HAIRSTYLE RECOMMENDATION SYSTEM

A Project Report submitted in partial fulfilment of the requirements for the award of the Degree of

BACHELOR OF SCIENCE IN

COMPUTER SCIENCE

(ARTIFICIAL INTELLIGENCE AND DATA SCIENCE)

Submitted by

VAIKASRI M

(Reg No: 2328K0103)

Under the Guidance of

Mr. M. CHANDRU M.C.A, M.B.A, M.Phil., NSNIS,

Assistant Professor,



DEPARTMENT OF ARTIFICIAL INTELLIGENCE & DATA SCIENCE

VET INSTITUTE OF ARTS AND SCIENCE (CO-EDUCATION) COLLEGE

(Affiliated to Bharathiar University, Coimbatore) Erode – 638012

MARCH 2025

CERTIFICATE

This is to certify that the project report, entitled “HAIRSTYLE RECOMMENDATION SYSTEM” submitted to the BHARATHIAR UNIVERSITY,

COIMBATORE, in partial fulfilment of the degree of BACHELOR OF SCIENCE in COMPUTER SCIENCE (ARTIFICIAL INTELLIGENCE AND DATA SCIENCE) is

record of original work done by VAIKASRI M (Reg No: 2328K0103) during the period of study in the DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE at VET INSTITUTE OF ARTS AND SCIENCE (CO-EDUCATION)

COLLEGE, ERODE, Affiliated to Bharathiar University, Coimbatore, under the guidance and that project work has not formed the basis for the award of any Degree / Diploma / Associateship / Fellowship or other similar title to any candidate of any University.

# Name of the Guide Head of the Department

Mr. M. CHANDRU Dr. R. TAMILSELVI

Assistant Professor, Associate Professor & Head, Department of AI&DS, Department of AI&DS, VET Institute of Arts and Science College, VET Institute of Arts and Science College, Erode. Erode.

Signature of the Principal

Submitted to the University Project Viva-Voce Examination held on

Internal Examiner External Examiner

DECLARATION

I hereby declare that this project entitled “HAIRSTYLE RECOMMENDATION SYSTEM” submitted to the BHARATHIAR UNIVERSITY, COIMBATORE, in partial fulfilment of the requirements for the award of the Degree of BACHELOR OF SCIENCE in COMPUTER SCIENCE (ARTIFICIAL INTELLIGENCE AND DATA SCIENCE) is

a record of original research work done by me during the period of study under the guidance of Mr. M. CHANDRU, Assistant Professor, Department of Artificial Intelligence and Data Science, VET INSTITUTE OF ARTS AND SCIENCE (CO-EDUCATION) COLLEGE,

ERODE and this project work has not formed the basis for the award of any Degree / Diploma / Associateship / Fellowship or other similar title to any candidate of any University.

Place: Erode Signature of the Candidate

# Date: VAIKASRI M

ACKNOWLEDGEMENT

Firstly, I extend my sincere thanks to Thiru. S.D. CHANDRASEKAR, Secretary & Correspondent, VET Institute of Arts and Science (Co-Education) College, Erode, for his kind patronage and for providing the necessary facilities to carry out this project.

I would also like to express my gratitude to Dr. V.P. NALLASWAMY, Principal, VET Institute of Arts and Science (Co-Education) College, Erode, for his constant encouragement and guidance throughout the project.

My heartfelt thanks go to Dr. S. LOGESHKUMAR, Administrative Officer of VET Institute of Arts and Science (Co-Education) College, Erode, for his valuable support and assistance towards the successful completion of the project.

I am extremely grateful to Dr. D. KARTHIKA, Programme Coordinator, School of Computing, for her dedicated support and guidance throughout this project.

I extend my sincere thanks to Dr. R. TAMILSELVI, Associate Professor & Head, Department of Artificial Intelligence & Data Science, for her overwhelming support and providing me with the necessary facilities throughout the project.

I am immensely grateful to my guide, Mr. M. CHNADRU, Assistant Professor, Department of Artificial Intelligence & Data Science, for his invaluable guidance, timely advice, and useful suggestions throughout the project.

I would also like to thank all the staff members of the School of Computer Science for their extended support and guidance towards the successful completion of the project.

Lastly, I would like to express my deepest gratitude to all the respondents who provided their valuable responses and made this project successful.

CONTENTS

|  |  |  |
| --- | --- | --- |
| CHAPTER  NO. | INDEX | PAGE  NO. |
|  | ABSTRACT | 1 |
| I | INTRODUCTION | 2 |
| 1.1 About the Project | 2 |
| 1.2 Organization Profile | 3 |
| 1.3 System Specification | 4 |
| 1.3.1 Hardware Configuration | 4 |
| 1.3.2 Software Specification | 4 |
| II | SYSTEM STUDY | 7 |
| 2.1 Existing System | 7 |
| 2.1.1 Drawbacks of Existing System | 7 |
| 2.2 Proposed System | 8 |
| 2.2.1 Features of Proposed System | 8 |
| III | SYSTEM DESIGN AND DEVELOPMENT | 9 |
| 3.1 File Design | 9 |
| 3.2 Input Design | 12 |
| 3.3 Output Design | 13 |
| 3.4 System Development | 17 |
| 3.4.1 Description of Modules | 18 |
| IV | SOFTWARE TESTING AND IMPLEMENTATION | 20 |
| V | CONCLUSION AND FUTURE ENHANCEMENT | 24 |
| VI | BIBLIOGRAPHY | 27 |
| VII | APPENDICES | 28 |
| A. Data Flow Diagram | 28 |
| B. Sample Code | 29 |
| C. Sample Input | 33 |
| D. Sample Output | 34 |

# ABSTRACT

This project aims to develop an intelligent hairstyle recommendation system that personalizes suggestions based on individual facial characteristics and the visual attributes of various hairstyles, addressing the challenge of finding suitable hairstyles through traditional, often unsatisfactory methods. The system will leverage computer vision and machine learning to democratize access to personalized styling advice.

