ITIS/ITCS 4180/5180 Mobile Application Development
Midterm Makeup - Preparation

Basic Instructions:

1. This is the Midterm makeup preparation assignment. You are required to work on this assignment before the midterm makeup as the midterm makeup exam will be heavily based on this assignment. This assignment is to prepare you for the midterm makeup.
2. This is completely an individual effort. Each student is responsible for her/his own implementation. You should have a clear understanding of the app details, as the midterm makeup will be based on this assignment.
3. The midterm makeup grades will not be based on this assignment, instead it will be based on other specific functionalities that will be added to this assignment. So please spend the time to work on this assignment and to understand your implementation.
4. Make sure to create your own parse.com account to be used for this application.

Midterm Makeup - Preparation

In this assignment you will make HTTP requests and parse JSON data. Using the 500px API you will retrieve list of photos in JSON format and display them on your own app. The app is composed of three activities, namely **MainActivity**, **GalleryActivity** and **DetailsActivity**.

Initial Setup

1. Go to https://500px.com/login?r=%2Fflow and create a new account.
2. After creating an account, you will receive an email from 500px you need to confirm your account by clicking on the link in the received email.
3. Click on your profile and select Settings, see Figure 1.
4. Click on Applications, then click on the *"**register your application**"* at the bottom of the page, see Figure 2.
   a. Name your application "MidtermMakeup" and provide a short description.
   b. Set the application url to www.uncc.edu/~your-uncc-id
   c. Provide your email as the developer email.
5. Then the consumer key will be generated as in Figure 3. Copy the application "Consumer Key", so that you can use it later to search for photos.

