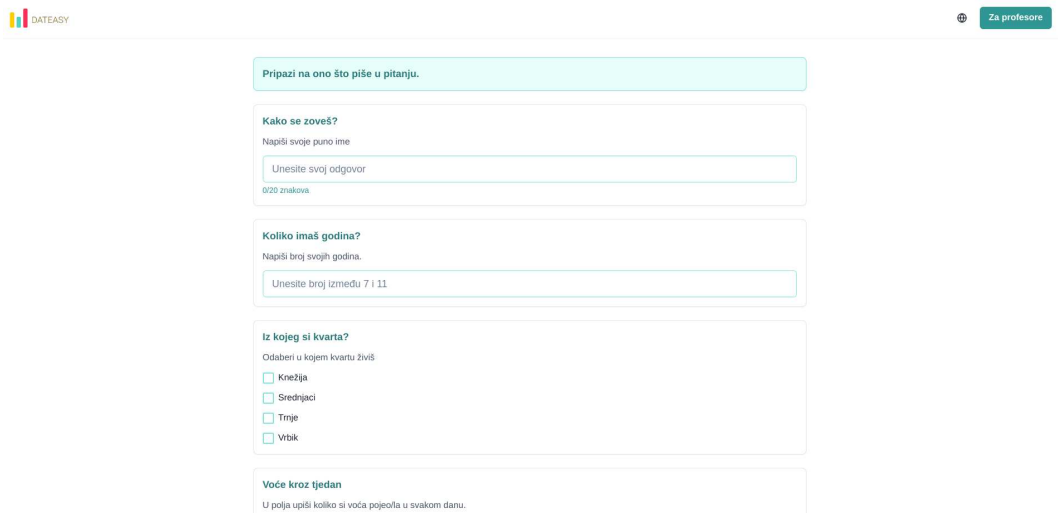
3.2.2. Korištenje boja i elemenata dizajna

Grafički dizajn ključan je za privlačenje dječje pažnje i olakšavanje interakcije s aplikacijom. Dizajn treba biti prilagođen njihovim kognitivnim sposobnostima i vizualnim preferencijama, s posebnim naglaskom na raspored elemenata, tipografiju i korištenje boja[8]. Vizualni identitet aplikacije temelji se na kombinaciji svijetlih i prijateljskih boja prilagođenih djeci[9]. Elementi dizajna uključuju:

- **Boje** – Svijetloplava, zelena i žuta koje potiču koncentraciju i osjećaj ugode.
- **Ikone i ilustracije** – Jednostavne i privlačne grafike koje olakšavaju navigaciju.
- **Prijelazi i animacije** – Diskretni efekti koji čine interakciju ugodnijom.

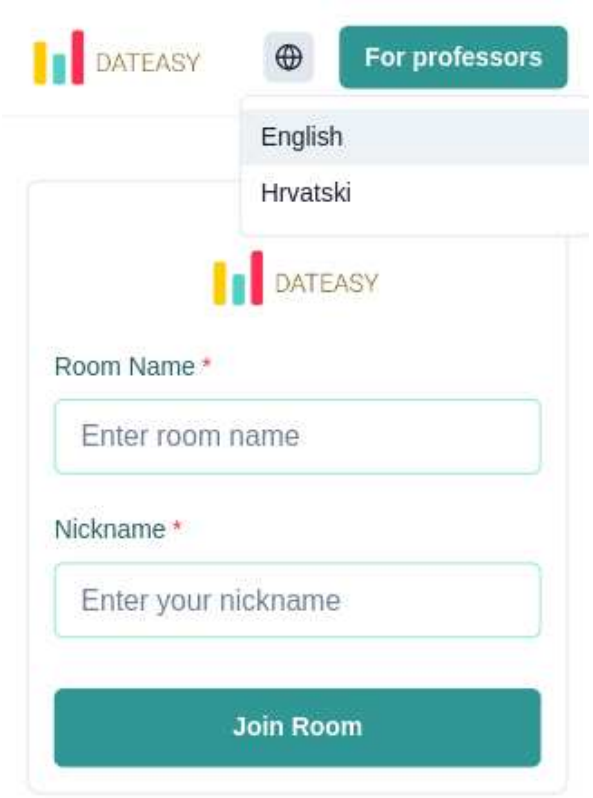
3.2.3. Responzivnost i višejezičnost

Korisničko sučelje dizajnirano je s naglaskom na responzivnost i višejezičnost. Responzivnost je implementirana kako bi aplikacija bila jednako funkcionalna i privlačna na računalima, tabletima i pametnim telefonima. Korisnici mogu jednostavno upravljati aplikacijom bez obzira na uređaje koje koriste.



Slika 3.4. Prikaz zadataka za učenike na ekranu visoke razlučivosti

Višejezičnost je implementirana kako bi aplikacija mogla imati što širu primjenu. Novi se jezici lako dodaju uz pomoć knjižnice specijalizirane za višejezičnost.



Slika 3.5. Primjer višejezičnosti

3.3. Opis funkcionalnih zahtjeva

Aplikacija pruža različite funkcionalnosti prilagođene dvjema glavnim ulogama: profesorima i učenicima. U ovom poglavlju opisane su ključne implementirane funkcionalnosti, kao i način na koji su korisničke akcije podržane unutar aplikacije.

3.3.1. Funkcionalnosti za profesore

Profesori imaju pristup sljedećim funkcionalnostima unutar aplikacije:

Stvaranje i upravljanje sobama

Profesori mogu:

