

ÇANKAYA UNIVERSITY

Software Design Document

Skinalyzer: AI-Based Skin Cancer Detection System

Melike Hazal Özcan 202011013

Bilgesu Fındık 202011407

Pelin Koz 202011048

Bekir Emrehan Şimşek 202011039

1.Introduction

1.1 Purpose

This document serves as a comprehensive guide to the design and architecture of the Skinalyzer: AI-Based Skin Cancer Detection System. It aims to detail the system's core components, structural framework, and workflow processes, providing a blueprint for its seamless implementation. By integrating advanced techniques such as Federated Learning and Incremental Learning, the system achieves a perfect balance of privacy, adaptability, and high diagnostic precision.

1.2 Scope

The scope of this project includes the development of a desktop application for diagnosing skin cancer using machine learning. The system will analyze skin lesion images uploaded by users, ensuring privacy through local data processing. A centralized server will aggregate updates from client devices to refine a global model, while Incremental Learning ensures continuous improvement with new data. Designed for healthcare professionals and individual users, the system prioritizes usability, scalability, and secure communication.

1.3 Glossary

AI-Based Skin Cancer Detection System: A system utilizing Artificial Intelligence (AI) to analyze skin lesion images for early detection of skin cancer.

Federated Learning (FL): A machine learning approach where data remains on client devices, and only model updates (e.g., gradients) are shared with a central server to maintain privacy.

Incremental Learning (IL): A technique that allows machine learning models to learn new data while retaining knowledge from previously trained data.

Central Server: A server that aggregates model updates from client devices, refines a global model, and redistributes it to clients.

Client Device: A user's local device (e.g., desktop, mobile) used for uploading skin lesion images, processing data, and training models locally.

Global Model: A centralized machine learning model that combines updates from multiple clients to improve performance and accuracy.

Analysis Result: The outcome of processing and analyzing a skin lesion image, including the diagnosis and confidence score.

Skin Lesion: A visible abnormality on the skin, which may indicate the presence of skin cancer.

Gradient Data: The data generated during the training of a machine learning model, used for updating the global model in federated learning.

Login History: Records of users' login attempts, including timestamps and user identifiers.

Authentication: The process of verifying a user's identity using credentials like email and password.

Verification Email: An email sent to users during the registration process to confirm their email address.

Base Model: An initial machine learning model distributed to clients for local training in a federated learning workflow.

Data Repository: A centralized storage system for user-uploaded images and other data needed for future analysis or training.

Update Log: A log of changes and updates made to the global model, including contributions from specific client devices.

User Interface (UI): The visual component of the system that allows users to interact with features such as image uploads and result viewing.

1.4 Overview of Document

This document provides a structured and detailed outline of the design specifications for the Skinalyzer: AI-Based Skin Cancer Detection System. It begins with an introduction to the project, highlighting its purpose, scope, and objectives. The document then delves into the system's architecture, providing a high-level overview of its components, including the user interface, client-side modules, server-side modules, and the database.

One of the key aspects of this document is the detailed design of the user interface (UI). The UI has been carefully crafted using Figma, a collaborative interface design tool, to ensure a user-friendly and intuitive experience. The design process focuses on enabling seamless navigation for users, incorporating essential functionalities such as uploading images, viewing analysis results, accessing past diagnoses, and providing feedback. Visual prototypes created in Figma are included to offer a clear representation of the intended look and feel of the application.

Subsequent sections detail the design methodologies, workflows, and communication mechanisms that ensure the system operates efficiently and securely. Diagrams such as the architectural layout, class diagrams, and entity-relationship diagrams (ERDs) are included to offer visual clarity and a deeper understanding of the system's structure.

The document also covers the integration of key machine learning technologies, including Federated Learning and Incremental Learning, and how these are leveraged to ensure privacy, scalability, and continuous improvement of the model. Finally, it provides implementation details for user interactions, model training, and system maintenance, serving as a practical guide for developers and stakeholders throughout the project's lifecycle.

1.5 Motivation

The Skinalyzer: AI-Based Skin Cancer Detection System is motivated by the critical need for early and accurate diagnosis of skin cancer, which can significantly improve patient outcomes and survival rates. Skin cancer remains one of the most common types of cancer worldwide, and its early detection is often hindered by the lack of accessible, reliable, and affordable diagnostic tools. This project aims to bridge this gap by leveraging cutting-edge machine learning techniques to provide an efficient, user-friendly, and privacy-preserving diagnostic solution.

The integration of Federated Learning ensures that users' sensitive data, such as skin lesion images, remain securely on their local devices, aligning with modern privacy standards and increasing user trust. At the same time, Incremental Learning enables the system to adapt and evolve by learning from new data without losing prior knowledge, ensuring its effectiveness over time.

