

HokieFinder Report

Team Members:

Taeesh Azal Assadi
Faraz Ulhaq Shah
Priscilla You
Asmi Panigrahi
Onjali McEachin

1. Product description

HokieFinder is a housemate matching tool created especially for Virginia Tech students, especially those who live nearby to the NCR campus, to address the particular difficulties faced by out-of-state and international students. Utilizing a proprietary algorithm that takes into account living patterns, cultural preferences, and budget, the application guarantees compatibility and provides individualized matching based on each user's demands. Its innovative swipe-based user interface and built-in messaging capabilities make finding a flat mate straightforward, fun, and stress-free.

By encouraging diversity and catering to varied tastes, HokieFinder not only streamlines the home search process but also fosters a sense of community, guaranteeing a smooth and supportive experience for students navigating their housing journey.

GitHub Link - <https://github.com/taeesh1309/RoomateFinderApp>

Presentation Link -

https://www.canva.com/design/DAGYokB3SyA/B0adFJZLLdUcQ9gv9HtJ2A/edit?utm_content=DAGYokB3SyA&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

2. Product Functionality

Below is the product functionality with a walkthrough.



Fig 1.

Fig 1. When users open the app, the first screen appears. After a few seconds, they can start using the app.

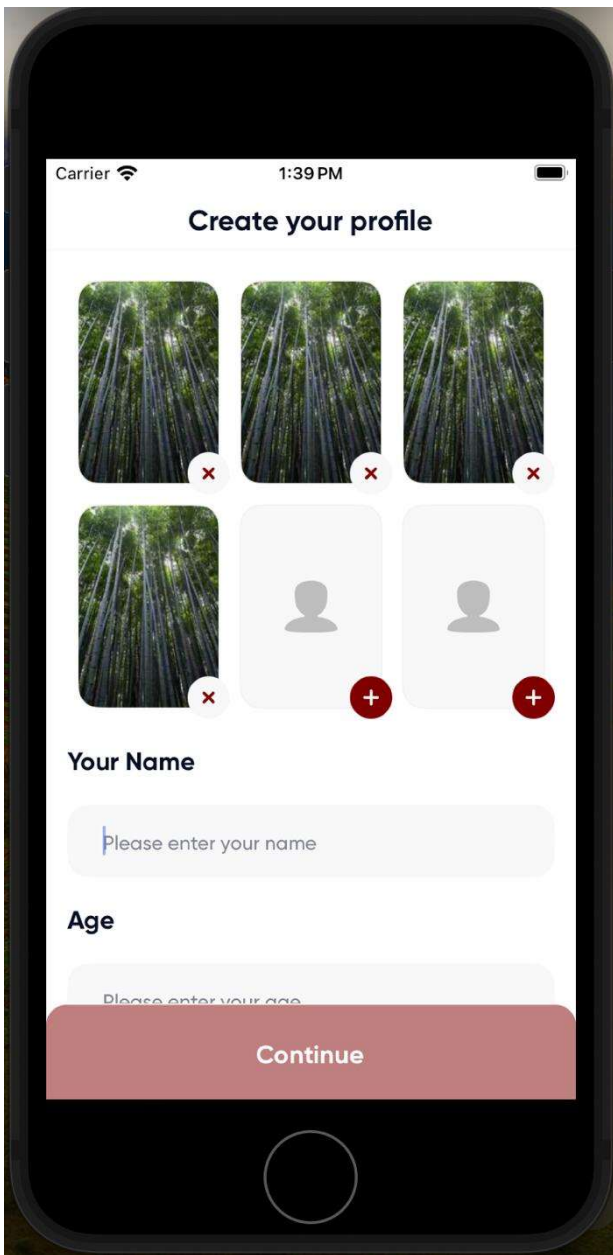


Fig 2.1

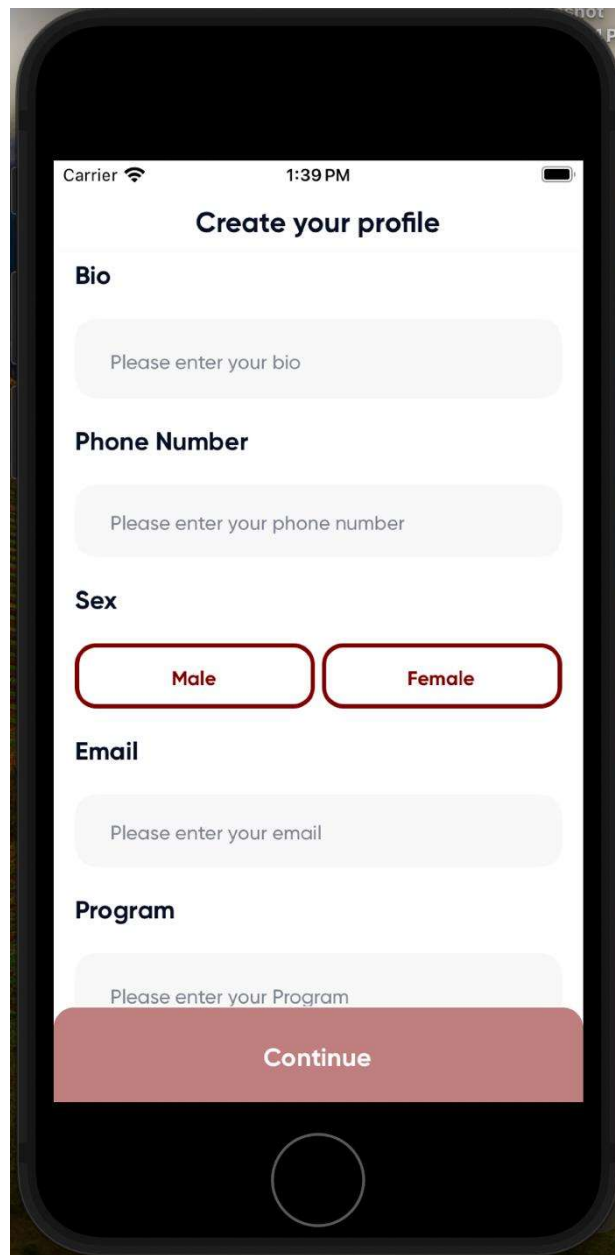
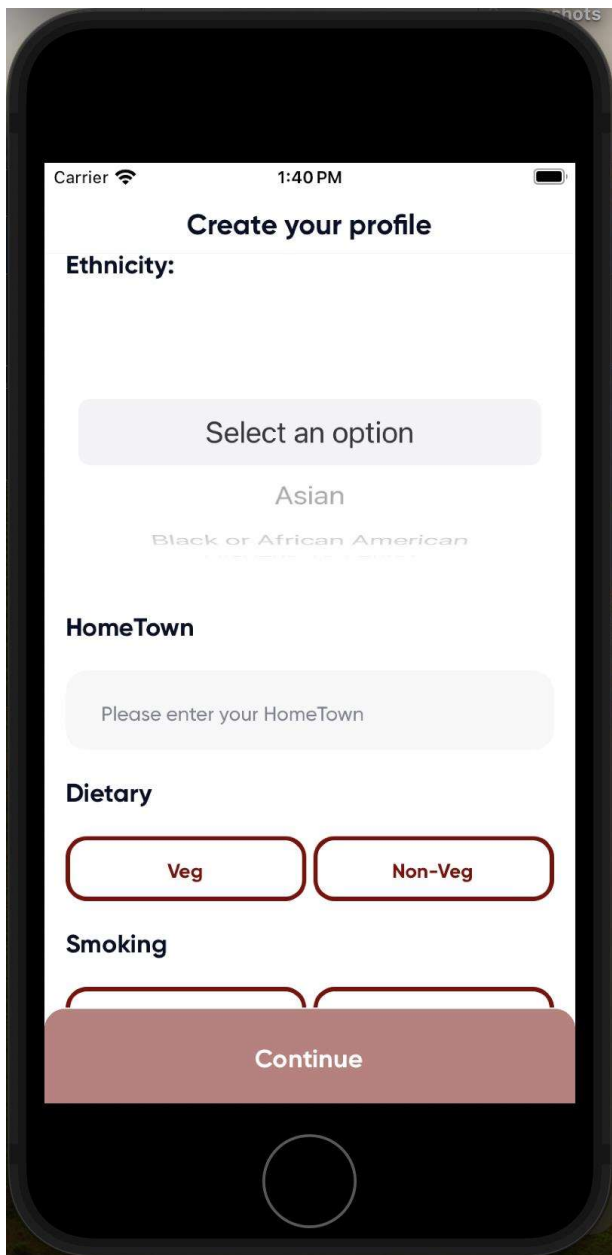


