

**SVEUČILIŠTE U SPLITU**  
**SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE**

Stručni diplomski studij Primijenjeno računarstvo

**ANAMARIJA PAPIĆ**

**D I P L O M S K I   R A D**

**RAZVOJ WEB APLIKACIJE ZA REPETICIJE**

Split, veljača 2026.

**SVEUČILIŠTE U SPLITU**  
**SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE**

Stručni diplomski studij Primijenjeno računarstvo

**Predmet:** Kriptovalute

**D I P L O M S K I R A D**

**Kandidat:** Anamarija Papić

**Naslov rada:** Razvoj web aplikacije za repeticije

**Mentor:** Nikola Grgić, viši predavač

Split, veljača 2026.

# Sadržaj

<b>Sažetak</b>	<b>1</b>
<b>1 Uvod</b>	<b>2</b>
<b>2 Tehnologije i teoretska podloga</b>	<b>4</b>
2.1 Next.js, React, TypeScript, Node.js, npm . . . . .	4
2.2 UI sloj: shadcn/ui, Radix UI, Tailwind CSS . . . . .	6
2.3 Baza podataka: PostgreSQL, Prisma ORM, Supabase . . . . .	7
2.4 WebRTC . . . . .	7
2.5 Verzioniranje kôda, hosting i deployment, email servis . . . . .	7
<b>3 Prikaz praktičnog dijela</b>	<b>8</b>
3.1 Potpoglavlje . . . . .	8
3.1.1 ABC . . . . .	8
3.1.1.1 123 . . . . .	8
3.1.1.2 456 . . . . .	8
3.1.2 DEF . . . . .	8
3.2 Još jedno potpoglavlje . . . . .	8
<b>4 Zaključak</b>	<b>9</b>
<b>Literatura</b>	<b>10</b>
<b>Dodatci</b>	<b>11</b>

## Sažetak

Cilj ovog diplomskog rada je razviti intuitivnu web aplikaciju koja će služiti kao platforma za povezivanje studenata i tutora s ciljem olakšavanja organizacije i održavanja repeticija, pojednostavljajući proces pružanja i pohađanja repeticija uživo ili online. Aplikacija omogućuje studentima jednostavno pronalaženje i rezervaciju repeticija, dok tutori mogu učinkovito upravljati vlastitim rasporedom, ponudama i rezervacijama. Sudjelovanje u online repeticijama podržano je integracijom *peer-to-peer* tehnologije WebRTC za video pozive. *Full-stack* web aplikacija razvijena je u razvojnem okviru za React – Next.js-u, dok se za pohranu podataka koristi relacijska baza podataka PostgreSQL upravljana putem Prisma ORM-a i smještena na platformi Supabase. U ovom pisanom radu predstavljene su korištene tehnologije, arhitektura sustava, poslovna logika te implementacija pojedinih funkcionalnosti aplikacije.

**Ključne riječi:** *sustav za upravljanje rezervacijama, Next.js, WebRTC, audio-video pozivi*

## Summary

### Development of a Web Application for Tutoring

The aim of this thesis is to develop an intuitive web application that serves as a platform for connecting students and tutors, with the goal of facilitating the organization and conduct of tutoring sessions and simplifying the process of providing and attending lessons, both in-person and online. The application enables students to easily find and book tutoring sessions, while tutors can efficiently manage their schedules, offers, and reservations. Participation in online tutoring sessions is supported through integrated WebRTC peer-to-peer technology for video calls. The full-stack web application is developed using the Next.js framework for React, while data storage is handled by a relational PostgreSQL database through Prisma ORM and managed on the Supabase platform. This thesis presents the technologies used, the system architecture, business logic, and the implementation of individual application functionalities.