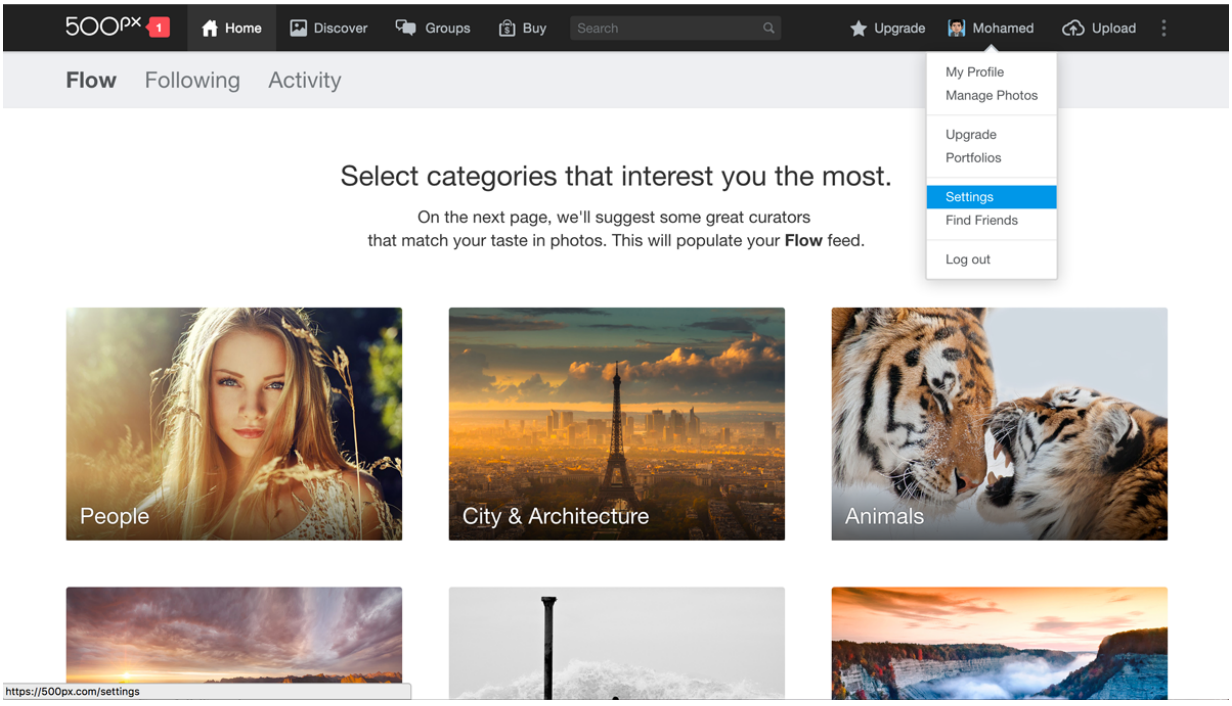


Figure 1, Settings of your account

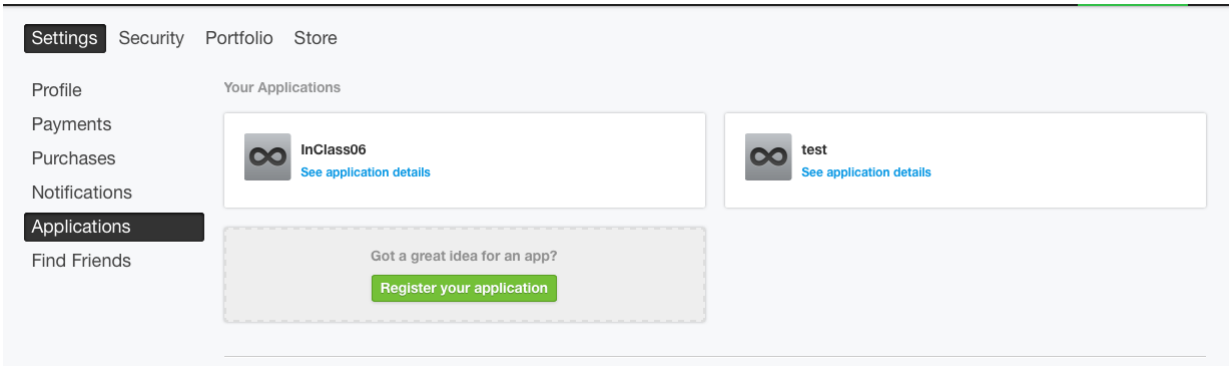


Figure 2, Applications

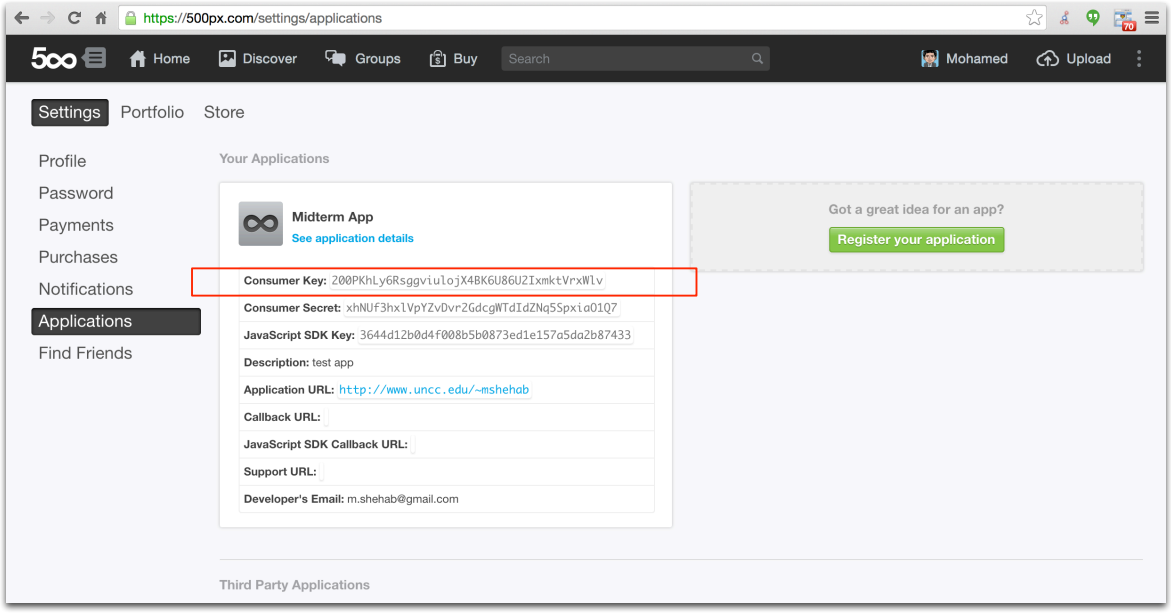
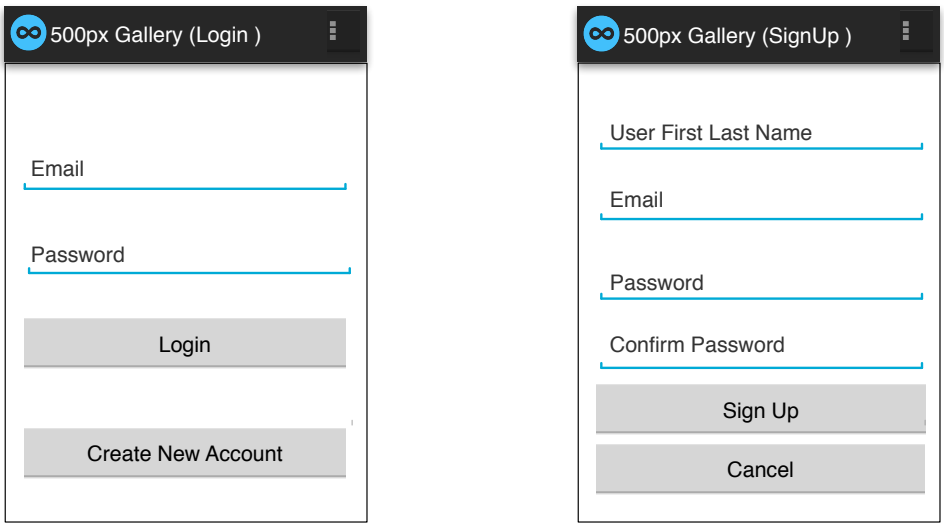


