

Comprehensive Specification Document for Scrim Statistics Webapp

1. Overall Vision

- **Target Users:**

Team coaches and managers.

- **Objectives:**

- Allow coaches to upload scrim match results with detailed performance data for both their team and enemy teams.
 - Support grouping of matches into scrim sessions (e.g., "ADMU vs UST Scrim" containing multiple games).
 - Provide a robust analytics suite including recent matches, team statistics, and leaderboards.
 - Enable comprehensive historical tracking (e.g., player IGN changes via alias history) to support advanced data analysis and future features.
-

2. Core Features

a. Core Feature 1: Match Upload

1. Match Entry & Meta-data

- **Date & Time:**

- *Component:* Date/Time Picker
- *Field:* `match_date_time`

- **Opponent Information:**

- **Opponent Category:**

- *Component:* Dropdown
- *Options:* "Collegiate", "Amateur", "Pro"
- *Field:* `opponent_category`

- **Opponent Team Name:**

- *Component:* Text Input
- *Field:* `opponent_team_name`

- **Opponent Team Abbreviation:**

- *Component:* Text Input

- *Field:* `opponent_team_abbreviation`
- **Additional Match Details:**
 - **Scrim Type:**
 - *Component:* Dropdown
 - *Options:* "Practice", "Tournament", "Friendly", etc.
 - *Field:* `scrim_type`
 - **Match Outcome:**
 - *Component:* Radio Buttons or Dropdown
 - *Options:* "Win", "Loss", "Draw"
 - *Field:* `match_outcome`
 - **General Notes:**
 - *Component:* Multi-line Text Area
 - *Field:* `general_notes`
 - **Game Number:**
 - *Component:* Numeric Input or Dropdown
 - *Field:* `game_number`
 - **Side Designation:**
 - *Component:* Radio Buttons or Dropdown
 - *Options:* "Blue Side", "Red Side"
 - *Field:* `team_side` (and optionally `enemy_side`)

2. Scrim Group Association

- **Association Field:**
 - *Component:* Searchable, filterable autocomplete input
 - *Field:* `scrim_group`
 - *Behavior:*
 - Filters existing scrim groups as the coach types.
 - Inline option to create a new scrim group if none match.
 - *Linkage:*
 - Associates the match entry with the selected scrim group (e.g., "ADMU vs UST Scrim"), grouping related games (e.g., Games 1–4).

3. Player Statistics Entry

- **Fixed Player Rows:**

- **Our Team Section:**
 - 5 fixed rows labeled "Our Player 1" through "Our Player 5"
 - *Prefill Behavior:*
 - On loading the form, automatically prefill each row's "Player Selection" field with the 5 players from the previous scrim.
 - *Replacement Option:*
 - The coach can click the dropdown in any row to replace the prefilled IGN with a different selection.
- **Enemy Team Section:**
 - 5 fixed rows labeled "Enemy Player 1" through "Enemy Player 5"
 - *Prefill Behavior:*
 - If previous enemy data is available, prefill these fields; otherwise, allow manual selection.
 - *Replacement Option:*
 - Coach can replace any prefilled enemy IGN as needed.
- **Each Fixed Row Includes the Following Fields:**
 1. **Player Selection:**
 - *Component:* Dropdown
 - *Data Source:*
 - For our team: Only players from your team roster.
 - For enemy team: Only players from the enemy roster.
 - *Field:* `player_ign`
 - *Behavior:*
 - The coach selects a player for that row.
 - If the coach enters an IGN manually that is not found or is ambiguous, the system prompts:

"The IGN 'Riptide' is new. Did one of these players change their IGN to 'Riptide'? Update player IGN? If not, create new player record."
 - Once confirmed, the entry is linked to the player's unique UID.
 2. **Hero Played:**
 - *Component:* Dropdown (populated from the hero database)
 - *Field:* `hero_played`

3. Raw KDA Inputs:

- *Components:* Three numeric inputs for `kills`, `deaths`, and `assists`

4. Computed KDA (Game Computed):

- *Component:* Numeric Input (or read-only if auto-calculated)
- *Field:* `computed_kda`

5. Other Performance Metrics:

- **Damage Dealt:** Numeric Input; Field: `damage_dealt`
- **Damage Taken:** Numeric Input; Field: `damage_taken`
- **Turret Damage:** Numeric Input; Field: `turret_damage`
- **Teamfight Participation:** Numeric or Percentage Input; Field: `teamfight_participation`
- **Gold Earned:** Numeric Input; Field: `gold_earned`

6. Player-Specific Notes:

- *Component:* Multi-line Text Area
- *Field:* `player_notes`

4. File Uploads

• Multi-file Uploader:

- *Component:* Drag-and-drop interface with a "Select Files" button
- *Supported Formats:* JPEG, PNG (optionally PDF)
- *Features:*
 - Preview thumbnails and progress indicators.
- *Association:*
 - Files are linked to the match entry and associated scrim group.
- *Future Enhancements:*
 - Option for OCR integration for automatic data extraction.

5. Submission Workflow

• Validation:

- Validate that all required fields (meta-data, scrim group, player statistics) are complete.
- Ensure each fixed player row is correctly linked to a unique UID via the player selection process.

• Final Submission:

- On submission, store:
 - All meta-data fields.

