Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.39
DOI: https://doi.org/10.24867/27BE08Potrebic

TROSLOJNA ARHITEKTURA KAO SOFTVERSKO REŠENJE SISTEMA ZA EVIDENCIJU RADA ZAPOSLENIH

THREE-TIER ARCHITECTURE AS A SOFTWARE SOLUTION FOR A WORK RECORD SYSTEM

Ljubica Potrebić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *u ovom članku opisana je implementacija i rad sistema za evidenciju rada zaposlenih. Radi se o troslojnom sistemu koji podrazumeva implementaciju klijentske i serverske strane, kao i kreiranje i upotrebu baze podataka koja se koristi za skladištenje istih. Projekat predstavlja pristup razvijanja web aplikacija upotrebom tehnolohija kao što su Node.js i React, služeći se programskim jezikom JavaScript. Baza podataka koja se koristila je MongoDB, koja predstavlja nerelacionu bazu podataka. Podržane su funkcionalnosti kao što je logovanje administratora i radnika, dodavanje novih radnika, zadataka, omogućen je pregled aktivnih i završenih zadataka, isplaćivanje radnika prema kalendaru, uklanjanje radnika i zadataka, unos odrađenih sati u toku izabranog datuma, završavanje zadataka i uvid u personalne informacije.*

Ključne reči: *Web, JavaScript, Node.js, React, MongoDB*

Abstract – *This article describes implementation and functionality of work record system. It is a three-tier system that includes the implementation of the client and server side, as well as the creation and use of a database used for storage. The project represents an approach to developing web applications using technologies such as Node.js and React, using the JavaScript programming language. The database used is MongoDB, which is a non-relational database. Supported functionalities include administrator and worker login, adding new workers, tasks, viewing active and completed tasks, paying workers according to the calendar, removing workers and tasks, entering worked hours during the selected date, completing tasks and insight into personal information.*

Keywords: *Web, JavaScript, Node.js, React, MongoDB*

1. UVOD

U cilju napredovanja poslovne zajednice neophodno je obezbediti uslove za olakšan rad, a to je najpre brza komunikacija između poslodavca i zaposlenog. Kao glavno sredstvo u toj komunikaciji korsti se internet, a preko interneta i *web* aplikacije koje su upravo alat za komunikaciju. Aplikacija za vođenje evidencije o zaposlenima omogućava upravo komunikaciju između poslodavca i zaposlenog, bez potrebe za fizičkim prisustvom lica na istom mestu. Glavne omogućene funkcionalnosti su da je moguće zadavanje zadataka radnicima, dodeljivanje plata u zavisnosti od toga kako se radnik isplaćuje, uvid u zadatke na kojima se radnik nalazi, a takođe je omogućen i uvid u ukupnu zaradu koju je radnik ostvario tokom rada u firmi, čime se na dodatan način zaposleni motivišu da rade savesno.

Projekat je napravljen tako da se sastoji od dva osnovna bloka. Prvi blok predstavlja serversku stranu, čiji se naziv u praksi sreće kao *Back-end*. Server je baziran na *Node.js* platformi [1]. Za rad na platformi je korišćen *JavaScript* programski jezik. Serverska strana takođe obuhvata i nerelacionu bazu podataka *MongoDB*, gde se čuvaju svi podaci [2]. Drugi blok je klijentska strana, češće nazivana *Front-end*, gde se koristi *React.js* korisnički interfejs koji u sebi obuhvata upotrebu *JavaScript* jezika, *HTML* i *CSS* jezika za kreiranje i dizajniranje *web* stranica [3].

Realizovana je i *Android* aplikacija, napravljena upotrebom *React Native*, koja se takođe koristi na klijentskoj strani kao alternativno rešenje za *Front-end* klijentsku komponentu aplikacije. Ona svakako pomaže pri poboljšanoj komunikaciji između poslodavca i zaposlenog, pošto ne podrazumeva svaki posao rad na računaru.

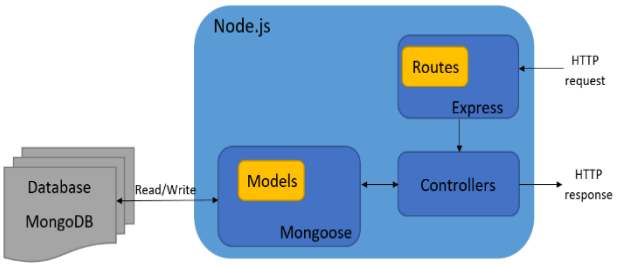
2. SISTEM

Poglavlje opisuje ključne aspekte i module sistema, uz teorijski osvrt.

