IMPLEMENTING A LOCATION BASED GAME WITH A PURPOSE
THE URBANOPOLY WEB CASE

LUIS ALBERTO BONILLA LEON

POLITECNICO DI MILANO

SCHOOL OF INDUSTRIAL AND INFORMATION ENGINEERING

POLO TERRITORIALE DI COMO

MASTER OF SCIENCE IN ENGINERING OF COMPUTING SYSTEMS

IMPLEMENTING LOCATION BASED GAMES WITH A PURPOSE
THE URBANOPOLY WEB CASE

Luis Alberto Bonilla León

In Partial Fulfillment of the Requirements for the Degree of:

Master of Science in Engineering of Computing Systems

Supervisor:

Emanuele Della Valle

Assistant Supervisor:

Irene Celino

CEFRIEL

(ICT Center of Excellence For Research, Innovation, Education and industrial Labs partnerships)

Politecnico di Milano

School of Industrial and Information Engineering

Master of Science in Engineering of Computing Systems

Abstract

Nowadays, phenomena like the rise of gamification and the increasing popularity of human-based computation games led to new ways in which mixed human-computational systems can carry out highly complex computational tasks, by delegating certain steps to human players in an entertaining way. The long-term engagement of the users has a special relevance to solve the computational tasks for which the games were designed. This is the reason why the adoption of multiple communication channels and the availability of such applications on the greatest number of platforms possible (e.g. Web and mobile) are required.

This is the case of Urbanopoly, a Game with a Purpose created to curate geographic data. Taking advantage of the interaction of the players with their environment, Urbanopoly provides an immersive experience of competition between the players; in its first version, Urbanopoly was available only in a mobile version for the Android platform. However, with the objective of raising the interest of more and more players in a market where new applications are daily published, the Urbanopoly development team identified the need to explore other platforms for the game.

The aim of this thesis was therefore to design and develop a Web-based version of the Urbanopoly game taking into account the following requirements: preserving all the functionality of the mobile version, enabling to play even when the geographical position of the player is not available and replacing the current map management technology with a non-proprietary alternative. Given those considerations, a critical element in the project development was to keep exactly the same functionalities without relying on the mobile application's native android functions. A big effort was required in order to search for possible patterns, modules or in some cases to develop the complete components in order to meet this fundamental requirement.

This document describes the process followed to develop the Urbanopoly Web prototype, a product which required over three hundred hours of effort and whose main features were validated by the quality team of the original version. This report presents the different technologies used for its implementation and provides a reference to common issues and challenges that may arise during the creation or migration of any mobile application to its respective web version.

Contents

List of Figures

Introduction

The Urbanopoly initiative in its mobile version was born under the project UrbanGames developed by PlanetData in 2012 with the goal of building geographic location based games for mobile platforms with the objective of providing an immersive experience in the geographical environment that provides tourist recommendations to the users while they are interacting with urban environments as it benefits from the ability of them to provide reliable feedback across the interaction with the environment under the concept of human computation.

The web version of Urbanopoly is part of a set of subsequent efforts to provide alternatives to the user community of the game, besides the implementation of a series of improvements to the game structure referenced by users and the staff in order to evolve into a game with increasingly active players. This document details the design and implementation of the web version of Urbanopoly, the similarities and differences with its mobile version and in addition, some recommendations for the next steps of the application.

The first section describes on a general level the game, in the second section the design of the web version its summarized, the third section makes reference to the details of the implementation of the prototype and finally a series of recommendations are listed for the next steps of the development in order to continue with the evolution of the game.

1. STATE OF THE ART

This section aims to present the state of art on the theme of Games with a Purpose, Location Based Games and Semantic Games with a Purpose, these concepts are the basis for the original development of Urbanopoly and define the characteristics of the game.

1.1. Games with a purpose

A Game with a Purpose (GWAP) is an online game that takes specific complex computational tasks and proposes them as a game to human users. These games are examples of the human computation approach but unlike crowdsourcing, they provide a more controlled environment to manage the allocation of the human effort. The GWAPs were first introduced by von Ahn as an approach to collect semantic information for multimedia data, such as images, audio, and text [1].

- <u>ESP game</u>
  The ESP game is the most famous example of a GWAP taking advantage of human computation to collect image tags as side effect of playing sessions. As its well-known image processing and annotation is currently a highly expensive computational process for computers, however this is a task humans are perfectly able to do.
  The game mechanics are basically as follows, the system randomly selects two users that will play together and then proposes the same image to both, they have to type textual metadata to describe the image content. The level ends when the two players post the same textual tag and the game proposes another image.

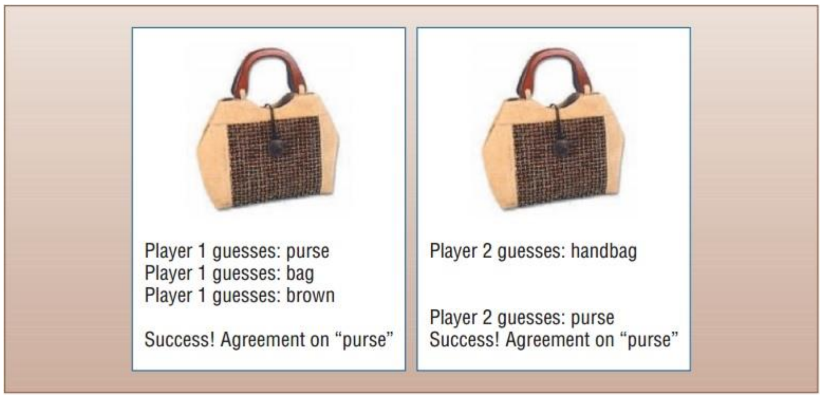


Figure 1 The ESP game

The goal of the players is to get as many points as possible so they have to insert the image description quickly. The system is designed to obtain relevant results so a black list of trivial words named “Taboo words” is provided to the players.
The levels are simple, short and repetitive in order to collect information efficiently, players have to play simultaneously to validate the submitted answers but in some cases one of the two players can be replaced by a simulated player using the answers provided in previous gaming sessions [2].

- <u>Fraxinus</u>
  A Facebook game developed by Team Cooper & The Sainsbury Laboratory aiming to respond to a recent phenomenon presented in the UK where the Ash trees are being attacked by a serious disease leaving some of the country’s few remaining ancient woodlands in serious danger.

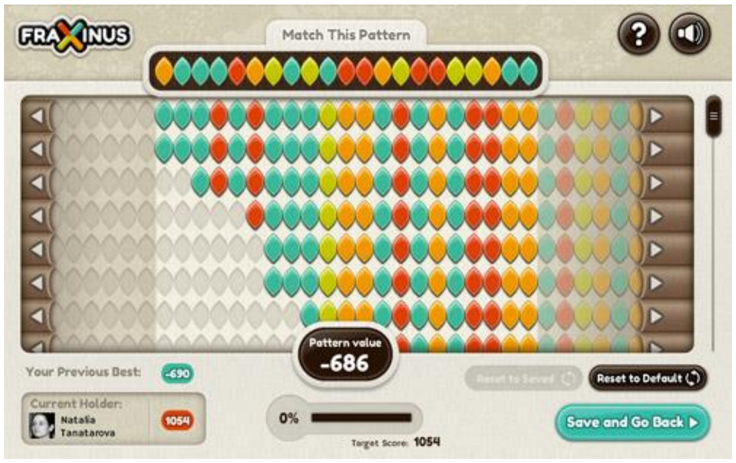


Figure 2 Fraxinus.

The game uses the tree’s real genetic data, and challenges the player to manipulate patterns to match sequences; the gameplay take advantage of human’s capacity for pattern recognition and gives scientists a helping hand sorting the genome of the Ash tree in pursuit of a cure for the disease [3].

- <u>EteRNA</u>
  The EteRNA game is a science based puzzle developed by Carnegie Mellon & Stanford University in which players help to solve real life RNA molecule folding problems.

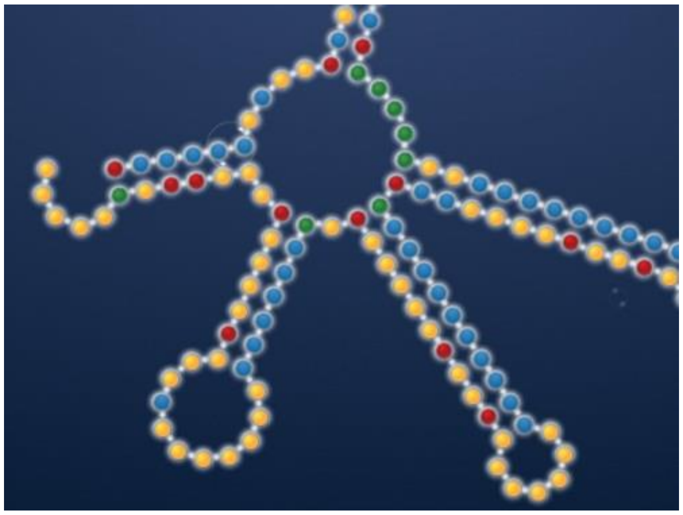


Figure 3 EteRNA

Starting out with simple, yet addictive puzzles, the player is trained up until they have the skills to contribute to regularly changing RNA problems posed by the scientists behind the game. Top voted solutions are synthesized in a lab and ultimately improve RNA folding research at a much faster rate than traditional computer models [4].

1.2. Location based web games

A location-based game or location-enabled game is a type of pervasive game in which the gameplay evolves and progresses based on players location, this type of game is made possible by the recent advances on the hardware of mobile phones and tablets. Players act not by pressing buttons or moving characters on a map, but by moving themselves around in the real world.

- <u>Ingress</u>
  Ingress is an augmented reality massively multiplayer online role playing GPS-dependent game created by Niantic Labs, a startup within Google. The gameplay consists of establishing "portals" at places of public art, landmarks, cenotaphs, etc., and linking them to create virtual triangular fields over geographic areas.

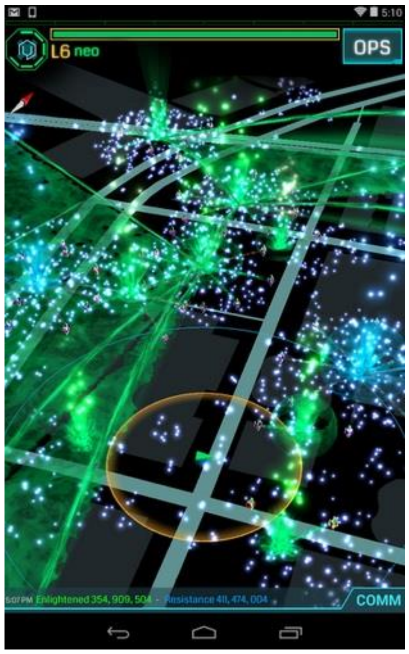


Figure 4 Ingress Gameplay

The progress in the game is measured by the number of Mind Units or people, nominally controlled by each faction and the necessary links between portals may range from meters to kilometers, to hundreds of kilometers in operations of considerable logistical complexity. International links and fields are not uncommon, as Ingress has attracted an enthusiastic following in cities worldwide amongst both young and old players [5].

- <u>The Walk: Fitness Tracker Game</u>
  The Walk is an adaptive fitness game created by the team of Six to Start. With over three months' worth of super immersive storytelling and hundreds of miles of walking this game helps to establish long term fitness habits. The history of The Walk begins with a terrorist attack at Inverness Station so you must escape the city and walk the length of the country, deciding who to trust along the way.

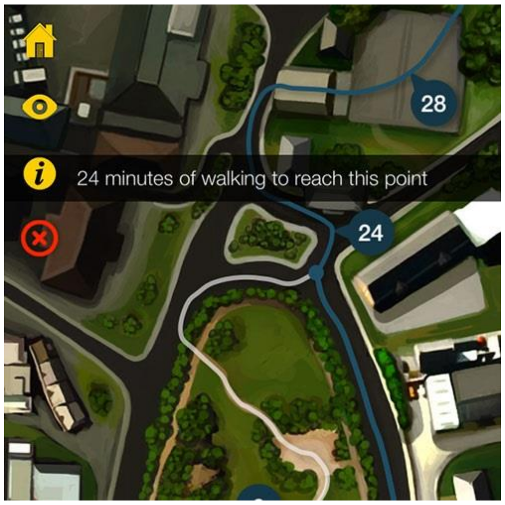
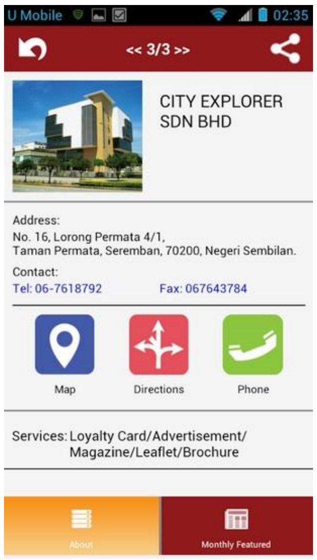


Figure 5 The Walk

The game tracks your steps, how long you've walked for, and rewards you with adventure to keep you moving [6].

- <u>City Explorer</u>
  This game aims to collect images, geographic positions and descriptions of points of interest (POI) of the cities, there are two groups of players the first one has to conquer POIs by sending geographic coordinates, photos or tags of them. The second group has to find those POIs and check the information introduced by the first group. If the information is confirmed as correct, the player who created that piece of information will gain points.



CITY EXPLORER SDN BHD

Address:
No. 16, Lorong Permata 4/1,
Taman Permata, Seremban, 70200, Negeri Sembilan.
Contact:
Tel: 06-7618792 Fax: 067643784

Services: Loyalty Card/Advertisement/Magazine/Leaflet/Brochure

Figure 6 City Explorer

The system uses this gameplay to collect and verify data and player actions can be done at different times, so this game is asynchronous. However the duration can be very long because depends on several factors such as the time that players needs to find places or the time for validate the information [7].

1.3. Semantic GWAP's on the web

The Semantic GWAP term make reference to a special type of Game with a Purpose that uses Semantic Web technologies. There are several games that use the semantic approach or were created in order to contribute to the area of semantic content creation.

- Waisda
  Similar to the ESP concept Waisda is a video annotation game but focusing on TV shows. The players have to annotate the objects in the TV shows they are provided with.

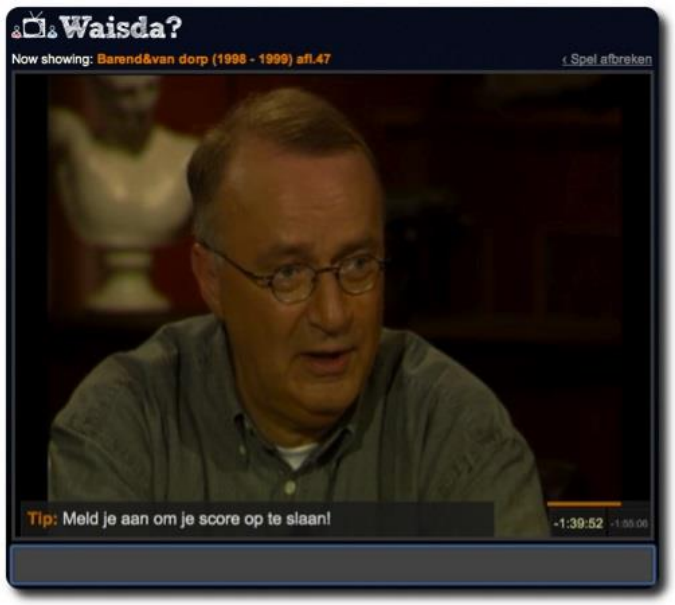
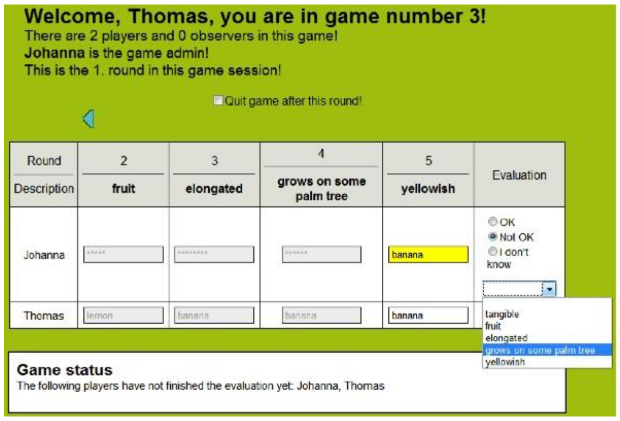

Waisda?
Now showing: Barend&van dorp (1998 - 1999) afl.47
‹ Spel afbreken
Tip: Meld je aan om je score op te slaan!
-1:39:52

Figure 7 Waisda

They annotate the same TV show and in case that two or more players use the same tag to annotate an object, they get points. The shorter the time difference between the answers the more points the players get [8].

