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

|  |  |
| --- | --- |
| Project name | Online Knowledge Testing System |

|  |
| --- |
| * Short project description (Business needs and system features) |
| Every office company has the repetitive, boring organisational tasks that people need to do - from ordering food, to making teambuilding plans and poking people to watch their work hours. This is what **AuxBot** is working on improving - an automated bot to do all the boring things with just one command. And this is what the system will help us with - maintain the commands, execute commands and see results real-time without having to read bot’s documentation and learn all the options it offers. 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. There will be also a real-time event streaming from the server to the web client using ***WSS*** and ***Server Sent Events (SSE)/WebSocket*** in order to allow clients to monitor chats and commands real-time. The main user roles (actors in UML) are:  • *Client* (*Registered User*) – can invoke already prepared bot commands  • *Maintainer* (extends *Client*) – can adjust settings and parameters to commands, can make proposals for new commands and functionalities  • *Administrator* (extends *Maintainer*) – can manage (create, edit user data and delete) all *Registered Users*, can stop/start the bot, can move bot between channels and edit it’s permissions, can add/remove commands |

|  |  |  |
| --- | --- | --- |
| * Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * **Browse commands and parameters** | The *User* can browse the information views (Home, Commands, About, Scheduled jobs) in AuxBot, and can choose to *run a command if he has permission for it*. | All users |
| **2.2 Execute commands** | All users can list possible commands. If they have permissions to execute given command - they can fill required parameters and execute it.  Otherwise they can request permission for this command’s execution | All Users |
| * **Register** | *Anonymous User* can register in the system by providing a valid e-mail address, first and last name, and choosing password.  Also users can connect to company’s active directory and log in with the AD account  *Administrator* can register new by entering *User Data* or adding him to the AD | *Anonymous User, Administrator* |
| * **Edit User Permissions** | *Administrator and Maintainer* can view and edit *User Data* of all *Users* and assign them *Permissions for specific commands execution / scheduling*. | *Registered User,*  *Maintainer, Administrator* |
| * **Schedule tasks** | *All maintainers and Administrators can schedule tasks to be done every X hours.*  *Tasks are bot command invocations, with option to read channel history before that and decide parameters based on the history automatically* | *Administrator, Maintainer* |
| * **Manage Commands** | *Administrators can manage commands - remove / disable certain commands.*  *Also they can edit default command parameters, manage channels allowed for commands, add/remove command parameters* | *Administrator* |
| * **Browse Command invocations and results** | *Maintainers and Administrators have access to activity log where they can see who invoked a command and what was the result of it* | *Maintainer, Administrator* |

|  |  |  |
| --- | --- | --- |
| * Main Views (SPA Frontend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * **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. | / |
| * **Commands** | Lists all commands available. Shows which one person has access to execute / read. | */commands* |
| * **Scheduled** | Lists all scheduled tasks, provides interface to remove schedule / schedule new tasks | */scheduled* |
| * **Activity log** | Lists all invoked commands and their results | */activity* |
| * **User Registration** | Presents a view allowing the *Anonymous Users* to register in *AuxBot* | */register* |
| * **Login** | Presents a view allowing the users to login. | */login* |
| * **User Panel** | Presents ability to view and edit personal *User Data*, as well as deregister from *OKTS*. | */user* |
| * **Command Proposal** | Presents interface to make new command request and shortly explain it’s purpose and functionality | */proposal* |
| * **Dashboard** | Presents in real time bot’s status, available channels and running commands | */dashboard* |
| * **Command result** | Watch real-time command action and the result of it | */commands/:invocationId* |
| * **About** | Presents information about the *AuxBot* project and his owner. | */about* |

|  |  |  |
| --- | --- | --- |
| * API Resources (Node.js Backend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * **Users** | GET *User Data* for all users, and POST new *User Data* (Id is auto-filled by *AuxBot* and modified entity is returned as result from POST request). Available only for *Administrators*. | */api/users* |
| * **User** | GET, PUT, DELETE *User Data* for *User* with specified *userId*, according to restrictions decribed in UCs. | */api/users/{userId}* |
| * **Login** | POST *User Credentials* (e-mail address and password) and receive a valid *JWT* to use in subsequent API requests. | */api/login* |
| * **Logout** | POST a logout request for ending the active session with *OKTS,* and invalidating the issued *Security Token*. | */api/logout* |
| * **All Commands** | GET all *Commands* | */api/commands* |
| * **Command** | GET permissions of user for specific command and command description,  POST command invocation  PUT default command parameter adjustments | */api/commands/{commandId}* |
| * **Command invocation** | GET session id for WS real-time watch of command invocation | */api/invocations/{invocationId}* |
| * **Schedule** | GET, PUT, DELETE *a new scheduled task* | */api/schedules/* |