|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Software Requirement Specifications**  **Version: [2.0]**   |  |  | | --- | --- | | Project Code | F24-102 | | Supervisor | Sir. Basit Jasani | | Co Supervisor | - | |  |  | | Project Team | Rayyan Ahmed(21K-3079)  Abdul Rafay(21K-3051)  Syeda Minal Alam(21K-3072) | | Submission Date | 15-05-2025 | |

Document History

[Revision history will be maintained to keep a track of changes done by anyone in the document.]

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Name of Person** | **Date** | **Description of change** |
| 1.0 | Rayyan Ahmed |  | Document Created |
| 2.0 | Syeda Minal Alam | 14-05-2025 | Document Updated |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Distribution List

[Following table will contain list of people whom the document will be distributed after every sign-off]

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | |
| Mr. Basit Jasani | | Supervisor |
| - | | Co- Supervisor |
|  | |  |

Document Sign-Off

[Following table will contain sign-off details of the document. Once the document is prepared and revised, this should be signed-off by the sign-off authority.

Any subsequent changes in the document after the first sign-off should again get a formal sign-off by the authorities.]

|  |  |  |
| --- | --- | --- |
| **Version** | **Sign-off Authority** | **Sign-off Date** |
| 1.0 | Supervisor |  |
| 2.0 | Supervisor |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

[1.](#_heading=h.30j0zll"﷟HYPERLINK "bookmark://_heading=h.30j0zll) ﷟HYPERLINK "bookmark://\_heading=h.30j0zll"Introduction 7

[1.1.](#_heading=h.3znysh7"﷟HYPERLINK "bookmark://_heading=h.3znysh7) ﷟HYPERLINK "bookmark://\_heading=h.3znysh7"Purpose of Document 7

[1.2.](#_heading=h.2et92p0"﷟HYPERLINK "bookmark://_heading=h.2et92p0) ﷟HYPERLINK "bookmark://\_heading=h.2et92p0"Intended Audience 7  
1.3 Abbreviations ………………………………………………………………………………………...7

[1.4.](#_heading=h.tyjcwt"﷟HYPERLINK "bookmark://_heading=h.tyjcwt) ﷟HYPERLINK "bookmark://\_heading=h.tyjcwt"Document Convention 7

[2.](#_heading=h.3dy6vkm"﷟HYPERLINK "bookmark://_heading=h.3dy6vkm) ﷟HYPERLINK "bookmark://\_heading=h.3dy6vkm"Overall System Description 8

[2.1.](#_heading=h.1t3h5sf"﷟HYPERLINK "bookmark://_heading=h.1t3h5sf) ﷟HYPERLINK "bookmark://\_heading=h.1t3h5sf"Project Background 8

[2.2.](#_heading=h.4d34og8"﷟HYPERLINK "bookmark://_heading=h.4d34og8) ﷟HYPERLINK "bookmark://\_heading=h.4d34og8"Project Scope 8

[2.3.](#_heading=h.2s8eyo1"﷟HYPERLINK "bookmark://_heading=h.2s8eyo1) ﷟HYPERLINK "bookmark://\_heading=h.2s8eyo1"Not In Scope 8

[2.4.](#_heading=h.17dp8vu"﷟HYPERLINK "bookmark://_heading=h.17dp8vu) ﷟HYPERLINK "bookmark://\_heading=h.17dp8vu"Project Objectives 8

[2.5.](#_heading=h.3rdcrjn"﷟HYPERLINK "bookmark://_heading=h.3rdcrjn) ﷟HYPERLINK "bookmark://\_heading=h.3rdcrjn"Stakeholders 8

[2.6.](#_heading=h.26in1rg"﷟HYPERLINK "bookmark://_heading=h.26in1rg) ﷟HYPERLINK "bookmark://\_heading=h.26in1rg"Operating Environment 8

