**Activity: Registration**

This activity allows the user to create a new user, using their name, email address, and password.

This Activity is set up to use multiple layouts, or pages, that can be accessed by swiping left or right, or by clicking the “next” button. Originally there were three pages, one for the user’s account, another for their contact information, and lastly a page for the user to create a new vehicle. However, contact information and vehicle information cannot be accessed unless the user has been verified AFTER registration. So while the Activity is still set-up to use multiple pages, the other pages have been commented out, and the user can only see the page associated with the ‘fragment\_user\_registration\_account.xml’ layout.

**Manifest Details**

This activity is implemented with UserRegistration.java. Like most of the Activities, its screenOrientation is restricted to “portrait.”

The Activity uses the theme”@style/AppTheme.NoActionBar”, which can be found in the res/values/styles.xml.

The Intent filter provides standard access when accessed by name.

**Layout Files**

activity\_user\_registration.xml

* The Registration Activity’s main file. It exists to display the account, contact, and vehicle Fragments, and to provide navigation between them with a button that changes functionality depending on the current layout. It also contains a link to the app’s Terms and Conditions Activity
* container (ViewPager) – Displays the current Fragment
* nextButton(Button) – Contains a default onClick method, but functionality changes in the UserRegistration.java file
  + onClick: nextButtonClick -- Default functionality changes the Fragment displayed in the container.
* termsTextView(TextView):
  + onClick: openTerms – Opens the Terms and Conditions Activity

fragment\_user\_registration\_account.xml

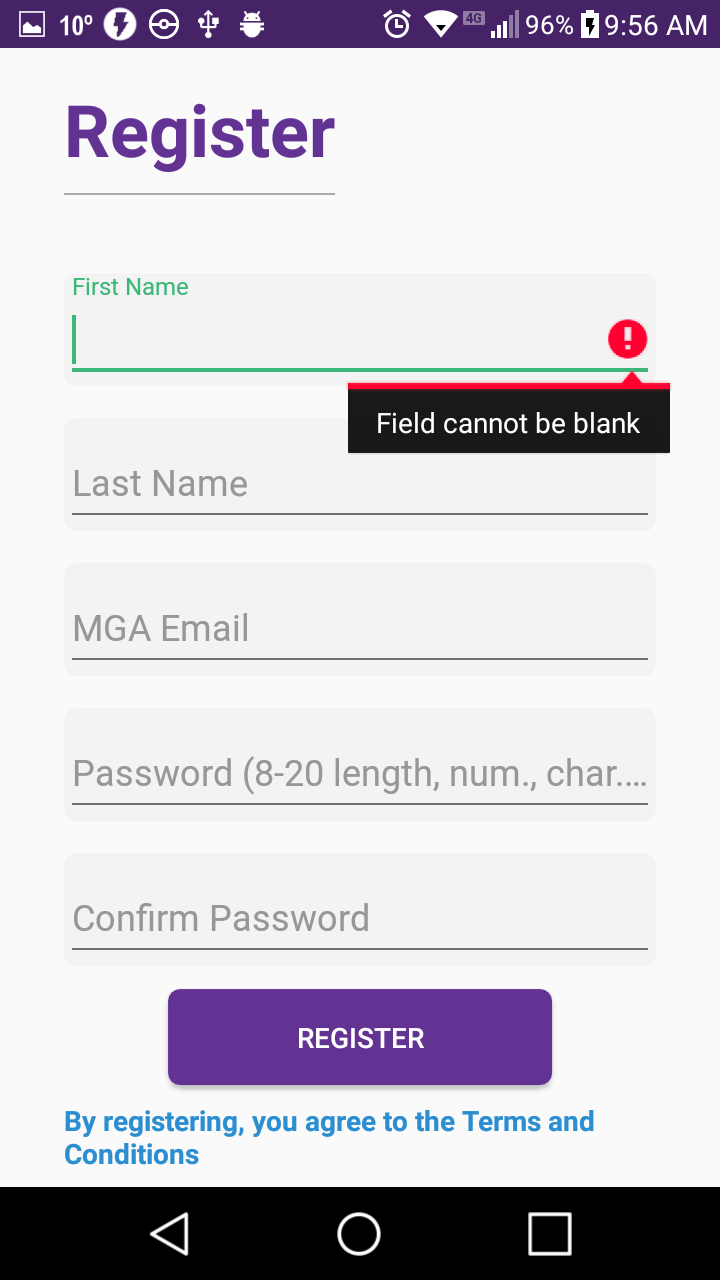
* Currently the only visible Fragment in the Registration Activity, as of writing this. This contains TextInputLayouts for a new user’s first name, last name, email address, and password. These values are pulled into UserRegistration.java

fragment\_user\_registration\_contact.xml

* Unused, because the user currently cannot add profile information unless they’ve confirmed their email address. This contains three fields for the user’s address, zip code, and phone number.

fragment\_user\_registration\_vehicle.xml

* Unused for the same reason as the contact layout. Users cannot create a vehicle for an unverified account. This contains fields to include a car’s make, model, year of manufacture, the vehicle’s nick name, and the seat capacity.



