To-Do List

Test Plan Document

Project 2, Team 6

Introduction

The software Product which we are working on is titled "To-Do List for Android". The To-do List allows users to manage tasks that they have to accomplish. The user can add tasks, set the priorities of each task, set the due date for each task, check-off items in the list and hide/show the checked items. The application will support multiple users and will be developed using Android Developer Tools. This document gives a brief about our testing strategy for the project. Its main objective is to set different quality standards for the unit, integration and system testing of the specified application.

Quality Control

Test Plan Quality

The main objective of this activity is to detect and fix as many bugs or defects in the code as possible. This will in turn help in improving the quality and reliability of the program code. The more rigorous the Test Strategy is, the more chances there are of getting a bug free program.

We will try to ensure the most thorough and efficient Test Plan for our project.

Adequacy criterion

The Adequacy Criterion depends on the extent to which the test cases made are successful in debugging the program code. Hence we need to set the range of the test data in such a way that most of the bugs get detected and fixed in this phase.

Our tests are exhaustive and based both on extreme and common cases, that's why it will ensure an adequate software code delivered to the customer.

Bug Tracking

The bugs found by implementing the test cases need to be documented and fixed properly. The Testing phase enables the testers to put in any random value and record the case if we get any erroneous or strange value as the output. This case is then recorded and reported back to the development team to get permanently fixed. The bugs found are stored for reporting and reference usage.

Test Strategy

Test Strategy is the plan by which we will analyse our program code and detect the differences between existing and required conditions. This strategy should be made in a way so that the gap between what exists and what is desired is minimized. We are carrying out two basic Test activities for our project – To-Do List.

Unit Testing- Each software module is tested internally at the development phase itself and all possibilities of internal bugs at each unit are removed.

Integration Testing- Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests to those aggregates, and delivers as its output the integrated system ready for system testing.

System Testing- The system once integrated is tested as a whole and the output is checked for any anomalies.

Regression Testing- Once bugs are fixed, regression testing is conducted to verify whether fix is correct and if it causes new bugs.

Testing Process

The testing process will consist of these procedural steps:

Procedural Steps

- As a part of **Unit testing**, we will test the individual modules of the program like Signup, Login, Add/Edit/Delete, Sort/View, Hide/Show modules.
- As a part of **Integration testing**, we plan to use the big bang approach to integrate all modules for testing as the application is not very complex.
- **System testing** will be conducted based on the system requirement to test the system as a whole.
- In **Regression testing**, we will execute the failed test cases to verify whether the fixes which have been implemented are correct

Technology

Since it's a code being implemented using the Evolutionary Prototype Model, there's a specific testing tool that we are using for testing purposes named JUnit. The Test Cases are being implemented in the same way as any Customer would try to use the App in his Android phone and exercise its in-built features.

Test Cases

The Test Cases that we formulate to test the code forms the basis of our testing process. It also shows us how strongly the Code can withstand against any arbitrary or unexpected input values without crashing. The Test Cases that we plan to execute include:

Module	Test Case#	Purpose	Steps	Expected Result	Actual Result	Pass/ Fail
Sign up	**01**	Test whether new users can sign up correctly	Try signing up with expected format of username and password	Should sign up successfully	The user can sign up successfully	Pass
	02	Test whether the program validates the input for signing up	Try signing up with empty field in username.	Should instruct users to give a correct username	Display error message: "Please enter username"	Pass
	03		Try signing up with empty field in password	Should instruct users to give a correct password	Display error message: "Please enter password"	Pass
	04		Try signing up without confirming password	Should instruct users to confirm password	Display error message: "Please confirm password"	Pass
	05		Try signing up without providing email	Should instruct users to provide email	Display error message: "Please enter a valid email"	Pass

	06	Test whether the program can identify an existing username	Try signing up with an existing username	Should instruct users to sign up with another username	Display error message: “User already exists”	Pass
	07	Test whether the user can cancel signing up	Click “Cancel” button in sign up page	Should cancel signing up	Signing up is cancelled	Pass
Login	**08**	Test whether the app can identify a pair of username / password which is already signed up on the app.	Try to login using a valid username / password pair	Should login correctly.	The user can login correctly	Pass
	09	Test whether the program identify an incorrect username / password pair	Try to login using an incorrect username/ password pair.	Should not login and prompt the user for correct entry.	Display error message: “No record”	Pass

