Programsko inženjerstvo ak.god 2023./2024

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

TeachABit

Tim: <G12.1> | Članovi | Uloge | |:----------|:----------| | Dorijan Jančić | Devops | | Matej Jurišić | Full stack | | Ivan Mitar | Frontend | | Dino Gabrić | Senior Memer, Frontend | | Martin Vidmar | Frontend, Dizajn | | Mateo Toić | Backend | | Tomislav Sesar | Backend |

Nastavnik: Vlado Sruk

Opis projekta

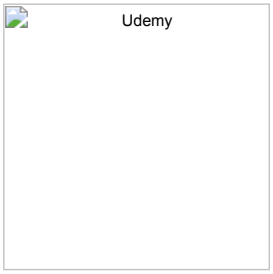
Ova platforma omogućuje korisnicima raznovrsne mogućnosti učenja i razmjene znanja. Korisnici mogu objavljivati vlastite tečajeve i radionice, te aktivno sudjelovati u diskusijama foruma. Forumi omogućuju razmjenu ideja, postavljanje pitanja i davanje odgovora. Cilj platforme je unaprijediti pristup obrazovanju i olakšati povezivanje između predavača i zainteresiranih korisnika.

Slična rješenja

Postojeća slična rješenja za tečajeve su Udemy (https://www.udemy.com/), Khan Academy (https://www.khanacademy.org/) i Brilliant (https://brilliant.org). Postojeća slična rješenja za forume su Reddit (https://www.reddit.com/) i Stack Overflow (https://stackoverflow.com/). Cilj projekta je ujediniti snage iz ova dva svijeta i povećati komunikaciju između predavača i korisnika.

Udemy

Udemy je platforma koja instruktorima omogućuje izradu online tečajeva. Korištenjem Udemy-jevih alata za razvoj tečajeva, instruktori mogu postavljati videozapise, izvorni kod za programere, PowerPoint prezentacije, PDF-ove, audio, ZIP datoteke i bilo koji drugi sadržaj. Ovo rješenje prati način prikaza instrukcija i radionica koje želimo implementirati, ali ne sadrži forume.



Khan Academy

Khan Academy je besplatna platforma za online tečaje. Koristi interaktivne video lekcije i vježbe za učenje. Prednost platforme je besplatni edukativni sadržaj, no ne omogućuje kreiranje svojih korisničkih lekcija, niti sadrži forume ili bilo kakav drugi način putem kojeg korisnici mogu kreirati svoj sadržaj.

Brilliant

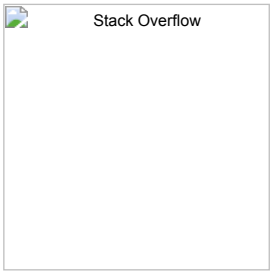
Brilliant je interaktivna online platforma koja omogućuje učenje kroz rješavanje problema. To je još jedan način online edukacije, ali nema mogućnost kreiranja ikakvog korisničkog sadržaja za učenje ili komuniciranje. Također koristi različite vježbe i testove za učenje te se ne oslanja previše na video lekcije.

Reddit

Reddit je platforma koja ne obuhvaća samo edukativni sadržaj. To je stranica za forume gdje korisnici mogu postavljati pitanja i pokretati rasprave. Za razliku od Reddita, projekt omogućuje kreiranje online tečaja i radionica. Reddit ne omogućuje označavanje točnog odgovora.

Stack Overflow

Stack Overflow je platforma koja se bavi računalnim temama. To je pitaj-odgovori stranica. Korisnik postavi pitanje, ostali korisnici pošalju svoje odgovore i na kraju vlasnik pitanja može označiti točan odgovor. Ova platforma također ne omogućuje kreiranje ikakvih tečajeva ili radionica.



Za koga je stranica namijenjena

Postoji širok spektar korisnika koji bi mogli biti zainteresirani za ovaj projekt. Učitelji, profesori, instruktori i predavači koji žele dijeliti svoja znanja i iskustva putem tečajeva i radionica. Studenti i učenici koji žele proširiti svoja znanja i unaprijediti vještine. Hobisti i entuzijasti koji se žele educirati o temama izvan svoje struke. Prosječna osoba koja jednostavno ima pitanja za koja mu trebaju tuđa mišljenja.

Korisničke grupe

- **Korisnik**

Korisnici koji nisu autentificirani u sustav imaju pravo pregleda javnih tečajeva i tuđih profila. Korisnici koji jesu autentificirani u sustav mogu pristupiti javnim tečajevima te ih ocjenjivati i ostavljati komentare. Također mogu prijavljivati loše/pogrešne tečajeve i objave koji onda idu Moderatorima na pregled. Imaju pravo odgovarati na postojeće i stvarati nove objave u forumu. Korisnici mogu i objavljivati svoje tečajeve. Imaju mogućnost slanja zahtjeva za verifikaciju profila na temelju kvalitete/ocjena svojih prethodnih tečaja. Kada korisnik bude verificiran, dobiva mogućnost organiziranja radionica i objavljivanja privatnih tečajeva koji zahtijevaju plaćanje kako bi im se pristupilo.

- **Moderator**

Moderatori održavaju platformu, imaju mogućnost brisanja objavljenih tečajeva, uklanjanja komentara/objava na forumima, te blokiranje korisničkih profila.

- **Administrator**

Administrator je odgovoran za nadgledanje i održavanje cijele platforme. Ima najviša prava, uključujući mogućnost postavljanja i uklanjanja moderatora kao i brisanje korisničkih računa.

Struktura platforme

Glavna navigacija

- Na početnom ekranu korisnik može navigirati između **tečajeva**, **foruma** i **radionica**.

Sadržaj stranice tečajeva

- Traka za pretraživanje tečajeva
- Popis istaknutih ili popularnih tečajeva
- Popis tečajeva sa njihovim osnovnim informacijama, klikom na tečaj otvara se stranica sa detaljima tečaja

Detalji tečaja

- Ako je za pristup tečaju potrebno plaćanje, otvara se ekran s ključnim informacijama i sučeljem za plaćanje

Ako je korisnik platio pristup tečaju ili je tečaj besplatan prikazuju se sljedeće stavke:

- Navigacija za odabir lekcije tečaja
- Klikom na lekciju skrivaju se osnovni podatci o tečaju i prikazuje se sadržaj lekcije
- Svaka lekcija sadrži razni sadržaj poput teksta, slika, poveznica, videa
- Korisnici mogu ostavljati komentare za svaku lekciju te ocjenjivati tečajeve
- Korisnici mogu označiti tečaj kao *favorite* kako bi ga kasnije lako pronašli na svom profilu

Sadržaj stranice radionica

- Popis svih dostupnih radionica
- Traka za pretraživanje radionica
- Osnovne informacije o radionicama poput teme, predavača i slično
- Broj prijavljenih i kapacitet radionice
- Datum i vrijeme održavanja

Detalji radionice

- Detaljan opis radionice (tematika, što će se sve obraditi i slično)
- Popis ishoda učenja
- Preduvjeti (npr. osnovno znanje programiranja)
- Mogućnost prijave i plaćanje
- Prije početka radionice predavač u sustav unosi poveznice/upute za pristup radionici koja se onda prosljeđuje svim prijavljenim korisnicima na email adresu
- Svaki predavač će imati odvojene statistike za recenzije radionica i tečajeva

Sadržaj stranice foruma

- Traka za pretraživanje i pronalazak foruma
- Popis objava
- Svaka objava će imati naslov, tekst i kategorije
- Klikom na objavu otvara se stranica s detaljima objave

Detalji objave foruma

- Stranica detalja objava sadrži diskusiju koja je popis odgovora i njihovih komentara
- Postoji mogućnost sortiranja odgovora po ocjeni i vremenu stvaranja
- Objava će imati poveznicu na korisnika koji ju je stvorio
- Korisnici mogu označiti objavu ili odgovor kao *favorite* isto kao što to mogu i za tečaj

Profil korisnika

- Informacije o korisniku poput profilne slike, imena, email adrese i slično
- Verifikacijski status
- Statistika za predavače (broj tečajeva, prosječna ocjena...)

Korisnik na svome profilu vidi:

- Tečaje/objave koje je kreirao
- Tečaje/objave koje je označio kao favorite
- Tečaje za koje je platio
- Tečaj/objave na koje je komentirao/odgovorio
- Radionice u kojima je sudjelovao

Opseg projektnog zadatka

Projekt obuhvaća razvoj platforme koja omogućuje stvaranje edukativnog online sadržaja. Tim se dijeli na dvije osnovne grupe frontend i backend.

Frontend

Frontend je zadužen za prikaz informacije poslane od backenda. Izgled stranice važna je komponenta za početni dojam novijih korisnika, stranica treba biti responzivna i laka za korištenje. Sve moguće oznake trebaju biti jasno vidljive i funkcionalne. Frontend će koristiti modernije alate za izradu web stranice kao što su React (https://react.dev/) i TypeScript (https://www.typescriptlang.org/).

Backend

Backend je zadužen za procesiranje zahtjeva od frontenda. To može biti dohvat ili spremanje podataka, kao i moguća izmjena podataka. Ponašat će se kao posrednik između frontenda i baze podataka. Backend treba biti efikasan što se tiče procesiranja i slanja podataka, također treba osigurati sigurnost i oporavak od pogreški za cjelokupni sustav. Backend će raditi s ASP.NET Core (https://dotnet.microsoft.com/apps/aspnet).

Baza podataka

Baza podataka mora biti zaštićena od zlonamjernih prijetnji. Integritet i dostupnost podataka od visoke su važnosti. Bazom podataka mora se pažljivo rukovati tako da se ne naruši integritet podataka. Za ovaj projekt koristit će se PostgreSQL (https://www.postgresql.org/).

