University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: ___1__

Date: _2024-02-09__

Group Number: ____94____

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Aaditya Desai	41381799	w8g6e	desai.aaditya.d@gmail.com
Aryaan Habib	11833316	z2a9c	habibaryaan@gmail.com
Devin Proothi	68619774	l9q5j	devinpr2021@gmail.com

<u>MILESTONE 1: SWIPEMATES</u>

A brief project description answering these questions:

a) **The domain of an application refers to the area of knowledge your application resides in. For example, if I am making an application for a hospital, the domain would be something like healthcare/patient management/logistics (it would depend on what the application is trying to do).**

- The domain of the application is student roommate matching, specifically designed to address the challenges faced by the University of British Columbia students in finding compatible roommates for shared accommodation on and off campus. It aids the process of finding roommates by allowing students to create profiles, and match with other students based on their entered preferences and requirements. The application aims to simplify the roommate search process, improve the living experience, and foster a sense of community among UBC students.

b) **What aspects of the domain are modeled by the database? In answering this question, you will want to talk about what your project is trying to address and how it fits within the domain. It is likely that in the process of answering these questions you will bring up examples of a real-life situation in the application could be applied to.**

- The database models several aspects of the domain to facilitate the matching process and enhance the overall experience of UBC students seeking compatible roommates. The various aspects of the domain are:

1. **User Profiles**: The database captures user profiles, including all the personal information like name, age, etc. This aspect allows students to create impressive

profiles that reflect their personalities and unique characteristics that other profiles might show interest in. This helps students not only find housing, but also meet potential roommates with common interests and therefore higher compatibility, while adding a fun twist to meeting other students and finding the perfect match.

2. **Preferences and Requirements:** The database stores information about each user's roommate preferences and requirements. Users can specify criteria such as gender, age range, location preferences, and rent. Based on these choices matches are made. Requirements are filters set by the user for attributes that are absoultely essential to them while preferences are attributes they would like to have in their future roommates but their absence are not deal-breakers. Both are determinants of the compatibility score.
3. **Matching:** The database allows the user to swipe right and left on other users after looking at their profiles, right meaning that they are interested and left meaning otherwise. Once two users swipe right on each other a match is made from where the users are provided with each other's socials.
4. **Notification and Compatibility Score:** Every match is accompanied by a notification and compatibility score. When a new match is found, the user gets a notification with details about the match. This is followed by a compatibility score and a message, which is calculated by comparing the preferences and requirements of the two users in the match.

– As UBC students ourselves, all three of us found it extremely difficult to look for housing in Vancouver and then looking for compatible roommates, especially in our second and third years where on-campus housing is not guaranteed. This is primarily where the idea for this project stems from since it makes finding both residence and roommates easier for not just students looking for off-campus housing but also first-year students coming to college for the first time searching for on-campus roommates.

Database specifications: (3-5 sentences)

a) **What functionality will the database provide? I.e., what kinds of things will people using the database be able to do?**
   - The database will provide functionality for facilitating the roommate matching process among UBC students. Users will be able to create detailed profiles, specify their preferences and requirements, and search for roommates based on these choices. The database will store all the user profiles created by students, their needs and preferences, their social and contact information, matches between students and even their compatibility scores. (Note: We plan on adding functionality for students to chat with each other instead of providing socials if we have time to incorporate the feature).

Description of the application platform: (2-3 sentences)

a. **What database will your project use (department-provided Oracle, MySQL, etc.)? See the "Project Platforms" section of this document for more information.**
   – As a group we have not yet finalized the RDBMS we will be using in our project. As of now our first choice would be MySQL, although there is a possibility to switch to Oracle (department provided) depending on what all members are more comfortable with.

b. **What is your expected application technology stack (i.e., what programming**

languages and libraries do you want to use)? See the "Project Platforms" section of this document for more information.

i. You can change/adjust your tech stack later as you learn more about how to get started for the project via latter tutorials

- Like our RDBMS, the tech stack to be used in our project is not finalized and is subject to change in the future depending on if we end up choosing to develop a web application, desktop application or android application. For now, we have boiled down to the following final choices and will pick one of each:
  - Frontend:
    - HTML/CSS/JavaScript
    - Python Libraries (tkinter/PyQt/PySite)
    - Java (Swing)
  - Backend:
    - Node.js
    - Django
    - PHP

An ER diagram for the database that your application will use. It is OK to hand-draw it but if it is illegible or messy or confusing, marks will be taken off. You can use software to draw your diagram (e.g., draw.io, GoogleDraw, Microsoft Visio, Powerpoint, Gliffy, etc.) The result should be a legible PDF or PNG document. Note that your ER diagram must

use the conventions from the textbook and the lectures. For example, do not use crow's feet notation or notation from other textbooks)