Fig 2.2

Fig 2.1 ~ Fig 2.6. To login and create profile, the user must provide their own information (Name, Age, Bio, Phone Number, Sex, Email, Program, Ethnicity, Hometown, Dietary Preference, Smoking, Drinking and Budget)

To get matched with a roommate, the user must fill in their preference information (match with, Dietary preference, Smoker, Drinker, Age, Home type, Apartment Type and Move in Date)



Carrier 1:40 PM

Create your profile

Ethnicity:

Select an option

Asian

Black or African American

HomeTown

Please enter your HomeTown

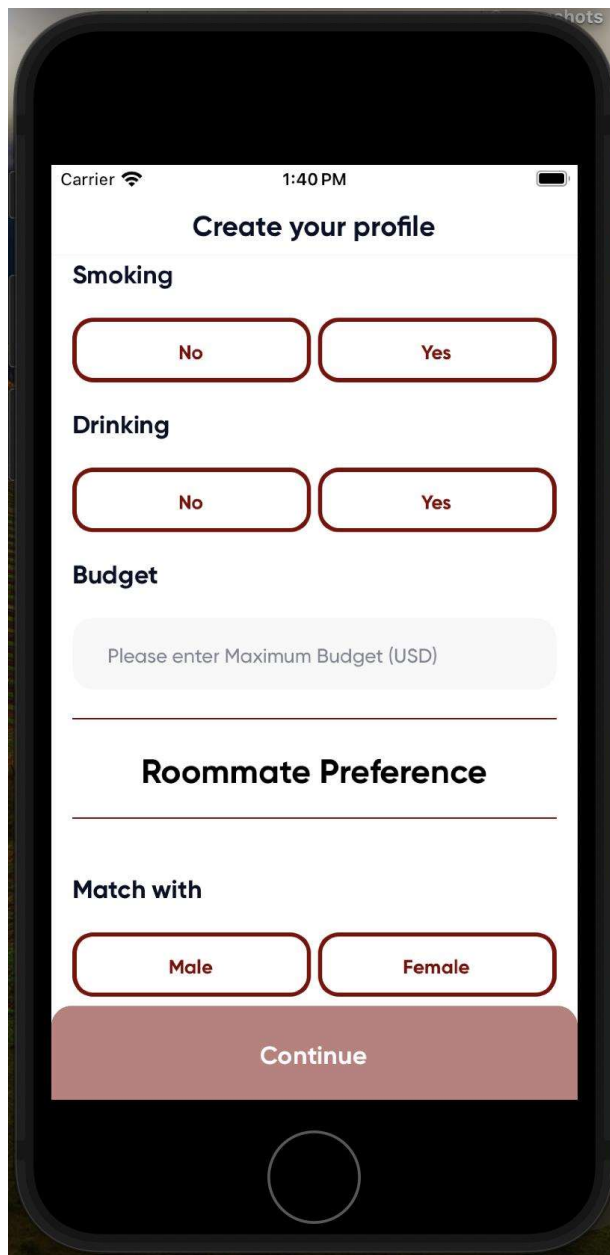
Dietary

Veg Non-Veg

Smoking

Continue

Fig 2.3



Carrier 1:40 PM

Create your profile

Smoking

No Yes

Drinking

No Yes

Budget

Please enter Maximum Budget (USD)

Roommate Preference

Match with

Male Female

Continue

Fig 2.4

Carrier 1:41 PM

Create your profile

Ethnicity:

Select an option

No Preference

Asian

Dietary Preference

Veg Non-Veg None

Smoker

No Yes Maybe

Drinker

No Yes Maybe

Continue

Fig 2.5

Carrier 1:41 PM

Create your profile

Drinker

No Yes Maybe

Age

Please enter your roommate's preferred age

Hometype

Apartment Type (number of bed)

