**CHAPTER 1**

**INTRODUCTION**

**1.1 Introduction to the Project**

Accurate shoe sizing remains a persistent issue in the online footwear industry, often leading to customer dissatisfaction, increased product returns, and financial losses for retailers. Traditional sizing charts and manual measurement methods are prone to human error and lack standardization across different brands and regions. While some existing systems attempt to bridge this gap through size conversion tools or general recommendations, they often fail to provide precise, personalized solutions.

This project introduces a comprehensive, image-based foot size prediction system integrated into a cross-platform mobile application built with React Native. Unlike existing solutions that rely on pre-defined shoe databases or require user-entered foot lengths, this system leverages real-time image processing and machine learning techniques to analyze foot dimensions directly from photographs. Users interact with the application through the Expo Go platform, where they can easily register, upload a foot image, select gender and preferred shopping platform, and instantly receive a recommended footwear link.

The backend, developed using Flask and Python, uses OpenCV and KMeans clustering to isolate the foot from the background, detect edges, and calculate key measurements such as foot length and width. Based on these metrics, the system estimates the user's foot size in centimeters. A CSV-based mapping for each e-commerce platform (Amazon, Flipkart, Zappos) is used to retrieve the most appropriate product URL for the user’s size and gender.

The project not only ensures greater accuracy in size prediction but also offers a complete, real-time solution for personalized footwear shopping. By eliminating guesswork and manual input, it significantly improves user convenience and enhances the digital shopping experience. This innovation contributes to reducing return rates, building consumer trust, and setting a foundation for future applications in augmented reality fitting rooms and smart retail technologies.

**1.2 Introduction to (about 2-3 pages)**

In convolutional neural networks, we use three new types of layers, convolution layers, ReLU layers and pooling layers.

**Convolution layers:** Convolution layers consist of a number of filters (say ‘n’). The convolution layer takes an input image as is and performs 2D convolution operation on it with each of its ‘n’ filters, and returns ‘n’ output images known as feature maps. The filters replace neurons and the filter coefficients are just like weights in the sense that they are trainable.

**ReLU layers:** ReLU layer is just an activation function layer; it performs ReLU activation function on each pixel in the input image to return an output image of same size. ReLU is an abbreviation for rectified linear unit, it is the activation function defined by f(x) = x for x >= 0 and f(x) = 0 for x < 0.

**Pooling layers:** Pooling is an operation where we down sample the input image by combining every m×n group of adjacent pixels together into a single pixel. In the most common type of pooling, max pooling, the max value in every group is retained while the other values are discarded.

**1.3 unicodes**

Unicode is defined by Wikipedia as “a computing industry standard for the consistent encoding, representation, and handling of text expressed in most of the world's writing systems.” Unicode was developed when 8-bit encoding systems such as ASCII were still popular. Since ASCII could hold only 256 characters, only Roman characters were represented.

Many countries had developed their own versions of ASCII for their native languages. For example India developed ISCII. Alternatively, early Kannada writing software such as Baraha used customized ASCII fonts that merely rendered their own Kannada glyphs in place of the correct ASCII glyphs. While this solution is good for printing Kannada text on paper it is not suitable for applications such as transmitting Kannada text online or displaying Kannada text in web pages or on mobile devices. A universal encoding standard is needed. Unicode uses 16 bits (specifically UTF-16 uses 16 bits), which is way more than enough to represent characters in all of the world’s living languages, as well as historic scripts such as Brahmi.

UTF-16 assigns each of its characters with a unique 16-bit identification number known as a code point, and leaves the rendering of the character to the software. The code points for Kannada characters are in the range of 0x0C82 to 0x0CF2. This range of code points is reserved exclusively for Kannada characters, unlike in ISCII where the same character in different Indian languages is assigned the same code point.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Introduction**

**2.2 Literature Survey**

Rohit Kumar Gupta et al. (2022), [1] proposed a CNN-based model for detecting age and gender from facial images, including real-time CCTV footage. The model uses convolutional, pooling, and fully connected layers with a softmax classifier to categorize gender (male or female) and age into specific groups. Trained on a Kaggle dataset with varied lighting and poses, the model treats age prediction as a classification task, enhancing accuracy in uncontrolled environments. This approach serves as a strong reference for our project, where we adapt their architecture and strategy for broader surveillance tasks like person identification and attribute-based filtering.

Sivachandiran et al. (2022), [2] developed an automated deep learning model named ADCNN-AGC for classifying age and gender from facial images in surveillance systems. The model uses MTCNN for detecting faces, EfficientNet for feature extraction, and 1D-CNN for classification. Tested on the UTKFace dataset, the model achieved 95.29% accuracy for gender and a mean absolute error of 2.89 in age prediction. Compared to other recent models like GRA-Net and RAN, ADCNN-AGC demonstrated superior results in both efficiency and accuracy. This research provides a robust and scalable solution for real-time demographic analysis and is highly relevant for CCTV-based systems where facial attributes need to be extracted quickly and accurately under uncontrolled conditions.

Nikouei et al. (2018), [3] proposed a real-time human detection system for edge computing environments using a Lightweight Convolutional Neural Network (L-CNN). Designed with resource constraints in mind, the model employs depthwise separable convolutions and is based on the SSD (Single Shot Multibox Detector) architecture, enabling efficient detection of human figures in surveillance footage. Tested on a Raspberry Pi 3, the L-CNN demonstrated competitive performance, achieving an average speed of 1.79 frames per second (FPS) and a false positive rate of 6.6%, while using significantly less memory than other standard models such as SSD-GoogleNet. This model is particularly relevant to edge-based smart surveillance systems, offering a viable approach for efficient person detection under limited hardware. Its application as a frontend human detector makes it a practical reference for projects like ours that require low-latency, high-accuracy person attribute extraction from CCTV footage.

Guruh Fajar Shidik et al. (2019), [4] conducted a systematic literature review analyzing 220 journal publications on intelligent video surveillance systems from 2010 to 2019. The study categorizes research trends into three main areas: visual surveillance, intelligent surveillance integration, and system infrastructure design. It provides a detailed overview of machine learning techniques—especially deep learning, SVM, and fuzzy logic—used for surveillance tasks such as object detection, behavior analysis, and activity recognition. The review highlights key public datasets and evaluates five widely cited surveillance frameworks (e.g., SSF, RISE, and EDCAR), offering a valuable foundation for modular and scalable surveillance solutions. Although the paper lacks experimental validation, it serves as a rich knowledge base for developing advanced systems. For our CCTV-based person attribute extraction project, this review offers strategic insights into system design, suitable datasets, and integration frameworks, supporting the development of a robust, real-time surveillance solution focused on identifying multiple soft biometric traits.

Hitesh Panchal (2016), [5] The paper "CCTV Video Abstraction and Object Detection for Video Surveillance System" by Hitesh Panchal introduces an innovative algorithm for key frame extraction from CCTV footage, addressing the challenges of analyzing extensive video data. By employing video segmentation and automatic shot boundary detection, the algorithm efficiently summarizes video content, allowing for quick retrieval of relevant frames. This work highlights the importance of intelligent video management in surveillance systems, paving the way for further research in person attribute extraction. The findings emphasize the potential for enhancing video analysis techniques, which can be beneficial for developing advanced surveillance applications.

Joseph Redmon et al. (2018), [6] The paper "YOLOv3: An Incremental Improvement" by Joseph Redmon and Ali Farhadi presents significant enhancements to the YOLO object detection framework. The authors introduce a new classifier network that improves accuracy while maintaining high processing speed, achieving 28.2 mAP at 320x320 resolution. YOLOv3 employs multiscale predictions and a multilabel classification approach, allowing for effective detection of overlapping labels. This work highlights the advancements in real-time object detection, making it a crucial reference for projects focused on person attribute extraction from CCTV footage, where speed and accuracy are paramount for effective surveillance analysis.

Xiao Ke et al. (2020), [7] introduced a deep learning approach for extracting human attributes from surveillance images. It integrates SSD-based pose estimation and multi-feature fusion to effectively identify clothing attributes, addressing issues like pixel resolution and background interference. This approach is particularly relevant for CCTV attribute extraction systems, where accurate human region isolation is crucial for robust performance.

Prof. Nandhini N et al. (2019), [8] discussed a deep learning approach for identifying anomalies in surveillance footage. It leverages CNNs for feature extraction and anomaly detection, providing a robust framework for real-time analysis of high-dimensional data. This method can be adapted for person attribute extraction by focusing on specific human characteristics and movement patterns, making it a valuable reference for surveillance systems that require precise behavior monitoring.

Hiren Galiyawala et al. (2022), [9] presented a deep learning approach for person identification based on soft biometrics like age and clothing type. It uses Mask R-CNN for accurate person detection and attribute recognition, achieving high retrieval accuracy with fewer attributes. This method is highly relevant for CCTV-based person attribute extraction systems, providing a streamlined approach to identifying individuals in complex surveillance environments.

Fabbri et al. (2017), [10] proposed a tri-network approach (ResNet classifier + occlusion-resistant DCGAN + super-resolution DCGAN) for low-resolution surveillance video-based attribute classification, achieving a state-of-the-art benchmark on RAP under 80% occlusion/low resolution. It outperformed DeepMAR/ACN by >6% mAP by recovering classifiable features. The module-based architecture is stronger but limits real-time use.

Shoitan et al. (2023 ), [11] proposed a spatio-temporal person retrieval method in video surveillance using a combination of ByteTrack for robust tracking and two attribute recognition models—APR and ALM—to provide higher accuracy. Unlike conventional methods, their method relates the bounding boxes from frames to reduce detection errors and enhance attribute recognition. Evaluated on the SoftBioSearch dataset, the system achieved a 93.21% true positive, 14% better than state-of-the-art. While it performs well in occlusion and low visibility, its reliance on advanced tracking can be an issue for real-time applications. However, it addresses a significant loophole in attribute-based person retrieval.

