May 27

Software Requirement Specification

Team Naegling

Team Members:
Johan Nilsson
David M. Szabo
Elsa Wide
Henrik Edholm
Sima Strybis
Mikaela Lidström

Supervisor: William Granli

Table of Contents

1. Introduction

In this section of the software documentation you will obtain information about the purpose of the developed system, including the scope and description of the software, constraints, product perspective, user stories, use cases, functional and nonfunctional requirements, logical databases and product functions i.e. what functions the system contains and who can use the system.

1.1 Purpose

The purpose of this document is to present a detailed description of the game, City of Assassins. The document will explain the features, the purpose, the interfaces, what the system will do, the constraints under which it must operate and how it will react to external factors. This document is intended for both the stakeholders and the developers.

1.2 Scope

This document will give more detail about the product and the different kinds of requirements and use cases. It helps the reader to have a deeper understanding about the product.

1.3 Definitions, Acronyms, and Abbreviations

This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.

SRS	**Software Requirement Specification**
SDD	Software Design Document
PHP	It "is a server-side scripting language designed for web development".[1]
Python	It "is a widely used general-purpose, high-level programming language".[2]
Java	It "is a programming language expressly designed for use in the distributed environment of the Internet."[3]
XML	"Extensible Markup Language, is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable."[4]
Android	It "is an operating system based on the Linux kernel with a user interface based on direct manipulation, designed primarily for touchscreen mobile devices such as smartphones and tablet computers".[5]

[1] http://en.wikipedia.org/wiki/PHP, downloaded 2014.05.24.
[2] http://en.wikipedia.org/wiki/Python_(programming_language), downloaded 2014.05.24.
[3] http://searchsoa.techtarget.com/definition/Java, downloaded 2014.05.24.
[4] http://en.wikipedia.org/wiki/XML, downloaded 2014.05.24.
[5] http://en.wikipedia.org/wiki/Android_(operating_system), downloaded 2014.05.24.

1.4 Overview

The SRS has 3 sections. The first one is introduction. The second one is general description, which will be about the product perspective, product functions, user characteristics, general constraints and assumptions and dependencies.

The third one is specific requirements, which will be about the use cases, functional and nonfunctional requirements.

2. General Description

2.1 Product Perspective

City of Assassins is a standalone product, not relating to any other products. It is an android application geared towards providing entertainment for its users, in which players can virtually assassinate each other using a GPS.

2.2 Product Functions

The main function of the product is to provide players a target which they can assassinate in an augmented reality app. For more detailed description see the functional requirement section.

2.3 User Characteristics

We mainly have one type of user for this system, these users can use it on their phone.

The user is expected to have basic understanding of android devices and apps. They are also expected to be Internet literates.

2.4 General Constraints

General design/implementation designs include the following:

- Python, C, Java, XML, and PHP languages will be used for programming the system.
- Developed software will run on Android devices.
- The developed system is required to incorporate Raspberry Pi into its function by using it scan phones and collect items.
- The system will be developed using Scrum software process.

2.5 Assumptions and Dependencies

2.5.1 Assumption

- The assumption is made that Google maps are free to use for non-commercial purposes and that we will be able to implement it in the app.
- The assumption is made that the GPS will be accurate and that it will update locations regularly.
- The assumption is made that the server will retrieve, process and display information in a reasonable amount of time.

2.5.2 Dependencies

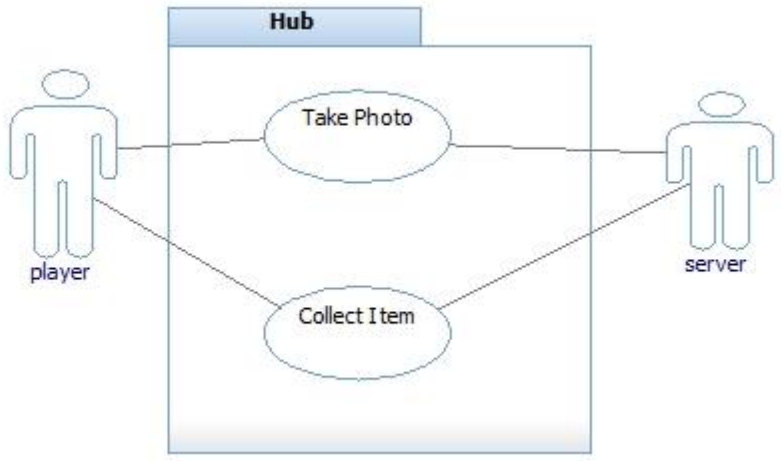
- The user should have an android device.
- The app is dependent on the internet connection.
- The game is dependent on users having the GPS on.
- The app is dependent on the NFC.