[2.7.](#_heading=h.35nkun2"﷟HYPERLINK "bookmark://_heading=h.35nkun2) ﷟HYPERLINK "bookmark://\_heading=h.35nkun2"System Constraints 8

[2.8.](#_heading=h.1ksv4uv"﷟HYPERLINK "bookmark://_heading=h.1ksv4uv) ﷟HYPERLINK "bookmark://\_heading=h.1ksv4uv"Assumptions & Dependencies 8

[3.](#_heading=h.2jxsxqh"﷟HYPERLINK "bookmark://_heading=h.2jxsxqh) ﷟HYPERLINK "bookmark://\_heading=h.2jxsxqh"External Interface Requirements 9

[3.1.](#_heading=h.z337ya"﷟HYPERLINK "bookmark://_heading=h.z337ya) ﷟HYPERLINK "bookmark://\_heading=h.z337ya"Hardware Interfaces 9

[3.2.](#_heading=h.3j2qqm3"﷟HYPERLINK "bookmark://_heading=h.3j2qqm3) ﷟HYPERLINK "bookmark://\_heading=h.3j2qqm3"Software Interfaces 9

[3.3.](#_heading=h.1y810tw"﷟HYPERLINK "bookmark://_heading=h.1y810tw) ﷟HYPERLINK "bookmark://\_heading=h.1y810tw"Communications Interfaces 9

[4.](#_heading=h.4i7ojhp"﷟HYPERLINK "bookmark://_heading=h.4i7ojhp) ﷟HYPERLINK "bookmark://\_heading=h.4i7ojhp"Functional Requirements 10

[4.1.](#_heading=h.2xcytpi"﷟HYPERLINK "bookmark://_heading=h.2xcytpi) ﷟HYPERLINK "bookmark://\_heading=h.2xcytpi"Functional Hierarchy 10

[4.2.](#_heading=h.1ci93xb"﷟HYPERLINK "bookmark://_heading=h.1ci93xb) ﷟HYPERLINK "bookmark://\_heading=h.1ci93xb"Use Cases 10

[4.2.1.](#_heading=h.3whwml4"﷟HYPERLINK "bookmark://_heading=h.3whwml4) ﷟HYPERLINK "bookmark://\_heading=h.3whwml4"[Title of use case] 10

[5.](#_heading=h.2bn6wsx"﷟HYPERLINK "bookmark://_heading=h.2bn6wsx) ﷟HYPERLINK "bookmark://\_heading=h.2bn6wsx"Non-functional Requirements 11

[5.1.](#_heading=h.qsh70q"﷟HYPERLINK "bookmark://_heading=h.qsh70q) ﷟HYPERLINK "bookmark://\_heading=h.qsh70q"Performance Requirements 11

[5.2.](#_heading=h.3as4poj"﷟HYPERLINK "bookmark://_heading=h.3as4poj) ﷟HYPERLINK "bookmark://\_heading=h.3as4poj"Safety Requirements 11

[5.3.](#_heading=h.1pxezwc"﷟HYPERLINK "bookmark://_heading=h.1pxezwc) ﷟HYPERLINK "bookmark://\_heading=h.1pxezwc"Security Requirements 11

[5.4.](#_heading=h.49x2ik5"﷟HYPERLINK "bookmark://_heading=h.49x2ik5) ﷟HYPERLINK "bookmark://\_heading=h.49x2ik5"User Documentation 11

[6.](#_heading=h.147n2zr"﷟HYPERLINK "bookmark://_heading=h.147n2zr) ﷟HYPERLINK "bookmark://\_heading=h.147n2zr"References 12

[7.](#_heading=h.3o7alnk"﷟HYPERLINK "bookmark://_heading=h.3o7alnk) ﷟HYPERLINK "bookmark://\_heading=h.3o7alnk"Appendices 13

1. **Introduction**

* 1. **Purpose of Document**

The purpose of this document is to define the software requirements for the Cricket Video Clip Extraction and Highlight Generation project. It outlines the system's functionality, interfaces, and constraints to ensure a clear understanding among stakeholders.