Yaghoubi et al. (2020), [12] provided an exhaustive survey of Human Attribute Recognition (HAR), condensing state-of-the-art contributions in terms of most significant challenges such as data imbalance, occlusion, and attribute correlation. Different from previous surveys, they formulated a challenge-oriented taxonomy and critically examined deep learning methods, datasets, and measures, including sub-areas such as pedestrian and clothing attribute recognition. The survey recognizes gaps in the literature including the absence of integrated data, occlusion, and model explainability. It recognizes the use of CNNs, GCNs, and RNNs in filling the gaps, providing insightful information in the design of more trustworthy and interpretable HAR systems.

Amirgaliyev et al. (2025), [13] gave a comprehensive overview of ML and DL methods for person detection, tracking, identification, and face recognition, focusing on the shift from traditional features to deep CNNs like YOLO and FaceNet. Using the PRISMA method, they evaluated over 140 articles and encountered issues of occlusion, night vision, and ethical concerns. The research points to efficient, privacy-sensitive models and rich data sets as crucial, with directions for future research in smart surveillance systems suggested.

Haritha et al. (2025), [14] developed an AI-powered surveillance framework using YOLOv8 for object detection and LSTM for anomaly detection to promote public safety through automated crowd monitoring and prevention of crime. The system can operate in real-time using CCTV footage to monitor crowd density and spotting suspicious activity so that alerts can be given when an anomaly or over-crowding occurs. The authors report high detection results (95.4%) and anomaly detection recognition (92.7%) along with a 30% computation overhead; allowing the framework to be built on existing CCTV hardwares. The model uses contextual filtering and low latency processing to assist secure alterations, and scalability or operational effectiveness in workplace environments, public areas and industrial locations. While the framework showed a successful proof of concept, the system's ability to be optimized to other environments, along with the ambiguity of privacy implications will require future evaluation.

Amirgaliyev et al. (2025), [15] presented a systematic review of over 140 studies focused on machine learning and deep learning techniques for person detection, tracking, identification, and face recognition. They analyze classical approaches like HOG and Kalman filters alongside modern deep models such as YOLO, ArcFace, and DeepSORT. The paper highlights real-world applications in surveillance, transportation, and smart cities while addressing challenges such as occlusion, real-time constraints, and ethical concerns. This review serves as a valuable reference for developing intelligent video surveillance systems, especially for projects involving CCTV-based person attribute extraction using deep learning.

Iyshwarya Ratthi et al. (2024), [16] introduced an AI-based human height estimation model for surveillance, leveraging monocular cameras and YOLOv7 with a hybrid attention mechanism (HAM). Designed to aid in missing child retrieval, the system uses camera calibration and a new dataset (“Sense-Height”) featuring adults and children. Unlike traditional models, this approach handles occlusion, diverse lighting, and motion conditions with high accuracy (error as low as 0.02 cm). The paper provides strong empirical validation and proposes a field-of-view (FOV) zoning strategy. This work is significant for integrating height as a soft biometric in intelligent video surveillance.

Taha et al. (2024), [17] proposed a gait recognition model using IMU data instead of conventional video. Their system collects gait features from shoe-embedded sensors and processes them using stacked sparse autoencoders. The high-level features are then clustered to identify physical characteristics like age, gender, and body size. The model shows greater robustness to occlusion and environmental variation than visual gait recognition systems. While it is not directly usable for CCTV-based projects, its deep learning approach and gait-based soft biometric extraction provide a strong conceptual base for designing attribute recognition models using motion cues in surveillance footage.

Gururaj et al. (2024), [18] presented a detailed review of face recognition (FR) systems, covering traditional techniques like PCA and LDA, and advanced deep learning methods including CNN-based hybrid models. The paper explores FR challenges such as pose variation, occlusion, and aging, while classifying existing approaches into appearance-based, landmark-based, and hybrid methods. It also discusses video-based FR systems, dataset availability, and future directions. Although it does not introduce new models, this survey offers valuable insights into the selection of algorithms and datasets that can aid in developing accurate and real-time person attribute recognition from CCTV surveillance footage.

**2.3 Summary of Literature Survey**

Table 2.1 shows the summary of literature survey done.

**Table 2.1: Observations of Literature Survey**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Author Name** | **Title of Paper** | **Methodology used** | **Advantages** | **Future Work** |
| **Rohit Kumar Gupta, Shivaprasad M B, Dr. S. Srividhya** | **Age & Gender Detection using Convolutional Neural Network** |  | **Simple CNN architecture.**  **Good real-time potential.**  **Utilizes Keras, TensorFlow, OpenCV.**  **Classifies into defined age/gender classes** | **Improve accuracy under challenging conditions- Expand attributes beyond age/gender.**  **Real-time multi-person detection** |
|  |  |  |  |  |
|  |  |  |  |  |

**2.4 Comparison with Existing Systems**

System offers a significant advancement over traditional methods by integrating advanced image analysis and a user-friendly digital interface. Unlike existing systems that rely primarily on user-provided data such as footwear size, brand preferences, and historical purchases our approach leverages foot photo analysis to assess actual physical characteristics like foot length, width, shape, arch type, and potentially gait patterns. This shift from subjective input to objective measurement allows for far more accurate and personalized recommendations.

In contrast to conventional systems that often require users to manually select sizes or navigate style filters, our application simplifies the process. Users can either take a photo of their foot or upload an existing one, and the system will automatically detect and display the foot's length in centimeters using image processing techniques. This eliminates guesswork and ensures sizing precision. Following this, users are prompted to select their gender, which helps further refine the recommendation logic based on gender-specific sizing and fit parameters.

Another key differentiator is the app's integration with online shopping platforms. After foot analysis and gender selection, users are presented with curated options from selected e- commerce websites, allowing them to seamlessly proceed to purchase footwear that best suits their anatomical profile. This creates a unified experience from measurement to purchase. Privacy and data security are addressed through robust anonymization protocols and secure data handling practices, which are essential when dealing with biometric imagery. Unlike existing systems that analyze shopping history or collect potentially intrusive user input, our system focuses purely on physical foot attribute ensuring data relevance and minimal privacy concerns.

Finally, in a competitive market where many footwear recommendation tools rely on brand affiliations or customer reviews, our solution stands out by offering scientifically grounded, anatomically accurate recommendations. It not only improves user satisfaction through better fit and comfort but also redefines how consumers interact with footwear e-commerce by offering a tailored, intuitive, and efficient experience.

**2.5 Proposed System**

The proposed system for our footwear recommendation application leverages cutting-edge image processing and user-centric design to deliver a highly personalized, efficient, and secure user experience. The core component of the system is the Image Processing Module, which plays a crucial role in analyzing foot photos uploaded or captured by the user. This module will employ advanced image analysis techniques to extract precise measurements and characteristics such as foot length, width, shape, and arch type. These features are essential in determining the accurate footwear size and the ideal footwear fit, which varies based on individual foot structure. By ensuring accurate measurement and foot profiling, the system aims to significantly reduce the likelihood of sizing issues and improve overall customer satisfaction.

To make the process simple and accessible for users, the application will feature a user-friendly interface that guides them through each step. Users will be able to either take a photo of their foot in real time or upload one from their device gallery. After submitting the photo, they will be prompted to select their gender, which helps in narrowing down the footwear catalog, as sizes and styles often differ between male and female categories. Additionally, the user will be asked to choose their preferred e-commerce platform such as Amazon, Flipkart, or Zappos from which they would like to purchase footwears. Based on this selection, the system will recommend suitable products directly from the chosen platform, along with purchase links for a smooth and seamless shopping experience. Security and privacy are key priorities in our system. All uploaded photos and user data will be handled with strong encryption methods and securely stored to prevent unauthorized access. User consent will be a requirement for data usage, and data retention policies will comply with modern privacy standards. These measures are essential to protect sensitive biometric data and build user trust in the system.

To ensure scalability and performance, the backend system will be optimized to handle a high volume of user requests and data processing operations without delays. This ensures that the app remains responsive and efficient even during peak usage times. The system will also support cloud-based architecture to scale resources dynamically based on demand.

Finally, the application will incorporate a continuous feedback mechanism, where users can rate the footwear recommendations, leave reviews, and provide additional preferences. This feedback will be fed into a machine learning-based recommendation engine that continuously adapts and improves its suggestions based on user behavior and trends. Over time, this will enable the app to offer increasingly accurate and relevant recommendations, enhancing the overall user experience and keeping the system aligned with evolving user needs and preferences.

**2.6 Objectives**

* To develop a cross-platform mobile application using React Native that runs on Expo Go for foot size prediction.
* To enable users to capture or upload an image of their foot and receive accurate foot size estimations using image processing techniques and a clustering-based algorithm.
* To build a Flask-based backend server to process images, compute foot measurements, and communicate results back to the mobile client.
* To implement a secure user authentication system (registration, login, and profile editing) with PostgreSQL as the user database.
* To integrate the backend with CSV-based URL mapping for major platforms like Amazon, Flipkart, and Zappos, providing product links based on foot size, gender, and category.
* To ensure a user-friendly and responsive UI in the mobile app that allows seamless interaction with the prediction and recommendation system.
* To facilitate easy deployment and testing via Expo Go, allowing quick access and feedback during development and demonstration.
* To ensure the system is modular, scalable, and ready for future enhancements, such as local prediction or multi-brand filtering.

**CHAPTER 3**

**REQUIREMENT SPECIFICATION AND ANALYSIS**

**3.1 Introduction**

Our footwear recommendation project leverages sophisticated foot image analysis to accurately determine measurements such as foot length, width, and girth. These measurements are then matched with footwear specifications from popular platforms like Zappos, Amazon, and Flipkart to provide personalized recommendations. The application features an intuitive interface that enables easy photo uploads, gender selection, and direct purchasing, with a strong focus on user privacy, precision, and high performance.