Komunikacija

Jedan od ključnih dijelova ovog projekta je sama komunikacija tima. Očekuje se da će svaki član tima pridonijeti ovom projektu, kao što će se i svaki član oslanjati na ostatak tima. Kroz ovaj projekt vodit će se kontinuirani sastanci kako bi se tim regrupirao i kako bi se organizirali sljedeći koraci unutar projekta. Za dokumentaciju, kod i upravljanje koristit će se Github (https://github.com/). Za komunikaciju koristit će se Discord (https://discord.com/) i WhatsApp (https://www.whatsapp.com/).

Hosting

Kako bi pokazali da je platforma spremna za pogon, potrebno je pokrenuti aplikaciju na namjenskom serveru. Za to će se koristiti hosting stranica za servere, u ovom slučaju Contabo (https://contabo.com/en/). Server je jednostavan Ubuntu operacijski sustav kojem se pristupa preko SSH protokola.

Vanjske tehnologije

Kako bi se ubrzala izgradnja projekta, koristit će se već gotovi API-i koji bi olakšali izgradnju web aplikacije. To bi bili:

- Stripe (https://stripe.com/en-hr) - online plaćanje
- Google Cloud (https://console.cloud.google.com/) - autentifikacija
- Twilio SendGrid (https://sendgrid.com/en-us) - slanje mailova

Moguća nadogradnja projektnog zadatka

- Dodati chat za korisnike, tako da dva ili više korisnika mogu komunicirati u privatnom forumu.
- Posebni *rankovi* za korisnike koji prikazuju njihovo iskustvo i aktivnost.
- Mogućnost prikazivanja reklama za povećanje prihoda.
- Dodavanje *tagova* za stavke kako bi njihovo pretraživanje bilo preciznije.
- Moderacija slika, npr. slike bi trebale biti provjerene prije nego što korisnik može objaviti objavu.
- Kreiranje mobilne aplikacije, koja bi imala mogućnost obavještavanja korisnika.
- Napredno pretraživanje poput googlea.
- Sustav za bedževe.
- Inbox za obavijesti.
- Dodati docker datotke.

Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriterij prihvaćanja
F-0.0	Sustav omogućuje korisnicima kreiranje računa pomoću e-mail adrese.	Visok	Korisnik	Korisnik se može registrirati e-mailom i kreirati korisnički račun.
F-0.1	Sustav omogućuje korisnicima kreiranje računa pomoću google autentikacije.	Visok	Korisnik	Korisnik se može registrirati i prijaviti na svoj korisnički račun preko google autentikacije.
F-0.2	Sustav omogućuje oporavak lozinke putem e-maila.	Srednji	Korisnik	Korisnik može zatražiti resetiranje lozinke, primiti poveznicu za resetiranje i uspješno resetirati lozinku.

ID zahtjeva	Opis	Prioritet	Izvor	Kriterij prihvaćanja
F-0.3	Sustav omogućuje prijavu korisnika putem e-maila / korisničkog imena.	Visok	Korisnik	Korisnik se prijavljuje u sustav koristeći svoj e-mail / korisničko ime i lozinku.
F-0.4	Sustav omogućuje pretraživanje tečaja / radionica / objava / profila.	Srednje	Korisnik	Korisnik upisuje u tražilicu važnu riječ što rezultira prikazom liste sličnih naslova stavki.
F-0.5	Prilikom pretraživanja sustav omogućuje filtraciju liste prema: broju lajkova, vremenu.	Srednji	Korisnik	Ispod tražilice korisnik može kliknuti na odgovarajući filter koji rezultira prikazom filtrirane liste.
F-0.6	Sustav omogućuje pregled specifičnog tečaja / radionice / objave / profila.	Visok	Korisnik	Klikom na specifičnu stavku korisniku se otvara prozor te stavke.
F-0.7	Sustav omogućuje prikaz cijelokupnog besplatnog tečaja.	Visok	Korisnik	Klikom na besplatni tečaj prikazan je cijeli tečaj zajedno sa njegovim lekcijama.
F-0.8	Sustav omogućuje plaćanje privatnog tečaja / radionice.	Visok	Korisnik	Korisnik može pristupiti privatnom tečaju / radionici nakon što je platio.
F-0.9	Sustav omogućuje kreiranje tečaja / radionice / objave.	Visok	Korisnik	Preko svog profila korisnik može napraviti svoj tečaj / radionicu / objavu na forumu.
F-0.10	Prilikom kreiranja tečaja korisnik mora specificirati naslov, opis, barem jednu lekciju i cijenu ukoliko je privatan.	Visok	Korisnik	Nije moguće stvoriti tečaj bez naslova, opisa ili lekcije.
F-0.11	Prilikom kreiranja radionice korisnik mora specificirati naslov, opis i cijenu.	Visok	Korisnik	Nije moguće stvoriti radionice bez naslova, opisa i cijene.
F-0.12	Prilikom kreiranja objave na forumu korisnik mora specificirati naslov i tijelo objave	Visok	Korisnik	Nije moguće stvoriti objavu bet naslova i tijela objave.
F-0.13	Sustav omogućuje modificiranje postojećeg tečaja / radionice / objave.	Visok	Korisnik	Korisnik može modificirati svoje tečajeve.
F-0.14	Sustav omogućuje brisanje postojećeg tečaja / radionice / objave / vlastiti profil.	Visok	Korisnik	Korisnik može izbrisati tečaj koji je on stvorio.
F-0.15	Nakon kreiranja objave na forumu vlasnik objave može označiti rješenje.	Nizak	Korisnik	Korisnik može označiti rješenje na svoje pitanje iz odgovora drugih korisnika.
F-0.16	Sustav omogućuje da korisnik komentira na tečaj / radionicu / objavu.	Srednji	Korisnik	Korisnik može dodati komentar na neki tečaj / radionicu / objavu.
F-0.17	Sustav omogućuje da korisnik ocijeni tečaj / radionicu.	Srednji	Korisnik	Korisnik može ocijeniti tečaj / radionicu
F-0.18	Sustav omogućuje da korisnik lajka objavu na forumu.	Nizak	Korisnik	Korisnik može označiti objavu sa like-om
F-0.19	Sustav omogućuje korisniku da označi tečaj / radionicu kao favorit.	Nizak	Korisnik	Korisnik može označiti tečaj / radionicu kao favorit, te ih kasnije naći u listi favorita.
F-0.20	Sustav omogućuje korisniku promjenu imena.	Nizak	Korisnik	Korisnik može promijeniti ime.
F-0.21	Sustav omogućuje korisniku promjenu profilne slike.	Srednji	Korisnik	Korisnik može promijeniti profilnu sliku
F-0.22	Sustav omogućuje korisniku prijavu tuđeg profila na neprimjereno ponašanje.	Nizak	Korisnik	Korisnik ima opciju prijave drugog korisnika za neprimjereno ponašanje, te se ta prijava šalje moderatoru.
F-0.23	Sustav omogućuje korisniku prijavu za verificiranje svoga profila.	Srednji	Korisnik	Korisnik može poslati zahtjev za verifikacijom profila moderatoru.
F-0.24	Sustav omogućuje korisniku upload slike.	Srednji	Korisnik	Korisnik može upload-ati slike.
F-1.0	Sustav omogućuje moderatoru brisanje tečaja / radionice / objave	Srednji	Moderator	Moderator može obrisati tečak / radionicu / objavu.
F-1.1	Sustav omogućuje moderatoru verificiranje profila.	Srednji	Moderator	Moderator može odobriti zahtjev za verifikacijom profila.
F-1.2	Sustav omogućuje moderatoru utišenje korisnika.	Nizak	Moderator	Moderator može utišati korisnika.
F-2.0	Sustav omogućuje administratoru brisanje profila	Srednji	Administrator	Administrator može obrisati profile.
F-2.1	Sustav omogućuje administratoru promoviranje korisnika u moderatora.	Nizak	Administrator	Administrator može promovirati korisnika u moderatora.
F-3.0	Sustav omogućuje slanje maila korisnicima.	Srednji	Sustav	Korisnici dobiju mail kada koriste funkcionalnosti za koje je on potreban.
F-3.1	Sustav podržava kartično plaćanje.	Visok	Sustav	Moguće je karticom platiti za neki tečaj / radionicu
F-3.2	Sustav obrađuje zahtjeve za bazu podataka.	Visok	Sustav	Sustav može obraditi zahtjev i komunicirati s bazom podataka.

