

Софийски университет "Св. Климент Охридски"

Факултет по математика и информатика

Учебна дисциплина „Разработка на клиент-сървър (fullstack) приложения с  
Node.js + Express.js + React.js“

Документация на курсов проект

---

на

Марам Баядсе, фак. № 81479

Име на проекта: ReviewIt

Научен ръководител

гл.ас. Траян Славчев Илиев

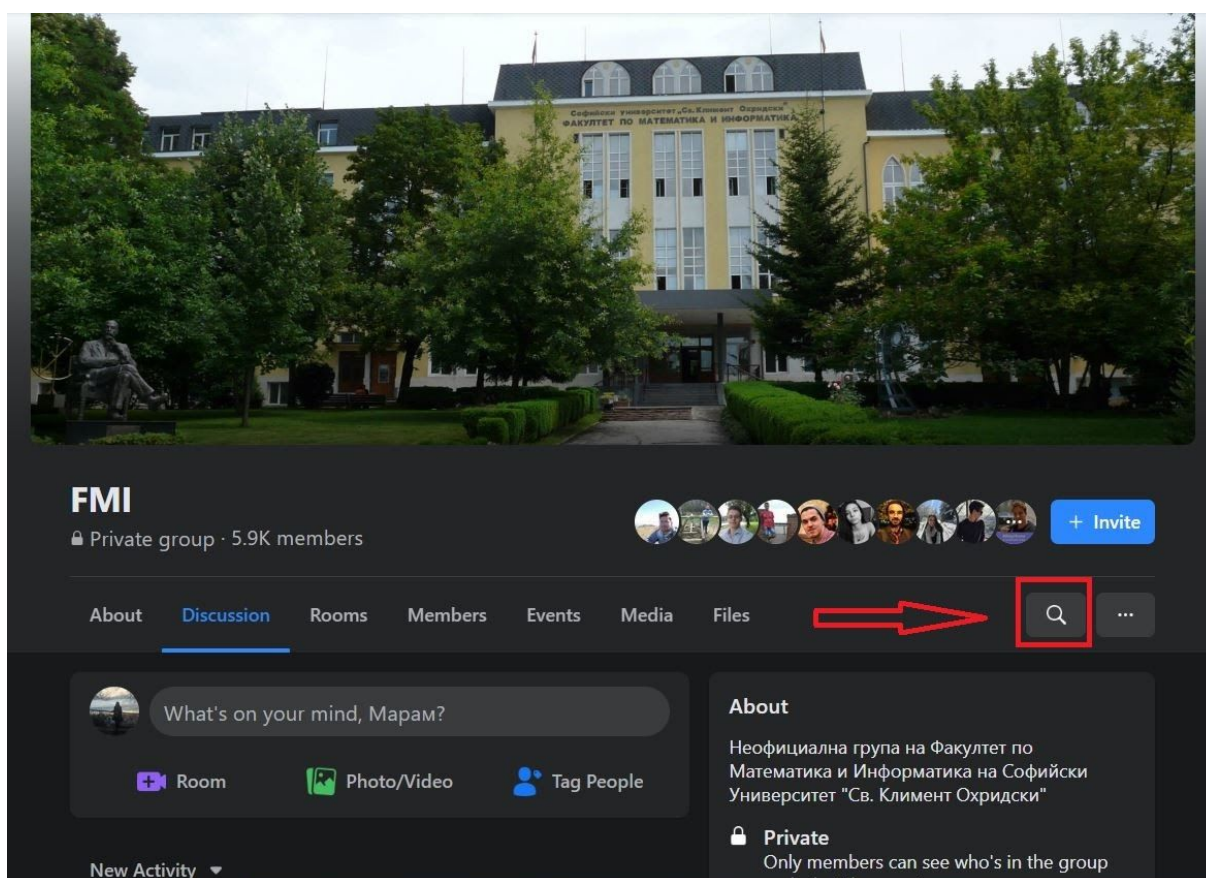
София 2020

## **Съдържание**

Увод	3
Функционални изисквания	4
Нефункционални изисквания	5
Използвани технологии и библиотечни модули	5
Архитектура и реализация	5
Описание на REST service API	7
Инсталиране и конфигуриране - Стъпки:	7
Заключение	8
Източници - документации на използваните библиотеки	8

## 1. Увод

В началото на всеки семестър стане ли дума за записване за избираеми дисциплини, студентите започват да разпитват свои познати за конкретни курсове. Появяват се доста често и публикации в общатата група на факултета в социалната мрежа Facebook с въпроси относно избираеми предмети. Аз лично за всяка избираема дисциплина, от която съм се интересувала, съм търсила допълнителна информация относно курса използвайки известната търсачка в групата на ФМИ във Facebook.