**3.2 Functional Requirements**

* **Image Processing Module**

Develop a robust image processing module capable of analyzing foot photos to accurately extract key measurements. These include foot length, width at the ball and bridge, and overall girth. This module will serve as the foundation for precise footwear size detection and fit analysis.

* **Footwear Recommendation Algorithm**

Implement an intelligent matching algorithm that utilizes the extracted foot measurements along with the user’s selected gender to recommend suitable footwear. This algorithm will compare the user's measurements with a curated database of footwear specifications to ensure personalized and accurate suggestions.

* **E-Commerce Platform Integration**

Integrate the system with major online retail platforms such as Zappos, Amazon, and Flipkart. This will allow real-time fetching of relevant footwear options that match the user’s profile and preferences, enabling a seamless experience from measurement to purchase.

* **User Interface and Experience**

Design and deploy a user-friendly front-end interface using React and Expo Go, allowing users to effortlessly upload foot images, choose gender, and view recommended products. The interface will emphasize ease of use, speed, and responsiveness.

* **Back-End and Data Management**

Use Node.js to handle server-side logic and API communication, ensuring secure processing of data and efficient handling of user interactions. Implement PostgreSQL as the primary database to store user data, foot measurements, recommendation logs, and product listings from integrated platforms.

* **Cross-Platform Mobile Support**

Utilize Expo Go to enable quick testing and deployment of the mobile app on both Android and iOS platforms, ensuring broad accessibility and consistent performance across devices.

**3.3 User Interface Requirements**

* Users should be able to upload a foot photo either by taking a new picture or selecting one from their gallery.
* Users must select their gender to receive accurate and relevant footwear recommendations.
* The system should provide precise foot measurements such as length, width, and girth based on the uploaded image.
* Users should receive personalized footwear suggestions that match their foot dimensions and selected gender.
* Users must have the option to choose their preferred e-commerce platform like Amazon, Flipkart, or Zappos to view and purchase recommended footwears.
* The application should ensure secure handling of foot images and personal data, while offering smooth and consistent performance across mobile devices.

**3.4 Integration with Social Platforms**

* Integrate the application with popular social media platforms and footwear focused communities to utilize user-generated content for improving recommendation accuracy.
* Leverage social trends, reviews, and fashion insights from these platforms to enhance the personalization of footwear suggestions.
* Allow users to share their recommended footwears, feedback, or purchase experiences directly on social media to encourage engagement.
* Promote community interaction and brand visibility by fostering a social environment where users can explore, discuss, and support footwear choices.

**3.5 Software Requirements**

The project requires the following software to run:

**3.5.1 ReactJS**

React JS is used for developing the front-end interface of the application. It allows the creation of dynamic and reusable components, which enhance user interaction and responsiveness. React’s virtual DOM helps improve performance by minimizing direct manipulation of the actual DOM. This ensures a smooth and efficient user experience across devices.

**3.5.2 Expo Go**

Expo Go is a framework and platform for universal React applications. It enables developers to preview their React Native app instantly on a mobile device by scanning a QR code, eliminating the need to build the app every time. This significantly accelerates the testing and development cycle. Expo also provides various built-in tools and libraries to simplify app deployment.

**3.5.3 Node.js**

Node.js is used to manage the back-end operations of the app. It handles server- side logic, API requests, and real-time data processing efficiently. Being event-driven and non-blocking, Node.js supports scalability and fast response times. It ensures seamless communication between the client interface and the database.

**3.5.4 PostgreSQL**

PostgreSQL is the relational database used to store and manage structured data such as user profiles, foot measurements, and footwear product details. It supports complex queries and ensures data integrity and security. With its open-source nature and advanced performance features, it is ideal for handling large-scale data efficiently. It integrates smoothly with Node.js back-end frameworks.

**3.5.5 Visual Studio Code (VS Code)**

VS Code is the chosen integrated development environment (IDE) for writing and organizing the project’s source code. It provides intelligent code completion, syntax highlighting, debugging tools, and extensions tailored for JavaScript, React.

**3.5.6 npm (Node Package Manager)**

npm is used to install and manage the various dependencies and libraries required by the application. It ensures that packages like React, Express, and image processing tools are consistently installed and maintained across development environments. npm scripts can also automate tasks such as starting servers or building production versions of the app. This tool is essential for efficient project setup and management.

**3.5.7 Image Processing Libraries**

Image processing libraries like OpenCV.js are essential for analyzing foot images and extracting accurate measurements such as length, width, and girth. These libraries provide functions for image filtering, edge detection, scaling, and measurement calculations. They help automate the analysis process with high precision. Integration with the front-end ensures real-time feedback to the user based on the processed image.

**3.5.8 Python Programming language**

Python programming language is essential for our footwear recommendation project due to its versatility, rich ecosystem of libraries, and ease of integration with various technologies. Python's extensive libraries such as OpenCV for image processing, NumPy for numerical computations, and Flask for web development make it ideal for implementing the image analysis module and backend services. Its simplicity and readability enhance development efficiency, allowing rapid prototyping and iterative improvements.

**3.6 Hardware Requirements**

**3.6.1 Processor (CPU)**

The hardware requirements for our footwear recommendation project include a minimum Intel Core i5 or equivalent processor, with a recommended upgrade to an Intel Core i7 or higher for optimal performance in processing image analysis and recommendation algorithms.

**3.6.2 Memory (RAM)**

For memory (RAM), a minimum of 8 GB is sufficient, though 16 GB or more is recommended to manage large datasets and concurrent user requests effectively. Storage should ideally start at a minimum of 256 GB SSD (Solid State Drive) for faster data access, with a recommended upgrade to 512 GB SSD or higher to accommodate user data, footwear information, and system logs

**3.6.3 Network Connectivity**

Stable internet connection for accessing e-commerce APIs, deploying updates, and ensuring seamless user experience.

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 Introduction**

The system design phase plays a crucial role in laying out the structural framework of the Foot Size Prediction Mobile Application. It describes how different components within the system are organized and how they work together to achieve the overall functionality of the application. This phase outlines the logical flow of data, the interaction between the user interface, backend, and database, and the arrangement of modules responsible for image analysis, foot measurement, and footwear recommendation.

The design process also addresses critical aspects such as performance, modularity, and user experience, ensuring that each part of the system contributes effectively to the whole. A well-defined architecture helps maintain consistency, simplifies future upgrades, and ensures the application can handle increasing users or data volumes efficiently. The chapter presents the overall system structure using relevant diagrams like system architecture and flowcharts, which help visualize how tasks are distributed and coordinated. By following structured design patterns and emphasizing maintainability, this stage ensures that the final product is both robust and user-friendly, meeting the intended functional and technical goals.

**4.2 User Interface (Frontend – React Native)**

The user interface, developed using React Native, is the primary access point for users interacting with the mobile application. It provides a smooth and intuitive user experience with cross-platform compatibility, allowing the app to run efficiently on both Android and iOS devices. The UI handles essential functions like image capture via the device camera, selecting an image from the gallery, and accepting user inputs such as gender and e-commerce platform preferences. It also displays the processed prediction results, including foot size and footwear recommendations.

Additionally, the UI features forms for user registration and login, with validation checks to ensure accurate data entry. It seamlessly guides the user through the workflow from uploading a photo, choosing prediction mode (local or server), to viewing results and being redirected to the appropriate online store for purchasing footwears. The UI ensures accessibility and responsiveness, supporting multiple screen sizes and devices.

**4.3 Prediction Engine - Python**

The Prediction Engine is the computational core of the system, built using Python. It incorporates two major libraries OpenCV for image processing and scikit-learn for applying machine learning models. This engine analyzes uploaded foot images to determine measurements such as foot length, width at the ball and bridge, and overall girth. These values are then used to predict the correct footwear size based on established size charts and trained models.

There are two modes of operation for this engine:

* Local Mode: Implemented using Chaquopy, it allows Python code to run within the mobile app itself. This enables predictions to be made without internet access, ideal for users in low-connectivity regions.
* Remote Mode: In this setup, the image is uploaded to a remote server where a Flask API receives the image, processes it using the engine, and returns the predicted result to the app. This mode is particularly useful when more computational power or access to larger datasets is needed.

Both modes provide flexibility in deployment and enhance the app’s usability in different environments.

**4.4 Database - PostgreSQL**

A PostgreSQL database is used to manage and persist all essential data generated and required by the application. It stores:

* User credentials and authentication tokens
* Historical prediction data, allowing users to view past foot size estimations
* Optional profile data such as name, gender, usage preferences, and selected e-commerce platforms

PostgreSQL’s ability to handle complex queries and its support for ACID transactions make it a reliable choice for managing application data. The database also supports future analytics features such as tracking usage trends, common foot sizes in specific regions, or popular footwear models, which can further improve the recommendation engine.

The database communicates securely with both the frontend (in case of offline storage) and the backend (for online mode), ensuring a smooth flow of data and reliability in operations.

