**SE330-PZ - MiniLinkedIn Student**

Projekat: 4. Mini "LinkedIn" za studente (portfolio, veštine)

Student: Stevan Stojanović 5607

Asistent: Anđela Grujić

Sadržaj:

0) Uvod i vizija ………………………………………………………………………………………………..3

1) Agilna struktura i konvencije ……………………………………………………………………….……3

2) Metodologija procene i prioritizacije ……………………………………………………………………4

3) Product Backlog ………………………………………………………………………………………….4

4) Plan sprintova

Sprint 1……………………………………………………………………………………………….5

Sprint 2……………………………………………………………………………………………….7

Sprint 3……………………………………………………………………………………………..10

Sprint 4……………………………………………………………………………………………..12

Zavrsni sprint………………………………………………………………………………………15

5) Testiranje …………………………………………………………………………………………………16

6) Organizacija rada i pracenje …………………………………………………………………………..16

7) Primena 12 principa aginlong manifesta ……………………………………………………………..16

8) Prilozi ……………………………………………………………………………………………………..17

**0) Uvod i vizija**

MiniLinkedIn Student je jednostavna veb aplikacija fokusirana na kreiranje studentskih portfolija (osnovne informacije, vestine, projekti) i pregled profila drugih studenata. Sistem koristi jednostavan backend (Node.js, Express) i React interfejs, sa akcentom na jednostavnost, brz rad i minimalnu arhitekturu.

**Vizija**

Glavni korisnik – student (pocetnik / siroka publika):  
Za studenta koji zeli brzo i jednostavno da kreira profesionalni portfolio, MiniLinkedIn Student je minimalisticka veb aplikacija koja omogucava urednu prezentaciju vestina, projekata i akademskih informacija. Za razliku od kompleksnih profesionalnih mreza, ovaj proizvod je jednostavan, brz, prilagodjen studentima i ne zahteva naprednu registraciju ili komplikovane naloge.

Ciljevi MVP-a (ostvareno / delimicno):

* Kreiranje studentskog profila (ime, email, fakultet, smer, godina)
* Dodavanje vestina sa nivoima znanja
* Dodavanje projekata sa opisima i linkovima
* Lista studenata i kartice profila
* Pretraga po imenu, fakultetu i biografiji
* Filtriranje po vestinama
* Pregled javnog profila
* React frontend, Node.js backend (in-memory)

Ciljevi narednih sprintova:

* Vizuelno unapredjenje prikaza profila
* Napredniji filteri i sortiranje
* Validacija sa korisnicima radi provere korisnickog iskustva
* Stabilizacija, testiranje, refaktorisanje
* Mogucnost "recommendation" ili endorse funkcija (prosirenje)

1. **Agilna struktura i konvencije**

**1.1 Scrum ceremonije**

* Sprint Planning: izbor od 5 do 8 prica, do 30 SP, definisanje Sprint Goal-a i taskova
* Daily Scrum (najvise 15 min): Sta sam radio juce? Sta radim danas? Da li imam blokere?
* Sprint Review: demonstracija inkrementa, povratne informacije, update backloga
* Retrospektiva: analiza procesa i od 1 do 3 akcione tacke za sledeci sprint

**1.2 Definition of Ready (DoR)**

* Prica je napisana korisnickim jezikom
* Ima jasnu poslovnu vrednost
* Acceptane kriterijumi definisani (Given/When/Then)
* Dodeljen prioritet (MoSCoW)
* Procenjeni Story Points (Fibonacci)
* Poznate zavisnosti i rizici
* Spreman dizajn/mock

**1.3 Definition of Done (DoD)**

* Funkcionalnost implementirana (frontend i backend)
* Provereno manuelnim testiranjem
* Kod pregledan i formatiran
* Nema kriticnih bagova
* UI stabilan i responzivan
* Dokumentacija azurirana
* Funkcionalnost demonstrirana na review sastanku

**2) Metodologija procene i prioritizacije**

**2.1 MoSCoW**

Must – kriticne funkcionalnosti

Should – vazne, ali ne kriticne

Could – nije lose da se ima

Won’t – van obima trenutne iteracije

**2.2 $100 tehnika**

Stakeholderi rasporedjuju 100 poena na najvaznije price i prica sa vise ‘dolara' ima vecu poslovnu vrednost i veci prioritet.

**2.3 Story Points (Fibonacci)**

Procena: 1, 2, 3, 5, 8, 13  
SP predstavljaju relativnu slozenost + napor + rizik  
Procene potvrdene pomocu tehnike Planning Poker

**3) Product Backlog**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Naziv korisnicke price | MoSCoW | $100 | SP |
| US-01 | Kao student zelim da kreiram profil da bih imao svoj portfolio | MUST | $12 | 5 |
| US-02 | Kao student zelim da se prijavim izborom postojeceg naloga | MUST | $10 | 3 |
| US-03 | Kao student zelim da izmenim svoje osnovne informacije | MUST | $8 | 5 |
| US-04 | Kao student zelim da dodam vestine sa nivoima znanja | MUST | $10 | 5 |
| US-05 | Kao student zelim da obrisem vestinu | SHOULD | $4 | 2 |
| US-06 | Kao student zelim da dodam projekat sa opisom i linkom | MUST | $10 | 5 |
| US-07 | Kao student zelim da uklonim projekat | SHOULD | $5 | 3 |
| US-08 | Kao student zelim da azuriram postojeci projekat | SHOULD | $5 | 3 |
| US-09 | Kao student zelim da napisem kratak bio opis | MUST | $6 | 2 |
| US-10 | Kao posetilac zelim da vidim profil drugih studenata | MUST | $8 | 3 |
| US-11 | Kao posetilac zelim da vidim listu svih studenata | MUST | $10 | 5 |
| US-12 | Kao posetilac zelim da pretrazujem studente po imenu i fakultetu | SHOULD | $6 | 5 |
| US-13 | Kao posetilac zelim da filtriram studente po vestini | SHOULD | $6 | 5 |
| US-14 | Kao student zelim da vidim vestine na profilu drugog korisnika | COULD | $2 | 2 |
| US-15 | Kao sistem zelim da cuvam sve podatke o studentima | MUST | $8 | 3 |
| US-16 | Kao sistem zelim da omogucim izmenu profila PUT zahtevom | MUST | $8 | 3 |
| US-17 | Kao sistem zelim da generisem listu svih vestina iz profila | SHOULD | $3 | 3 |
| US-18 | Kao student zelim da vidim potvrdu da su moje izmene sacuvane | COULD | $2 | 1 |
| US-19 | Kao student zelim da imam vizuelno moderan prikaz profila | COULD | $3 | 3 |
| US-20 | Kao student zelim da aplikacija radi brzo i bez reloadovanja | COULD | $2 | 1 |

