CS 411
FALL 2024

Project Proposal

Team - 055 - desi

Race Rewind

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

1. Project Summary:

This project aims to develop a comprehensive web application for Formula 1 enthusiasts, offering unique insights and alternative scenarios based on historical and recent F1 data. The application, called RaceRewind, will allow users to explore how past championships might have unfolded under current scoring systems, analyze driver performances based on sector times, and track the evolution of pit stop efficiency. By leveraging advanced data analysis techniques, this platform will provide F1 fans with a deeper understanding of the sport's nuances and "what-if" scenarios that go beyond traditional race statistics.

RaceRewind will feature a user-friendly interface that enables fans to interact with complex data sets in an intuitive manner. Users will be able to visualize historical championship recalculations, compare theoretical race times with actual results, and assess driver consistency in qualifying sessions. Additionally, the application will offer insights into the correlation between qualifying and race performance, and identify standout performances across different seasons. By combining these features, RaceRewind aims to become an indispensable tool for passionate F1 fans seeking to deepen their engagement with the sport through data-driven analysis and exploration.

2. Description:

RaceRewind is designed to be a comprehensive web application that addresses the growing demand for in-depth, data-driven analysis in Formula 1 fandom. Our primary goal is to bridge the gap between readily available F1 statistics and the deeper, more nuanced analysis that passionate fans crave. The problem we aim to solve is the lack of easily accessible, sophisticated analytical tools for F1 enthusiasts who want to explore alternative scenarios, dissect performance metrics, and gain insights that go beyond surface-level race results.

At its core, RaceRewind will offer six key features:

1. Historical Championship Recalculation
2. Theoretical Ideal Race Time Calculator
3. Qualifying Driver Consistency Index
4. Pit Stop Evolution Tracker
5. Qualifying vs. Race Performance Correlation
6. Performance Outlier Identifier

By implementing these features, we want to create a platform that satisfies the curiosity of hardcore F1 fans, provides new perspectives on driver and team performances, and enhances the

overall viewing experience of the sport. RaceRewind aims to be more than just a statistics database; it will be an interactive tool that encourages fans to dive deep into the data, form their own conclusions, and engage in informed discussions about various aspects of Formula 1.

Moreover, we envision this application serving as an educational resource for newcomers to the sport, helping them understand the nuances of F1 performance metrics and the multifaceted nature of success in Formula 1. By offering both historical analysis and insights into recent seasons, RaceRewind will cater to long-time fans as well as those just beginning to explore the rich history and complexity of Formula 1 racing.

3. Technically Challenging Features:

1. Historical Championship Recalculation Engine:
   - Develop a robust algorithm to apply current points systems to historical race results (1950-2010).
   - Implement efficient data storage and retrieval methods for handling large datasets spanning six decades.
   - Create a visualization system to display alternative championship outcomes clearly and interactively.

2. Theoretical Ideal Race Time Calculator:
   - Design an algorithm to compile fastest sector times for each driver over a race weekend.
   - Implement a system to calculate theoretical best lap times and extrapolate to full race distances.
   - Develop a comparison mechanism to contrast theoretical times with actual race results.

3. Qualifying Driver Consistency Index:
   - Create a sophisticated algorithm to evaluate driver consistency in qualifying sessions.
   - Implement a ranking system based on the difference between theoretical best laps and actual qualifying laps.
   - Design an intuitive visualization of consistency scores across multiple races and seasons.

4. Pit Stop Evolution Tracker:
   - Develop a data mining system to extract historical pit stop data.
   - Implement trend analysis algorithms to track pit stop time evolution.
   - Create interactive visualizations to display pit stop efficiency across teams and seasons.

5. Qualifying vs. Race Performance Correlation:
  - Design a correlation analysis system to evaluate the relationship between qualifying and race positions.
  - Implement track-specific analysis to determine varying importance of qualifying across different circuits.
  - Develop a predictive model to estimate race outcomes based on qualifying performance.

6. Performance Outlier Identifierr:
  - Create an algorithm to establish baseline performance metrics for each driver.
  - Implement a system to identify and categorize over- and under-performances based on predefined criteria.
  - Develop an alert system to highlight significant outliers in real-time during race weekends.

These technically challenging features will significantly enhance the functionality and appeal of RaceRewind. They require advanced algorithms, efficient data handling, and sophisticated visualization techniques, pushing the boundaries of what's typically available in F1 analysis tools.

4. Usefulness:

RaceRewind offers unique value to Formula 1 fans by providing deep, data-driven insights that are not readily available elsewhere. Its usefulness stems from several key factors:
- It satisfies the curiosity of hardcore fans who often speculate about "what-if" scenarios in F1 history.
- It provides a new perspective on driver and team performances, going beyond traditional statistics.
- It offers a platform for fans to engage more deeply with the sport, enhancing their viewing experience.
- It can serve as an educational tool for newcomers to understand the nuances of F1 performance metrics.

