Android App on

Class Management System

Software Requirements Specification

Version 1.0

<Date>

Abhinaba Audhya (M120360CA) Leader

Gagandeep Kaur (M120388CA)

Jay Shankar Yadav(M120387CA)

Sourav Das (M120358CA)

Vikash Kumar (M120354CA)

In partial fulfillment of

Object Oriented Programming Project

Instructor: Mr. Kranti Kumar

2014

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| <date> | <Version 1> | <Your Name> | <First Revision> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  | <Your Name> | Lead Software Eng. |  |
|  | A. David McKinnon | Instructor, CptS 322 |  |
|  |  |  |  |

**Table of Contents**

Revision History ii

Document Approval ii

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 References 1

1.5 Overview 1

2. General Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

3. Specific Requirements 2

3.1 External Interface Requirements 3

3.1.1 User Interfaces 3

3.1.2 Hardware Interfaces 3

3.1.3 Software Interfaces 3

3.1.4 Communications Interfaces 3

3.2 Functional Requirements 3

3.2.1 <Functional Requirement or Feature #1> 3

3.2.2 <Functional Requirement or Feature #2> 3

3.3 Use Cases 3

3.3.1 Use Case #1 3

3.3.2 Use Case #2 3

3.4 Classes / Objects 3

3.4.1 <Class / Object #1> 3

3.4.2 <Class / Object #2> 3

3.5 Non-Functional Requirements 4

3.5.1 Performance 4

3.5.2 Reliability 4

3.5.3 Availability 4

3.5.4 Security 4

3.5.5 Maintainability 4

3.5.6 Portability 4

3.6 Inverse Requirements 4

3.7 Design Constraints 4

3.8 Logical Database Requirements 4

3.9 Other Requirements 4

4. Analysis Models 4

4.1 Sequence Diagrams 5

4.3 Data Flow Diagrams (DFD) 5

4.2 State-Transition Diagrams (STD) 5

5. Change Management Process 5

A. Appendices 5

A.1 Appendix 1 5

A.2 Appendix 2 5

# 1. INTRODUCTION

## 1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to provide a detailed description of the functionalities of the Class Management System application on the Android platform. This document will cover each of the system’s intended features. It will also cover hardware, software and other technical dependencies.

**1.2 Intended Audience**

This document is intended for all individuals participating in and/or supervising the Class Management System. Readers interested in a brief overview of the product should focus on the rest of Part 1 (Introduction), as well as Part 2 of the document (Overall Description), which provide a brief overview of each aspect of the project as a whole.

Readers who wish to explore the features of Class Management System in more detail should read on to Part 3 (System Features), which expands upon the information laid out in the main overview.

Readers interested in the non-technical aspects of the project should read Part 5, which covers performance, safety, security, and various other attributes that will be important to users.

## 1.3 Scope

The Class Management System is an Android application built to help the faculty of NITC to manage the classes in a better way. It is composed of two main components: a client-side application which will run on Android handsets of the faculty, and a server-side application which will support and interact with various client-side features. The system is designed to facilitate the process of viewing student details, taking attendance and many other features right from their mobile devices.

## 

## 1.4 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.*

## 1.5 References

*This subsection should:*

*(1) Provide a complete list of all documents referenced elsewhere in the SRS, or in a separate, specified document.*

*(2) Identify each document by title, report number - if applicable - date, and publishing organization.*

*(3) Specify the sources from which the references can be obtained.*

*This information may be provided by reference to an appendix or to another document.*

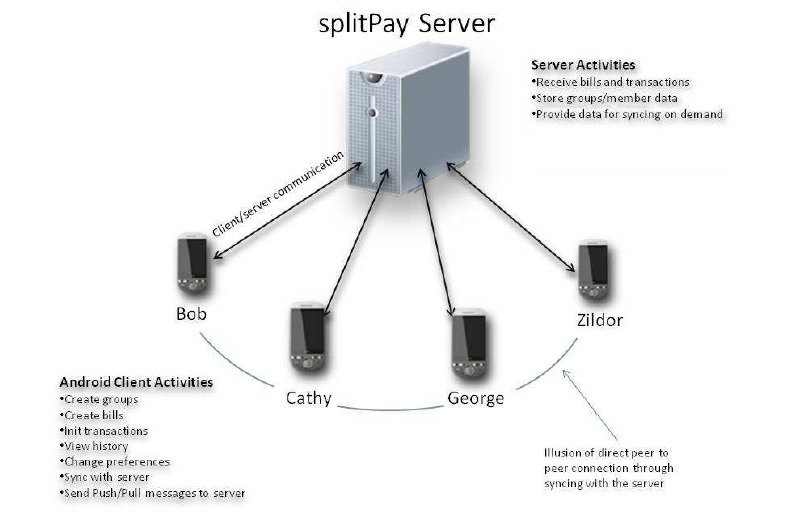
# 

# 2. GENERAL DESCRIPTION

## 2.1 Product Perspective

The Class Management System is a new, self-contained program intended for use on the Android platform. While the mobile application is the main focus of the project, the server side is equally important. The server side is responsible for the database and the synchronizing of the data and the services. The scope of this project encompasses both the server and the client sides, so both the aspects are covered in details in this document.

