
SOFTWARE REQUIREMENTS SPECIFICATION

for

Attendance Application

Version 1.0 approved

Prepared by:	Sanskar Mittal	21CS10057
	Yash Sirvi	21CS10083
	Ashwin Prasanth	21CS30009

Indian Institute of Technology Kharagpur

March 27, 2023

Contents

1	Introduction	5
1.1	Purpose	5
1.2	Document Conventions	5
1.3	Intended Audience and Reading Suggestions	5
1.4	Project Scope	6
1.5	References	6
2	Overall Description	7
2.1	Product Perspective	7
2.2	Product Functions	7
2.3	User Classes and Characteristics	7
2.4	Operating Environment	7
2.5	Design and Implementation Constraints	7
2.6	User Documentation	8
2.7	Assumptions and Dependencies	8
3	External Interface Requirements	9
3.1	User Interfaces	9
3.2	Hardware Interfaces	9
3.3	Software Interfaces	9
3.4	Communications Interfaces	9
4	System Features	10
4.1	System Feature 1	10
4.1.1	Description and Priority	10
4.1.2	Stimulus/Response Sequences	10
4.1.3	Functional Requirements	10
4.2	System Feature 2 (and so on)	10
5	Other Nonfunctional Requirements	11
5.1	Performance Requirements	11
5.2	Safety Requirements	11
5.3	Security Requirements	11
5.4	Software Quality Attributes	11
5.5	Business Rules	12
6	Other Requirements	13
6.1	Appendix A: Glossary	13

6.2	Appendix B: Analysis Models	13
6.3	Appendix C: To Be Determined List	13

Revision History

Name	Date	Reason For Changes	Version
21	22	23	24
31	32	33	34

1 Introduction

1.1 Purpose

The purpose of this document is to define the software requirements for the development of an Attendance Application. This software is intended to be used by educational institutions, businesses, and organizations to track the attendance of students, employees, or members. The application will be designed to provide a user-friendly interface, allowing easy and efficient management of attendance records.

The Attendance Application will be developed to provide an automated system that simplifies the attendance-taking process. The software will be designed to reduce the workload of teachers, administrators, and human resource personnel who are responsible for tracking attendance.

This document will outline the functional and non-functional requirements of the software, including the user interface.

1.2 Document Conventions

<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>

1.3 Intended Audience and Reading Suggestions

This document is intended for the following audience:

- The development team responsible for implementing the Attendance Application software
- System administrators and users who will be responsible for installing and using the Attendance Application software
- Any other parties involved in the development, testing, and deployment of the Attendance Application software

To fully understand the requirements and constraints outlined in this document, it is recommended that readers have a working knowledge of the following:

- Object-oriented programming concepts and practices
- Android application development technologies and tools such as Android Studio and React Native
- Relational database management systems and SQL

To read the document, readers should start with the Overview sections, including the Introduction and Overall Description, to gain a high-level understanding of the Attendance Application software and its purpose. From there, readers can move on to the sections that are most pertinent to their specific role or area of responsibility, such as the Other Nonfunctional Requirements section for developers, the External Interface Requirements section for project managers and testers. Finally, readers should review any relevant appendices for additional information.

1.4 Project Scope

The Attendance Application software is designed to provide an efficient and reliable way for educational institutions to track student attendance. The system is intended to be used by both students and faculty, with different roles and permissions assigned to each user type.

The Attendance Application software will include the following features and functionality:

- Student attendance tracking
- Attendance reports
- Notification system
- User authentication and security
- Proxy Detection
- Importing and Exporting data

The Attendance Application software will be designed as a smartphone-based application that can be accessed by both students and faculty using their mobile devices. The system will be developed using industry-standard technologies and will be scalable to accommodate additional users and institutions as needed.

1.5 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

2 Overall Description

2.1 Product Perspective

The Attendance Application software is a new, self-contained product that is being developed to provide an efficient and reliable way for educational institutions to track student attendance. The system will be designed as a smartphone-based application that can be accessed by both students and faculty using their mobile devices.

2.2 Product Functions

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>

2.3 User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

2.4 Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

2.5 Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing

requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

2.6 User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

2.7 Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

3 External Interface Requirements

3.1 User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

3.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

3.3 Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

3.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

4 System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 System Feature 1

<Don't really say "System Feature 1." State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1: REQ-2:

4.2 System Feature 2 (and so on)

5 Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

5.2 Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

5.5 Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

6 Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

6.1 Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

6.2 Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

6.3 Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>