* 1. **Intended Audience**
* Developers working on implementing the system.
* QA engineers for understanding the expected behavior.
* End-users and stakeholders who will evaluate the system.

**1.3**  **Abbreviations**

* **OCR**: Optical Character Recognition
* **FFmpeg**: Fast Forward Moving Picture Experts Group
* **AI:** Artificial Intelligence
* **LLM:** Large Language Model
  1. **Document Convention**
* Font Family = Arial
* Font Size = 12 for headings, 10 for the rest of the content

1. **Overall System Description**
   1. **Project Background**

In the rapidly evolving world of sports media, the demand for swift and engaging highlight reels has become a critical factor in driving audience engagement. Traditional manual methods of creating cricket highlights are time-consuming, require significant human intervention, and often introduce delays that reduce the immediacy and relevance of the content. Cricket, being a complex and widely followed sport, presents additional challenges due to its unpredictable flow and the necessity to identify crucial moments such as sixes, fours, and wickets accurately.

To address these challenges, this project proposes an AI-driven solution—**CrickHigh**—which automates the generation of cricket highlights. Leveraging technologies such as **EfficientNet** for event detection, **EasyOCR** for scoreboard extraction, and **FFmpeg** for video segmentation and processing, the system is designed to minimize manual effort, improve accuracy, and deliver faster turnaround times.

Additionally, CrickHigh integrates **Large Language Models (LLMs)** through the **Gemini API**, supported by a **local knowledge base**, to provide contextual insights, event explanations, and assist users in natural language query handling. This enhances the usability and intelligence of the system by offering smart interactions for highlight generation, validation, and contextual understanding.

This approach aims to optimize workflows for sports analysts, broadcasters, and content creators.

**Problem Statement:**

* The current process for generating cricket highlights faces multiple limitations:
* Manual highlight creation is slow, inefficient, and resource-intensive.
* Extracting key moments like sixes, fours, and wickets requires domain knowledge and continuous human supervision.
* Mistakes in identifying important events may lead to inaccurate or irrelevant highlights.
* Delays in publishing highlight reels reduce their effectiveness in attracting real-time viewer attention.
* The absence of automation results in inconsistent quality and wasted production effort
* Lack of intelligent assistance for validating or understanding event significance limits system usability for non-experts.
  1. **Project Scope**

**CrickHigh** is an AI-powered system focused on automating the creation of cricket match highlights using computer vision, video processing tools, and large language models. It aims to provide a seamless, accurate, and efficient solution for sports content production.

**Included Functionalities:**

* Automatic detection of cricket events (sixes, fours, wickets) using **EfficientNet**.
* Scoreboard extraction using **EasyOCR** to provide contextual information.
* Video segmentation and clip generation using **FFmpeg** based on predefined timestamps.
* Merging of extracted clips into a cohesive highlight reel.
* Web interface allowing users to upload match recordings and configure timestamp settings.
* **LLM-powered assistant** (via **Gemini API**) to answer user queries about match summaries, provide clip insights, and assist with system usage.
* Integration of a **semantic knowledge base** for contextual support during user interactions.

**Excluded Functionalities:**

* Manual tagging or identification of cricket events.
* Real-time processing of live cricket matches.
* Integration with external APIs for player or match statistics.
* Advanced post-production editing features such as adding commentary or overlays.
  1. **Not In Scope**
* Live match video processing or streaming.
* Real-time data retrieval from sports databases or feeds.
  1. **Project Objectives**
* **Automated Event Detection**  
   Utilize **EfficientNet** to detect and classify cricket events (sixes, fours, wickets) with high precision and minimal latency.
* **Video Clip Processing**  
   Employ **FFmpeg** for video segmentation to extract clips corresponding to key events, ensuring frame-accurate slicing and high-quality output.
* **Scoreboard Recognition**  
   Integrate **EasyOCR** for recognizing scoreboard details from video frames, enabling better contextual understanding for each clip.
* **Highlight Reel Compilation**  
   Seamlessly concatenate individual clips into a unified, polished highlight video for ease of sharing and viewing.
* **Intelligent Assistance & Summarization**

Use **Gemini API** and local knowledge base to provide users with natural language summaries, event insights, and real-time assistance by retrieving relevant context from a connected **knowledge base**.

