Software Requirements Specification

For

Team 1

September 23, 2010

Version 1

Prepared by:

Casey Droneburg

Table of Contents

1 Introduction 4

1.1 Overview 4

1.2 Goals and Objectives 4

1.3 Scope 4

1.4 Definitions 4

2 General Design Constraints 5

2.1 Roo Balance Application Environment 5

2.2 User Characteristics 5

2.3 Mandated Constraints 5

3 Nonfunctional Requirements 5

3.1 Operational Requirements 5

3.2 Performance Requirements 6

3.3 Security Requirements 6

3.4 Documentation and Training 6

3.5 External Interface 6

3.5.1 User Interface 6

3.5.2 Software Interface 6

4 Functional Requirements 7

4.1 Required Features 7

4.1.1 Use Case: 1 7

4.1.2 Use Case: 2 7

4.2 Optional Features 8

4.2.1 Use Case: 3 8

4.2.2 Use Case: 4 8

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Name** | **Description** |
| 1 | 09/23/10 | Casey Droneburg | Initial Document |
|  |  |  |  |

# Introduction

## Overview

The Roo Balance application will be a mobile app available to smartphone users with an iOS platform, essentially iPhone, iPad and iPod touch users. The application will provide access to a Roo Bucks account, a stored value account provided to UMKC One Card ID holders. The app will allow Roo Bucks account holders to check their balance and see where they can use their Roo Bucks.

This document provides information on the requirements for the Roo Balance software application. Project goals, scope and definitions are given in the introduction. Design constraints and application environment are described in the following section. Non-functional requirements are outlined for later verification. Functional requirements are given to show the system features and expected user interaction.

Project constraints will be included in separate documentation. The Software Project Management Plan will give specifics on project budget and schedule. A separate Test Plan document will address test specifications and procedures.

## Goals and Objectives

The main objective of this project is to allow students a way to access their Roo Bucks account information from a smartphone. The Roo Balance application is expected to:

1. Provide a mobile interface with UMKC.ManageMyID.com to access account information.
2. Function in a simple and intuitive manner.
3. Provide students with locations to use their Roo Bucks.

## Scope

The Roo Balance application will provide users with the ability to access information about Roo Bucks from a mobile device using the iOS platform. Users will be able to check their Roo Bucks account balance by logging in to their ManageMyID.com account through the app. Users will also be able to see locations that accept Roo Bucks without logging in to their account.

## Definitions

**Roo Balance Application** – the product that is being described here; the software system specified in this document.

**Project** – activities that will lead to the production of the Roo Balance application.

**Client** – the person or organization for which this Roo Balance application is being built.

**User** – the person or persons who will actually interact with the Roo Balance application.

**Use case** – describes a goal-oriented interaction between the system and an actor. A use case may define several variants called scenarios that result in different paths through the use case and usually different outcomes.

**Scenario** – one path through a user case

**Actor** – user or other software system that receives value from a user case.

**Developer** – the person or organization developing the system, also sometimes called the supplier.

**Stakeholder** – anyone with an interest in the project and its outcomes. This includes clients, customers, users, developers, testers, managers and executives.

# General Design Constraints

## Roo Balance Application Environment

The Roo Balance product will include a mobile app designed to work on an iOS platform. This application will interface with a proxy server of our design. This proxy server will interface with the umkc.managemyid.com website. The proxy server will allow more efficient maintenance of the software system.

Manage My ID Website

Roo Balance

User Interface

iOS Application

Roo Balance

Proxy Server

## User Characteristics

**Roo Bucks Users**: UMKC students, faculty or staff who own a smartphone. Students are working on their college education and as smartphone owners are likely proficient with mobile applications.

## Mandated Constraints

The application will run on an iOS platform. This platform was chosen based on experience with Objective-C and team consensus.

# Nonfunctional Requirements

## Operational Requirements

Usability: 95% of users will not need to read the user manual to be able to use the application.

## Performance Requirements

Maintainability: Changes made to the Manage My ID website can be adopted without altering the iOS application.

## Security Requirements

The Roo Balance app has two features. For the first feature, Use Case 1, Roo Bucks account security is provided by secure login to the managemyid website. Login information input via the Roo Balance app will not be stored. For the second feature, Use Case 2, no security is required and access to Roo Bucks locations is available to all users.

## Documentation and Training

The Roo Balance application will be delivered to users as a download without documentation or training. A user guide and system documentation will be provided to project stakeholders.

## External Interface

### User Interface

The user interface will be eye-catching and visually appealing. When users access their Roo Bucks accounts, the interface will provide a smooth transition with the Manage My ID website which has a straightforward, understated look and feel.

The interface will be intuitive. As a mobile app it will be streamlined and simple to use. No training will be provided and it is expected that 95% of users will be able to use the app without any training.

### Software Interface

The Roo Balance proxy server will serve as an interface between the iOS application and Manage My ID website. It will enable interaction between the user and the remote website.

# Functional Requirements

## Required Features

### Use Case: 1

**Description: User Login / Check Roo Bucks Balance**

Actors: student or any Roo Bucks user

Value = high

Cost = high

Basic Path

1. User clicks icon for Roo Bucks application.
2. System prompts user to select an option: View Account or View Locations.
3. User clicks View Account option.
4. System prompts user to enter user e-mail and password.
5. User enters correct user e-mail and password and clicks Login.
6. System displays Account Summary with Roo Bucks balance with options to logoff or view transactions.
7. User clicks Logoff.
8. System exits.

Alternate Path

1. User clicks icon for Roo Bucks application.
2. System prompts user to select an option: View Account or View Locations.
3. User clicks View Account option.
4. System prompts user to enter user e-mail and password.
5. User enters incorrect user e-mail and/or password and clicks Login.
6. System displays error message: “Invalid Email Address and / or Password for user@wrongaddress.com ..... Or you may have exceeded the number of consecutive attempts allowed. Please try again later.”
7. User may choose to login again, returning to step 1, or exit.
8. System exits.

### Use Case: 2

**Description: Find Where to Spend Roo Bucks**

Actors: student or any Roo Bucks user

Value = high

Cost = medium

Basic Path

1. User clicks icon for Roo Bucks application.
2. System prompts user to select an option: View Account or View Locations.
3. User clicks View Locations option.
4. System displays Roo Bucks locations, possibly on more than one screen, with an exit option.
5. User clicks exit.
6. System exits.

## Optional Features

### Use Case: 3

**Description: Check Roo Bucks Recent Transactions**

Actors: student or any Roo Bucks user

Value = medium

Cost = high

Basic Path

1. Following Login [Use Case 1 Step 6]: System displays Account Summary with Roo Bucks balance with options to logoff or view transactions.
2. User clicks View Transactions.
3. System displays recent transactions with option to logoff.
4. User clicks Logoff.
5. System exits.