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: kathbena

FlickPick

Description

Ever try picking a flick to watch and have a hard time finding something that suits your mood? Trying to find something you would not normally choose? Staying in and trying to find a great movie to rent? Wondering what good movie is out in theaters to watch? The FlickPick app is

here to make those decisions easier! The FlickPick app helps find a movie for you based on a genre you select and whether you are going out or staying in. With access to a large movie database, you can find results from the most recently released to good ole classics! In addition to finding a movie, the FlickPick app directs you to Fandango to buy movie tickets or Amazon to rent your desired movie.

Intended User

The intended user for this app is anyone who loves watching movies.

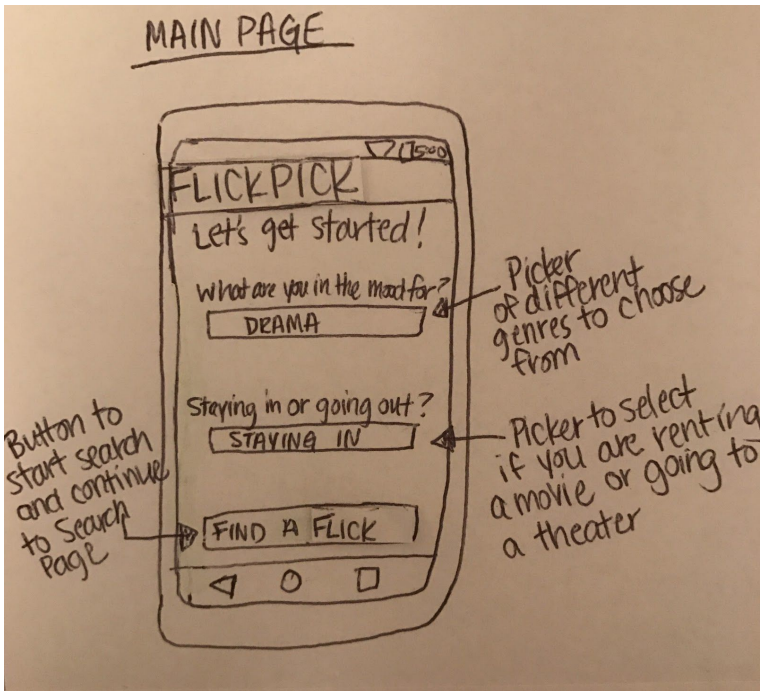
Features

- Picks a movie based on genre and location
- Links to Fandango to buy movie tickets
- Links to Amazon to rent movie
- Provides movie details such as name, description, rating, and movie poster

User Interface Mocks

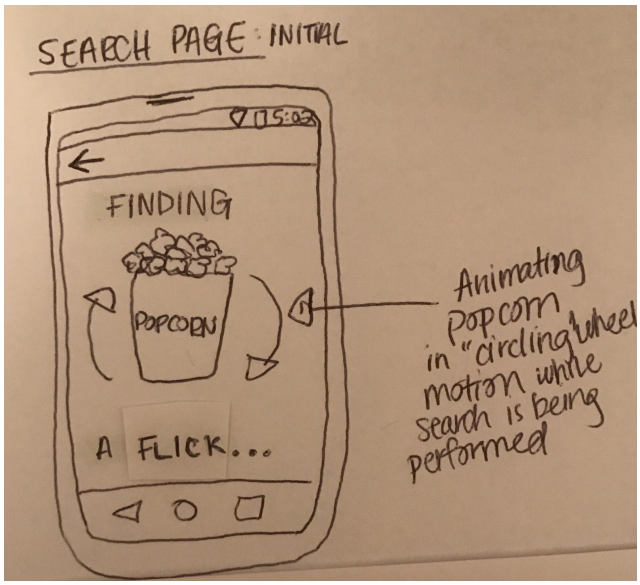
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

