|  |  |  |  |
| --- | --- | --- | --- |
| Assignment 1 | | Project Summary | |
| Course | | Fullstack Application Development with Node.js + Express.js + React.js - 2019/2020 | |
|  | | | |
| Project author | | | |
| № | Pseudonym | | Face-to-face/ online |
| 1 | ivo | | face-to-face |
| 2 | nick | | face-to-face |
|  |  |  |  |

|  |  |
| --- | --- |
| Project name | Event management system |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| The event system allows you to schedule events, specifying various parameters and custom questions that event participants must answer. After creating an event users can invite other people either by email or by sending a link with the invitation form. Invited people fill some personal information and answer the custom questions. The results of this will be presented to the host of the event. If the event requires a ticket (for example concert) users will be able to purchase those through the website. The system will also allow creating virtual meetings as an online chat. The system will be developed as a *Single Page Application (SPA)* using ***React.js*** as front-end, and ***Node.js + express*** as backend technologies.  All users can create events. For each event the main user roles will be:  Host – The creator of the event.  - Update event parameters  - Invite people  - View statistics  - Manage other users’ roles  - Manage payments  - Manage meetings  Moderator  - Invite people  - View statistics  - Fill invitation form  - Manage meetings  Regular user  - Fill invitation form  - View event information  - See event pass (after payment) |

|  |  |  |
| --- | --- | --- |
| 2. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| **2.1.** **Register** | Create an account filling parameters such as first and last name, email, password, avatar. | All users |
| **2.2.** **Login** | Authentication with email and password. | *All users* |
| **2.3.** **Create/Delete event** | Create an event. Some parameters will be mandatory and some optional. The user who creates an event becomes that event’s host.  Deleting an event will notify those who have been invited via email. | *Registered User* |
| **2.4.** **Update event** | Update event data. | *Host* |
| **2.5.** **View event** | Show details about an event. There will be public and private events. Public will be visible for everybody. Private are visible only for invited people. | *All users* |
| **2.6.** **Send invitation** | This can either be done by sending an invitation link, which leads to an invitation form or by sending an email which contains the invitation link. The invitation link will contain a key (random string) so that only people who have received an invitation can view the invitation form. | *Host* |
| **2.7.** **Fill invitation** | People with an invitation link can fill their data and answer the predefined questions. | *All users* |
| **2.8.** **See event statistics** | All people who have submitted their invitation forms will be listed. Their information and answers to the questions will be shown. Also for each question the host or moderator will be able to see the most preferred answer. | *Host, Moderator* |
| **2.9.** **Pay for ticket** | If the event requires buying a ticket, invited people will be able to pay for the ticket by submitting their credit card information | *All users* |
| **2.10 Event pass** | After succesful payment the user will receive an event pass which will be used at the event to verify the user has paid. The event pass will contain basic infoamtnion about the event and a QR code (or some other kind of unique code) | *All users* |
| **2.11.** **See payments transactions** | Hosts will be able to see who has successfully paid for their ticket and verify those users at the event. | *Host* |
| **2.12. Create/Update/Delete meeting** | The meeting will be a chat room, which can be opened / closed by its creator. It could have an optional configuration whether the meeting is for a specific event. The creator will be able to update or delete the meeting. | *All users* |
| **2.13. Organizations** | Apart from users events will also be able to be created by organizations. Examples of an organization are: a company, a night club, a school etc | *All users* |
| **2.14. Feed** | Events and organizations will have the ability to publish posts. This could be used for example to inform users about new events, or changes to existing ones. | *Hosts* |

|  |  |  |
| --- | --- | --- |
| 3. Main Views (SPA Frontend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| **3.1.** **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. | / |
| **3.2.** **Login** | Login using email and password | /login |
| **3.3.** **Register** | Register | /register |
| **3.4.** **Create event** | Contains the form for creating an event. This view will probably have multiple tabs | */event/create* |
| **3.5.** **Update event** | Contains the form for updating an event | */event/{eventId}/edit* |
| **3.6.** **View event** | Here are shown all details for the event. If the event is public this view will be visible by everybody. If the event is private the view will be visible only by people who have been invited. Hosts and moderators will be able to send invitations and publish posts. | */*event/{eventId} |
| **3.7.** **Event statistics** | Here Hosts and Moderators will be able to see event statistics | */event/{eventId}/statistics* |
| **3.8.** **Payment** | A form that contains card billing information fields | */event/{eventId}/payment* |
| **3.9.** **Payments transactions** | Show payment transactions | */event/{eventId}/payment/transactions* |
| **3.10.** **Users** | A view for hosts to manage users who are invited to the event. Hosts will be able to promote or demote users. | */event/{eventId}/users* |
| **3.11.** **Create meeting** | Contains the form for creating a meeting. This view will have multiple tabs | */meeting/create* |
| **3.12.** **Update meeting** | Contains the form for updating an event | */meeting/{meetingId}/edit* |
| **3.13 Meeting dashboard** | This is the view with the chat | */meeting/{meetingId}* |
| **3.14 Create organization** | Create organization form | */organization/create* |
| **3.15 Organization dashboard** | Dashboard for managing an organization: add members, change details, publish posts, view events | */organization/{organizationName}* |
| **3.16. Event pass** | Contains some event details and a QR code | */event/{eventId}/pass* |

|  |  |  |
| --- | --- | --- |
| 4. API Resources (Node.js Backend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| **4.1.** **Register** | POST user info that will be stored and retrieve JWT to store in front end | */api/register* |
| **4.2.** **Login** | POST email and password and retrieve JWT | */api/login* |
| **4.3.** **Create event** | POST event data. Return JSON with validation errors or success message | */api/event* |
| **4.4.** **Event** | GET event with feed, PATCH – to update event data, DELETE event | */api/event/{eventId}* |
| **4.5.** **Submit invitation** | POST Invitation data | */api/event/{eventId}/{invitationKey}* |
| **4.6.** **Statistics** | GET Generate and return event statistics. Request made by user without the necessary permissions will return 403 | */api/event/{eventId}/statistics* |
| **4.7.** **Payment** | POST Card information and create a call to a simulated third party api to perform payment transactions. Return the result. | */api/event/{eventId}/payment* |
| **4.8. Event pass** | GET Event details and QR code png photo | */api/event/{eventId}/pass* |
| **4.9. Meeting** | POST Create meeting with filled details.  GET Generate all meetings the user has created/is invited to  PATCH – to update meeting details, in case the user is creator  DELETE – to delete meeting in case the user is the creator | */api/meeting*  */api/meeting/{meetingId}* |
| **4.10. Meeting message** | POST meeting message and store it | */api/meeting/{meetingId}/messages* |
| **4.11. Create Organization** | POST organization data | */api/organization/create* |
| **4.12. Update, Retreive, Delete organization** | PATCH organization data  GET organization data with feed  DELETE organization | */api/organization/{organizationId}* |
| **4.13. Event feed create** | POST details | */api/event/{eventId}/posts/create* |
| **4.14. Event feed delete** | DELTE post | */api/event/{eventId}/posts/{postId}* |
| **4.15. Organization feed create** | POST details | */api/organization/{organizationId} }/posts/create* |
| **4.16. Organization feed delete** | DELETE post | */api/organization/{organizationId}/posts/{postId}* |