**4) Plan sprintova**

**4.1 Sprint 1 – Osnova + profili + stabilnost (najvise 30 SP)**

Sprint Goal: postaviti osnovu samog sistema, odnosno kreiranje i prijava korisnika, prikaz liste studenata i uredjivanje osnovnih podataka profila. Sprint mora da donese stabilnu osnovu za dalji razvoj vestina i projekata.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Naziv | SP | Prioritet |
| US-01 | Kreiranje profila | 5 | MUST |
| US-02 | Prijava studenta | 3 | MUST |
| US-03 | Izmena osnovnih informacija profila | 5 | MUST |
| US-09 | Dodavanje bio opisa | 2 | MUST |
| US-10 | Pregled profila drugih studenata | 3 | MUST |
| US-11 | Lista svih studenata | 5 | MUST |
| US-15 | Cuvanje podataka u sistemu | 3 | MUST |
| US-16 | PUT izmena profila | 3 | MUST |

Ukupno: 29 SP

**Acceptance kriterijumi (Gherkin format):**

US-01 – Kreiranje profila

**Given** da student unosi ime i email  
 **When** klikne na ‘Kreiraj profil’  
 **Then** sistem kreira novi profil i cuva ga u bazi (in-memory)

US-02 – Prijava studenta

**Given** da postoji lista studenata  
 **When** student izabere postojece ime i potvrdi  
 **Then** sistem ga prijavljuje i preusmerava na njegov profil

US-03 – Izmena osnovnih informacija

**Given** da je student ulogovan  
 **When** izmeni ime, email, fakultet, smer ili godinu  
 **Then** sistem azurira podatke putem PUT zahtevа

US-09 – Dodavanje bio opisa

**Given** da student pise bio  
 **When** sacuva promene  
 **Then** prikazuje se azurirani bio na njegovom profilu

US-10 – Pregled profila drugog studenta

**Given** da postoji vise studenata  
 **When** posetilac klikne na ‘Pogledaj profil’  
 **Then** vidi javne informacije o studentu

US-11 – Lista svih studenata

**Given** da aplikacija ucitava pocetnu stranu  
 **When** lista se prikazuje  
 **Then** svaki student se prikazuje u formi kartice

US-15 – Cuvanje podataka

**Given** da sistem poseduje backend  
 **When** kreira ili menja profil  
 **Then** podaci se cuvaju u memoriji i vracaju kroz API

US-16 – PUT izmena

**Given** da student azurira svoj profil  
 **When** forma se potvrdi  
 **Then** backend vrsi izmenu preko PUT /api/users/:id

**Task breakdown:**

Backend (Node + Express)

* Setup Express projekta i folder strukture
* Endpoint: POST /api/users za kreiranje korisnika
* Endpoint: GET /api/users za listu korisnika
* Endpoint: GET /api/users/:id za detalje
* Endpoint: PUT /api/users/:id za azuriranje
* In-memory model i validacija
* Manualni testovi API-ja (Postman)

Frontend (React)

* Setup React projekta
* Rutiranje
* Login stranica (izbor postojeceg studenta)
* Kreiranje novog profila (forma i backend poziv)
* Stranica “Moj profil” (osnovne informacije i biografija)
* PUT azuriranje profila
* Stranica “Lista studenata”
* Stranica “Pregled profila”
* UI stilizacija (kartice, layout)

Testiranje i dokumentacija

* Smoke test backend API-ja
* Smoke test frontenda
* Kratka tehnicka dokumentacija API ruta
* Changelog + DoD zapis

**Definition of Done (Sprint 1)**

* Sve price implementirane bez kriticnih bugova
* Backend API radi stabilno
* Frontend prikazuje sve osnovne funkcionalnosti
* Login i kreiranje profila funkcionalni
* Lista studenata i pregled profila se ispravno ucitavaju
* PUT izmena radi pouzdano
* Code review odradjen (self-review)
* Minimalna dokumentacija upisana
* Demo pripremljen za Review

**Plan testiranja (Sprint 1)**

Unit / API:

* Test kreiranja korisnika (POST)
* Test dobijanja liste (GET)
* Test pojedinacnog profila (GET)
* Test azuriranja profila (PUT)

UI testovi (manuelni):

* Kreiranje profila –> odjava –> ponovna prijava
* Izmena imena i fakulteta
* Vidljivost bio opisa
* Navigacija studentska lista –> detaljan profil

Validacija sa korisnikom:

* 2–3 studenta prolaze kroz tok “kreiraj profil i pregledaj druge”

**Rizici i mitigacija**

|  |  |  |
| --- | --- | --- |
| **Rizik** | **Uticaj** | **Mitigacija** |
| In-memory backend gubi podatke na restartu | Srednji | Dokumentovati ogranicenje, stabilizovati API |
| UI moze biti konfuzan | Srednji | Rano korisnicko testiranje |
| Veliki broj fetch poziva | Nizak | Memoizacija i minimalno ponovno ucitavanje |
| Moguci CORS problemi | Nizak | Proxy u Vite konfiguraciji |

