

# Praktični ispit iz kursa Uvod u Veb i Internet Tehnologije

Grupa 2

14.06.2019.

Broj indeksa	Ime i prezime

**Obavezno se potpisati i ostaviti papir na tastaturi na kraju rada!**  
**Student koji ponese papir sa sobom automatski dobija 0 poena!**

Ispit se radi najviše 180 minuta. Maksimalan broj poena je 45. Broj poena po zahtevima je:

Modeli	Server	GET 1	POST 1	GET 2	POST 2	Bootstrap	Suma	Ukupno
15	15	30	20	35	35	2	150	45 (+2)

## Pre početka rada

U direktorijumu `Desktop` se nalazi direktorijum `uvit_materijali` koji je potrebno preimenovati u format `InicijaliAsistenta_ImePrezime_Indeks_2`. Indeks zapisati u formatu `AAAGGGG`, gde je `AAA` trocifreni broj indeksa, a `GGGG` godina izdavanja indeksa (na primer, za indeks `50/2013` jedini ispravan format je `0502013`). U direktorijumu se nalazi arhiva koju je potrebno otpakovati (desni klik na arhivu, pa odabrati opciju "Extract here"). Za otpakivanje arhive je neophodna lozinka: `sms`.

## Uputstvo za rad

Potrebno je napraviti serversku aplikaciju koja će upravljati porukama među korisnicima zajedno sa sistemom za upravljanje bazom podataka i prateću klijentsku aplikaciju pomoću koje će korisnici imati mogućnost da šalju poruke jedni drugima.

### Modeli - 5 + 10 poena

Svaki korisnik (sa odgovarajućim ključevima sheme `Korisnik` u zagradama) ima informacije o: korisničkom imenu (`korisnickoIme`, niska) i lozinki (`lozinka`, niska). Sve informacije su neophodne.

Poruke (sa odgovarajućim ključevima sheme `Poruka` u zagradama) imaju naredne informacije: identifikator korisnika koji je poslao poruku (`posiljalac`, identifikator), identifikator korisnika koji je primio poruku (`primalac`, identifikator), datum slanja poruke (`datum`, niska) i tekst poruke (`tekst`, niska). Sve informacije su neophodne.

Podatke čuvati u MongoDB sistemu za upravljanje bazom podataka, a za naziv baze podataka izabrati `201819Jun1_2`.

### Arhitektura - 5, CORS - 5, Obrada grešaka - 5 poena

Korišćenjem `Node` razvojnog okruženja, kreirati serversku aplikaciju koja opslužuje zahteve na portu 3000 i koja implementira naredne zahteve nad odgovarajućim lokacijama koje su navedene ispod. Arhitekturu aplikacije i rutiranje zahteva postaviti pomoću biblioteke `Express`, a komunikaciju sa bazom podataka izvršiti pomoću biblioteke `Mongoose`. Svaki model i ruta treba da budu u posebnim datotekama i grupisane po direktorijumima `routes` i `modules`. Ispravno implementirati CORS mehanizam zaštite. Ispravno izvršiti obradu grešaka koje mogu nastati i ispravno obavestiti klijenta o nastalim greškama.

## Opisi zahteva

U direktorijumu `klijent/` se nalaze neophodni resursi za veb prezentaciju, dok se u direktorijumu `klijent/final/` nalaze slike koje sadrže finalni prikaz stranice i njenih delova (više o tome u ostatku teksta).

## Zahtevi ka `http://localhost:3000/korisnik`

### GET 1: Serverski deo - 10, Klijentski deo - 20 poena

Klikom na dugme “Uloguj se”, prikupiti podatke iz polja formulara iznad dugmeta i poslati asinhroni **GET** zahtev. Potrebno je proslediti **samo** korisničko ime kao parametar zahteva. Oba polja moraju biti popunjena. Ukoliko neko polje nije popunjeno, prikazati korisniku obaveštajni prozor sa porukom koje polje nije popunjeno. Na serverskoj aplikaciji dohvatiti identifikator i lozinku korisnika sa zadatim imenom iz baze podataka (može se pretpostaviti da je ime jedinstveno). U telu odgovora smestiti izdvojene podatke, a status postaviti na 200. Ukoliko nije pronađen korisnik sa zadatim imenom, vratiti status 404 i prazno telo. Na klijentu je potrebno proveriti da li se uneta lozinka poklapa sa lozinkom koju vraća server. Ukoliko je lozinka ispravna, prikazati prozor sa porukom **Dobrodosli!**, a u suprotnom poruku **Pogresna lozinka!**. U slučaju ispravnog logovanja, potrebno je izbrisati sadržaj elementa sanduče i sakriti formular za novu poruku. Obraditi sve potencijalne greške na serveru i klijentu.