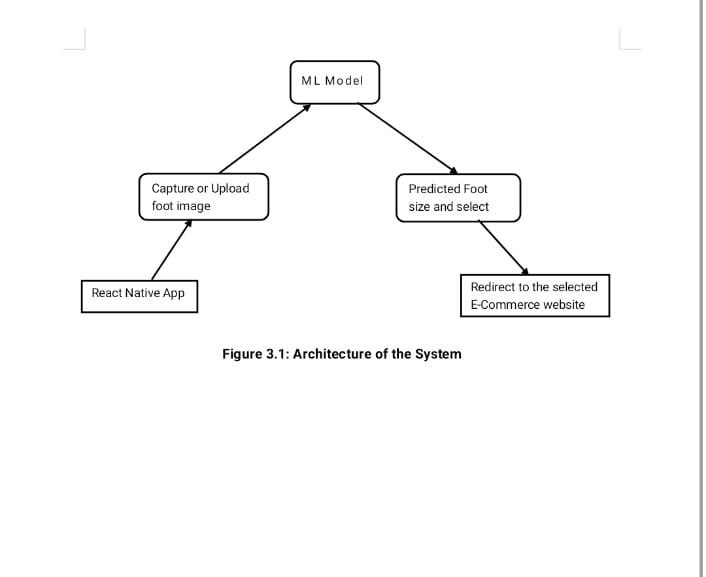
**4.5 Flask**

The backend, developed using Flask, acts as the bridge between the frontend app and the Python-based Prediction Engine in online mode. It provides RESTful API endpoints that handle user authentication (login/register), image upload, and the triggering of prediction tasks on the server.

When a user selects server mode, the image is sent to the Flask backend, where it is processed by the prediction engine. The result foot dimensions and recommended footwear size is then sent back to the app. The backend ensures that requests are handled securely and efficiently and can be scaled depending on the user load.

In addition to image processing, the backend also manages user sessions, stores intermediate data, logs prediction activities, and optionally integrates with analytics tools for future enhancements.

**4.6 System Architecture**



User Input Output

**Figure 4.1: System Architecture**

These are the steps as mentioned in the Figure 4.1:

**1. ML Model**

The ML model is responsible for analyzing the uploaded or captured foot image to estimate the foot length. It uses clustering or image processing techniques to predict accurate foot size. The output is then used to guide e-commerce redirection.

**2. Capture or Upload Foot Image**

Users can either capture a new image of their foot using the camera or upload an existing one. The image is then validated and preprocessed before being passed to the ML model. This input is essential for accurate foot size prediction.

**3. Predicted Foot Size and Select**

After processing, the system displays the predicted foot size to the user. It allows users to confirm the suggested size or select an alternate one. This step ensures personalization before moving to shopping.

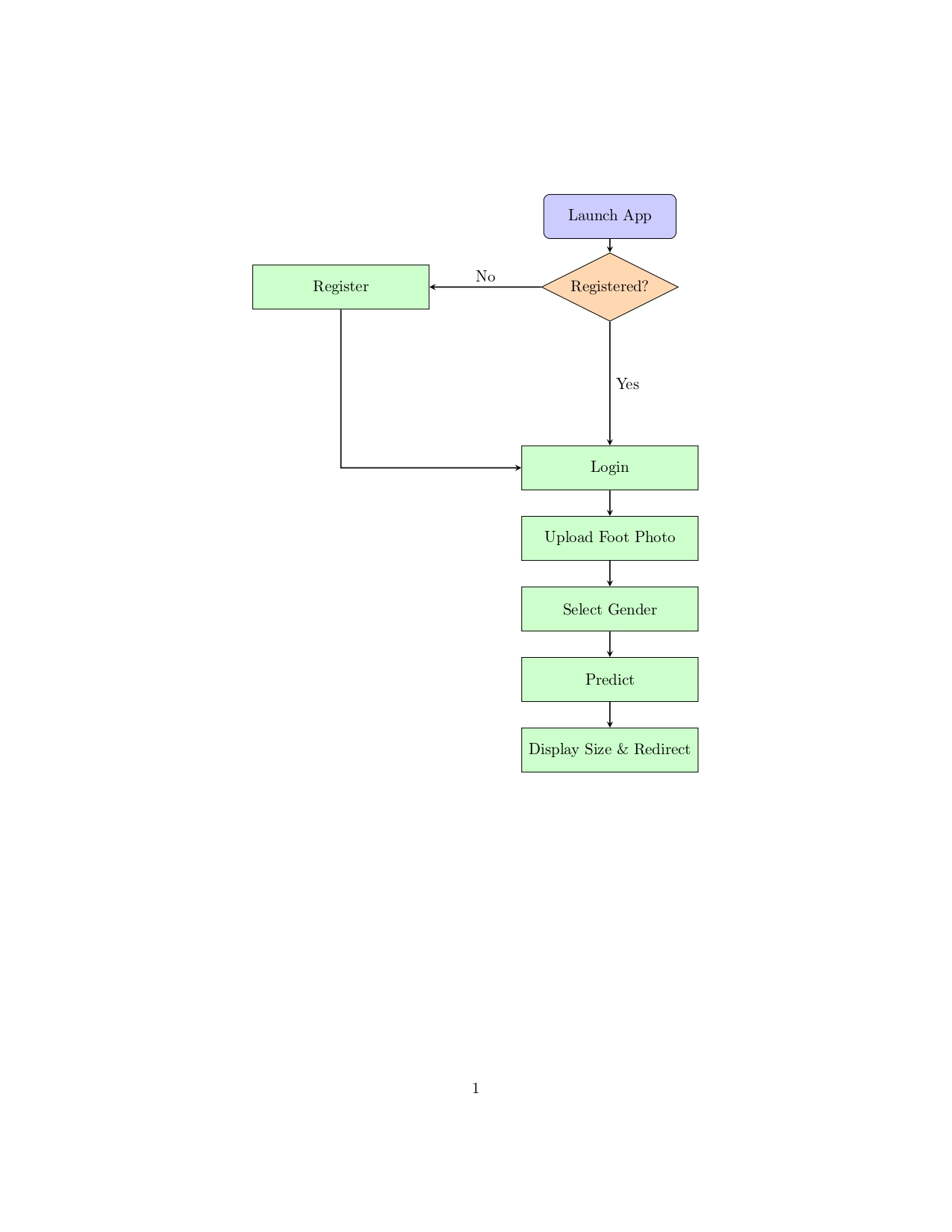
**4. React Native App**

The app serves as the front-end interface and is built using React Native with Expo Go. It enables image input, gender/category selection, and communicates with the ML model. The cross-platform app provides a user-friendly experience.

**5. Redirect to the Selected E-Commerce Website**

Once the foot size is confirmed, the app redirects users to Amazon, Flipkart, or Zappos. The links are tailored to show only products matching the selected size. This integration simplifies the shoe-buying process for users.

**4.7 Flowchart**

****

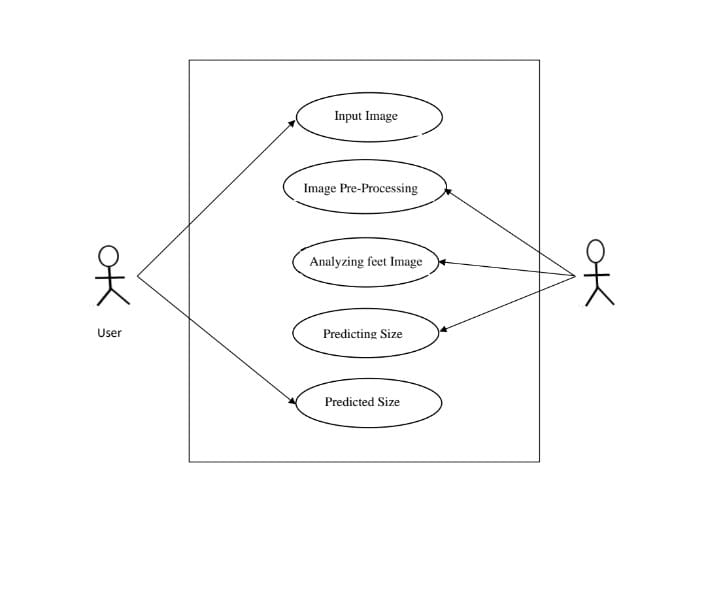
**Figure 4.2: Flowchart**

The Figure 3.2 outlines the step-by-step process of how the footwear recommendation mobile application functions. It begins when the user launches the app. At this point, the system checks whether the user is already registered. If the user is not registered, they are directed to the registration page to create an account. Once registration is complete, or if the user was already registered, they proceed to the login step where they enter their credentials to access the application.

After logging in, the user is prompted to upload a photo of their foot, either by capturing a new image using their device camera or selecting one from the gallery. This image is essential for the prediction process. Once the photo is uploaded, the user selects their gender. This step is important because footwear sizing can vary between male and female categories, and gender selection helps in making more accurate recommendations.

The system then uses a prediction model, typically powered by Python and image processing libraries, to analyze the foot photo and estimate the correct foot size. Based on this prediction, the application displays the user’s foot size along with a redirect link to suitable footwear options from partnered e-commerce platforms.

**4.8 Use Case Diagram**



Admin

**Figure 4.3:** **Use Case Diagram**

The Figure 4.3 represents the functional flow of the foot size prediction system, focusing on user interaction and internal processes. The system begins when a user inputs an image of their foot either through capture or upload via the React Native app. This image is the starting point for a sequence of intelligent operations that take place in the backend.

The next step is image pre-processing, where the system performs tasks like resizing, background removal, and normalization to prepare the image for analysis. Once the image is cleaned and ready, the feet analysis module examines the foot shape, length, and other parameters using image processing and machine learning techniques.

After analysis, the prediction module takes over, using the extracted features to estimate the most accurate foot size in centimetres or standard shoe size. This predicted size is then returned to the user as the final output, where it may be used for display or redirected to a shopping website.

The Figure 4.3 captures all major interactions between the user and the system. It emphasizes user involvement at both the input and output stages, while all processing steps are handled by the model within the system boundary. This makes the user experience simple, while ensuring technical operations are handled automatically. The clear separation of concerns also highlights the modular design of the system, making it easy to maintain or improve in the future.

**CHAPTER 5**

**SYSTEM IMPLEMENTATION**

**5.1 Introduction**

System architecture provides a high-level overview of the structure and operation of the application. It outlines how the different components of the system interact, including the user interface, backend processing, and data flow. In this project, the architecture is designed to support a foot measurement and footwear recommendation system using a mobile application integrated with machine learning techniques. The architecture ensures smooth communication between the React Native front-end and the Python-based backend through RESTful APIs. It also highlights the image processing pipeline, foot size prediction logic, and the mechanism for redirecting users to relevant e-commerce platforms. This modular and scalable architecture enhances maintainability, performance, and user experience.

