**VELEUČILIŠTE U RIJECI**

**Poslovni odjel**

Specijalistički diplomski stručni studij Informacijske tehnologije u poslovnim sustavima Kolegij: Programsko inženjerstvo

Igor Đorđević

**Aplikacija za pregled događaja u Gorskom kotaru (GOREvent)**

Rijeka, 2021.

**VELEUČILIŠTE U RIJECI**

**Poslovni odjel**

Specijalistički diplomski stručni studij Informacijske tehnologije u poslovnim sustavima Kolegij: Programsko inženjerstvo

**Aplikacija za pregled događaja u Gorskom kotaru**

**(GOREvent)**

MENTOR: STUDENT:

dr. sc. Marin Kaluža, prof. visoke škole Igor Đorđević

MBS:

Rijeka, 2021.

**Sadržaj**

1. Opis sustava
2. Specifikacija zahtjeva
3. Arhitektura sustava
4. Opis korištenih tehnologija

3.1. React

3.2. SQLite

1. Postavljanje razvojnog okruženja
2. Dijagram akcija
3. Dijagram sekvenci
4. Dijagram klasa

**Opis sustava**

Izradom ove aplikacije biti će omogućen pregled/prikaz svih događaja na području Gorskog kotara. Ideja je da se njome koriste stanovnici Republike Hrvatske kao i turisti pa bi bio omogućen prikaz na hrvatskom i engleskom jeziku.

Unose se događaji po datumu, vrsti, lokaciji, ciljanoj publici sa opisom događaja.  
Mogućnost pregleda svih događaja, a moći će se i filtrirati događaji po navedenim parametrima (datum, vrsta, lokacija, ciljana publika).

Korisnici bi se po želji mogli registrirati/prijaviti (osobni podaci uključujući godine, spol, interese…) u aplikaciju te bi imali mogućnost spremanja događaja za koji su zainteresirani na listu favorita te eventualno dodati podsjetnik za određeni događaj u budućnosti. Neregistrirani korisnici mogu pregledavati buduće te eventualno prošle događaje.

Ideja za izradom aplikacije ovoga tipa proizlazi iz činjenice da (za sada) ne postoji nešto slično čime su objedinjeni svi događaji na području GK već lokalno općine (na službenim stranicama općina ili turističkih zajednica) objavljuju svoje događaje.

**Specifikacija zahtjeva**

U aplikaciju se može prijaviti administrator te korisnik ako za to postoji potreba/želja. Administrator nakon prijave može upravljati događajima – može kreirati nove događaje, ažurirati postojeće te brisati prošle.

Korisnik može pregledavati događaje i njihove značajke – tip, vrijeme i mjesto izvođenja itd.

Po potrebi korisnik se može prijaviti u aplikaciju čime mu se otvaraju dodatne mogućnosti – nakon pregleda događaja koje želi može dodati u listu favorita, a sustav mu onda za te događaje određeno vrijeme prije njihova termina šalje obavijest o nadolazećem događaju. Prošli događaji iz liste favorita se automatski brišu.