**Keywords:** *booking management system, Next.js, WebRTC, audio-video calls*

## 1. Uvod

U hrvatskom se obrazovanju posljednjih dvadesetak godina jasno potvrđuje snažan rast „sjene obrazovanja” – razgranat obrazovni biznis plaćenih privatnih instrukcija i organiziranih priprema za državnu maturu i prijemne ispite, koji funkcionira paralelno s formalnim obrazovnim sustavom. Dodatne poduke postale su uobičajen dio školovanja i svakodnevica mnogih hrvatskih đaka svih uzrasta – od osnovne škole do fakulteta, a ne iznimka, unatoč tome što ovakav oblik obrazovanja otvara niz pitanja jednakosti pristupa znanju (navedene usluge dostupnije su učenicima iz socioekonomski povoljnijih obitelji te iz većih geografskih centara) te stvara pritisak na učenike i komercijalizira obrazovanje. Istraživanja Borisa Jokića i Zrinke Ristić Dedić pokazuju da je gotovo 40,0% učenika u svakoj ispitanoj generaciji (8. razred osnovne te 2. i završni razred srednje škole) koristilo privatne instrukcije u školskoj godini 2020./2021., a među maturantima ih je 38,0%. Svaki drugi maturant gimnazije poхађa instrukcije (i to 17,1% njih redovito), dok 39,2% učenika završnih razreda strukovnih i umjetničkih škola također koristi takvu pomoć, najčešće iz matematike i ključnih predmeta za upis na studij. Većina učenika i dalje preferira poхађanje repeticija uživo, međutim 20,6% maturanata i 15,9% osmaša poхађalo je privatne instrukcije u šk. god. 2020./2021. u online obliku [1]. 56,0% maturanata poхађalo je pripremne tečajeve za državnu maturu, a njih 36,1% poхађalo ih je u nekom obliku kod privatnih tvrtki. 62,9% gimnazijalaca i 50,3% učenika strukovnih škola koji planiraju upisati studij poхађalo je pripremne tečajeve u šk. god. 2020./2021., pri čemu 48,6% gimnazijalaca i 26,4% *strukovnjaka* kod privatnih tvrtki [2].

Trenutno se ponuda privatnih repeticija u Hrvatskoj najčešće pronalazi putem usmenih preporuka, online oglasnika ili grupa na društvenim mrežama, kao i na zalijepljenim oglasima na javnim površinama, gdje svoje usluge podučavanja najčešće nude uspješniji studenti i (umirovljeni) nastavnici kako bi zaradili dodatni prihod. U domaćem se medijskom prostoru povremeno izvještava da su tutori stoga i česta meta Državnog inspektorata zbog rada „na crno” jer mnogi od njih nisu registrirani kao obrtnici, a s obzirom na popularnost ovakvog oblika dodatne zarade, nadzori su učestali. Razvijena aplikacija zasad ipak ne nalazi u ovu pravnu i poreznu problematiku, već samo ističe cijene usluga te plaćanje usluga ostaje između tutora i učenika izvan same aplikacije.

Praktični dio ovog rada, aplikacija *repeticije-hr*, nastaje upravo unutar prethodno pred-

stavljenog konteksta, kao pokušaj da se postojeće, često netransparentno i nejednako tržište privatnih instrukcija učini pravednijim, preglednijim i dostupnijim većem broju učenika i studenata. Temeljna ideja aplikacije je digitalno povezati učenike i tutore na jednom mjestu, uz jasan uvid u kvalifikacije, ponudu predmeta, cijene, termine, lokaciju (uživo/online) i povratne informacije drugih korisnika, čime se smanjuje ovisnost o „vezi“ i uskim krugovima preporuka. Time se barem djelomično adresiraju problemi koje ističu Jokić i Ristić Dedić – neravnomjerna dostupnost instrukcija, netransparentnost ponude i snažna socijalna selektivnost – jer učenici iz različitih dijelova Hrvatske, neovisno o tome žive li u većim centrima ili manjim sredinama, dobivaju strukturiran digitalni prostor u kojem mogu lakše pronaći podršku koja im je potrebna.

U poglavljima koja slijede prvo će kratko biti predstavljene korištene tehnologije pri izradi aplikacije i objasnjena teoretska podloga, a zatim će se detaljno i pregledno opisati poslovna logika i način na koji je sama aplikacija implementirana.

## 2. Tehnologije i teoretska podloga

### 2.1. Next.js, React, TypeScript, Node.js, npm

**Next.js** je razvojni okvir (engl. *framework*) otvorenog kôda za React koji omogućava objedinjeni (engl. *full-stack*) razvoj web aplikacija, što znači da obuhvaća razvoj i klijentskog (engl. *front-end*) i poslužiteljskog (engl. *back-end*) dijela aplikacije. Razvijen je od strane tvrtke Vercel, a izvorni kôd dostupan je na platformi GitHub [3] gdje je prva verzija objavljena 2016. godine.