**Review (agenda)**

* Demo kreiranja profila
* Demo liste studenata
* Demo uredjivanja osnovnog profila
* Povratne informacije
* Azuriranje backlog prioriteta

**Retrospektiva (akcije)**

* Ujednaciti stil komponenti (UI konzistencija)
* Uvesti simple validator za unos podataka
* Podeliti taskove jos granularnije za Sprint 2

**4.2 Sprint 2 – Vestine, projekti, pretraga**

Sprint Goal: dodati kljucne funkcije portfolija, odnosno upravljanje vestinama, dodavanje projekata i uvodjenje pretrage i filtriranja studenata. Ovo povecava vrednost aplikacije jer studenti sada mogu da prikazu svoje znanje i projekte, a drugi mogu da ih lakse pronadju.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Prica | SP | Prioritet |
| US-04 | Dodavanje vestina | 5 | MUST |
| US-05 | Brisanje vestine | 2 | SHOULD |
| US-06 | Dodavanje projekta | 5 | MUST |
| US-07 | Brisanje projekta | 3 | SHOULD |
| US-12 | Pretraga studenata po tekstu | 5 | SHOULD |
| US-13 | Filtriranje po vestini | 5 | SHOULD |

Ukupno: 25 SP

**4.2.1 Detaljni plan**

Trajanje: 2 nedelje

Tim i uloge:

* 1 React developer
* 1 QA
* 1 Product Owner

**Task breakdown:**

Backend (Node.js i Express):

* Endpoint: dodavanje vestine u profil (PUT /api/users/:id)
* Endpoint: brisanje vestine
* Endpoint: dodavanje projekta
* Endpoint: brisanje projekta
* Prosirenje modela studenta (skills, projects)
* Pretraga u GET /api/users?q=
* Filtriranje u GET /api/users?skill=
* Endpoint: GET /api/skills (automatska lista vestina)

Frontend (React):

* UI komponenta za dodavanje vestine
* UI brisanje vestine iz liste
* UI forma za dodavanje projekta
* UI brisanje projekta
* Prosirenje ProfilePage.jsx (sekcije, layout)
* StudentsPage.jsx – polje za pretragu
* StudentsPage.jsx – dropdown za filtriranje po vestini
* Integracija sa backend API-jem

Testiranje i dokumentacija:

* Smoke test API-ja (projekti, vestine)
* UI testovi: dodavanje/brisanje vestine
* UI testovi: dodavanje/brisanje projekta
* Testiranje pretrage i filtriranja
* Azuriranje dokumentacije i changelog

**Acceptance kriterijumi:**

US-04 – Dodavanje vestine

**Given** student je na svom profilu  
 **When** unese naziv vestine i nivo i klikne ‘Dodaj’  
 **Then** nova vestina se prikazuje u listi i cuva preko PUT zahteva

US-05 – Brisanje vestine

**Given** student ima vestinu u profilu  
 **When** klikne ‘Ukloni’  
 **Then** vestina nestaje iz liste i backend je azuriran

US-06 – Dodavanje projekta

**Given** student je na svom profilu  
 **When** unese naziv, opis i link projekta  
 **Then** projekat se pojavljuje u listi i cuva u backend-u

US-07 – Brisanje projekta

**Given** postoji projekat u profilu  
 **When** klikne ‘Ukloni projekat’  
 **Then** projekat je uklonjen iz frontenda i backend-a

US-12 – Pretraga studenata

**Given** postoji vise studenata  
 **When** korisnik unese tekst u polje za pretragu  
 **Then** prikazuju se samo studenti koji odgovaraju pojmu

US-13 – Filtriranje po vestini

**Given** postoji vise studenata sa razlicitim vestinama  
 **When** korisnik odabere vestinu iz padajuce liste  
 **Then** prikazuju se samo studenti koji poseduju tu vestinu

**Definition of Done (Sprint 2)**

* Vestine i projekti 100% funkcionalni
* Pretraga i filtriranje rade fluidno
* UI sekcije profila kompletirane
* Backend i frontend dosledni u strukturi podataka
* Nema kriticnih bugova
* Dokumentacija dopunjena
* Testovi sprovedeni (API i UI)
* Feature demonstriran tokom review-a

**Plan testiranja (Sprint 2)**

Unit/API testovi:

* PUT dodavanje/brisanje vestine
* PUT dodavanje/brisanje projekta
* GET pretraga i filtriranje
* GET lista vestina

UI testovi (manuelni):

* Dodavanje vestine –> odmah prikazano
* Brisanje vestine –> odmah nestaje
* Dodavanje projekta –> vidljivo u listi
* Brisanje projekta –> nestaje iz liste
* Pretraga po imenu –> radi ocekivano
* Filtriranje po vestini –> radi ocekivano

Validacija sa stvarnim korisnikom:

* Student uspeva da doda realne vestine
* Student dodaje svoj GitHub projekat
* Posetilac moze da pronadje profil po ‘React’ vestini

**Rizici i mitigacija**

|  |  |  |
| --- | --- | --- |
| Rizik | Uticaj | Mitigacija |
| Preklapanje vestina sa istim nazivom | nizak | deduplikacija skill liste |
| Projekti bez opisa | srednji | obavezna polja u formi |
| Pretraga spora zbog vise filtera | nizak | jednostavno pretrazivanje u memoriji |
| UI moze biti nejasan | srednji | feedback studenata u review-u |

**Review (agenda):**

* Demo dodavanja i brisanja vestina
* Demo dodavanja i brisanja projekata
* Demo pretrage i filtriranja
* Feedback korisnika
* Moderacija backlog prioriteta za Sprint 3

