PROJECT DESCRIPTION DOCUMENT

GenderLens

AI-Powered Face Recognition & Gender Classification

Project Overview

GenderLens is a web-based academic application developed to detect human faces in an uploaded image and classify each detected face into the model's Male or Female output categories. The application offers a simple interface where a user can learn about the project, upload an image, run face detection and view the annotated image together with individual prediction details.

The computer-vision workflow uses OpenCV and a Haar Cascade classifier to locate faces. Detected face regions are preprocessed and passed to a machine-learning pipeline based on Principal Component Analysis (PCA/Eigenfaces) for feature extraction and a Support Vector Machine (SVM) for classification.

Technology Stack

Area	Technology / Purpose
User Interface	HTML5 and CSS3
Client-side scripting	JavaScript
Back-end / Web framework	Python and Flask
Computer vision	OpenCV (cv2)
Face detection	Haar Cascade Classifier
Feature extraction	PCA (Eigenfaces)
Machine learning	Support Vector Machine (SVM)

Main Application Flow

Home Page → Face Detection Page → Select Image → Detect Faces → View Results

The project has only two user-facing pages. It does not use a database or database tables to store uploaded images, user information or prediction records. Results are displayed for the current detection request in the application interface.

Project Modules Description

The application is organised into two main pages and a supporting image-processing workflow.

1. Home Page Module

The Home page is the entry point of GenderLens. It introduces the project, displays its name and a short description, and explains the basic AI face-detection workflow. The page highlights the main technologies used in the application and provides navigation to the Face Detection page through the navigation link and the Explore Face Detection button.

2. Face Detection Module

The Face Detection page is the main functional area of the application. It allows a user to choose an image from the computer and submit it for processing. The page displays the uploaded image with labels around detected faces. A results section shows the total number of faces detected and a separate result item for each face.

3. Image Processing and Classification Workflow

- Image input: the user selects a supported image file, such as JPG, JPEG or PNG.
- Face localization: OpenCV loads the image and the Haar Cascade classifier identifies face regions.
- Face preprocessing: each detected region is prepared in the format and dimensions expected by the trained model.
- Feature extraction: PCA projects the processed face data into a lower-dimensional feature space using the Eigenfaces approach.
- Classification: the SVM model predicts the Male or Female class and provides a confidence/probability score where available.
- Result presentation: the application draws a bounding box and a compact label on the image, then lists details for each detected face.

4. Result Details Displayed

For each detected face, the results list can show the face number, predicted category, confidence score, face position (X and Y coordinates), face dimensions and a face thumbnail. The total face count is shown separately above the individual result items. The displayed image and the result thumbnails are intended to help users understand which region each prediction refers to.

Data Storage Note

GenderLens does not contain a database layer. No database tables are created for users, uploaded images or prediction history. The uploaded image is processed for the current request and the results are returned to the page; the application does not provide a persistent history module.

