Technological Educational Institute
Of Crete

Winter semester 2018/2019

Software Engineering & Big-data modeling

Petra Cendelinova
Petr Kalas

November 22, 2018

Contents

1 Software Analysis

1.1 System Textual Analysis

1.1.1 Textual description

For human beings, it is natural to socialize and interact with each other. Technology has become more and more important in the social aspects of life. Nowadays people prefer to spend more time on portable devices. Social networking has moved to mobile platforms, which are accessible anywhere and anytime.

The objective of our work is to develop a social application for organizing events any type. As events can be considered birthday parties, new year eve, weddings, baby showers etc. As we want to make this application more general for any kind of events, we have decided to call it by name "EventMate". The main goal of the application is to provide tasks management (create, assign, close) and also to create a communication channel among event owners and guests. The application is going to content gamification elements such as a scoreboard and badges.

The following chapter explains major features. In order to use the application it is necessary to create a user's account or log in via existing social accounts. In our application, there are various user roles such as owners, assignees and guests. As we mentioned earlier, the main goal is to provide task management for an event. For all practical purposes, it means that user can create a task with corresponding attributes such as name, deadline, and persons to be assigned to it. All these fields are saved and continuously maintained. During the whole event, gamification principles are applied which help users to feel that they are a part of the game. When the event has finished, event summary is provided to particular users.

Storyboard

In order to understand the meaning of the application storyboard was created. The following storyboard describes one of possible reasons the application usage.

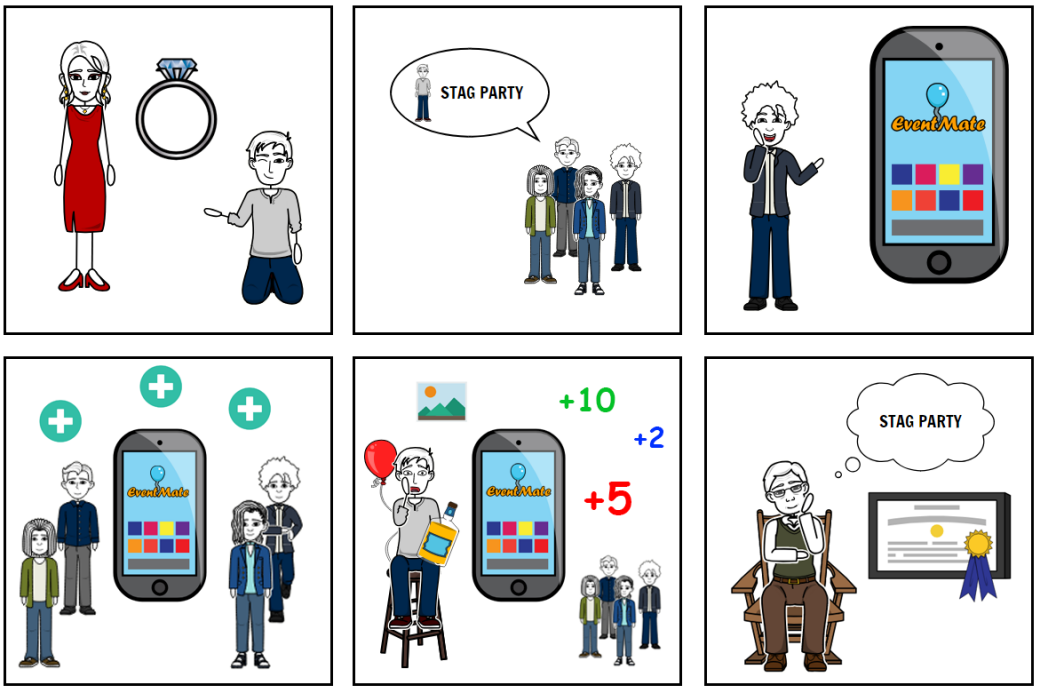

Figure 1: Storyboard - stag party

1.1.2 Requirements list

In order to describe requirements of the system a list of requirements and requirement diagram are used.

Functional requirements

- User registration
- User registration via Facebook / Google account
- User login
- Create events
- Create events from template
- List events
- Filter events
- Modify events
- Delete events
- Change event state
- Create tasks
- List tasks
- Modify tasks
- Change task state
- Submit task results
- Edit task photo
- Assign points for accomplished tasks
- Create reports
- Share reports
- Send private messages
- Send group messages
- Show user profile

- Change their own settings
- Setup notifications

Non-functional requirements

- Supported Android version 6 – 8
- Supported Web version by all browsers
- Responsivity Android frontend
- Android app available in portrait mode
- General usable REST API
- Secured REST API
- Maximum response time of 2 seconds
- Backend platform support
- Multilanguage support
- Password security
- Log rotation ability

