CHAPTER 1

INTRODUCTION

Effective online product recommendation has become one of the major requirements in the consumer industry now a day. Its application ranges from online stores like Amazon, Flipkart, snap deal etc to video sharing sites like YouTube to Social networks sites such as Face Book, Google+ etc. Recommending right product to the right user at the right time will certainly impact the buying characteristics of the user. Technologies for communication have traditionally been developed based on the senses that play a major role in human interaction.

Python is used for developing this innovative project and it is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. It is a high-level, interpreted, interactive and object-oriented scripting language. It is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. Programs and other executable files can be in many directories, so operating systems provide a search path that lists the directories that the Operating System searches for executable.

Setting path is stored in an environment variable, which is a named string maintained by the operating system. This variable contains programs. To add the Python directory to the path for a particular session in Windows, At the command prompt − type path %path%;C:\Python and press Enter.

Python is Interpreted − Python is processed at runtime by the interpreter. It does not need to compile your program before executing it. This is similar to Practical Extraction and Reporting Language and Personal Home Page. Python is Interactive − can actually sit at a Python prompt and interact with the interpreter programs. It is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects. Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple. Python's feature includes like easy learning, it has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

Easy-to-read; Python code is more clearly defined and visible to the eyes, Easy-to-maintain where Python's source code is fairly easy-to-maintain. A broad standard library has its bulk of the library is very portable and cross-platform compatible on Windows, and Macintosh.

It has Interactive Mode by supporting for an interactive mode which allows interactive testing and debugging of snippets of code. It is Portable for running on a wide variety of hardware platforms and has the same interface on all platforms.

It is more Extendable by adding low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient. It provides the Databases Facilities with more interfaces to all major commercial databases.

Graphical User Interface Programming supports Graphical User Interface applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix. It is more Scalable which provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features; few are listed as by supporting its functional and structured programming methods as well as Object Oriented Programming. It can be used as a scripting language or can be compiled to byte-code for building large applications. It provides very high-level dynamic data types and supports dynamic type checking. It supports automatic garbage collection. It can be easily integrated with C, C++, Common Operating Machine, ActiveX, Consolidated Omnibus Budget Reconciliation Act, and Java. Python is available on a wide variety of platforms including Linux and Mac Operating System X. Let's understand how to set up our Python environment.

Python distribution is available for a wide variety of platforms. Initially needed to download only the binary code applicable for platform and install Python. If the binary code for the platform is not available, It needs a C compiler to compile the source code manually. Compiling the source code offers more flexibility in terms of choice of features for installation.

The use of such technologies that recognize voice and language, there are artificial intelligence robots that can interact closely with real life, in such ways as managing the daily Schedules of people and playing their favourite music. However, sensory acceptance is required for interactions more precisely mirroring those of humans. Therefore, the most necessary technology is a vision sensor, as vision is a large part of human perception in most interactions. In artificial intelligence robots using interactions between a human and a machine, human faces provide important information as a clue to understand the current state of the user.

Therefore, the field of facial expression recognition has been studied extensively over the last ten years. Recently, with the increment of relevant data and continued development of deep learning, a facial expression recognition system which accurately recognizes facial expressions in various environments has come to be actively studied. Facial expression recognition systems FERs are fundamentally based on an ergonomic and evolutionary approach. Based on universality, similarity, physiological, and evolutionary properties, emotions in FER studies can be classified into six categories: happiness, sadness, fear, disgust, surprise, and anger. In addition, emotions can be classified into seven categories with the addition of a neutral emotion. Previous methods on FER can be categorized into two groups: detecting facial actions that are related to a specific expression or making classification based on the extracted image features. In Facial Action Coding System (FACS), expressions are encoded by Action Units (AUs), which refer to some tiny but discriminable facial muscle changes. Thus, researchers usually convert FER problem to the task of AU detection. Some other methods represent expressions by some hand-crafted patterns or features, which can be utilized to train an expression classifier. However, some intractable problems are inevitable for these methods. For example, when encoding expressions by AUs, it‟s hard for researchers to accurately detect every AU in an image as facial muscle moves are sometimes difficult to be tracked. It is also hard to design a type of feature that can adapt to different environments. Variations such as illuminations and image rotations can weaken the representative capacity of hand-crafted features. In recent years, Convolutional Neural Network (CNN) has achieved great success in the field of computer vision.

It can adaptively adjust its convolutional kernels of each layer to obtain some desired features, which makes it adapt to various classification problems without much prior knowledge. In general, the output of CNN, i.e. the feature maps of the last layer, will be treated as the high-level semantic concept to represent the input. In FER problem, previous studies have revealed that expressional changes usually occur on some salient facial regions such as neighbourhood of mouth, eyes and nose. This implies that details of local facial regions can be discriminative for expressional recognition. However, most current CNN-based methods on FER only extract features from the whole expressional image. These methods emphasize the internality of a facial expression but ignore the information of local details. In the other words, typical CNN can‟t take full advantage of recognition-effective information encoded in Age Recognition is used to find the user‟s age from their facial image. After obtaining the feature vector of face image age, the age estimation function can be learned and the corresponding age estimator can be trained. When it compares two face ages, it is easy for humans to identify which is older, but to accurately guess the age of the face is not so easy. When inferring a person‟s exact age, it may compare the input face with the faces of many people whose ages are known, resulting in a series of comparisons, and then estimate the person‟s age by integrating the results. This process involves numerous pairwise preferences, each of which is obtained by comparing the input face to the faces in the dataset. Based on this, it proposes a divide-and- rule age estimation learning scheme.

Ethnicity recognition is added as the enhancement for the product recommendation. Such features are extracted using various approaches such as geometrical feature, holistic feature, local feature, and fusion features. Chan and Bledsoe 1965 analysed the facial features of the White by using the distance and ratio of facial geometrical features. According to geometrical relationship of eyes, mouth, and under jaw, Kanade 1977 matched face images in a dataset constructed by him. Brunelli and Poggio 1993 measured face similarity using facial geometrical features, which include nose length, mouth width, and under jaw shape, and the results indicated that geometrical features could be used to identify ethnical groups quite well. Brooks and Gwinn 2010 analysed the differences between the White and Black using the skin colour. According to their proposed skin colour model, Gwinn extracted the facial features from Asian and European. Akbari and Mozaffari Mar. 2012 explored the relations of facial skin colour using south Indian, Australian, and African. Anzures, Pascalis, Quinn, Slater, and Lee 2011 confirmed that skin color was very sensitive to illumination, so that the skin color was usually fused in combined features to classify people primarily. Since Turk and Pentland 1991 proposed principal component analysis PCA in facial feature analysis including eyes, nose, and mouth successfully, PCA has been a popular method in face recognition. Based on PCA, Levine 1996 conducted facial feature extraction between Burman and non-Burman. Awwad, Ahmad, and Salameh 2013 accomplished facial features analysis for Arabian, Asian, and Caucasian. Based on scale, illumination and pose, Yan and Zhang (2009) used PCA to analyze the facial features on CMU and UCSD databases. Recently, many deep neural network methods are also used for face analysis and recognition (Chen, Zhang, Dong, Le, & Rao, 2017; Luan et al., 2018; Trigeorgis, Snape, Kokkinos, & Zafeiriou, 2017; Zhang, Song, & Qi, 2017). Srinivas et al. 2017focused on predicting ethnicity using a convolutional neural network (CNN) with the Wild East Asian Face Dataset.

Local features can reduce the influence of illumination and obstacle occlusion, which are usually performing better than holistic features. For example, wavelet and local binary pattern (LBP) had shown their effectiveness on FERET database Kumar, Berg, Belhumeur, & Nayar, 2011; Salah, Du, & Al-Jawad, 2013. In addition, Fu, Yang, and Hou 2011 analyzed facial expression using embedded topographic independent component analysis (TICA), the results showed the advantages of local features. However, the combined features which usually include skin coloUr features, local wavelets features, and holistic features were used in practice instead of a single type of facial features. Ding, Huang, Wang, and Chen 2013 described face representations using texture and geometrical shape. Previously, It also combined several different geometric features to represent ethnical groups, such as length, angular, and proportion features Li et al., 2017. Also the semantic descriptions for ethnical groups were constructed based on Axiomatic Fuzzy Set (AFS) theory, and the manifolds of ethnical groups were learned in our recent study Duan, Li, Wang, Zhang, & Liu, 2016; Wang, Duan, Liu, Wang, & Li, 2016.

SR has been intensively used in the field of face recognition, expression analysis, age estimation, and facial image super-resolution Dian, Fang, & Li, 2017. Wright, Yang, Ganesh, Sastry, and Ma 2009 proposed sparse representation based classification (ESRC) approach and brought the SR into face recognition. It assumed that a face image could be viewed as a sparse linear representation of other face images for the same person. Aharon, Elad, and Bruckstein 2006 applied the ESRC approach directly for occluded facial expression recognition. The performance is not as good as expected due to the fact that the identity information of human face is more obvious than that of expression, which implies that the features of identity would affect facial expression recognition severely. Recently, the SR has been extended to some recognition tasks with small sample size. Mairal, Leordeanu, Bach, Hebert, and Ponce 2008 proposed the extended ESRC approach, which has refined SR by adding general learning in the framework of ESRC. This method improved the performance for small sample size face recognition problem and single sample based face recognition problem by unitizing the information extracted from other datasets. Yang, Zhang, Yang and Zhang 2010 proposed the sparse variation dictionary learning SVDL approach, in which one could obtain the projection matrix according to a training set. The SVDL was then embedded in ESRC to conduct face recognition. However, SVDL needs plenty of training data which contained all type of images for each class to learn an effective dictionary. Yang, Zhang, Yang, and Niu 2007 proposed sparse illumination learning and transfer (SILT) approach. This approach could match a few targets for obtaining information of face images with different illuminations. The methods mentioned above can improve face recognition performance to different extent, and also have significant achievement in solving small sample size problem in face recognition. This aims to use the SR approach to solve the ethnical group recognition with extracted regional features via data mining. Furthermore automatic zooming is achieved through the frontal cortex dislocation and eye movement.

CHAPTER 2

LITERATURE SURVEY

1. Multi-Objective Based Spatio-Temporal Feature Representation Learning Robust to Expression Intensity Variations for Facial Expression Recognition

Facial expression recognition (FER) is increasingly gaining importance in various emerging affective computing applications. In practice, achieving accurate FER is challenging due to the large amount of inter-personal variations such as expression intensity variations. It proposes a new spatio-temporal feature representation learning for FER that is robust to expression intensity variations. The proposed method utilizes representative expression states (e.g., onset, apex and offset of expressions) which can be specified in facial sequences regardless of the expression intensity. The characteristics of facial Expressions are encoded in two parts. As the first part, spatial image characteristics of the representative expression-state frames are learned via a convolutional neural network. Five objective terms are proposed to improve the expression class separability of the spatial feature representation. In the second part, temporal characteristics of the spatial feature representation in the first part are learned with a long short-term memory of the facial expression. Comprehensive experiments have been conducted on a deliberate expression dataset (MMI) and a spontaneous micro-expression dataset (CASME II). Experimental results showed that the proposed method achieved higher recognition rates in both datasets compared to the state-of-the-art methods.

