PROJECT REPORT

Project Name: Crime Vision: Advanced Crime Classification With Deep Learning

Developers:
Ambika Panse
Diya Ramani
Ridhima Taneja

1. INTRODUCTION

1.1 Project Overview

Our project employs advanced deep learning techniques to transform crime identification through the utilization of neural networks to analyse images and video footage of crime scenes. It focuses on elevating criminal justice procedures. Our methodology entails training these networks on extensive datasets, enabling the meticulous classification of diverse crimes based on visual patterns. This innovative approach proves invaluable in crime scene investigations, forensic analyses, and surveillance endeavours. By harnessing the capabilities of deep learning algorithms, we empower law enforcement to accurately detect crime types, extract insights from substantial datasets, and identify trends within surveillance footage. Ultimately, our project functions as a strategic instrument for crafting dynamic interventions and tailored strategies, representing a technological advancement in crime prevention and law enforcement endeavors.

1.2 Purpose

The purpose of the project is to leverage sophisticated deep learning techniques to enhance the identification and classification of various types of crimes by employing neural networks to analyze images from crime scenes.

The primary goal is to provide means for precision crime detection and provide efficient technological tools to law enforcement to enhance crime detection process.

The secondary goals include expanding to multiple types of data (eg. Audio, video) from various sources, identifying crime hotspots and lookinging into the possibility of detecting a crime before it takes place.

2. LITERATURE SURVEY

2.1 Existing problem

Following are the existing problems to solve which the need to develop an Advanced Crime Classification With Deep Learning arose:

1. Traditional crime classification often relies on manual analysis of evidence, which is time-consuming and delays the investigative processes
2. Human judgement in crime classification can be subjective and may be biased due to social, economic or personal factors.
3. Due to bulk of data generated by the modern crime scenario, it is possible that manual inferences are erroneous.
4. Identifying subtle or complex patterns indicative of specific types of crimes may not always be feasible for the person analysing the data.
5. Manual analysis of crime scene data is a time and labour intensive task.

2.2 References

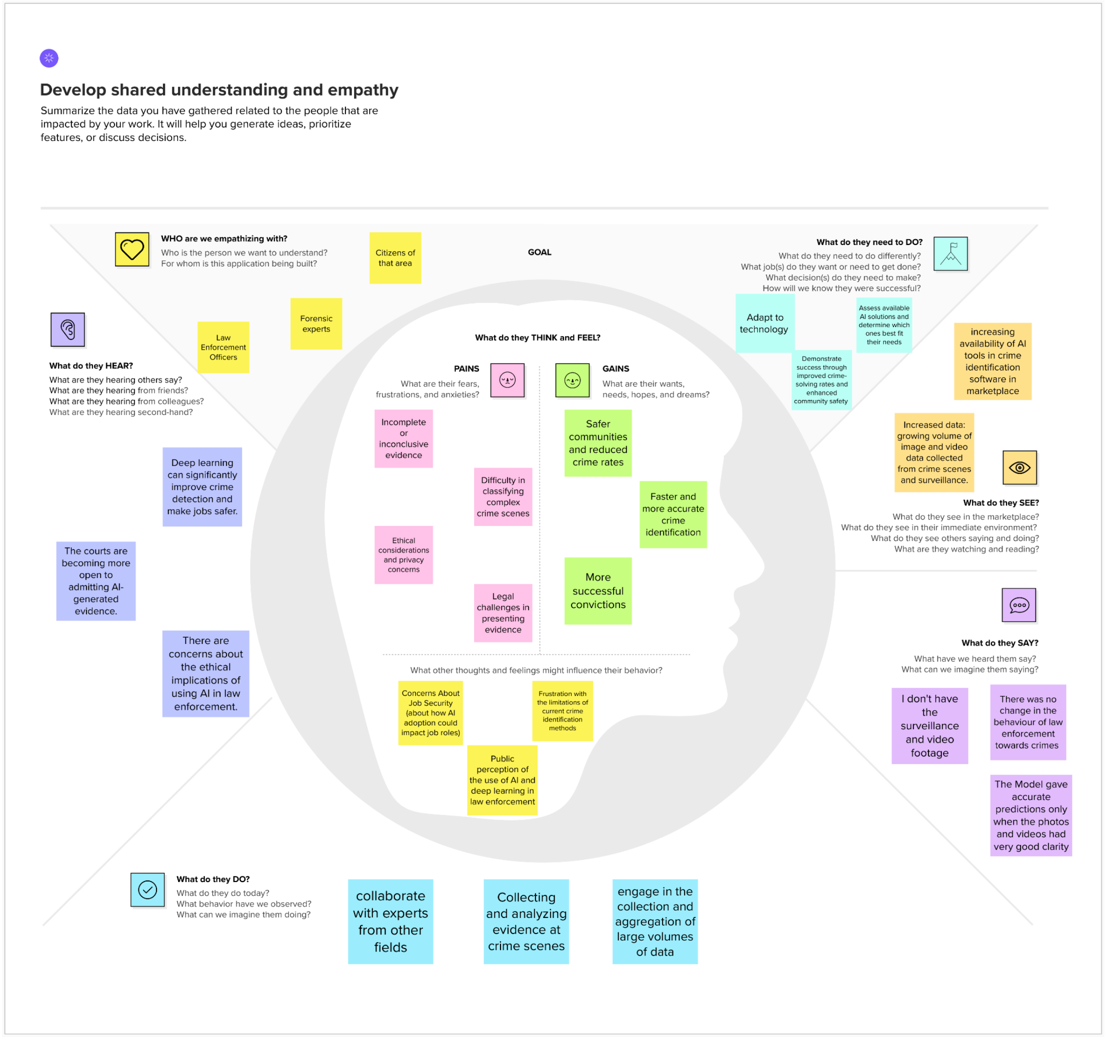
1. Shaikh, Qhubaib & Yadav, Yogesh & Sankhe, Saish & Chaudhari, Reshma. (2023). Survey Paper on Criminal Identification System. 10.6084/m9.figshare.22233061.
2. Shakil, Mohammad. (2022). Barriers in crime scene investigation: A study on the inefficiency & procrastination in solving criminal cases.
3. Brayne, S. (2017). Big Data Surveillance: The Case of Policing. American Sociological Review, 82(5), 977-1008. https://doi.org/10.1177/0003122417725865
4. Bhowmik, Subhranil. (2023). The Evolution of Crime: The Dynamic Definition of Crime as Per Society. International Journal of Law and Management. 6. 3438 - 3489. 10.10000/IJLMH.115284.

