Software Requirements Specification

for

Club Management Application

Version 1.0 approved.

Prepared by:

Syed Irfan Ahmed (SRN: PES2201800041)

Naik Bhavan Chandrashekhar (SRN: PES2201800047)

Siddharth Shenoy (SRN: PES2201800499)

Individual Contribution: Researched about Google's OAuth and filled up sections 1 and 5 (Introduction and System Features) in the SRS doc.

PES University, EC Campus

29th January 2020

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This document expresses the need of an application for a club belonging to any University. As there are various clubs which are growing across different campuses, it may be difficult for the managing committee to generate forms for attendance, send notifications and perform other club related activities on a single platform. Hence, we have come up with an idea of an app for a club of our campus which is going to be implemented using some general and some specific features for that club.

1.2 Intended Audience

This document is a conscious effort made for all the users of the application, whether it would be the club heads, the managing committee of the club, the members of sub-teams of the club or just the mere members of the club. Also, we have explained more about our product features and its requirements in this document ahead.

1.3 Product Scope

The idea that we develop will intend to serve all the requirements of a functional club associated with any University. This is specifically made for the club heads/ members who find it difficult to maintain details about the club at one place considering they generally use various types of forms for the relevant work of the club. This application will have features like automatic attendance, instant pop-up notifications via app, direct log-in to related groups of the club, etc.

1.4 References

Our main reference is the PESU Application used by our University. Though it is a completely educational-based app, we aim at providing something similar to it because it has a lot of features which are the same to the app we will be developing.
A similar application is also used by the college football teams in the United States.

2. Overall Description

2.1 Product Perspective

The product is an end-to-end club management system designed for a club to manage themselves without the use of additional platforms and tools. It is a full system with multiple subsystems leveraging cloud infrastructures to dynamically update and manage members and their interactions.

2.2 Product Functions

- Member registration
  Admins can open and close registrations for their club, thereby having a single source to accept users directly into the club

- Member list management
  Members can be viewed by other members. Admins can manually add or delete members from the club as needed.

- Attendance marking and management
  Attendance marking for sessions conducted by clubs. Session based attendance lists for admins, and user-based attendance lists for all users

- Event scheduling
  Admins can schedule events in app, and members can view all past events and upcoming events

- Special event notice boards
  Any urgent notices or special events can be showed on a page to users. These events are entered in app by the admins

- Attendance registers
  Attendance on a per user, per session basis is available for all admins, and users can track their own attendance throughout their history

- Live push notifications
  Admins can notify all members via a push notification system, similar to PESU Academy's push system. These notifications are sent in app.

- Admin management pages
  Admins have a separate interface for managing the club and the members, and they are automatically shown a page by their credentials without a special login

2.3 User Classes and Characteristics

1. Admin/club managers:
   Privileged user class. Has read and write access to all application features as stated above. These users are responsible for managing the entire club, and they are manually entered by existing admins who have access to the core database infrastructure.

2. General club participants/members:
   Read only access to all features. They have the ability to mark their attendance during slots opened by admins which denotes their write access.

2.4 Operating Environment

Application will operate on Android, versions ranging from API level 21+, which will be the core interface and UI for both admins and users.
The application will be interacting with Firebase, which will be the cloud database and platform.

The application will have all classes corresponding to the use cases on devices and interact with the database to push and pull data from.

2.5 Design and Implementation Constraints

- The application is designed for a single club for this project's purposes. For security purposes, each database is assigned to a single club. In order to make this general purpose, this would entertain extensive functional design and additional server setup on GCP to allow such functionality.
- Application is restricted to Android. Deploying and testing applications for iOS requires access to the Apple ecosystem which none of the project members have.
- Application uses Firebase as a backend database. This provides us with an existing stack to build on, allowing faster iterations and prototyping.

- Application communicates via Firebase API calls, interfacing with the database and server.
- Security is implemented by using Google's Sign in functionality, allowing us to leverage Google's existing sign in protocols and allowing us to use their "Sign in via Google" features.
- Application built using Java for application logic and XML for UI and setup.
- Android uses Gradle as a build platform, which executes and runs all setup tasks to build an application.

