System Specification
ennui.at

Project Name	Ennui.at
Project Manager	Markus Geilehner
Document Owner	Martin Singer
Created on	25/11/2016
Last modified	18/04/2017 22:54
State	in Work vorgelegt x Released
Dokumentablage	SystemSpecification.docx

Content

1 Initial Situation and Goal

1.1 Initial Situation

Currently, searching for spare time activities is a task which has to poll a big bunch of different data sources.

If one only considers a very specific target group like teenagers from 15 to 20 years of age it turns out that there is no real central source for information about proper spare time activities.

Even social media platforms like Facebook, Twitter, SnapChat etc. give only partly useful information. For example, not every company will have their Facebook site where they advertise the activities. This way of searching also claims lots of time because it is not filterable and not known activities can only be found coincidently or via a friend sharing it.

The more complicated it gets if one broadens the target group to children, or more adults, young parents, mid-agers, grand-parents, elderly people, etc. Here some offline calendars of events, like written programs of local cinemas, etc. have to be consulted additionally.

The problem here is two-fold: First of all there is no central source for upcoming events available which enables people to do a focused search for spare time activities.

But on the other hand it is also difficult for event hosts to promote their events. Here again FB is kind a basis but here we also have the same problem as mentioned before, Users will most of the time not find the event.

The only similar website that already exists is szene1.at. There the events can already be filtered by location, they also have public transport support and photos can be watched after the event. But there are several problems: Szene1 is only for parties, really outdated design-wise and half of the people do not even know that there is an event finder, because szene1 is only known for their event photos.

1.1.1 Application Domain

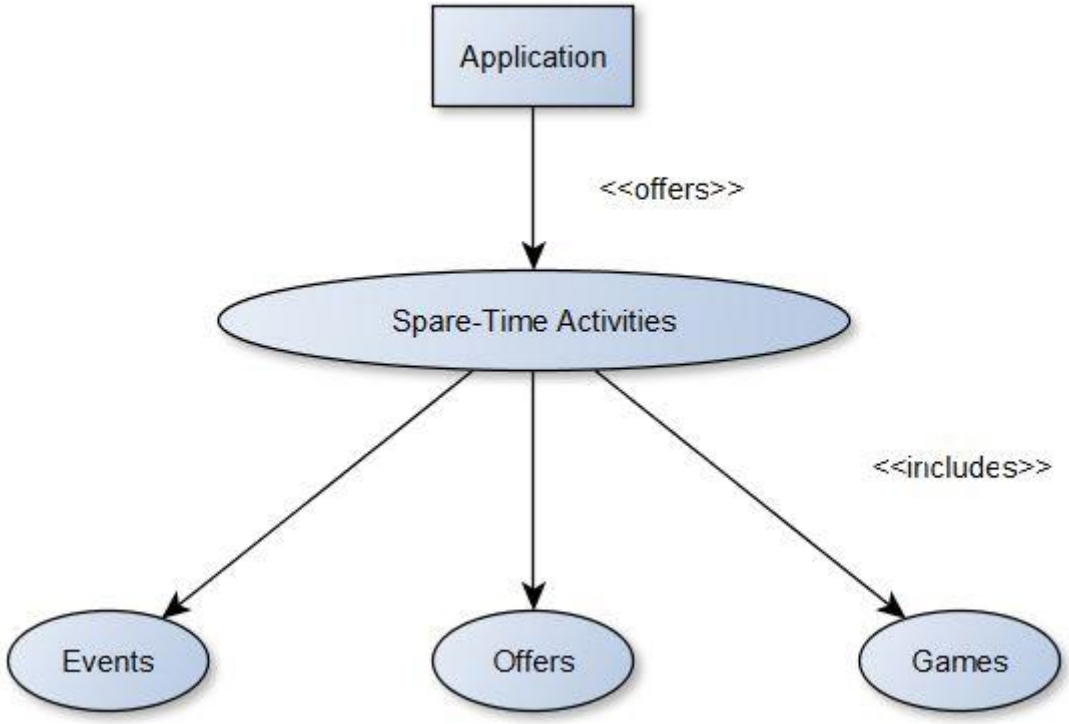
The plan of our application is to show spare time activities. We divide these activities into three sections:
Events: Events are activities that are only temporarily available (for instance: parties, concerts, musicals). Normally this kind of data can be found on different sources like Facebook, weekly magazines, etc.
Offers: Offers are activities that are always on certain times available (for example: museums, climbing areas, hiking routes). This kind of data can normally be found on Google via Maps, or in local tourist brochures.
Games: Games are activities that can be anywhere played with friends, for example card games. Users can post their game ideas and share them with others.

To ensure the best quality in all three categories, different sources like Facebook or Google Maps have to be minded

1.1.2 Glossary

API: Application Programming Interface
Spare time activities/Activities: Normally consisting of 3 Categories (Events, Offers, Games), in the Use Case Section only consisting of 2 (Events and Offers), because Games are most of the time handled differently.
Events: Activities that are only temporarily available (concerts, parties)
Offers: Activities that are always on certain times available (museums, climbing areas)
Games: Classical card or board games which can be played with friends

1.1.3 Model of the Application Domain



1.2 Goal Definition

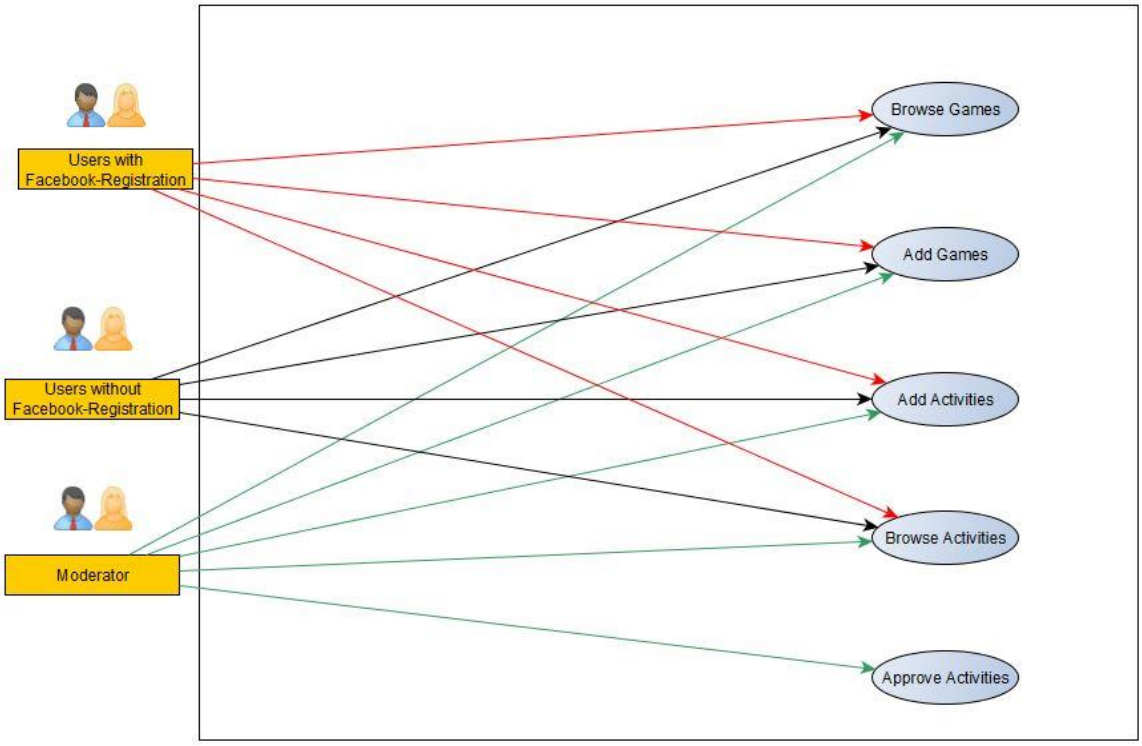
The main goal of our idea is to ease finding the perfect spare time activity for everyone. To make this happen we are working on our website called ennui.at which searches all activities available nearby your location plus additional information to the event and routes. There will also be filters available for public transport options, price range and tags that can be added, to specify the event searched.