Nefunkcionalni zahtjevi

ID zahtjeva	Opis
NF-0.1	Stranica mora omogućiti vrijeme učitavanja manje od 10 sekundi
NF-1.1	Osjetljivi korisnički podaci moraju biti enkriptirani
NF-1.2	Sav promet između klijenta i servera mora koristiti HTTPS protokol
NF-1.3	Kolačići za autentifikaciju moraju koristiti opciju HttpOnly
NF-1.4	Sustav mora omogućiti autentifikaciju s pomoću korisničkog imena / lozinke ili Google OAuth2.0
NF-2.1	Sustav mora biti sposoban podržavati dodavanje novih značajki bez značajnih promjena postojeće infrastrukture
NF-3.1	Sustav mora biti dostupna 99 % vremena
NF-4.1	Stranica mora koristiti responzivan dizajn

Dionici

- Korisnici
- Moderatori i Administratori
- Server
- Razvojni tim

Aktor	Funkcionalnosti
Korisnik	F-0.0 do F-0.24
Moderator	F-0.0 do F-0.25, F-1.0 do F-1.2
Administrator	F-0.0 do F-0.25, F-1.0 do F-1.2 , F-2.0, F-2.1
Server	F-3.0 do F-3.2

Obrasci upotrebe

Dijagram obrazaca upotrebe

Korištenje sustava

Administracija

Prijava

Opis obrasca upotrebe

Element	Opis
Redni broj	U1
Glavni sudionik:	Korisnik
Cilj:	Pretraživanje tečaja, radionica i foruma s pomoću tražilice i različitih filtera
Sudionici:	Korisnik, Server, Baza podataka
Preduvjet:	-

Element	Opis
Opis osnovnog tijeka:	1. Korisnik upisuje u tražilicu ključne riječi. 2. Korisnik pokreće traženje pritiskom na gumb za pretraživanje ili tipkom Enter. (F-0.4) 3. Server prima zahtjev, kreira upit za bazu podataka i šalje odgovor s dohvaćenim podacima. 4. Korisniku se na stranici prikazuje odgovor od servera kao rezultati pretraživanja. 5. Korisnik uključuje specifični filter koji ponovno pokreće pretraživanje od koraka 3. (F-0.5)
Opis mogućih odstupanja:	-

Element	Opis
Redni broj	U2 (F-0.0, F-0.1, F-0.3, F-0.4)
Glavni sudionik:	Korinsik
Cilj:	Prijava korisnika u sustav
Opis osnovnog tijeka:	1. Korisnik odabire način prijave/registracije
Opis mogućih odstupanja:	-

Element	Opis
Redni broj	U12
Glavni sudionik:	Korisnik
Cilj:	Prijava korisnika putem emaila i lozinke
Sudionici:	Korisnik, Server, Mail
Preduvjet:	Korisnik ima email adresu
Opis osnovnog tijeka:	1. Korisnik odabire opciju za prijavu. 2. U slučaju da korisnik nema račun, korisnik odabire opciju za registraciju. (F-0.0) 2.a Korisnik prima potvrdu za mail preko upisane email adrese. 2.b Klikom na link, korisnik verificira svoj korisnički račun. 3. Korisnik upisuje svoje podatke korisničko: ime/email i lozinku. 4. Korisnik odabire opciju za prijavu. (F-0.3)
Opis mogućih odstupanja:	4.a Neispravna lozinka ili korisničko ime. Povećava se brojač mogućih prijava. U slučaju da je postignut maksimalan broj prijava pokreće se time-out prijave.

Element	Opis
Redni broj	U13
Glavni sudionik:	Korisnik
Cilj:	Prijava korisnika putem google računa.
Sudionici:	Korisnik, Server, Google server
Preduvjet:	Korisnik ima google račun.
Opis osnovnog tijeka:	1. Korisnik klikne na gumb za prijavu preko google računa. 2. Korisniku se otvara dialog za biranje google računa. 3. Korisnik se prijavljuje u google račun ako već nije. 4. Google vraća token korisniku koji se zatim prosljeđuje na server. 5. Server provjerava token. 6. Korisnik se uspješno prijavljuje / registrira. (F-0.1).
Opis mogućih odstupanja:	-

Element	Opis
Redni broj	U14
Glavni sudionik:	Korisnik
Cilj:	Oporavak lozinke
Sudionici:	Korinsik, Server, Mail
Preduvjet:	Korisnik treba imati korisnički račun.
Opis osnovnog tijeka:	1. Korisnik pokreće "Zaboravio/la sam lozinku". (F-0.2) 2. Server generira link za oporavak lozinke koji se šalje na email adresu korisnika. 3. Korisnik pokreće i ispunjava obrazac za zaboravljenu lozinku: nova lozinka, potvrda lozinke. 4. Server ažurira podatke.
Opis mogućih odstupanja:	4.a Nova lozinka i potvrda lozinke se ne podudaraju.

Element	Opis
Redni broj	U3
Glavni sudionik:	Korisnik
Cilj:	Kreiranje tečaja, radionice ili objave na forumu.
Sudionici:	Korisnik, Server
Preduvjet:	Samo verificirani korisnici mogu kreirati radionice. Korisnik mora biti prijavljen u sustav.
Opis osnovnog tijeka:	1.a Korisnik ispunjava obrazac za kreiranje tečaja: naslov, opis, skup lekcija. (F-0.9, F-0.10) 1.b Korisnik ispunjava obrazac za kreiranje radionice: naslov, opis. (F-0.9, F-0.11) 1.c Korisnik ispunjava obrazac za kreiranje objave: naslov, tijelo. (F-0.9, F-0.12) 2. Podaci se šalju na server gdje se pohranjuju u bazu podataka.
Opis mogućih odstupanja:	1.a Sva potreba polja u upitu nisu ispunjena.

Element	Opis
Redni broj	U4
Glavni sudionik:	Korisnik

Element	Opis
Cilj:	Modificiranje postojećeg tečaja, radionice ili objave na forumu.
Sudionici:	Korisnik, Server
Preduvjet:	Korisnik mora biti ulogiran i mora biti vlasnik stavke.
Opis osnovnog tijeka:	1. Korisnik pritišće gumb za uređivanje. (F-0.13, F-0.14) 2. Otvara se obrazac za uređivanje stavke. 3. Nakon promijene korisnik šalje upit na server. 4. Server ažurira bazu podataka.
Opis mogućih odstupanja:	3.a Nisu ispunjena sva potrebna polja.

Element	Opis
Redni broj	U5
Glavni sudionik:	Korisnik
Cilj:	Korisnik pristupa tečaju, radionici ili objavi.
Sudionici:	Korisnik, Server
Preduvjet:	Korisnik treba biti ulogiran.
Opis osnovnog tijeka:	1. Korisnik odabire stavku. 2. Korisniku se otvara odabrana stavka. (F-0.6, F-0.7) 3. U slučaju tečaja ili radionice moguća je opcija za upis. 4. Pritiskom na opciju za upis pokreće se upisivanje korisnika na tečaj. 5. Server ažurira bazu podataka.
Opis mogućih odstupanja:	4.a Za privatni tečaj ili radionicu pokreće se obrazac za plaćanje. (F-0.8)

Element	Opis
Redni broj	U6 (F-0.16, F-0.17, F-0.18)
Glavni sudionik:	Korisnik
Cilj:	Korisnik komentira tečaj, radionicu ili objavu.
Sudionici:	Korisnik, Server
Preduvjet:	Korisnik pohađa zadani tečaj ili radionicu ili je ulogiran kako bi slao objave.
Opis osnovnog tijeka:	1. Korisnik klikom na gumb za komentiranje pokreće obrazac za komentiranje. 2. Nakon ispunjenog obrasca potvrđuje to klikom na gumb za komentiranje. 3. Server prihvaća obrazac i sprema ga na svoju bazu podataka.
Opis mogućih odstupanja:	-

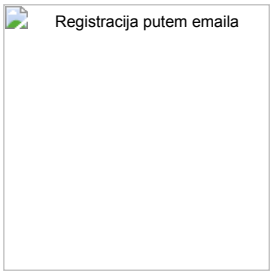
Element	Opis
Redni broj	U8
Glavni sudionik:	Korinik
Cilj:	Prijava korisnika na neprimjereno ponašanje.
Sudionici:	Korisnik, Server, Moderator
Preduvjet:	Korisnik je prijavljen u sustav.
Opis osnovnog tijeka:	1. Korisnik prijavljuje korisnički profil preko gumba za prijavu profila. (F-0.22) 2. Server pohranjuje prijavu u bazu. 3. Moderator dohvaća listu prijava. 4. Uloga moderatora je provjera prijavljenog korisničkog računa. 5. U slučaju potvrđenih sumnji određuje pravilnu kaznu. (F-1.0, F-1.2)
Opis mogućih odstupanja:	

Element	Opis
Redni broj	U9
Glavni sudionik:	Moderaotr
Cilj:	Verificirnje korisničkog računa.
Sudionici:	Korisnik, Moderator
Preduvjet:	Korisnik ima moderatorske privilegije.
Opis osnovnog tijeka:	1. Korisnik šalje zahtjev za verifikaciju računa. (F-0.23) 2. Server sprema zahtjev u bazu podataka. 3. Moderator dohvaća listu zahtjeva. 4. Na temelju provjere moderator verificira korisnika. (F-1.1)
Opis mogućih odstupanja:	4.a Zahtjev je odbijen.

Element	Opis
Redni broj	U10 i U11
Glavni sudionik:	Administrator
Cilj:	Modificiranje profila.
Sudionici:	Administrator, Korisnik
Preduvjet:	Korisnik ima administratorske privlegije.
Opis osnovnog tijeka:	1. Administrator dohvaća listu za zadani obrazac. 2. Na temelju provjere odlučuje postupak koji će poduzeti. (F-2.0, F-2.1)
Opis mogućih odstupanja:	-

Sekvencijski dijagram

Registracija putem emaila



Korisnik ispunjava obrazac za registraciju, koji se zatim šalje na server. Server kreira račun i sprema ga u bazu podataka. Preko mail API-a šalje se google mail na priloženu email adresu. Od korisnika se očekuje da klikne na poslanu poveznicu preko emaila. Klikom na poveznicu generira se token koji se zatim šalje serveru kako bi se verificirao korisnički račun, nakon čega je korisnički račun uspješno kreiran i funkcionalan.

Prijava putem emaila

Korisnik ispunjava i šalje obrazac za prijavu. Server prima obrazac, dohvaća podatke na temelju obrasca i provjerava ispravnost podataka. Ako su podaci točni stvara se kolačić koji se zatim šalje korisniku. U slučaju krive lozinke, šalje se obavijest i smanjuje se broj mogućih pokušaja za 1. Ako se postigne maksimalni broj neuspješnih prijava, korisnik se blokira na određeno vrijeme.

