A logo with blue text

Description automatically generated

**CrickHigh**

**PROJECT SUPERVISOR**

Sir Basit Jasani

**PROJECT TEAM**

**Shaikh Abdul Rafay 21K-3051**

**Syeda Minal Alam 21K-3072**

**Rayyan Ahmed 21K-3079**

**Submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.**

**FAST SCHOOL OF COMPUTING**

**NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES KARACHI CAMPUS**

**May 2024**

|  |  |
| --- | --- |
| Project Supervisor | Mr Basit Jasani |
| Project Team | Shaikh Abdul Rafay K21-3051  Syeda Minal Alam K21-3072  Rayyan Ahmed K21-3079 |
| Submission Date | May 15, 2025 |

Mr Basit Jasani Dr. Muhammad Rafi

**Supervisor Head of Department**

**FAST SCHOOL OF COMPUTING**

**NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES**

**KARACHI CAMPUS**

**Acknowledgement:**

All the thanks to the supervisor Mr. Basit Jasani and, the members of the of the project Shaikh Abdul Rafay, Rayyan Ahmed, Syeda Minal Alam, and people who cooperated in providing the data for the project. All the mentioned people have contributed to the project and helped us on this journey. Without their help we could not have done this.

Table of Contents

[Abstract 4](#_Toc198030540)

[Introduction 6](#_Toc198030541)

[Related Work 6](#_Toc198030542)

[Requirements 7](#_Toc198030543)

[Use Cases 8](#_Toc198030544)

[Highlight Generation 8](#_Toc198030545)

[Get Cricket Insights via AI 8](#_Toc198030546)

[Non-Functional Requirements are given below: 10](#_Toc198030547)

[Design 11](#_Toc198030548)

[UserApp 12](#_Toc198030549)

[Implementation 13](#_Toc198030550)

[Test cases 14](#_Toc198030551)

[Conclusion 15](#_Toc198030552)

[References 16](#_Toc198030553)

# Abstract

CrickHigh is an AI-driven platform designed to automate the generation of cricket highlights from full-length match videos, streamlining the traditionally manual and time-consuming editing process. By leveraging state-of-the-art computer vision techniques, including EfficientNet-B0 trained on bowler run-up data, and Optical Character Recognition (OCR) via EasyOCR and image processing, the system accurately detects key moments such as ball deliveries, boundaries, sixes, wickets, and scoreboard changes. It then extracts and compiles these clips into concise highlight reels, operating at approximately 40–60% of the original video’s duration.

To enrich user engagement, CrickHigh integrates a win prediction module based on historical ODI World Cup ball-by-ball data. This module uses a two-stage LightGBM model to forecast match outcomes using pre-match features and real-time match progression. Additionally, the system offers a chatbot interface powered by the Gemini API, allowing users to query a vast knowledge base constructed from over 18,000 matches including international, league, and club games. Semantic search is performed using FAISS, with boosted document retrieval for more accurate and natural language responses.

Combining video automation, predictive analytics, and interactive AI, CrickHigh provides a fast, intelligent, and user-friendly solution for modern cricket content consumption and analysis.

# Introduction

CrickHigh addresses this challenge through a smart, end-to-end automated system for cricket highlight generation. Leveraging advanced computer vision and Optical Character Recognition (OCR) technologies, CrickHigh intelligently detects key moments—such as sixes, boundaries, and wickets—directly from match footage, drastically reducing editing time while maintaining high accuracy. The system uses EfficientNet-B0, trained on bowler run-up footage, to identify delivery start points, and combines it with scoreboard changes detected via OCR to pinpoint highlight-worthy events.

Beyond highlights, CrickHigh brings an analytical edge with a machine learning-based win predictor, trained on historical ODI World Cup ball-by-ball data. It uses a two-stage LightGBM model to generate pre-match predictions and update them dynamically based on live match progress.