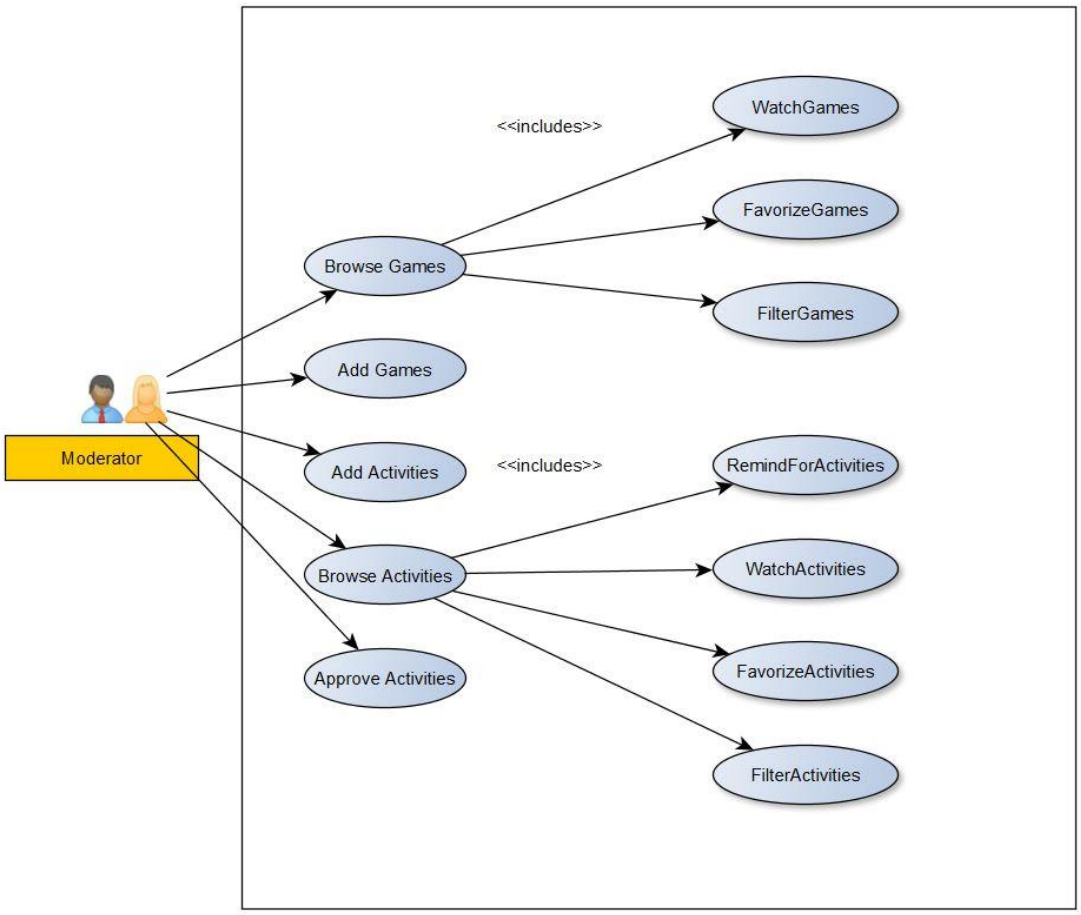
Under the term “spare time activities” we defined three different categories:
Events: Events are activities that are only temporarily available (for instance: parties, concerts, musicals).
Offers: Offers are activities that are always on certain times available (for example: museums, climbing areas, hiking routes).
Games: Games are activities that can be anywhere played with friends, for example card games. Users can post their game ideas and share them with others

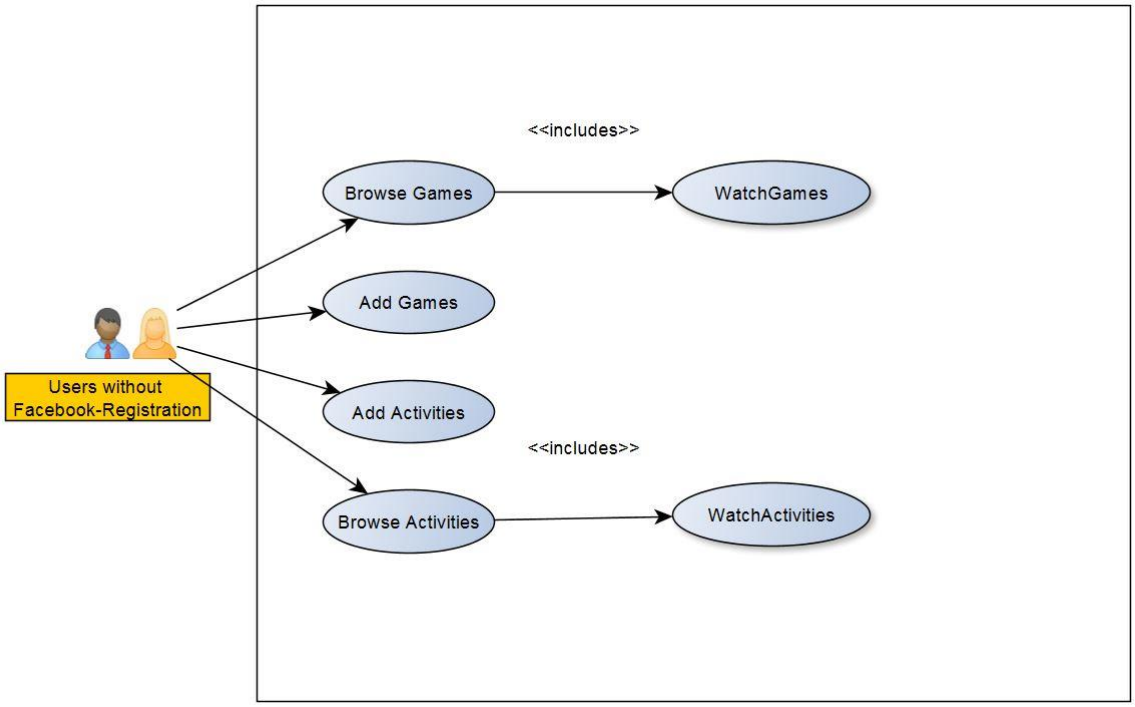
The main target group for our project are teenagers, but also adults or families, more precisely: Everybody who wants to get the best events available in a short time. ennui.at will be easy to handle and nearly everybody has an internet access today, which is sufficient to run our service. We will also release apps for android, iOS, Windows and macOS.