**Retrospektiva (akcije):**

* Standardizovati format vestina (stringovi bez duplikata)
* Dodati validaciju za obavezna polja u projektu
* Uvesti vizuelne potvrde uspesnog dodavanja/brisanja

**4.3 Sprint 3 – Validacija sa korisnikom i hardening**

Sprint Goal: potvrditi vrednost aplikacije kroz testiranje sa stvarnim korisnicima i poboljsati kvalitet sistema na osnovu rezultata. Pored toga, uvodimo funkcionalnosti koje unapredjuju iskustvo korscenja: bolji vizuelni prikaz profila, empty state-ove i sistem notifikacija o uspesnom cuvanju.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Prica | SP | Prioritet |
| US-18 | Poruka o uspesnom cuvanju | 1 | COULD |
| US-19 | Moderniji vizuelni prikaz profila | 3 | COULD |
| US-14 | Prikaz vestina u javnom profilu (razrada) | 2 | COULD |
| US-12 (prosirenje) | Pretraga – poboljsanja UX-a | 3 | SHOULD |
| US-13 (prosirenje) | Filtriranje – poboljsanja UX-a | 3 | SHOULD |
| US-03 (prosirenje) | Empty state za neuspesno ucitavanje profila | 2 | SHOULD |
| US-10 (prosirenje) | Poboljsanja pregleda tudjih profila | 3 | SHOULD |
| US-XX | Sistem mini-onboardinga pri prvom ulasku | 4 | COULD |

UKUPNO: 21 SP

**4.3.1 Detaljni plan**

Trajanje: 2 nedelje

Tim: 1 dev, 1 QA, 1 PO

**Task breakdown:**

Frontend

* Implementacija ‘saved successfully’ notifikacije
* Redizajn ProfilePage (layout sekcija, spacing, vizuelne kartice)
* Empty state na stranici /students kada nema rezultata
* Empty state kada profil ne postoji (‘Profil nije pronađen’)
* Poboljsanje StudentsPage (pretraga brza, debounce)
* Poboljsanje filtriranja (dropdown UX, loading states)
* Mini onboarding modal (1–2 ekrana pri prvom ulasku)
* Refaktorisanje UI komponenti za bolju citljivost

Backend

* Dodavanje poruke pri uspesnom cuvanju (status i msg)
* Poboljsanje strukture vracenih podataka (dosledniji format)
* Error handling za lose ID-e i invalidne projekte
* Dodavanje fallback vrednosti (empty arrays)

Testiranje i validacija

* Kreiranje scenarija za test korisnike
* Sprovodjenje od 3 do 5 korisnickih sesija
* Analiza komentara i identifikacija problema
* Popravke na osnovu feedback-a
* UI regression test (pretraga i filtriranje)
* Finalna dokumentacija sprinta

**Acceptance kriterijumi:**

US-18 – Poruka o cuvanju

**Given** student je uneo izmene  
 **When** klikne ‘Sacuvaj’  
 **Then** prikazuje se vizuelna potvrda

US-19 – Moderniji profil

**Given** korisnik otvara profil  
 **When** otvori svoj ili tudji profil  
 **Then** prikazuje se novi UI (kartice, sekcije, istaknut bio)

US-14 – Vestine u javnom profilu

**Given** student ima vestine  
 **When** posetilac otvori njegov profil  
 **Then** vestine su prikazane kao chipovi sa nivoima

US-12 (prosirenje) – Pretraga UX

**Given** korisnik brzo kuca pojam  
 **When** pauzira kucanje  
 **Then** aktivira se ‘debounced’ pretraga i rezultati se azuriraju bez skokova

US-13 (prosirenje) – Filtriranje UX

**Given** postoji dropdown sa vestinama  
 **When** korisnik izabere vestinu  
 **Then** prikazuje se loading stanje i jasna lista filtriranih studenata

US-03 (prosirenje) – Empty state profila

**Given** ID ne postoji  
 **When** korisnik poseti nepostojeci profil  
 **Then** prikazuje se poruka ‘Profil nije pronađen’

US-10 (prosirenje) – Bolji pregled tudjih profila

**Given** student otvara tudji profil  
 **When** gleda bio i projekte  
 **Then** sekcije su preglednije, uz naglasene naslove

US-XX – Onboarding

**Given** korisnik prvi put otvara aplikaciju  
 **When** otvori glavni ekran  
 **Then** prikazuje se onboarding modal koji objasnjava osnovne funkcije

**Definition of Done (Sprint 3)**

* Vizuelna unaprdjenja u potpunosti implementirana
* UX pretrage i filtriranja poboljsan
* Onboarding funkcionise samo prvi put
* Empty state-ovi dosledni
* Svi kriticni bugovi uklonjeni
* Povratne informacije iz korisnickih testova obradjene
* Dokumentacija azurirana
* Sprint Review demo pripremljen

**Plan testiranja:**

Unit / API testovi:

* Test konzistencije backend odgovora
* Test error-handlinga za pogresne ID-je
* Test pretrage sa i bez parametara
* Test filtriranja po vestini

UI testovi:

* Onboarding modal –> pojavljuje se samo jednom
* ‘Sacuvano’ poruka radi u svim formama
* Empty state se prikazuje kada treba
* Pretraga radi debounced (bez trzanja)
* Filtriranje ispravno smanjuje rezultate

Korisnicko testiranje:

Scenariji:

* ‘Napravi profil → dodaj vestine → pregledaj druge studente’
* ‘Pretrazi studente po React → otvori prvi profil’
* ‘Pokusaj da otvoris nepostojeci profil’

Pitanja:

* Šta ti je bilo jasno?
* Šta ti je bilo zbunjujuce?
* Šta nikako ne radi kako treba?
* Šta bi promenio?