2.3 Problem Statement Definition

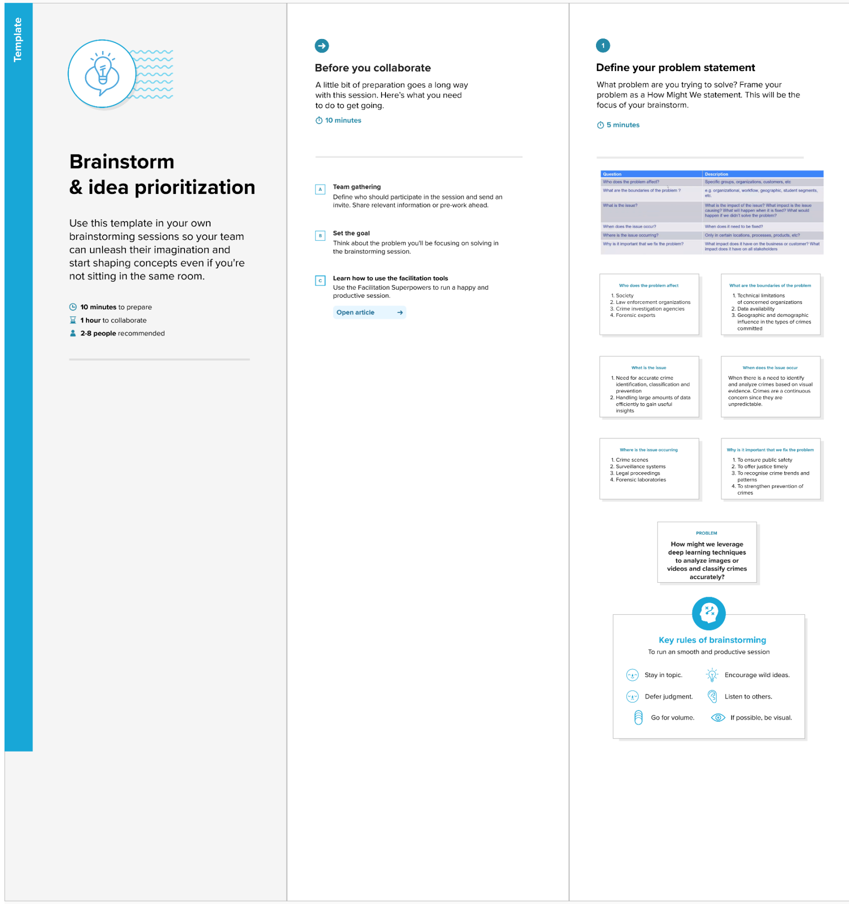
Law enforcement agencies face growing challenges in effectively managing and categorizing the vast array of criminal activities captured within the ever-expanding volume of visual data. The Advanced Crime Classification With Deep Learning initiative seeks to revolutionize crime analysis, enabling precise and rapid detection of complex criminal patterns, thereby enhancing public safety

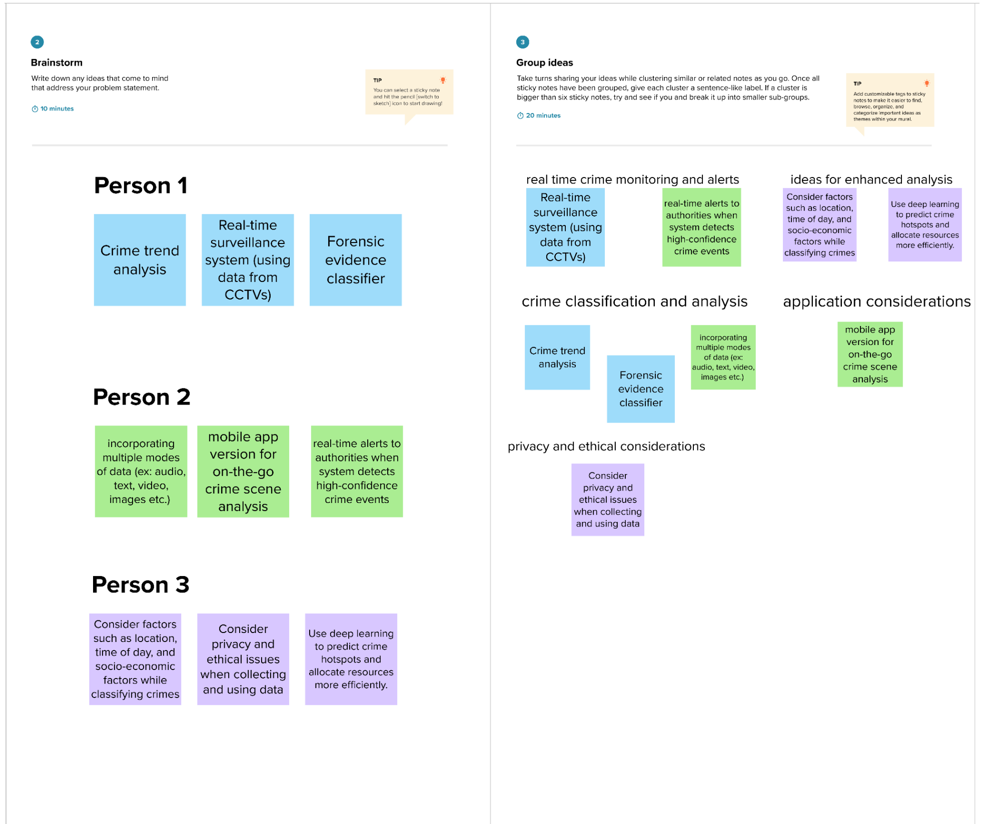
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming





4

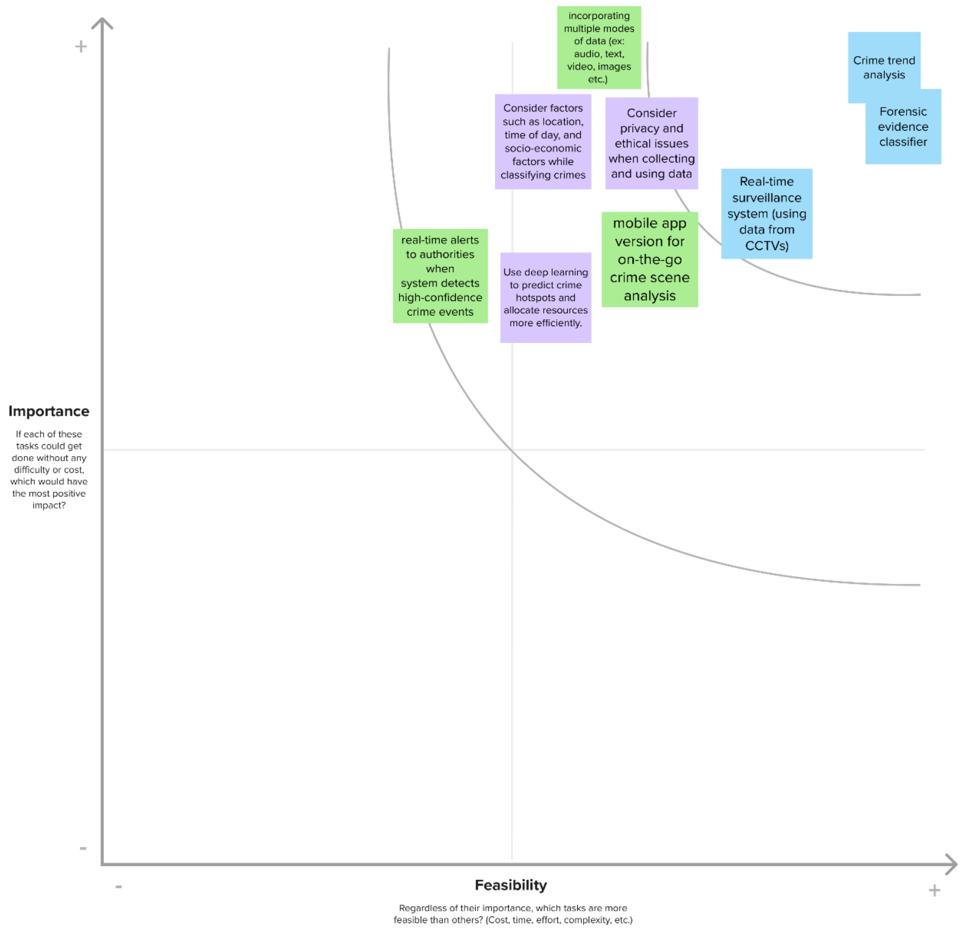
Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.