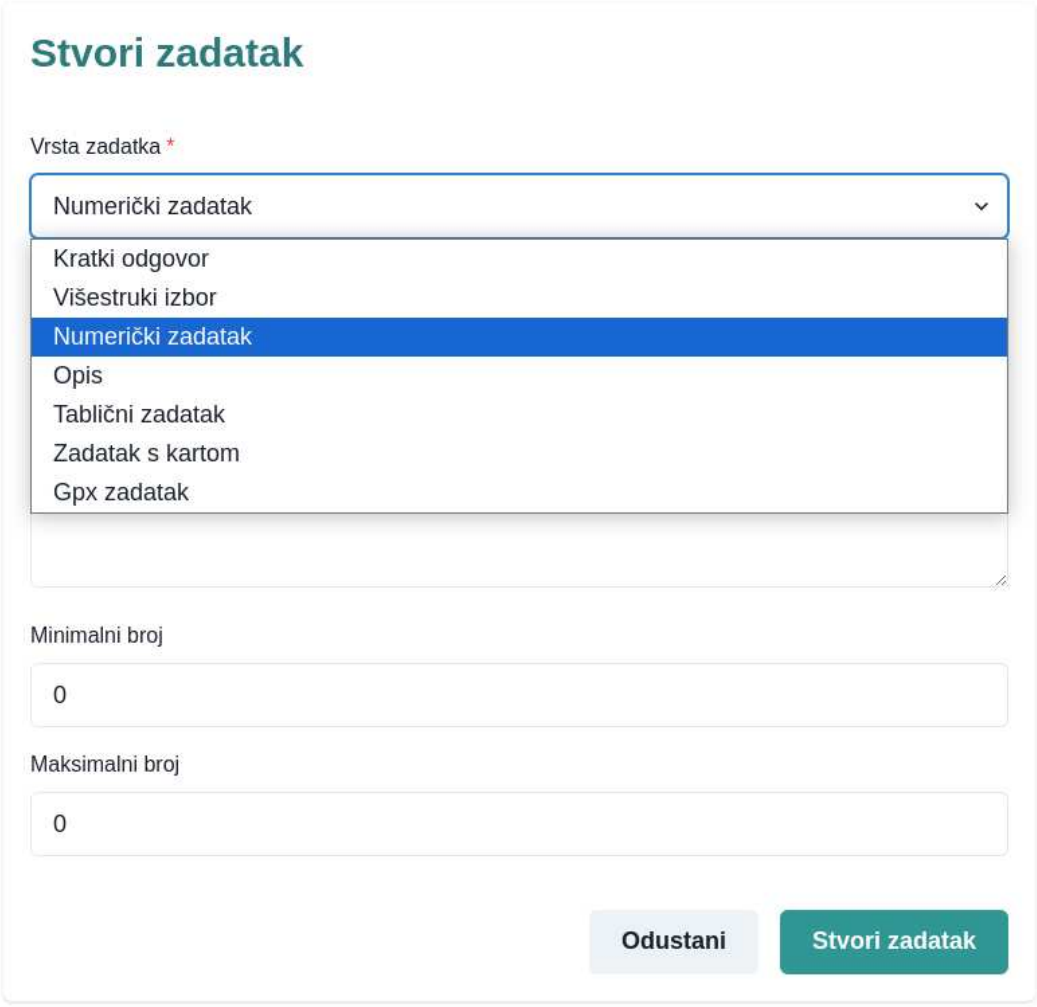
- Stvoriti novu sobu s jedinstvenim nazivom i lozinkom.

- Pristupiti postojećoj sobi pomoću unosa odgovarajuće lozinke.

Dodavanje zadataka

Unutar svake sobe, profesor može dodavati zadatke za učenike. Zadaci mogu biti različitih tipova:

- **Kratki tekst:** Učenici unose kratki odgovor u tekstualnom obliku.
- **Numerički zadatak:** Učenici unose brojčanu vrijednost.
- **Višestruki odabir:** Profesor definira opcije, a učenici odabiru jednu ili više njih.
- **Tablični zadatak:** Učenici unose odgovore u strukturi tablice.
- **Zadatak s kartom:** Učenici označavaju lokacije na interaktivnoj karti.
- **Zadatak s *GPX*-trakom:** Učenici učitavaju *GPX*-datoteku s informacijama o rutama ili stazama.
- **Opis:** Profesor dodaje upute za učenike.

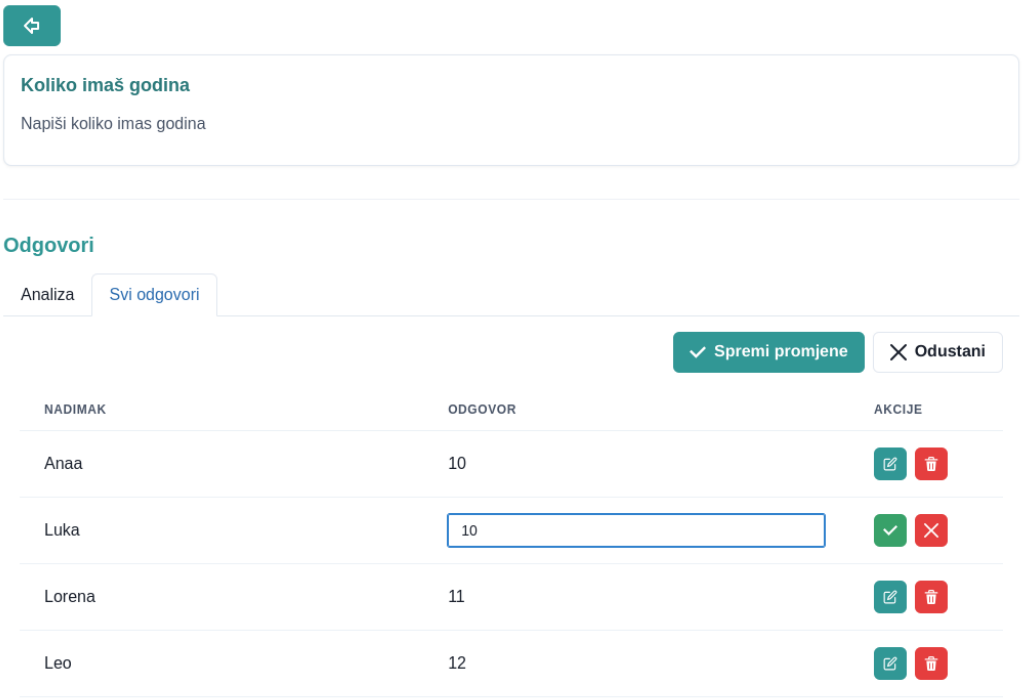


Slika 3.6. Primjer odabira tipa zadatka

Upravljanje odgovorima učenika

Profesori imaju mogućnost:

- Pregleda i uređivanja odgovora učenika.
- Brisanja odgovora koji nisu prikladni ili su uneseni pogreškom.

