Team 10 - Qiwen Luo, Gary Yao, Peifeng Hu, Ha Do

FINAL PROJECT REPORT
“Hi, Stranger”

1. General description:

“Hi, Stranger” is an online dating website that helps users find their other half by meeting compatible people based on their criteria for a partner. Users can create their account by uploading their picture, fill out their personal information such as name, age, gender, occupation and wage, and their criteria for a partner regarding gender, age, wage, and occupation. Then the system will introduce potential partners based on the users’ criteria. Users can look at their potential partners’ profiles and choose to “like” these people, then the users’ matchlist will be updated every time they “like” someone. Then matched users can chat to get to know each other and start dating if possible.

2. How we fulfill the requirements:

- Technology requirements:
  - Our application server is able to run on the class Linux server, and the application database is MySQL: Through PuTTY, we can log in the class Linux server eecslab-9 using our Case ID to get access to our project’s MySQL database. Then we can use SQL queries to view the tables and interact with the data from them.
  - The user interface runs in common web browsers such as Chrome and Firefox.
  - There are no back-end frameworks.
- Database interaction requirements:

- Our application can create, read, update, and delete data:
  - Create: Our application 1) creates 7 tables to store data for this website: user, tomatch, photo, hasphoto, matchtable, aggregate, liketable 2) insert data when user sign up or like other users
  - Read: Our application can read the data in the tables in order to 1) give out potential people for liking or matching 2) check whether the username is unique when creating an account 3) check whether username and password are matched when signing in 4) perform calculations in aggregate functions, such as the total number of users, the average age and wage of the users.
  - Update: Our application can update data in the tables, such as 1) data in “user” can be updated when the user changes his information (username, password, wage, etc.), data in “matchtable” can be updated when a user

has a new matched user, and so on. 2) data in aggregate table when a new user sign up or an old user deletes his/her account
    - Delete: Our application can delete data in the tables, such as 1) information about a matched user in the table “liketable” can be removed when a user “unlikes” someone 2) all information of a user when he/she chooses to delete account
- Our application includes some queries that utilize aggregate functions in SQL: The website administrators can see the application’s calculation of the total number of users, the average age and wage of the users.

3. What we did beyond the requirements:
    - We filter users to give out potential people for liking/matching according to user’s requirement.
    - We filter users to give out a list of people who have already been liked by the current user.
    - Everytime a user signs up, we filter our user table to make sure that none of our users has that username and email.
    - We use validator in every form user submit to make sure all the input information is in correct format. Therefore, we can increase the rate of successfully matching with somebody when we filter the potential people for liking.
    - We use CSS to make our web pages look better.
4. Contribution by each team member:
    · Qiwen Luo: Front-end development: Front-end planning, interface design, client-side coding, interface documentation
    · Gary Yao: Database administration: Data modeling, data modeling implementation, retrieving data from Wikipedia, query writing, database documentation
    · Peifeng Hu: Back-end development: Server planning, software design, server-side coding, server documentation
    · Ha Do: Project management: Process flowchart, E-R diagram and relational model design; report writing; presentation planning

5. What we have learned from the project:
    a. Technical skills:
        - Front-end development: Basic syntax of HTML5, sending/receiving data in HTML, using CSS to make the webpage look better

        - Back-end development: flask, SQLAlchemy, using the flask to connect frontend and database, and use SQLAlchemy to connect the database

and modify the data in database, the backend will also monitor current user to make sure all the action are done based on current user's wishes.

b. Planning skills:
    - Instead of starting frontend, backend and database simultaneously, we should finish frontend and database first. Because there can be mistakes when the variable names in frontend or database do not match the variable name in backend
    - Our plan could have been more detailed so that front-end, back-end and database design can accomplish the same functions at the same levels. Thus, we can spend less time integrating the three parts.