Univerzitet u Beogradu

Fakultet organizacionih nauka

Katedra za elektronsko poslovanje

Testiranje klijentske veb aplikacije

Domaći zadatak 3

|  |  |  |
| --- | --- | --- |
| Ime | Prezime | Broj indeksa |
| Filip | Popovic | 2022/0538 |
| Mihailo | Radulovic | 2022/0542 |
| Petar | Ristic | 2022/0552 |
| Link ka GitHub-u | <https://github.com/elab-development/klijentske-veb-tehnologije-2024-2022-0542-interaktivni-kalendar> | |

* Tema seminarskog rada

Tema je interaktivni kalendar. Korisniku je omoguceno da na ovoj stranici unosi svoje buduce planove uz opciju da ih edituje ili brise. Planove moze unositi na danasnji dan ili u buducnosti.

Korisnik se takodje moze ulogovagti na ovu stranicu kao i da se registruje ako vec nema napravljeni nalog. Ako pored toga ima bilo koju sugestiju ili pitanje korisnik nam moze poslati u delu koji je predvidjen za to.

* Biblioteke i paketi

Za pocetak je potrebno instalirati sve potrebne plakete za vitest i @testing-library/jest-dom. Komandom koja se nalazi na slici ispod instaliracemo sve biblioteke za testiranje u React aplikaciji.



Objasnjenje koda:

1.npm install

Ovo je onsovna komanda koje se koristi u Node.js ekosistemu za instalaciju paketa (biblioteka) preko Node Package Manager-a(NPM). Kada se pokrene ova komanda, NPM preuzima navedene biblioteke iz svog registra i dodaje ih u projekat.

2. –save-dev

Ova opcija govori i NPM-u da instalira pakete kao razvojne zavisnosti (dev dependecies), a ne kao regularne zavisnosti. Ove zavisnosti su potrebne samo za razvoj i testiranje aplikacije, a ne za njeno pokretanje u produkcionom okruzenju.

3.Instalirani paketi

a.vitest

vitest je brza, moderna i lagana biblioteka za testiranje. Slicno kao jest, koristi se za pisanje i pokretanje testova unutrar JavaScript/TypeScript aplikacija,ukljucujuci React aplikacije. Vitest je optimizovan za brzinu i dobro radi sa ES modulima.

Upotreba: Koristi se za definisanje testova, pokretanje testova i generisanje izvestaja o testiranju.

b. @testing-library/react

@testing-library/react je biblioteka za testiranje React komponenti. Ona omogucava pisanje testova koji se fokusiraju na ponasanje i interakcije komponenti, simulirajuci kako bi korisnici koristili aplikaciju.

Upotreba: Koristi se za testiranje kako React komponente rederuju i reaguje na dogadjaje (klikove, promene unosa itd.) unutrar virtuelnog DOM-a

c. @testing-library/jest-dom

@testing-library/jest-dom je dodatak za jest (ili vitest) koji prosiruje njegove aseritrivne metode za bolju proveru DOM elemenata. Dodaje prilagodjene matchers metode koje omogucavaju lakse i citljivije testiranje React komponenti i njihovih elemenata.

Upotreba:Omogucava pisanje koji proveravaju stvari poput:da li neki element ima odredjeni tekst, atribut ili klasu(npr. Expect(element).toHaveTextContent('Hello')).

Kada pokrenemo ovu komandu:

Vitest ce omoguciti da pisemo i pokrecemo testove,

@testing-library/react omoguvava testiranje React komponenti,

@testing-library/jest-dom dodaje korisne metode za proveru DOM elemenata u testovima

Sve ove biblioteke ce biti instalirane kao razvojne u zavisnosti u nasem projektu

Takdoje je potrebno da napravimo datoteku vitest.config.js, ukoliko je prethodno nismo imali.Ona treba da izgleda ovako:



Za jest,obicno se instaliraju sledeci paketi:

Jest:Glavni paket za testiranje. Obezbedjuje osnovnu funkcionalnost za izvodjenje testova,kao i alatke za testiranje JavaScript koda.

Npm install –save-dev jest

**@testing-library/react**: Biblioteka koja olaksava testiranje React komponenti. Pruza metode kao sto su render, fireEvent, i screen za manipulaciju i ispitivanje React komponenti.

