****

**COMSATS University Islamabad**

**Abbottabad, Pakistan**

**Video Surveillance System**

***By***

**Zain Asif CIIT/FA20-BSE-136/ATD**

**Nabeel Saleem CIIT/FA20-BSE-122/ATD**

**Hammad Ur Rehman CIIT/FA20-BSE-126/ATD**

***Supervisor*Dr. Saad Mustafa**

***Bachelor of Science in Computer Science (2020-2024)***

**The candidate confirms that the work submitted is their own and appropriate  
 credit has been given where reference has been made to the work of others**.

****

**COMSATS University, Islamabad Pakistan**

**Video Surveillance System**

**A project presented to**

**COMSATS Institute of Information Technology, Islamabad**

**In partial fulfillment**

**of the requirement for the degree of**

***Bachelor of Science in Computer Science (2020-2024)***

**By**

**Zain Asif CIIT/FA20-BSE-136/ATD**

**Nabeel Saleem CIIT/FA20-BSE-122/ATD**

**Hammad Ur Rehman CIIT/FA20-BSE-126/ATD**

**DECLARATION**

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other. We will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

Zain Asif Nabeel Saleem Hammad Ur Rehman

--------------------------- --------------------------- ---------------------------

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (SE) “Video Surveillance System” was developed by **Zain Asif (CIIT/FA20-BSE-136),** **Nabeel Saleem(CIIT/FA20-BSE-122)** and **Hamad Ur Rehman** **(CIIT/FA20-BSE-126)** under the supervision of **Dr**. **Saad Mustafa** and co supervisor **Dr. Numan ul Haq** and that in (their/his/her) opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Sciences.

---------------------------------------

**Supervisor**

---------------------------------------

**External Examiner**

---------------------------------------

**Head of Department**

**(Department of Computer Science)**

**EXECUTIVE SUMMARY**

The rising crime rates in Pakistan pose a significant threat to public safety, necessitating proactive measures to address the increasing incidents of violence. Currently, instances of violence are primarily captured through CCTV camera recordings, but this data often reaches authorities after the fact, limiting their ability to respond promptly. In response to this challenge, our proposed solution is to develop a real-time violence detection system.

The aim of this system is to swiftly detect violent activities as they occur and notify authorized users, including administrators and security officials. Traditional manual monitoring systems face challenges in identifying and responding to violence in real-time due to human limitations such as fatigue and distraction. Our proposed solution leverages artificial intelligence (AI) technology to automatically analyze video feeds for suspicious and violent activities.

By dividing videos into manageable chunks and passing them through AI models trained to detect violence, our system can quickly identify potential threats. Upon detection, authorized users are promptly notified, enabling them to take appropriate actions, such as informing responsible authorities or deploying security personnel. Additionally, users have the option to upload videos for analysis, enhancing the system's ability to detect violence in specific footage.

Our intelligent video surveillance system offers a cost-effective and efficient solution to strengthen security, reduce response times, and create a safer environment for communities. By automating the detection process and providing real-time alerts, our solution addresses the limitations of manual monitoring systems and enhances the effectiveness of security measures.

**ACKNOWLEDGEMENT**

All praise is to Almighty Allah who bestowed upon us a minute portion of His boundless knowledge by virtue of which we were able to accomplish this challenging task.

We are greatly indebted to our project supervisor “**Dr**.  **Saad Mustafa**” and our Co-Supervisor “**Dr. Numan ul Haq**”. Without their personal supervision, advice and valuable guidance, completion of this project would have been doubtful. We are deeply indebted to them for their encouragement and continual help during this work.

And we are also thankful to our parents and family who have been a constant source of encouragement for us and brought us the values of honesty & hard work.

Zain Asif Nabeel Saleem Hammad Ur Rehman

--------------------------- --------------------------- ---------------------------

**ABBREVIATIONS**

|  |  |
| --- | --- |
| **App** | Application |
| **UX** | User Experience |
| **UI** | User Interface |
| **SDLC** | Software Development Life Cycle |
| **SQE** | Software Quality Engineering |
| **HCI** | Human Computer Interaction |
| **OS** | Operating System |
| **SRS** | Software Requirement Specification |
| **SDD** | Software Design Description |
| **SE I** | Software Engineering I |
| **OOSE** | Object Oriented Software Engineering |
| **OOP** | Object Oriented Programming |
| **SRE** | Software Requirements Engineering |
| **SPM** | Software Project Management |
| **DBS** | Database Systems |
| **SD** | Sequence Diagram |
| **SSD** | System Sequence Diagram |
| **UC** | Use Case |
| **FR** | Functional Requirement |

**TABLE OF CONTENTS**

[**1 Introduction** 11](#_Toc520773734)

[1.1 Brief 11](#_Toc520773735)

[1.2 Relevance to Course Modules 11](#_Toc520773736)

[1.3 Project Background 11](#_Toc520773737)

[1.4 Literature Review 11](#_Toc520773738)

[1.5 Analysis from Literature Review (in the context of your project) 11](#_Toc520773739)

[1.6 Methodology and Software Lifecycle for This Project 11](#_Toc520773740)

[1.6.1 Rationale behind Selected Methodology 11](#_Toc520773741)

[**2** **Problem Definition** 12](#_Toc520773742)

[2.1 Problem Statement 12](#_Toc520773745)

[2.2 Deliverables and Development Requirements 12](#_Toc520773746)

[2.3 Current System (if applicable to your project) 12](#_Toc520773747)

[**3** **Requirement Analysis** 13](#_Toc520773748)

[3.1 Use Cases Diagram(s) 13](#_Toc520773750)

[3.2 Detailed Use Case 13](#_Toc520773751)

[3.3 Functional Requirements 13](#_Toc520773752)

[3.4 Non-Functional Requirements 13](#_Toc520773753)

[**4 Design and Architecture** 14](#_Toc520773754)

[4.1 System Architecture 14](#_Toc520773759)

[4.2 Data Representation [Diagram + Description] 14](#_Toc520773760)

[4.3 Process Flow/Representation 14](#_Toc520773761)

[4.4 Design Models [along with descriptions] 14](#_Toc520773762)

[**5** **Implementation** 15](#_Toc520773763)

[5.1 Algorithm 15](#_Toc520773765)

[5.2 External APIs 15](#_Toc520773766)

[5.3 User Interface 15](#_Toc520773767)

[**6 Testing and Evaluation** 16](#_Toc520773768)

[6.1 Manual Testing 16](#_Toc520773771)

[6.1.1 System testing 16](#_Toc520773772)

[6.1.2 Unit Testing 16](#_Toc520773773)

[6.1.3 Functional Testing 17](#_Toc520773774)

[6.1.4 Integration Testing 17](#_Toc520773775)

[6.2 Automated Testing: 18](#_Toc520773776)

[Tools used: 18](#_Toc520773777)

[**7** **Conclusion and Future Work** 19](#_Toc520773778)

[7.1 Conclusion 19](#_Toc520773780)

[7.2 Future Work 19](#_Toc520773781)

[**8** **References** 20](#_Toc520773782)

**LIST OF FIGURES**

Fig 3.1: Use case Diagram...……………………………………………….………………17

Fig 4.1: Deployment Diagram……………………………………………………………..26

[Fig 4. 2: Package](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156570201)Diagram……………………………………………..…………….……..27

[Fig](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156570202) 4.3: Class Diagram..…………………………………….……………………………..28

[Fig 4.4:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156570204) Admin Activity Diagram…..…...……………………………………..…………..29

[Fig 4.5:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156570205) User Acitvity Diagram……….…………………………….……………………..30

Fig 4.6: JSON Tree ……………………………………………………………………..... 31

[Fig 4.7:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156570207) Add User SD…………………………….………………..………………..……..32

[Fig 4.8:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156570208) Remove User SD…………………………..………...…………….…….……..…33

[Fig 4.9:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156570208) Live Violence Detection SD……..……………………..……………….………..34

[Fig 4.10:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156570210)Violence DetectionSD…………………….……………………….…………….35

[Fig 4.11:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156394315) NotificationSD…………………………..………………………………………36

[Fig 4.12:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156394316) Add User SSD……………….……..……..……………………………………..37

[Fig 4.13: Remove User SSD……………….…….………………………………………..38](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156394317)

[Fig 4.14:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156394318) NotificationSSD…………………………………………………….…………...39

[Fig 4.15:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156394319) Live Violence Detection SSD…………………………………………………...40

[Fig 4.16:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156394320) Detect Violence SSD…………………………………………………...…….....41