- <u>GuessWhat?</u>
  The GuessWhat game focuses on creating ontologies by presenting users with concept fragments from Linked Open Data in order to have the users guessing a described concept. In addition each user can judge on the concept choice of an opponent in order to reach consensus on the final concept to use.



Welcome, Thomas, you are in game number 3!
There are 2 players and 0 observers in this game!
Johanna is the game admin!
This is the 1. round in this game session!

☐ Quit game after this round!

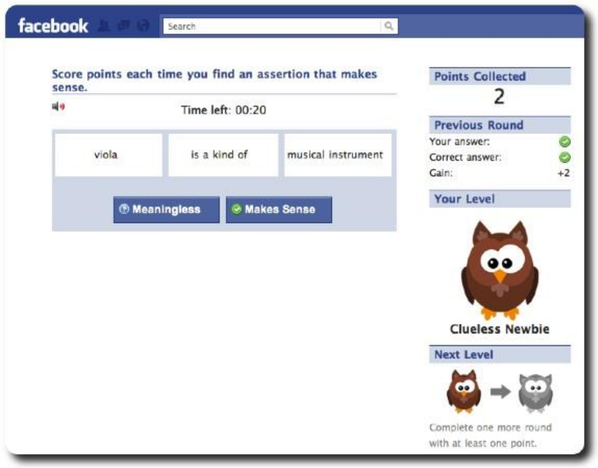
Round / Description	2 / **fruit**	3 / **elongated**	4 / **grows on some palm tree**	5 / **yellowish**	Evaluation
Johanna	*****	********	******	banana	OK [✓ Not OK] I don't know
Thomas	lemon	banana	banana	banana	

Game status
The following players have not finished the evaluation yet: Johanna, Thomas

Figure 8 GuessWhat?

The starting point is a seed concept which serves as basis for crawling DBpedia, Freebase and OpenCyc for concept fragments. From then the game continues round based in which each round provides a new concept for the ontology to be built [9].

- <u>Concept Game</u>
  In Concept Game the user is provided with a randomly chosen assertion consisting of three parts.



Score points each time you find an assertion that makes sense.
Time left: 00:20

viola	is a kind of	musical instrument

Points Collected
2

Previous Round
Your answer: ✓
Correct answer: ✓
Gain: +2

Your Level
Clueless Newbie

Next Level
Complete one more round with at least one point.

Figure 9 Concept Game

The user then has to decide whether the assertion is meaningful or not. In case the user is right, he gets points and additional time for further assertions. This way, Concept Game collects common sense knowledge [10].

1.4. Additional considerations

Although the concepts previously presented defined the classification and typology of the game, this examples also recognized that the adoption in the market for this type of game is not only due to features such as how engaging and fun is the gameplay but also how well integrated is the data collection and how the validation mechanisms meets the purpose for which the game was conceived.

Games with a Purpose have to infer knowledge from player's choices, the mechanism to achieve this is usually player's agreement with a proper statistical analysis. However this approach implies a multiplayer game mode using synchronous interactions, a requirement very difficult to apply to urban environments but that can be relaxed by simulating other player from previous executions.

Another consideration about data validation with games is the complexity of the task, if the task is too complex the design of the game seems to be harder or in the worst case, the task complexity negatively influences the fun and engagement of the game, the more difficult the action, the less motivated the player.

2. THE URBANOPOLY CONCEPT

In this section the concept developed by Urbanopoly in its original version is presented across a detailed description of the game, the gameplay rules are explained and an overview of the purpose behind the game is provided.

2.1. Description of the game

Urbanopoly is a location-based game inspired by the famous game Monopoly, in which the players have to trade hotels, pay rents and so on. Similarly to Monopoly, the player’s goal in Urbanopoly is to become the richest entrepreneur that owns the most prestigious areas of a city. Urbanopoly, however, shows some important difference from the popular board game.

Urbanopoly is a location-aware game played in a real urban context, instead of a fictitious map; players are physically located in a geographical position where they have to move from one place to another in order to play the game. The objective is to make the players feel completely immersed into the game while living their city life.

The real estate trade within the game is based on the exchange of venues representing relevant locations on the city; these venues are accessed via players "check in" and are traded during the game.

Urbanopoly players increment or decrease their possessions, in terms of venues or money by interacting with the surrounding relevant places. When a player checks-in a venue owned by another player, the game presents a casual component (a rotating wheel) that determines the possible actions to continue the game, the player can be asked to pay a rent or can be offered the chance to buy the property, additionally the player can be forced to overcome challenges in exchange for money, this challenges can be "advertise" the venue (by adding new information), perform a "quiz" about the venue (by answer a series of questions) or “judging” other players’ advertisements (by rating the venue information provided).

2.2. Gameplay Overview

The game has the following six options in its main menu: myVenues, Map, Leaderboard, Notifications, Tutorial and Credits. Among this options the Map one stands out because a large amount of the available actions takes place in this option, some of this actions are described below.

2.2.1. Surrounding interactions

As main view on the map option, the game shows to the player a map centered on his current location and containing some icons representing nearby venues, additionally if the player moves around, the map and the venues on it change accordingly. When the player selects a venue icon the game shows the corresponding information about it based on the venue status: a venue can be free (that means it has no owner), occupied or owned by the player.

As was mentioned before the game offers the opportunity to acquire new venues by selecting "Free" properties on the map or by an action of type "Take" in the case of visiting a place that is already occupied (see next section). In the process of acquiring a venue, the game requires in any case the data corresponding to the name of the venue and the respective category (e.g. Transportation - Bus stop for public transport). Once this process is completed, the venue become part of the properties owned by the player and its value is taken from the money belonging to that player.

In the case a player attempts to acquire a venue with a value greater than its money available, the application will block the purchase but offering the possibility of selling or mortgaging the owned venues in order to acquire the necessary money.

In the case of sale a venue the application simply allows the player to recover the funds equivalent to the value of the venue losing the ownership of it.

The action of mortgaging a venue, on the other hand, allows the player to recover half the value of the property with the condition of being able to redeem the mortgage within 24 hours, otherwise the player will lose the other half of the venue in question.

2.2.2. The Game Wheel

When a player spins the game wheel the system chooses, among the available actions, which action assign to him. The wheel returns one of five possible output: take, pay, quiz, advertise or skip. Let's briefly introduce them:

In the case of the "Take" option, the player have the opportunity to take the ownership of a venue, if indeed he decides to take it the price to pay in order to acquire the venue is equal to 150% of its normal value.

If the wheel results in the "Pay" action, the player is forced to pay a rent fee to the owner of the venue.

In the "Quiz" option the player is encouraged to answer questions in order to get money while the server that coordinates the game feeds on the information provided by the players. These sessions are composed by closed questions where the answer may be single or multiple and can be presented in different number, with a minimum of one question.

In the "Advertise" option, in addition to the options proposed in the quiz session, the player has the ability to answer openly to questions through an input field. In addition to the questions, that can vary in number and content, the advertising session requests the name and category of the venue (as seen before in the acquisition phase) and finally, the players are requested to take a picture of the venue. Once the advertise session is completed a poster shows the details of the responses to the player. Alternatively in the advertising session, the player can skip some questions (only the name and category of the venue are mandatory).

The last result that can be obtained from the wheel is the "Skip" action, presenting the opportunity to the player of returning to the map without taking any action.

2.3. The purpose of Urbanopoly

2.3.1. Human Computation Objectives

The main purpose of the Urbanopoly game is the quality assurance on urban (linked) data by exploiting a social sensing approach via Human Computation. To this end the game starts from a pre-existing open dataset of geospatial information, the popular OpenStreetMap project that creates and provides free geographic data and mapping.

The OpenStreetMap data is collected in a wiki mode from the community and rely on a crowdsourcing mechanism, however this approach suffers from all the problems that affect several community efforts like varying or scattered contributions, imprecise or outdated information or different levels of granularity.

In order to overcome these problems the main objectives of Urbanopoly from a Human Computation perspective are:

- To verify the data already provided by the OpenStreetMap community to validate the correct data and to identify the incorrect or doubtful data.
- To identify and correct the pre-existing OpenStreetMap data when imprecise or inaccurate data are recognized.
- To collect new information to complement the OpenStreetMap data, gathering both the pre-defined information about geo-spatial elements and additional data [11].

2.3.2. Task and Game Alignment

Urbanopoly was designed to generalize the approach to solve a single Human Computation task to a larger set of tasks. However different assignments represent different challenges for the user involved in their resolution, for this reason Urbanopoly was designed as a multifaceted game in which each Human Computation task is encapsulated in a different “mini-game”.

The Human Computation tasks and the respective mini-games within the Urbanopoly gameplay can be categorized as follows:

- <u>Data collection tasks</u> are encapsulated in creative challenges, the player is asked to provide a number of elementary inputs to create a complex artifact like an advertisement poster.
- <u>Data validation tasks</u> are presented in two forms, in the Quiz challenges the options to be validated are presented together with possible alternative values and the player is asked to select the “right” answer; this is aimed to validate the individual assertion and in the Judge session a set of information about the same entity is presented as a single artifact, a poster in this case, and the player is asked to judge the artifact on a rating scale, this is aimed to validate a set of assertions.

- <u>Data ranking tasks</u> are again proposed as quizzes where some alternative options are presented to the player who is asked to select the “best” one, like the most representative image of a place, between two or more photos.

Urbanopoly puts together those mini-games in a meaningful way, so the players are motivated to address and solve them to proceed and become successful in the game. In this way, players are requested to solve all different tasks in different moments of the game and can be rewarded differently within the game on the basis of the effort required to solve the task [12].

2.3.3. The Social Incentive

Urbanopoly uses Facebook as social media channel, so players log in the game with their account and can share their game achievements by posting success messages on their wall. Being a location-based game, those messages work also as a mechanism to share the player position with their friends. Moreover, in the leaderboard, a player can compare his own game performance with friends and thus be stimulated to continue playing to beat them.

Additionally the public sharing of game achievements on a players wall acts as an advertising for the game, letting Urbanopoly to be known to a larger audience and also offering an easy way to uniquely identify players, working as a sort of authentication authority.

3. THE URBANOPOLY WEB CASE

3.1. Requirement Analysis

3.1.1. Overall Description

In order to diversify the game and to provide the alternative for new users or enable the users to access to the game from a different platform, the Urbanopoly development team have proposed an alternative web client as the first step towards the evolution of the game, thinking ahead in a greater number of users and several different platforms to play.

3.1.1.1. Use-Case Model Survey

The following use cases have been defined taking like reference the original design of the mobile application and the mobile version itself, as many of the features originally considered were not implemented or were modified someway for the final version.

- <u>Login / Signup</u>
  It requires the user to authenticate to the game through a valid Facebook account since it is using the Facebook API as delegated authentication mechanism, if the user is logging in for first time the game will request the necessary permissions and then it will register the user on the data server to store its game data.

- <u>Map Interaction</u>
  Allows the player to interact with the properties around its geographical position if is available or alternatively with the properties of any of the default locations available in the game.

- <u>Game Wheel</u>
  The game wheel is the mechanism to play with the different options of the game, when a player spins the wheel the application returns one of five different alternatives which are described below

  Take: The player wins the opportunity to take a property having the chance to decide whether or not to take it, in the first case the player must pay 150% of the value of the property and the application will proceed to the payment phase.

Pay: The player must pay a rent to the property owner.

Quiz: The player is subjected to a series of questions about the property, obtaining a reward for each of the responses; the game rewards the activity answering the questions, in other words the game not discriminate between right or wrong answers.

Advertise: The player is subject to make a promotional poster for the property, the game will ask the player some optional data related to the property and to provide a photograph, the player will get a reward for the amount of data provided.

Skip: The player gains the ability to pass through the property without paying the rent and additionally earn a small amount of money.

- Mortgages Management
  Allows a player to mortgage a property obtaining 50% of its total value, however if the player does not redeem the mortgage within the next 24 hours it will lose the property leaving it free to buy by another player.

- Payments
  Allows the player to make payments on the two possible cases, when the player decides to purchase a property or it have to pay the rent of a property.
  In any case the player must have enough money to pay the value, if the player have not enough money the game allow to earn money by selling or mortgaging of one or more of the properties owned by the player.

- Properties Management
  Enables players to buy free properties in its location or sell its own properties in any case the game will proceed to the payment phase performing the money calculation according to the action performed and the property value.

- Judge
  The game provides this functionality to allow players to evaluate the promotional posters produced by other players for their venues; the game rewards players who perform the evaluation and allows players to report the misuse of these posters.

- Share
  Allows players to share the purchase of a venue using Facebook in order to be seen by friends encouraging new players to enter the game.

3.1.1.2. Assumptions and Dependencies

For the development of the web version of Urbanopoly the following assumptions and dependencies were considered

- The web version of Urbanopoly should be an alternative to the mobile version of the game so all the operations must have the same behavior and must not offer additional features to give any advantage to players of either platform.
- The web version like the mobile version depends of the same data server, so the information used is stored on a central component and displayed in the same way on all platforms.
- The web client of the game has been developed under a defined version of the tools and components, the eventual upgrade or changes in these components could represent changes in the behavior of the game so any update process should be considered in detail.
- At the time of editing this document the web version of the game is subject to testing, however is considered a full functional prototype and it's an artifact in a very close shape to a production-ready application. The eventual release of the game on its web version is subject to responsibility of the development team and the quality assurance team of Urbanopoly.

3.1.2. Specific Requirements

3.1.2.1. Use-Cases Overview

This section lists the identified use cases according to the UML standard notation.

