Target ship date: 2022-12-23

Squirtle Squadron, Andrew Piatetsky, Ayman Habib, Weichen Liu, Jeffrey Zou, Raven (Ruiwen) Tang
SoftDev
P01: ArRESTed Development Design Doc
2022-12-06

Feedback From other teams:

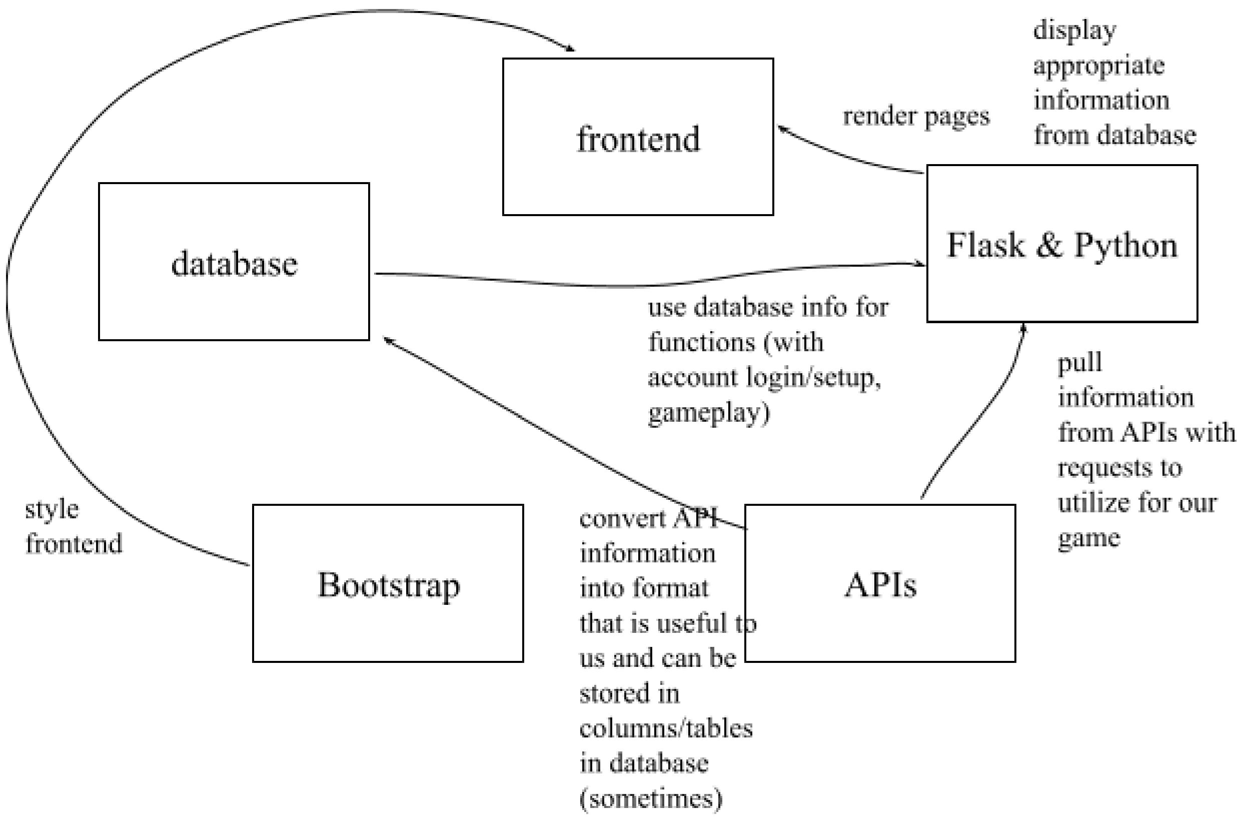
Add api cards
~~Rethink tables~~
Revamp site map
~~Add spotify API~~
~~Explain more about APIS~~
~~Rethink routes~~
Add templates
~~Revamp roles~~

Idea: Create a higher-lower game that involves comparing the population and weather of cities.
Breakdown of Tasks & Member Assignments