Npm install –save-dev @testing-library/react

**@testing-library/jest-dom**: Ekstenzija za Jest koja dodaje korisne DOM asertacije za testiranje u @testing-library/react. Omogucava upotrebu metoda kao sto su toBelnTheDocument(), toHaveTextContent(),itd. Npm install –save-dev @testing-library/jest-dom

**Babel-jest**:Paket koji omogucava Jest-u da koristi Babel za transformaciju JavaScript koda pre nego sto se izvrsi testiranje.Ovo je posebno korisno za testiranje modernog JavaScript-a (ES6+) I JSX.

Npm install –save-dev babel-jest

**Babel-core i babel-preset-env**:Ovi paketi omogucavaju u Babel-u transpaljuje JavaScript kod prema potrebama.

Npm install - -save-dev @babel/core @babel/preset-env

**Babel-preset-react**: Ovaj omogucava Babel-u da transpaljuje JSX kod.

Npm-install –save-dev @babel/preset-react



Ovo je konfiguracija za Jest,alat za testiranje u JavaScript-u.Evo sta svaka sekcija radi:

**Transform**

'^.+\\.(js|jsx)$': 'babel-jest'

Konfigurise Jest da koristi babel-jest za transformaciju JavaScript i JSX fajlova pre testiranja.Ovo omogucava Jest-u da razume i testira moderne JavaScript funkcionalnosti.

**transformIgnorePatterns**

“/node\_modules/(?!reactjs-popup).+\\.js$“

Ova opcija kaze Jest-u da ignorise transformaciju svih JavaScript fajlova u node\_modules osim onih paketa react.js-popup.Ovo moze biti korisno ako neki paketi koriste specificne sintakse koje ne treba transformisati.

**testEnviroment**

“jsdom“

Postavlja testno okruzenje na jsdom,sto omogucava simulaciju DOM-a u Node.js okruzenju. Ovo je korisno za testiranje React komponenti koje koristi DOM.

**moduleNameMapper**

'[\\.(css|less)$](file://\\.(css|less)$\)': 'roodDir>/\_\_mocks\_\_/styleMock.js'

Uskladjuje importovanje stilskih fajlova(.css,.less) u testovima sa styleMock.js fajlom.Ovo je potrebno jer testiranje stilove moze uzrokovati greske,pa se cesto koriste “mock“ fajlovi za stilove.

Ova konfiguracija omogucava Jest-u da pravilno transformise kod,upravlja stilovima i pruza okruzenje za testiranje koje simulira rad sa DOM-om.

* Prikaz i objašnjenje kreiranih testova
* Test jedne funkcionalnosti

U ovom testu proverava se osnovna funkcionalnost jednof dela aplikacije.To moze biti funckija ili metoda koja vrsi odredjene operacije(npr.sortiranje,racunanje,validacija podataka).

Unutar komponente CalendarApp definisana je funckija sortByDate,koja prima dogadjaja i sortira ih prema datumu.Funckija koristi Array.sort() metodu kako bi poredila datume svakog dogadjaja.



Test za funkcionalnosti sortByDate osigurava da su dogadjaji pravilno sortirani.U ovom testu kreiran je niz dogadjaja s razlicitim datumima, a zatim je pozvana funkcija sortByDate kako bi se proverilo da su dogadjaji pravilno sortirani



Objasnjenje testa

Kreiranje dogadjaja:Definisana su tri dogadjaja sa razlicirim datumima(2024-07-18,2024-08-18,2024-09-18)

Poziv funkcije:Funkcija sortByDate se poziva sa ovim nizom dogadjaja.

Ocekivanja (Assertions):Koristeci expert,proverava se da li je prvi dogadjaj u sortiranom nizu dogadjaj s najranijim datumom,zatim sledeci,i tako dalje.Ocekuje se da ce dogadjaji biti sortirani u rastucem redosledu datuma.

Test uspesno prolazi ako su dogadjaji pravilno sortirani.

Ovaj test je kljucan za osiguravanje ispravnosti logike sortiranja dogadjaja prema datumu,sto je vazna funkcionalnost unutar aplikacije CalendarApp.Test proverava da funkcija radi korektno,bez obzira na redosled u kojem se dogadjaji inicijalno nalaze u nizu.