Studio 1 2+

Move-in Date

Dec 10, 2024

Continue

Fig 2.6

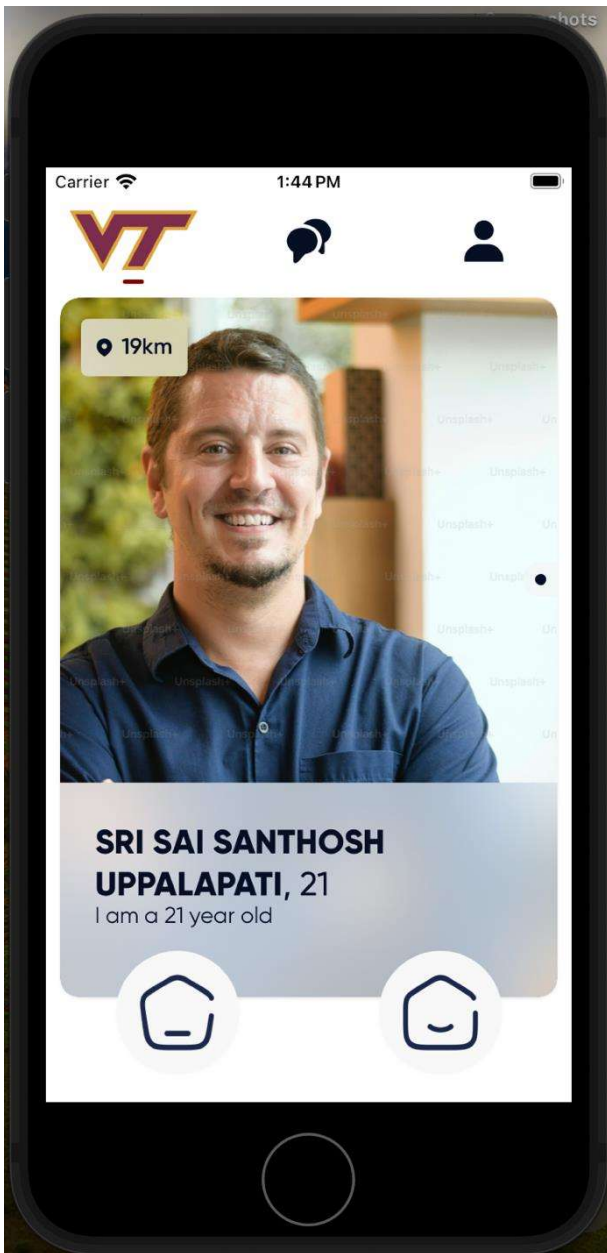


Fig 3.1

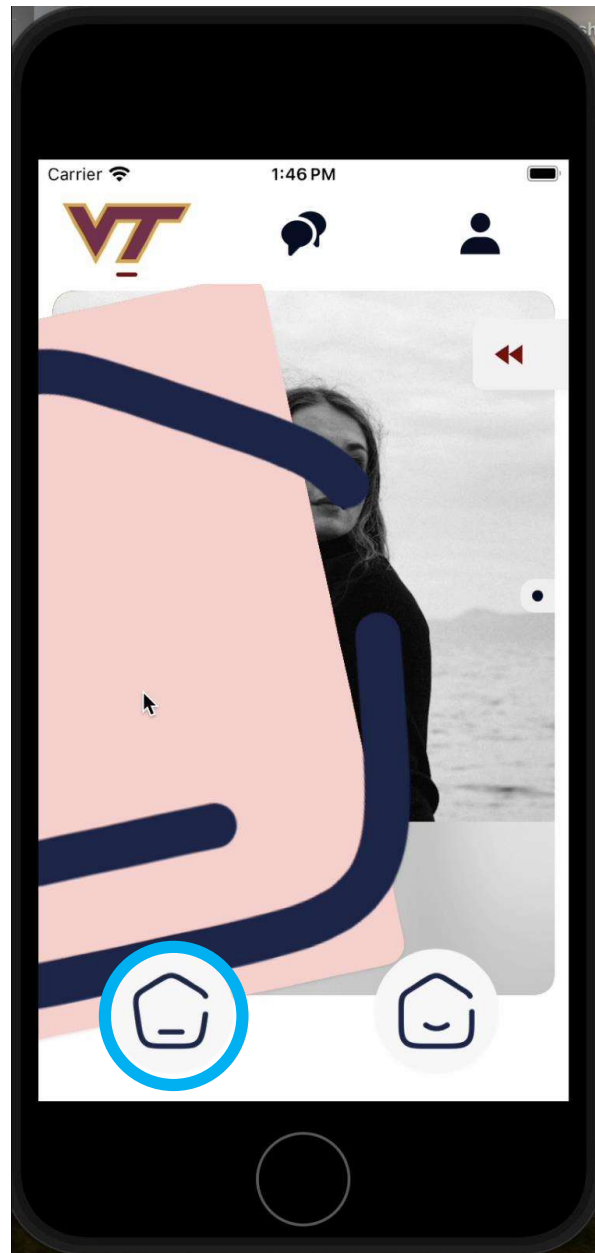


Fig 3.2

Fig 3.1. After creating a profile, the app provides the user with a recommended list of potential roommates based on their preferences. The user can view name, age, bio information and picture of the matched peoples.

Fig 3.2. If the user is not interested in the suggested people, they can swipe left or click the left button (marked in blue).