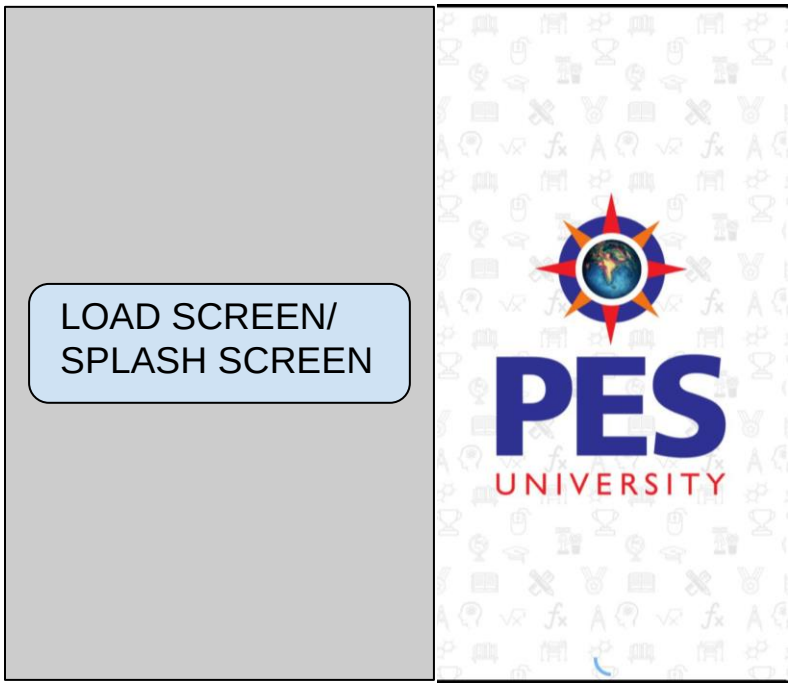
2.6 Assumptions and Dependencies

Project relies on Firebase's reliability as stated by Google for uptime and service accesses.
Application relies on vendor adherence to Google's guidelines on using the open source Andoid OS to build their flavours of Android. The application will be built and tested on Google's official Android release, on which all vendor ROMs are built on.

3. External Interface Requirements

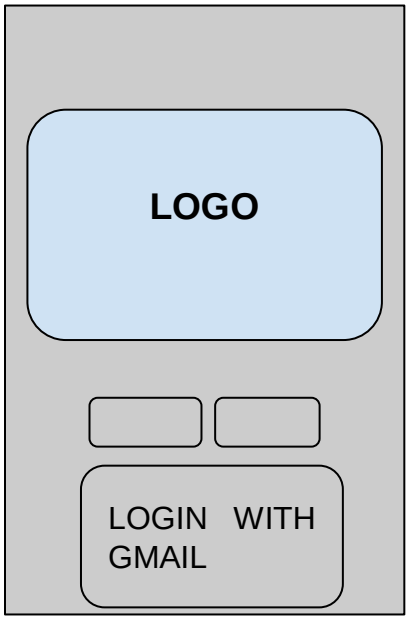
3.1 User Interfaces

Splash Screen
The first step in the user interface would be a splash screen.



When we open the app, we don’t want the user to be confused with if the app is hung or if the app is still running so instead of giving the user a blank white screen, we are planning on giving the user a splash screen. This screen will have the logo of the company/Application name.

Sign in/login screen



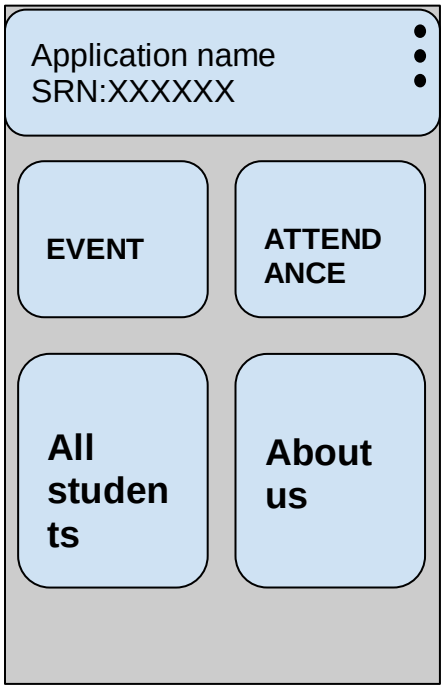
The next page that would show up would be a login/sign up page. In this page we are also planning on including login with Gmail this would be done by using Google API as it is much easier faster and less typing
As soon as the login is successful the app should provide you with a specific unique identification number through which a student would be identified.

