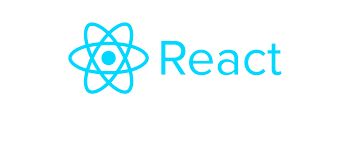
**Internet softverske arhitekture**

**-Projektna dokumentacija za završni ispit-**

****

****

**Uvod**

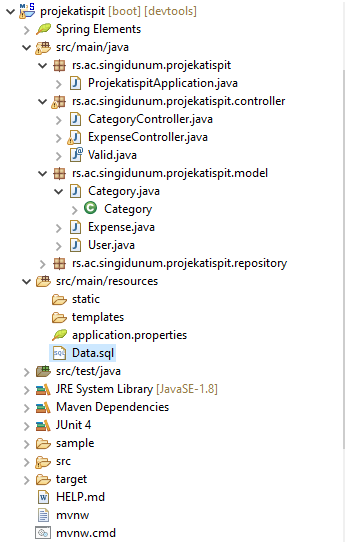
Aplikacija je kreirana u Eclipse okruženju uz pomoć Maven alata. Aplikacija se sastoji iz **Spring** dela (projekta) koji predstavlja **back-end** ove aplikacije, kao i iz **React** dela koji predstavlja **front-end** impelemntirane aplikacije. React u ovom projektu zapravo predstavlja grafički korisnički interfejs (GUI). Aplikacija **YourMovies** omogućava svojim korisnicima dodavanje na listu želja filmove koje žele da pogledaju. Takođe, korisnik kasnije može da ukloni odgledani film jednim klikom na dugme. Aplikacija je vrlo jednostavna za upotrebu.

**Prikaz korišćenih tehnologija i detalji implementacije**

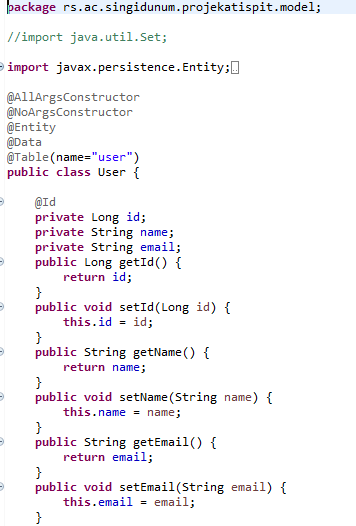
Prvo je potrebno otići na zvanični Spring Initializer veb sajt, gde se konfigurišu osnovna podešavanja Spring projekta u RESTful web servisa kroz sledeće jednostavne korake:

1. Prvi zahtevi su da projekat bude Maven u Java tehnologiji i Spring Boot verziji
2. Group: rs.ac.singidunum (naziv grupe je isto što i naziv paketa)
3. Artifact: projekatIspit
4. Name: projekatIspit (naziv projekta)
5. Description: Demo Singidunum Project for Spring Boot
6. Dodavanje zavisnosti: veb, JPA, H2, DevTools(vrši se dodavanje Maven zavisnosti)
7. Klikom na dugme Generate Project generiše se spring Boot projekat prema željenim specifikacijama koji se preuzima u formi zip arhive

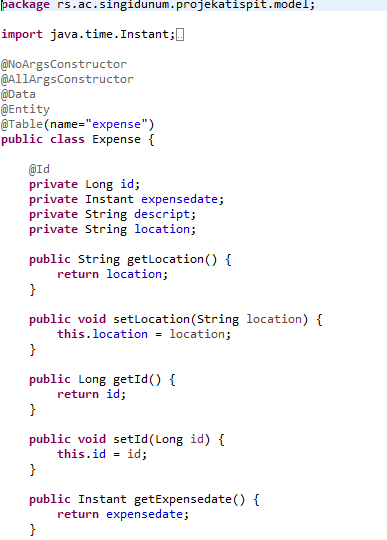
Nakon pokretanja **Eclipse** okruženja potrebno je izabrati opciju **Create New Project** i odabrati **Maven** za tip projekta. Projekat je inicijalizovan pomoću Spring Boot alata. U nastavku je prikazana struktura ovog projekta. Glavni paket ovog projekta je **rs.ac.singidunum.projekatIspit**.

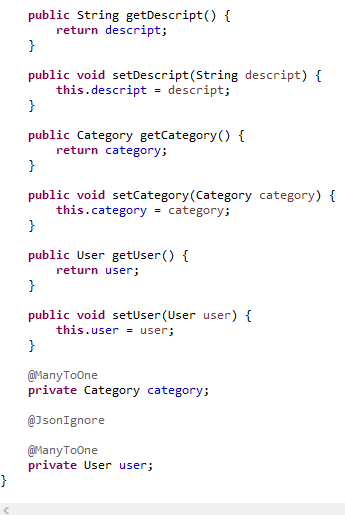


Kod klase **User** je prikazan u nastavku. Klasa User se nalazi u paketu **rs.ac.singidunum.projekatIspit.model**.