2.1. Serverska strana

Entiteti koji sačinjavaju server su:

- *Node.js* – za pokretanje servera
- *MongoDB* – za upravljanje bazom podataka
- *Express*.js – za rutiranje
- Kontroleri – za obradu zahteva



Slika 1. Blok šema serverske strane

Node.js

Node.js jeste *JavaScript* platforma koja radi uz pomoć *Google V8 Engine*-a, što znači da ima odlične

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Predrag Teodorović, docent.

performanse, gde je najbitnija stavka brzina rada platforme. Svrha u koju se *Node.js* koristi je da omogući rad *Back-end* dela aplikacije bez korišćenja pretraživača (eng. *Browser*), te je korisničko iskustvo znatno poboljšano. Imajući u vidu prirodu *JavaScript*-a, svakako je neophodno da *Node.js* omogućava asinhroni rad *web* aplikacije, što je i učinjeno. Izvršavanje zahteva odvija se u samo jednom procesu, bez potrebe za kreiranjem novih niti (eng. *Thread*). Nije neophodno da se prethodni upit završi kako bi se izvršio novi – čim se dobije zahtev za izvršavanje upita, novi upit se izvršava, a čim se on završi, prethodni upit se izvršava počev od linije koda gde je prekinut. Na taj način se omogućava brz rad platforme bez dodatnog čekanja i blokiranja niti.

Kako bi korisničko iskustvo korišćenja *Node.js*-a bilo kompletno, neophodno je spomenuti i *Node Package Manager – npm*. Radi se o ugrađenoj podršci koja služi za dobavljanje i skladištenje spoljašnjih *Node.js* paketa koji služe za olakšani i brži rad na samoj platformi. Uz instalaciju *Node.js*-a automatski dolazi i *npm*. Najčešći način interakcije sa *npm*-om je preko komandne linije, odnosno preko terminala, gde je neophodno ukucati odgovarajuću *npm* instalacionu komandu, nakon čega se paketi preuzimaju i smeštaju u listu zavisnosti (eng. *Dependency*) u datoteku *package* koja na kraju ima *json* ekstenziju. Paketi koji su se koristili na serverskoj strani su sledeći:

- *Mongoose* – za rad sa *MongoDB* bazom podataka
- *Express* – za rutiranje
- *Cors* – za ograničavanje pristupa podacima sa drugih servera

MongoDB

MongoDB je baza podataka koja se koristi za smeštanje i čuvanje podataka dobijenih sa serverske strane. S obzirom da je za funkcionalnost serverske strane zaslužan *Node.js*, korišćen je paket *Mongoose* za pristupanje navedenoj bazi podataka i manipulacijom istih. Radi se o paketu koji ima izuzetno jednostavan interfejs za modeliranje podataka objekata (eng. *ODM – Object Data Modeling*). Koristi se kao sprega između objekata u kodu i njihove reprezentacije u bazi podataka. *MongoDB* spada u vrstu nerelacionih baza podataka, odnosno *NoSQL* baza. Predstavlja dokumentacionu bazu podataka gde se podaci smeštaju u *JSON* formatu.

Express

Express predstavlja jedan od paketa *Node.js*-a koji služi za rad sa *HTTP* zahtevima koji pristižu sa klijentske strane. Koristi se i u svrhu interakcije između serverske strane i baze podataka, gde server šalje odgovarajući zahtev ka modelu kako bi se izvršila odgovarajuća operacija nad bazom podataka. Za rad su korišćene dve metode, a to su *GET* i *POST* metode.

Ka serveru pristižu različite vrste upita od strane klijenta, te je neophodno da se oni klasifikuju radi bolje preglednosti i funkcionalnosti. U te svrhe se kreiraju putanje, odnosno rute (eng. *Routes*) za tačno određene krajnje tačke (eng. *Endpoint*). Na samom početku razvoja, početna putanja definisana je kao http://localhost:3001 i ova putanja podrazumeva da se rad obavlja na mašini koja je povezana na lokalnu mrežu. Za ostale uređaje bi bilo neophodno proslediti konkretnu *IP* adresu uređaja na kojoj je server aktivan, umesto *localhost* ključne reči. Rute služe za slanje zahteva dobijenih od klijentske strane ka odgovarajućim kontrolerima, koji dalje izvršavaju željenu funkcionalnost.