### POST 1: Serverski deo - 10, Klijentski deo - 10 poena

Klikom na dugme “Registruj se”, prikupiti podatke iz polja formulara iznad dugmeta i poslati asinhroni **POST** zahtev. U telu zahteva smestiti prikupljene podatke. Sva polja moraju biti popunjena. Ukoliko neko polje nije popunjeno, prikazati korisniku obaveštajni prozor sa porukom koje polje nije popunjeno. Na serverskoj aplikaciji dohvatiti informacije iz tela zahteva i upisati novi podatak u bazu podataka (u kolekciju sa korisnicima). Nakon toga, u telu odgovora treba smestiti ime kreiranog korisnika. Na klijentu je potrebno prikazati obaveštajni prozor u kojem piše **Korisnik <IME> je uspesno registrovan** kao na slici **post1.png**. Obraditi sve potencijalne greške na serveru i klijentu.

## Zahtevi ka `http://localhost:3000/poruke`

### GET 2: Serverski deo - 15, Klijentski deo - 20 poena

Klikom na dugme “Sanduče”, poslati asinhroni **GET** zahtev. Proslediti identifikator ulogovanog korisnika kroz parametar zahteva. Ukoliko nijedan korisnik nije ulogovan, prikazati prozor sa porukom **Niste ulogovani!**. Na serverskoj aplikaciji dohvatiti informacije (pošiljalac, datum i tekst poruke) iz baze podataka o svim porukama koje je primio korisnik sa zadatim identifikatorom. Rezultat sortirati po datumu opadajuće. Rezultat obogatiti podacima o pošiljaocu iz sheme **Korisnik**. U telu odgovora smestiti niz poruka. Na klijentu je potrebno prikazati podatke u formatu kao na slici **get.png** u elementu ispod dugmeta “Sanduče”. Obraditi sve potencijalne greške na serveru i klijentu.

### POST 2: Serverski deo - 20, Klijentski deo - 15 poena

Klikom na dugme “Nova poruka” efektom klizanja se prikazuje / skriva formular za slanje nove poruke. Klikom na dugme “Pošalji”, prikupiti podatke iz formulara iznad dugmeta i poslati asinhroni **POST** zahtev. U telu zahteva smestiti prikupljene podatke, kao i podatke o datumu kada je poslata poruka (trenutni datum) i identifikatoru korisnika koji šalje poruku (ulogovan korisnik). Ukoliko nijedan korisnik nije ulogovan, prikazati prozor sa porukom **Niste ulogovani!**. Sva polja moraju biti popunjena. Ukoliko neko polje nije popunjeno, prikazati korisniku obaveštajni prozor sa porukom koje polje nije popunjeno. Na serverskoj aplikaciji dohvatiti informacije iz tela zahteva. Prvo treba dohvatiti identifikator primaoca, za zadato korisničko ime. Ukoliko ne postoji korisnik sa zadatim imenom, vratiti klijentu statusni kod 404 sa praznim telom. Inače, koristiti dohvaćene podatke za upisivanje novog podatka u bazu podataka (u kolekciju sa porukama). U telo odgovora smestiti ime primaoca. Na klijentu je potrebno prikazati obaveštajni prozor u kojem piše **Uspesno ste poslali poruku korisniku: <IME>**, kao na slici **post2.png** i obrisati sadržaj polja formulara. Obraditi sve potencijalne greške na serveru i klijentu.

## Obavezni zahtevi i napomene

- Stilizovanje dokumenta pomoću biblioteke **Bootstrap**, tako da izgleda kao na slici **preview.png**, može doneti **dodatna 2 poena**. Ukupno je moguće osvojiti najviše 45 poena.
- Neophodno je barem jednom izvršiti stilizovanje elementa korišćenjem **JavaScript** programskog jezika ili pomoću biblioteke **jQuery**.
- Sintaksne greške u bilo kom delu koda rezultovaće umanjnjem ukupne sume poena za 30%.