* 1. **Stakeholders**
* Employees
* HODs
* Software Developer
* HR
* DataBase Administrator
* organization(Fast NU)
  1. **Operating Environment**

CrickHigh is designed to run in environments with modern computational infrastructure and stable internet connectivity. It integrates video processing, OCR, machine learning, and LLM-based natural language querying.

**Hardware and Network Requirements:**

* A mid- to high-performance machine capable of running:
* Video processing using FFmpeg
* CNN-based event detection (e.g., EfficientNet)
* OCR and LLM response generation

**Software Prerequisites:**

* + Pre-installed FFmpeg for video processing.
  + EasyOCR for extracting scoreboard information from video frames.
  + A Python environment configured with necessary libraries such as OpenCV, NumPy, and Pandas.
  + Local storage for saving extracted clips and compiled highlight reels.
  + Gemini API access for utilizing the Large Language Model (LLM) to handle natural language queries and generate summaries.
  + FAISS and SentenceTransformers libraries for implementing the local knowledge base and semantic search for context-aware LLM responses.

**User Environment:**

* + Suitable for use in office settings, production studios, or other professional environments where sports highlights need to be generated or analyzed.
  + Designed to support users interacting with an AI-powered chatbot that retrieves relevant cricket insights using a combination of Gemini API (LLM) and a structured knowledge base built from historical match data.
  1. **System Constraints**

**Third-Party Dependencies:**

* EasyOCR and FFmpeg must be pre-installed and configured correctly for the system to operate.
* Any interruption in these third-party tools may affect the accuracy or timeliness of results.
* Gemini API (LLM) must be accessible via a valid internet connection and properly configured. Failure in API access may disable chatbot or query-response features.

**Processing Environment:**

* Video resolution should ideally be 720p or higher for optimal scoreboard detection and clip extraction accuracy.
* System performance depends on the computational capacity of the machine being used (CPU/GPU performance impacts processing speed).
* LLM response time depends on internet bandwidth and the efficiency of local document retrieval using FAISS.

**Storage Requirements:**

* Sufficient local storage for saving extracted clips and temporary data during processing.
* Additional space required for storing document embeddings used in LLM knowledge base (e.g., Cricsheet-based match data).

**Ethical and Practical Constraints:**

* OCR-based scoreboard recognition must respect the intellectual property of broadcasters.
* The system must ensure that extracted data is not misused or shared without proper authorization.
* LLM-generated insights must not fabricate or misrepresent factual match events or statistics.

**User Constraints:**

* The interface should be designed to accommodate users with varying technical expertise, offering clear instructions and minimal manual intervention.
* LLM-based chatbot must provide clear, explainable answers and avoid overwhelming users with technical jargon..

**Environmental Constraints:**

* The system must be capable of operating in shared or noisy environments where sound notifications may not be practical.
* The chatbot interface should use visual cues or on-screen messages in place of audio-based alerts
  1. **Assumptions & Dependencies**

**Data Integrity:**

* Employees will provide accurate and truthful information during research paper submissions and performance tracking processes.
* Employees will only change the status of scoreboard entries or cricket highlights (e.g., from "pending" to "verified") after ensuring their correctness, as no automated mechanism exists to validate manual edits.
* Clips extracted through FFmpeg will be free from errors and will align with the project's predefined standards for quality and content.
* Responses generated through the Gemini API will be reviewed to ensure they align with retrieved knowledge base documents and reflect accurate cricket match insights.