2. Facial expression recognition with Convolutional Neural Networks Coping with few data and the training sample order.

Facial expression recognition has been an active research area in the past 10 years, with growing application areas including avatar animation, neuro marketing and sociable robots. The recognition of facial expressions is not an easy problem for machine learning methods, since people can vary significantly in the way they show their expressions. Even images of the same person in the same facial expression can vary in brightness, background and pose, and these variations are emphasized if considering different subjects (because of variations in shape, ethnicity among others). Although facial expression recognition is studied in the literature, few works perform fair evaluation avoiding mixing subjects while training and testing the proposed algorithms. Hence, facial expression recognition is still a challenging problem in computer vision. In this work, it proposes a simple solution for facial expression recognition that uses a combination of Convolutional Neural Network and specific image pre-processing steps. CNN achieve better accuracy with big data. However, there are no publicly available datasets with sufficient data for facial expression recognition with deep architectures. Therefore, to tackle the problem, It apply some pre-processing techniques to extract only expression specific features from a face image and explore the presentation order of the samples during training. The experiments employed to evaluate our technique were carried out using three largely used public databases (CKþ, JAFFE and BU-3DFE). A study of the impact of each image pre-processing operation in the accuracy rate is presented. The proposed method: achieves competitive results when compared with other facial expression recognition methods – 96.76% of accuracy in the CKþ database – it is fast to train, and it allows for real time facial expression recognition with standard computers.

3. Facial Expression Recognition Based on Deep Evolutional Spatial-Temporal Networks.

One key challenging issue of facial expression recognition is to capture the dynamic variation of facial physical structure from videos. It proposes a Part-based Hierarchical Bidirectional Recurrent Neural Network to analyse the facial expression information of temporal sequences. Our PHRNN models facial morphological variations and dynamical evolution of expressions, which is effective to extract "temporal features" based on facial landmarks from consecutive frames. Meanwhile, In order to complement the still appearance information, a Multi-Signal Convolutional Neural Network is proposed to extract “spatial features” from still frames. It uses both recognition and verification signals as supervision to calculate different loss functions, which are helpful to increase the variations of different expressions and reduce the differences among identical expressions. This deep Evolutional Spatial-Temporal Networks extract the partial whole, geometry appearance and dynamic-still information, effectively boosting the performance of facial expression recognition. Experimental results show that this method largely outperforms the state-of-the-art ones. On three widely used facial expression databases (CK+, Oulu-CASIA and MMI), our method reduces the error rates of the previous best ones by 45.5%, 25.8% and 24.4%, respectively.

4. Local Directional Ternary Pattern for Facial Expression Recognition

Face descriptor, local directional ternary pattern, for facial expression recognition. LDTP efficiently encodes information of emotion-related features (i.e., eyes, eyebrows, upper nose, and mouth) by using the directional information and ternary pattern in order to take advantage of the robustness of edge patterns in the edge region while overcoming weaknesses of edge-based methods in smooth regions. Our proposal, unlike existing histogram-based face description methods that divide the face into several regions and sample the codes uniformly, uses a two level grid to construct the face descriptor while sampling expression-related information at different scales. It use a coarse grid for stable codes (highly related to non-expression), and a finer one for active codes (highly related to expression). This multi-level approach enables us to do a finer grain description of facial motions, while still characterizing the coarse features of the expression. Moreover, it learns the active LDTP codes from the emotion related facial regions. It tested our method by using person dependent and independent cross-validation schemes to evaluate the performance. This shows improvement of overall accuracy of facial expression recognition on six datasets.

5. Object Detection with Discriminatively Trained Part-Based Models.

It describes an object detection system based on mixtures of multi scale deformable part models. Our system is able to represent highly variable object classes and achieves state-of-the-art results in the PASCAL object detection challenges. While deformable part models have become quite popular, their value had not been demonstrated on difficult benchmarks such as the PASCAL data sets. Our system relies on new methods for discriminative training with partially labelled data. It combine a margin sensitive approach for data-mining hard negative examples with a formalism It call latent SVM. A latent SVM is a reformulation of MI-SVM in terms of latent variables. A latent SVM is semi convex, and the training problem becomes convex once latent information is specified for the positive examples. This leads to an iterative training algorithm that alternates between fixing latent values for positive examples and optimizing the latent SVM objective function.

6. A new algorithm for age recognition from facial images.

` A new algorithm for age-group recognition from frontal face image is presented. The algorithm classifies subjects into four different age categories in four key stages: Pre-processing, facial feature extraction by a novel geometric feature-based method, face feature analysis, and age classification. In order to apply the algorithm to the problem, a face image database focusing on people's age information is required. Because there were no such databases, It created a database for this purpose, which is called Iranian face database. IFDB contains digital images of people from 1 to 85 years of age. After pre-processing, then primary features of the faces in the database will be accurately detected. Finally, a neural network is used to classify the face into age groups using computed facial feature ratios and wrinkle densities. Experimental results show that the algorithm identifies the age group with accuracy of 86.64%.

7. Ethnicity- and Gender-based Subject Retrieval Using 3-D Face-Recognition Techniques.

While the retrieval of datasets from human subjects based on demographic characteristics race is ability with wide-ranging application, it remains poorly-studied. In contrast, a large body of work exists in the field of biometrics which has a different goal: the recognition of human subjects. Due to this disparity of interest, existing methods for retrieval based on demographic attributes tend to lag behind the more well studied algorithms designed purely for face matching. The question this raises is whether a face recognition system could be leveraged to solve these other problems and, if so, how effective it could be. In the current work, It explore the limits of such a system for gender and ethnicity identification a ground truth of demographically labeled, textureless 3-D models of human faces and a state-of-the-art face-recognition algorithm. Once trained, our system is capable of classifying the gender and ethnicity of any such model of interest. Experiments are conducted on 4007 facial meshes from the benchmark Face Recognition Grand Challenge v2 dataset.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Online product recommendation is done using facial image recognition and emotional feedback. It detects the faces of the users from the live camera stream along with gender identification and product recommendation algorithms for targeting products to the right user. It also uses an emotion detection technique for getting a feedback about the recommended product. The developed system improves user interaction and can be used in online shopping websites and other places where efficient product recommendation methods are required.

3.1.1 DISADVANTAGES

- Fails to recommend customized products
- Same account is being used by multiple users.
- Accuracy range is low
- Product recommendation mismatching occurs when multiple user sign in to same account
- Capturing image and processing takes more time.

3.2 PROPOSED SYSTEM

For online product recommendation using facial expression, age and ethnicity recognition by Fisher Faces algorithm further more Yoda algorithm is used for product recommendation. It identifies customer‟s facial image classifies the facial texture with age and race categories these thing are used for prescribing appropriate product. If the customer‟s texture matched with Indian dataset, products are recommended based on the Indian‟s Likely hood then it checks the facial expression of customer if their expression is likelihood about that product then such product will be added to the cart. Instead of manual zooming automatic zooming method is introduced for ease of inspecting products. It is carried by monitoring the frontal cortex landmarks and eye movement. When these attributes gets adjusted from normal range product image gets up scaled automatically.

3.2.1 ADVANTAGES

- Improve the accuracy range.
- Recommends the appropriate product from user‟s preference.
- Which impacts great boom in online shopping platform
- Acts more User friendly.

CHAPTER 4

SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS

• Processor	:	Intel processor 3.0 GHz
• RAM	:	4GB
• Hard disk	:	500 GB
• Compact Disk	:	650 Mb
• Keyboard	:	Standard PS/2 keyboard
• Mouse	:	PS/2 compatible Mouse
• Monitor	:	15 inch color monitor

SOFTWARE REQUIREMENTS

• Front End	:	Python
• Back End	:	MYSQL
• Operating System	:	Windows OS
• System type	:	32-bit or 64-bit Operating System
• IDE	:	Python 3.5.1

CHAPTER 5

MODULE DESCRIPTION

A module can be defined as part of program or a single independent unit of program. There are four types of modules namely.

- Face Recognition Module
- Age Recognition Module
- Ethnicity Recognition Module
- Facial Expression Recognition Module
- Auto Zooming Module

5.1 FACE RECOGNITION MODULE

Face detection is the first step in any face recognition system. Many face detection techniques have been proposed in the past decade. They can be classified into geometry-based face detectors and colour-based face detectors. Among the geometry-based face detectors, a method examines the triangle relationship between eye and mouth regions to identify the face region.

In addition to, the traditional eye detection methods can be simply and efficiently implemented for frontal face images but can be difficult for complex images.

Moreover, skin colour has been proven to be an effective image feature for face detection. In the pre-processing stage of the proposed system, a facial region based on skin colour detection is cropped from an input image. The obtained facial region is then resized into an 8×8 pixel image to make the face recognition system scale invariant. After then, histogram equalization is applied to enhance the image brightness and contrast. Feature extraction methods are based on the transforms such as the DWT, DCT and Sobel edge detection also commonly represent the face images with a large set of features.

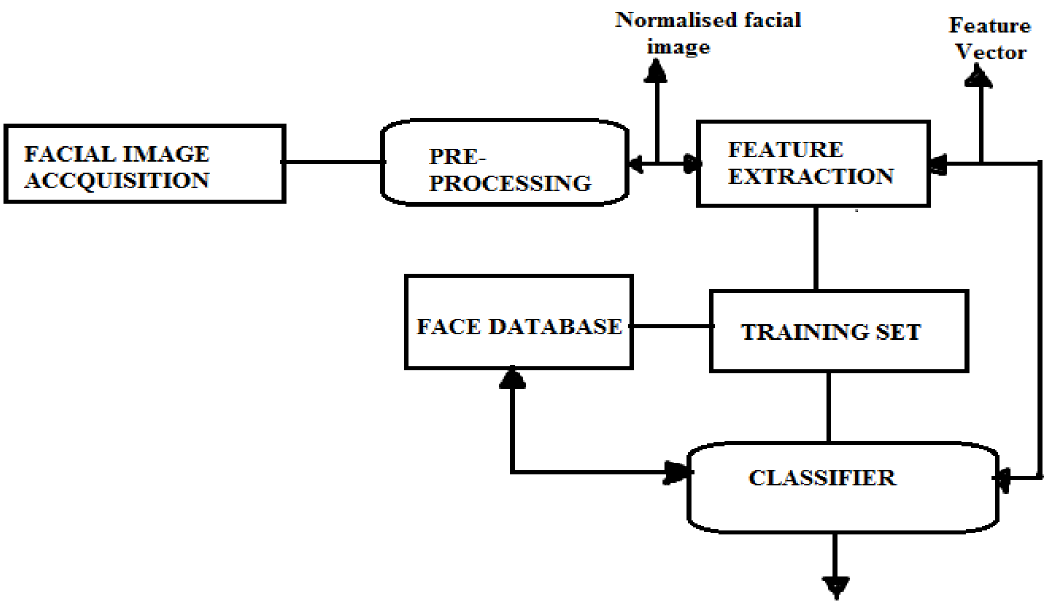


Figure 5.1 Facial Recognition

The features of every image stored in our data based are extracted and then stored into the feature vector. Once the feature vectors for all existing images are developed the new database consist of all feature vectors is formed and then stored inside our storage device. To retrieve all images that are similar to the target image, we must extract the features of the target image and compare it with all features vectors. DCT is a powerful transform to extract proper features for face recognition. After applying DCT to the entire face images, some of the coefficients are selected to construct feature vectors.