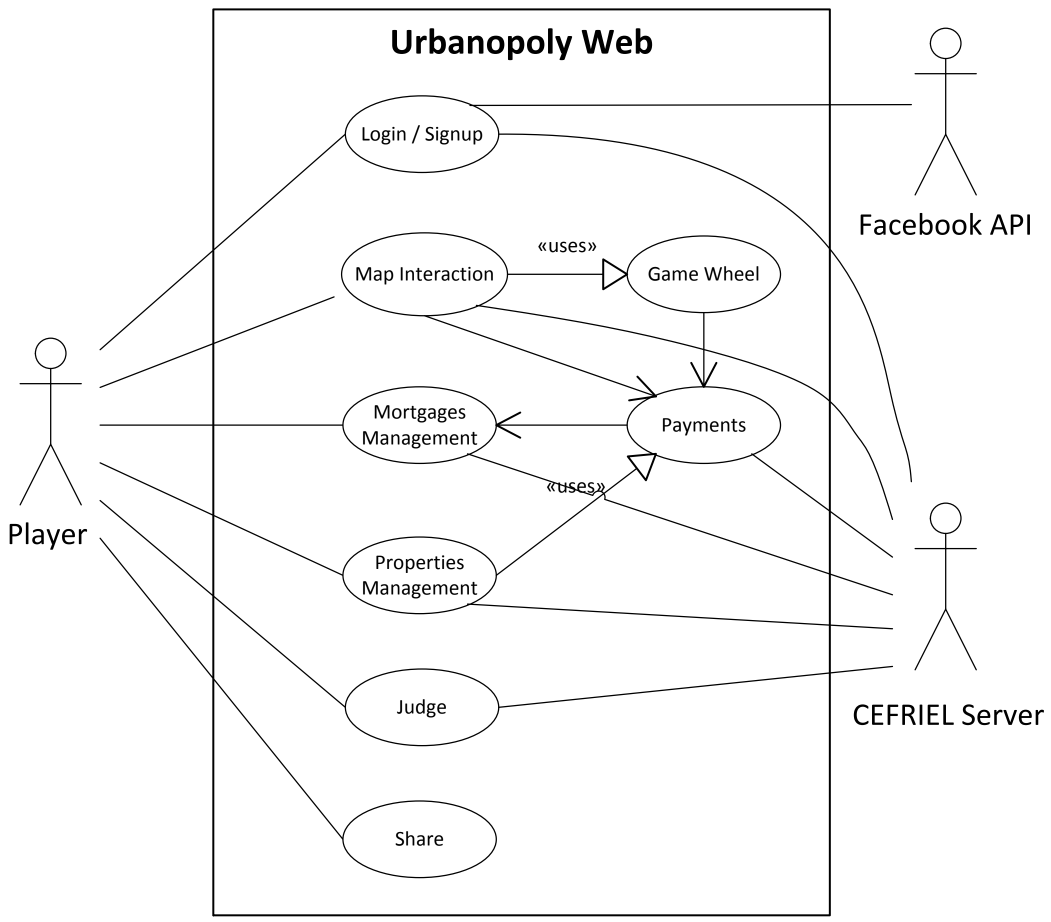


Figure 10 Urbanopoly Web use-case diagram

3.1.2.2. Use-Case Activity Diagrams

The different activity diagrams for use cases identified for the application are presented in this section

- Login / Signup

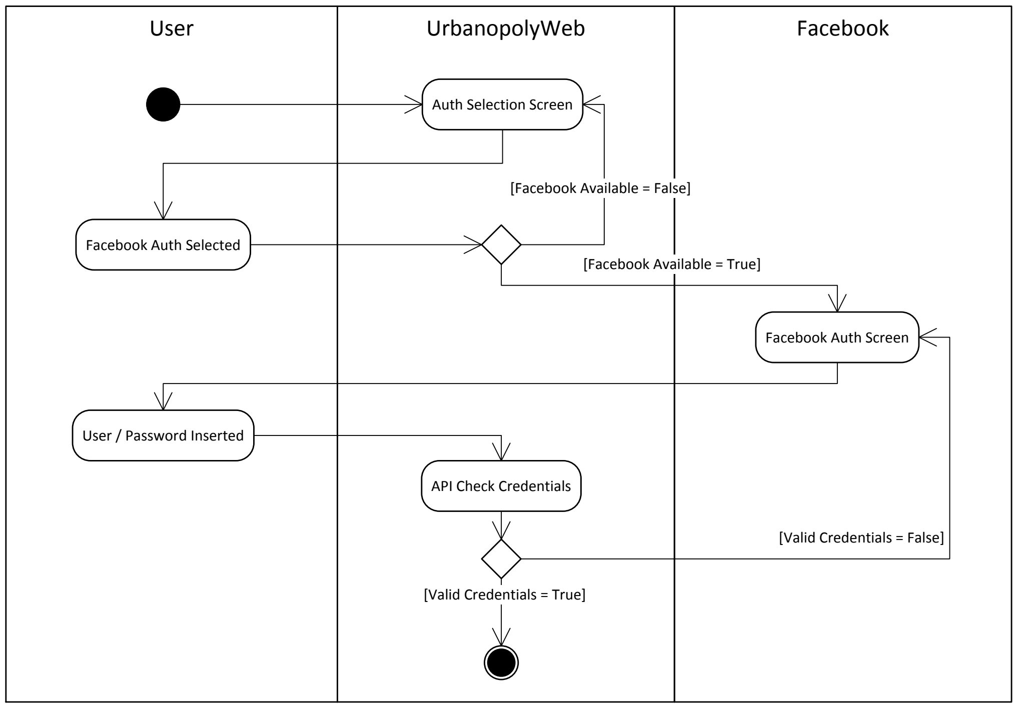


Figure 11 Login/Signup Activity Diagram

The interaction with the application starts by selecting an authentication method, in this case Facebook, then the system confirms the authentication alternative is available and redirects to the login screen, once the credentials are introduced these must be validated enabling the access on the success case or otherwise, returning to the login screen for a new attempt.

- Map Interaction

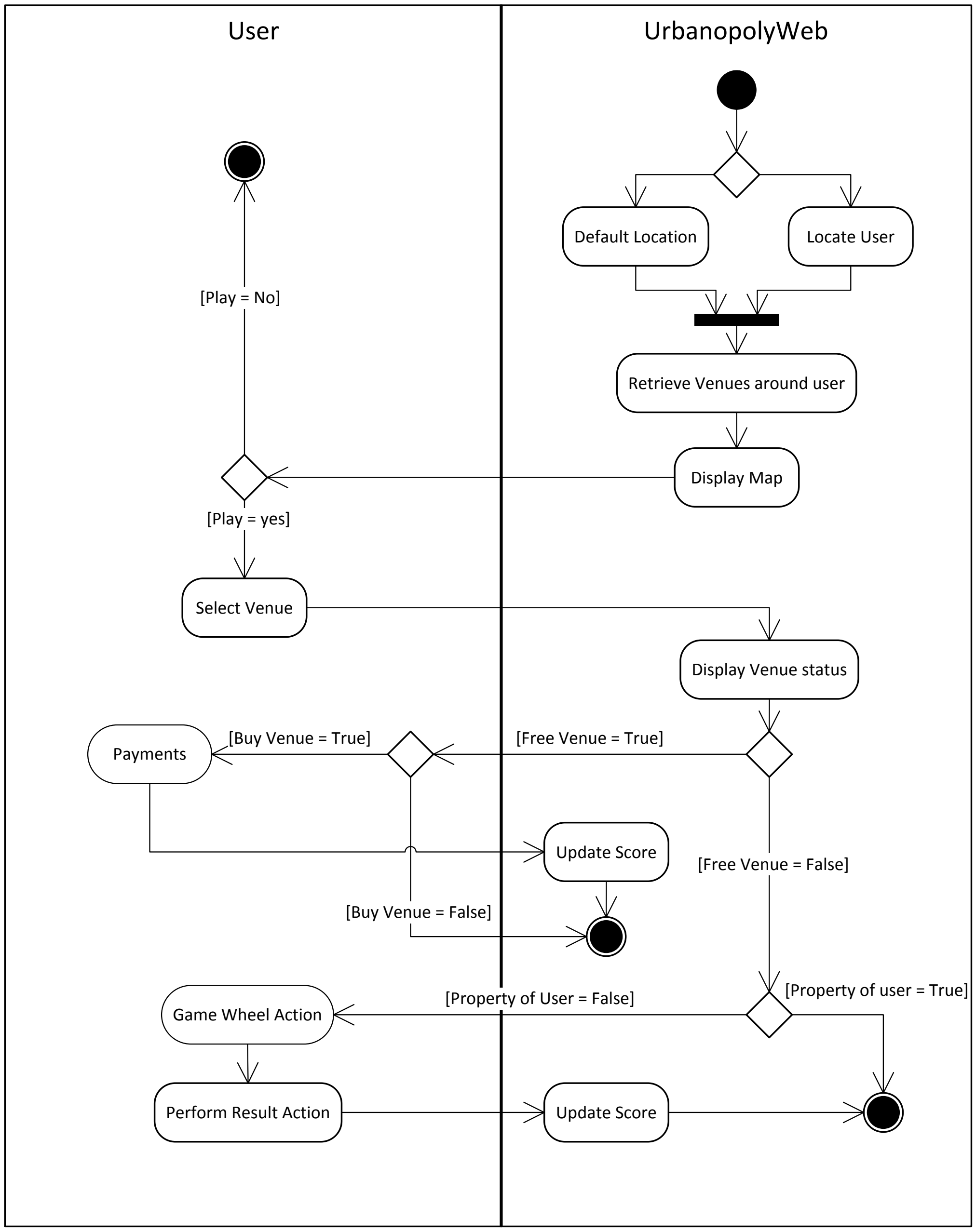


Figure 12 Map Interaction Activity Diagram

The map interaction begins with the retrieving of the geographic location of the player, this can be detected by the browser under the authorization of the user or alternatively can be one of the default locations provided by the game, in any case the application will display a set of nearby venues to play. Among this venues the user selects one of his interest, if the venue is free the player will have the opportunity to buy it or alternatively if the venue is occupied the player should play with the game wheel game to determine the next game action.

- Game Wheel

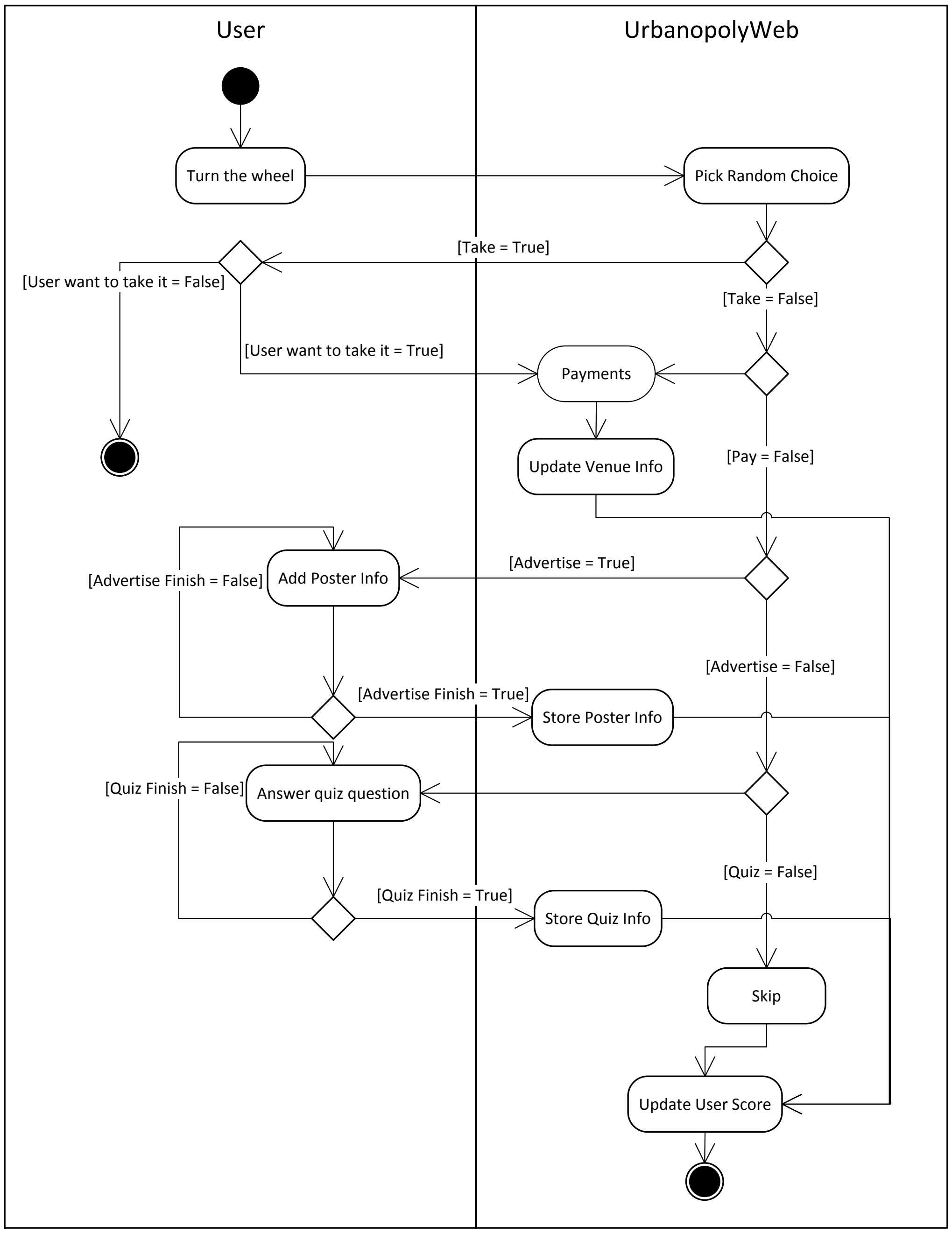


Figure 13 Game Wheel Activity Diagram

Once the player spin the game wheel, one of the five possible outcomes will be displayed on the screen, if the result is "Take" the player will decide if he wants to take the ownership of the venue making the correspondent payment, in the "Pay" case the player should pay the corresponding rent of the venue, in the "Quiz" case a set of questions available for the venue will be displayed, in the "Advertise" case the application will request to introduce some information about the venue and finally in the "Skip" case the player will be enable to return to the map without performing any action, for each alternative a reward will be granted.

- Mortgages Management

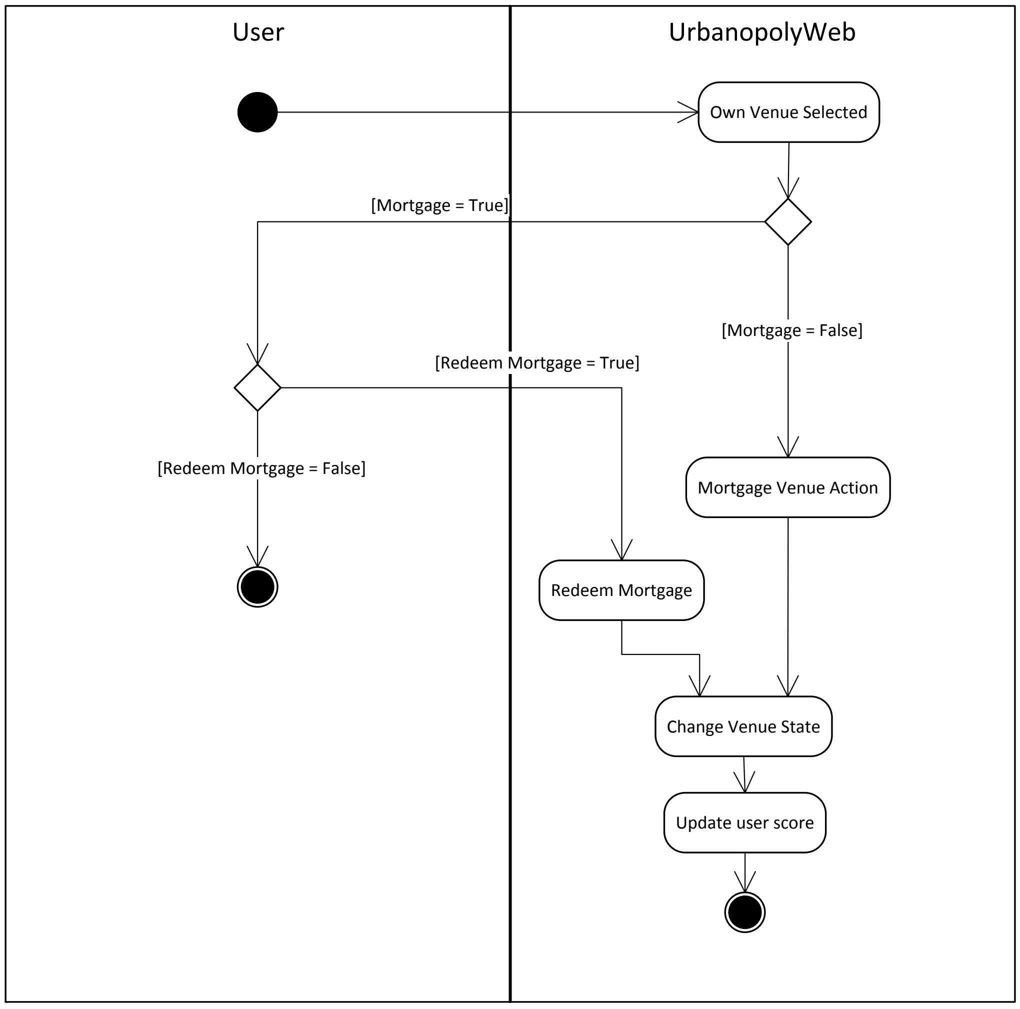


Figure 14 Mortgages Management Activity Diagram

The mortgage management is simply the action of mortgage a venue owned by the player in order to receive part of its value in cash, and the ability to redeem these mortgages by making the correspondent payment of the value received to recover the ownership of the venue.