By designing a system that is accessible to both healthcare professionals and individual users, the Skinalyzer project aspires to democratize advanced diagnostic technologies. It seeks to empower users with actionable insights while reducing the burden on medical professionals, ultimately contributing to improved public health outcomes. The motivation behind this project is rooted in making AI-driven healthcare innovations more inclusive, secure, and reliable for widespread use.

2. SYSTEM OVERVIEW

This project aims to develop a desktop application for supporting the diagnosis of skin cancer by analyzing skin lesion images. The system utilizes advanced machine learning techniques, including Federated Learning (FL) and Incremental Learning (IL), to ensure privacy, adaptability, and efficiency in model training and predictions.

The application provides a user-friendly interface where users can upload images, view analysis results, and access past diagnoses. The system processes uploaded images locally on client devices, contributing to the federated learning process, which ensures that raw user data remains on their devices, maintaining privacy and compliance with data protection standards.

The centralized server collects and aggregates updates from clients to improve the global model. Incremental learning allows the system to update the model with new data while preserving knowledge from previous datasets, ensuring continuous model enhancement without data loss.

The project integrates key components such as:

- **User Interface (UI):** A responsive and intuitive UI for users to interact with the system.
- **Client Module:** Handles local model training and communication with the server.
- **Server Module:** Manages global model updates and provides coordination for the federated learning workflow.
- **Database:** Stores user information, image data, analysis results, and model metadata.

This system architecture ensures efficient data flow, secure communication, and accurate predictions, making it a robust tool for early skin cancer detection.

3. SYSTEM ARCHITECTURE

The **System Architecture** of this project is designed to facilitate a privacy-preserving, scalable, and efficient solution for skin cancer detection using Federated Learning (FL) and Incremental Learning (IL) techniques. The architecture integrates client devices, a centralized server. Below are the key architectural components:

3.1 High-Level Architecture

The system operates in a decentralized manner, where client devices locally process and train models using user-provided data (skin lesion images). The server aggregates these updates, improves the global model, and redistributes the updated model to the clients.

Key Components:

- **Clients:** Local data processing, and model training.
- **Server:** Coordinates model aggregation, versioning, and distribution.

3.2 Federated Learning Workflow

The Federated Learning workflow is at the core of the system. It ensures data privacy by keeping user data local while still enabling collaborative model improvement.

1. Model Initialization:

- The server initializes a base model and distributes it to all clients using Flower libraries.

2. Local Training:

- Clients train the model locally on their datasets (skin lesion images).
- Incremental Learning ensures that new training data is incorporated without losing knowledge from previous datasets.

3. Model Update Communication:

- Clients send model updates (gradients) to the server using Flower libraries.

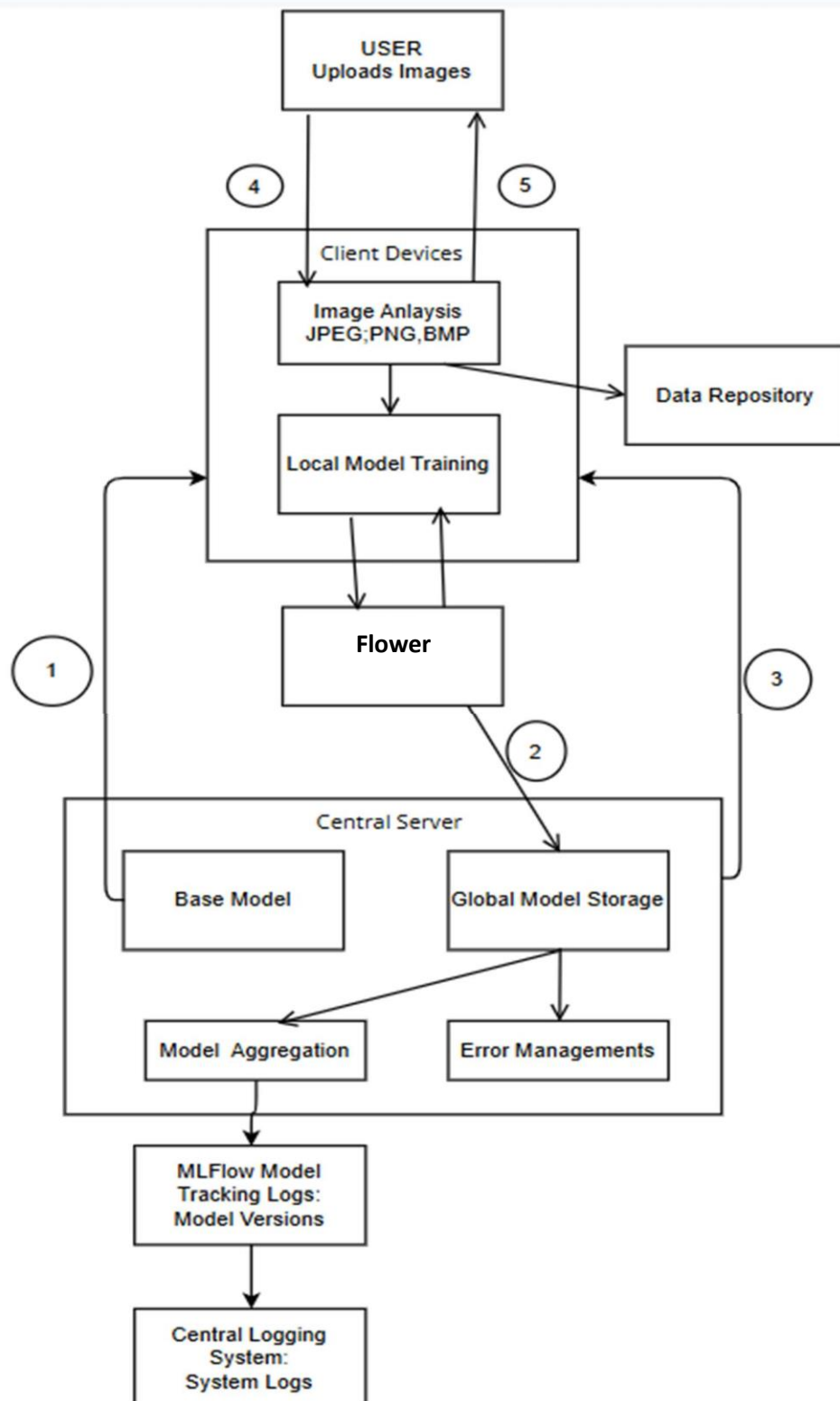
4. Global Model Aggregation:

- The server aggregates updates using Federated Averaging or similar techniques.
- The aggregated model is stored and versioned in Server.

5. Redistribution:

- The updated global model is sent back to clients for further training cycles.

3.3 Architectural Diagram



1. We send the base model to multiple clients and train the model.
2. We combine the trained models to create a global model.
3. We send the global model back to the clients.
4. The user uploads a photo for testing

Loop: Is the Data Repository empty?

- **Yes:**
 - Step 4: The photos accumulate in the repository.
 - Step 5: The client analyzes the photo using the model and shows the result to the user.
- **No:**
 - Go back to Step 2.

1. User

- **Action:** The user uploads skin lesion images to the system through the user interface.
- **Purpose:** Users can view the analysis results and upload new images for diagnosis.

2. Client Devices

- **Components:**
 - **Image Analysis:**
 - Processes uploaded images.
 - Supported formats include JPEG, PNG, and BMP.
 - Ensures compatibility with a wide range of image types.
 - **Local Model Training:**
 - Performs training of the model using local data on the client devices.
 - Sends model weights or gradient updates to the server for aggregation.
- **Role:** Acts as the intermediary between the user and the central server, handling local data processing and model updates.

3. Data Repository

- **Function:**
 - Stores processed images for future reference or analysis.
 - Provides a centralized location for storing user-uploaded data.
- **Role:** Supports data persistence and allows historical analysis if needed.

4. Central Server

- **Components:**
 - **Model Aggregation:**
 - Aggregates model updates (gradient data) from multiple client devices.
 - Combines the updates to improve the global model.
 - **Error Management:**
 - Identifies and handles faulty updates or issues with client-server communication.
 - Ensures stability and reliability of the aggregation process.
- **Role:** Acts as the core of the federated learning system, orchestrating updates and maintaining the global model.

5. Global Model Storage

- **Function:**
 - Stores the aggregated global model.
 - Updates the global model when a predefined threshold is reached (e.g., after receiving enough updates from clients).
- **Purpose:** Serves as the main storage for the machine learning model used across all devices.