	10	Test whether the program can identify an empty field in the usernam e.	Try to login using an empty field in either username.	Should prompt the user for an empty field.	Display error message: “Please enter username”	**Pass**
	11	Test whether the program can identify an empty field in the password field.	Try to login using an empty field in either password.	Should prompt the user for an empty field.	Display error message: “Please enter password”	**Pass**
Manage	**12**	Test whether the user is able to add a task with specified priority and due date.	Try adding a task from the main page and specifying priority and due date.	Should add a task in the tasks list view with specified priority and due date.	The user can add a task, but the priority display is wrong	**Pass**
	13	Test whether the user is able to add a task with default priority and due date.	Try adding a task from the main page without changing priority and due date.	Should add a task in the tasks list view with default priority and due date.	The user can add a task with default priority and due date.	**Pass**

14	Test whether the user is able to change the task name for a particular task.	Go to the task details of a particular task, and then try changing the task name.	Should be able to edit without any errors.	The user can edit the item name, but will create a new item after saving and priority/due date will be the default value.	Pass
15	Test whether the user is able to change the priority of a particular task	Click on edit task and try changing the priority.	Should be able to change the priority.	The user can change the priority, but the priority cannot be saved correctly and a new item will be created.	Pass
16	Test whether the user is able to change the due date of a particular task.	Try changing the due date of a task.	Due date should change without any errors.	The user can change the due date, but a new item will be created.	Pass
17	Test whether the application can identify an invalid due date when	Try adding task and specifying the due date to a past date	Should remind the user to change the due date.	A pop-up appears saying “You can’t enter past date.”	Pass

	adding a task				
18	Test whether the applicatio n can identify an invalid due date when editing a task	Try editing task and specifying the due date to a past date	Should remind the user to change the due date.	The user can save the change successfully.	Fail
19	Test that if the user gets a phone call while using the app, then app is restored after the phone call in the same state as before the call.	Try calling the cell phone on which the app is installed and running and check for the behaviour.	Should be running smoothly and in the same state after the call ends.	The application runs smoothly and in the same state after the call ends.	Pass
20	Test whether the user is able to check the check box in the list view	Try checking the checkbox in the main list view.	Should check correctly.	The user can check the check box correctly.	Pass
21	Test whether the user is able to	Try scrolling down in the task list view.	Should scroll without any errors.	The user can scroll down the list.	Pass

		scroll down the list without any errors.				
	22	Test whether the user is able to delete a selected task	Click on a particular task, wait the menu to appear and click delete button	A pop-up appears to ask users to confirm deletion; delete item if the user confirms.	A pop-up appears to ask users to confirm deletion; delete item if the user confirms.	**Pass**
	23	Test whether the applicatio n can automati cally save a task when adding a task	Try adding a task without click “Save” button, and click “back” button	The task should be saved automaticall y	The task is saved automaticall y and displayed in the list	**Pass**
	24	Test whether the applicatio n can automati cally save a task when editing a task	Try editing a task without click “Save” button, and click “back” button	The task should be saved automaticall y	The change is not saved	**Fail**
View/Sort	**25**	Test whether the user is able to	Click on a particular task, wait the menu to	Should sort by due date	The user can sort by due date	**Pass**

		sort the tasks by due date	appear and click sort by due date			
	26	Test whether the user is able to sort the tasks by priority	Click on a particular task, wait the menu to appear and click sort by priority	Should sort by priority	The user can sort by priority	Pass
	27	Test whether the user is able to view detailed informati on about a task	Click on a particular task, wait the menu to appear and click "Details"	Should open a new page displaying detailed information about the task	A new page appears and displays detailed information.	Pass
Hide/Show	**28**	Test whether the user can show hidden tasks	Click on 'hidden items' in the menu	Should show all hidden items	Shows the list of hidden items	Pass
	29	Test whether the user can hide the tasks	Click on check box and then confirm hiding	Should hide the item	Hides the Item from the main list	Pass
Logout	**30**	Test whether the user can log out successfu lly	Try to click the "log out" button	Should log out and return the login page	The user cannot log out	Pass