[Fig 4.17:](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156394321) Histroy Module SSD……………………………………………………..……..42

**LIST OF TABLES**

[Table](#_Toc156394406) 1.1: Literature Review……………………………………………...………………....13

[Table 2.1:](#_Toc156394410) Development Requirement……………………...………………………………..17

[Table 3.0:](#_Toc156394410) UC1- Add User……….…………………………………………………………..19

[Table 3.1:](#_Toc156394410) UC2- Remove User ...…………….…………………………………………........19

[Table 3.2:](#_Toc156394410) UC3- Detect violence …………..…………...……………………………...…….20

[Table 3.3:](#_Toc156394410) UC4- Notification...………...………………………….........................................20

[Table 3.4: UC5- Live Violence Detection………………………………](#_Toc156394410) …………...……...21

[Table 3.5:](#_Toc156394410) UC6- View History ………………………………..….………………………….22

[Table 3.6:](#_Toc156394410) FR01- Add user…………………………………………………………………...22

[Table 3.7:](#_Toc156394410) FR02-Remove User…………………………………………………….………....23

[Table 3.8:](#_Toc156394410) FR03-Detect Violence ….…………………………………………….…………..23

[Table 3.9:](#_Toc156394410) FR04-Notification.………………………………………………………………..24

[Table 3.10: FR](#_Toc156394410)05-Live Violence Detection………………………………………………….24

[Table 3.11:](#_Toc156394410) FR06-View History……………………………………………………..…….....25

1. **Introduction**

We are developing an AI system which ensures the safety of people by detecting the anomaly or suspicious activity. It divides video in chunks and the feed them into our Deep learningmodel which is trained on our dataset. Then our model process the video if it finds any violence activity related to the classes we defined like Fighting, Road Accident and Fire then it will notify the authorized user and will send detected frames.

* 1. **Brief**
* Project: Video Surveillance System android application.
* Outcome: Automates the violence/anomaly detection process, reduces time and cost of existing monitoring system.
* Features: Add user, Remove user, Detect violence/anomaly, Upload video, notification, History.
* Tools: Android Studio development platform, Flask framework for client-server development, Python and dart.
* Methodology: Iterative and incremental software development approach.
  1. **Relevance to Course Modules**

Our project, Video Surveillance System, is related to several courses studied during the Bachelor of Software Engineering (BSE) program.

* + 1. **Software Engineering**: We kept in mind the concepts we learned in software engineering about how the project lifecycle progresses, which process model is used for what kind of project etc.
    2. **Software Engineering II:** In software engineering II we studied Use case Modelling. How necessary it is to properly understand the requirements and all the aspects we will cover in the scope of our project.
    3. **Object-Oriented Programming (OOP):** OOP concepts, such as classes, objects, inheritance, and polymorphism, form the backbone of our project's implementation. Our project is developed using Dart language, which follows the OOP paradigm, enabling modular, reusable, and maintainable code.
    4. **Database Systems (DBS):** Our application leverages Firebase, a real-time DBMS, for data storage and retrieval. Concepts learned in DBMS courses, such as data modelling, normalization, and query optimization, are directly applicable to designing efficient database structures for admin and users.
    5. **Mobile Application Development:** The BSE program covers mobile application development using frameworks like Flutter for cross-platform development. Our project aligns with this coursework by utilizing Flutter to create an intuitive and responsive user interface for Video Surveillance System.
    6. **Software Quality Engineering**: This course taught us how to improve the development process so that certain issues can be prevented before they cause major problems in production. How to make quality software according to the non-functional requirements specified.
    7. **Software Testing:** Testing is a major step in software development life cycle. We studied how to test our system manually and through automation. Making test cases and how to perform different types of testing like black box and white box, unit, integration, system, and user testing.
    8. **Project Management:** Effective project management practices, including planning, scheduling, resource allocation, and risk management, are vital for project success. Our team applies these concepts to coordinate tasks, set milestones, and ensure timely delivery of project deliverables throughout the development lifecycle.
    9. **Human Computer Interaction:** We kept in mind the major principles of HCI so that the interface is more user friendly and understandable by the end user.
  1. **Project Background**

This project emphasizes the urgent requirement to tackle the increasing crime rates in Pakistan, which presents a major danger to public safety. Crime cases are going up, which demands proactive actions to ensure security and save lives of people. At the moment, instances of violence are mainly seen through CCTV camera recordings; however, this data often reaches the authorities only after the incident has happened in response to this challenge, we suggest creating a real-time violence detection System

The aim of this System is to quickly detect violent activities as they happen, and inform the admin or security official’s registered through admin access. The objective is to speed up the process, ultimately resulting in catching wrongdoers and saving lives and maintaining security. As not everyone can afford to hire a dedicated security person to continuously monitor live camera feeds due to the high associated costs. Furthermore, even with human surveillance, keeping constant watch on a computer screen all day is not practical.

Thus, the suggested solution proposes an intelligent software system that automatically detects potential threats and informs the registered users without needing constant human oversight. This approach provides a more cost-effective and efficient way to strengthen security, reduce response times, and ultimately create a safer environment for the community

Thus, the suggested solution proposes an intelligent software system that automatically detects potential threats and informs the registered users without needing constant human oversight. This approach provides a more cost-effective and efficient way to strengthen security, reduce response times, and ultimately create a safer environment for the community.

* 1. **Literature Review**

|  |  |  |
| --- | --- | --- |
| **Application Name** | **Weakness** | **Proposed Project Solution** |
| Efficient Violence Detection in Surveillance | Efficient Violence Detection in Surveillance uses LSTM model which is more complicated than traditional RNNs and require more training data in order to learn effectively  . | Our system will use Pre train model (VGGA, Reset 50, and mobile net V2) which excels in processing spatial and temporal information in videos. |
| The Lahore Safe City Project | The main objective of the Safe City project is to develop the security and surveillance system in the city on modern lines, for which 10 thousand new CCTV cameras will be installed at important places in the city But this system is only for monitoring purposes not for violence detection | The primary purpose of our Violence Detection System is to detect violence activities. |

*Table 1. 1: Literature Review*

* 1. **Analysis from Literature Review (in the context of your project)**

Video Surveillance System addresses the limitations identified in the literature review projects while offering a comprehensive solution for enhancing public safety through violence detection.

* **Utilization of Pre-Trained Models:** By leveraging pre-trained models, our system simplifies the architecture and training process, overcoming the complexity associated with LSTM models.
* **Focus on Violence Detection:** While projects like the Lahore Safe City initiative enhance surveillance capabilities, our system specifically targets violence detection, providing a proactive approach to public safety.
  1. **Methodology and Software Lifecycle for This Project**

We chose the object-oriented approach for our project.

* The object-oriented approach concentrates on capturing the structure and behaviour of information systems into small modules that combine data and processing. The primary goal of Object-Oriented Design is to improve the efficiency and effectiveness of system design by making it easier to use.

In terms of the software development life cycle (SDLC), the Incremental and iterative methodology was chosen for this project.

* **Continuous Improvement:** The nature of our project involves complex tasks such as creating a dataset developing AI models, integrating various modules, and ensuring real-time performance. Incremental and iterative development allows for continuous improvement and refinement of the system through multiple iterations.
* **Flexibility and Adaptability:** The requirements for our project evolve over time, requiring flexibility in the development process. Incremental and iterative approaches enable us to adapt the changing requirements and incorporate changes in iterations.
  + 1. **Rationale behind Selected Methodology**

The above methodology is selected because following advantages of Object- Oriented Approach:

1. **Reduced Maintenance:** The main objective is to guarantee that the system will last longer with significantly lower maintenance costs. The behaviors can be reused and incorporated into new behaviors because most system processes are encapsulated.
2. **Real World Modeling:** Object oriented systems often model the real world more thoroughly than traditional approaches. Objects are classified into classes, and objects and behaviors are linked together. The model is built on objects rather than data and computation.
3. **Improved Reliability and Flexibility:** Because new behaviors may be "built" from pre-existing objects, object-oriented systems are far more likely to be trustworthy than traditional systems. Because existing objects can be dynamically called upon and accessed, the construction of new objects is always feasible. Other objects may provide one or more data properties to the new objects. Super-class behaviors can be inherited, and new behaviors can be introduced with no effect on how the system currently works.
4. **High Code Reusability:** When a new object is generated, it inherits the data attributes and properties of the class from which it was produced. The data and behaviors of all super classes in which the new object participates will likewise be inherited by the new object. When a user creates a new type of widget, the new item acts "wiggly," with new behaviors defined to the system.