5.2 AGE RECOGNITION MODULE

The proposed age estimation algorithm realizes multiclass classification approach for each ages (from 1 to N) a binary classifier is constructed deciding whether a person on input image looks older than the given age or not. Input fragments are pre-processed to align their luminance characteristics and to transform them to uniform scale. Pre-processing includes colour space transformation and scaling, both similar to that of gender recognition algorithm. Additionally image normalization was performed by histogram equalization procedure. Transformation to LBP feature space and SVM training procedure are used for binary classifier construction. To predict direct age binary classifier outputs are statistically analysed and the most probable age becomes the algorithm output. To test age estimation algorithms performance standard metrics were calculated. There are conventional databases that are widely used in a field of age estimation from facial images. Each image in such data sets contains information about biological age of a person on this image. The most commonly used database is MORPH. It contains over 55 thousand facial images of more than 13000 different persons: men and women of various races, nationalities and ages.

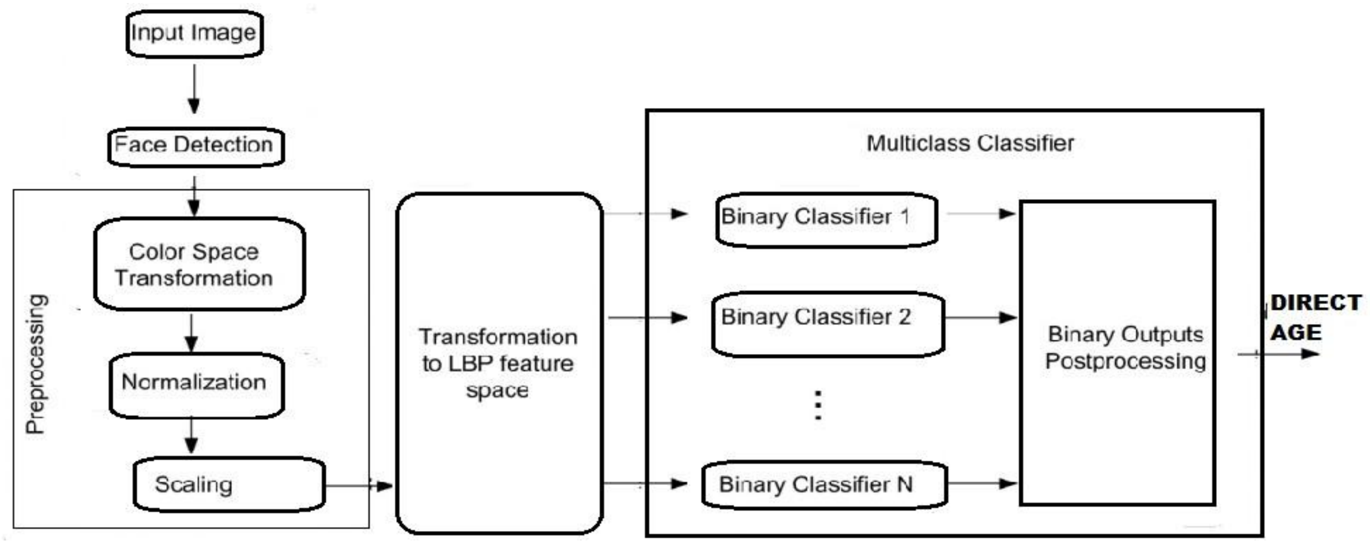


Figure 5.2 Age Recognition

5.3 ETHNICITY RECOGNITION MODULE

A „PCA+LDA‟ scheme is used to reduce the dimensionality of the input space; LDA is used to extract the discriminant projection directions by taking into account the class label information. Images at different scales provide different levels of information as the visual stimuli. Multi scale analysis is widely used in. Face images with different resolutions construct different manifolds in the input space of different dimensionalities. A classifier at each scale can provide confidence of the assigned class membership for each test face image. The final decision may be enhanced by integrating the confidence from different scales. A person identification system by combining outputs from classifiers based on visual cues. Although images with high resolutions tend to provide detailed information, their dimensionality is very high relative to the limited number of training samples, which makes it difficult to correctly estimate the statistical distribution. While the low resolution images may not contain enough clues for the individual identity recognition, it can still be used for other human trait identification, such as race and gender. It achieved an average accuracy rate of 92% for the ethnic classification part of the task. At each scale, a classic appearance-based face recognizer based on the LDA representation is developed under the Bayesian statistical decision framework. An ensemble is then constructed by integrating the classification results to arrive at the final decision. The product rule is used as the integration strategy, which results the appropriate ethnicity of the user.

5.4 FACIAL EXPRESSION RECOGNITION MODULE

The FER system includes the major stages such as face image pre-processing, feature extraction and classification. Pre-processing is a process which can be used to improve the performance of the FER system and it can be carried out before feature extraction process Image pre-processing includes different types of processes such as image clarity and scaling, contrast adjustment, and additional enhancement processes is to improve the expression frames. The cropping and scaling processes were performed on the face image in which the nose of the face is taken as midpoint and the other important facial components are included physically. Bessel down sampling is used for face image size reduction but it protects the aspects and also the perceptual worth of the original image. The Gaussian filter is used for resizing the input images which provides the smoothness to the images. Normalization is the pre-processing method which can be designed for reduction of illumination and variations of the face images with the median filter and to achieve an improved face image. The normalization method also used for the extraction of eye positions which make more robust to personality differences for the FER system and it provides more clarity to the input images. Localization is a pre-processing method and it uses the Viola-Jones algorithm to detect the facial images from the input image. Detection of size and location of the face images using Adaboost learning algorithm and haar like features. The localization is mainly used for spotting the size and locations of the face from the image. Face alignment is also the pre-processing method which can be performed by using the SIFT flow algorithm. For this, first calculate reference image for each face expressions. After that all the images are aligned through related reference images. ROI segmentation is one of the important types of pre-processing method which includes three important functions such as regulating the face dimensions by dividing the colour components and of face image, eye or forehead and mouth regions segmentation. In FER, ROI segmentation is most popular because for convenient segmentation of face organs from the face images. The histogram equalization method is used to conquer the illumination variations. This method is mainly used for enhancing the contrast of the face images and for exact lighting also used to improve the distinction between the intensities. In FER, more pre-processing methods are used but the ROI segmentation process is more suitable because it detects the face organs accurately which organs are is mainly used for expression recognition. Next the histogram equalization is also another one important pre-processing technique for FER because it improves the image distinction.

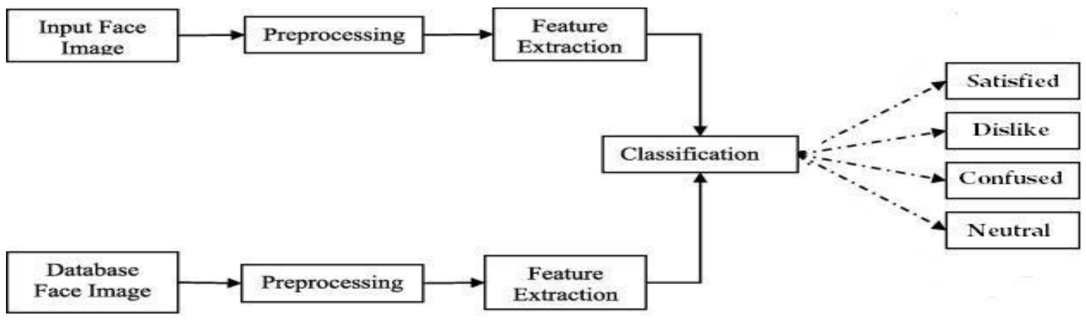


Figure 5.4 Facial Expression Recognition

5.5 AUTO ZOOMING

Instead of manual zooming, auto zooming is done by monitoring the user‟s frontal cortex region and eye movement if it is adjusted from the normal value the product image gets up scaled up to user‟s comfort view. When user blinks one time, it initially checks the user‟s normal blinking range from the live stream camera. If the blinking rate, Frontal cortex region, eye movements are adjusted from the normal rate it up scales the product automatically for the user‟s deeper inspection. By this method it resolves the irritation in viewing the product and facilitates user to feel more comfortable.

CHAPTER 6

SYSTEM DESIGN

6.1 ARCHITECHTURE DIAGRAM

Facial image is captured from the live stream camera. That image identifies the user‟s age and ethnicity for product recommendation. Using FER if the user gets satisfied from the product it gets added to the cart.