1.1.3 Requirement diagram

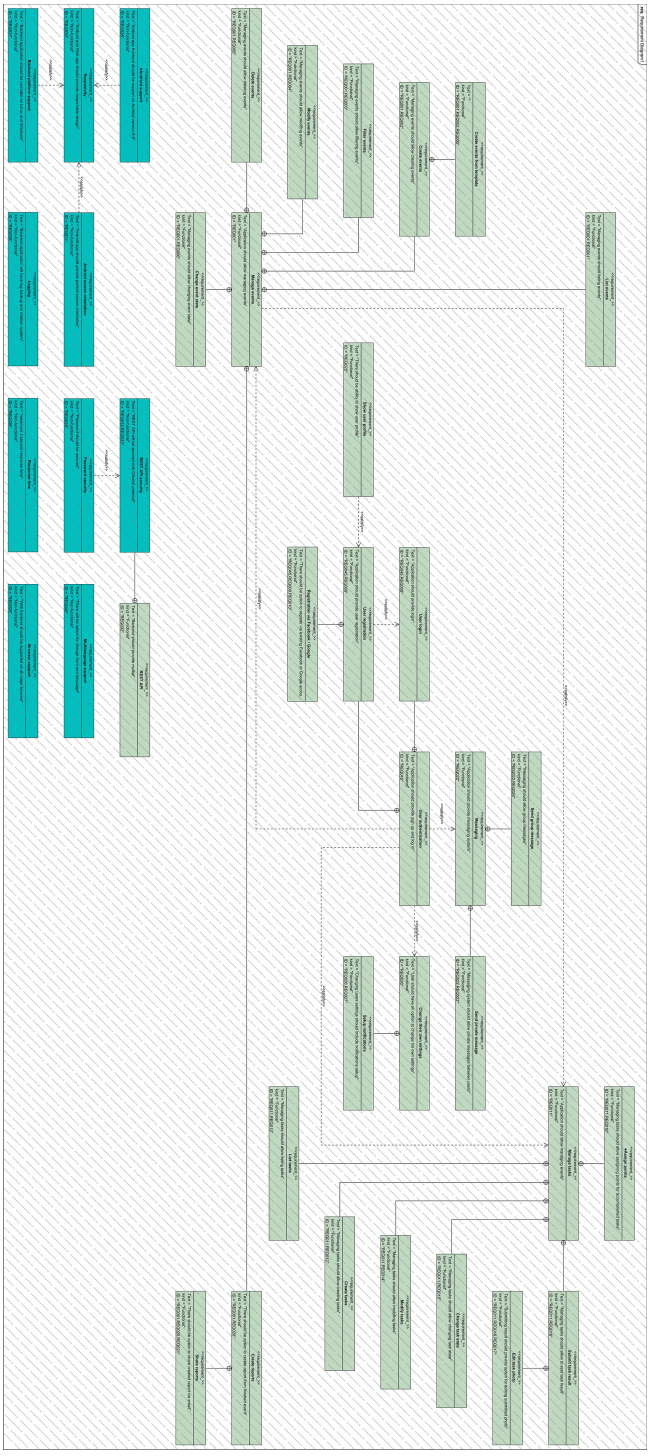

Figure 2: Requirements diagram

1.2 Mind map

In order to visually organize relationships among components mind map was created.

