**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 1](#_Toc74863933)

[2. SPECIFIKACIJA ZAHTJEVA 2](#_Toc74863934)

[3. ARHITEKTURA SUSTAVA 4](#_Toc74863935)

[4. OPIS KORIŠTENIH TEHNOLOGIJA 5](#_Toc74863936)

[4.1. React 5](#_Toc74863937)

[4.2. Express.js 6](#_Toc74863938)

[4.3. MySQL 6](#_Toc74863939)

[5. POSTAVLJANJE RAZVOJNOG OKRUŽENJA 8](#_Toc74863940)

[6. DIJAGRAM AKCIJA 9](#_Toc74863941)

[7. DIJAGRAM SEKVENCI 11](#_Toc74863942)

[8. DIJAGRAM KLASA 13](#_Toc74863943)

[9. STRUKTURA PROJEKTA 14](#_Toc74863944)

[10. MAPIRANJE 15](#_Toc74863945)

[10.1. Navigacijska traka 16](#_Toc74863946)

[10.2. Novi događaj – AddDogadaj.js 17](#_Toc74863947)

[10.3. Lista događaja – DogadajiList.js 18](#_Toc74863948)

[10.4. Detalji o odabranom događaju – Dogadaj.js 19](#_Toc74863949)

[11. DIZAJN KORISNIČKOG SUČELJA 20](#_Toc74863950)

[12. TESTIRANJE SIGURNOSTI 21](#_Toc74863951)

[12.1. Greške srednje razine: 22](#_Toc74863952)

[12.2. Greške niske razine: 22](#_Toc74863953)

[12.3. Greške informacijske razine: 23](#_Toc74863954)

[13. INSTALACIJA 24](#_Toc74863955)

[14. KORISNIČKE UPUTE 26](#_Toc74863956)

[ZAKLJUČAK 26](#_Toc74863957)

[POPIS SLIKA 28](#_Toc74863958)

# 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. |

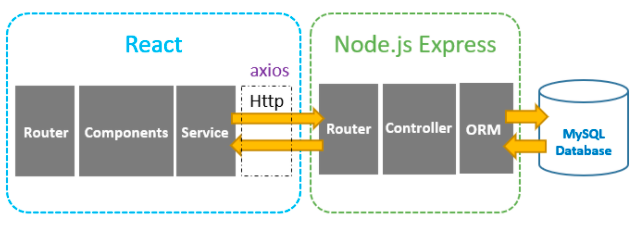
Izvor: autor

# 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 (MySQL).

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 MySQL 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).

## Express.js

Express predstavlja fleksibilan okvir web aplikacija Node.js koji pruža skup značajki za razvoj web i mobilnih aplikacija. Omogućuje brzi razvoj web aplikacija utemeljenih na čvorovima. Osnovne značajki Express-a su:

* omogućuje postavljanje posredničkog softvera za odgovaranje na HTTP zahtjeve
* definira tablicu usmjeravanja koja se koristi za izvođenje različitih radnji na temelju HTTP metode i URL-a.
* omogućuje dinamičko prikazivanje HTML stranica na temelju prosljeđivanja argumenata na predloške.

Express instaliramo globalno pomoću NPM-a tako da se može koristiti za stvaranje web aplikacije pomoću node terminala (npm install express --save). Ova naredba lokalno sprema instalaciju u direktorij node\_modules i stvara direktorij ekspres unutar node\_modules. Važni moduli koji trebaju biti instalirani zajedno s express-om su:

* body-parser - node.js posrednički softver za rukovanje podacima s obrascima kodiranim JSON-om, tekstom i URL-om.
* multer - node.js posrednički softver za rukovanje podacima s više dijelova / obrazaca
* cookie-parser

## MySQL

MySQL je besplatan sustav za upravljanje bazom podataka otvorenog koda. Uz PostgreSQL, MySQL je čest izbor baze za projekte otvorenog koda, poput Linuxovih servera, no postoje inačice i za ostale operacijske sustave poput macOS-a, Windowsa i dr.