6. Central Logging System

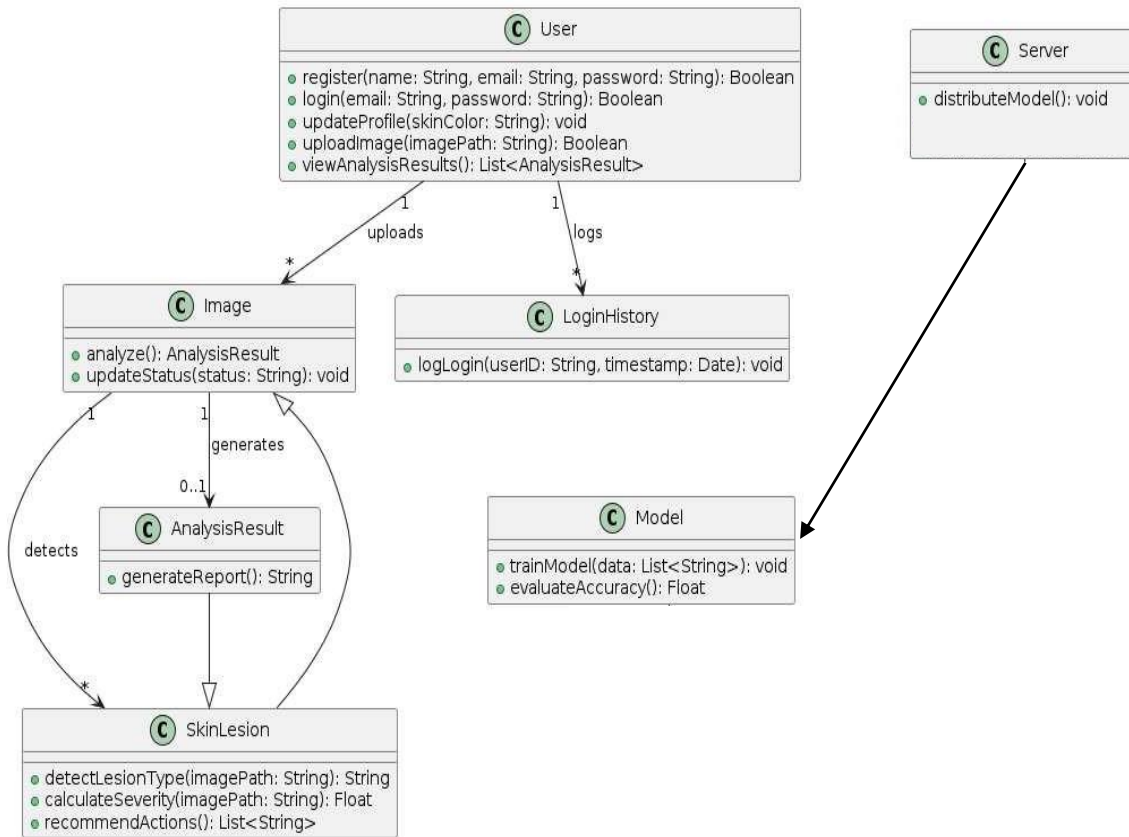
- **Function:**
 - Stores system logs, including events, errors, and updates.
- **Role:** Ensures system transparency and aids in debugging or auditing.

4. Detailed Design

4.1 Class Diagram

Class Descriptions:

1. **User:** Represents the user entity with attributes for name, email, and password. It includes methods for registration, login, profile update, image upload, and viewing analysis results.
2. **Image:** Represents images uploaded by users, with a method to analyze the image and update its status.
3. **AnalysisResult:** Represents the results of image analysis, including generating a report.



- SkinLesion:** Represents skin lesions detected in images, with methods to detect lesion type, calculate severity, and recommend actions.
- LoginHistory:** Logs user login activities with user ID and timestamp.
- Model:** Represents the machine learning model used for training on data and evaluating accuracy.
- UpdateLog:** Logs updates for models, detailing the update specifics.
- Client:** Represents client devices that sync with the server and send model updates.
- Server:** Manages the distribution of models in the federated learning system.

Relationships:

- Users upload images, which are then analyzed, generating results that can detect skin lesions.
- Users' login activities are recorded in the login history.
- The model is trained with data and its accuracy is evaluated, with updates being logged.
- Clients perform updates which are sent to the server, and the server distributes these updates back to clients.

This class diagram captures the flow and interaction between different components in a federated learning system tailored for skin cancer analysis.

