Term Project Outline

Member Details

- Riki Fameli
  - Intro Course: CS19
  - Strengths:
    - Prior experience with basic web development
      - Python, Django, HTML+CSS, database migrations in Django using SQL
    - Functional programming and recursion
    - S/NCing two other courses for this one xd
    - Sophomore
      - Imagine being a freshman
  - Weaknesses:
    - Little experience with OOP and Java outside of this course
    - No experience with Javascript outside of this course
- Thet Htay Zaw
  - Intro Sequence: CS17/18
  - Strengths:
    - Prior Experience with OOP and Java
    - Prior Experience with functional programming and putting OOP and functional programming together (Scala)
  - Weaknesses:
    - No front end, web development, or Javascript experience before CS32
    - As a visual learner, using SQL is a nightmare for me
- Siddharth Diwan
  - Intro Sequence: 19
  - Focus
    - Experience with Java, Python, Pyret
    - I like to test but only if there's no code to write
  - Working on
    - Some JS, but not much

        - SQL is fine if I have a command visualizer
- ➢ Theo Fernandez
    - Intro Sequence: 19
    - Strengths:
        - Quick Learner
        - Good presentation skills
        - Amazing sense of humor 👉😎👉
        - I have used python before (but still not much experience)
        - Freshman :)
    - Weaknesses:
        - Little prior experience with Java/Javascript/HTML/CSS/SQL
        - SQL is pretty difficult for me
        - Testing

Project Ideas

JournalTexter

JournalTexter is a journaling app that makes it easy and engaging for users to type up their thoughts. After a user signs up, they can make their first entry, in which they'll be prompted by a Journaller. The Journaller will give them a prompt, and once they answer, the Journaller will provide a few related questions to get more of their thoughts down. Unlike a chatbot, the Journaller is just there to feed you related questions/prompts to help you explore and document different aspects of your life that you may not have considered writing down. Once the user is done journaling, they can save the entry to look back on later. This makes it easy for users to keep writing, however much they want.

Base Requirements

Our program must be able to:

- Authenticate users and track data for each user
- Allow users to create accounts
- Allow users to input and potentially customize/format text
- Store and display previous journal entries in a user-friendly way

- Use an intuitive interface to allow users to message the journal
- Present users with suggested questions to answer in their entries
- Suggest questions and/or respond to user input using a complex algorithm (see Algorithmic complexity below)

User Accounts and Authentication

- Like a physical journal, users should have the ability to store entries and revisit old journal entries
- To save these journal entries, users will need to create accounts
- Journal entries should be stored and presented in a user friendly manner when you log into your account

Journalizing an Entry

- When a user decides to create a new entry, the journaller will start off with a prompt to set the "mood" of the entry.
    - This can take the form of the journaller asking "How are you feeling today?" or the journaller asking the user to quantify their mood from a scale of 0 to 100.
- Once the "mood" of the journal entry has been established, the journaller will begin choosing questions from a questions bank that fits the appropriate "mood"
- The journaller will provide multiple questions and the user will click one to answer
    - The user will have the ability to reject the offered questions and get new options
- This takes the question out of the bank of available questions for the entry
- As the back and forth between the user and the journaller continue, the journaller will amass information that allows it to pick even more specific or tailored questions
    - See Algorithmic Complexity below
- Once the user is satisfied with the "conversation," they may have the option to stop the journaller from prompting so that they may finish their thoughts uninterrupted

- After the journal entry is done, the user can save their entry (with or without the journaller prompts) and be done for the day.

Algorithmic Complexity

- Sentiment analysis
    - Analyse the sentiment of journal entries in response to previous prompts to determine the next set of prompts to give the user.
    - Eg, if the sentiment is low, ask more favorable or light-hearted questions.
- Word Count Vectorization
    - Use the frequency of certain words in the previous journal entry to determine the next set of prompts to give the user.
    - Eg, if piano is used frequently in the previous journal entry, ask a question specific to music.
    - Warning:
        - This could create a loop where the JournalEditor will then only ask about music. For this, introduce randomness in which ranked word-to-vecs with highest frequency are chosen for the next set of prompts.
    - Or, make sure that only word-to-vecs with ‘middle’ to ‘low’ frequency are used to determine the next set of questions because words with higher frequency have likely already been addressed by the user in their previous entry.

Nice to Have / Stretch Goals

- A streak system or point system where passing a certain threshold unlocks new characters, icons, or customization
- An interface like SIMULACRA, where unlocked characters are group members you talk to (with slightly different personalities)
    - Each character may have their own ‘local lingo’ eg. British, Australian, by passing through some API
- A ‘donate question’ feature that allows users to suggest questions after picking the proper category for then

    - This would not immediately add the question into the question bank, any user submitted questions would be vetted
- A functionality that allows users to edit, hide, or delete previous journal entries they don't want to see

StudyBuddy

StudyBuddy is a system which will match you with people who are close in location and are studying similar topics to you. StudyBuddy will give users an initial survey asking them about their study habits, including where they like to study, when they like to study, where they live, their courses, their intended concentrations, and general interests/hobbies. Users can also select whether they want to be matched with people with similar answers, or if they would want to be matched with everyone. Then, the algorithm will suggest other profiles to the user. As the user decides to meet with certain people, the algorithm can suggest other profiles similar to the ones that the user likes. Users will also have the option to message other users and to plan meetups in the app. Ideally, as the user uses the StudyBuddy more, the app will get better at suggesting other users to study with.

Requirements

- Authenticate users and track data for each user
- Allow users to create accounts
- Present an easy to understand interface for users to navigate
- Implement an in app messaging system
- Implement an algorithm which suggests profiles which are similar to profiles that the user has liked before

User Accounts and Authentication

- Like any software that matches people, users will need to create accounts and fill them with details StudyBuddies will keep track of
- Upon logging in, the dashboard will present the people you are currently talking with and any new suggestions that might have occured while the user was logged out

Algorithmic Complexity

The algorithmic complexity comes from the apps ability to suggest new profiles to the user. Initially, the app will suggest based on the answers to the survey, but as the user uses the app more, the app will be able to determine which types of profile the user is more likely to meet up with. The goal of the algorithm would be to predict which users are most likely to meet up with each other, and then to suggest those users to each other.

Stretch Goals

- Display a map showing all users/matches within the vicinity and their match score
    - People will say they are in certain locations, and will have the ability to toggle this visibility on and off or share only with specific users
- Ability to set up and keep track of study dates within the app
- Food break functionality
    - Possibly link to the open dining halls and their menus
- High quality graphics
- Blacklist/blocking functionality