- [ ] Backend database creation, population, and management (Weichen, Jeffrey)
- [ ] Python associated with databases (Weichen, Jeffrey)
- [ ] API usage (Andrew)
- [ ] Connecting API to frontend and backend (Andrew)
- [ ] Python Flask work (Andrew, Jeffery, Weichen)
- [ ] Account signup and login (Ayman, Raven)
- [ ] HTML (Ayman, Raven)
- [ ] Bootstrap framework for styling (Ayman, Raven)
- [ ] Project management (Andrew)
- [ ] Devlog updating (All)

Components:

1. Database
    a. Store user account information
2. Frontend
    a. Display the game to the user
3. Flask & Python
    a. Render frontend pages
    b. Game functionality
    c. Account login/setup functionality
    d. Database setup
4. Bootstrap framework
    a. Make our frontend pretty, intuitive, and user-friendly!
5. APIs
    a. Provide info for population and temperature of cities



frontend
database
Flask & Python
Bootstrap
APIs
display appropriate information from database
render pages
use database info for functions (with account login/setup, gameplay)
pull information from APIs with requests to utilize for our game
convert API information into format that is useful to us and can be stored in columns/tables in database (sometimes)
style frontend

SQLite3 Database

We’re currently thinking about not using tables overall for the objects we’re comparing and use lists that we can get a randomized object from.

- Tables:
    1. **Users**

Username (primary key)	text
Password	text
Points	integer
Profile Picture (potentially)	Link (text)

<u>Frontend</u>

- Bootstrap. It is very responsive.
    - Easy to use cards which we can customize
    - Upgraded form controls
    - Cool features such as dropdown menu and progress bar
    - Overall, very customizable and easy to use
- HTML
    - Use templates to connect with flask application

<u>Templates</u>

- country.html
- create_account.html
- error.html
- game.html
- home_page.html
- index.html
- leaderboard.html
- login.html
- result.html
- sign_up_success.html

<u>Flask Site Map</u>

@app.route(‘/create_account’)
@app.route(‘/’)

- Login page

@app.route('/home')

- Buttons to other routes

@app.route('/logout')

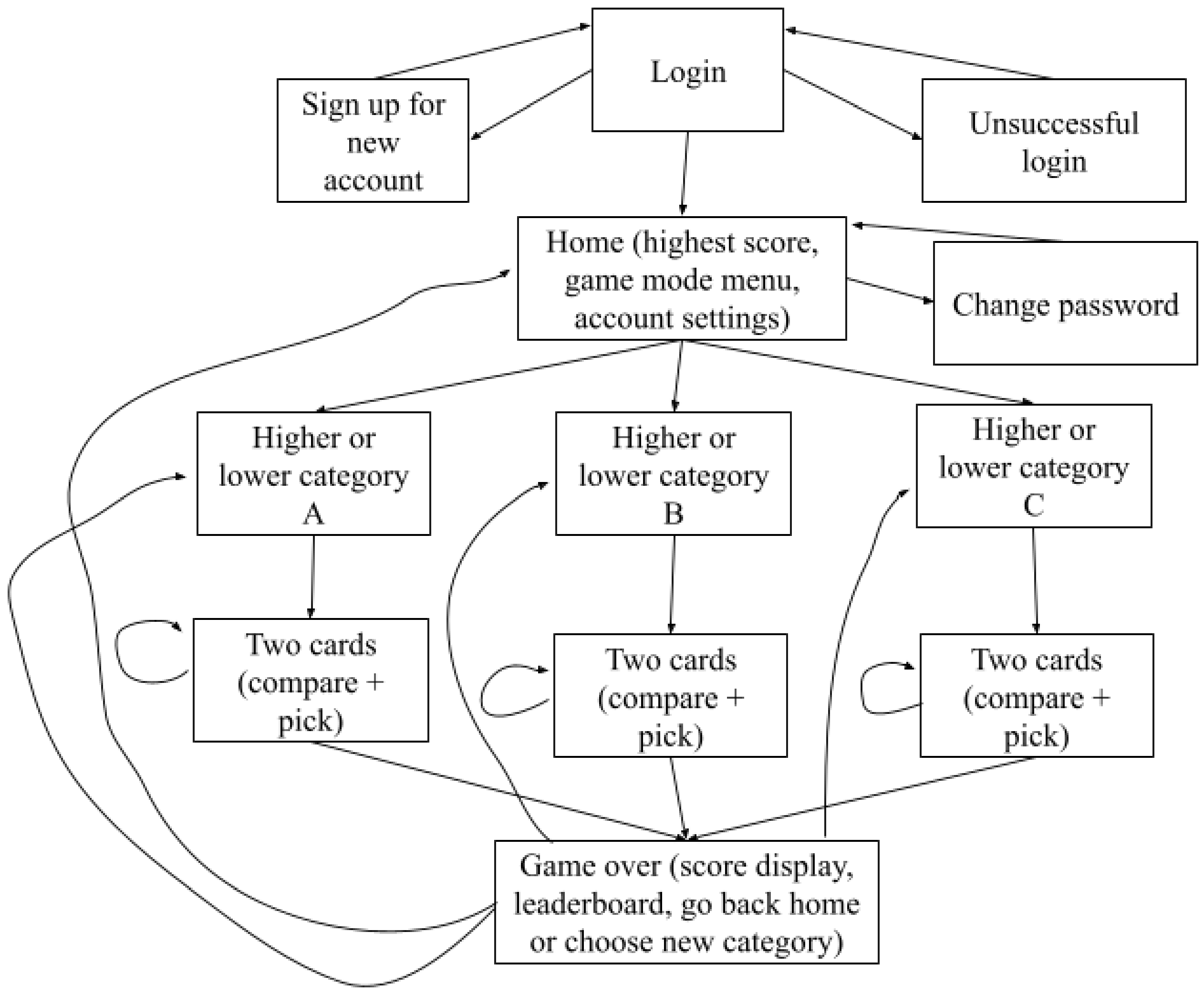
@app.route('/game')