Below is a diagram of the Class Management System which illustrates the interaction between the server and the client sides.



## 

## 2.2 Product Functions

This subsection of the SRS should provide a summary of the functions that the software will perform.

## 2.3 User Characteristics

The Class Management System Android application is meant to provide the faculty of NIT Calicut a better and easier way to manage the classes from their mobile devices. It will reduce the paperwork and data entry job at the end of each semester. The interface is as simple as possible, and the application itself has no learning curve.

Users can be defined by how they will use the product in a particular situation. The following list categorizes the scenarios in which Class Management System is expected to be utilized by the faculty:

1. Quickly check a student’s details (like mobile number or email) in class or from anywhere.
2. Quickly check the number of days a student has been absent, and warn him if he/she is beyond the 80% rule.
3. Quickly check a student’s marks in that subject, and notify him.
4. Take a note of the students absent in class, and update it directly to the database.

These above scenarios can be supported by the core features of the application. There can be some additional scenarios depending on the implementation of the additional features:

1. A student may check the number of days he has been absent for a particular subject, at any time.
2. A student may check his marks in any subject, whenever he needs to check it.

**2.4 Operating Environment**

The main component of the Class Management System project is the software application, which will be limited to the Android operating system (specifically Android 2.3 and above). The application is not resource or graphics-intensive, so there are no practical hardware constraints. The app will rely on several functionalities built into Android’s Application Programming Interface (API), so ensuring appropriate usage of the API will be a major concern. Beyond that, the application is a self-contained unit and will not rely on any other Android-related software components.

The application will, however, frequently interact with the Class Management System server, a virtual dedicated server hosted by GoDaddy.com. The server operates on a Linux platform with 1GB of RAM and 100GB of allocated storage space. The Class Management System database will be stored on the server using MySQL and will be interfaced with a wrapper written in PHP 5.

## 2.5 Design and Implementation Constraints

The primary design constraint is the mobile platform. Since the application is designated for mobile handsets and tablets, limited screen size and resolution will be a major design consideration. Creating a user interface which is both effective and easily navigable will pose a difficult challenge. Other constraints such as limited memory and processing power are also worth considering. The Class Management System is meant to be quick and responsive, even when dealing with large groups of data and transactions, so each feature must be designed and implemented with efficiency in mind, reducing the overall response time of the application.

Another main constraint is the availability and speed of data connection on the mobile device. Since the application has to send requests to the server for each and every task, having a good data connection all the time is necessary.

## 2.6 Assumptions and Dependencies

# 

# 3. Specific Requirements

This will be the largest and most important section of the SRS. The customer requirements will be embodied within Section 2, but this section will give the D-requirements that are used to guide the project’s software design, implementation, and testing.

Each requirement in this section should be:

* Correct
* Traceable (both forward and backward to prior/future artifacts)
* Unambiguous
* Verifiable (i.e., testable)
* Prioritized (with respect to importance and/or stability)
* Complete
* Consistent
* Uniquely identifiable (usually via numbering like 3.4.5.6)

Attention should be paid to the carefuly organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

### 3.1.2 Hardware Interfaces

### 3.1.3 Software Interfaces

### 3.1.4 Communications Interfaces

## 3.2 Functional Requirements

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

### 3.2.1 <Functional Requirement or Feature #1>

3.2.1.1 Introduction

3.2.1.2 Inputs

3.2.1.3 Processing

3.2.1.4 Outputs

3.2.1.5 Error Handling

### 3.2.2 <Functional Requirement or Feature #2>

…

## 3.3 Use Cases

### 3.3.1 Use Case #1

### 3.3.2 Use Case #2

…

## 3.4 Classes / Objects

### 3.4.1 <Class / Object #1>

3.4.1.1 Attributes

3.4.1.2 Functions

<Reference to functional requirements and/or use cases>

### 3.4.2 <Class / Object #2>

…

## 3.5 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

### 3.5.1 Performance

### 3.5.2 Reliability

### 3.5.3 Availability

### 3.5.4 Security

### 3.5.5 Maintainability

### 3.5.6 Portability

## 3.6 Inverse Requirements

State any \*useful\* inverse requirements.

## 3.7 Design Constraints

Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

## 3.8 Logical Database Requirements

Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc.

## 3.9 Other Requirements

Catchall section for any additional requirements.

# 4. Analysis Models

List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.

## 4.1 Sequence Diagrams

## 4.3 Data Flow Diagrams (DFD)

## 4.2 State-Transition Diagrams (STD)

# 5. Change Management Process

Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.

# A. Appendices

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.

*Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.*

## A.1 Appendix 1

## A.2 Appendix 2