Programiranje internet aplikacija

Elektrotehnički fakultet, Univerzitet u Beogradu

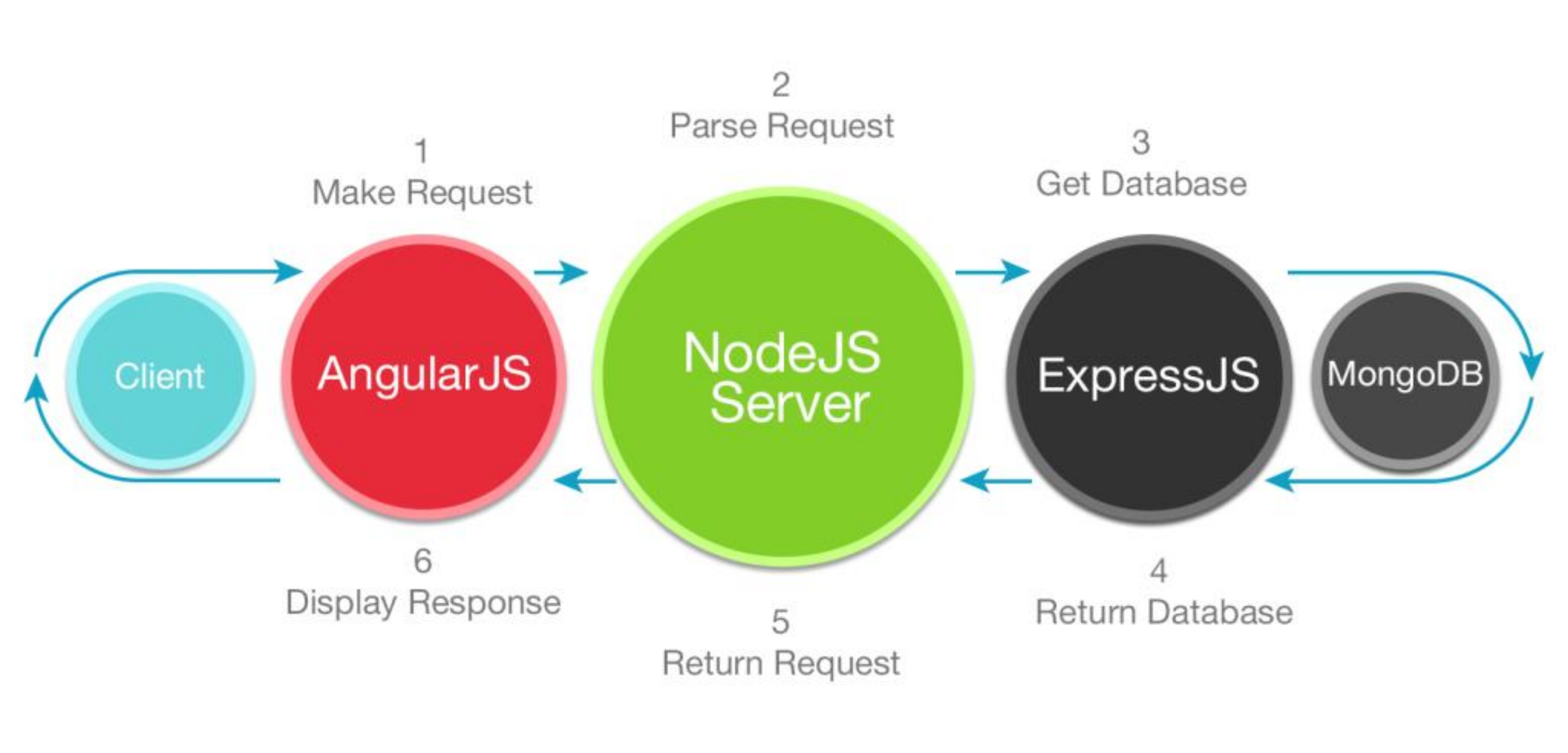
2024/2025

Uvod

- Deo MEAN steka – *open-source* JavaScript softverski stek

- M – MongoDB
- E – Express
- A – Angular
- N – Node.js

MEAN stack


2
Parse Request
1
Make Request
3
Get Database
Client
AngularJS
NodeJS Server
ExpressJS
MongoDB
6
Display Response
5
Return Request
4
Return Database