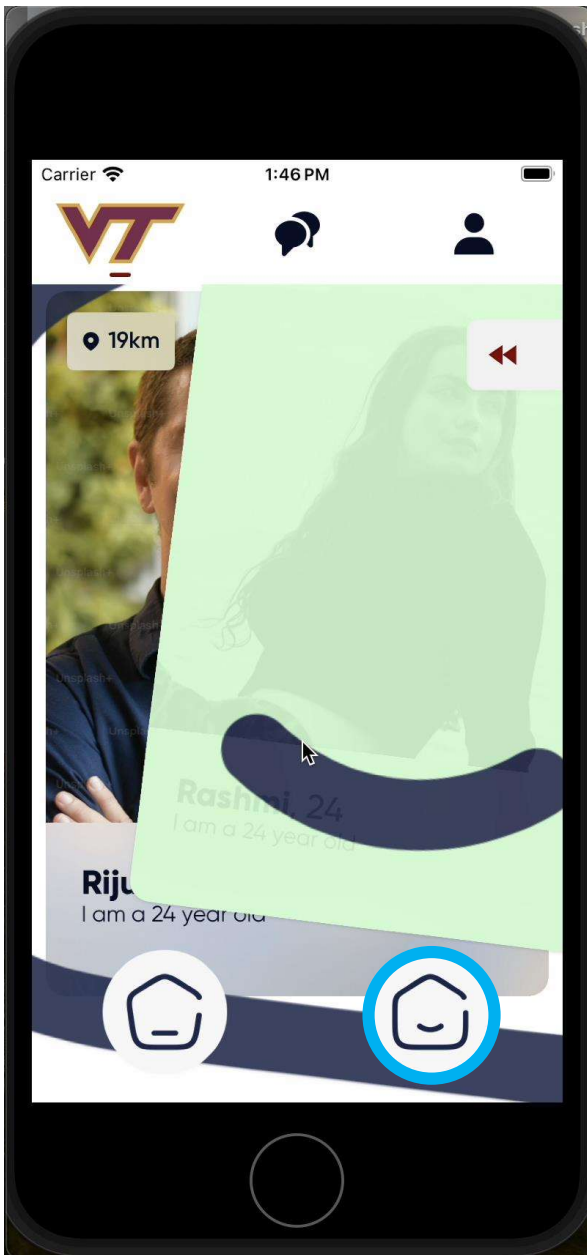


Fig 3.3

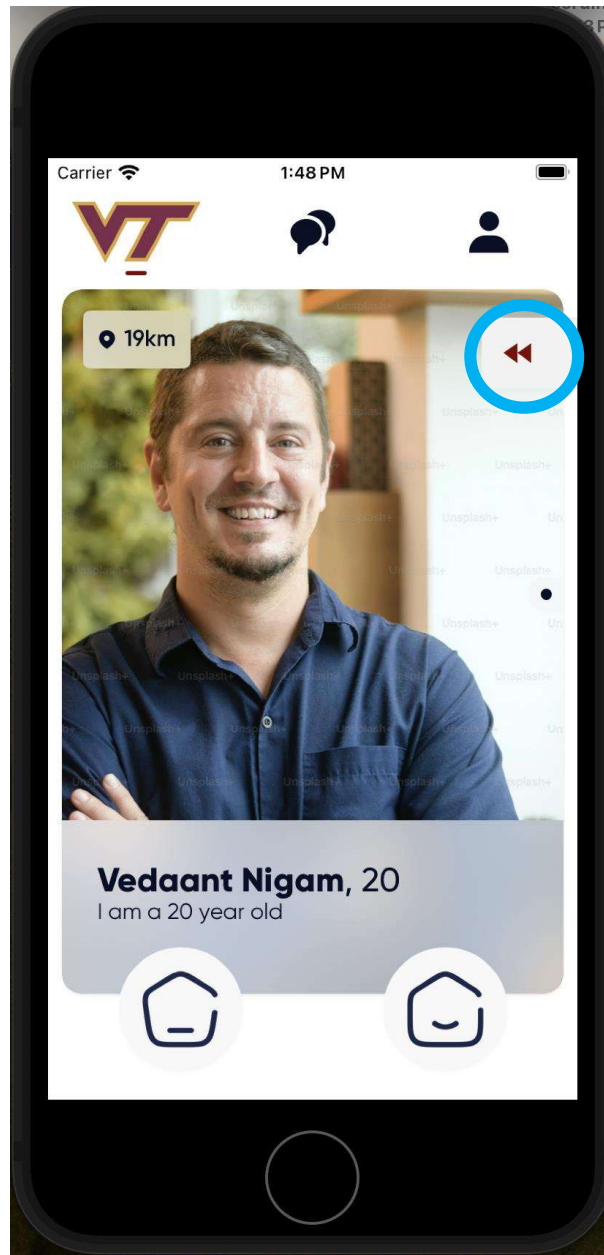


Fig 3.4

Fig 3.3. If the user is interested in the suggested person, they can swipe right or click the right button (marked in blue). After that, the user can start chatting with them.

Fig 3.4. If the user swipes or clicks the button incorrectly, the reverse button (marked in blue) allows them to undo their action and reselect the person they want to match with.

