JUNIORDev DIGITALNA DALMACIJA

Finalna verzija aplikacije

Jednom kada smo (barem privremeno) dovršili razvoj React aplikacije, potrebno ju je pretvoriti u oblik prikladan za objavu. Gotovo sve promjene na aplikaciji smo radili u direktoriju "***src***" (skraćeno od *source*) u kojem imamo sav naš kôd podijeljen u višestruke direktorije i datoteke.

Takva struktura nam olakšava razvoj i upravljanje cijelom aplikacijom jer su komponente smisleno imenovane i cijela aplikacija je posložena u logičke cjeline (ako smo uredno strukturirali projekt). S druge strane, takva struktura nije praktična za iz perspektive poslužitelja na kojem će se eventualno naša aplikacija nalaziti. Osim toga, čitav kôd je napisan na način da je razumljiv ljudima, što u praksi znači da imamo nepotrebno (iz konteksta računala) duga imena varijabli i funkcija, komentare, razmake i slično. Na manjim aplikacijama koje se rijetko otvaraju to možda i nije toliki problem ali kod većih projekata i stranica kod kojih poslužitelj mora obraditi veliki broj zahtijeva, veličina datoteka postaje važan faktor. Također, već smo spomenuli da se JXS sintaksa koju koristimo na kraju prevodi u "obični" JavaScript i da postoje posebni alati za tu namjenu.

Uzevši sve u obzir, očito je se aplikacija u jednom trenutku mora prilagoditi i optimizirati za pokretanje na poslužitelju. Taj postupak se najčešće zove ***build*** proces, a samo pokretanje aplikacije se naziva ***deployment***.

Izgradnja aplikacije

Proces izgradnje aplikacije uključuje posebne alate koje je potrebno konfigurirati i prilagoditi za konkretnu aplikaciju koju izgrađujemo. Srećom, većina okvira za React sadrži unaprijed konfigurirane alate pa možemo probati proces izgradnje bez ulaženja u detalje vezane uz same alate.

Za primjer ćemo uzeti jednu od aplikacija koje smo radili u sklopu prethodnih radionica, i usput se upoznati sa još jednom *git* naredbom. Ponekad (kao u ovom primjeru) ne počinjemo sa praznim repozitorijem već želimo nastaviti raditi na nekom već postojećem projektu, a nemamo odgovarajući lokalni repozitorij na našem računalu. U takvim situacijama koristimo naredbu `git clone` koja služi za kloniranje nekog već postojećeg repozitorija.

Otvorite terminal u nekoj praznoj mapi tj. na lokaciji gdje želite klonirati postojeći repozitorij. Pri tome samo pripazite da lokacija terminala nije u direktoriju koji već sadrži *git* repozitorij (pod-direktoriji ih mogu imati). Nakon toga upišite naredbu:

```
git clone https://github.com/gzaharija-pmfst/jdev-primjer.git
```

Na ovoj adresi se nalazi primjer aplikacije koja dohvaća i ispisuje imena astronauta koji se trenutno nalaze u svemiru. Nakon što se završio proces kloniranja potrebno je odraditi postupak koji radimo i prilikom stvaranja nove aplikacije - pozicionirati se u novi direktorij, instalirati sve potrebne *dependency* biblioteke te pokrenuti lokalni razvojni poslužitelj.

```
cd jdev-primjer
npm install
npm run dev
```

Otvorite aplikaciju u VS Code i po želji modificirajte aplikaciju. Također, ako otvorite novi terminal i pogledate popis udaljenih repozitorija uočiti će da je automatski postavljen `origin` repozitorij na adresu sa koje ste ga upravo klonirali. Ovisno o postavkama repozitorija možete ili ne morate imati pravo slanja (*push*) promjena na originalni repozitorij ali zato možete klonirani repozitorij povezati sa nekim drugim udaljenim repozitorijem kod kojega imate pravo uređivanja.

Vratimo se na proces izgradnje aplikacije. Vite okruženje koje smo koristili kao predložak za nove aplikacije sadrži ugrađenu skriptu za pokretanje *build* procesa. Sve što je potrebno je u terminalu (naravno, unutar direktorija projekta) upisati naredbu:

```
npm run build
```

Nakon završetka procesa izgradnje možemo pogledati koje su se promjene dogodile unutar našeg projekta. S obzirom da u sklopu projekta imamo *git* repozitorij, naredba `git status` bi nam trebala ispisati sve promjene u odnosu na posljednji *commit* ali ako pokušate izgledati će kao da nema razlika.