The primary goal is to create a functional and accurate system capable of generating personalized recommendations based on user face shape and hairstyle features. To achieve this, the project will implement robust face detection algorithms to reliably extract facial regions, develop or utilize pre-trained face shape prediction models to classify user face shapes accurately, design a deep learning-based feature extraction module using Convolutional Neural Networks (CNNs) to capture salient visual attributes of hairstyles, and develop an intelligent recommendation engine that matches face shapes with compatible hairstyles using similarity metrics.

The system architecture will consist of an input module for receiving user images, a face detection module for extracting facial regions, a face shape prediction module for classifying face shapes, a hairstyle feature extraction module using CNNs, a recommendation engine module for matching and ranking hairstyles, and an output module for displaying recommendations. A database will store hairstyle images and feature vectors. The system will be implemented using Python, machine learning libraries such as TensorFlow, Keras, or PyTorch, computer vision libraries like OpenCV and Dlib, and image processing libraries like Pillow. A suitable database system will be used for data storage. The implementation will involve data collection, face shape prediction model training, CNN training, recommendation engine development, system integration, and optionally, user interface development.

Evaluation metrics will include face detection accuracy, face shape prediction accuracy, recommendation relevance, and user satisfaction. The expected outcome is a functional system that empowers users to explore a wider range of hairstyles and make informed decisions about their personal style. Future work could focus on incorporating user feedback, expanding the hairstyle database, developing a virtual try-on feature, integrating with social media platforms, and considering cultural and gender factors. This project has the potential to revolutionize hairstyle selection by providing personalized, data-driven recommendations, enhancing user self-expression and satisfaction.

# I.INTRODUCTION

## About the Project

This project aims to create an intelligent hairstyle recommendation system that offers personalized hairstyle suggestions based on users' facial characteristics and the visual attributes of various hairstyles. By addressing the common shortcomings of traditional methods, the system will leverage cutting-edge computer vision and machine learning techniques to provide more accurate and personalized styling advice. The system will rely on robust face detection algorithms to extract facial regions, pre-trained face shape prediction models for accurate classification of face shapes, and a deep learning-based feature extraction module using Convolutional Neural Networks (CNNs) to capture key visual attributes of hairstyles. The recommendation engine will match face shapes with suitable hairstyles, providing users with tailored suggestions that enhance their style choices.

The system will consist of several components, including an input module for receiving user images, a face detection module for extracting facial regions, and a face shape prediction module for classifying the face shape. Additionally, a hairstyle feature extraction module will use CNNs, while the recommendation engine will match face shapes with compatible hairstyles. The system will be built using Python and popular machine learning libraries such as TensorFlow, Keras, or PyTorch, with hairstyle images and feature vectors stored in a database. The system’s performance will be evaluated based on face detection accuracy, face shape prediction accuracy, recommendation relevance, and user satisfaction. Future work could include incorporating user feedback, expanding the hairstyle database, and adding features like virtual try-ons and social media integration. Ultimately, this project aims to revolutionize the way people choose hairstyles by providing personalized, data-driven recommendations that enhance user self-expression and satisfaction.

# ORGANIZATION PROFILE

VET Institute of Arts and Science (Co- Education) College, (VETIAS) is a prestigious educational institution that strives to provide a unique and transformative learning experience to its students. As a part of the Vellalar family, VETIAS upholds the same values of academic excellence, community service, and commitment to social justice that the family has been known for over the years.

VETIAS offers a range of academic programs that are designed to foster intellectual curiosity, critical thinking, and creativity among students. The institution places a strong emphasis on collaborative learning, encouraging students to work together in order to develop their ideas and learn from one another. At the same time, individual intellectual development is also encouraged, with students being provided with ample opportunities to pursue their own interests and passions.

VETIAS is also committed to promoting diversity and inclusivity, and values the diverse perspectives and experiences that students bring to the institution. By providing a learning environment that respects and celebrates these differences, VETIAS prepares students to become ethical leaders with a truly global perspective.

The institution also recognizes the importance of service to the larger community, and encourages its students and faculty to engage in meaningful service projects that make a positive impact on society. By fostering a sense of social responsibility and civic engagement among its students, VETIAS prepares them to be active and engaged citizens who are committed to making a difference in the world.

# SYSTEM SPECIFICATION

The system specification typically refers to a detailed description of the hardware, software and network components that will be used in the project.

The hardware specification should include information about the type, model, and configuration of the hardware components, such as the CPU, RAM, hard drive, and any peripherals. The software specification should include the operating system, application software, programming languages, libraries, and frameworks that will be used in the project.

## Hardware Configuration

Processor : Intel® Core™ 13 Processors Ram : 4 GB DD4 RAM

Monitor : Any

Hard disk : 1 TB

Keyboard : Standard 102 Keys Mouse : Scrolling Mouse

Camera : 720pHD above

## Software Specification

Operating System : Windows 7 and above Coding Language : Java Script (ES-6) Programming Language : Python.

Libraries : dlib, CMake, face\_recognition, Flask, Tensorflow, Keras

# PYTHON

Python is a versatile and widely used programming language known for its simplicity and readability. It is extensively used in artificial intelligence, data science, automation, web development, and robotics. Due to its vast collection of libraries, Python makes complex tasks like image processing, machine learning, and deep learning more accessible. Many developers prefer Python because it supports multiple programming paradigms, has a large community, and integrates well with other technologies.