Tablica 1: Specifikacija zahtjeva

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Uloga** | **Radnja** | **Rezultat** |
| 1. | Prijava - administrator | Za ažurno vođenje evidencija događaja zaslužan je administrator sustava koji se prijavljuje u sustav. | Unosom korisničkog imena i lozinke ostvaruje se pravo pristupa i rada sa podacima. |
| 2. | Dodavanje događaja | Nakon prijave administrator po potrebi dodaje nove/nadolazeće događaje sa pripadajućim podacima. | Novi događaj vidljiv je prilikom odabira svih događaja. |
| 3. | Ažuriranje događaja | Nakon prijave administrator po potrebi ažurira postojeće događaje sa pripadajućim/promijenjenim podacima. | Ažurirani događaji vidljivi su prilikom odabira svih događaja. |
| 4. | Brisanje događaja | Nakon prijave administrator briše događaje koji su prošli ili su iz nekog razloga otkazani. | Obrisani događaji više nisu vidljivi prilikom odabira svih događaja. |
| 5. | Korisnik | Svaki korisnik može pregledavati informacije o nadolazećim događajima. | Korisnik vidi trenutno aktualne događaje sa pripadajućim informacijama. |
| 6. | Prijava - korisnik | Korisnik se unosom korisničkog imena i lozinke može prijaviti u sustav. | Nakon prijave korisnik ima uvid u aktualne događaje sa mogućnošću spremanja događaja u favorite. |
| 7. | Favoriti | Prijavljeni korisnik ima mogućnost dodavanja određenih događaja u favorite. | Ažuriranjem favorita korisnik si osigurava lakše praćenje događaja. |
| 8. | Obavijest – nadolazeći događaj | Stavljanjem određenog događaja u favorite korisnik ima bolji uvid u događaje | Korisnik dobiva obavijest o nadolazećem događaju. |

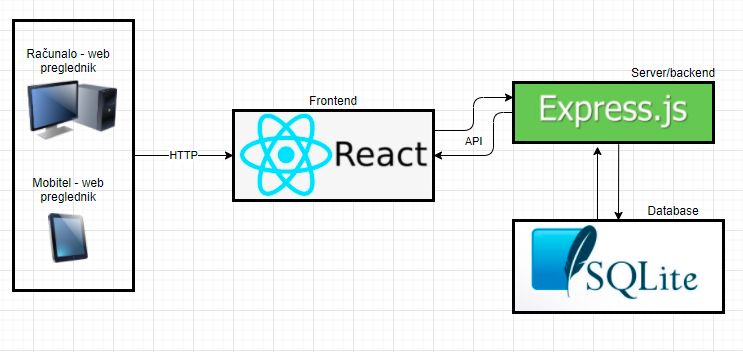
Izvor: autor

**Arhitektura sustava**

Arhitektura aplikacije za evidenciju događaja u Gorskome kotaru sastojala bi se od korisničkih sučelja kako za računalo tako i za mobilni uređaj, web poslužitelja i baze podataka za spremanje osnovnih podataka.

Aplikaciji bi se pristupalo koristeći web preglednik (mogućnost preko računala ili mobilnog uređaja) gdje korisnik šalje zahtjev web poslužitelju. Rezultat je prikaz traženog sadržaja (React framework) sa podacima koji bi se preuzimali iz baze podataka na poslužitelju (SQLite).

Slika 1: Arhitektura sustava



Izvor: autor

**Opis korištenih tehnologija**

Tehnologije koje će biti korištene za izradu ove aplikacije su React framework kao front-end te SQLite za bazu podataka. Kao backend server koristiti će se Express.js.

**React**

React (React.js ili ReactJS) je JavaScript knjižnica otvorenog koda (open-source) koja nudi pregled podataka koji se generiraju kao HTML. Specifičnosti Reacta su:

- pisanje komponenti koje se prikazuju kao posebne HTML oznake (tags)

- "nizvodni tok podataka" (engl. "data flows down"), odnosno nemogućnost komponenti da izmijene svoje nadkomponente

- jasno razdvajanje komponenti koje je vrlo korisno za razvoj modernih jednostraničnih aplikacija (SPA), budući da je glavna ideja da komponente budu ponovno iskoristive u različitim aplikacijama, i kada se podaci koje prikazuju promijene