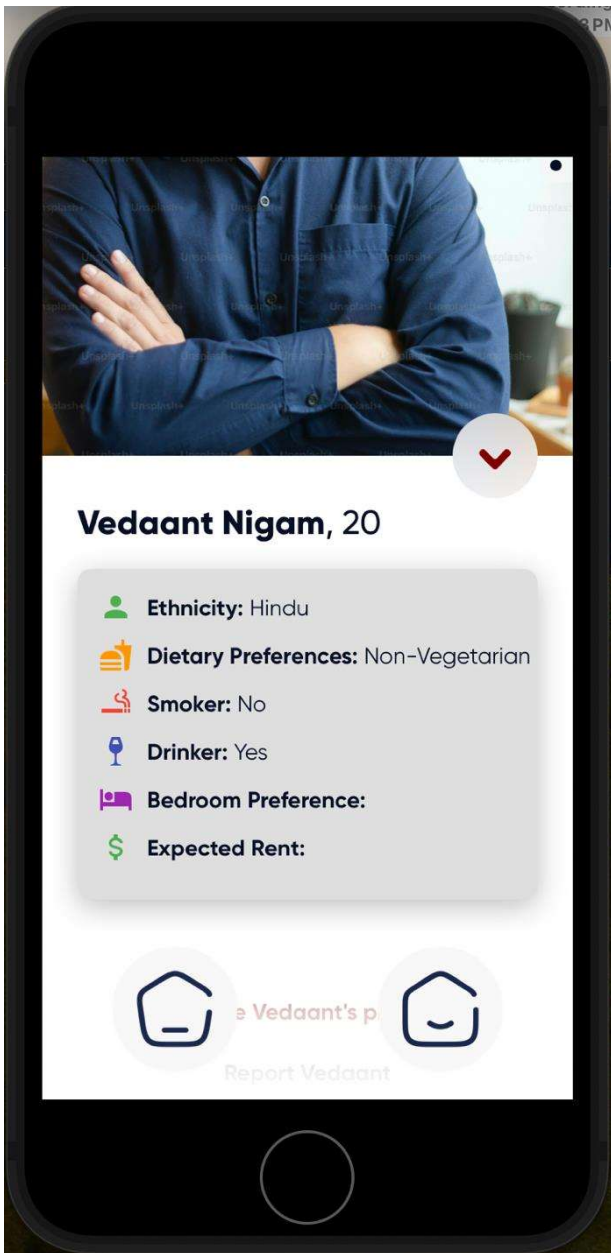


Fig 4

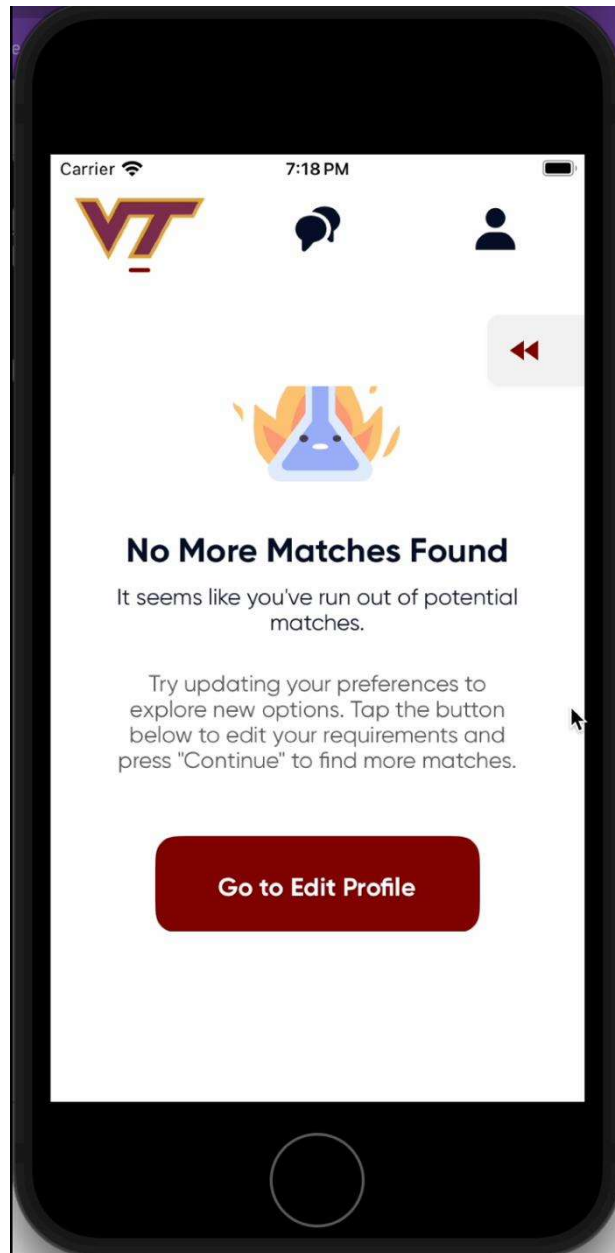


Fig 5

Fig 4. On the main page, which displays the roommate list, if the user wants to see detailed information about someone, they can click on that person's profile. This detailed information can help the user make an informed decision when choosing a roommate.

Fig 5. After the user has reviewed all the recommended roommate matches, they will see the new screen. On this screen, clicking the "Go to Edit Profile" button will navigate to a page where users can update their preference information. And then the user can receive a refreshed list of recommended roommates.