Basic functions of the web application include:
- Viewing alternative historical championship outcomes
- Analyzing theoretical race times based on sector performances
- Exploring driver consistency in qualifying sessions
- Tracking pit stop evolution over time
- Assessing the impact of qualifying on race results

- Identifying standout performances and underperformances

Complex features include the ability to customize analysis parameters, compare multiple seasons or drivers simultaneously, and receive real-time updates during race weekends.

While there are existing F1 statistics websites like F1.com and Motorsport.com, our application differentiates itself by:
1. Offering hypothetical scenarios (e.g., applying current points systems to historical data)
2. Providing deeper performance analysis (e.g., theoretical best lap times)
3. Focusing on often overlooked aspects (e.g., pit stop evolution, qualifying consistency)
4. Offering interactive, customizable data exploration tools

By combining these unique features with a user-friendly interface, RaceRewind aims to become an essential tool for F1 enthusiasts seeking a more analytical approach to their passion.

Realness

For this project, we identified two key datasets related to Formula 1 racing. One is a comprehensive data source that captures detailed data on race results, drivers, constructors, and race events across multiple seasons. The other dataset contains sector timings for each driver for each race from 2018 to 2019. These datasets will serve as the foundation for our application, enabling comprehensive analysis and feature extraction.

1. Race Results Data

- Source: race-results.yml
- Format: YAML
- Cardinality: This dataset captures race results for each Grand Prix event, providing detailed data on the drivers' positions, lap times, constructor teams, tire manufacturers, and penalties.
- Degree: Key attributes include position, driverId, constructorId, laps, time, gap, points, and gridPosition.

2. Driver Standings

- Source: driver-standings.yml
- Format: YAML
- Cardinality: This dataset captures the standings of drivers for each season, reflecting the cumulative points each driver has earned throughout the season.
- Degree: Attributes include position, driverId, and points.

3. Constructor Standings

- Source: constructor-standings.yml
- Format: YAML
- Cardinality: This dataset provides the overall standings of constructor teams, capturing their performance in races over the season.
- Degree: Attributes include position, constructorId, engineManufacturerId, and points.

4. Qualifying Results

- Source: qualifying-results.yml
- Format: YAML
- Cardinality: This dataset contains results from the qualifying sessions of each Grand Prix, detailing how drivers performed during the qualifying rounds.
- Degree: Key attributes include position, driverId, constructorId, time, q1, q2, and q3 times.

5. Free Practice Results

- Source: free-practice-1-results.yml, free-practice-2-results.yml, free-practice-3-results.yml
- Format: YAML
- Cardinality: Each dataset represents results from the practice sessions during a race weekend, providing lap times and gaps for drivers.
- Degree: Attributes include position, driverId, constructorId, time, laps, and gap.

6. Fastest Lap Data

- Source: fastest-laps.yml
- Format: YAML
- Cardinality: Contains data on the fastest laps completed by each driver in the race.
- Degree: Attributes include driverId, time, and lap.

7. Pit Stop Data

- Source: pit-stops.yml
- Format: YAML
- Cardinality: This dataset captures the pit stop timings for each driver, including the number of pit stops, laps, and time spent in the pit lane.
- Degree: Attributes include position, driverId, constructorId, stop, lap, and time.

8. Race Metadata

- Source: race.yml
- Format: YAML
- Cardinality: Contains high-level metadata about each race, such as the race date, circuit type, course length, and laps.
- Degree: Attributes include raceId, grandPrixId, officialName, qualifyingFormat, circuitId, circuitType, and laps.

9. Starting Position

- Format: YAML
- Data Size:
- Cardinality: This dataset contains the results of individual races for each Grand Prix event, capturing essential data on the performance of drivers and constructors. In the provided example, the dataset represents results for two drivers.
- Degree: Key attributes include position, driverNumber, driverId, constructorId, engineManufacturerId, tyreManufacturerId, time, and gridPenalty.

Each dataset offers a granular view into the various aspects of race weekends, driver and constructor performance, and the progression of the racing season.

The second database is of sector timings of each driver racing in each of the Grand Prix's from 2018 to 2019. The dataset contains Year, Location, Sector, Driver Name, Time. This dataset gives us access to individual sector times that make up an entire lap, which gives us more opportunities to further analyze and process data to extract new information for the user.