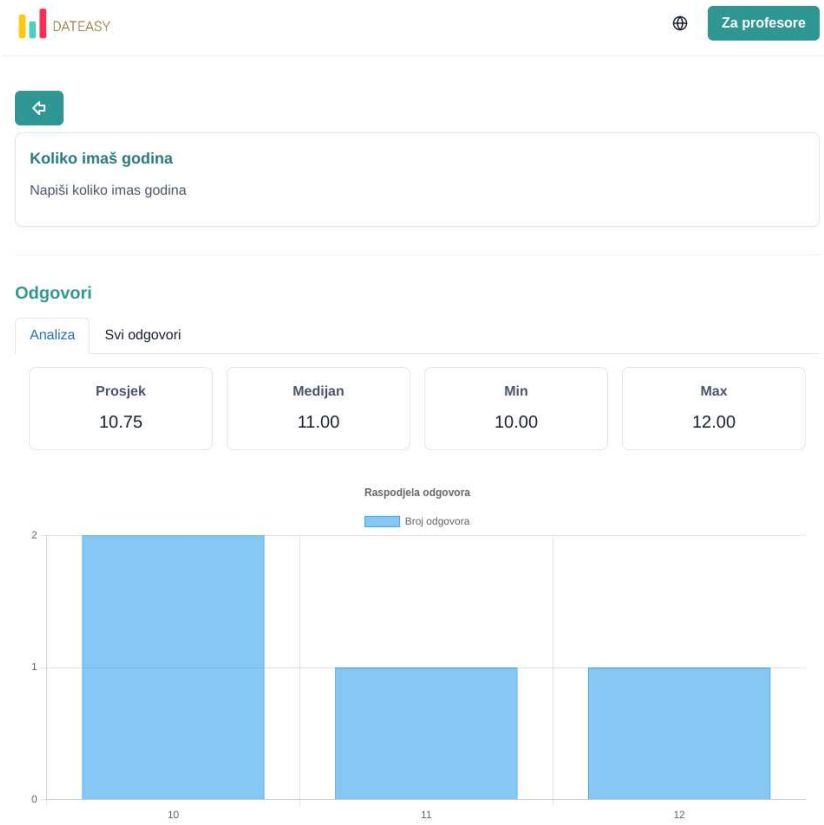


Slika 3.7. Primjer uređivanja svih odgovora

Prezentacijski prikaz rješenja

Profesori mogu koristiti funkciju prezentacijskog prikaza kako bi prikazali zadatke učenicima u stvarnom vremenu. Ključne značajke ove funkcionalnosti uključuju:

- Prikaz svakog zadatka pojedinačno, uz opciju kretanja kroz zadatke.
- Uključivanje ili isključivanje anonimiziranih odgovora učenika radi zaštite privatnosti.



Slika 3.8. Primjer prezentacijskog prikaza svih odgovora

3.3.2. Funkcionalnosti za učenike

Učenici koriste aplikaciju na sljedeći način:

Prijava u sobu

Učenici ulaze u sobu unosom svog nadimka i naziva sobe. Ovaj proces omogućuje jednostavnu prijavu i zaštitu privatnosti.

Rješavanje zadataka

Nakon prijave, učenici ispunjavaju zadatke koje je profesor definirao:

- Za svaki zadatak pruža se odgovarajuće sučelje prilagođeno tipu zadatka.
- Učenik unosi svoje odgovore i klikom na gumb *Predaj* šalje odgovore.

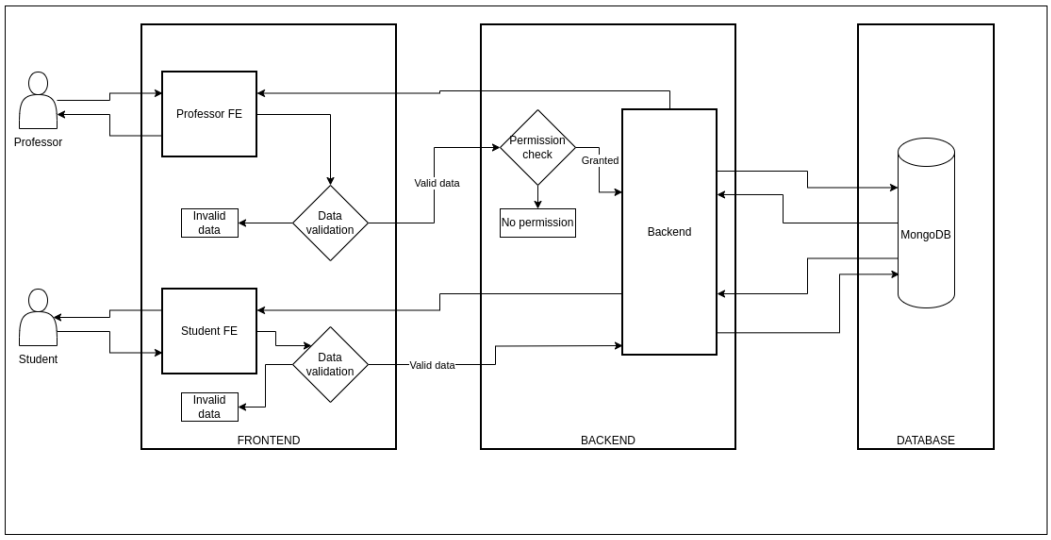
Sudjelovanje u dijeljenom skupu podataka

Svi uneseni odgovori učenika spremaju se u zajednički skup podataka unutar sobe. Ovi podaci mogu se kasnije koristiti za analizu ili prezentaciju tijekom rada grupe.

4. Implementacija i tehničke značajke

4.1. Arhitektura aplikacije

Arhitektura aplikacije podijeljena je na tri glavna sloja: klijentski sloj, poslužiteljski sloj i baza podataka. Svaki sloj ima jasno definirane funkcionalnosti i uloge.



Slika 4.1. Arhitektura aplikacije

4.1.1. Klijentski sloj

Klijentski sloj sastoji se od dva korisnička sučelja:

- **Sučelje za profesore (Professor FE)** – namijenjeno profesorima za upravljanje podacima.
- **Sučelje za učenike (Student FE)** – prilagođeno učenicima mlađe školske dobi za unos i pregled podataka.

Korisnici unose podatke u aplikaciju, koji prolaze kroz proces validacije podataka. Ako

uneseni podaci nisu valjani, aplikacija vraća povratnu informaciju o grešci.

4.1.2. Poslužiteljski sloj

Nakon uspješne validacije na klijentskom sloju, podaci se šalju u sloj poslužitelja. Poslužitelj obavlja sljedeće funkcije:

- Provjerava dozvole za pristup korisnika na temelju njihove uloge (profesor ili učenik).
- Odbija pristup ako korisnik nema potrebne dozvole.
- Komunicira s bazom podataka kako bi pohranjivao ili dohvaćao podatke.

4.1.3. Baza podataka

Poslužitelj koristi *MongoDB* kao bazu podataka za pohranu svih korisničkih unosa i zajedničkih skupova podataka. Ova baza služi kao središnje spremište koje omogućuje siguran i učinkovit rad s podacima.

4.1.4. Tijek podataka

Cjelokupni tijek podataka prikazan je sljedećim koracima:

1. Korisnik unosi podatke putem korisničkog sučelja (*Frontend*).
2. Podaci se validiraju i prosljeđuju poslužiteljskom sloju (*Backend*).
3. Poslužitelj provjerava dozvole korisnika i, ako su ispunjeni uvjeti, komunicira s bazom podataka (*MongoDB*) za pohranu ili dohvat podataka.

Ova arhitektura omogućuje siguran, strukturiran i skalabilan rad aplikacije s naglaskom na validaciju i kontrolu pristupa.

4.2. Tehnologije za razvoj aplikacije

Za razvoj aplikacije korištene su moderne tehnologije koje omogućuju brzu, skalabilnu i intuitivnu implementaciju funkcionalnosti, prilagođenu specifičnim zahtjevima pro-

jekta. Sama aplikacija razvijana je na više razina – poslužitelj, klijent i baza podataka – uz upotrebu alata i knjižnica koji osiguravaju rješenje otpornije na pogreške.