To enhance interactivity, CrickHigh includes a chatbot powered by the Gemini API and a semantic search engine (FAISS). Backed by a comprehensive knowledge base derived from over 18,000 matches (including club leagues like IPL, PSL, and international games for both men and women), the chatbot enables users to ask natural language questions and receive insightful, data-driven responses on players, match summaries, and stats.

By seamlessly integrating automated video editing, predictive modeling, and conversational AI, CrickHigh offers a transformative experience for both fans and media professionals—making cricket content smarter, faster, and more interactive than ever before.

# Related Work

Here is the list of the platforms that provided similar services which are:

**1. Wisden (by ESPN / CricViz)**

**Wisden, developed by ESPN** in collaboration with CricViz, is widely used by major broadcasters such as Sky Sports and Fox Cricket. It leverages ball-tracking technology to automatically generate highlights and deliver live win probability updates. However, ball-tracking is computationally expensive and the system is tailored for broadcasters, lacking a user-friendly interface. In contrast, Crichigh offers a more efficient and accessible approach that maintains high performance while significantly reducing computational overhead.

**2. Nexus Cricket (by Sportradar)**

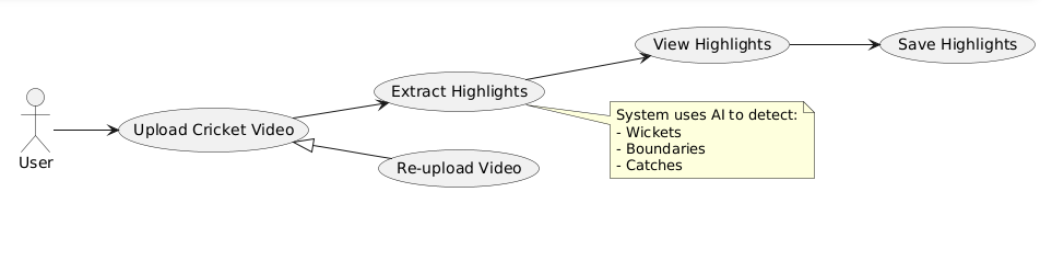
**Nexus Cricket (by Sportradar)** targets enterprise clients like leagues and fantasy platforms. It offers automated video tagging and predictive models for team and player analytics. Unlike Nexus, CrickHigh focuses on general users, combining highlight detection, score prediction, and natural language interaction for a more engaging and accessible experience.

# Requirements

Functional requirements and the diagrams are given below:

## Use Cases

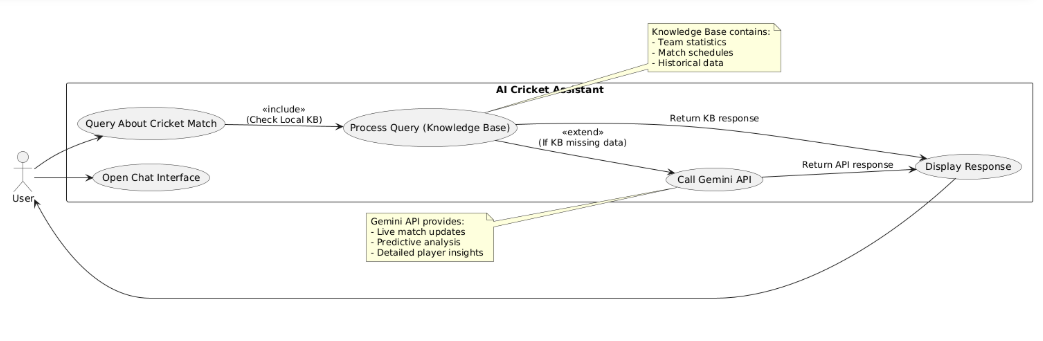
### Highlight Generation



**Figure 1: Use case of Highlight Generation**

|  |  |  |  |
| --- | --- | --- | --- |
| ***UC1: Highlight Generation*** | | | |
| ***Use case Id:*** | | *Uc1* | |
| ***Actors:***  *Users* | | | |
| ***Feature:*** *Highlight Generation* | | | |
| ***Pre-condition:*** | | *User has a video file* | |
| ***Step#*** | ***Action*** | | ***Software Reaction*** |
| ***1*** | *User clicks "Upload Video" and selects a cricket match video and highlight is generated.* | | *System validates the file format (e.g., MP4, AVI).* |
| ***2.*** |  | | *System displays extracted highlights video* |
| ***3.*** | *User saves the highlights or reuploads another video.* | | *System saves the highlights to the user’s device or repeats the process upon reupload* |
| ***Alternate Scenarios:*** | | | |
| ***1:***  *User uploads an unsupported file (e.g., PDF).*  *System shows error: "Invalid file format. Please upload a video (MP4/AVI).”* | | | |
| ***2:*** *User clicks reuploads Video after previewing incorrect highlights.*  *System returns to Step 1.* | | | |  |
| ***Post Conditions:*** *Highlights are successfully generated and saved.* | | | |

### Get Cricket Insights via AI



**Figure 2: Match Information Query via AI**

|  |  |  |  |
| --- | --- | --- | --- |
| ***UC2: Get Cricket Insights via AI*** | | | |
| ***Use case Id:*** | | *Uc2* | |
| ***Actors:***  *User* | | | |
| ***Feature:*** *AI-Powered Cricket Query Resolution* | | | |
| ***Pre-condition:*** | | *System has internet connectivity (for Gemini API)* | |
| ***Step#*** | ***Action*** | | ***Software Reaction*** |
| ***1.*** | *User open AI Chat interface* | | *Displays input field with* |
| ***2.*** | *User enters query* | | *1. Retrieves relevant information from local knowledge base* *2. Uses Gemini API to produce a natural language response* |
| ***Alternate Scenarios:*** | | | |
| ***1:*** *In case of invalid query, the system shows invalid query message* | | | |
| ***Post Conditions:*** *User sees the answer on screen* | | | |

## Non-Functional Requirements are given below:

**Operating Environment**

The CrickHigh platform is designed for cross-platform compatibility, supporting both Windows and Linux operating systems. It runs efficiently on modern web browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge, ensuring broad accessibility for end users. The system supports both cloud-based and on-premise deployments, providing flexibility for different infrastructure needs.

CrickHigh is built using Flask as its backend framework and a responsive frontend powered by Tailwind CSS and React. The platform leverages state-of-the-art machine learning and computer vision libraries, including PyTorch (≥2.0.0), TorchVision (≥0.15.0), TensorFlow, and OpenCV (≥4.7.0), to enable real-time video analysis and prediction tasks. Additional libraries such as NumPy, Pillow (≥9.4.0), EasyOCR (≥1.6.2), and FFmpeg are used for image processing, OCR, and media handling. Sentence-transformers, FAISS-CPU, and Google Generative AI are integrated to power semantic search, vector indexing, and generative NLP features.

The application is optimized for Python 3.11 and benefits from CUDA-enabled GPUs for accelerated performance during model training and inference, making it suitable for high-throughput and real-time workloads.

**System Constraints**

The system benefits from a dedicated GPU for efficient performance, particularly in tasks involving convolutional neural networks (CNNs) used for video processing, object detection, and model inference. However, it does not require a high-end GPU—mid-range options like the NVIDIA GTX 1060 or better are sufficient to ensure smooth execution of these tasks.

For OCR operations, a strong CPU is more important, as text extraction relies heavily on frame-by-frame analysis rather than GPU acceleration. The system is capable of generating highlights in approximately 40–60% of the original video’s runtime. While faster output is possible by reducing processing intensity, this often comes at the cost of accuracy.

With a better CPU, frame processing becomes significantly faster, enabling the system to maintain high performance while also improving speed—minimizing the usual trade-off between accuracy and efficiency.

**Assumptions & Dependencies**

* Users have a stable internet connection for real-time processing.
* Users understand English for system interactions and queries.
* Users access the platform on compatible devices with supported operating systems.
* Users have basic technical knowledge to navigate the platform.

**System Dependencies**:

* The system relies on a hosting service to connect to servers.
* A stable internet connection is essential for real-time operations.
* The system requires an active database connection for data storage and retrieval.