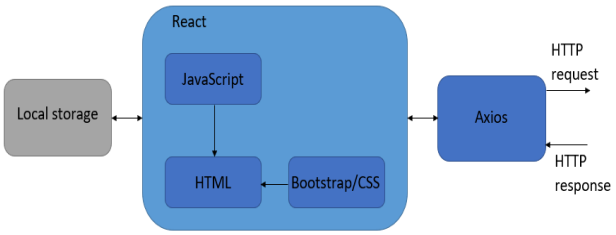
Kontroleri

Kontroleri se koriste u svrhu komunikacije između servera i baze podataka. Komunikacija se obavlja tako što kontroleri preko modela šalju odgovarajuće upite ka bazi. Kreiranjem upita definišu se uslovi koje je potrebno ispuniti kako bi se kao odgovor od baze dobili željeni podaci. Za primer se može navesti dodavanje novog zaposlenog na *web* stranici. Na početku je neophodno popuniti sve podatke na klijentskoj strani, kao što su ime, prezime, datum prijema u firmu, način isplate i plata u zavisnosti od načina isplate. Zatim se ti podaci smeštaju u *HTTP* zahtev ka ruti na serverskoj strani http://localhost:3001/home-admin/newEmployee, koji potom server preuzima i prosleđuje odgovarajućem kontroleru. Kontroler na kraju ima zadatak da ispuni željenu funkcionalnost, pri čemu će na pravilan način interagovati za bazom podataka, a potom i poslati povratnu informaciju o izvršenju zadatka.

2.2. Klijentska strana

Klijentska strana zapravo predstavlja korisnički interfejs gde korisnik šalje određene upite serveru. U svrhe realizacije klijentske strane, korišćena je *React* biblioteka. Preduslov za kreiranje *React* dela aplikacije je upotreba *npm*-a, koji se koristio i na serverskoj strani. Glavni entiteti korišćeni na klijentskoj strani su sledeći:

- *React* – osnova za kreiranje korisničkog interfejsa
- *Bootstrap* – za izgled korisničkog interfejsa
- *Axios* – za slanje *HTTP* zahteva



Slika 2. Blok šema klijentske strane

React

React predstavlja biblioteku za izradu korisničkih interfejsa i omogućava kreiranje aplikacije koristeći *JavaScript*. Uvodi sintaksnu ekstenziju za *JavaScript* pod nazivom *JSX* (eng. *JavaScript Syntax Extension*). Navedena ekstenzija omogućava pisanje *HTML* elemenata u *JavaScript*-u, pri čemu se vrši konverzija *HTML* elemenata u *React* elemente, a takođe je omogućeno pisanje *HTML* strukture i *JavaScript* koda u okviru iste datoteke. Za upravljanjem elementima koriste se komponente. Objekti mogu biti jednostavne strukture, ako koriste samo elemente, a mogu biti i složene ukoliko koriste skup više različitih elemenata za kreiranje jedinstvene komponente.

Za izgled korisničkog interfejsa se uz *HTML*, koji se koristi za definisanje strukture i sadržaja, dodatno može kombinovati i *CSS*. Radi se o jeziku formatiranja pomoću

kog se definiše izgled elemenata *web* stranice. Sintaksa *CSS*-a je takva da opisuje izgled elemenata u dokumentu. Opis može da definiše više elemenata, a moguće je i da više opisa bude vezano za jednu komponentu, primenjujući svoj sadržaj da bi se formirao konačni izgled jednog elementa.

Bootstrap

Bootstrap predstavlja *framework* otvorenog koda koji koristi kombinaciju *HTML*-a, *CSS*-a i *JavaScript*-a. Razvijen je sa ciljem da omogući i olakša razvoj *web* formi kao što su interfejs ili *layout*, kao i razvoj naprednih *web* komponenti. Njegova glavna upotrebna vrednost ogleda se u izgledu korisničkog interfejsa. Radi se o kolekciji *CSS* i *JavaScript* alata i biblioteka. Modularnog je tipa, te je moguća i izmena komponenti ukoliko programer želi da ubaci dodatne funkcionalnosti vezane za sam interfejs. Osim što omogućava i olakšava integrisanje raznih komponenti, predstavlja i odličnog „saradnika" u radu sa *jQuery* bibliotekama.