4.2.1. Programski jezik *Go*

Poslužitelj aplikacije razvijen je u programskom jeziku *Go* (*Golang*), poznatom po svojoj visokoj izvedbi, jednostavnoj sintaksi i snažnoj podršci za konkurentnost. *Go* je idealan za aplikacije koje zahtijevaju obradu velikih količina podataka ili rad s više korisnika istovremeno. Njegove prednosti koje su korištene u ovom projektu uključuju:

1. Brzo izvršavanje zahvaljujući prevođenju u nativni kôd.
2. Jednostavan razvoj skalabilnih i pouzdanih programskih sučelja (*API*).
3. Široku knjižnicu standardnih paketa koja pokriva raznovrsne potrebe, uključujući *HTTP*-poslužitelje, koji omogućuju prijenos sadržaja putem *HTTP*-protokola, i rad s datotekama.

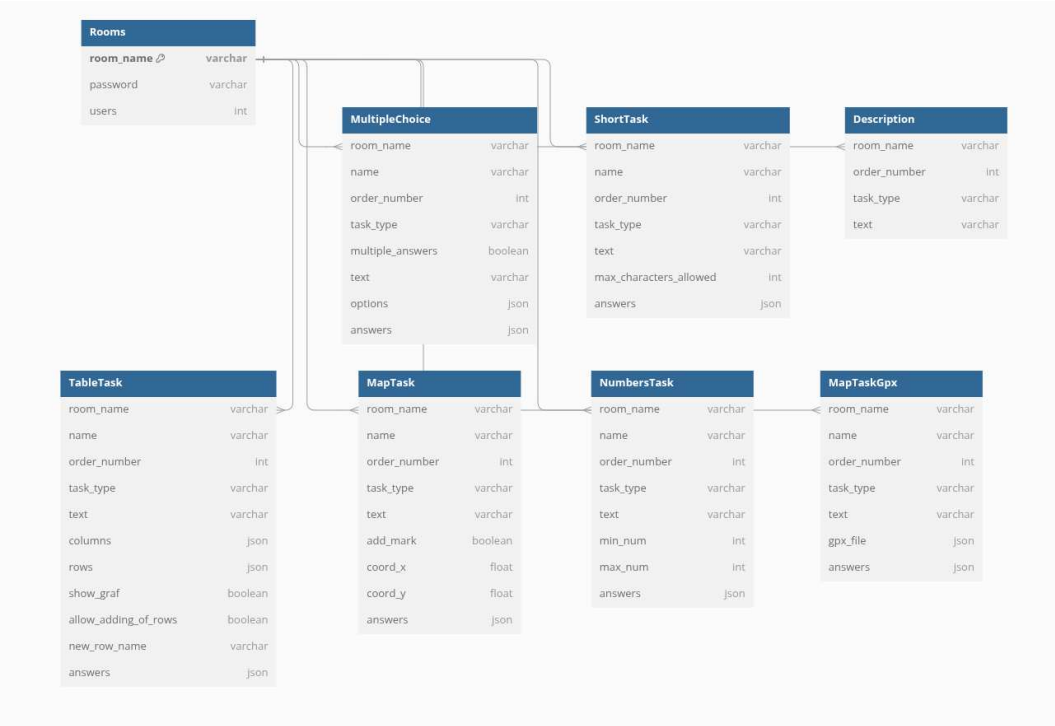
Poslužitelj osigurava komunikaciju s bazom podataka, obradu podataka i podršku za REST API, omogućujući nesmetanu integraciju s klijentom.

4.2.2. Baza podataka: *MongoDB*

Za pohranu podataka korištena je *NoSQL* baza podataka *MongoDB*, poznata po svojoj fleksibilnosti i sposobnosti da rukuje nestrukturiranim ili polustrukturiranim podacima. Razlozi za odabir baze *MongoDB* uključuju:

1. Skladištenje podataka u formatu kao što je format JSON, što olakšava rad s podacima i njihovu manipulaciju.
2. Fleksibilnost u dizajnu modela podataka, što je korisno za rad s podacima različitih tipova, poput tekstualnih vrijednosti, brojeva, geografskih lokacija i *GPX*-zapisa.
3. Mogućnost rada s velikim količinama podataka i podrška za horizontalnu skalabilnost.

MongoDB omogućuje brzu i učinkovitu pohranu podataka unesenih od strane korisnika te njihovo dohvaćanje za vizualizaciju i analizu.



Slika 4.2. Dijagram baze podataka

Kao što je vidljivo i u dijagramu, za spremanje odgovora koristi se format *JSON*. Svaki odgovor sprema se u listu odgovora *answers* u obliku *nadimak: odgovor*.

```
{
    "id": "67865f426043b72cef0718c8",
    "room_name": "soba1",
    "name": "Koliko imas godina",
    "task_type": "numbers_task",
    "order_number": 2,
    "text": "Napisi koliko imas godina",
    "min_num": 10,
    "max_num": 12,
    "answers": [
        {
            "username": "Anaa",
            "answer": "10"
        },
        {
            "username": "Luka",
```

```
            "answer": "10"
        },
        {
            "username": "Lorena",
            "answer": "11"
        },
        {
            "username": "Leo",
            "answer": "12"
        }
    ]
}
```

Kôd 4.1. Primjer objekta u bazi

4.2.3. Korisničko sučelje: *React* + *TypeScript* + *Chakra UI*

Korisničko sučelje aplikacije razvijeno je koristeći knjižnicu *React* u kombinaciji s jezikom *TypeScript* i knjižnicom komponenti *Chakra UI*. Ovaj skup alata omogućuje razvoj responzivnog, prilagodljivog i vizualno privlačnog korisničkog sučelja.

Knjižnica *React*

React je korišten za izradu dinamičkog korisničkog sučelja koje se brzo ažurira prema korisničkim unosima. Njegova komponentno orijentirana arhitektura omogućila je modularnost i ponovnu upotrebu kôda.

Programski jezik *TypeScript*

TypeScript donosi statičko tipiziranje i dodatne značajke iznad standardnog jezika JavaScript, čime je povećana sigurnost kôda i smanjen broj potencijalnih pogrešaka.

Knjižnica komponenti *Chakra UI*

Chakra UI pruža unaprijed definirane i prilagodljive komponente za izradu modernog i dosljednog dizajna. Omogućuje brzo implementiranje sučelja i visoku fleksibilnost u prilagodbi dizajna koji je primamljiv korisnicima mlađe školske dobi.

4.2.4. Knjižnice korištene u aplikaciji

Za razvoj i proširenje funkcionalnosti korištene su različite knjižnice. Najbitnije su:

Za vizualizaciju podataka

- ***D3.js*** i ***d3-cloud*** [1]: Za napredne vizualizacije podataka poput grafikona i oblaka riječi.
- ***Chart.js***[2]: Za jednostavnu integraciju prilagođenih grafova u aplikaciju.
- ***Leaflet***[3]: Za vizualizaciju podataka na interaktivnim kartama.