.

Incremental and iterative approaches offer several advantages for developing a video surveillance system.

1. **Early Delivery of Key Features**: Incremental development allows for the early delivery of essential functionalities, such as real-time monitoring and basic detection capabilities.
2. **Flexibility to Adapt**: Iterative development enables the system to adapt to changing requirements and emerging security threats over time. This flexibility allows for the incorporation of new technologies, algorithms, and user feedback, ensuring the system remains relevant and effective in the long term.
3. **Continuous Improvement**: Incremental and iterative approaches facilitate continuous improvement through frequent iterations and feedback loops. By iterating on the system in small increments, we can identify and address issues promptly, leading to a more refined and reliable surveillance solution.
4. **Reduced Risk and Cost**: Breaking the development process into manageable increments reduces the risk of project failure and allows for better risk management. Additionally, by focusing on delivering small, achievable goals, iterative development can help control costs and avoid unnecessary expenses associated with large-scale implementations.
5. **Problem Definition**

The problem at hand revolves around the need for a robust and efficient video surveillance system capable of detecting and preventing violent and suspicious activities in various environments. Traditional surveillance methods are often limited in their effectiveness, relying heavily on human, leading to delayed responses and compromised security.

3. 1. **Problem Statement**

The manual monitoring systems faces a critical challenge in effectively identifying and responding to violence and suspicious activities in real-time. Humans can be fatigued and easily distracted when monitoring video feeds for extended periods. This can result in missed events or delayed reactions to suspicious activities. Employing human security personnel 24/7 is expensive, and the cost increases with the need for multiple shifts.

* 1. **Deliverables and Development Requirements**
     1. **Deliverables:**
* Design and development of an intuitive and user-friendly interface for the Android application.
* Fully functional mobile app from which user can interact with system.
* Implementation of user authentication mechanisms, including login functionality.
* Implementation of a notification system to alert users in real-time upon detecting violent or suspicious activities.
* Implementing upload video module allowing users to upload videos from their devices for violence detection.
* Designing the architecture of the deep learning model for violence/anomaly detection.
* Testing the trained model on our Data set to validate its performance in real-world scenarios.
* Integration of Android application with Flask backend APIs to ensure seamless communication between the client and server.
  + 1. **Development Requirements:**

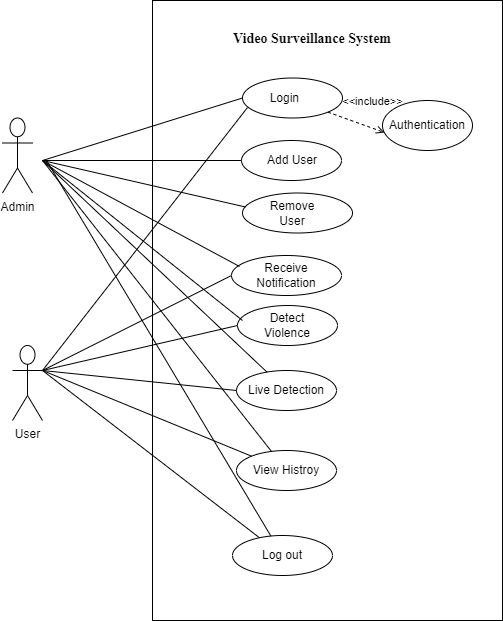
|  |  |  |  |
| --- | --- | --- | --- |
| **Tools**  **And**  **Technologies** | **Tools** | **Version** | **Rationale** |
| Android Studio | 2022 | IDE |
| Flutter SDK | 2023 | UI toolkit |
| Firebase | CSC 6 | Real-time DBMS |
| Star UML | 3.2.2 | System Modeling |
| MS Word | 2022 | Documentation |
| MS Power Point | 2022 | Presentation |
| Figma | 2.0.5 | Mockups Creation |
| **Technology** | **Version** | **Rationale** |
| Dart | 6.0 | Programming language |
| UML | V5.1.0 2013 | Modeling Language |
| Flutter | 3.13 | Mobile Application development |

*Table 1.* ***2****: Development Requirement*

1. **Requirement Analysis**
3. 1. **Use Cases Diagram(s)**

In this section we will discuss the use case diagram for Video Surveillance System.

Fig 3.1 represents the use case diagram Video Surveillance System.

****

*Fig 3. 1: use case*

* 1. **Detailed Use Cases**
     1. **Module 1: Add users**

|  |  |
| --- | --- |
| **Use Case ID:** | UC1 |
| **Use Case Name:** | Add user |
| **Actors:** | Admin |
| **Description:** | Admin will add user by setting their credential which include email and password |
| **Trigger:** | This use case is triggered when an admin decides to create a new user account within the system |
| **Preconditions:** | PRE-1 Internet working  PRE-2 should be an Admin |
| **Post conditions:** | POST-1 User will added successfully |
| **Normal Flow:** | 1. Admin will go to add new user. 2. Admin will set an email. 3. Admin will set a password. 4. User will be added in system. |
| **Exceptions:** | None |
| **Business Rules:** | New User will be added easily. |
| **Assumptions:** | The system has all the capabilities to add user |

*Table 3.0: Add User UC*

* + 1. **Module 2: Remove users**

|  |  |
| --- | --- |
| **Use Case ID:** | UC2 |
| **Use Case Name:** | Remove user |
| **Actors:** | Admin |
| **Description:** | Admin will remove user from authorized user list |
| **Trigger:** | This use case is triggered when an admin clicks on remove user to remove an existing user account within the system |
| **Preconditions:** | PRE-1 Internet working  PRE-2 should be an Admin  PRE-3 User to be removed should be member of organization. |
| **Post conditions:** | POST-1 User will removed successfully |
| **Normal Flow:** | 1. Admin will go to remove user. 2. Admin will select the user to be removed. . 3. Admin will remove the user |
| **Exceptions:** | None |
| **Business Rules:** | User will removed from system easily. |
| **Assumptions:** | The system has all the capabilities to remove user |

*Table 3.1: Remove User UC*

* + 1. **Module 3: Detect Violence**

|  |  |
| --- | --- |
| **Use Case ID:** | UC3 |
| **Use Case Name:** | Detect violence |
| **Actors:** | User |
| **Description:** | User will upload the video from the gallery in which user want to detect the Violence. |
| **Trigger:** | This use case is triggered when user clicks on detect violence/anomaly to detect violence/anomaly in a particular video |
| **Preconditions:** | PRE-1 Internet working  PRE-2 should be an authorized user  PRE-3 Video should be in proper format  PRE-4 video size should be within 5 minutes duration range.  PRE-5 Type of violence/anomaly should belong to any of our define classes |
| **Post conditions:** | POST-1 Video should be upload on the server  POST-2 server will send the response |
| **Normal Flow:** | 1. User will go to upload video module. 2. User will select the video from gallery. 3. User will click on upload video. 4. Video will converted into frames 5. Video will be uploaded to server. |
| **Alternative Flows:** | System will ask user to recheck the size and format of the video and redirect to upload video module. |
| **Exceptions:** | None |
| **Business Rules:** | Video should be upload within five minute duration range. User should have stable internet connection. |
| **Assumptions:** | The system can accept video within five minute duration range and mp4 format |

*Table 3.2: Detect Violence UC*

* + 1. **Module 4: Notification**

|  |  |
| --- | --- |
| **Use Case ID:** | UC4 |
| **Use Case Name:** | Notification |
| **Actors:** | User ,Admin |
| **Description:** | Notification module will notify user and admin if any violence/anomaly is detected by our model and also give what type of violence/anomaly is detected. |
| **Trigger:** | This use case is triggered by server when any violence is detected by model the server will send the notification. |
| **Preconditions:** | PRE-1 Internet working  PRE-2 should be an authorized user |
| **Post conditions:** | POST-1 All users will receive the notification of violence detection. |
| **Normal Flow:** | 1. Video will be uploaded to the server 2. Violence will be detected by model 3. Server will send the violence detection notification to user and admin. |
| **Alternative Flows:** | None |
| **Exceptions:** | None |
| **Business Rules:** | System will send notification to the user efficiently. |
| **Assumptions:** | When violence is detected server will send notification successfully. |

