Project Reflection Report (Team 92)

Demo Video:
https://drive.google.com/file/d/1jaycFVbYDr2dQFj8kj-fqUPOhp8HdEYv/view?usp=sharing

1. Please list out changes in the directions of your project if the final project is different from your original proposal (based on your stage 1 proposal submission).
   - We decided against implementing the daily habits checklist for users.
   - We refined the UI design of the application for improved user experience.
   - Our current user information editing functionality only updates their name, while our original plan was to allow changes to more fields.

2. Discuss what you think your application achieved or failed to achieve regarding its usefulness.
   Our application successfully enabled users to input characteristics of what they were experiencing in terms of symptoms, conditions, etc., and gain insights based on others' experiences. We implemented a successful input functionality that took into account the experiences of people with the symptom or condition, allowing users to input information and adjusting our information accordingly. One area where our application could have improved is providing users with assistance in understanding the information presented to them about their illness, instead of merely listing it.

3. Discuss if you changed the schema or source of the data for your application.
   We initially considered a different dataset for our project, which focused on using roadmaps to find the optimal travel route for users. However, we realized that this proposal would not require much database manipulation. We then switched to our current project and developed a new proposal for an application that helps users share their experiences with certain medical conditions. After a deep look into the new dataset, we found out that the data on treatment, weather and other factors are quite unclear, and the most significant parts are only the conditions and symptoms. Therefore, we decided to only focus on those parts.

4. Discuss what you change to your ER diagram and/or your table implementations. What are some differences between the original design and the final design? Why? What do you think is a more suitable design?
   The ER diagram remained largely consistent with our original design, with only a few minor adjustments made during implementation. For example, we renamed the "Illnesses" table to "Conditions" to ensure consistency in naming conventions. We also altered the data types of certain attributes from 'int' to 'VARCHAR(255)' and vice versa to better reflect the nature of the data being stored. Additionally, we introduced a unique user ID for each user profile to address potential issues related to users sharing the same first and last names. This modification allowed us to better manage user data and prevent accidental modifications to multiple users with the same name. The final design proved to be more suitable as it provided a clear and logical representation of the data, allowing for efficient querying and data management while adhering to the project's requirements and assumptions.

5. Discuss what functionalities you added or removed. Why?
   During the development process, we made a few changes to our application's functionalities in order to improve efficiency and better address the project requirements.

   Added functionalities:

   - Update functionality: We realized that we had overlooked the need to update data in the initial design. To address this, we introduced update and delete conditions in our SQL CREATE TABLE statements by using the ON DELETE CASCADE and ON UPDATE CASCADE functions.
   - Dataset filtering: The original dataset included 9,000 conditions and 23,000 symptoms, which could have slowed down the application. To enhance performance, we decided to focus on the top 1,000 conditions and symptoms.
   - Security enhancements: We realized the importance of protecting user data, so we implemented additional security measures such as input validation, secure password storage, and protection against common web application vulnerabilities.

   Removed functionalities:

   - Daily habits checklist: We initially planned to include a checklist of daily habits that might affect users' health, but we decided to remove this functionality as it required a more extensive dataset and greater complexity in data management.

   These adjustments helped us better adhere to the project requirements while improving the overall efficiency and user experience of the application.

6. Explain how you think your advanced database programs complement your application.
   Our advanced database programs allowed our application to efficiently retrieve the information users were searching for. Given certain constraints, we provided helpful and relevant information to users about their designated symptoms or conditions. Other functionalities, such as adding a trigger, enabled the updating of displayed information based on other users' inputs in real-time. As a result, the database programs facilitated an informative and collaborative experience for users.

7. Each team member should describe one technical challenge that the team encountered. This should be sufficiently detailed such that another future team could use this as helpful advice if they were to start a similar project or where to maintain your project.

   Meghna: Developing the data management feature posed a challenge in efficiently querying and updating the database while maintaining data integrity. We addressed this by implementing database transactions and using appropriate locking mechanisms to prevent data inconsistencies.

   Hsiang-Yin: Initially, the Updating and Deleting function on the USER data table was based solely on the user's first name. However, after receiving feedback from TA, we discovered that this method could result in modifying multiple users with the same first name simultaneously.

The task of rectifying this issue was not particularly challenging, but it did require careful consideration to avoid similar problems in the future when building a platform like this. To solve the problem, I updated the code to modify data based on both the first and last names of the user. Another approach could be modifying the data using the unique user ID.

Shreshta: One challenge we faced was creating queries that are useful to the user using the data that was present. Since not all the tables were filled sufficiently, our results did not provide a useful display for the user. After inputting more data for the queries to work with however, the results became more meaningful.

Richard: We faced challenges with getting a good advanced SQL query and we had to change it once because it wasn't making much sense when we tried to integrate it into the frontend. This also made us have to redo the indexing analysis. So I would recommend that for future groups they should think carefully about the meaning of their advanced queries so they don't have to repeat work.

8. Are there other things that changed comparing the final application with the original proposal?
   Throughout the project, there were minor adjustments to the user interface and feature set to improve usability and user experience. For example, we refined the symptom input process to be more user-friendly and added more interactive visualizations to display data more effectively. We also realized the need for additional security measures to protect user data and ensure the privacy and safety of our users and implemented security enhancements accordingly to prevent common web application vulnerabilities.

9. Describe future work that you think, other than the interface, that the application can improve on.
   There are several areas where the application can be improved:
   - Incorporate machine learning algorithms to predict potential chronic illnesses more accurately based on user data.
   - Expand the database to include more diverse and comprehensive data on chronic illnesses, treatments, and environmental factors.
   - Enhance the user experience with personalized health recommendations based on their individual profiles and data.
   - Improve the data visualization to better represent complex patterns and trends.

10. Describe the final division of labor and how well you managed teamwork.
    The final division of labor remained consistent with the original proposal, with each team member taking responsibility for their respective features. To manage teamwork effectively, we relied on messaging and discussion during extra time at the end of class, as well as occasional Zoom meetings, to stay on track and work effectively as a team. By sharing updates and discussing any issues that arose, we were able to ensure that we all stayed on the same page. We also used Git for version control, which allowed us to collaborate on the codebase effectively and merge changes from different team members.