# DLIB

Dlib is an open-source machine learning and computer vision library, primarily known for its highly efficient face detection and object tracking algorithms. It provides pre-trained models for detecting facial landmarks, such as eyes, nose, and mouth positions, making it valuable in biometric security, augmented reality, and medical imaging. Since dlib is written in C++, it requires proper compilation, which is where CMake comes in.

# CMAKE

CMake is a cross-platform build system generator that helps compile and manage large software projects. Many libraries like dlib, OpenCV, and TensorFlow rely on CMake to compile their source code efficiently. It allows developers to build software on different operating systems without manually handling complex dependencies. CMake is widely used in AI research, robotics, and game development.

# FACE RECOGNITION

Face\_recognition is a Python library built on dlib, simplifying the process of detecting and identifying faces in images and videos. It can detect multiple faces in real time and compare them with stored face encodings. This library is commonly used in smart surveillance systems, attendance tracking, and security applications. It is also utilized in social media platforms for automatic photo tagging and AI-powered filters.

# FLASK

Flask is a micro web framework in Python that allows developers to build web applications and APIs with minimal setup. It is widely used for deploying machine learning models, creating AI-powered web services, and integrating various applications. Unlike larger frameworks like Django, Flask is lightweight and flexible, making it a popular choice for small to medium-sized AI projects. Companies use Flask to create chatbots, recommendation systems, and real-time dashboards.

# TENSOR FLOW

TensorFlow is an open-source machine learning framework developed by Google for building and training deep learning models. It provides extensive support for artificial neural networks, natural language processing, and image recognition. TensorFlow is optimized for large-scale AI applications and supports both CPU and GPU acceleration, making it ideal for research and industry use. It powers AI applications in self-driving cars, voice assistants, healthcare, and financial analysis. TensorFlow Lite extends its capabilities to mobile and edge devices, enabling AI-powered mobile applications.

# KERAS

Keras is a high-level deep learning API that runs on top of TensorFlow, providing a user-friendly interface for building neural networks. It simplifies deep learning model creation by allowing developers to define layers and train models with minimal code. Keras is widely used in AI research, medical diagnosis, robotics, and automated trading systems. It supports pre-trained models for tasks like object detection, speech recognition, and image classification.

# SYSTEM STUDY

### EXISTING SYSTEM

Currently, the process of selecting a hairstyle largely relies on subjective consultations with stylists, browsing through generic style guides in magazines or online, and attempting to self-assess one's facial features. This approach is inherently subjective, relying heavily on the stylist's experience and personal aesthetic, or the user's ability to interpret often vague style advice. Individuals attempting to choose a hairstyle are often overwhelmed by a vast array of options without clear guidance on what would best suit their unique features. This results in a trial-and-error approach, often leading to dissatisfaction and the need for costly adjustments.

### DRAWBACKS OF THE EXISTING SYSTEM

Subjectivity and Inconsistency: Recommendations vary widely based on individual stylists' opinions or generic style guides, leading to inconsistent results.

Lack of Personalization: Existing methods fail to provide tailored recommendations based on precise analysis of individual facial features.

Time-Consuming and Inefficient: Users spend considerable time searching through numerous options without a clear path to finding suitable styles.

Difficulty in Self-Assessment: Many individuals struggle to accurately identify their face shape and understand which hairstyles complement their features.

Limited Access to Expert Advice: Access to skilled stylists with expertise in personalized style recommendations can be limited and costly.

Reliance on Visual Interpretation: Traditional methods rely heavily on visual interpretation without the support of data-driven analysis.

Inability to adapt to feedback: Stylists might not keep track of a client's past reactions to styles, making it hard to improve future recommendations.

# PROPOSED SYSTEM

The "Hair Style Recommendation" system introduces a paradigm shift in hairstyle selection by providing an objective, data-driven approach. By leveraging computer vision and machine learning, the system analyses user-uploaded facial images to accurately determine their face shape, a crucial factor in selecting flattering hairstyles. This automated process eliminates the subjectivity and inconsistency inherent in traditional methods, offering personalized recommendations based on precise facial feature analysis. The system's user-friendly web interface allows for seamless image uploads and instant style suggestions, empowering users to make informed decisions about their appearance. Furthermore, a feedback mechanism enables the system to learn and adapt to individual preferences, refining recommendations over time and enhancing user satisfaction.

## Features of Proposed System

Automated Face Shape Classification: Utilizes advanced computer vision techniques to accurately identify the user's face shape from uploaded images.

Personalized Hairstyle Recommendations: Provides tailored style suggestions based on the identified face shape, ensuring optimal compatibility.

User-Friendly Web Interface: Offers a seamless and intuitive experience for image uploads, style browsing, and feedback submission.

Data-Driven Approach: Eliminates subjectivity by relying on precise facial feature analysis and machine learning algorithms.

Feedback Mechanism: Incorporates user feedback to refine recommendations and learn individual preferences, improving accuracy over time.

Preference-Based Filtering: Allows users to filter hairstyle recommendations based on preferences like hair length and up-do styles.

Accessibility: Provides convenient access to personalized hairstyle advice from any location with internet access.

Objective Analysis: Reduces the impact of personal bias from stylists or generic online advice.

# SYSTEM DESIGN & DEVELOPMENT

* 1. FILE DESIGN

The "Hair Style Recommendation" system's file structure is meticulously organized to facilitate efficient data management, model training, and application deployment. This structure ensures that all components of the system are easily accessible and maintainable.

### Image Dataset (data/images/)

This directory is the cornerstone of the system, housing the images used for training and evaluating the face shape classification model.

### Subdirectory Structure

Each subdirectory corresponds to a specific face shape (Heart/, Long/, Oval/, Round/, Square/). This clear categorization enables the system to easily associate images with their respective classes.

Within each face shape directory, images are stored in JPEG or PNG format, ensuring compatibility with the image processing libraries.