Temeljen je na JavaScriptu i izvršava se u okruženju **Node.js**, pri čemu Node.js omogućava pokretanje JavaScript kôda izvan preglednika, na poslužitelju, što je preduvjet za poslužiteljsko renderiranje i izgradnju aplikacije [4]. U praksi se Next.js projekti gotovo uvek razvijaju uz **TypeScript**, nadskup JavaScripta razvijen od strane Microsofta, koji uvodi statičku tipizaciju i provjeru kôda u vrijeme prevođenja, čime se smanjuje broj pogrešaka u složenijim aplikacijama [5].

Upravljanje ovisnostima obavlja se putem **npm**-a (engl. *Node Package Manager*) ili alternativno `pnpm/yarn/bun` upravitelja paketa, koji omogućuju jednostavnu instalaciju, ažuriranje i skriptiranje razvojnih zadataka u Node.js okruženju. Paketi (biblioteke) se preuzimaju iz javnog repozitorija [npmjs.com](https://npmjs.com), a definiraju se u datoteci `package.json` unutar projekta, a pri instalaciji se kreira i datoteka `package-lock.json` koja zaključava točne verzije ovisnosti radi konzistentnog okruženja. U direktoriju `node_modules` pohranjuju se sve instalirane ovisnosti projekta [6].

**React** (React.js) je slobodan softver otvorenog kôda tzv. FOSS (engl. *Free and Open Source Software*). Radi se o JavaScript biblioteci za izgradnju korisničkih sučelja, razvijenoj od strane Facebooka (sada Meta). Programski kôd javno je objavljen 2013. godine, a od tada je prigrđen od strane zajednice i pozicionirao se kao jedna od najpopularnijih biblioteka za razvoj web aplikacija [7].

React se često opisuje kroz tri osnovne značajke: *declarative*, *component-based* i sloganom „*learn once, write anywhere*”. Deklarativni pristup znači da razvojni programer opisuje kakvo korisničko sučelje želi u određenom stanju aplikacije, a React se brine za učinkovit prikaz i osvježavanje DOM-a (engl. *Document Object Model*), što olakšava razumijevanje i održavanje kôda. Komponentni pristup podrazumijeva da se sučelje sastoji od

malih, ponovno iskoristivih komponenti koje enkapsuliraju logiku i prikaz pa se iste komponente mogu koristiti na više mesta i u različitim projektima. Načelo „*learn once, write anywhere*” odnosi se na činjenicu da se ista znanja i koncepti mogu primijeniti u web aplikacijama, ali i u izradi mobilnih (React Native) i desktop aplikacija, bez ponovnog učenja potpuno novog programskog modela [8].

React se temelji na konceptu komponenti, pri čemu se u povijesnom razvoju razlikuju klasne (engl. *class*) komponente i funkcione (engl. *function*) komponente, koje su trenutno preporučeni način pisanja komponenti. Uvođenjem kuka (engl. *hooks*) preko Hook API-ja (engl. *Application Programming Interface*) funkcione komponente dobine su mogućnost upravljanja stanjem i *side* efektima bez potrebe za klasama, čime se dodatno pojednostavilo strukturiranje logike i ponovna iskoristivost kôda. React koristi virtualni DOM – laganu, u memoriji pohranjenu reprezentaciju strukture korisničkog sučelja kako bi učinkovito uspoređivao prethodno i novo stanje te primjenjivao samo nužne promjene u stvarnom DOM-u, što značajno poboljšava performanse složenijih aplikacija. JSX (engl. *JavaScript XML*) je sintaksno proširenje za JavaScript koji omogućuje zapis komponenti u obliku HTML-u slične sintakse, pri čemu se JSX u pozadini prevodi u pozive funkcije `React.createElement`, a razvojnom programeru olakšava čitanje i strukturiranje kôda [9]. U kombinaciji s TypeScriptom koristi se proširenje TSX (engl. *TypeScript JSX*). U novijim verzijama React uvodi i *Server Components* – komponente koje se izvršavaju isključivo na poslužitelju, generiraju HTML bez slanja JavaScript kôda na klijent i tako smanjuju veličinu paketa te povećavaju sigurnost i performanse, osobito u kombinaciji s razvojnim okvirima poput Next.js-a.

