**Project Scope: Stream Management and Bowler Lookup System**

**1. Objective**

Develop a robust system that allows **Joomla** to dynamically display **stream tiles** based on the state of streams managed by a **Flask app**. Additionally, provide **Event Details Management** and **Bowler Lookup** functionality to enhance the user experience in the bowling center context.

**2. Features**

1. **Dynamic Stream Tiles on Joomla**:
   * **Description**: Display clickable stream tiles on the Joomla site. The tiles should only appear when the corresponding **stream** is **enabled** in the **Flask app**.
   * **Requirements**:
     + The Flask app will expose the current **stream status** (enabled/disabled) for each lane pair.
     + Each active stream will display a **YouTube iframe** embedded within a tile on Joomla.
     + The **stream title** will be dynamic and correspond to the lane pair (e.g., "Lane 1 & 2").
     + Tiles will be **lazy-loaded** for improved performance.
2. **Event Details Page in Flask**:
   * **Description**: A page within the Flask app to manage event information, including uploading lane draw data.
   * **Requirements**:
     + Event name, dates, and venue should be entered by an admin.
     + The system must allow admins to **upload a CSV** containing **lane draw** data for each event (e.g., bowlers' names, lanes, and scheduled times).
     + This page should store event data and lane draw information in a structured format for future use.
     + The page should include **validation** for required fields (event name, venue, etc.) and handle CSV uploads properly.
3. **Bowler Lookup**:
   * **Description**: Allow users to **search for a bowler’s name** on the Joomla front-end and get the corresponding **lane and time** from the uploaded lane draw.
   * **Requirements**:
     + The Flask app will provide an API to **search for a bowler** in the **lane draw** and return the relevant lane and time.
     + The Joomla front-end will include a form where users can **enter a bowler’s name**.
     + Once the bowler is found, the system will display **the lane and time** they are scheduled for.
     + Provide appropriate **error handling** when a bowler is not found or when there is an issue with the CSV data.

**3. Workflow**

1. **Joomla Front-End**:
   * **Dynamic Tile Display**:
     + The Joomla site will make **AJAX requests** to the **Flask app** to retrieve stream data (enabled lanes, stream URLs).
     + Active streams will result in **dynamic tiles** appearing on the page, each with an embedded **YouTube iframe**.
     + Each tile will have a **label** such as "Lane 1 & 2" to match the lane pairing set in the Flask app.
   * **Bowler Lookup**:
     + Users will enter a **bowler's name** in the form on the Joomla site.
     + Joomla will make an **AJAX request** to the Flask API to retrieve the **lane and time** associated with that bowler.
     + The results will be displayed dynamically on the page (e.g., "Lane 1 at 2:00 PM").
2. **Flask Backend**:
   * **Event Details Page**:
     + Admins will enter and submit event data, including **event name**, **event dates**, and **venue**.
     + The app will process **CSV uploads** containing lane draw information (bowlers' names, lanes, times).
     + Data will be **stored in a structured format** for easy lookup and retrieval.
   * **Bowler Lookup API**:
     + Flask will provide a **search endpoint** where Joomla can request **bowler details** (lane and time) based on the uploaded CSV.
     + This will return a response with the **lane and time** or an error if no match is found.

**4. Technical Specifications**

1. **Flask Backend**:
   * **API Endpoints**:
     + /event\_details: Manages event data entry and CSV upload.
     + /lookup\_bowler: Handles searching for a bowler’s lane and time based on their name.
   * **CSV Parsing**:
     + The **CSV file** should contain the following columns: Bowler Name, Lane, Time.
     + The data should be stored in a structured **JSON format** or **database** for future use.
2. **Joomla Front-End**:
   * **Dynamic Stream Tiles**:
     + Use **AJAX** to pull active stream data from the Flask app and dynamically generate stream tiles with **YouTube iframes** embedded.
     + **Lazy-load** the tiles as the user scrolls the page.
   * **Bowler Lookup**:
     + Use **AJAX** to interact with Flask’s **/lookup\_bowler** API to return lane and time data based on the bowler's name entered in the form.

**5. User Stories**

1. **As an admin**, I want to manage the event details and upload a lane draw CSV, so that I can configure the event for the bowling center.
2. **As a user**, I want to search for a bowler’s name, so that I can see which lane and time they are scheduled for.
3. **As a user**, I want to see stream tiles for active lanes, so that I can watch the live streams of the lanes I’m interested in.

**6. Milestones & Deliverables**

1. **Flask App Development**:
   * **Event Details page** (for managing event data and uploading CSV).
   * **Bowler Lookup API** (to handle searches for bowler name and return lane/time).
2. **Joomla Front-End**:
   * **Dynamic stream tiles** for each active lane.
   * **AJAX call** for bowler name lookup and displaying the results.
3. **Testing & Debugging**:
   * **End-to-end testing** of stream tiles, bowler lookup, and CSV upload functionality.
   * **Bug fixes** and UI adjustments as needed.

**7. Timeframe:**

This will depend on the complexity and integration requirements, but a rough estimate would be:

* **Flask Backend Setup**: 2–3 days for API creation, CSV processing, and bowler lookup.
* **Joomla Front-End Integration**: 2–3 days for dynamic tiles and bowler search functionality.
* **Testing and Deployment**: 2–3 days for testing and ensuring all components are functioning correctly.

**End of Scope Document**

This scope document should now serve as a **comprehensive guide** for building the event management system and stream control features within your project.

Let me know if you'd like any changes or further details. We can begin with the implementation steps now!