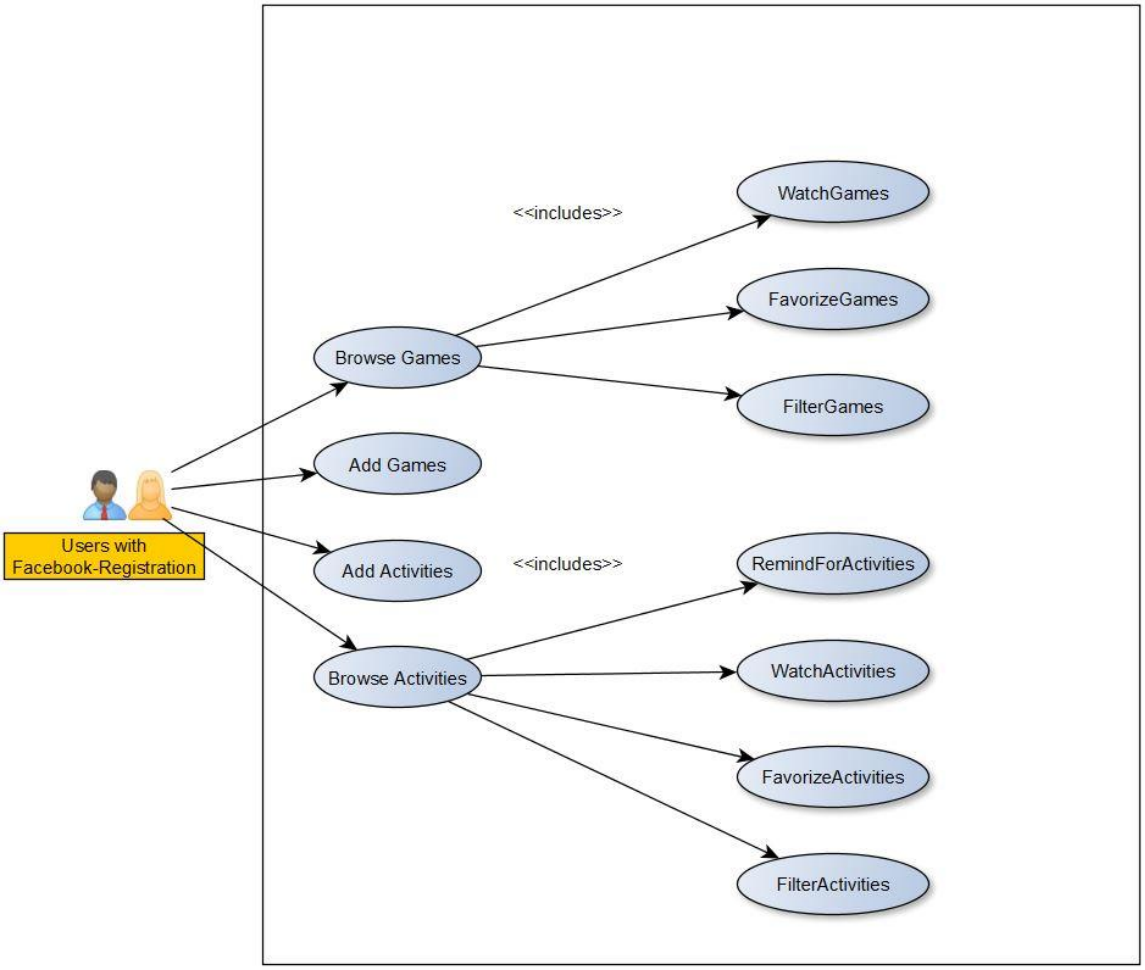
2 Functional Requirements

2.1 Use Case Diagrams




Moderator
Browse Games
Add Games
Add Activities
Browse Activities
Approve Activities
<<includes>>
<<includes>>
WatchGames
FavorizeGames
FilterGames
RemindForActivities
WatchActivities
FavorizeActivities
FilterActivities


<<includes>>
Browse Games
WatchGames
Add Games
Users without Facebook-Registration
Add Activities
<<includes>>
Browse Activities
WatchActivities


WatchGames
<<includes>>
FavorizeGames
Browse Games
FilterGames
Add Games
Users with Facebook-Registration
Add Activities
<<includes>>
RemindForActivities
Browse Activities
WatchActivities
FavorizeActivities
FilterActivities

2.2 Description of use cases

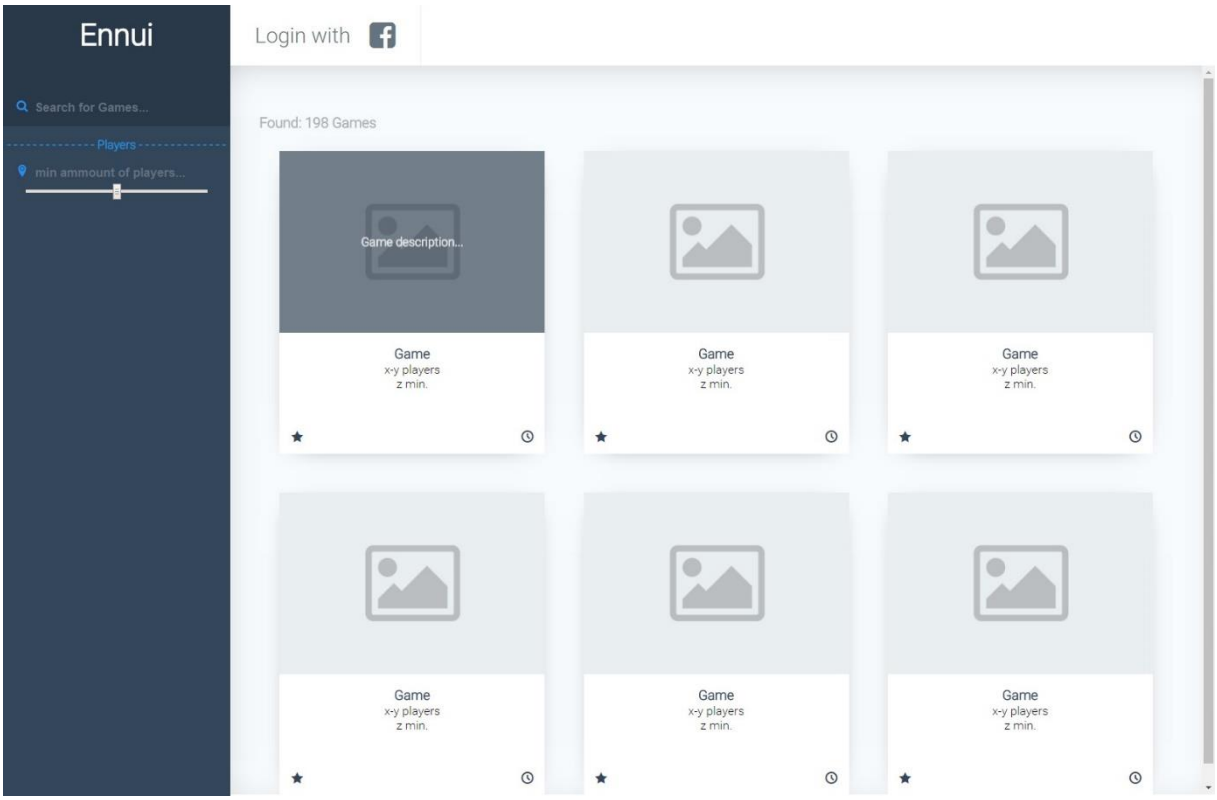
2.2.1 BrowseGames

In the game section of the website, the user is able to watch, favourize or add games.

Characteristic Information

Goal:	The user finds a game he likes and is able to favorize it if he wants to
Precondition:	The user switches to the Game section and wants to find a game He wants to add a game to his favourites
Postcondition:	The user has found his game and is able to play it The game is added to the favourites of the user
Involved User:	User: Searches for a game and favorizes it Server: Hands the game informations for the games. After the user favorizes a game, the server adds it to the Favourites section.
Triggering Event:	Pressing the favorize button

GUI to call the use case



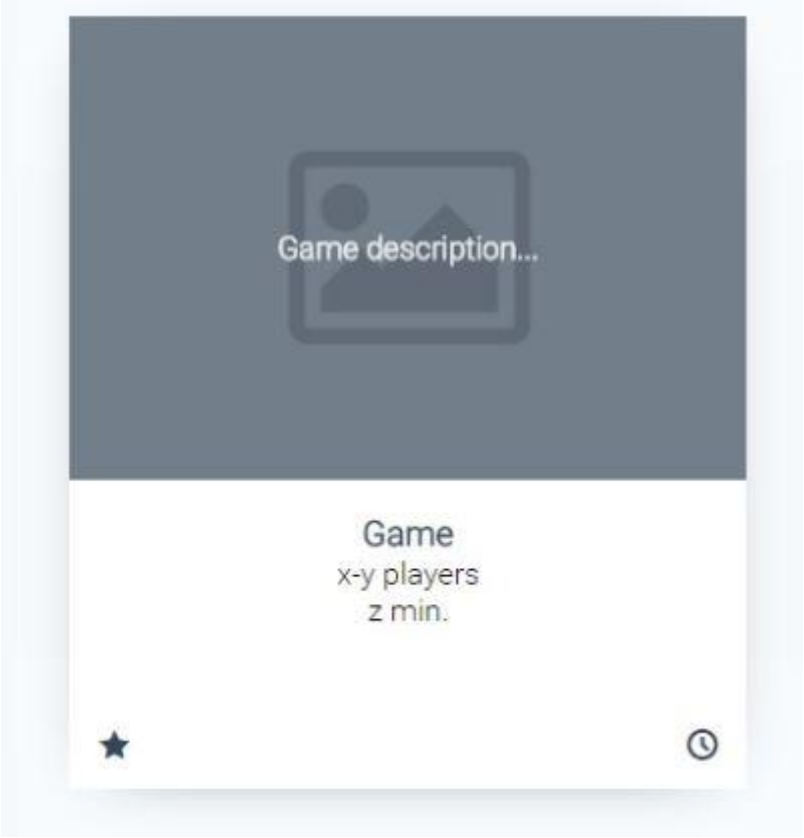
2.2.1.1 Watch Games

The user can watch games created by other users, including how to play them, the minimum or maximum of people that are able to play them and things that are required, like dices for example.

