Beta Release - Five Girls

Testing Documents 25%

Test Plan

This is the testing for the application - Test Cracker. The testing aims to test whether all functions of the application are implemented well.

Goals for the tests:

- System meets all the user requirements
    - Can correctly search courses, add and delete favorite courses, upload and download course materials
    - Can mark and delete the calendar easily
    - Can only see the course materials when the user has already logged in
- System works under normal circumstances
    - Show notification when saving the image successfully
    - Show notification when registering successfully
    - Show notification when logging in and logging out
    - Show notification when marking calendar successfully
- System works in the corner cases
    - Show notification when entering incorrect username and incorrect password
- System response correctly in different situations
    - Open the course materials when the user has already logged in
    - Open the login page when the user hasn't logged in yet

Basic testing strategy used:

The testing is conducted manually. The tester tests each function one by one.

Test Approach:

Black Box Testing

Test Environment:

Java Development Kit (JDK)
Android Studio 3.3.2 for Mac

Testing Tools:

A MacBook Pro with macOS Mojave

A list of resources used for the tests:

- Computers
- People
- Time
- Backtest
- SIS

Schedule for testing:

Write all the test cases before coding and run the test case right after finish function. Repeat the testing until the application passes all the test cases.

Test Case

According to the use cases we come up with, we analyze different actions that might occur during the actual running of the application by going through the sequence of action.

TC1: AddFavoriteCourse

- Precondition: Must has course
- Sequence of action:
    - Clicks add button
      System add the course to Student's MyCourse list, which is in MyProfile page.
      User may able to see the course in the MyCourselist

TC2: UserLogin

- Precondition: Must has one register account
- Sequence of action:
    - Enter correct username and password
      System should login
    - Enter invalid username
      System should return incorrect username
    - Enter invalid password
      System should return incorrect password

TC3: UserLogout

- Precondition: Must has account login
- Sequence of action：
    - Click logout
      System should logout and return to the login page
      Next time user enter the system, system should ask the user to login and do not allow the user who haven't login to get access to any data in the system

TC4: CreateNewAccount

- Precondition: the account we are going to create have not exist in the system
- Sequence of action:
    - Enter correct username email and password
      System should verifies information and register a new account
    - Enter invalid email
      System can not verifies information and return invalid email address
    - Enter invalid password
      System should return invalid password format
    - Enter invalid username
      System should return invalid username

TC5: SearchCourse

- Precondition: System must has a list of course
- Sequence of action:
    - Hit the search bar
      System should prompt keyboard to type the keyword of the course
    - User type the keyword in the list
      System should show the list of course name which includes the keyword.
    - User type the keyword not in the list
      The System should show no result.

TC6: ViewCourseMaterial

- Precondition: System must contains the course and material for selecting course
- Sequence of action:
    - Hit the title of the course
      System should prompt page of material under the selected course.
      System will show an empty page if there is no material uploaded for this course
    - Hit the title of the material
      System should show enlarged corresponding material

TC7: UploadMaterial

- Precondition: System must contains the course
- Sequence of action:
    - Hit the upload button
      System should open upload course
    - Add the upload photo
      System should add photo to queue in upload tool.
    - Enter the title and descriptionfor the photo
      System should show the title and decsription for this photo after the photo be added
    - Add multi photos
      System should add all photos to queue in upload tool.
    - Upload invalid format photo (like pdf)

System should return invalid format
    - Click ‘x’ to remove the photo
      System should remove photo from queue
    - Click the upload button
      System should upload photos to database

TC8: SaveMaterial

- Precondition: System must contains the course and the course material
- Sequence of action:
    - Click the material
      System should open the material
    - Long press the image
      System should show save successfully

TC9: MarkCalender:

- Precondition: System must has the calendar
- Sequence of action:
    - Click the calendar
      System should open the calendar page
    - Long press date
      System should promote the adding page
    - Enter the title and information
      System should add the information in queue
    - Click add
      System should add the information to database, there will be a blue circle on the date

TC10: DeleteCalendar:

- Precondition: System must has the calendar and the mark
- Sequence of action:
    - Long press mark
      System should promote the delete page
    - Click delete
      System should delete the mark in database
    - Click cancel
      System should return to the calendar page

Testing Result

Testing iteration: 4/16/19 2:00
Tester: Jennifer

Test Case	Result	Comments
TC1	Pass	
TC2	Pass	
TC3	Pass	
TC4	Pass	
TC5	Pass	
TC6	Pass	
TC7	Fail	Cannot upload multi photos
TC8	Fail	Cannot save photos locally
TC9	Pass	
TC10	Pass	

Code Review 15%

In the class, we first review the code from Team Great. The code is well organized with sufficient comments. By having comments for each function in each class, the code will be easily read for developers who join the project later and team members to read each other's work. We probably want to adopt this type of commenting. The class diagram is drawn in the standard format and the code perfectly match the diagram. Each class in the code only have a single task and the data inside the class cannot be modified directly. During the code reviewing, Team Great plans to add a feature to the Spectator class. It would be fun if spectators get more involved in the game. They mentioned allowing the spectators to see cards players hold. The UI can be revised so that users, players, and spectators can play and chat more conveniently without scrolling up and down the page.JavaScript front-end part can be formatted so can be read easily. Overall, the idea is really good. Team Great has good coding style.

Honex Ind reviewing our coding in the class. So far our coding is well-organized readable. However, we need to add more comment to the code to help others outside our team to understand the code. Each event work as its own task and they are the subclasses of the BaseActivity class. The encapsulation is great cause we cannot modify the data directly. And all the activities classes inherit the BaseActivity. Our code matches the static class diagram we had for the interim release. Our UI talks to activities which talk to DB helpers. We may need to default ‘not logged in’ statement to the login page so that our login page has a default statement. Overall, more comment is needed and we have a great coding style.

Contribution Summary and Status Report 5%

Contribution Summary

Yiran Zheng

- Work on the code reviewing section by summary the comments and advice from Honex Ind.
- Go through the source code and adding comments
- Suggest several test case we might need to go through
- Implement status report
- Read over the other work and suggest improvement/modification

Zhou Lu

- Implement status report
- Read over the other work and suggest improvement/modification
- Organize Git repository

Sylvia Hua

- Wrap up the source code and add comments.
- Work for the calendar feature.
- Read over the other work and suggest improvement/modification.
- Status report

Jennifer Fu

- Write the test case
- Write the status report

Chenhao Pan

- Write the test plan and test strategy
- Suggest more test cases
- Help with the testing
- Read over the other work and suggest improvement/modification

Status Report

In Sprint 5-9, we realized all the functions as described in use cases in Sprint 4 deliverables and this app is ready for the beta release. Now we are working on tests and debugging process. Trying to make this app bug free, we created a couple of test cases. Unfortunately, the uploading feature and saving image feature broke down today. While feeling frustrated, we are trying to fix them and figure out what are the reasons. Later in the evening, we find out the reason why the uploading picture and saving the image to local feature could not work on the class. Since we change the virtual machine we usually used and create a new virtual machine on a new device, we haven't authorized the application on this new virtual machine. After authorizing the application on the machine, the uploading and saving feature can work correctly now. We will add more exception handlers in the testing.

For the last two week, we are going to prepare the presentation and final release. We will use the last two week to debug the code and try to import all the courses in the course list.