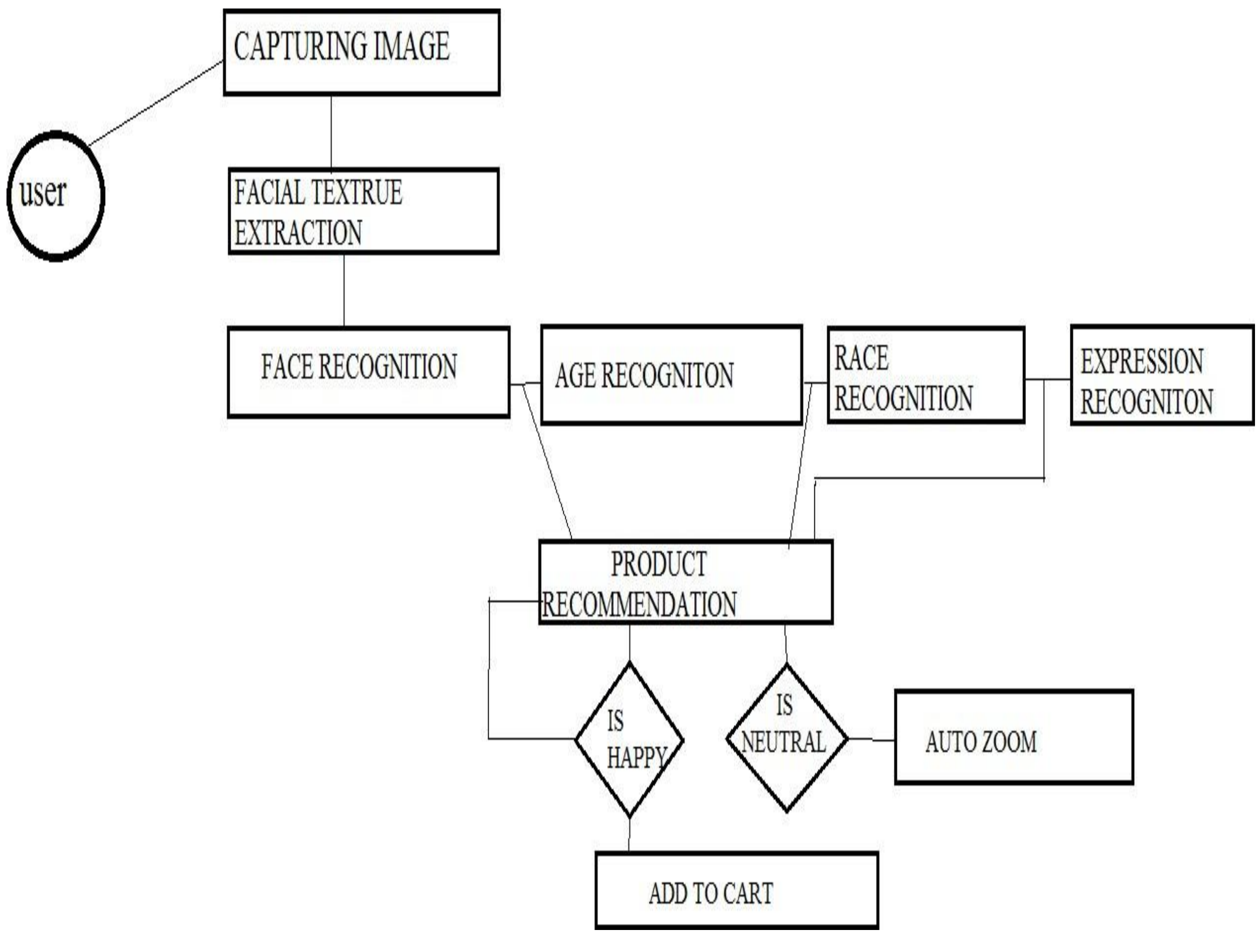


Figure 6.1 Architecture of Online Product Recommendation

6.2 USE CASE DIAGRAM

A use case is a list of steps, typically defining interactions between a role (known in Unified Modeling Language (UML) as an” actor”) and a system, to achieve a goal. The actor can be human, an external system, or time. In Systems engineering, uses cases are used at a higher level than without software engineering, often representing missions or stakeholder goals. The detailed requirements may then be captured in systems modelling language (SysML) or as contractual statements.

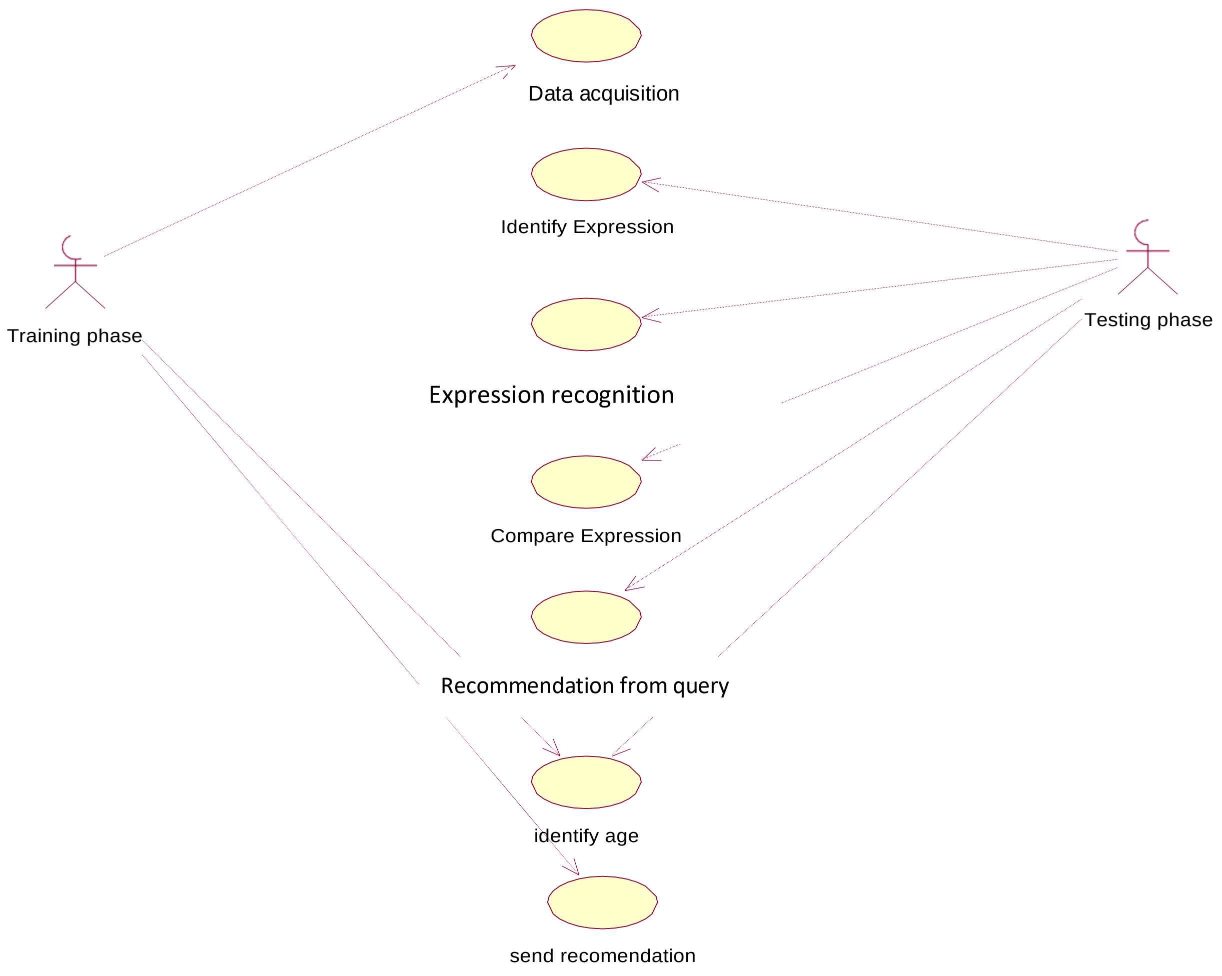


Figure 6.2 Use case Diagram

6.3 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanges between the objects needed to carry out the functionality of the scenario.

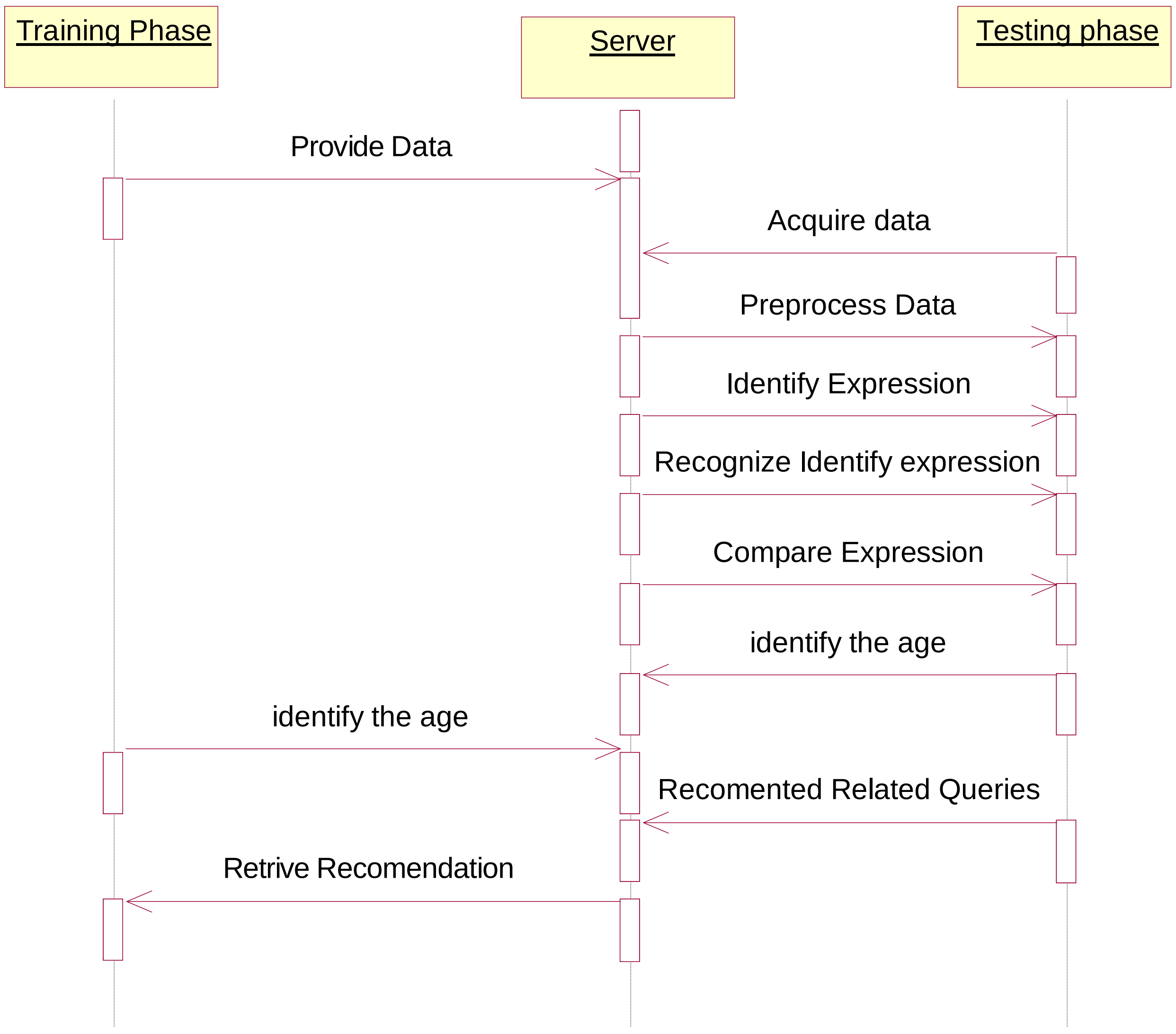


Figure 6.3 Sequence Diagram

6.4 COLLABORATION DIAGRAM

A collaboration diagram is a type of visual presentation that shows how various software objects interact with each other within an overall IT architecture and how users can benefit from this collaboration. A collaboration diagram often comes in the form of a visual chart resembles a flow chart.