MySQL baza je slobodna za većinu uporaba. Ranije u svom razvoju, MySQL baza podataka suočila se s raznim protivnicima MySQL sustava organiziranja podataka jer su joj nedostajale neke osnovne funcije definirane SQL standardom. Naime, MySQL baza je optimizirana kako bi bila brza nauštrb funkcionalnosti. Nasuprot tome, vrlo je stabilna i ima dobro dokumentirane module i ekstenzije te podršku od brojnih programskih jezika: PHP, Java , Perl, Python...

MySQL baze su relacijskog tipa, koji se pokazao kao najbolji način skladištenja i pretraživanja velikih količina podataka i u suštini predstavljaju osnovu svakog informacijskog sustava, tj. temelj svakog poslovnog subjekta koji svoje poslovanje bazira na dostupnosti kvalitetnih i brzih informacija.

Kao i ostali sustavi za upravljanje bazama, i MySQL poštuje ACID načela pri izvođenju transakcija nad podatcima.

Osnovi element koji se pohranjuje u bazi je entitet koji može biti bilo što: osoba, neki objekt, događaj, služba u nekoj organizaciji i sl. - stvari iz stvarnog života o kojima želimo čuvati informacije. Drugi važan pojam u teoriji baza podataka je relacija. Kao što u stvarnom životu postoje određeni međusobni odnosi između dvije ili više osoba, događaja isl. tako se i u bazama podataka mogu pojaviti određeni odnosi ili relacije između raznih entiteta, koji se na odgovarajući način predstavljaju unutar same baze.

# POSTAVLJANJE RAZVOJNOG OKRUŽENJA

Početni korak za postavljanje razvojnog okruženja je preuzimanje Visual Studio Code nakon čega je potrebno na Visual Studio Code IDE-u preuzeti dodatak GitLens.

Postavljene git repozitorija na GitHab okruženju.

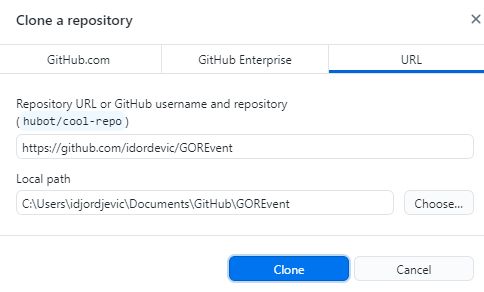
Prijavimo se sa korisničkim imenom i lozinom (//<https://github.com/>).

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.

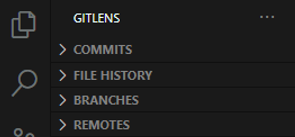
Slika 2: povezivanje GitHub Desktop/GitHab repozitorij



Izvor: autor

U Visual Studio Code-u instaliramo plug-in GitLens i pomoću linka unutar VS Code-a kloniramo repozitorij.

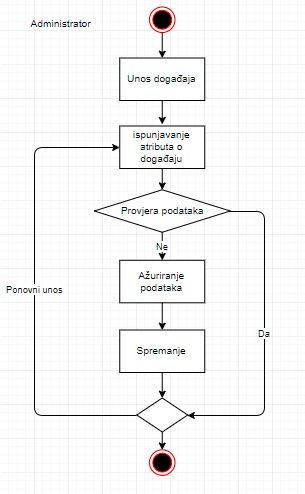
Slika 3: Kloniranje repozitorija



Izvor: autor

# DIJAGRAM AKCIJA

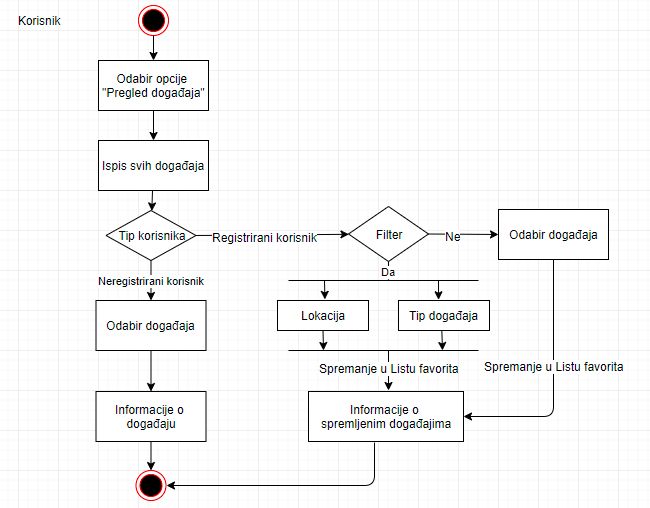
Slika 4: Dijagram akcija - administrator