Резултатите от търсенето не винаги са релевантни, а мнението на различните студенти за един курс могат да бъдат много различни. С разработването на системата ReviewIt се цели съхранението на впечатленията и мненията на студентите, свързани с избираемите дисциплини, в една организирана система.

## 2. Функционални изисквания

- **Регистрация:**

Всеки анонимен потребител има възможност за регистрация в системата чрез въвеждане на основна лична информация, включваща: имена, потребителско име и парола.

- **Вход в системата:**

Всеки регистриран потребител има възможност за вход в системата. Това става чрез въвеждане на потребителско име и парола.

- **Разглеждане на курсове:**

Всеки потребител, влязъл в системата, има възможност за разглеждане на списъка от всички налични курсове в системата. Също така има възможността да ги филтрира по категория на курса (математика, практикум, ОКН, ЯКН) и да търси по име. Има възможност и за сортиране по различни критерии.

При избор на даден курс от таблицата с курсове, потребителя вижда по подробна информация за съответния курс, включваща: преподаватели, категория, кредити, и коментари на други потребители за курса.

- **Следене/отказване от следене:**

Всеки потребител, влязъл в системата, има възможност да започне да следи или да се откаже от следене на даден курс.

- **Действия с коментари:**

Всеки потребител, влязъл в системата, има възможност да добави нов коментар към даден курс или да подкрепи или не други коментари. Последното става чрез гласуване с палец, обърнат нагоре или надолу.

- **Предлагане на нови курсове:**

Всеки потребител, влязъл в системата, има възможност да попълни форма с информация за курс. Като така дава свое предложение за допълване на каталога от всички курсове в системата.

- **Разглеждане на предложенията за курсове:**

Всеки потребител с админска роля, влязъл в системата, има възможност за разглеждане на списъка от предложените курсове в системата. Има опции за изтриване на предложението или за одобряването му чрез редактиране и/или допълване на информацията към него.

- **Действия с предложение**

Всеки потребител с админска роля има възможност за одобряване или изтриване на предложение от предложените курсове. След одобряване на предложението се създава курс в каталога от курсовете, видими за потребителите.

### 3. Нефункционални изисквания

- **Достъпност:**

Системата трябва да бъде достъпна за всички - всеки анонимен потребител да има възможност за регистрация и използването и.

- **Лесна използваемост:**

Използването на системата трябва да бъде интуитивно. Видимите компоненти в системата трябва ясно да обозначават каква роля изпълняват.

- **Съвместимост:**

Системата трябва да се поддържа от всички налични браузъри на стационарни и преносими компютри.

- **Изпълнение:**

Системата трябва бързо да реагира на събитията предизвикани от потребителите.

### 4. Използвани технологии и библиотечни модули

- За фронтенд частта:

- [ReactJS](#)
- [React Router](#)
- [Redux](#)
- [Formik](#) & [Yup](#)
- [Material-UI](#)

- За бекенд частта:

- [Express](#)
- [MongoDB](#)

### 5. Архитектура и реализация

- Основните единици в реализация на фронтенда са компоненти, контейнери и стор (redux-store).
  - Компонентите получават информация от контейнерите и имат ролята да ги визуализират. Те нямат пряк достъп до стора - не могат нито да извличат данни, нито да модифицират пряко от него.
  - Контейнерите служат като връзка между компонентите и данните. Те извличат необходимата на компонента информация от стора и ги предават на компонентите. Също така представят междинни функции за модификация на данните в него.
  - Стора служи за съхранение на информацията в рамките на една браузър сесия. Чрез използването му се гарантира наличието на единен източник на информация в цялото приложение. Стора съдържа няколко подразделения - по едно за Курсове, за Предложения за курсове и за Потребители. Всяко