*Table 3.3: Notification UC*

* + 1. **Module 5: Live Violence Detection**

|  |  |
| --- | --- |
| **Use Case ID:** | UC5 |
| **Use Case Name:** | Live Violence Detection |
| **Actors:** | Server |
| **Description:** | Live Violence Detection module will detect the violence in live footage. This footage will pass to the server in small patches and model we examine these small patch one by one and give result back. |
| **Trigger:** | This use case is triggered by camera to check any kind of violence in live footage. |
| **Preconditions:** | PRE-1 Internet working  PRE-2 camera should be working properly  PRE-3 There should be proper light |
| **Post conditions:** | POST-1 Admin and user will receive the notification if there is any violence activity in a patch. |
| **Normal Flow:** | 1. Live footage will divided into small patches and each patch will send on a server. 2. Violence detector model will process the frames of patch and give result back to the server. |
| **Alternative Flows:** | System ask user to check if camera is working properly or not and footage is divided into patches correctly or not. |
| **Exceptions:** | None |
| **Business Rules:** | Model will try to accurately identify the violence in the live footage |
| **Assumptions:** | None |

*Table 3.4: Live Violence Detection UC*

* + 1. **Module 6: View History**

|  |  |
| --- | --- |
| **Use Case ID:** | UC6 |
| **Use Case Name:** | View History |
| **Actors:** | User, Admin |
| **Description:** | In View History module user and admin can view the patch of footage that contain violence. |
| **Trigger:** | This use case is triggered by user and admin to view the violence evidence. |
| **Preconditions:** | PRE-1 Internet working  PRE-2 Authorized User |
| **Post conditions:** | POST-1 Admin and user will view the violence history |
| **Normal Flow:** | 1. User or admin will login to system. 2. User will go to the view history module. 3. User will view the history. |
| **Alternative Flows:** | None |
| **Exceptions:** | None |
| **Business Rules:** | After violence detected it should be send back to application efficiently. |
| **Assumptions:** | System can save list of violence footages. |

*Table 3.5: View History UC*

* 1. **Functional Requirements**
     1. **Module 1: Add User**

|  |  |
| --- | --- |
| **Identifier** | REQ001 |
| **Title** | Add User Functionality |
| **Requirement** | Add user functionality allow admin to add new user to the system:  Admin can add new user by setting there credential like email and password |
| **Source** | Admin |
| **Rationale** | This requirement aims to provide admin with a simple, efficient, and flexible way to add new user to the system. |
| **Business Rule** | User should be member of that particular organization no outside person will allow to become user. |
| **Dependencies** | None |
| **Priority** | High |

*Table 3.6: Add User FR 01*

* + 1. **Module 2: Remove users**

|  |  |
| --- | --- |
| **Identifier** | REQ002 |
| **Title** | Remove users Functionality |
| **Requirement** | Remove user functionality allow admin to Remove user from system. Only admin can remove user from the system. |
| **Source** | Admin |
| **Rationale** | This requirement aims to provide admin with a simple and efficient way to remove existing user from system. |
| **Business Rule** | None |
| **Dependencies** | None |
| **Priority** | Medium |

*Table 3.7: Remove User FR 02*

* + 1. **Module 3: Detect Violence**

|  |  |
| --- | --- |
| **Identifier** | REQ003 |
| **Title** | Upload Video Functionality |
| **Requirement** | User can able to upload video on the server from the gallery to check is there any violence in the footage or not. |
| **Source** | User |
| **Rationale** | This requirement aims to provide user a method from which user can upload the footage on the server from its gallery and check whether it contain any violence activity or not. |
| **Business Rule** | Upload video should only in allowed format and size |
| **Dependencies** | None |
| **Priority** | High |

*Table 3.8: Detect Violence FR 03*

* + 1. **Module 4: Notification**

|  |  |
| --- | --- |
| **Identifier** | REQ004 |
| **Title** | Notification Functionality |
| **Requirement** | Notification Functionality will send notification to the admin and user if violence activity detected in the uploaded footage. Notification will be receive on user mobile app. |
| **Source** | Server |
| **Rationale** | This requirement aims to timely notify the user and admin about any violence activity. The will also save time and improve efficiency as user don’t have to view continuously for violence activity. |
| **Business Rule** | None |
| **Dependencies** | None |
| **Priority** | High |

*Table 3.9: Notification FR 04*

* + 1. **Module 5: Live Detection**

|  |  |
| --- | --- |
| **Identifier** | REQ005 |
| **Title** | Live Violence/ anomaly Detection Functionality |
| **Requirement** | Live Violence/anomaly Detection model will detect violence activity in real time. Footages from camera are divided into small patches and are sent on server where model detects violence/ anomaly activity. |
| **Source** | User, Admin |
| **Rationale** | This requirement aims to detect violence in real time to reduce the human error factor. Continuously monitoring can cause human fatigue and as a result important event will be missed. |
| **Business Rule** | Camera should be working proper and in correct position. |
| **Dependencies** | None |
| **Priority** | High |

*Table 3.10: Live Detection FR 05*

* + 1. **Module 6: View History**

|  |  |
| --- | --- |
| **Identifier** | REQ006 |
| **Title** | View History Functionality |
| **Requirement** | View History functionality will allow user to view the video frames that contain violence. This functionality is also use for evidence purpose. |
| **Source** | User |
| **Rationale** | This requirement aims to save video frames that contain violence activity. These violent frames can also use as video evidence against the criminals. |
| **Business Rule** | None |
| **Dependencies** | None |
| **Priority** | High |

*Table 3.11: Add User FR 06*

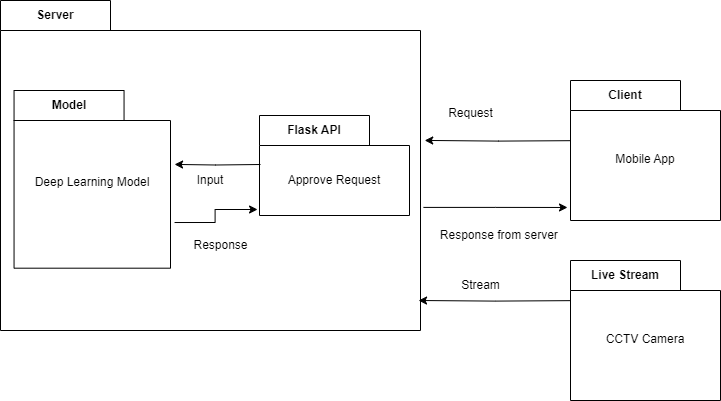
* 1. **Non-Functional Requirements**
     1. **NFR 1: Usability**
* **User Interface (UI):** The application should have an intuitive and user-friendly interface, allowing easy navigation for both administrators and registered users.
* **Accessibility:** The system should be accessible to all users ensuring that the registration process and violence detection results are easily understandable.
  + 1. **NFR 2: Performance**
* **Response Time:** The system should provide real-time notifications, ensuring that authorized users receive alerts upon the detection of violent incidents.
* **Throughput:** The uploaded video will be processed in few seconds and after that system will be available for uploading another video.
* **Scalability:** The system should be scalable to accommodate an increasing number of registered users without compromising performance.
  + 1. **NFR 3: Security**
* **Authentication and Authorization:** User authentication and authorization mechanisms should be robust to ensure that only authorized personnel can access the system and receive notifications.
  + 1. **NFR 4: Maintainability**
* **Modularity:** The system should be modular, allowing for easier updates and maintenance of individual components without affecting the entire application.
  + 1. **NFR 5: Interoperability**
* **API Compatibility:** The system should support integration with external APIs for future enhancements or collaboration with other security systems.
* **Data Exchange:** The application should facilitate data exchange with other authorized security systems if necessary.