Characteristic Information

Goal:	The user finds a game he likes
Precondition:	The user switches to the Game section and wants to find a game
Postcondition:	The user has found his game and is able to play it
Involved User:	User: Searches for a game Server: Hands the game informations for the games.
Triggering Event:	Pressing the Games button

GUI to call the use case



Click anywhere on the game to show detailed information

GUI for the standard use

Step	User	Activity
1	user	Clicks on the games section
2	server	Gets the games posted by users
3	server	Lists the games

Scenarios for non-standard uses (bad cases or work around cases)

Step	User	Activity
1	user	Clicks on the games section
2	server	Gets the games posted by users
3	client	Connection interrupt

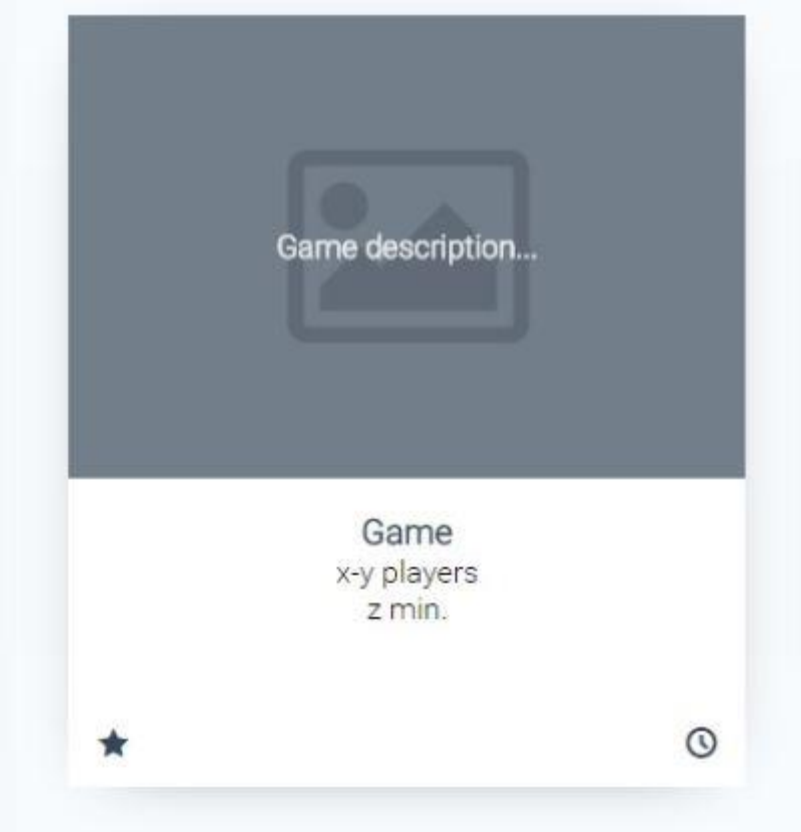
2.2.1.2 FavourizeGames

If the user likes a game, he is able to favorize it. After that, the game is added to the Section MyFavourites, next to his favourite activities.

Characteristic Information

Goal:	The user finds a game he likes and is able to favorize it if he wants to
Precondition:	The user wants to add a game to his favourites
Postcondition:	The game is added to the favourites of the user
Involved User:	User: Favorize a game it Server: After the user favorizes a game, the server adds it to the Favourites section.
Triggering Event:	Pressing the favorize button

GUI to call the use case



Press the star-icon to favorize a game

GUI for the standard use

Step	User	Activity
1	user	Clicks on the “star”-Icon from a game to favorize it
2	server	Server saves the game to the user’s favourite page

Scenarios for non-standard uses (bad cases or work around cases)

Step	User	Activity
1	user	Clicks on the “star”-Icon from a game to favorize it
2	client	Connection interrupt

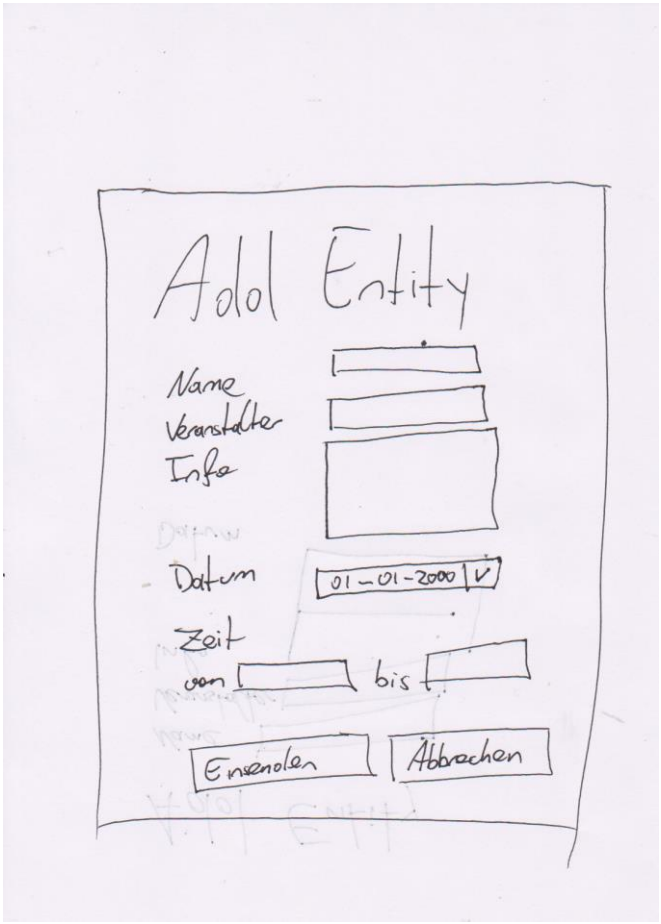
2.2.2 AddGames

If one clicks at the plus button in the right bottom corner, a form opens where he is able to add his own game. After an algorithm has checked that the game is not spam or junk, it gets added to the games section.

Characteristic Information

Goal:	Successfully add a game
Precondition:	The user has a game idea and wants to share it via our website
Postcondition:	After filling the fields of the registration formular with right information, the game gets handed to the moderator-section
Involved User:	User: Needs to fill out the formular properly Server: hands the filled-out formular to the moderator-section Moderator: Checks the formular and adds it to the page if it is complete and serious (no junk or spam)
Triggering Event:	After filling in correct information, pressing the Send Form-Button will save the informations and handle it to the moderator section

GUI to call the use case



GUI for the standard use

Step	User	Activity
1	user	Presses + Button in the right Corner of the Game section
2	user	Fills out the form and presses submit
3	server	hands form to moderator section
4	moderator	checks if form is not junk or spam
5	server	if form is not spam, adds game to website
6	user	gets a notification if the game is successfully added

Scenarios for non-standard uses (bad cases or work around cases)

Step	User	Activity
1	user	Presses + Button in the right Corner of the Game section
2	user	Fills out the form and presses submit
3	server	invalid form data

Step	User	Activity
1	user	Presses + Button in the right Corner of the Game section
2	user	Fills out the form and presses submit
3	server	hands form to moderator section
4	moderator	checks if form is not junk or spam
5	server	if form is not spam, adds activity to website
6	client	connection interrupt

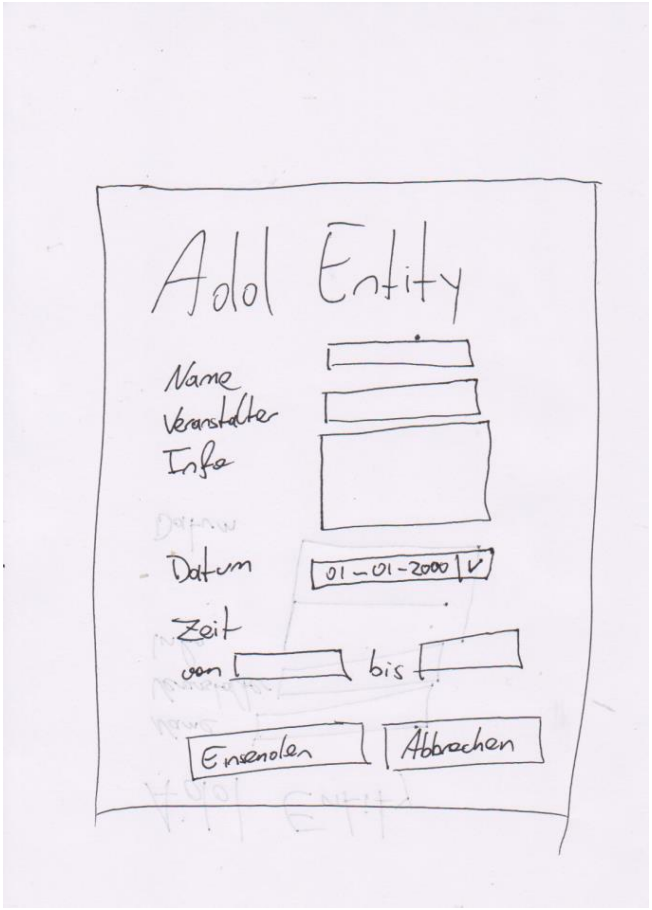
2.2.3 AddActivities

If one clicks at the plus button in the right bottom corner, a form opens where he is able to add his own activity. After an algorithm has checked that the activity is not spam or junk, it gets added to the event or offer section.