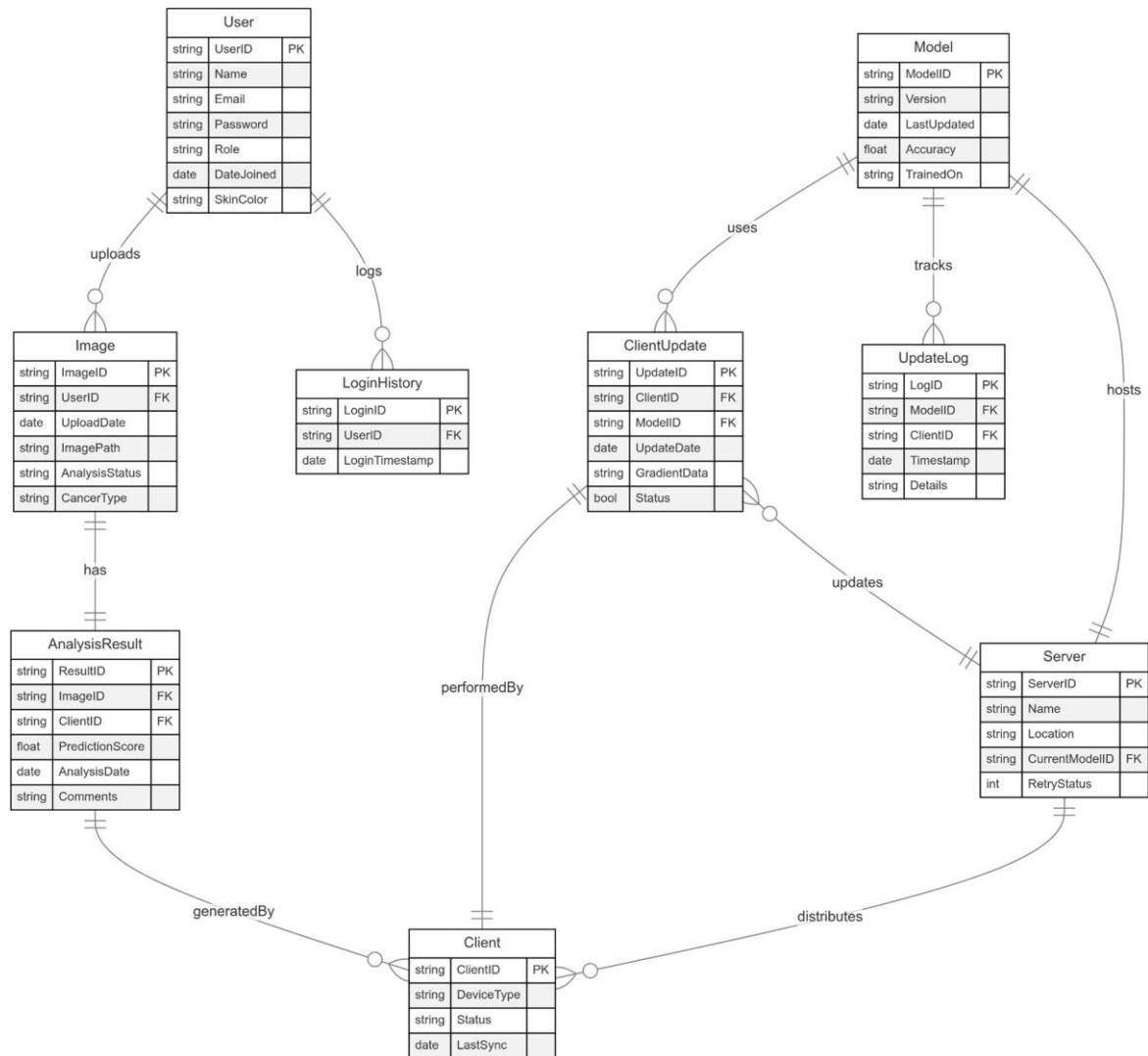
4.2 Entity Relationship Diagram (ERD)

This diagram is an **Entity-Relationship (ER) Diagram** representing the key entities, their attributes, and relationships in a system designed for federated learning and skin cancer analysis.

Key Entities and Their Attributes:

1. User

- **Description:** Represents the primary users of the system who upload images and view analysis results.
- **Attributes:**
 - UserID: Uniquely identifies each user.
 - Name, Email, Password: Stores user credentials and roles.
 - DateJoined: Tracks when the user joined the system.



- **SkinColor**: Captures skin color information, useful for specific analyses.
- **Relationships**:
- **uploads**: A user uploads images stored in the Image table.
- **logs**: A user's login history is recorded in the LoginHistory table.

2. Image

- **Description:** Represents images uploaded by users for analysis.
- **Attributes:**
 - ImageID: Uniquely identifies each image.
 - UserID: References the user who uploaded the image.
 - UploadDate, ImagePath: Tracks the upload date and file path of the image.
 - AnalysisStatus, CancerType: Indicates the analysis status and detected cancer type.
- **Relationships:**
 - **has:** Each image is associated with an analysis result in the AnalysisResult table.

3. AnalysisResult

- **Description:** Captures the results of analyzing uploaded images.
- **Attributes:**
 - ResultID: Uniquely identifies each analysis result.
 - ImageID: Links the result to the analyzed image.
 - PredictionScore: Confidence level or probability of the prediction.
 - AnalysisDate: Date of the analysis.
 - Comments: Additional notes or remarks about the analysis.
- **Relationships:**
 - **generatedBy:** The analysis is performed by a Client device.

4. Client

- **Description:** Represents devices (e.g., desktops) used for local model training and analysis.
- **Attributes:**
 - ClientID: Uniquely identifies each client device.
- **Relationships:**

- **performedBy:** A client performs the analysis for an image.
- **generatedBy:** Links a client to the analysis it generates.

5. LoginHistory

- **Description:** Tracks the login history of users.
- **Attributes:**
 - LoginID: Uniquely identifies each login event.
 - UserID: References the user who logged in.
- **Relationships:**
 - **Logs:** Tracks which user performed the login.

6. Model

- **Description:** Represents machine learning models hosted on the central server.
- **Attributes:**
 - ModelID: Uniquely identifies each model.
 - Version, LastUpdated: Stores the version and last update timestamp of the model.
 - Accuracy, TrainedOn: Indicates model accuracy and training data source.
- **Relationships:**
 - **tracks:** Model updates are recorded in the UpdateLog table.
 - **uses:** Model updates are linked to ClientUpdate.

