Software Requirements Specification

for

Notification Application

Prepared By Group 6

Athul V – 17
Gilu Vijayan – 28
Gouri P – 29
Kailas Unni - 35

CONTENTS

1. Introduction

1.1 Purpose

The purpose of a notification system that helps prioritize academic-related notifications is to streamline and manage the flow of information to students in a more effective way. With so much information coming from various sources such as official sites, faculties, and class representatives, it can be overwhelming and time-consuming to sift through everything to find what is most relevant and urgent.

By managing academic-related notifications in a centralized system, students can also avoid missing important deadlines or events that may impact their academic progress. Additionally, the system can provide a more personalized and tailored experience by filtering notifications based on the student's course load, interests, and academic standing. Overall, a notification system that helps prioritize academic-related notifications can help students stay on top of their academic responsibilities and ensure they are getting the information they need to succeed.

1.2 Document Conventions

- Use Times font for the entire document.
- Main Heading:

  Font size: 18

  Bold
- Sub Heading:

  Font size: 14

  Bold
- Body

  Font size: 12

1.3 Intended Audience and Reading Suggestions

The intended audience for a software requirements specification (SRS) document for a notification application for students would include project managers, software developers, quality assurance teams, system architects, and other technical stakeholders who are involved in the development and delivery of the software product. In addition, the intended audience for this particular SRS document would also include students, professors, and other academic staff who will be using the notification application. The document consists of a detailed description of all the functional and non-functional requirements and various perspectives of the product. It has all the features that are to be implemented in the product.

1.4 Product Scope

The product scope for a notification application for students that helps prioritize notifications would include features and functionalities that improve the flow of academic-related information, streamline communication between students and academic staff, and provide a personalized and user-friendly experience.

1. Notification Management System: The application should have a robust notification management system that can effectively manage the flow of notifications from various sources such as official sites, faculties, and class representatives.
2. Notification Prioritization: The application should prioritize academic-related notifications such as upcoming exams, assignment deadlines, and important announcements over general notifications and event invitations.
3. Personalization: The application should provide a personalized experience by filtering notifications based on the student's course load, interests, and academic standing.
4. Security and Privacy: The application should include security and privacy requirements to protect students' personal information and academic data.
5. User Interface Design: The application should have a user-friendly interface that is easy to navigate for students, professors, and academic staff.

2. Overall Description

2.1 Product Perspective

The product perspective for a notification application for students that helps to prioritize notifications should consider the user experience, system integration, business objectives, technology, data collection and analysis, security and privacy, compliance, and maintenance.

2.2 Product Functions

The product functions for a notification application for students that helps to prioritize notifications should focus on improving the flow of academic-related information, streamlining communication between students and academic staff, and providing a personalized and user-friendly experience.

1. Notification Management: The application should manage notifications from various sources such as official sites, faculties, and class representatives. The application should filter notifications based on their priority and relevance to the student.
2. Notification Prioritization: The application should prioritize academic-related notifications such as upcoming exams, assignment deadlines, and important announcements over general notifications and event invitations.
3. Personalization: The application should provide a personalized experience by filtering notifications based on the student's course load, interests, and academic standing.

4. User Interface Design: The application should have a user-friendly interface that is easy to navigate for students, professors, and academic staff. The application should also include features such as search and filter options to help students find the information they need quickly.
5. Technical Support: The application should provide technical support to students, professors, and academic staff to help them troubleshoot issues and provide feedback to improve the application.
6. Security and Privacy: The application should include security and privacy requirements to protect students' personal information and academic data. This includes data encryption, access control, and authentication mechanisms.

2.3 User Classes and Characteristics

The notification system should be designed to meet the needs of different user classes and provide a personalized and user-friendly experience for all users.

