|  |
| --- |
|  |
| Crimson Clubs |
| Software Requirements Specifications |
|  |
|  |
| **Max Livingston**  **Maclane May** |
| **Adam Gabriel**  **Travis Redfield** |
| **2/27/2018** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision History** | | | |
| **Date** | **Version** | **Description** | **Author** |
| 2-27-2018 | 0.1 | Initial Draft | Crimson Clubs |
|  |  |  |  |

Table of Contents

[1. Preface 3](#_Toc506835075)

[2. Introduction 3](#_Toc506835076)

[3. Glossary 3](#_Toc506835077)

[4. User Requirements Definition 3](#_Toc506835078)

[4.1 Functional Requirements 3](#_Toc506835079)

[4.2 Non-Functional Requirements 4](#_Toc506835080)

[5. Competition 4](#_Toc506835081)

[6. System Architecture 4](#_Toc506835082)

[6.1 Main Screen 4](#_Toc506835083)

[6.2 Options 4](#_Toc506835084)

[7. System Requirements Specification 5](#_Toc506835085)

[8. System Models 6](#_Toc506835086)

[9. System Evolution 6](#_Toc506835087)

[10. Index 7](#_Toc506835088)

# Preface

This is version 0.1 of the requirements documentation for the Android-based Crimson Clubs application. This document is intended for developers and user to guide them through the process of development during the course of the project.

# Introduction

Crimson Clubs is an Android App/framework that will allow clubs and intramurals at universities keep track of their statistics and schedules. In addition to this functionality, we intend to include tools that will allow admins and referees to update and track past / previous games.

# Glossary

Android: a software stack for mobile devices that includes an operating system, middleware and key applications.

Application framework: framework that enables reuse and replacement of components

Framework: an abstraction in which software providing generic functionality can be selectively changed by user code, providing application specific software.

API: application programming interface, a particular set of rules and specifications that software programs can follow to facilitate interaction.

# User Requirements Definition

## Functional Requirements

1. An admin user shall be able to create events.
2. An admin user shall be able to add other people or players / approve people.
3. An admin user shall be able to add custom push notifications.
4. An admin user shall be able to remove people.
5. An admin user shall be able to modify descriptions, info’s, and tags.
6. A normal user shall be able to request to join clubs, then once approved able to join club.
7. A normal user shall be able to search clubs.
8. A normal user shall be able to look at personal calendar.
9. A normal user shall be able to leave clubs.
10. A normal user shall be able to create a profile.
11. A normal user shall be able to view event stats.
12. A referee user shall gain access to stopwatch asset.
13. A referee user shall be able to upload scores.
14. A referee user shall be able to update statistics.

## Non-Functional Requirements

1. The system should be able to operate navigation outside of a network.
2. The system must not be a significant drain on battery life.
3. The database with be T-SQL.
4. The backend will be C# .NET using MVC.
5. The frontend website will use AJAX calls instead of form submissions, to provide a faster, more seamless experience to the user.
6. Performance is critical, any visible delays or appearance of inefficiency will need to be corrected so that the user a quick, smooth interaction with the software.
7. Role base access control will be implemented.

# Competition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | User-Friendliness | Schedule | Nonintrusive Advertising | Efficiency | Statistics |
| IMLeagues | None | Full | None | None | Partial |
| Crimson Clubs | Full | Full | Full | Full | Full |

# System Architecture

The system will be composed of five main activities: the main screen in which to begin the application, a set of options to choose a store and list, a scanner and/or upload to upload a store or item list, a list of items which can be edited, and finally a navigation activity in which the user can navigate the store.

## Main Screen

The main screen will be the first thing the user sees when opening the application. From here, they can choose to start shopping or view some basic application information. It will extend the Activity class.

## Options

The options screen will be the activity in which the user decides which store and list they would like to use. It will be composed of several Spinner widgets, so that the user can pick the appropriate store and list. This screen will also contain a button to open the Scan Store and Upload List activity, a button to open the Edit List activity, and a button to Navigate. No actual computation/queuing will occur in this activity; the user will just choose a predefined/favorite option.

# System Requirements Specification

The base structure of the project will revolve around specific activities, and use other classes to implement specific features that these activities will need. These system helper classes include

# System Models

# System Evolution

In addition to the main requirements, there are plans to partner with multiple universities on the same platform. As well as expand with more options, flexibility, features, etc…

# Index

Accelerometer 1, 3, 7

Aisle 3

AisleNav 3

Android 3, 5

Architecture 4

Camera 1, 5, 7

Communications 7

Computation 5

Data Class 6

Data connectivity 5

Database Class 6

Edit List 5

Evolution 12

Framework 3

Functional Requirements 4

Grid 1, 6, 7

Hardware 7

Hardware Requirements 5

List 6

List Class 7

Main Screen 4

Map 1, 3, 6, 7

Models 8

Navigate 5

Non-Functional Requirements 4

Options 4

QR Code 3

Queries 5

Scan Store 4

Screenshots 11

Software Requirements 5

TCP 7

Upload List 4

USB 7

USB Driver 5