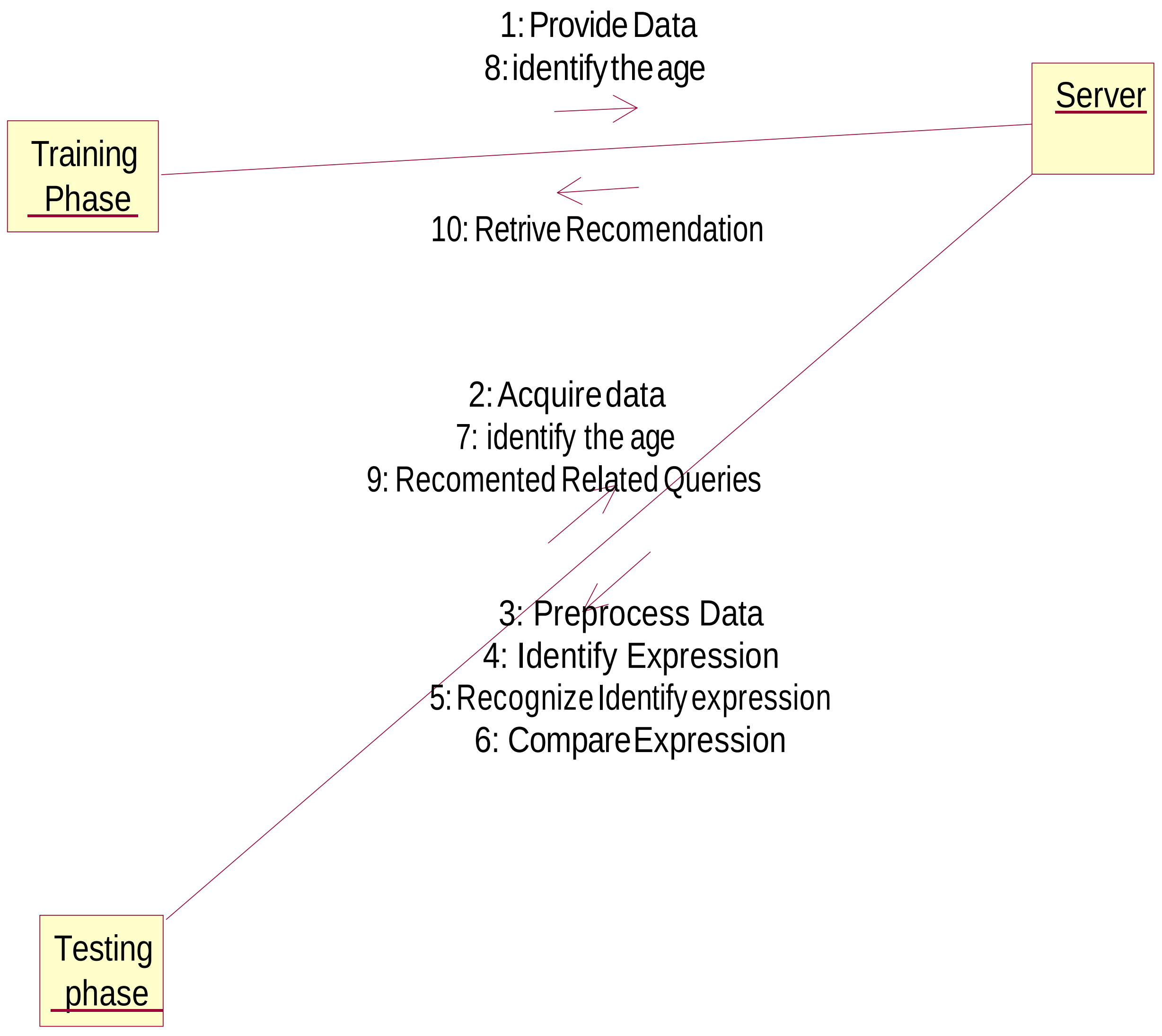


Figure 6.4 Collaboration Diagram

CHAPTER 7

TESTING

7.1 SYSTEM TESTING

System implementation is made up of many activities. The six major activities used in the project development are as follows. Coding is the process of whereby the physical design specification created by the analysis team turned into working computer code by the programming team.

Once the coding processed in parallel, as each module tested. Installation is the process during which the current system is replaced by the new system. This includes conversion of existing data, software, and documentation and work procedures to those consistent with the new system.

It is result from the installation process, user guides provides the information of how to use the system and its flow. Each learner gets personalized attention and could get their specific answer to the questions.

7.1 UNIT TESTING

Unit testing is a software development process in which the smallest testable parts of an application called units which are individually and manually but is often automated. Unit testing is a component of test driven development, a pragmatic methodology that takes a meticulous approach to building a product by means of continual testing and revision. Test-driven development requires that developers first write failing unit tests.

Then they write code and refractor the application until the test passes. TDD typically results in an explicit and predictable code base. It involves only those characteristics that are vital to the performance of the unit under test. This encourages developers to modify the source code without immediate concerns about how such changes might affect the functioning of other units or the program as a whole. Once all of the units in a program have been found to be working in the most efficient and error-free manner possible, larger components of the program can be evaluated by means of integration testing.

7.2 INTEGRATION TESTING

Integration testing also known as integration and testing the integrated module is a software development process which program units are combined and tested as groups in multiple ways. In this context, a unit is defined as the smallest testable part of an application. Integration testing can expose problems with the interfaces among program components before trouble occurs in real world program execution. Integration testing is a component of extreme programming, a pragmatic method software development that takes a meticulous approach to building a product by means of continual testing and revision.

There are two major ways of carrying out an integration testing called the bottom-up method and the top-down method. Bottom-up integration testing begins with unit testing, followed by tests of progressively higher level combinations of unit s called modules or builds. In top-down integration testing, the highest-level modules are tested first and progressively lower level modules are tested.

7.3 FUNCTIONAL TESTING

This type of testing ignores the internal parts and focuses only on the output to check if it is as per the requirement or not .It is Black-box type testing geared to the functional requirements of an application. Functional testing is more effective when the test conditions are created directly from user/business requirements. When test conditions are created from the system documentation (system requirements/ design documents), the defects in that documentation will not be detected through testing and this may be the cause of end-users‟ wrath when they finally use the software.

7.4 ACCEPTANCE TESTING

Accepting Testing is often used interchangeably with „Stress" and „load" testing. Performance Testing is done to check whether the system meets the performance requirements. Different performance and load tools are used to do this testing. It is a formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.

7.5 REGRESSION TESTING

Testing an application as a whole for the modification in any module or functionality is termed as Regression Testing. It is difficult to cover all the system in Regression Testing, so typically automation testing tools are used for these type of testing. This testing is done to make sure that new code changes should not have side effects on the existing functionalities. It ensures that the old code still works once the latest code changes are done.

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENTS

8.1 CONCLUSION

The current system can be improved by using advanced algorithms for facial recognition, age and ethnicity Recognition for product Recommendation based on user's facial images. Using FER dataset Recommendation of products is enhanced. In Order to enhance the size of products automatic zooming facility is offered using the frontal cortex region range from live stream camera.

8.2 FUTURE ENHANCEMENTS

The current system can be improved by using advanced algorithms for facial recognition, gender identification and product recommendation and emotion detection. With slight modifications the same system can be used in mobile applications, retail stores, super markets, shopping malls etc. This system can be used for developing advertisement boards that displays customized advertisements for different persons.

APPENDIX A

SAMPLE CODINGS

MAIN.PY

```
import os
from flask import Flask, render_template, Response, redirect, request, session, abort, url_for
from camera import VideoCamera
import cv2
import shutil
import datetime
import PIL.Image
from PIL import Image
import imagehash
import mysql.connector
from werkzeug.utils import secure_filename
mydb = mysql.connector.connect(
host="localhost",
  user="root",
  password="",
  charset="utf8",
  database="shopping_face”
)
app = Flask(__name__)
app.secret_key = 'abcdef'
```

```
UPLOAD_FOLDER = 'static/upload'
ALLOWED_EXTENSIONS = { 'png', 'jpg', 'jpeg', 'gif'}
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
@app.route('/')
def index():
    '''cursor = mydb.cursor()
    cursor.execute('SELECT * FROM student')
    data = cursor.fetchall()
    cutoff = 20
    for rows in data:
        m1="static/photo/"+rows[0]+".jpg"
        print(rows[0])
        i=1
        while i <= 2:
            m2="faces/f"+str(i)+".jpg"
            hash0 = imagehash.average_hash(Image.open(m1))
            hash1 = imagehash.average_hash(Image.open(m2))
            cc=hash0 - hash1
            print("s="+m2+" "+str(cc))
            i += 1'''
    return render_template('index.html')
def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
```

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    msg=""
    uname=""
    if request.method=='POST':
        uname=request.form['uname']
        pwd=request.form['pass']
        cursor = mydb.cursor()
        cursor.execute('SELECT * FROM register WHERE uname = %s AND pass =
%s', (uname, pwd))
        account = cursor.fetchone()
        if account:
            session['username'] = uname
            return redirect(url_for('home',uname=uname))
        else:
            # Account doesnt exist or username/password incorrect
            msg = 'Incorrect username/password!'
    return render_template('login.html',msg=msg,uname=uname)
@app.route('/login_admin', methods=['GET', 'POST'])
def login_admin():
    msg=""
    if request.method=='POST':
        uname=request.form['uname']
        pwd=request.form['pass']
        cursor = mydb.cursor()
        cursor.execute('SELECT * FROM admin WHERE username = %s AND
password = %s', (uname, pwd))
        account = cursor.fetchone()
        if account:
            #session['loggedin'] = True
            #session['username'] = account['username']
            # Redirect to home page
            return redirect(url_for('admin'))
        else:
            # Account doesnt exist or username/password incorrect
            msg = 'Incorrect username/password!'
    return render_template('login_admin.html',msg=msg)
@app.route('/admin', methods=['GET', 'POST'])
def admin():
    cursor = mydb.cursor()
    cursor.execute('SELECT * FROM register')
    data = cursor.fetchall()
    return render_template('admin.html',data=data)
@app.route('/home', methods=['GET', 'POST'])
def home():
    return render_template('home.html')
@app.route('/home2', methods=['GET', 'POST'])
def home2():
    if 'username' in session:
        uname = session['username']
    qr=""
    txt=""
    f2=open("mess.txt","r")
    ms=f2.read()
    cursor = mydb.cursor()
    if ms=="1":
        cursor.execute('select p.id,p.product,p.price,p.details,p.pimage from product
p,selected s where p.id=s.pid and s.uname=%s and s.status=1',(uname, ))
    elif ms=="2":
        cursor.execute('select p.id,p.product,p.price,p.details,p.pimage from product
p,selected s where p.id=s.pid and s.uname=%s and s.status=2',(uname, ))
    else:
        cursor.execute('SELECT * FROM product')
    data = cursor.fetchall()
    if request.method=='POST':
        txt=request.form['txt']
        cursor1 = mydb.cursor()
        likeString = "%" + txt + "%"
        cursor1.execute('SELECT * FROM product where product like %s',(likeString,
))
        data = cursor1.fetchall()


    return render_template('home2.html',data=data)
@app.route('/analyze', methods=['GET', 'POST'])
def analyze():
    txt=0
    f2=open("mess.txt","r")
    ms=f2.read()
    return render_template('analyze.html',txt=ms)
@app.route('/register', methods=['GET', 'POST'])
def register():
    #import student
    msg=""
    if request.method=='POST':
        name=request.form['name']
        mobile=request.form['mobile']
        email=request.form['email']
        address=request.form['address']
        uname=request.form['uname']
        cursor = mydb.cursor()
        sql = "INSERT INTO register(name,mobile,email,address,uname,pass)
VALUES (%s, %s, %s, %s, %s, %s)"
        val = (name,mobile,email,address,uname,pass1)
        cursor.execute(sql, val)
        mydb.commit()
        print(cursor.rowcount, "Registered Success")
        result="sucess"
        if cursor.rowcount==1:
            return redirect(url_for('login'))
        else:
            msg='Already Exist'
    return render_template('register.html',msg=msg)
@app.route('/capture', methods=['GET', 'POST'])
def capture():
    act=""
    if 'username' in session:
        uname = session['username']
    print(uname)
    if request.method=='GET':
        act = request.args.get('act')
    if request.method=='POST':
        act="yes"
        ftype=request.form['ftype']
        print(ftype)
        ss=uname+ftype+".jpg"
        path="static/photo/"+ss
        shutil.copy('faces/f1.jpg', path)
        return redirect(url_for('capture',act=act,uname=uname))
    return render_template('capture.html',act=act, uname=uname)
@app.route('/product', methods=['GET', 'POST'])
def product():
    cursor = mydb.cursor()
    cursor.execute('SELECT * FROM product')
    data = cursor.fetchall()
```