3. FUNKCIONALNOSTI

Poglavlje predstavlja opis ključnih funkcionalnosti korišćenih u izradi *web* aplikacije.

3.1. Prijavljivanje korisnika

Na početnoj strani *web* aplikacije prikazan je interfejs za prijavljivanje korisnika. Preduslov za prijavljivanje korisnika je da prethodno budu upisani u bazu podataka. Nakon unošenja korisničkog imena i lozinke, neophodno je izabrati i odgovarajuću ulogu – *Administrator* ili *Employee*, pri čemu ako se izabere pogrešna uloga, korisnik neće biti pronađen u bazi podataka i ispisaće se odgovarajuća poruka koja će naglasiti da uneseni korisnik ne postoji.
Nakon unosa ispravnog korisničkog imena i lozinke, klikom na dugme *Login* prelazi se na narednu stranu, koja će zavisiti od toga da li je uloga korisnika *Administrator* ili *Employee*. Ukoliko je izabrana uloga *Administrator*, prelazi se na početnu stranicu administratora, gde se nalaze *tab*-ovi sa odgovarajućim namenama. U suprotnom, ako je izabrana uloga *Employee*, prelazi se na početnu stranicu zaposlenog, gde zaposleni ima *tab*-ove sa drugačijom namenom u odnosu na administratora. Logovanjem korisnika čuvaju se korisnički podaci na *Local Storage*-u, kako bi se znalo ko je ulogovani korisnik. Podaci se čuvaju u vidu korisničkog imena i uloge.

3.2. Dodavanje novog zaposlenog

Podrazumevani *tab* na administratorskoj strani je *tab Add new employee*, gde je neophodno uneti ime, prezime, datum rođenja, broj telefona, datum prijema u firmu, šifru, tip zarade (da li se zaposleni isplaćuje prema satu, danu ili mesecu), zaradu u zavisnosti od tipa zarade i broj radnih sati koje radnik treba da uradi tokom radnog dana. Korisničko ime se generiše automatski nakon unosa imena i prezimena. Ukoliko su sva polja popunjena, klikom na dugme *Add new employee* šalje se zahtev serveru za upis novog zaposlenog u bazu podataka. Ispunjavanjem uslova, koji podrazumevaju da ne postoji korisnik sa istim korisničkim imenom, novi korisnik se dodaje u bazu podataka. U suprotnom, neophodno je promeniti korisničko ime novog zaposlenog.

3.3. Dodavanje novog zadatka

Naredni *tab* je *tab Add new task*, koji služi za dodavanje novog zadatka. Na početku je neophodno uneti naziv zadatka, potom se iz padajućeg menija biraju zaposleni koji će izvršiti dati zadatak, i njihovim odabirom se njihovo korisničko ime smešta u polje za odabrane zaposlene kako bi izbor bio pregledniji.
Ukoliko se poslodavac pak odluči da ne želi da neko od odabranih zaposlenih radi na zadatku, može ga jednostavno obrisati iz polja za odabrane zaposlene. Potrebno je odrediti i da li za zadatak postoji jednokratni bonus koji će se isplaćivati radnicima.
Klikom na jedno od dugmadi *yes* ili *no* se određuje da li postoji bonus, te ako bonus postoji, javlja se dodatno polje koje služi za unos novčanog iznosa jednokratnog bonusa. Na kraju, klikom na dugme *Add new task* serveru se šalje upit za kreiranje novog taska.

3.4. Uvid u listu zaposlenih

Sledeći *tab*, *List of employees*, namenjen je prikazu informacija o odabranom zaposlenom. Odabir zaposlenog lica se vrši iz padajućeg menija, klikom na korisničko ime zaposlenog. Nakon toga se u poljima izlistavaju informacije kao što su ime i prezime, korisničko ime, datum prijema u firmu, zarada i tip zarade, a takođe se izlistavaju i aktivni i završeni zadaci radnika, gde se prikazuje ocena koju je radnik dobio na zadatku ukoliko je ocenjen za taj zadatak.