Za internacionalizaciju i lokalizaciju

- ***i18next***[4]: Omogućuje podršku za više jezika u aplikaciji, što je važno za prilagodbu korisnicima različitih jezičnih skupina.

Za poboljšanje korisničkog iskustva

- ***Framer Motion***[5]: Za animacije i prijelaze u sučelju koje povećavaju interaktivnost i privlačnost aplikacije.
- ***Lucide-react***[6]: Za prilagođene i vizualno privlačne ikone.

4.2.5. Razvojni alati i infrastruktura

Osim knjižnica, korišteni su i alati za razvoj i održavanje aplikacije:

- ***Vite***[7]: Alat za brzi razvoj i izgradnju *React* aplikacija, koji omogućuje brzo pokretanje i pregled promjena tijekom razvoja.
- ***ESLint***[8]: Alat za analizu i osiguravanje kvalitete kôda, s posebnim dodatcima za *React* i *TypeScript*.

[1] D3 by Observable, https://d3js.org
[2] *Chart.js*, https://chartjs.org/
[3] Leaflet, https://leafletjs.com
[4] i18next, https://www.i18next.com/
[5] Framer Motion, https://github.com/motiondivision/motion
[6] Lucide React, https://lucide.dev/guide/packages/lucide-react
[7] Vite, https://vite.dev/
[8] ESLint, https://eslint.org/

4.3. Detalji implementacije

U ovom poglavlju prikazani su odabrani isječci kôda i datoteka aplikacije koji implementiraju ključne funkcionalnosti. Isječci su organizirani prema kategorijama funkcionalnosti.

4.3.1. Rute na sučelju(*API*)

Sve pristupne točke (*endpoints*) na sučelju su organizirane po kategorijama ovisno o akcijama za koje služe. Neke od najbitnijih su:

- `/createRoom` - za kreiranje sobe
- `/tasks` - dohvaćanje svih zadataka za sobu
- `/addTask/{taskType}` - dodavanje zadatka ovisno o tipu
- `/editTask/{taskType}` - uređivanje zadatka ovisno o tipu
- `/giveAnswers` - pošalji odgovore učenika u bazu i pospremi ih po zadacima

4.3.2. Poslužitelj - kreiranje sobe

Funkcija *HandleCreateRoom* služi za kreiranje nove sobe u aplikaciji, gdje prvo provjerava ispravnost unesenih podataka (ime sobe i lozinka moraju biti duži od 4 znaka) i provjerava postoji li već soba s tim imenom. Nakon provjere, stvara novu sobu s kriptiranom lozinkom, inicijalizira popis korisnika, sprema sobu u bazu, te na kraju generira i vraća *JWT* token koji identificira administratora nove sobe. Posljednji korak je da vraća kreirani token klijentu kao *JSON* odgovor s *HTTP* statusom 200 (*OK*).

```
1 func (s *APIServer) HandleCreateRoom(w http.ResponseWriter, r
     *http.Request) error {
2   c_r_req := new(types.JoinAsAdminRequest)
3   if err := json.NewDecoder(r.Body).Decode(c_r_req); err !=
       nil {
4     return err
5   }
6
```

```
7   // validacija imena sobe i lozinke
8   if len(c_r_req.RoomName) < 4 {
9     return WriteJSON(w, http.StatusBadRequest, ApiError{Error:
          "Room name must be longer than 4 characters"})
10  }
11  if len(c_r_req.Password) < 4 {
12    return WriteJSON(w, http.StatusBadRequest, ApiError{Error:
          "Password must be longer than 4 characters"})
13  }
14
15  // provjeri postoji li soba
16  if _, err := s.store.GetRoomByName(c_r_req.RoomName); err ==
        nil {
17    return WriteJSON(w, http.StatusBadRequest, ApiError{Error:
          "Room name already taken"})
18  }
19
20  // stvori novu sobu s lozinkom
21  room := types.NewRoom(c_r_req.RoomName)
22  room.Password = hashPassword(c_r_req.Password)
23  room.Users = make(map[string]bool) // inicijaliziraj
      korisnicku kartu
24
25  // spremi sobu
26  if _, err := s.store.CreateRoom(room); err != nil {
27    return err
28  }
29
30  // stvori token za sobu
31  tokenString, err := createJWT(map[string]interface{}{
32    "username": "Admin",
33    "room_name": c_r_req.RoomName,
34  })
35  if err != nil {
```

```
36      return err
37    }
38
39    // vrati token
40    return WriteJSON(w, http.StatusOK, map[string]interface{}{
41      "token": tokenString,
42    })
43  }
```

Kôd 4.2. Primjer kreiranja sobe

4.3.3. Poslužitelj - dohvaćanje zadatka

Primjer dohvaćanja zadatka na tipu zadatka "Kratki tekst". U funkciji *GetShortTasks* dohvaća se kolekcija *Short Task* iz baze te se zadaci za određenu sobu filtriraju po nazivu sobe. Podaci se nakon dohvaćanja dekodiraju iz strukture *MongoDB* u strukturu *Go* i dodaju u listu.

```
1  func (s *MongoStore) GetShortTasks(roomName string) ([]*types.
       ShortTask, error) {
2    collection := s.client.Database("taskdb").Collection("
       ShortTask")
3
4    var tasks []*types.ShortTask
5    filter := bson.M{"room_name": roomName}
6
7    cursor, err := collection.Find(context.Background(), filter)
8    if err != nil {
9      return nil, fmt.Errorf("error trying to get multiple
         choice tasks: %v", err)
10   }
11   defer cursor.Close(context.Background())
12
13   for cursor.Next(context.Background()) {
14     var task types.ShortTask
15     if err := cursor.Decode(&task); err != nil {
```

```
16          return nil, fmt.Errorf("error decoding task: %v", err)
17        }
18        tasks = append(tasks, &task)
19      }
20      if err := cursor.Err(); err != nil {
21        return nil, err
22      }
23
24      return tasks, nil
25  }
```