Figure 3, Copy Consumer Key

User Signup and Login

Your app should implement both login and signup functions. You should use parse.com to store the user first name, last name, email address and password in the Parse User class. The requirements are as follows:

1. The launcher activity should be set to the Login activity. When the app first starts, the Login activity should check if there is a current user session, by using the parse provided methods to check if there is a valid current user:
   a) If there is a current valid user, then start the Main activity, and finish the Login activity.
   b) If there is no current valid user, then the Login activity should be used to provide user login.



(a) Login (b) Sign Up

Figure 4, User SignUp and LogIn Activities

2. Create a Login activity (Figure 4(a)):
   a) The user should provide their email and password.
      - Check the user input, if the email or password field is left empty, print a toast message indicating that these fields are required and don't submit the login information to parse.com.
   b) The provided credentials should be used to authenticate the user using parse.com. Clicking the "Login" button should submit the login information to parse.com to verify the user's credentials.
      - If the user is successfully logged in then start the Main activity, and finish the Login activity.
      - If the user is not successfully logged in, then show a toast message indicating that the login was not successful.
   c) Clicking the "Create New Account" button should start the Signup activity and finish the login activity.
3. Create a Signup activity (Figure 4(b)):
   a) Clicking the "Cancel" button should finish the Signup activity and start the Login activity.
   b) The user should provide their first name, last name, email and password. The provided credentials should be stored in the User class in parse.com. Clicking the "Sign Up" button should submit the user's information to parse.com to verify the user's credentials.
      - Check the user input, if any of the fields is left empty, or if the password and confirm password do not match, print a toast message indicating the corresponding error message and don't submit the provided information to parse.com.
      - If an account with the same email already exists, display an error message indicating that the account account was not created and the user should select a different email.

- If an account with the provided credentials does not already exist, then store the new account information and display a Toast indicating that the user has successfully login. Then start the Main activity and finish the Signup activity.
- Note that, the username and email should set to the same value in the user's table.

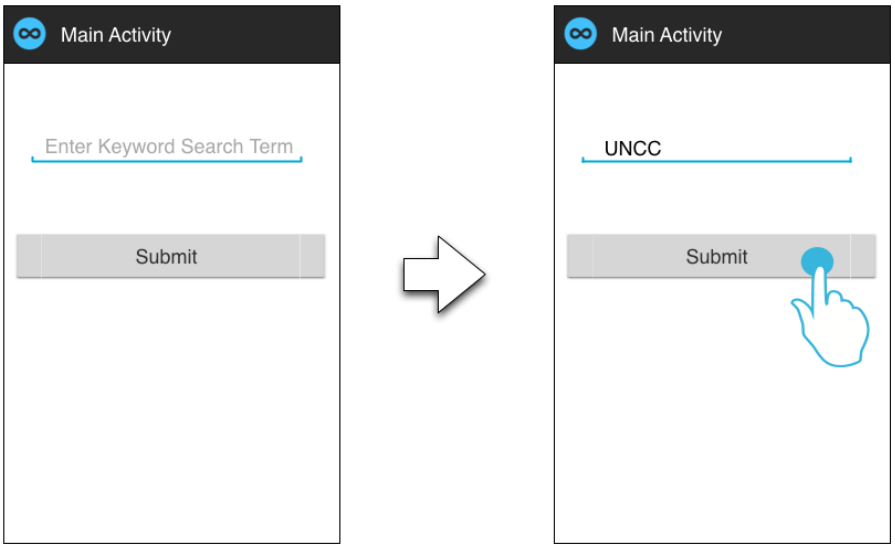


Figure 5. The Main Activity

MainActivity

The Main activity displays a EditText that enables the user to enter a keyword search term. Figure 4 shows the Main activity. The requirements include:

5. Upon selecting a keyword from the list, the EditText text should be set to the selected keyword, See Figure 5.
6. Upon clicking the submit button start the Gallery Activity. The intent should include the search term using extras. You should not use a static variable to share data between activities.
7. If the submit button is clicked and the EditText is empty display a Toast Message indicating that the field was empty and do not start the next activity.