3.5. Uvid u listu aktivnih zadataka

Nakon liste zaposlenih, neophodno je prikazati i zadatke koji nisu završeni, a to se čini odabirom *tab*-a *Active tasks*. Moguće je odabrati zadatak klikom na dugme sa njegovim imenom. Izlistaće se osnovne informacije, kao što su ime zadatka, da li postoji bonus za dati zadatak, ukoliko bonus postoji prikazaće se informacija koliko iznosi bonus, i naravno izlistaće se radnici koji rade na datom zadatku.

3.6. Uvid u listu završenih zadataka i ocenjivanje radnika

Pored liste aktivnih zadataka, podrazumeva se i prikaz liste završenih zadataka, koja se nalazi na *tab*-u *Finished tasks & Rate employee*. Klikom na odabrani završeni zadatak prikazuju se slične informacije kao i kod aktivnih zadataka, uz izmenu da je moguće odabrati zaposlenog koji će se oceniti za dati zadatak. To se čini klikom na njegovo korisničko ime, te se popunjavaju osnovne informacije o datom radniku.

Ocenjivanje radnika vrši se klikom na jedno od dugmadi, koji su označeni brojevima između 1 i 5. Dodatna opcija koja nije obavezna je da se ostavi poruka za radnika u polju *note*. Ukoliko radnik nije prethodno ocenjen za taj zadatak, klikom na dugme *Rate employee* radnik se uspešno ocenjuje i podaci se upisuju u bazu podataka.

3.7. Dodeljivanje plata

Jedan od najbitnijih *tab*-ova jeste *Salaries* koji služi za dodeljivanje plata radniku. Odabirom zaposlenog izlistavaju se njegove osnovne informacije, potom je neophodno kliknuti na polje za odabir datuma, gde se odabira datum za isplaćivanje radnika. Ukoliko je datum obeležen crvenom bojom, znači da radnik nije uneo

odrađenu satnicu za taj dan, te se radnik u tom slučaju ne može isplatiti. Ukoliko je datum obeležen žutom bojom, to znači da je radnik uneo odrađenu satnicu za taj dan ali nije isplaćen. Jedino se u tom slučaju radnik može isplatiti za odabrani datum. U poljima ispod će biti prikazana satnica koju je radnik obavio tog dana i proračunata plata u tom slučaju. Klikom na dugme *Pay employee*, zaposleni se isplaćuje. Poslednji slučaj koji se može videti u kalendaru jeste da je datum obeležen zelenom bojom, što znači da je radnik isplaćen za taj dan i nije moguće ponovno isplaćivanje radnika. Ukoliko je radnik plaćen na mesečnom nivou, prikaz i isplaćivanje se menjaju u smislu da se prikazuje ukupna satnica koja je odrađena za taj mesec i izračunata plata u tom slučaju. Poslodavac ne može da isplati mesečnu platu ako mesec nije završen. Takođe, ako je radnik uneo satnicu za manji broj dana nego što je predviđeno, biće isplaćen za ceo mesec u zavisnosti od broja odrađenih sati i radnik više neće biti u mogućnosti da unosi odrađene sate za taj mesec.

3.8. Uklanjanje zaposlenih

Odabirom zaposlenog izlistavaju se osnovne informacije na osnovu kojih se odlučuje da li će radnik biti uklonjen. Klikom na dugme *Remove employee*, zaposleni se briše iz baze podataka svuda gde se prethodno nalazio, pri čemu se briše iz zadataka koje je prethodno radio.

3.9. Uklanjanje zadataka

Logično je da pored uklanjanja zaposlenih postoji i uklanjanje zadataka. Odabirom zadatka se prikazuju osnovne informacije o zadatku, a klikom na dugme *Remove task*, zadatak se efikasno uklanja iz baze podataka.

3.10. Pregled osnovnih informacija zaposlenog

Ako je korisnik koji se ulogovao zaposleni, dolazi na početnu stranicu za zaposlene, gde je podrazumevani *tab Personal information*. Tu se izlistavaju sve informacije ulogovanog zaposlenog lica, uključujući i ukupna dosadašnja primanja, što može da pruži podsticaj zaposlenom da radi kvalitetno i savesno, a to svakako ide u prilog firmi u kojoj je radnik zaposlen.

