



Софийски университет „Св. Климент Охридски“

Факултет по математика и информатика

КУРСОВ ПРОЕКТ ПО МУЛТИМЕДИЙНИ ТЕХНОЛОГИИ

Droid assembly

Автори: Николай Кънев, Христо Загоров

Специалност: Вградени системи

ОКС: Магистър

Фак. номера: 25789, 25637

Ръководител:

ас. Траян Илиев

София, 28.06.2018 г.

Съдържание

| | Страница |
|--------------------------------------|----------|
| 1 Увод | |
| 2 Изложение | 4 |
| 2.1 Функционални изисквания | 4 |
| 2.2 Използвани технологии | 5 |
| 2.3 Описание на сървърната част | 6 |
| 2.4 Начин на работа със системата | 9 |
| 4 Заключение | 10 |
| 5 Използвани източници | 11 |

1. Увод

В сферата на информатиката в последните години все повече популярност набира областта на Интернет на нещата (Internet of things) и хоби електрониката. Интернет пространството съдържа изобилие от уроци за сглобяване и програмиране на роботи и умни устройства с компоненти, налични онлайн за свободно закупуване.

Развитието на тази област се дължи на няколко основни фактора:

- За да програмира робот, човек вече не трябва да има дългогодишен опит с езици на ниско ниво като C/C++ и Assembly. За по-широко разпространените микроконтролери вече са създадени среди за разработка, които позволяват на програмиста да пише по-опростен код и предоставят достъпност до базовата им функционалност.
- Компонентите за изграждането на робот или друго електронно устройство са лесно достъпни за закупуване и на ниски цени. Уеб сайтове като Ebay и Aliexpress предлагат богат каталог от компоненти, сензори, актуатори, микроконтролери и всякакви други части на много изгодни цени с безплатна доставка до самия потребител.
- Широко са разпространени хардуерни платформи като Ардуино, които позволяват лесното свързване и програмиране на различни електронни устройства.

Когато човек реши да направи свой собствен робот, първият етап е планирането на необходимите за проекта части. В зависимост от района, части биха могли да се закупят и лично от магазин за електроника, но обикновено по-удобно и по-евтино излиза те да се поръчат онлайн. Съставянето на списъка с компоненти може да се окаже дълъг и объркващ процес, тъй като почти всеки бива предлаган в различни форми и разновидности, на различни цени. Целта на проекта е да улесни този етап, предлагайки удобен интерфейс за подбиране и сглобяване на частите и предоставяйки информация в текстов вид, придружена от видео уроци относно тяхната употреба.

Основните свойства на проекта са:

- Позволява на потребителя да създаде свой проект за робот, в който да търси, добавя и подрежда части чрез drag and drop.
- Да предостави на потребителя възможност да състави интерактивна диаграма за свързването на компонентите.
- За всяка част показва налични видео уроци (tutorial), за да може потребителят да види примери за използването ѝ и с какви други компоненти е съвместима.
- Да визуализира цената на проекта с избраните части, крайната дата, на която се очаква частите да са пристигнали, и линкове към налична техническа документация (datasheet).
- Да предостави на потребителите възможността да публикуват готовите си списъци от части в достъпна за всички потребители на сайта колекция.

2. Изложение

2.1. Функционални изисквания

От гледна точка на проекта съществуват три потребителски роли:

- Регистриран потребител – може да създава свои проекти, да търси части за тях и да ги запазва в профила си. Ще може и да ги публикува в достъпна за всички потребители колекция от готови проекти.
- Нерегистриран потребител – може да разглежда публикувани проекти, но за да създаде свой проект, ще трябва да си направи профил.

