Final Project Report: Intelligent Math Exercise Management Platform

1. Changes in Directions from the Proposal

During the initial phase, our plan was to develop a personalized math exercise management platform capable of dynamically recommending related math concepts and generating exercises based on user input. However, due to time and resource constraints, we made the following adjustments:

- **Data Sources**: While the initial plan involved using real user data for account creation and activity logs, we opted to generate virtual user data. This decision was made to save time and protect privacy.
- **Scope of Implementation**: We successfully implemented user login, exercise filtering, note creation, answer verification, history and stars. However, due to time constraints and concerns about the complexity of the project, features such as dynamic difficulty adjustment were not realized**.**

Achievements:

- Successfully implemented user account management (login and registration) and exercise filtering functionalities, allowing users to efficiently search for relevant exercises.
- Introduced a note management system enabling users to create and access notes for each exercise.
- Supported bookmarking and answer validation, aiding users in reviewing and consolidating their learning.
- Supports the practice history function so that users can view the practice questions they have done before

Shortcomings:

- However, we failed to realize the function of dynamically adjusting the difficulty of the questions and recommending related types of exercises

according to the user's historical correct rate of practice and the difficulty of each question, limiting the platform’s ability to adapt exercises based on user performance.

- The AI-driven real-time question recommendation system was only partially realized.
- Certain features (e.g., the UI for generating exercises) remained backend-focused and lacked frontend integration.

3. Data Schema and Source

Our data consists of three main components:

- **Exercises and Math Concepts**: Derived from real mathematical databases, including exercises of varying difficulty and types.
- **Users and Interaction Data**: Simulated using auto-generated virtual user data.
- **Data Structure**: Despite adjustments in data sources, the core architecture (ER diagram) remained unchanged, ensuring efficient relational design.

4. ER Diagram and Table Design

The initial ER diagram clearly outlined relationships among entities and was adhered to throughout implementation:

- Entities include User, Exercise, and MathConcept.
- Relationships cover user-exercise interactions (e.g., Stars, Completes) and concept-exercise associations. This design provided a balance between performance and scalability, demonstrating its suitability for the project.

5. Functionalities Added or Removed

Added:

- **Note Creation**: Users can create and save notes for each exercise, facilitating better review and understanding.
- **Password Verification**: User could login with their email and password, and view their personal exercise history and record.
- **Submit Answer:** After users review the answer of some question, they are unable to do the exercise again.

Removed:

- **Dynamic Difficulty Adjustment**: Originally planned to adjust exercise difficulty based on user performance, but not implemented due to algorithm complexity.
- **Fully Integrated AI Recommendation**: While part of the exercise filtering logic was implemented, the complete integration of AI-powered real-time analysis was not achieved.

6. Advanced Database Techniques

To support platform functionalities, we utilized advanced database techniques, including:

- **Complex Queries**: Developed SQL queries to analyze exercise performance, user activity, and completion rates.
- **Stored Procedures**: Encapsulated common operations (e.g., user data management, answer submission) to streamline database logic.
- **Transaction Management**: Ensured data consistency during operations such as bulk user deletion or exercise updates.
- **Triggers**: Automated user activity logging to enhance data management and tracking.

7. Technical Challenges

1) Haojun Li

Challenges:

- **ER Diagram Design**: Initially, determining the relationships between entities and their connections caused confusion. This led to logical inconsistencies in early designs and resulted in point deductions during Stage 2.
- **Database Implementation**: The complex relationships and varied query requirements increased the difficulty of schema design.
- **Time Management**: Team members faced challenges balancing this project with other commitments, delaying early progress.

Solutions:

- The team revisited and refined the ER diagram, ensuring all relationships and functionalities were accurately represented.
- Clear division of responsibilities and iterative testing helped optimize database queries and ensure functionality.
- Improved communication and prioritization in later stages allowed the team to focus on delivering key features effectively.

2) Ruichao Chen

Challenge and solution:

Front-end design: In terms of front-end design, significant challenges were encountered. During the initial stages of front-end development, issues arose with the improper connection to the back-end database. Additionally, after the preliminary completion of the front-end interface and the implementation of some basic functionalities, problems such as program crashes and failures in data retrieval occurred when adding subsequent features for further enhancement, such as submitting answers and saving user notes. Therefore, it is essential to anticipate

potential issues or functional conflicts from multiple perspectives at the outset of the project. Furthermore, the front-end code should be closely integrated with the database structure and logic during its development.