3.11. Pregled aktivnih zadataka zaposlenog

Kako je neophodno videti aktivne zadatke sa serverske strane, tako je potrebno videti ih sa klijentske strane. Odabirom *tab*-a *Your active tasks* izlistavaju se svi zadaci za koje je odabran ulogovani korisnik. Mogu se videti osnovne informacije zadatka, kao i zaposleni koji su odabrani za zadatak. Za razliku od serverske strane, na klijentskoj strani zaposleni ima mogućnost da završi zadatak klikom na dugme *Finish task*, čime se zadatak premešta na listu završenih zadataka.

3.12. Pregled završenih zadataka zaposlenog

Na *tab*-u *Your finished tasks* mogu se videti svi gotovi zadaci na kojima se nalazio ulogovani korisnik. Ukoliko je zaposleni ocenjen za dati zadatak, prikazaće se i ocena koja je dobijena na zadatku, a ako je uneta i beleška o radniku od strane poslodavca, ona će takođe biti vidljiva radniku.

3.13. Unos odrađenih sati

Kako je na serverskoj strani najbitnije dodeljivanje plata zaposlenima, tako je na klijentskoj strani najbitnije da zaposleni unese odrađene sate za odgovarajući dan. To se čini klikom na *tab Enter worked hours*, gde zaposleni ima uvid o informacijama koje se tiču njegove zarade. Na sličan način kao i kod dodeljivanja plata, neophodno je kliknuti na dugme kako bi se otvorio kalendar. Zaposleni može da unese odrađene sate samo za datume obeležene crvenom bojom, odnosno kada nisu prethodno uneseni odrađeni časovi. U slučaju da je unos sati već odrađen, datumi su obeleženi žutom bojom i nije moguće uneti sate za taj dan iznova. Takođe se odrađeni sati ne mogu uneti za dane kada je radnik plaćen, gde se ti dani u kalendaru obeležavaju zelenom bojom.

3.14. Odjavljivanje korisnika

Poslednja, ali ne manje bitna funkcionalnost koju treba spomenuti jeste odjavljivanje korisnika, koja se i kod administratora i kod zaposlenog obavlja pritiskom dugmeta *Log Out*. Nakon toga se iz *Local Storage*-a brišu prethodno sačuvano korisničko ime i uloga, vraćajući se na osnovnu putanju gde se korisnik prijavljuje.

4. ZAKLJUČAK

Na osnovu prethodnog izlaganja, izvodi se zaključak da je sistem baziran na primeni klijent-server komunikacije sa praktičnom primenom uspešno realizovan. Projekat je podrazumevao razvijanje *web* sajta koristeći jedan od najaktuelnijih jezika za razvoj, a to je *JavaScript,* uz upotrebu *Node.js*-a, *React*-a i paketa koje oni pružaju.
Mesta za napredak ima mnogo, a navode se samo neki od predloga za poboljšanje rada aplikacije:

- Uvođenje evidencije o zaradi na mesečnom i dnevnom nivou
- Uvođenje obaveštenja za zaposlene radi brže komunikacije između poslodavca i zaposlenog
- Uvođenje odabira radnih dana, kao i slobodnih dana vikendom, uvođenje godišnjeg odmora i mogućnost bolovanja, gde bi se plaćanje radnika vršilo shodno situaciji
- Prelazak sa *HTTP* na *HTTPS* protokol radi sigurnosti i bezbednosti podataka na *web* stranici

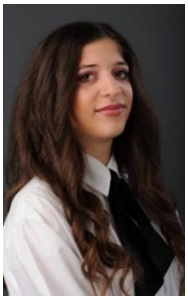
5. LITERATURA

[1] SHAH, DHRUTI NA. *Node .JS*. BPB PUBLICATIONS, 2018.

[2] Amit Phaltankar, Juned Ahsan, Michael Harrison, Liviu Nedov. *MongoDB Fundamentals*. Packt Publishing Ltd, December 2020.

[3] Cory Gackenheimer. *Introduction to React*. Apress, 2015.

Kratka biografija:



Ljubica Potrebić rođena je u Novom Sadu 1999. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnsika i računarstvo – Primenjena elektronika odbranila je 2022. god.

kontakt: potrebicljubica@gmail.com