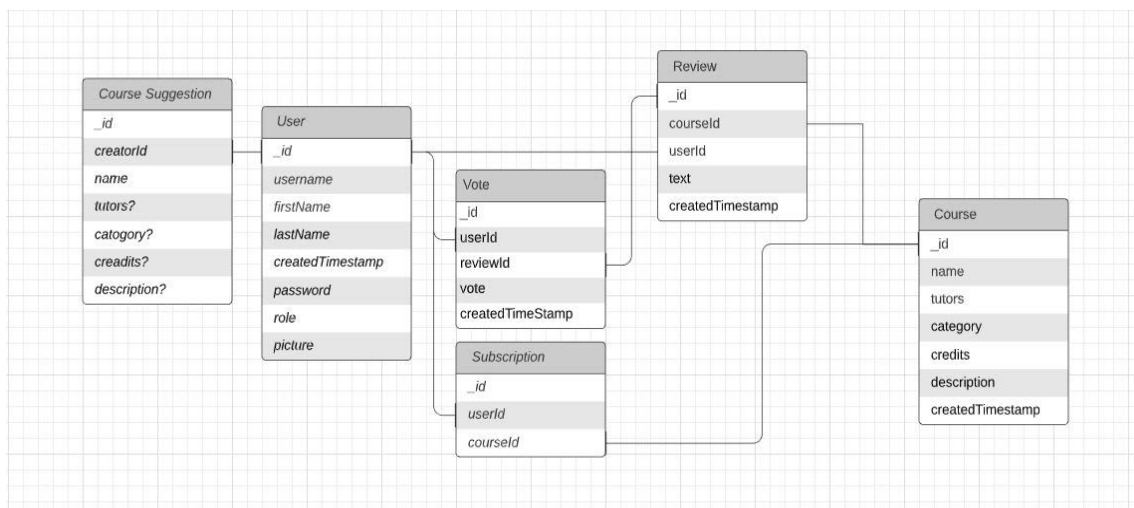
подразделение се грижи информацията на данните, които са свързани с него. Да я предоставя и да я модифицира.

- Маршрутизация

- Всички страници в приложението освен началната, тази за регистрация и тази за вход, не са достъпни за анонимни потребители. Това е осъществено чрез реализацията на защитени маршрути, които се намират на по високо ниво в йерархията.
- Съществуват страници в приложението които са достъпни само за потребители с админска роля. Това също е реализирано на по-високо ниво чрез въвеждане на **AdminProtectedRoute**, в който се случва пренасочването на потребителите, които не са с админска роля към други страници.

- Бекенда е имплементиран като **REST/JSON API** използвайки сериализация на данните в **JSON** формат.

- Единици в базата данни и връзките между тях. Схема.



- Потребител: при регистрация в системата се създава нов запис в БД в таблица **Users**.
- Предложение за курс: всеки потребител, влязъл в системата, има възможност да създаде предложение за курс. Предложението се записва в таблицата **Course Suggestion**. Записа съдържа полето **creatorId**, чрез което се свързва потребител със създадените от него предложения за курсове.
- Курсове: одобрените от админа предложения се записват в базата от данни като запис в таблицата **Course**.
- Следване: Всеки потребител може да запозне да следи даден курс. При това се създава запис в таблицата **Subscription**, съдържащ идентификатор на потребителя и този на курса. При отказване от следването, този запис се изтрива.
- Коментари: Всеки потребител може да добавя коментари към даден курс. Те се записват в таблицата **Review**. Съдържат полетата за идентификатор на

курса към който принадлежат и идентификатор на потребителя, който е направил ревюто,

- Гласове: Всеки потребител може да гласува в подкрепа или срещу даден коментар, При това се записва нов запис в таблицата *Vote*, съдържащ идентификатор на потребителя, на коментара и стойността “за”/”против”.