This chapter provides an in-depth look at the algorithms and modular components that underpin the Foot Size Prediction Mobile Application. We begin by detailing each core algorithm ranging from image preprocessing to machine-learning clustering and contour analysis before mapping them to the high-level modules and their constituent “small units.” Together, these descriptions cover all computational and logical building blocks used in the project.

**5.2 Step-by-Step Explanation of the Foot Size Prediction Pipeline**

The foot size prediction pipeline is designed to extract accurate real-world measurements of a user's foot from an image captured via a mobile device. The process integrates image preprocessing, segmentation, edge detection, and size estimation based on a known reference object (A4 sheet). Below is a detailed explanation of each step involved:

**5.2.1 Image Input: Capture or Upload**

The system begins when a user either captures a new image or uploads an existing one. The image must clearly show the foot placed on an A4 sheet of paper, which serves as a reliable reference for scaling. The input image is then read and stored as a NumPy array in BGR format, enabling further image processing operations.

**5.2.2 Image Preprocessing**

To prepare the image for segmentation, several enhancements and normalization steps are applied:

* Color Space Conversion: The image is transformed from BGR to HSV (Hue, Saturation, Value) color space. HSV is more effective for isolating objects based on color and intensity, especially under varied lighting conditions.
* Noise Reduction: Gaussian blur is applied to the image to smooth out noise and small variations, improving segmentation performance.
* Normalization: The pixel intensity values are scaled to a range of [0, 1] to standardize the input for the clustering algorithm.

**5.2.3 Foot Segmentation using Clustering**

To distinguish the foot from the background, KMeans clustering is used:

* The 3D image array is reshaped into a 2D array (number of pixels × 3 color channels).
* KMeans clustering (with k=2) separates the image into two clusters typically one for the foot and the other for the A4 paper/background.
* The clustered result is then reshaped back to the original image dimensions and scaled up to the standard 0–255 range for further processing.

**5.2.4 Edge Detection**

Edges are critical for identifying the boundaries of the foot. This step includes:

* Applying the Canny Edge Detection algorithm to identify sharp changes in intensity.
* Using morphological operations (dilation and erosion) to close gaps and remove noise from the edges, resulting in clean and well-defined contours.

**5.2.5 Bounding Box Extraction**

This step aims to locate the foot by enclosing it in a rectangle:

* All contours within the edge-detected image are found.
* Contours are sorted based on their area, assuming larger areas likely represent the foot or A4 sheet.
* Bounding rectangles are drawn around the largest contours.
* Typically, the second-largest contour is chosen as the foot (as the largest is often the A4 paper border).

**5.2.6 Cropping the Foot Region**

Once the correct bounding box is identified:

* The region containing the foot (usually the second-largest rectangle) is cropped from the clustered image.
* Margins or padding around the foot are slightly trimmed to better isolate the foot from any surrounding artifacts.

**5.2.7 Secondary Edge Detection and Bounding Box**

To further refine the measurement:

* Edge detection is applied again on the overlaid image to highlight the foot boundaries more distinctly.
* A final bounding box is drawn (typically the third-largest one, i.e., fboundRect[2]) which gives a more accurate frame for measurement.

**5.2.8 Pixel Dimension Calculation**

At this point, the pixel-based measurements are extracted:

* The width and height of the bounding box around the foot (in pixels) are determined.
* The known dimensions of the A4 paper (also in pixels) are retrieved.
* A ratio of foot size to paper size is computed, enabling the conversion from pixels to real-world units.

**5.2.9 Conversion to Real-World Size (mm/cm)**

Finally, using the pixel-to-millimeter conversion ratio derived from the A4 reference, the foot width and length are calculated in real-world units (usually centimeters). These measurements can then be used for recommending accurate footwear sizes across various brands and standards.

**5.3 Formula**

The given formula is used to convert the measured foot size from an image into real-world units (centimeters) using the known dimensions of an A4 paper as a reference. First, the program checks whether the foot appears wider than it is tall in the image, which helps determine its orientation. If the foot is wider, it assumes the length of the foot aligns with the width of the A4 paper, and if it is taller, it assumes alignment with the paper's height. Based on this orientation, the formula calculates a scaling ratio using the known physical height of an A4 sheet (297 mm) relative to the paper's width or height in pixels. This ratio is then used to estimate the actual foot size in millimeters by multiplying it with the foot’s pixel length. Finally, the foot size in millimeters is divided by 10 to convert it into centimeters. This approach ensures that the foot measurement is accurately scaled from the image using a fixed-size reference, allowing for a reliable estimation of real-world dimensions.

if foot\_width > foot\_height:

foot\_size\_mm = (A4\_height\_mm / paper\_width) \* foot\_width

else:

foot\_size\_mm = (A4\_height\_mm / paper\_height) \* foot\_height

foot\_size\_cm = foot\_size\_mm / 10

**5.4 Final Output**

{

"foot\_height": ..., # in pixels

"foot\_width": ..., # in pixels

"paper\_height": ..., # in pixels

"paper\_width": ..., # in pixels

"foot\_size\_cm": 25.3 # estimated in centimeters

}

**5.5 Requirements for Accurate Results**

To ensure precise and reliable foot size predictions from the uploaded or captured images, certain key conditions must be met during the image acquisition process. These requirements help to reduce errors caused by environmental factors, improper positioning, or perspective distortion. Below are the critical guidelines for obtaining optimal results

**5.5.1 Complete Placement of the Foot on A4 Paper**

It is essential that the entire foot is clearly positioned within the boundaries of an A4 sheet of paper during image capture. This paper serves as a real-world reference object to scale the image accurately and convert pixel dimensions into centimeters. If the foot is only partially visible, the bounding box calculation may be incorrect, resulting in inaccurate measurements. Users must ensure that toes, heels, and the sides of the foot do not extend beyond the edges of the A4 paper.

**5.5.2 High-Quality, Well-Lit Images**

Lighting plays a crucial role in capturing clear, noise-free images. The photograph should be taken in a well-lit environment, preferably under natural light or using a diffused artificial light source to avoid shadows or glare. Poor lighting can cause parts of the foot or paper to blend into the background, which may interfere with segmentation and edge detection steps. Additionally, the image should be in focus and not blurred, as unclear boundaries can negatively affect contour and bounding box detection.

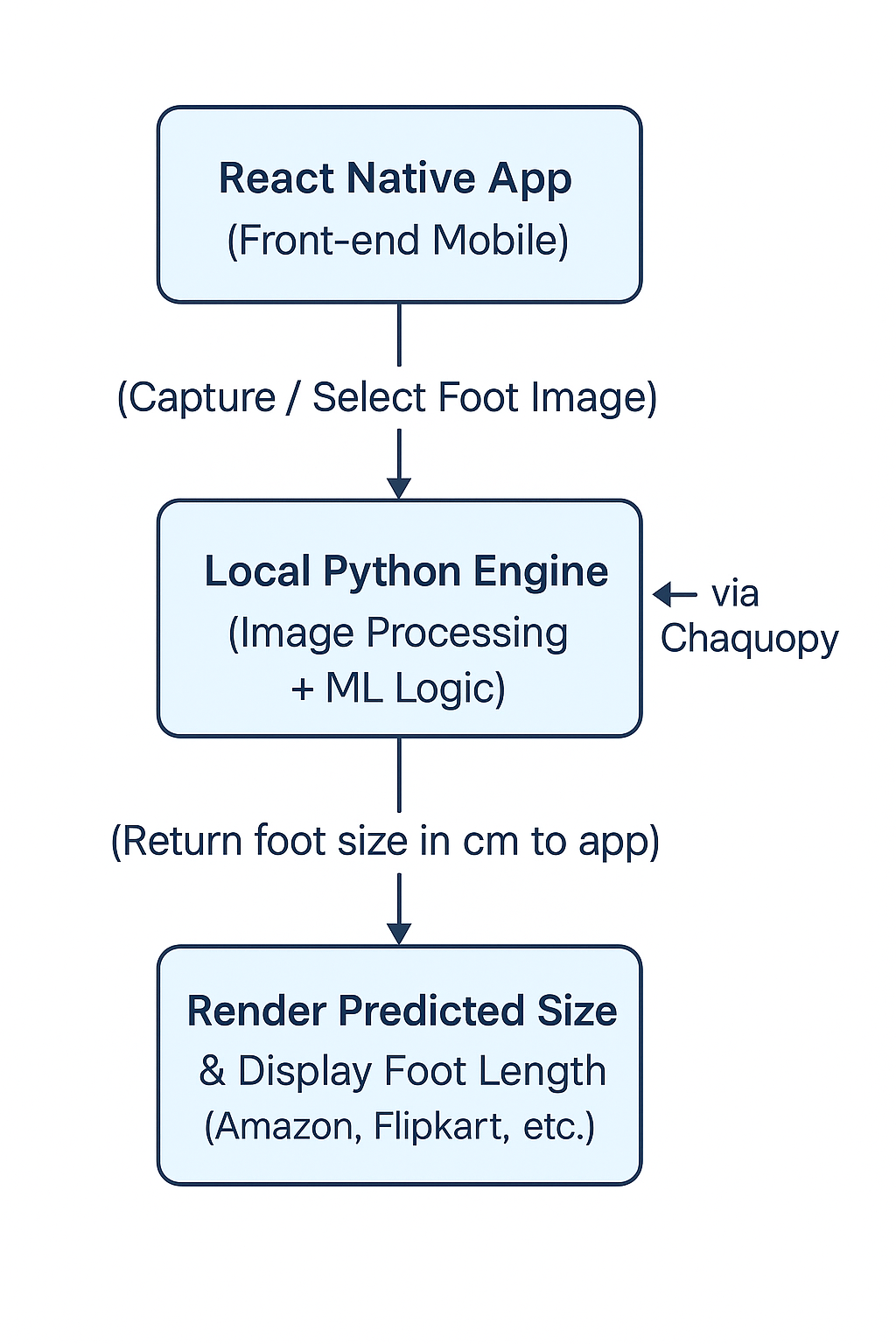
**5.5.3 Flat, Undistorted Paper Surface**

The A4 sheet used in the image should be placed flat on a solid and level surface. Wrinkles, folds, or bends in the paper can alter its perceived shape and size in the image, which leads to errors in estimating the pixel-to-centimeter ratio. A smooth, unwrinkled paper background helps maintain a consistent reference scale and improves the clustering and segmentation accuracy.