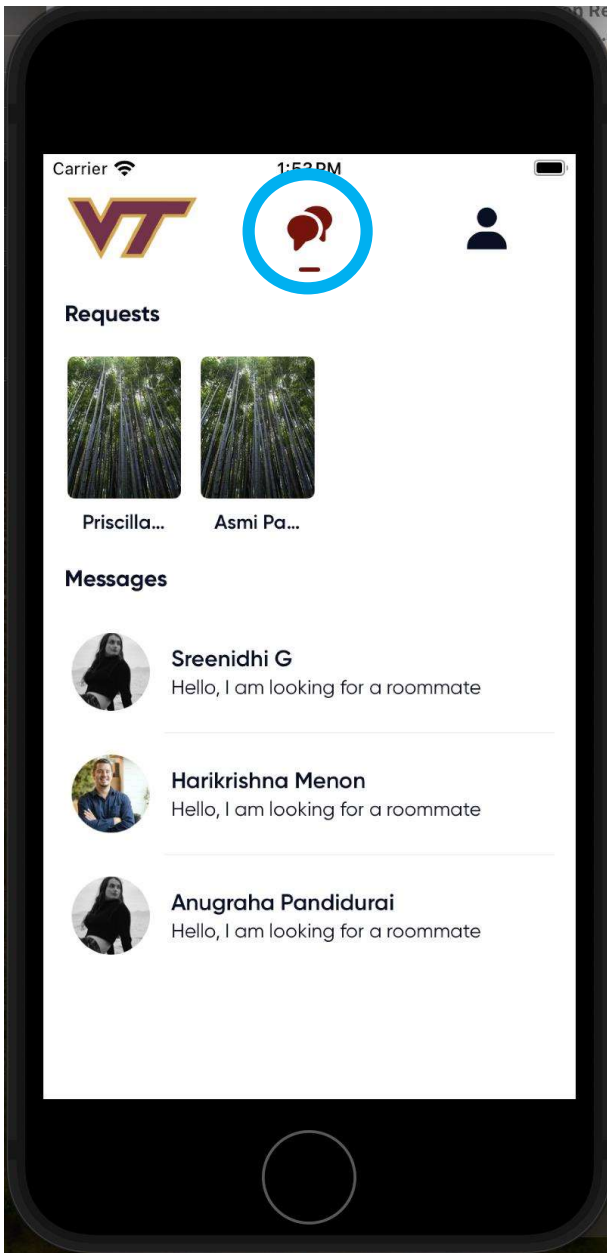


Fig 6.1

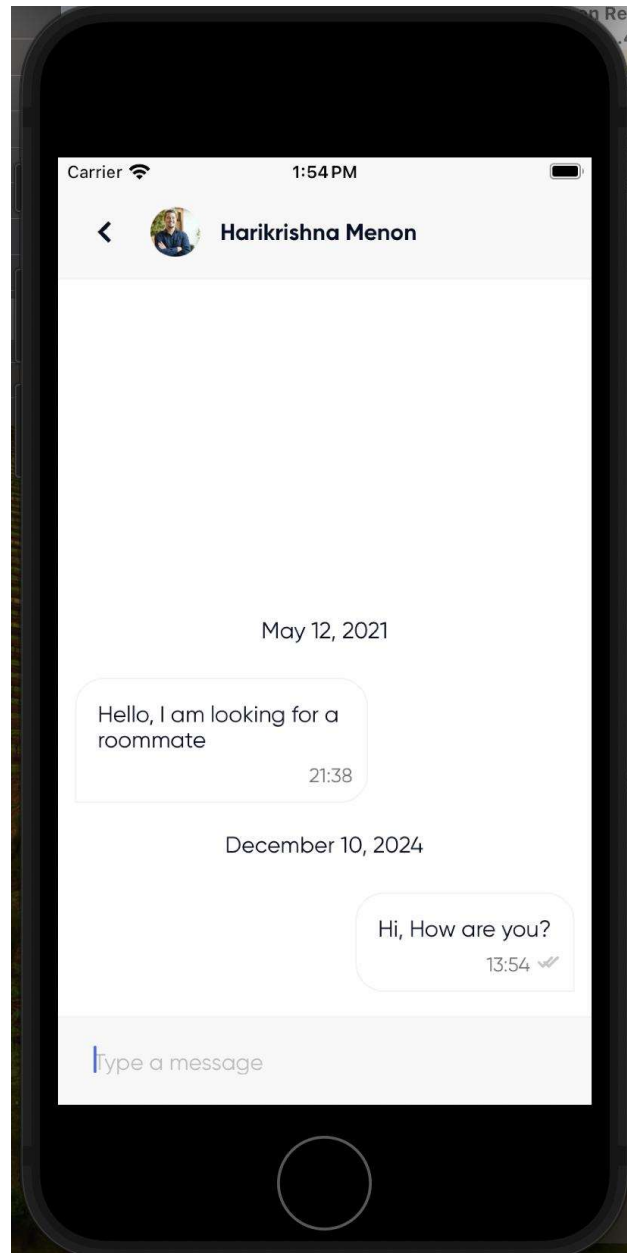


Fig 6.2

Fig 6.1. If the user clicks the button (marked in blue), they will be taken to the chat section. This section shows a list of people with their pictures and names, whom the user previously selected as potential roommates. The user can chat with them directly.

Fig 6.2. This is the chat screen. When the user clicks on a person in the list from 6.1, they will enter the 6.2 screen, where they can communicate to find the right roommate.

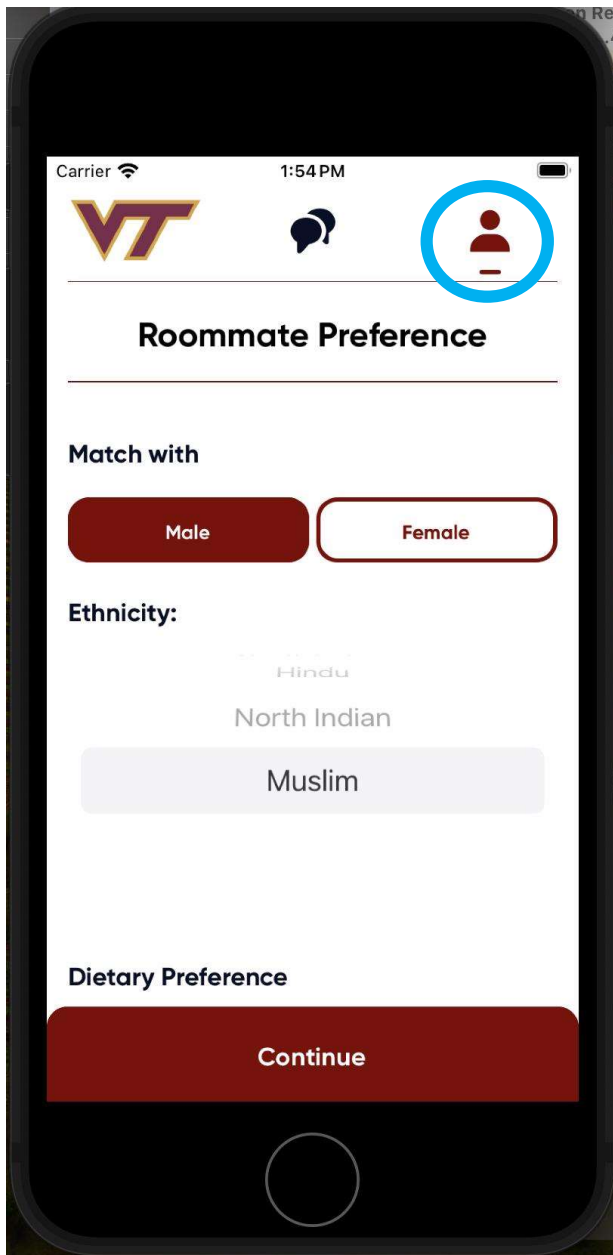


Fig 7

Fig 7. If the user clicks the button (marked in blue), they will be directed to the profile edit page. On this page, the user can update both their personal information and roommate preferences. After editing their preferences, they will receive a new list of matched roommates.