Error messages related to this page

- If during the login process if you entered the wrong password or a wrong user id then a error message should appear
- If the internet is too slow and the login time expires then an error message should appear.

If you have logged into your app before and haven't logged out then next time when you open the app this page should appear it should directly go to the home page.

Home page



This is the home page, or you can also call the main page of our application. This page starts with the header in this part of the page there would be 3 main things that we plan on including first the application name, the SRN or unique identification number that the app provided after a successful login and the three dots would provide you a menu/drop down box with logout option and with an option to change the details of your account.

Next this page would mainly include 4 things as we are doing a club management application first and most important thing that we would need is the upcoming event that would take place in the club this event would be shown the event page when clicked on the event option a page should open which would show the upcoming event and who to contact to take part in the event.

Next in the importance list should be who and all in the total members of the club actually attend the meeting this can be done by an attendance system.
Admin and user will have a separate attendance option that admin can keep attendance on the days when the meeting is going to be held and the users can go to the attendance option and click on the attendance if they are present in the meeting. A “proxy” attendance can be solved by using a SSID clause in the attendance system.

All students should consist of all the members of the club so that the admin knows who and all are the part of the club as a result he has the required information to whom to give what work.

3.2 Software Interfaces

During login we are planning to link our application with firebase to run in the backend so as soon as there is a successful login the email id and the the unique srn and the password gets stored in the database therefore the main database we are planning to use would be firebase, GCP.

Similarly, we are planning if we conduct any kind of event in the department then a push notification should go to all the members of the club with details of the event and a link to get more details of the link like registration.

Next would-be attendance when an admin conducts a meeting, he/she will provide set a meeting in the events and then after the end of meeting he/she will open the attendance and only the users) members are supposed to mark themselves present by their own apps and this would be made proxy free by using a clause of local hotspot SSID.

The main operating systems we would be aiming for would be all android ranging from API level 21+.
We will be using multiple libraries like Squareup, Picasso, Bumptech, Glide, Junit and many more inbuilt libraries.

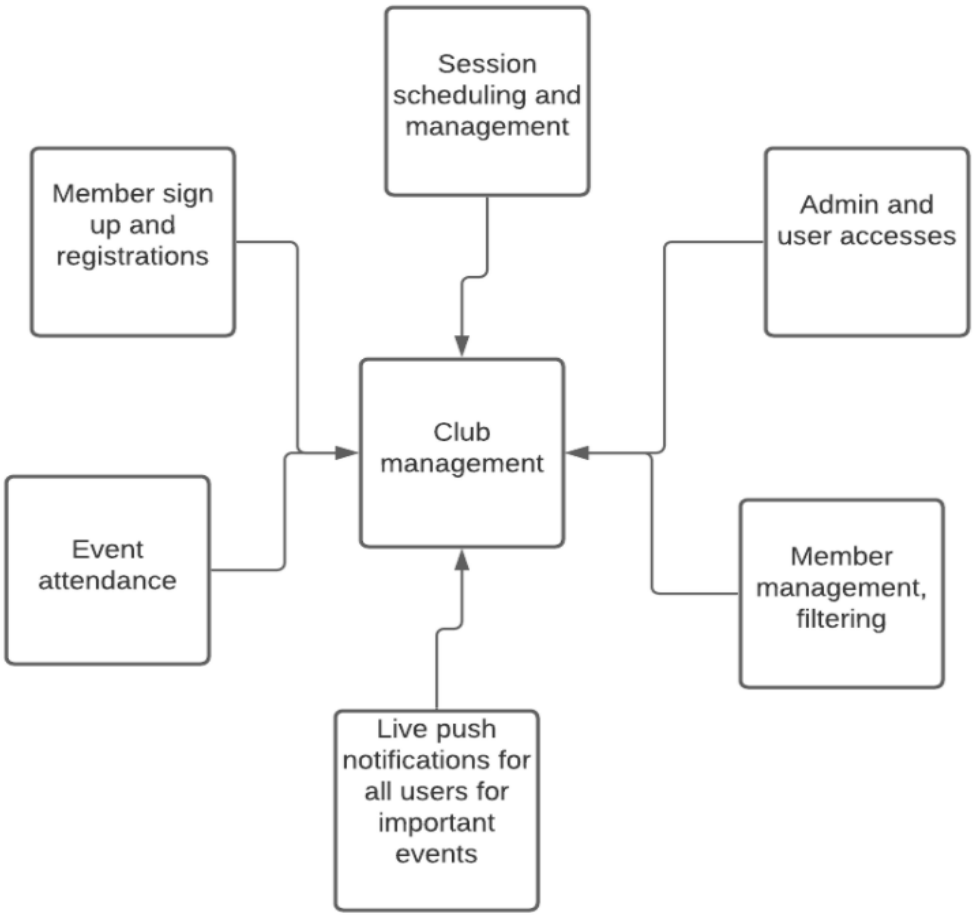
The only data that would be shared through software components would be all kind of data such as:

Attendance marking for sessions conducted by clubs. Session based attendance lists for admins, and user-based attendance lists for all users. Admins can schedule events in the app, and members can view all past events and upcoming events.Any urgent notices or special events can be shown on a page to users. These events are entered in the app by the admins. Attendance on a per user, per session basis is available for all admins, and users can track their own attendance throughout their history.
Admins can notify all members via a push notification system, similar to PESU Academy’s push system. These notifications are sent in the app.
Moreover, admins have a separate interface for the entire app from where they can control the information that would be displayed on the app.

3.3 Communications Interfaces

We are planning to send a push notification to every member of the club as and when the admin plans to keep a meeting, or a new event is going to take place this would be done by using firebase via the API calls.
The communication security is maintained by encryption and this would be done directly by the cloud service provider which we are planning to be google cloud service provider.
Synchronization is achieved by using atomic transaction interaction with Firebase which would be done via API calls. All the communication standards like FTP, HTTP would be internally handled by the API.
The major kind of message formatting that we would be using would be only the push notification; this would be only for the new upcoming events which the admin would update, and the entire text data of the event would be sent to all the members who have logged into the application.

4. Analysis Models



5. System Features

5.1 Registration

5.1.1 Description and Priority:

The most general feature of any application is the registration. Most of the apps would require the user to login/register into their app using their linked accounts like google, yahoo, etc. This is a high priority feature as registration should be done by everyone who enters the application.

5.1.2 Stimulus/Response Sequences:

Firstly, the user enters the app where the system asks to login to the app.
The user goes to the login page and enters the relevant details for login.
The user will be given a specific unique user identification number.
And hence, the user would be successfully registered with the app.

5.1.3 Functional Requirements:

The only requirement we would need here would the login credentials of the user which the user will themselves input to successfully register with the app.
It will first check if the credentials are correct, if they are not, then it will say “Invalid Credentials” and take the user back to the registration/login page.
REQ-1: Login Credentials (Google, Yahoo, etc.)

5.2 Attendance

5.2.1 Description and Priority:

One of the main uses of this application is to maintain attendance. We have mostly observed that it is very difficult for the club committee to maintain the attendance of its members or they face many problems doing that. Hence, we have this feature in the app itself where the club members can directly give their attendance. This feature is of high priority.

5.2.2 Stimulus/Response Sequences:

When all the members have entered the event, the admin has the option to start attendance.
All the present members will then mark themselves as present for that event.
The admin will disable this feature (SSID lock feature) after everyone has marked themselves present.

5.2.3 Functional Requirements:

The students need to have their attendance feature open in their app while doing this attendance for every event. Only if the member of the club is present for the event, they would know when the admin has switched the option for attendance on.
There would be instances where someone would have forgotten their phone or have no internet on their phone, then they have to contact the admin at the end of the event. The admin would take the UID and mark the member as present.

5.3 Events

5.3.1 Description and Priority:

Events is the most important feature of this app. We usually know that some members do not know about an upcoming event or what is going on in the club. To address this, we have included this feature so that all the members know about the ongoing and upcoming events of the club. This is a high priority feature.

5.3.2 Stimulus/Response Sequences:

Once the event date and time is finalized by the club committee, it asks the admin to update this event to their app. The admin describes the name, time and a small description about the event and adds it to the app. This would send a notification to everyone having the app that the particular event has been scheduled.