**Rizici i mitigacija**

|  |  |  |
| --- | --- | --- |
| Rizik | Uticaj | Mitigacija |
| UI previse zbijen | Srednji | Unapredjenje spacinga i layouta |
| Pretraga previse opterecuje frontend | Nizak | Debounce na 250ms |
| Los UX pri filtriranju | Srednji | Loading states i vizuelni feedback |
| Korisnici ne shvataju funkciju onboarding modala | Nizak | Pojednostaviti tekst |

**Review (agenda):**

* Demo onboarding modala
* Demo poboljsane liste studenata
* Demo empty state-ova
* Demo novih UI elemenata
* Analiza feedbacka korisnika

**Retrospektiva (akcije):**

* Uvesti globalni UI spacing sistem
* Razmotriti dohvat svim vestina iz baze pri svakom load-u
* Poboljsati vizuelni prikaz projekata (thumbnail, link ikonica)

**4.4 Sprint 4 – Stabilnost i napredne funkcije**

Sprint Goal: uvesti naprednije funkcije portfolija (sortiranje, mini-analitika profila, prosirena pretraga), optimizovati performanse i stabilnost, i pripremiti sistem za finalni sprint i demonstraciju.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Prica | SP | Prioritet |
| US-12 (prosirenje 2) | Napredni filteri i sortiranje studenata | 8 | SHOULD |
| US-13 (prosirenje 2) | Filtriranje po vise vestina | 5 | COULD |
| US-19 (prosirenje) | Modernizacija UI profila (verzija 2) | 5 | COULD |
| US-11 (prosirenje) | Poboljsanja liste studenata (paging/lazy load) | 5 | COULD |
| US-03 (prosirenje 2) | Form validation i UX poboljsanja | 3 | SHOULD |

UKUPNO: 26 SP

**4.4.1 Detaljni plan**

Trajanje: 2 nedelje

Tim: 1 dev, 1 QA

**Task breakdown:**

Backend (Node.js)

* Prosiriti upit /api/users sa parametrom sort= (ime, godina, fakultet)
* Implementirati filtriranje po vise vestina: ?skills=react,sql
* Optimizacija pretrage u memoriji (indeksirane strukture)
* Dodavanje limit i offset parametara zbog paging-a
* Standardizacija svih API odgovora (status, message, data)
* Refaktorisanje error middleware-a

Frontend (React)

* UI za sortiranje studenata (dropdown i live update)
* UI za selekciju vise vestina (multi-select)
* Lazy loading/paging studentske liste
* Redizajn profila (verzija 2 — bolji layout i spacing)
* Validacija formi: obavezna polja, format linkova, prikaz gresaka
* Refaktorisanje StudentsPage u manje komponente

Tehnicke optimizacije i QA

* Monitoring performansi frontenda (React Profiler)
* Smarterni cache kod pretrage i filtera
* Uklanjanje zastarelog koda i cleanup
* UI regresioni test (studentska lista i pretraga)
* Testiranje validacija i edge case podataka
* Azuriranje dokumentacije (API, UI, struktura)

**Acceptance kriterijumi:**

US-12 – Napredna pretraga/sortiranje

**Given** korisnik je na studentskoj listi  
 **When** klikne na sortiranje (A-Z, Z-A, godina, fakultet)  
 **Then** lista se odmah azurira bez loading trzanja

US-13 – Filtriranje po vise vestina

**Given** student ima vise vestina  
 **When** korisnik izabere npr. React i SQL  
 **Then** prikazuju se samo studenti koji imaju obe vestine

US-19 – Modernizacija profila (v2)

**Given** korisnik otvori profil  
 **When** profil se ucita  
 **Then** sekcije su vizuelno jasne (bio, obrazovanje, projekti, vestine)

US-11 – Paging studentske liste

**Given** ima 30+ studenata  
 **When** korisnik scroll-uje  
 **Then** lista se dopunjuje automatski (infinite scroll)

US-03 – Validacija formi

**Given** korisnik pokusava da sacuva pogresne podatke  
 **When** pokusa da sacuva  
 **Then** prikazuju se jasne validacione poruke (email format, prazan naziv…)

**Definition of Done (Sprint 4):**

* Napredni filteri i sortiranja rade
* Lista studenata optimizovana
* Profil vizuelno uskladjen i stabilan
* Validacije uvedene globalno
* API odgovori dosledni
* Svi kriticni bugovi uklonjeni
* Dokumentacija uredno dopunjena
* Review demo pripremljen

**Plan testiranja:**

API testovi

* Filtriranje po vise vestina
* Sortiranje po imenu/fakultetu
* Error handling standardizovan
* Paging vraca konzistentne rezultate

UI testovi

* Sortiranje radi bez skokova
* Multi-select filter
* Infinite scroll ponasanje
* Validacija svih formi (bio, projekti, profil)

Performanse

* React Profiler: render vreme manje od 16 ms
* Pretraga najvise 250ms sa debounce-om

Korisnicka validacija

* ‘Pronadji React developera iz generacije 2026’
* ‘Dodaj projekat –> unesi loš link –> vidi gresku’

**Rizici i mitigacija**

|  |  |  |
| --- | --- | --- |
| Rizik | Uticaj | Mitigacija |
| Previse kompleksan multi-filter | srednji | ograniciti izbor na max 3 vestine |
| Infinite scroll moze trošiti resurse | srednji | server-side paginacija |
| Vizuelna konzistencija narusena | nizak | centralizovati stilove |
| Preveliki broj front-end re-rendera | srednji | memoizacija i useCallback |

**Review (agenda):**

* Demo naprednih filtera
* Demo sortiranja
* Demo infinite scroll-a
* Demo unapredjenog profila
* Demo form validation
* Prikupljanje povratnih informacija

**Retrospektiva (akcije):**

* Definisati jasne UI stilove (Design Tokens)
* Prebaciti deo logike u custom hook-ove (performance improvement)
* Pojednostaviti StudentsPage rendering strukturu

**4.5 Zavrsni Sprint – Stabilizacija i demonstracija**

Sprint Goal: stabilizovati aplikaciju, ukloniti sve kljucne bugove, zavrsiti dokumentaciju i pripremiti demonstraciju za zavrsnu predaju. Ovo je finalna faza projekta. Fokus nije na novim funkcijama, vec na kvalitetu, stabilnosti i prezentaciji gotovog proizvoda.