Functionality and User Interaction:

1. Home Page: Users are greeted with a dashboard showcasing the main features of the F1 Insights Hub. They can navigate to different sections using a menu bar or interactive tiles representing each feature.
2. Historical Championship Recalculation:
    - Users select a year range between 1950 and 2010.
    - They can choose to apply different scoring systems (e.g., current, 2010-2018, pre-2010).
    - The application displays an interactive chart showing how championships would have changed.
    - Users can hover over data points for detailed race-by-race breakdowns.
    -

3. Theoretical Race Time Analysis (2018-2019):
    - Users select a specific race from the 2018 or 2019 season.
    - The app displays actual race results alongside theoretical results based on fastest sector times.
    - Users can toggle between drivers to see individual performance breakdowns.
    - A slider allows users to adjust the impact of theoretical times on the results.
4. Qualifying Consistency Index:
    - Users choose a driver and a season (2018 or 2019).
    - The app presents a graph showing the driver's theoretical best lap vs. actual qualifying lap for each race.
    - A consistency score is calculated and displayed, with the option to compare multiple drivers.
5. Pit Stop Evolution Tracker:
    - Users can view an interactive timeline of pit stop times from 1950 to present.
    - They can filter by team, driver, or specific time periods.
    - Clicking on a data point reveals detailed information about that pit stop.
6. Qualifying vs. Race Performance Correlation:
    - Users select a season or specific Grand Prix.
    - The app displays a scatter plot showing qualifying position vs. final race position.
    - Users can hover over points to see driver names and specific race details.
    - A correlation coefficient is calculated and displayed.
7. Performance Outlier Identifier:
    - Users choose a season or driver to analyze.
    - The app highlights races where a driver significantly over- or under-performed relative to their average.
    - Users can set custom thresholds for what constitutes an "outlier" performance.

User Interaction:

- Create: Users can create custom analyses by selecting specific parameters (e.g., date ranges, drivers, tracks).
- Update: They can modify their selections in real-time to see how results change.
- Delete: Custom analyses or comparisons can be removed from their saved items.
- Search: A search function allows users to quickly find specific drivers, races, or seasons.

Project distribution:

1. Data Acquisition and Preprocessing (Jay, Aarush)

- **Task Overview:** Responsible for sourcing and preparing all datasets needed for the application.
- **Subtasks:**
    1. **Data Sourcing:** Collect data from F1 historical datasets (race results, sector times, pit stops, etc.).
    2. **Data Cleaning:** Ensure all datasets are cleaned (removing duplicates, handling missing values).
    3. **Data Transformation:** Convert raw data into standardized formats for backend consumption (e.g., convert sector times into race time formats).
    4. **Database Setup:** Prepare and store datasets in a relational database or NoSQL (depending on the system design).
- **Skills Required:** Data preprocessing, experience with F1 data formats, familiarity with SQL

2. Backend Systems Development (Yash, Aarush, Neel, Jay)

- **Task Overview:** Develop the backend architecture for processing data, implementing algorithms, and serving responses to the frontend.
- **Subtasks(Yash, Aarush) :**
    1. **API Development:** Develop RESTful APIs for the frontend to interact with the backend.
    2. **Championship Recalculation Algorithm:** Implement the logic for recalculating historical championships using modern points systems. This involves large-scale dataset manipulation and the development of a ranking algorithm.
    3. **Theoretical Lap Time Calculation:** Implement the algorithm to calculate the theoretical fastest race times based on sector data.
    4. **Performance Optimization:** Ensure that the system handles large datasets efficiently (race data from 1950-2010).
- **Subtasks(Jay, Neel):**

1. **Pit Stop Analysis Algorithm:** Build the backend logic for tracking pit stop evolution, including data mining and trend analysis.
2. **Qualifying Consistency Index:** Implement algorithms to assess driver qualifying consistency, comparing theoretical vs actual lap times.
3. **Performance Outlier Identifier:** Develop a system to detect race performances where drivers over/underperformed, including setting thresholds for outlier identification.
4. **Correlation Analysis:** Create the backend logic to assess the impact of qualifying on race outcomes (using historical data for track-specific analysis).

- **Skills Required:** Python, SQL, algorithm development, backend performance optimization, RESTful API design.