```
if request.method=='GET':
        act  =  request.args.get('act')
        did = request.args.get('did')
        if act=="del":
            cursor = mydb.cursor()
            cursor.execute('SELECT * FROM product WHERE id = %s', (did, ))
            drow = cursor.fetchone()
            fn=drow[3]
            #os.remove("static/upload/"+fn)
            cursor1 = mydb.cursor()
            cursor1.execute('delete FROM product WHERE id = %s', (did, ))
            mydb.commit()
            return redirect(url_for('product'))
    return render_template('product.html',data=data)
@app.route('/view', methods=['GET', 'POST'])
def view():
    act=""
    pid=""
    if 'username' in session:
        uname = session['username']
    if request.method=='GET':
        act  =  request.args.get('act')
        pid = request.args.get('pid')
        cursor1 = mydb.cursor()
        cursor1.execute('SELECT count(*) FROM selected WHERE uname=%s &&
pid = %s', (uname, pid))

        num = cursor1.fetchone()[0]

        if num==0:

            #print("s")

            mycursor = mydb.cursor()

            mycursor.execute("SELECT max(id)+1 FROM selected")

            maxid = mycursor.fetchone()[0]

            if maxid is None:

                maxid=1

sql = "INSERT INTO selected(id, uname, pid, status) VALUES (%s, %s, %s, %s)"

            val = (maxid, uname, pid, '0')

            mycursor.execute(sql,val)

            mydb.commit()

        else:

            f2=open("mess.txt","r")

            ms=f2.read()

            if ms!="":

                cursor2 = mydb.cursor()

                cursor2.execute('update selected set status=%s WHERE uname=%s and pid
= %s', (ms, uname, pid))

                mydb.commit()

        cursor = mydb.cursor()

        cursor.execute('SELECT * FROM product WHERE id = %s', (pid, ))

        data = cursor.fetchall()
        if act=="zoom":
            return redirect(url_for('view',act=act, pid=pid))
    return render_template('view.html',act=act, pid=pid, data=data)
@app.route('/view2', methods=['GET', 'POST'])
def view2():
    act="1"
    pid=""
    zo=""
    ss=0
    if 'username' in session:
        uname = session['username']
    if request.method=='GET':
        act  =  request.args.get('act')
        pid = request.args.get('pid')
        if act is None:
            act="1"
        ss=int(act)+1
        if ss==5:
            act="1"
        else:
            act=str(ss)
         cursor1 = mydb.cursor()
        cursor1.execute('SELECT count(*) FROM selected WHERE uname=%s &&
pid = %s', (uname, pid))
        num = cursor1.fetchone()[0]
        if num==0:
            #print("s")
            mycursor = mydb.cursor()
            mycursor.execute("SELECT max(id)+1 FROM selected")
            maxid = mycursor.fetchone()[0]
            if maxid is None:
                maxid=1
              sql = "INSERT INTO selected(id, uname, pid, status) VALUES (%s, %s, 
%s, %s)"
            val = (maxid, uname, pid, '0')
            mycursor.execute(sql,val)
            mydb.commit()
        else:
            f2=open("mess.txt","r")
            ms=f2.read()
            if ms!="":
                cursor2 = mydb.cursor()
                cursor2.execute('update selected set status=%s WHERE uname=%s and pid 
= %s', (ms, uname, pid))
                mydb.commit()
        cursor = mydb.cursor()
        cursor.execute('SELECT * FROM product WHERE id = %s', (pid, ))
        data = cursor.fetchall()
    return render_template('view2.html',act=act, pid=pid, data=data)
@app.route('/cart', methods=['GET', 'POST'])
```

```
def cart():
    act=""
    pid=""
    did=""
    if 'username' in session:
        uname = session['username']
    cursor = mydb.cursor()
    cursor.execute('SELECT c.id,p.product,p.price,p.details,p.pimage,c.rdate FROM
cart c,product p where c.pid=p.id and c.uname=%s', (uname, ))
    data = cursor.fetchall()
    if request.method=='GET':
        act = request.args.get('act')
        pid = request.args.get('pid')
        did = request.args.get('did')
        if act=="ok":
            mycursor = mydb.cursor()
            mycursor.execute("SELECT max(id)+1 FROM cart")
            maxid = mycursor.fetchone()[0]
            if maxid is None:
                maxid=1
            now = datetime.datetime.now()
            rdate=now.strftime("%d-%m-%Y")
             sql = "INSERT INTO cart(id, uname, pid, rdate) VALUES (%s, %s, %s,
%s)"
            val = (maxid, uname, pid, rdate)
            mycursor.execute(sql,val)
            mydb.commit()
            return redirect(url_for('cart',data=data))
        if act=="del":
            cursor = mydb.cursor()
            cursor.execute('delete FROM cart WHERE id = %s', (did, ))
            mydb.commit()
            return redirect(url_for('cart',data=data))

    return render_template('cart.html', data=data)
@app.route('/report', methods=['GET', 'POST'])
def report():
    act=""
    pid=""
    did=""
    if 'username' in session:
        uname = session['username']
    cursor = mydb.cursor()
    cursor.execute('SELECT c.id,p.product,p.price,p.details,p.pimage,c.rdate,c.uname
FROM cart c,product p where c.pid=p.id order by c.id desc')
    data = cursor.fetchall()
    return render_template('report.html', data=data)
@app.route('/add_product', methods=['GET', 'POST'])
def add_product():
    msg=""
    if request.method=='POST':
        product=request.form['product']
        price=request.form['price']
        details=request.form['details']
        file = request.files['file']
        mycursor = mydb.cursor()
        mycursor.execute("SELECT max(id)+1 FROM product")
        maxid = mycursor.fetchone()[0]
        if maxid is None:
            maxid=1
        imgname="P"+str(maxid)+file.filename
        if 'file' not in request.files:
            flash('No file part')
            return redirect(request.url)
        if file.filename == '':
            flash('No selected file')
            return redirect(request.url)
        if file and allowed_file(file.filename):
            filename = secure_filename(imgname)
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            sql = "INSERT INTO product(id, product, price, details, pimage) VALUES
  (%s, %s, %s, %s, %s)"
            val = (maxid, product, price, details, imgname)
            mycursor.execute(sql,val)
            mydb.commit()
            msg="Product Added success"
    return render_template('add_product.html',msg=msg)
@app.route('/logout', methods=['GET', 'POST'])
def logout():
    return render_template('login.html')
@app.route('/monitor', methods=['GET', 'POST'])
def monitor():
    if 'username' in session:
        uname = session['username']
    return render_template('monitor.html')
def gen(camera,uname):
    print(uname)
    m1="static/photo/"+uname+"1"+".jpg"
    m2="static/photo/"+uname+"2"+".jpg"
    m3="static/photo/"+uname+"3"+".jpg"
    a=0
    b=0
    c=0
    st=0
    cutoff=15
    while True:
        frame = camera.get_frame()
        hash0 = imagehash.average_hash(Image.open(m1))
        hash1 = imagehash.average_hash(Image.open("faces/f1.jpg"))
```

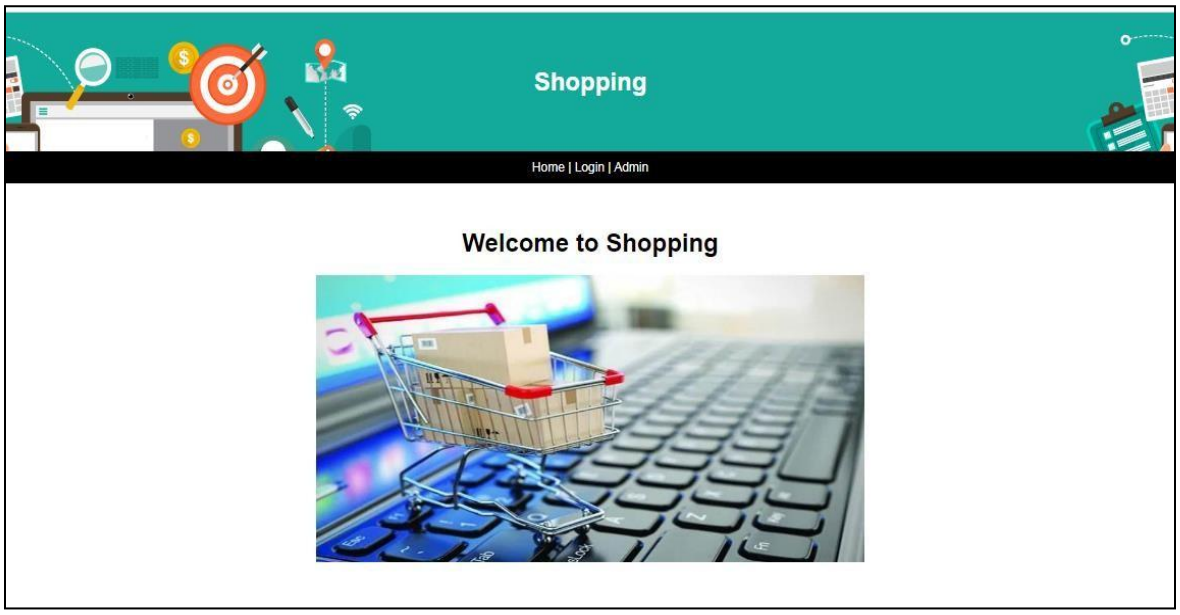
```
cc=hash0 - hash1
if hash0 - hash1 <= cutoff:
    a=hash0 - hash1
hash20 = imagehash.average_hash(Image.open(m2))
hash21 = imagehash.average_hash(Image.open("faces/f1.jpg"))
cc=hash20 - hash21
if hash20 - hash21 <= cutoff:
    b=hash20 - hash21

