**Project Title**: **Facial Recognition and Emotion Detection System**  
**Group Members**: Nikita Godinho (30011687) , Missand Carvalho (30013124)

**Tasks and Responsibilities:**

**Nikita**

* Designed and implemented the FaceCropper class.
* Ensured accurate face detection using OpenCV Haar Cascade.
* Verified the cropped faces were saved correctly in the clipped\_faces/ folder.

**Missand**

* Developed the ImageModifier class for image transformations.
* Implemented pixelation, grayscale conversion, and random noise addition.
* Ensured all modified images were saved in the altered\_images/ folder.

**Nikita & Missand**

* Designed and implemented the EmotionRecognizer class.
* Integrated DeepFace for emotion analysis and ensured accurate emotion detection.
* Managed the saving of emotion results in the emotion\_results/ folder.
* Compiled the final report and organized project results.

**Introduction:**

This project demonstrates a facial recognition and emotion detection system. The system uses Python and libraries such as OpenCV for image processing and DeepFace for emotion detection. It implements an object-oriented design to ensure modularity and scalability. The primary objectives are:

* Detect and crop faces from input images.
* Apply modifications such as pixelation, grayscale, and noise to the cropped faces.
* Analyze the emotions in the modified images and display confidence levels.

The project provides practical insights into computer vision and machine learning techniques for image processing and emotion recognition.

**System Design:**

**Architecture Overview**

The system consists of three main components:

1. **FaceCropper**: Detects and crops faces from raw input images.
2. **ImageModifier**: Applies image modifications such as pixelation, grayscale, and noise.
3. **EmotionRecognizer**: Analyzes emotions from the modified images.

**UML Diagram**

The following UML diagram illustrates the design of the system:

A screenshot of a computer

Description automatically generated

**Class Descriptions**

* **FaceCropper**:
  + Attributes: input\_folder, output\_folder
  + Methods: crop\_faces()
* **ImageModifier**:
  + Attributes: input\_folder, output\_folder
  + Methods: pixelate\_image(), grayscale\_image(), add\_random\_noise(), modify\_images()
* **EmotionRecognizer**:
  + Attributes: input\_folder
  + Methods: analyze\_emotions()

**Implementation:**

**Face Cropping**

The FaceCropper class uses OpenCV’s Haar Cascade to detect and crop faces from input images. Cropped faces are saved in the clipped\_faces/ folder.

**Image Modification**

The ImageModifier class modifies the cropped faces by applying:

* Pixelation
* Grayscale
* Random noise

The altered images are saved in the altered\_images/ folder.

**Emotion Detection**

The EmotionRecognizer class uses DeepFace to analyze the emotions in the modified images. It outputs the dominant emotion and the confidence level for each image. Results are stored in the emotion\_results/ folder as .txt files.

**Results:**

**Cropped Faces**

* Cropped faces are saved in the clipped\_faces/ folder.
* Example: A screenshot of a computer

  Description automatically generated

**Modified Images**

* Altered images (pixelated, grayscale, noisy) are saved in the altered\_images/ folder.
* Example:

A screenshot of a computer

Description automatically generated

**Emotion Detection Results**

The following table summarizes the detected emotions and confidence levels:

|  |  |  |
| --- | --- | --- |
| Image Name | Emotion | Confidence **(%)** |
| image0\_face\_1\_pixelated.jpg | neutral | 48.72 |
| image3\_face\_0\_pixelated.jpg | happy | 99.85 |
| IMG\_5411\_face\_0\_noisy.jpg | fear | 48.03 |
| IMG\_5412\_face\_0\_grayscale.jpg | fear | 72.77 |
| image4\_face\_1\_grayscale.jpg | neutral | 87.04 |
| IMG\_5408\_face\_1\_pixelated.jpg | fear | 97.52 |
| image0\_face\_1\_noisy.jpg | fear | 67.83 |
| image4\_face\_1\_noisy.jpg | neutral | 97.48 |
| image4\_face\_1\_pixelated.jpg | neutral | 73.22 |
| IMG\_5408\_face\_1\_grayscale.jpg | happy | 99.99 |
| IMG\_5412\_face\_0\_pixelated.jpg | fear | 88.04 |
| IMG\_5414\_face\_0\_noisy.jpg | happy | 50.71 |
| image0\_face\_1\_grayscale.jpg | fear | 72.68 |
| image1\_face\_0\_noisy.jpg | surprise | 98.93 |
| image1\_face\_0\_pixelated.jpg | surprise | 88.86 |
| image2\_face\_1\_pixelated.jpg | neutral | 95.14 |
| image3\_face\_0\_grayscale.jpg | happy | 98.59 |
| IMG\_5407\_face\_0\_grayscale.jpg | neutral | 94.89 |
| IMG\_5407\_face\_0\_noisy.jpg | neutral | 99.93 |
| IMG\_5411\_face\_0\_pixelated.jpg | neutral | 97.15 |
| IMG\_5414\_face\_0\_pixelated.jpg | sad | 96.11 |
| image1\_face\_0\_grayscale.jpg | sad | 58.04 |
| image2\_face\_1\_grayscale.jpg | neutral | 75.62 |
| IMG\_5407\_face\_0\_pixelated.jpg | neutral | 99.20 |
| IMG\_5408\_face\_1\_noisy.jpg | happy | 99.72 |
| IMG\_5411\_face\_0\_grayscale.jpg | fear | 64.41 |
| IMG\_5412\_face\_0\_noisy.jpg | fear | 62.36 |
| IMG\_5414\_face\_0\_grayscale.jpg | fear | 98.34 |
| image2\_face\_1\_noisy.jpg | neutral | 95.42 |
| image3\_face\_0\_noisy.jpg | happy | 99.27 |

**Graphical User Interface (GUI)**

The graphical user interface (GUI) allows users to interact with the facial recognition and emotion detection system. The interface provides options to:

* Upload images.
* View processed results, including cropped faces, modified images, and detected emotions.

The screenshot below demonstrates the GUI in action:

A screenshot of a person

Description automatically generated

**Challenges and Solutions:**

**Challenges**

1. **Face Detection in Poor Lighting**: Some images failed to detect faces due to low lighting conditions.
2. **Dependency Issues**: Errors occurred while installing DeepFace and other libraries.
3. **Missing or Corrupted Results**: Some images did not generate results in the emotion\_results/ folder.

**Solutions**

1. Ensured input images were well-lit and properly aligned.
2. Updated the Python environment and resolved dependency conflicts by reinstalling libraries.
3. Fixed script logic to handle list-based results from DeepFace.

**Conclusion:**

The project successfully implemented facial recognition and emotion detection. Cropped faces and modified images were processed effectively, and emotions were detected with confidence levels. This demonstrates the practical application of computer vision techniques. Future improvements could include:

* Fine-tuning the emotion detection model for higher accuracy.
* Adding support for additional image modifications.
* Enhancing the GUI for better user interaction.

**References:**

1. OpenCV Documentation: https://docs.opencv.org/
2. DeepFace GitHub Repository: https://github.com/serengil/deepface
3. Python Official Documentation: https://docs.python.org/