4. REQUIREMENT ANALYSIS

4.1 Functional Requirement

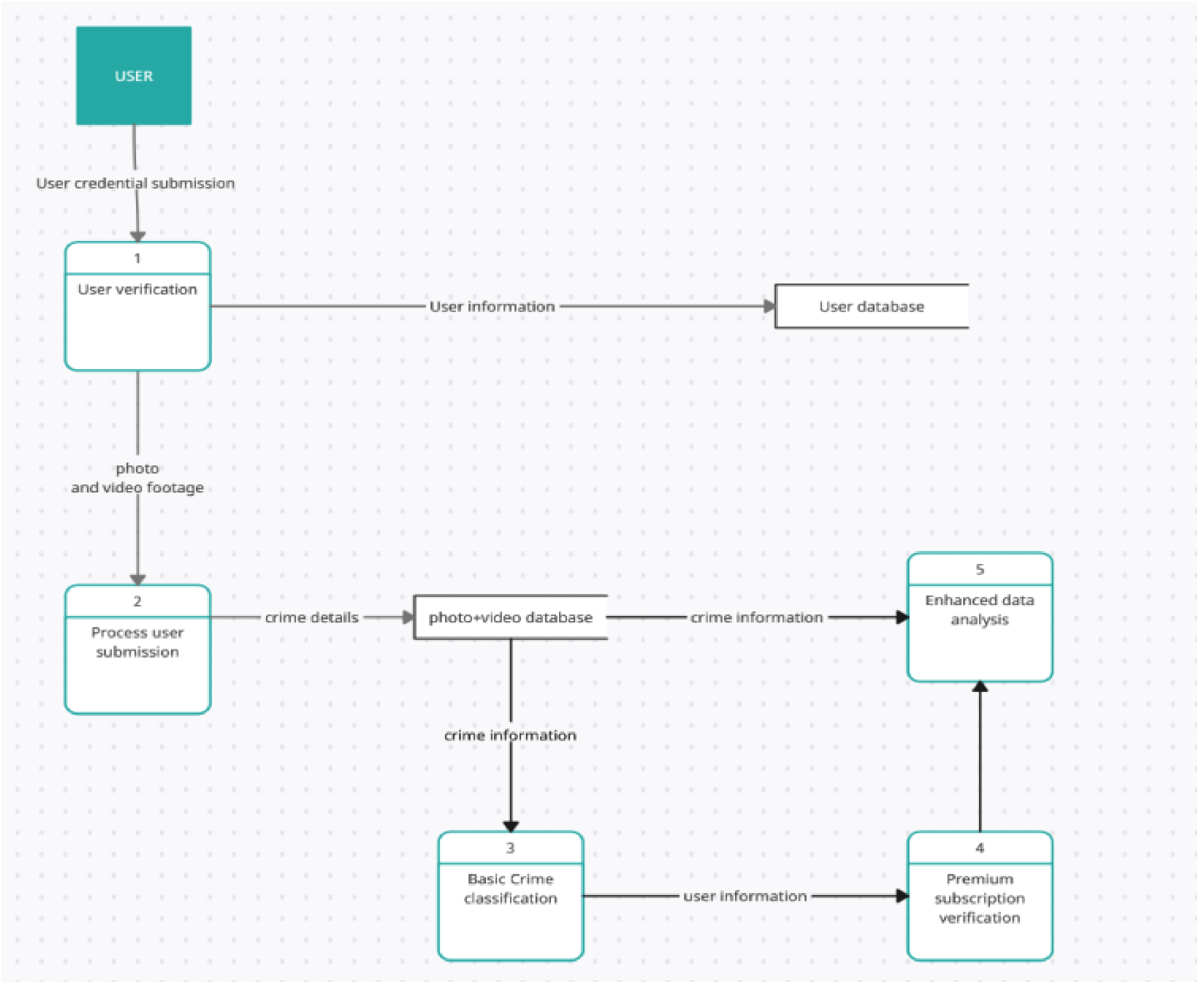
1. Prediction Confidence Scores: Integrating a mechanism that assigns a confidence score to each crime prediction, reflecting the model's level of certainty and implementing a scoring system that accurately conveys the model's confidence in its predictions, enabling informed decision-making by law enforcement personnel.
2. Intuitive User Interface: Develop a user-friendly dashboard specifically designed for law enforcement officials, presenting crime predictions, confidence scores, and relevant image details in a clear and organized manner and prioritize accessibility and ease of use, ensuring that the interface effectively conveys the model's insights to users with varying levels of technical expertise.
3. Scalable System Architecture: Design a scalable system architecture that can efficiently handle large volumes of crime scene images, ensuring adaptability to future data growth which implements optimized algorithms and infrastructure to meet increasing data and processing demands without compromising performance or stability.
4. Comprehensive Documentation: Prepare comprehensive documentation that thoroughly details system functionalities, training procedures, and deployment instructions, providing a valuable resource for users and developers alike. It may also include user manuals, developer guides, and architecture documentation to promote effective system utilization, maintenance, and future enhancements.

4.2 Non-Functional Requirements

1. Performance: Achieve real-time performance by optimizing algorithms and infrastructure for efficient processing.
2. Reliability: Deliver accurate predictions under diverse conditions by deploying robust error handling mechanisms and conducting thorough testing to validate model reliability.
3. Availability: Maintain consistent availability by designing failover mechanisms and redundancy for continuous system operation.
4. Security: Safeguard against unauthorized access and misuse by implementing secure authentication protocols and encryption for data transmission.
5. Accuracy and Precision: Ensure high accuracy and precision in crime predictions by regularly assessing and fine-tuning the model, using metrics like precision, recall, and F1 score for improvement.