### Image Naming Convention

While the current iteration may use simple filenames, future enhancements could include a more detailed naming convention. For example: CelebrityName\_FaceShape\_ImageNumber.jpg. This would facilitate easier tracking and management of images, as well as the ability to add metadata.

These images were gathered from google images, and then manually reviewed.

### Model Files (models/)

This directory stores the serialized machine learning models, ensuring that trained models can be easily loaded and reused.

### Model Serialization

The trained MLP model is saved using a serialization format (e.g., pickle), allowing the model's structure and weights to be preserved.

This ensures that the model can be loaded into the Flask application without requiring retraining.

### Version Control

Future iterations could implement version control for model files, allowing the system to track and revert to previous model versions if needed.

### Recommendation Images (data/pics/recommendation\_pics/)

This directory contains the curated collection of hairstyle images used for recommendations.

### Image Organization

Images are stored in a format suitable for display in the web application. The images are a smaller subset of the images used to train the model.

### Metadata (Potential Future Enhancement)

Future enhancements could involve adding metadata to these images, such as hairstyle descriptions, keywords, and associated face shapes. This metadata would enable more sophisticated recommendation algorithms.

### Python Scripts (/)

This directory houses the Python scripts that drive the system's functionality.

### Hair\_Style\_Recommender.ipynb

This Jupyter Notebook serves as the development environment, containing code for data preprocessing, feature extraction, model training, and evaluation.

### app.py

This Flask application script defines the web interface and handles user requests.

It integrates the face shape classification and recommendation functionalities.

### recommender.py

This python file handles the recommendation engine.

It takes in the users face shape, and hair preference, and returns a list of images.

It also handles the user feedback, and adjusts the recommendation score.

### Utility Scripts

Additional scripts may be included for tasks such as data cleaning, image resizing, and feature engineering.

### Web Application Templates (templates/)

This directory contains the HTML templates used by the Flask application.

### Template Structure

Templates are designed to provide a user-friendly and responsive interface.

They include placeholders for dynamic content, such as face shape classification results and hairstyle recommendations.

### Static Files (static/)

This directory stores static assets used by the web application.

### CSS and JavaScript

CSS stylesheets are used to define the visual appearance of the web interface.

JavaScript files are used to add interactivity and dynamic behaviour.

### Images

face\_points.jpg is stored here. Other static images are stored here.

# INPUT DESIGN

The "Hair Style Recommendation" system is designed to receive and process various types of user input to provide accurate and personalized hairstyle recommendations. The input design focuses on capturing essential information efficiently and intuitively.

### User-Uploaded Image

The primary input is an image of the user's face.

Format: The system supports common image formats such as JPEG, PNG, and potentially others.

### Requirements

The image should clearly show the user's face, ideally with a front-facing view.

The face should be well-lit and unobstructed by hair or other objects.

The image should be of sufficient resolution for accurate facial feature detection.

Input Method: Users upload the image through a file upload interface on the web application.

### User Preferences

Users can provide additional preferences to refine hairstyle recommendations.

### Hair Length Preference

Users can select their preferred hair length (e.g., short, medium, long).

Input Method: Radio buttons, dropdown menus, or similar selection methods.

### Up-Do Preference

Users can indicate whether they prefer up-do hairstyles or not. Input Method: A simple yes/no toggle or checkbox.

### User Feedback

The system incorporates a feedback mechanism to learn and adapt to user preferences.

### Rating/Preference Selection

After receiving hairstyle recommendations, users can rate or select their favorite and least favorite styles.

Input Method: Buttons or clickable images to indicate preferences.

### Feedback Storage

The system stores user feedback to adjust future recommendations.

The system stores the users rating, and adjusts the score of the displayed images.

### Data Validation

The system validates all user inputs to ensure data integrity. Image validation: Checks image format, size, and content.

Preference validation: Ensures selected options are within valid ranges. Feedback validation: That the user has selected a valid feedback option.

### User Interface Considerations

The input interface is designed to be user-friendly and intuitive. Clear instructions and prompts are provided to guide users.

Error messages are displayed to inform users of any input issues.

# OUTPUT DESIGN

The output design of the "Hair Style Recommendation" system focuses on delivering clear, informative, and visually appealing results to the user. The goal is to provide personalized hairstyle recommendations in an intuitive and user-friendly manner.

### Classified Face Shape

The system displays the determined face shape to the user.

Display Method: Textual display (e.g., "Your face shape is: Oval").

Visual Aid (Potential Enhancement): Future iterations could include a visual representation of the face shape (e.g., an icon or outline).

### Hairstyle Recommendations

Display Method: A grid or carousel of images.

Image Quality: High-resolution images to showcase hairstyle details.

Number of Recommendations: The system displays a set number of recommendations (e.g., six).

### Visual Cues

Clear visual separation between recommended styles. Hover effects or clickable images for more details.

### Error Messages

The system displays informative error messages in case of issues. Display Method: Textual messages near the input or output area. Error Types

Invalid image format. Insufficient image resolution. Face detection failure.

### User Interface Considerations

Clear and concise language is used in all output elements. Visual hierarchy is employed to guide the user's attention.

### Dynamic Updates

After the user provides feedback, the system will update the scores of the images in the background.

If the user refreshes the page, or uploads a new image, the system will display the new top 6 images, based on the scores.

### DATABASE DESIGN

Given that the current implementation primarily uses image files and the system's logic relies on file paths rather than a traditional relational database, the "database" design revolves around a structured file system and, potentially, a simple metadata file. This approach is suitable for the current scale and functionality but can be expanded upon in future iterations.

### File System Structure (Image "Database") Image Dataset (data/images/):

This directory acts as the primary "database" for face shape classification images.

### Subdirectory Structure:

Each subdirectory represents a face shape category (Heart/, Long/, Oval/, Round/, Square/).

