How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

GitHub Username: Patrick-Wallin

College Information

Description

College Information finds college and provide information such as cost, aid, admission, location, and link to its website based on search input by users.

The app interface will meet the Material Design principles that makes a lot easier for users for searching and reading information in clear and loud. The search and sort will be flexible. Users will have an opportunity to search at least one or more in categories to narrow down the search results. The sort will be cost (low to high), location (closest to further), and aid (highest to lowest). The information could be stored into local database when users like one of the colleges in list of search result.

Intended User

Parents and students

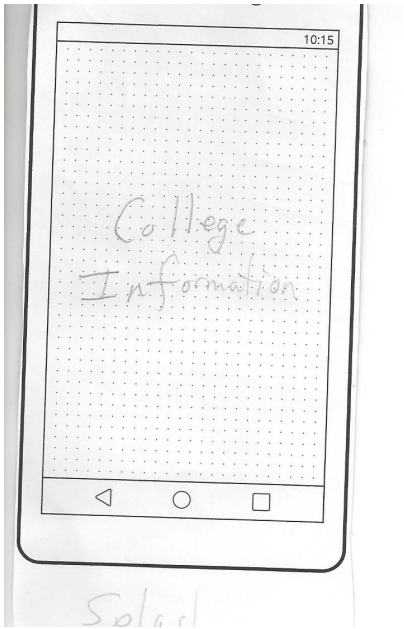
Features

List the main features of your app. For example:

- Search for college based on degrees, programs, location, or name
- Display search results
- “Like” or “Not Like” (Using Thumb-up image instead of heart)
- “Like” college would store information into local database
- Interface meets Material Design principles
- Display detail information including cost, aid, admission, website, and location on map

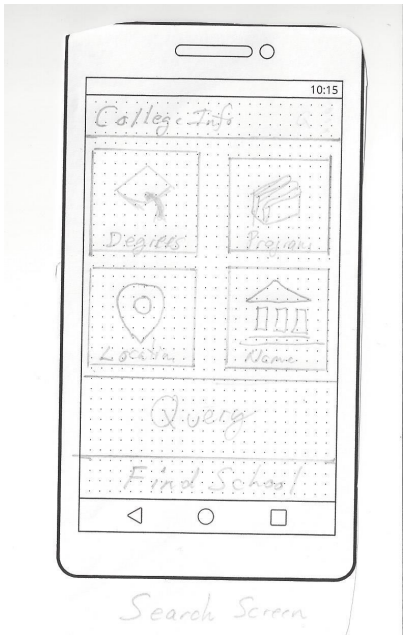
User Interface Mocks - Phone

Screen 1



Splash screen appears when user taps on the app icon. It might be showing a picture of college's building or a person with diploma hat with the word "College Information" on the screen (I just found out that it wasn't worth to have splash screen since it would waste time on users). I will not include this screen for this app.

Screen 2

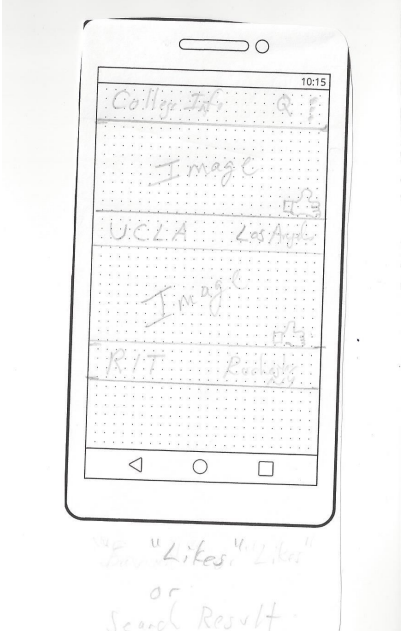


Search screen will appear after splash if there is no “like” record in local database. Otherwise, it will show “Screen 3” below. Screen screen will show four pictures or buttons (probably using card view) along with list view below it (one row in list view). The “query” indicates what user has chosen the search options or selections. “Find School” is a button or in one row on list view. When user taps on it, it will use query and bring the result on “Screen 3” below.
05/29/2017 - Updated - Let use floating button instead of big button of “Find School”. We can use search icon on the floating button and put it on the bottom right corner.