**Zadaci završnog sprinta:**

|  |  |  |
| --- | --- | --- |
| Br. | Task | Vreme |
| 1 | Bug-fix lista (P1 –> P3 prioritet) | kontinuirano |
| 2 | Poliranje UI (spacing, boje, tipografija, kontrast) | 4h |
| 3 | Finalna UX provera (usaglasenost svih ekrana) | 3h |
| 4 | Refaktorisanje kriticnih delova frontenda (pretraga, studenti lista) | 6h |
| 5 | Refaktorisanje backend strukture odgovora | 3h |
| 6 | Finalni regression testovi (API i UI) | 6h |
| 7 | Testiranje edge-case scenarija (prazni nizovi, nepostojeci ID, duplikati) | 2h |
| 8 | Stabilnost paginacije i pretrage | 2h |
| 9 | README / API dokumentacija / How-to guide | 4h |
| 10 | Zavrsni changelog | 2h |
| 11 | Kratko korisnicko uputstvo (korak po korak) | 3h |
| 12 | Screenshot dokumentacija (relevantni ekrani) | 2h |
| 13 | Snimanje kratkog demo videa (screen-record) | 3h |
| 14 | Finalna priprema za Review i odbranu | 2h |
| 15 | Retro: ‘sta smo naucili / sledeci koraci’ | 2h |

**Definition of Done (Zavrsni sprint):**

* Svi kriticni bugovi zatvoreni (P1 i P2 = 0)
* UI/UX vizuelno uskladjeni i bez nedovrsenih mesta
* Validacija inputa radi globalno u aplikaciji
* Dokumentacija kompletna (SRS deo, backlog, sprintovi, testovi)
* API dokumentacija za sve rute azurirana
* README, How-to-run i Changelog gotovi
* Kompletan demo pripremljen
* Aplikacija je spremna za predaju i demonstraciju

**Plan testiranja (Finalni sprint):**

API testovi:

* GET /api/users (empty –> populated)
* GET /api/users/:id (validan i nevalidan ID)
* PUT /api/users/:id (validacija i edge-cases)
* Pretraga i filtriranje — kombinovani testovi

UI testovi:

* Kreiranje novog profila –> izmena –> brisanje vestine –> dodavanje projekta
* Pretraga po imenu i fakultetu
* Filtriranje po vestini i po vise vestina
* Infinite scroll / paging
* Onboarding (prikazuje se samo prvi put)
* Notifikacija ‘Sacuvano’ radi globalno

UX testovi:

* Sve stranice imaju koherentan izgled
* Profil citljiv i vizuelno balansiran
* Studenti lista stabilno reaguje na filtere
* Projekti lepo prikazani
* Vestine prikazane kao chipovi bez prelamanja

Security / Stability

* Validacija inputa
* Ocuvanje tipova podataka
* Bez padova aplikacije pri sporoj mrezi

**Review (agenda):**

* Prikaz kompletne aplikacije (end-to-end tok)
* Kreiranje profila, izmena, prikaz vestina i projekata
* Lista studenata sa pretragom i filtrima
* Prikaz tudjeg profila
* UI/UX poboljsanja iz prethodnih sprintova
* Performanse i stabilnost
* Analiza dokumentacije
* Diskusija o sledecim koracima (ako se projekat nastavlja)

**Retrospektiva (akcije)**

* Poboljsati strukturu komponenti (modularnost i reusable blokovi)
* Izvuci vise logike u custom hook-ove
* Isplanirati mogucnost buduceg prelaska na bazu (Mongo/Postgres)
* Uvesti CI skriptu za automatsko testiranje
* Razmotriti prosirenje aplikacije za realnu upotrebu studenata