Home Page

Figure 1: GenderLens home screen

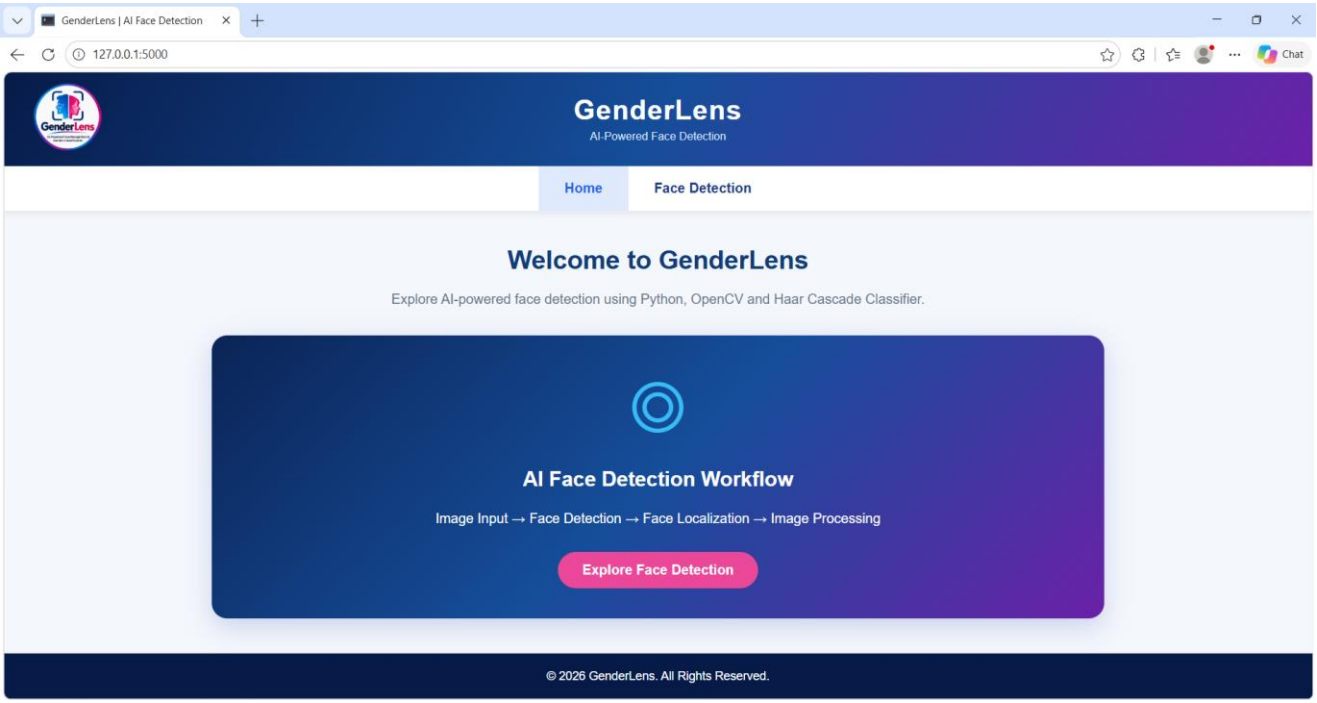

Figure 1: Home Page

The Home page welcomes visitors to GenderLens and introduces the purpose of the application. The header contains the GenderLens branding and the navigation menu with Home and Face Detection options. The main content briefly explains that the project demonstrates AI-powered face detection using Python, OpenCV and the Haar Cascade Classifier.

The central workflow panel presents the high-level sequence: Image Input → Face Detection → Face Localization → Image Processing. The Explore Face Detection button takes the visitor to the page where an image can be uploaded and analysed. A footer at the bottom of the page displays the application copyright notice.

Main Features of the Home Page

- Introduces the GenderLens project and its purpose.
- Displays the technologies used at a high level.
- Explains the basic face-detection workflow.
- Provides navigation to the Face Detection page.