Example: I tap on “Degrees” and it will open “Screen 4” below and I select one of the degrees such as “Bachelors” and then go back to this screen and tap on “Programs” to choose “Mathematics” and go back to this screen and I would see the “query” screen as the following:
Degrees: Bachelors
Programs: Mathematics

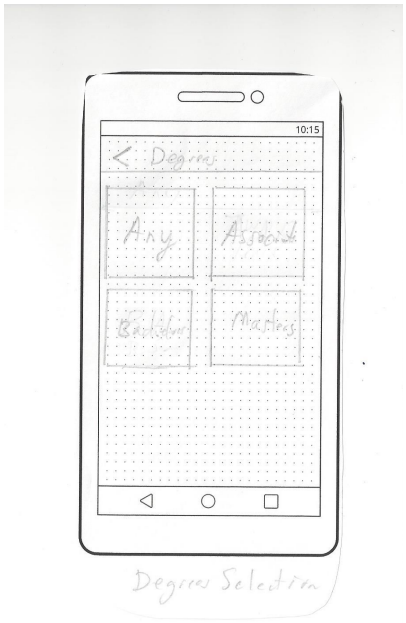
This app will store the search input into local database so in future, when they go to search screen, it will show the query.

Screen 3



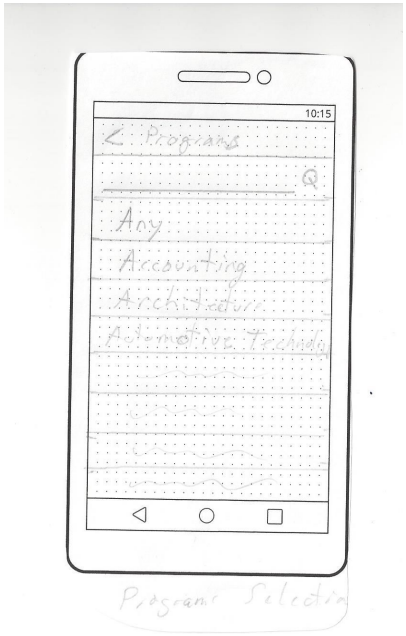
This will appear after user taps on “Find School” or go to “Like” (using menu on action bar). The screen will include image of college along with college name and location. Each item will have an event of tap for going to detail page.

Screen 4



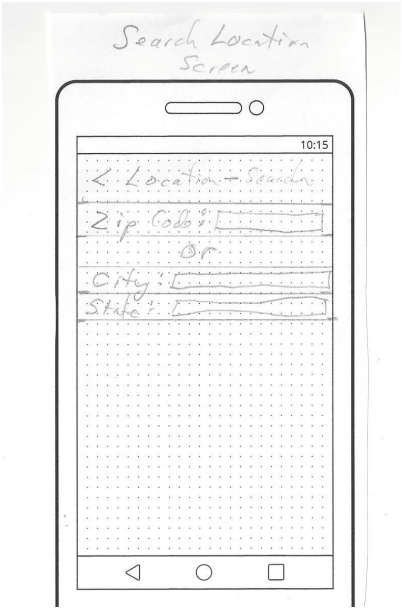
This is a screen of “Degrees” selection for search.

Screen 5



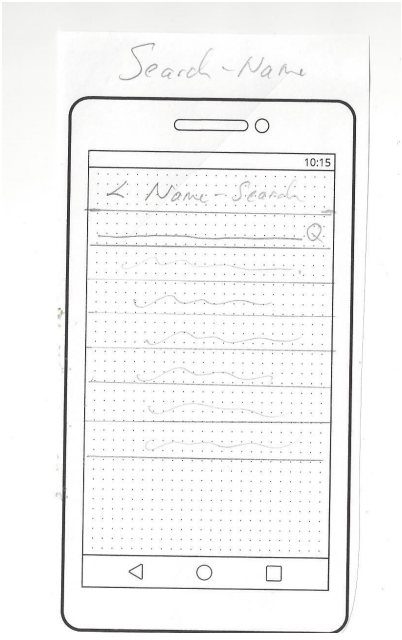
“Programs” selection screen will populate listing of programs with search input that user can enter to find program that he/she wants to be included in search query.