**5) Testiranje**

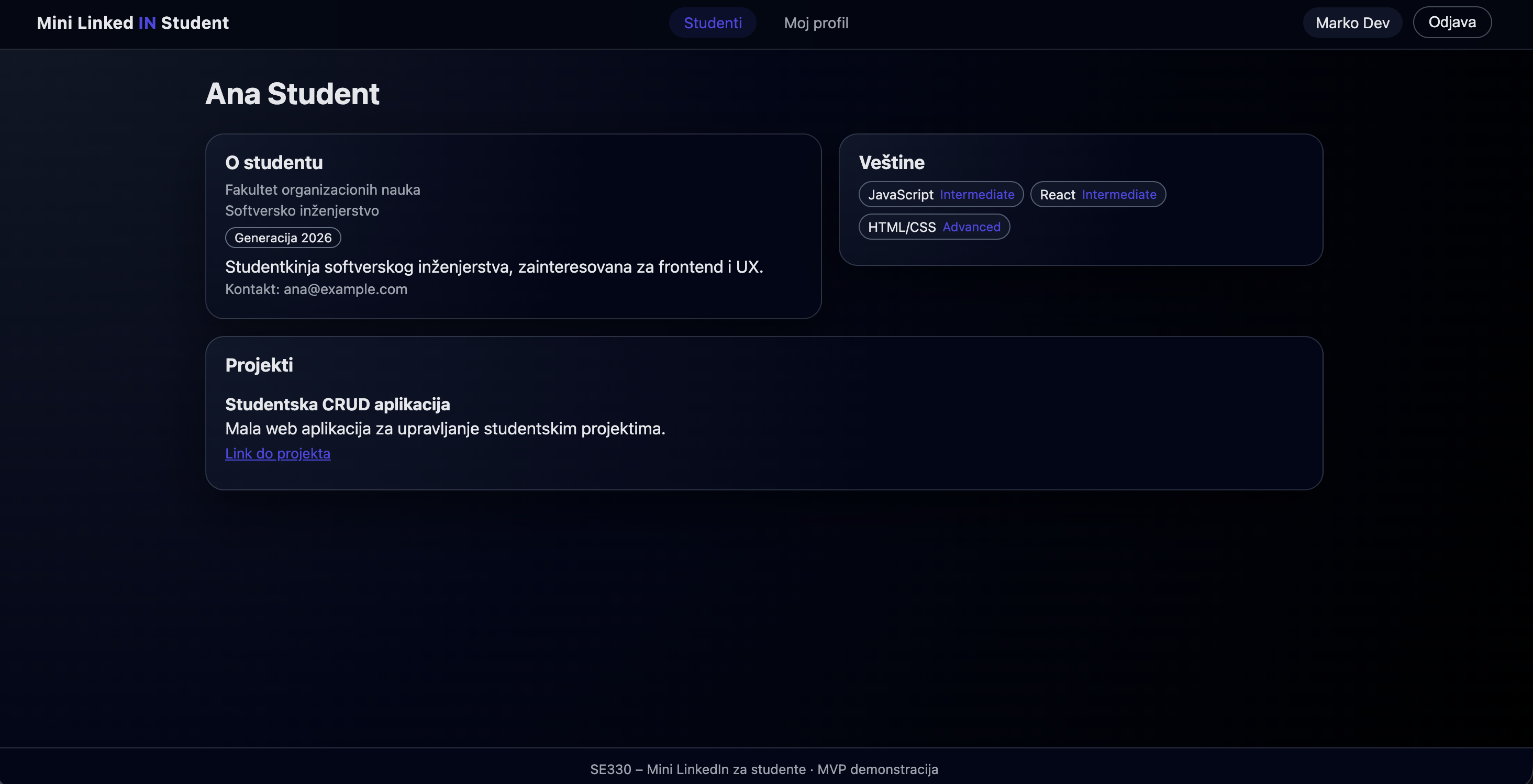
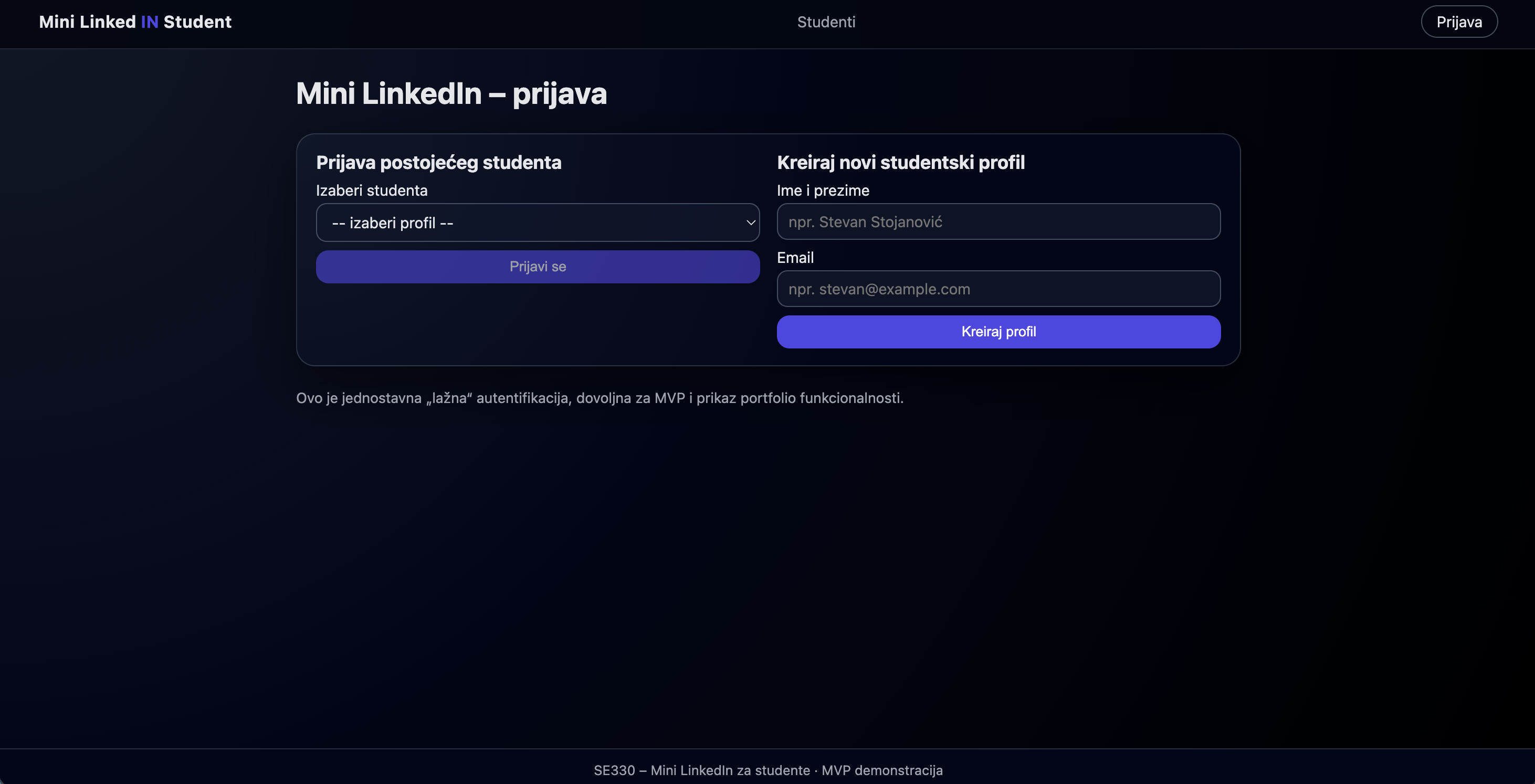
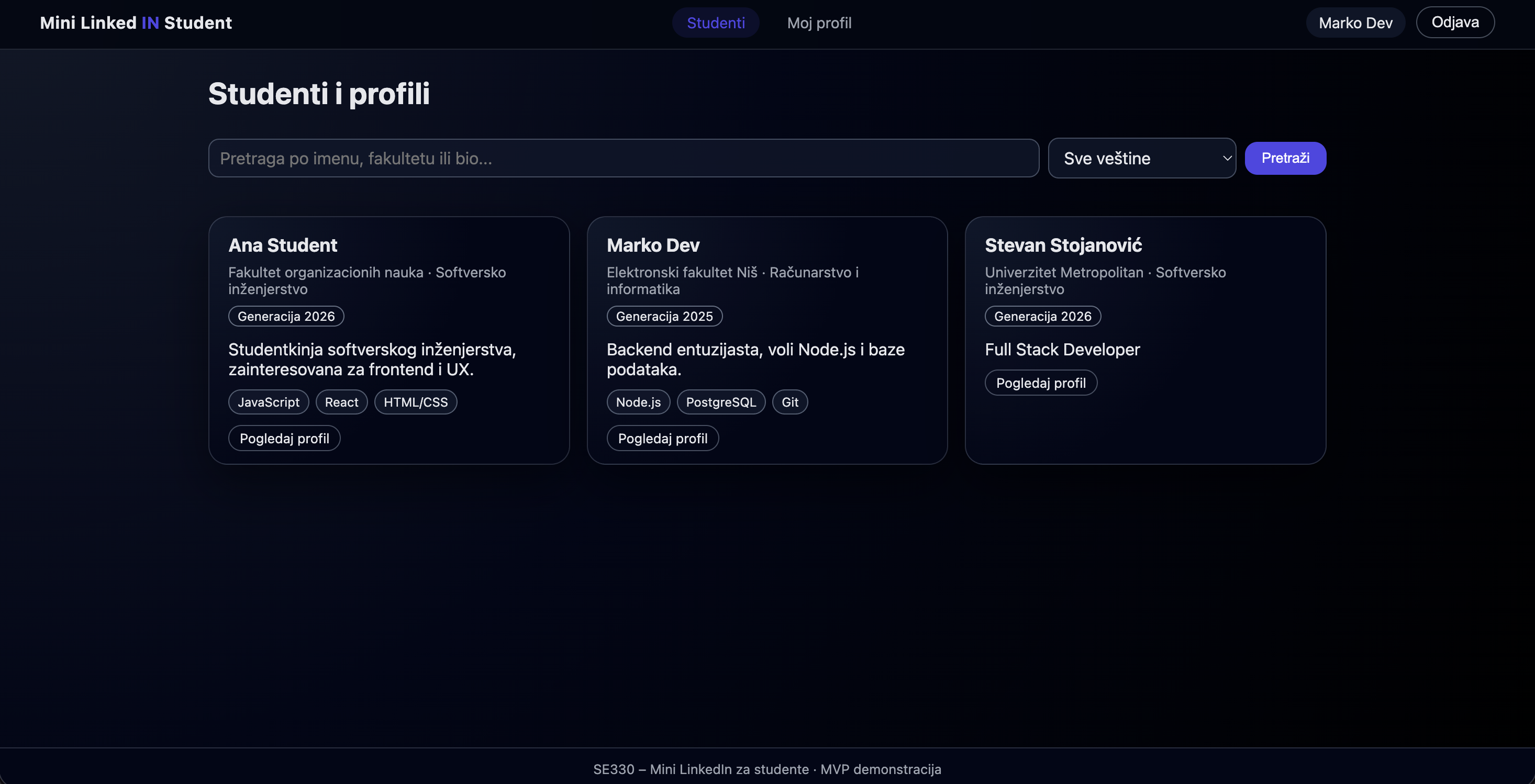
• Unit testovi backend logike  
• UI testovi kljucnih tokova  
• Validacija sa korisnicima (od 3 do 5 studenta)