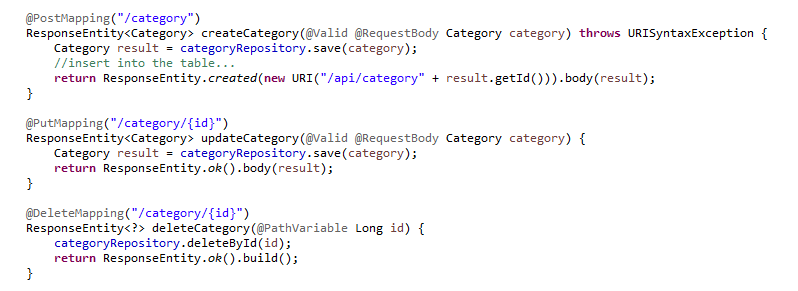
Kod klase **Expense** (koja predstavlja filmove) u paketu rs.ac.singidunum.projekatIspit.model:



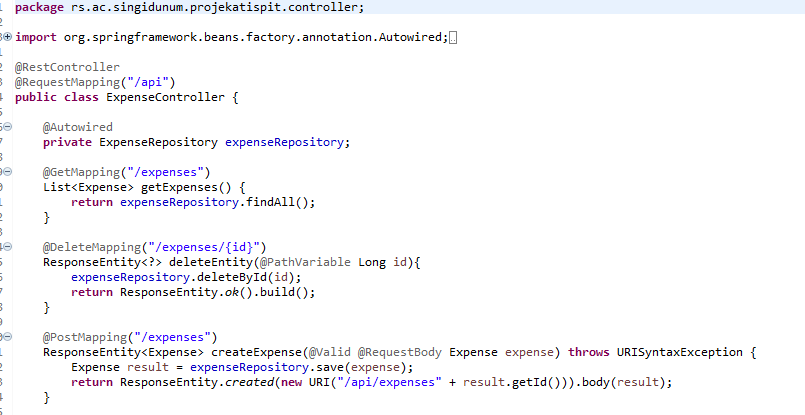


**CategoryController** klasa u paketu **rs.ac.singidunum.projekatIspit.controller**:

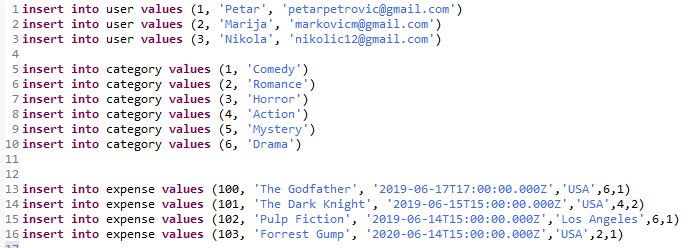




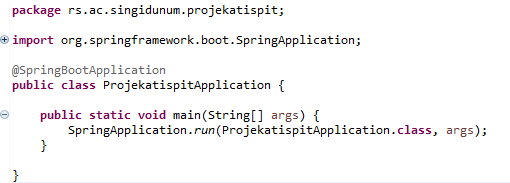
Klasa **ExpenseController** koja se takođe nalazi u paketu **rs.ac.singidunum.projekatIspit.controller**:



Potrebno je da napravimo našu **bazu podataka**. To ćemo učiniti u samom Eclipse okruženju tako što ćemo instalirati SQL editor preko Eclipse Marketplace-a.

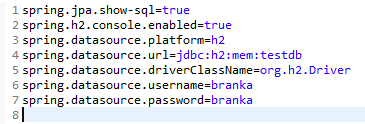


Kod glavne klase (**main**) ovog projekta:

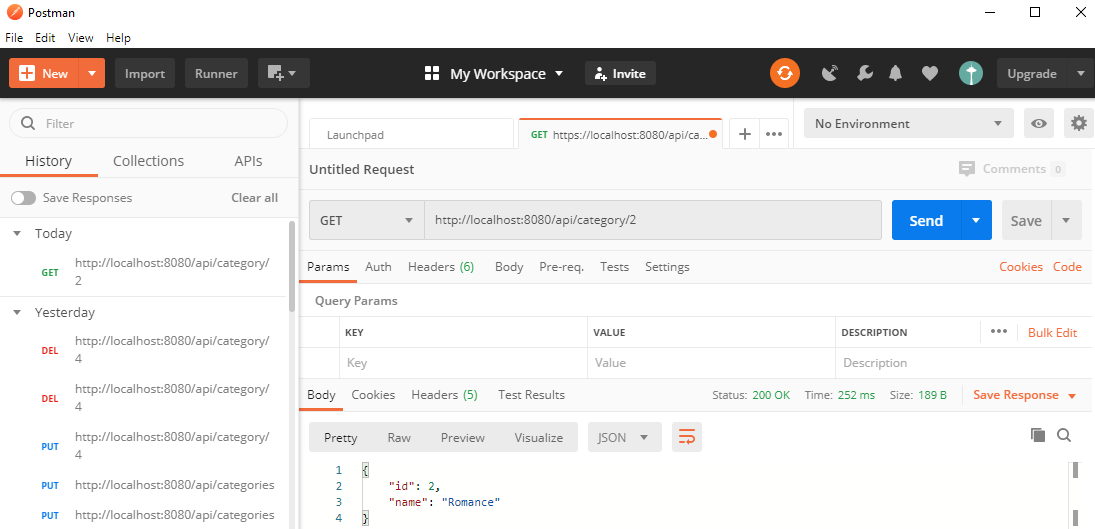


Anotacija **@SpringBootApplication** mora da se doda ispred deklaracije glavne klase zato što ova anotacija upućuje da se radi o Spring Boot aplikaciji. Pokretanjem aplikacije (desni klik na kod klase SingidunumSpringBootApplication -> Run pokreće se aplikacija zajedno sa ugradjenim (engl. embedded) Tomcat serverom na podrazumevanom **portu 8080**. Medutim, URL adresa http://localhost:8080/ nije u funkciji zato što još uvek nije napravljena klasa kontroler (engl. controller class) koja upravlja HTTP zahtevima klijenata.