* Test komponente

Komponenta CalendarApp je srz aplikacije i prikazuje interfejs za upravljanje dogadjajima u kalendaru.

Cilj ovog testa je da proveri da se komponenta CalendarApp pravilno renderuje i da sadrzi odredjeni tekst koji se ocekuje na ekranu prilikom njenog renderovanja.



Objasnjenje testa

**Importi**

Import React from 'react';

Uvozi Reacr biblioteku koja je potrebna za rad sa JSX-om(iako je ponekad automatski ukljucena)

Import CalendarApp from'../Components/CalendarApp';

Uvozi komponentu CalendarApp iz datoteke na odredjenoj putanji

Import{render,screen} from '@testing-library/react';

Uvozi funkcije render(za renderovanje komponente) i screen(za pretragu elemenata u DOM-u).

Import '@testing-library/jest-dom';

Uvozi dodatne matcher-e za Jest koji omogucavaju bolje asertacije kao sto je toBelnTheDocument().

**Strukturiranje Testa**

Describe('CalendarApp Component',() = >{...}};

Grupisnje testova za komponentu CalendarApp.

It('should render the CalendarApp component',() = >{...}};

Testira da li se komponenta CalendarApp ispravno renderuje.

Redner(<CalendarApp/>);

Renderuje komponentu u virtuelni DOM za testiranje.

Expect(screen.getByText('Calendar')).toBelnTheDocument().

Proverava da li se tekst “Calendar“ nalazi u dokumentu.

Ovaj test proverava da li komponenta CalendarApp pravilno prikazuje tekst “Calendar“.

* Test komponente koja zahteva *mock* funkciju

Na log in stranici imamo mesto gde mozemo nas mail i sifru,dugme koje nas vodi na prethodnu stranicu,dugme kojim se loginujemo,linkove do drugih poznatih mreza,kao i link za registrovanje ako nemamo nas nalog.Klikom na link “Sign up“ bicemo preusmereni na sledeci stranicu.







**Uvoz paketa**

React:Importuje React kako bi se mogao koristiti u JSX sintaksi.

Render i fireEvent:Funkcije iz @testing-library/react koje omogucavaju renderovanje komponente u virtuelnom DOM-u i simulaciju korisnickih dogadjaja(poput klika i promene).

MemoryRouter:Komponenta iz react-router-dom koja omogucava simulaciju rutiranja u testovma.

LoginPage:Komponenta koju testiramo,u ovom slucaju stranica za prijavu.

useNavigate:Hook iz react-router-dom za navigaciju izmedju stranica.

**Mockovanje useNavigate**

Jest.fn():Kreira mock funkciju koja ce pratiti pozive,argumente i broj poziva.

Jest.mock('react-router-dom'):Mock-uje modul react-router-dom,a funkcija useNavigate vrcaca mockNavigate.Ovo omogucava da se testira kako se useNavigate funkcionise bez stvarnog navigacionog okvira.

**Test za prikazivanje upozorenja sa pogresnim kredencijalima**

Global.alert = jest.fn():Mock-uje globalnu alert funkciju kako bismo mogli da proverimo da li je pozvana sa ispravnim porukama.

Render:Renderuje LoginPage unutrar MemoryRouter koji emulira okruzenje rute za testiranje.

fireEvent.change i fireEvent.click:Simuliraju unos u polja i klik na dugme.

Expect(global.alert).toHaveBeenCalledWith:Proverava da li je alert funkcija pozvana sa ocekivanom porukom.

**Test za navigaciju na stranciu profila sa ispravim kredencijalina**

Mock.Navigate.mockClear():Ocisti prethodne pozive na mockNavigate kako bi se osiguralo da test ne bude pogodjen na prethodnim pozivima.

fireEvent.change i fireEvent.click:Simuliraju unos ispravnih kredencijala i klik na dugme za prijavu.

Expectc(mockNavigate).toHaveBeenCalledWith:Proverava da li je mockNavigate pozvan sa ocekivanim putem (/profil),sto bi znacilo da je navigacija uspesno izvrsena.