Datoteka .gitignore

Ponovno ćemo se kratko vratiti na *git*. Prilikom stvaranja novog repozitorija spomenuli smo kako se stvara i datoteka ***.gitignore*** ali se nismo posebno osvrtali na nju. Ova datoteka sadrži popis direktorija i datoteka za koje **NE** želimo da budu praćene od strane repozitorija niti uključene u *commit*, čak ni ako napišemo naredbu `git add .` koja dodaje sve datoteke iz glavnog direktorija u repozitorij.

Razlog ovome je što neke dijelove projekta nije potrebno (niti poželjno) uključivati u repozitorij. Jedan od takvih primjera je direktorij ***node_modules*** koji se nalazi na popisu u "*.gitignore*" datoteci. Direktorij *node_modules* sadrži sve potrebne *dependency* biblioteke potrebne za pokretanje projekta ali radi se o bibliotekama koje se nalaze u sklopu javnog repozitorija (***Node Package Manager*** - *npm*). Zbog toga nema potrebe da svaki projekt sadrži vlastitu kopiju tih biblioteka kad ih jednostavno možemo preuzeti u bilo kojem trenutku sa naredbom `npm install` koju smo koristili svaki puta pri stvaranju novog projekta. Naravno, svaki projekt mora sadržavati popis biblioteka koje su mu potrebne za izvršavanje, a to možete pronaći u datoteci ***package.json*** koja se nalazi u sklopu projekta.

Build verzija

Vratimo se na proces izgradnje aplikacije. Ako bolje promotrimo direktorij projekta, vidjeti ćemo da ipak postoji razlika. Nakon pokretanja *build* skripte, u sklopu projekta se stvorio ***dist*** direktorij koji sadrži verziju aplikacije optimiziranu za objavu na poslužitelju.

Promotrimo li sadržaj *dist* direktorija, vidjeti ćemo da sadrži samo jednu HTML, jednu JS i jednu

CSS datoteku (sadrži i sliku koja je dio početnog predloška a koju nismo izbrisali). Ako otvorite CSS ili JS datoteku, vidjeti ćete da se radi o minimiziranim verzijama koje su napisane kako bi zauzimale što manje memorije (tj. imale što manju veličinu datoteka.)

Sve odvojene datoteke i komponente koje smo imali unutar projekta su skupljene u jednu JS datoteku kako bi se olakšao posao poslužitelju i smanjila količina prometa između klijenta i poslužitelja. Kod većih aplikacija će *build* verzija imati nešto više datoteka ali opet značajno manje od njene razvojne verzije.

Pokretanje build verzije

Vite omogućava i jednostavno pokretanje *build* verzije aplikacije, korištenjem naredbe:

```
npm run preview
```

Ovo će pokretni lokalni poslužitelj koji simulira klasične web poslužitelje i pomoću kojeg možemo testirati kako će se aplikacija "ponašati" jednom kada bude objavljena. Za razliku od svih dosadašnjih slučajeva, ovo nije razvojni React poslužitelj i ovaj poslužitelj ne prevodi naš izvorni JSX kôd u gotovu aplikaciju već samo poslužuje "index.html" datoteku koja se nalazi unutar *dist* direktorija, što naravno možemo vrlo jednostavno testirati ukoliko napravimo neku promjenu u izvornom kôdu aplikacije.

Bilo kakva promjena u izvornom kôdu će se odmah pokazati u poslužitelju kojeg smo pokrenuli sa naredbom `npm run dev` ali neće biti vidljiva na poslužitelju koji je pokrenut sa `npm run preview`. Ukoliko nam je to cilj, potrebno je ponovno pokrenuti skriptu za izradu *build* verzije.

Ukoliko imate na raspolaganju neki vlastiti poslužitelj na web-u, možete jednostavno kopirati *dist* direktorij i imati ćete potpuno funkcionalnu aplikaciju. Alternativno, možete koristiti neki od *online* alata za posluživanje statičkih (samo *frontend*) web aplikacija kao što su GitHub pages, Vercel, i Netlify ili možete (uz plaćenje) pokrenuti svoj virtualni poslužitelj (VPS) koristeći *online* platforme kao što su Digital Ocean, Linode ili Vultr

Last updated on April 11, 2023

Udaljeni repozitorij

Praktični dio