Characteristic Information

Goal:	Successfully add an activity
Precondition:	The user has an event/offer and wants to share it via our website
Postcondition:	After filling the fields of the registration formular with right information, the activity gets handed to the moderator-section
Involved User:	User(both): Needs to fill out the formular properly Server: hands the filled-out formular to the moderator-section Moderator: Checks the formular and adds the event to the page if it is complete and serious (no junk or spam)
Triggering Event:	After filling in correct information, pressing the Send Form-Button will save the informations and handle it to the moderator section

GUI to call the use case



GUI for the standard use

Step	**User**	**Activity**
1	User(both)	Presses + Button in the right Corner of the Event/Offer section
2	User(both)	Fills out the form and presses submit
3	server	hands form to moderator section
4	moderator	checks if form is not junk or spam
5	server	if form is not spam, adds event to website
6	User(both)	gets a notification if the event is successfully added

Scenarios for non-standard uses (bad cases or work around cases)

Step	User	Activity
1	user	Presses + Button in the right Corner of the Event/Offer section
2	user	Fills out the form and presses submit
3	server	invalid form data

Step	User	Activity
1	user	Presses + Button in the right Corner of the Event/Offer section
2	user	Fills out the form and presses submit
3	server	hands form to moderator section
4	moderator	checks if form is not junk or spam
5	server	if form is not spam, adds event to website
6	client	connection interrupt

2.2.4 BrowseActivities

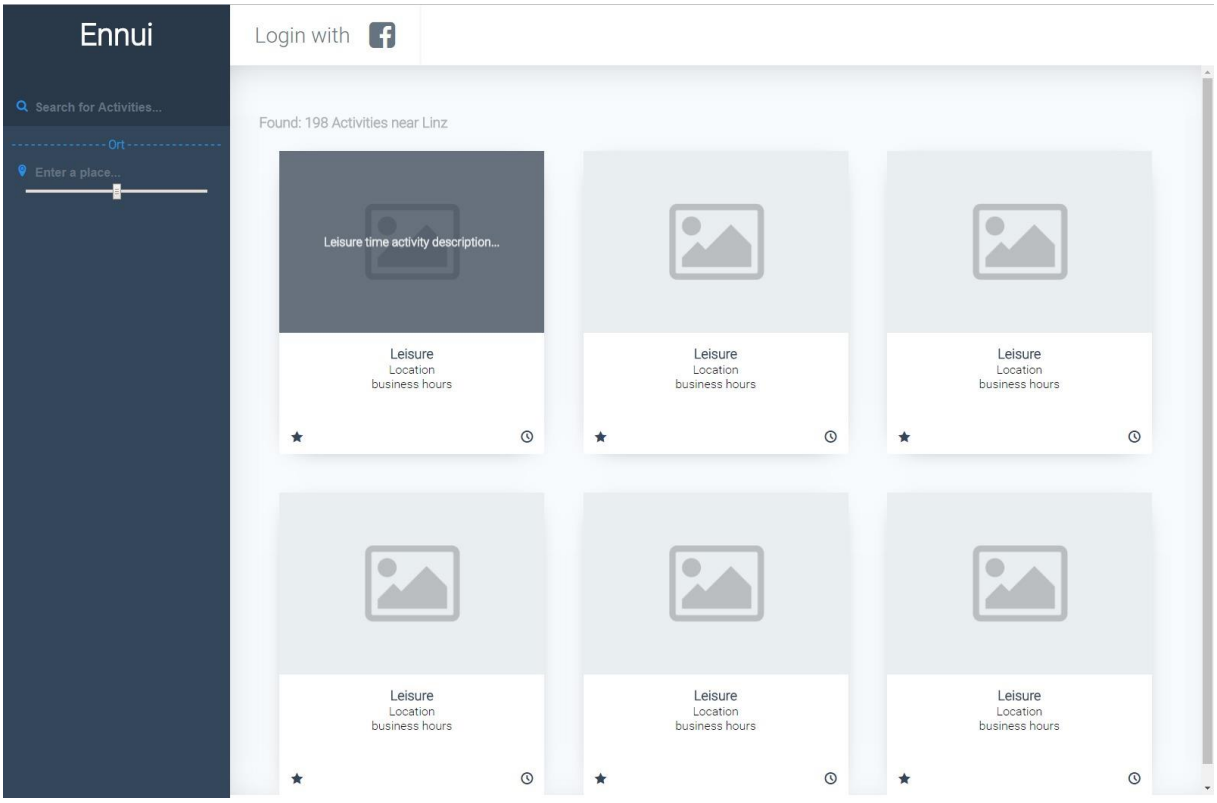
The user is able to filter, favorize, watch or add activities on the website. He can also set a reminder for an Activity.

Characteristic Information

Goal:	The user is able to watch and favorize an Activity, he is able to filter the informations and set a reminder for it.
Precondition:	The user switches to the Events/Offer section and wants to find an Activity He wants to filter the Activities He wants to set an Reminder for an Activity He wants to add a Activity to his favourites
Postcondition:	The user found his Activity and has enough information to attend to it easily He finds now only Activities which apply to the filters he chose He now gets an alert on the day of the Activity The Activity is added to the favourites of the user
Involved User:	User: Searches for an Activity

	Chooses filters for searching Chooses an Activities he wants to be reminded of Favorizes an Activity Server: Hands the Activity informations to the users Applies filters Sets an reminder for the Activity Sets an Activity as a favourite
Triggering Event:	Pressing the Activities-Button Pressing the Apply Filters-Button Pressing the Remind me-Button Pressing the favorize button

GUI to call the use case



2.2.4.1 WatchActivities

The user can watch Activities in the events or offers section, created by other users or directly imported from Facebook.

Characteristic Information

Goal:	The user is able to watch an Activity
Precondition:	The user switches to the Events/Offers section and wants to find an Activity
Postcondition:	The user found his Activity and has enough information to attend to it easily
Involved User:	User: Searches for an Activity Server: Hands the Activity informations to the users
Triggering Event:	Pressing the Events/Offers-Button

GUI to call the use case

Press anywhere on the Activity to watch a detailed page of it

GUI for the standard use

Step	User	Activity
1	user	Clicks on the Events/Offers Section
2	server	Gets location of the user and sends it to the server
3	server	Retrieves information and sends back Activities near the user

Scenarios for non-standard uses (bad cases or work around cases)

Step	User	Activity
1	user	Clicks on the Events/Offers Section
2	server	Is unable to get the location, because the client is blocking it

Step	User	Activity
1	user	Clicks on the Events/Offers Section
2	server	Gets location of the user and sends it to the server
3	client	Connection interrupt

2.2.4.2 FavorizeActivities

If the user likes an Activity, he is able to favorize it. After that, the Activity is added to the Section MyFavourites, next to his favourite games. In this section are also the Activities the user is interested on in Facebook.