3. Specific Requirements

3.1 Functional Requirements: Use cases

This section describes specific features of the software project in a use case format.

3.1.1 Hub



Created by Simas Strybis, 20.05.2014., Hub Use Case Diagram

1. Take Photo

Brief Description:

The Player can take his/her profile picture for the application from the PI.

Flow of Events

Basic Flow

The use case starts when the Player scans his phone with the NFC scanner.
The Hub receives and records the signal.
The Hub takes the photo with the camera module.
The Hub uploads the picture to the server. The use case ends.

Preconditions

The Player is logged in.
The System is connected to the internet.
The Player has NFC on their phone.

Postconditions

The Players profile picture has changed.

2. Collect Item

Brief Description:

When the Player assassinates the target he / she can collect item from the PI.

Flow of Events

Basic Flow

Player assassinates target.
The System shows loot and option to collect item.
Player selects option to collect item.
The System registers the item.
The System instructs the Player to scan phone on hub.
Player scans phone on hub.
The System shows the option to select, which item to take, old or new.
Player selects the item and it is stored in profile.

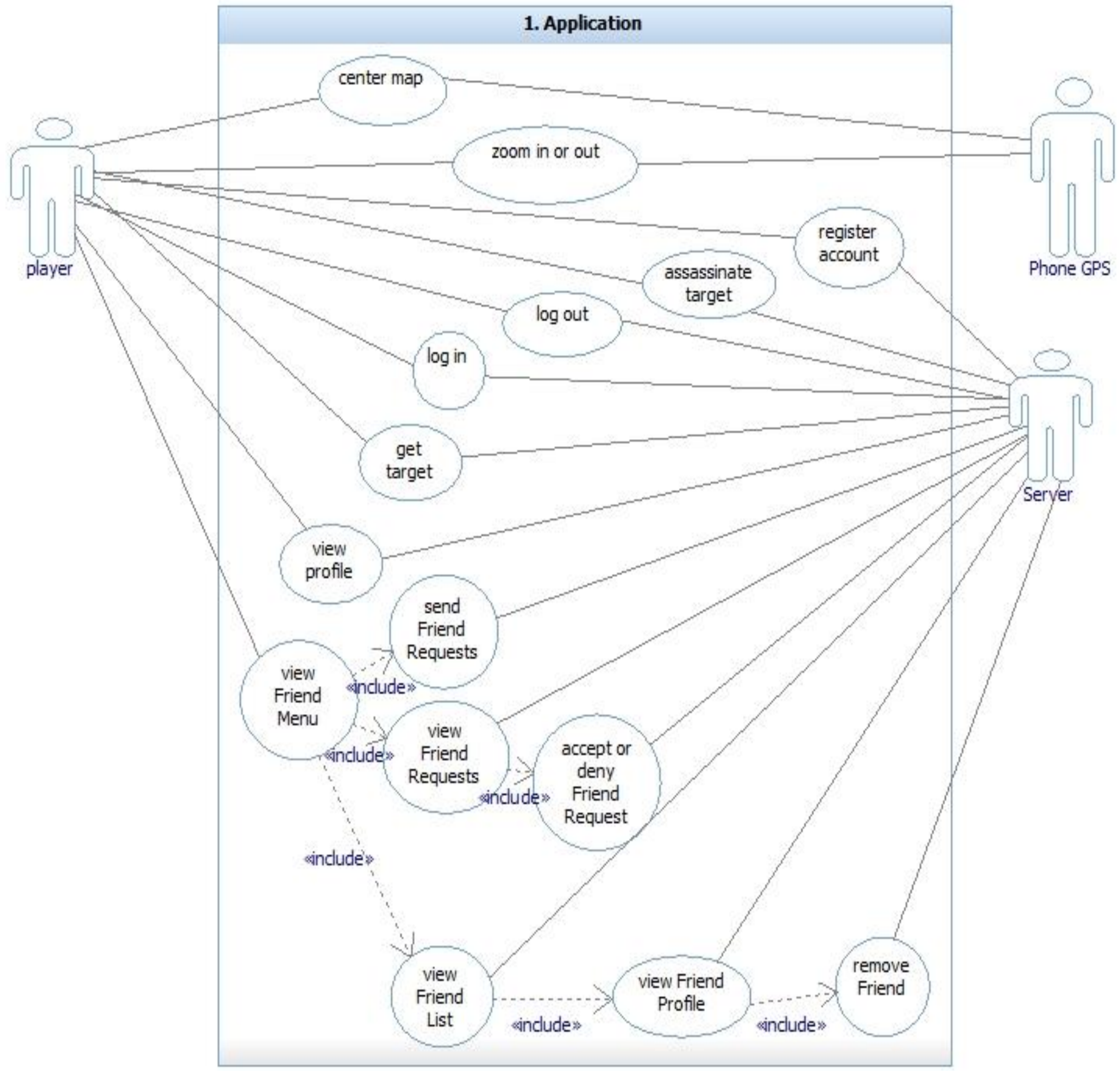
Preconditions

The Player is logged in.
The System is connected to the internet.
The Player has NFC on their phone.

