Final Project Report

Stage 4

CS 411

Fall 2024

Team - 055 - desi

Project Race-rewind

Neel Ghoshal, Yash Jain, Aarush Shah, Jay Sunil Goenka

Introduction & Modifications from Proposal:

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):

Our initial goal was to develop a comprehensive web application for Formula 1 enthusiasts, offering unique insights and alternative scenarios based on both historical and recent F1 data. We planned to include features that would allow users to explore how past championships might have unfolded under current scoring systems, analyze driver performance based on sector times, and track the evolution of pit stop efficiency.

As a team, we feel we have largely achieved the objectives outlined in our proposal. We successfully implemented most of the key features, enabling users to access data and insights that are typically not available elsewhere. Additionally, we managed to integrate complex database operations while implementing these features.

However, we were unable to fully realize the planned feature of performance outlier/consistency. This was primarily due to the lack of sufficient and accurate data, which we had not anticipated during the proposal phase. While this was an unfortunate limitation, we believe the features we've successfully implemented provide valuable insights and enhance the user experience.

Did we achieve desired usefulness ?:

Discuss what you think your application achieved or failed to achieve regarding its usefulness:

We believe our application has successfully captured both the essence and usefulness we aimed for at the outset of the project. As outlined in our proposal, one of our key objectives was to differentiate ourselves from other existing F1 platforms, and we are proud to say that we have achieved this. By focusing on offering unique insights that aren't readily available on other platforms, we have created a tool that provides value to F1 enthusiasts and analysts alike.

Our application enables users to explore alternative scenarios that might have occurred under different historical conditions, such as how past championships would have unfolded under current scoring systems. This feature alone provides a fresh perspective on the sport, allowing fans to rethink and engage with F1 history in a new way.

Additionally, the detailed analysis of driver performance based on sector times and the tracking of pit stop efficiency evolution offer valuable insights that are difficult to find in traditional media. These features provide a deeper level of analysis, catering to users who seek a more granular understanding of the sport's dynamics.

Changes in schema / source of data:

Discuss if you change the schema or source of the data for your application:

Yes we did switch our sources for one of our databases after the initial project proposal. When we started working with our original dataset we identified several missing values and gaps that were too hard to fulfill. Hence after extensive digging we found a new kaggle dataset that was comprehensive and most importantly had accurate and updated data. This database source switch was also the reason that limited us to fulfill all desired functions. As the table and data changed our original schema also needed some slight alterations some tables were added and removed. We also added a table for user profile and authentication.

Design Differences:

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?:

We actually were able to implement our UML diagram completely. We had accounted for all normalization and optimizations for the database while submitting the database design, which actually worked to our benefit as we were able to implement it completely.

Functionalities added/removed:

Discuss what functionalities you added or removed. Why?:

We followed the outline of our proposal for the functionalities. However we did add user authentication allowing the user to manage a profile allowing them to store their favorite team and driver. Moreover we modified / added 2 functionalities:

Feature: Qualifying and Race Position Analysis with Consistency Index

This feature calculates the qualifying and finishing positions for a specific driver in a given race year, along with the positions gained or lost during each race. Additionally, it computes a Consistency Index, which evaluates the driver’s performance across all races in that year using:

Standard Deviation of Positions Gained/Lost: Measures the variability in the driver's race performance, Average Positions Gained/Lost: Indicates whether the driver, on average, gained or lost positions during the races.

This feature provides valuable insights into a driver's race-day performance and consistency over a season.

Feature: Hypothetical Fastest Lap vs. Qualifying Performance

This feature analyzes a driver's qualifying performance by comparing their actual qualifying times to their hypothetical fastest lap time, calculated using the sum of their fastest sector times. It highlights the driver's ability to maximize their potential during qualifying sessions.

The feature is particularly useful for identifying performance gaps or inefficiencies, as it allows teams, analysts, and fans to evaluate how close a driver came to achieving their theoretical best performance. By uncovering the differences between actual and potential lap times, this feature provides valuable insights into driver consistency, car performance, and the ability to handle pressure during qualifying.

Advanced database programs:

- **Triggers**

Explain how you think your advanced database programs complement your application.