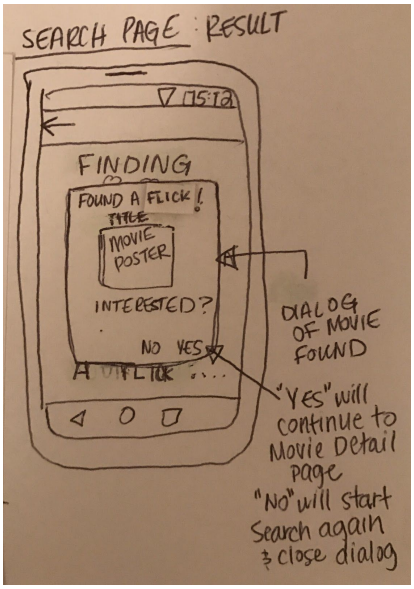
Screen 1 - Main Page

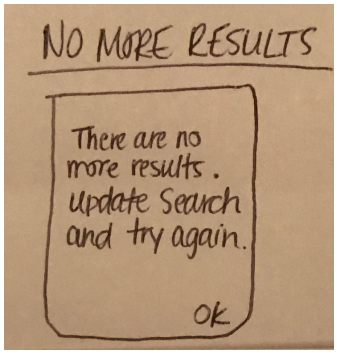


This page is the start page and main screen for the app. There are 2 questions that will be answered by the user to find a movie suitable to genre and location (staying in vs movie theatre) for the user. The first picker will contain a variety of genres for the user to choose from. The second picker will have only two options: Staying in and Going to the movies. Only one option can be selected between each picker. The “Find a Flick” button will direct the user to the Search page.