- Higher or lower gamemode in here

@app.route('/weather')

@app.route('/leaderboard')

- Possible a leaderboard implementation


Login
Sign up for new account
Unsuccessful login
Home (highest score, game mode menu, account settings)
Change password
Higher or lower category A
Higher or lower category B
Higher or lower category C
Two cards (compare + pick)
Two cards (compare + pick)
Two cards (compare + pick)
Game over (score display, leaderboard, go back home or choose new category)

APIs

Will create a spam email for this

- Email: sqsquad53@gmail.com
- Fake Name: Steve Rja
- Password: "   " (highlight to see)

- Alt pass: “   ”

1. ~~Spotify API (how many listeners)~~
    a. Don’t offer it through their API, only way to get it is through scraping
2. ~~Instagram API (to get how many followers they have)~~
    a. https://developers.facebook.com/docs/instagram-api/guides/business-discovery
    b. Will get how many followers someone has
    c. need a phone number
3. ~~Google Trends (how many people are searching for that thing)~~
    a. Implementations are not restful
4. ~~Twitter followers (and possibly retweets or something of that nature)~~
    a. https://developer.twitter.com/en/docs/twitter-api/users/follows/introduction
    b. Will just fetch the twitter followers of a certain profile (celebrity or other famous person)
    c. Not wanting to work - also need phone number
5. ~~Youtube subscribers~~
    a. https://developers.google.com/youtube/v3/docs/channels
    b. Will just fetch the number of subscribers that a person has
    c. Doesn’t fit into the theme anymore
6. ~~Census Population/Demographics~~
    a. https://www.census.gov/data/developers/data-sets.html
    b. Population age, size, average income
    c. No useful information for us
7. ~~City weather (OpenWeather from the API cards)~~
    a. Which city is hotter right now
    b. https://openweathermap.org/current
8. ~~City demographics~~
    - https://population.un.org/dataportalapi/index.html
9. City population
    - https://api-ninjas.com/api/city
    - Gives the population of a city
    - Need to create api card
10. ~~City area~~
    - https://dbpedia.org/sparql
    - Returns area of a city in meters squared, should display it as football fields
    - Test: https://dbpedia.org/sparql?default-graph-uri=http://dbpedia.org&query=select ?area where { dbr:San_Francisco dbo:areaTotal?area}&format=json&signal_void=on&signal_unconnected=on

11. Returns a random city every time (with a bit of
    - http://geodb-cities-api.wirefreethought.com/demo
12. Returns images of a given city (by querying for key words)

Roadmap:

- Construct SQLITE database along with various helper functions
- Build HTML templates along with flask web application
- Combine API and database into flask application
- Make functions for features
- Bootstrap and CSS to make app look pretty
- Testing and debugging