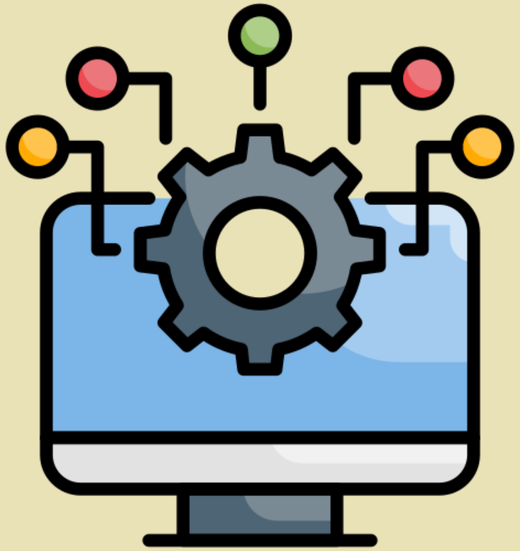
Node.js

- JavaScript ili TypeScript izvorni kod
  - TypeScript izvorni kod se prevodi u JavaScript kod
- Asinhroni *event-driven JavaScript enviroment* za izvršavanje na serverskoj strani
- Jedan proces koji omogućava asinhrone neblokirajuće pozive ka API-jima, čime se omogućava velika konkurentnost bez eksplicitnog *thread management*-a
- `npm` *package installer*

Express

- *Backend framework* za posredovanje između Node.js i baze podataka
- REST API
- CRUD operacije (create, read, update, delete)

Instalacija, kreiranje i pokretanje aplikacije



- Node.js (20.17.0)
  - https://nodejs.org/download/release/v20.17.0/

  - npm init
    - za entry point staviti dist/server.js

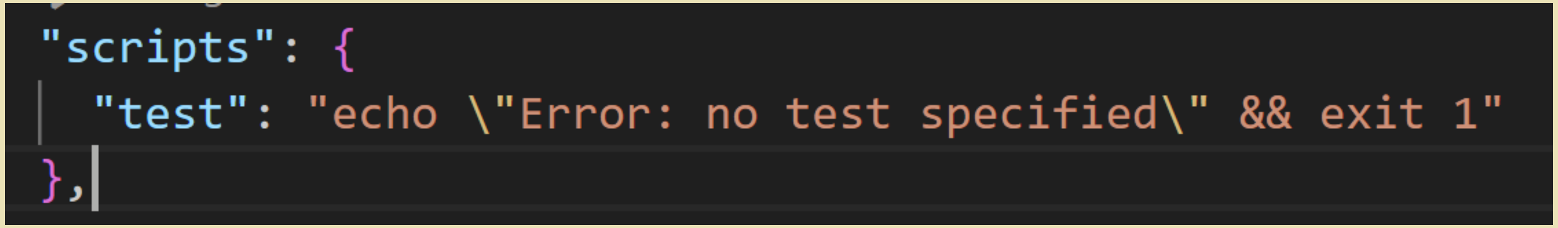
```
npm install typescript --save-dev
npm install @types/node --save-dev
npm install express
npm install @types/express --save-dev
npm install cors
npm install @types/cors --save-dev
npm install mongoose
npm install mongodb
```

package.json

```
npx tsc --init --outDir dist
```

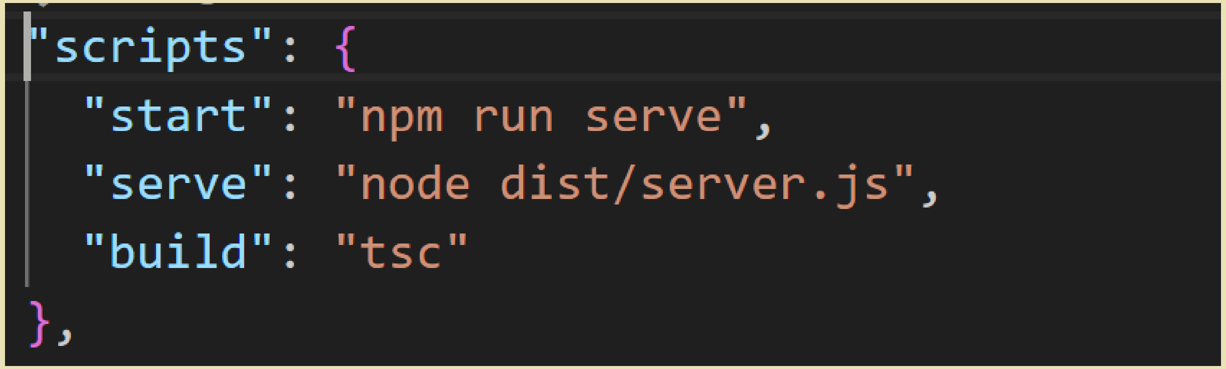
tsconfig.json

U package.json izmeniti još i:



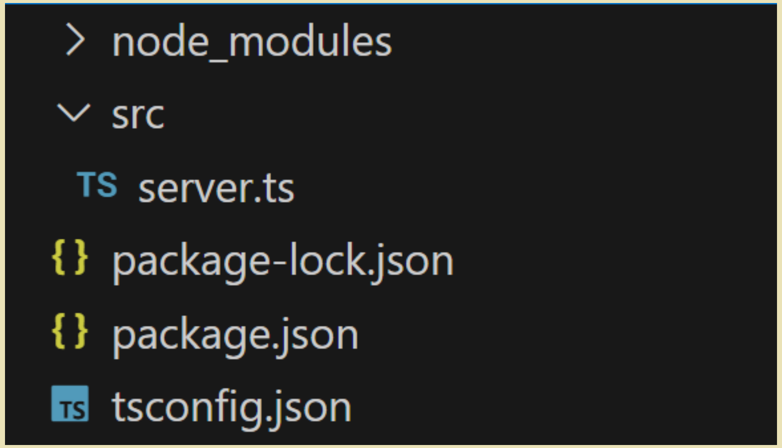
```
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1"
},
```

sa:

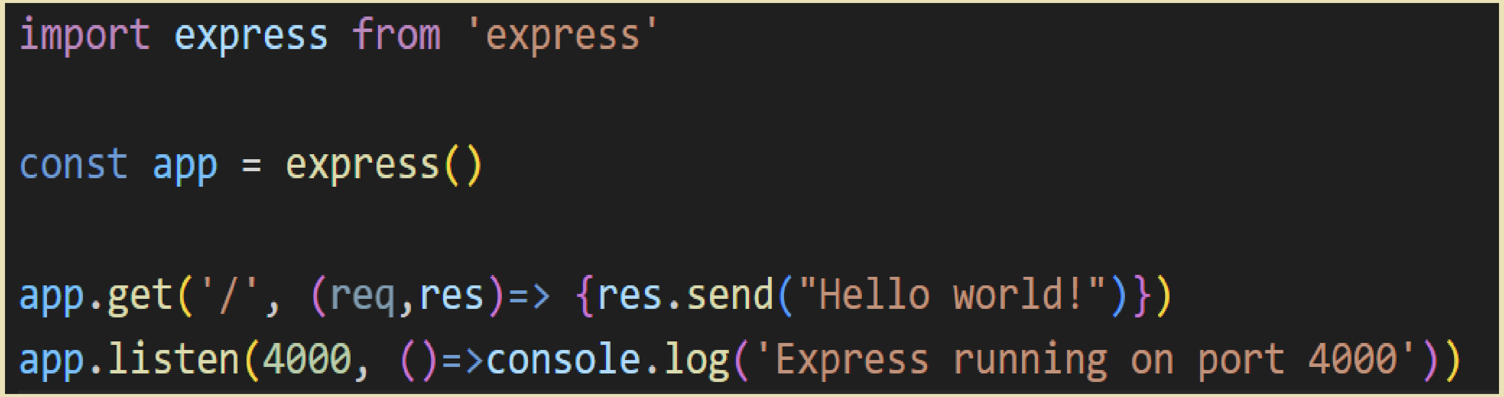


```
"scripts": {
  "start": "npm run serve",
  "serve": "node dist/server.js",
  "build": "tsc"
},
```

Struktura projekta i početni kod



```
> node_modules
∨ src
   TS server.ts
{} package-lock.json
{} package.json
TS tsconfig.json
```

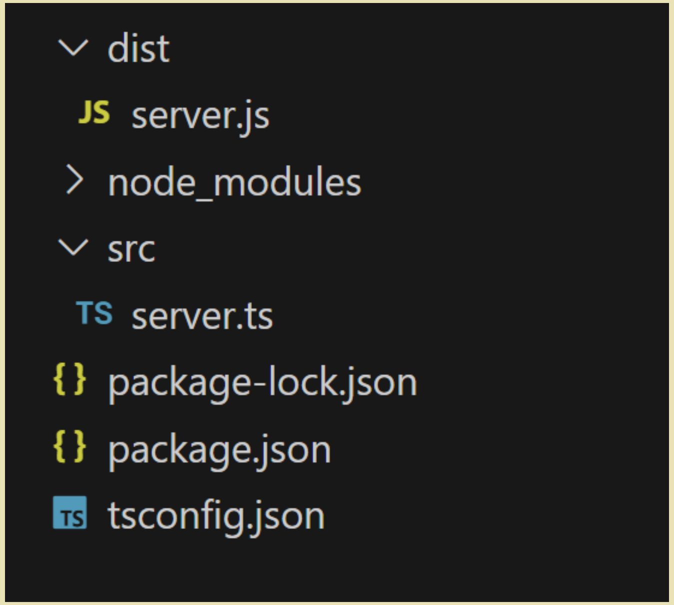


```
import express from 'express'

const app = express()

app.get('/', (req,res)=> {res.send("Hello world!")})
app.listen(4000, ()=>console.log('Express running on port 4000'))
```