Кратко описание на потребителските случаи (Use cases)

| Име на потребителския случай | Кратко описание (Brief Descriptions) | Кратко описание на актьорите (Actor Brief Descriptions) |
|------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------|
| 1. Регистриране на нов потребител | Потребителят се регистрира в системата. | Нерегистриран потребител |
| 2. Създаване на нов проект | Потребителят създава нов проект, в който да добавя части за работа си. | Регистриран потребител |
| 3. Търсене на компонент | Потребителят търси компонент по ключови думи, което стартира заявка към Web API на Ebay и връща списък с резултати. | Регистриран потребител |
| 4. Търсене на видео урок | Потребителят маркира част и от YouTube се визуализира списък с налични за нея видео уроци, които могат да се гледат. | Регистриран потребител |
| 5. Създаване на дефиниция на компонент | Потребителят създава дефиниция на нова част от схемата на проекта си като въвежда име, по желание URL на картинка и цвят и брой и имена на входни и изходни пинове на частта. | Регистриран потребител |
| 5. Създаване на схема на свързването на компонентите | Потребителят маркира желаните части и задава какъв брой пинове имат, както и имената на пиновете. Те се добавят в графичен прозорец, в който могат да се свързват една с друга (вж. ресурс [5]), за да може да се получи цялостно описание на проекта, частите му и как да се сглобят. | Регистриран потребител |
| 6. Запазване на проекта | Регистрираният потребител запаметява проекта си, за да може да го отвори друг път от профила си. | Регистриран потребител |
| 7. Редактиране на проект | Регистрираният потребител избира проект от списъка си със запаметени проекти и го отваря за редакция. | Регистриран потребител |
| 8. Публикуване на коментар | Регистрираният потребител отваря страницата на даден проект и създава коментар от свое име | Регистриран потребител |
| 9. Изтриване на коментар | Регистрираният потребител отваря страницата на профила си и изтрива коментар, който е публикувал за даден проект. | Регистриран потребител |
| 11. Разглеждане на налични проекти | Потребителят разглежда списък с налични публикувани проекти и може да избере да отвори някой от тях за | (Не)регистриран потребител |

| | | |
|--|-----------------------------------|--|
| | разглеждане, но не и за редакция. | |
|--|-----------------------------------|--|

Структура на проекта с използвани мултимедийни материали и интерактивна хипермедия

- 1. Заглавна страница (Home)** – съдържа списък с публикувани вече проекти и линкове към тях, с които да могат да се отвори цялото налично за тях съдържание.
- 2. Създаване/редактиране на проект** – съдържа полета за търсене на компоненти, плочки с резултатите, предоставени от Ebay REST API, и drag and drop поле, в което да могат плочките да се пускат, за да се направят част от проекта. Drag and drop-натият компонент съдържа хиперлинк към Ebay страницата на компонента, неговата цена и иконка, препращаща към YouTube за възпроизвеждане на видео урок към маркирана част.
- 3. Потребителски профил** – показва профила на потребителя и списъка с неговите проекти и коментари. Има възможност всеки от проектите да бъде изтрит или редактиран, а коментарите да бъдат изтривани.
- 4. Login страница** – стандартен компонент, чрез който регистрираният потребител да може да се впише в приложението със своите потребителско име и парола.
- 5. Register страница** - компонент, чрез който нерегистрираният потребител да може да се регистрира в приложението със свои имейл адрес, потребителско име и парола.

2.2. Използвани технологии

Проектът е реализиран с помощта на следните уеб и мултимедийни технологии:

- 1. HTML 5, CSS3 – стилизиране и структуриране на различните компоненти в приложението.
- 2. Angular - client-side framework за създаване на динамични и интерактивни single page уеб приложения.
- 3. Node.js – софтуерна платформа за създаване на уеб сървър и back-end частта на приложението.
- 4. Express.js – Node.js framework, улесняващ разработката на сървърната част.
- 4. PrimeNG – колекция от UI компоненти за Angular.
- 5. GoJS – JavaScript библиотека за създаване на интерактивни диаграми.
- 6. Bootstrap 4 – JavaScript framework за разработка на responsive уеб сайтове.
- 7. REST – архитектурен стил, с който се осъществява взаимодействието между сървъра и клиента, базиран на HTTP протокола.
- 8. MongoDB – система за обработване на NoSQL база данни. MongoDB съхранява структурираната информация в JSON формат, която я прави изключително удобна за работа с Node.js.
- 9. Mongoose – ODM (Object Data Modeling) библиотека за MongoDB и Node.js. Използва schema-based модели за по-улеснено записване в MongoDB.