- Payments

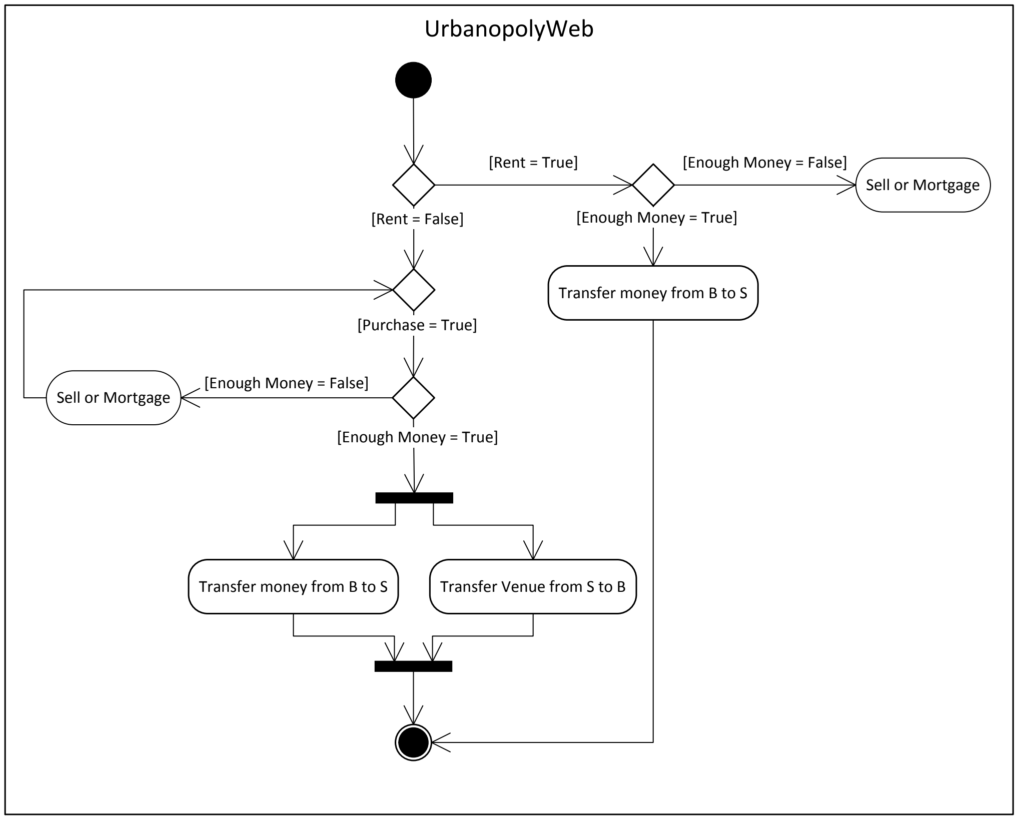


Figure 15 Payments Activity Diagram

The payment action includes transactions corresponding to sales, mortgages and the payment of rents in which the payment is made to the owner of the venue. Both operations require a previous validation of the amount of money available to perform the action.

- Properties Management

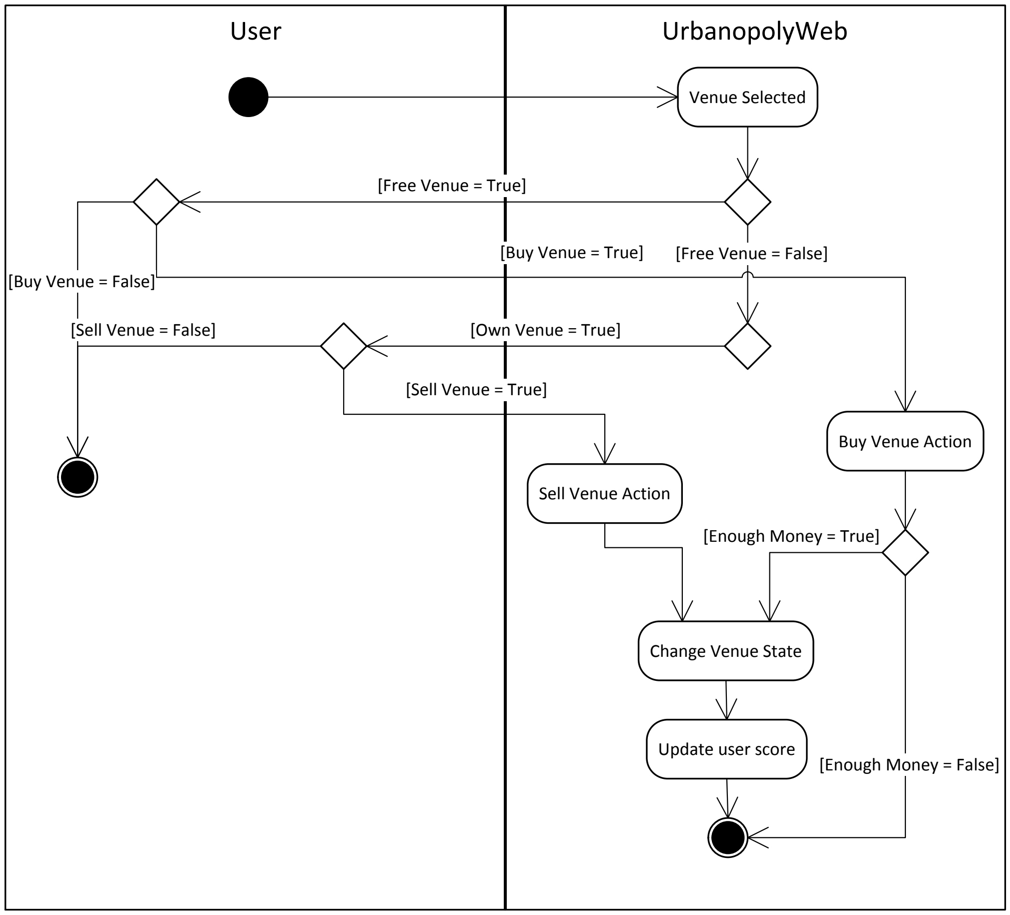


Figure 16 Properties Management Activity Diagram

In the properties management all the actions of purchase and sale of venues are handled, the player selects a venue, if it's free the player can buy it making the correspondent payment, alternately if the venue is already owned by the player he can sell it and receive the corresponding value in cash.

- Judge

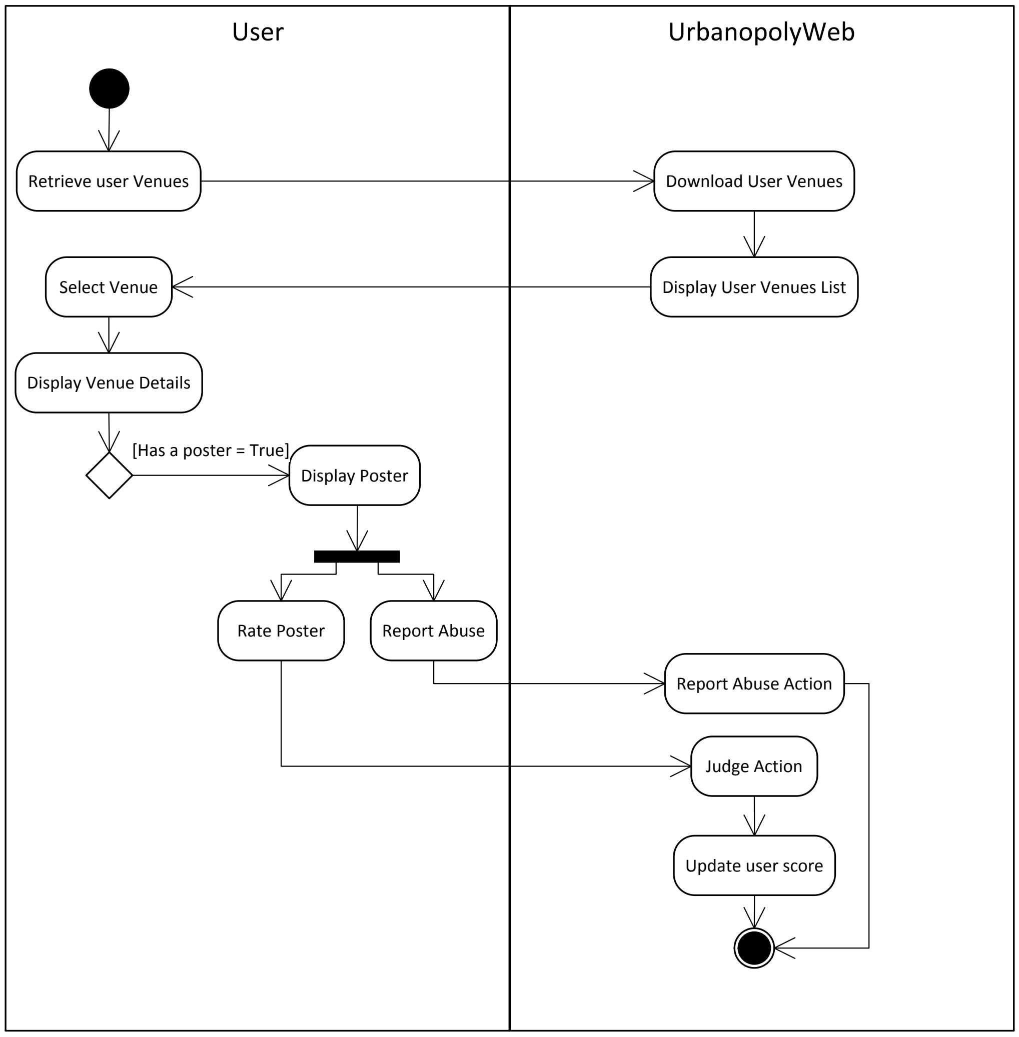


Figure 17 Judge Activity Diagram

In the judge action the player have the possibility of review all the venues he owns, the system performs the query and displays them on the screen, then the player can select the one he want to review and if there's a poster available for the venue (made across the advertise action) he can check it and provide an evaluation according to his criteria using the star rating tool available or if its required to report the abuse of the functionality.

- Share

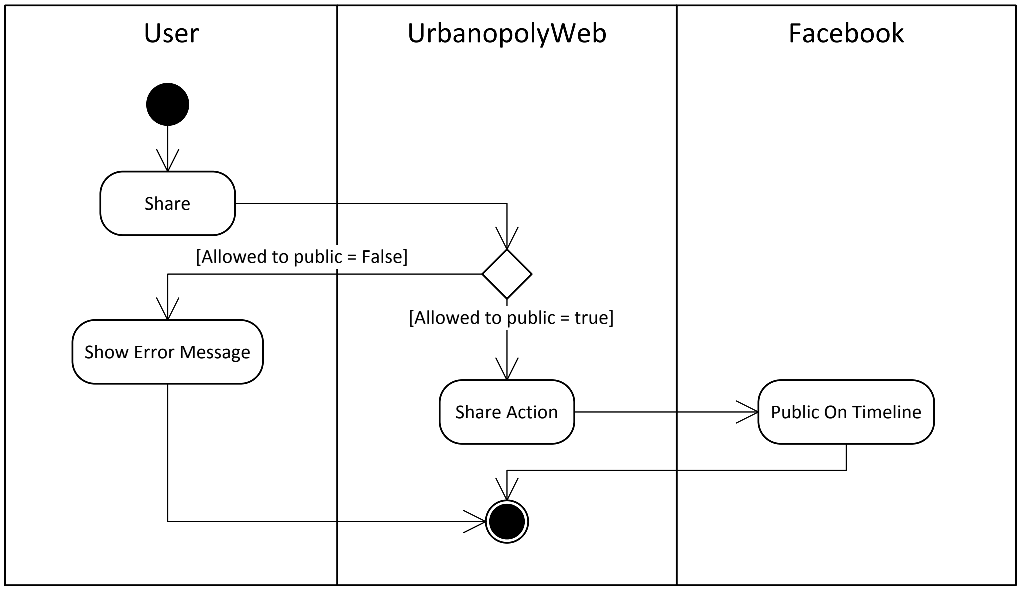


Figure 18 Share Activity Diagram

In the case of the Share action, the player selects the option share in the application; this option verifies if the permits to post on player's Facebook wall are available and if this is the case, the applications proceeds to publish the action on Facebook.

3.2. Architecture Definition

In order to develop a web client that conserves Urbanopoly dynamics, usability and responsiveness as its mobile version, different technologies have been considered for the development, their advantages and disadvantages have been evaluated for the project resulting in the selection presented in this section. A comparative figure of the architecture between the original version and the web version is presented below; it is noteworthy that the most of the processing in the original version is performed on the mobile client, while in the web version this processing is performed mainly on the web server.

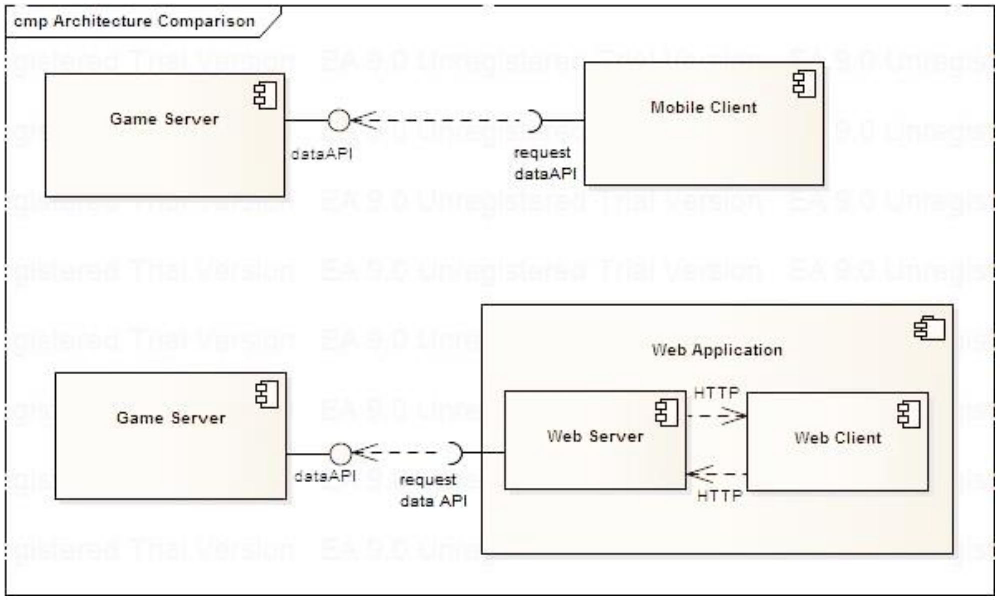


Figure 19 Architecture Comparison

Urbanopoly Web is an application fully implemented on the JavaScript programming language over an infrastructure supported by Node.js and its architecture is based on the Model-View-Controller pattern. In the following paragraphs the detail of the selected technologies is presented for both the server side and the client side.

3.2.1. Server Side

At next an overview of the technologies selected on the server side is presented with a brief description of each one of them.