# Design

The system is designed with a frontend-backend architecture to ensure modularity and ease of development. The frontend of the platform is built using ReactJS, providing users with an interactive interface to view match highlights, get score predictions, and interact with the LLM-powered chatbot. All user requests and video interactions are handled through this interface.

The backend is developed in Python, where the main processing occurs. This includes video frame extraction, highlight detection using computer vision, scorecard recognition through OCR, and real-time win prediction using machine learning. The backend also handles communication with the LLM (via Gemini API) to respond to user queries. The frontend and backend communicate through API calls, allowing smooth and efficient data flow between components.

# Implementation

The CrickHigh platform comprises a ReactJS-based frontend and a Python backend. The frontend enables users to upload cricket match videos, view automatically generated highlights, and interact with a chatbot for match-related queries. Communication between the frontend and backend is managed via Flask, which exposes well-defined route endpoints to handle API requests and data flow.

### Highlight Extraction Pipeline

The highlight generation module combines computer vision and OCR techniques for accurate event detection:

* **Frame Processing & Ball Detection**: OpenCV is used to extract individual frames from the video. A **Convolutional Neural Network based on EfficientNet-B0** is used to detect the start of a bowler’s run-up, marking the beginning of a delivery. This model was trained on a dataset of approximately 5,000 labeled images capturing bowler run-up sequences.
* **Score Change Detection via OCR**: **EasyOCR**, combined with traditional image processing techniques, is used to extract on-screen scoreboards. It tracks changes in metrics like runs, wickets, and overs. Events such as boundaries, sixes, and wickets are flagged when significant scoreboard changes are detected.
* **Event Matching**: Timestamps from CNN-based ball start detection are matched with OCR-detected score changes to isolate highlight-worthy segments.
* **Highlight Compilation**: The matched segments are extracted using FFmpeg and concatenated into a final highlights video, which is then made accessible through the frontend interface.

### Chatbot and Match Insights

* **Knowledge Base Construction**: Using ball-by-ball data from Cricsheet (covering over 18,000 matches from 2001 to 2025, including IPL, PSL, and international matches), a structured knowledge base was built encompassing batter and bowler career stats, and match summaries.
* **Semantic Indexing**: These documents were indexed using FAISS to enable fast and relevant semantic retrieval.
* **Query Handling**: User queries are optimized using the Gemini API to form effective semantic search prompts. Relevant documents—especially those with perfect query matches—are boosted in the retrieval process.
* **Natural Language Response**: The Gemini API uses the retrieved documents to generate accurate and fluent responses in natural language.

### Win Predictor Module

CrickHigh includes a real-time **Win Predictor** built on historical **ODI World Cup** ball-by-ball data:

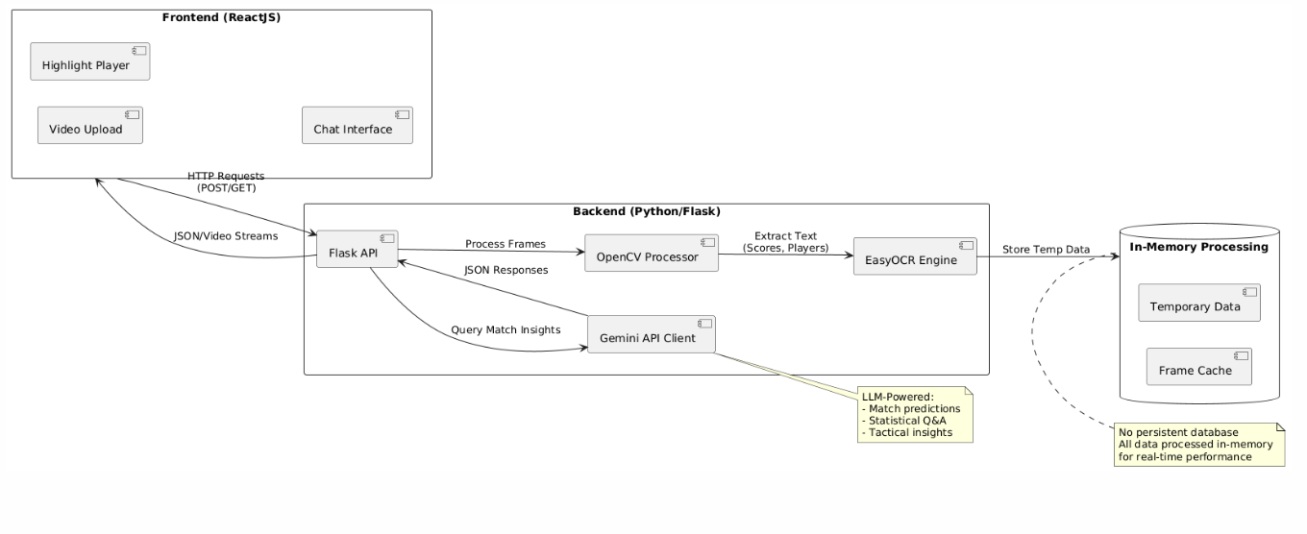
* **Pre-Match Prediction**: A **LightGBM model** trained on features like venue, historical win rates, and player profiles provides an initial win probability before the match begins.
* **In-Match Prediction**: A second LightGBM model takes the pre-match output and updates it dynamically using live ball-by-ball inputs such as current score, wickets, and over progress, providing accurate in-game win probability.

### System Architecture & Performance

* The platform supports both Windows and Linux environments, with frontend compatibility across modern browsers (Chrome, Firefox, Edge).
* GPU acceleration (e.g., NVIDIA GTX 1060 or higher) boosts CNN inference performance, while a strong CPU improves OCR speed and efficiency.
* Highlights are generated in **40–60% of the original video’s runtime**, with performance-speed trade-offs customizable based on available hardware.
* All processing is done in-memory, eliminating the need for external databases and ensuring a fast, lightweight system.

CrickHigh delivers an end-to-end solution for automated cricket highlight generation, real-time match analytics, interactive insights, and predictive modeling—powered by a seamless integration of deep learning, machine learning, and natural language technologies.

Following diagram shows the implementation



**Figure 9: System architecture Diagram**

# Test cases

|  |  |  |
| --- | --- | --- |
| **Test Cases** | **Names** | **Expected Result** |
| TC1:Video Upload and Highlight Generation | User uploads a cricket video for highlight generation. | Video is processed, and a highlights video is generated based on key moments like boundaries, sixes, and wickets. |
| TC2: Highlight Generation with Missing Scoreboard | User uploads a video where the scoreboard is partially or completely missing. | *Result:* No output as Scoreboard is crucial for the system |
| TC3: Performance with Large Video Files | User uploads a high-resolution video (e.g., 1080p or 4K) of a long match (e.g., full ODI). | The system processes the video efficiently and generates highlights within a reasonable time frame (e.g., 40-60% of the original video runtime). |
| TC4:Accuracy of Event Detection (Boundary/Six/Wicket) | User uploads a video with clear boundaries, sixes, and wickets. | The system accurately identifies these events and includes them in the final highlights video. |
| *TC5:*Multiple Event Detection | User uploads a video with overlapping events (e.g., a wicket followed by a boundary). | The system correctly detects all events and ensures all are represented in the final highlights video without missing or duplicating events. |
| TC6:Pre-Match Win Prediction | System is provided with pre-match features (venue, teams, player stats, etc.). | The LightGBM model outputs an initial win probability before the match starts. |
| TC7: Real-Time Win Prediction Update | System receives real-time ball-by-ball data during a live match. | The second LightGBM model updates the win probability based on ongoing match conditions, such as runs, wickets, and overs remaining. |
| TC8: Prediction Accuracy Based on Historical Data | System receives data from a historical match (e.g., an ODI World Cup match). services | The system outputs an accurate win prediction based on the historical patterns and ball-by-ball data. |
| TC9: Basic Query Response (Player Stats) | User asks for a player's career stats (e.g., "Show me Virat Kohli's batting average"). | The chatbot returns accurate player statistics, pulled from the ball-by-ball dataset or other match data sources. |
| TC10: Complex Query Handling (Match Insights) | User asks a complex question (e.g., "Who is the highest run-scorer in the last IPL final?"). | The chatbot processes the query through the Gemini API, retrieves relevant data, and responds in natural language with the correct answer. |
| TC11: Handling Unsupported Queries | User asks a question that is not related to cricket or supported by the knowledge base (e.g., "What is the weather today?"). | The chatbot informs the user that it cannot process the query, perhaps directing them to a different service or asking if they want to know something else cricket-related. |

