**Domain Model: TheLabRats**

| **Class:** | **Attributes:** | **Functions:** |
| --- | --- | --- |
| User | * UID: String * Token: String * Password: String * Email: string | > Login(UID, Password, Token)  > Logout()  > verifySession(token)  > setUser()  > getUser()  > deleteUser()  > verifyEmail(Email) |
| Profile | * UID: String * Name: String * D.O.B: String * Age: Int * Sex: String * Biography: String * Pictures[5]: .jpg, .png * Preferences: Object * AccountType: Object * Address[2]: Object * LinkedUserID: String * AccessibilityReqs: String | > setProfile()  > getProfile()  > verifyAge(D.O.B)  > linkUser(UID, LinkedUserID)  > unlinkUser(UID, LinkedUserID)  > addPictures(Pictures[5]) |
| Regular  (Account type 1) | * UID: String * HasAddress: Boolean * Job: String * Rent: float * CriminalHistory: Boolean * CriminalBackground: String * Pets: Boolean * Smoker: Boolean | > setRegular()  > getRegular()  > updateRegular(UID) |
| Student  (Account type 2) | * UID: String * University: String * Grade: Int * HasDorm: Boolean * Major: String * Hobbies: string * Religion: String | > setStudent()  > getStudent()  > updateStudent(UID) |
| Address | * UID: String * Address: String * Town: String * State: String * Zip: int * Country: String | > setAddress()  > getAddress()  > updateAddress(UID, Address, Town, State, Zip, Country) |
| RegularPreferences | * UID: String * HasAddress: Boolean * Job: String * Rent: float * CriminalHistory: Boolean * CriminalBackground: String * Pets: Boolean * Smoker: Boolean | > getPreferences()  > setPreferences()  > updatePreferences() |
| StudentPreferences | * UID: String * University: String * Grade: Int * HasDorm: Boolean * Major: String * Hobbies: string * Religion: String | > getPreferences()  > setPreferences()  > updatePreferences() |
| Whitelist | * UID: String * OtherUID: String | > setWhitelist()  > getWhitelist()  > notifyOtherUser(UID, OtherUID)  > updateWhitelist(UID, OtherUID) |
| Blacklist | * UID: String * OtherUID: String * BlacklistTime: float | > setBlacklist()  > getBlacklist()  > updateBlacklist(UID, OtherUID, BlacklistTime)  > blacklistTimer(OtherUID) |
| Matches | * UID: String * OtherUID: String | > setMatches()  > getMatches()  > updateMatches(UID, OtherUID) |
| ChatRoom | * ChatroomUID: String * SenderUID: String * RecieverUID: String * Timestamp: Float * Message: String * IsMatched: Boolean | > createChatroom(ChatroomUID, SenderUID, RecieverUID, Timestamp)  > sendMessage(ChatroomUID, SenderUID, RecieverUID, Message, Timestamp)  > verifyMatches(SenderUID, RecieverUID) |

**\*Refer to** [**TheLabRats\_Deliverable\_3\_[DomainModelDiagram].pdf**](https://drive.google.com/file/d/1LumUG09IO6rOEC5cdDP9kFeLA_ZASoHz/view?usp=share_link) **in the Deliverable Folder for the Domain Model Diagram.**

[**https://drive.google.com/file/d/1LumUG09IO6rOEC5cdDP9kFeLA\_ZASoHz/view?usp=share\_link**](https://drive.google.com/file/d/1LumUG09IO6rOEC5cdDP9kFeLA_ZASoHz/view?usp=share_link)

**Domain Model Description:**

***Users, Accounts, Preferences:***

Everyone who signs up with RoomUp will create a user instance, which will generate tokens, their UID and password. With this class, users can log in and log out of the application. However, having a user instance is not enough, as they will need to set their profile once they create a user (after initially signing up with their email and creating a password). Users can set their profile, verify their emails, and delete their user.

Once the user signs up, they will set their profile, which includes attributes such as name, date of birth, sex, biography, etc. This instance also allows users to include up to five profile pictures which will later be used for swiping. Additionally, users will have other instances of an address (if they already have one), account types (student versus regular), and preferences according to account type. In this class, users can set their profile and will need to verify their age. They will also have the option to link or unlink other profiles if multiple people who already live together are trying to find another roommate.

There are two account types, the first being on campus students, and the second being off campus students/regular users. For each account type there will be different questions. For example, regular users will not be in dorms, but will either have apartments, condos, or renting a house, so their questions will ask if they do or don’t have a place to live as of now, occupation, how much rent they are willing to pay, and criminal history, if they smoke, and if they have pets. We will not ask on-campus students these questions as many don’t apply to dorm lifestyles (smoking is illegal on campus, pets are not allowed in dorms, rent does not apply, and universities do background criminal history reports on students). However, questions that apply more to on-campus students than regular users include: what university they go to, their grade, if they are already in a dorm or not, their major, hobbies, etc.Many dorms are organized by major.

The regular and on-campus student preferences ask the same questions as what we asked them in their account type creation, but are specifically geared towards qualities they want their roommate to have. We would not pair on-campus students with regular users as that would be problematic, so there needs to be separate preferences for on-campus versus off-campus roommates. We simplified the names of the classes to be regular versus student to keep the class names shorter and less complicated. Finally, we have another class for addresses if the user has one, as there are many aspects to creating an address such as the address line, town/city, state, zip, and country. In rare instances, some people might have multiple instances of addresses, but we only allow up to two instances.

***Chat Room, Swipes, and Matches:***

We have another class for chat rooms, matches, whitelist, and blacklist. It starts with a swipe in the UI, in which a user will swipe up or down if they like a potential roommate. If they swipe down, the other user will be sent to a blacklist of roommates so they don’t show up on the original user's swipe feed in the future after a certain amount of time. If a user swipes up on someone, then that person will be added to a whitelist of potential roommates. Once a user is added to a whitelist, the program will notify the whitelisted user to see if they want to swipe up with the user who added them. If not, then the program will remove the user from the original user’s whitelist and put them on the blacklist. We have a specific function that calculates the necessary time that a user will remain in another user’s blacklist.

The class that specifically creates an instance for all the matches a user has is named Matches. In this class you can set matches, get matches, and update the matches list. This class is different from the whitelist class, as that class has many instances of possible matches, while the matches class is definite matches. One user will have multiple matches, and thus multiple instances of the Matches class. This class will later be used for our final class listed: ChatRoom. A user will have multiple chat rooms with other users. These other users can only be matched, thus a need for the function verifyMatches(), to make sure when a new chat room is created, the two users are matched. This class will contain attributes for a chatroom UID, SenderUID, RecieverUID, timestamps, messages, and the result from the verifyMatches function.This will allow us to store multiple instances of chat rooms for each user.