Screen 6



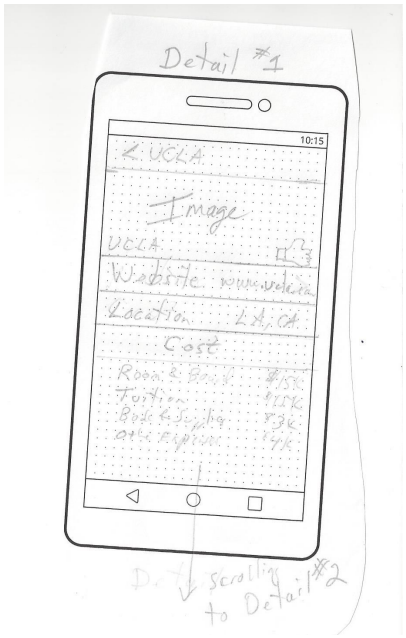
The “location” search screen will have three input fields which are zip code, city or state.

Screen 7



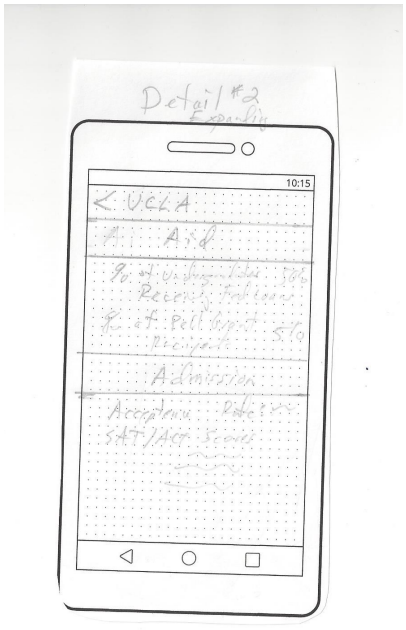
The “Name” search screen will allow them to find name based on search input.

Screen 8



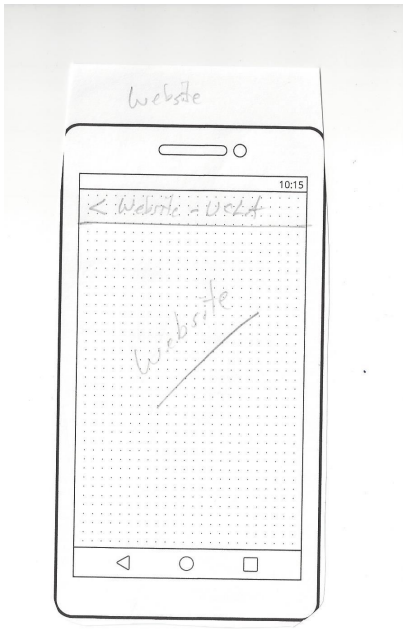
This screen will appear from screen #3 after user taps on one of the items on it. It will show image with location, name, website, cost, aid, and admission (it will appear on next screen when you scroll down on this screen).

Screen 9



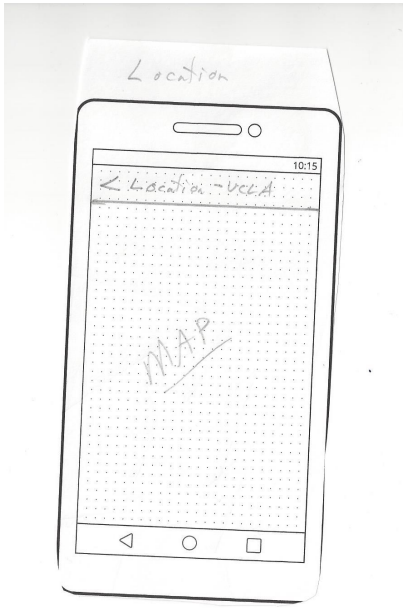
This is the part of detail page and it will appear when you scroll down from screen #8 and the image will disappear.

