Configuration Manual

MSc Research Project
Programme Name

Michael O’Toole
Student ID: x22192131

School of Computing
National College of Ireland

Supervisor: Michael Pantridge

Directory structure

```
deepfake_app/
├── app.py              # The main Python file to run the Flask app
├── uploads/            # Directory to temporarily store uploaded files
├── templates/          # Directory for HTML templates
│   ├── index.html      # Upload form for users
│   ├── result.html     # Page to display analysis results
├── static/             # Static files like CSS, JS, and images
│   ├── css/
│   │   └── styles.css  # Optional CSS file for custom styles
│   ├── js/
│   │   └── script.js   # Optional JavaScript file
└── requirements.txt    # File to list Python dependencies
```

The provided code implements a Flask-based web application that detects potential deepfakes in uploaded videos or images using basic face detection.

- Flask framework is used to handle HTTP requests and render HTML templates.
- The / route serves an HTML form (index.html) where users can upload a file.
- The /upload route processes uploaded files via HTTP POST requests.
- Uploaded files are saved temporarily in the uploads directory using secure_filename to sanitize filenames.

The detect_faces_and_analyze function uses OpenCV to analyze whether a file is likely to be a deepfake.

For videos, the function processes up to 50 frames:

- Converts each frame to grayscale.
- Detects faces using a pre-trained Haar Cascade Classifier (haarcascade_frontalface_default.xml).
- Counts the number of frames with detected faces.

For images, the function checks for faces using the same classifier.
If no faces are detected in a file, it is flagged as a potential deepfake.

After analysis, the application generates a result message (Deepfake detected or No deepfake detected).

The result is displayed on the result.html template.

The application provides meaningful error messages for common issues, such as:

- No file uploaded.
- Issues during file analysis.

Uploaded files are deleted after analysis to maintain a clean working environment.

Uploads Directory

This folder (`uploads/`) is where uploaded files are stored temporarily. The Flask app automatically creates it if it doesn't exist.

Templates Directory

Contains the HTML templates for your web pages:

- **index.html**: The upload form.
- **result.html**: The result display page.

Static Directory

Holds static files like CSS for styling or JavaScript for additional interactivity. You can customize these files as needed.

Commands to Set Up

Create the directory structure:

```
mkdir -p deepfake_app/uploads deepfake_app/templates deepfake_app/static/css
deepfake_app/static/js
```

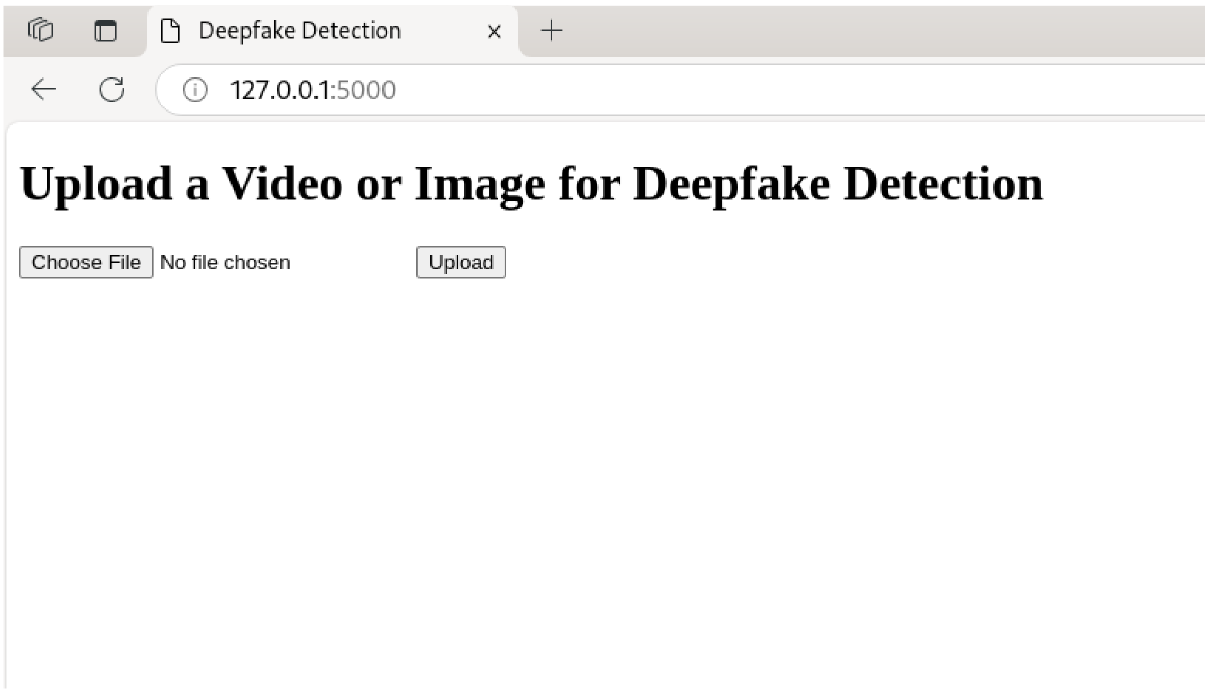
Move files:

Place the Python file (app.py) in the deepfake_app/ directory.
Place the index.html and result.html in deepfake_app/templates/.
Install dependencies:

```
pip install -r requirements.txt
```

Once installed run the application using python app.py

When the application is running then open a browser and visit http://127.0.0.1:5000



Upload a Video or Image for Deepfake Detection

Choose the file that you wish to check and click upload. The image will then be analysed and a determination will be made as to whether it is a deepfake image or not.