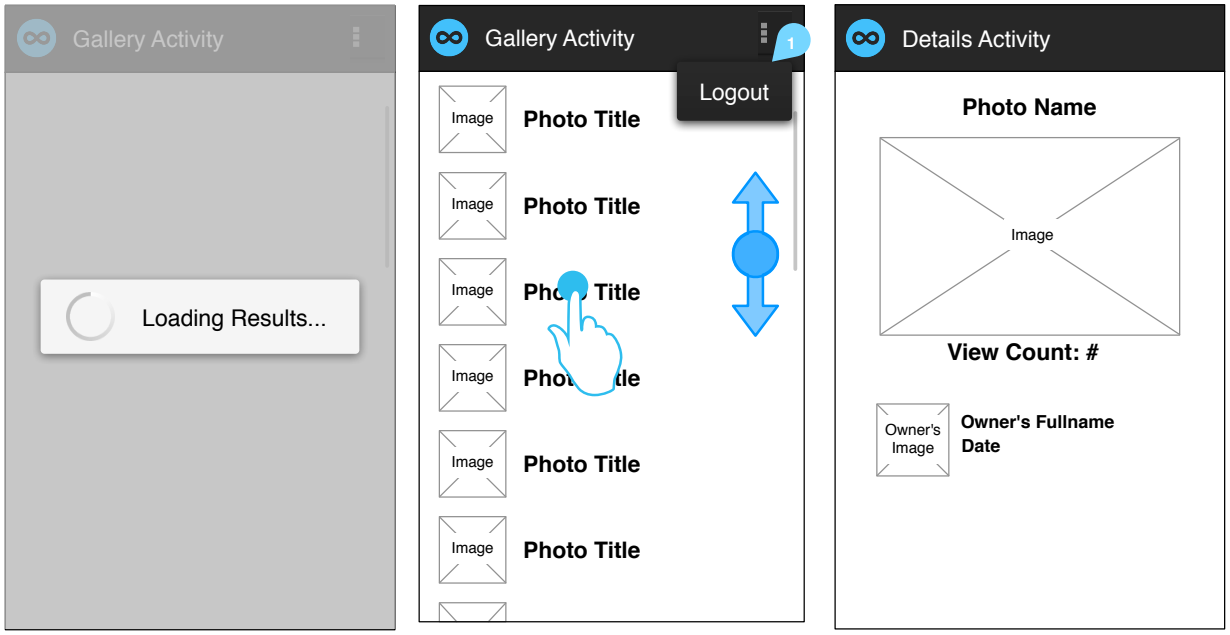
Gallery Activity

The Gallery activity is responsible for the retrieving the photo search results based on the search keyword term sent from the Main Activity. The Gallery Activity wireframe is shown in Figure 6. The requirements include:

1. The 500px photos/search api should be used to retrieve the photos matching the requested keyword term:
    - Use the api documentation provided in https://github.com/500px/api-documentation/blob/master/endpoints/photo/GET_photos_search.md
    - You should set the "**consumer_key**" parameter based on your 500px app's consumer key.
    - You should set the "**term**" parameter to the search term keyword.

- You should set the "**image_size**" parameter to 4.
- You should set the "**rpp**" parameter to 50.
- Use the api endpoint https://api.500px.com/v1/photos/search?

2. Send a GET request and retrieve the photo search results for the selected keyword term. All the parsing and HTTP connections should be performed by a worker thread or an AsyncTask and should not be performed by the main thread. While, the JSON is being downloaded and parsed you should display a progress dialog see Figure 6.
3. Create a photo class that stores the photo title, photo url, photo view count, owner's name and owners photo. The JSON parser should return a list of photo objects.
4. The progress dialog should be dismissed after the parsing is completed, and the ListView or RecyclerView should be setup and displayed, as shown in Figure 6(b). We recommend you use a RecyclerView.
5. Clicking an item should start the Details Activity. The intent should include the required photo information to be sent to the Details activity using extras. You should not use a static variable to share data between activities.
6. Clicking the logout menu item should logout the user, end the activity and show the login activity.



(a) Gallery Progress (b) Gallery Activity (c) Details Activity

Figure 6. The Gallery + Details Activity Wireframe

DetailsActivity

This activity should display the photo details, which includes the photo title, owner name and the photo itself. The DetailsActivity should receive the photo information from the GalleryActivity. Image Download should be done in a child thread. Figure 6(c) shows the DetailsActivity. The implementation requirements include:

1. Use a thread pool (or AsyncTask) to retrieve the selected photo. If there is no photo image, display a default image of your choice to indicate no photo found.
2. Pressing back should end this activity and go back to the Gallery activity.