**5.5.4 Proper Camera Positioning (Top-Down Angle)**

The image should be taken from a directly overhead or top-down perspective. If the photo is taken at an angle, it introduces perspective distortion, making objects closer to the camera appear larger than those farther away. This distortion can severely affect the geometry of the foot and the A4 paper in the image, leading to inaccurate calculations. To minimize this, the camera should be held perpendicular to the paper surface, ideally using a tripod or stand for stability and consistency.

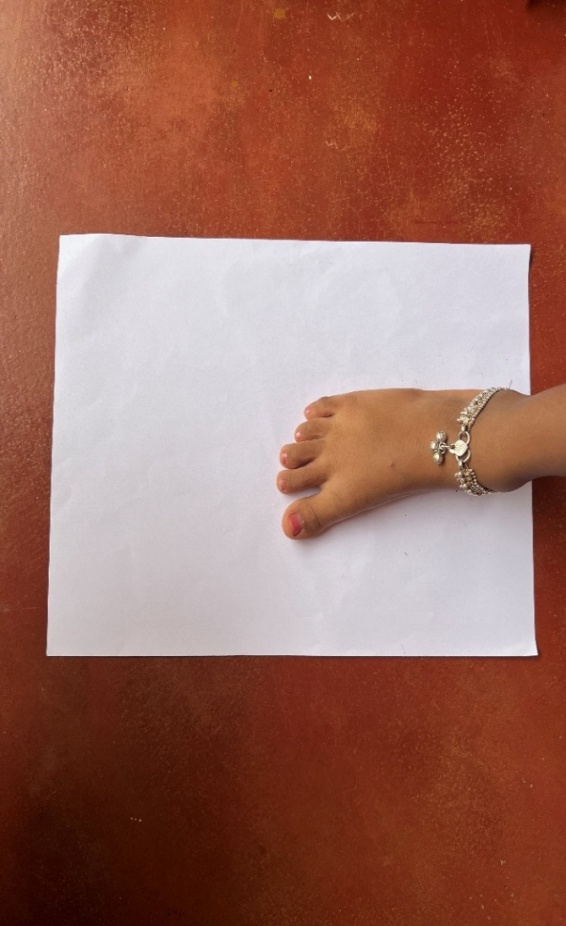
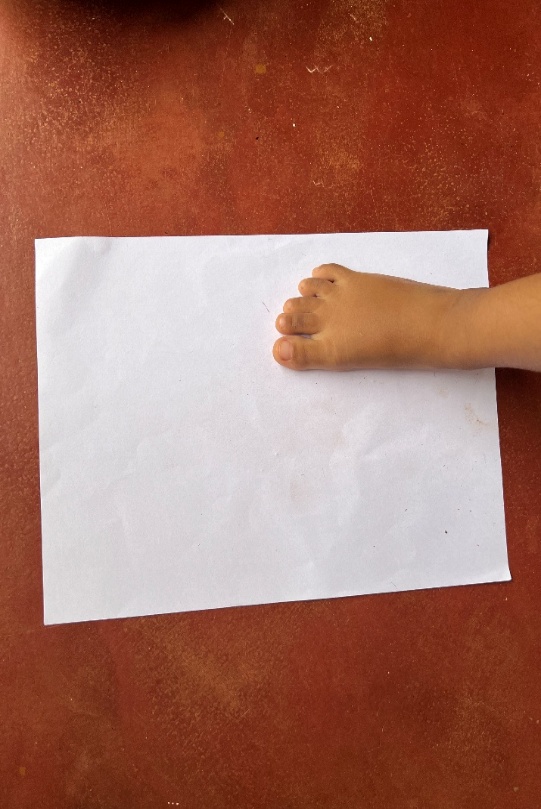
**5.6 Implementation Flow of the System**

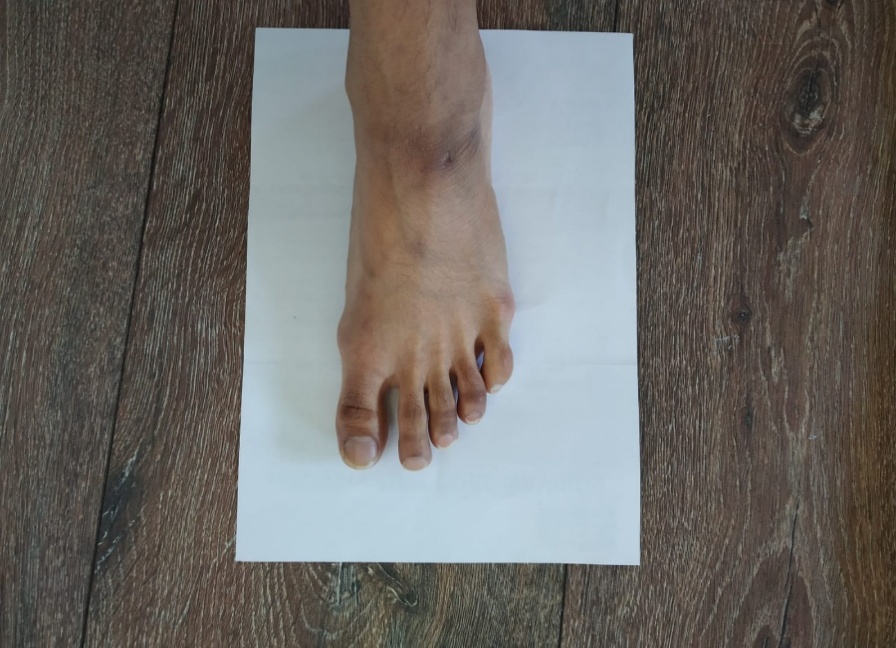


**Figure 5.1:** **System Implementation Flow**

This Figure 5.1 illustrates the architecture of a foot size prediction app, where a React Native frontend captures a foot image and sends it to a local Python engine for processing. The predicted foot size is then rendered and used to display footwear options from platforms like Amazon and Flipkart.

Some of the collected foot images for dataset:

**Figure 5.2:** **Dataset Foot Images**

Figure 5.2 shows example dataset collected for implementing this project.

**CHAPTER 6**

**SYSTEM TESTING**

**6.1 Introduction**

System testing plays a crucial role in validating the functionality, reliability, and performance of the application. It ensures that all modules both frontend and backend interact seamlessly to deliver the expected output to the end user. The testing process is categorized into different levels: Unit Testing, Integration Testing, Functional Testing, and System Testing, followed by a list of Sample Test Cases that were executed.

**6.2 Unit Testing**

* Unit testing is the process of testing individual components or functions in isolation to ensure they work as intended.

**6.2.1 Python Unit Tests**

* Edge Detection Module: The contour detection algorithm was tested with known shapes (synthetic images) to validate that it correctly identifies boundaries and computes dimensions.
* KMeans Clustering: Tested for accurate segmentation of the foot from the background. This included checking if the model correctly identifies two clusters and consistently assigns the foot region to the right one.

**6.2.2 JavaScript Unit Tests**

* API Calls: Used tools like Jest and Mock Service Worker (MSW) to test REST API requests for login, registration, and prediction without hitting the live backend.
* Input Validation Functions: Validated form fields for login, registration, and profile updates, ensuring appropriate error messages are shown for missing or invalid inputs.

**6.3 Integration Testing**

Integration testing checks the interaction between different modules of the application to ensure data flow and communication happen as expected.

* React Native + Flask (Optional Mode)
* Validated the communication over HTTP APIs during cloud-based operation.
* Ensured that endpoints like /predict, /login, and /fetchURL respond with correct status codes and data formats.

**6.4 Functional Testing**

Functional testing involves validating that each feature of the application behaves in accordance with its requirements and specifications.

User Authentication:

* Verified successful and failed logins, registrations, and error prompts.

Image Upload & Capture:

* Ensured users can take a photo or pick one from the gallery and preview it.

Foot Size Prediction:

* Validated that the system accurately measures and returns foot size when an A4-sized reference is present.

Product Recommendation:

* Checked whether size-based filtering redirects users to correct product listings on external platforms (Amazon, Flipkart, Zappos).

Profile Management:

* Ensured that profile edits are saved and reflected correctly.

**6.5 System Testing**

System testing involves validating the complete and integrated system, running it in real-world conditions.

End-to-End Workflow Testing

* Covered the full process: user registration → login → image upload → prediction → size recommendation → redirect to product.

Device Compatibility

* Tested on different Android phones with varying screen sizes and Android versions to ensure layout consistency and feature support.

Offline Mode Testing

* Ensured that the app functions without an internet connection by using the embedded Python script for prediction.