****

Izvor: autor

Aplikacija je zamišljena na načina da administrator sustava unosi podatke o određenim događajima, provjerava i održava te podatke te po završetku briše ili ih proglasi neaktivnima, a samim time i nevidljivima krajnjem korisniku (radi bolje preglednosti aktualnih informacija). Svi aktivni događaji vidljivu su krajnjem korisniku bez obzira ne njegov status (da li je korisnik prijavljen ili nije).

Slika 5: Dijagram akcija - korisnik



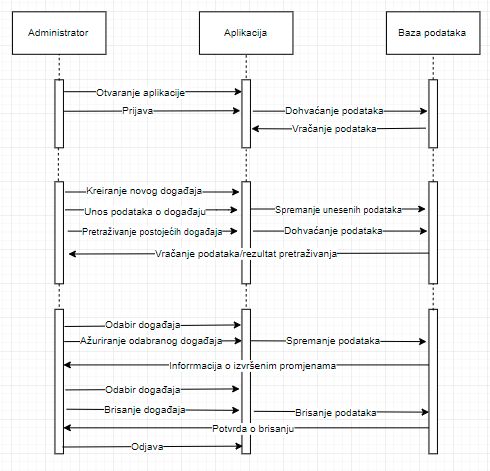
Izvor: autor

Korisnik ima mogućnost pregleda informacija o svim aktivnim događajima. Također korisnik ima mogućnost prijave u aplikacije. Za neprijavljene korisnike na raspolaganju je prikaz svih događaja te odabirom na željeni događaj prikaz detalja o tom događaju.

Prijavljeni korisnik osim pregledavanja događaja ima mogućnost spremanja događaju u listu favorita sa povratnim informacija (notifikacija) o onim događajima koji slijede unutar kratkog vremena (5 dana) te ima mogućnost filtriranja događaja prema lokaciji i tipu događaja ukoliko korisnik preferira određenu vrstu/tip događaja.