3. Design:

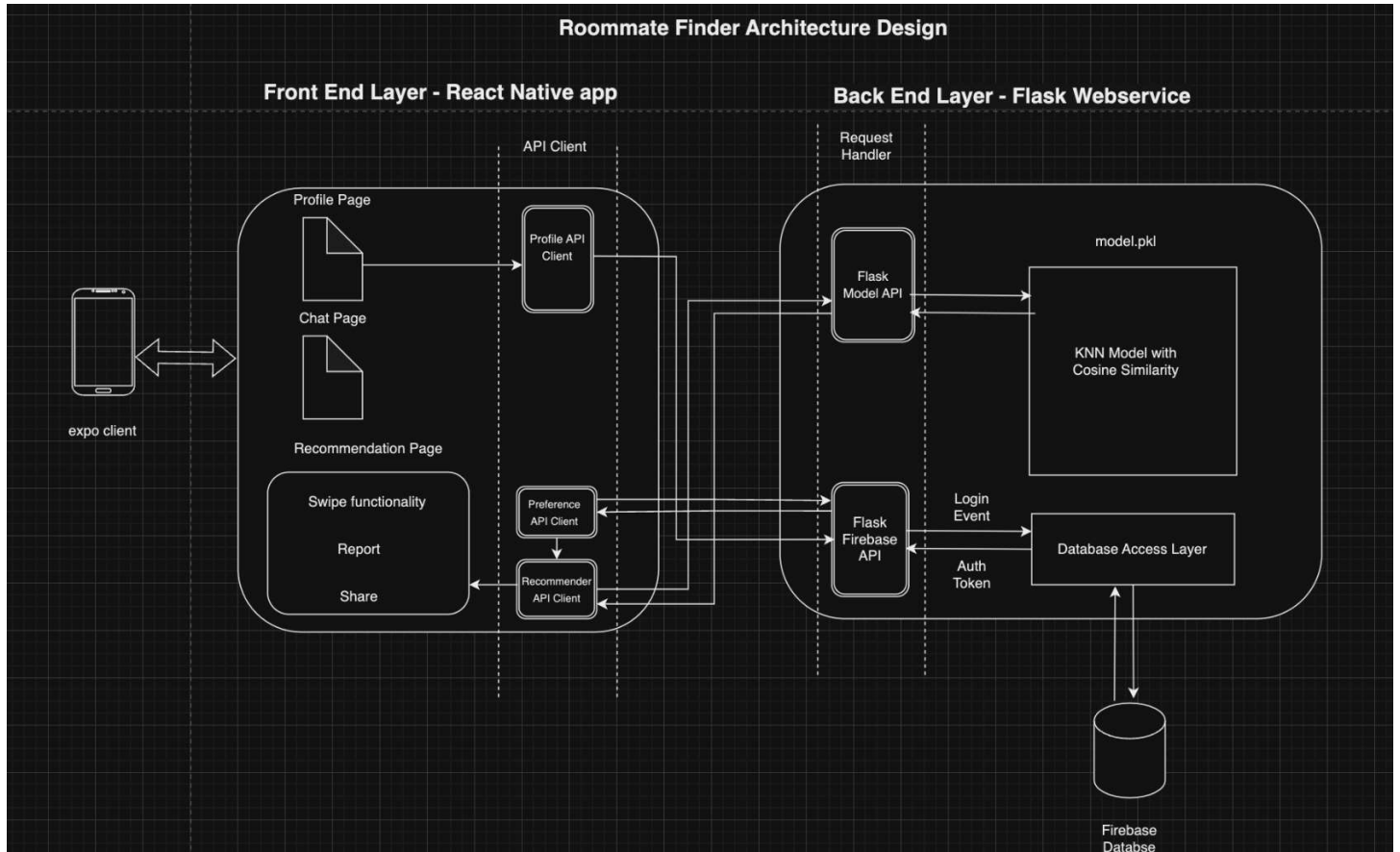


Fig 8: HokieFinder's Architecture Design

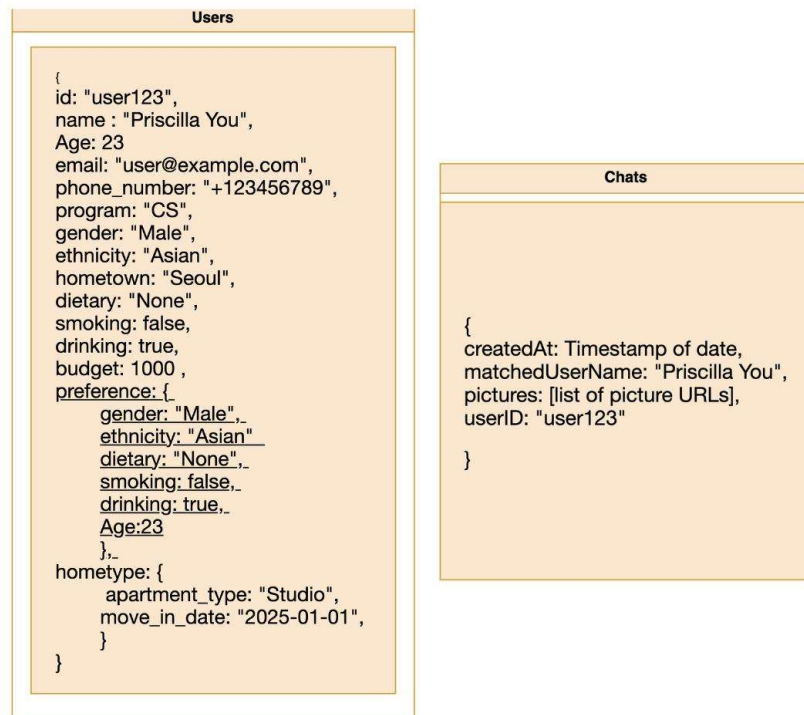


Fig 9: HokieFinder's Firebase Schema

1. Design Overview

The Roommate Finder application was developed to improve on the current options for local/international students of Virginia Tech DC area in finding roommates. This application, through intelligent recommendation algorithms, user-focused design principles, and a scalable architecture, will ensure that the matches are accurate but intuitive and engaging for the user. Inclusion of meaningful customization options further enhances the usability of the app, thus making it adaptable to a wide array of preferences and requirements.

2. Algorithm and Model Implementation

The Roommate Finder recommendation technique is based on the concept of the K-nearest neighbors' algorithm, enhanced by the cosine similarity criterion for user similarity, at a basic level. As a well-performing alternative, KNN was highly suitable, especially against the data volume of mere user profiles. Its interpretability marries well with the end goal of the whole project-targeting an intuitive system worthy of users' trust. Cosine similarity measures the angle between two feature vectors and functions well in high-dimensional data where the magnitudes may differ. In this application, user preferences ranging from drinking and smoking to budget to preferred ethnicities were encoded as feature vectors and compared to find their closest matches. Early iterations of the model showed low accuracy due to poor tuning of parameters.

After trying different hyperparameters, a tolerance of 2.0 was found that balanced precision and recall giving an overall accuracy rate of 80%. The implementation first preprocessed the users' data into numerical vectors, then applied the KNN algorithm, comparing feature vectors. Finally, it presented the top 10 matches for every user, making sure recommendations are relevant and meaningful.

3. Frameworks and Implementation

The Roommate Finder application is developed on the SOLID principle to drive home such code architecture that should be maintainable, scalable, and robust. In designing one easy-to-understand system, it should be pretty easy for the whole software module to take on single responsibilities. The application of Open/Close Principles has enabled the integration of new features in the code without alteration to old codes-for designs in view, this open extensibility but with modifications being closed-off for modification. For instance, there was seamless integration of in-app messaging. Segregation of the Interface' principle was observed. That is, not one component exposes, or API shows functionalities that are not needed to satisfy the purpose of those components; Second, the last dependency inversion was done in such a manner that high-level modules dependent on a lower abstraction are inverted by a form of abstraction at this level. That means changes in those lower frameworks or either Firebase or Flask would not break down their crucial functionalities. Following these and, hence, all SOLID principles mean the app will be far more long-term reliable and flexible.