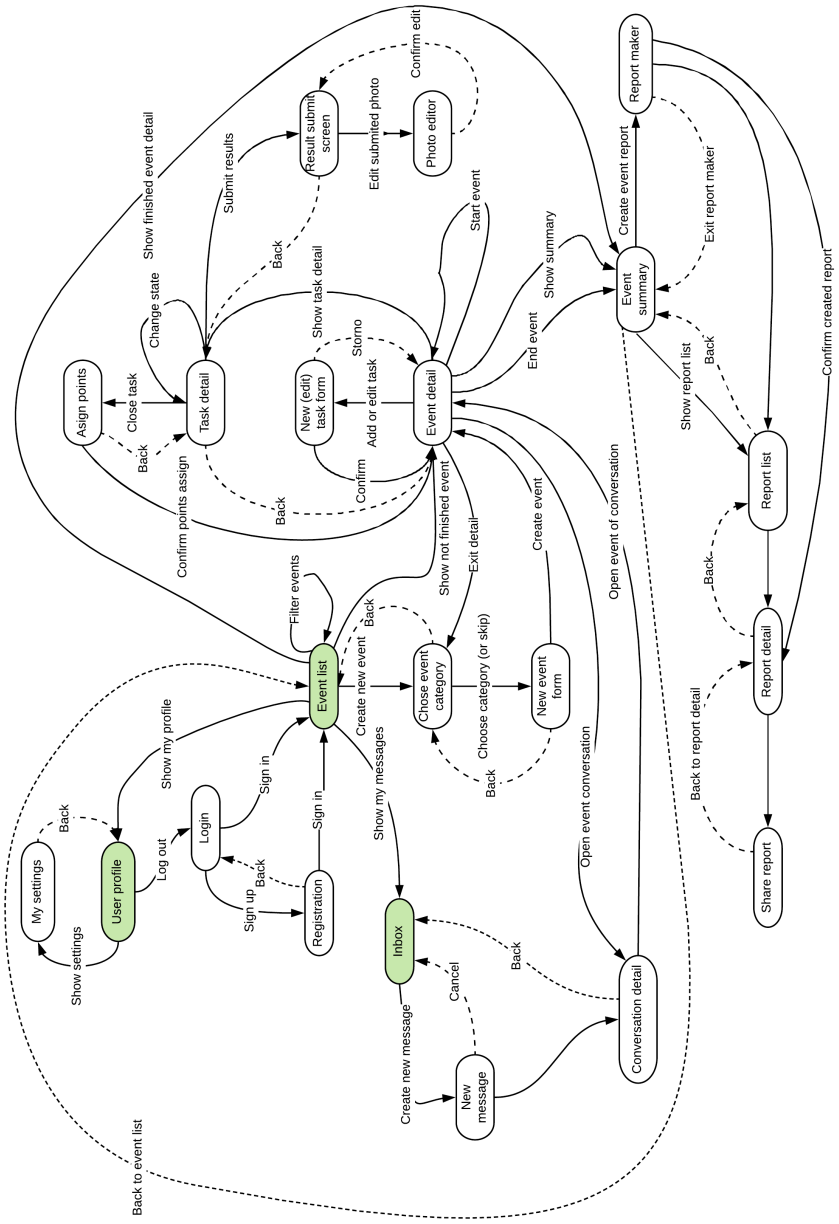

Figure 3: Mind map

1.3 Use case

The following sections describes use case diagram with two actors. The principal use cases are described in detail. Particularly speaking about creating a new event, adding a new task and generating event summary.

1.3.1 List of roles

The application contains 4 user roles with predefined behaviour and permission.

Event owner
Event owner is considered as a user that has created an event. Event creation gives this user all permissions to manage event such as edit, delete, lock, start and close.

Task owner
Task owner is an event guest that has created a task. As its owner has a right to edit, delete, start (in case of time limit task), assign points to assignees and close task. Task owner can also assign his own task to himself.

Assignee
It is a type of user that has been assigned to a particular task. He has a right to upload his answers and view results of others.

Guest
Guest is simply a regular user that has been invited to an event. He has not created any task yet or been assigned to any task. He can view event detail with its tasks.

1.3.2 Use case diagram

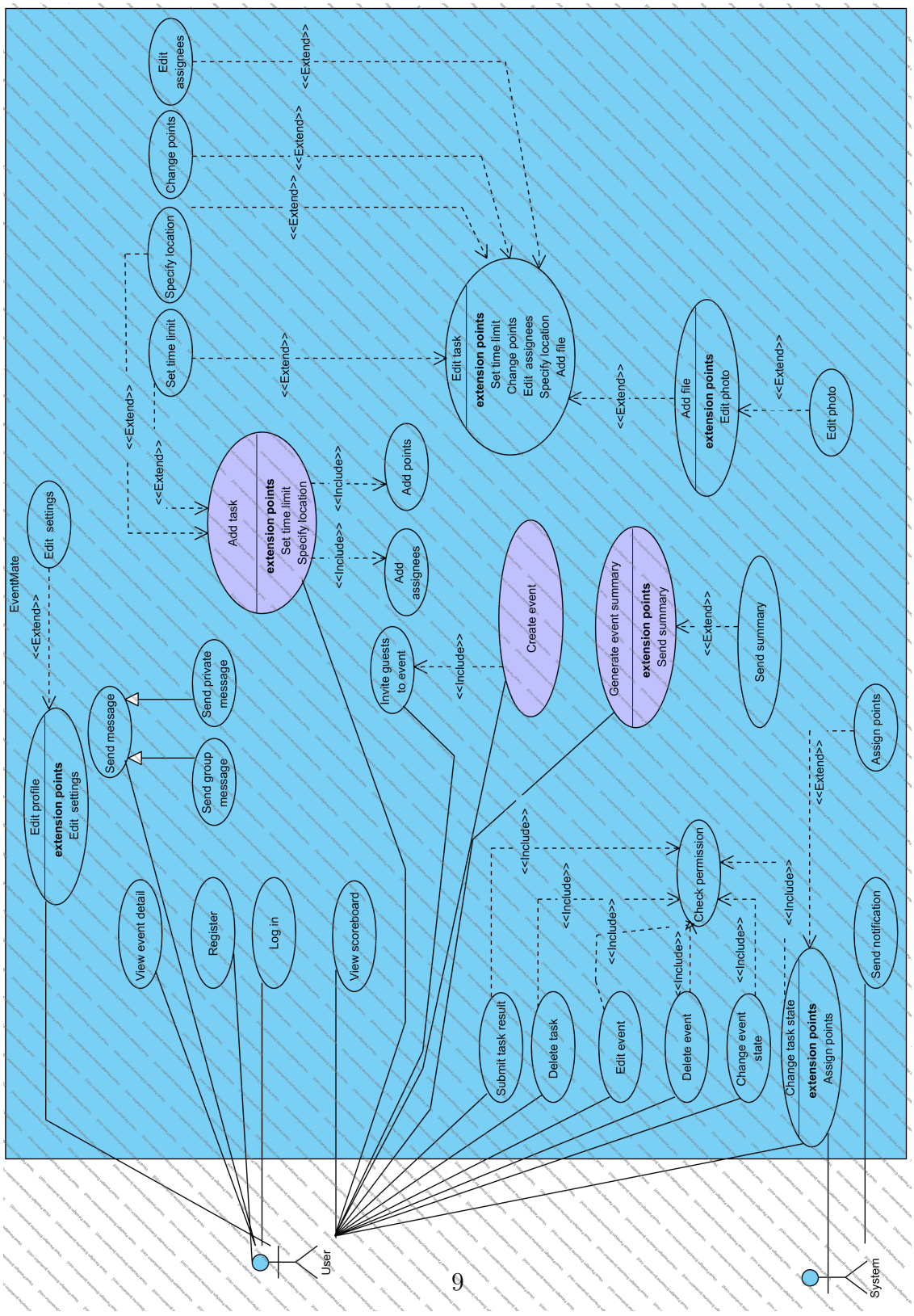

Figure 4: Use case diagram

1.3.3 Textual Description of use cases

This section describes the most important use cases in textual form. Particularly speaking about 'Create event', 'Add task' and 'Create summary'.

