# Software Engineering - UE19CS302
# Assignment-3: Software Testing

Date: 24/11/2021
Section: H

| | |
|---|---|
| Sumukh Raju Bhat | PES1UG19CS519 |

**Scenario #1** - Mobile Application to scan documents for various customers with a need to function on a variety of hardware and OS.

**a) Strategy being planned:**

1. *Test mindset/models:*
    - As testing is constrained to a certain time frame, Preventive mindset/model could be the best as it prevents faults in early phases itself through careful planning and design. If not this, we know that the faults found in the later part of the lifecycle are harder, time consuming and costly to fix. Reviews and test driven development are a part of this model which is required as many types of customers use the application.
    - If not a preventive model, we can always use a demonstrative model, where we need to write all possible test cases for all compatible hardwares and OS. This model although meets all the given constraints of our application, however could break the time constraints, which is a necessity.
2. *Test types:*
    - Functionality testing: This is mandatory as for any application, the minimum requirements would be that it should be properly functional with respect to what it is actually supposed to do. Speed, smoothness etc which are non-functional requirements are secondary. (Eg: FC1 test case written in test cases section of the report)
    - Exploratory Testing: This should be done by experienced software engineers in the field of mobile applications. This would ensure maximum coverage of bug/fault finding, also in a minimum time.
    - Cross-Platform Testing: The necessity is to test the application on all hardwares and OSes.
    - Performance testing: We need to test the non-functional features of the application with particular workloads like heavy workloads as multiple users will be using the application. (Eg: NFC1 test case written in test cases section of the report)
    - Back-end testing: This is a necessity for our application as the application has to work on n-number of hardwares and OSes. Here we test if the data is stored in the right place and in the right format.
    - Storage testing: Mobile devices have relatively less memory and will typically be filled with multimedia files which consume a lot of space. Hence we need to avoid any memory related exceptions and overflows
3. *Test environments*: Setup of software and hardware to execute test-cases
    - Hardware/Devices: Smartphones, Tablet
    - OS: AndroidOS, iOS
    - Databases: MongoDB, Firebase, Realm, SQLite

- Network: WiFi/3G/4G internet connectivity, WPA2/WPA3 for security

4. *Test automation:*
   - Testing should be automated for the reasons that it increases the effectiveness and efficiency of testing as we have to test on multiple combinations of hardware and OS.
   - Steps:
     1. Define the goals: functional and non-functional requirements as in the scenario description.
     2. Deciding test approach/types: Determining the level of testing, roles and responsibilities of the team members. (Types are already discussed above)
     3. Selecting an automation framework/tool:
        - Appium is one of the best for mobile applications as it is open-source, allows for cross-platform testing and is json based which is easier to learn.
     4. Create and run the test scripts/cases on it
     5. Maintain scripts/cases

**b) Test Cases:**

1. Test Case ID: FC1
   Title: Scan and Save
   Description: A user must be able to scan documents using the application using a camera phone/smartphone.
   Precondition: User must have logged in as a valid user of the application and the application should have access to the phone's camera.
   Assumption: The phone is a smartphone or a tablet, should have either AndroidOS or iOS and AndroidOS should have version 7 or later and iOS should have version 9 or later. The phone should have a working camera.
   Testing steps:
   1. Open the application
   2. Click on the scan option on the homescreen
   3. Hit on capture option to capture the scan
   4. Crop the image to necessary dimensions
   5. Click on save

   Expected results: If the scanning device has a AndroidOS, the scanned document must be saved in the User's gallery folder. If it has iOS, the scanned document must be saved in the Photos folder. It shouldn't have any loss of information when compared to the scanned document. The above mentioned results are the same for both smartphones and tablets.

2. Test Case ID: NFC1
   Title: Time to scan
   Description: A application should scan the document with a certain time constraints
   Precondition: User must have logged in as a valid user of the application and the application should have access to the phone's camera.
   Assumption: The application passes Testcase FC1. The phone is a smartphone or a tablet, should have either AndroidOS or iOS and AndroidOS should have version 7 or later and iOS should have version 9 or later. The phone should have a working camera.

Testing steps:

1. Open the application
2. Click on the scan option on the homescreen
3. Take a note of the present timestamp(before scan) from the system clock, say I (in milliseconds)
4. Hit on capture option to capture the scan
5. After it successfully scans, note the timestamp from the system clock, say F (in milliseconds)
6. Find the difference between F and I, and call it D i.e. D = F - I (in milliseconds)
7. Discard the scanned copy
8. Quit the application

Expected results: The upper bound value of D for various hardwares and OS are summarized in the below table(in milliseconds):

| | Smartphone | Tablet |
|---|---|---|
| AndroidOS | 1 | 2 |
| iOS | 0.8 | 1.5 |

(to be got from past experience)

**c) METRICS:**

1. Cost of quality: Total cost of prevention, appraisal, rework/failure to the total cost of the project.
2. Defect discovery rate: Number of defects found per line of code.
3. Defect removal cost: Cost associated with finding and fixing a defect.

Even though the usual aim would be to reduce the values got in the above metrics, our application has high quality related demands. Hence the team shouldn't worry if the value overshoots the usual values obtained from their past experiences. At the same time, the team should aim at minimizing the metric as much as possible, wherever possible, if not 'the minimal'. For example, if it was observed in the past that cost of quality was 15%, and the current project has it at 20%, it is fine. But a value, say greater than 60% is not acceptable when compared to the 15% value. This can be judged by someone with greater experience of working with such mobile applications in the past.