|  |  |  |  |
| --- | --- | --- | --- |
| Assignment 1 | | Project Summary | |
| Course | | Fullstack Application Development with Node.js + Express.js + MongoDB + React.js - 2020 | |
|  | | | |
| Project author | | | |
| № | MustHaveFate | | Face-to-face/ online |
| 1 | Instructor | | face-to-face |

|  |  |
| --- | --- |
| Project name | Online Clothing Store |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| The rapid development of the web influenced a lot in our daily lives – the way we communicate, our ability to explore the word .... and the way we shop. The ***Online Clothing Store (OCS)*** provides ability for clients to shop online and for sellers to expand their clientbase. In addition to that it allows users(buyers) to register, and administrators to manage them and add items for sale. The system will be developed as a *Single Page Application (SPA)* using ***React.js*** as front-end, and ***Node.js, express and MongoDb*** 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:  • Unregistered *Client* – can only view items but cannot place orders.  • *Buyer* (extends *Registered User*) – can purchase items and can review the history of his/hers purchase history.  • *Administrator* (extends *Registered User*) – can manage (create, edit user data and delete) all *Registered Users, add items for sale and view the history of their purchases*. |

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| **2.1. Browse items and view descriptions** | The *User* can browse the information views (Home, Items, About) and view specific information about each item. | All users |
| **2.2. Register** | *A buyer can register by entering User Data and choosing a Role (buyer).*  *Administrator* can register by entering *User Data* and choosing a Role (*admin*). | *Byuer,*  *Administrator* |
| **2.3. Change Data** | *Registered Buyer* can view and edit his/hers personal *User Data*.  *Administrator* can view and edit *User Data* of all *Users*. | *Registered Buyer, Administrator* |
| **2.4. Manage *Buyers*** | *Administrator* can browse and filter users based on different criteria: first and last name, email.  *Administrator* can choose a *User* to manage, and can manage the chosen User - edit (using Change User Data) or delete. | *Administrator* |
| **2.5. Manage Orders** | A buyer can order items and has access to the order history.  *Administrator* can dismiss the buyer’s order if necessary. | *Buyer, Administrator* |
| **2.6. Add/Edit Items** | *Administrator can add new items to the store and update item data.* | *Administrator* |
| **2.7. Store purchased items in the cart** | *Every purchased item will be stored in the cart until checkout of the order.* | *Buyer* |
| **2.8. Browse Orders** | *Buyer* can browse his/hers purchase history.  *Administrator* can see if one of his/hers item was puchased and by which buyer as well as have a list of the items added by him/her. | *Buyer, Administrator* |

|  |  |  |
| --- | --- | --- |
| 1. Main Views (SPA Frontend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Home** | Presents the introductory information for the purpose of the system as well as detailed instructions how to start using it. Prominently offers ability to register. | / |
| * 1. **Items** | Presents items available . Offers abilities to browse and view item specifics. | */items* |
| **3.3. My items** | Presents items added by administrator . Offers abilities to create, read, update, delete (CRUD) Items. Available only for Administrators. | */my-items* |
| **3.4. Checkout** | Presents the checkout page, before payment. | */checkout* |
| **3.5. Complete Order** | Presents the complete order information after payment. | */complete-order* |
| **3.6. User Registration** | Presents a view allowing Buyers and Administrators to register to the website. | */register* |
| **3.7. Login** | Presents a view allowing the users to login. | */login* |
| **3.8. User Data** | Presents ability to view and edit personal *User Data*, *as well as keep track of your purchases*. | */personal* |
| **3.9. Users** | Presents ability to manage (CRUD) Users and their User Data (available for *Administrators* only). | */users* |
| **3.10. About** | Presents information about the website and his owner. | */about* |

|  |  |  |
| --- | --- | --- |
| 1. API Resources (Node.js Backend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Users** | GET *User Data* for all users, and POST new *User Data* (Id is auto-filled by *OCS* and modified entity is returned as result from POST request). Available only for *Administrators*. | */api/users* |
| * 1. **User** | GET, PUT, DELETE *User Data* for *User* with specified *userId*. | */api/users/{userId}* |
| * 1. **Login** | POST *User Credentials* (e-mail address and password) and receive a valid *Security Token* to use in subsequent API requests. | */api/login* |
| * 1. **Logout** | POST a logout request for ending the active session with OCS*,* and invalidating the issued *Security Token*. | */api/logout* |
| **4.5. Orders** | GET user orders (according to *User's Role* and identity) | */api/orders* |
| **4.6. Order** | *POST a order from a buyer.* | */api/orders/{buyerid}* |
| **4.7. Order Information** | GET, PUT, DELETE Order *Data* for *Order* with specified *orderid* | */api/orders/{orderid}* |
| **4.8. Items for sell** | *GET the collection of items available.* | */api/items* |
| **4.9. Item** | *POST a item to sell. Available for administrators only.* | */api/my-items* |