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories

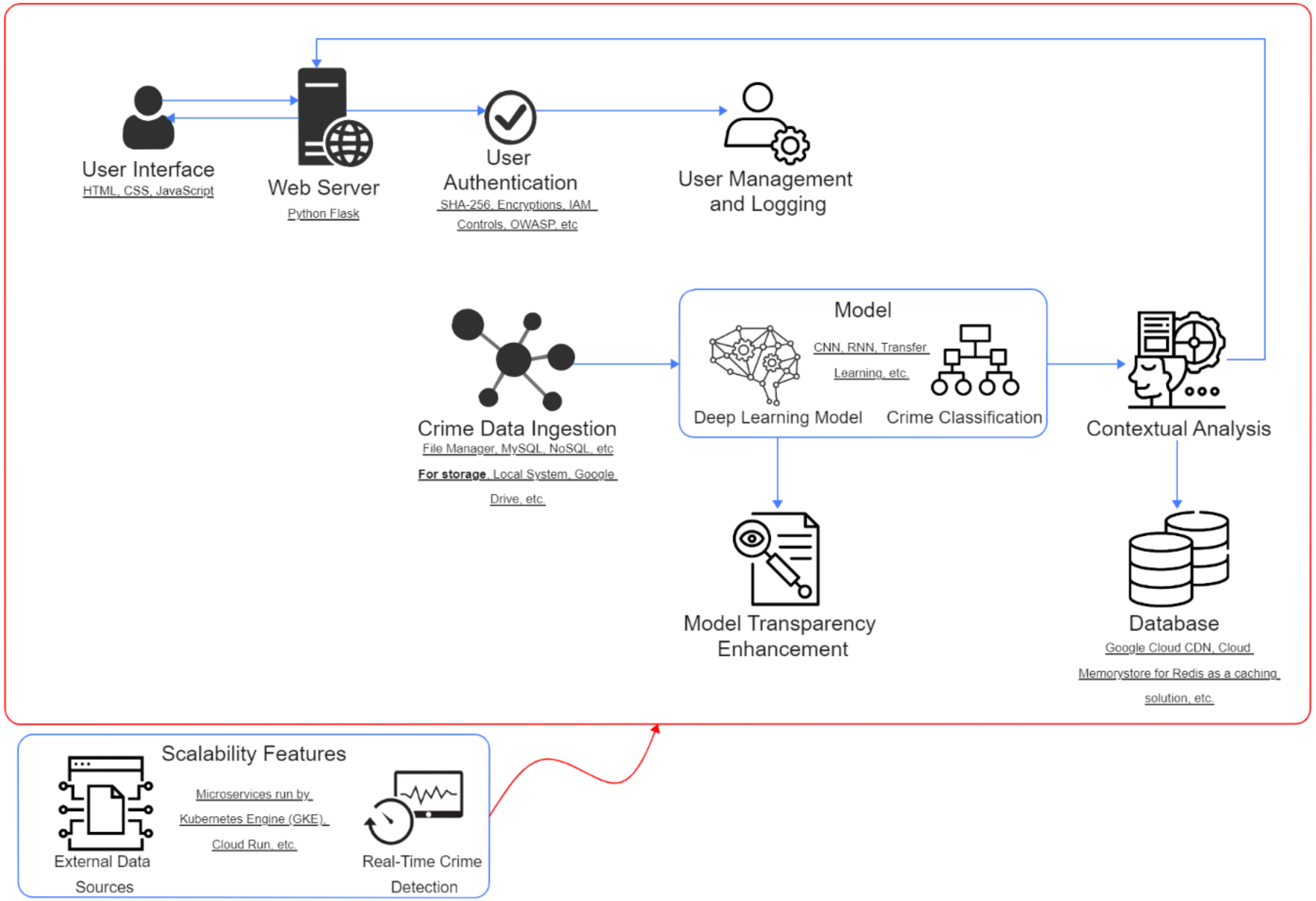


User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile/Web user)	Registration	USN-1	As a user, I can register for the application by entering my email and confirming my password.	I can access my account/dashboard	High	Sprint-1
		USN-2	As a user, I will receive a confirmation email once I have registered for the application	I can receive a confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Gmail Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering my email & password	I can log in to my profile by using my credentials	High	Sprint-1
	Dashboard	USN-6	As a user, I can select the 'Submit Crime Evidence' button to submit photo and video footage from my device (mobile/desktop/laptop)	I can submit photo/video input data through any device	High	Sprint-1
	Dashboard	USN-7	As a user, I can choose 'Basic Classification' to	I can get basic	High	Sprint-1

			receive a preliminary classification.	classification of the uploaded evidence		
	Dashboard	USN-8	As a user, I can choose 'Premium Analysis' to subscribe and access detailed crime analysis.	I can subscribe and receive detailed crime analysis	High	Sprint-1
	Dashboard	USN-9	As a user, I can navigate to ‘History’ so as to review previous classifications and analysis	I can view all my previous uploads and their respective analysis/classification	Low	Sprint-2
Customer Care Executive	User assistance	USN-10	As a customer care executive, I can assist users in the process of submitting crime evidence.	I can guide the user through the submission process.	High	Sprint-2
	User assistance	USN-11	As a customer care executive, I can address user issues related to basic or premium crime classification.	I can investigate and resolve user complaints promptly.	High	Sprint-3
Administrator	Platform management	USN-12	As an administrator, I can monitor user activity and interactions with the platform.	I can access a dashboard displaying user activity	High	Sprint-2
	Platform management	USN-13	As an administrator, I can manage premium subscriptions, including user upgrades, renewals, and cancellations.	I can access a subscription management interface.	Medium	Sprint-3

5.2 Solution Architecture



6. PROJECT PLANNING AND SCHEDULING

6.1 Technical Architecture

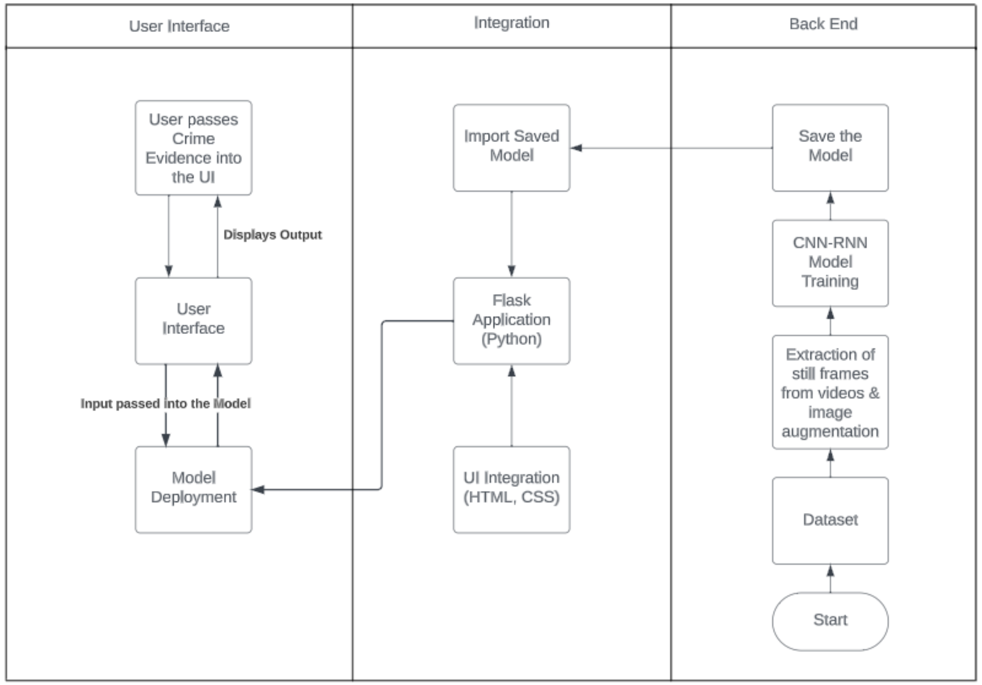


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How the user interacts with the application e.g. Web UI	HTML, CSS, JavaScript
2.	Application Logic-1	Logic for a process in the application	Python

3.	Database	Collect the Dataset Based on the Problem Statement	File Manager, MySQL, NoSQL, etc.
4.	File Storage/ Data	File storage requirements for Storing the dataset	Local System, Google Drive, etc.
5.	Frame Work	Used to Create a web Application, Integrating Frontend and Back End	Python Flask
6.	Deep Learning Model	Purpose of Model	CNN, RNN, Transfer Learning, etc.
7.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Kubernetes, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Python's Flask
2.	Security Implementations	List all the security/access controls implemented, use of firewalls, etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASP, etc.
3.	Scalable Architecture	Justify the scalability of architecture (3-tier, Micro-services)	Microservices run by Kubernetes Engine (GKE), Cloud Run, etc.
4.	Availability	Justify the availability of the application (e.g. use of load balancers, distributed servers, etc.)	Google Cloud Load Balancing, Google Cloud Spanner, etc.
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDNs), etc.	Google Cloud CDN, Cloud Memorystore for Redis as a caching solution, etc.

6.2 Sprint Planning and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Diya
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Ridhima
Sprint-2	Registration	USN-3	As a user, I can register for the application through Facebook	2	Low	Ambika, Ridhima
Sprint-1	Registration	USN-4	As a user, I can register for the application through Gmail	2	Medium	Ambika
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Diya, Ridhima
Sprint-1	Dashboard	USN-6	As a user, I can select the ‘Submit Crime Evidence’ button to submit photo and video footage from my device (mobile/desktop/laptop)	8	High	Ridhima
Sprint-1	Dashboard	USN-7	As a user, I can choose 'Basic Classification' to receive a preliminary classification.	5	High	Ambika, Diya
Sprint-1	Dashboard	USN-8	As a user, I can choose 'Premium Analysis' to subscribe and access detailed crime analysis.	8	High	Ridhima
Sprint-2	Dashboard	USN-9	As a user, I can navigate to ‘History’ so as to review previous classifications and analysis	3	Low	Ambika

Sprint-2	User assistance	USN-10	As a customer care executive, I can assist users in the process of submitting crime evidence.	3	High	Ambika, Ridhima
Sprint-3	User assistance	USN-11	As a customer care executive, I can address user issues related to basic or premium crime classification.	3	High	Diya
Sprint-2	Platform management	USN-12	As an administrator, I can monitor user activity and interactions with the platform.	5	High	Diya, Ridhima
Sprint-3	Platform management	USN-13	As an administrator, I can manage premium subscriptions, including user upgrades, renewals, and cancellations.	5	Medium	Ambika, Ridhima

6.3 Sprint Delivery Schedule

Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	27	10 Days	31 Oct 2023	9 Nov 2023	27	9 Nov 2023
Sprint-2	13	6 Days	10 Nov 2023	15 Nov 2023	12	15 Nov 2023
Sprint-3	8	4 Days	16 Nov 2023	19 Nov 2023	9	19 Nov 2023