Face Detection Page

Figure 2: Image upload and face detection results

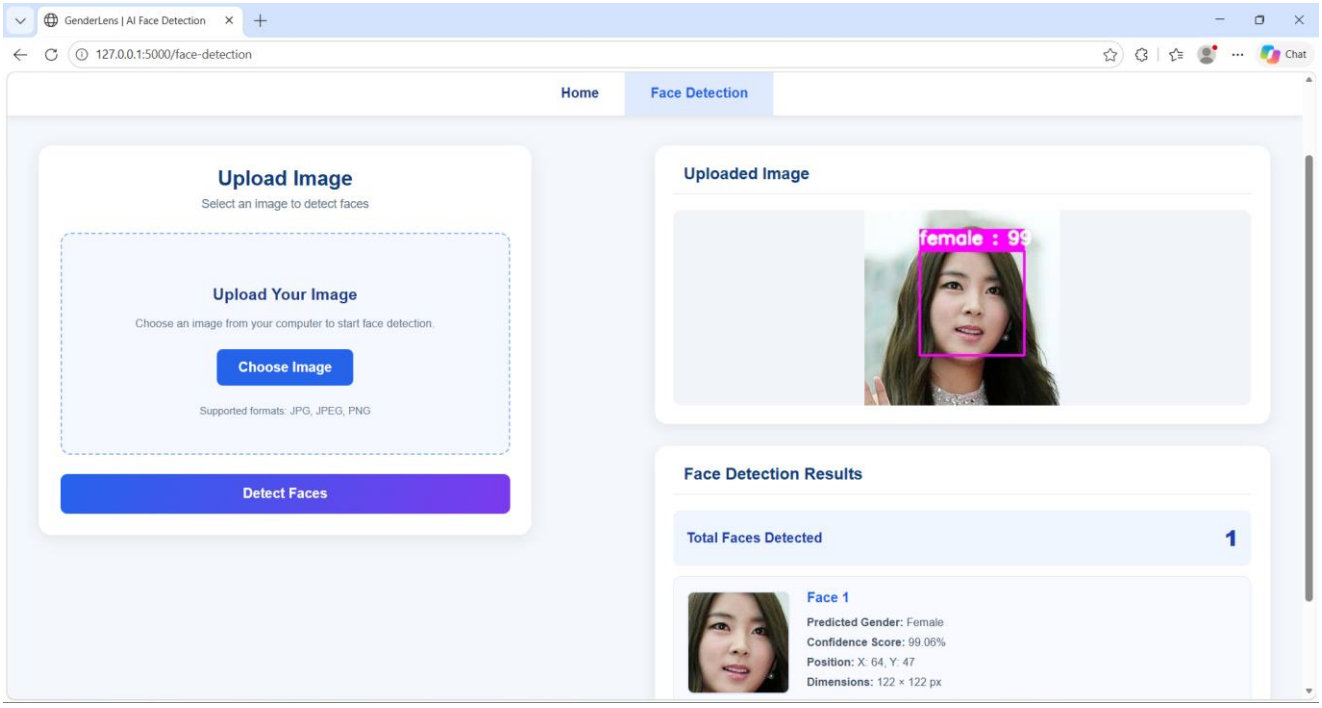

Figure 2: Face Detection Page

The Face Detection page provides the main image-analysis functionality. On the left side, the Upload Image section contains a file-selection control and a Detect Faces button. The user selects an image from the computer and starts the detection process by clicking the button.

On the right side, the Uploaded Image panel displays the selected image after processing. Detected faces are marked with rectangular bounding boxes and compact labels showing the predicted category and score. The Face Detection Results panel displays the total number of detected faces and an individual result item for each face.

Information Presented in the Results List

- Total Faces Detected: number of face regions found in the uploaded image.
- Face Number: sequential identifier for each detected face.
- Predicted Gender: the class label returned by the trained classifier.
- Confidence Score: the model's associated prediction score.
- Position: X and Y coordinates of the detected face region.
- Dimensions: width and height of the detected face region in pixels.
- Face Thumbnail: a cropped preview associated with the corresponding detected face.

Project Flow and Functional Explanation

End-to-end sequence followed when a user performs face detection.

Step-by-Step Working

Step 1 – Open the application: The user opens GenderLens in a web browser and views the Home page.

Step 2 – Navigate to Face Detection: The user selects Face Detection from the navigation menu or clicks Explore Face Detection.

Step 3 – Select an image: The user chooses a supported image file using the Upload Image control.

Step 4 – Start detection: The user clicks Detect Faces. The Flask back-end receives the request and invokes the image-processing pipeline.

Step 5 – Detect and classify faces: OpenCV and Haar Cascade locate face regions. The preprocessing, PCA/Eigenfaces and SVM stages prepare and classify each detected face.

Step 6 – Review the output: The page displays the annotated uploaded image, total face count and per-face details in the results list.

Functional Requirements

- The application should provide Home and Face Detection pages.
- The user should be able to select a supported image from the local computer.
- The application should detect and mark faces in the uploaded image.
- The application should display the predicted class and relevant face details for each detected face.
- The application should show a suitable message when no face is detected or the selected image cannot be processed.

Limitations and Academic Use

The output depends on image quality, lighting, face orientation, image resolution and the training data used to build the classifier. A prediction score should not be interpreted as a guarantee of correctness. The Male/Female labels represent the categories learned by this particular model and do not determine a person's gender identity. The project is developed for learning and demonstration of computer-vision and machine-learning concepts; it should not be used to make consequential decisions about people.

Conclusion

GenderLens demonstrates how a Python and Flask web application can combine OpenCV-based face detection with PCA feature extraction and SVM classification. Its two-page interface keeps the workflow simple: users learn about the project on the Home page, upload an image on the Face Detection page, and review the annotated image and structured prediction details. The application operates without a database or persistent prediction-history feature.