1. **Design and Architecture**



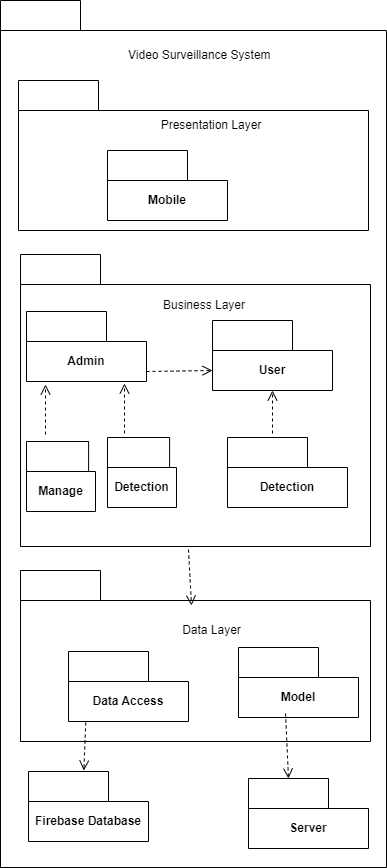
6. 1. **System Architecture**

**4.1.1. Deployment Architecture**

****

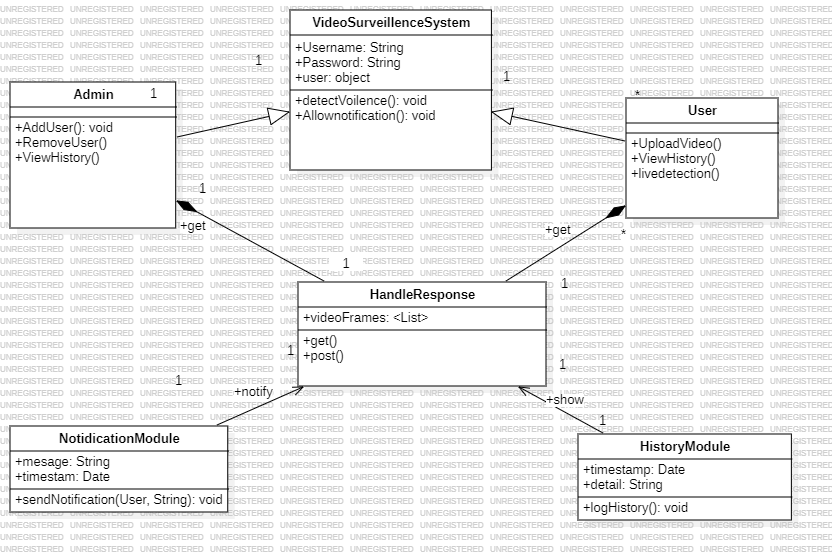
*Fig 4.1: Deployment Architecture*

* + 1. **Package Diagram**

****

*Fig 4.2: Package Diagram*

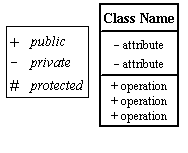
* 1. **Data Representation [Class Diagram ]**

****

*Fig 4.3: Class Diagram*

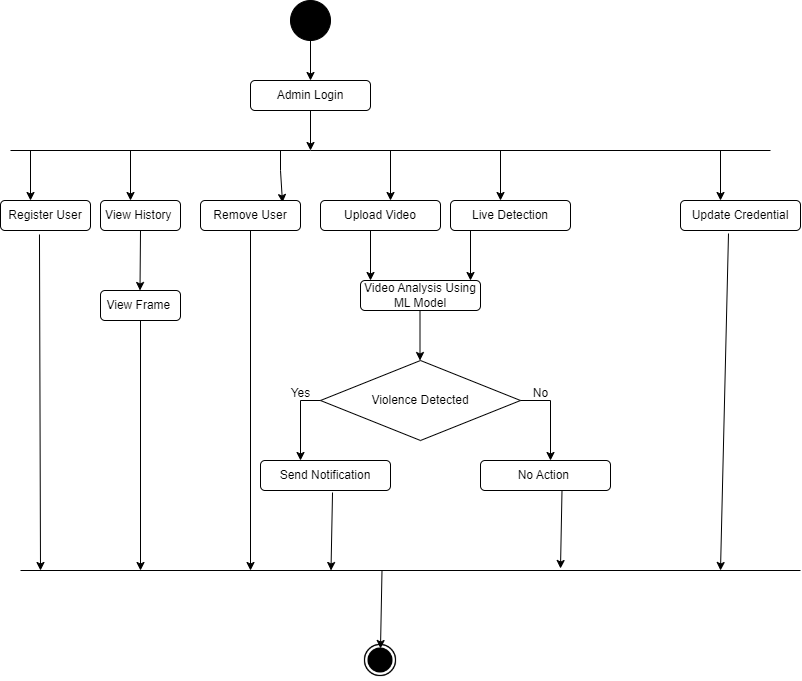
**Legends**

A white background with black text

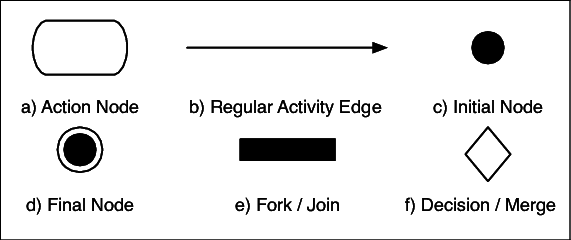
Description automatically generatedA black line on a white background

Description automatically generated

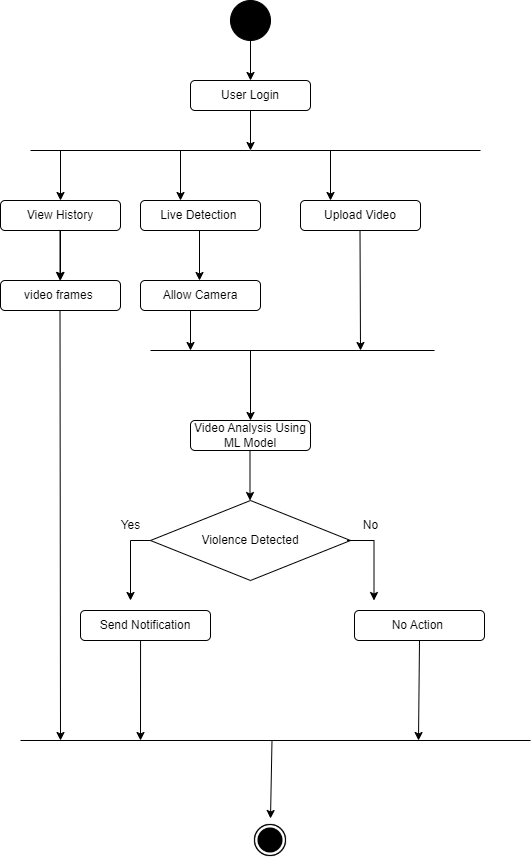
* 1. **Process Flow/Representation**
     1. **Admin Activity Diagram:**