Average Velocity = 27 + 12 + 9 / 3 = 16

Estimate Effort

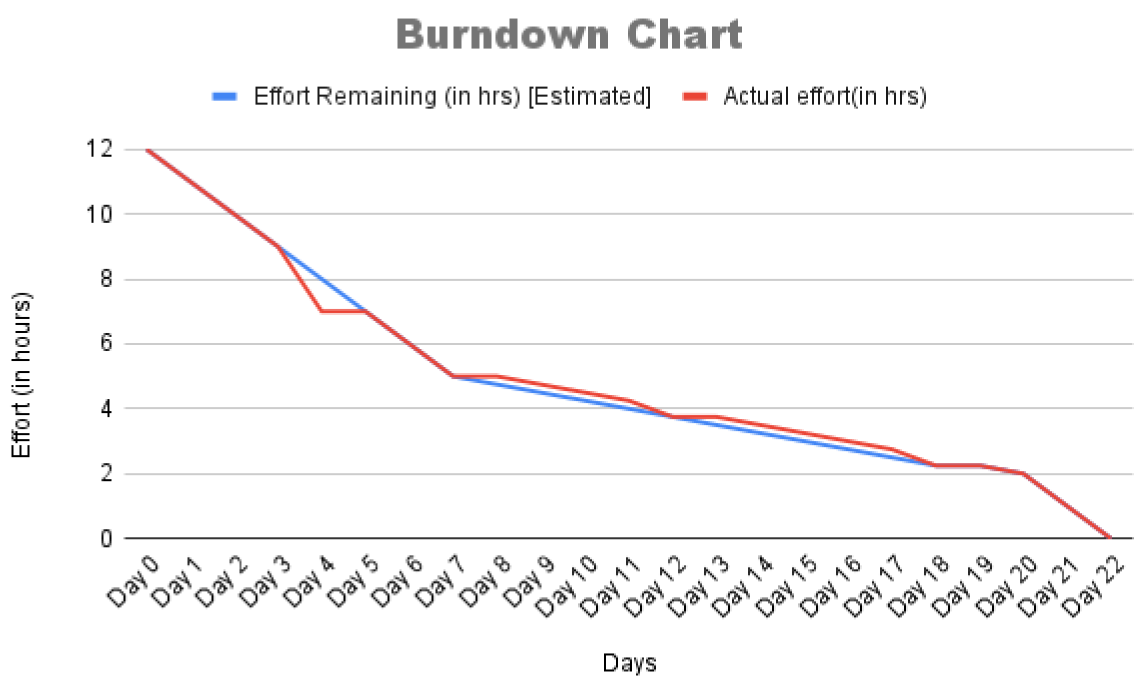
	Day 0	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12	Day 13	Day 14	Day 15	Day 16	Day 17	Day 18	Day 19	Day 20	Day 21	Day 22
Effort Remaining (in hrs)	12	11	10	9	8	7	6	5	4.75	4.5	4.25	4	3.75	3.5	3.25	3	2.75	2.5	2.25	2.25	2	1	0

Daily Progress Tracking

Daily Progress	Task	Completion	Time Required
	Empathy Map Canvas	Day 1	1
	Brainstrom and Prioritize Ideas	Day 2	1
	Proposed Solution	Day 3	1
	Solution Architecture	Day 4	1
	Data Flow Diagram	Day 5	1
	Technology Stack	Day 6	1
	Project Planning Details	Day 7	1
	Project Development Phase	Day 8-Day 20	3
	Solution Performance	Day 21	1
	Project Documentation	Day 22	1

Actual Effort

Days	Day 0	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12	Day 13	Day 14	Day 15	Day 16	Day 17	Day 18	Day 19	Day 20	Day 21	Day 22
Actual effort	12	11	10	9	7	7	6	5	5	4.75	4.5	4.25	3.75	3.75	3.5	3.25	3	2.75	2.25	2.25	2	1	0
Effort Remaining (in hrs) [Estimated]	12	11	10	9	8	7	6	5	4.75	4.5	4.25	4	3.75	3.5	3.25	3	2.75	2.5	2.25	2.25	2	1	0


Burndown Chart
Effort Remaining (in hrs) [Estimated]
Actual effort(in hrs)
Effort (in hours)
12
10
8
6
4
2
0
Day 0
Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7
Day 8
Day 9
Day 10
Day 11
Day 12
Day 13
Day 14
Day 15
Day 16
Day 17
Day 18
Day 19
Day 20
Day 21
Day 22
Days

7. CODING AND SOLUTIONING

7.1 Feature 1-Deep Learning-powered Image Analysis:

The project employs advanced deep learning algorithms to analyze image data from past crime scenes. These algorithms are trained to identify intricate patterns and features, enabling the model to predict the type of crime taking place solely based on visual evidence. This capability holds immense potential for law enforcement, as it can provide valuable insights into criminal activities without the need for physical evidence or eyewitness accounts.

7.2 Feature 2-Crime Type Prediction:

The core functionality of the project lies in its ability to predict the type of crime occurring in a given scenario. By leveraging historical data and the learned patterns from deep learning, the model can classify and identify the specific nature of criminal activities depicted in images. This feature can be instrumental for law enforcement agencies, as it can provide them with a preliminary assessment of a crime scene before they arrive, allowing them to allocate resources more effectively and make informed decisions about their response strategy.

8. PERFORMANCE TESTING

8.1 Performance Metrics

<table>
<tr><th>S.No.</th><th>Parameter</th><th>Values</th><th>Screenshot</th></tr>
<tr><td>1.</td><td>Model Summary</td><td>Total params: 4243278
Trainable params: 1983758
Non-trainable params: 2259520</td><td><pre>
 Layer (type)   Output Shape   Param #
===
 mobilenetv2_1.00_224 (Func (None, 2, 2, 1280)   2257984
 tional)

 module_wrapper_21 (ModuleW (None, 5120)   0
 rapper)

 module_wrapper_22 (ModuleW (None, 256)   1310976
 rapper)

 batch_normalization_6 (Bat (None, 256)   1024
 chNormalization)

 module_wrapper_23 (ModuleW (None, 256)   0
 rapper)

 module_wrapper_24 (ModuleW (None, 512)   131584
 rapper)

 batch_normalization_7 (Bat (None, 512)   2048
 chNormalization)

 module_wrapper_25 (ModuleW (None, 512)   0
 rapper)

 module_wrapper_26 (ModuleW (None, 1024)   525312
 rapper)

 module_wrapper_27 (ModuleW (None, 14)   14350
 rapper)

===
Total params: 4243278 (16.19 MB)
Trainable params: 1983758 (7.57 MB)
Non-trainable params: 2259520 (8.62 MB)
</pre></td></tr>
</table>