hash30 = imagehash.average_hash(Image.open(m3))
hash31 = imagehash.average_hash(Image.open("faces/f1.jpg"))
cc=hash30 - hash31
if hash30 - hash31 <= cutoff:
    c=hash30 - hash31
if a<b and a<c:
    st=1
elif b<c:
    st=2
else:
    st=0

ff=open("mess.txt","w")
ff.write(str(st))
ff.close()
```

```
        print("abc="+str(a)+" "+str(b)+" "+str(c))
        print("st="+str(st))
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')
@app.route('/video_feed')
def video_feed():
    if 'username' in session:
        uname = session['username']
    return Response(gen(VideoCamera(),uname),
                    mimetype='multipart/x-mixed-replace; boundary=frame')
if_name____== '__main__':
    app.run(host='0.0.0.0', debug=True)
```

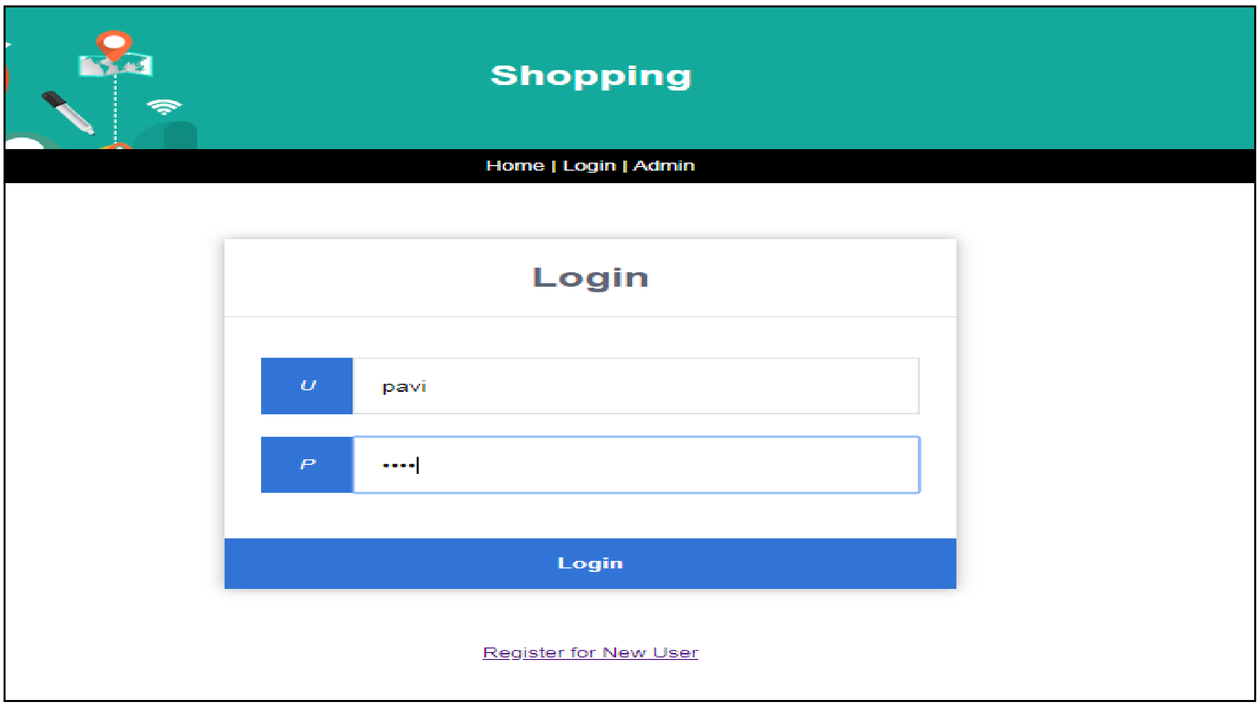
APPENDIX B

SCREENSHOTS

Main Webpage

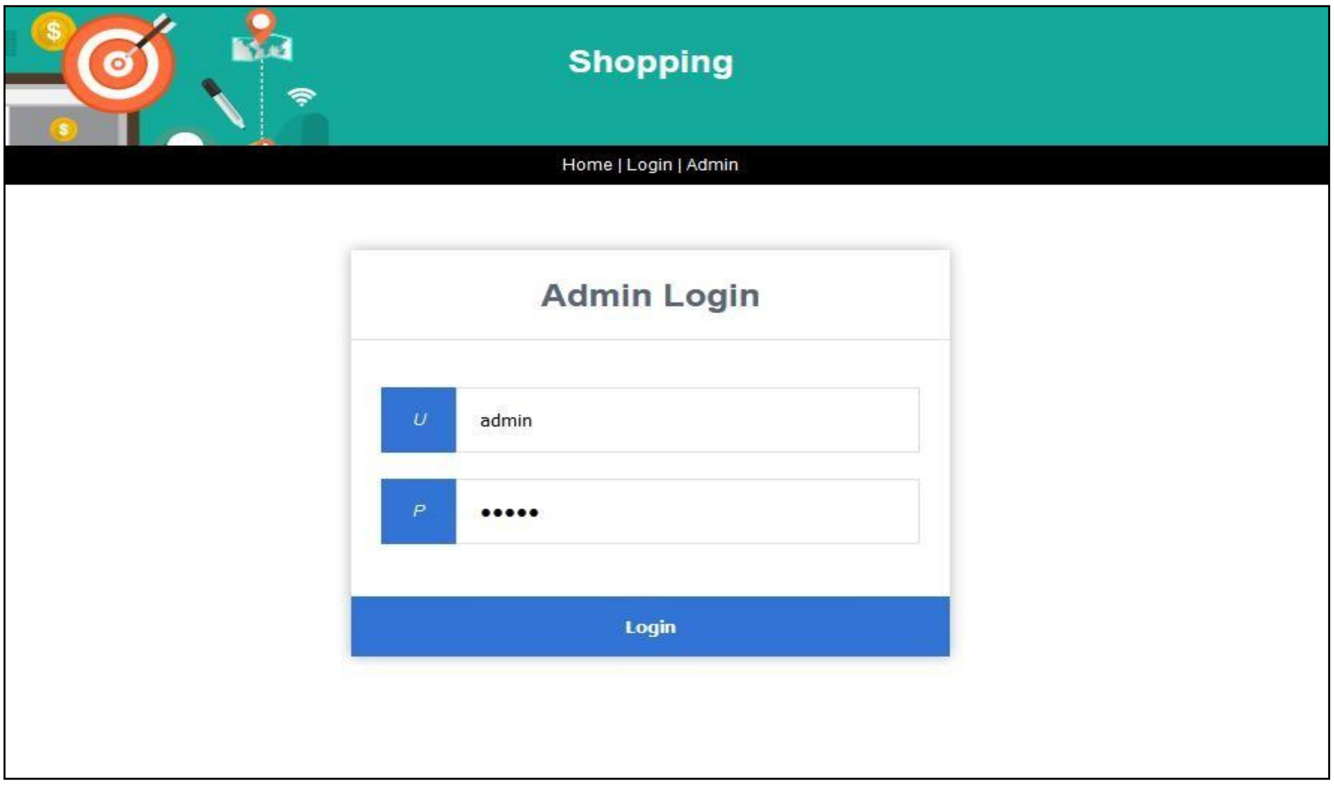


User Login



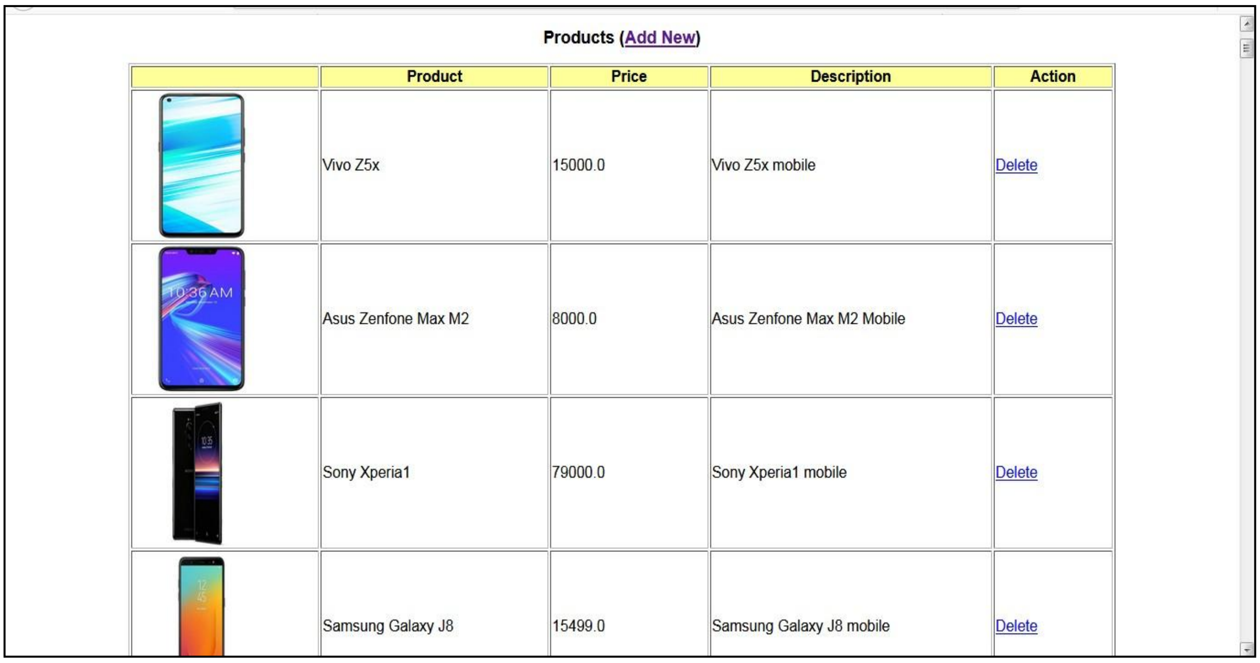
pavi

Admin Login



admin
•••••

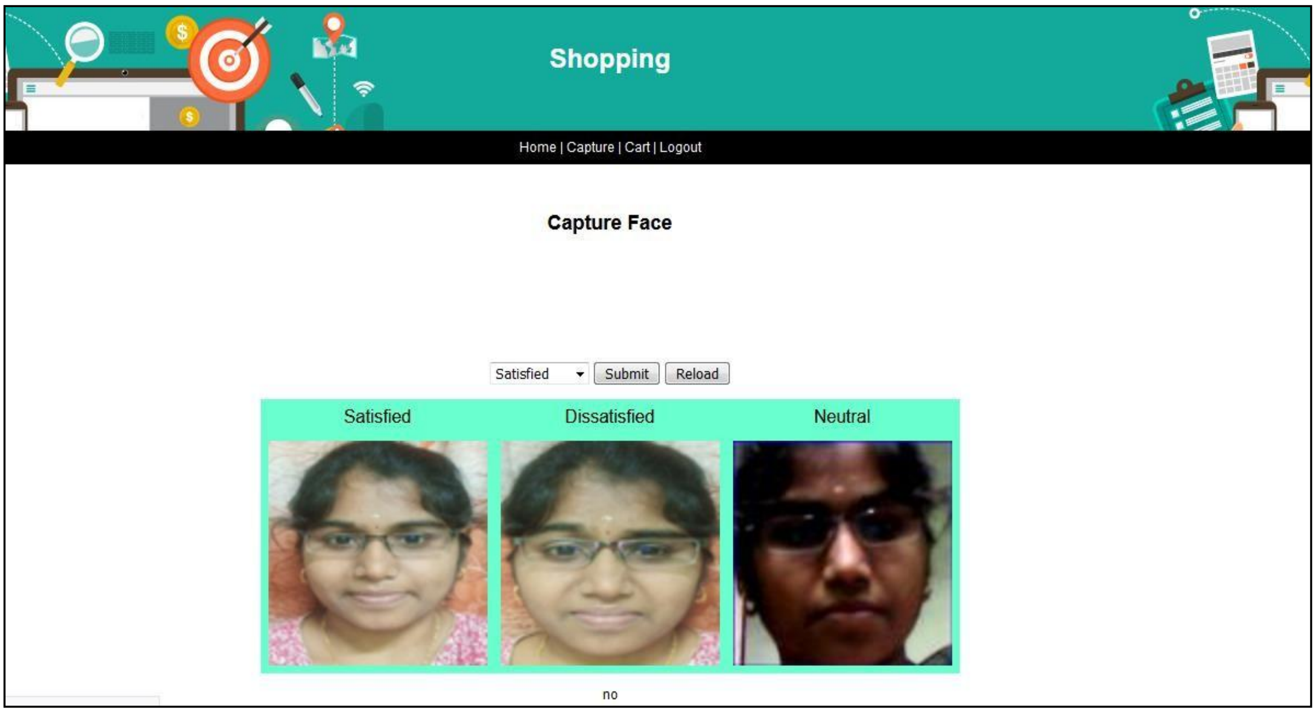
Products for Recommendation



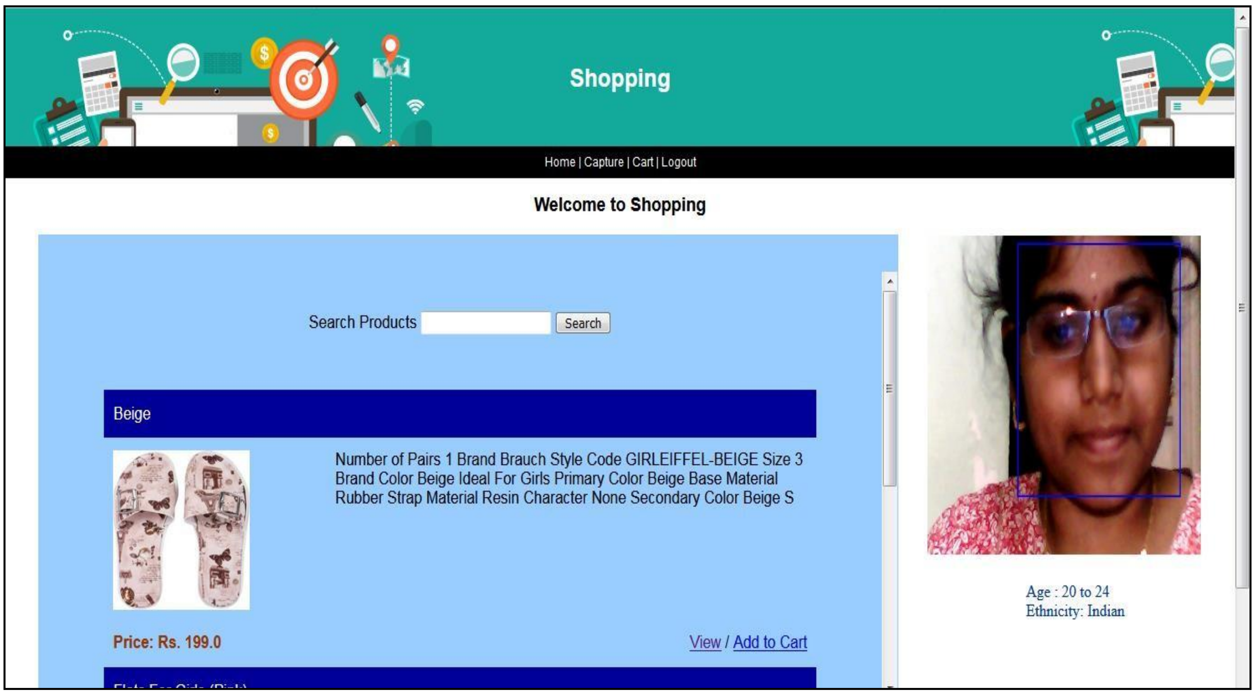
Products (Add New)

	Product	Price	Description	Action
	Vivo Z5x	15000.0	Vivo Z5x mobile	Delete
	Asus Zenfone Max M2	8000.0	Asus Zenfone Max M2 Mobile	Delete
	Sony Xperia1	79000.0	Sony Xperia1 mobile	Delete
	Samsung Galaxy J8	15499.0	Samsung Galaxy J8 mobile	Delete

Data Training



Facial expression, Age and ethnicity Recognition

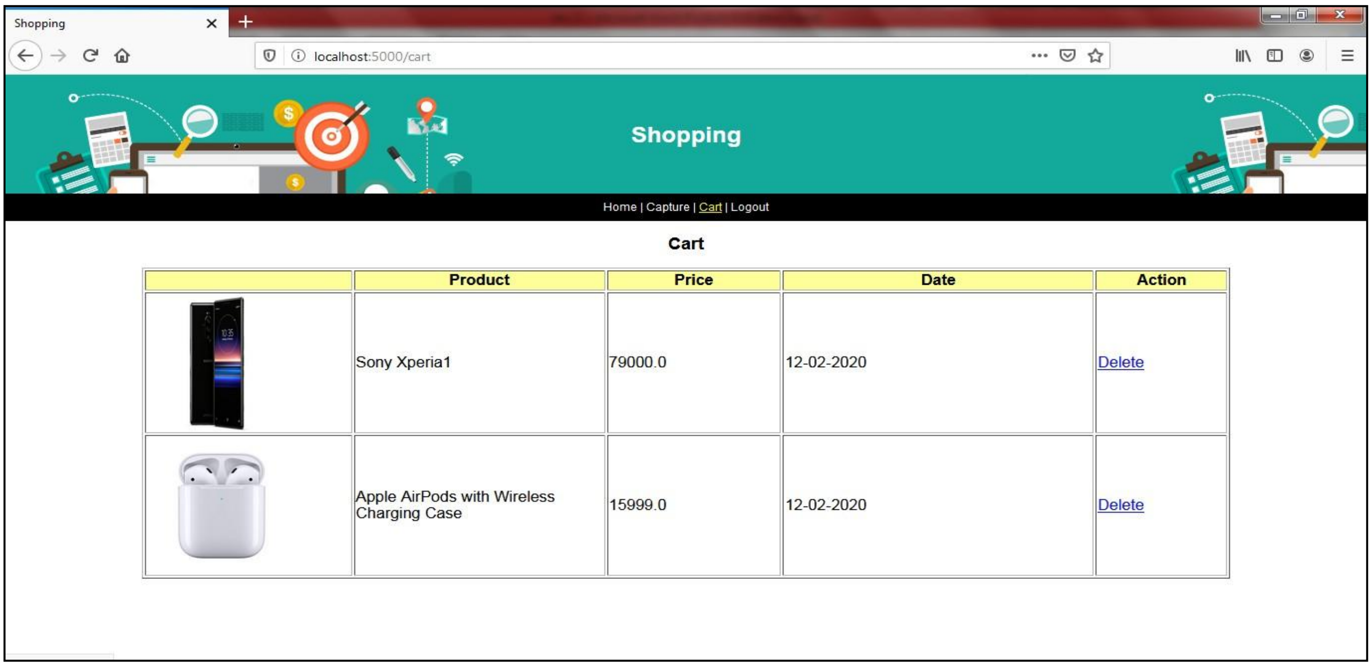


Number of Pairs 1 Brand Brauch Style Code GIRLEIFFEL-BEIGE Size 3 Brand Color Beige Ideal For Girls Primary Color Beige Base Material Rubber Strap Material Resin Character None Secondary Color Beige S

Price: Rs. 199.0

Age : 20 to 24
Ethnicity: Indian

Satisfied Products added to the Cart



Cart

	Product	Price	Date	Action
	Sony Xperia1	79000.0	12-02-2020	Delete
	Apple AirPods with Wireless Charging Case	15999.0	12-02-2020	Delete

REFERENCES