**Class File**

UserRegistration.java implements the Registration Activity, but it also includes internal classes to implement the Activity’s Fragments, and one additional class for the ViewPager in activity\_user\_registration.xml

Variables:

* -mSectionsPagerAdapter: SectionsPagerAdapter – holds a references to the SectionsPagerAdapter internal class, which displays Fragments in order.
* -mViewPager: ViewPager – The ViewPager in activity\_user\_registration.xml
* +PAGE\_COUNT: int – Tracks how many pages the ViewPager has available. This is used by the SectionPagerAdapter class to know if it is on the last page.
* -accountFragment: AccountFragment – References fragment\_user\_registration\_account.xml
* -contactFragment: ContactFragment – UNUSED. References fragment\_user\_registration\_contact.xml
* -vehicleFragment: VehicleFragment – UNUSED. References fragment\_user\_registration\_vehicle.xml
* -includeVehicle: boolean – UNUSED. Allowed the user to call the registration method if they left the vehicle information blank.
* -queue: RequestQueue – Holds JSON requests

Methods and Internal Classes

* \*onCreate(savedInstanceState): void
  + Sets up the User Registration Activity. This tells the Pager Adapter to display the first Fragment (in this case, the user’s account information) and to display it on the ViewPager.
  + It also initializes the RequestQueue to send JSON Object Requests.
  + It also modifies the functionality of nextButton.
    - nextButton’s text is changed from “Next” to “Register”
    - The onClick event also calls the register() method.
* +onBackPressed(): void
  + Lets the user return to previous Fragments in the ViewPager using Android’s Back Button.
  + Also restores the nextButton’s functionality if it had changed.
* +openTerms(view): void
  + Opens the Terms and Conditions Activity
* +nextButtonClick(view): void ---UNUSED---
  + Allowed the user to change the current Fragment on the ViewPager.
  + This would also change the nextButton’s functionality to start the register() method if the user was on the last page
  + It would also show a clickable TextView to start the register() method WITHOUT any vehicle information
  + It would also hide the link to the app’s Terms and Conditions until final page was loaded for the first time.
* -register(): void
  + Creates Hashmaps to store information from the Account, Contact, and Vehicle classes by calling their “getValues()” methods. Afterward, createUser() is called, passing the values as paramaters
  + Would also call createCar() immediately after creating a user, but this has been commented out.
* -createUser(values): void
  + Sends a JSON Object POST Request to the server to make an account, passing the values returned in register() as the parameters.
  + If account creation is successful, the Registration Activity closes itself, otherwise, errors are shown.
* -createCar(values): void ---UNUSED---
  + Sends a request to the server to create a new vehicle for the newly created user.
* -getUserId(username, password): String ---UNUSED---
  + Log in with the newly created user to retrieve the account ID, and use the ID when accessing the createCar API
* +AccountFragment
  + Class for the account’s login information
  + Variables
    - -firstNameInput: TextInputEditText
    - -lastNameInput: TextInputEditText
    - -emailIput: TextInputEditText
    - -passwordInput: TextInputEditText
    - -passwordConfirmInput: TextInputEditTExt
  + Method
    - +onCreateView(inflater, container, savedInstanceState): View
      * Inflates the fragment\_user\_registration\_account layout and initializes the TextInputEditText variables
    - +getAccountValues(): HashMap<String, String>
      * Retrieves the values stored in the TextInputEditTexts and returns them in a HashMap.
* +ContactFragment
  + Class for contact information
  + Variables
    - -addressInput: TextInputEditText
    - -zipInput: TextInputEditText
    - -phoneInput: TextInputEditText
  + Methods
    - +onCreateView(inflater, container, savedInstanceState): View
      * inflates the fragment\_user\_registration\_contact layout and initializes the TextInputEditText variables
    - +getContactValues(): HashMap<String, String>
      * Retrieves contact information from the input fields and returns them in a HashMap
* +VehicleFragment
  + Class for vehicle information. This class is used depending on if the user wanted to add a new Vehicle or not.
  + Variables
    - -makeInput: TextInputEditText
    - -modelInput: TextInputEditText
    - -yearInput: TextInputEditText
    - -vehicleNameInput: TextInputEditText
    - -seatsInput: TextInputEditText
  + Methods
    - +onCreateView(inflater, container, savedInstanceState): View
      * Inflates the fragment\_user\_registration\_vehicle layout and initializes the TextInputEditTexts
    - +getCarValues(): HashMap<String,String>
      * Retrieves vehicle information and stores it in a HashMap
* +SectionsPagerAdapter
  + Tracks all the available pages the ViewPager can swap between.
  + +SectionsPagerAdapter(fm)
    - Constructor
  + +getItem(position): Fragment
    - Returns and display a Fragment depending on the value of position
  + +getCount(): int
    - Returns the number of pages.