# 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 make an instance called alarmActivity of Reminder class.  2. @After reallocates the used resources after their work is finished | Activity Test rule for Alarm class is needed to work if the Intent is created. |
| Test Case 12 (Espresso Input Space Partioning ) | Successful | **testMedicineNameInput(),** **testMedicineNameInput2(), testMedicineNameInput3()** uses Espresso framework to test input type of medicinename field to be only alphabets | To allow users to only input valid medicine names |
| Test Case 13 | Successful | **testMedicineNameInput4InvalidInput()** tests the medicinename field for invalid input types. i.e. 123napa | Patients need to be check before entering the medicine name |
| Test Case 14 | Successful | **improperTimeFalseCase()** invokestocheck the reminder time if it is the right time to set the alarm. | Proper time should be between 5 am to 11 pm. |
| Test Case 15 | Successful | **improperTimeTrueCase()** invokestocheck the reminder time if it is the wrong time to set the alarm. | Improper time should be after 11pm to 5 am. |

**Input space partitioning graph:**

MedicineName TextField Input Space Partition:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| q1 = Medicine Name Input text type | All capital letters i.e. NAPA | ALL small letters  i.e. napa | Small and Captail mixed letters i.e. NaPa | Invalid Input i.e. 12napa, 12\_napa, #napa |

Input Space partitioning for TimeCheck:

|  |  |  |
| --- | --- | --- |
| **Function** | **True case** | **False case** |
| **improperTime()** | **{ 1 ,3,4,24}** | **{6, 8, 11, 16}** |

**Graph Partioning:**

Medicine-Name Input type testing

Medicine Name

testMedicineNameInput4InvalidInput() //123napa

testMedicineNameInput() //Napa

testMedicineNameInput3() //NAPA

testMedicineNameInput2() //napa

testCase failed.

testCase Passed.

Fig: Graph for Medicine Name text testing

Alarm setting ImproperTimeCheck:

**improperTime():**

**! improperTime()**

int hour = 20;

assertEquals(false,result)

**improperTime()**

int hour = 23;

assertEquals(true,result)

**Return false**

**Return true**

Fig: Graph for Improper Time-Check