[1] Viola, P, Jones, M: "Rapid object detection using a boosted cascade of simple features ". In Proceedings of 200i IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 511- 518, 2001.

[2] Freund, Y., Schapire, RE.: "A short introduction to boosting". In J Jap. Soci. Artif. intel!. 14(5), pp.771- 780, 1999.

[3] Leinhart, R, Maydt, "An extended set of Haar-like features", In Proceedings of International Conference on image Processing, pp. 900- 903, 2002.

[4] Gong, Songjie. "A collaborative filtering recommendation algorithm based on user clustering and item clustering." In Journal of Software 5, no. 7 (2010), pp.745-752, 2010.

[5] Ma, Hao, Haixuan Yang, Michael R Lyu, and Irwin King. "Sorec: social recommendation using probabilistic matrix factorization." In Proceedings of 17th ACM conference on Information and knowledge management, pp. 931-940, ACM, 2008.

[6] Dae Hoe Kim, Wissam J. Baddar, Jinhyeok Jang, and Yong Man Ro , Senior Member, IEEE" Multi-Objective Based Spatio-TemporalFeature Representation Learning Robust toExpression Intensity Variations for Facial Expression Recognition" IEEE Transactions On Affective Computing, Vol. 10, no. 2, April-June 2019.

[7] S. H. Lee, K. Plataniotis, N. Konstantinos, and Y. M. Ro, “Intraclass variation reduction using training expression images for sparse representation based facial expression recognition,” IEEE Transactions On Affective Computing., Vol. 5, no. 3, pp. 340–351, Jul.-Sep. 2014.

[8] Xie and H. Hu,” Facial expression recognition using hierarchical features with deep comprehensive multipatches aggregation convolutional neural networks” vol. 21, no. 1, pp. 211220, Jan. 2019.

[9] A.T. Lopes, E. de Aguiar, A. F. de Souza, and T. Oliveira-Santos, “Facialexpression recognition with convolutional neural networks: Coping withfew data and the training sample order” Pattern Recognit., vol. 61,pp. 610628, Jan. 2017

[10] Xin Geng, Zhi-Hua ZhouKate, Smith-Miles ” Automatic Age Estimation Based on Facial Aging Patterns” vol. 29 , Issue: 12 , Dec. 2007