Postconditions

The Player has a new item in his inventory

3.1.2 Application



Created by Simas Strybis, 20.05.2014., Hub Use Case Diagram

1. Register Account

Brief Description

The Player creates an account in the game.

Flow of Events

Basic Flow

The use case starts with the Player selecting Sign up option.
The System asks the Player to fill in name, email, password and password again.
The Player fills in the information.
The System registers the Player in the server. The use case ends.

Alternative Flows

At basic flow step 3.

The Player enters an existing name or email.

The System asks the Player to enter a new name or email. The use case returns to the basic flow step 3.

At basic flow any step

The Player cancels registration.
The information is not added to the database. The use case ends.

At the basic flow step 3

The Player enters a password that is too short. The system informs the user enter a password of at least 8 characters. The use case returns to the basic flow step 3.

Preconditions

The Player is logged out.
The System is on the Log in activity.
The System is connected to the internet.

2. *Zoom in / out*

Brief Description:

The Player can zoom in and out on the map.

Flow of Events

Basic Flow

The use case starts with the Player selecting the zoom in option.
The System zooms in. The use case ends.

Alternative Flows

At basic flow step 1

The Player select the zoom out option.
The System zooms out. The use case ends.

Preconditions

The Player is logged in.
The System is connected to the internet.
The Player has NFC on their phone.

3. *Log out*

Brief Description:

The Player logs out from the game.

Flow of Events

Basic Flow

The use case starts when the Player selects the log out option.
The System logs the Player out. The use case ends.

Preconditions

The Player is logged in.
The System is connected to the internet.
The Player has NFC on their phone.

4. *Get Target*

Brief Description:

The Player gets a new target.

Flow of Events

Basic Flow

The use case starts when the Player selects the get target option.
The System selects a new online target for the Player. The use case ends.

Preconditions

The Player is logged in.
The target is logged in.
The System is connected to the internet.
The Player has NFC on their phone.

5. *View Profile*

Brief Description:

The Player views his / her profile.

Flow of Events

Basic Flow

The use case starts when the Player selects the view profile option.
The System shows the Player’s profile. The use case ends.

Preconditions

The Player is logged in.
The System is connected to the internet.
The Player has NFC on their phone.

Postconditions

The Player is logged out from the game.

6. *Send Friend Request*

Brief Description:

The Player sends a friend request.

Flow of Events

Basic Flow

The use case starts when the Player chooses to add friends.
The System asks the Player for a name.
The Player enters the friend's name who he / she wants to add.
The System shows the written name.
The Player chooses the send option.
The System adds the friend to the friend list. The use case ends.

Alternative Flows

At basic flow step 5

The System doesn't find the required friend.
The System gives an error message. The basic flow resumes at step 2.

At basic flow step 5

The selected Player was already the Player's friend.
The System shows this information to the Player. The basic flow resumes at step 2.

Preconditions

The Player is logged in.
The System is connected to the internet.
The Player has NFC on their phone.
The Player is in the View Friend list menu.

Scenario: Sending Friend request

David decides to add a friend to his friend list. He enters a name and chooses to send the friend request. The person receives his friend request.

7. *View Friend Request*

Brief Description:

The Player views his / her friend requests.

Flow of Events

Basic Flow

The use case starts when the Player selects the view friend requests option.
The System shows the Player’s friend requests. The use case ends.

Preconditions

The Player is logged in.
The System is connected to the internet.
The Player has NFC on their phone.
The Player is in the View Friend list menu.

Extension Point

Accept or deny Friend Request use case.

8. *Log in*

Brief Description:

Log in function allows Players to log into their Player account.

Flow of events:

Basic flow:

Use case starts when the Player turns on the System.
The System displays the start screen.
Player enters his details and opts to log in.
The System fetches the details from the server and logs in the Player. Use case ends.

Alternative Flow:

At step 2, Player enters incorrect details. The System notifies the Player the details are incorrect.
Use case return to step 1.

Preconditions:

Player has an already existing account.
Player has a functioning Android device.
Player’s device has access to internet.