<table>
<tr><td>2.</td><td>Accuracy</td><td>Training Accuracy - 75.25

Validation Accuracy - 57.86</td><td>Epoch 1/10
100/100 [==============================] - ETA: 0s - loss: 1.4299 - accuracy: 0.6900
Epoch 1: val_loss improved from inf to 3.14930, saving model to crime.h5
100/100 [==============================] - 731s 7s/step - loss: 1.4299 - accuracy: 0.6900 - val_loss: 3.1493 - val_accuracy: 0.4037
Epoch 2/10
100/100 [==============================] - ETA: 0s - loss: 1.2662 - accuracy: 0.7228
Epoch 2: val_loss improved from 3.14930 to 2.31910, saving model to crime.h5
100/100 [==============================] - 771s 8s/step - loss: 1.2662 - accuracy: 0.7228 - val_loss: 2.3191 - val_accuracy: 0.4849
Epoch 3/10
100/100 [==============================] - ETA: 0s - loss: 1.2276 - accuracy: 0.7178
Epoch 3: val_loss improved from 2.31910 to 2.28730, saving model to crime.h5
100/100 [==============================] - 710s 7s/step - loss: 1.2276 - accuracy: 0.7178 - val_loss: 2.2873 - val_accuracy: 0.5766
Epoch 4/10
100/100 [==============================] - ETA: 0s - loss: 1.1405 - accuracy: 0.7412
Epoch 4: val_loss improved from 2.28730 to 2.02291, saving model to crime.h5
100/100 [==============================] - 711s 7s/step - loss: 1.1405 - accuracy: 0.7412 - val_loss: 2.0229 - val_accuracy: 0.5782
Epoch 5/10
100/100 [==============================] - ETA: 0s - loss: 1.0682 - accuracy: 0.7553
Epoch 5: val_loss did not improve from 2.02291
100/100 [==============================] - 771s 8s/step - loss: 1.0682 - accuracy: 0.7553 - val_loss: 2.2640 - val_accuracy: 0.5779
Epoch 6/10
100/100 [==============================] - ETA: 0s - loss: 1.1062 - accuracy: 0.7444
Epoch 6: val_loss did not improve from 2.02291
100/100 [==============================] - 687s 7s/step - loss: 1.1062 - accuracy: 0.7444 - val_loss: 2.2243 - val_accuracy: 0.5700
Epoch 7/10
100/100 [==============================] - ETA: 0s - loss: 1.0371 - accuracy: 0.7566
Epoch 7: val_loss improved from 2.02291 to 1.97179, saving model to crime.h5
100/100 [==============================] - 710s 7s/step - loss: 1.0371 - accuracy: 0.7566 - val_loss: 1.9718 - val_accuracy: 0.5319
Epoch 8/10
100/100 [==============================] - ETA: 0s - loss: 1.0441 - accuracy: 0.7541
Epoch 8: val_loss improved from 1.97179 to 1.94027, saving model to crime.h5
100/100 [==============================] - 683s 7s/step - loss: 1.0441 - accuracy: 0.7541 - val_loss: 1.9403 - val_accuracy: 0.5652
Epoch 9/10
100/100 [==============================] - ETA: 0s - loss: 0.9986 - accuracy: 0.7663
Epoch 9: val_loss did not improve from 1.94027
100/100 [==============================] - 710s 7s/step - loss: 0.9986 - accuracy: 0.7663 - val_loss: 2.1234 - val_accuracy: 0.5587
Epoch 10/10
100/100 [==============================] - ETA: 0s - loss: 1.0380 - accuracy: 0.7525
Epoch 10: val_loss did not improve from 1.94027
100/100 [==============================] - 709s 7s/step - loss: 1.0380 - accuracy: 0.7525 - val_loss: 1.9445 - val_accuracy: 0.5786</td></tr>
<tr><td>3.</td><td>Confidence Score (Only Yolo Projects)</td><td>Class Detected - NA

Confidence Score - NA</td><td>Not Applicable</td></tr>
</table>

Performance Metric	Score
Accuracy	54.59%
Precision	35.09%
Recall	54.59%
F1-Score	42.07%

Classification Report:

```
              precision    recall  f1-score   support

           0       0.03      0.00      0.01       297
           1       0.00      0.00      0.00      3365
           2       0.03      0.02      0.02      2793
           3       0.00      0.00      0.00      2657
           4       0.00      0.00      0.00      7657
           5       0.06      0.00      0.01      6510
           6       0.00      0.00      0.00      1231
           7       0.58      0.93      0.72     64952
           8       0.00      0.00      0.00      2663
           9       0.01      0.00      0.00       835
          10       0.00      0.00      0.00      7630
          11       0.08      0.00      0.00      7623
          12       0.01      0.04      0.02      1984
          13       0.00      0.00      0.00      1111

    accuracy                           0.55    111308
   macro avg       0.06      0.07      0.06    111308
weighted avg       0.35      0.55      0.42    111308
```

Confusion Matrix:

```
Confusion Matrix:
[[    1     0     6     0     0     1     0   278     0     0     0     0
     11     0]
 [    0     0    59     0     0    12     1  3148     0     6     0     2
    137     0]
 [    0     1    47     0     0     7     3  2582     0     4     0     4
    145     0]
 [    1     0    57     0     0    15     1  2449     0     4     0     3
    127     0]
 [    2     1   121     0     0    15     4  7142     1    10     0     8
    353     0]
 [    2     1   107     0     0    19     4  6071     0     7     0     8
    291     0]
 [    0     0    19     0     0     5     0  1144     0     1     0     2
     60     0]
 [   18     8  1023     0     2   188    20 60611     1   107     0    89
   2884     1]
 [    3     1    38     0     0    11     0  2470     0     4     0     4
    132     0]
 [    1     0    11     0     0     3     0   781     0     2     0     1
     36     0]
 [    0     1   143     0     0    25     5  7080     1    13     0     9
    353     0]
 [    2     1   116     0     0    24     7  7107     0    15     0    12
    339     0]
 [    0     0    30     0     0     5     1  1870     0     3     0     2
     73     0]
 [    1     0    15     0     0     3     0  1028     0     0     0     2
     62     0]]
```

9. RESULTS

9.1 Output Screenshots

-Data Preprocessing

```
train_set = train_datagen.flow_from_directory(train_path,
                                              target_size = (64, 64),
                                              batch_size = 32,
                                              class_mode = 'categorical')

Found 1266345 images belonging to 14 classes.
```

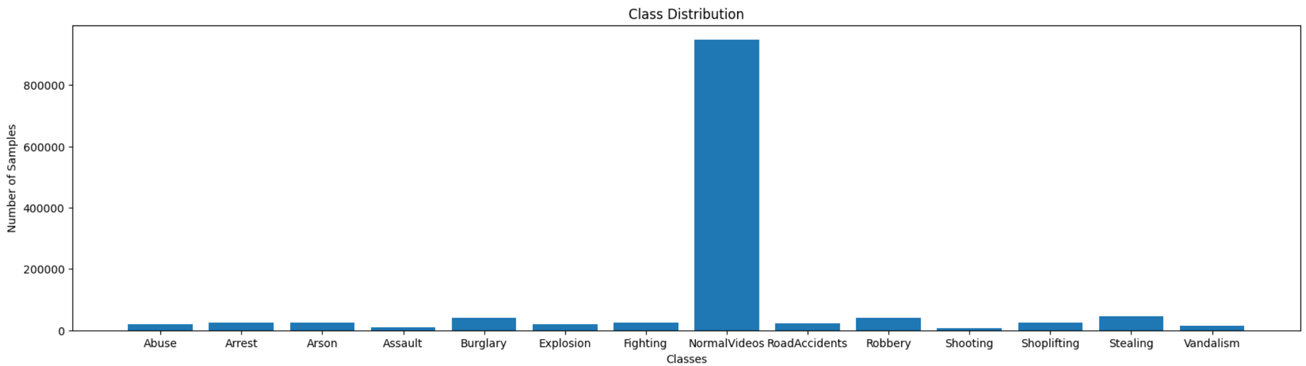
```
[ ] test_set = test_datagen.flow_from_directory(test_path,
                                                target_size = (64, 64),
                                                batch_size = 32,
                                                class_mode = 'categorical')

Found 111308 images belonging to 14 classes.
```

```
[ ] class_names = list(train_set.class_indices.keys())
    class_names = np.array(class_names)
    class_names

array(['Abuse', 'Arrest', 'Arson', 'Assault', 'Burglary', 'Explosion',
       'Fighting', 'NormalVideos', 'RoadAccidents', 'Robbery', 'Shooting',
       'Shoplifting', 'Stealing', 'Vandalism'], dtype='&lt;U13')
```