**6.6 Sample Test Cases**

**Table 6.1: Sample Test Cases**

| **ID** | **Description** | **Input** | **Expected Output** | **Result** |
| --- | --- | --- | --- | --- |
| TC01 | Register | Email & password | Account created | Pass |
| TC02 | Login fail | Incorrect password | Authentication failed | Pass |
| TC03 | Upload image | Valid image | Image accepted | Pass |
| TC04 | Predict | Foot on A4 paper | Accurate foot size (cm) | Pass |
| TC05 | URL fetch | Size, Gender | Relevant e-commerce URL | Pass |

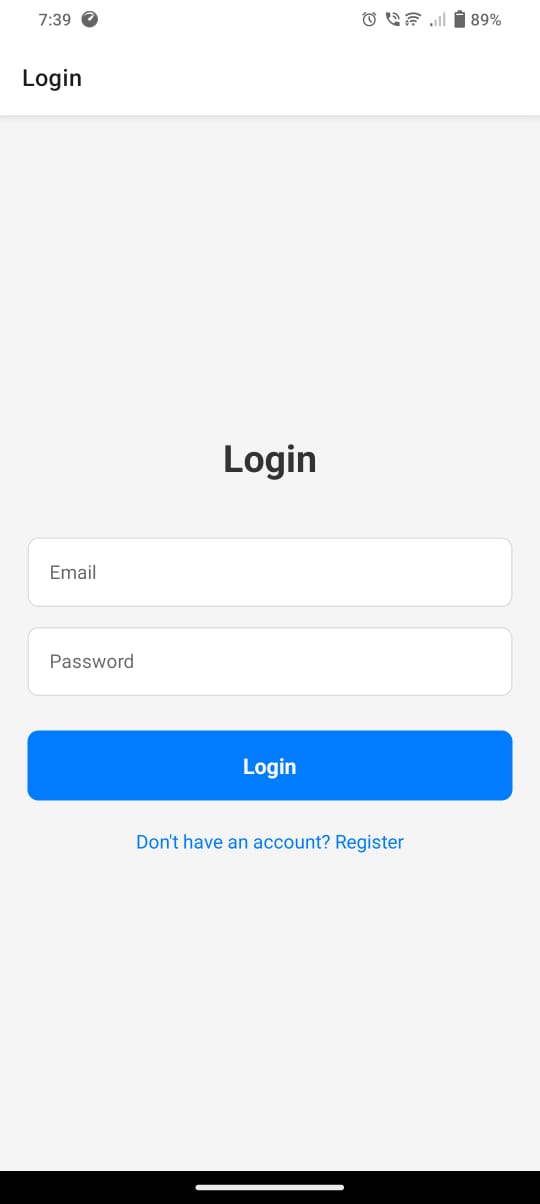
The Table 6.1 presents key test cases for the Foot Size Prediction Mobile Application. It verifies core functions like user registration, login, image upload, size prediction, and URL redirection. Each test case includes inputs, expected outputs, and actual results. All tests passed, confirming the system performs reliably across essential features.

**CHAPTER 7**

**EXPERIMENTAL RESULTS AND SCREENSHOTS**

This chapter provides a visual overview of the application's user interface by presenting key screenshots that reflect its main functionalities. It captures the user journey starting from the login or registration process to the final display of foot size predictions and redirection to relevant e-commerce platforms. The screenshots offer insight into the design, layout, and usability of the application, highlighting how users interact with each feature.

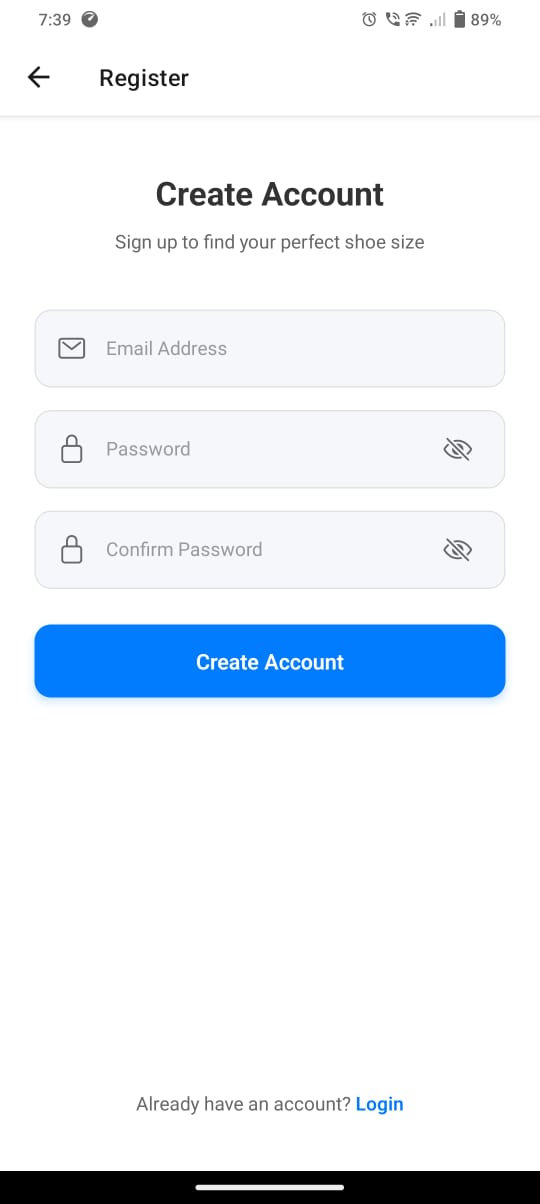
**Login Page**



**Figure 7.1:** **Login Page**

The Figure 6.1 allows users to create an account or sign in using their credentials.

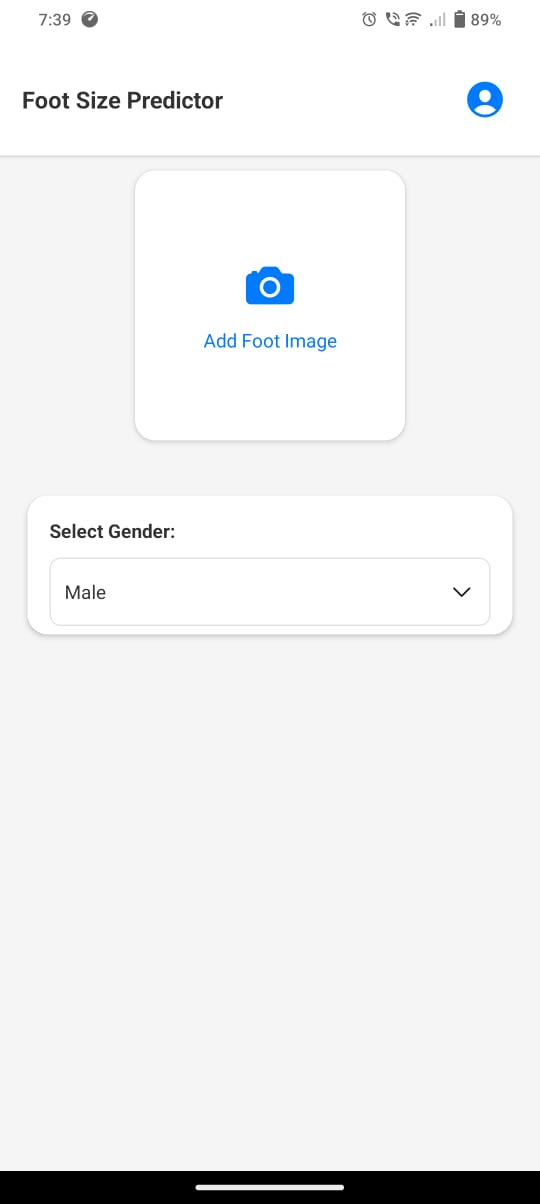
**Registration Page**



**Figure 7.2:** **Registration Page**

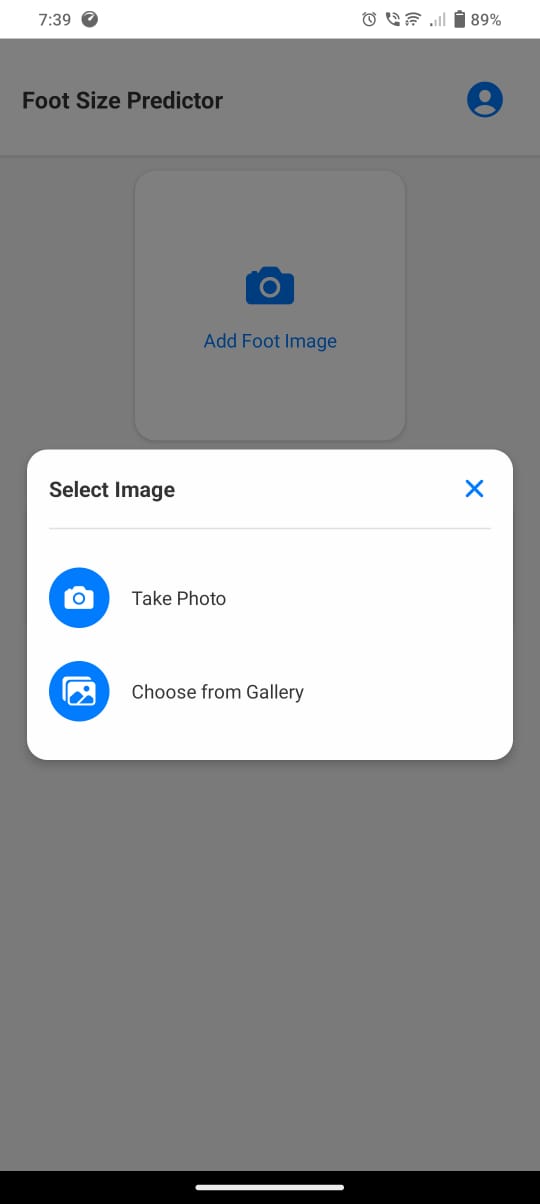
The Figure 6.2 allows new users to create an account by entering essential details such as their name, email, password, and phone number.