**User Knowledge:**

* Users (developers and admins) will have the necessary knowledge to operate and maintain the EasyOCR and FFmpeg tools used in the system.
* Users are familiar with cricket-related terminologies, scoreboard formats, and highlight structures to accurately validate and process extracted data.
* Users should also understand how to query the AI-based chatbot using cricket-relevant questions, and how to interpret or verify LLM-generated responses.

**Tool Dependencies:**

* EasyOCR will function effectively with the pre-processed frames of video clips for accurate text extraction from scoreboards.
* FFmpeg will reliably extract video clips based on the timestamps provided by the system, without skipping or introducing errors.
* Gemini API must be integrated and authenticated properly for natural language interaction with users, allowing question-answering based on cricket match data.
* FAISS and SentenceTransformers libraries are required to implement the semantic search and retrieval system powering LLM responses.

**Technical Dependencies:**

* The system relies on the availability of high-quality video clips (minimum 720p resolution) for accurate text and object recognition.
* The hardware running the software must support necessary libraries like TensorFlow/PyTorch (for AI-based features) and have sufficient processing power for smooth functioning.
* Internet connectivity is required for updating models, accessing external APIs such as Gemini API, and interacting with online services.
* The knowledge base must be indexed using FAISS for efficient retrieval and accurate LLM input prompts.

**Evaluation Logic:**

* Configurations for AI evaluation parameters (e.g., thresholds for detecting highlights like sixes or wickets) will be logically set and verified for accuracy.
* Pre-trained models or custom weights used in the AI module will align with the specific requirements of cricket video analysis.
* LLM-generated answers will be compared with retrieved knowledge base snippets to ensure factual consistency and meaningful user interaction.

**Human Interaction:**

* Developers or admins will manually validate and refine any results flagged as ambiguous or incorrect by the system.
* LLM chatbot responses will be monitored periodically to ensure relevance, clarity, and adherence to the cricket data domain, with feedback loops incorporated for improvement.
  1. **Dependencies**

1. **FFmpeg:**  
    Required for extracting clips from video files. FFmpeg is a key tool for processing video data, as it is used to trim, concatenate, and manipulate video files. The availability and proper configuration of FFmpeg on the system are crucial for the seamless extraction and manipulation of cricket clips.
2. **EasyOCR:**  
    EasyOCR is used for Optical Character Recognition (OCR) to extract textual information (such as the scoreboard) from cricket video clips. The proper installation and configuration of EasyOCR are essential to ensure accurate recognition of text from video frames.
3. **Python Libraries:**  
    The project depends on several Python libraries for data manipulation and video processing. These include:

* numpy for numerical computations.
* opencv for video frame extraction and manipulation.
* pandas for handling and organizing timestamp data.
* matplotlib for visualizations, if necessary.
* moviepy for video editing tasks (if applicable).
* faiss for implementing semantic similarity search in the knowledge base.
* sentence-transformers for converting cricket-related text documents into embeddings for LLM retrieval.

1. **5Video File Formats:**  
    The system requires specific video file formats (e.g., MP4) to process clips correctly. Videos must be in compatible formats for FFmpeg and EasyOCR to extract frames and textual information accurately.
2. **Hardware Requirements:**  
    Adequate CPU and memory resources are required for processing high-quality video clips (such as 720p or higher). For efficient video extraction, it is recommended to have a system with sufficient processing power and memory to handle multiple clips in parallel.
3. **Network Connection (for API Access):**  
    If the system accesses external APIs (e.g., for real-time score updates or Gemini LLM queries), a stable internet connection is required. This is particularly important if the system pulls live data, uses the Gemini API for generating summaries, or interacts with online services for additional features.
4. **External Video Files:**  
    The system depends on the availability of cricket video clips for processing. Users must ensure that the video files are accessible and stored in an appropriate directory (e.g., /content/faces/Train) for the system to perform analysis.

1. **External Interface Requirements**

[This section is intended to specify any requirements that ensure that the new system will connect properly to external components. Place a context diagram showing the external interfaces at a high level of abstraction.]