-Data Visualisation



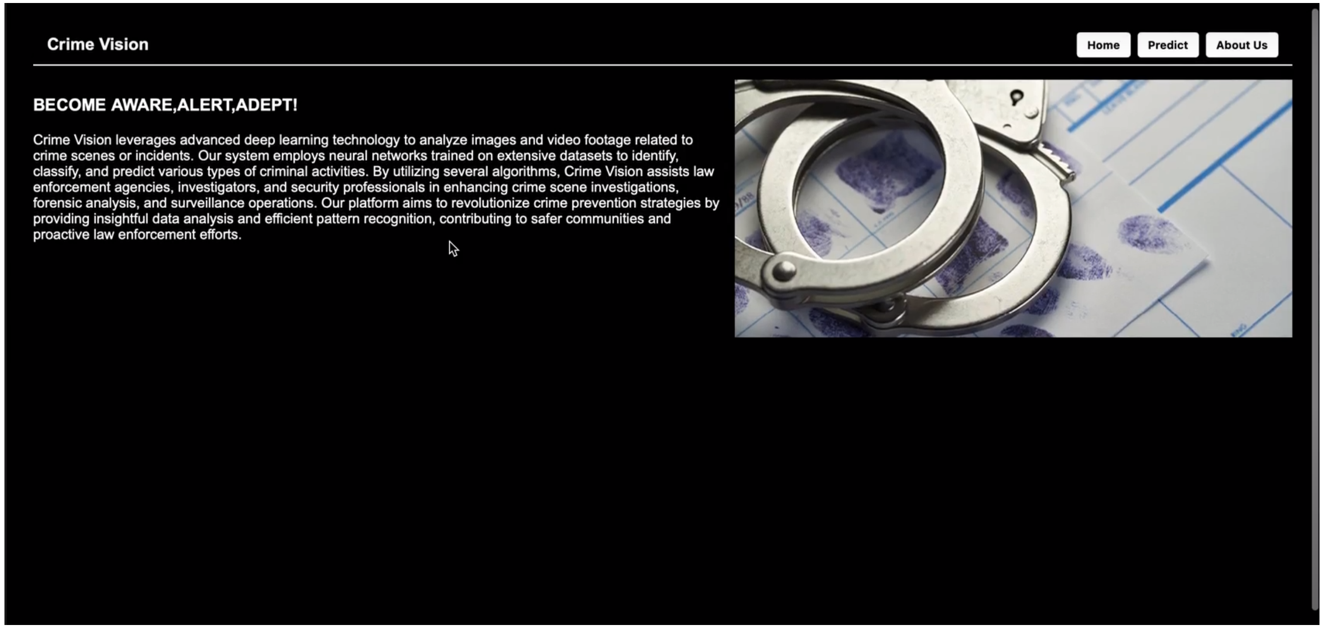
-Model Testing

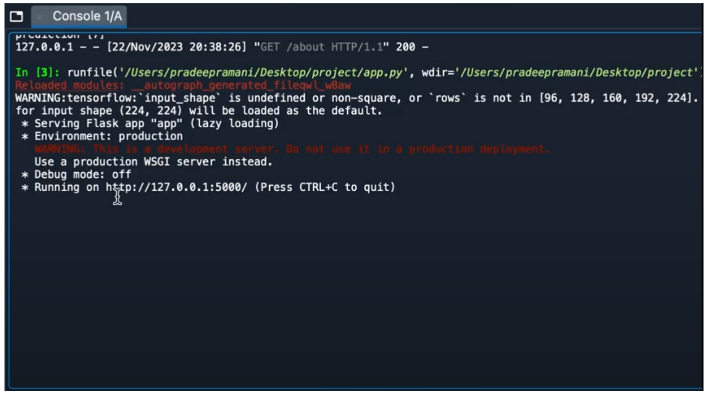
```
#Running the model on a single image
from keras.utils import load_img
img = image.load_img('/content/Test/NormalVideos/Normal_Videos_003_x264_2200.png',target_size=(64,64))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
prediction = np.argmax(model_predict.predict(x))
class_names[prediction]
```

```
1/1 [==============================] - 1s 1s/step
'NormalVideos'
```

```
#Running the model on Test_set
predictions = model_predict.predict(test_set)
predicted_labels = np.argmax(predictions, axis=1)
```

```
3479/3479 [==============================] - 699s 201ms/step
```


Crime Vision
Home
Predict
About Us
BECOME AWARE,ALERT,ADEPT!
Crime Vision leverages advanced deep learning technology to analyze images and video footage related to crime scenes or incidents. Our system employs neural networks trained on extensive datasets to identify, classify, and predict various types of criminal activities. By utilizing several algorithms, Crime Vision assists law enforcement agencies, investigators, and security professionals in enhancing crime scene investigations, forensic analysis, and surveillance operations. Our platform aims to revolutionize crime prevention strategies by providing insightful data analysis and efficient pattern recognition, contributing to safer communities and proactive law enforcement efforts.


Console 1/A
127.0.0.1 - - [22/Nov/2023 20:38:26] "GET /about HTTP/1.1" 200 -
In [3]: runfile('/Users/pradeepramani/Desktop/project/app.py', wdir='/Users/pradeepramani/Desktop/project'
Reloaded modules: __autograph_generated_fileqwl_w8aw
WARNING:tensorflow:`input_shape` is undefined or non-square, or `rows` is not in [96, 128, 160, 192, 224].
for input shape (224, 224) will be loaded as the default.
* Serving Flask app "app" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)


Crime Vision
Home
Predict
About Us
Upload Image Here For Classification
Choose...
Result: The classified crime is:
NormalVideos

10. ADVANTAGES AND DISADVANTAGES

Advantages:

1. Deep learning algorithms excel at identifying crimes accurately, reducing the likelihood of errors and enhancing the reliability of law enforcement operations.
2. By minimizing human bias, deep learning ensures unbiased crime assessments, fostering fairness and upholding justice in law enforcement practices.
3. Deep learning models continually learn from new image data, ensuring adaptability to changing crime patterns. This dynamic approach identifies emerging trends and variations in criminal behavior effectively.
4. In situations with sparse textual information, depending solely on images is advantageous. Deep learning extracts valuable insights, reducing reliance on text.
5. Emphasizing image data sets the stage for automating crime classification. As the model gains precision, it holds promise for automating elements of crime analysis, allowing resource allocation to strategic tasks.

Disadvantages:

1. Relying solely on image data, deep learning models lack context and struggle to grasp situational nuances, potentially resulting in inaccurate crime predictions. Additionally, capturing static scenes hinders understanding dynamic criminal activities.
2. The accuracy of the model depends on high-quality, representative training data. Inadequate diversity may introduce biases, hindering generalization to real-world variations.
3. Complex deep learning models frequently lack transparency, making it challenging to understand and justify their decision-making processes, impacting the trustworthiness of crime classification systems.
4. Adapting to new or evolving crime patterns poses a challenge for image-based models, potentially reducing their effectiveness in predicting novel crimes.

11. CONCLUSION

The Advanced Crime Classification With Deep Learning project represents a significant advancement in applying technology to crime analysis. By utilizing image data to predict past crimes, the project showcases the potential of deep learning in interpreting visual information. The model's ability to classify crime types based on collected data highlights its contribution to law enforcement.

However, inherent limitations exist, primarily the model's static representation of dynamics and its difficulty in adapting to emerging crime patterns. The reliance on only one type of and limited data may pose challenges in identifying evolving criminal tactics, potentially reducing the model's effectiveness in predicting novel crime types. Additionally, concerns regarding limited context awareness and the need for diverse training data emphasize areas for improvement.

Moving forward, a comprehensive strategy that incorporates various data forms, addresses ethical considerations, and enhances adaptability will be critical. While the project marks progress, it is important to understand that as crime evolves in nature and execution, innovation in detecting and solving it is imperative. A holistic approach, considering the dynamic nature of crimes and incorporating advancements in technology, will be essential for the development of crime classification systems that better serve the complexities of the real-world.

12. FUTURE SCOPE

The project is based on using a deep learning model to classify crimes from images of a crime scene. Understandably, the model can be modified to serve various purposes based on how and where the model is being used. In the near future, keeping the basic frame work of the model and its program the same, additional features can be added and the pre-existing ones updated. Following are some goals which can be achieved through an enhanced version of the model built in this project.