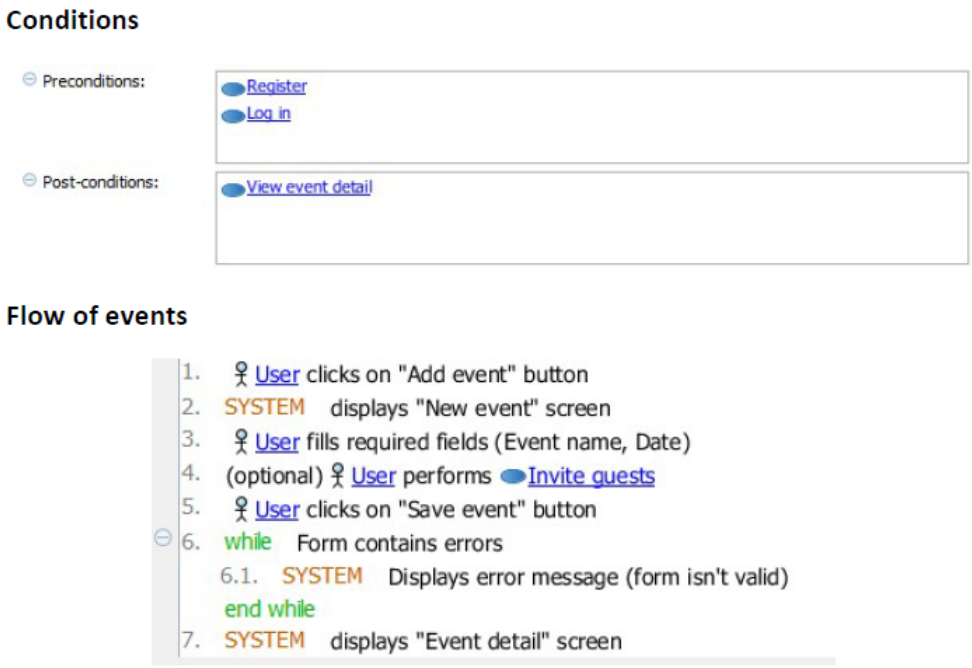

Figure 5: Use case detail - create event

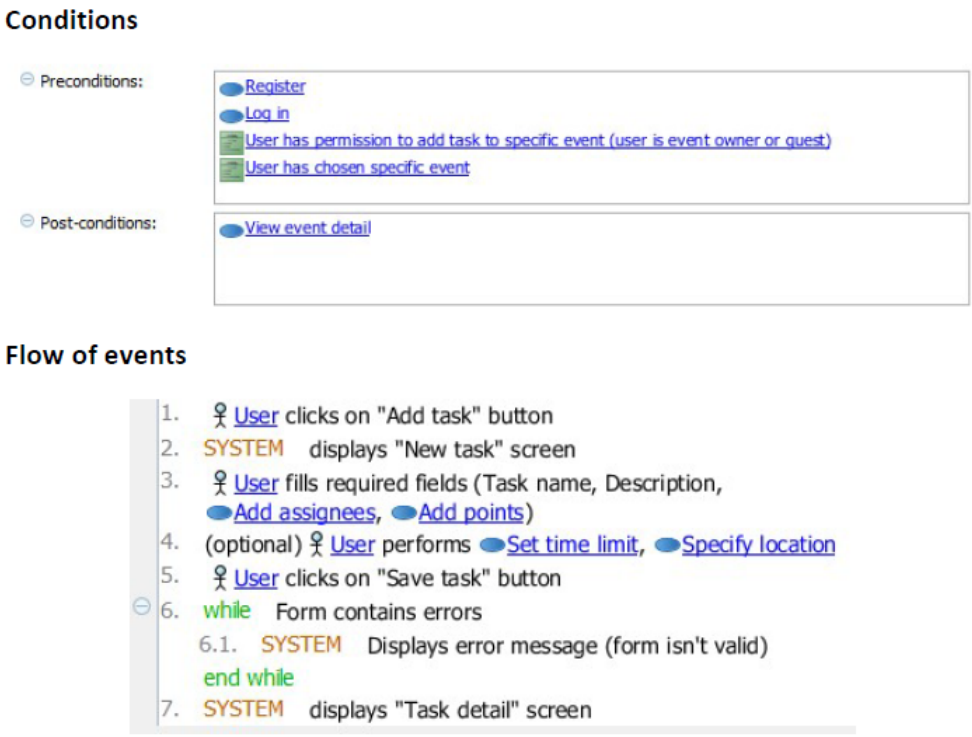

Figure 6: Use case detail - add task

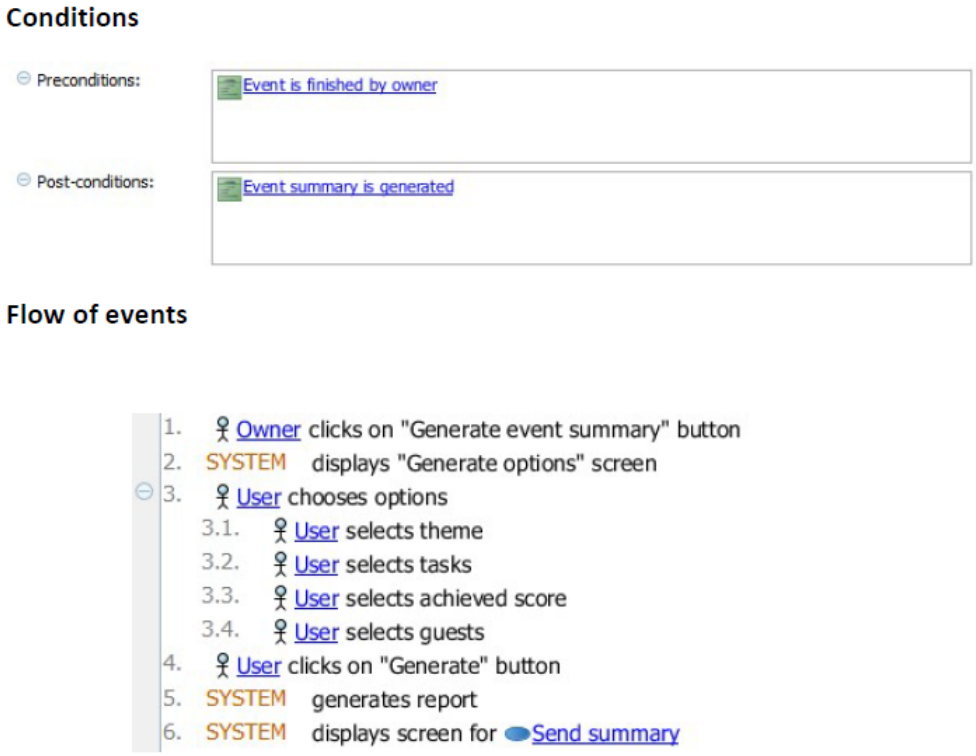

Figure 7: Use case detail - create summary

1.4 State machine diagram

The aim of this section is to clarify possible states of events and tasks using state machine diagram.

Firstly, event states will be presented. Any event can exist in four states. After its creation it is considered as 'editable' state. During this phase any user can add a new task to this event. Later on, the event owner has a right to lock his event to prevent guests from adding new tasks. Afterwards, the event owner is entitled to trigger his own event. The state is known as 'in play'. The event can changed its state to 'finished' when the owner close it or assign points for all tasks.