Next.js nadograđuje React dodajući brojne značajke i optimizacije koje olakšavaju izgradnju skalabilnih i visokoučinkovitih web aplikacija. Uvodi višestruke načine renderiranja stranica i napredne optimizacije koje su posebno važne za performanse i SEO. Osim klasičnog renderiranja na strani klijenta – CSR (engl. *Client-Side Rendering*), Next.js podržava poslužiteljsko renderiranje – SSR (engl. *Server-Side Rendering*), statičko generiranje stranica – SSG (engl. *Static Site Generation*) te inkrementalno statičko obnavljanje – ISR (engl. *Incremental Static Regeneration*), pri čemu se većina sadržaja isporučuje kao statički HTML, a pojedine stranice se povremeno regeneriraju kada se podaci promijene. Next.js također automatski dijeli kôd u manje dijelove (engl. *code splitting*) i koristi usmjeravanje (engl. *routing*) bazirano na strukturi direktorija, poznato kao *file-based routing*. To znači da se samo nužni dijelovi kôda učitavaju za svaku stranicu, čime se smanjuje vrijeme učitavanja

i poboljšava korisničko iskustvo. U novijim verzijama uveden je App Router s podrškom za React Server Components, strujanje (engl. *streaming*) sadržaja i Server Actions, što omogućuje da se dio logike obrade podataka izvršava na poslužitelju uz manju količinu potrebnog JavaScript kôda na klijentu, što pruža brži prikaz sadržaja korisniku [10].

Osim spomenutih značajki, Next.js sadrži integrirane alate za optimizaciju slika automatski generirajući različite veličine, koristi lijeno učitavanje (engl. *lazy loading*) i moderne formate slika, čime se poboljšavaju Core Web Vitals metrike i ukupne performanse web aplikacija. U kombinaciji s podrškom za TypeScript „*out-of-the-box*”, integracijom s Vercelom za jednostavnu isporuku te bogatim ekosustavom dodataka, Next.js se pozicionira kao cjelovit razvojni okvir za izradu skalabilnih, produkcijski spremnih aplikacija [10].

U trenutku izrade rada najnovija stabilna verzija Next.js-a je 16, dok je React dosegao verziju 19 te su iste korištene u izradi praktičnog rada, uz TypeScript verzije 5.x. Korištena verzija Node.js-a je 24.x LTS (engl. *Long Term Support*) uz npm verzije 11.x.

## 2.2. UI sloj: `shadcn/ui`, Radix UI, Tailwind CSS

Sloj korisničkog sučelja (engl. UI – *User Interface*) aplikacije temelji se na kombinaciji biblioteka `shadcn/ui`, Radix UI i Tailwind CSS, koje su usko povezane i nadograđuju se jedna na drugu.

**shadcn/ui** je zbirka unaprijed pripremljenih React komponenti (npr. avatar, botuni, bedževi, kartice za prikaz sadržaja, dijaloški okviri, padajući izbornici, polja i grupe za unos, navigacijski izbornici, tablice itd.) koje su izgrađene na Radix UI „primitivima” i stilizirane pomoću Tailwind CSS-a. Umjesto da se isporučuje kao klasična biblioteka ovisnosti, `shadcn/ui` generira stvarne izvorišne datoteke komponenti koje se kopiraju u projekt, čime razvojni programer zadržava potpunu kontrolu nad kôdom, može ga prilagoditi specifičnim potrebama aplikacije i izbjegava dodatni teret zasebne UI biblioteke dok se aplikacija izvršava (engl. *runtime*). Ovo je osobito korisno u većim projektima, gdje se očekuju dugoročno održavanje, prilagodba dizajna i dosljedan vizualni identitet [11].

**Radix UI** predstavlja skup pristupačnih (engl. *accessible*) niskorazinskih komponenti tzv. „primitiva” poput dijaloških okvira, skočnih prozora, kartica (engl. *tabs*), skočnih izbornika i slično. Fokus Radix UI-ja je na ispravnom ponašanju i pristupačnosti: osigurava ispravne ARIA (engl. *Accessible Rich Internet Applications*) atributi, upravljanje fokusom

i podršku za tipkovničku navigaciju u skladu s WAI-ARIA (engl. *Web Accessibility Initiative – ARIA*) smjernicama, dok izgled i stilizacija ostaju u potpunosti u domeni razvojnih timova [12].