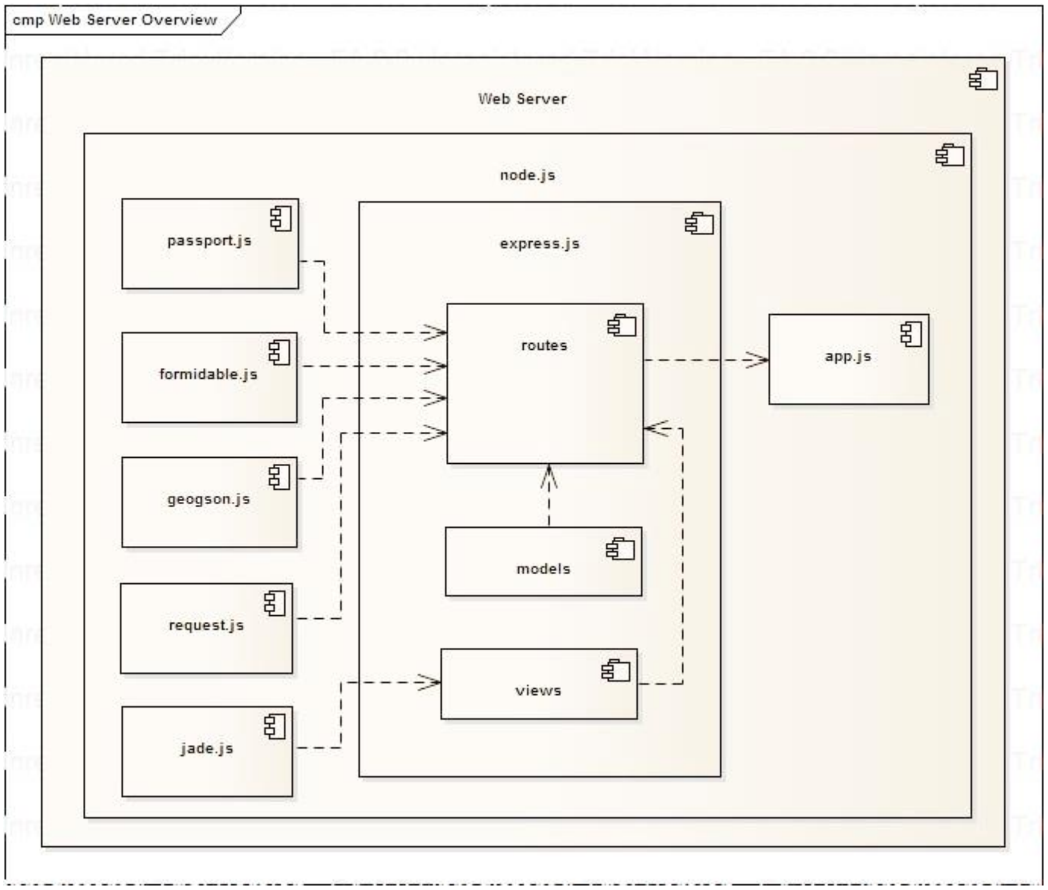


Figure 20 Web Server Overview

3.2.1.1. Node.js

Node.js is a server-side software system designed for writing scalable Internet applications, notably web servers. Programs are written on the server side in JavaScript, using event-driven, asynchronous I/O to minimize overhead and maximize scalability. Node.js contains a built-in HTTP server library, making it possible to run a web server without the use of external software, such as Apache or Lighttpd and allowing more control of how the web server works. Node.js enables web developers to create an entire web application in JavaScript, both server-side and client-side [13].

3.2.1.2. Express.js

Express.js is a minimal and flexible node.js web application framework, providing a robust set of features for building single and multi-page, and hybrid web applications.

Like every other framework express handles some complex characteristics common to all web applications like the http request handling, middleware and routing providing a pre constructed structure for web applications [14].

3.2.1.3. Passport.js

Passport is authentication middleware for Node. It is designed to serve a singular purpose: authenticate requests, so passport delegates all other functionality to the application. This separation of concerns keeps code clean and maintainable, and makes this component extremely easy to integrate into any application.

In modern web applications, authentication can take a variety of forms. Traditionally, users log in by providing a username and password. However with the rise of social networking, single sign-on using an OAuth provider such as Facebook or Twitter has become a popular authentication method and services that expose an API often require token-based credentials to protect access.

Passport recognizes that each application has unique authentication requirements and provides different authentication mechanisms, known as strategies for a huge variety of social providers packaged as individual modules so the applications can choose which strategies to employ, without creating unnecessary dependencies [15].

3.2.1.4. Jade

Jade is a web template engine for node.js enabling a flexible presentation management across the use of functions and programming logic as conditionals and loops to generate html code to be consumed by the client, as any other web template engine Jade encourages separation of business logic from the presentation logic making the development and deployment of web applications more flexible and easily maintainable [16].

3.2.2. Client Side

In this section an overview of the technologies selected on the client side is presented with a brief description of each one of them.

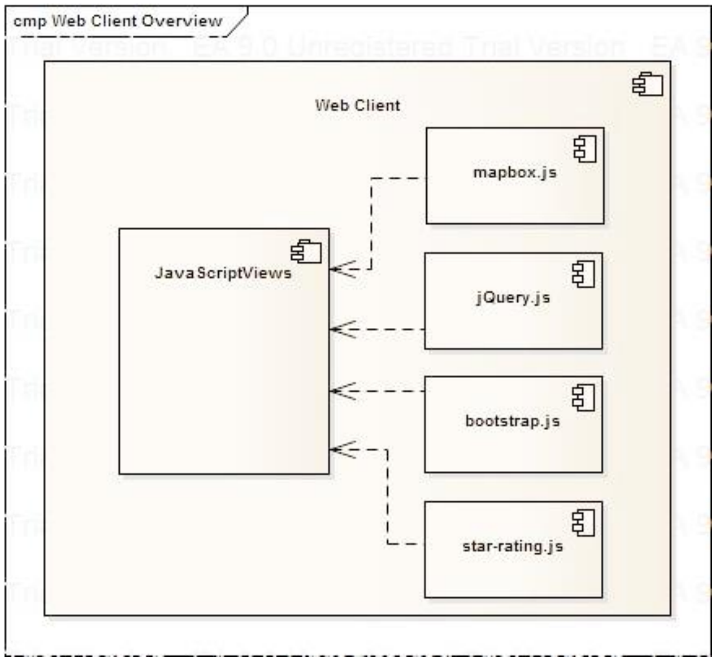


Figure 21 Web Client Overview

3.2.2.1. Leaflet

Leaflet is a modern open-source JavaScript library for mobile-friendly interactive maps. It has all the features most developers ever need for online maps and it was designed with simplicity, performance and usability in mind. It works efficiently across all major desktop and mobile platforms out of the box, taking advantage of HTML5 and CSS3 on modern browsers while still being accessible on older ones. It can be extended with many plugins and has a beautiful, easy to use and well-documented API with a simple and readable source code [17].

3.2.2.2. Mapbox.js

Mapbox is a one of the biggest providers of custom online maps for major websites and is the creator of, or a significant contributor to many popular open source mapping libraries and applications, including the MBTiles specification, the TileMill cartography IDE, the Leaflet JavaScript library, the CartoCSS map styling language and parser, and the mapbox.js JavaScript library. This last one enable the use of beautiful full customizable maps making easy the development of geographic compatible applications across a well-documented API and a huge active community of programmers [18].

3.2.2.3. JQuery

JQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. Its syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications and also to provide capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, theme-able widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and web applications [19].

3.2.2.4. Bootstrap

Bootstrap is a free collection of tools for creating websites and web applications. It contains HTML and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Bootstrap is compatible with the latest versions of all major browsers and it gracefully degrades when used on older browsers such as Internet Explorer 8. Since its version 2.0 it also supports responsive web design, this means the layout of web pages adjusts dynamically, taking into account the characteristics of the device used (desktop, tablet, mobile phone) [20].

3.2.2.5. Star-Rating

Is simple yet powerful JQuery star rating plugin for Bootstrap which supports advanced features like fractional star fill and RTL input support, developed with a focus on utilizing pure CSS-3 styling to render the control the plugin uses Bootstrap markup and styling by default, but it can be overridden with any other CSS markup [21].

3.3. Data Model Definition

The data model of the web version of Urbanopoly as in its mobile version was conceived only to represent the entities provided by the ontological model defined in the data server that provides all the information for the application.

This model originally defined using Open Street Map (OSM) and LinkedGeoData (LGD) as primary data providers has been used to model the categories of the venues in Urbanopoly and the additional features were used to represent them as ontological classes in the system, the complete ontology is available on-line at *http://swa.cefriel.it/ontologies/urbanopoly*.

However at the web client level the representation of data is made across the definition of models that represent each of the entities provided by the interaction with the data server and whose attributes are presented below.

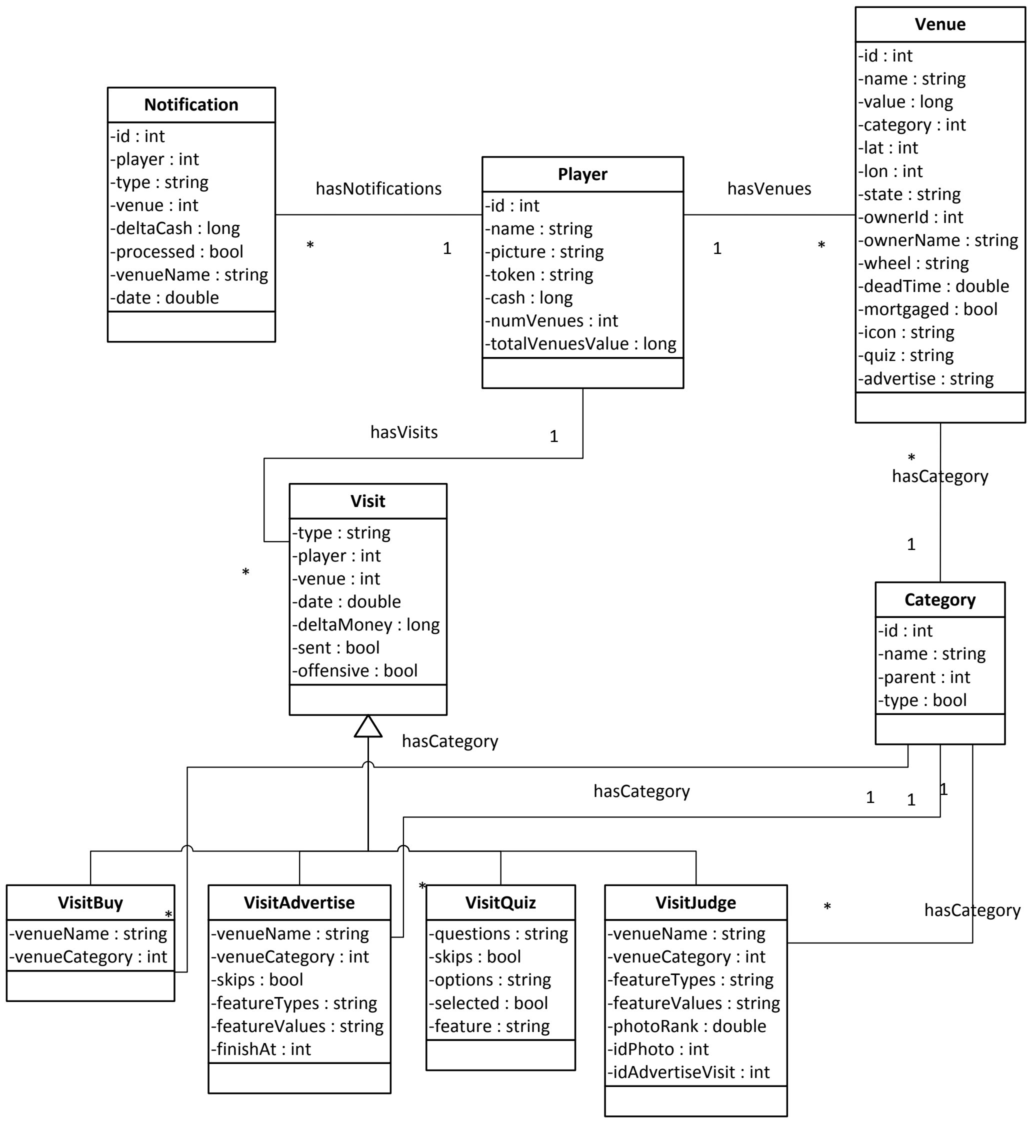


Figure 22 Data Model Overview

- Player: Entity responsible of all the data about the players including the money, the number of venues and the accumulated value in cash and on venues value.

- Venue: Entity responsible of storing the data of the venues, among the fields stand out latitude and longitude to determine the location, the state of the venue and the corresponding questions for the quiz and advertise sessions.

- Notification: Entity that stores data for the player's notifications, among the fields stand out the venue to which the notification is related, the deltaCash representing the gain or loss of money in the action and the processed field which represents if the notification has been consumed or not.

- Category: Entity representing the data of the categories, a field to reference the category parent is included.

- Visit: Entity in which all the game actions are stored, among the fields stands out the references to the player who performs the action, the venue on which the action is performed, the money involved in the transaction and whether the visit has been processed by the system or not. From this entity other types of visits are built with the specific data required for each of the actions of the game.

3.4. UI Design Proposal

This section presents a consolidated design of the user interface proposal for the web version of Urbanopoly, the design was directly influenced by the existing mobile version keeping the same look and feel and tries to keep a consistency in the different game options thinking in the player who has previously used the game making it feel comfortable during their experience in the web version and perceive that it is indeed the same game.

Below the different characteristics of the prototype are presented in order to illustrate some of the gameplay mechanics introduced in the previous sections.

3.4.1. Map Interaction

Each of the main features of the game are based on the interaction with the game map, this map shows the properties found around the geographical position of the player or any of the default locations depending on the case.

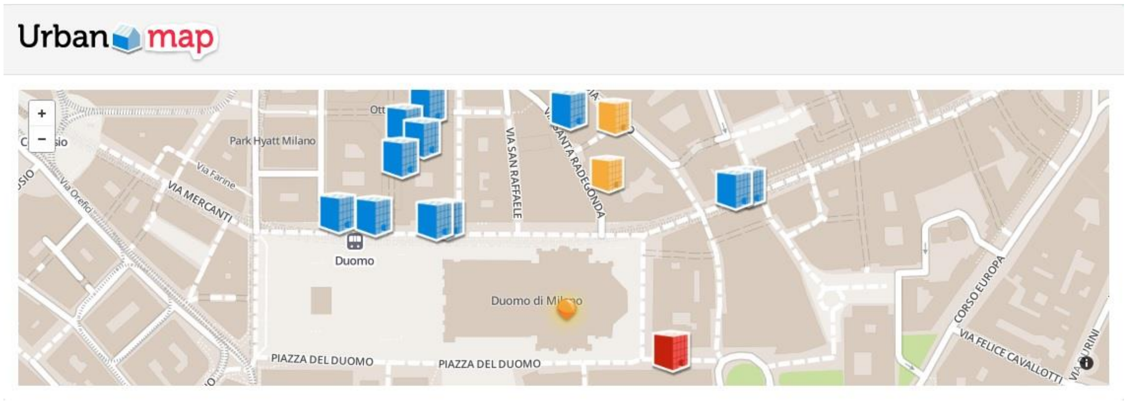
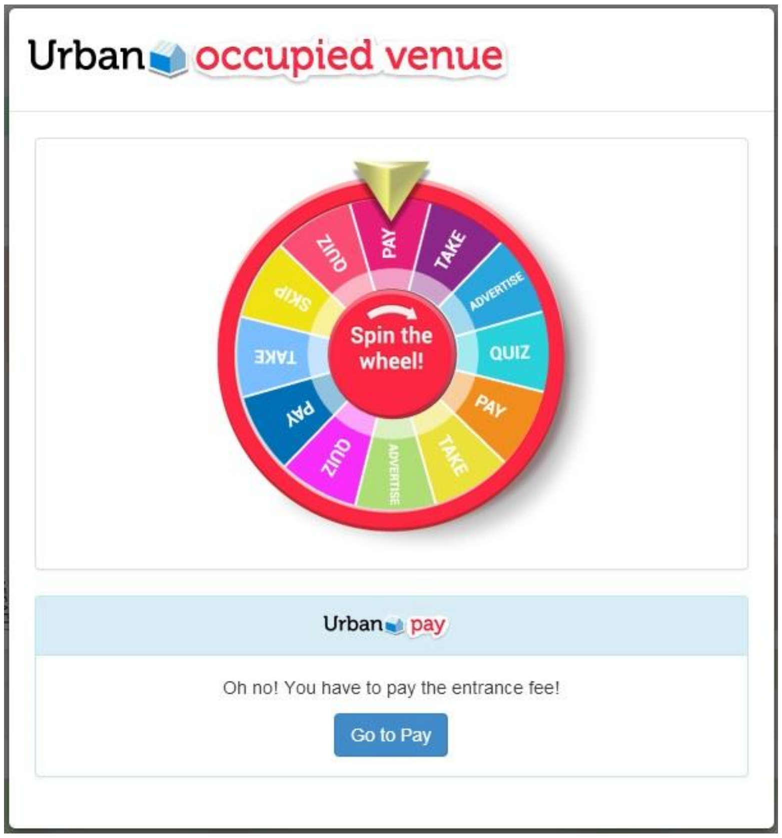


Figure 23 Map Interaction Visualization

The properties have a characteristic icon depending on its value (expensive, medium, cheap) and a characteristic color depending on their status (free / occupied, mine, mortgaged) which determines the operations available for the property.