Screen 10



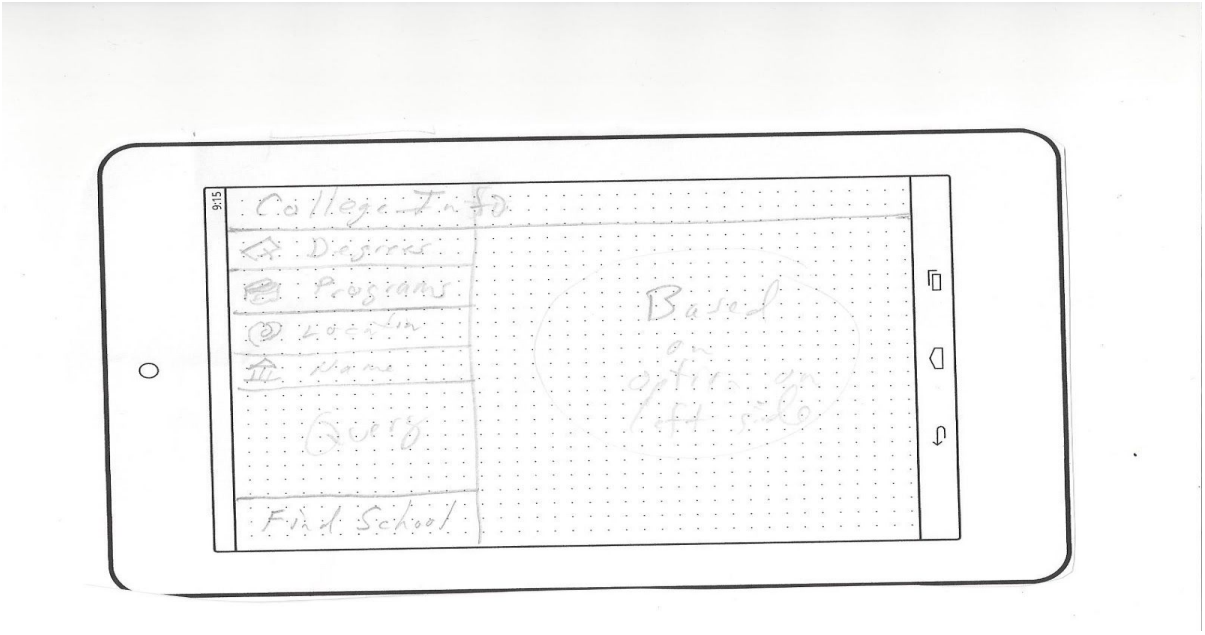
When user taps on “website” on detail page, it will open the website of college on this screen.
05/29/2017 - Updated - Let use default browser so user can keep it open while using this app.