Ovaj kod testiea LoginPage komponentu kako bi se uverilo da pravilno reaguje na pogresne i ispravne kredencijalne.Mock-uje useNavigate da proveri navigaciju bez stvarnog preusmeravanja,i mock-uje globalnu alert funckiju da proveri da li se prikazuje ispravna poruka u slucaju pogresnh kredencijala.

* Test modela
* Event test

Ovaj kod testira razlicite aspekte funkcionalnosti klase Event,ukljucujuci:

Inicijalizaciju sa ispravnim vrednostima

Formatiranje detalja.

Rukovanje praznim i nedostajucim vrednostima.

Rukovanje razlicitim tipovima podataka.

Testovi su napisani kako bi se osiguralo da klasa Event pravilno funkcionise u razlicitim uslovima.





**Uvoz i Opis**

Import Event from '../Models/Event';:Uvozi klasu Event iz modula koji se nalazi u ../Models/Event.

Describe ('Event',() =>{:Definise grupu testova koji su povezani sa klasom Event.

**Priprema za testove**

Let event;:Deklarise varijablu event koja ce se koristiti u testovima.

beforeEach(() => {:Pre svakog testa,kreira novu instancu klase Event sa zadatim vrednostima.

**Testovi**

Test 1:Provera inicijalizacije

Test('should create an event with name,date and description',() =>{:Testira da li objekat ima ispravne vrednosti za ime,datum i opis,

Test 2:Provera Formatiranja Detalja

Test('should return event details in the correct format',() = >{:Testira da li metoda getEventDetails vraca detalje dogadjaja u pravilnom formatu.

Test 3:Rukovanje Praznim Imenom i Opisom

Test('should handle empty name and description',() =>{:Testira da li klasa pravilno obradjuje prazne stringove za ime i opis.

Test 4:Rukovanje Nedostajacim Datumom

Test('should handle missing date',() = >{:Testira da li klasa pravilno obradjuje slucaj kada datum nije naveden.

Test 5:Rukovanje Razlicitim Tiopvima Podataka

Test('should handle different types of data',()=>{:Testira da li klasa moze da rukuje razlicitim tipovima podataka za ime,datum i opis,i proverava da li ispravno formatuju detalji.

* User test

Ovaj kod testira razlicite aspekte funkcionalnosti klase User,ukljucujuci:

Inicijalizaciju sa ispravnim vrednostima.

Formatiranje informacija o korisniku.

Rukovanje praznim vrednostima.

Rukovanje razlicitim tipovima podataka.

Testovi su napisani kako bi se osiguralo da klasa User pravilno funkcionise u razlicitim uslovima.





**Uvoz i opis**

Import User from '../Models/User';:Uvozi klasu User iz modula koji se nalazi u ../Models/User.

Desribe('User',() => {:Definise grupu testova klase User.

**Priprema za Testove**

Let user;:Deklarise varijablu user koja ce se koristiti u testovima.

beforeEach(() =>{:Pre svakog testa,kreira novu instancu klase User sa zadatim vrednostima.

**Testovi**

Test 1:Provera inicijalizaije

Test('should create a user with username,password,and email',() =>{:Testira da li objekat user ima ispravne vrednosti za korisnicko ime,lozinku i e-mail.

Test 2:Provera Formatiranja Informacija

Test('should return user info in correct format',() =>{:Testira da li metofa getUserInfo vraca informacije o korisniku u pravilnom formatu.

Test 3:Rukovanje Praznim Imenom i E-mailom

Test('should handle empty username and email',() =>{:Testira da li klasa pravilno obradjuje prazne stringove za korisnicko ime,lozinku i e-mail.

Test 4:Rukovanje Razlicitim Tipovima Podataka

Test('should handle different types of data',() =>{:Testira da li klasa moze da rukuje razlicitim tipovima podataka za korisnicko ime,lozinku i e-mail,i proveravaa da li se ispravno formatiraju informacije o koristniku.

* Prikaz rezultata testiranja

Za prikaz svih testova koristicemo komandu npm test u terminalu



Da bismo prikazali svaki test pojedinacno potrebno je da u terminalu pozovemo komandu npm test --<putanja-do-test-fajla>.











Za detaljnije rezultate koristimo komadu npm test -- --verbose.