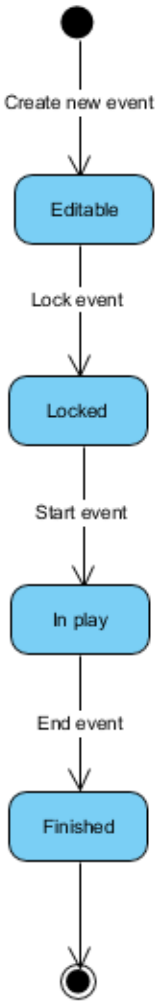

Figure 8: Event state diagram

The application supports two types of tasks. Particularly speaking about tasks without defining time to finish and tasks with specified time limit. After its creation a task becomes editable. When its parent event has been triggered, the task passes to a new state 'In play' which supports result submission for assigned users. After submission of results or stopping the task by its owner, the task becomes only readable known as 'In review' state. Transition to last state requires granting points to assignees done by the task owner.

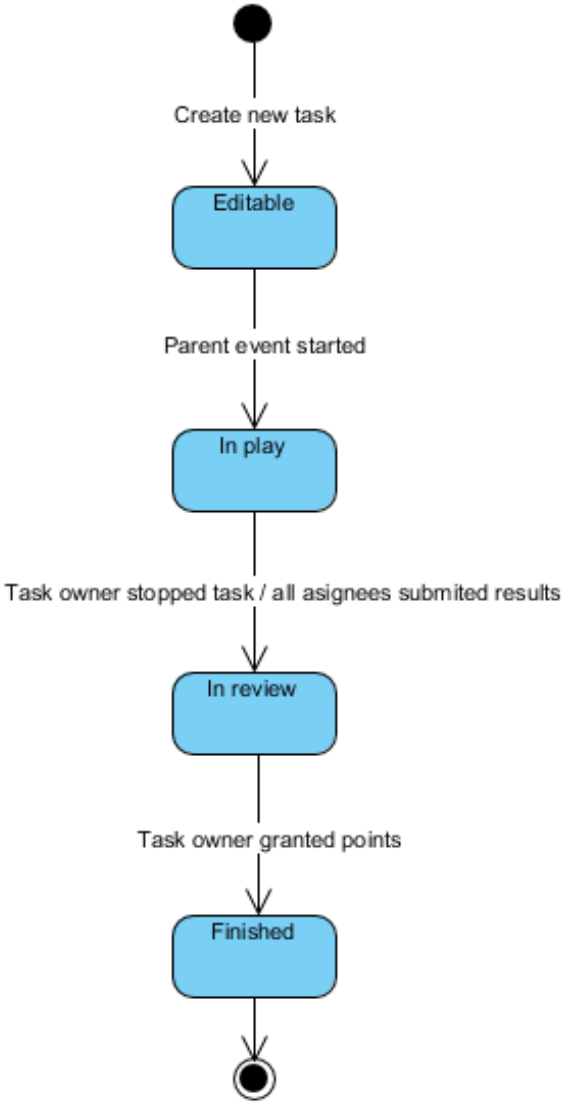

Figure 9: Task state diagram

As figure shown tasks with defined time limit contain one state more named ‘Ready to start’. Transition to next state requires activation by its owner.