Screen 11



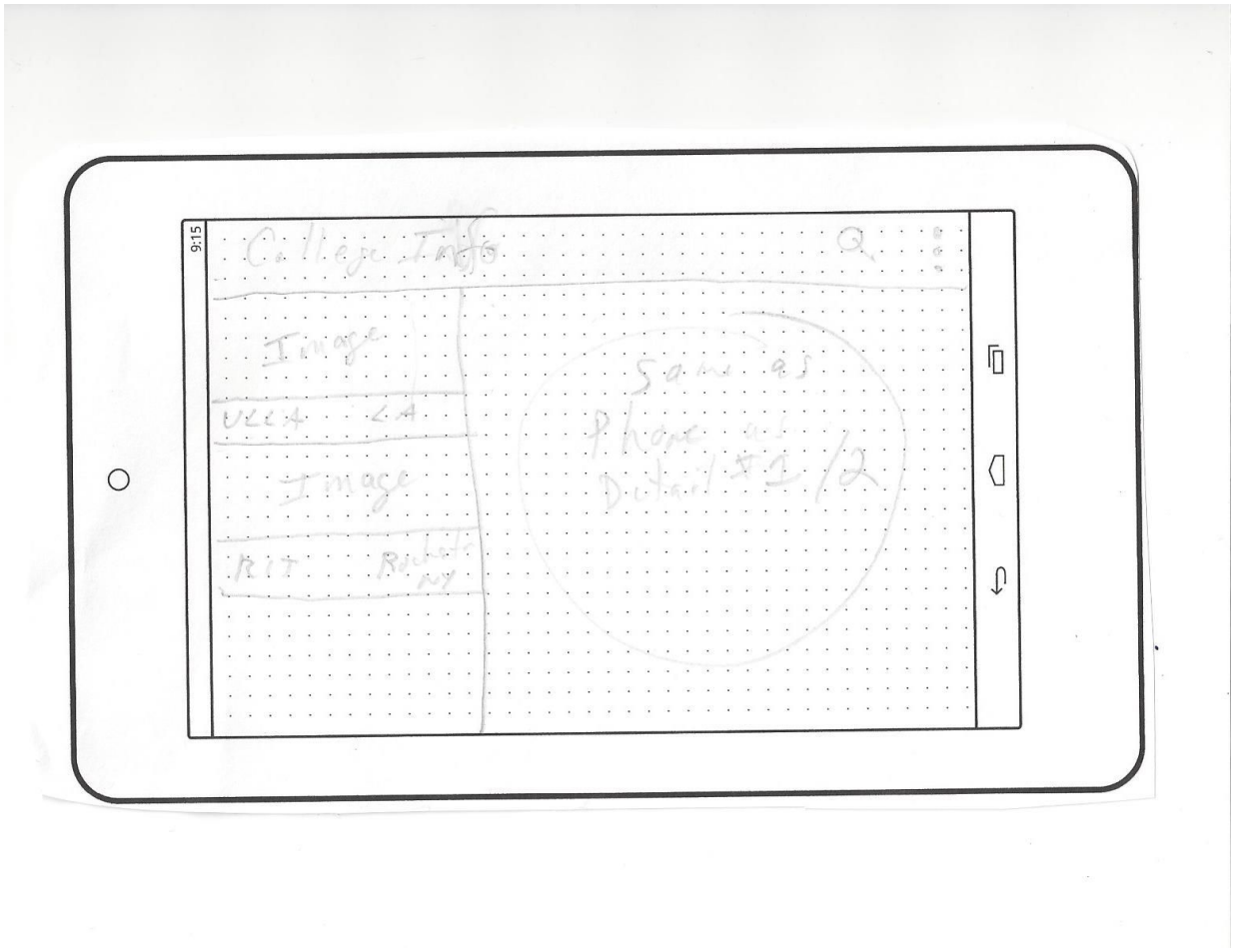
When users tap on “location” on detail page, it will open google map. Let launch default map instead of inside this app.

Screen 12



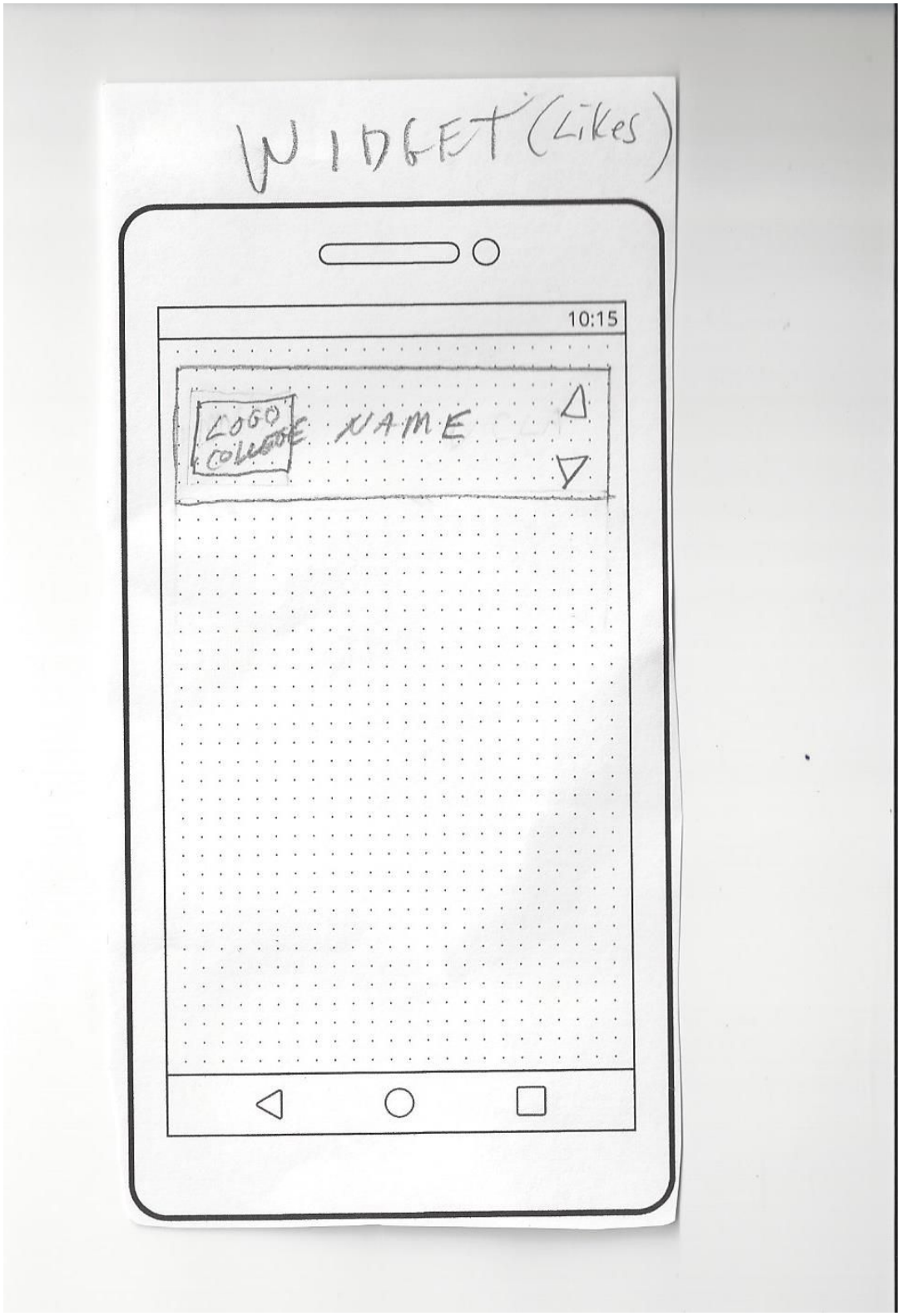
When user launches the app on tablet in landscape (portrait will be same as phone) and it has no “like” record, then it will show search screen like above. When user taps on the left side, it will load the activity page based on the option. Example: if user taps on “Degrees”, it will load “Degrees” page as screen #4 into right side.

