Софийски Университет

Разпределени Системи и Мобилни Технологии

Проект по Fullstack with Node.js, React.js, Express.js

Тема: Блог за домашни любимци

Изработил: Антоан Андонов, ф.н. 25613

Преподавател: ……..
/ас. Т. Илиев/

Дата: 25.06.2018

Съдържание

Резюме

Настоящият документ е изготвен във връзка с разработването на проект по курса FullStack with Node.js, React.js, Express.js, 2017-2018, летен семестър към СУ, ФМИ, с преподавател ас. Траян Илиев.

Предназначението на настоящия документ е да даде най-общо концептуално описание на целената реализация на уеб базирана система. Курсът разглежда задълбочено React.js и Redux от гледна точка на тяхната практическа употреба за реализиране на реални уеб и мобилни приложения (SPA). Facebook React.js се налага като една от най-разпространените в практиката библиотеки за разработка на съвременни SPA уеб приложения, която съчетава простота и лекота на употребата с ефективност на реализацията.

Въведение

Представлява уеб-базирано приложение, което е разработено на JavaScript/ ECMAScript и използва следните технологии:

- Node.js
- React.js
- Express.js
- My SQL

Заданието, което е реализирано е на тема “Блог за домашни любимци”, където всеки може да търси информация за различните породи любимци. Реализацията на това уеб-приложение ще съдържа следните функционалности:

1. Регистрация на потребител
2. Търсене на информация за дадена порода
3. Добавяне на нов пост
4. Редактиране на пост
5. Изтриване на пост
6. Администраторите имат достъп до акаунтите на всички потребители
7. Администраторите имат достъп до акаунтите на всички любимци

Технологии

- axios - v0.18.0 - https://github.com/axios/axios
- bootstrap - v4.1.1 - https://getbootstrap.com/docs/4.1/getting-started/introduction/
- cors - v2.8.4 - https://www.npmjs.com/package/cors
- npm - v6.1.0 - https://docs.npmjs.com
- react - v16.4.1 - https://reactjs.org/docs/hello-world.html
- react-dom - v16.4.1 - https://reactjs.org/docs/react-dom.html
- react-router - v4.3.1 - https://reacttraining.com/react-router/web/guides/philosophy
- react-router-dom - 4.3.1 - https://www.npmjs.com/package/react-router-dom
- mysql - v2.15.0 - https://dev.mysql.com/doc/refman/8.0/en/
- body-parser - v1.18.3 - https://github.com/expressjs/body-parser
- express - v4.16.3 - https://expressjs.com/en/4x/api.html
- joi - v13.3.0 - https://github.com/hapijs/joi
- morgan - v1.9.0 - https://github.com/expressjs/morgan
- nodemon - v1.17.5 - https://github.com/remy/nodemon#nodemonАрхитектура

Архитектура

В тази секция ще опиша архитектурата, която следвам в създаването на приложението, като ще дам пример за съответните end-point-и как са дефинирани от страната на сървъра и как се използват/извикват от страната на клиента.

Rest API Server-side

```
userValidation = (obj) => {
  const schema = {
    id: Joi.number().required(),
    firstName: Joi.string().min(2).required(),
    lastName: Joi.string().min(2).required(),
    userName: Joi.string().min(3).required(),
    email: Joi.string().min(5).required(),
    password: Joi.string().min(5).required(),
    isAdmin: Joi.boolean().required()
```

```
    }
    return  Joi.validate(obj, schema);
}

router.get('/api/pets', (req, res)
router.get('/api/posts', (req, res)
router.get('/api/users', (req, res)
router.get('/api/:userName/pets', (req, res)
router.get('/api/profile/:userName/', (req, res)

router.post('/api/login', (req, res)
router.post('/api/signin', (req, res)
router.post('/api/pets/add', (req, res)
router.post('/api/posts/add', (req, res)

router.put(`/api/:userName/edit`, (req, res)
router.put('/api/pets/:petId/', (req, res)
router.put('/api/posts/:postId/', (req, res)

router.delete('/api/posts/:postId', (req, res)
router.delete('/api/pets/:petId', (req, res)
```

Rest API Client-side

```
axios.get(`${host}/api/pets`)
axios.get(`${host}/api/posts`)
axios.get(`${host}/api/users`)
axios.get(`${host}/api/${this.state.user.userName}/pets`)
axios.get(`${host}/api/profile/` + this.state.user.userName)

axios.post(`${host}/api/login`, user)
axios.post(`${host}/api/signin`, user)
axios.post(`${host}/api/posts/add`, post)
axios.post(`${host}/api/pets/add`, pet)

axios.put(`${host}/api/${user.userName}/edit`, user)
axios.put(`${host}/api/posts/${post.id}`, post)
axios.put(`${host}/api/pets/${pet.id}`, pet)

axios.delete(`${host}/api/posts/${postId}`)
axios.delete(`${host}/api/pets/${petId}`)
```

Инсталация

За правилната инсталация и поддръжка на приложението е необходимо да се инсталира Node.js. След това се инсталират другите депендънсита с npm или yarn, като за стартирането е необходимо да се пусне първо сървъра с в съответната директория с команда: ***nodemon index.js***. С тази команда ще се стартира сървъра, който ще слуша на порт 3003. След като тази стъпка е изпълнена може да се стартира и приложението или или още UI-частта, което представлява самия сайт. Това става като в съответната директория се изпълни командата ***yarn***, която да build-не всички инсталирани с npm пакети, а след това ***yarn start***, с която команда стартираме клиентската част на приложението, която работи на порт 3000.