Prijava i registracija putem googla

Prilikom prijave sustava, korisnik pokreće prijava putem Googla. U slučaju da korisnik nije ulogiran u svoj Google račun otvara se prozor za prijavu na svoj Google račun. Nakon što se korisnik prijavio, na Google se šalje zahtjev za tokenom. Token se vraća korisniku koji se prosljeđuje serveru. Server provjerava validnost tokena, a u slučaju da korisnik nema svoj račun kreira se novi u bazi podataka. Zatim se korisniku šalje kolačić čime je korisnik verificiran.

Plaćanje

Korisnik pokreće obrazac za plaćanje. Server preko API-a za plaćanje kreira namjeru za plaćanje, a API vraća korisničku tajnu koja se koristi za potvrdu plaćanja. Ta tajna se šalje korisniku gdje on potvrđuje transakciju. Nakon potvrđene transakcije server ažurira svoju bazu podataka.

Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe

Obrazac	Funkcionalni zahtjevi
U1	F-0.4, F-0.5, F-3.2
U2	F-0.0 do F-0.3, F-0.20, F-0.21, F-0.23, F-0.24, F-3.0, F-3.2
U3	F-0.9 do F-0.12, F-3.2
U4	F-0.13, F-0.14, F-0.15 , F-3.2
U5	F-0.6, F-0.7, F-0.8, F-0.19, F-3.0, F-3.1, F-3.2
U6	F-0.16 do F-0.18, F-3.2
U8	F-0.22, F-1.0, F-1.2, F-3.2
U9	F-1.1, F-3.0, F-3.2
U10	F-2.0, F-3.0, F-3.2
U11	F-2.1, F-3.0, F-3.2

Arhitektura sustava

Opis arhitekture

Sustav koristi klijent-poslužitelj arhitekturu. To je model organizacije koji aplikaciju razdvaja na dvije ključne komponente: klijent i poslužitelj. U ovakvom modelu klijent šalje zahtjev za resurse, a poslužitelj obrađuje te zahtjeve i pruža tražene usluge. Ovakav način arhitekture postao je standard za web aplikacije, a model je danas široko prihvaćen i dobro poznat u indsutriji.

Sustav se sastoji od tri važna podsustava: frontend, backend i baza podataka. Frontend je zadužen za prikaz korisničkog sučelja te omogućuje korisniku interakciju sa sustavom. Backend je zadužen za obradu različitih zahtjeva korisnika, kao i provjeru tih podataka i upravljanje korisnicima. U ovom slučaju također služi kao i posrednik između frontenda i baze podataka. Baza podataka je zadužena za pohranu i upravljanje podacima o svim relevantnim informacijama koje sustav koristi.

Sustav će biti implementirana na unajmljenom poslužitelju. Bit će potrebno direktno instalirati potrebne servise i aplikaciju. Ovakav način gubi automatizaciju i portabilnost, ali izbacuje dodatnu kompleksnost virtualizacije.

Klijent i poslužitelj su dva odvojena procesa, a komunikacija između njih ostvaruje se preko HTTP ili HTTPS protokola. Standardna interakcija započinje tako da frontend šalje HTTP / HTTPS zahtjev backendu. Backend prima zahtjev, procesira ga i po potrebi pristupa bazi podataka. Baza podataka služi kao pohrana podataka koja odgovora na zahtjeve backenda. Baza podataka vraća tražene podatke backendu, koji se zatim šalju na frontend. Frontend prikazuje dobivene podatke. Za slanje maila koristit će se SMTP protokol.


Klijent-poslužitelj

Obrazloženje odabira arhitekture

Dizajn arhitekture uveliko je ovisio o današnjim standardima i samom iskustvu tima. Svaka komponenta u arhitekturi ima jasno definirane uloge. Ovakva podjela olakšava razvoj i održavanje, kao i brzu identifikaciju problema unutar specifičnih dijelova sustava. Kako su klijent i poslužitelj dosta neovisni jedan o drugome, to omogućuje visoku fleksibilnost za dodavanje novih funkcionalnosti na specifične dijelove sustava.

Organizacija aplikacije

Frontend

Frontend je podijeljen u sljedeće foldere:

- **api** - Apstrakcija komunikacije sa backendom.
- **component** - Sadrži react komponente.
- **context** - Sadrži react context datoteke.
- **enums** - Sadrži enum datoteke.
- **hooks** - Sadrži react hooks.

- **images** - Slike koje se koriste na frontendu (logo aplikacije i slično).
- **models** - Interface-ovi koji opisuju tipove podataka.
- **pages** - Page level komponente.
- **stores** - Mobx store komponente.
- **styles** - Globalne css datoteke.

Backend

Backend koristi Controller-Service-Repository arhitekturu

TeachABit.API (Controller layer)

- Configurations - dodavanje servisa i omogućavanje dependency injectiona za potrebne klase
- Controllers - sve Controller klase kojima je uloga samo poziv Service layera i kreiranje HTTP odgovora
- Middleware - middleware datoteke

TeachABit.Service (Service layer)

- Services - sve Service klase koje sadrže svu poslovnu logiku i komuniciraju s bazom preko Repository layera
- Util - utility klase za komunikaciju sa vanjskim servisima ili slično

TeachABit.Repository (Repository layer)

- Repositories - sve Repository klase koje komuniciraju s bazom podataka

TeachABit.Model

- DTOs - klase koje definiraju tipove podataka koji se šalju na frontend
- Mapping - AutoMapper profili
- Migrations - migracije za postavljanje baze podataka
- Models - klase koje predstavljaju tablice u bazi podataka
- ValidationAttributes - definicije custom validacija nad DTO poljima

Baza podataka

Za bazu podataka koristimo PostgreSQL s code first pristupom koristeći ef core migracije.

Opis tablica

Radionica

Atribut	Tip podatka	Opis varijable
Id	int4	Jedinstveni identifikator radionice.
Naziv	text	Naziv radionice.

Tecaj

Atribut	Tip podatka	Opis varijable
Id	int4	Jedinstveni identifikator tečaja.
Naziv	text	Naziv tečaja.

Objava

Atribut	Tip podatka	Opis varijable
Id	int4	Jedinstveni identifikator objave.
Naziv	text	Naziv objave.
Sadržaj	text	Tekst komentara.
VlasnikId	text	Jedinstveni identifikatora korisnika koji je objavio objavu.

Komentar

Atribut	Tip podatka	Opis varijable
Id	int4	Jedinstveni identifikator komentara objave.
Sadržaj	text	Sadržaj komentara.
VlasnikId	text	Jedinstveni identifikator korisnika koje je objavio komentar.
ObjavaId	int4	Jedinstveni identifikator objave na koju se odnosi komentar.
CreatedDateTime	timestamptz	Vrijem i datum kreiranja komentara.

AspNetUserRoles

Atribut	Tip podatka	Opis varijable
UserId	text	Jedinstveni identifikator korisnika.
RoleId	text	Jedinstveni identifikator uloge.

AspNetRoles

Atribut	Tip podatka	Opis varijable
Id	text	Jedinstveni identifikator uloge.
Name	varchar(256)	Ime uloge.
NormalizedName	varchar(256)	Normalizirano ime.
ConcurrencyStamp	text	Token istodobnosti.

AspNetRoleClaims

Atribut	Tip podatka	Opis varijable
Id	int4	Jedinstveni identifikator korisničke tvrdnje za ulogu.
RoleId	text	Jedinstveni identifikator uloge.
ClaimType	text	Tip tvrdnje.
ClaimValue	text	Vrijednost tvrdnje.

__EFMigrationHistory

Atribut	Tip podatka	Opis varijable
MigrationId	varchar(150)	Jedinstveni identifikator migracije.
ProductVersion	varchar(32)	-

AspNetUserClaims

Atribut	Tip podatka	Opis varijable
Id	int4	Jedinstveni identifikator korisničke tvrdnje.
UserId	text	Jedinstveni identifikator korisnika.
ClaimType	text	Tip tvrdnje.
ClaimValue	text	Vrijednost tvrdnje.

AspNetUserLogins

Atribut	Tip podatka	Opis varijable
LoginProvider	text	Način prijave u sustav.
ProviderKey	text	Token za prijavu.
ProviderDisplayName	text	Ime načina prijave.
UserId	text	Jedinstveni identifikator korisnika.

AspNetUserTokens

Atribut	Tip podatka	Opis varijable
UserId	text	Jedinstveni identifikator korisnika.
LoginProvider	text	Način prijave u sustav.
Name	text	Naziv tokena.
Value	text	Vrijednost tokena.

AspNetUsers

Atribut	Tip podatka	Opis varijable
Id	text	Jedinstveni identifikator korisnika.
UserName	varchar(256)	Ime korisnika.
NormalizedUserName	varchar(256)	Normalizirano korisničko ime.
Email	varchar(256)	Email korisnika.
NormalizedEmail	varchar(256)	Normaliziran email korisnika.
EmailConfirmed	bool	Je li potvrđen email račun.
PasswordHash	text	Sažetak lozinke.
SecurityStamp	text	Sol za šifru.
ConcurrencyStamp	text	Token istodobnosti.
PhoneNumber	text	Mobilni broj korisnika.
PhoneNumberConfirmed	bool	Je li potvrđen mobilni broj korisnika.
TwoFactorEnabled	bool	Je li omogućena dvo faktorska autentifikacija.
LockoutEnd	timestampz	Trajanje zabrane prijave.