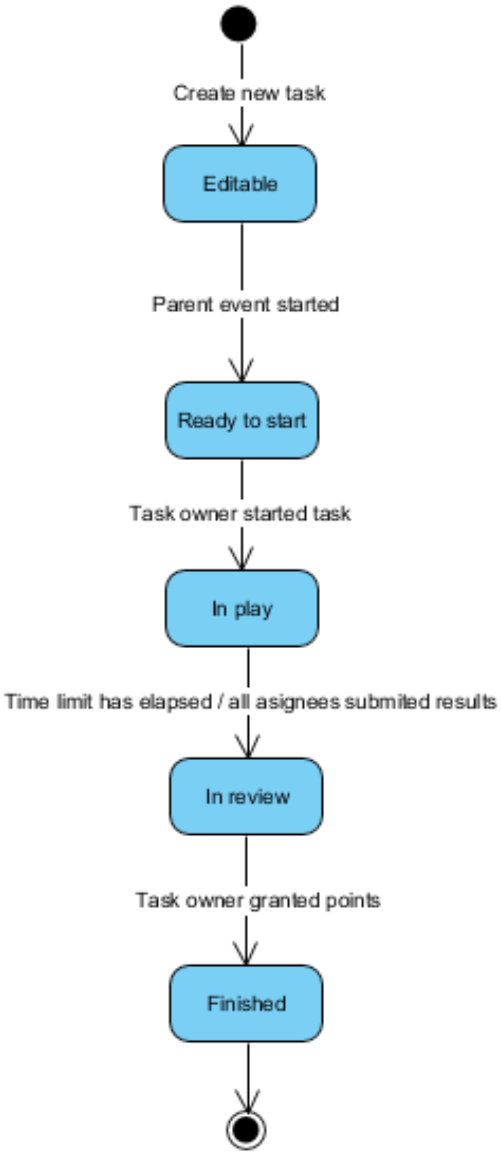

Figure 10: Task state diagram

1.5 Mockups

Mockups were created for scenarios below and they can be found in the following files in the submission folder.

Event owner role (Android) - *scenario_owner_android.pdf*
Event owner role (Web) - *scenario_owner_web.pdf*
Event assignee role (Android) - *scenario_asignee_android.pdf*

2 Software Design

2.1 Architectural Design

Architecture style is based on client-server. The project consists of two client side parts and server side with a database.

There are several reasons of choosing client-server architecture. The first one is need of persistent storage of data. These data should be available to several clients at once. Data should be also consistent during cooperation of many clients. Second reason is operation of more than one kind of clients (specifically mobile application and web application based clients). With client-server architecture, it is possible to create one unified interface that will be usable for both kind of clients.

P2P architecture was not used as there are no clear benefits of using it.