7. ClientUpdate

- **Description:** Tracks updates sent by client devices to the central server.
- **Attributes:**
 - UpdateID: Uniquely identifies each update.
 - ClientID: Links the update to a specific client.
 - ModelID: Indicates the model being updated.

- UpdateDate, GradientData: Tracks the update timestamp and data gradients sent.
- Status: Indicates whether the update was successful.
- **Relationships:**
 - **updates:** Links the updates back to the central server.

8. UpdateLog

- **Description:** Logs updates and changes made to machine learning models.
- **Attributes:**
 - LogID: Uniquely identifies each log entry.
 - ModelID: Links the log entry to a specific model.
 - ClientID: Tracks which client contributed to the update.
 - Timestamp, Details: Logs the date and details of the update.
- **Relationships:**
 - **tracks:** Maintains a history of all updates to the model.

9. Server

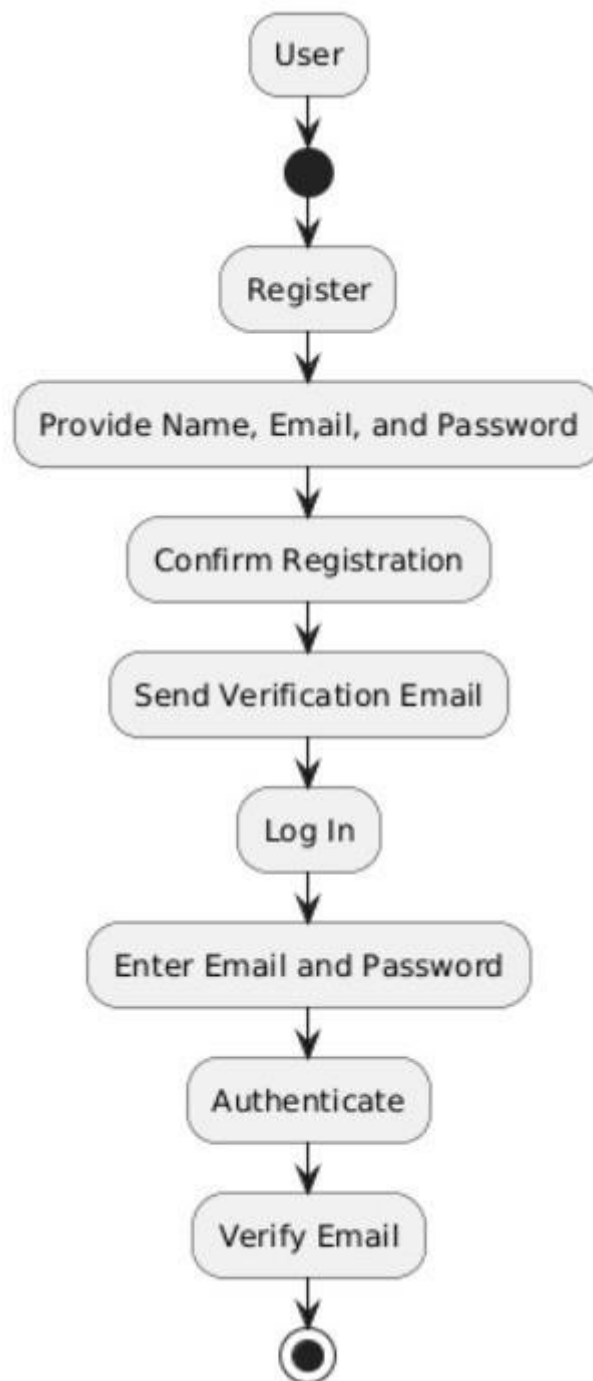
- **Description:** Represents the central server that aggregates updates and distributes the base model.
- **Attributes:**
 - Name, Location: Metadata about the server.
 - CurrentModelID: Links the server to the currently active model.
 - RetryStatus: Tracks whether retry mechanisms are triggered for updates.
- **Relationships:**
 - **distributes:** The server distributes the base model to client devices.
 - **hosts:** The server hosts and aggregates updates from clients.

Summary of Relationships:

- **Users** upload images, and are tracked via their login history.
- **Images** are analyzed, with analysis results linked to specific images and the clients that performed the analysis.
- **Login history** records user activity, tracking timestamps and user IDs.
- **Client devices** perform local model training, generate analysis results, and send updates to the central server.
- **Client updates** contribute gradient data and model updates to improve the global model managed by the central server.
- The **server** aggregates model updates from clients, distributes base models to devices, and tracks model versions using logs.
- The **UpdateLog** records all updates to models, associating them with specific clients and maintaining a history for traceability.

