Web Application

Pitch

Currently, the system to randomly match roommates isn’t very efficient or known to students. Students have to track down roommates on their own, which is hard to do and often results in pairs that don’t work well together. ZoomieRoomie will allow students to find a “perfect” roommate match based on their personal habits and the habits they are looking for in a roommate. Students will later provide feedback on the match, which ZoomieRoomie will use to improve its suggestions for future roommate pairings.

Functionality

1. Users can fill out a form about their personal habits and qualities
2. Users can fill out a form about the habits and qualities they are looking for in a roommate
3. Users can retrieve their “top roommate matches”
4. Users can see how roommate pairings with similar characteristics worked together in the past
5. Users can create an account using their Illinois email address
6. Users can rank dorm preferences

Continuous Integration:

- The library we will be using to run our tests is Catch.
- The style guide we will follow is camelCase. The automated tool we will use to check the style guide is linters for each language, such as JaSCS for JavaScript.
- We will compute test coverage using JUnit for our Java code.
- We expect that each team member will submit a pull request when they feel they have accomplished their specific task successfully, efficiently, and cleanly. We think the main reviewer for each PR should be the other team member who is proficient in the specific framework, tool, etc. In the case that this is not possible, we are expecting that all team members will learn every framework, so somebody else can review that specific PR. In order to avoid merge conflicts, we will make sure that each team member is assigned a specific task and that these tasks don’t overlap in a way that will cause any conflicts.

Components

- Backend: We will write the backend using Java and MongoDB for our database needs. All of our team members have experience with Java, so this will allow us to focus on learning the new frameworks and platforms we chose. We will use the Spring framework to provide Java infrastructure support. We will also use Python and PyTorch for the machine learning components of our project. We will use a combination of unit testing and integration testing to check the functionality of all pieces of the back end. The majority, however, will be integration testing because that will provide tests that are most similar to how the users will use the program. The components will interact with each other through RestAPI calls.

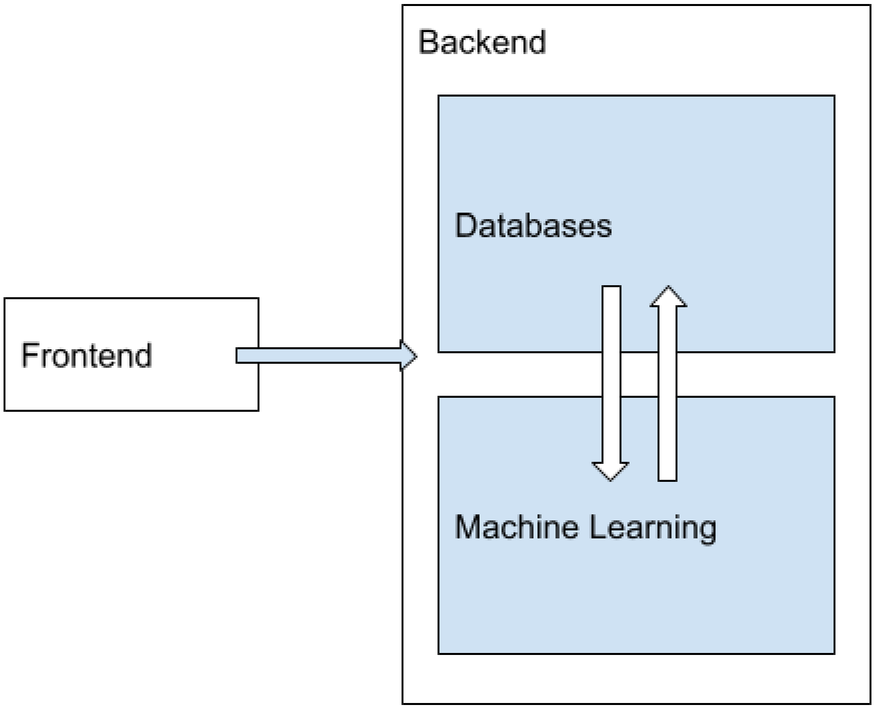
The backend has the following responsibilities (the parts of the program that the user cannot see or interact with):

  - Store a list of user accounts
  - Store users' form responses
  - Analyze data and improve roommate matching system

- Frontend: We will write the frontend using JavaScript and the React framework. Most of our team members have familiarity with JavaScript, but not React. We chose to use the React framework because of its ability to create custom components as well as its quick rendering capabilities. React is also widely used, so there is documentation and community support we could follow to enhance our UI.

  The frontend has the following responsibilities (the parts of the program that the user can see and interact with):

  - Graphical user interface
  - Login page/User authentication
  - Access to forms and submission capabilities
  - Viewing roommate matches



Weekly Planning

1. Create databases that will include the data for the roommate matching. Also create methods for extracting data from them.
2. Create Illinois student authentication and log in compatibility. Start to write unit tests for the front end.
3. Create forms for users to fill out habits and preferences. Devise a method of inputting information into databases.
4. Start working on the frontend using React, and learn how to use it. Create homepage that shows available forms and top roommate matches

5. Learn about the machine learning program and how to implement it into the rest of our project. Begin devising machine learning algorithms from form responses.
6. Continue tuning machine learning algorithms. Begin to connect this to roommate suggestions. Write unit tests for the backend.
7. Create sample data by asking current roommate pairs to fill out the forms. After getting data, using this data in the algorithm and tweaking it.
8. Create a way to visualize how roommate pairings with certain characteristics worked together in the past. Finish running the unit and integration tests and develop a final product.

Potential Risks

1. As we are planning to tackle Machine Learning and none of our group members have worked in depth with Machine Learning before, we may encounter difficulties with the PyTorch application. We are not very familiar with it, so if we have any issues, we plan on using the internet's community support on how to develop projects using PyTorch, and also refer to our mentor. We have given ourselves plenty of time in our schedule so that we can deal with these issues.
2. In order to create our project, we need sample data to train our machine learning algorithm. We don't think we will be able to get access to the current UIUC roommate system and database. Instead, we will try to obtain data from current roommate pairings, as well as how they have worked together so far. This will be even more beneficial because we can get first-hand responses about roommate pairs with certain characteristics. Getting data from current students will probably take a while, so we will do this throughout the entire semester on the side, so that we can follow our schedule.
3. There is a chance that we do not have enough data to be able to create reasonable outputs from our machine learning algorithms. In this case, we will still devise the algorithm, but we will plan to do more in-depth training as our program is used and receives more data.
4. Another potential risk we might encounter is our scheduling, and how our team might have conflicting timings. We could possibly be busy one week, and might not be able to accomplish the tasks that we needed to complete for that week. We will try to fix this by having some cushion from week to week, so that some weeks are easier, and some weeks are harder, and that way we can push back the task we couldn't finish for the next week. Accordingly, if we were to have conflicting schedules that impacted our weekly planned schedules, making us not meet our deadlines, we would try to add that task to a week where we either have more time or the task we have planned for that week takes less time.

Teamwork

We will use a Docker Environment in order to keep our work across all of our devices consistent, and we plan on using Visual Studio Code as our editing environment.

In order to work together efficiently, we will all use a trello board to keep track of each person's to-do items, and their finished items. This way, we avoid confusion and will work on separate parts of the project and not overlap work.

Since we are a team of four people we will divide into two sub teams of two people each: a front end team (Vineetha and Ayush) working on React and JavaScript for the UI, and a backend team (Julie and Peter) working with Java. However, as a team, we are all interested in Machine Learning, so we will be working collaboratively with PyTorch with Python.