Assignment 1	Project Summary
Course	Fullstack Application Development with Node.js + Express.js + React.js - 2020

Project author		
№	Pseudonym	Face-to-face/ online
1	Eva	face-to-face
2	Pesho	face-to-face

Project name	BookRater

1. Short project description (Business needs and system features)

We have all heard of IMDb (Internet Movie Database). Well, if you haven’t heard of it, basically this is an online database of movies, where you can rate, give review and be up to date with the newest movies. Wouldn't it be great if there was such an application for books? The aim of this project is to develop an application, called ***BookRater*** which will provide the opportunity to review and rate books, as well as keeping a reading list of your favourite books, be up to date with the newest ones and etc. People will have the ability to register and also to explore as an anonymous users. The system will be developed as a *Single Page Application (SPA)* using ***React.js*** as front-end, and ***Node.js + express*** as backend technologies. Each view will have a distinct URL, and the routing between pages will be done client side using ***React Router***. The backend will be implemented as a ***REST/JSON API*** using JSON data serialization.

The main user roles (actors in UML) are:

- *Anonymous User* – can only view the information pages, read book catalogues
- *Author* (extends *Registered User*) – can upload new books to the system, view the information pages, read book catalogues
- *Reader* (extends *Registered User*) – can rate books, keep a reading list, view the information pages, read book catalogues
- *Administrator* (extends *Registered User*) – can manage (create, edit user data and delete) all *Registered Users*, as well as *books, reading lists.*

2. Main Use Cases / Scenarios

Use case name	Brief Descriptions	Actors Involved
2.1. Browse information, read book catalogues, and articles	The *User* can browse the information views (Home, Catalogue, Book, About) in BookRater.	All users
2.2. Register	*Anonymous User* can register in the system by providing a valid e-mail address, first and last name, and choosing password. New registered users can choose to be *Readers* or *Authors*. *Administrator* can register new by entering *User Data* and choosing a Role (*Student, Instructor*, or *Administrator*) or approve become an *Author* requests.	*Anonymous User, Reader, Administrator*
2.3. Change User Data	*Registered User* can view and edit her personal *User Data*. *Administrator* can view and edit *User Data* of all *Users* and assign them *Roles*: *Reader, Author*, or *Administrator*.	*Registered User, Administrator*
2.4. Rate a book	*Reader* can browse books, and give them ratings.	*Reader*
2.5. Add review	*Reader* can add a text review for a book. *Administrator* can edit, or delete *Readers* reviews.	*Instructor, Administrator*
2.6. Add a book to reading list	*Readers* can add a book to their reading lists which can be reviewed at later point.	*Reader*
2.7. Become an Author	*Reader* can request to become an *Author* by providing the necessary information. *Administrator* can approve "Become an Author" requests	*Reader, Administrator*
2.8. Add a book to catalogue	*Author* can add a new book to the catalogue to be reviewed by *Reader*. *Administrator* can edit the book catalogue.	*Author, Administrator*

3. Main Views (SPA Frontend)		
View name	**Brief Descriptions**	**URI**
3.1. Home	Presents the introductory information for the purpose of the system.	/
3.2. Book Catalogue	Presents the book catalogue, containing all of the books in the application.	*/catalogue*
3.3. Book	Presents a concrete book, with detailed information about it and reviews and ratings.	*/book*
3.4. User Registration	Presents a view allowing the *Anonymous Users* to register in BookRater as a *Reader* or an *Author*.	*/register*
3.5. Login	Presents a view allowing the users to login.	*/login*
3.6. User Data	Presents ability to view and edit personal *User Data* and an opportunity for *Reader* to *Become Author*.	*/personal*
3.7. Reading list	Presents ability to browse, add and edit books in the personal reading list.	*/reading-list*
3.8. Users	Presents ability to manage (CRUD) Users and their User Data (available for *Administrators* only, as described in UCs).	*/users*
3.9. About	Presents information about the *OKTS* project and his owner.	*/about*

4. API Resources (Node.js Backend)		
View name	**Brief Descriptions**	**URI**
4.1. Users	GET *User Data* for all users, and POST new *User Data* (Id is auto-filled by *BookRate* and modified entity is returned as result from POST request). Available only for *Administrators*.	*/api/users*

4.2. User	GET, PUT, DELETE *User Data* for *User* with specified *userId*, according to restrictions decribed in UCs.	*/api/users/{userId}*
4.3. Login	POST *User Credentials* (e-mail address and password) and receive a valid *Security Token* to use in subsequent API requests.	*/api/login*
4.4. Logout	POST a logout request for ending the active session with *BookRater,* and invalidating the issued *Security Token*.	*/api/logout*
4.5. Book catalogue	GET Book catalogue, and POST new *Book*(Id is auto-filled by *BookRater* and modified entity is returned as result from POST request), according to *User's Role* and identity security restrictions.	*/api/catalogue*
4.6. Book	GET, PUT, DELETE *Book* with specified *bookid*.	*/api/books/{bookid}*
4.7. Reading list	GET *Books*(according to *Reader’s* identity) and PUT new *Reading list* (modified entity is returned as result from PUT request).	*/api/reading-list/{userId}*