5. Use Case Realization

5.1 USER REGISTER FLOW

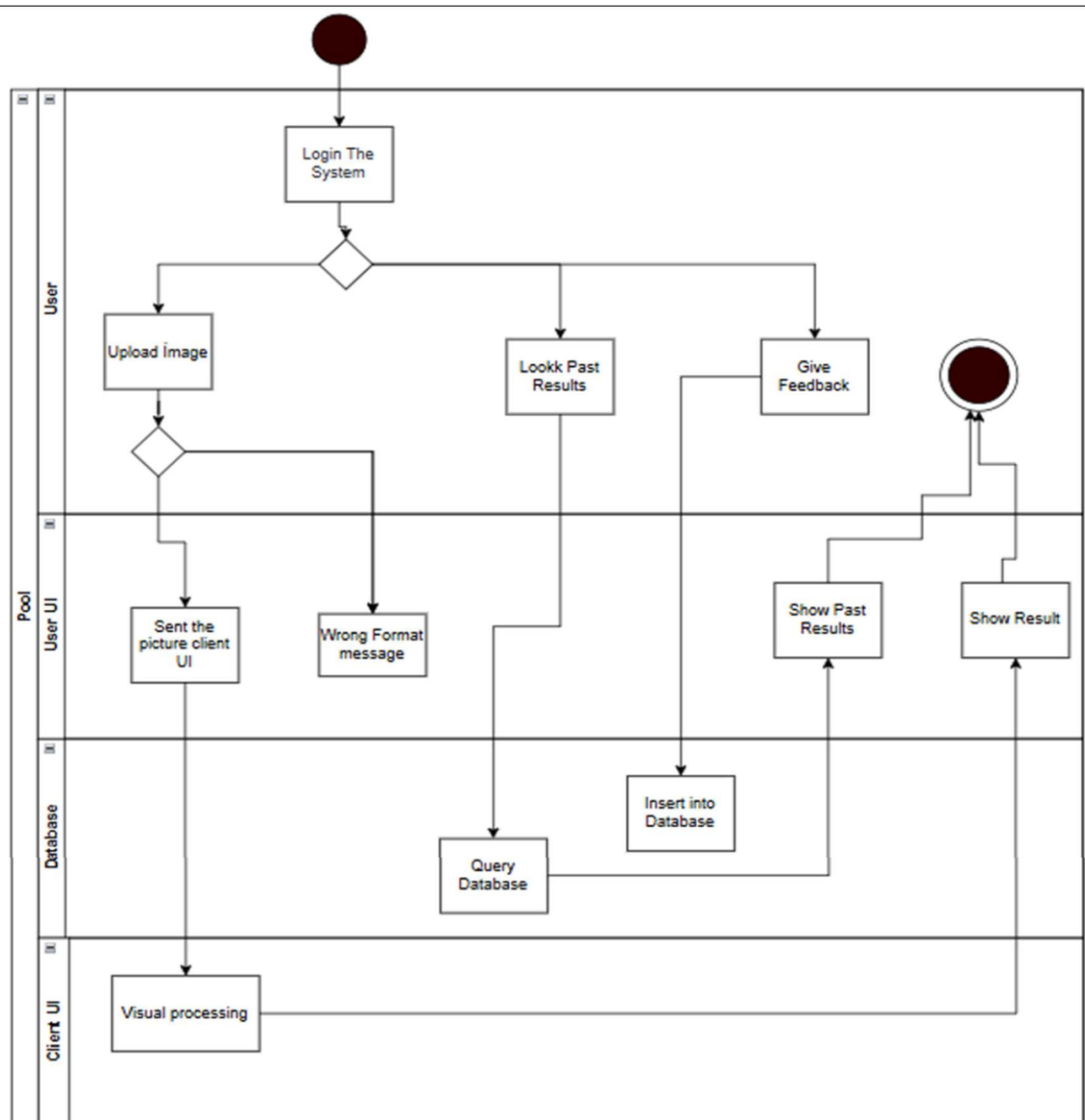


This class diagram represents the user registration and authentication process within a system. The process begins with the user providing necessary details, such as their name, email, and password, during registration. Once these details are submitted, the system confirms the registration and sends a verification email to ensure the provided email address is valid. The user must verify their email through the link or code

provided in the email. After successful verification, the user can log in by entering their email and password. The system then authenticates the user's credentials to ensure they match the records. As an additional security measure, the system verifies the email before granting full access to the user. This sequential process ensures both user convenience and system security by validating user information at multiple stages.