* 1. **Hardware Interfaces**

**Desktop Devices:**

* Minimum 4GB RAM, dual-core processor, running Windows 10, macOS 10.13, or Linux (kernel 4.x or later).
* The system should be able to handle video processing tasks effectively with the minimum hardware requirements.
* Additionally, devices used for video input and processing should have a web browser installed to interact with the user interface and run the required Python libraries.

**Video Capture Devices (Optional):**

Devices capable of capturing cricket match footage, such as cameras or video recorders, for processing in the system. The video file should be uploaded in compatible formats (e.g., MP4).

**AI Model Deployment:**

The AI model used for event detection and score recognition will be deployed locally to reduce latency and ensure fast processing of video frames. The system should be capable of storing the model in an efficient manner to minimize resource consumption and maximize speed during real-time processing.

* 1. **Software Interfaces**

**OpenCV:**  
 This software library will be used for video frame extraction and manipulation. The system uses OpenCV to load video clips, extract frames, and identify the key moments of the match (e.g., fours, sixes, wickets).

**AI Model (TensorFlow or PyTorch):**  
 The AI model, responsible for event detection (e.g., recognizing specific cricket events like boundaries or wickets), will be deployed locally using TensorFlow (v2.x) or PyTorch (v1.x). The model will process the video frames in real time.

**FFmpeg:**  
 FFmpeg will be used to extract clips from the video, convert video formats, and merge video files for cricket highlights. The system will rely on FFmpeg’s APIs to perform the necessary video editing tasks.

**EasyOCR:**  
 This is used to perform OCR on video frames to detect text (e.g., scoreboards, player names). EasyOCR will communicate with the system through a Python interface to extract and process text from frames.

**Gemini API (LLM):**  
 The system will interact with the Gemini API to allow users to ask natural language questions (e.g., "Show highlights where player X hit boundaries") and receive intelligent responses. LLM outputs are enriched using a local knowledge base for contextual understanding of match events

* 1. **Communications Interfaces**

N/A

**Functional Requirements**

#### 1. Video Upload and Processing

* **FR1.1**: The system shall allow users (admins and employees) to upload cricket match videos (e.g., MP4 format) to the platform.
* **FR1.2**: The system shall validate the uploaded video to ensure it is in a supported format (e.g., MP4, AVI) and meets the required video quality (minimum resolution: 720p).
* **FR1.3**: The system shall automatically extract frames from the uploaded video at a specified frame rate (e.g., 30 frames per second) for further processing.

#### 2. Event Detection

* **FR2.1**: The AI model shall automatically detect key events (e.g., fours, sixes, wickets) within the video based on predefined criteria.
* **FR2.2**: The system shall use video frame analysis (via OpenCV) to identify the specific moments of the match where significant events occur.
* **FR2.3**: The system shall apply object recognition to detect players, boundaries, and wickets using pre-trained machine learning models (e.g., TensorFlow or PyTorch).
* **FR2.4**: The system shall generate a list of detected events along with their timestamps within the video.

#### 3. Scoreboard and Text Detection

* **FR3.1**: The system shall extract text from video frames (e.g., scoreboards, player names, over details) using OCR technology (e.g., EasyOCR).
* **FR3.2**: The system shall accurately detect and extract information such as runs, wickets, and overs from the scoreboard during the video.
* **FR3.3**: The extracted text shall be displayed alongside the video clip for user reference.

#### 4. Highlight Generation

* **FR4.1**: Based on the detected events, the system shall generate cricket highlights by extracting specific segments from the original video.
* **FR4.2**: The system shall allow users to customize which events to include in the highlight, such as boundaries, wickets, or player-specific actions.
* **FR4.3**: The system shall merge video clips into a final highlight reel, ensuring smooth transitions between the detected events.
* **FR4.4**: The system shall allow users to preview the generated highlight before finalizing the video.