2.3. Описание на back-end (сървърната) част

Back-end частта представлява REST-API, изградено с Node.js, Express и MongoDB база данни, с което всички ресурси на проекта биват достъпвани чрез http-endpoints (routes) и CRUD (Creat, read, update and delete) операции. Клиентът посредством HTTP заявки към даден route извиква изпълнението на функция, изпълняваща CRUD операция към базата и документа, който иска да модифицира. Например <https://localhost:4200/api/projects> изпраща заявка към проху сървъра, работещ на порт 4200, който проксира към <https://localhost:9000/>, където слуша сървърът. Заявката се обработва от middleware, рутиращ към файла с всички api/projects endpoints. В случая се изпълнява read (GET) операция, извличаща всички проекти записани в базата, „опакова ги“ в JSON обект, който накрая добавя в отговора (response) на заявката. Методът изглежда по следния начин:

```
// GET projects list
router.get('/', function (req, res) {
  const db = req.app.locals.db;
  const query = req.query;
  const skip = parseInt(query.page);
  const limit = parseInt(query.limit);
  const collection = db.db('login').collection('projects');
  var options = { "limit": limit ,
    "skip": skip  };
  collection.find({}, options).toArray(
    function (err, projects) {
      if (err) throw err;

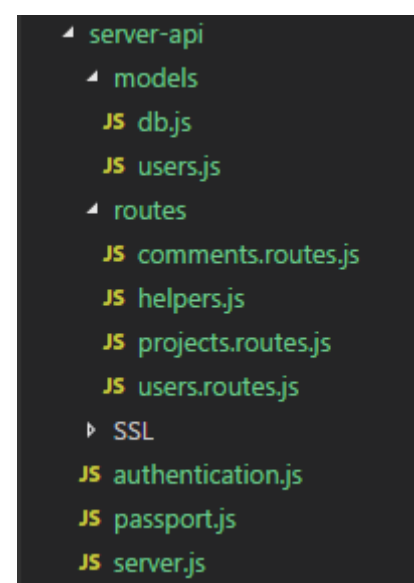
      projects.forEach((project) => replaceId(project));
      res.json({ data: projects });
    });
});
```

Handler функцията приема request (заявката) и response (отговора) обекти и извършва paginating според подадените query параметри в URL-а.

Рутиране

В сървърното приложение са дефинирани три главни route-a, чийто http endpoints са дефинирани в папка routes:

- api/comments – всички endpoints под този URL са дефинирани в routes/comments.routes.js, където са имплементирани заявки към базата за получаване, изтриване и добавяне на коментари под проектите.



Фиг.1 Структура на REST api-то

- `api/projects` – всички заявки се рутират към `projects.routes.js`, където са функциите за обработка на `projects` колекцията в базата.
- `api/users` – рутира към `user.routes.js`, където се намират админските заявки за работа с потребителите – добавяне, изтриване, редактиране.

В `helper` е `export`-ната `replaceId` функция, която преименува `_id` свойството на записите в базата в `id`, улесняващо работата на клиента с получените обекти.