1. Multimodal Integration: Expanding the model to include additional data modalities, such as audio and text, will provide a more comprehensive understanding of crime scenarios by enabling the analysis of a wider range of information. This will allow for a more accurate and nuanced assessment of criminal activities.
2. Real-time Analysis: Enabling the model to operate in real-time will allow for immediate crime detection and response. This will enable law enforcement to take proactive measures to prevent crimes from occurring or to apprehend suspects in the act.
3. Contextual Awareness: Integrating contextual information into the model will improve its understanding of crime situations. This will allow the model to make more nuanced interpretations of dynamic crime situations, taking into account factors such as time, location, and surrounding environment.

13. APPENDIX

13.1 Source Code

```
!pip install -q kaggle

#upload kaggle.json to access Kaggle Datasets

from google.colab import files

uploaded = files.upload()

! mkdir ~/.kaggle

! cp kaggle.json ~/.kaggle/

! chmod 600 ~/.kaggle/kaggle.json

#download ucf-crime-dataset

!kaggle datasets download -d odins0n/ucf-crime-dataset

!unzip ucf-crime-dataset.zip

#import necessary libraries

import matplotlib.pyplot as plt

import numpy as np

import tensorflow as tf

import os

import random
```

```
from random import sample

from tensorflow import keras

from tensorflow.keras import layers

from tensorflow.python.keras.layers import Dense, AveragePooling2D, Dropout,MaxPooling2D, Conv2D, Flatten

from tensorflow.keras.models import Sequential

from tensorflow.keras.optimizers import Adam

from keras.preprocessing.image import ImageDataGenerator

from keras.applications.resnet import preprocess_input

from keras.applications import ResNet50

from keras.callbacks import ModelCheckpoint

from keras.callbacks import EarlyStopping

from keras.preprocessing import image

from tensorflow.keras.utils import load_img

from collections import Counter

from keras.layers import BatchNormalization

from tensorflow.keras.optimizers import SGD

train_path = '/content/Train'

test_path = '/content/Test'
```

```
train_datagen = ImageDataGenerator(
    preprocessing_function=preprocess_input,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest')


test_datagen = ImageDataGenerator(
    preprocessing_function=preprocess_input,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
```

```
    horizontal_flip=True,
    fill_mode='nearest')


train_set = train_datagen.flow_from_directory(train_path,
                                                 target_size = (64, 64),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
test_set = test_datagen.flow_from_directory(test_path,
                                                 target_size = (64, 64),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
class_names = list(train_set.class_indices.keys())
class_names = np.array(class_names)
class_names
#Visualisation
dataset_path = '/content/Train/'
num_samples=14
```

```
class_folders = [folder for folder in os.listdir(dataset_path) if os.path.isdir(os.path.join(dataset_path, folder))]

class_counts = Counter()

for classes in class_names:

    class_path = os.path.join(dataset_path, classes)

    class_counts[classes] = len(os.listdir(class_path))

classes, counts = zip(*class_counts.items())

dcount = dict()

i=0

for crimes in classes:

  dcount[crimes] = counts[i]

  i=i+1



print(dcount)

plt.figure(figsize=(20,5))  # Set the figsize here

plt.bar(classes, counts)

plt.xlabel('Classes')

plt.ylabel('Number of Samples')
```

```
plt.title('Class Distribution')
plt.show()


from keras.applications.mobilenet_v2 import MobileNetV2



mobilenet_v2 = MobileNetV2(
    input_shape=(64, 64, 3),
    alpha=1.0,
    include_top=False,
    weights='imagenet',
    input_tensor=None,
    pooling=None,
    classes=14,
    classifier_activation='softmax'
)
```

```
for layer in mobilenet_v2.layers:
    layer.trainable = False
optimizer = SGD(learning_rate=0.001)
model = Sequential()
model.add(mobilenet_v2)
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.2))
model.add(Dense(1024, activation='relu'))
model.add(Dense(len(class_names), activation='softmax'))


# Display model summary
model.summary()
```

```
# Compile the model
optimizer = Adam(learning_rate=0.001)
model.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['accuracy'])
#Checkpoint used to save the best performances only
checkpoint = ModelCheckpoint(
    filepath='crime.h5',
    monitor="val_loss",
    mode="min",
    save_best_only=True,
    save_weights_only=True,
    verbose=1)


from keras.callbacks import LearningRateScheduler


def scheduler(epoch, lr):
    if epoch < 10:
```

```
        return lr

    else:

        return lr * tf.math.exp(-0.1)



lr_schedule = LearningRateScheduler(scheduler)

#training the model (taking a lotttt of time, but it works)

callbacks = [checkpoint]

epochs=10

   #Manually setting the priority of 'NormalVideos' (majority) class

#to balance the dataset

class_weight = {0: 1,

                1: 1,

                2: 1,

                3: 1,

                4: 1,

                5: 1,

                6: 1,
```

```
                7: 0.7,
                8: 1,
                9: 1,
                10: 1,
                11: 1,
                12: 1,
                13: 1,}

model_history = model.fit(
  train_set,
  validation_data=test_set,
  epochs=epochs,
  steps_per_epoch = 100,
  callbacks=callbacks,
  # class_weight = class_weight
)
from keras.models import load_model
```

```
mobilenet_v2 = MobileNetV2(
    input_shape=(64, 64, 3),
    alpha=1.0,
    include_top=False,
    weights='imagenet',
    input_tensor=None,
    pooling=None,
    classes=14,
    classifier_activation='softmax'
)
for layer in mobilenet_v2.layers:
    layer.trainable = False
optimizer = Adam(learning_rate=0.001)


model_predict = Sequential()
model_predict.add(mobilenet_v2)
model_predict.add(Flatten())
```

```
model_predict.add(Dense(256, activation='relu'))

model_predict.add(BatchNormalization())

model_predict.add(Dropout(0.2))

model_predict.add(Dense(512, activation='relu'))

model_predict.add(BatchNormalization())

model_predict.add(Dropout(0.2))

model_predict.add(Dense(1024, activation='relu'))

model_predict.add(Dense(len(class_names), activation='softmax'))

model_predict.load_weights('crime.h5')



#Running the model on a single image

from keras.utils import load_img

img = image.load_img('/content/Test/NormalVideos/Normal_Videos_003_x264_2200.png',target_size=(64,64))

x=image.img_to_array(img)

x=np.expand_dims(x,axis=0)

prediction = np.argmax(model_predict.predict(x))

class_names[prediction]
```

```
#Running the model on Test_set

predictions = model_predict.predict(test_set)

predicted_labels = np.argmax(predictions, axis=1)

true_labels = test_set.classes



from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report, confusion_matrix

accuracy = accuracy_score(true_labels, predicted_labels)

print(f'Accuracy: {accuracy * 100:.2f}%')



precision = precision_score(true_labels, np.argmax(predictions, axis=1), average='weighted')

recall = recall_score(true_labels, np.argmax(predictions, axis=1), average='weighted')

f1 = f1_score(true_labels, np.argmax(predictions, axis=1), average='weighted')

print(f'Precision: {precision * 100:.2f}%')

print(f'Recall: {recall * 100:.2f}%')

print(f'F1-score: {f1 * 100:.2f}%')

# Print classification report

print('Classification Report:')
```

```
print(classification_report(true_labels, np.argmax(predictions, axis=1)))



# Print confusion matrix

print('Confusion Matrix:')

print(confusion_matrix(true_labels, np.argmax(predictions, axis=1)))
```

13.2 GitHub and Project Demo Link

GitHub Link: https://github.com/smartinternz02/SI-GuidedProject-610134-1699032511

Demo Link: https://drive.google.com/file/d/1sGpPwZVmi43qZYacQm060RmJw2QZXJAc/view?usp=sharing