Kôd 4.3. Poslužitelj - dohvaćanje zadatka

4.3.4. Zaštićene rute

Zaštićene rute služe za autorizaciju i rutiranje u aplikaciji, gdje prvo provjerava korisničku ulogu (profesor ili učenik) i ima li korisnik potrebna prava pristupa. Ako korisnik nema odgovarajuća prava, prikazuje se poruka o pogrešci (*toast*, skočna obavijesna poruka) i korisnik se preusmjerava - učenici se šalju na pregled zadataka (*/room/:roomName/view*), a ostali na početnu stranicu. U dijelu za rute vidimo definicije javnih ruta (stranice za prijavu) i zaštićenih ruta (npr. upravljanje zadacima) koje su dostupne samo profesorima kroz komponentu *ProtectedRoute*.

```
1   // dohvati korisnicku ulogu
2   const userRole = roomHandler.getUserRole();
3
4   // provjeri je li uloga valjana
5   if (!userRole || !allowedRoles.includes(userRole)) {
6     toast({
7       title: t("error.accessDenied"),
8       description: t("error.insufficientPermissions"),
9       status: "error",
10      duration: 3000,
11      isClosable: true,
12    });
13
```

```
14      // ako ucenik pokusa pristupiti ekranu namijenjenom za
            profesore, preusmjeri ga
15      if (userRole === "student") {
16        const roomName = location.pathname.split("/")[2];
17        return <Navigate to={`/room/${roomName}/view`} replace
              />;
18      }
19
20      // za sve ostale slucajeve, vrati na pocetnu
21      return <Navigate to="/" replace />;
22    }
23
24    ...
25
26    <Routes>
27      {/* Javne rute */}
28      <Route path="/" element={<StudentLoginPage />} />
29      <Route path="/prof" element={<ProfessorLoginPage />} />
30
31      {/* Profesorske rute */}
32      <Route
33        path="/room/:roomName/edit"
34        element={
35          <ProtectedRoute allowedRoles={["professor"]}>
36            <ProfessorTaskManagementPage />
37          </ProtectedRoute>
38        }
39      />
40      ...
```

Kôd 4.4. Zaštićene rute

4.3.5. Validacija unosa

Validacija unosa provjerava svaku unesenu vrijednost od strane korisnika. Ova funkcija *validateNumberTask* služi za validaciju zadatka s brojevima, gdje provjerava jesu li minimalni i maksimalni broj ispravno definirani kao brojevi te je li maksimalni broj stvarno veći od minimalnog. Ako bilo koji od ovih uvjeta nije ispunjen, u objekt *errors* se dodaje odgovarajuća poruka o grešci koja je lokalizirana pomoću prosljeđene funkcije za prijevod (‘t‘).

```
const validateNumberTask = (
  data: ExtendedTask,
  errors: FormErrors,
  t: TFunction
): void => {
  if (typeof data.min_num !== "number") {
    errors.min_num = t("error.invalidMinNumber");
    return;
  }

  if (typeof data.max_num !== "number") {
    errors.max_num = t("error.invalidMaxNumber");
    return;
  }

  if (data.max_num <= data.min_num) {
    errors.max_num = t("error.maxGreaterThanMin");
  }
};
```

Kôd 4.5. Validacija unosa

4.3.6. Dohvaćanje zadataka

Ovaj dio kôda služi za dohvaćanje zadataka s *API*-ja, koji prvo provjerava je li korisnik autentificiran i ima li spremljen token u lokalnoj pohrani preglednika. Ako su autentifikacijski uvjeti zadovoljeni, kôd šalje *HTTP*-zahtjev na *endpoint* /tasks s *JWT*-tokenom

u zaglavlju, a ako dođe do pogreške pri dohvaćanju (bilo da je odgovor neispravan ili se ne može parsirati), baca se odgovarajuća pogreška. Nakon uspješnog dohvata, podaci o zadacima se parsiraju iz formata *JSON* u niz objekata tipa *ExtendedTask*.

```
    try {
      if (!roomHandler.isAuthenticated()) {
        navigate("/");
        throw new Error("Not authenticated");
      }

      const authToken = localStorage.getItem("token");
      if (!authToken) {
        navigate("/");
        throw new Error("Not authenticated");
      }

      const response = await fetch("http://localhost:3000/
          tasks", {
        headers: {
          "Content-Type": "application/json",
          "x-jwt-token": authToken,
        },
      });

      if (!response.ok) {
        const error = await response.text();
        try {
          const parsedError = JSON.parse(error);
          throw new Error(parsedError.error || "Failed to
              fetch tasks");
        } catch {
          throw new Error("Failed to fetch tasks");
        }
      }

```

```
30        const data: ExtendedTask[] = await response.json();
```

Kôd 4.6. Dohvaćanje zadataka

4.3.7. Slanje odgovora

Funkcija *allAnswers* služi za slanje odgovora na zadatke prema serveru, koji prvo sortira zadatke po rednom broju, filtrira van zadatke tipa "description", te mapira preostale zadatke u format koji sadrži ID zadatka i odgovor iz reference. Nakon pripreme podataka, kôd šalje *POST* zahtjev na *endpoint* /*giveAnswer* s pripremljenim odgovorima u *JSON* formatu, gdje se također uključuje *JWT* token iz lokalne pohrane za autentifikaciju.

```
1  const allAnswers = tasks
2        .sort((a, b) => a.order_number - b.order_number)
3        .filter((task) => task.task_type !== "description")
4        .map((task) => ({
5          id: task.id,
6          answer: taskRefs.current[task.order_number] || "",
7        }));
8
9      try {
10       const authToken = localStorage.getItem("token");
11       if (!authToken) {
12         throw new Error("Not authenticated");
13       }
14
15       const response = await fetch("http://localhost:3000/
             giveAnswer", {
16         method: "POST",
17         headers: {
18           "Content-Type": "application/json",
19           "x-jwt-token": authToken,
20         },
21         body: JSON.stringify(allAnswers),
22       });
```

Kôd 4.7. Slanje odgovora

4.3.8. Prikaz odgovora na karti

Ova komponenta služi za prikaz karte (mape) korištenjem knjižnice *React-Leaflet*, gdje se karta centrira na zadane koordinate i postavlja na određenu razinu zumiranja. Unutar karte se prikazuje *OpenStreetMap* sloj s mapom, a ako postoje odgovori (*hasAnswers*), prikazuje se i sloj s markerima (*MarkerLayer*) koji može prikazivati anonimne ili imenovane markere ovisno o postavci *isAnonymous*.

```
<Box h="500px" position="relative" borderRadius="lg" overflow=
   "hidden">
        <MapContainer
          center={[center.lat, center.lng]}
          zoom={13}
          style={{ height: "100%", width: "100%" }}
        >
          <TileLayer
            attribution='&copy; <a href="https://www.
               openstreetmap.org/copyright">OpenStreetMap</a>
               contributors'
            url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y
               }.png"
          />
          {hasAnswers && (
            <MarkerLayer answers={answers} isAnonymous={
               isAnonymous} />
          )}
        </MapContainer>
      </Box>
```

Kôd 4.8. Prikaz odgovora na karti

4.3.9. Prikaz grafa

U odgovorima podaci se prikazuju kroz dva različita grafikona - kružni (*Pie*) i stupčasti (*Bar*) graf, gdje se definiraju opcije za stupčasti graf poput responzivnosti i veličine koraka (*stepSize*).

```
const barOptions = {
    responsive: true,
    maintainAspectRatio: false,
    scales: {
      y: {
        beginAtZero: true,
        ticks: {
          stepSize: 1,
        },
      },
    },
    plugins: {
      legend: {
        display: false,
      },
    },
  };
  ...
  <TabPanel>
    <HStack spacing={8} justify="center" align="start">
      <Box w="300px" h="300px">
        <Heading size="sm" textAlign="center" mb={4}>
          {t("pieDistribution")}
        </Heading>
        <Pie data={chartData} />
      </Box>
      <Box w="400px" h="300px">
        <Heading size="sm" textAlign="center" mb={4}>
          {t("barDistribution")}
        </Heading>
```

```
        <Bar data={chartData} options={barOptions} />
      </Box>
    </HStack>
  </TabPanel>
```