Screen 2 - Search Page: Initial and Dialog

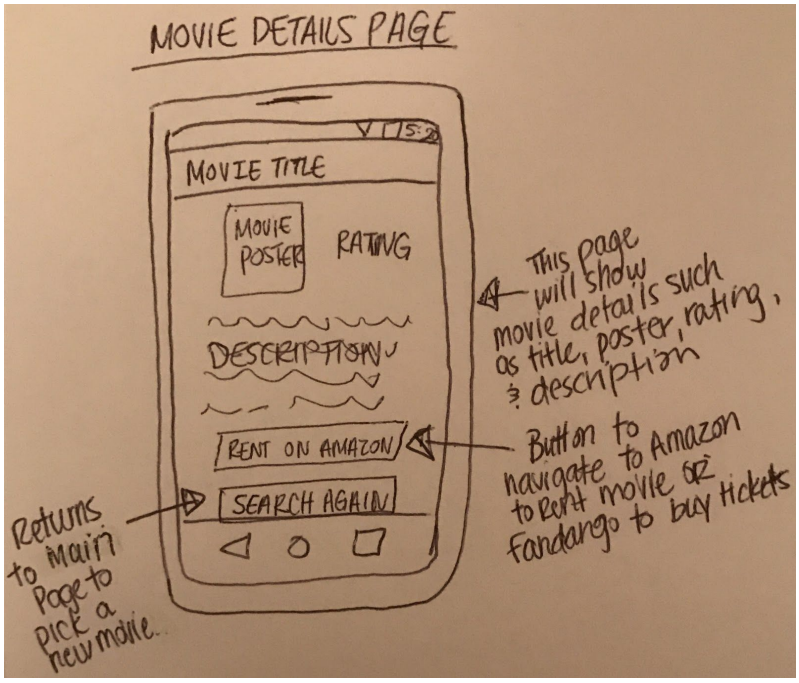






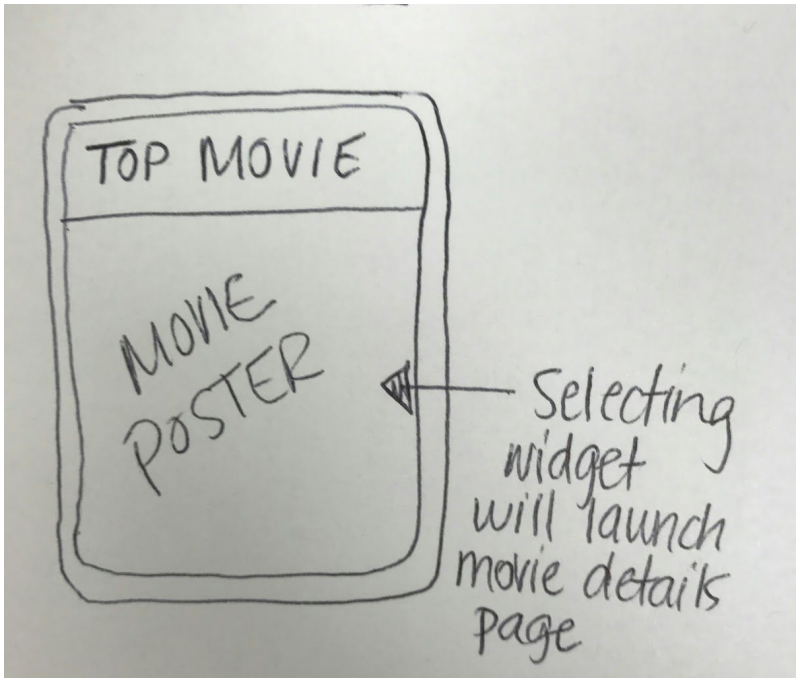
The Search page will have an animation of a spinning popcorn image (similar to a spinning wheel) to show the search is active while the movie is being chosen. The movie results will be ordered by highest rating to lowest rating and the first item will be shown to the user in a dialog. If the user selects “YES”, the user will proceed to the Movie detail page with those details. If the user selects “NO”, the dialog will close, and the next movie in the results will be shown in the dialog. If there are no more movies left in the results, the dialog will show an alert for the user to change search criteria and will then redirect them back to the main page upon selecting “OK”.

Screen 3 - Movie Detail Page



The Movie Detail page shows the user movie details such as title, poster, rating, and description. If the user selected “Staying in” on the main page, the first button will show “Rent on Amazon” and direct the user to the Amazon site. If the user selected “Going to the movies”, the first button will show “Buy Tickets” and will direct the user to the Fandango site. If the button is for Fandango, then location will be checked for the user in order to send to Fandango in the redirect to have a location for nearby theatres. The “Search Again” button will take the user back to the Main page to allow the user the option to modify their search or search again.

Screen 4 - Widget



The widget will show the user the current top movie in the box office by displaying the poster of the movie. If selected, the app will open and the user will be directed to the Movie Detail page with details of the movie. The first button will show “Buy Tickets” and will direct the user to the Fandango site. It will show the same “Search Again” button with same action described in Movie Details page.

Key Considerations

How will your app handle data persistence?

The movie data will come from themoviedb.org website. The data will be fetched using an AsyncTask to perform the requests of the data. This app also uses a Loader to move its data to its views. This app also implements a ContentProvider to access locally stored data such as already seen movies and store data in SQLiteDatabase.

Describe any corner cases in the UX.

- If the user selects the “Search Again” button on the Movie Details page, the title will be saved and used to remove from results, so it does not show again.
- If the user selects the back button on any of the pages, they will return to the previous page.

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

- I will be using the Picasso library to load and cache images for the movie posters on the Search and Movie details pages.
- I will be using MovieDB Api for more information

Describe how you will implement Google Play Services.

- Google Location to get location of user to send to Fandango for local theaters
- Google Address API to also get location if user wants to enter address to show theaters near there for Fandango 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.

Task 1: Project Setup

- Create project and add to repository
- Set up libraries and google play services
- Structure packages in project
- Create basic structure for app and main menu for already seen movies

Task 2: Implement Main Page Activity

- Create the view for displaying questions
- Create two pickers for genre and location

- Create objects for user input
- Launch search page when button on this page is selected

Task 3: Implement Search Page Activity

- Integrate with API
- Create AsyncTask to perform requests and response for the movie data
- Create animation of image on page to move in circle motion
- Pass inputs from main page to this page to create request for data
- Create dialog to show movie poster and title and YES/NO actions
- Setup Picasso library to retrieve and show movie poster in dialog

Task 4: Implement Movie Detail Page Fragment

- Implement Google Play Services for location
- Create view for displaying movie details
- Populate UI with data
- Pass in user input for staying in or going out for Rent/Buy tickets button
- Add logic to show and use Amazon vs Fandango link
- Launch Main page when user selects “Search Again” to create new search

Task 5: Implement Widget

- Create widget view
- Create widget action to display movie details page
- Pass in movie data for displaying top movie for widget display
- Populate view for displaying movie details

Add as many tasks as you need to complete your app.

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**”