Triggers Implemented in the Project

1. User Registration Logging Trigger

This trigger logs every new user registration into the UserRegistrationLog table. It is triggered after a new user is inserted into the users table. This ensures a reliable audit trail of all user registrations for monitoring and debugging purposes.

```
CREATE TRIGGER AfterUserInsert
AFTER INSERT ON users
FOR EACH ROW
BEGIN
    INSERT INTO UserRegistrationLog (user_id, username)
    VALUES (NEW.id, NEW.username);
END;
```

2. Prevent Duplicate Usernames Trigger

This trigger ensures that no duplicate usernames are allowed in the users table. It is triggered before inserting a new row. If the username already exists, the trigger raises an error and prevents the insertion.

```
CREATE TRIGGER PreventDuplicateUsernames
BEFORE INSERT ON users
FOR EACH ROW
BEGIN
    DECLARE duplicate_count INT;
    SELECT COUNT(*)
    INTO duplicate_count
    FROM users
    WHERE username = NEW.username;
    IF duplicate_count > 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Duplicate username not allowed';
    END IF;
END;
```

Purpose of the Triggers:

- User Registration Logging Trigger: Automates the process of tracking new user registrations for auditing.
- Prevent Duplicate Usernames Trigger: Enforces username uniqueness, ensuring data integrity and preventing conflicts.

These triggers enhance the functionality and reliability of the authentication system by automating critical checks and tasks.

- **Stored Procedures:**

We implemented two of our features as stored procedures for the user to access on the web application. These stored procedures allow the user to access 2 of our features that were fastest lap comparator and Driver Consistency Index.

- **Constraints:**

Constraints allow us to ensure each row and column has expected values. These constraints allow us the maintain the database and ensure data integrity. We implemented constraints using primary and foreign keys.

- **Transaction:**

We have implemented transactions in our user authentication table with the help of a stored procedure. We ensure that All updates for favorites are handled within the stored procedure, ensuring consistency and ensuring no concurrent transactions can modify the same row.
The use and advantages of this transaction were:
Centralized Logic: All updates for favorites are handled within the stored procedure, ensuring consistency.
Reduced Code Duplication: You don't need to implement the transaction logic repeatedly in your application.
Efficient Conflict Handling: Using SELECT…FOR UPDATE ensures no concurrent transactions can modify the same row.

Technical Challenges:

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.

Aarush:

One major challenge was implementing the trigger to prevent duplicate usernames in the users table. The difficulty lay in ensuring the check worked correctly for both INSERT and UPDATE operations without causing conflicts or performance bottlenecks in a high-traffic system. The solution was to use a BEFORE trigger that validates the new username against existing entries and raises an error if a duplicate is found. Future teams should carefully test this feature in scenarios with concurrent user registrations to ensure the trigger doesn't inadvertently block valid operations or cause deadlocks.

Jay:

We faced some issues with database alignment and GCP updates. The csv to database loading caused some issues at times when the csv did not align exactly to the table created which led us to modify the csv to accommodate the format we created the table with on gcp.

Neel:

We faced some issues with the backend integration, especially trying to integrate different SQL queries and syntax on the backend. Even though I had prior full stack knowledge, the use of GCP and tying it into a project came very suddenly and was challenging at first. This required extra attention.

Yash:

Understanding of gcp functionality was challenging at the start. Some more emphasis on how to use GCP for various aspects will ease the process. Furthermore more emphasis on future project requirements at the start of the semester would enable students to choose a project more efficiently and design it to align better with the final product.

Other Changes:

Are there other things that changed comparing the final application with the original proposal?

No, other changes apart from the ones listed above.

Future Steps:

Describe future work that you think, other than the interface, that the application can improve.

We think there is a huge scope of features we wanted to add but couldn't due to limitations. A possible feature we were thinking of was to integrate a dream team feature something similar to but different from normal fantasy websites.

We could possibly allow users to push some database modifications to enable some predictive features like allowing users to simulate a season by deciding driver order and some sort of simulation feature.

Work Distribution:

We all took responsibility for the project and tried to help in implementing each step.

- Aarush and Jay worked with the SQL queries and GCP end of the project trying to update and store all the data into the database accurately while Yash and Neel helped with data verification and formatting.
- All team members were in charge of writing and verifying 1 sql query each for our 4 features, so the work was divided very evenly here.
- Aarush and Jay also worked on stored procedures and triggers while Neel worked on implementing Transactions.
- Yash and Neel wrote the frontend, backend, inter connections, and database SQL implementation for the web app on VSCode.