Ръководство на потребителя

В тази секция ще опиша как да се използва реализираната система.

Като за начало при първо посещение в сайта, се зарежда главната Home страница, където се показва текущата прогноза за времето за София. До Home бутона се вижда и бутон Search, където като посетим страницата, може да търсим различна информация, която събира от няколко различни източника и зарежда отдолу. Дотук бяха описани функционалностите, които важат за всички нерегистрирани потребители.

Най-вдясно се виждат два бутона Sign in и Login, които както по името им става ясно, служат за регистрация на потребител и за вписване в системата.

След като попълним формата и си направим профил, се вписваме в системата, чрез мейл или потребителско име последвано от парола и това, което се променя е, че се показват още няколко връзки в системата, а именно Posts, Users и Pets. Съответно, само администратор може да преглежда данните в Users и Pets, като има права да редактира и/или изтрива информацията за даден потребител и/или домашен любимец за съответната секция. След като вече сме вписани в системата и посетим отново Search секцията, при търсене на информация за домашен любимец, ще се види най-отдолу един бутон Add post, който ще добави нов пост, към всички останали в секция Posts.

При успешно вписване в сайта на мястото на Login бутона, се вижда потребителското име на текущия потребител, където ако се посети тази секция, се зарежда профил на потребителя. Там може персонално да се редактира профила, а в секция My Petsмогат да се редактират добавените домашни любимци, също могат да се трият или да се добавят нови любимци. В под-секцията Calculator може всеки да си изчисли по колко количество храна трябва да изяжда домашния любимец, според възраст и килограми.

Заключение

За разработването и реализирането на този продукт са необходими солидни знания по уеб дизайн, HTML 5, CSS, JavaScript/ ECMAScript, jQuery и познание на технологии като Node.js, React.js, Express.js, както и задълбочени познания за софтуерен архитектурен стил REST, разработка на уеб услуги (APIs) и бази данни (в случая mysql).

По време на разработката срещани трудности и проблеми в клиентската част са не-добрата съвместимост на react с bootstrap, което доведе до неуспешна реализация на някои от формите, които предага bootstrap.

Друг срещан проблем бе при извличането на данни в JSON формат. На обекта, който е върнат от заявката не може да бъдат извлякани под-обектите в дълбочина, освен, ако не се запишат в нови обекти, които да се съхраняват в *this.state* в момента на тяхното извличане.

Следващите стъпки, които предстои да бъдат реализирани, ще бъдат свързани с подобрението на функционалността, потребителския интерфейс и добавянето на нови функционалности. Някои от идеите са:

- Подобряване на търсачката и включване на други източници на информация
- Добавяне на повече опции за менажиране на потребителския профил
- Добавяне на нови функционалности в частта с постовете, където всеки да може да коментира, харесва и добавя към любими дадена статия
- Подобряване на прогнозата за времето, която да взима текущия град за всеки потребител и да показва не само прогнозата в момента, а и за цялата седмица с подробна информация
- Добавяне на форма за изпращане на мейл до администратора на сайта

Разрешените проблеми бяха свързани с добавянето на cors, който успя да преодолее защитите на външните API-та, които използвам за извличане на прогнозата за времето, за извличането на изображение на домашен любимец и за извличане на информация за домашен любимец. Преди това, не се получаваха данните, поради ограничение в header на request-а от страна на *access-control-allow-headers*. Друг разрешен проблем, бе от страна на сървъра - с връзката към базата данни и с извличането на параметрите от даден request.

Източници

- Уеб сайт на W3C – http://w3.org
- Уеб сайт на платформата Node.js – https://nodejs.org/en/
- Уеб сайт на библиотеката React.js – https://facebook.github.io/react/
- Уеб сайт на Express.js – http://expressjs.com/
- Уеб сайт на My SQL – https://dev.mysql.com/doc/refman/8.0/en/
- Уеб сайт на Bootstrap – https://getbootstrap.com/docs/4.1/getting-started/
- Уеб сайт на Joi валидатор – https://github.com/hapijs/joi
- Уеб-сайт на React-router – https://reacttraining.com/react-router/web/guides/philosophy
- Уеб-сайт на axios – https://github.com/axios/axios
- Уеб-сайт за уроци – https://www.w3schools.com/Js/