3.4.2. Game Wheel

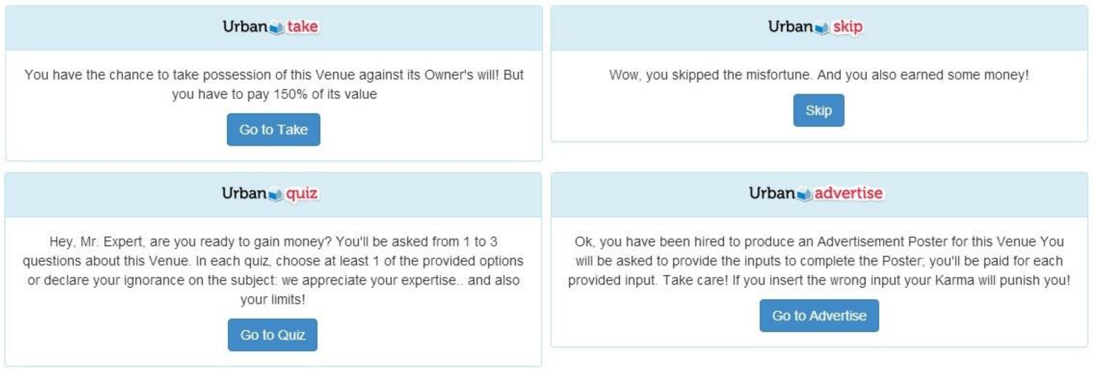
The game wheel is the main feature of the gameplay of Urbanopoly, through it the game flow is determined and possibilities of each player to become the greatest landlord in the game or just finish on bankruptcy.



Oh no! You have to pay the entrance fee!

Figure 24 Game Wheel Visualization

The results of the plays made in the game wheel are presented in the lower panel with a brief explanation of the result, of course once the wheel has determined the fate of the player they have no way to escape.



You have the chance to take possession of this Venue against its Owner's will! But you have to pay 150% of its value

Wow, you skipped the misfortune. And you also earned some money!

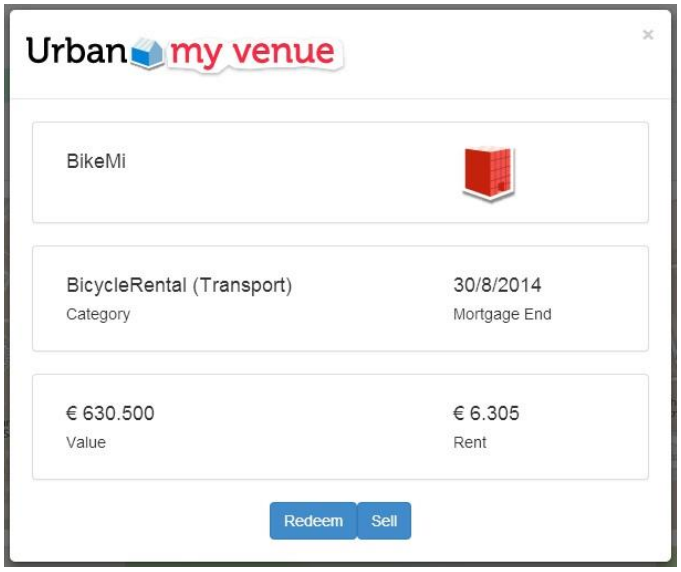
Hey, Mr. Expert, are you ready to gain money? You'll be asked from 1 to 3 questions about this Venue. In each quiz, choose at least 1 of the provided options or declare your ignorance on the subject: we appreciate your expertise.. and also your limits!

Ok, you have been hired to produce an Advertisement Poster for this Venue You will be asked to provide the inputs to complete the Poster; you'll be paid for each provided input. Take care! If you insert the wrong input your Karma will punish you!

Figure 25 Different Game Wheel Results

3.4.3. Mortgages Management

The mortgage management is summarized in two operations, given a venue this can be mortgaged by 50% of its value or given a mortgaged venue this can be redeemed by paying the mortgage value.



BikeMi

BicycleRental (Transport)
Category

30/8/2014
Mortgage End

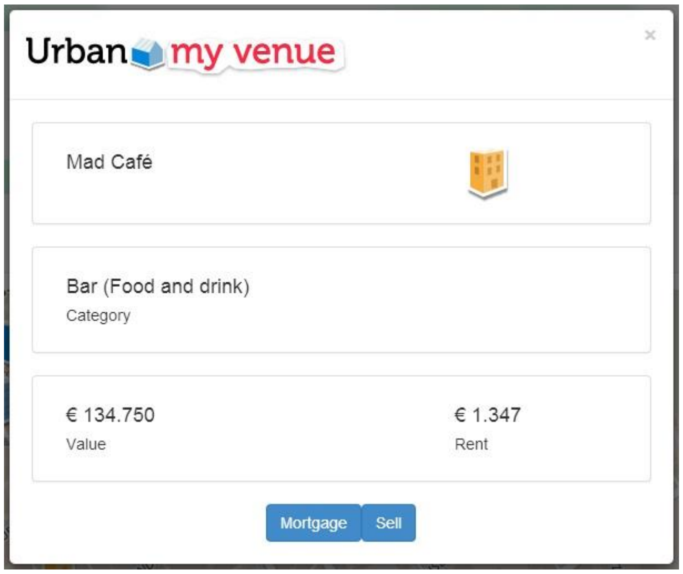
€ 630.500
Value

€ 6.305
Rent

Figure 26 Mortgages Management Visualization

3.4.4. Properties Management

The property management is basically that given a free venue this can be acquired by a player or if the venue belongs to the player this can be sold.



Mad Café

Bar (Food and drink)
Category

€ 134.750
Value

€ 1.347
Rent

Figure 27 Properties Management Visualization

3.4.5. Judge

The judge function enable the user to review a poster made it by other players for its venues and rate it according to its judgment, alternatively the user can report a player if the poster is considered offensive.

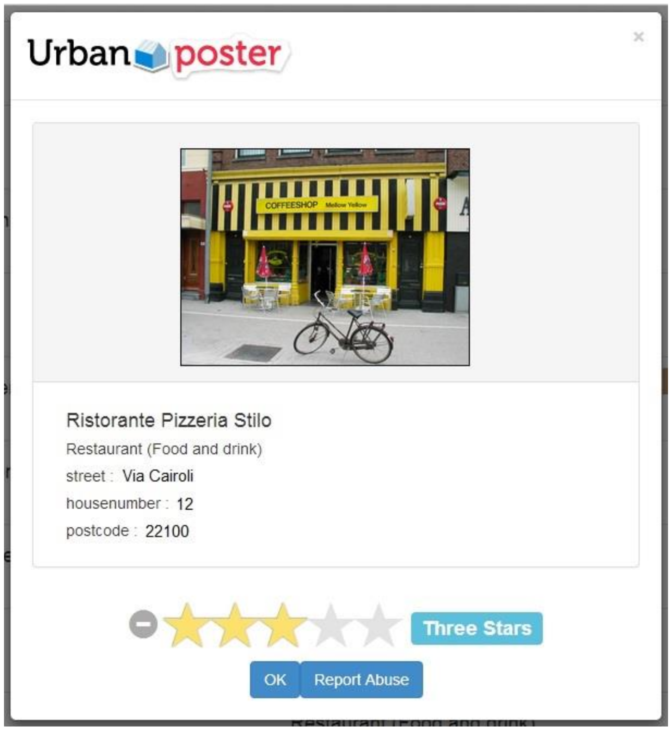

Ristorante Pizzeria Stilo
Restaurant (Food and drink)
street : Via Cairoli
housenumber : 12
postcode : 22100

Figure 28 Judge Poster Visualization

3.5. External Interactions

In the architecture of the application is expected the existence of a central server in which different clients interacts through web service type calls, the exchange of information with these services is done via JSON format requests and the details of the required parameters and the information provided by these is referenced below. For convenience these interactions have been separated on GET requests, which only retrieve information without performing any modification to the data, and POST requests, which perform modifications to the information either updating data or deleting data.

- **<u>GET Requests:</u>**

  - **GetHighscore:** Service requested to obtain the player ranking in the game. The response is always 10 players: the first 3 players in the ranking, the actual player and the 6 players around the actual player position.

    <u>INPUT:</u> uid user id, lower threshold, upper threshold.
    <u>OUTPUT:</u> vector of score objects.

  - **GetNotifications:** returns the notifications not sent to the player yet. As soon as the server sets these notifications as seen they will no longer be available to the player. There are different types of notifications and for each of these there are predefined messages to display to the player.
    <u>INPUT:</u> uid user id.
    <u>OUTPUT:</u> Array of notifications not sent yet and the status of the player updated. The server applies the notifications so the status of the player already contains the notifications applied. It serves only to notify the user of the changes applied.

  - **GetPlayArea:** The main feature of the game returns the area and the venue which the player can interact. The number of returned venues and the maximum distance is calculated on the server side based on the area of the game. Also in this case the status of the player and the venue info must be updated even if was already downloaded previously. There are also some predefined views. With this service are generated and returned the visits of QUIZ and ADVERTISE type. However the JUDGE visit must be generated by the client.
    <u>INPUT:</u> uid user id, latitude, longitude.
    <u>OUTPUT:</u> Array of venues, Array of possible visits, the day game and the player updated.

  - **GetPlayerVenues:** used to obtain the properties of the player updated.
    <u>INPUT:</u> uid user id.
    <u>OUTPUT:</u> Array of player's venues, array of visits of JUDGE type available, and the user updated.

- **GetVersion:** is the first request of the client and returns the current version of the server. If the client version is different you have to upgrade one of the two.

- **<u>POST Requests:</u>**

  - **PlayerSubscribe:** used to register the user in the application, since the second call is used every time at starting time on the app as is needed to update or create the user.
    <u>INPUT:</u> uid facebook and username.
    <u>OUTPUT:</u> If the user already exists returns the state of the player updated, if the user does not exist creates it and returns the user status again.

  - **SetVisit:** used to send the user actions to the server.
    INPUT: Array of visits. If the visit contains offensive content, send an email to the game administrator.
    OUTPUT: Status of the operation. If it fails you have to resubmit the visits.

  - **UploadPhoto:** receives and save the picture.
    INPUT: uid user id, venue id, bitmap.
    OUTPUT: Status of the operation. If the operation fails you have to perform the retry.

4. CASE REVIEW AND NEXT STEPS

The experience of implementing the web version of Urbanopoly has been an enriching experience in the aspect of learning new tools and technologies applied to the web development, however different challenges have been found on the way. In this section some of the issues encountered during the project are presented both from the methodological as from the technical point of view. Additionally some measures about the project are presented, enabling to size the effort from each one of the phases, the section ends with a series of next steps for the following versions of the game. The resulting prototype can be found on *https://github.com/labonillal/UrbanopolyWeb/tree/master/v1.5.*

4.1. Methodology Issues

4.1.1. Out of date documentation

At the beginning of the project and partly to perform the estimation of time and effort required to the implementation of the web version the project scope was considered a migration of the mobile client to a web platform; however this consideration assumed a complete and updated documentation of the mobile version which unfortunately was not the case.

The mobile version was implemented under the idea of evolutionary prototype encouraging the agile development but degrading considerably the quality of project documentation, in first place several of the use cases originally considered in the documentation were completely modified or even eliminated from the final version, consequently the corresponding artifacts of these use cases like activity diagrams, basic flows or sequence diagrams were missing in the documentation.

Because of these problems and in order to identify the features present in the application a detailed source code inspection was performed on the mobile version and the documentation necessary was produced in order to initiate the development of the web version.

4.1.2. Limited amount of comments on the source code

As mentioned in the previous section an inspection of the source code was performed like a reverse engineering exercise to produce the necessary documentation for the development of a web version of the game. However at the first look in the code was clear that any policy of code documentation was used while the code was built, while some sections present a complete description of the operations others do not have any comments at all, which may be understandable in sections such as front end classes, data models or the serialization utilities but the comments were missing also in critical sections of the game logic.

4.1.3. Inherited problems

As a result of the original game design that considers the existence of a central server for the game data storage some problems have been inherited from this design into the web version, this does not mean the architecture was designed in the wrong way but thinking about the evolution of the game eventually some changes should be made to the data server and the services it provides. Some of the features of the web version which should be modified with a possible improvement of the data server services are the following.

- New user attributes such as profile picture must be included in the data server, these attributes are currently stored in session and since they are frequently updated consuming memory resources.
- The venue categories should be stored in the data server and provided via web service as other data; currently they are defined in a text file and retrieved when the application starts the creation objects of type category.
- The visits control (restriction of a visit by day of play) should be managed via database on the central server through a relationship between the venue and the player, both to improve the visits control and to ensure that is horizontal for the mobile and the web version, so if one venue is visited on the mobile version the restriction should persist on the web version.

4.2. Technical problems

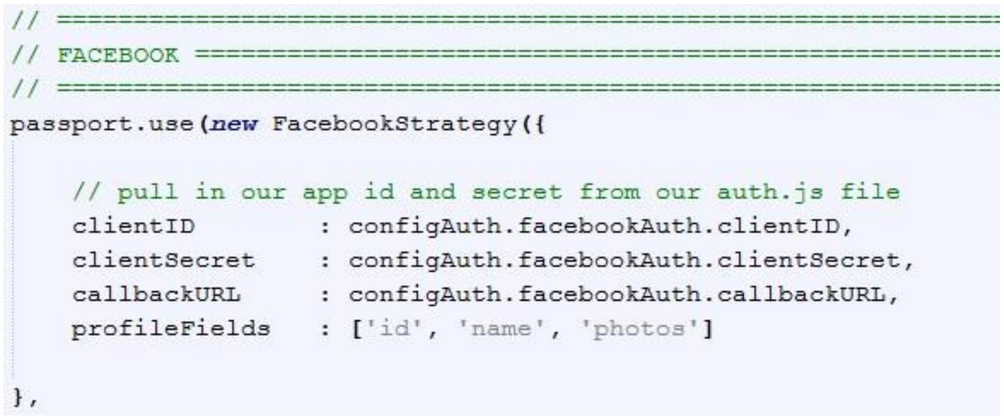
In this section some of the technical problems encountered during the development of the web version of the game are mentioned, although some are complex and others simple all of them represent a knowledge base which will be useful as a reference for those who have a similar project or have been fighting against this kind of problems in another context.

4.2.1. Displaying Facebook profile picture

Like many other applications using social authentication for the game we wanted to show the Facebook profile picture of the player inside the game interface to give a sense of personalization of the interface, however the settings of passport.js to perform this operation were pretty hard to find but quite simple.

Problem: Access the Facebook profile picture of the authenticated user.

Solution: In the parameters of initialization of the passport Facebook strategy the 'profileFields' field should be included, in this field we can define additional parameters to be returned by the Facebook API through passport and here we include the 'photos' field.