User login and register

За аутентикация на въведените от клиента `credential`-и за влизане в системата се използва `passport.js`. `Passport` е `authentication middleware` за `node.js`, който предоставя множество „стратегии“ – различни техники при процеса на аутентикация на потребители, като логване с `Facebook`, `Twitter`, `Oauth` и т.н. Използваната стратегия е `LocalStrategy`, дефинирана в `passport.js` файла, и се използва за `username/password` аутентикация. `Callback` функцията на `LocalStrategy` приема въведените от потребителя име и парола и търси за съвпадение в базата данни. Комуникацията с базата при логина се извършва с `mongoose`, чиято конфигурация и връзка с `mongodb` се намира в `models/db.js`, и `user schema` в `users.js`, където е описан `User` обектът с три свойста – име, парола и и-мейл, заедно с два метода за проверка на валидност и записване на нова парола в базата. Ако бъде намерен потребител с име, съвпадащо с въведеното от клиента, се проверява за валидността на паролата. Всички потребителски пароли при регистриране преминават през `salt` и хеширане с `bcrypt` модула, след което резултатът от хеширането се записва в базата вместо истинската комбинация. При всяко последващо логване на регистриран потребител въведената парола се хешира по същия метод и се сравнява с хеша, записан в базата. Ако двете хеш стойности съвпадат, то аутентикацията е преминала успешно и `passport.authenticate()` връща `User` обектът на логнатия потребител.

За да се избегне тази проверка при всяка заявка, сървърът връща `JWT (JSON web token)` при успешно логване, който той на свой ред очаква в `body`-то на всяка следваща заявка от страна на потребителя. `JWT` се генерира с `sign` функцията от `jsonwebtoken.js` модула, към която се подават данните, които да се приложат в токена, и `secret key` за неговото кодиране. Всяка последваща заявка бива проверявана за наличие на токен и валидирана с `express-jwt middleware`.


HTTPS

За да е напълно подсигурана комуникацията със сървъра е препоръчително да се използва `HTTPS` протокол, при който връзката е криптирана с `SSL` сертификат, даващ възможност клиент-сървър приложения да комуникират без те да бъдат подслушвани. “Сертификатът обикновено съдържа информация за органите, които са го издали (`Certificate Authority`), името на сървъра и публичния ключ, с който ще се криптира връзката” [1].

За генериране на сертификата се използва `OpenSSL toolkit`. Частният (`private`) ключ и сертификатът се намират в `SSL` папката на апи-то. Те се добавят към променлива `certOptions` заедно с `passphrase` и накрая се подават на сървъра при неговото стартиране:

```
var server = https.createServer(certOptions,app).listen(9000);
```

След инсталирането на сертификата URL-ът на приложението изглежда по следния начин:

 Secure | <https://localhost:4200/sign-in>

Описание на REST API:

- POST /api/sign-in
- POST /api/register
- GET /api/projects?page=X&limit=Y
- GET /api/projects/count
- GET /api/projects/:projectId
- POST /api/projects
- GET /api/:projectId/comments
- GET /api/:projectId/name
- POST /api/projects/comments
- PUT /api/projects/:projectId
- DELETE /api/projects/:projectId
- GET /api/users
- GET /api/users/:userId
- GET /api/:userId/projects
- GET /api/:userId/name
- POST /api/users
- DELETE /api/users/:userId
- GET /api/comments
- GET /api/comments/:userId
- PUT /api/comments/:commentId
- DELETE /api/comments/:commentId

2.4 Начин на работа със системата

При първоначалното влизане в сайта потребителят се навигира към home страницата, където се показва списъкът с публикуваните проекти. В горната част на страницата е поставена лента за навигиране към останалите компоненти на приложението – профил на потребителя и създаване на нов проект при извършено логване в системата, и логин компонент, който освен логин формата за въвеждане на име и парола, съдържа и препратка към страницата за регистрация.

При процеса на регистрация е нужно потребителят да въведе валидни и-мейл адрес, име и парола. При успешно регистриране се визуализира pop-up съобщение, потвърждаващо успешната регистрация, и потребителят автоматично се навигира към логин страницата, откъдето новорегистрираният потребител може да влезе в системата и своя профил.

След успешното логване потребителят вече може да създава проекти, да коментари свои и чужди, и да разглежда своя профил, използвайки лентата за навигация на сайта. При създаването на нов проект потребителят има възможност да въведе заглавие и описание на проекта (задължителни полета), да добавя тагове и Ebay компоненти чрез drag and drop. Всеки създаден проект се запазва в профила на потребителя, където може да бъде редактиран или изтрит от него, и се добавя към общия списък с проекти, наличен на началната страница.

