

Praktični ispit iz Programiranja za veb – Primer ispita

18. maj 2020.

Ispit se polaže najviše 180 minuta. Broj poena po zadacima je:

Redni broj zadatka	1	2	3	4	5	6	7	8	9	10	11	12	Suma
Najviše poena	20	15	5	5	5	5	5	5	5	5	5	5	70*

*Poeni za teoriju se skaliraju sa ukupnih 50 na 35.

Zadaci

1. Napisati Node.js aplikaciju koja ispunjava naredne zahteve:

- U MongoDB bazi podataka *pueb_primer_ispita* nalaze se dve kolekcije:
 - Kolekcija *goals* sadrži informacije o zadacima koje treba ispuniti: jedinstveni identifikator zadatka, kratak naziv zadatka, opis zadatka i njegovu važnost. Važnost zadatka je numerička vrednost od 1 do 3 (1 označava vrlo važan zadatak, 2 umereno važan zadatak, a 3 malo važan zadatak).
 - Kolekcija *steps* sadrži pojedinačne korake vezane za realizaciju ovih zadataka: jedinstveni identifikator koraka, jedinstveni identifikator koraka, redni broj koraka i njegov opis.

Kreirati sheme korišćenjem Mongoose biblioteke na osnovu informacija iz datih kolekcija. Na osnovu ovih shema kreirati modele *Goal* i *Step*.

- Korišćenjem Express i Mongoose biblioteka implementirati REST API koji je opisan u narednoj tabeli:

Metod	Parametri	Značenje	Rezultat
GET	/goals	Dohvatanje svih zadataka.	JSON reprezentacija kolekcije zadataka.
GET	/goals/{id}	Dohvatanje pojedinačnog zadatka.	JSON reprezentacija jednog zadatka.
POST	/goals	Dodavanje novog zadatka.	JSON opis uspešnosti unosa.

2. Napisati Angular aplikaciju koja ispunjava naredne zahteve:

- Na početnoj strani aplikacije *http://localhost:4200/* dohvatiti informacije o zadacima iz BP slanjem GET zahteva serverskoj aplikaciji. Prikazuju se imena svih zadataka uređenih po važnosti (slika 1). Imena zadataka su prikazani na karticama (*<h*>* ili *<div>* elementima) čija je boja pozadine (*background-color*) prilagođena važnosti zadatka. Crvenom bojom se prikazuju važni zadaci, narandžastom bojom umereno važni zadaci, a malo važni zadaci žutom bojom (slika 1).
- Klikom na neku karticu koja predstavlja jedan zadatak, dohvatiti informacije o tom zadatku iz BP slanjem GET zahteva serverskoj aplikaciji. Ove informacije se prikazuju u sakrivenom elementu ispod te kartice.
- Iznad ove liste zadataka, nalazi se serija opcija za filtriranje. Izbor važnosti omogućava prikaz zadataka iz odabrane kategorije. Izbor složenosti omogućava prikaz zadataka po složenosti: zadatak je jednostavan ako ima samo jedan korak, umereno složen ako ima dva ili tri koraka, a težak ako ima 4 ili više koraka.
- Klikom na “Add goal”, otvara se stranica *http://localhost:4200/add-goal* na kojoj se prikazuje panel za dodavanje novog zadatka (slika 2). Korisnik mora popuniti polja za ime, opis i važnost zadatka, kao i bar jedan od koraka. Naslov zadatka treba da bude maksimalne dužine 50 karaktera, a opis dužine od 10 do 100 karaktera. Omogućiti validaciju ovog formulara putem reaktivnog pristupa. Omogućiti prikaz poruka o greškama u slučaju pogrešnih unosa.
- Klikom na “Save” unete podatke treba trajno sačuvati u bazi slanjem POST zahteva serverskoj aplikaciji. U telu zahteva smestiti unete podatke iz formulara.
- Klikom na “Close” vratiti se na početnu stranicu.

Slika 1: Prikaz svih ciljeva.

Slika 2: Formular za unos novog cilja.

Teorijska pitanja

Uputstvo: U radnom direktorijumu na *Desktop*-u nalazi se datoteka *PrezimeIme_alasNalog_grupa_teorija.txt* u kojoj je potrebno čuvati odgovore na naredna pitanja. Obavezno preimenovati datoteku da sadrži Vaše informacije. Ako je neki odgovor potrebno obrazložiti, to uraditi u ≤ 5 redova dužine 80 karaktera.

Sledeća pitanja služe samo kao demonstracija u kom formatu će pitanja na teorijskom delu ispita izgledati. Biće deset pitanja koja nose po pet (neskaliranih) poena, mada ih je u ovom primeru ispita samo pet...

3. Ako bismo pokušali da komponente veb strane uklopimo u *Model-View-Controller* arhitekturu, šta bi bio virtuelni DOM i obrazložiti zašto?

4. Da li je svaki *JavaScript* program u isto vreme i *TypeScript* program?

5. Dobili ste zadatak da implementirate mehanizam čuvanja *JavaScript* objekata. Ako nije bitna brzina izvršavanja i možete da “otaljate” implementaciju, koliko vremena bi Vam za to bilo potrebno (obrazložiti odgovor).

6. Šta je *CORS*?

7. Da li je operacija $+$ u *JavaScriptu* komutativna (obrazložiti odgovor)?

8. ...

9. ...

10. ...

11. ...

12. ...