5.3.3 Functional Requirements:

There aren't any functional requirements as such. It is just a feature that would send a pop-up notification on everyone's phones when an event is scheduled. The admin would just create that session given the name and time for the event, If the members miss that pop-up notification on their phone, they can check it regularly on the app where the regular updates take place.

5.4 Members

5.4.1 Description and Priority:

This app also manages all the members of the app. The admin can add, remove, or update the details of a member. The members themselves can update their details. This is a medium priority feature as it is not used always.

5.4.2 Stimulus/Response Sequences:

If a new member has joined the club and is not able to login due to some credential issues, the admin has the feature to directly add that member to the club.
If an old member has left the club, then they need to inform the admin about it and they would be removed from the club app and they won't receive any updates.
If a member feels that their details need to be updated, they can either ask the admin to do it or they are themselves allowed to make some changes on the app itself.

5.4.3 Functional Requirements:

There aren't any requirements here as well. Just in case of any of the actions (adding, removing a member, or updating details of a member), they need to submit their UID for the following changes to take place on their account. If any of the updated details of any member is invalid, it will switch back to the previous details.

5.5 Info

5.5.1 Description and Priority:

"Info" feature is usually for the new members of the club who do not know about the club leads, committee of the club or the important members of the club. It is for the members to know which person holds what position at the club. This is a low priority feature available in the app.

5.5.2 Stimulus/Response Sequences:

If a member wants to know about the students holding important positions, they can go to the info section where we can see all the details of those students and can even directly get into contact with them given their mail-id/ phone number if the new members need any help.

5.5.3 Functional Requirements:

This is just an extension of the members feature but only a few selected members who hold very important positions will show their profile consisting of a few details like name, email-id, their position in the club, user ID number of the club, etc.

There is also a feature to join the whatsapp group of the club directly just by one tap on the app given in the info section.

6. Other Nonfunctional Requirements

6.1 Performance Requirements

Event scheduling is done in real time, which requires atomic transactions with independent listeners observing live data changes in the database. Push notifications are sent via GCP's messaging platform, accessed via Firebase.

6.2 Safety Requirements

Data loss is prevented by having backups of user data on Firebase as fallbacks. All user data is stored on Google's auth infrastructure and has its own security and backup features. This data is handled as per international guidelines and standards. These standards match all international safety certifications.
Admin user accesses are enabled through privileged data locks on the database, which can only be edited by admins who have access to the Firebase cloud. All other accesses are read-only, apart from attendance writes for all users in their own personal data nodes.

6.3 Security Requirements

All user data is stored securely on Google's cloud infrastructure, which manages its own security and access protocols. User authentication is handled by Google's OAuth integration for SSO options. All user data is first authenticated via Google before being accessed by us. Firebase access is securely accessed only by superusers or privileged admins with access to the database.

6.4 Software Quality Attributes

The application must be robust to handle simultaneous users and admin accesses, especially in the attendance and session management features. Each module is built separately in order to be tested and verified independently.

6.5 Business Rules

The admin is an important part of this app. This must be one or few of the core members of the club. The admin can allow registration, add members, remove members, update the details of the members, schedule events, start and end attendance sessions, and even modify the info page of the club app as the positions change frequently.
The club can also select everyone from the leads of the club to the organizing committee as the admin. In the sense, that anyone can take the attendance for that particular event.

The members of the club register themselves in the app, can mark their attendance when the admin allows, update their details, look into the upcoming events and can contact the important members of the club.

Appendix A: Glossary

- SSID: Service Set Identifier
- iOS: iPhone Operating System
- ROM: Read-only memory
- GCP: Google Cloud Platform
- XML: Extensible Markup Language
- UI: User interface
- API: Application Programming Interface
- FTP: File Transfer Protocol
- HTTP: Hypertext Transfer Protocol
- Android Studio: An app used to develop Android applications easily.

Appendix B: Field Layouts

Sample sheet with information required to register the member of the club:

Field	Length	Data Type	Description	Is Mandatory
Email-id	any	Alphanumeric	Mail id of the member	Y
Password	>8	Alphanumeric	Password of the account	Y

Appendix C: Requirement Traceability Matrix

Sl. No	Requirement ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID
1.	REQ-1	Login Account					