Screen 13



When user has some “like” records or taps on “Find school”, it will bring up this screen that will show list of college on left side while showing detail page on right side. The detail page is same as screen#8 and #9.

Screen 14 - Widget


WIDGET (Likes)
10:15
LOGO
NAME

The widget will contain small logo of college (if we could otherwise, no image) with college name and the arrows on right side. The arrows will be like scrolling to go to next list of college that user had chosen as “likes” via the app.

Key Considerations

How will your app handle data persistence?

The app will store college information into local database when user taps on “like” thumb-up image on one of the colleges. Also, we will store the search query into local database as well.

Describe any corner cases in the UX.

The app will be simple and easier to use that some activity will automatically go back to the previous page for you. Example: if you tap on “Degrees” button, it goes to “Degrees” page to choose one of four “degrees” (any, associate, bachelors, or master). When you tap on it, it will scroll back to “Search Query” screen.

Describe any libraries you’ll be using and share your reasoning for including them.

I will be using one of the following libraries:

- Picasso to handle the loading and caching of images
- ButterKnife to handle all the view items since there will be a lot of them in this app.
- Snackbar to show nice error message or show a warning message such as no search result.
- Fast Android Networking to load the JSON from data.gov to receive data of college information. I haven’t decided whether to use this library or not. I will research some more whether I should use this library or others.
- SparkButton to show nice flashing on floating button.
- Parceler to handle the parcelable in easy way. According to it, it would reduce code by not having including read/write in order.
- Timber to show the message through debugging that will help me to know what class and method are coming from.

Describe how you will implement Google Play Services.

I will use Google Maps API to display the location of college on the app.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

General Feature

- Provide widget to provide basic information to the users on the home screen
- Develop database that may contain a few tables such as information about college that user likes, list of majors, list of college names, and search query information. List Of majors are rarely being changed. It is best to leave it on local table so app does not need to download it all the time.
- Implements a ContentProvider to access into the data on local database.
- Use Loader to move data into views
- Use RecyclerViewHolder which is easy to move data into items during RecycleView when users scroll it to save performance or battery
- All variables that contain information (hard coded) should be in strings.xml or dimen.xml file
- Use Google Maps API via Google Play Service
- Make smooth transition between search result and detail activity pages.
- Uses an app bar and toolbars
- Make image on top on detail page disappear while scrolling up
- Pull data from www.data.gov's API when search is performed
- Pull data from www.data.gov's API for populating list of majors and college names when user gets into the "search selection" page (major or name).