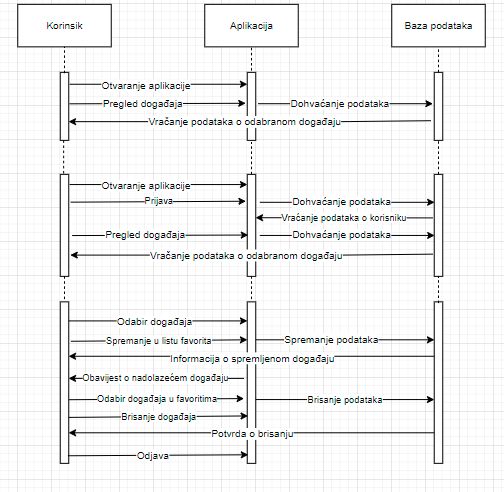
# DIJAGRAM SEKVENCI

Slika 6: Dijagram sekvenci – administrator



Izvor: autor

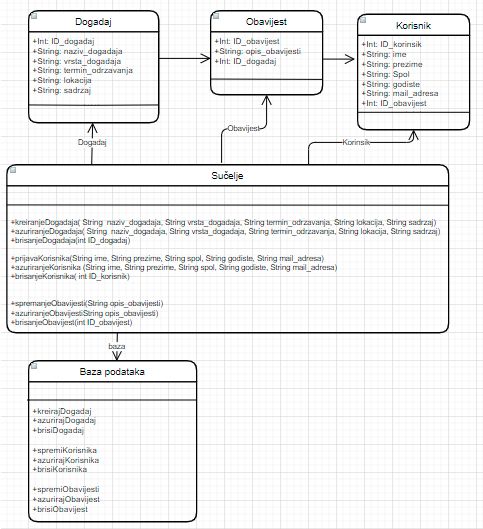
Slika 7: Dijagram sekvenci – korisnik



Izvor: autor

# DIJAGRAM KLASA

Slika 8: Dijagram klasa

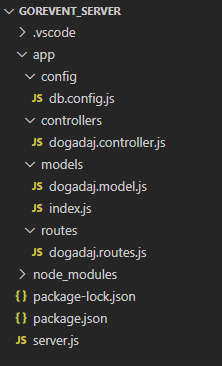
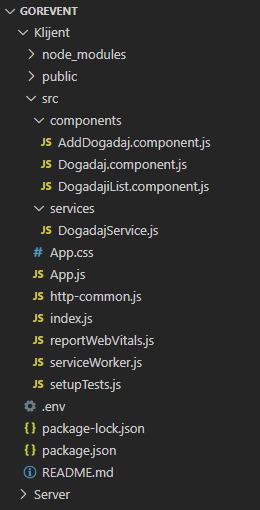


Izvor: autor

# STRUKTURA PROJEKTA

React.js– frontend Node.js Express - backend

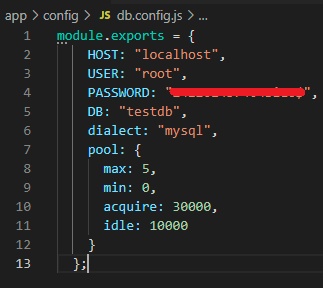
Slika 9: Struktura projekta



Izvor: autor

# MAPIRANJE

Slika 10: db.config.js



Izvor: autor

Prvih pet parametara definira potrebne podatke za vezu sa bazom podataka (MySql) – podaci o host-u, korisničkom imenu i lozinci usera, naziv baze.

Pool parametar nije obavezan –

* max: maksimalan broj veza
* min: minimalan broj veza
* acquire: maksimalno vrijeme (u milisekundama) da veza može biti neaktivna prije puštanja
* idle: maksimalno vrijeme (u milisekundama) za uspostavu vezu prije poruke o grešci

## 10.1. Navigacijska traka

Slika 11: App.js



Izvor: autor

Navigacijska traka (unutar App.js-a) – prikazuje se naziv/logo aplikacije (GOREvent), sa opcijama za odabir gumba Događaji (otvara nam se popis svih trenutno aktivnih događaja) i gumba Dodaj gdje se otvara forma za unos novog događaja sa definiranim podacima o samom događaju.

## 10.2. Novi događaj – AddDogadaj.js

Slika 12: AddDogadaj.js



Izvor: autor

Prvo definiramo i postavljamo početno stanje (dogadaj& submit).

Dalje, kreiramo funkciju handleInputChange () za praćenje vrijednosti ulaza i praćenja stanja za promjene. Funkciju za dobivanje stanja i slanje zahtjeva web API-ju (poziva metodu DogadajDataService.create). Za povratak provjeravamo novo stanje (true) - gumb Dodaj za upis novih vrijednosti.

## 10.3. Lista događaja – DogadajiList.js

Slika 13: DogadajList.js



Izvor: autor

Prikazuju se događaji po nazivu. Odabirom na željeni događaj („Klikni na odabrani događaj“) podaci o događaju otvaraju se s desne strane.

Upotrijebit ćemo naredbu useEffect () za dohvaćanje podataka s web API-ja. Ova naredba govori Reactu da komponenta mora nešto poduzeti nakon prikaza ili izvođenja DOM ažuriranja. Na taj način vršimo dohvaćanje podataka iz API-ja.

## 10.4. Detalji o odabranom događaju – Dogadaj.js

Slika 14: Dogadaj.js



Izvor: autor

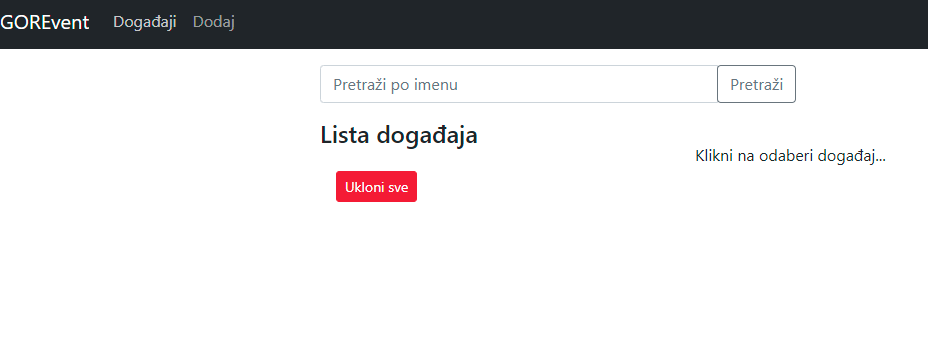
Za dobivanje podataka i ažuriranje/brisanje događaja, ova komponenta koristi 3 funkcije DogadajDataService:

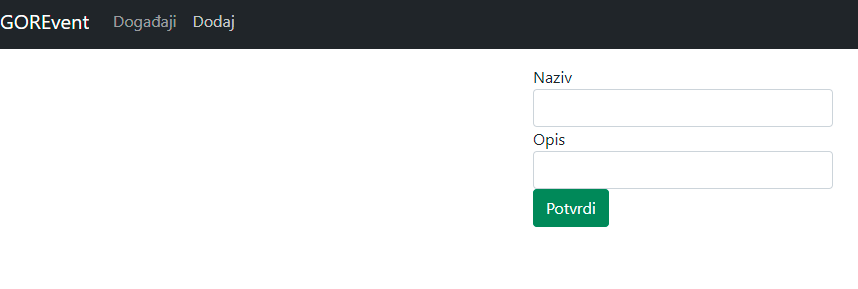
* get()
* update ()
* delete()

Također koristimo naredbu useEffect () za dobivanje parametara po id-u.

# DIZAJN KORISNIČKOG SUČELJA

Slika 15: Dizajn korisničkog sučelja





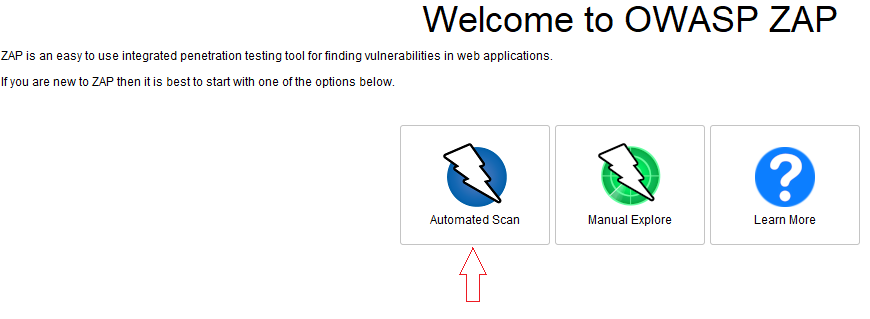
Izvor: autor

Osnovni prikaz dizajna korisničkog sučelja – događaji se unose po nazivu, a pod opisom se zapisuju podaci o događaju (vrsta, termin, lokacija, sadržaj). Događaji se prikazuju pod karticom događaji, a klikom na odabrani događaj prikazuju se informacije o događaju. Registrirani korisnici prilikom pregleda informacija mogu spremiti događaj u listu favorita.

# TESTIRANJE SIGURNOSTI

Zed Attack Proxy alat odabran je kontrolu testa sigurnosti. ZAP predstavlja jedan od vodećih alata za provedbu penetracijskog testa. Nakon preuzimanja alata sa službenih stranica i uspješne instalacije pokretanjem samog alata dobijemo jednostavno i pregledno, lako za koristiti sučelje koje nam između ostaloga nudi i opciju automatskog testiranja s ciljem dobivanja relevantnijih podataka.

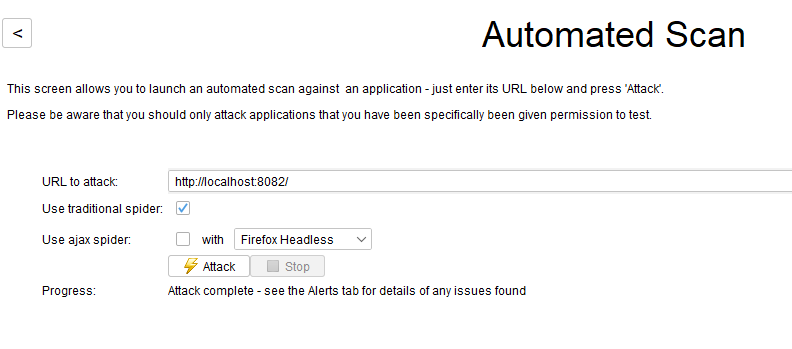
Slika 16: Testiranje sigurnosti (ZAP)



Izvor: autor

Odabirom opcije Automated Scan otvara se novi prozor sa podacima za unos – unosimo lokaciju naše web aplikacije koja je preko VSC-a pokrenuta na localhost-u. Testiranje započinjemo odabirom opcije Attack.