Atribut	Tip podatka	Opis varijable
LockoutEnabled	bool	Je li blokiranje omogućeno.
AccessFailedCount	int4	Broj neuspješnih prijava.
ProfilnaSlikaVersion	text	Verzija profilne slike.

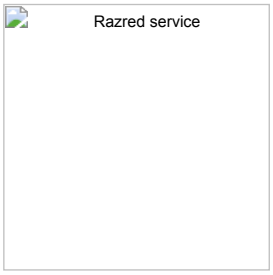
Dijagram baze podataka

Dijagram razreda

Controller

- Controller klase koje nasljeđuju BaseController
- ModelStateFilter koji provjerava ispravnost dolaznih podataka svih endpointa koji koriste isti
- DependencyRegistrations i ServiceExtensions za konfiguraciju sustava

Service



- Service klase sa pripadnim interface-ima

Model

- Result pattern klase za standardizaciju return tipova
- MessageResult klase za opis poruka
- Dto klase
- Model klase

- Extensions klase s extension metodama
- Context klasa koja opisuje bazu podataka
- Atributi koji se koriste pri validaciji Model state-a
- AutoMapper klase koje definiraju mapiranja među klasama

Repository

- Repository klase sa pripadnim interface-ima

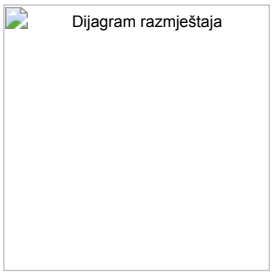
Dinamičko ponašanje aplikacije

UML dijagrami stanja

UML dijagrami aktivnosti

Dijagram komponenata

Dijagram razmještaja



Ispitivanje komponenti

Ispitivanje komponenti

1. Prijava korisnika

Ulaz:

- korisničko ime = "demo"
- lozinka = "Password0"

Koraci:

1. Pokreće se web preglednik na localhost:3000
2. Klikni na gumb "Prijava"
3. Unesi korisničko ime i lozinku
4. Klikni na gumb "Prijava"

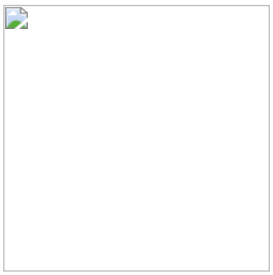
Izlaz:

Korisnik je prijavljen na stranicu.

```
import { test } from "@playwright/test";
import performLogin from "./helpers/login";

test("Login", async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();

    await performLogin(page);
});
```



2. Neispravna lozinka

Ulaz:

- korisničko ime = "demo"
- lozinka = "Password0"

Koraci:

1. Pokreće se web preglednik na localhost:3000
2. Klikne se gumb prijava
3. Unesi korisničko ime i lozinku
4. Klikni na gumb "Prijava"

Izlaz:

Korisnik nije prijavljen, pojavljena je poruka "Pogrešna lozinka".

```
import { test } from "@playwright/test";
import performLoginGreska from "./helpers/login_greska";

test("LoginGreska", async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();

    await performLoginGreska(page);
});
```

3. Kreiranje tečaja

Ulaz:

- naziv = "Tečaj1"
- opis = "Opis1"
- cijena = 100
- korisničko ime = "demo"
- lozinka="Password0"

Koraci:

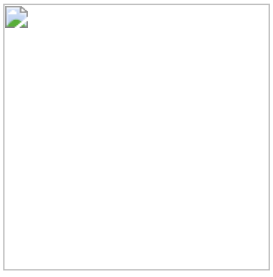
1. Pokreće se web preglednik na localhost:3000
2. Klikni na gumb "Prijava"
3. Unesi korisničko ime i lozinku
4. Klikni na gumb "Prijava"
5. Klikne se na gumb "Stvori tecaj"
6. Unesu se naziv, opis i cijena
7. Klikne se gumb "Stvori tečaj"

Izlaz: Tečaj je prikazan na listi tečaja.

```
import { test } from "@playwright/test";
import performLogin from "./helpers/login";
import createTecaj from "./helpers/createTecaj";

test("CreateTecaj", async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();

    await performLogin(page);
    await createTecaj(page);
});
```



4. Kreiranje komentara (na tečaju)

Ulaz:

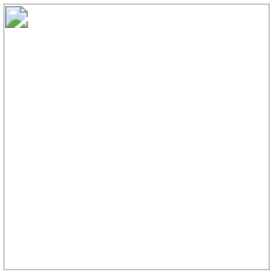
- Naziv = "Tečaj1"
- Opis = "Opis1"
- Cijena = 100
- Korisničko ime = "demo"
- lozinka = "Password0"
- komentar = "tekst"

Koraci:

1. Pokreće se web preglednik na localhost:3000
2. Klikni na gumb "Prijava"
3. Unesi korisničko ime i lozinku
4. Klikni na gumb "Prijava"
5. Klikne se na gumb "Stvori tecaj"
6. Unesu se naziv, opis i cijena
7. Klikne se gumb "Stvori tečaj"
8. Klikne se gumb "Stvori komentar"
9. Unesi komentar
10. Klikne se gumb stvori komentar

Izlaz: Komenatr je stvoren na stranici tečaja.

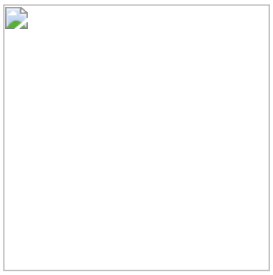
```
import { test } from "@playwright/test";
import performLogin from "./helpers/login";
import createKomentarTecaj from "./helpers/createKomentarTecaj";
import createTecaj from "./helpers/createTecaj";
test("CreateKomentarTecaj",  async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();
    await performLogin(page);
    //await createTecaj(page);
    await createKomentarTecaj(page);
});
```



5. Kreiranje lekcije na tečaju:

Ulaz:

- Naziv = "Tečaj1"
- Opis = "Opis1"
- Cijena = 100
- Korisničko ime = "demo"
- Lozinka = "Password0"
- Naziv lekcije = "Lekcija_1"
- Sadržaj lekcije = "Objasnjenje"



Koraci:

1. Pokreće se web preglednik na localhost:3000
2. Klikni na gumb "Prijava"
3. Unesi korisničko ime i lozinku
4. Klikni na gumb "Prijava"
5. Klikni na gumb "Stvori tecaj"
6. Unesu se naziv, opis i cijena
7. Klikni na gumb "Stvori tečaj"
8. Klikni na gumb "Stvori lekciju"
9. Unesi naziv lekcije i sadržaj lekcije
10. Klikni na gumb "Stvori lekciju"

Izlaz: Lekcija je stvorena na stranici tečaja.

```
import { test } from "@playwright/test";
import performLogin from "./helpers/login";
import createLekcija from "./helpers/createLekcija";

test("CreateLekcija", async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();

    await performLogin(page);
    await createLekcija(page);
});
```

IZLAZ: Korisnik je prijavljen na stranicu

6. Kreiranje objave

Ulaz:

- Korisničko ime = "demo"
- Lozinka = "Password0"
- Naziv objave = randomUUID()
- Sadržaj objave = "Testing sadrzaj"

Koraci:

1. Pokreće se web preglednik na localhost:3000
2. Klikni na gumb "Prijava"
3. Unesi korisničko ime i lozinku
4. Klikni na gumb "Prijava"
5. Klikne se na gumb "Forum"
6. Klikne se na gumb "Stvori objavu"
7. Unesi naziv objave i sadržaj objave
8. Klikni na gumb "Stvori komentar"

Izlaz: Objava je stvorena.

```
import { test, expect, chromium } from "@playwright/test";
import performLogin from "./helpers/login";
import createDeleteObjava from "./helpers/createDeleteObjava";

// Nije implementirano delete dio jer test nije nikako mogo prepoznat delete button

test("CreateAndDeleteObjava", async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();

    await performLogin(page);
    await createDeleteObjava(page);
});
```

7. Kreiranje radionice

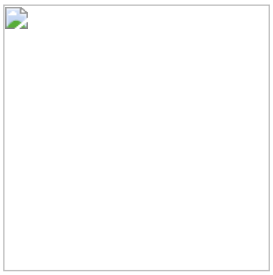
Ulaz:

- Korisničko ime = "demo"
- Lozinka = "Password0"
- Naziv = randomUUID()
- Opis = "Testing sadrzaj"
- Cijena = 5
- Kapacitet = 6
- Vrijeme = "02/24/2025 10:30"

Koraci:

1. Pokreće se web preglednik na localhost:3000
2. Klikni na gumb "Prijava"
3. Unesi korisničko ime i lozinku
4. Klikni na gumb "Prijava"
5. Klikni na gumb "Radionice"
6. klikni na gumb "Stvori radionicu"
7. Unesi naziv, opis, cijena i kapacitet
8. Klikni na gumb stvori radionicu

Izlaz: Radionica je stvorena na listi radionica.



