Department of Computer Science Engineering

SRM IST, Kattankulathur – 603 203

18CSC206J – SOFTWARE ENGINEERING AND PROJECT MANAGEMENT

Experiment No	10
Title of Experiment	Develop a Testing Framework/User Interface
Name of the Candidate	Sai Rohit P
Team Members	Sai Rohit (RA2111003010806) Pavan Sagar (RA2111003010809)
Date of Experiment	

Mark Split Up			
S.No	**Description**	**Maximum Mark**	**Mark Obtained**
1	Exercise	5	
2	Viva	5	
Total		**10**	

Staff Signature with date

Aim:

To develop the testing framework and user interface framework for TimetableSOS

Team Members:

S. No.	Register Number	Name	Role
1	RA2111003010806	Sai Rohit	Rep
2	RA2111003010809	Pavan Sagar	Member

Project Title: TimetableSOS

Executive summary

The software application in question is a timetable management app for students. The app aims to help students manage their schedules by providing real-time notifications for their classes. The app's primary objective is to ensure that students never miss a class and are always aware of the important details related to their classes.

The scope of the project is to develop an easy-to-use app that allows students to input their class timetables and receive timely notifications. The app's focus is to provide an in-dismissible notification for the current class happening, which includes important information such as the name of the class, start and end time, classroom details, and the name of the professor conducting the class.

To ensure the app's effectiveness, the software application will undergo several testing approaches. These approaches include functional testing, usability testing, and performance testing. Functional testing will be conducted to ensure that the app's features are working correctly. Usability testing will be done to ensure that the app is easy to use and navigate. Performance testing will be conducted to ensure that the app's response time is quick, and it can handle a large number of users.

Test Plan:

Scope of Testing: The scope of testing for this project includes both functional and non-functional testing. The functional testing will cover all the modules of the software application to ensure that all features are working correctly. There will be no exceptions for any modules. The testing will include both manual and automated testing. The automation will cover all the functional test cases or regression-critical path test cases.

Non-functional testing will ensure that all non-functional requirements (NFR) are covered. This includes testing the performance, reliability, scalability, usability, and security of the software application. The testing will be done using appropriate tools and techniques to ensure that all the NFRs are met.

The testing approach will follow the following steps:

1. Test Planning: This will involve identifying the testing objectives, identifying the testing team, and defining the testing scope.
2. Test Design: This will involve creating the test cases, test scenarios, and test scripts to be used in testing.
3. Test Execution: This will involve executing the tests and analyzing the results.
4. Defect Management: This will involve tracking and managing any defects found during testing.
5. Test Closure: This will involve reviewing the test results and providing a summary report.

In summary, the testing scope for this project includes functional testing, which covers all modules of the software application, and non-functional testing, which covers all non-functional requirements. The testing approach will follow the steps of test planning, test design, test execution, defect management, and test closure.

Types of testing

Category	Methodology	Tools Required
Input validation	Manual	None
Timetable creation	Manual	None
Real-time notification	Automatic	Test automation tool (e.g. Appium, Espresso), Notification testing libraries (e.g. FCM Test Lab, AWS Device Farm)
Class information display	Manual	None
Classroom details	Manual	None
Professor information	Manual	None

Note: The tools required for input validation and timetable creation are not applicable as they are done manually. The tools required for real-time notification testing could include a test automation tool like Appium or Espresso, along with notification testing libraries like FCM Test Lab or AWS Device Farm. No specialized tools are needed for testing class information display or classroom details as they can be manually verified.

Result: Thus, the testing framework/user interface framework has been created for TimetableSOS