**Table 1: Test cases**

# Conclusion

In conclusion, the CrickHigh platform represents an innovative and comprehensive solution for cricket fans and service providers, leveraging cutting-edge technologies in computer vision, machine learning, and natural language processing. The system efficiently handles multiple complex tasks, including real-time highlight generation, win prediction, and an interactive chatbot for match insights. By combining EfficientNet-B0 for event detection, EasyOCR for score extraction, and LightGBM models for win prediction, CrickHigh offers an engaging, data-driven experience for users while maintaining high performance even with large video files and live match data.

The platform also features a seamless user interface for booking services and providing real-time updates, making it a versatile tool for cricket enthusiasts. The robust test cases ensure that all components of the system, from video processing to predictive modeling and chatbot interactions, are thoroughly validated under various scenarios. With flexible deployment options, CrickHigh is poised to enhance the cricket viewing experience, providing both fans and service providers with a powerful tool for engagement and interaction.

Through its combination of advanced algorithms, real-time data processing, and user-friendly design, CrickHigh stands as a prime example of how technology can revolutionize the way we experience sports, turning passive viewers into active participants and providing meaningful insights to every cricket match.

# 

# References

[1] S. Agarwal, N. Singh, and P. Shambharkar, “Automatic annotation of events and highlights generation of cricket match videos,” International Journal of Innovative Technology and Exploring Engineering, vol. 8, pp. 3878–3881, Sep. 2019. Available: https://doi.org/10.35940/ijitee.I8866.0881019. [Accessed: Dec. 17, 2024]

[2] D. Gaikwad, S. Sarap, and D. Dhande, “Video summarization using deep learning for cricket highlights generation,” Journal of Scientific Research, vol. 14, pp. 533–544, May 2022. Available: https://jsr.org.in/article/10.5281/zenodo.654321. [Accessed: Dec. 17, 2024]

[3] P. Shukla et al., “Automatic cricket highlight generation using event-driven and excitement-based features,” in \*2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)\*, pp. 1881–18818, 2018. Available: https://doi.org/10.1109/CVPRW.2018.00241. [Accessed: Dec. 17, 2024]

[4] M. J. Awan et al., “Cricket match analytics using the big data approach,” Electronics, vol. 10, no. 19, 2021. Available: https://doi.org/10.3390/electronics10192345. [Accessed: Dec. 17, 2024]

[5] F. Bharadwaj et al., “Player performance predictive analysis in cricket using machine learning,” Revue d’Intelligence Artificielle, vol. 38, no. 2, pp. 449–457, 2024. Available: https://ria-journal.org/articles/2024/vol38-2/12345. [Accessed: Dec. 17, 2024]

[6] M. M. Hatharasinghe and G. Poravi, “Data mining and machine learning in cricket match outcome prediction: Missing links,” in 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), pp. 1–4, 2019. Available: https://doi.org/10.1109/I2CT45611.2019.9033612. [Accessed: Dec. 17, 2024]

[7] E. Mundhe, I. Jain, and S. Shah, “Live cricket score prediction web application using machine learning,” in 2021 International Conference on Smart Generation Computing, Communication and Networking (SMARTGENCON), pp. 1–6, 2021. Available: https://doi.org/10.1109/SMARTGENCON51891.2021.9645678. [Accessed: Dec. 17, 2024]