1. Students: Students are the primary users of the system. Students may have different academic needs and interests, and the system should be able to filter and prioritize notifications based on these needs.
2. Faculty: Faculty members are another user class that may interact with the system. They may use the system to post important course announcements, assignments, and other academic-related information. The system should provide a way for faculty to easily post and manage notifications.
3. Class Representatives: Class representatives may also use the system to post information and notifications on behalf of their respective classes. They may be responsible for organizing events, study groups, and other academic-related activities. The system should provide a way for class representatives to easily manage and post notifications.
4. Administrators: System administrators are responsible for managing the overall system, including user accounts, security, and data privacy. They may also be responsible for monitoring system performance and resolving technical issues.

2.4 Operating Environment

The operating environment for a notification application that helps to prioritize academic-related notifications should be designed to ensure high availability, scalability, and security. The application should be compatible with different mobile platforms and devices and provide a consistent user experience across different devices and networks.

1. Mobile Platform: The application should be designed for mobile platforms such as iOS and Android. The application should be optimized for different screen sizes and resolutions.
2. Network Connectivity: The application should be designed to operate on different network infrastructures like Wi-Fi, 3G/4G, and 5G. It should be designed to handle intermittent connectivity, network disruptions, and low bandwidth conditions.

3. Server Infrastructure: The application should be hosted on a reliable and scalable server infrastructure. The application should be able to handle a large number of concurrent users and provide high availability and fault tolerance.
4. Security and Data Privacy: The application should be designed to protect user data and privacy. The application should use encryption and access controls to secure user data and prevent unauthorized access. Additionally, the application should comply with data protection laws and regulations.

2.5 Design and Implementation Constraints

Design and implementation constraints for a notification application that helps to prioritize academic-related notifications from official sites, faculties, and class representatives may include the following:

1. User Interface: The application should have a user-friendly and intuitive interface that is easy to use and navigate. The interface should be designed to work well on different devices and screen sizes.
2. Data Integration: The application should be able to integrate with different data sources such as official academic websites, faculty portals, and class representative communication channels. The application should be designed to handle different data formats and ensure data accuracy and integrity.
3. Performance: The application should be designed to provide fast and responsive performance. The application should be optimized to handle large amounts of data and provide quick access to notifications.
4. Security: The application should be designed to provide a high level of security to protect user data and prevent unauthorized access. The application should use encryption and access controls to secure user data.
5. Scalability: The application should be designed to handle a large number of users and a high volume of data traffic. The application should be able to scale up or down depending on user demand.

Overall, the design and implementation of a notification application that helps to prioritize academic-related notifications should consider the above constraints to ensure that the application is user-friendly, performs well, is secure, compatible, and scalable.

2.6 User Documentation

User documentation for a notification application that helps to prioritize academic-related notifications from official sites, faculties, and class representatives should include the following:

1. Overview: Provide an overview of the application, including its purpose, features, and benefits.
2. System Requirements: List the system requirements for the application, including the supported mobile devices and operating systems.

3. Installation: Provide step-by-step instructions on how to download and install the application.
4. User Interface: Provide a detailed description of the user interface, including how to navigate the application and how to access and manage notifications.
5. Notifications: Explain how the application prioritizes and manages academic-related notifications from official sites, faculties, and class representatives.
6. Settings: Describe the different settings available in the application, including notification preferences, account settings, and privacy settings.
7. Troubleshooting: Provide troubleshooting tips and common issues that users may encounter while using the application.
8. Support: Provide contact information for customer support, including phone numbers, email addresses, and support forums.
9. FAQ: Provide a list of frequently asked questions and their answers related to the application

Overall, user documentation for a notification application that helps to prioritize academic-related notifications should be clear, concise, and easy to follow. It should provide users with a comprehensive guide on how to use the application effectively, troubleshoot common issues, and get support if needed.