Characteristic Information

Goal:	The user is able to favorize an Activity
Precondition:	He wants to add a Activity to his favourites
Postcondition:	The Activity is added to the favourites of the user
Involved User:	User: Favorizes an Activity Server: Sets an Activity as a favourite
Triggering Event:	Pressing the favorize button

GUI to call the use case

Press the star-icon to favorize an Activity

GUI for the standard use

Step	User	Activity
1	user	Clicks on the “star”-Icon from an Activity to favorize it
2	server	Saves the Activity to the user’s favourite page

Scenarios for non-standard uses (bad cases or work around cases)

Step	User	Activity
1	user	Clicks on the “star”-Icon from an Activity to favorize it
2	client	Connection interrupt

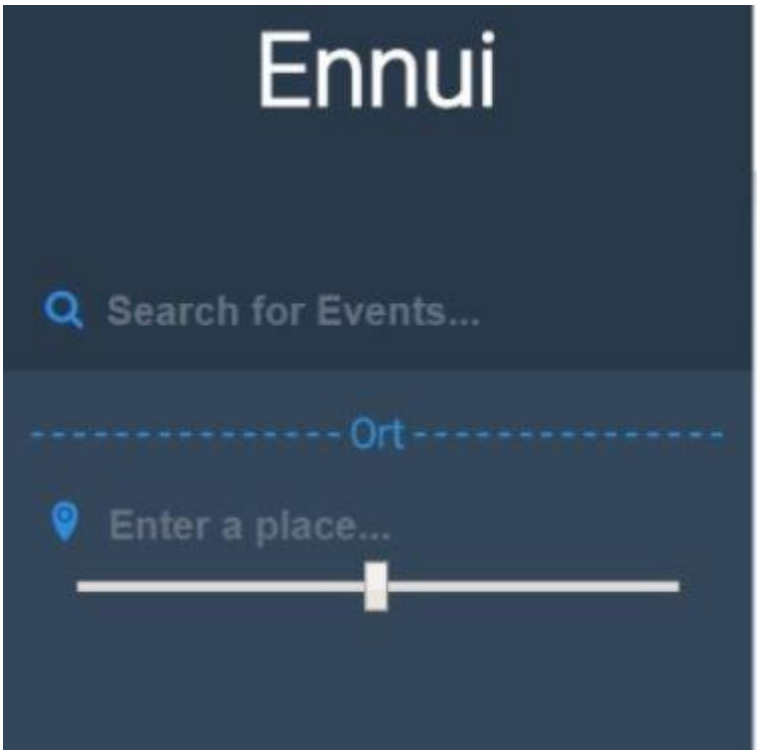
2.2.4.3 FilterActivities

The website has a filter menu to search for Activities more specific. Planned options for filtering are: location(current position of the user by default), radius around the location(25km by default), timespan between day x and y(a timespan of two weeks starting with the current day by default), categories. If there is any way to reach it with public transport or not and price range.

Characteristic Information

Goal:	The user is able to filter the required informations
Precondition:	The user wants to filter the Activities
Postcondition:	He finds now only Activities which apply to the filters he chose
Involved User:	User: Chooses filters for searching Server:Applies filters
Triggering Event:	Pressing the Apply Filters-Button

GUI to call the use case



Change the radius or search for Activities, more filters will be added soon

GUI for the standard use

Step	User	Activity
1	user	Clicks on the filter button
2	user	can set options to filter for (Location,Date,Radius)
3	user	clicks on the “filtern” button
4	server	responds with a new list of Activities filtered by the user input

Scenarios for non-standard uses (bad cases or work around cases)

Step	User	Activity
1	user	Clicks on the filter button
2	user	can set options to filter for (Location,Date,Radius)
3	user	clicks on the “filtern” button
4	client	connection interrupt

Step	User	Activity
1	user	Clicks on the filter button
2	user	can set options to filter for (Location,Date,Radius)
3	user	clicks on the “filtern” button
4	server	invalid form data

2.2.4.4 RemindForActivities

The user is able to set a reminder for an Activity. He then gets a push notification at the day of the Activity, to make him not forget about it.

Characteristic Information

Goal:	The user is able to set a reminder for an Activity.
Precondition:	The user wants to set an Reminder for an Activity
Postcondition:	He now gets an alert on the day of the Activity
Involved User:	Chooses an Activities he wants to be reminded of Server: Sets an reminder for the Activity
Triggering Event:	Pressing the Remind me-Button

GUI to call the use case

Press the "watch"-icon to set an reminder for an Activity

GUI for the standard use

Step	User	Activity
1	user	Clicks on the remind button of an Activity
2	server	saves a reminder for the user the user

Scenarios for non-standard uses (bad cases or work around cases)

Step	User	Activity
1	user	Clicks on the remind button of an Activity
2	client	Connection interrupt

2.2.5 ApproveActivities

Users with moderator permissions, are also able to approve Activities. In the moderator section of the website, all “form fillings” from our users can be found. Before we have the algorithm finished to check the fillings automatically, our moderators have to read over them before they get posted on the site.

GUI to call the use case

GUI for the standard use

Step	User	Activity
1	moderator	accepts Activity/game
2	client	Add Activity/Game to the website

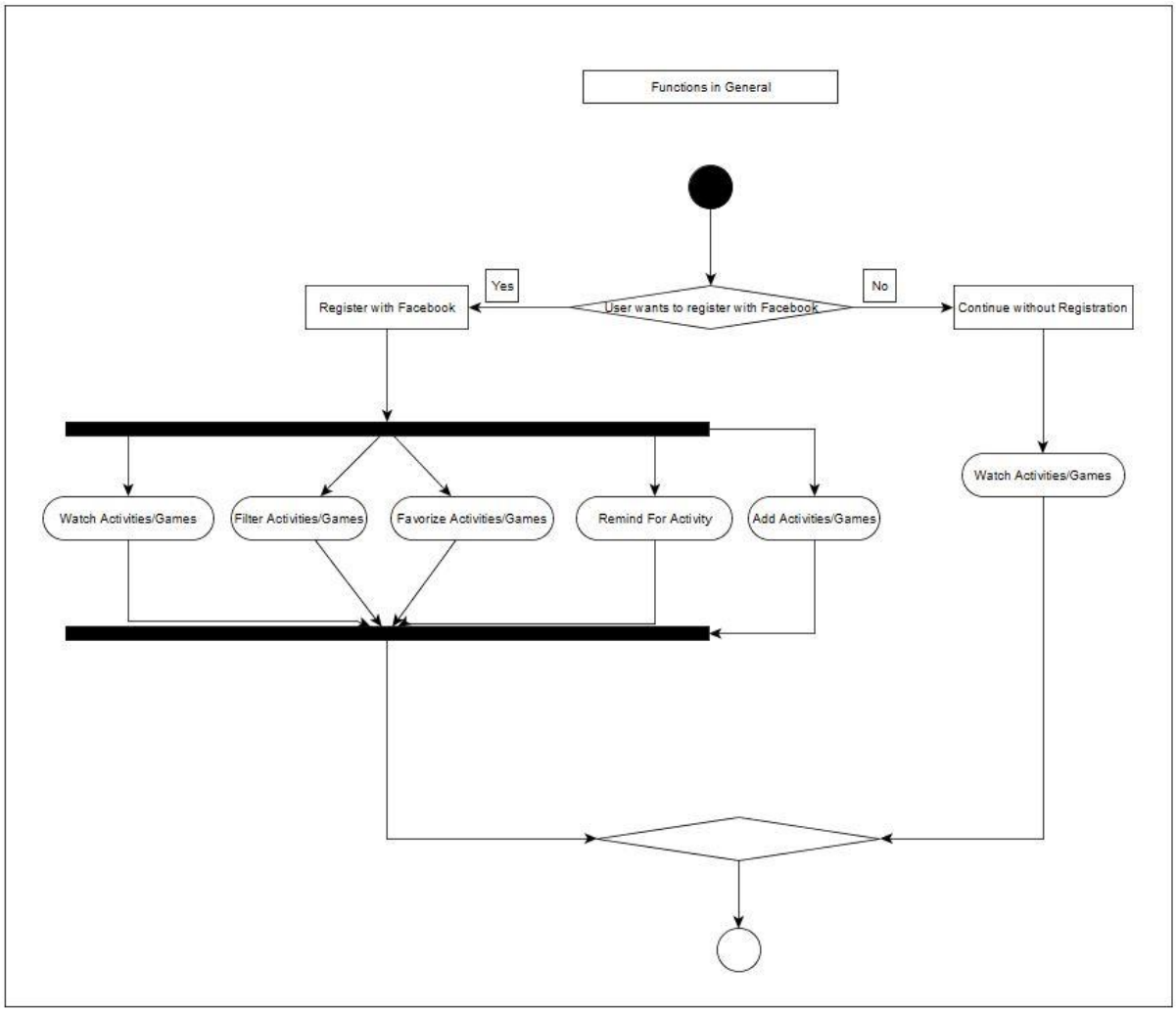
Step	User	Activity
1	moderator	declines Activity/game
2	client	Delete Activity/Game