****

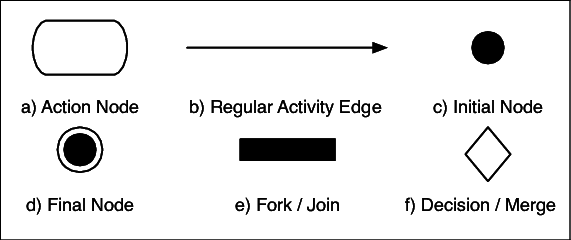
*Fig 4.4: Admin Activity Diagram*

**Legends**

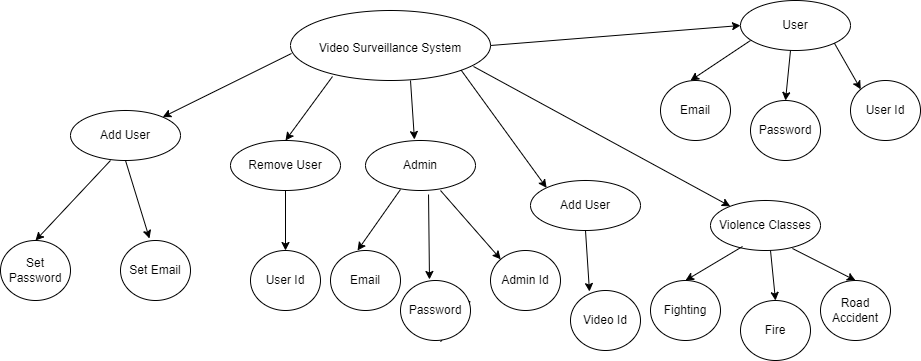
* + 1. **User Activity Diagram :**

****

**Legends**

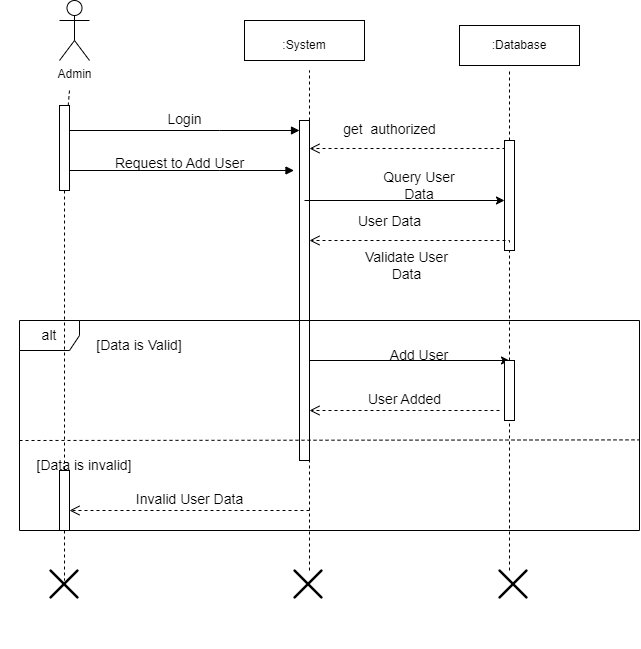
 *Fig 4.5: User Activity Diagram*

* + 1. **JSON Tree :**

****

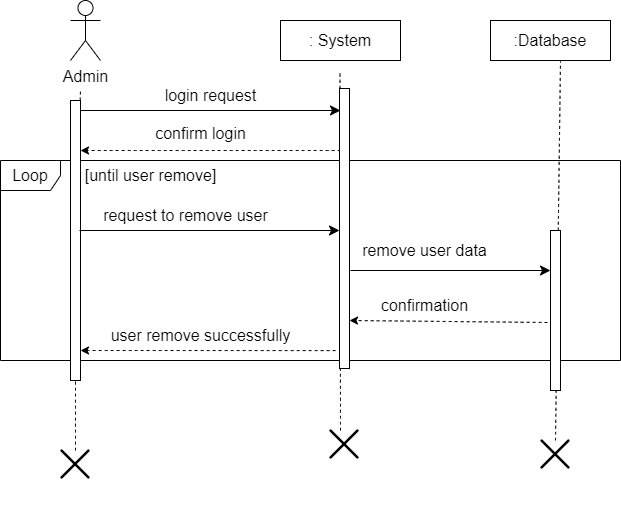
*Fig 4.6: JSON Tree*

* 1. **Design Models** 
     1. **Add User Sequence Diagram**

****

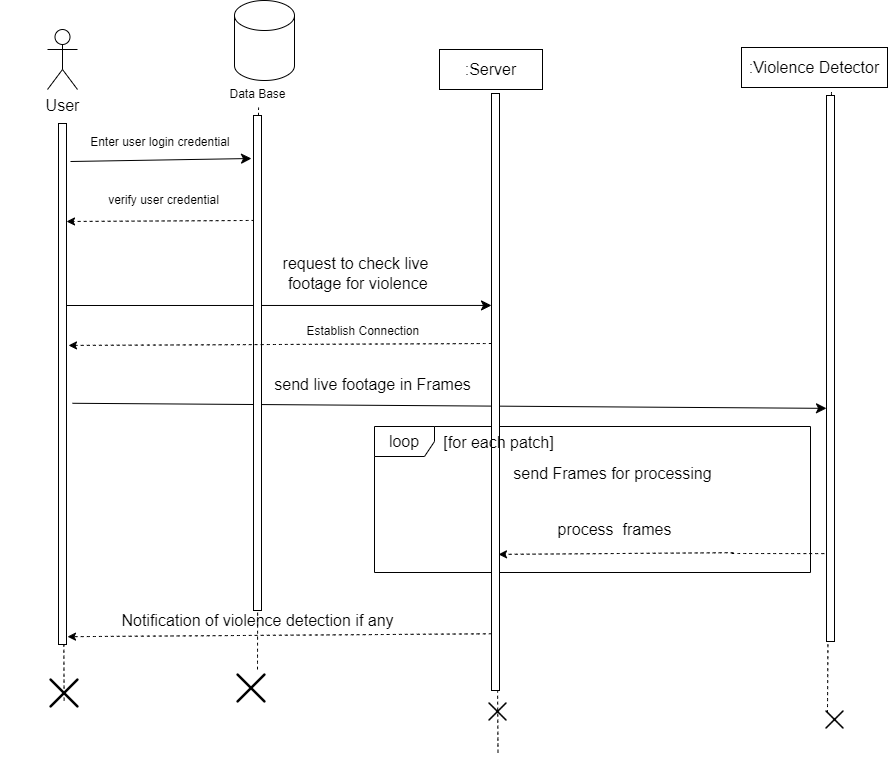
*Fig 4.7: Add User Sequence Diagram*

* + 1. **Remove User Sequence Diagram**

****

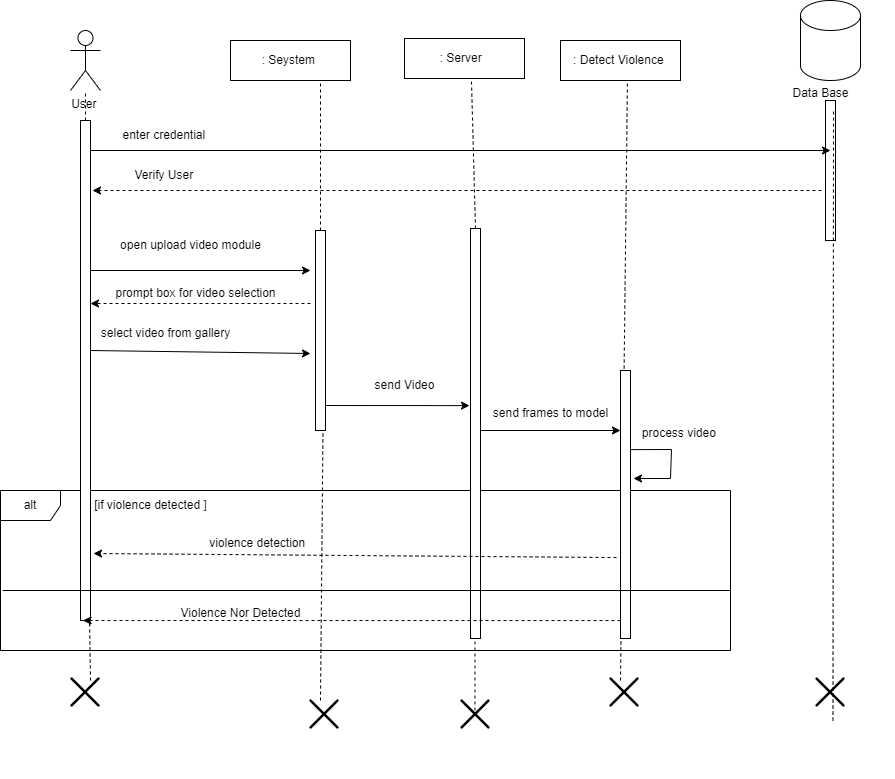
*Fig 4.8: Remove User Sequence Diagram*

* + 1. **Live Violence Detection Sequence Diagram**

****

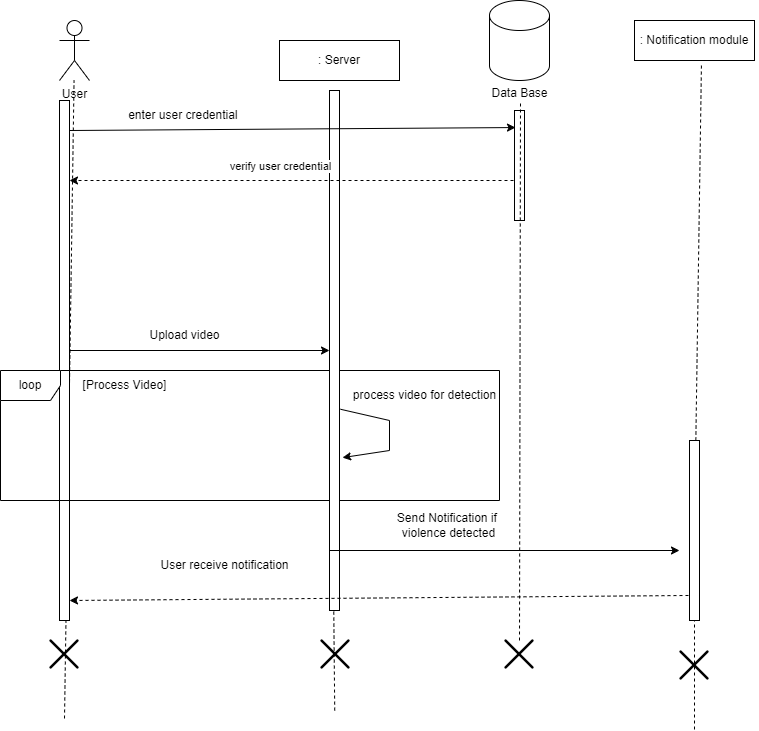
*Fig 4.9: Live Violence Detection Sequence Diagram*