8. Modificiranje tečaja

Ulaz:

- Naziv = "Tečaj1"
- Opis = "Opis1"
- Cijena = 100
- Korisničko ime = "demo"
- lozinka = "Password0"
- komentar = "tekst"
- Promjena opis = "PromjenaOpisa"

Koraci:

1. Pokreće se web preglednik na localhost:3000
2. Klikni na gumb "Prijava"
3. Unesi korisničko ime i lozinku
4. Klikni na gumb "Prijava"
5. Klikne se na gumb "Stvori tecaj"

6. Unesu se naziv, opis i cijena
7. Klikne se gumb "Stvori tečaj"
8. Klikne se gumb "Stvori komentar"
9. Unesi komentar
10. Klikne se gumb stvori komentar
11. Klikni na gumb za modificiranje
12. Unesi promjenu opisa
13. Klikni na gumb "Spremi promjenu"

Izlaz: Promjenjen je opis tečaja na stranici tečaja.

```
import { test } from "@playwright/test";
import performLogin from "./helpers/login";
import ModificiranjeTecaj from "./helpers/ModificiranjeTecaja";

test("ModificiranjeTecaj", async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();

    await performLogin(page);
    await ModificiranjeTecaj(page);
});
```

```
import { test, expect, chromium } from "@playwright/test";
import performLogin from "./helpers/login";
import createDeleteRadionica from "./helpers/createDeleteRadionica";

// brisanje ne radi jer test ne vidi delete opciju

test("CreateAndDeleteRadionice", async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();

    await performLogin(page);
    await createDeleteRadionica(page);
});
```

Helper funkcije

```
import { test, expect, chromium } from "@playwright/test";
import performLogin from "./helpers/login";
import createDeleteRadionica from "./helpers/createDeleteRadionica";

test("CreateAndDeleteRadionice", async ({ browser }) => {
    const context = await browser.newContext({ ignoreHTTPSErrors: true });
    const page = await context.newPage();

    await performLogin(page);
    await createDeleteRadionica(page);
});
```

createDeleteObjava.ts

```
import { expect, Page } from "@playwright/test";
import { randomUUID } from "crypto";

// Ovaj file ne radi delete zato jer je bilo problema pri postavljanju parametra za test
// No delete radi

export default async function createDeleteObjava(page: Page) {
    await page.goto("https://localhost:3000");

    const naziv = randomUUID();

    const korisnik = page.locator("#navigationUser-korisnik");
    try {
        await expect(korisnik).toBeVisible();
    } catch (ex) {
        throw new Error("Korisnik mora biti ulogiran.");
    }

    const forumButton = page.locator('span:has-text("Forum")');
    await forumButton.click();

    const createObjavaButton = page.locator('button:has-text("Stvori objavu")');
    await createObjavaButton.click();

    const createObjavaForm = page.locator("#objavaEditor");
    await expect(createObjavaForm).toBeVisible();

    const objavaNazivInput = page.locator('input[name="naziv"]');
    await objavaNazivInput.fill(naziv);

    const tiptapEditor = page.locator(".ProseMirror");
    await tiptapEditor.fill("Testing sadrzaj");

    const createObjavaFormButton = page.locator("#objavaEditorStvoriObjavu");
    await createObjavaFormButton.click();

}
```

createDeleteRadionica.ts

```
import { expect, Page } from "@playwright/test";
import { randomUUID } from "crypto";

export default async function createDeleteRadionica(page: Page) {
    await page.goto("https://localhost:3000");

    const naziv = randomUUID();

    const korisnik = page.locator("#navigationUser-korisnik");
    try {
        await expect(korisnik).toBeVisible();
    } catch (ex) {
        throw new Error("Korisnik mora biti ulogiran.");
    }

    const forumButton = page.locator('span:has-text("Radionice")');
    await forumButton.click();

    const createRadionicaButton = page.locator('button:has-text("Stvori radionicu")');
    await createRadionicaButton.click();

    const createRadionicaForm = page.locator("#radionicaEditor");
    await expect(createRadionicaForm).toBeVisible();

    const radionicaNazivInput = page.locator('input[name="naziv"]');
    await radionicaNazivInput.fill(naziv);

    const radionicaOpisInput = page.locator('textarea[name="opis"]');
    await radionicaOpisInput.fill("Testing sadrzaj");

    const radionicaCijenaInput = page.locator('input[name="cijena"]');
    await radionicaCijenaInput.fill("5");




    const vrijeme = page.locator('input[placeholder="MM/DD/YYYY hh:mm"]');
    await vrijeme.fill('02/24/2025 10:30')

    const broj = page.locator('input[name="cijena"]');
    await broj.fill("6");

    const createRadionicaFormButton = page.locator("#stvoriRadionicuButton");
    await createRadionicaFormButton.click();

    const createdRadionica = page.locator(`#radionica:has-text("${naziv}")`);
    await createdRadionica.scrollIntoViewIfNeeded();
    await expect(createdRadionica).toBeVisible();

    const radionicaNavigator = page.locator(
        `#radionica:has-text("${naziv}") >> button`
    );
    await radionicaNavigator.click();

    // const deleteObjavaButton = page.locator("#objavaPage-deleteButton");
    // await deleteObjavaButton.click();

    // const deletedObjava = page.locator(`#objava:has-text("${naziv}")`);
    // await expect(deletedObjava).toHaveCount(0);
}
```

createKomentarTecaj.ts

```
import { expect, Page } from "@playwright/test";
export default async function createKomentarTecaj(page: Page){
    await page.goto("https://localhost:3000");
    const korisnik = page.locator("#navigationUser-korisnik");
    try {
        await expect(korisnik).toBeVisible();
    } catch (ex) {
        throw new Error("Korisnik mora biti ulogiran.");
    }
    const Button1 = page.locator('div.MuiButtonBase-root.MuiListItemButton-root.MuiListItemButton-gutters._navButton_1swuc_9._active
    await Button1.click();

    const Button2 = page.locator('button:has-text("Stvori tecaj")')
    await Button2.click();

    const createCekajForm = page.locator('div.MuiDialog-container[role="presentation"]');
    await expect(createCekajForm).toBeVisible();

    const inputField = page.locator('input[name="naziv"]');
    await inputField.fill("Tecaj1");

    const inputField1 = page.locator('div[contenteditable="true"].ProseMirror')
    await inputField1.fill("Opis1");

    const inputField2 = page.locator('input[name="cijena"]');
    await inputField2.fill("100");

    const Button3 = page.locator('button.MuiButtonBase-root.MuiButton-root.MuiButton-contained.MuiButton-containedPrimary.MuiButton-
    await Button3.click()

    await page.waitForTimeout(10000)
    const addCommentButton = page.locator('button:has-text("Dodaj Komentar")');
    await addCommentButton.click();

    const editor = page.locator('div[contenteditable="true"]');
    await editor.fill("tekst");

    const button = page.locator('button:has-text("Stvori komentar")');
    await button.click();



    /*
    const addButton = page.locator('button.MuiButtonBase-root.MuiIconButton-root.MuiIconButton-sizeMedium.css-hefddw');
    await addButton.click();

    const label = page.locator('label:has-text("Naziv")');
    await expect(label).toBeVisible();

    const inputField3 = page.locator('input[name="naziv"]');
    await inputField3.fill('Lekcija_1');

    const editor = page.locator('div[contenteditable="true"]');
    await editor.fill("Objasnjenje");

    const button = page.locator('button:has-text("Stvori lekciju")');
    await button.click();
    */

}
```

createLekcija.ts

```
import { expect, Page } from "@playwright/test";
export default async function createKomentarTecaj(page: Page) {
    await page.goto("https://localhost:3000");
    const korisnik = page.locator("#navigationUser-korisnik");
    try {
        await expect(korisnik).toBeVisible();
    } catch (ex) {
        throw new Error("Korisnik mora biti ulogiran.");
    }
    const Button1 = page.locator('div.MuiButtonBase-root.MuiListItemButton-root.MuiListItemButton-gutters._navButton_1swuc_9._active
    await Button1.click();

    const Button2 = page.locator('button:has-text("Stvori tecaj")')
    await Button2.click();

    const createCekajForm = page.locator('div.MuiDialog-container[role="presentation"]');
    await expect(createCekajForm).toBeVisible();

    const inputField = page.locator('input[name="naziv"]');
    await inputField.fill("Tecaj1");

    const inputField1 = page.locator('div[contenteditable="true"].ProseMirror')
    await inputField1.fill("Opis1");

    const inputField2 = page.locator('input[name="cijena"]');
    await inputField2.fill("100");

    const Button3 = page.locator('button.MuiButtonBase-root.MuiButton-root.MuiButton-contained.MuiButton-containedPrimary.MuiButton-
    await Button3.click()

    await page.waitForTimeout(10000);

    const addButton = page.locator('button:has-text("Dodaj Lekciju")');
    await addButton.click();

    const label = page.locator('label:has-text("Naziv")');
    await expect(label).toBeVisible();

    const inputField3 = page.locator('input[name="naziv"]');
    await inputField3.fill('Lekcija_1');

    const editor = page.locator('div[contenteditable="true"]');
    await editor.fill("Objasnjenje");

    const button = page.locator('button:has-text("Stvori lekciju")');
    await button.click();

}
```

createTecaj.ts

```
import { expect, Page } from "@playwright/test";
export default async function createTecaj(page: Page) {
    await page.goto("https://localhost:3000");
    const korisnik = page.locator("#navigationUser-korisnik");
    try {
        await expect(korisnik).toBeVisible();
    } catch (ex) {
        throw new Error("Korisnik mora biti ulogiran.");
    }
    const Button1 = page.locator('div.MuiButtonBase-root.MuiListItemButton-root.MuiListItemButton-gutters._navButton_1swuc_9._active
    await Button1.click();

    const Button2 = page.locator('button:has-text("Stvori tecaj")')
    await Button2.click();

    const createCekajForm = page.locator('div.MuiDialog-container[role="presentation"]');
    await expect(createCekajForm).toBeVisible();

    const inputField = page.locator('input[name="naziv"]');
    await inputField.fill("Tecaj1");

    const inputField1 = page.locator('div[contenteditable="true"].ProseMirror')
    await inputField1.fill("Opis1");

    const inputField2 = page.locator('input[name="cijena"]');
    await inputField2.fill("100");

    const Button3 = page.locator('button.MuiButtonBase-root.MuiButton-root.MuiButton-contained.MuiButton-containedPrimary.MuiButton-
    await Button3.click()
}
```