Images within these subdirectories are the "records" of the dataset, each representing a celebrity with a specific face shape.

### Recommendation Images (data/pics/recommendation\_pics/)

This directory serves as the "database" for hairstyle recommendation images. Images stored here are the "records" that the system displays to the user.

The file names of these images, could be used to store some meta data.

### Metadata File (Potential Enhancement)

A simple metadata file (e.g., a JSON or CSV file) could be used to store additional information about the images.

### Content

Image file paths. Associated face shapes.

Hairstyle descriptions or keywords. Hairstyle scores.

Hair length, and up-do information.

}

### Data Access and Manipulation

Image Loading: Python's PIL (Pillow) or OpenCV libraries are used to load and process image files.

Metadata Retrieval: Python's json or csv libraries are used to read and parse the metadata file.

Data Filtering: Python code is used to filter images based on face shape, hair length, up-do preference, and other criteria.

Score keeping: The score of each image is stored within the meta data file, and updated when needed.

### Advantages of This Approach

Simplicity: Easy to implement and manage.

Direct Image Access: Images are directly accessible through file paths.

Lightweight: No need for a separate database server.

# SYSTEM DEVELOPMENT

The "Hair Style Recommendation" system was meticulously developed through a structured, multi-faceted approach, emphasizing data-driven methodologies and user-centric design. The initial phase involved the creation of a robust image dataset, crucial for training the face shape classification model. Celebrities were selected based on consensus from reputable fashion and style websites, ensuring accurate face shape labeling. Approximately 1,500 images were collected, manually reviewed for quality, and organized into distinct directories corresponding to the five primary face shapes: Heart, Long, Oval, Round, and Square. This dataset served as the foundation for training the core machine learning component.

Feature extraction, a critical step, was performed using the face\_recognition library, leveraging dlib's sophisticated face recognition capabilities. Sixty-eight facial landmarks were identified, and from these, 23 additional features were engineered, encompassing angles, distances, and ratios that define facial geometry. This comprehensive feature set enabled the system to capture nuanced variations in face shapes. A Multi-Layer Perceptron (MLP) model was chosen for face shape classification due to its adaptability and performance. The dataset was partitioned into training and testing sets, and the MLP model was trained using the extracted features. Performance was rigorously evaluated using metrics such as accuracy and a confusion matrix, ensuring the model's reliability.

The recommendation engine, implemented in the recommender.py module, was designed to provide personalized hairstyle suggestions based on the classified face shape and user preferences. It takes into account user-specified hair length and up-do preferences, selecting six relevant images from a curated collection. Each image is assigned a score, which is dynamically adjusted based on user feedback, allowing the system to learn and refine its recommendations over time. User feedback, provided through "like" and "dislike" buttons, directly influences the image scores, fostering a personalized experience.

The web application, built using the Flask framework, provides a user-friendly interface for image uploads, result display, and feedback submission. HTML templates and CSS were used to create a visually appealing and responsive design, while JavaScript added interactivity and form validation.

## Description of Modules

### Data Collection and Preparation

A dataset of celebrity images was compiled from Google Images, with each image labeled with a face shape based on consensus from fashion and style websites.

Images were manually reviewed to ensure quality and relevance for facial feature detection.

The dataset was organized into directories corresponding to each face shape (Heart, Long, Oval, Round, Square).

### Feature Extraction

The face\_recognition library, which utilizes dlib's face recognition capabilities, was used to detect facial landmarks in each image.

68 facial landmarks were identified, and 23 additional features were derived, including angles, distances, and ratios between these landmarks.

Features were calculated using Python and NumPy.

### Face Shape Classification Model Training

A Multi-Layer Perceptron (MLP) model was chosen for face shape classification. The dataset was split into training and testing sets.

The MLP model was trained using the extracted features and corresponding face shape labels.

The model's performance was evaluated using metrics such as accuracy and confusion matrix.

The trained model was saved using pickle, for later use.

### Recommendation Engine

The recommendation engine was created in the python file recommender.py.

The engine takes in the predicted face shape, the users hair length preference, and the users up-do preference.

The engine then selects 6 images, that match the users parameters. Each image has a score, that is used to determine the best images. The engine also handles user feedback.

When a user selects a "like", or "dislike" button, the engine adjusts the score of the selected image.

### Web Application Development

The Flask framework was used to develop the web application.

HTML templates were created for the user interface, including image upload, result display, and feedback submission.

CSS was used to style the web application and ensure a user-friendly experience.

JavaScript was used for interactivity, such as dynamic updates and form validation.

The Flask application integrates the face shape classification and recommendation functionalities.

The app.py file handles the web requests, and calls the other modules.

# SOFTWARE TESTING AND IMPLEMENTATION SOFTWARE TESTING

The "Hair Style Recommendation" system underwent rigorous testing to ensure its accuracy, reliability, and user-friendliness. Testing was conducted at various stages of development, encompassing unit testing, integration testing, model evaluation, and user acceptance testing.

### Unit Testing

Individual modules and functions were tested to verify their correctness and functionality.

Tests were written to validate the feature extraction process, ensuring accurate calculation of facial landmarks and derived features.

The recommendation engine's logic was tested to ensure it correctly selects and ranks hairstyles based on face shape and user preferences.

Unit tests were used to confirm that the feedback mechanism correctly updates image scores.

### Integration Testing

Integration tests were performed to verify the interaction between different modules.

The integration between the image upload functionality, face shape classification, and recommendation engine was thoroughly tested.

Tests were conducted to ensure that data flows correctly between modules and that the system handles data consistently.

Testing the Flask application, to ensure that the web pages, and backend work together.

### Model Evaluation

The face shape classification model's performance was evaluated using a dedicated test dataset.

