|  |
| --- |
| INTHER SOFTWARE DEVELOPMENT |
| MotiVision |
| Software Requirements Specification |
|  |
| Version 1.0 |
|  |
| **Alex Zavtur, Dumitru Dîrul** |
| **2-Jul-2018** |

|  |
| --- |
| Software Requirements Specification for “MotiVision” project to be implemented during the internship session in June-July 2018. |

Contents

[1. Introduction 2](#_Toc518056526)

[1.1 Purpose 2](#_Toc518056527)

[1.2 General Description 2](#_Toc518056528)

[2. Specific Requirements 2](#_Toc518056529)

[2.1 Functional Requirements 2](#_Toc518056530)

[2.2 Architecture 4](#_Toc518056531)

[2.2.1 Server 4](#_Toc518056532)

[2.2.2 Clients 4](#_Toc518056533)

[2.3 Non Functional Requirements 4](#_Toc518056534)

[2.3.1 Project Methodology 5](#_Toc518056535)

[2.3.2 Source Code 5](#_Toc518056536)

[2.3.3 Software Tools 5](#_Toc518056537)

[2.3.4 Assets 5](#_Toc518056538)

### Introduction

##### 1.1 Purpose

This document describes general Software Requirements Specification for the “MotiVision” project that is to be implemented during the Internship Session in July 2018. The document will serve as a requirements base for the project team. Furthermore, requirements and details will be identified, discussed and agreed on in an Agile development method.

##### 1.2 General Description

The project that will be implemented during Internship Program aims to create an interactive mobile/desktop application that would motivate people practicing sport activities during small breaks. Usually, at ISD, employees perform push-up/pull-up actions during workday as a break, to keep themselves more fit. The purpose of this application is to motivate employees to practice this sort of activities while keeping it interactive and entertaining.

From technical point of view, the application will represent a distributed system, based on Client-Server pattern. The Server side will share resources and services that large number of distributed Clients wish to access.

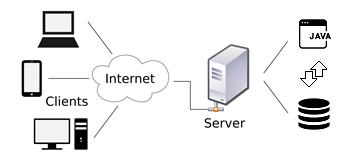


Fig. 1 Client-Server pattern

Clients initiate interactions with the server, invoking services as needed from the server and waiting for the result of those requested.

The communication between these two components will be ensured by a REST API. The data the application will work with has to be stored into a database.

### 2. Specific Requirements

##### 2.1 Functional Requirements

Below is a list of User Stories and the acceptance criteria every of them must cover:

1. (Logo page) As a user when launching the application I want to see ISD logo image fading in and out.
   1. No interaction between user and application during this period of time will be allowed.
2. (Login/Register page) As a user, I want to be able to login or register into the system, so that I could use it.
   1. I can login by providing my username and password.
   2. I can register myself into the application providing new username and password.
   3. When registering a new user, validate that the username is not already taken.
3. (Character selection page) As a user, I want to be able to see the list of all character profiles created under my account or create a new profile for it.
   1. I can select one of the character profiles from available list (in case there is at least one profile available).
   2. I can create a new character (Click the “Create” button from the bottom of available profiles list).
   3. I can delete one of the profiles from the list. A confirm pop-up dialogue will show up, asking for action confirmation.
   4. I can logout from the system.
4. (Character creation page) As a user, I want to be able to create new character providing following information:
   1. Specify the character’s name
   2. Select the gender (male or female)
   3. Select head type from a predefined set of options
   4. Select body type from a predefined set of options
   5. Specify the team to join (validation of existing team should be done).
   6. Checkbox “Create new team”
   7. Select Team logo (in case “Create new team” is selected).
5. (Character profile page) As a user, I want to be able to see following information:
   1. View my character with all equipped items.
   2. I want to see the name of the character and the team he belongs to.
   3. Buy new items for my character from the store (In case that the item has not been purchased yet, the cost of the item must be displayed).
   4. View points balance of my currently selected character.
   5. Go back to character selection page.
   6. Access team members list.
   7. Access last battle screen in case a battle occurred already.
   8. Access Earning points screen.
   9. Access admin page (only available to the admin of the team).
6. (Team members page) As a user, I want to be able to see the list of all members of the team I belong to:
   1. Team members list should be ordered by name.
   2. Delete a member from my team (only available to admin).
   3. Go back to profile page.
7. (Earning points page) As a user, I want to see the list of available activities I can choose to complete:
   1. View all available activities defined per team with their awarding points.
   2. Once I confirm the completion of an activity an animation will appear of my character battling a monster.
   3. Go back to my profile
8. (Admin page) As an admin of a team, I want to be able to:
   1. Change the team logo.
   2. Manage activities list defined per team (Add/Edit/Delete (name/value)).
   3. Set battle frequency.
   4. Go back to profile page.
9. (Battle page) As user, I want to be able to see the last battle scene that occurred for the team.
   1. When battle starts, an animation containing (from left to right): the image of team castle, team members, enemy army members and the enemy castle image.
   2. Scene of fighting will last for Y seconds (to be defined later).

1. (Settings pop up dialogue) As a user, I want to access settings page from every screen (except the Logo page) where I can enable/disable sound and music.

##### 2.2 Architecture

Given the fact that the project is going to be implemented as a distributed system, the application can be split into several parts.

##### 2.2.1 Server

The server handles CRUD operations, user authentication, necessary computations and provides requested resources to clients.

##### 2.2.2 Clients

For instance a mobile application will be developed that based on REST API will communicate with server. The REST interface API is to be defined by the project team.

#### 2.3 Non Functional Requirements

1. The password used for registration has to be encrypted.
2. When creating a new character and selecting the options to create a new team – the created character becomes the admin of this team.
3. During last week of the internship, mobile application will be placed on Play Store while server side application has to be deployed on one of the company’s VMs. Details to be provided...

##### 2.3.1 Project Methodology

The project itself will be implemented using Agile SCRUM methodology. There will be 4 Sprints, one week long each. The development team size is limited to 5 people. Specific detailed project requirements will be refined for each sprint apart, depending on the actual project progress.

##### 2.3.2 Source Code

Delivered source code should be of high quality, easy maintainable, tested and documented. All the source code will be open source.

Project should be versioned using Git and hosted on GitHub. High quality documentation is a must, both as properly commented code and as a proper documentation document delivered with the project.

Documentation should contain technical knowledge, for future developers, but not forget end user documentation as well.

##### 2.3.3 Software Tools

The project team is to decide on the tools to be used during the development process. The only requirement is to use freeware user friendly tools. All the used tools should be mentioned in the documentation. All the tools should fully operational on Microsoft Windows.

Below is a list of recommendations the team might consider to use:

* Recommended IDE (Server side application): NetBeans.
* Recommended IDE (Client side application): Eclipse, Android Studio
* Recommended Framework LibGDX
* Preferable database: PostgreSQL.
* Server/Client side programming language: Java 8.

##### 2.3.4 Assets

The project team is to decide on the assets (images, sounds, music) to be used during the development process. The only requirement is that all assets should be freeware. All the sources for the assets should be mentioned in the documentation.