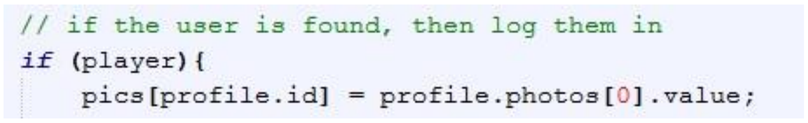
```
// =================================================================
// FACEBOOK ========================================================
// =================================================================
passport.use(new FacebookStrategy({

    // pull in our app id and secret from our auth.js file
    clientID        : configAuth.facebookAuth.clientID,
    clientSecret    : configAuth.facebookAuth.clientSecret,
    callbackURL     : configAuth.facebookAuth.callbackURL,
    profileFields   : ['id', 'name', 'photos']

},
```

Figure 29 Facebook Strategy Parameters

After this passport will include a 'photos' field in the profile object containing the user photos, however this field is an array of urls of the available photos, in order to access the profile picture we should take the value of the first element the array.



```
// if the user is found, then log them in
if (player){
    pics[profile.id] = profile.photos[0].value;
```

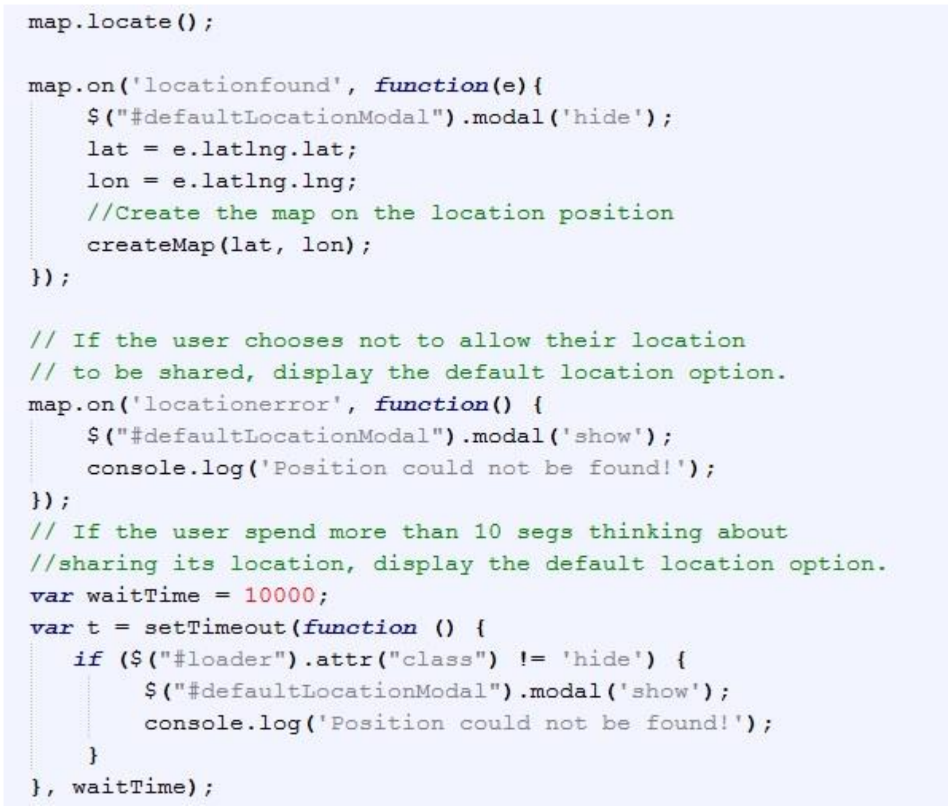
Figure 30 Select the profile picture

4.2.2. Managing Firefox location sharing

In order to access the geographic location of the user through the browser mapbox.js contains the method `map.locate()`, this method can return two callbacks `locationfound` in the case the user share its location or `locationerror` otherwise, however for the specific case of Mozilla Firefox was found that if the user selects the option "Not Now" when the browser asked to share its geographic location that does not throw the `locationerror` callback so this option works as if the user continue thinking whether or not to share its location making impossible to the game to detect the event.

Problem: The user selects the option "Not Now" when the browser asked to share their geographic location.

Solution: As the game is unable to detect if the user shared his location through the callbacks of the `locate()` method the alternative was to implement a timeout when requesting the location, so if the user does not respond or select the option "Not Now" the game will display the option to select a default location.



```
map.locate();

map.on('locationfound', function(e){
    $("#defaultLocationModal").modal('hide');
    lat = e.latlng.lat;
    lon = e.latlng.lng;
    //Create the map on the location position
    createMap(lat, lon);
});

// If the user chooses not to allow their location
// to be shared, display the default location option.
map.on('locationerror', function() {
    $("#defaultLocationModal").modal('show');
    console.log('Position could not be found!');
});
// If the user spend more than 10 segs thinking about
//sharing its location, display the default location option.
var waitTime = 10000;
var t = setTimeout(function () {
   if ($("#loader").attr("class") != 'hide') {
       $("#defaultLocationModal").modal('show');
       console.log('Position could not be found!');
    }
}, waitTime);
```

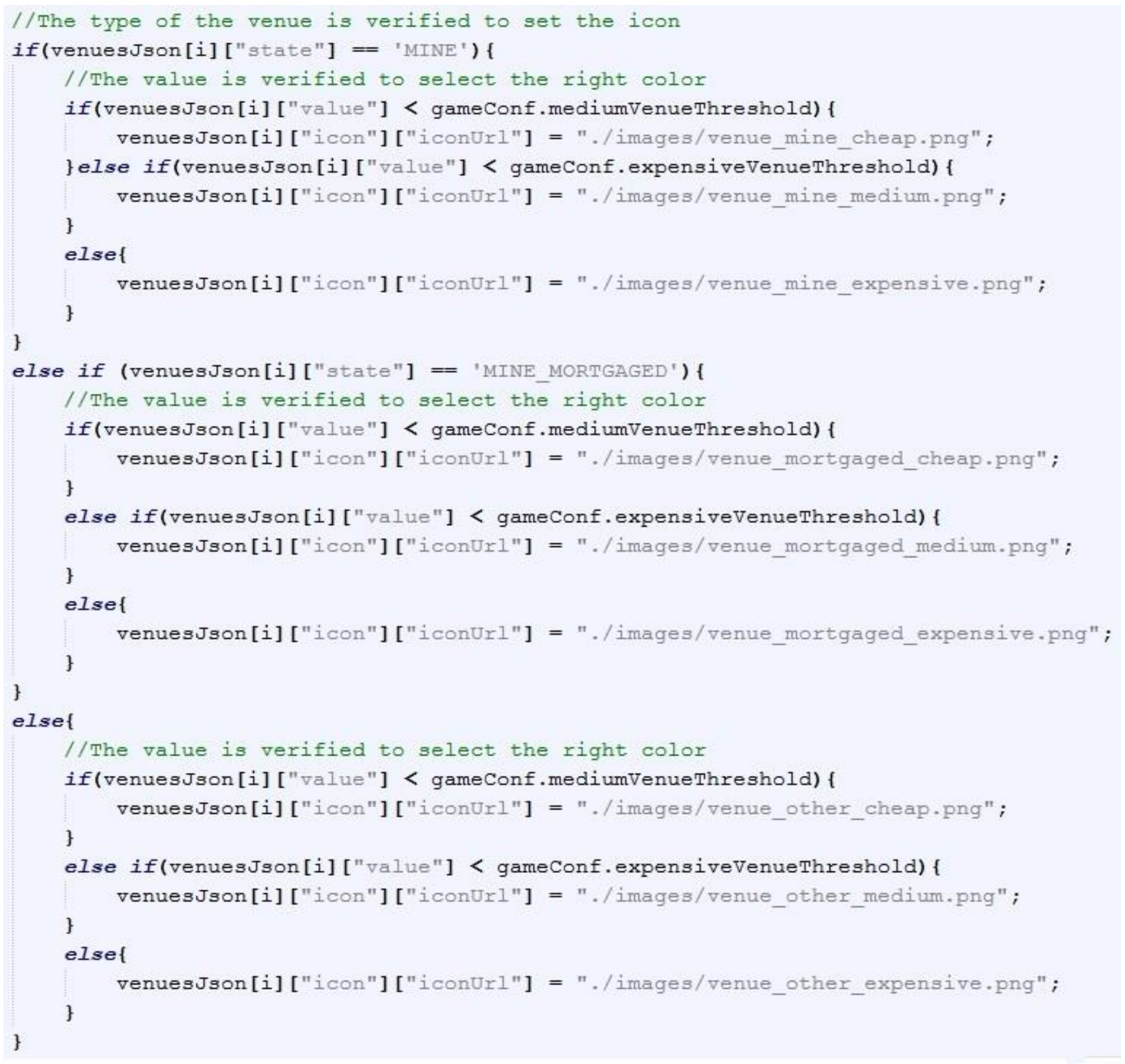
Figure 31 Location Timeout

4.2.3. Dynamic display of icons on the map

When game map is displayed the markers that represent the venues around the player must have a distinctive icon depending on its current state and value for assign icons to markers mapbox.js provides the method `marker.setIcon()` but the information corresponding to the icon must be available for all properties when the markers layer is loaded.

Problem: Information about icons should be available at the time of adding the markers layer to the map.

Solution: As the information icon have to be available when the map loads the solution implemented was once downloaded the venues information from the data server we iterate among them adding the url of the icon.



```
//The type of the venue is verified to set the icon
if(venuesJson[i]["state"] == 'MINE'){
    //The value is verified to select the right color
    if(venuesJson[i]["value"] < gameConf.mediumVenueThreshold){
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_mine_cheap.png";
    }else if(venuesJson[i]["value"] < gameConf.expensiveVenueThreshold){
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_mine_medium.png";
    }
    else{
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_mine_expensive.png";
    }
}
else if (venuesJson[i]["state"] == 'MINE_MORTGAGED'){
    //The value is verified to select the right color
    if(venuesJson[i]["value"] < gameConf.mediumVenueThreshold){
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_mortgaged_cheap.png";
    }
    else if(venuesJson[i]["value"] < gameConf.expensiveVenueThreshold){
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_mortgaged_medium.png";
    }
    else{
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_mortgaged_expensive.png";
    }
}
else{
    //The value is verified to select the right color
    if(venuesJson[i]["value"] < gameConf.mediumVenueThreshold){
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_other_cheap.png";
    }
    else if(venuesJson[i]["value"] < gameConf.expensiveVenueThreshold){
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_other_medium.png";
    }
    else{
        venuesJson[i]["icon"]["iconUrl"] = "./images/venue_other_expensive.png";
    }
}
```

Figure 32 Icon Management

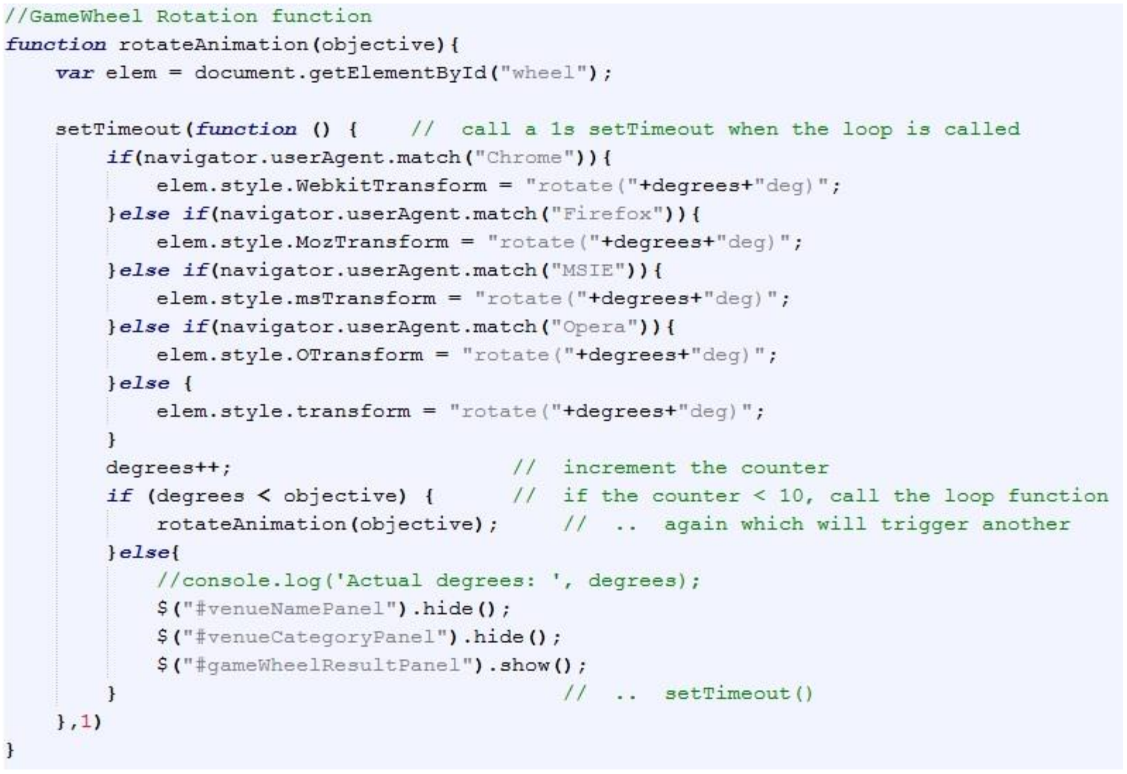
In this way the icon data will be available in the properties of each of the markers on load time so the icon can be added to each marker when the markers layer is added to the map.

4.2.4. Cross-Browser Game Wheel animation

The game wheel is the most important component of the game, this component determines the plays of the users and its animation have two key challenges, the rotation had to be smooth for all browsers and the desired result had to be unequivocally indicated by the pin at the top of the image.

Problem: Make the game wheel animation smoother on all the most popular browsers and correctly indicate the result on every play.

Solution: In the absence of a utility function as in the case of `android.view.animation.RotateAnimation` in the mobile version and to ensure the compatibility with all the browsers the following custom rotation function was implemented for the game.



```
//GameWheel Rotation function
function rotateAnimation(objective){
    var elem = document.getElementById("wheel");

    setTimeout(function () {    //  call a 1s setTimeout when the loop is called
        if(navigator.userAgent.match("Chrome")){
            elem.style.WebkitTransform = "rotate("+degrees+"deg)";
        }else if(navigator.userAgent.match("Firefox")){
            elem.style.MozTransform = "rotate("+degrees+"deg)";
        }else if(navigator.userAgent.match("MSIE")){
            elem.style.msTransform = "rotate("+degrees+"deg)";
        }else if(navigator.userAgent.match("Opera")){
            elem.style.OTransform = "rotate("+degrees+"deg)";
        }else {
            elem.style.transform = "rotate("+degrees+"deg)";
        }
        degrees++;                      //  increment the counter
        if (degrees < objective) {      //  if the counter < 10, call the loop function
            rotateAnimation(objective);     //  ..  again which will trigger another
        }else{
            //console.log('Actual degrees: ', degrees);
            $("#venueNamePanel").hide();
            $("#venueCategoryPanel").hide();
            $("#gameWheelResultPanel").show();
        }                                   //  ..  setTimeout()
    },1)
}
```

Figure 33 Game Wheel Rotation Function

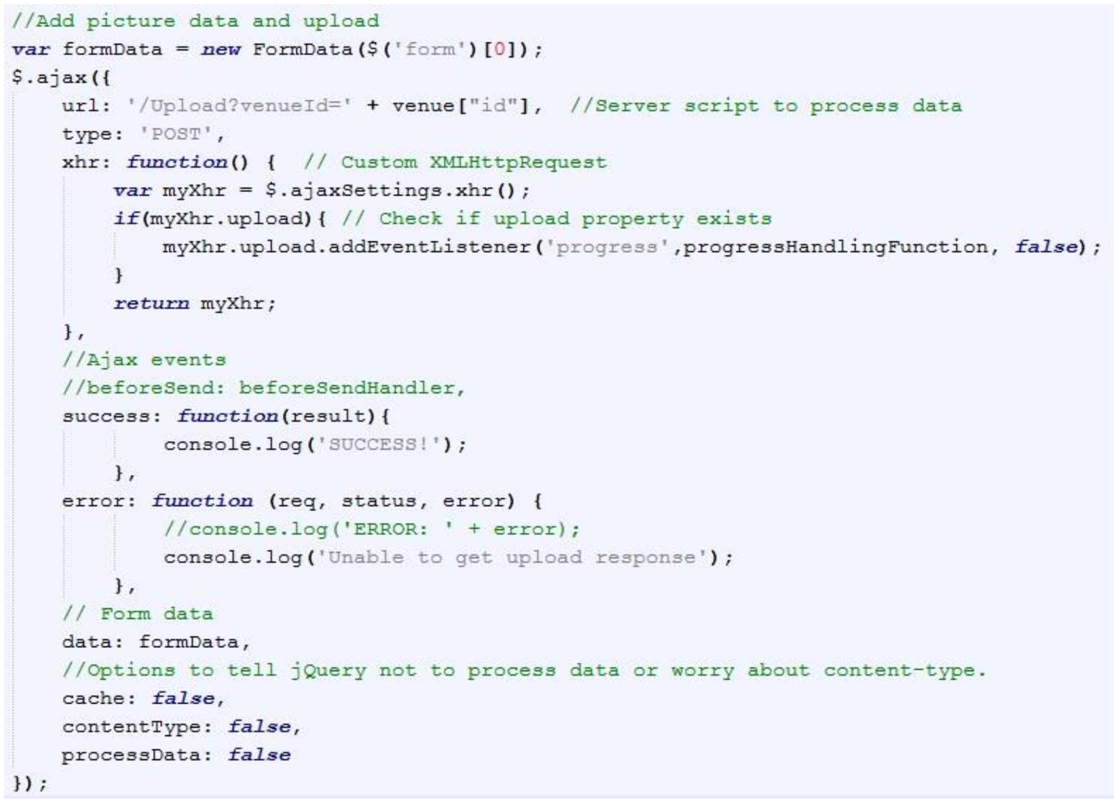
This function receives a parameter `objective` that represents the value in degrees where the animation should stop, this value was previously determined for each of the possible outcomes ensuring the wheel game will always indicate the correct value in the animation.

