# SceneSolver: AI-Powered Forensic Analysis Platform

## ****Team Information****

**Team Number:** G348  
**Team Members:**

* **Rythma Reddy Lakkady** (23BD1A056K)
* **Rishab Deshpande** (23BD1A056G)
* **Koppula Tusshar** (23BD1A660Y)
* **Leela** (23BD1A664H)
* **Aditya Panyala** (23BD1A0505)
* **Tripura** (23BD1A057W)

## ****1. Introduction****

### ****1.1 Purpose****

SceneSolver is an AI-powered forensic platform that automates crime scene analysis using CLIP and Vision Transformers (ViT). It processes crime scene images to identify crime types, extract key evidence, and generate crime scene summaries. The system enhances forensic investigations by enabling professionals to analyze large volumes of visual data efficiently.

### ****1.2 Scope****

SceneSolver provides forensic professionals with AI-driven tools to:

* **Identify crime types** from images and videos.
* **Extract key visual evidence** to aid investigations.
* **Generate automated crime scene summaries** for reporting.
* **Support batch processing** to analyze multiple crime scene images efficiently.
* **Store and manage crime scene data** securely using MongoDB.

The platform integrates **CLIP for crime classification** and **ViT for evidence extraction** while using a React.js-based frontend and a backend powered by **Node.js, Express.js, and Flask**.

### ****1.3 Definitions, Acronyms, and Abbreviations****

* **AI** – Artificial Intelligence
* **CLIP** – Contrastive Language-Image Pretraining
* **ViT** – Vision Transformer
* **CRUD** – Create, Read, Update, Delete
* **ML** – Machine Learning
* **MongoDB** – NoSQL database used for storing crime scene data

### ****1.4 References****

* CLIP model documentation (OpenAI)
* Vision Transformer documentation
* UCF Crime Dataset reference
* MongoDB, Node.js, Express.js, Flask, and React.js documentation
* Deployment guides for AWS and Vercel

### ****1.5 Overview****

This document provides a comprehensive specification of SceneSolver, covering:

* Project perspective
* Features and functionalities
* System interfaces
* Assumptions and constraints

## ****2. Overall Description****

### ****2.1 Project Perspective****

SceneSolver operates as a standalone forensic tool, integrating AI models to assist in crime investigation. The platform will function as a web-based application accessible via browsers and support batch processing for handling large crime scene datasets.

### ****2.2 Project Functions****

* **Crime Classification:** Identifies crime types from images and videos.
* **Evidence Extraction:** Extracts significant crime scene details using ViT.
* **Automated Reporting:** Generates structured crime scene summaries.
* **Batch Processing:** Handles multiple crime scene analyses efficiently.
* **Data Storage:** Stores processed crime scene images and extracted evidence securely.

### ****2.3 User Classes and Characteristics****

* **Forensic Experts:** Utilize AI tools for crime scene analysis.
* **Law Enforcement Officers:** Use extracted data to support investigations.
* **Crime Analysts:** Analyze summarized reports for further case studies.

### ****2.4 Operating Environment****

* **Frontend:** React.js for web-based UI
* **Backend:** Node.js and Express.js for API handling, Flask for AI processing
* **Database:** MongoDB for storing images, evidence, and reports
* **Deployment:** Hosted on AWS or Vercel

### ****2.5 Design and Implementation Constraints****

* Integration with **CLIP and ViT models** for AI-based crime analysis.
* **Web-based access only**, no native mobile app.
* **Batch processing support**, requiring optimized server performance.

### ****2.6 Assumptions and Dependencies****

* The UCF Crime Dataset will be available for model training.
* Users will access the system through modern browsers (Chrome, Firefox, Edge).
* AI models will require periodic fine-tuning for improved accuracy.

## ****3. System Features****

### ****3.1 Feature 1 - Crime Classification****

#### ****3.1.1 Description and Priority****

This feature classifies crime scene images into predefined categories using CLIP. It is a **high-priority** feature as it forms the core functionality of the platform.

#### ****3.1.2 Functional Requirements****

* **FR1.1:** Users can upload crime scene images for classification.
* **FR1.2:** The AI model classifies the crime type based on visual data.
* **FR1.3:** Classified results are stored in the database with metadata.

### ****3.2 Feature 2 - Evidence Extraction****

#### ****3.2.1 Description and Priority****

This feature extracts key visual evidence from images using Vision Transformers (ViT). It is **critical** for assisting forensic investigations.

#### ****3.2.2 Functional Requirements****

* **FR2.1:** Users can select images for evidence extraction.
* **FR2.2:** The system identifies significant visual elements (e.g., weapons, objects, bloodstains).
* **FR2.3:** Extracted evidence is stored and linked to crime scene reports.

### ****3.3 Feature 3 - Crime Scene Summarization****

#### ****3.3.1 Description and Priority****

This feature generates textual summaries of crime scenes based on extracted data. It is **essential** for simplifying forensic reporting.

#### ****3.3.2 Functional Requirements****

* **FR3.1:** Users can generate an automated summary for a crime scene.
* **FR3.2:** The system compiles evidence data into structured reports.
* **FR3.3:** Summarized reports can be exported or shared with law enforcement

**3.4 Feature 4 - Batch Processing**

**3.4.1 Description and Priority**

This feature enables bulk analysis of multiple crime scene images. It is **high-priority** for handling large-scale forensic cases efficiently.

**3.4.2 Functional Requirements**

* **FR4.1:** Users can upload multiple images for simultaneous analysis.
* **FR4.2:** The system processes each image sequentially and generates reports.
* **FR4.3:** Results are stored and can be reviewed later.

**4. External Interface Requirements**

**4.1 User Interfaces**

* **Dashboard:** Displays crime scene images, classifications, and reports.
* **Crime Analysis Panel:** Allows users to upload images and view results.
* **Reports Section:** Enables users to generate, edit, and export crime scene summaries.

**4.2 Hardware Interfaces**

* No specific hardware requirements; works on standard computing devices.

**4.3 Software Interfaces**

* **MongoDB:** Stores processed crime scene data.
* **Flask API:** Handles AI model inference.
* **Express.js:** Manages backend API calls.

**4.4 Communication Interfaces**

* **HTTP/HTTPS:** Secure communication between frontend and backend.
* **REST API:** For accessing AI models and retrieving analysis results.

**5. Non-Functional Requirements**

* **Security:** Uses authentication (JWT) and encryption for data protection.
* **Scalability:** Supports high-volume data processing for forensic agencies.

**6. Other Requirements**

* Compliance with **legal and ethical guidelines** for forensic AI usage.
* Users must be able to **review and edit AI-generated reports** before submission.