Tradicionalno, korisnička sučelja web aplikacija grade se pomoću HTML predložaka (engl. templates), koji definiraju sve mogućnosti koje se nude za izradu korisničkog sučelja. React za izgradnju komponenti koristi pravi i potpuni programski jezik, JavaScript, koji ima nekoliko prednosti nad korištenjem predložaka. JavaScript je fleksibilan i moćan programski jezik koji ima mogućnost izgradnje apstraktnih klasa, što je od izuzetne važnosti pri izradi velikih aplikacija, a sjedinjenje prezentacijskih dijelova s odgovarajućom logikom olakšava održavanje i proširenje React koda. Prednosti Reacta najviše se ističu kada se podaci koje prikazuje mijenjaju tijekom vremena. U tradicionalnim JavaScript aplikacijama, nakon svake promjene podataka potrebno je napraviti odgovarajuće promjene u DOM-u. React eliminira potrebu za ručnim izmjenama. Kada se komponenta prvi put inicijalizira, poziva se render metoda koja generira jedan prikaz. Kada se podaci promijene, ponovno se poziva render metoda, ali React uspoređuje prethodni poziv s trenutnim i generira minimalni set promjena koje će se primijeniti na DOM. Ovaj se proces naziva reconciliation. Budući da je takvo generiranje jako brzo, nije potrebno eksplicitno definirati podatkovno uparivanje (engl. data bindings - procesi koji povezuju korisničko sučelje s aplikacijskom logikom).

**SQLite**

Najjednostavnije rečeno, SQLite je programski paket otvorenog koda (engl. open-source) koji služi za upravljanje DBMS-om. SQLite je izvorno napisan pomoću programskog jezika C i pripadnih biblioteka pa se često u literaturi definira i kao programski jezik namijenjen za upravljanje podacima unutar DBMS-a. Za usporedbu, najpoznatiji paketi za upravljanje relacijskim bazama koji su temeljeni na otvorenom kodu su MySQL i PostgreSQL. Od komercijalne programske podrške ističu se Oracle Database, Microsoft SQL Server i IBM DB2 u manjoj mjeri. Prilikom imenovanja softvera ili općenito proizvoda danas je uobičajen sufiks Lite koji u ovom slučaju označava manju opću složenost, lakšu administraciju i upravljanje te ono što je najvažnije, veoma malu potrebu za resursima uređaja u odnosu na navedenu konkurenciju. Osim navedenih prednosti, SQLite se najviše razlikuje od ostalih DBMS sustava po komunikacijskoj arhitekturi. Ostali sustavi su kao i većina aplikacija zasnovani na klijentsko - poslužiteljskoj arhitekturi. To uvjetuje izvršavanje svih upita na poslužiteljskom računalu iako je sam program instaliran i na klijentskom računalu. SQLite je pokrenut na uređaju gdje je instaliran i upiti te sva ostala logika obavlja se na klijentskoj strani što ne bi bilo moguće sa zahtjevnijim inačicama DBMS-a (u smislu potrebnih resursa).

**Postavljanje razvojnog okruženja**

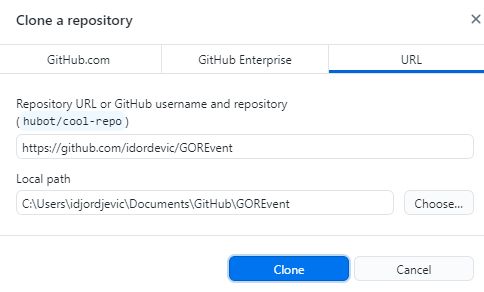
Postavljene git repozitorija na GitHab okruženju.

Prijavimo se sa korisničkim imenom i lozinom (//<https://github.com/>).

Kreiramo novi projekt/repozitorij sa traženi podacima – opisom, opcijom Public/Private itd.

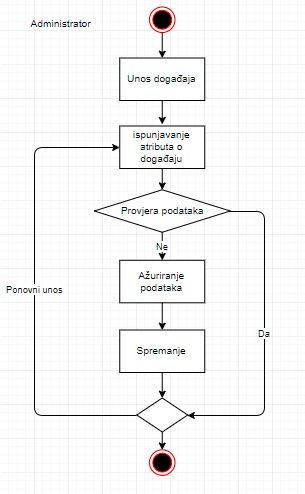
Na kreirani projekt/repozitorij dodajemo nastavnika kao suradnika.

Instaliramo GitHub Deskop Git klijent i povezujemo ga sa GitHab repozitorijem.