Postconditions:

Player is shown the map on his screen.

Scenario 1: Logging in

John starts the System on his Android device and is taken to the log in screen. He enters his details and is logged into the game.

Scenario 2: Failing logging in

If John enters incorrect information, he will be notified that he could not be logged in.

9. *Center Map*

Brief Description:

Center map use case zooms to the Player's location on the map.

Flow of events:

Basic flow:

Use case starts when Player chooses the Center Map option.
The System zooms to the Player's location. Use case ends.

Preconditions:

Player has access to the internet.
Player's Android device has built in GPS device which is able to get a signal.
Player is viewing the map on his Android device.

Scenario: Centering the map

While viewing the map of the game on his Android device, John presses the Center Map button. He is then zoomed into his location on the map.

10. *Assassinate Target*

Brief Description:

The use case allows the Player to assassinate his / her target.

Flow of events:

Basic flow:

Use case starts when the Player opts to get a target. The System displays the target's location.
Player goes to the target's location. The System makes the "Assassinate Target" option enabled.
Player chooses to assassinate his target. The System does a random roll in favor of assassinating the target.
The Player and target are notified of results. The System assigns Player a new target. The use case ends.

Alternative Flow:

At step 3, the System does a random roll NOT in favor of assassinating the target. Use case resumes at step 4.

Special Requirements:

By default, the random roll gives the Player a 60/100 chance to kill his target.
Items carried by Player and/or his target can affect the random roll for better or worse.
Player has to be within 50 meters of his target for the assassinate button to be enabled.

Preconditions:

Player is playing the game.
Player has internet connection on his android device and a valid target.

Postconditions:

Player obtains a new target.
Player's target obtains a message notifying them if the assassination succeeded or failed.
Player's and his target's kills and deaths are adjusted accordingly in the database.

Scenario 1: Assassinating a target

John is playing the game and he has acquired a target. He approaches his target until he is within 50 meters or closer. The "Assassinate Target" button on his screen becomes enabled. John presses the button and the System does a random roll. The roll comes out in favorable to John and he succeeds in assassinating his target. He and his target are notified of the result, and John's kill count is increased by 1, while his target's death count is increased by 1. John is then given a new target.

Scenario 2: Failing to assassinate a target

In case the random roll comes out favorable for John's target, he fails to assassinate his target. The System makes no changes to kill and death counts and just assigns John a new target.

11. View Friend Menu

Brief Description:

View Friend menu use case allows Players to handle System options concerning their friends.

Flow of events:

Basic flow:

Use case starts when Player chooses "Friends" option if the System menu. System displays three tabs on the screen. Use case ends.

Preconditions:

Player has access to internet on his Android device.

Postconditions:

When this use case ends, it activates View Friend List use case.

Extension points:

View Friend List
Send Friend Requests
View Friend Requests

Scenario: Viewing friend menu

While playing the game, John decided to view Friend menu. He prompts up the menu and chooses "Friends" option. The System displays different tabs, with Friend List tab being selected.

12. View Friend List

Brief Description:

View Friend List use case displays the Player all of his friends in the game.

Flow of events:

Basic flow:

Use case starts when the Player chooses to view his friends. The System displays the Player a list of all of his friends. Use case ends.

Special Requirements:

This use case is activated automatically every time View Friend Menu use case ends (although Player can activate it manually under other circumstances).

Preconditions:

Player has to have internet access on his Android device.

Postconditions:

Use case displays a list of friends, from which the Player can choose to activate View Friend Profile use case.
Players can enable Send Friend Request or View Friend Requests use cases from this use case.

Extension points:

View Friend Profile.

Scenario: viewing friend list

John wants to view his friends, so he chooses an appropriate tab while in Friend menu. A list of his friend is then displayed on the screen.

13. View Friend Profile

Brief Description:

View Friend Profile use case allows Players to view their friend's information.

Flow of events:

Basic flow:

Use case starts when the Player chooses to view information of one of his friends. The System display's their friend's profile. The use case ends.

Preconditions:

Player has to have friends for this use case to be available.
Player has to have internet access on his android device.
This use case is only available while View Friend List use case is active.

Postconditions:

Display screen show Player's friend's profile, from which use case Remove Friend can be activated.

Extension points:

Remove Friend

Scenario: viewing friend profile

While viewing his friends, John chooses to view a particular friend's information. He chooses a friend and the screen displays his friend's profile.

14. Remove Friend

Brief Description:

This use case allows Player to remove his friend from his friend list.