- The scrim group association.
 - All player statistics, each linked to the appropriate UID.
 - Associated file uploads linked to the match record.
 - **Post-Submission:**
 - The match entry appears in the Recent Matches section.
 - Aggregated data flows into dashboards and leaderboards.
-

6. Authentication & Authorization (for Reference)

- **User Authentication:**
 - Use Django's built-in authentication or JWT-based authentication (e.g., Django REST Framework Simple JWT) to secure endpoints.
 - Store tokens or session cookies on the React frontend.
 - **Data Access:**
 - Each match entry includes a `submitted_by` field linked to the User model.
 - API endpoints filter data based on the authenticated user, ensuring that users can only access data they submitted.
-

3. Core Feature 2: Recent Matches

1. Filter Options

- **Time-Based Filters:**
 - Preset filters: Past Week, Past Month, Past 3 Months
 - Custom Date Range: Date range picker for manual selection.
- **Team & Category Filters:**
 - Team Filter: Dropdown or checkboxes to filter by a specific team.
 - Opponent Type Filter: Options to filter by "Collegiate", "Amateur", or "Pro".

2. Aggregate Summary Cards

- **Display Cards at Top:**
 - **Match Winrate Card:** Displays overall scrim winrate (e.g., 75%).
 - **Match Win-Loss Record Card:** Displays aggregated win-loss record (e.g., 3-1).

3. Scrim History Listing

- **Scrim Group Cards:**
 - Each card represents a scrim session (e.g., "ADMU vs UST Scrim") with details:

- Scrim group name.
- Aggregate win-loss record (e.g., "3-1").
- Date or date range.
- **Show More Button:**
 - Each card has a "Show More" button to expand and display individual match entries within that scrim session.

4. Detailed Match Results Page (Boxscore)

- **Access:**
 - Clicking an individual match entry navigates to a detailed match results page.
 - **Boxscore Layout:**
 - Header with meta-data (date/time, opponent, scrim type, game number, side designation).
 - Side-by-side team player statistics (including raw and computed KPIs).
 - Display of associated file uploads.
 - Navigation controls (back button, breadcrumbs).
-

4. Core Feature 3: Team Statistics

1. Team Selection

- **Team Selector:**
 - Component: Dropdown or searchable input listing all teams.
 - When a team is selected, update the view accordingly.

2. Team Overview

- **Aggregate Team Metrics:**
 - Display overall statistics such as:
 - Total matches played.
 - Overall winrate.
 - Average computed KDA.
 - Other key metrics (total damage, gold, etc.)
 - Visualized using summary cards or charts.

3. Player List & Aggregated Statistics

- **Player List Display:**
 - Component: Table or card view listing all players in the selected team.

- For each player, display:
 - Player Name (current IGN)
 - Role/Position (if applicable)
 - Key metrics: winrate, average computed KDA, total matches played, most played hero.
- Each player entry is clickable for a detailed view.

4. Detailed Player View

- **Player Overview:**
 - Header with player name, profile image, and basic stats.
- **Recent Matches Section:**
 - List/table of the player's recent matches with basic details (date, opponent, outcome, key metrics).
 - Filtering options (e.g., past 7 days, past month).
- **Hero-Specific Statistics:**
 - Display most played heroes.
 - Show winrates with specific heroes (via charts or tables).
 - Include trends over time (e.g., graph of winrate or KDA trends).

5. Database Schema (Summary)

- **Teams:**
 - `teams(team_id, team_name, team_abbreviation, team_category, created_at, updated_at)`
- **Players:**
 - `players(player_id, team_id, current_ign, role, profile_image_url, created_at, updated_at)`
- **Player Aliases:**
 - `player_aliases(alias_id, player_id, alias, created_at)`
- **Opponent Teams (Optional):**
 - `opponent_teams(opponent_team_id, opponent_category, team_name, team_abbreviation, created_at, updated_at)`
- **Scrim Groups:**
 - `scrim_groups(scrim_group_id, scrim_group_name, start_date, end_date, notes, created_at, updated_at)`
- **Matches:**
 - `matches(match_id, scrim_group_id, submitted_by, match_date_time, scrim_type, match_outcome, score_details, general_notes, game_number, team_side, opponent_category, opponent_team_name, opponent_team_abbreviation, created_at, updated_at)`
- **Player Match Statistics:**

- `player_match_stats(stats_id, match_id, player_id, hero_played, kills, deaths, assists, computed_kda, damage_dealt, damage_taken, turret_damage, teamfight_participation, gold_earned, player_notes, is_our_team, created_at, updated_at)`

- **File Uploads:**

- `file_uploads(file_id, match_id, file_url, file_type, uploaded_at)`

6. Architecture & Technology Stack

- **Backend:**

- Django with Django REST Framework, PostgreSQL as the database.
- Use JWT or Django's session authentication.

- **Frontend:**

- React with a UI library (Material-UI, Ant Design, etc.), React Router for navigation.

- **Deployment & Tools:**

- Use v0 for deployment, Cursor for collaborative development.

- **Authentication:**

- Secure API endpoints so users can only access data they submitted (using `submitted_by` fields and API filtering).

7. File Structure (High-Level)

Django Backend (scrimstats_backend)

```
sql
Copy
scrimstats_backend/
├── manage.py
├── requirements.txt
├── scrimstats_backend/
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
├── apps/
│   ├── matches/
│   │   ├── models.py
│   │   ├── serializers.py
│   │   └── views.py
│   ├── urls.py
│   ├── teams/
│   │   ├── models.py
│   │   └── serializers.py
```


- Build models, serializers, views, and URLs as per the database schema.
- Write unit tests and configure authentication.

3. **Frontend Development:**

- Develop React components and pages as outlined.
- Integrate API services using Axios or fetch.

4. **Integration & Testing:**

- Connect the React frontend with the Django REST API.
- Perform unit tests and end-to-end testing.

5. **Deployment:**

- Deploy the backend and frontend using v0 (or similar platforms) and configure CI/CD pipelines.

6. **Monitoring & Iteration:**

- Implement logging, error tracking, and gather user feedback for future improvements.