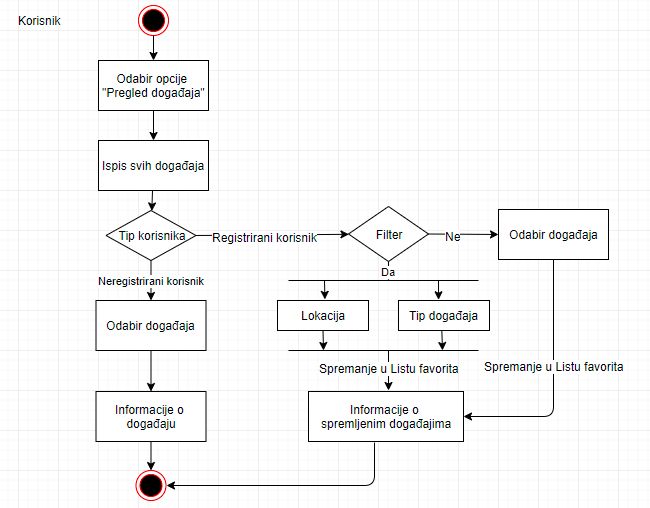
**Dijagram akcija**

Slika 2: Dijagram akcija - administrator

****

Izvor: autor

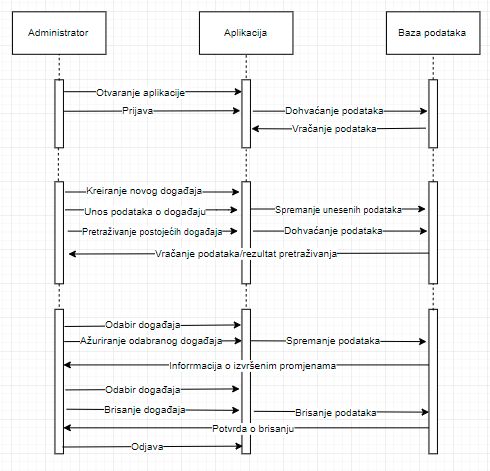
Slika3: Dijagram akcija - korisnik



Izvor: autor

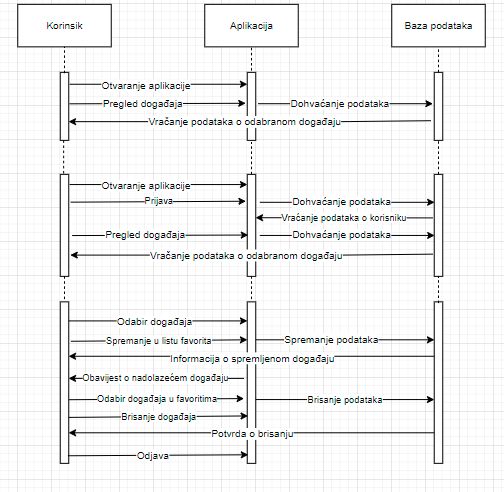
**Dijagram sekvenci**

Slika4: Dijagram sekvenci – administrator



Izvor: autor

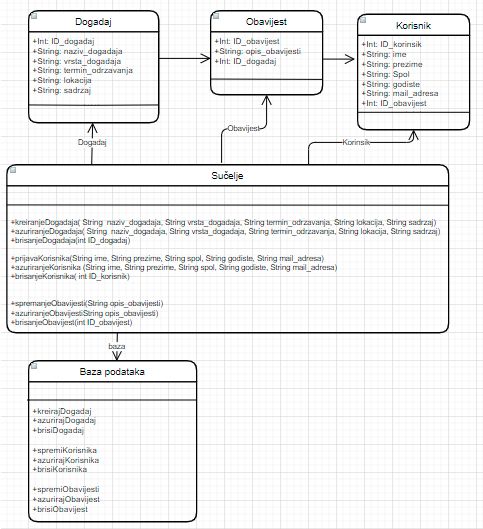
Slika5: Dijagram sekvenci – korisnik



Izvor: autor

**Dijagram klasa**

Slika5: Dijagram sekvenci – korisnik



Izvor: autor