Kako bismo mogli našu aplikaciju da otvorimo u **browser**-u ili u **Postman**-u potrebno je da u **application.properties** koji se nalazi u src/main/resources dodamo sledeće linije koda:



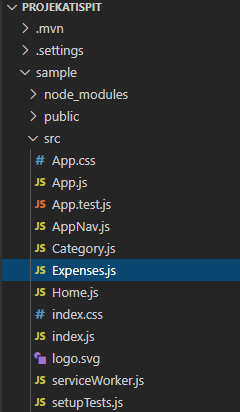
Za pristup koristimo **url:** [**http://localhost/api**](http://localhost/api)**.** Prikaz preko **Postman** aplikacije:



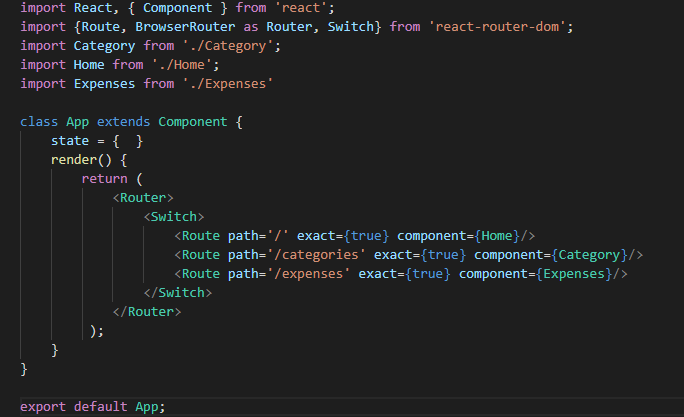
Za kraj, da bi aplikacija radila potrebno je da kliknemo desnim klikom na projekat i idemo na **Run As – Spring Boot App.**

Kada smo uradili i završili back-end naše aplikacije, prelazimo na front-end deo koji će biti prikazan