* + 1. **Detect Violence Sequence Diagram**

****

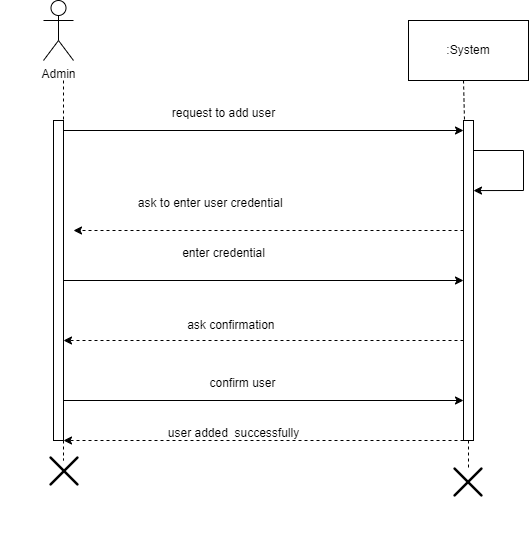
*Fig 4.10: Detect Violence Sequence Diagram*

* + 1. **Notification Sequence Diagram**

****

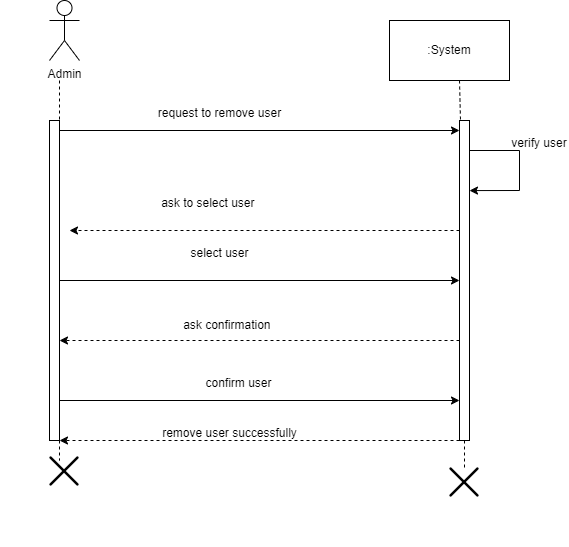
*Fig 4.11: Notification Sequence Diagram*

* + 1. **Add User System Sequence Diagram**

****

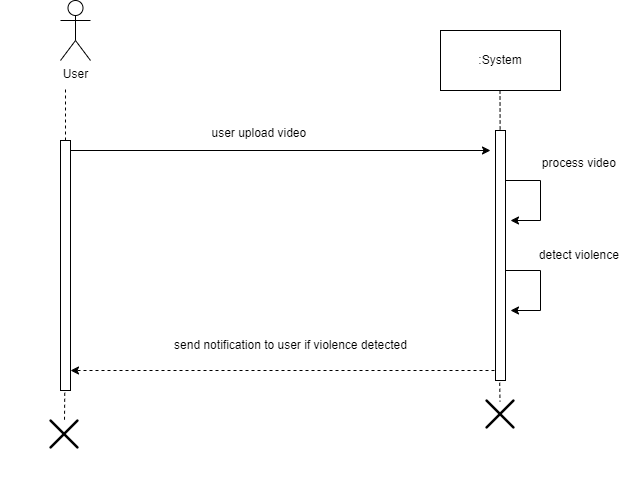
*Fig 4.12: Add User System Sequence Diagram*

* + 1. **Remove User System Sequence Diagram**

****

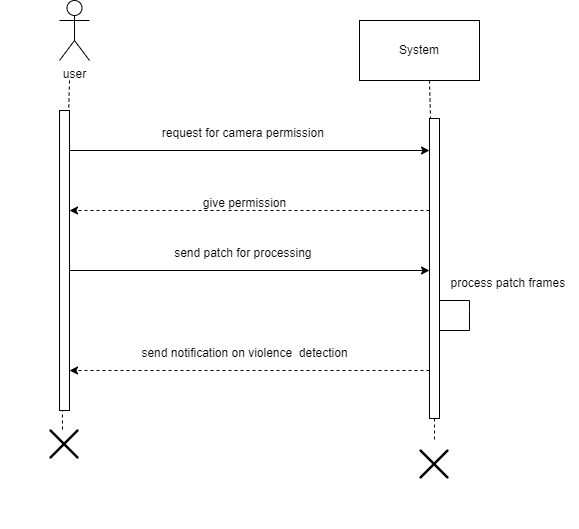
*Fig 4.13: Remove User System Sequence Diagram*

* + 1. **Notify User System Sequence Diagram**

****

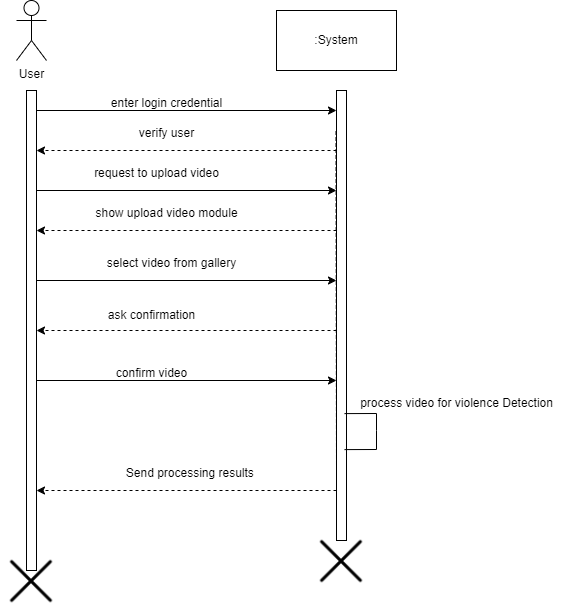
*Fig 4.14: Notification System Sequence Diagram*

* + 1. **Live Detection System Sequence Diagram**

****

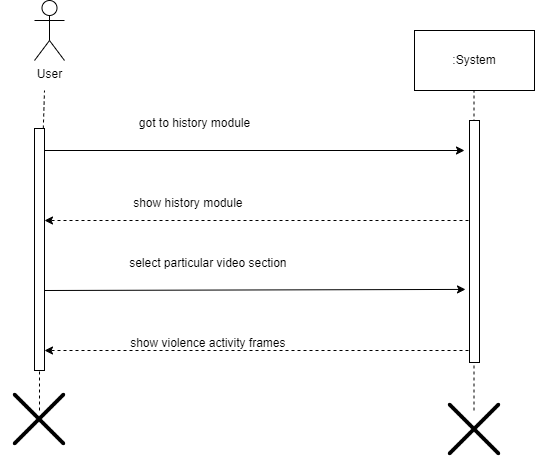
*Fig 4.15: Live Violence Detection System Sequence Diagram*

* + 1. **Detect violence System Sequence Diagram**

****

*Fig 4.16: Detect Violence System Sequence Diagram*

* + 1. **History Module System Sequence Diagram**

****

*Fig 4.17: History System Sequence Diagram*

1. **Testing and Evaluation**
   1. **Manual Testing**
      1. **System testing**

* + 1. **Unit Testing**
* **Unit Testing 1:** Login
* **Testing Objective:** To ensure the login is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Verify login with valid admin credentials | Email:  admin@gmail.com  Password: admin@1234 | Successfully log into the admin dashboard | Pass |
| 2 | Verify login with valid user credentials | Email:  nabeelsaleem@gmail.com  Password: user@1234 | Successfully log into the user dashboard | Pass |
| 3 | Verify login with incorrect admin email | Email:  zain@gmail.com  Password:  admin@1234 | Error message: "Invalid Credential" | Pass |
| 4 | Verify login with incorrect user email | Email:  nabeel@gmail.com  Password:  user@1234 | Error message: "Invalid Credential" | Pass |
| 5 | Verify login with incorrect admin password | Email:  admin@gmail.com  Password:  123456 | Error message: "Invalid Credential" | Pass |
| 6 | Verify login with incorrect user password | Email:  nabeelsaleem@gmail.com Password:  123456 | Error message: "Invalid Credential" | Pass |
| 7 | Verify login with empty email | Email: (empty)  Password: user1234 | Error message: "please enter your email" | Pass |
| 8 | Verify login with empty password | Email: nabeelsaleem@example.com Password: (empty) | Error message: " please enter your password" | Pass |
| 9 | Verify login with both email and password empty | Email: (empty)  Password: (empty) | Error message: " please enter your email "  please enter your password | Pass |