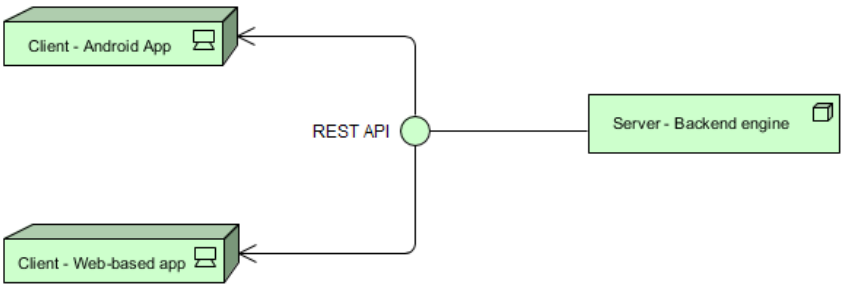

Figure 11: Architecture diagram

2.1.1 Component diagram

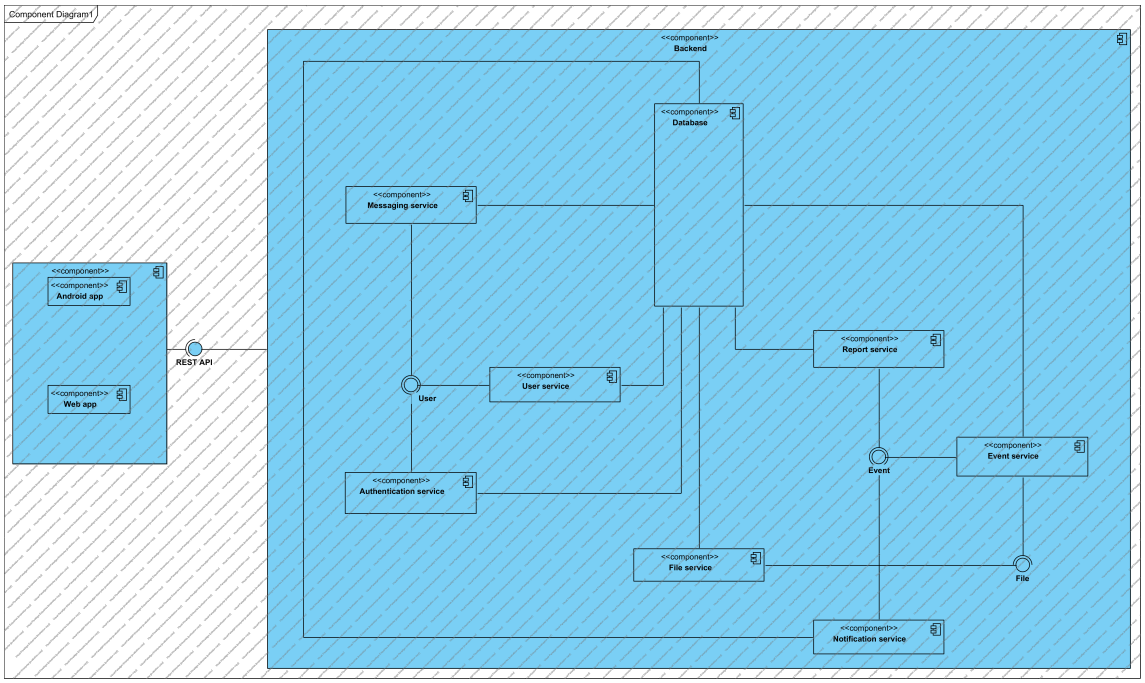

Figure 12: Component diagram

2.2 Class diagram

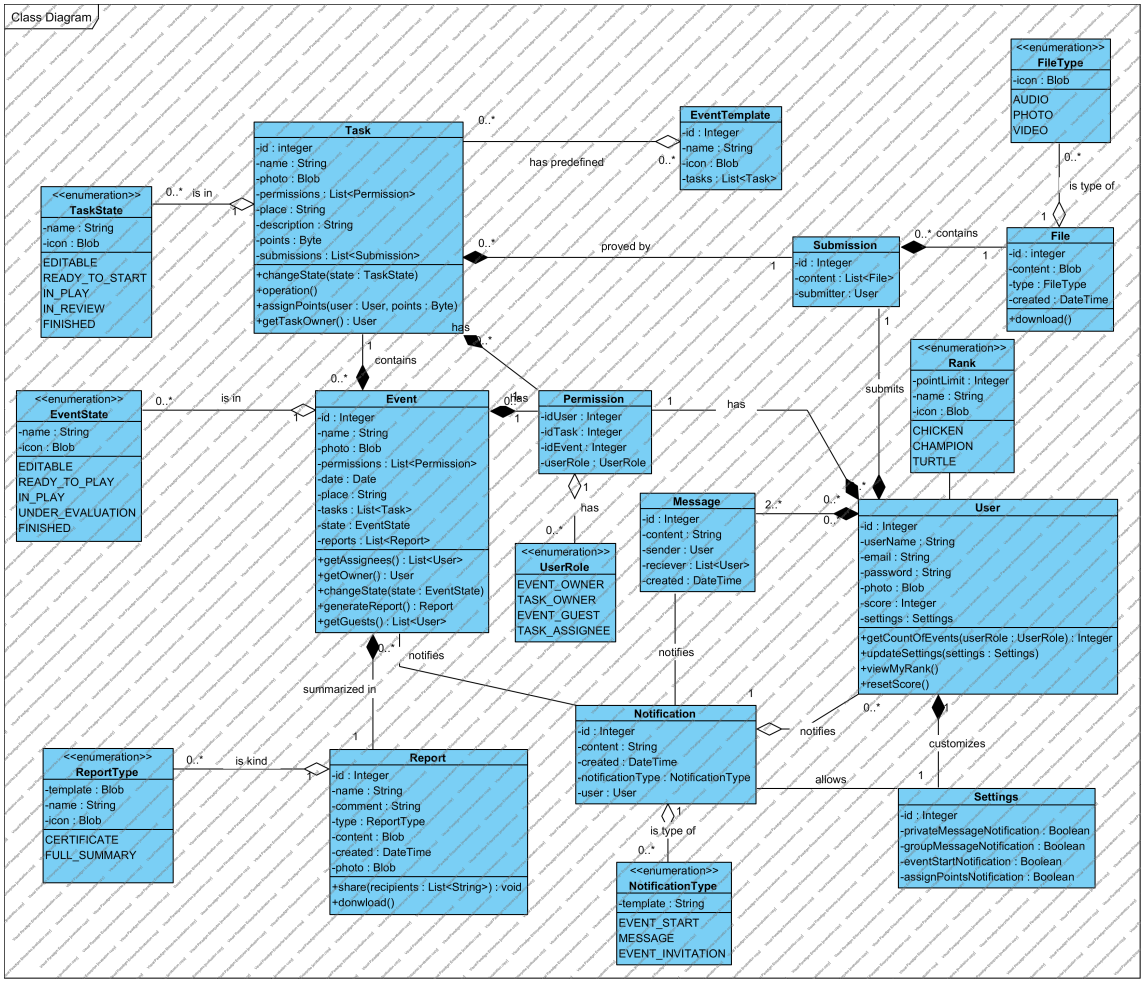

Figure 13: Class diagram

2.3 Sequence diagrams

Sequence d were created for two essential actions of the application, i.e. Create event and create task. This diagram also shows invitation phase that as result invokes sending notifications to invited guests.