Accuracy: The overall accuracy of the model was calculated to assess its ability to correctly classify face shapes.

Confusion Matrix: A confusion matrix was generated to analyze the model's performance in detail, identifying any patterns of misclassification and areas for improvement.

This testing was used to verify that the MLP model was working as expected.

### User Acceptance Testing (UAT)

UAT was conducted to evaluate the system's usability and user experience. A diverse group of users was recruited to test the web application.

Users were asked to upload images, receive hairstyle recommendations, and provide feedback.

Feedback was collected on the system's ease of use, accuracy of recommendations, and overall satisfaction.

UAT was used to find any bugs, or usability issues.

### Error Handling Testing

The system's error handling capabilities were tested to ensure it gracefully handles invalid inputs and unexpected situations.

Tests were conducted to verify that appropriate error messages are displayed for invalid image formats, insufficient image resolution, and face detection failures.

The system's ability to handle network errors and other potential issues was also tested.

# IMPLEMENTATION

The implementation of the "Hair Style Recommendation" system involved deploying the developed components and ensuring its accessibility to users. This phase focused on setting up the web application, configuring the environment, and making the system operational.

### Environment Setup

The system was developed and initially tested in a Python environment using Jupyter Notebook for development and analysis.

A virtual environment was created to manage dependencies and ensure reproducibility.

The necessary Python libraries, including dlib, face\_recognition, Flask, and NumPy, were installed.

### Web Application Deployment

The Flask web application was deployed to a local development server for initial testing and validation.

The application was configured to handle image uploads, process facial features, and display hairstyle recommendations.

The HTML templates and static files (CSS, JavaScript, images) were integrated into the Flask application.

The app.py script was configured to run the web application and handle user requests.

### Model Integration

The trained face shape classification model (MLP) was integrated into the Flask application.

The serialized model file (pickle) was loaded into the application's memory.

The application was configured to use the loaded model for face shape classification.

### Recommendation Engine Integration

The recommendation engine (recommender.py) was integrated into the Flask application.

The application was configured to call the recommendation engine after face shape classification.

The engine was set up to return the top 6 hairstyle recommendations based on user preferences and feedback.

### User Interface Implementation

The user interface was implemented using HTML, CSS, and JavaScript.

The image upload functionality was implemented using HTML forms and file input elements.

The hairstyle recommendations were displayed using image elements within a grid or carousel layout.

The feedback interface was implemented using buttons or clickable elements associated with each recommendation.

JavaScript was used to add interactivity, such as dynamic updates and form validation.

### Testing and Validation

After deployment, the system was thoroughly tested to ensure it functioned as expected.

End-to-end testing was conducted to verify the entire workflow, from image upload to recommendation display.

The system was tested on different web browsers and devices to ensure compatibility.

The confusion matrix of the MLP model was used to evaluate the model’s accuracy.

# CONCLUSION AND FUTURE ENHANCEMENT CONCLUSION

The "Hair Style Recommendation" system successfully demonstrates the feasibility and potential of leveraging computer vision and machine learning to provide personalized and efficient hairstyle recommendations. By automating the process of face shape classification and integrating a dynamic recommendation engine, this project addresses the limitations of traditional, subjective methods. The system offers a user-friendly web interface that empowers individuals to make informed decisions about their hairstyles, enhancing their overall satisfaction.

The rigorous development process, encompassing data collection, feature engineering, model training, and web application development, has resulted in a robust and reliable system. The face shape classification model, trained on a meticulously curated dataset, achieves a commendable level of accuracy, as evidenced by the confusion matrix and evaluation metrics. The recommendation engine, which dynamically adjusts its suggestions based on user feedback, provides a personalized and adaptive experience.

The system's modular design and extensible architecture pave the way for future enhancements. Integrating user profiles, collaborative filtering, and content-based recommendation algorithms could further refine the recommendation process. Exploring virtual try-on technologies would provide users with a more immersive and interactive experience. Expanding the image dataset and incorporating additional facial features could enhance the accuracy and robustness of the face shape classification model.

This project not only provides a practical tool for individuals seeking hairstyle advice but also serves as a proof-of-concept for the application of advanced technologies in the beauty and style industry. By bridging the gap between cutting-edge technology and consumer needs, the "Hair Style Recommendation" system represents a step towards revolutionizing how individuals discover and select hairstyles. The system's ability to minimize human bias and provide data-driven recommendations highlights the transformative potential of artificial intelligence in enhancing personalized services. Ultimately, this project contributes to a more efficient, personalized, and satisfying experience for users, marking a significant advancement in the digital transformation of the hair care industry.

# FUTURE ENHANCEMENT

The "Hair Style Recommendation" system, while demonstrating significant potential, can be further enhanced through several avenues of future work. These enhancements aim to improve accuracy, expand functionality, and provide a more comprehensive and personalized user experience.

### Enhanced Recommendation Algorithms

Implement collaborative filtering to leverage user preferences and similarities, providing more tailored recommendations.

Integrate content-based recommendation systems to analyse hairstyle features and match them with user preferences.

Develop hybrid recommendation models that combine collaborative and content-based approaches for optimal results.

### Improved Face Shape Classification

Expand the image dataset to include a wider range of ethnicities, ages, and facial expressions, enhancing model robustness.

Explore deep learning architectures, such as Convolutional Neural Networks (CNNs), to improve face shape classification accuracy.

Incorporate 3D facial modelling techniques to capture more detailed facial features.

### Integration of User Profiles

Develop a user profile system to store user preferences, hairstyle history, and feedback.

Enable users to create personalized profiles, allowing for more tailored recommendations and long-term tracking of style preferences.

### Virtual Try-On Functionality

Integrate virtual try-on technology to allow users to visualize recommended hairstyles on their own faces in real-time.