## 6. Описание на REST service API

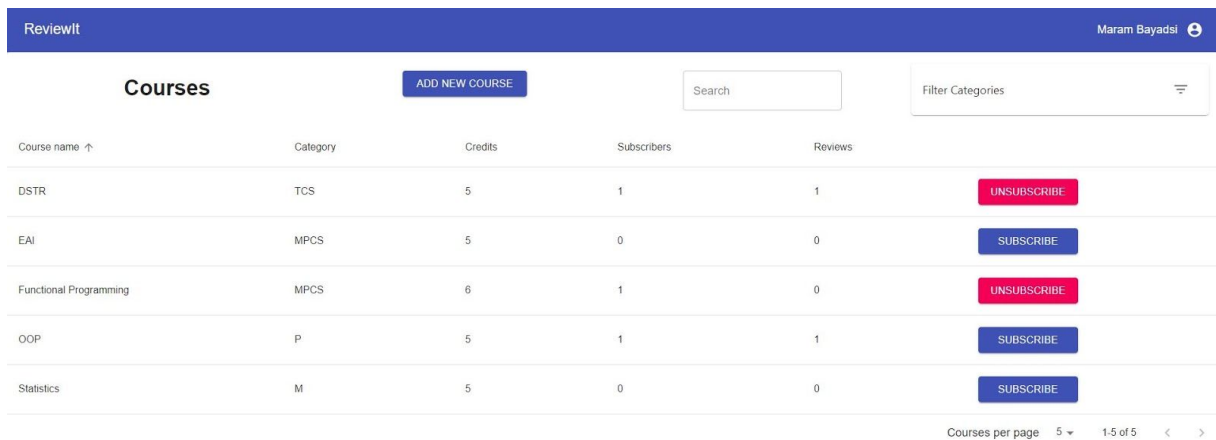
- **Users - /api/users**
  - GET User Data for all users
  - POST new User Data
- **User - /api/users/{userId}**
  - GET, PUT, DELETE User Data for User with specified userId
- **Login - /api/login**
  - POST User Credentials - username and password
- **Courses - /api/courses**
  - GET all available courses on the system;
  - POST new course - only administrator
- **Course suggestion - /api/course-suggestions**
  - GET all course suggestions - admin only
  - POST course information.
- **Reviews - /api/courses/{courseId}/reviews**
  - POST new review;
  - GET all reviews;
- **Vote like/dislike - /api/courses/{courseId}/reviews/{reviewId}/votes**
  - POST - add a new Vote

## 7. Инсталиране и конфигуриране - Стъпки:

1. cd review
2. yarn
3. yarn start
4. cd ../review-backend
5. yarn
6. yarn start

## 8. Заключение

Всички функционални изисквания с изключение на едно бяха изпълнени. Снимка на крайния резултат:



The screenshot shows the 'ReviewIt' application interface. At the top, there is a blue header with the text 'ReviewIt' on the left and 'Maram Bayadsi' with a user icon on the right. Below the header, there is a section titled 'Courses' with a blue button 'ADD NEW COURSE' and a search bar. A 'Filter Categories' dropdown is also visible. The main content is a table with the following columns: 'Course name', 'Category', 'Credits', 'Subscribers', and 'Reviews'. The table contains five rows of course data, each with a corresponding 'SUBSCRIBE' or 'UNSUBSCRIBE' button. At the bottom right, there is a pagination control showing 'Courses per page: 5' and '1.5 of 5'.

Course name	Category	Credits	Subscribers	Reviews	
DSTR	TCS	5	1	1	UNSUBSCRIBE
EAI	MPCS	5	0	0	SUBSCRIBE
Functional Programming	MPCS	6	1	0	UNSUBSCRIBE
OOP	P	5	1	1	SUBSCRIBE
Statistics	M	5	0	0	SUBSCRIBE

Възможност за бъдещо развитие: имплементиране на функционалност за изпращане на известия чрез комуникация в реално време между клиента и сървъра. Това беше предвидено като част от този проект, но осъществяването на това функционално изискване не бе имплементирано.

## 9. Източници - документации на използваните библиотеки

- <https://redux.js.org/recipes/configuring-your-store>, последно посетен на 24.07.2020г.
- <https://reactjs.org/>, последно посетен на 24.07.2020г.
- <https://material-ui.com/getting-started/installation/>, последно посетен на 24.07.2020 г.
- <https://reactrouter.com/web/guides/quick-start>, последно посетен на 24.07.2020 г.