Flow of events:

Basic flow:

Use case starts when Player opts to remove one of his friends from his friend list. The System asks Player for confirmation.
Player confirms he wants to remove his friend. The System removes the friend and brings the Player back to his friend list. Use case ends.

Alternative Flow:

At step 2, Player does not give confirmation. System brings the Player back to the friend's profile page. Use case ends.

Preconditions:

Player has to have access to internet on his Android device.

Postconditions:

If friend is deleted, Player is displayed his friend list.
If deleting friend query is aborted, Player id displayed the friend profile.

Scenario 1: Removing a friend

While looking at his friend's Steve profile, John decided to remove him from his friends list. The System asked John to confirm that he really want to remove Steve from his friend's list, and after confirming, John was brought back to his friends list, with Steve no longer there.

Scenario 2: Not removing a friend

In case John canceled friend deletion when the System asked him to confirm it, John will be brought back to Steve's profile.

15. Accept or Deny Friend Request

Brief Description:

Accept or Deny Friend Request use case allows Players to accept or deny friend requests.

Flow of events:

Basic flow:

Use case starts when Players selects a friend request. The System asks the Player if he wants to accept friend request.
Player chooses to accept friend request. System removes the friend request and ads accepted Player to Player's friend list. Use case ends.

Alternative Flow:

At step 2, Player chooses to deny the friend request. The System removes the friend request. Use case ends.
At step 2, Player chooses to ignore the friend request. The System does nothing. Use case ends.

Preconditions:

Player has internet access on his Android device.
Player has friend requests.

Scenario 1: Accepting friend requests

While viewing his friend requests, John chose William's friend request. The System asked John if he wants to accept William's friend request. John chose yes and the System removed the friend request and added William to John's friend list.

Scenario 2: Denying friend requests

While viewing his friend requests, John chose William's friend request. The System asked John if he wants to accept William's friend request. John chose no and the System removed the friend request.

Scenario 3: Ignoring friend requests

While viewing his friend requests, John chose William's friend request. The System asked John if he wants to accept William's friend request. John chose to ignore for now, and the System returned him to the friend request list.

16. Forgot Password

Brief Description:

The Player forgot his / her password use case allows user to get a new password.

Flow of Events

Basic Flow

1. The use case starts when the Player forgets his / her password. He or she selects the Forgot Password option.
2. The system displays input option for the Player with to input an email address.
3. The Player inputs his / her email. And confirms.
4. The system sends a random generated new password to the Player's email address.
5. The use case ends.

Alternative flows:

At basic flow 3:

1. The Player enters an invalid email.
2. The system displays an error message that the email is invalid.
3. Use case continues at basic flow 3.

Preconditions

The System is connected to the internet.
Player is in login view.

17. Change Password

Brief Description:

This use case allows the Player to changes his / her password.

Flow of Events

Basic Flow

1. The use case starts when the Player wants to change his / her password.
2. The system provides the option to change the password.
3. The Player selects that option.
4. The system asks the Player for the old password, and the new password twice.
5. The Player enters his / her old password and new passwords and selects the confirm option.
6. The system displays a confirmation.
7. The use case ends.

Alternative Flow

At basic flow 5:

1. The user inputs incorrect data.
2. System displays error message.
3. Use case continues at basic flow 5.

Preconditions

The Player is logged in.
The System is connected to the internet.

18. View Ranking

Brief Description:

The player views the ranking of all players.

Flow of Events

Basic Flow

The use case starts when the user chooses “Ranking in menu” option. The system displays Ranking List. The use case ends.

Preconditions

The Player is logged in.
The System is connected to the internet.
The Player is in the ProfileActivity.

19. View Friends Ranking

Brief Description:

The player views the ranking of all players.

Flow of Events

Basic Flow

The use case starts when the user chooses Show Friend Ranking option. The system displays Friends Ranking list. The use case ends.

Preconditions

The Player is logged in.
The System is connected to the internet.
The Player is in the RankingActivity.

3.2 Non-Functional Requirements

3.2.1 Performance

3.3.1.1 Getting a GPS location should not take over 1 minute.
3.3.1.2 The GPS location should update every 5 seconds.
3.3.1.3 Login in to the application should not take more than 10 seconds

3.2.2 Reliability

3.3.2.1 The application should have less than 438 hours downtime per year.

3.2.3 Availability

3.3.3.1 The application should be available 99.99% of the time.
3.3.3.3 The application should be available in all of the Gothenburg area.
3.3.3.4 The application should be available for all Androids users.

3.2.4 Security

3.3.4.1 All passwords are encrypted and only known by the user.
3.3.4.2 The users should be able to disable the GPS tracking.

END OF SRS