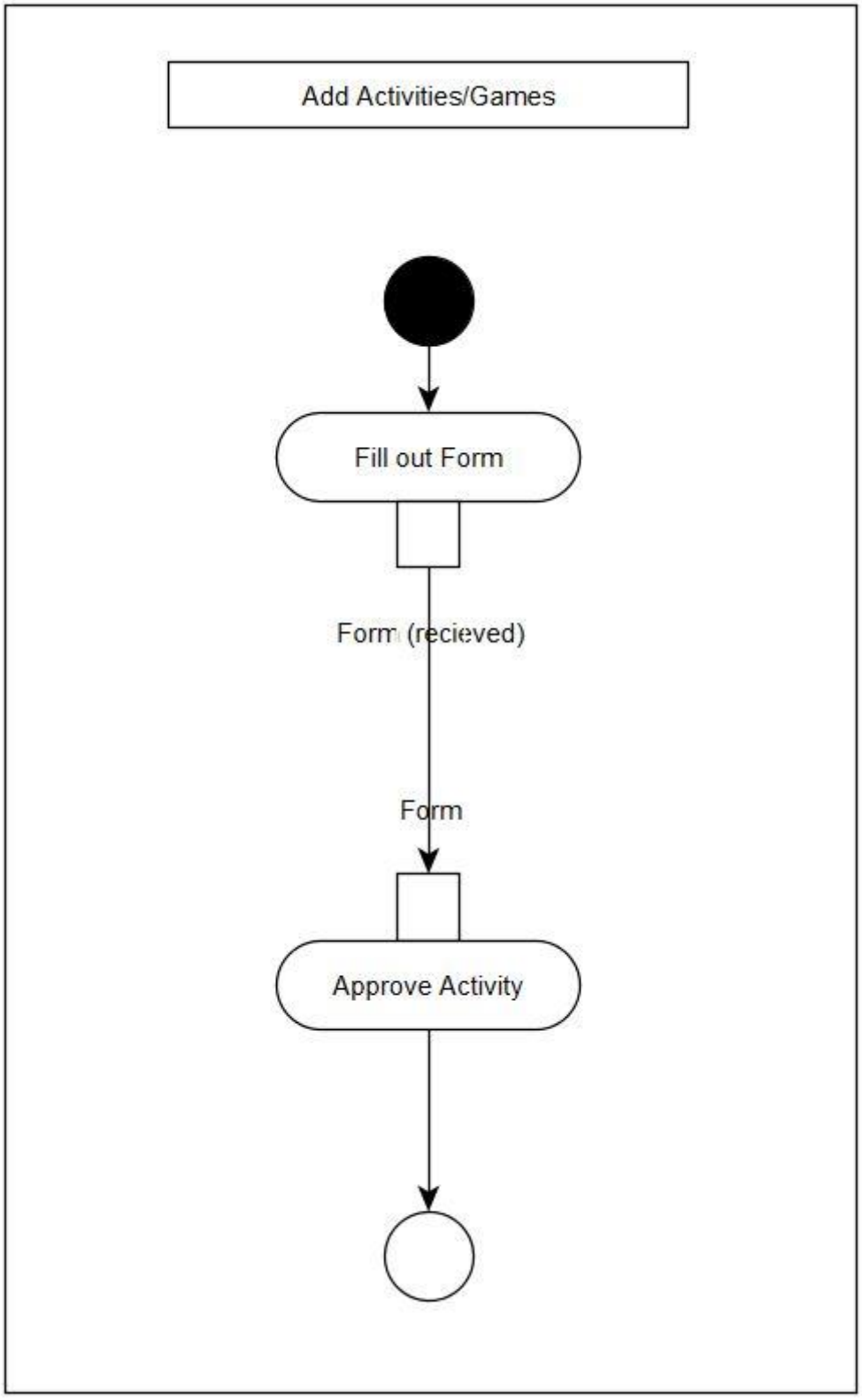
Scenarios for non-standard uses (bad cases or work around cases)

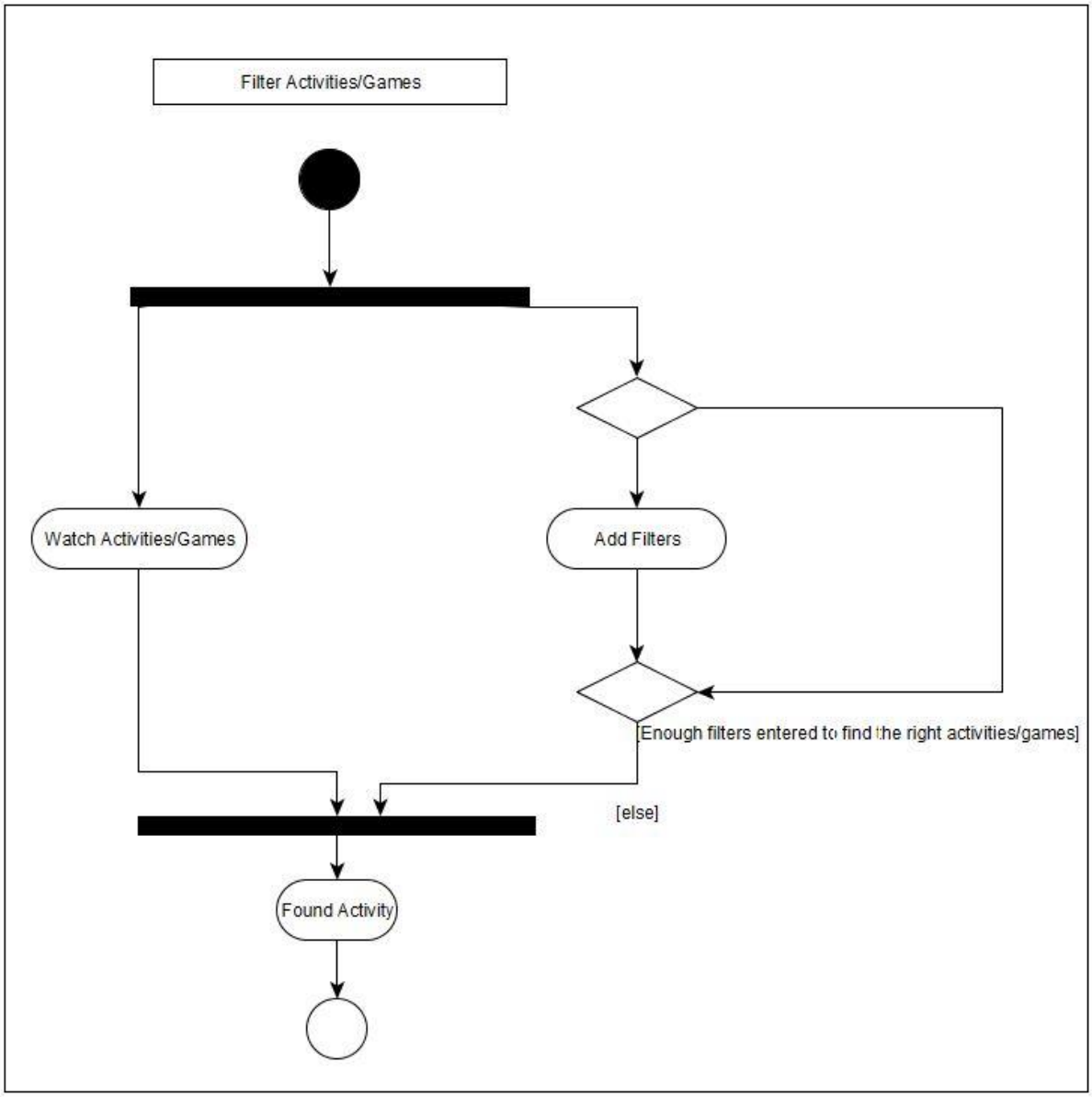
Step	User	Activity
1	admin	accepts Activity/game
2	client	connection interrupt

Step	User	Activity
1	admin	declines Activity/game
2	client	connection interrupt

2.2.3 Workflow




Add Activities/Games
Fill out Form
Form (recieved)
Form
Approve Activity



2.2.4 Open Points

·Plans for the Diploma thesis

- Big Data Analysis: The user gets event suggested, which could be relevant for him, based on the events he already attended and on the events his friends are attending. He also gets alerts if he is in the vicinity of an event, and if he is leaving, gets tips for cabs he could call or public transport to take him home
- Android and iOS native apps: To make the best performance possible, and making additional features which are only available if the app is natively programmed, an android and an ios app are planned.