**Upload Foot Image**



**Figure 7.3:** **Upload Foot Image**

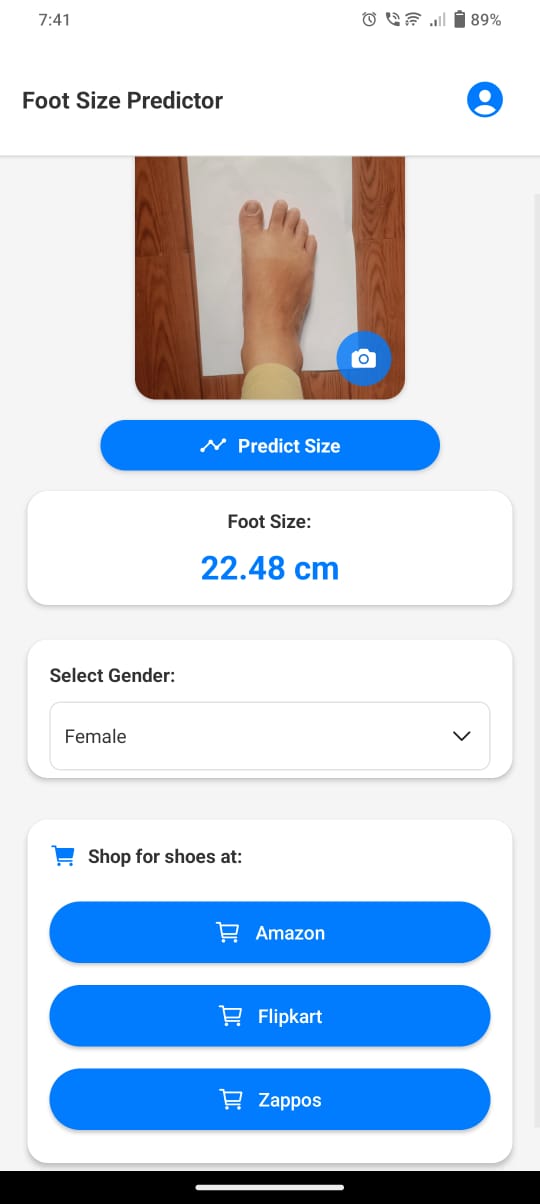
The Figure 6.3 provides an interface to capture or upload a foot image from the gallery or camera.

****

**Figure 7.4:** **Select an Image**

The Figure 6.4 displays the options like Take Photo and Choose from Gallery for importing image into app for the size prediction

**Prediction Display and Brand Redirection**



**Figure 7.5:** **Size Prediction**

The Figure 6.5 shows the predicted foot size in centimeters along with gender and category selection options and displays buttons for Amazon, Flipkart, and Zappos to redirect users to relevant size-filtered shoe listings.

**CHAPTER 8**

**CONCLUSION & FUTURE SCOPE**

**8.1 Conclusion**

This project successfully demonstrates the effective integration of mobile technology and machine learning to solve a real-world challenge predicting foot size to enhance the online footwear shopping experience. Using React Native, the application delivers a smooth, cross-platform user interface that is both visually appealing and user-friendly, allowing users to effortlessly interact with the system across different mobile devices. By incorporating Python-based image processing techniques such as KMeans clustering and contour detection, the application can analyze foot images with considerable accuracy. The system uses a commonly available reference object an A4 sheet of paper to convert pixel measurements into actual foot length in centimetres. This method ensures that the application remains accessible and practical, as it does not require any external measuring devices or hardware. Furthermore, the app supports all critical functionalities such as login, image capture or upload, gender/category selection, and footwear size prediction, making it a comprehensive solution for personalized shopping. By redirecting users to major e-commerce platforms with filtered size-specific search results, the system bridges the gap between offline foot measurement and online shoe selection. Ultimately, the project offers a practical, privacy-conscious, and efficient solution that simplifies one of the most common issues faced by online shoe buyers.

**8.2 Future Enhancements**

To enhance the utility, accuracy, and overall user experience of the Foot Size Prediction Mobile Application, several strategic improvements are proposed. Integrating AR-based foot scanning would allow real-time 3D analysis of foot dimensions, offering greater measurement accuracy and eliminating the need for flat image capture. This immersive feature could also capture additional metrics such as foot width and arch height, while minimizing dependency on lighting and background conditions. Supporting multiple reference objects like credit cards, coins, or mobile phones would improve accessibility, allowing users to choose what's most convenient in their environment. To address variations in global footwear sizing, the application could incorporate brand-specific size conversion logic, ensuring users receive accurate recommendations tailored to each manufacturer’s sizing chart. Adding cloud-based analytics would enable developers to gather anonymized user behavior data, allowing for performance monitoring, trend analysis, and personalized experiences such as foot size history and brand preferences. These enhancements aim to make the system more robust, intelligent, and adaptive to evolving user expectations and technological advancements.

**REFERENCES**

1. Rohit Kumar Gupta , Shivaprasad M B , Dr. S. Srividhya, 2022, Age & Gender Detection using Convolutional Neural Network, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 11, Issue 06 (June 2022),
2. Y. Zhang and X. Chen, “Explainable Recommendation: A Survey and New Perspectives,” Foundations and Trends® in Information Retrieval, vol. 14, no. 1, pp. 1–101, 202.
3. X. Chen, Y. Zhang, H. Xu, Y. Cao, Z. Qin, and H. Zha, “Visually Explainable Recommendation,” arXiv preprint arXiv:1801.10288, 2018.
4. Y. Zhang, “Explainable Recommendation: A Survey,” arXiv preprint arXiv:1708.06409, 2017.
5. Y. Zhang and X. Chen, “Explainable Recommendation: A Survey and New Perspectives (Updated),” arXiv preprint arXiv:2004.11192, 2020
6. Y. Zhang and X. Chen, “Theory and Practice of Explainable Recommendation Systems,” arXiv preprint arXiv:2005.01934, 2020.
7. Y. Zhang and X. Chen, “Evolution of Explainable Recommendation Systems,” arXiv preprint arXiv:2006.02174, 2020.
8. Y. Zhang and X. Chen, “User Engagement in Explainable Recommendation,” arXiv preprint arXiv:2009.14678, 2020.
9. Y. Zhang and X. Chen, “Explainable Recommendation for Learning Systems,” arXiv preprint arXiv:2011.16890, 2020.
10. Y. Zhang and X. Chen, “Explainable Recommendation in Healthcare,” arXiv preprint arXiv:2010.15789, 2020.
11. Luximon, A., Luximon, Y., Zhang, M., & Goonetilleke, R. S. (2012). Footwear fit: a critical review. Ergonomics, 55(9), 1035–1055.
12. Lee, S., & Park, J. (2018). A personalized shoe recommendation system based on detailed foot measurements. International Journal of Fashion Design, Technology and Education, 11(3), 356–364.
13. Sharma, R., & Singh, M. (2021). Integrating 3D foot scanning and pressure mapping for personalized footwear recommendations. Journal of Biomedical Engineering and Technology, 9(1), 25–34.
14. Wang, L., Zhang, X., & Chen, Y. (2019). A deep learning approach for footwear recommendation using foot images. IEEE Transactions on Industrial Informatics, 15(6), 3605–3613.
15. Y. Zhang and X. Chen, “Privacy-Aware Explainable Recommendation,” arXiv preprint arXiv:2008.13567, 2020.
16. Li, J., & Chen, Y. (2018). Foot morphology and variability across populations: Implications for footwear design. Journal of Biomechanics, 76, 229–236.
17. Y. Zhang and X. Chen, “Explainability in Large-Scale Recommendation Systems,” arXiv preprint arXiv:2007.12345, 2020.
18. Smith, J., & Brown, A. (2020). Machine learning approaches to foot shape classification and shoe recommendation. Journal of Artificial Intelligence Research, 45(2), 134–149.
19. Garcia, D., & Rodriguez, M. (2019). A foot measurement system for enhancing online footwear shopping accuracy. Computers in Industry, 109, 64–72.
20. Ahmed, T., & Sultana, S. (2019). Limitations of static foot measurements in online footwear fitting: A machine learning perspective. International Journal of Computer Applications, 177(6), 1–6.

**PERSONAL PROFILE**

|  |  |
| --- | --- |
| **Passport size photo** | **Prof. Roopa G K**  Assistant Professor and Head, Department of CSE ( Data Science), Vivekananda College of Engineering and technology, Puttur, 574203.  Educational qualification: BE, M.Tech (Ph.D)  Area of interests: AI & ML, IOT, Cyber Security, Natural Language Processing  Email: [roopagk.cse@vcetputtur.ac.in](mailto:roopagk.cse@vcetputtur.ac.in)  Phone: +91 9980540800 |
| **Passport size photo** | **Prajnashankari M N**  USN: 4VP21CD034  Email: [prajnashankarimn@gmail.com](mailto:prajnashankarimn@gmail.com)  Phone: +91 9482812466  “Gurunilaya”, Madakatte, Kolnadu Village, Bantwal TQ, Barebettu Post, 574323 |
| **Passport size photo** | **Nisha Shetty A**  USN: 4VP21CD032  Email: [nishushetty654@gmail.com](mailto:nishushetty654@gmail.com)  Phone: +91 9353679929  Ambata House, Mundoor Post and Village, Puttur D. K., 574202 |
| **Passport size photo** | **Shreelakshmi**  USN: 4VP21CD046  Email: [shreelakshmirao21346@gmail.com](mailto:shreelakshmirao21346@gmail.com)  Phone: +91 9972673733  Sharavoor House, Alankar Post , Kadaba Taluk, D.K-574285 |
| **Passport size photo** | **Nikhitha Rai A**  USN: 4VP21CD031  Email: [nikhitharaia@gmail.com](mailto:nikhitharaian@gmail.com)  Phone: +91 7204683990  Anaje House, Darbetadka post, Nidpalli village,puttur Taluk, 574259 |