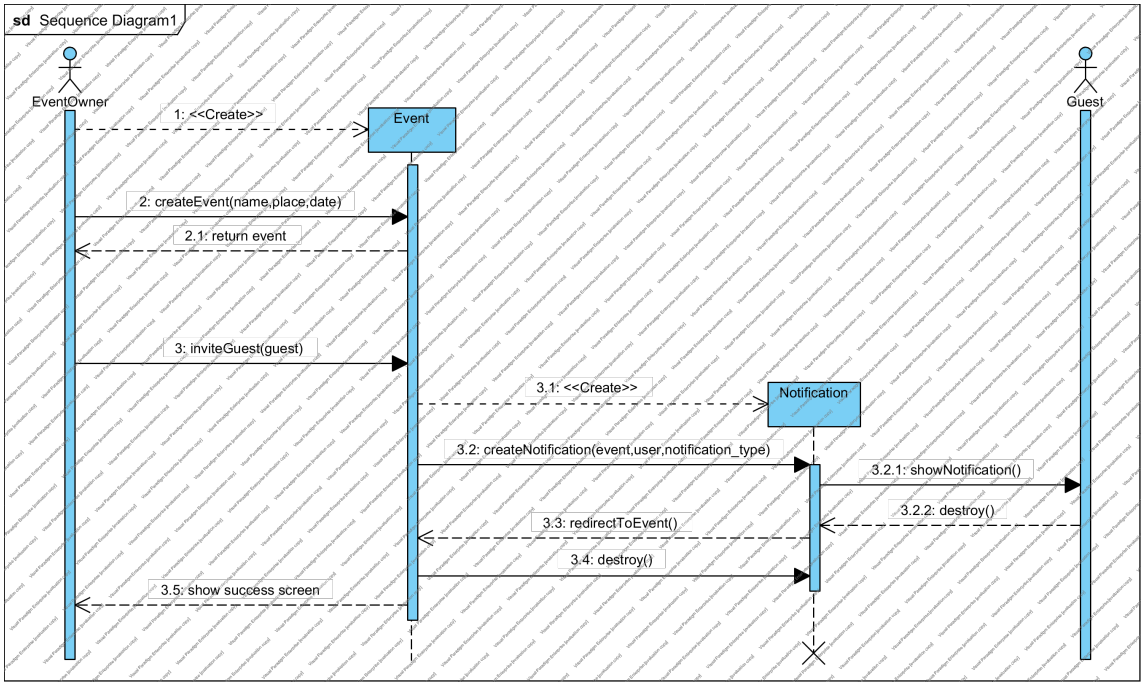

Figure 14: Sequence diagram - create event

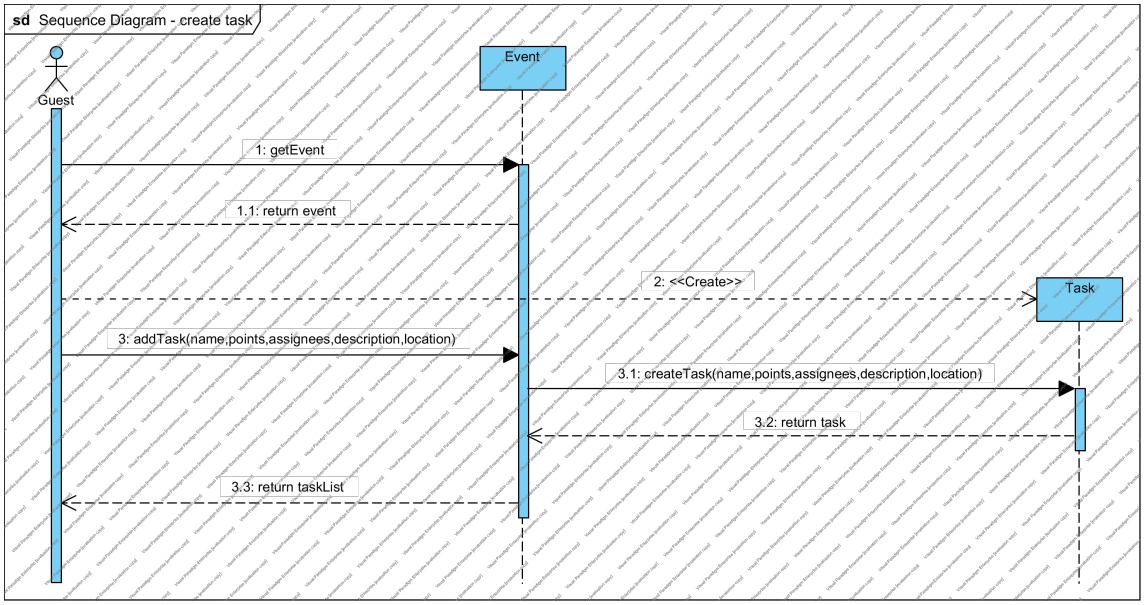

Figure 15: Sequence diagram - create task