**6) Organizacija rada i pracenje**

• Task board: Backlog –> Ready –> In Progress –> Review –> Testing –> Done  
• WIP limit: max 2 aktivna taska  
• Taskovi manji od 1 dana  
• Price da je najvise 8 SP u sprintu

**7) Primena 12 principa agilnog manifesta (sazeto)**

* Zadovoljstvo korisnika – cesti inkrementi aplikacije (svaki sprint donosi vidljive funkcije: profili, vestine, projekti)
* Dobrodosle promene – grooming i re-prioritizacija backlog-a prema feedback-u studenata
* Cesta isporuka – funkcionalni inkrement svakih 2 nedelje (Sprint Review)
* Tim i biznis zajedno – stalna komunikacija sa mentorom i korisnicima kroz sve faze sprintova
* Motivisani pojedinci – jasni ciljevi sprinta i ownership nad kljucnim modulima (frontend, backend, UX)
* Licem-u-lice komunikacija – kratki daily (ili async update ako radi jedna osoba)
* Radni softver – fokus na DoD, svaki sprint donosi potpuno funkcionalne profile, listu studenata i pretragu
* Odrzivi tempo – balansirani sprintovi (<=30 SP), WIP limiti, realna procena kapaciteta
* Tehnicka izvrsnost – refaktorisanje UI logike, API stabilizacija, validacija inputa i testiranje
* Jednostavnost – minimalni UI otpor (jednostavne forme, jasno prikazani podaci)
* Samoorganizacija – sloboda da se tehnicka resenja i struktura komponenti biraju prema najboljoj praksi
* Kontinuenta refleksija – retros sa akcionim tackama posle svakog sprinta (poboljsanje pretrage, validacije, layout-a)

**9) Prilozi**