**Tailwind CSS** je utilitaristički (engl. *utility-first*) razvojni okvir za CSS koji umjesto gotovih vizualnih komponenti nudi velik broj malih, jednoznačnih klasa za oblikovanje (npr. razmaci, boje, tipografija, raspored – fleksibilni i mrežni (engl. *grid*) rasporedi). Te se klase kombiniraju izravno u markup-u ili JSX/TSX kôdu, čime se smanjuje potreba za pisanjem zasebnih CSS datoteka i olakšava održavanje dosljednog dizajna u komponentnom pristupu Reacta. Tailwind je visokokonfigurabilan putem konfiguracijske datoteke koja definira dizajnerske tokene (boje, tipografiju, radijuse, razmake), a u produkcijskom okruženju alat automatski uklanja neiskorištene klase (engl. *tree shaking*), što rezultira malim konačnim CSS paketom i boljim performansama [13].

U razvijenoj aplikaciji kombinacija predstavljenih UI biblioteka omogućava izradu modernog, responzivnog i pristupačnog korisničkog sučelja uz dobru ravnotežu između brzine razvoja, kontrole nad dizajnom i tehničke kvalitete.

### **2.3. Baza podataka: PostgreSQL, Prisma ORM, Supabase**

**PostgreSQL, Prisma ORM, Supabase** (CLI, lokalni Docker razvoj, hosting, real-time funkcionalnosti)

### **2.4. WebRTC**

### **2.5. Verzioniranje kôda, hosting i deployment, email servis**

**Git, GitHub, Vercel, Resend**

### **3. Prikaz praktičnog dijela**

#### **3.1. Potpoglavlje**

*TODO: Add your content here*

##### **3.1.1. ABC**

*TODO: Add your content here*

###### **3.1.1.1. 123**

*TODO: Add your content here*

###### **3.1.1.2. 456**

*TODO: Add your content here*

##### **3.1.2. DEF**

*TODO: Add your content here*

#### **3.2. Još jedno potpoglavlje**

*TODO: Add your content here*

## **4. Zaključak**

U zadnjem poglavlju dan je zaključak.

## Literatura

- [1] B. Jokić and Z. Ristić Dedić, “Privatne instrukcije u Republici Hrvatskoj: biznis iz sjene kojeg pandemija nije ugrozila,” <http://idiprints.knjiznica.idi.hr/id/eprint/1048> (posjećeno 30.1.2026.), Institut za društvena istraživanja, Zagreb, Project Report, 2022.
- [2] ——, “Organizirane pripreme za državnu maturu i prijemne ispite: obrazovni biznis koji u Hrvatskoj i dalje raste,” <http://idiprints.knjiznica.idi.hr/id/eprint/1046> (posjećeno 30.1.2026.), Institut za društvena istraživanja, Zagreb, Project Report, 2022.
- [3] <https://github.com/vercel/next.js> (posjećeno 30.1.2026.).
- [4] O. Foundation, “About Node.js®,” <https://nodejs.org/en/about/> (posjećeno 30.1.2026.).
- [5] Microsoft, “TypeScript Docs,” <https://www.typescriptlang.org/docs/> (posjećeno 30.1.2026.).
- [6] I. npm, “npm Docs,” <https://docs.npmjs.com/> (posjećeno 30.1.2026.).
- [7] <https://github.com/facebook/react> (posjećeno 30.1.2026.).
- [8] I. Meta Platforms, “React,” <https://legacy.reactjs.org/> (posjećeno 30.1.2026.).
- [9] ——, “React Reference Overview,” <https://react.dev/reference/react> (posjećeno 30.1.2026.).
- [10] I. Vercel, “Next.js Docs,” <https://nextjs.org/docs> (posjećeno 30.1.2026.).
- [11] shadcn at Vercel, “shadcn/ui Docs,” <https://ui.shadcn.com/docs> (posjećeno 30.1.2026.).
- [12] R. by WorkOS, “Radix UI - Primitives: Introduction,” <https://www.radix-ui.com/primitives/docs/overview/introduction> (posjećeno 30.1.2026.).
- [13] I. Tailwind Labs, “Tailwind CSS Docs - Core Concepts: Styling with Utility Classes,” <https://tailwindcss.com/docs/styling-with-utility-classes> (posjećeno 30.1.2026.).

# **Dodatci**

**Popis slika**

**Popis tablica**

**Popis ispisa kôda**