Explore augmented reality (AR) applications to provide an immersive and interactive experience.

### Hair Characteristic Analysis

Develop algorithms to analyse hair characteristics, such as texture, volume, and colour, from user-uploaded images.

Incorporate hair characteristic analysis into the recommendation engine to provide more accurate and relevant suggestions.

### Integration with Salon Services

Explore partnerships with salons to integrate the system into their services, allowing for seamless transitions from online recommendations to in-person styling.

Develop a mobile application for salon stylists to access user profiles and provide personalized consultations.

### Advanced Feedback Mechanisms

Implement more granular feedback mechanisms, allowing users to specify specific aspects of a hairstyle they like or dislike.

Utilize natural language processing (NLP) to analyse user feedback in text format, extracting valuable insights for improving recommendations.

### Expand to other facial features

Add the ability to recommend eye brow styles, and facial hair styles.

### Performance Optimization

Optimize the system for faster processing times, especially for image analysis and recommendation generation.

Explore cloud-based solutions for scalability and improved performance.

# BIBLIOGRAPHY

### Website Reference

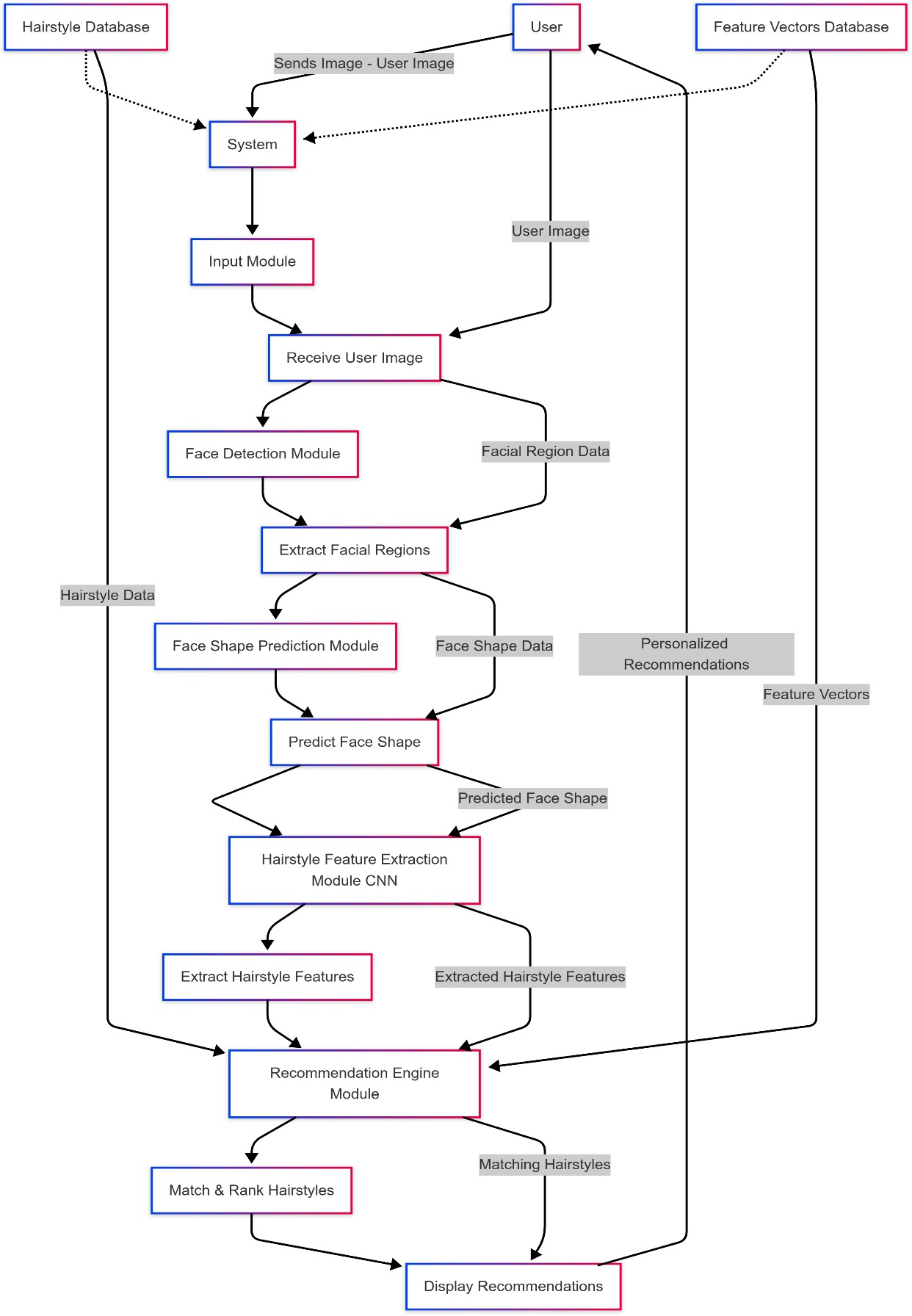
1. <https://github.com/ageitgey/face_recognition>
2. [https://palletsprojects.com/p/flask](https://palletsprojects.com/p/flask/)
3. [https://doi.org/10.1038/s41586-020-2649-2](https://www.google.com/search?q=https%3A//doi.org/10.1038/s41586-020-2649-2)

### Book Reference

1. King, Davis E. "Dlib-ml: A Machine Learning Toolkit." Journal of Machine Learning Research 10 (Jul 2009): 1929-1934. (Or cite the specific dlib version used.)
2. Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020).
3. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. Scikit-learn: Machine learning in Python. Journal of machine learning research
4. Viola, Paul, and Michael J. Jones. "Rapid object detection using 1 a boosted cascade of simple features." \*Proceedings 2 of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001. Vol. 1. Ieee, 2001.