- app.get – get zahtev
- req – request, res – response
- app.listen – osluškivanje na portu

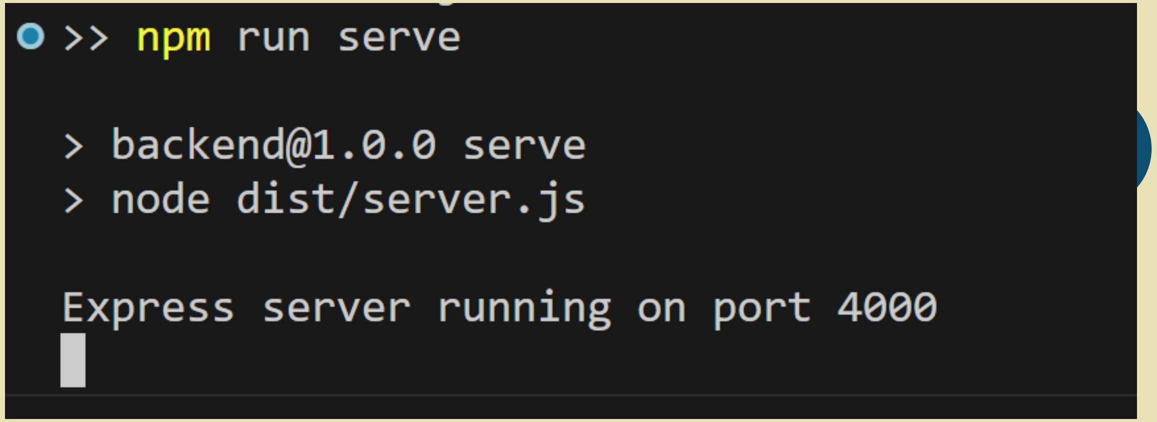
Pokretanje projekta

- Komanda `tsc`
  - Kompajlira sve iz src
  - `dist` folder

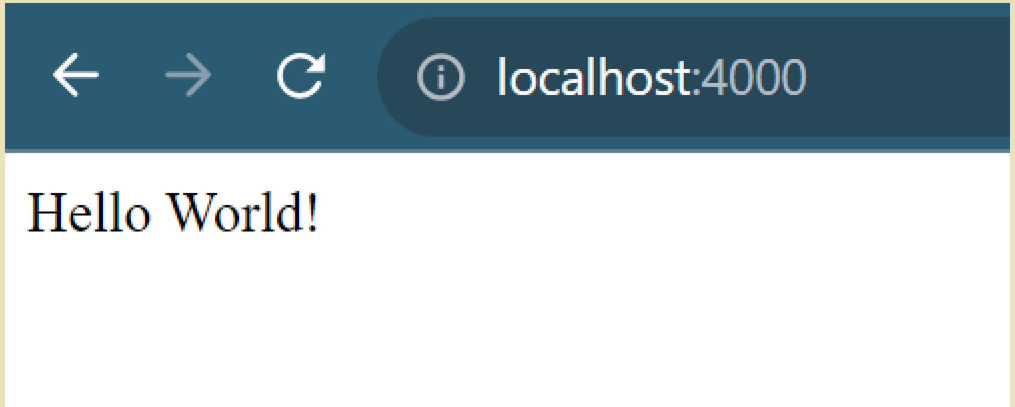


```
∨ dist
   JS server.js
> node_modules
∨ src
   TS server.ts
{} package-lock.json
{} package.json
TS tsconfig.json
```

- Pokretanje – `npm run serve`



```
>> npm run serve

> backend@1.0.0 serve
> node dist/server.js

Express server running on port 4000
```



Hello World!

Kontroleri

```
// controllers/user.controller.ts
import express from 'express’
// import modela itd

export class UserController {
    login = (req: express.Request, res: express.Response) => {
        // npr.
        let username = req.body.username;
        let password = req.body.password;
        let result = . . .;

        // . . .
        res.json(result);
    }

    register = (req: express.Request, res: express.Response) => {
        . . .
    }
}
```

Rutiranje

```
// routers/user.router.ts
import express from 'express'
import { UserController } from '../controllers/user.controller';

const userRouter = express.Router();

userRouter.route('/login').post(
    (req, res) => new UserController().login(req, res)
)

export default userRouter;
```

Rutiranje – korišćenje u server.ts

```
const router = express.Router();
router.use('/users', userRouter)

// const app = express();
app.use('/', router);
```

CORS

```
// server.ts
import cors from 'cors' // za cross-origin

// const app = express();
app.use(cors())
app.use(express.json()) // označavamo da podatke šaljemo u JSON formatu
```

HVALA NA PAŽNJI!