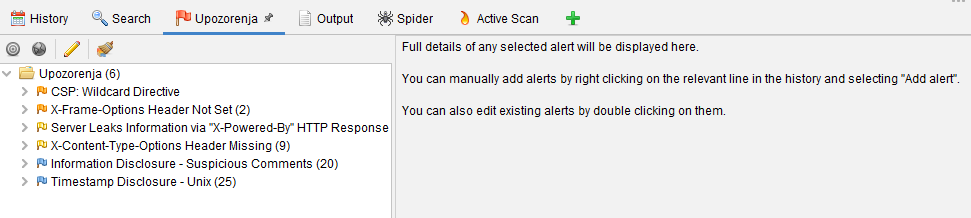
Slika 17: Testiranje sigurnosti (ZAP)



Izvor: autor

Rezultati se prikazuju u obliku prikazanom na slici niže. Iz rezultata možemo vidjeti da imamo ukupno 4 prijetnje od kojih su dvije srednjeg rizika te dvije nižeg rizika (intenzitet boja žuta/narančasta).

Slika 18: Testiranje sigurnosti (ZAP)



Izvor: autor

## Greške srednje razine:

1. CSP: Wildcard Directive – ova prijetnja pokazuje da ne postoji nikakav „Content-Security-Policy“ instaliran na web aplikaciji, (primarna svrha mu je detekcija i ublažavanje data injection napada)
2. X-Frame-Options Header Not Set- prijetnja koja pokazujene postoji zaštita od „ClickJacking“ napada (sakriveni/nevidljivi gumbovi na stranici ) čijim se slučajnim odabirom pokreću radnje koje mogu naštetiti računalu korisnika sa kojega je pokrenuta aplikacija

## Greške niske razine:

1. Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)- greška niske razine koja govori kako postoji mogućnost od proboj informacija iz baze podataka (ukoliko se u bazi nalaze osobni/povjerljivi podaci tada to može predstavljati problem i po mome mišljenju biti greška neke više razine)
2. X-Content-Type-Options Header Missing- greška niske razine koja ukazuje da ne postoji nikakav x-content (sprječava neželjeno „njuškanje“ od strane potencijalnog napadača)

## Greške informacijske razine:

1. Information Disclosure - Suspicious Comments – mogućnost postojanja komentara u kodu koje bi potencijalni napadači mogli iskoristiti s ciljem dobivanja informacije iz baze podataka
2. Timestamp Disclosure – Unix – web server pruža *timestamp* (trenutno vrijeme po *Unix* standardu)

# INSTALACIJA

Za izradu aplikacije GOREvent koristilo se:

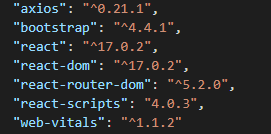
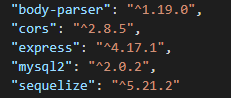
Razvojno okruženje: Visual studio Code.

Frontend alat: React (ver. 17.0.2)

Backend: Node.js Express (ver. 4.17.1)

Za uspješno pokretanje aplikacije kao i za njenu izradu koristili su određeni plug-inovi kako na klijentskoj strani tako i na serverskoj. Svi potrebni plug-inovi prikazani su na slici niže.

Slika 19: Potrebni plug-inovi (klijent/server)

Izvor: autor

Na računalo na kojem želimo pokretati aplikaciju ukoliko nemamo moramo skinuti i instalirati node.js (<https://nodejs.org/en/download/>).

Gotovi projekt možemo preuzeti sa Github-a.

Projekt se nalazi na poveznici: <https://github.com/idordevic/GOREvent/tree/master>

Potrebni plug-inovi instaliraju se redom naredbom (primjer: „npm install axios, npm install body-parser “). Prilikom instalacije moramo voditi računa da smo pozicionirani na klijentskoj ili serverskoj strani – ovisno o tome kamo želimo/imamo potrebu instalirati određeni plug-in.