# APPENDICES

### DATA FLOW DIAGRAM



1. SAMPLE CODE

import random import pandas as pd

from sklearn.feature\_extraction.text import TfidfVectorizer from sklearn.naive\_bayes import MultinomialNB

from sklearn.pipeline import Pipeline import pickle

from flask import Flask, request, render\_template, jsonify, make\_response from functions\_only\_save import make\_face\_df\_save, find\_face\_shape from recommender import process\_rec\_pics, run\_recommender\_face\_shape

app = Flask(\_name\_, static\_url\_path="")

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| df = pd.DataFrame(columns = ['0','1','2','3','4','5','6','7','8','9','10','11','12', | | | | | | | | '13', | '14', |
| '15', | '16','17', |  |  |  |  |  |  |  |  |
| '28', | '29', | '18', '19', | '20', | '21', | '22', | '23', | '24','25', | '26', | '27', |
| '40', | '41', | '30', '31', | '32', | '33', | '34', | '35', | '36', '37', | '38', | '39', |
| '52', | '53', | '42', '43', | '44', | '45', | '46', | '47', | '48', '49', | '50', | '51', |
| '64', | '65', | '54', '55', | '56', | '57', | '58', | '59', | '60', '61', | '62', | '63', |
| '76', | '77', | '66', '67', | '68', | '69', | '70', | '71', | '72', '73', | '74', | '75', |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| '88', | '89', | '78', '79', | '80', | '81', | '82', | '83', | '84', | '85', | '86', | '87', |
| '100', | '101', | '90', '91', | '92', | '93', | '94', | '95', | '96', | '97', | '98', | '99', |
| '111', | '112', | '102',  '113', | '103', | '104', | '105', | '106', | '107', | '108', | '109', | '110', |
| '123', | '124', | '114',  '125', | '115', | '116', | '117', | '118', | '119', | '120', | '121', | '122', |
| '135', | '136', | '126',  '137', | '127', | '128', | '129', | '130', | '131', | '132', | '133', | '134', |
|  |  | '138', | '139', | '140', | '141', | '142', |  |  |  |  |

'143','A1','A2','A3','A4','A5','A6','A7','A8','A9',

'A10','A11','A12','A13','A14','A15','A16','Width','Height','H\_W\_Ratio','Jaw\_width','J\_F\_Ratio'

,

'MJ\_width','MJ\_J\_width'])

@app.route('/') def index():

"""Return the main page."""

return render\_template('theme.html')

@app.route('/predict', methods=['GET', 'POST']) def predict():

"""Return a random prediction.""" data = request.json

test\_photo = 'data/pics/recommendation\_pics/' + data['file\_name']

file\_num = 2035

style\_df = pd.DataFrame()

style\_df = pd.DataFrame(columns = ['face\_shape','hair\_length','location','filename','score']) hair\_length\_input = 'Updo'

updo\_input = data['person\_see\_up\_dos'] if updo\_input in ['n','no','N','No','NO']:

hair\_length\_input = data['person\_hair\_length'] if hair\_length\_input in ['short','Short','s','S']:

hair\_length\_input = 'Short'

if hair\_length\_input in ['long','longer','l','L']: hair\_length\_input = 'Long'

make\_face\_df\_save(test\_photo,file\_num,df) face\_shape = find\_face\_shape(df,file\_num) process\_rec\_pics(style\_df)

img\_filename = run\_recommender\_face\_shape(face\_shape[0],style\_df,hair\_length\_input) return jsonify({'Face Shape': face\_shape[0], 'img\_filename': img\_filename})

@app.route('/predict\_user\_face\_shape', methods=['GET', 'POST']) def predict\_user\_face\_shape():

"""Return a user face shape.""" data = request.json

test\_photo = 'data/pics/recommendation\_pics/' + data['file\_name'] file\_num = 2035

make\_face\_df\_save(test\_photo,file\_num,df) face\_shape = find\_face\_shape(df,file\_num) return jsonify({'face\_shape': face\_shape[0]})

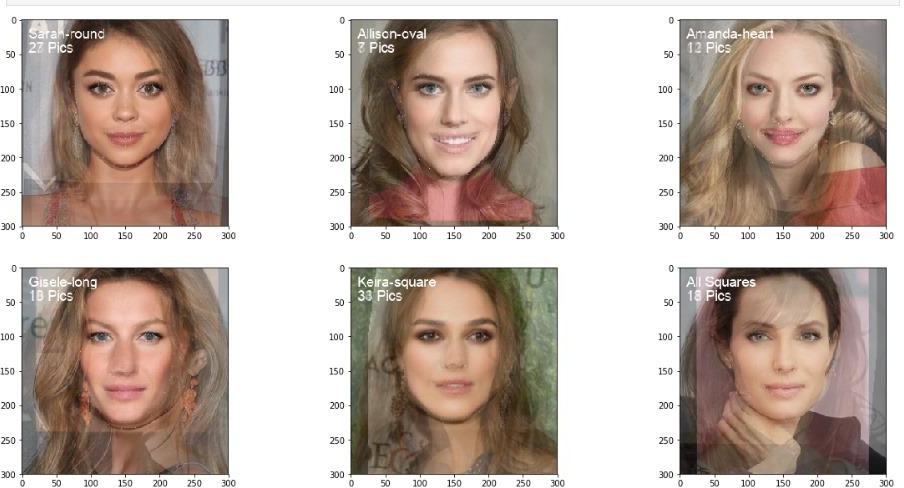
@app.route('/output/<img\_filename>') def output\_image(img\_filename):

"""Send the output image."""

with open(f"output/{img\_filename}", 'rb') as f: img\_data = f.read()

response = make\_response(img\_data) response.headers['Content-Type'] = 'image/png' return response

1. Sample Input



1. Sample Output

