**Offroad Challenge Tracker - Project Documentation**

**Introduction**

The **Offroad Challenge Tracker** is a Flutter-based application with SQLite database integration to manage rankings and scoring for **Rain Forest Challenges (RFC)**. The application allows users to:

* Register participants
* Manage track details
* Enter track performance data
* Calculate rankings automatically
* Display leaderboard filtered by categories

**Tech Stack**

* **Frontend**: Flutter (Dart)
* **Database**: SQLite (sqflite package)
* **UI Design**: Flutter Widgets
* **Version Control**: GitHub

**Features Implemented**

**1. Participant Registration**

* Users will be able to add participant details:
  + Participant Number
  + Driver Name
  + Co-driver Name
  + Category Selection (Stock, Mod Petrol, Mod Diesel, Pro, Ladies + Pro)
* Options to **add, edit, remove**, or **clear entries** will be implemented.
* Data will be stored in the SQLite database.

**2. Track Management**

* Users will be able to add multiple tracks.
* Tracks will be stored in the SQLite database with unique track numbers.
* Allows modification of track details if needed.

**3. Scoring System (To Be Implemented)**

* Users will be able to enter the performance details for each participant:
  + Track Completion Time
  + Track Score (before penalty)
  + Track Penalty
* Once submitted, the data will be stored in SQLite.
* Users will be able to **edit scores** before final submission.

**4. Rank Calculation & Leaderboard (To Be Implemented)**

* Automatically calculates total scores after all tracks are entered.
* Ranks participants based on total scores.
* Filters leaderboard by category dynamically.

**Database Schema**

**Tables Created**

1. **Participants Table**
   * id (INTEGER, PRIMARY KEY)
   * participant\_number (INTEGER, UNIQUE, NOT NULL)
   * driver\_name (TEXT, NOT NULL)
   * co\_driver\_name (TEXT, NOT NULL)
   * category (TEXT, NOT NULL)
2. **Tracks Table**
   * id (INTEGER, PRIMARY KEY)
   * track\_name (TEXT, NOT NULL)
   * track\_number (INTEGER, UNIQUE, NOT NULL)
3. **Scores Table** (To Be Implemented)
   * id (INTEGER, PRIMARY KEY)
   * participant\_id (INTEGER, FOREIGN KEY REFERENCES Participants(id))
   * track\_id (INTEGER, FOREIGN KEY REFERENCES Tracks(id))
   * completion\_time (REAL, NOT NULL)
   * track\_score (INTEGER, NOT NULL)
   * penalty (INTEGER, DEFAULT 0)
4. **Rankings Table** (To Be Implemented)
   * id (INTEGER, PRIMARY KEY)
   * participant\_id (INTEGER, FOREIGN KEY REFERENCES Participants(id))
   * total\_score (INTEGER, NOT NULL)
   * rank (INTEGER, NOT NULL)

**Database Implementation in Flutter**

**Steps Taken**

1. Installed sqflite and path dependencies in pubspec.yaml.
2. Created database\_helper.dart for database interactions.
3. **Next Steps:** Implement CRUD functions for:
   * Adding, fetching, updating, and deleting participants.
   * Adding and retrieving tracks.
   * Entering scores and penalties.
   * Calculating total scores and rankings.
4. **Future Plans:** Integrate database functions in the main.dart UI.
5. Use **FutureBuilder** to fetch and display data dynamically.

**Next Steps**

* Implement **CRUD operations** for Participants, Tracks, and Scores.
* Implement **Export & Import Database** feature.
* Enhance UI with **Material Design & Animations**.
* Optimize queries for **faster ranking calculations**.