Test Plan Documentation:

Project: Unit Testing on “Pill Reminder” Mobile Application

Course: CSE427

Course Title: Software Testing and Quality Assurance

Section: 01

Semester: Spring 2019

Submitted to: Shaikh Shawon Arefin Shimon

Submitted by: Group 06

1. Noshin Islam 1521733642

2. Asik Azad 1510612042

Name	Status	Step	Expected Result
@Before Test (Reminder Class)	Successful	1. First we create a Test rule of “mainactivity” type. We make an instance of the rule called “mActivityTestRule” 2. By using the @Before annotation, we can test if the required activity can be invoked by the created rule, which is preliminary. In the @Before method we try to invoke the Reminder class activity which was a success.	The activity rule instance should be able to invoke the main activity fille which is “ReminderActivity ”. Runs once before any of the test methods in the class
@After Test	Successful	It runs after all the test cases of the class. As we made an activity instance with the rule instance we have to make that class instance null after we are done with it. We use the **TearDown** method to do this	If we allocate external resources in a Before method you need to release them after the test runs.
Test Case 1	Succesful	First test is to test I if the Text view of where patients will enter their message to be invoked is successfully launched or not. The name of the method is **testLaunchActivity_TextView** , First **View** class instance is created to find the necessary id name of the field.	Text View should be successfully shown on the screen
Test Case 2	Successful	Second test is to see if the Cancel button that cancels the written text is working or being viewed or not. The method being tested here is **testLaunchActivity_cancelbuttonView**	Cancel Button should be sucessfully shown on the app
Test Case 3	Successful	Third test is to see if the Setbutton view is working or not. **testLaunchActivity_setbuttonView**	Set button should be shown and work

Test Case 4	Successful	There is a timepicker in the app, by which patients can select time precisely. To see if the timepicker is shown: **testLaunchActivity_timepickerView**	Timepicker should be shown on the main activity view and should be able to navigate properly
Test Case 5	Successful	To test if the medicine name view where the medicine nme will be put is invoked or not we tested **testLaunchActivity_medicinenameView**	Medicine name text field should be shown
Test Case 6	Successful	To test if the iconreminder view is shown or not this is tested testLaunchActivity_iconreminderView	Icon reminder should be successfully shown
Test Case 7	Successful	To test if the layoutreminder view is shown or not this is used: testLaunchActivity_layoutreminderView	Reminder layout should be shown
Test Case 8	Successful	We test the Launch of set button. The activities that needs to be checked are: setbutton view is not null, setbutton should be able to perfom if a click happens, Intent class instances can be made to connect the reminder class and alarm class. All these are tested by the method: **testLaunchOfSetButtonActivity**	Set button should meet all the requirements to have a successful click operation.
Test Case 9	Successful	The test method: **testonClick** It tests the View class instance and its workability, It tests the Timepicker actitivity. It also tests the getTimeInMillis method	Timepicker should be able to pick the time (hour, minute) in miliseconds.
Test Case 10	Successful	We test the **getTimeInMillis()** separately because it is crucial that we get the hour and minute by using the Calendar class instance	Time should be get is Hour, Minute format and converted to miliseconds.
Test Case 11 (Alarm Class)	Successful	1. @Before We make an ActivityRule for Alarm class and make an instance of to try to test if the Alarm class activity can be invoked or not. We	Activity Test rule for Alarm class is needed to work if

		make an instance called alarmActivity of Reminder class. 2. @After reallocates the used resources after their work is finished	the Intent is created.
Test Case 12 (MyBroadcastReceiverTest class)	Partly Successful	**testIntentHandling** Alarm Receiver works, serviceIntent functions so that Intent calling is raised.	To handle Intents the connection between BrodcastReceiver and Alarm class should be smooth.
Test Case 13	Successful	**onReceive** method takes a context and intent as its parameter, it is testing if the notification manager can connect between the contexts and trigger the intent to get system service which is to get a notification in due time.	Patients need to be sent notification in the stored time.