Task 1: Project Setup

Write out the steps you will take to setup and/or configure this project. See previous implementation guides for an example.

- Sign up for data.gov API to receive data, www.data.gov
- Develop small java program in netbeans to test receiving the JSON from www.data.gov and see what it looks like and see if it works based on input values into API to get correct data.
- Create Project in Android Studio. The project would be called CollegeInformation.
- Set up Picasso by adding the link of Picasso to dependencies block.
- Set up Butterknife by adding the link of Butterknife to dependencies block.

- Set up Fast Android Networking by adding the link of Fast Android Networking to dependencies block.
- Set up SparkButton by adding the link of Spark Button for “thumb up” floating button.
- Set up Parceler by adding the link of Parceler which is easier to use than Parcelable since it would not require us to write a lot of code in it as in order to add code in both read/write methods.
- Set up Timber (Logger) by adding the link of Timber that will help us to trace any issue via debugger.

Task 2: Your Next Task

- Create class for handling the Fast Android Networking to receive JSON
- Create class for converting JSON into Array.
- Create Parcelable class that use data that will be easy to pass it between activities.
- Create AsyncTask to receive data from www.data.gov smoothly.

Task 2: Implement UI for Search Screen

- Create layout and AppCompatActivity class for Main page
- Create “Search Screen” layout and AppCompatActivity class for search screen.
- Create CardView on that “Search Screen” page. CardView would have four items.
    - Degrees
    - Programs
    - Location
    - Name
- It would be in two columns and two rows.
- Create Listview on “Search Screen” page below cardview. It would have “query” list.
- Create floating button on the bottom of the screen using search icon.
- In Main Activity page, load “Search Screen” at launch to see if the screen looks okay for now. We will get into the part whether we load list of college or search screen later.

Task 3: Implement UI for Degrees Selection Screen

- Create layout and AppCompatActivity for Degrees Selection
- Create card view on that page that contains four items: Any, Associate, Bachelors, and Masters.
- Go to “Search Screen” activity and add event of “click” on “Degrees” button.
- When the click’s event has been fired, go to “Degrees Selection” screen.
- Add click’s event on all buttons on “Degrees Selection” screen.

- When the click's event has been fired, store the value: "Any", "Associate", "Bachelors" or "Masters". Use const value for each of them and apply number to it. When it has been stored, go back to the page of "Search" screen and populate the value into "Query" list on "Search" screen.

Task 4: Implement UI for Programs Selection Screen

- Create layout and AppCompatActivity for Programs Selection
- Create Listview using recyclerview with search on top of it.
- Create table of listing of majors if table does not exist (sqliteopenhelper to create table called tblMajors)
- If table does not exist, load list of majors from www.data.gov (if they do not offer that, I will research and find another API).
- Create Adapter and RecyclerViewHolder classes.
- Load data from the table into listview (recyclerview).
- Allow search input to filter the listview while user types. Example: if user types Ant, then list view will be filtered based on that search input. (except keeping "Any" on top of listview).
- When users tap on one of the selection, it stores the value of major and go back to search screen and populate it into "query" listview.

Task 5: Implement UI for Search Location Screen

- Create linear layout and AppCompatActivity for "Search Location" screen.
- Create label and textfield for "Zip Code"
- Create "Or" on next row after "Zip Code"
- Create label and textfield for "City" after "Or"
- Create label and textfield for "State" after "City"
- Create three buttons after "State" and buttons would be labeled as "OK" , "Clear", and "Cancel"
- When the "Cancel" button has been pressed, go back to "Search" screen and do nothing.
- When "Ok" button has been pressed, store the values and then go back to "Search" screen and populate it in "Query" listview.
- When "Clear" button has been pressed, clear the input values.

Task 6: Implement UI for Search Name Screen

- Create linear layout and AppCompatActivity for "Search Name" screen.
- Create Listview using recyclerview with search on top of it.