login_greska.ts

```
import { expect, Page } from "@playwright/test";

export default async function performLoginGreska(page: Page) {
    await page.goto("https://localhost:3000");

    // Click the login button to open the form
    const loginButton = page.locator("#authForm-prijavaButton");
    await loginButton.click();
    // Wait for form to be visible
    const form = page.locator("#loginForm");
    await expect(form).toBeVisible();

    // Fill in the email/username field
    const emailInput = page.locator('input[name="credentials"]');
    await emailInput.fill("demo");

    // Fill in the password field
    const passwordInput = page.locator('input[name="password"]');
    await passwordInput.fill("Password");

    // Submit the form
    const submitButton = page.locator("#loginForm-prijavaButton");
    await submitButton.click();

    // Verify successful login by checking for navigationUser element
    /*const korisnik = page.locator("#navigationUser-korisnik");
    await expect(korisnik).toBeVisible();*/

    await page.waitForSelector('div.MuiAlert-colorError.MuiAlert-standardError', {
        state: 'visible' // Wait until the element is visible
    });
    console.error('Pogrešni podaci za prijavu');
}
```

login.ts

```
import { expect, Page } from "@playwright/test";

export default async function performLogin(page: Page) {
    await page.goto("https://localhost:3000");

    // Click the login button to open the form
    const loginButton = page.locator("#authForm-prijavaButton");
    await loginButton.click();

    // Wait for form to be visible
    const form = page.locator("#loginForm");
    await expect(form).toBeVisible();

    // Fill in the email/username field
    const emailInput = page.locator('input[name="credentials"]');
    await emailInput.fill("demo");

    // Fill in the password field
    const passwordInput = page.locator('input[name="password"]');
    await passwordInput.fill("Password0");

    // Submit the form
    const submitButton = page.locator("#loginForm-prijavaButton");
    await submitButton.click();

    // Verify successful login by checking for navigationUser element
    const korisnik = page.locator("#navigationUser-korisnik");
    await expect(korisnik).toBeVisible();
}
```

ModificiranjeTecaja.ts

```
import { expect, Page } from "@playwright/test";
export default async function ModificiranjeTecaj(page: Page) {
    await page.goto("https://localhost:3000");
    const korisnik = page.locator("#navigationUser-korisnik");
    try {
        await expect(korisnik).toBeVisible();
    } catch (ex) {
        throw new Error("Korisnik mora biti ulogiran.");
    }
    const Button1 = page.locator('div.MuiButtonBase-root.MuiListItemButton-root.MuiListItemButton-gutters._navButton_1swuc_9._active
    await Button1.click();

    const Button2 = page.locator('button:has-text("Stvori tecaj")')
    await Button2.click();

    const createCekajForm = page.locator('div.MuiDialog-container[role="presentation"]');
    await expect(createCekajForm).toBeVisible();

    const inputField = page.locator('input[name="naziv"]');
    await inputField.fill("Tecaj1");

    const inputField1 = page.locator('div[contenteditable="true"].ProseMirror')
    await inputField1.fill("Opis1");

    const inputField2 = page.locator('input[name="cijena"]');
    await inputField2.fill("100");

    const Button3 = page.locator('button.MuiButtonBase-root.MuiButton-root.MuiButton-contained.MuiButton-containedPrimary.MuiButton-
    await Button3.click()

    await page.waitForTimeout(10000)
    const addCommentButton = page.locator('button:has-text("Dodaj Komentar")');
    await addCommentButton.click();

    const editor = page.locator('div[contenteditable="true"]');
    await editor.fill("tekst");

    const button = page.locator('button:has-text("Stvori komentar")');
    await button.click();

    const olovka= page.locator('[data-testid="EditIcon"]');
    await olovka.click();

    const tekst =page.locator('div[contenteditable="true"].ProseMirror');
    await tekst.fill("PromjenaOpis");

    const a=page.locator('button:has-text("Spremi promjene")');
    await a.click();

}
```

Korištene tehnologije i alati

Programski jezici

Za izradu korisničkog dijela aplikacije koristi se **TypeScript** (verzija 5.5.3). TypeScript je programski jezik sa strogim tipovima, temeljen na JavaScriptu, koji omogućuje precizniju kontrolu nad strukturom koda. Zbog statičke provjere koda ubrzava razvoj i održavanje te smanjuje rizik od pogrešaka. Njegova jasnoća i modularnost posebno su korisni u timskim projektima.

Za izradu serverskog dijela aplikacije koristi se **C#** (verzija 12). C# je objektno orijentirani programski jezik koji se koristi za razvoj aplikacije unutar .NET okvira.

Radni okviri i biblioteke

Na frontendu se još koristi **React** (verzija 18.3.1), popularni alat za izradu interaktivnih korisničkih sučelja. React je radni okvir koji omogućuje stvaranje samostalnih komponenti, što doprinosi lakšem održavanju i ponovnoj upotrebi koda.

Za prikaz stranice koristi se **Material UI** (verzija 6.1.6). Ova biblioteka pruža gotove komponente, što omogućava brzo kreiranje modernih korisničkih sučelja. Pruža fleksibilnu prilagodbu tema tako da se može namjestiti prema specifičnim potrebama aplikacije.

Kako bi olakšali upit za izradu određenog tečaja, radionice ili objave na forumu koristi se **Tiptap** (verzija 2.9.1). Napredni editor koji se lako ugrađuje u React aplikaciju. Korisnicima omogućuje unos sadržaja unutar aplikacije uz podršku za razne prilagodljive stilove.

Za serversku stranu aplikacije koristi se **ASP.NET C#** (verzija 8.0.110). C# je objektno orijentirani programski jezik koji se koristi unutar .NET okvira. Koristi se zbog lakog povezivanja s bazom podataka i ostalih dijelova sustava.

Osim klasičnog korisničkog imena i lozinke za prijavu na sustav, aplikacija koristi i prijavu preko Google servisa. To je ostvareno s pomoću **react-oauth/google** (verzija 0.12.1), biblioteka za autentifikaciju korisnika putem Google OAuth protokola.

Za spremanje slika koristi se **ASWSDK.S3** (verzija 3.7.405.11). To je biblioteka za rad s Amazon S3 servisom koji se koristi za pohranu i dohvaćanje statičkih resursa (poput slika, videa, datoteka i sl.).

Plaćanje je realizirano pomoću **Stripe** (verzija 2024-12-18.acacia) sustava. Stripe API omogućuje jednostavno upravljanje plaćanjima, pretplatam i povratima.

Za slanje mailova koristi se **SendGrid** (verzija 9.29). SendGrid je email API i SMTP servis koji omogućuje slanje mailova.

Baza podataka

PostgreSQL (verzija 17) je relacijska baza podataka poznata po svojoj skalabilnosti, sigurnosti i podršci za složene upite.

Razvojni alati

Razvojna okruženja:

- Visual Studio
- Visual Studio Code
- JetBrains Rider

Za verzioniranje aplikacije koristio se **Git** (verzija 2.46).

Alati za ispitivanje

Alati za razmještaj

Nginx (verzija 1.18.0) je konfiguriran kao reverse proxy za preusmjeravanje zahtjeva frontendu. Ima jednostavnu konfiguraciju i postavljanje, te omogućuje razne dodatne mogućnosti koje poboljšavaju performansu.

Cloud platforma

Aplikacija je hostana na **Contabo** platformi, koja pruža virtualne privatne servere.

Instalacija

Potrebno je instalirati sljedeće komponente:

- Node.js
- Nginx
- Git
- Dotnet SDK
- TeachABit repozotorij
- PostgreSQL

Ubuntu:

```
sudo apt install node nginx git postgresql \
sudo apt-get install -y dotnet-sdk-8.0 \
git clone https://github.com/DorijanJ/TeachABit.git

dotnet tool install --global dotnet-ef --version 8.*
```

Postavljanje baze podataka

Kreirajte bazu podataka s imenom "teachabit" i shemom "backend". Napravite novog korisnika s imenom "teachabit_backend" i dodijelite mu lozinku. Dodijelite sva prava kreiranom korisniku za bazu podataka "teachabit".

Ulazak u bazu podataka:

```
sudo -i -u postgres \
psql
```

Postavljanje baze podataka:

```
# kreiranje baze
create database teachabit;

# spajanje na bazu
\c teachabit

# kreiranje sheme
create schema backend;

# kreiranje korisnika
create user teachabit_backend with password lozinka;

# dodavanje prava
grant usage, create on schema backend to teachabit_backend;
grant all privlages on all tables in schema backend to teachabit_backend;
```

Postavljanje backenda

Navigirajte se u TeachABit/src direktorij i preimenujte "appsettings.Development.json.txt" u "appsettings.Development.json".

Ubuntu:

```
mv appsettings.Development.json.txt appsettings.Development.json
```

Potrebno je popuniti sve tajne/ključeve.