našim krajnjim korisnicima. Front-end ove aplikacije je urađen u **React**-u u **Visual Studio Code**-u.

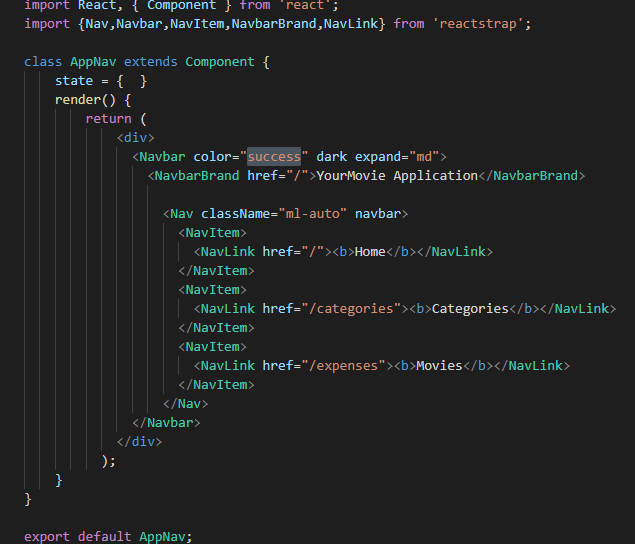
Prvo je potrebno da u Visual Studio Code-u instaliramo sledeće: **Simple React Snippets** i **Prettier React**. Kada smo to uradili, otvaramo terminal i prvo pravimo projekat tako što upišemo komandu **npx create-react-app sample** (**sample** je naziv React projekta). Struktura ovog React projekta data je u nastavku:



Kod fajla **App.js** prikazan je u nastavku:



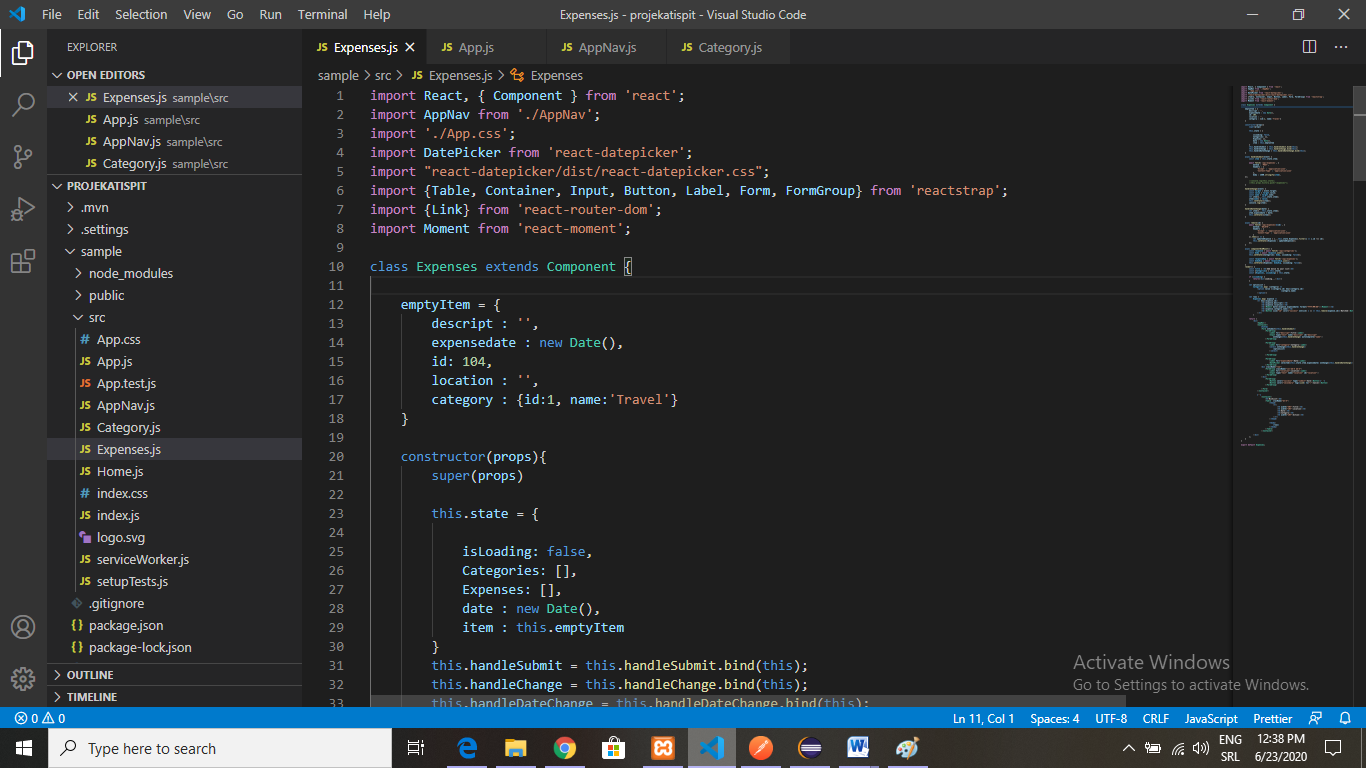
**AppNav.js** fajl u kom definišemo navigacioni meni naše aplikacije:

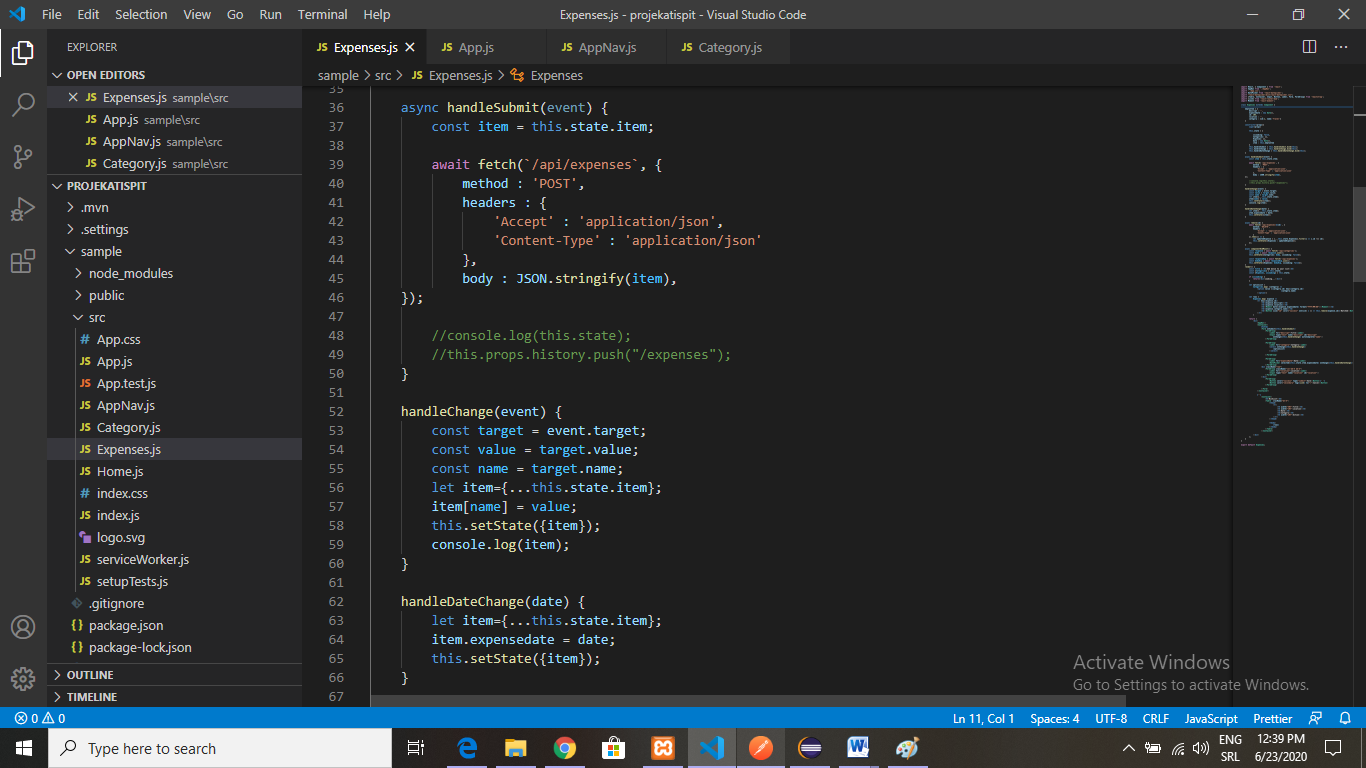


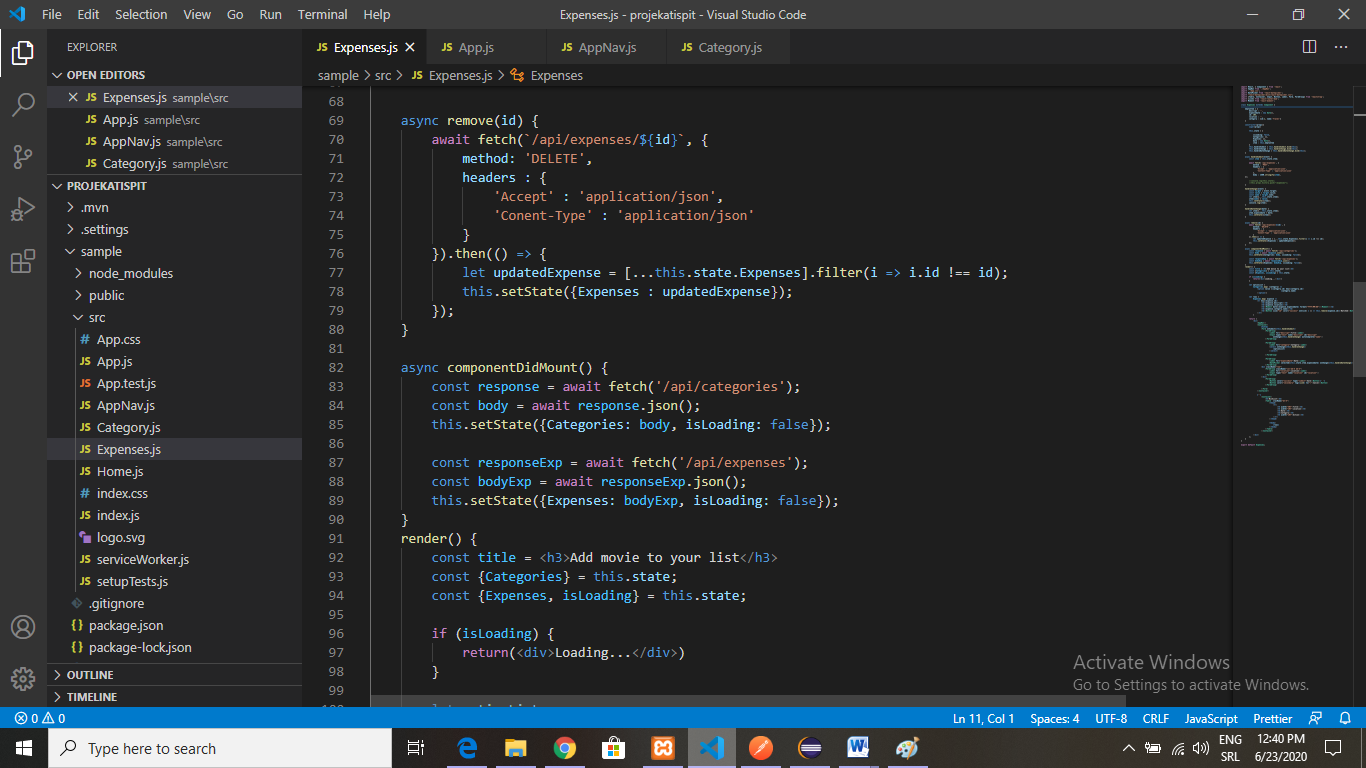
**Category**.**js** fajl:

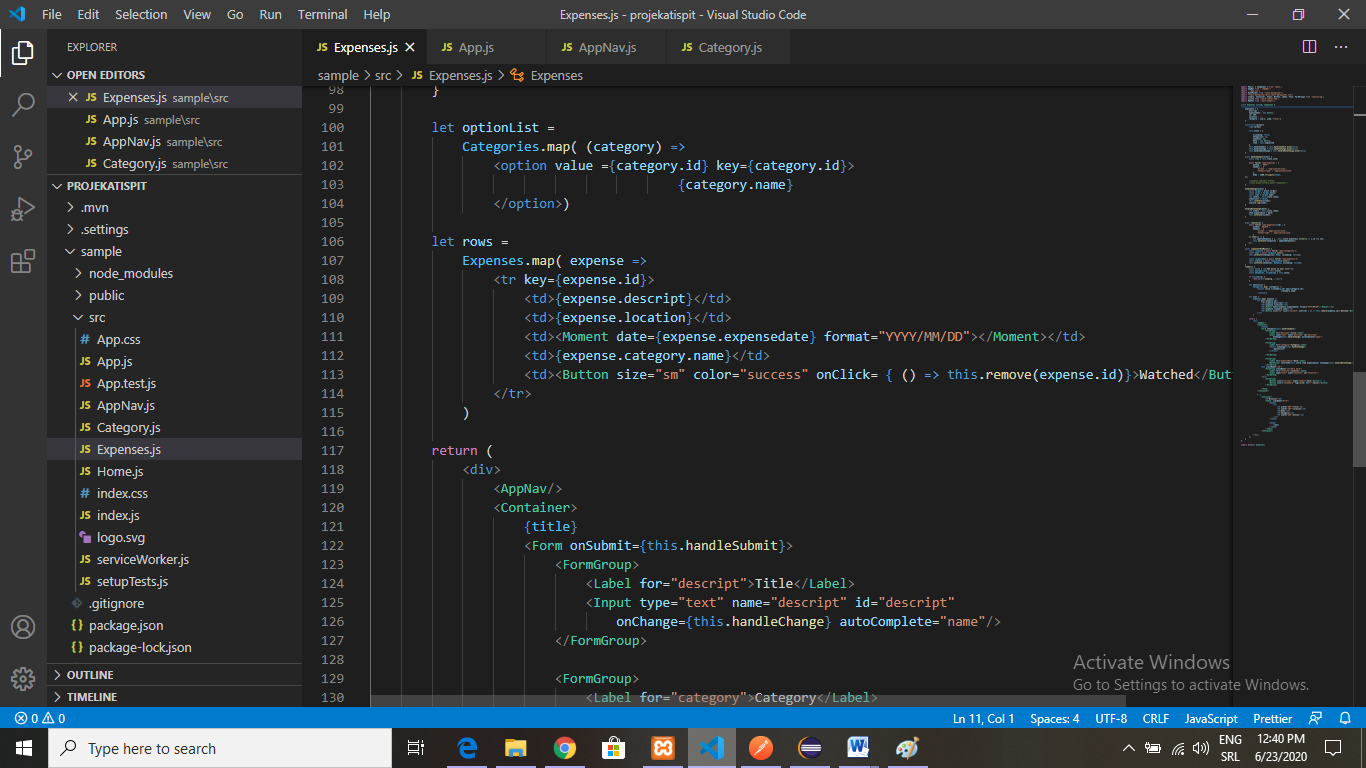


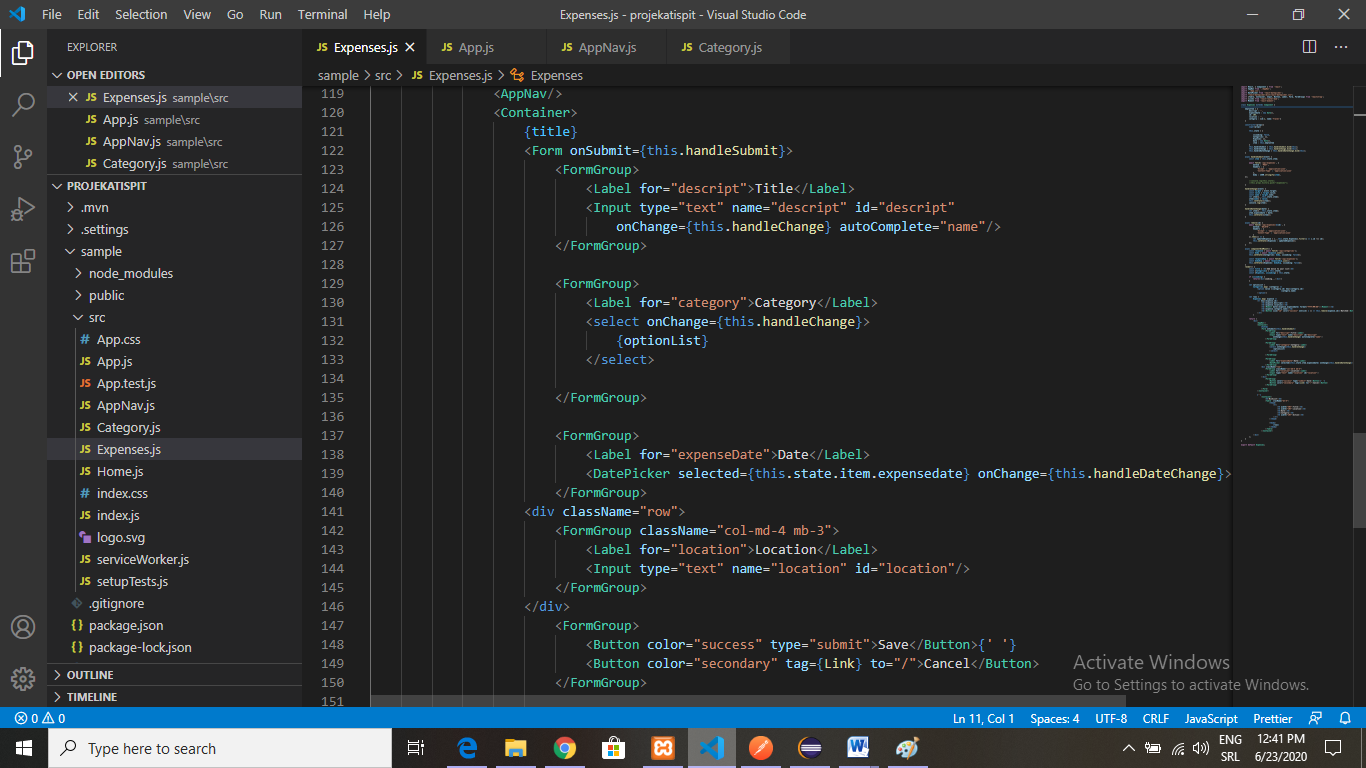
Fajl **Expenses.js** je prikazan u nekoliko sledećih slika:

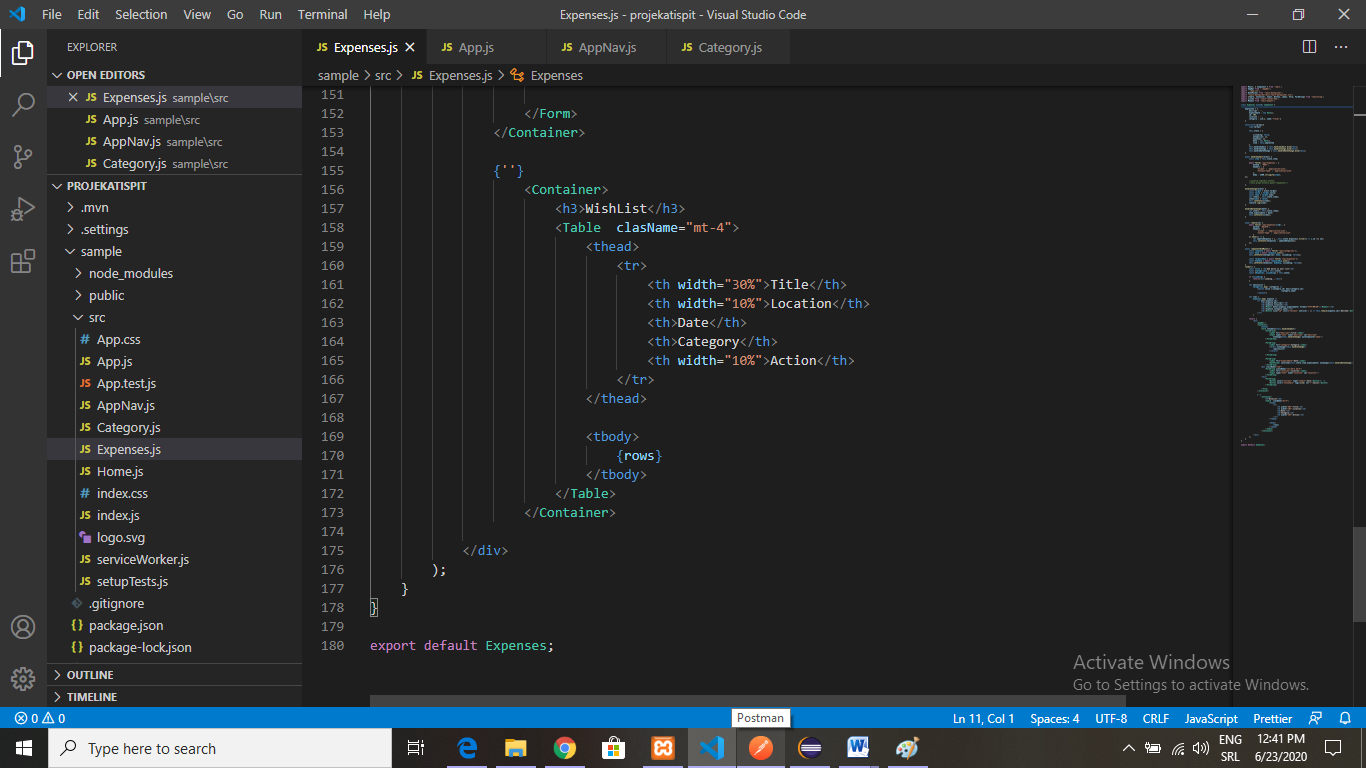




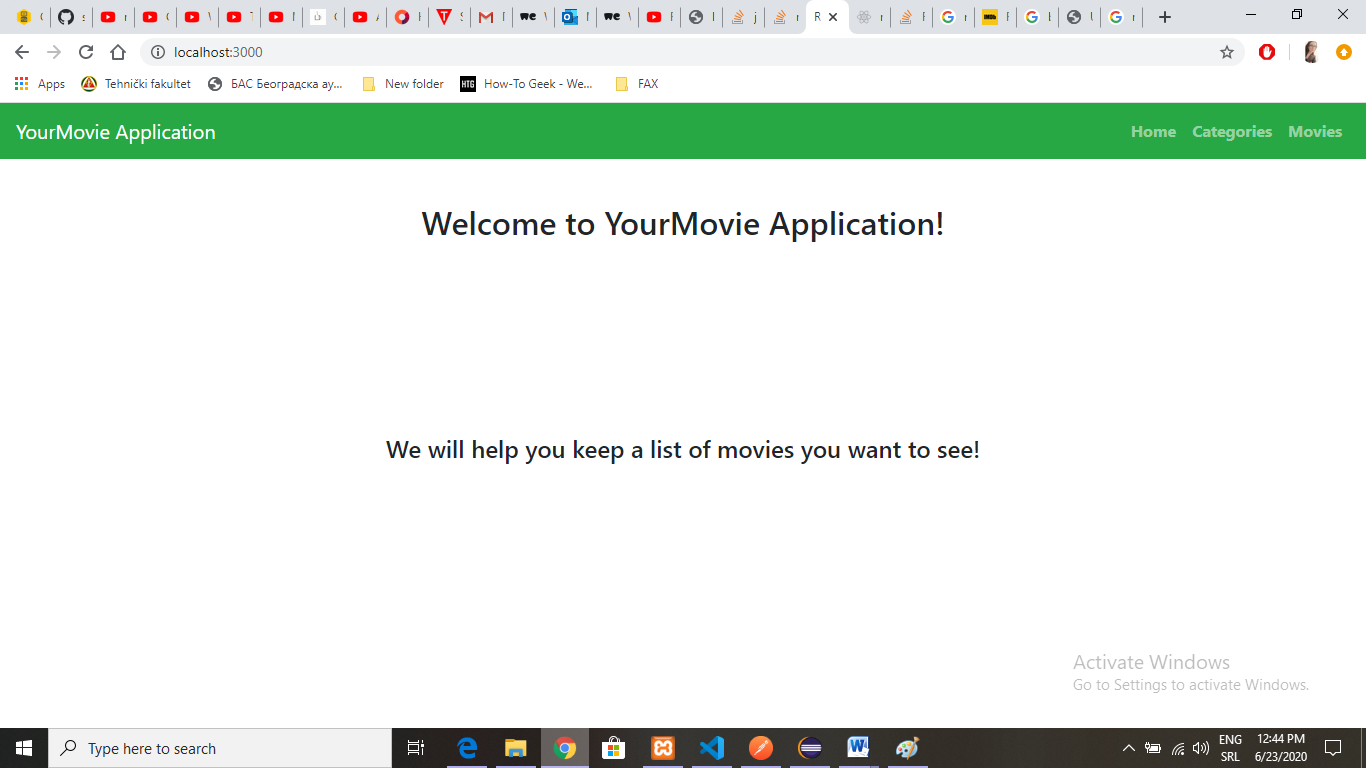


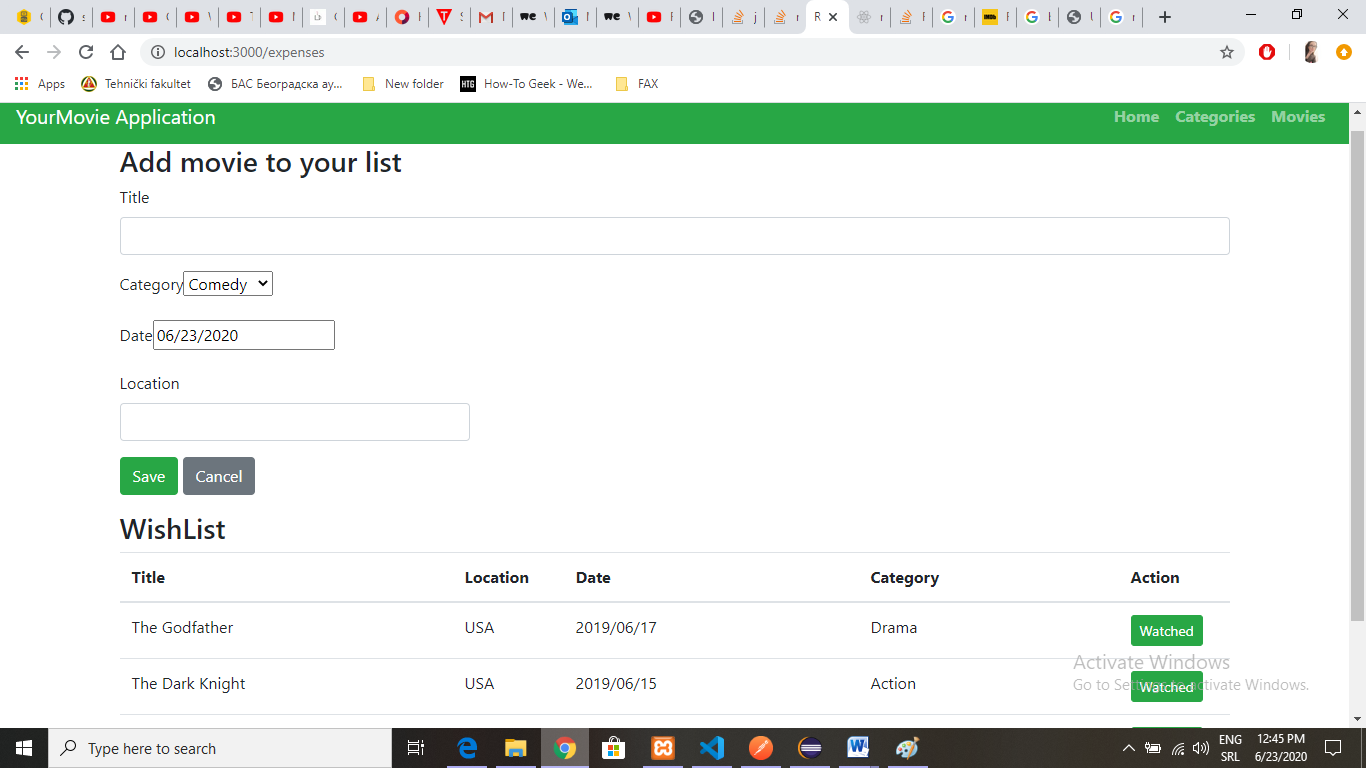






Da bismo pokrenuli React aplikaciju, potrebno je da u terminalu upišemo sledeću komandu: **npm netstat.** Aplikacija se otvara u browser-u na URL adresi: [**http://localhost:3000**](http://localhost:3000). Konačan izgled ove aplikacije koji vide krajni korisnici prikazan je u nastavku.





**Zaključak**

**REST** (Representational StateTransfer) se koristi za izgradnju lakih veb servisa u konekstu korišćenja računarskih resursa, lako održivih i skalabilnih. Koristi isključivo HTTP protokol za komunikaciju. Za reprezentaciju podataka koristi XML, JSON ili neki drugi format. REST definiše način na koji klijenti mogu da pristupe da pristupe resursima. Najbitnija odlika web servisa je omogućavanje komunikacije izmedju heterogenih aplikacija, svaka aplikacija radi na uređajima u smislu pemetni telefona,televizora takodje je prilagodjena za izvršavanje na Klaud okruženju. Ovaj standard koriste popularni web poratli kao sto su Facebook, Twitter. Ova aplikacija predstavlja osnove prikaza rada restful web servisa sa klijent server arhitekturom. Način funkcionisanja ovakvog sistema je sledeći - Server hostuje RESTIfuUI veb servis koji obezbedjuje potrebnu funkcionalnost klijentu. Klijent šalje zahtev veb servisu koji je implementiran na serveru, kada servis ili odbija zahtev,ili ga izvršava i generiše odgovarajući odgovor koji klijentu šalje nazad.