Mobile App Development
Midterm Exam

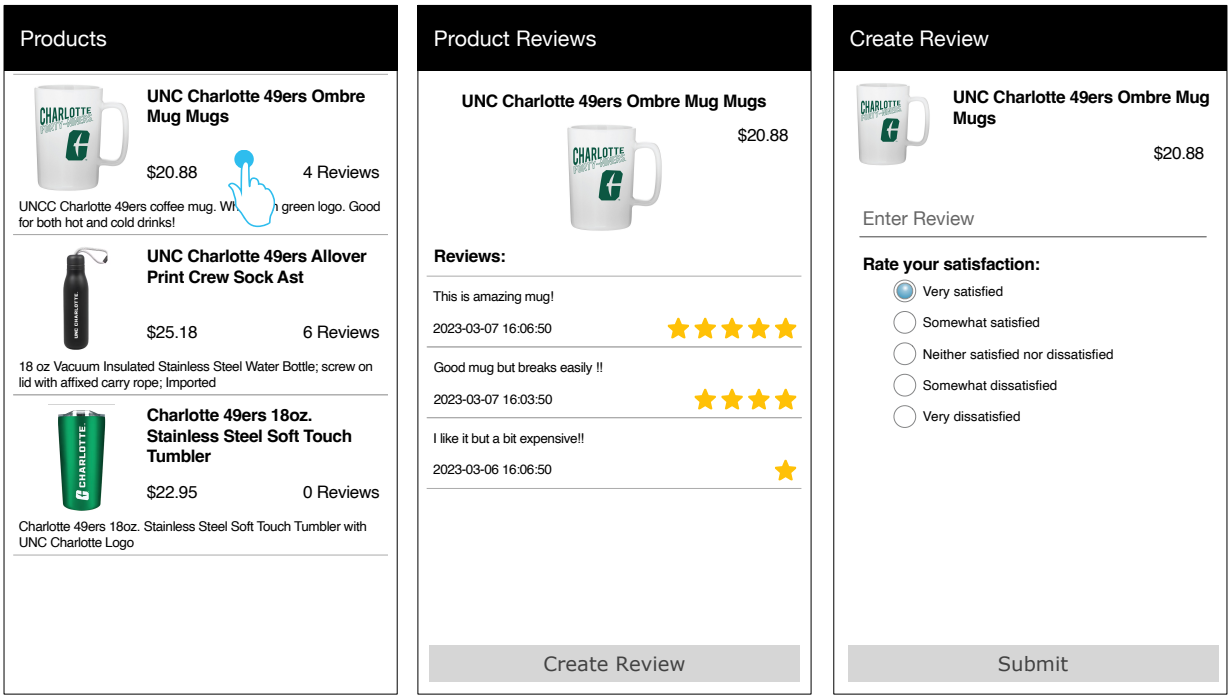
Basic Instructions:

1. This is the Midterm Exam, which will count for 15% of the total course grade.
2. This Midterm is an individual effort. Each student is responsible for her/his own Midterm and its submission.
3. Once you have picked up the exam, you may not discuss it in any way with anyone until the exam period is over.
4. During the exam, you are allowed to use the course videos, slides, and your code from previous home works and in class assignments. You can use the internet to search for answers. You are NOT allowed to use code provided by other students or solicit help from other online persons.
5. Answer all the exam parts, all the parts are required.
6. During the midterm the teaching assistants and Instructors will pass by each student and ask them to demonstrate their application. Your interaction with the teaching assistants and instructors will be taken into consideration when grading your exam submission.
7. Please download the support files provided with the Midterm and use them when implementing your project.
8. Your assignment will be graded for functional requirements and efficiency of your submitted solution. You will loose points if your code is not efficient, does unnecessary processing or blocks the UI thread.
9. Create a zip file which includes all the project folder, any required libraries, and your presentation material. Submit the exported file using the provided canvas submission link.
10. **Do not try to use any Social Messenger apps, Emails, Or Cloud File Storage services in this exam.**
11. **Failure to follow the above instructions will result in point deductions.**
12. **Any violation of the rules regarding consultation with others will not be tolerated and will result disciplinary action and failing the course.**

Midterm Exam (100 Points)

In this assignment you will implement a product reviews application.