#### **5. Natural Language Interaction with LLM**

#### **FR5.1:** The system shall allow users to query match details or highlight segments using natural language (e.g., "Show me all sixes by Player A").

#### FR5.2: The system shall send user queries to the Gemini API and retrieve context-aware answers based on indexed cricket knowledge.

#### FR5.3: The system shall highlight or display relevant video clips or metadata corresponding to the user’s question.

#### FR5.4: The system shall support follow-up questions or clarifications to enable conversational interactions.

* + 1. **[Use Case Diagram]**

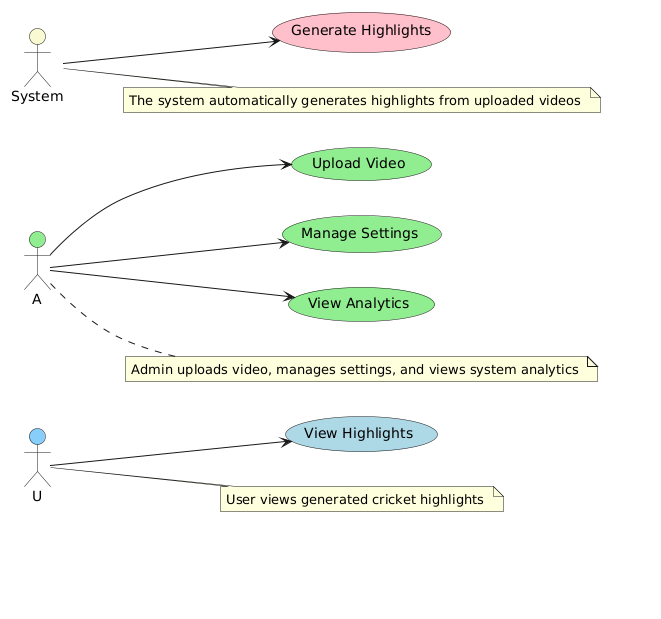


Fig1.1

1. **Non-functional Requirements**

#### Performance Requirements

* **Concurrency**: The system should handle multiple concurrent users, allowing them to view highlights or upload videos without noticeable lag in processing or response time.
* **Video Processing Time**: The system should retrieve and display generated highlights within 2 seconds of a request from the user.
* **CRUD Operations**: CRUD operations (Create, Read, Update, Delete) on video or highlight data should execute within 1 second on average.
* **Highlight Generation**: The AI model should generate cricket highlights from uploaded video clips within 15 seconds of video submission.
* **Video Uploading**: The system should confirm the successful uploading of videos within 5 seconds.
* **System Uptime**: The system must maintain 99.9% uptime to ensure reliability, especially during live match analysis and highlight generation.
* **LLM Query Response Time:**  
   Natural language queries to the Gemini API, integrated with the knowledge base, must respond within 3 seconds on average to ensure a smooth user experience.
* **Knowledge Base Retrieval:**  
   Semantic search and retrieval from the local knowledge base (FAISS) must occur in under 1 second to assist the LLM in providing context-aware answers.

* 1. **Safety Requirements**

1. **Data Integrity**: The system should prevent unauthorized deletion or modification of critical data, such as video clips, generated highlights, or user details.
2. **Backup Mechanism**: A backup mechanism must be implemented to automatically store video and highlight data snapshots daily, ensuring quick recovery in case of system failure.
3. **Error Handling**: The system must include error handling to prevent crashes under invalid input scenarios (e.g., incorrect video format or network failure during highlight generation).
4. **Accidental Overwriting**: Safeguards must be in place to prevent accidental overwriting of highlight data, particularly for ongoing live video uploads or highlights already created.
5. **Evaluation Accuracy**: Ensure the AI model evaluates highlights accurately, providing fair and consistent results. This is critical to avoid misrepresentation in match highlights.
6. **LLM Query Validation:** Input queries to the Gemini API must be validated to prevent malformed or inappropriate requests that could compromise performance or cause incorrect responses.
   1. **Security Requirements**