Kôd 4.9. Prikaz grafa

4.3.10. Prikaz oblaka riječi

Prikaz oblaka riječi omogućuje analizu učestalosti pojmova. Ovaj kôd služi za kreiranje oblaka riječi (*word cloud*) koristeći knjižnicu *D3.js*, gdje se riječi raspoređuju s različitim rotacijama i veličinama fonta. Svaka riječ je obojana jednom od definiranih boja iz palete, a cijeli oblak je smješten unutar *SVG*-elementa koji je responzivan.

```
...
    cloud<CloudWord>()
      .size([600, 300])
      .words(words)
      .padding(5)
      .rotate(() => (Math.random() - 0.5) * 30)
      .fontSize((d) => d.size)
      .on("end", (words) => {
        d3.select(svgRef.current)
          .attr("viewBox", "0 0 600 300")
          .append("g")
          .attr("transform", "translate(300,150)")
          .selectAll("text")
          .data(words)
          .join("text")
          .style("font-size", (d) => `${d.size}px`)
          .style("font-family", "Impact")
          .style("fill", (d) => {
            const colors = [
              "#2D3282", // blue
              "#E6447D", // pink
              "#28A745", // green
```

```
                "#9333EA", // purple
                "#F59E0B", // amber
                "#10B981", // emerald
                "#3B82F6", // bright blue
                "#EC4899", // hot pink
              ];
              return colors[d.value % colors.length];
            })
            .style("cursor", "pointer")
            .attr("text-anchor", "middle")
            .attr(
              "transform",
              (d) => `translate(${d.x ?? 0},${d.y ?? 0})rotate(${d.rotate ?? 0})`
            )
            .text((d) => d.text as string);
        })
        .start();
    }, [answers, t]);

    return (
      <Box w="100%" px={4} bg={containerColor} borderRadius="md" py={4}>
        <svg
          ref={svgRef}
          style={{
            width: "100%",
            height: "auto",
            maxHeight: "70vh",
          }}
        />
      </Box>
    );
};
```

Kôd 4.10. Prikaz oblaka riječi

4.3.11. Obrazac za učenika

Kada se dohvate svi zadaci, učeniku se generira formular za ispunjavanje. Slijedi primjer generiranja prozora za prijenos *GPX*-datoteke, gdje je prikazan okvir s isprekidanom linijom koji služi kao područje za *drag-and-drop* ili klik za odabir datoteke. Komponenta vizualno pokazuje je li datoteka odabrana promjenom ikone (iz *FileUp* u *FileCheck*) i boje teksta, a cijeli element je omotan u *TaskWrapper* komponentu koja dodaje naslov i opis zadatka.

```
<TaskWrapper title={title} description={description}>
  <FormControl>
    <Box
      border="2px dashed"
      borderColor="teal.200"
      borderRadius="md"
      p={6}
      _hover={{ borderColor: "teal.400" }}
      transition="all 0.2s"
    >
      <Input
        type="file"
        accept=".gpx"
        onChange={handleFileChange}
        hidden
        id="gpx-file-input"
      />
      <label htmlFor="gpx-file-input">
        <VStack spacing={3} cursor="pointer">
          {selectedFile ? (
            <FileCheck color="#319795" size={24} /> // teal
                .500 color
          ) : (
```

```
            <FileUp color="#4FD1C5" size={24} /> // teal.300 color
          )}
          <Text color={selectedFile ? "teal.500" : "gray.500"}>
            {selectedFile ? selectedFile.name : t("uploadGpxFile")}
          </Text>
        </VStack>
      </label>
    </Box>
  </FormControl>
</TaskWrapper>
```

Kôd 4.11. Obrazac za učenike

4.3.12. Internacionalizacija

Ovaj kod služi za postavljanje internacionalizacije (*i18n*) u aplikaciji, gdje se definiraju prijevodi za engleski i hrvatski jezik sa svim potrebnim tekstovima. Konfiguracija postavlja hrvatski kao glavni jezik i engleski kao rezervni (*fallback*), a koristi *react-i18next* knjižnicu za integraciju s komponentama.

```
import i18n from "i18next";
import { initReactI18next } from "react-i18next";

const resources = {
  en: {
    translation: {
      respondent: "Response",
      forProfessors: "For professors",
      forStudents: "For students",
      createRoom: "Create Room",
      joinRoom: "Join Room",
      ...
  hr: {
```

```
    translation: {
      respondent: "Odgovor",
      forProfessors: "Za profesore",
      forStudents: "Za ucenike",
      createRoom: "Stvori sobu",
      joinRoom: "Udi u sobu",
      ...
i18n.use(initReactI18next).init({
  resources,
  lng: "hr",
  fallbackLng: "en",
  interpolation: {
    escapeValue: false,
  },
});
```

Kôd 4.12. Internacionalizacija

4.3.13. Anonimizacija

Ovaj kod implementira sustav za upravljanje anonimnošću u aplikaciji kroz *React Context*. *AnonymityProvider* komponenta upravlja *boolean* stanjem *isAnonymous* i pruža funkciju za njegovo mijenjanje, dok *AnonymityToggle* komponenta kreira kontrolu za mijenjanje anonimnosti. Anonimnost se koristi u zadacima za skrivanje korisničkih imena kada je aktivirana.

```
interface AnonymityContextType {
  isAnonymous: boolean;
  toggleAnonymity: () => void;
}

const AnonymityContext = createContext<AnonymityContextType |
   undefined>(
  undefined
);

```

```
export const AnonymityProvider: React.FC<{ children: React.ReactNode }> = ({
  children,
}) => {
  const [isAnonymous, setIsAnonymous] = useState(false);

  const toggleAnonymity = () => {
    setIsAnonymous((prev) => !prev);
  };

  return (
    <AnonymityContext.Provider value={{ isAnonymous, toggleAnonymity }}>
      {children}
    </AnonymityContext.Provider>
  );
};

export const useAnonymity = () => {
  const context = useContext(AnonymityContext);
  if (context === undefined) {
    throw new Error("useAnonymity must be used within an AnonymityProvider");
  }
  return context;
};
```

Kôd 4.13. Kontekst za anonimizaciju

```
const AnonymityToggle = () => {
  const { isAnonymous, toggleAnonymity } = useAnonymity();
  const { t } = useTranslation();

  return (
    <HStack spacing={2}>
```

```
      <Switch
        colorScheme="teal"
        isChecked={isAnonymous}
        onChange={toggleAnonymity}
        id="anonymous-mode"
      />
      <Text fontSize="sm" color="gray.600">
        {isAnonymous ? t("anonymous") : t("showUsernames")}
      </Text>
    </HStack>
  );
};

// ako je zastavica isAnonymus true, nemoj pokazati nadimke
<Thead>
    <Tr>
      <Th>{isAnonymous ? t("respondent") : t("username")}</Th>
      <Th>{t("answer")}</Th>
    </Tr>
</Thead>
```