3 Non-functional Requirements

ID:	NFR_001
Name:	Design
Type:	USE
Description:	Design Conditions for Android apps, iOS-Apps and JavaFX applications have to be kept to create a familiar environment for the user
Assigned use cases:	----------

ID:	NFR_003
Name:	Language
Type:	USE
Description:	We want to implement our software in the languages English and German
Assigned use cases:	----------

ID:	NFR_005
Name:	Correct Data Insertion
Type:	SEC
Description:	In the Event-Form, inserts have to be checked if they are not spam or junk
Assigned use cases:	AddEvent(2)

ID:	NFR_006
Name:	Avoidance of redundancy
Type:	EFFIC
Description:	The website/app has to ensure that there is not redundancy in the database(for example an event is registered twice)
Assigned use cases:	AddEvent(2)

ID:	NFR_007
Name:	Implementing of another language
Type:	MAINT
Description:	In a later process we also want to publish our site in german, because it is our mother language and is one of the biggest languages in the world
Assigned use cases:	----------

4 Quantity Structure

The following informations about the user get saved in our database:

- Name, E-Mail
- Favorized Events, Reminders for Events as well as for Games
- All Attended Events(see Open Points, big data analysis)

The database will be a plain, Oracle MySQL Database. The estimated size will be in the range of 1GB to 10GB once about 10000 users will be registered. For the database and the Backend a virtual server will be rented from the provider hosteurope.de. For a space of 100gb SSD 9.99€ get monthly charged, if more space is needed an upgrade can be made.

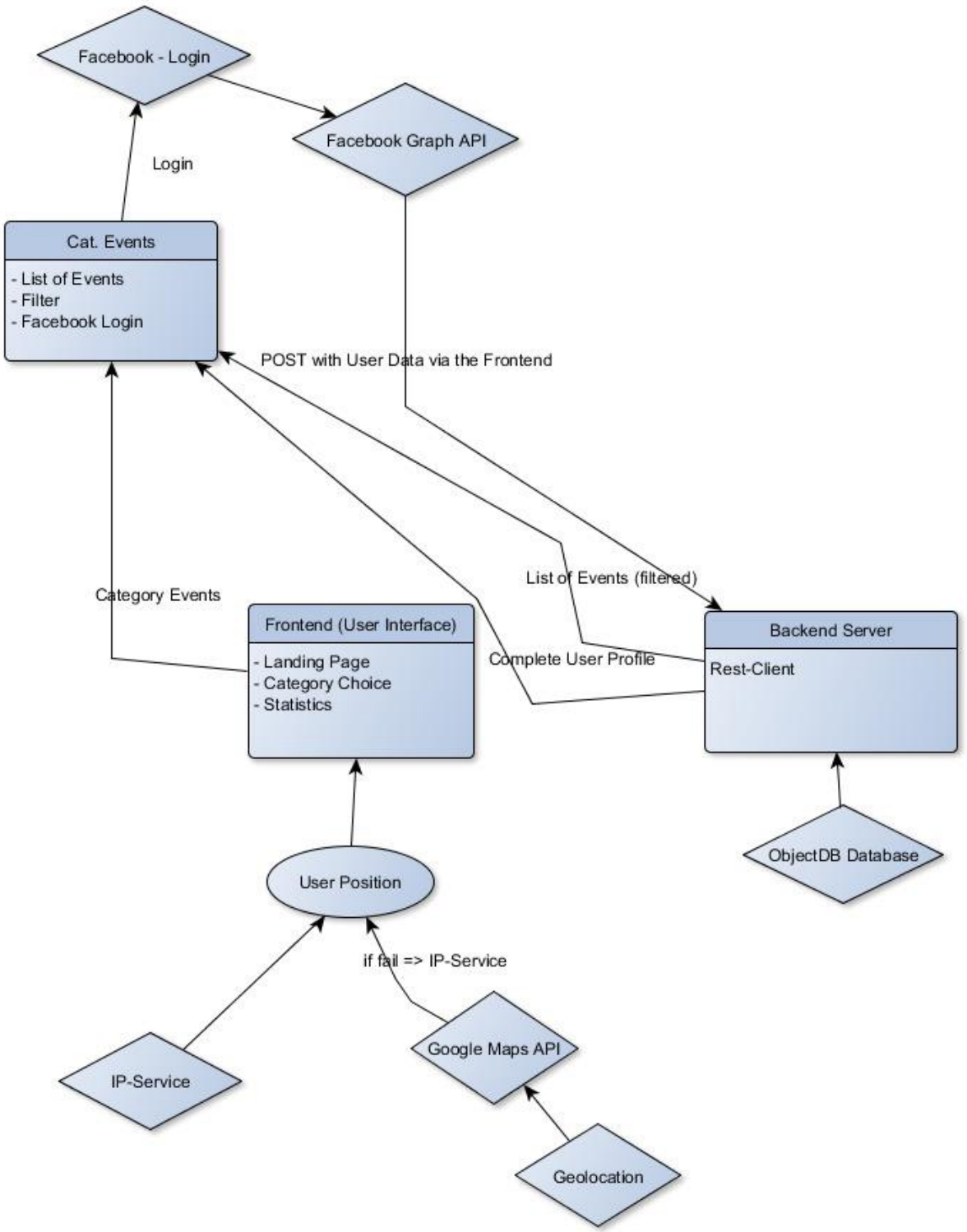
As our Webhost we chose world4you.at. For the domain only (ennui.at) world4you.at charges 13.90€ a year, the first year is free. Because we will only rent the domain, no additional costs will be charged by the company.

2 System Architecture and Interfaces

Our website works with the following APIs. To find the events, the Facebook-API is used. If a user uses our site with Facebook, his whole public events get recognized and put into our event finder, the private events remain private, of course. To get as many events as possible, we also use several “crawlers” which are Facebook users, just made to get as many events as possible on the website.
To get travel routes, taxi-drivers nearby, events including their prices, we use the Google Maps API. For the public transport information, ÖBB Scotty API is used.
Connected with the user’s location, we can provide information about the best events in range, public transport, travelling routes and taxis nearby.

To list the event on our website we take the data provided by Facebook. If this is not sufficient, we send an automatically generated message via Facebook to the event owner. If he wants to provide better information on our website, he is able to submit extra data with our included form. The same form is also directly available on our website, for adding a new event or spare time activity. After submission, an algorithm checks if the form is not spam or junk, if the program cannot certainly detect any of these an email gets sent to us.

For the Frontend we use HTML 5, Sass and Typescript, the Backend is written in Java 8 with additional Frameworks like Spring. To make our web application accessible for Smartphones, we use Electron to convert it into a mobile application for Android, iOS and Windows, instead of developing a native application each.



Component diagram for the category events

3 Acceptance Criteria

Test Step	Expected Behaviour	Test fails if...

Favorize Game:

User with connects with Facebook	User is logged in	conection interrupt, user not connected with Facebook
User presses the Games Button	Games section opens	Connection interrupt
User presses the favourize button attached to a certain game	Game gets added to the MyFavourites section	Connection interrupt, game is already a favourite

Add Game

User clicks on the Game-Button	Game section opens	Connection interrupt
User fills the form	user enters correct Data	User enters incorrect data
User submits the form	add game to list of games to approve for the moderator	invalid form data entered by the user, connection interrupt

Add Event

User presses the Events/Offers Button	Events/Offers section opens	Connection interrupt
User fills the form	user enters correct Data	User enters incorrect data
User fills out and submits the form	add event to list of events to approve for the moderator	invalid form data entered by the user, connection interrupt

Favorize Activities:

User with connects with Facebook	User is logged in	conection interrupt, user not connected with Facebook
User presses the Events/Offers Button	Events/Offers section opens	Connection interrupt
User presses the favourize button attached to a certain Event/Offer	Event/Offer gets added to the MyFavourites section	Connection interrupt, game is already a favourite

Approve Event:

Moderator clicks the admin panel button	Admin panel opens	Connection interrupt
Moderator approves an event added by an user	event gets added to event-list	connection interrupt

Decline Event

Moderator clicks the admin panel button	Admin panel opens	Connection interrupt
Moderator declines an event added by an user	event gets deleted	connection interrupt