Assignment 1	Project Summary
Course	Fullstack Application Development with Node.js + Express.js + React.js - 2018

Project author			
№	First name, last name	E-mail	Face-to-face/ online
1	Simona Tsenova	simona.tsenova17@gmail.com	face-to-face
2	Yoana Georgieva	yoanageorgieva96@gmail.com	face-to-face

Project name	BooksHub

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

Nowadays, in our highly dynamic everyday life, we have less and less time to spend in a library or a bookstore. But that doesn't necessarily mean we can't browse thousands of books right from our coach or office, or while in the bus. **BooksHub** can be your very personal library, keeping collections of the books you have read, want to read or are currently reading. With **BooksHub** stores detailed information about thousands of books - summaries, reviews, ratings, and their authors. In **BooksHub** users can create and join events and take part in discussions with other passionate readers.

The main user roles (actors in UML) are:

- *Anonymous User* – can search books by different criteria and view the details pages about books; can register in the system.
- *Reader* (extends *Registered User*) – can create and join events, write reviews, rate books, mark book as read, want to read or currently reading; can follow other readers and recommend books.

- *Administrator* (extends *Registered User*) – can create and edit *events* and *books*; can manage users in the system and their access rights.

2. Main Use Cases / Scenarios		
Use case name	**Brief Descriptions**	**Actors Involved**
2.1. Browse information about books	All users can view details pages of books added in the system.	All users
2.2. Register	Each Anonymous User can create new account by providing a username, an e-mail address, a first and a last name, and a password. By registering the user will get access to more of the functionalities of the system.	Anonymous user
2.3. Add/Edit/Delete Book	Administrator can manage books by getting a list of all existing books, adding new ones, and editing and deleting already existing ones.	Administrator
2.4. Make friendship	Readers can add new friends (must be Readers as well) to their friends' list by sending them a friendship invitation. Each reader can browse their pending invitations and choose to accept or decline them. Two people become friends only of the receiver of the invitation has accepted it.	Reader
2.5. Join Event	Readers can join events and be part of the community of people attending or be interested of it.	Reader
2.6. Rate Book	Reader can rate any existing book by giving it stars between 1 and 5. Rating a book by a certain user is allowed only once.	Reader

2.8. Write Review	Readers can express their opinion about a book by writing a review on the book's page. Each reader can publish only one review for a certain book.	*Reader*
2.9. Comment Event	Readers can discuss events by leaving a comment on the event's page. They can ask questions, make propositions or just state their opinion about the event. Each reader can post multiple comments on a single event's page.	*Reader*
2.10. Recommend Book	Readers can recommend books to everyone from their friends' list and view the books their friends have recommended to them.	*Reader*
2.11. Add/Edit/Delete Event	Registered user can create a new event and they must provide at least a name, a location, a starting time and a maximum number of guests for the event. Registered users can manage all the events they have created by editing or deleting them. Administrators can edit or delete any event in the system.	*Registered user*
2.12. Manage users	Administrator can view all registered users in the system. Administrator can then manage users by changing their role in the system (reader or admin) or by deleting and restoring their account.	*Administrator*
2.13. Search books	All users can search books by author, title, year published or genre.	*All users*
2.14. Manage books collections	Readers have three special collection of books - books they have read, books they are currently reading and books they want to read. Each Reader can mark a certain book as Read, Want to Read or Currently Reading and by this the book goes to one of the user's respective collections.	*Reader*

2.15. Browse and search events	Readers can search events by name, city, place or date and can view each event's details page.	*Reader*

3. Main Views (SPA Frontend)		
View name	**Brief Descriptions**	**URI**
3.1. Home	Shows top books and offers ability to register or login.	/
3.2. User Data	It offers ability to view, edit personal User Data and send a friend request to other user.	*/users*
3.3. Book Data	It offers ability to view all books, search a book by specified criteria, view details of a book, rate it or leave a review, recommend it to a friend.	*/books*
3.4. Event Data	It offers ability to view all events, search an event by specified criteria, view details of an event and comment an event.	*/events*
3.5. Admin Panel	Shows all users, books, authors and events. Offers ability to create, put, delete a book or an author.	*/admin*
3.6. Author Data	It offers ability to view all authors and view information about an author.	*/authors*

4. API Resources (Node.js Backend)		
View name	**Brief Descriptions**	**URI**
4.1. User	GET, PUT data for *User* with specified *username*.	*/api/users/{username}*
4.2. Add friend	POST friend to *User* with specified *username*.	*/api/users/{username}/friends*

4.3. Send friend request	GET, POST friend *requests* for *User* with specified *username*.	*/api/users/{username}/requests*
4.4. Users	GET all *Users* and POST new *User* (Id is auto-filled and modified entity is returned as result from POST request).	*/api/users*
4.5. Book	GET, PUT, DELETE data for *Book* with specified *bookId.*	*/api/books/{bookId}*
4.6. Write Review	POST *Review* for *Book* with specified *bookId.*	*/api/books/{bookId}/reviews*
4.7. Rate Book	PUT *Rating* of *Book* with specified *bookId.*	*/api/books/{bookId}/rating*
4.8. Recommend Book	POST *Recommendation* about *Book* to *User* with specified *username*.	*/api/users/{username}/recommendations*
4.9. Books	GET *Books* and POST new *Book* (Id is auto-filled and modified entity is returned as result from POST request).	*/api/books*
4.10. Event	GET, PUT data for *Event* with specified *eventId.*	*/api/events/{eventId}*
4.11. Comment Event	POST *Comment* to *Event* with specified *eventId*.	*/api/events/{eventId}/comments*
4.12. Events	GET *Events* and POST new *Event* (Id is auto-filled and modified entity is returned as result from POST request).	*/api/events*
4.13. Join Event	POST *User* to *Event* with specified *eventId*.	*/api/events/{eventId}/users*
4.14. Author	GET, PUT, DELETE data for Author with specified authorId	*/api/author/{authorId}*
4.15. Authors	GET *Authors* and POST new *Author* (Id is auto-filled and modified entity is returned as result from POST request).	*/api/authors*