2.7 Assumptions and Dependencies

Assumptions and dependencies for a notification application that helps to prioritize academic-related notifications from official sites, faculties, and class representatives may include the following:

1. Network Connectivity: The application assumes that users have access to a stable and reliable internet connection to receive notifications and access academic-related information.
2. Reliable Data Sources: The application depends on reliable and accurate data sources, such as official academic websites, faculty portals, and class representative communication channels, to receive academic-related notifications.
3. User Authentication: The application assumes that users will have valid login credentials and that the authentication process will be secure and reliable.
4. Device Compatibility: The application depends on users having compatible devices, such as smartphones with the required operating system version to install and run the application.
5. User Availability: The application assumes that users will be available to receive and respond to notifications in a timely manner.
6. User Privacy: The application depends on users willingness to share personal information, such as their name, email address, and academic information, to create a profile and receive relevant notifications.

Overall, assumptions and dependencies for a notification application that helps to prioritize academic-related notifications should be clearly defined and communicated to stakeholders to ensure that the application is designed and implemented in a way that meets user requirements and is reliable and effective.

3. System Features

3.1 Functional Requirements

For Users (Student/Faculty)

1. Register
2. Login
3. Notification log
4. Filter notification
5. View notification
6. Set reminders
7. View reminders
8. Notification history
9. Create custom reminders
10. View predicted notifications

For Admins

1. Register
2. Login
3. View notification
4. Notification history
5. Create notifications
6. Send reminders

Description

Registration

- Users can register themselves by typing their credentials.
- Unique username.
- Same person can have multiple roles.
- Password should contain minimum of 8 characters, should include atleast one character, one digit and one special character.
- Password is encrypted.

User database

Variable	Datatype	Constraints
fname	varchar	
lname	varchar	
user name	varchar	primary key
password	varchar	
role	varchar	
dob	date	
email id	varchar	
ph no	long int	10 digits

Login

Users can login by typing their credentials.

- Users can select their role.
- 5 chances are given to enter password.
- Password recovery through email.

Notifications Log

This is the home page that comes after login. Users can view their notifications.

- Notifications are auto prioritized based on keywords.
- Users can filter their notifications in any manner including topic wise, priority, time, source, etc.

View Notifications

Users can select a particular notification and view in detail.

- Modify reminders.
- Modify priority manually.

View reminders

Users can view their reminders and modify these accordingly.

- Modify already set reminders.

Notification History

Users can view notification history.

- These include previous notifications that are already expired.

Create custom reminders

Users are able to create custom reminders for their personal needs.

- These reminders can also be auto prioritised.

Functions exclusively for admins

Create Notifications

Admins can create notifications.