Kreiranje tablica u bazi podataka:

```
dotnet ef database update -s TeachABit.API -p TeachABit.Model
```

Kreiranje dotnet aplikacije:

```
dotnet publish -c Release -r linux-x64 --self-contained --output ../bin
```

Postavljanje frontenda

Navigirajte se do TeachABit.Frontend i preimenujte ".env.txt" u ".env".

```
mv .env.txt .env
```

Popunite sve potrebne varijable okruženja i nakon toga kreirajte frontend.

```
npm install
npm run build
```

Postavljanje Nginx

Navigirajte se do nginx/sites-available. Dobra je praksa nazvati konfiguracijske datoteke odgovarajućim imenom.

```
cd /etc/nginx/sites-available \
mv default teachabit.org
```

Potrebno je postaviti lokacije za frontend.

```
nano teachabit.org


//////////////////////////////////////////////////////
teachabit.org datoteka
//////////////////////////////////////////////////////

...

root .../TeachABit/src/TeachABit.Front/dist;

server_name teachabit.org; # koristi točnu IP adresu / DNS ime, ili izbriši ovu liniju

location / {
    try_files $uri $uri/ /index.html;
}

location * {
    try_files $uri $uri/ /index.html;
}

location /api/ {
    proxy_pass      http://127.0.0.1:5000;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection keep-alive;
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
}

...

//////////////////////////////////////////////////////

sudo systemctl reload nginx
```

Pokretanje backenda

Navigirajte se do build direktorija TeachABit/bin i pokrenite dotnet aplikaciju.

```
chmod +x TeachABit.API.dll \
nohup dotnet TeachABit.API.dll &
```

Opis prisutpa aplikaciji na javnom poslužitelju

Nakon provedene konfiguracije korisnici mogu pristupiti aplikacije upisivanjem URL-a u preglednik: teachabit.org.

Kroz jedan semestar zadatak ovog tima bio je izraditi platformu koja omogućuje razmjenu znanja putem raznih tečajeva i foruma. Prvi sastanak bio je ključan za međusobno upoznavanje, odabir teme i raspodjelu zadataka. Tim je brojao sedam članova: troje je radilo na frontendu (od čega je jedna osoba bila zadužena i za dizajn), dvoje na backendu, jedan član je bio full-stack (zbog prijašnjeg iskustva), dok je posljednji član bio zadužen za deployment i dokumentaciju.

Tijekom projekta pojavili su se brojni tehnički problemi, no uspješno su riješeni uz pomoć iskusnijeg člana tima. U implementaciji funkcionalnosti, frontend je zaostajao za backendom, no to nije stvaralo veće poteškoće. Jedan od izazova bio je nedostatak detaljnog planiranja i dizajna aplikacije na početku projekta, što je povremeno uzrokovalo nejasnoće među članovima tima.

S obzirom na to da je samo jedan član imao iskustva u izradi web stranica, odabrani su alati u kojima je taj član radio: React za frontend i C# za backend. Tijekom projekta članovi tima stekli su nova znanja: frontend tim je savladao React, dok je backend tim naučio raditi s ASP.NET-om u C#.

Iskustvo o korištenim alatima i bolje planiranja pridonijelo bi brzini realizacije projekta. Znanja vezana uz arhitekturu koda i apstrakciju koda može pridonijeti kvaliteti razvoja programske potpore.

U budućnosti se može razmotriti dodavanje novih funkcionalnosti te uklanjanje onih koje se pokažu nepotrebnima. Poboljšanja u planiranju i suradnji tima mogla bi ubrzati razvoj i povećati kvalitetu konačnog proizvoda.

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Stvorena stranica "Home"	Dorijan Jančić	23.10.2024
0.2	Stvorena stranica "Opis projektnog zadatka"	Dorijan Jančić	23.10.2024
0.3	Stvorena stranica "Specifikacija programske potpore"	Dorijan Jančić	23.10.2024
0.4	Stvorena stranica "Popis literature"	Dorijan Jančić	23.10.2024
0.5	Stvorena stranica "Prikaz aktivnosti grupe"	Dorijan Jančić	23.10.2024
0.6	Stvorena stranica "Backend upute"	Matej Jurišić	28.10.2024
0.7	Stvorena stranica "Analiza zahtjeva"	Dorijan Jančić	28.10.2024
0.8	Stvorena stranica "Github workflow"	Matej Jurišić	22.10.2024
0.9	Dopunjena stranica "Opis projektnog zadatka"	Dorijan Jančić	9.11.2024
0.10	Dopunjena stranica "Specifikacija programske potpore"	Dorijan Jančić	9.11.2024
0.11	Stvorena stranica "Arhitektura i dizajn sustava"	Dorijan Jančić	9.11.2024
0.12	Dopunjena stranica "Arhitektura i dizajn sustava"	Dorijan Jančić, Matej Jurišić	13.11.2024
1.1	Preprava stranica "Specifikacija zahtjeva sustava"	Dorijan Jančić	12.12.2024
1.2	Stvorena stranica: "Arhitektura komponenata i razmještaja"	Dorijan Jančić	1.1.2025
1.3	Stvorena stranica: "Ispitivanje programskog rješenja"	Dorijan Jančić	1.1.2025.
1.4	Stvorena stranica: "Tehnologije za implementaciju aplikacije"	Dorijan Jančić	1.1.2025.
1.5	Stvorena stranica: "Upute za puštanje u pogon"	Dorijan Jančić	1.1.2025.
1.6	Stvorena stranica: "Zaključak i budući rad"	Dorijan Jančić	1.1.2025.
1.x	Dopunjena stranica: "Upute za puštanje u pogon"	Dorijan Jančić	15.1.2025
1.x	Dopunjena stranica: "Tehnologije za implementaciju aplikacije"	Dorijan Jančić	16.1.2025.
1.x	Dopunjena stranica: "Zaključak i budući rad"	Dorijan Jančić	24.1.2025.

1. Programsko inženjerstvo, FER ZEMRIS, http://www.fer.hr/predmet/proinz (http://www.fer.hr/predmet/proinz)
2. Astah Community, http://astah.net/editions/uml-new (http://astah.net/editions/uml-new)

Denevnik sastajanja

1. sastanak

- Datum: 14. listopada 2024.
- Prisustovali: D. Gabrić, I. Mitar, M. Vidmar, M. Jurišić, M. Toić, T. Sesar, D. Jančić
- Teme sastanka:

> - *Upoznavanje tima*
> - *Podjela uloge (backend, frontend, devops)*
> - *Odabir teme i razrada teme*

2. sastanak

- Datum: 21. listopada 2024.
- Prisustovali: D. Gabrić, I. Mitar, M. Vidmar, M. Jurišić, M. Toić, T. Sesar, D. Jančić
- Teme sastanka:

> - *Koordinacija projekta*

3. sastanak

- Datum: 28. listopada 2024.
- Prisustovali: I. Mitar, M. Vidmar, M. Jurišić, T. Sesar, D. Jančić

> - *Detaljna razrada strukture projekta*
> - *Definiranje prioriteta*
> - *Definiranje prvih zahtjeva*

4. sastanak

- Datum: 4. studenog 2024.
- Prisustovali: D. Gabrić, I. Mitar, M. Vidmar, M. Jurišić, M. Toić, T. Sesar, D. Jančić

> - *Koordinacija projekta*

5. sastanak

- Datum: 11. studenog 2024.
- Prisustovali: D. Gabrić, I. Mitar, M. Vidmar, M. Jurišić, M. Toić, T. Sesar, D. Jančić

> - *Revizija projekta*

6. sastanak

- Datum: 9. prosinca 2024.
- Prisustovali: D. Gabrić, I. Mitar, M. Vidmar, M. Toić, T. Sesar, D. Jančić

> - *Podjela bodova*

7. sastanak

- Datum: 16. prosinca 2024.
- Prisustovali: M. Vidmar, T. Sesar, T. Sesar, D. Jančić

> - *Revizija projekta*

8. sastanak

- Datum: 13. siječnja 2024.
- Prisustovali: D. Gabrić, I. Mitar, M. Vidmar, M. Jurišić, M. Toić, T. Sesar, D. Jančić

> - *Lock in*

9. sastanak

- Datum: 20. siječnja 2024.
- Prisustovali: D. Gabrić, I. Mitar, M. Vidmar, M. Jurišić, M. Toić, T. Sesar, D. Jančić

> - *Završno planiranje, podijela poslova*

10. sastanak

- Datum: 24. siječnja 2024.
- Prisustovali: D. Gabrić, I. Mitar, M. Vidmar, M. Jurišić, M. Toić, T. Sesar, D. Jančić

> - *Implementiranje završnih funkcionalnosti, izada dokumentacije*

Tablica aktivnosti

Dokumentacija

Komponenta	Imena
Upravljanje projektom	D. Jančić
Opis projektnog zadatka	D. Jančić, M. Jurišić
Funkcionalni zahtjevi	D. Jančić
Opis pojedinih obrazaca	D. Jančić
Dijagram obrazaca	D. Jančić
Sekvencijski dijagrami	D. Jančić
Opis ostalih zahtjeva	D. Jančić
Arhitektura i dizajn sustava	D. Jančić, M. Jurišić
Baza podataka	D. Jančić
Dijagram razreda	D. Jančić, M. Jurišić
Dijagram stanja	D. Jančić
Dijagram aktivnosti	D. Jančić
Dijagram komponenti	D. Jančić
Korištene tehnologije i alati	D. Jančić
Ispitivanje programskog rješenja	D. Jančić, T. Sesar, M. Jurišić
Dijagram razmještaja	D. Jančić
Upute za puštanje u pogon	D. Jančić
Dnevnik sastajanja	D. Jančić
Zaključak i budući rad	D. Jančić
Popis literature	D. Jančić

Praktično

Komponenta	Imena
Dizajn aplikacije	M. Vidmar
Frontend naslovna stranica	D. Gabrić, M. Vidmar, M. Jurišić
Frontend tražilica i prikaz	I. Mitar
Backend radionice	M. Toić
Backend tečajevi	T. Sesar
Backend forumi	M. Jurišić
Backend registracija / prijava	M. Jurišič
Backend slanje maila	M. Jurišić
Backend slike	M. Jurišić
Postavljanje poslužitelja	D. Jančić
Frontend radionice	D. Gabrić
Frontend tečajevi	I. Mitar
Frontend tečajevi	M. Vidmar