Za pokretanje na serverskoj strani potrebo je upisati naredbu „node server.js“ a na klijentskoj strani „npm start“.

# KORISNIČKE UPUTE

Aplikacija GOREvent rađena je s idejom i služi za evidenciju događaja (manifestacija, kulturno/zabavnih programa itd.) na području Gorskoga kotara.

Zamišljena je na način da bude dostupna tj. da je koriste „obični“ korinsnici kao i oni koji se svoji podacima mogu prijaviti u aplikaciju.

Običan korisnik ima opcije pregledavanja svih događaja skupno te pregledavanja detalja o odabranom događaju. Odabirom opcije „Klikni na odabrani događaj“ otvara se nova forma sa detaljima o samom događaju (tip, događaja, vrijeme održavanja, lokacija...).

Prijavljeni korisnik dodatno može pretraživati događaje po određenim parametrima te ukoliko želi može određeni događaj odabirom opcije Favorit spremiti u Listu favorita te na taj način pratiti informacije samo o onim događajima za koje je stvarno zainteresiran.

Događaji koji su prošli brišu se iz baze kao i iz liste favorita. Na taj način smanjuje se količina nepotrebnih informacija te povećava funkcionalnost i preglednost same aplikacije.

Po završetku pregleda odabirom opcije Odjava korisnik se odjavljuje iz aplikacije.

# ZAKLJUČAK

Izrada ove aplikacije za mene kao izvođača sigurno predstavlja jedno pozitivno i novo iskustvo. Rad i upoznavanje sa meni novim tehnologija i tehnologijama kojima nisam prije radio predstavljalo je veliki izazov. Prije svega shvaćanjem načina na koji pojedina tehnologija radi, pa onda i načina kako sve povezati i dobiti jednu funkcionalnu sredinu.

U navedenom/zadanom roku nisu ispunjene sve početno definirane funkcionalnosti ali svakako postoji želja i način da se one ipak odrade.

Prva takva funkcionalnost je dodavanje mogućnosti prijave u aplikaciju koja dalje nudi/povlači određene radnje (favoriti događaja, pretraga događaja.)

Jedna od zanimljivosti koja bi digla aplikaciju na novi/viši nivo je dodavanje slika (npr. plakata za određeni događaj) te mogućnost postavljanja kratkih videa (s ciljem reklamiranja događaja).

Korišteni alati u svakom slučaju omogućuju aktivnosti koje nas koristeći na ispravan način dovode u situaciju i mogućnost stvaranja zavidnih aplikacija.

# POPIS SLIKA

[Slika 1: Arhitektura sustava 4](#_Toc74864005)

[Slika 2: povezivanje GitHub Desktop/GitHab repozitorij 8](#_Toc74864006)

[Slika 3: Kloniranje repozitorija 8](#_Toc74864007)

[Slika 4: Dijagram akcija - administrator 9](#_Toc74864008)

[Slika 5: Dijagram akcija - korisnik 10](#_Toc74864009)

[Slika 6: Dijagram sekvenci – administrator 11](#_Toc74864010)

[Slika 7: Dijagram sekvenci – korisnik 12](#_Toc74864011)

[Slika 8: Dijagram klasa 13](#_Toc74864012)

[Slika 9: Struktura projekta 14](#_Toc74864013)

[Slika 10: db.config.js 15](#_Toc74864014)

[Slika 11: App.js 16](#_Toc74864015)

[Slika 12: AddDogadaj.js 17](#_Toc74864016)

[Slika 13: DogadajList.js 18](#_Toc74864017)

[Slika 14: Dogadaj.js 19](#_Toc74864018)

[Slika 15: Dizajn korisničkog sučelja 20](#_Toc74864019)

[Slika 16: Testiranje sigurnosti (ZAP) 21](#_Toc74864020)

[Slika 17: Testiranje sigurnosti (ZAP) 21](#_Toc74864021)

[Slika 18: Testiranje sigurnosti (ZAP) 22](#_Toc74864022)

[Slika 19: Potrebni plug-inovi (klijent/server) 24](#_Toc74864023)