3) Chengyi Wang

Complex Networking Configurations

Challenge: Setting up networking (VPCs, subnets, and firewalls) can be daunting, particularly when connecting multiple services across different regions.

Solution:

Use the GCP Cloud Console or Terraform scripts to standardize and simplify network setup.

Employ default VPC configurations for simple projects to avoid unnecessary complexity.

Test networking configurations with sample workloads before deploying actual services.

Trigger Implementation in Cloud Functions

Challenge: Implementing triggers, such as those based on changes in Cloud Storage, Pub/Sub messages, or Firestore events, often involves configuring dependencies correctly. Failing to do so can cause triggers to misfire or not fire at all.

Solution:

Ensure the triggering source is configured correctly in the Cloud Function's deployment settings.

Debug issues using GCP logs to trace event firing and Cloud Function execution.

Utilize testing tools like the GCP Emulator to simulate trigger events during development.

Debugging and Logging

Challenge: Debugging distributed systems in GCP can be challenging due to the lack of a centralized view of logs and metrics.

Solution:

Integrate Cloud Logging and Monitoring from the start to gain insights into application performance and failures.

Use structured logging to make logs searchable and easier to analyze.

Set up alerting policies for critical metrics to detect issues proactively.

4) Cheng Wang

Challenges:

1. Encounter the issue that connect GCP sql to GCP Virtual Machine.
2. The front design of how to control ExerciseCard to appear and how to control Notes and submit answer card to appear is really hard.
3. The state variable in react is controlled by how react render the web page, and it makes the code often unable to call async function ‘axios.post’ or ‘axios.get
4. The backend code of how to properly submit sql command from VM to real sql server is also hard to debug.

Solution:

1. Read GCP documentation and figure out the solution, and proper code for connecting VM backend to SQL server.

2. Implement three different components ExerciseCard, Notes, and SubmitAnswer. Two routed webpage ExerciseList and ExploreHistory would show a list of ExerciseCard, and each ExerciseCard can trigger Notes component and SubmitAnswer component to apper.
3. Put state variable together and control them together in ExerciseCard component, and call axios get or post function in Notes, and SubmitAnswer individually.
4. Test some possible SQL commend submitted by backend in GCP console shell. Then switch back to backend code add them.

8. Changes between the original proposal and final application

1. Simplified the function to star exercise questions

2. The functionality for dynamically recommending questions based on difficulty level and user performance was removed.

3. A user login interface was introduced, enabling multiple users to independently and simultaneously utilize the platform.

4. Pagination and keyword search functionalities were added to allow users to search for target exercises more conveniently and efficiently.

9. Things that can be improved

1. Add the platform-wide online interaction feature, enabling diverse forms of user engagement. For instance, a competitive mode was added, allowing users to select question banks of varying difficulties and types to participate in one-on-one, real-time speed-solving competitions. Scoring is based on both accuracy and submission speed. Additionally, a real-time, dynamically updated leaderboard showcasing user rankings across the platform was implemented to enhance user motivation and engagement.

2. Add dynamic question adjustment, leveraging each user's performance and question type preferences to provide personalized exercise recommendations. This feature assists users in planning their practice schedules, helping them improve their mathematical skills more effectively.

10. Fianl devision of labor and management

UML Diagram and Table Design: Haojun Li, Ruichao Chen

Data Collection and Cleaning: Cheng Wang

SQL Filter Search: Haojun Li, Ruichao Chen, Cheng Wang, Chengyi Wang

Frontend Design and Development: Ruichao Chen, Haojun Li

Frontend and Backend interaction: Cheng Wang, Chengyi Wang

In order to manage our teamwork and optimize our final project, we had an online meeting during every stage of the project and discessed the direction and devision of labor. Also, we had a chat group for members to discuss the detail questions and some of the problems met daily. As a result, every member could complete his own part of work on time, and the design and optimization of our project can get onto the right track.