1. You are provided with a skeleton application file, which you should use for this app.
2. You are provided with a PostMan File that includes the APIs required for this app.
3. Use the OkHttp library in this app in order to make all the API calls. All the data returned by the APIs is in JSON format.
4. All the network calls should be done in a background thread. All UI changes, updates and edits should be performed on the main thread.
5. This app will have one Activity and 3 Fragments, **all communication between fragments should be managed by the activity.**



(a) Products Fragment (b) Reviews Fragment (c) Create Review Fragment

Figure 1, App Wireframe

Part 1 : Products Fragment (35 Points)

This screen displays the list of products (see Fig 1(a)). The requirements are as follows:

1. When the fragment loads the fragment should call the **/api/products** api to retrieve the list of products:
   a. Use OkHttp library to get the list of products. Use the provided Product class to parse each product and add it to the ArrayList of products.
   b. Upon completing the parsing of the products, the ListView should show each product item as shown in Fig 1(a).
   c. Use the Picasso library to display the product image.
2. Clicking on a product row item should communicate with the Main Activity to:
   a. Replace the current fragment with the Reviews fragment.
   b. Send the selected product object to the Reviews fragment.
   c. Add the current fragment to the back stack.

Part 2 : Reviews Fragment (40 Points)

This screen displays the list of reviews for the selected product (see Fig 1(b)). The requirements are as follows:

1. At the top of the fragment display the product name, product price and the product image as shown in Fig 1(b).
    a. Use the Picasso library to display the product image.
2. When the fragment loads (onViewCreated) call the **/api/product/review/{pid}** api to retrieve the list of reviews for the selected product:
    a. Use OkHttp library to get the list of reviews. Use the provided Review class to parse each review and add it to the ArrayList of reviews.
    b. Upon completing the parsing of the reviews, the ListView should show each review item as shown in Fig 1(b).
    c. For each review the api returns “rating” value (1 to 5) which indicates the number of stars. You are provided with drawable files for each star rating. For example to show ***5 stars*** use ***stars_5*** drawable resource.
3. Clicking on the “Create Review” button should communicate with the Main Activity to:
    a. Replace the current fragment with the Create Review fragment.
    b. Send the product object to the Create Review fragment.
    c. Add the current fragment to the back stack.

Part 3 : Create Review Fragment (25 Points)

This allows the user to create a review (see Fig 1(c)). The requirements are as follows:

1. At the top of the fragment display the product name, product price and the product image as shown in Fig 1(c).
    a. Use the Picasso library to display the product image.
2. Clicking on the “Submit” button should:
    a. If the review text is not entered then show a toast indicating it is required.
    b. If the all the fields are entered correctly then call the **/api/product/review** API.
        i. This api requires the pid, review and the rating. The pid is the product ID, review is the text entered for the review, and the rating is an integer from 1 to 5 representing the selected rating (where Very satisfied is 5, and Very dissatisfied is 1).
        ii. If the request is successful: then contact the Main Activity which should pop the back stack to go back to the Reviews fragment. Which should reload the list of reviews and show the newly added review.
        iii. If the request is not successful: then display a toast message summarizing the error.

Student Name:	
Student ID:	

P#	Features	Total	Grade	Comments
P1	**Products Fragment:** Connect to the api, parse JSON response and put each product in the ArrayList	15		
P1	**Products Fragment:** Created a custom ArrayAdapter and display the products correctly.	15		
P1	**Products Fragment:** Clicking product row item: - Sends the selected product to the Reviews fragment - Pushes the current fragment back stack. **Interface Implementation REQUIRED**	5		
P2	**Reviews Fragment:** Display the product name, price and image correctly.	5		
P2	**Reviews Fragment:** Connect to the api, parse JSON response and put each review in the ArrayList	15		
P2	**Reviews Fragment:** Created a custom ArrayAdapter and display the reviews correctly.	15		
P2	**Reviews Fragment:** Clicking “Create Review” button: - Sends the product to the Create Review fragment - Pushes the current fragment back stack. **Interface Implementation REQUIRED**	5		
P3	**Create Review Fragment:** Display the product name, price and image correctly.	5		
P3	**Create Review Fragment:** Clicking “Submit” button the validation done correctly.	5		
P3	**Create Review Fragment:** Clicking “Submit” button: - Connects to the api and upon successful return contacts the Main Activity to pop the back stack. **Interface Implementation REQUIRED**	15		
Total		**100**		

Table 1: Grading Key