Project name	IT events meetups management system

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

The **IT event meetups management system** provides ability to the hosts manage events. In addition to that it allows users to register and be active attending latest events. The system will be developed as a ***MVC*** using ***Tkinter*** as front-end, and ***JSON file persistence*** technologies.

The main user roles (actors in UML) are:

- *Anonymous User* – can view the information pages and view events and can attend when is registered to the system
- *Administrator*– can manage (create, edit user data and delete) all *Registered Users*, as well as *Events*.
- Registered user– can sign up to the events with their saved data
- *Host* – type of responsible person who manages own created events

2. Main Use Cases / Scenarios

Use case name	Brief Descriptions	Actors Involved
2.1. Browse information sign up for events	The *User* can browse the information views (Home, Events, About) in EMMS, and can choose to sign up for event.	All users
2.2. View	*Anonymous User* can only view events.	*Anonymous User*
2.3. View and enroll	Registered User can view and enroll events. If he is already enrrolled the button for enrolling is not active.	Registered User
2.4. Sign up	*Administrator* can register new by entering *User Data* and choosing a Role (*Registered user, Host*, or *Administrator*). Anonymous user can register account. The assigned role is REGISTERED_USER	*Anonymous User, Registered User*
2.5. Change User Data	*Registered User* can view and edit her personal *User Data*.	*Registered User , Administrator*

	Administrator can view and edit *User Data* of all *Users* and assign them *Roles*: *Registered User, Host*, or *Administrator*.	
2.6. Manage Users	*Administrator* can browse and filter users based on different criteria: first and last name, email, Role. *Administrator* can choose a *User* to manage, and can manage the chosen User - edit (using Change User Data UC) or delete. *Administrator* can create a new user using *Register UC*.	*Administrator*
2.7. Manage Events	*Host* can browse Events, add new *Event* using *Add/Edit Event UC*, and delete a *Event*, as well as view the *Event Responses* for her own *Events*. *Administrator* can browse events of all Hosts, edit and delete them.	*Host, Administrator*
2.8. Add/Edit Event	*Host* or *Administrator can moderate* ***only their events*** specifies/edits *Event* meta-data such as: event name, subject, description, event inviation data can follow responses from participants.	*Host, Administrator*
2.9. Complete Registration Event process	*Registered User can enroll event if event is in status "Open for registrations"*	*Registered User*
2.10. Receive invitation to the participant calendar	*Registered User* can click to add the event with place/link date and time of the event directly to their calendar	*Registered User*

3. Main Views	
View name	**Brief Descriptions**

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. Events	Presents events available according to *User's Role* and identity. Offers abilities to create, read, update, delete (CRUD) *Events*.
3.3. Event Details	Provides ability to enter/edit Event details for individual *Group of people*. *Presents chosen Event and has action button for registration to event.* *Provides ability to browse event responses of the particular event*
3.4. User Data	Presents ability to view and edit personal *User Data*, as well as deregister from *EEMS*.
3.5. Users	Presents ability to manage (CRUD) *Users* and their *User Data* (available for *Administrators* only, as described in UCs).
3.6. About	Presents information about the *EMMS* project and his owner.

4. Domain object description

4.1 All **Users** should have following common attributes:

package UserManagment

- id - (generated automatically) - long number;

- first_name - string 2 to 70 characters long;
- last_name - string 2 to 70 characters long;
- email - should be valid email address, unique within the system, cannot be changed;
- password
- bio - string 2 to 255 characters long;
- is_active – bool default True

4.2 Each **Role** has the following structure: .

package UserManagment

- name -enumeration including "Admin", "Host"," Guest"

4.3 Each **Group** has the following structure:

package GroupManagment

- id - (generated automatically) - long number;
- name - string 2 to 100 characters long;
- description - string 2 to 255 characters long;

4.4 Each **Event** has the following structure:

package EventManagment

- id - (generated automatically) - long number;

- name - name - string 2 to 70 characters long;
- description - string 2 to 255 characters long;
- creation_date - date
- creation_user_id – long number
- registration_end_date – registration end date of the event
- start_datetime – start date of the event
- end_datetime – end date of the event
- place - string 2 to 255 characters long or url format;
- is_public - bool
- capacity - int
- price – float value
- status_id binded with event status

4.5 Each **AllowedEventGroup** has the following structure:

package GroupManagment

- has_a (relationship) event_id long number;
- has_a (relationship) group_id long number;

4.6 Each **EventInvitation** has the following structure:

package EventManagment

- has_a (relationship) event_id long number;
- has_a (relationship) user_id long number;
- sent_date – invitation sent date
- has_a (relationship) invitation_response_id long number;
- text_response - string 2 to 255 characters long
- response_date – invitation response date

4.7 Each **InvitationResponseType** has the following structure:

package EventManagment

- name - enumeration including “Accept”, “Reject”, “”Maybe”

4.8 Each **EventTicket** has the following structure:

package EventManagment

- has_a (relationship) event_id long number;
- has_a (relationship) owner_id long number;
- is_paid – bool default False
- paid_date timestamp

4.9 Each **EventStatus** has the following structure:

package EventManagment

- name - enumeration including “Draft”, “Open to registration”, “ClosedToRegistration”, “Ongoing”, “Past”, “Cancelled”

4.10 Each **EventPost** has the following structure:

package EventManagment

- has_a (relationship) event_id long number;
- text- string 2 to 70 characters long
- creation_date – date of the event post creation
- has_a (relationship) creation_user_id long number – user id of the organizer

5. UML diagram of relationships