4.2.5. Managing asynchronous file uploads

At the moment of implementing the step of upload a picture for the selected venue an unexpected difficulty arises by default to upload a file the html form should be submitted and therefore a page reload should be performed which was unacceptable to the game because this step is part of a modal panel over the map interface, which would mean reloading the entire map in order to send the photo.

Problem: Upload the selected image asynchronously to avoid reloading the page to complete the process.

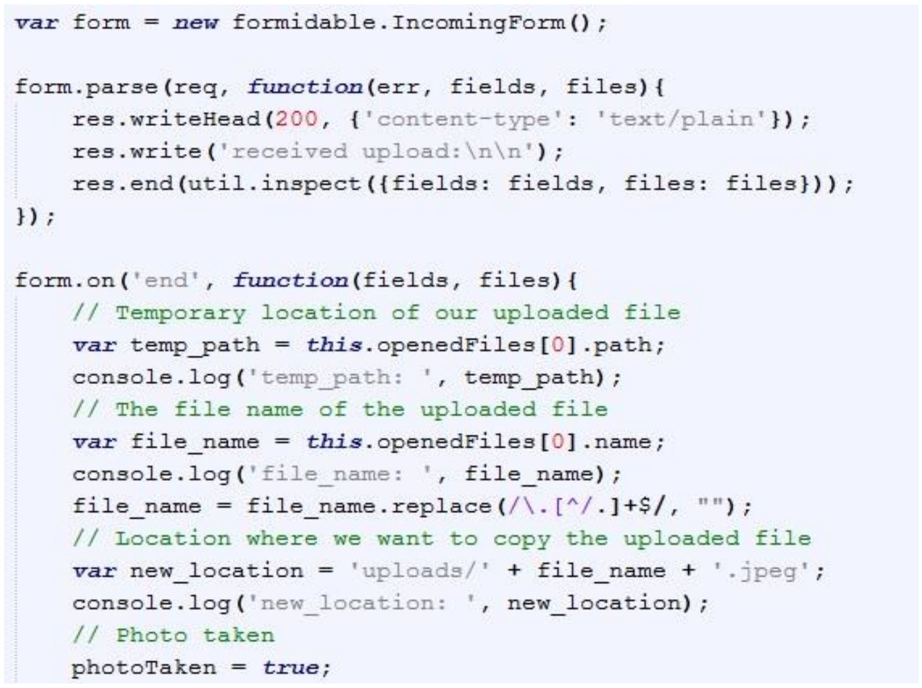
Solution: In order to perform the image upload the `FormData` interface was used to manage the form data to be sent on the ajax request this because we don't have the possibility of building an entity of type `org.apache.http.entity.mime.MultipartEntity` as in the case of the mobile version.



```
//Add picture data and upload
var formData = new FormData($('form')[0]);
$.ajax({
    url: '/Upload?venueId=' + venue["id"],  //Server script to process data
    type: 'POST',
    xhr: function() {  // Custom XMLHttpRequest
        var myXhr = $.ajaxSettings.xhr();
        if(myXhr.upload){ // Check if upload property exists
            myXhr.upload.addEventListener('progress',progressHandlingFunction, false);
        }
        return myXhr;
    },
    //Ajax events
    //beforeSend: beforeSendHandler,
    success: function(result){
            console.log('SUCCESS!');
        },
    error: function (req, status, error) {
            //console.log('ERROR: ' + error);
            console.log('Unable to get upload response');
        },
    // Form data
    data: formData,
    //Options to tell jQuery not to process data or worry about content-type.
    cache: false,
    contentType: false,
    processData: false
});
```

Figure 34 Upload photo request

On server-side in order to save the image with the corresponding venue and player identifier the `Formidable` module was used as parsing utility to manage the information on the forms sent by the client.



```
var form = new formidable.IncomingForm();

form.parse(req, function(err, fields, files){
    res.writeHead(200, {'content-type': 'text/plain'});
    res.write('received upload:\n\n');
    res.end(util.inspect({fields: fields, files: files}));
});

form.on('end', function(fields, files){
    // Temporary location of our uploaded file
    var temp_path = this.openedFiles[0].path;
    console.log('temp_path: ', temp_path);
    // The file name of the uploaded file
    var file_name = this.openedFiles[0].name;
    console.log('file_name: ', file_name);
    file_name = file_name.replace(/\.[^/.]+$/, "");
    // Location where we want to copy the uploaded file
    var new_location = 'uploads/' + file_name + '.jpeg';
    console.log('new_location: ', new_location);
    // Photo taken
    photoTaken = true;
```

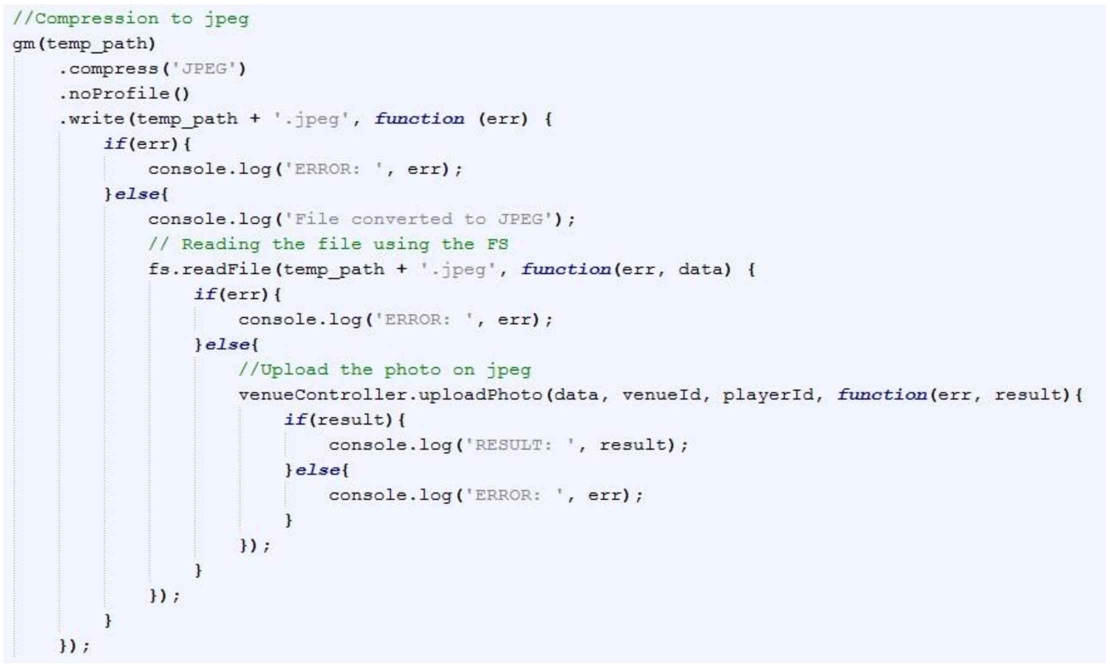
Figure 35 Server side photo manage

4.2.6. Enabling image compression

After receiving the image on the server side another challenge arises, the image had to be compressed before being sent to the data server as this restricts the size of it forcing its reduction, this made each of the photos sent to be transformed on some kind of image in a range of red colors, so it was necessary to the web server have some image management utility.

Problem: The image sent to the central data server must be compressed.

Solution: To perform the compression of the picture ImageMagick was used as image processing utility and was installed on the web server, in addition the `gm` module was used to access this utility in node so the image is compressed in the web server before be sent to the central data server of the game.



```
//Compression to jpeg
gm(temp_path)
    .compress('JPEG')
    .noProfile()
    .write(temp_path + '.jpeg', function (err) {
        if(err){
            console.log('ERROR: ', err);
        }else{
            console.log('File converted to JPEG');
            // Reading the file using the FS
            fs.readFile(temp_path + '.jpeg', function(err, data) {
                if(err){
                    console.log('ERROR: ', err);
                }else{
                    //Upload the photo on jpeg
                    venueController.uploadPhoto(data, venueId, playerId, function(err, result){
                        if(result){
                            console.log('RESULT: ', result);
                        }else{
                            console.log('ERROR: ', err);
                        }
                    });
                }
            });
        }
    });
```

Figure 36 Image Compression

4.3. Project Measures

In this section some metrics in relation to the development of the overall project are presented, a summary of the effort spent in hours for each of the project phases is provided and a brief description of each of the stages performed on the Urbanopoly Web project is included.

Activity	Effort (hours)
Project Management	24
Learning Process	40
Application Design	16
Application Development	170
Application Test	80
Documentation	24
Total	354

Table 1 Project Measures

- Project Management: To this phase, considered horizontally across all the project lifetime, are associated activities of logistic type and tracking tasks as the project plan elaboration, the effort estimation, progress track meetings etc.

- Learning Process: Are associated to this phase all the activities related to the acquisition of knowledge about the technologies used during the project, among them the reading, lessons, tutorials and all the different tasks that have contributed to acquire the necessary knowledge needed in the project implementation.
- Application Design: Among the design phase are counted component design activities, services and methods developed during the project as well as the diagram creation and the documentation necessary to support them.
- Application Development: In the development phase are included all the development activities of the application components including activities such as coding, unit testing, code documentation, communication implementation, and infrastructure installation and configuration management.
- Application Test: Test phase activities make reference to all the tasks to perform any kind of testing on the application, both at the end of the development phase and in the acceptance of the prototype, additionally the activities of code debugging and correction of submitted issues are considered in this phase.
- Documentation: Finally in the documentation phase are counted activities like production of documents, reports, plans and other artifacts produced during the project and which were used as the main source for the creation of this document.

4.4. Next Steps

4.4.1. Perform a comparative evaluation between versions

A similar evaluation like the originally made for the mobile version should be performed on the web version in order to determine different issues that can arise from the use of different platforms and devices. Such evaluation should measure the same characteristics considered during the evaluation of the mobile version to provide a mechanism for comparison between the two versions.

4.4.2. Increase the amount of geographical data

One of the most important improvements in the short term to be applied to the game is definitely an increase in the amount of geographic information available, in this moment the only available information is the region of Lombardy in Italy, the city of Boston in USA and the city of Amsterdam in the Netherlands which is stored in the data server. The priority of the application is to be used in any geographic location so a data provider that delivers this information dynamically is an unquestionable need for Urbanopoly as a game and for the growth of its community of players.

4.4.3. More social methods to authenticate

With the election of passport.js as the utility module for delegated authentication via social network opens endless possibilities to expand Urbanopoly authentication methods, as well as passport provide a strategy to allow authentication with Facebook there are many other strategies to allow the authentication with Twitter, Google, Flickr, GitHub to name a few. The reason why these methods of authentication have not yet been included in the web version is because Urbanopoly players are identified inside the data server using the id of the Facebook account, being this an external identifier there is no guarantee that other authentication methods ids does not collides with those provided by Facebook which means that in the future changes in how the players are identified in the data server should be performed to ensure that the not matter what authentication method was selected the id of the player is unique within the game.

4.4.4. Implement collectible rewards

After the game experience with Urbanopoly on par with the progress of the prototype of the web version and from the feedback made by test users a possible improvement to the reward system of the game was conceived, as is well known the sense of competition has a significant impact on the players returning to the game, based on this concept, the implementation of a series of collectible rewards (such as Foursquare medals) depending on user activity in the game arises so provide those rewards like achievements to identify the user progress in the game like a huge sum of money, the acquisition of representative sites, recurrence in the game and other activities.

5. CONCLUSIONS AND RECOMMENDATIONS

This document is a compendium of the detailed design and development of the web version prototype of Urbanopoly, the experience of the implementation process besides the problems and challenges encountered are described in detail. The design and characteristics of the original version were preserved and implemented within the limitations of the web architecture.

The source code of the prototype is the result of several months of learning and development and not meant to be a final product for a production environment, instead its intention is to provide a reference to confirm the feasibility of a web version maintaining the features of its mobile equivalent and even raise the possibility of extend this features in the near future.

The development process and the overall result achieved has been verified by the authors of Urbanopoly in its mobile version and determination for the realization of the proposed next steps is high so we expect the web version will be a step forward in the successful evolution of the game and we are confident that the project will continue to expand to more players and platforms in the future.

6. BIBLIOGRAPHY

[1] L. v. Ahn, "Games with a purpose," *Computer,* vol. 39, no. 6, pp. 92 - 94, 2006.

[2] L. v. Ahn, "Labeling images with a computer game," in *Proceedings of the SIGCHI Conference on Human factors in computing systems*, New York, 2014.

[3] gameswithpurpose.org, "Fraxinus," [Online]. Available: http://gameswithpurpose.org/fraxinus/. [Accessed September 2014].

[4] gameswithpurpose.org, "EteRNA," [Online]. Available: http://gameswithpurpose.org/eterna/. [Accessed September 2014].

[5] wikipedia.org, "Ingress," NianticLabs, December 2013. [Online]. Available: http://en.wikipedia.org/wiki/Ingress_(game). [Accessed September 2014].

[6] gameswithpurpose.org, "The Walk," Six to Start, February 2014. [Online]. Available: http://gameswithpurpose.org/the-walk/. [Accessed September 2014].

[7] S. Matyas, C. Matyas, C. Schlieder, P. Kiefer, H. Mitarai and M. Kamata, "Designing location-based mobile games with a purpose: collecting geospatial data with CityExplorer," in *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, New York, 2008.

[8] semanticgames.org, "Waisda," 2011 November. [Online]. Available: http://semanticgames.org/2011/11/waisda/. [Accessed September 2014].

[9] "http://semanticgames.org/," November 2011. [Online]. Available: http://semanticgames.org/2011/11/guesswhat/. [Accessed September 2014].

[10] "semanticgames.org," November 2011. [Online]. Available: http://semanticgames.org/2011/11/concept-game/. [Accessed September 2014].

[11] I. Celino, D. Cerizza, S. Contessa, M. Corubolo, D. Dell'Aglio, E. Della Valle, S. Fumeo and F. Piccinini, "Urbanopoly - A Social and Location-Based Game with a Purpose to Crowdsource Your Urban Data," in *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012*

International Confernece on Social Computing (SocialCom), Amsterdam, 2012.

[12] I. Celino, D. Cerizza, S. Contessa, M. Corubolo, D. Dell'Aglio, E. Della Valle, S. Fumeo and F. Piccinini, "Urbanopoly - Collection and Quality Assessment of Geo-spatial Linked Data via a Human Computation Game," in *Submissions Semantic Web Challenge 2012*, Boston, 2012.

[13] P. Teixeira, Professional Node.js: Building Javascript Based Scalable Software, Wrox, 2012.

[14] expressjs, "Express.js," [Online]. Available: http://expressjs.com/. [Accessed September 2014].

[15] J. Hanson, "Passport.js," [Online]. Available: http://passportjs.org/. [Accessed September 2014].

[16] Wikipedia, "Web Template System," August 2014. [Online]. Available: http://en.wikipedia.org/wiki/Web_template_system. [Accessed September 2014].

[17] V. Agafonkin, "Leaflet.js," 2010. [Online]. Available: http://leafletjs.com/. [Accessed September 2014].

[18] Wikipedia, "Mapbox," 2010. [Online]. Available: http://en.wikipedia.org/wiki/Mapbox. [Accessed September 2014].

[19] T. j. Foundation, "jQuery," 2006. [Online]. Available: http://jquery.com/. [Accessed September 2014].

[20] Wikipedia, "Bootstrap," 2011. [Online]. Available: http://en.wikipedia.org/wiki/Bootstrap_(front-end_framework). [Accessed September 2014].

[21] K. Visweswaran, "Star Rating," 2014. [Online]. Available: http://plugins.krajee.com/star-rating. [Accessed September 2014].