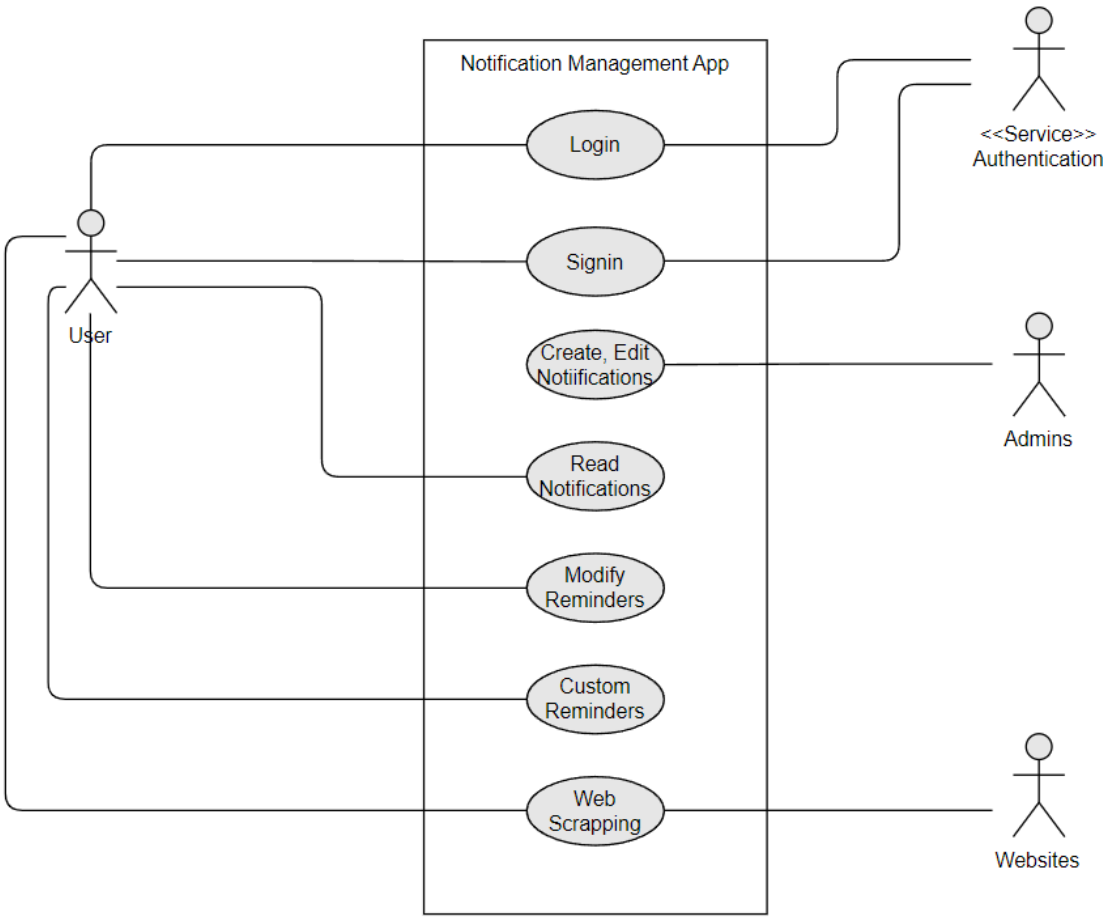
- The data includes which class or student group that particular notification is being referred to.
- The priority of the notification can be set manually.

Modify Reminders

Admins are able to send reminders to already send notifications.

- These may be to modify a particular notification like if the due date of an assignment has been changed.

3.2 Use Case Diagram



4. External Interfaces Requirements

4.1. User Interfaces

A. Sign in page
B. Login page
C. Home page
D. Profile page
E. Settings page
F. History page

4.2. Hardware Interfaces

A. Operating system: Windows 11
B. Hard disk: 512 RAM: 4GB
C. Processor: AMD Ryzen 7

4.3. Software Interfaces

A. Figma
B. Android Studio IDE
C. Flutter
D. Django
E. Mongodb server

5. Other Non-Functional Requirements

Non-functional requirements for a notification application that helps to prioritize academic-related notifications should be defined and communicated to stakeholders to ensure that the application meets user expectations and provides a high-quality user experience.

5.1. Performance

The app shall be responsive and provide notifications in a timely manner. The system should be able to handle large amounts of traffic at a time. It should be fast and efficient.

5.2. Safety and Security Requirements

The app shall maintain the user's privacy and prevent unauthorized access to user data. Sensitive notifications should be delivered to appropriate users only.

There shouldn’t be any partiality and every user should receive intended notifications.

5.3. Software Quality Attributes

A. Reliability

The app shall function consistently and without errors.

B. Compatibility

The app shall be compatible with Android platform.

C. Usability

The app shall be user-friendly and easy to use.

D. Scalability

The app shall be designed to support future updates and enhancements.

E. Availability

The application should be available to users at all times, with minimal downtime for maintenance or upgrades.

F. Accessibility

The application should be accessible to users with disabilities, including support for assistive technologies and compliance with accessibility standards.

6. Other Requirements

6.1. Database Schema

Users

username	String
role	Type:string Enum:[“USER”,”ADMIN”]
fullname	String
email	String
phone_no	String
password	String

Notifications

notification_id	String
message	String
priority	Integer
date	Date