* **Unit Testing 2:** Add User
* **Testing Objective:** To ensure Add user is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Results** |
| 1 | Add user with valid credentials | Email:  hammadahmed@gmail.com Password: hammad@123 | User is successfully added to the system | Pass |
| 2 | Add user with existing email | Email:nabeelsaleem@gmail.com Password: newpass@123 | Error message: “email address is already use by another user ” | Pass |
| 3 | Add user with invalid email format | Email:  hammad.com  Password: hammad123 | Error message: "please enter a valid email address " | Pass |

* **Unit Testing 3:** Remove User
* **Testing Objective:** To ensure Remove User is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Verify admin login | Email: admin@gmail.com Password: admin@123 | Successfully log into the admin panel | Pass |
| 2 | Remove selected user | Checkbox: Checked | Selected user is remove | Pass |
| 3 | If no user selected | Checkbox: Unchecked | Error message: "please select the user to delete" | Pass |

* **Unit Testing 4:** Changing Admin Password
* **Testing Objective:** To ensure Admin Password change successfully.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Change admin password with incorrect old password | Old Password:  Zain123 | Error Message” please enter the correct old password” | pass |
| 2 | Change admin password with empty old password | Old Password:  empty | Error Message:” please enter your password” | pass |
| 3 | Set and confirm new admin password with correct old password | Old Password:  admin@1234 | Admin password change successfully | pass |

* + 1. **Functional Testing**
* **Functional Testing 1:** Login as Admin
* **Objective**: To ensure that app correctly navigate to Admin dashboard

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Login as Admin with correct credentials | Username: zainasif@gmail.com  Password: zain@123 | Successfully logs into the main page as admin | Pass |
| 2 | Login with incorrect username | Username: zain@gmail.com  Password: zain@123445 | Displays error message "Invalid Credential" | Pass |
| 3 | Login with incorrect password | Username: zainasif@gmail.com  Password: z123 | Displays error message "Invalid Credential" | Pass |
| 4 | Login with empty username | Username:  Password: zain@123 | Displays error message "please enter the email" | Pass |
| 5 | Login with empty password | Username: zainasif@gmail.com  Password: | Displays error message ""please enter the password" | Pass |
| 6 | Login with empty username and password | Username: (Empty)  Password: (Empty) | Displays error messages ""please enter the email "  "please enter the password | Pass |

* **Functional Testing 2:** Login as User
* **Objective**: To ensure that app correctly navigate to User dashboard

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Login as User with correct credentials | Username: nabeelsaleem@gmail.com  Password: nabeel@123 | Successfully logs into the main page as user | Pass |
| 2 | Login with incorrect username | Username: nabeel@gmail.com  Password: nabeel@123 | Displays error message "Invalid credential" | Pass |
| 3 | Login with incorrect password | Username: nabeelsaleem@gmail.com  Password: n123 | Displays error message "Invalid username or password" | Pass |
| 4 | Login with empty username | Username:  Password: nabeel123 | Displays error message "please enter your email" | Pass |
| 5 | Login with empty password | Username: nabeelsaleem@gmail.com  Password: | Displays error message " please enter your password" | Pass |
| 6 | Login with empty username and password | Username: (Empty)  Password: (Empty) | Displays error messages "please enter the email "  "please enter the password | Pass |

* **Functional Testing 3:** Upload Video
* **Testing Objective:** To ensure Upload Video is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Upload video with valid format | Video File: validvideo.mp4 | Video is successfully uploaded | Pass |
| 2 | Upload video with invalid format | Video File: file.pdf | File not accepted | Pass |
| 3 | No video selected | Video File: empty | Error message: "No video selected from gallery" | Pass |

* **Functional Testing 4:** Detect Violence
* **Testing Objective:** To ensure Detect Violence is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Upload a valid video for violence detection | Video file: validvideo.mp4 | Video is successfully uploaded and processed for violence detection Result is returned | Pass |
| 2 | Upload a video with an unsupported format | Video file: file.pdf | Displays error message "file will not selected"  Video is not uploaded | Pass |
| 3 | No video selected from gallery | Video file: empty | Displays error message "please select the video from gallery" | Pass |
| 4 | Upload a video with no violence detected | Video file: noviolence.mp4 | Video is processed  Result: "No frames return" | Pass |

* **Functional Testing 5:** Live Violence Detection
* **Testing Objective:** To ensure Live Violence Detect is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Detect violence in live footage | Live footage: Contains violence | Violence is detected in live footage  Notification is sent to user and admin | Pass |
| 2 | No violence detected in live footage | Live footage: No violence | No violence is detected  No notification is sent | Pass |

* **Functional Testing 6:** Notification
* **Testing Objective:** To ensure Notification is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Receive notification for detected violence | Detected violence in video: yes | Notification is sent to user and admin  Notification contains details of detected violence | Pass |
| 2 | No notification for no detected violence | Detected violence in video: no | No notification is sent | Pass |

* **Functional Testing 7:** History
* **Testing Objective:** To ensure History Module is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | View history of detected violence | User: authorized | User/admin can view all frames that contain violence | Pass |
| 2 | View history with no detected violence | User: authorized | No violence frames are showed" | Pass |

* + 1. **Integration Testing**

**7 Conclusion and Future Work**

* 1. **Conclusion**

The proposed video surveillance system offers a significant advancement in enhancing public safety by addressing the urgent need for real-time violence detection in high-crime areas like Pakistan. Using advanced machine learning models such as VGGA, ResNet 50, and MobileNet V2, the system can accurately identify violent activities as they occur and immediately notify authorities, facilitating quick interventions. This automated approach not only improves accuracy but also reduces the need for constant human monitoring, making it a cost-effective and practical solution. With user-friendly features for managing access, uploading videos, and reviewing incident history, the system is versatile and scalable, suitable for various settings from small businesses to large urban areas. By offering a proactive, efficient, and reliable method of surveillance, this system represents a crucial step towards creating safer communities.

* 1. **Future Work**

1. **Add more violence Classes:**

Adding more violence classes to the system make system more diverse and efficient and help us to detect different kind of violence activity.

1. **Real-Time Data Processing :**

Enhance the system's ability to process and analyze live video feeds more efficiently using real-time analytics to reduce latency and improve response times.

1. **Enhanced User Interface:**

Develop a more intuitive and user-friendly interface with better visualization tools, such as interactive dashboards, to help users easily navigate and manage the system.

1. **Scalability and Flexibility:**

Improve the system’s scalability to handle more cameras and higher data volumes, and make it flexible to adapt to different environments and requirements.

1. **Improved Notification System:**

Enhance the notification system to include multiple channels (SMS, email, mobile app alerts) and provide more detailed and contextual information about detected incidents.

1. **Privacy and Data Security:**

Implement advanced security methods and secure data storage solutions to protect sensitive information and ensure user privacy.

1. **Regular Updates and Maintenance:**

Regular software updates and system maintenance to keep the technology as per current standard and address any emerging threats or vulnerabilities.

1. **Feedback and Continuous Improvement:**

Create a feedback mechanism where users can report issues and suggest improvements, allowing continuous refinement and enhancement of the system based on real-world usage and requirements.

# **References**

Al-Shahmeh, M. M. (2023). Flutter Apps Development: Build Cross-Platform Flutter Apps with Trust. Damascus, Syria: Independantly Published.

# FIREBASE -OVERVIEW AND USAGE August 2022 Journal of Engineering and Technology Management

Esraa Hassan, H. T. (January 2021). Medical Prescription Recognition using Machine Learning. 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0973-0979). NV, USA: IEEE.

Galen, D. (2018). Software Quality, Concepts and Practice. Hoboken, New Jersey: Wiley, IEEE Press.

Karl Wiegers, J. B. (2013). Software Requirements, Third edition. Unterschleissheim, Germany: Microsoft Press.

Marco Ortu, G. D. (May 2015). Measuring and Understanding the Effectiveness ofJIRA Developers Communities. Research Gate, 8-9.

Pavel Hamet, J. T. (April 2017). Artificial intelligence in medicine. Metabolism, S36-S40.

Payne, R. (December 2019). Beginning App Development with Flutter: Create Cross-Platform Mobile Apps. New York: Apress.

Phillip A. Laplante, M. K. (2022). Requirements Engineering for Software and Systems. New York: Auerbach Publications.

Raschka, S. (September 2015). Python Machine Learning. Birmingham, UK: Packt Publising Ltd.

# 