* **User Authentication**: All users (Admins, HODs, and Employees) must authenticate with multi-factor authentication (MFA) to access their respective dashboards.
* **Data Encryption**: All sensitive data, such as user credentials, video data, highlight results, and match statistics, must be encrypted using AES-256 encryption to ensure privacy.
* **Role-based Access Control (RBAC)**: The system should implement role-based access control (RBAC) to ensure users only have access to the features relevant to their role (Admin, HOD, Employee).
* **Blockchain Security**: All communication with the blockchain (e.g., writing promotion criteria) must be encrypted to ensure data security and integrity during interactions.
* **Data Privacy**: The system must ensure the privacy of employees' and users' data, ensuring that personal or performance data is only visible to authorized users and not shared with unauthorized parties.
* **Network Security**: Ensure secure communication between the system and external services, such as video platforms and blockchain, using SSL/TLS encryption.
* **LLM Access Control:** Access to the Gemini API must be restricted to authenticated users only, and all interactions must be logged to ensure traceability and prevent misuse of the language model.
  1. **User Documentation**
* **User Manual**: A detailed user manual will be provided to guide Admins, HODs, and Employees through the system features, including uploading videos, viewing highlights, and using the AI model for match analysis.
* **Online Help**: Integrated help within the system will provide step-by-step instructions for using key features such as video uploading, highlight viewing, and AI configuration. This will be accessible from the homepage.
* **Tutorial Videos**: For complex operations such as configuring AI weights or generating specific highlights from a full match video, tutorial videos will be available. These videos will guide users on how to perform these tasks efficiently.
* **Context-Sensitive Help**: The system will offer context-sensitive help, displaying relevant guidance based on the user’s current action or the page they are on (e.g., uploading a video, viewing highlights).
* **Troubleshooting Guide**: A troubleshooting guide will be available to help users resolve common issues, such as failed video uploads, system crashes, or issues with highlight generation. It will include clear instructions on how to report issues or get assistance.

1. **References**

* **Journal Article**  
  M. R. Wicaksono, A. K. Sari, and A. F. Habiburrahman, “On the Design of a Blockchain-based Fraud-prevention Performance Appraisal System,” International Journal of Computer and Communication Systems, vol. 14, no. 1, pp. 1–9, 2020. [Online]. Available:<https://doi.org/10.22146/ijccs.67669>
* **Conference/Journal Paper (PDF from IRJET)**  
  S. S. Jadhav, A. P. Mahesh, and S. J. Patil, “Corruption-less Appraisal System using Blockchain,” International Research Journal of Engineering and Technology (IRJET), vol. 9, no. 6, pp. 1442–1445, June 2022. [Online]. Available:<https://d1wqtxts1xzle7.cloudfront.net/91711775/IRJET_V9I6193-libre.pdf>
* **Government/Policy Document**  
  Higher Education Commission (HEC), HEC Journals and Publication Policy 2024. HEC, Pakistan, 2024. [Online]. Available:<https://www.hec.gov.pk/english/services/faculty/journals/Documents/HEC%20Journals%20and%20Publication%20Policy%202024.pdf>
* **Website**  
  Higher Education Commission (HEC), “HEC Services and Resources.” [Online]. Available:<https://www.hec.gov.pk>
* **Web-based Resource/Database**  
  Higher Education Commission, “HEC Journal Recognition System (HJRS),” HEC Pakistan. [Online]. Available:<https://hjrs.hec.gov.pk>
* **Book**  
  Author(s), Title of the Book. Unpublished resource, Google Drive. [Online]. Available:<https://drive.google.com/file/d/1N9Hd2NumMq8DrEhjA2XxnwyS9NXG1IYn/view>
* **Web-based Blockchain Development Resource**  
  Multichain, “Blockchain Resources for Developers,” Multichain Developers. [Online]. Available:<https://www.multichain.com/developers>

1. **Appendices**

[This section should include supporting detail that would be too distracting to include in the main body of the document.]