- Create table of listing of college names if table does not exist.
- If table does not exist, load list of college names from www.data.gov (if they do not offer that, I will research and find another API).
- Create Adapter and RecyclerViewHolder classes.
- Load data from the table into listview (recyclerview).
- Allow search input to filter the listview while user types. Example: if user types Ant, then list view will be filtered based on that search input (except keeping “Any” on top of the listview)
- When users tap on one of the selection, it stores the value of college names and go back to search screen and populate it into “query” listview.

Task 7: Query the search result

- When the “Find School” button has been clicked, gather query and pass it to the class of Fast Android Networking and combine the values into URL and then call it to get result.
- If the result comes out an error, call snackbar to bring up an error.
- If the result comes out with no record, call snackbar to inform user that there is no record to be found based on search input or query.
- If there is one or more records from JSON, then load the “Detail” screen (it will be on next task).

Task 8: Implement UI for College List screen

- When the “Find School” button has been clicked, gather query and pass it to the class of Fast Android Networking and combine the values into URL and then call it from data.gov to get result in JSON.
- If there is no record, stay on “Search” screen. Otherwise, go to “Detail” screen
- Create layout and activity for “Detail” screen.
- Create cardview item that would contain one imageview, one floating button, and two textviews. The floating button would be “thumb up” image on the bottom right corner on the imageview.
- When user taps on the “Thumb Up” image, it will turn into dark color indicating that user likes that college and then the app stores information into local database.
- When user taps on the “Thumb Up” image that was already “Like”, then remove dark color that shows white indicating that user does not like that college anymore. And then app removes the record from the local database.
- Develop the event of “click” on imageview and it would go to “Detail” screen which is on next task.

Task 9: Implement UI for College Detail screen

- Create layout, and activity for “Detail” screen.
- The data would be passed from College list screen to this screen. Probably using parcelable.
- Read the data and load it into items
- The items on the screen would be imageview, “thumb up” floating button, textview (college name), listview below the image. The one row on listview would have two textviews which would be “website” and link or URL. (maybe it would be just one textview such as URL so user may know he/she can tap on it). Second row on listview would have two textviews which are “Location” and name of location (maybe it would be just one textview such as name of location so user may know he/she can tap on it).
- Third row of listview would have textview as “Cost”
- Fourth row would be populated the list of costs (another listview with it?)
- Fifth row would have title (textview) as “Aid”
- Sixth row would be showing two rows such as percentages. Please see screenshot on Screen #9.
- Seventh row would have a title (textview) as “Admission”
- Last row would contain information of “Admission” such as Acceptance, SAT/ACT scores.

Task 10: Implement UI for College “Detail - Location” screen

- Create layout, and activity for “Detail - Location” screen.
- This screen would be opened when user taps on “Location” on “Detail” screen.
- This screen will receive the location such as longitude and latitude from “Detail” screen that has data from parcelable.
- It then load the location into Google Map API within this screen.

Task 11: Implement UI for College “Detail - Website” screen

- Create layout, and activity for “Detail - Website” screen.
- When user taps on “Website” on “Detail” screen, it will pass the website’s link or URL to this screen.
- Then this screen will load it into webview in this screen.
- This is undecided whether we do this way or we should just launch the default browser on mobile phone.

Task 12: Implement UI for “Search” screen on tablet

- Create layout, and activity for “Screen” screen for tablet.
- Create split layout to have two columns on landscape.
- The left side would have similar to “Search” screen for phone except it has to be on listview instead of buttons. See screenshot, Screen #12.
- When user taps on one of the list, it would load other “search” activity into right side.
- Example: if user taps on “Degrees”, it would load the “Search - Degrees Selection” screen on right side.

Task 13: Implement UI for “Result or Likes and Detail” screen on tablet

- Create layout, and activity for “Result/Likes and Detail” screen for tablet on landscape.
- Create split layout to have two columns on landscape.
- The left side would be similar to Screen #3. When user taps on image, it would load the “Detail” screen on right side (Screen #8 and 9).

Task 14: Handle the network error message

- At launch time, check if there is connection or not.
- If there is no connection, bring up a message and warn them that search may not be possible.
- If there is no connection and there are some “likes” record, load that list on the screen while still showing a warning message.
- If there is no connection and there is no “likes” record, load “Search” screen with a warning message.

Submission Instructions

1. After you’ve completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
3. Add this document to your repo. Make sure it’s named “**Capstone_Stage1.pdf**”