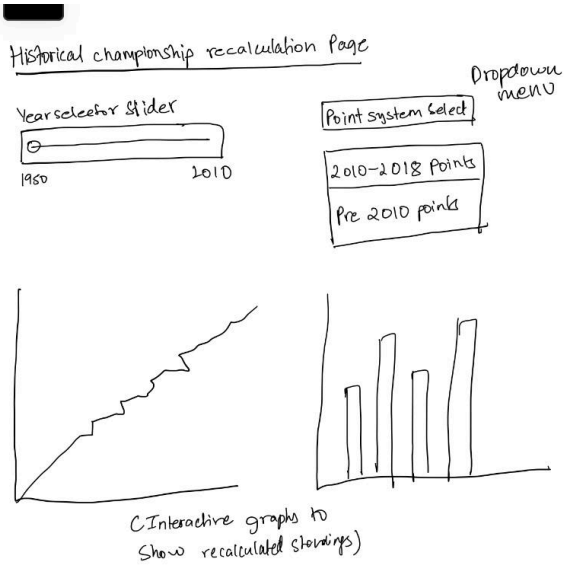
3. Frontend Development (Neel)

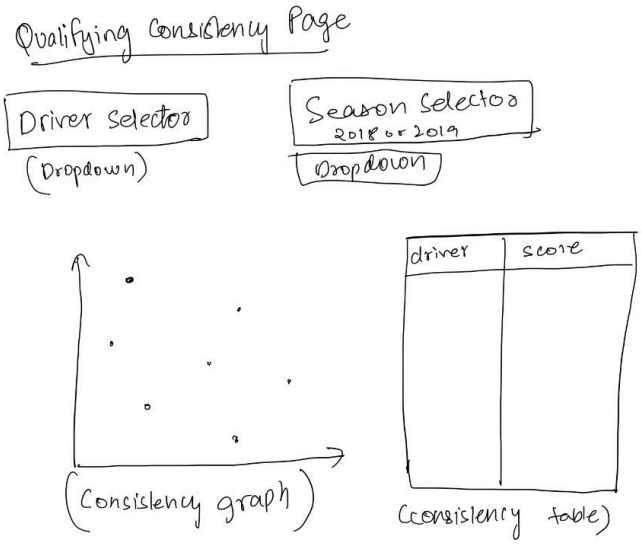
- **Task Overview:** Build the user-facing components of the RaceRewind application, ensuring a seamless and interactive experience for users.
- **Subtasks:**
    1. **Frontend Framework Setup:** Set up the frontend architecture using React, Angular, or another modern framework.
    2. **Dashboard Design:** Design and implement the main dashboard where users can explore all the features (championship recalculation, race time analysis, etc.).
    3. **Interactive Data Visualization:** Develop interactive charts/graphs for data (race results, pit stop trends, performance outliers) using libraries like D3.js or Chart.js.
    4. **User Input Forms:** Allow users to input their desired analysis criteria (e.g., choose a season, driver, or track).
    5. **User Experience Testing:** Ensure all frontend features are intuitive and responsive for users.
- **Skills Required:** HTML/CSS, JavaScript, React/Angular, experience with data visualization libraries.

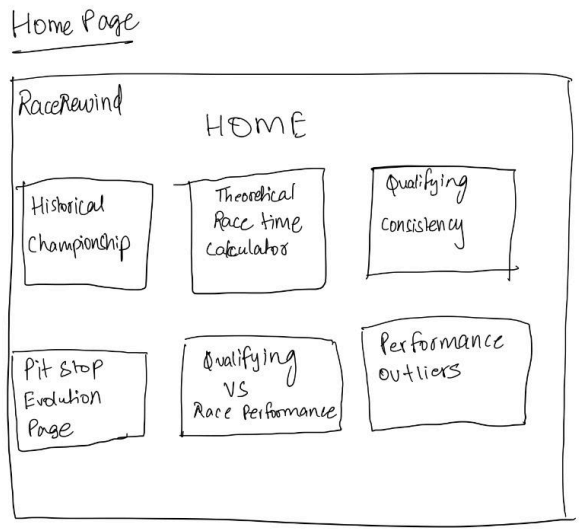
4. System Integration and Testing (Yash)

- **Task Overview:** Ensure seamless integration of backend and frontend systems and implement thorough testing for system robustness.
- **Subtasks:**
    1. **Integration Testing:** Test communication between the frontend and backend to ensure APIs respond correctly and efficiently.
    2. **Unit Testing:** Write and implement unit tests for each module (e.g., algorithms, database queries, APIs).
    3. **Performance Testing:** Stress test the application to handle large datasets and multiple users simultaneously.
    4. **Bug Fixing:** Address any bugs or performance issues identified during testing.
- **Skills Required:** Experience with testing frameworks, familiarity with both frontend and backend systems.

UI Mockup


Historical championship recalculation Page
Year selector Slider
1950
2010
Point system Select
Dropdown menu
2010-2018 Points
Pre 2010 points
(Interactive graphs to show recalculated standings)


Qualifying Consistency Page
Driver Selector
(Dropdown)
Season Selector
2018 or 2019
Dropdown
(Consistency graph)
driver
score
(Consistency table)


Home Page
RaceRewind
HOME
Historical Championship
Theoretical Race time calculator
Qualifying consistency
Pit Stop Evolution Page
Qualifying VS Race Performance
Performance outliers