Kôd 4.14. Primjer korištenja anonimnosti

5. Zaključak

Razvoj web-aplikacije za unapređenje podatkovne pismenosti djece mlađe školske dobi predstavlja inovativan i značajan korak u modernizaciji obrazovanja prilagođenog digitalnom dobu. U sklopu ovog rada istraženi su ključni principi dizajna korisničkog sučelja i metode poučavanja podataka, fokusirajući se na specifične kognitivne i vizualne potrebe djece ovog uzrasta. Kroz detaljnu analizu postojećih edukacijskih alata i primjenu najboljih praksi u području obrazovne tehnologije, razvijeno je sveobuhvatno rješenje koje omogućuje prikupljanje, dijeljenje i analizu podataka u sigurnom i poticajnom digitalnom okruženju.

Implementirana web-aplikacija obuhvaća širok spektar funkcionalnosti prilagođenih djeci osnovnoškolskog uzrasta. Ključne značajke uključuju intuitivan unos brojčanih, tekstualnih i geografskih podataka, dinamičku vizualizaciju kroz interaktivne grafove, kreativne oblake riječi i geografske karte, te jednostavne ali učinkovite alate za analizu podataka. Inovativni koncept "virtualnih soba" potiče suradničko učenje i timski rad među korisnicima, omogućujući djeci da zajedno istražuju i analiziraju podatke. Posebna pažnja posvećena je dizajnu korisničkog sučelja koje je izrađeno prema principima pristupačnosti i prilagođenosti djeci, osiguravajući optimalnu intuitivnost, responzivnost na različitim uređajima i vizualnu privlačnost koja održava motivaciju i angažman korisnika.

Ovaj projekt potvrđuje da se moderna tehnologija može koristiti za stvaranje alata koji spajaju igru, edukaciju i suradnju, pružajući djeci priliku da usvoje ključne vještine rada s podacima već u ranoj dobi. U budućnosti, razvoj ovakvih alata može značajno doprinijeti razvoju podatkovno pismenih generacija spremnih za izazove digitalnog društva.

Literatura

[1] E. Carmi i S. J. Yates, “Digital inclusion and data literacy”, *Internet Policy Review*, 5 2020. https://doi.org/10.14763/2020.2.1474

[2] L. Bowler, A. Acker, W. Jeng, i Y. Chi, “"it lives all around us": Aspects of data literacy in teen’s lives”, *Proceedings of the Association for Information Science and Technology*, sv. 54, br. 1, str. 27–35, 2017. https://doi.org/10.1002/pra2.2017.14505401004

[3] Hrčak, https://hrcak.srce.hr/263400/, [mrežno; stranica posjećena: siječanj 2025.].

[4] J. Robertson i E. K. Tisdall, “The importance of consulting children and young people about data literacy”, *Journal of Media Literacy Education*, sv. 12, str. 58–74, 12 2020. https://doi.org/10.23860/JMLE-2020-12-3-6

[5] D. Kuhn, “Defining and developing data literacy”, *Routledge Open Research*, sv. 2, str. 44, 10 2023. https://doi.org/10.12688/routledgeopenres.18015.1

[6] E. Deahl, “Better the data you know: Developing youth data literacy in schools and informal learning environments”, diplomski ili magistarski rad, Massachusetts Institute of Technology, Department of Comparative Media Studies, Cambridge, MA, USA, 2014., [CrossRef].

[7] A. Dangol i S. Dasgupta, “Constructionist approaches to critical data literacy: A review”, u *Proceedings of the Interaction Design and Children (IDC ’23)*. Chicago, IL, USA: ACM, 2023., str. 12. https://doi.org/10.1145/3585088.3589367

[8] T. S. M. T. Wook i S. S. Salim, “Guideline for the graphic design of web application for children’s interface”, *TELKOMNIKA*, sv. 11, br. 6, str. 3130–3133, 2013. https://doi.org/10.11591/telkomnika.v11i6.2658

[9] C. Boyatzis i R. Varghese, “Children’s emotional associations with colors”, *The Journal of genetic psychology*, sv. 155, str. 77–85, 04 1994. https://doi.org/10.1080/00221325.1994.9914760

Sažetak

Web-aplikacija za razvoj podatkovne pismenosti djece mlađe školske dobi

Marija Šantek

U digitalnom dobu, podatkovna pismenost postaje ključna vještina koja omogućuje razumijevanje i korištenje podataka u svakodnevnom životu. Ovaj rad istražuje kako se ovakve vještine mogu unaprijediti kod djece mlađe školske dobi kroz razvoj web-aplikacije prilagođene njihovim potrebama. Fokus aplikacije je na omogućavanju jednostavnog unosa, vizualizacije i analize podataka, uz istovremeno poticanje igre i suradnje. Razvijena aplikacija omogućuje unos brojčanih i tekstualnih vrijednosti, geografskih podataka i *GPX*-traka kretanja, dok se podaci vizualiziraju putem interaktivnih grafova, karata i oblaka riječi. Kroz rad u "virtualnim sobama", učenici surađuju na zajedničkim skupovima podataka, dok profesori mogu dodavati zadatke, pregledavati odgovore i analizirati podatke na jednostavan i učinkovit način. Sučelje aplikacije je responzivno, vizualno privlačno i prilagođeno djeci mlađe školske dobi. Ova aplikacija predstavlja korak naprijed u razvoju edukacijskih alata koji spajaju igru i učenje, pružajući djeci temelj za razvoj podatkovne pismenosti.

Ključne riječi: podatkovna pismenost; edukacija; web-aplikacija; vizualizacija podataka; djece mlađe školske dobi

Abstract

Web application for data literacy development of lower school age children

Marija Šantek

In the digital era, data literacy is becoming an essential skill that enables individuals to understand and utilize data in everyday life. This thesis explores how such skills can be developed in young school-age children through the creation of a web application tailored to their needs. The application focuses on providing an intuitive way for children to input, visualize, and analyze data while fostering playfulness and collaboration. The developed application allows children to input numerical and textual values, geographic data, and GPX activity tracks, with data being visualized through interactive graphs, maps, and word clouds. Through "virtual rooms", students collaborate on shared datasets, while teachers can create tasks, review responses, and analyze data efficiently. The application's interface is responsive, visually appealing, and tailored to the cognitive and motor abilities of young children. This application marks a step forward in creating educational tools that merge play and learning, providing children with a foundation in data literacy and preparing them for the challenges of a modern, data-driven society.

Keywords: data literacy; education; web application; data visualization; lower school age children