3.1 Front-End Layer

This application's frontend was made with React Native, which was used because of its cross-platform compatibility by reusing code for both iOS and Android. Development is smoother thanks to Expo CLI that allows

one to debug quicker and has live previews while coding. It contains therein a Profile Page for preferences input and updates, and a Recommendation Page where suggested matches are shown and swiped for acceptance or rejection.

While the full implementation of the Chat Page is for future releases, the dynamic features which directly link with the recommendations engine take precedence in the present work.

Especially, the swipe functionality is directly bound to real-time recommendation data and not to static templates. Different UI designs were implemented to have a smooth and user-friendly interface.

3.2 Back-End Layer

The backend makes use of Flask, which is lightweight and modular; thus, ideal for API request management. It has been used for multiple API endpoint creations like Profile API to perform the CRUD on the user information, Recommendation API for fetching matches generated from the machine learning model, and Preference API, which dynamically updates the preferences of the users. Firebase was integrated to handle authentication and store data securely, while tokens were used to ensure safe communication between the front-end and back-end layers. The KNN model, saved as a.pkl file, is loaded at runtime to provide real-time recommendations. Extra attention was paid to the optimization of the recommendation pipeline such that the latency would be low, yielding fast and responsive user interactions.

4. Unique Features and Design Decisions

The Roommate Finder application is unique in many ways, considering the traditional roommate matching tools. The recommendation system-which has been developed over a customized implementation of KNN, using cosine similarity-suits particularly to the dataset and requirements of the user. Variability in different levels of tolerance makes this algorithm scalable and flexible regarding different applications. Firebase integration adds an additional degree of security and scalability with seamless management of authentication and storage.

Therefore, it saves the user lots of effort while giving them curated matches for partners instead. The dynamic functionality with swipe features, along with being able to update preference criteria in real time, results in an interactive app and website platform. The internal infrastructure has a modular architecture that provides leeway for scalability and adding up integrations in the times to come, such as live communications on-app for matched customers, or area-based filter enabling for hyper-local recommendations to crop in.

5. Challenges and Solutions

The development process had its challenges.

Early versions of the recommendation algorithm produced poor results in matching. This issue was improved by refining the user feature vectors and hyperparameter tuning, finally reaching an 80% accuracy rate. Firebase authentication using the Recommendation API also brought complex technical challenges in custom middleware for token management, thereby guaranteeing fluent communication among system components. These challenges were overcome through iterative testing and problem-solving.

Another significant challenge was the SDK version mismatch between the Expo Client and the React Native Animated library. This caused unexpected runtime errors and delays, particularly in features involving dynamic animations like swiping. Resolving this required upgrading the Expo Client to a compatible version from SDK 49 to SDK 51 and updating dependencies across the project to ensure stability. Thorough testing after the fixes guaranteed a seamless user experience with no disruption in app functionality.

6. What Makes Our Product Unique?

The Roommate Finder application leads by offering intelligent recommendations of quality matches according to user preferences. Firebase, Flask, and React Native blend together in strong harmony, giving an ecosystem that is bulletproof in guaranteeing data security, efficiency, and scalability. The app boasts a polished, intuitive user interface, a definite distinction from prior tools. It is modular in nature, thus allowing for enhancements to be made in the future, which makes the solution very forward-thinking in regard to roommate matching.

4. Ethical implications:

1. Our method meticulously takes into account ethical issues, especially those of security and data protection. Risks include misuse or illegal access when gathering user preferences and personal data. To combat this, we have implemented anonymization techniques to protect user identities, limited access to authorized personnel, and encrypted all data storage and transit. The safe handling of sensitive data and user confidence are guaranteed by these measures.
2. Another concern is bias in recommendations from our ML model, which could unintentionally lead to unfair or discriminatory outcomes. To address this, we trained our model on diverse datasets and regularly audited its outputs to ensure fairness and inclusivity in roommate matching.
3. Lastly, preserving trust depends on user consent and openness. While our platform currently gathers and uses data without explicit user consent, we recognize the importance of providing students with alternatives to opt out of data gathering or have their information deleted if they desire. Going forward, we aim to incorporate clear terms of service and privacy policies that explicitly explain how user data is handled. These steps guarantee that our platform functions morally while putting user confidence and inclusion first.

5. Retrospection

1. What went well:

- The team worked cohesively, dividing the tasks efficiently and maintaining clear communication throughout the project.
- Collaborative sessions led to the successful implementation of features like swipe-based functionality and ML-powered recommendations.
- Building and optimizing the ML model for personalized recommendations was a key success.
- Integration of the swipe functionality into the app provided an intuitive user experience.
- Adhering to timelines and meeting project milestones allowed the team to deliver the core features on - time.
- Debugging and testing were emphasized to address compatibility problems in React Native.

2. What went wrong:

- Faced issues on accessing user data making sure the interface respected student privacy, especially when designing profile sections.
- Issues with React Native app not functioning as expected.
- Challenges arose during component integration due to varying development speeds and differing opinions on app's design and functionality.
- Combining the ML model with the app's interface was more complex than expected, requiring multiple iterations and extra effort from the team.

6. Recommendations for future

1. Hokie Finder's future development can be guided by several suggestions, guaranteeing that it becomes a valuable and enduring resource for the Virginia Tech community. First, launching the app on major platforms like the Google Play Store and Apple App Store will expand accessibility, making it easier for all Hokies to use regardless of location.
2. Secondly, increasing the app's personalized flat mate ideas would ensure a tailored experience, addressing the inadequacies of websites like Apartments.com.
3. Third, engagement with the Virginia Tech NCR Campus may foster creativity, reduce reliance on costly external platforms, and provide technological expertise.
4. Making Hokie Finder an open-source project will also allow students and community members to contribute new ideas, improve algorithms, and offer new features, creating a community-driven approach.

Finally, broadening the app's scope to target more general housing issues, such as affordability and accessibility, will boost its value and relevance. These initiatives will not only improve the app but will also establish Hokie Finder as a comprehensive housing solution that matches the changing demands of the Virginia Tech community.