Основна характеристика на приложението е възможността потребителят да създаде интерактивна схема на свързването на компонентите на проекта си. Това става като се натисне бутонът Add component. Това води до отварянето на диалогов прозорец, в който да бъде въведено име, изображение, цвят на компонента, както и имена на входящи и изходящи пинове. След като се потвърди създаването на компонента, той се появява в списъка с активните компоненти към проекта. Генерирането на празна схема с компонентите се извършва с натискане на иконката с гаечен ключ. Тя поставя по една инстанция от всеки създаден компонент в интерактивното поле за чертаене на схема. След това потребителят може да размества компонентите и да създава чрез влачене връзки между входните и изходните пинове.

Списъкът с проекти представлява множество страници с полета, съдържащи име на проекта, авторът и описанието. При кликане на името на проекта се отваря страница, съдържаща всички негови детайли – кога и от кого е въведен проектът, неговите части, описанието, тагове и поле за добавяне на коментар.

След приключването на работата си в сайта, потребителят може да излезе от системата с logout бутонът от навигационната лента. Всяка потребителска сесия трае 24 часа (времето за изтичане на валидността на дадения от сървъра токен), след което е нужно повторно логване в система.

3. Заключение – срещнати/преодолени трудности и перспективи за бъдещо развитие на системата

В настоящия си вид системата предлага функционираща платформа за създаване на библиотека от любителски проекти. Криптирането на данните и използването на HTTPS предлага и известна степен на сигурност при използването ѝ. Разбира се, за да може да се нарече напълно функционална платформа, е необходимо да се добави известен брой допълнителни характеристики.

В момента тя предлага единствено основните CRUD операции за манипулиране на данните. В действителност, за да се гарантира добър user experience, има нужда от още възможности.

- Добавяне на рейтинг към проектите. Това е основна характеристика на всяка информационна платформа, която предлага публикувани от потребители данни. По този начин с времето става възможно да се отсеят качествените проекти от тези с ниска стойност или лоша документация и липсващи технически данни. За тази цел е нужно в базата данни да се направи колекция таблица Ratings, която да съдържа всички оценки на потребител към проект. По този начин се предотвратява възможността един и същи потребител да даде неограничен брой оценки на даден проект.
- Добавяне на рейтинг към резултатите от Ebay. Тъй като всяка част бива предлагана от множество търговци, може да е по-лесно за потребителите да получават с преференция резултати от даден списък с части, които и други потребители вече са ползвали и са доволни от качеството, цената и времето за доставка.
- Добавяне на потребител от тип администратор. Това би улеснило много изтриването или коригирането на дадена информация в приложението. Той ще може да използва платформата като всеки останал потребител, но ще има допълнителни правомощия да манипулира съдържанието. Така ще може по бърз и ефективен начин да бъдат изтривани нежелани коментари, проекти или потребители.
- Добавяне на верификация на имейл адресът на регистриращ се потребител. Това става като акаунтът се активира чак след натискането на линк, изпратен му на адреса, който е посочил за свой.
- Добавяне на интерактивна схема на сглобяването на проекта, в случай, че има нужда от такова. Това би включило не само свързването на електрониката, но и сглобяването на механични части.
- По-прецизно показване на списъци с видео уроци, така че да се избегнат несъществени резултати.

4. Използвани източници

- [1] Какво е SSL сертификат - <https://www.icn.bg/bg/help/ssl-sertifikati/kakvo-e-ssl-sertifikat/> -
- [2] Документация на Web API-тата на Ebay – <https://go.developer.ebay.com/api-documentation>
- [3] Сайт на PrimeNG - <https://www.primefaces.org/primeng/#/>
- [4] Search engine за техническа документация – <http://zuken.componentsearchengine.com/>
- [5] Демо на библиотеката GoJS – <https://gojs.net/latest/samples/dataFlow.html>
- [6] Документация за MongoDB - <https://docs.mongodb.com/>