- **Entities (with attributes)**

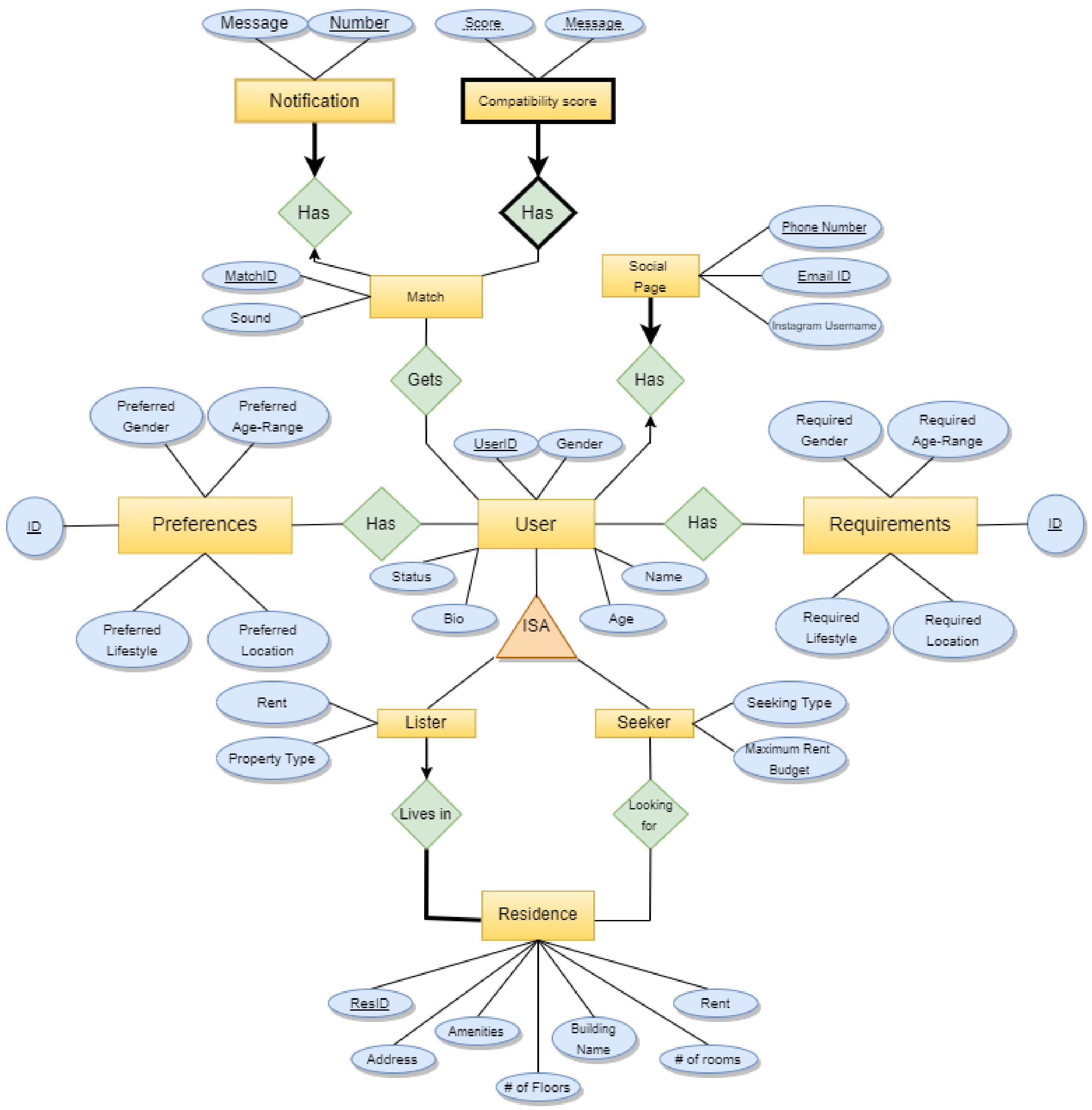
1. **User** (Primary Key: UserID, Name, Gender, Status, Bio, Age)
2. **Preferences** (Primary Key: ID, Preferred Gender, Preferred Age-Range, Preferred Lifestyle [non-smoking, non-alcohol], Preferred Location)
3. **Requirements** (Primary Key: ID, Required Gender, Required Age-Range, Required Lifestyle [non-smoking, non-alcohol], Required Location)
4. [User ISA] **Lister** (Rent, Property Type)
5. [User ISA] **Seeker** (Seeking Type, Maximum Rent Budget)
6. **Residence** (Primary Key: ResID, Address, Amenities, Number of Floors, Building Name, Number of Rooms, Rent)
7. **Socials Page** (Primary Key: Email ID, Primary Key: Phone Number, Instagram Username)
8. **Match** (Primary Key: MatchID, Sound)
9. **Notification** (Primary Key: Number, Message)
10. [Weak Entity] **Compatibility Score** (Primary Key: Score, Message and MatchID)

(NOTES: All the attributes are not finalized and might get changed during development)

- **Relationships (cardinalities)**

  - **User-Preferences (Many to Many):** One user can have many preferences and many users can have the same preference.

- **User-Requirements (Many to Many):** One user can have many requirements and many users can have the same requirements.
- **User-Match (Many to Many):** One user can have various matches with other users and one match requires exactly 2 users.
- **User-Socials Page (One to One):** One user will only have one social page, and a social page is unique to every user.
- **Lister-Residence (Many to One):** One lister lives in only one residence, but one residence can have multiple listers.
- **Seeker-Residence (Many to Many):** One seeker could be looking for multiple residences at the same time, and one residence can have multiple seekers looking for it.
- **Match-Notification (One-to-One):** Every match gets one notification, and every notification is unique for a match.
- **Match-Compatibility Score (Weak Entity Relationship):** Every match gets a compatibility score, and multiple matches can have the same score.


Message
Number
Score
Message
Notification
Compatibility score
Has
Has
Phone Number
MatchID
Social Page
Email ID
Match
Sound
Instagram Username
Gets
Has
Preferred Gender
Preferred Age-Range
UserID
Gender
Required Gender
Required Age-Range
ID
Preferences
Has
User
Has
Requirements
ID
Status
Name
Bio
ISA
Age
Preferred Lifestyle
Preferred Location
Required Lifestyle
Required Location
Rent
Lister
Seeker
Seeking Type
Property Type
Maximum Rent Budget
Lives in
Looking for
Residence
ResID
Rent
Amenities
Building Name
Address
# of rooms
# of Floors