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: Jaren831

Judge!

Description

Write a brief summary of what your app does. What problem does your app solve?
Judge! Provides a one stop shop for all Magic: The Gathering needs during play. Consists of three components: a card search feature that allows player to search for cards, a life counter interface during a game, and an up to date and easily navigable rulebook. The life counter

interface is more UI focused, allows player to tracks life during a game and keep track of tokens, poison, counters, etc.

Intended User

Judge! Is intended for use by anyone who plays Magic: The Gathering and is looking for a quality companion app to their games.

Features

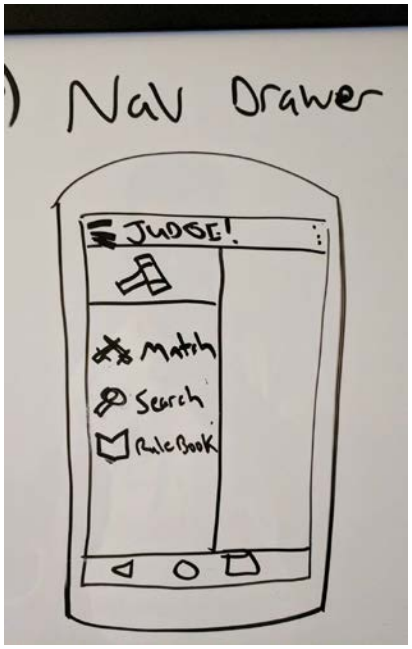
Judge! Features:

- Match information
- Card search
- Rulebook
- Provides beautiful UI for tracking life and other counters during match.

User Interface Mocks

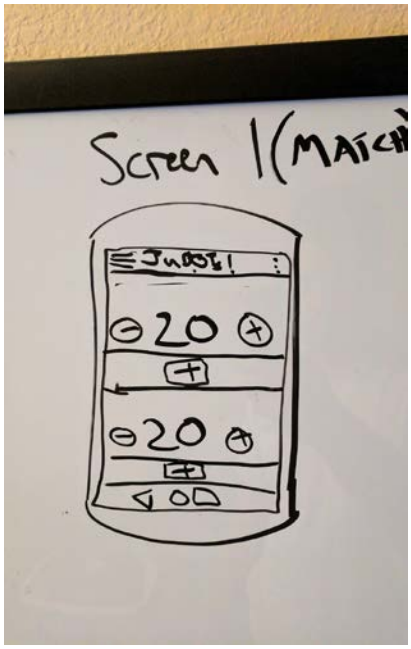
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 0



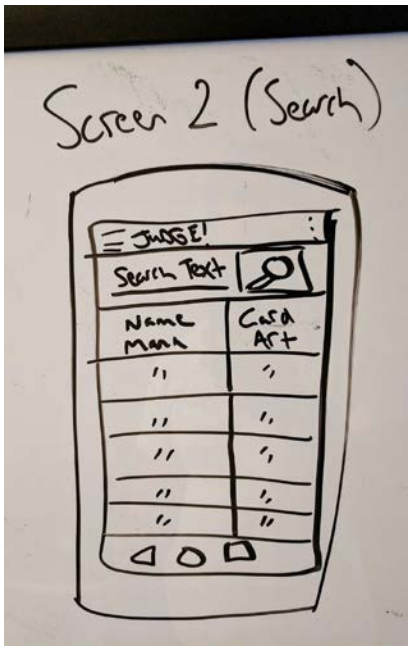
Navigation drawer. Provides means for user to navigate between the 3 features. Each is contained in a fragment.

Screen 1



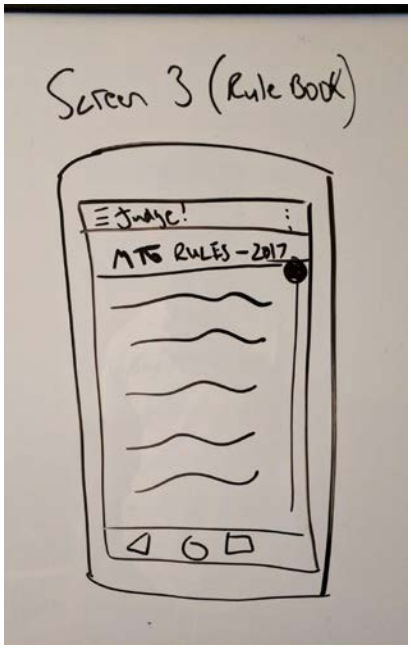
Displays a match with 2 players, 20 life by default, and blank token tracker. Settings button in top right will allow configurable starting life and colors. Pushing + or - by life will decrement or increment life. + button below life with add new token.

Screen 2



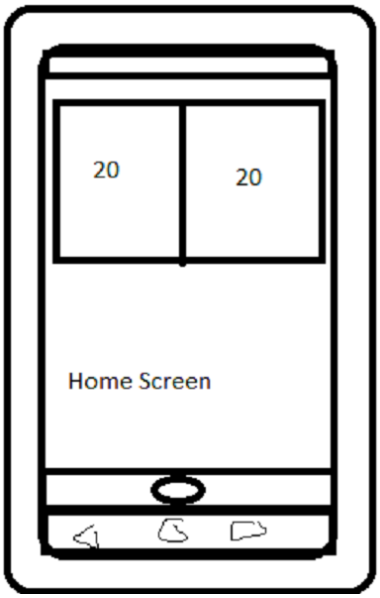
With empty search query will not display card list. When card name has been searched, will display relevant results below search box. Will display card name, mana cost/type, and card art. Pushing back button will take user back to match screen. Settings button will be hidden in this fragment.

Screen 3



Will display rule book in easily navigable format. Pushing back button will take user back to match screen. Settings button will be hidden in this fragment.

Screen 4



Will display life counts of current match. Pressing on widget will take user to match fragment. Is not changeable outside of app.

Key Considerations

How will your app handle data persistence?

I will build a content provider for card data from http://magicthegathering.io/ and store on firebase. Will build AsyncTask for card search to search for card information. Using firebase realtime database.

Describe any corner cases in the UX.

User pushing back button in screen other than Match screen (screen 1) return them to match screen. Pushing back button with nav drawer open will close nav drawer.

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

I will be using Glide to handle the loading and caching of images. http://magicthegathering.io/ API for card information.

Describe how you will implement Google Play Services.

I will be integrating Firebase in the app to enable easy patches. Admob will be used to display ads displayed on bottom of screen. Analytics will be used for tracking app data.

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

Subtasks:

- Create nav drawer layout
- Add Match screen fragment.
- Add empty search screen fragment.
- Add empty rulebook screen fragment.

Task 2: Implement UI for Each Activity and Fragment

Subtasks:

- Build UI for MainActivity
- Build 3 fragments to be contained in MainActivity(Match, Search, Rulebook)
- Build nav drawer to navigate between fragments.
- Build PreferenceFragment for Match fragment.

Task 3: Configure Firebase

Get firebase up and running. Configure proper backend for app.
Subtasks:

- Take firebase in a weekend udacity course elective.
- Implement. Using firebase real time database.
- Use MTG card api from http://magicthegathering.io/
- Build library to store information in firebase

Task 4: Configure AsyncTask for Card Search

Adds AsyncTask to get card data asynchronously when user searches
for card information:

- Create card search AsyncTask.
- Implement in Card search fragment.\

Task 5: Configure Loader for Views in Fragments

Loaders will be user to display data in each view:

- Loader for match fragment.
- Loader for card search fragment.
- Loader for rule book fragment.
- Loader for widget.

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**"