5.2 USER INTERACTION FLOW

Activity Diagram:

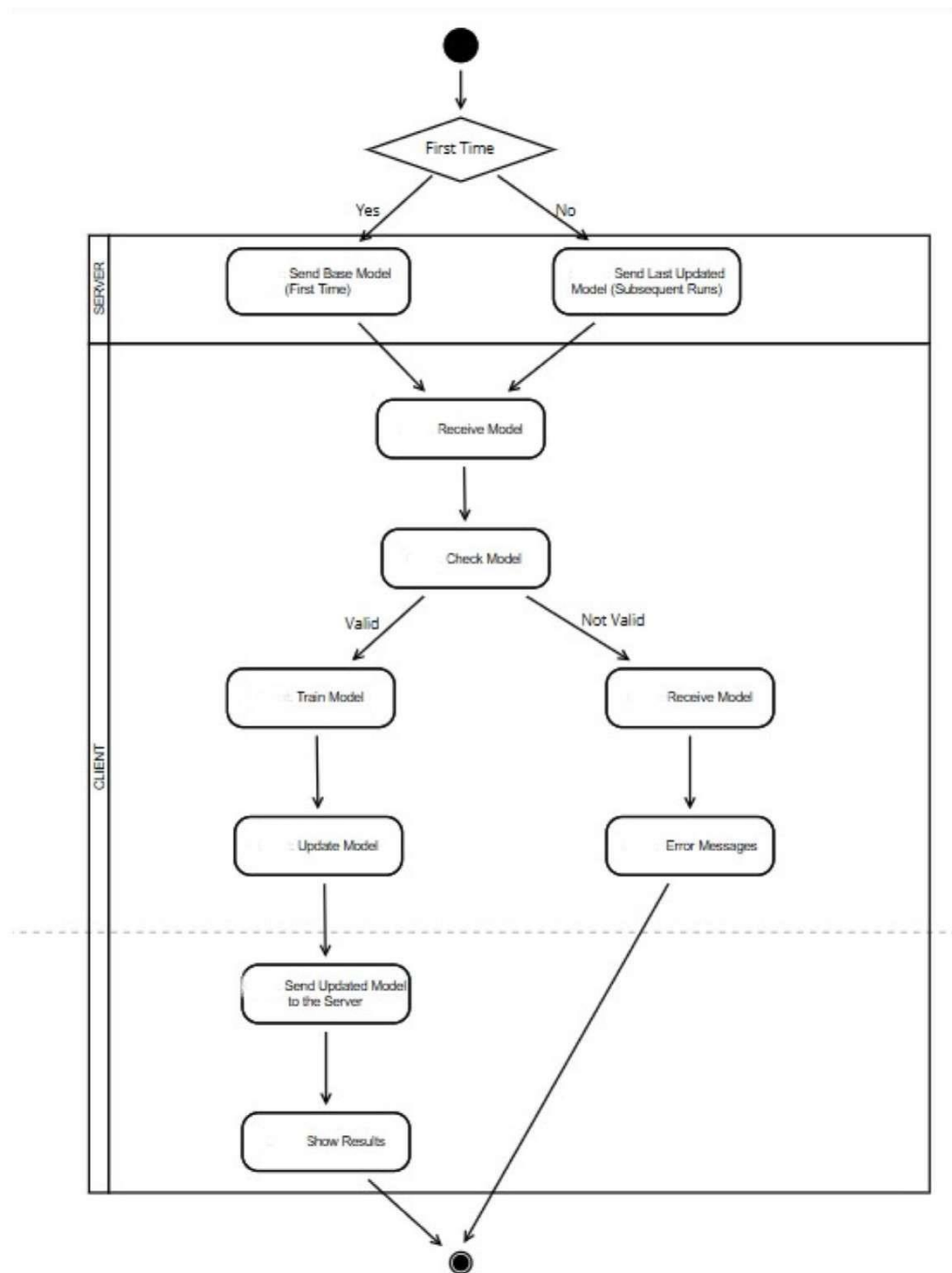


The activity diagram illustrates the workflow of the user's interaction with the system on the homepage. It begins with the user accessing the

homepage after a successful login. The user is prompted to upload a photo of their skin, which triggers a validation process. If the file format is incorrect, an error message is displayed; otherwise, the system proceeds with the image analysis. While the analysis is ongoing, a progress notification is shown. Once completed, the results are displayed on the screen.

The user can then navigate to the "Past Results" section to view a list of previous evaluations, including dates and outcomes. If assistance is needed, the "Help" section offers guidance and FAQs. The activity diagram captures this sequence of actions and decision points, representing the system's dynamic behavior in response to user interactions.

5.3 CLIENT



Overview of the Workflow

1. StartNode:

The process begins with the system being invoked.

2. Decision

First

Time?:

A decision node determines if it is the first invocation of the system:

- Yes → The server sends the Base Model.
- No → The server sends the Last Updated Model from previous runs.

3. ReceiveModel(Client):

The client receives the model (Base or Updated) and proceeds with the process.

4. CheckModel(Client):

The client checks the validity of the received model and training data.

5. Decision-ModelValid?:

A second decision node evaluates if the model is Valid:

- Valid: The client proceeds to train and update the model.
- Not Valid: The client handles errors and may display appropriate error **messages**.

6. Valid Path:

- Train Model: The client trains the model using local data.
- Update Model: The client updates the model with the newly trained data.
- Send Updated Model to the Server: The client uploads the updated model back to the server.
- Show Results: The system displays the training results to the client user.

7. Not Valid Path:

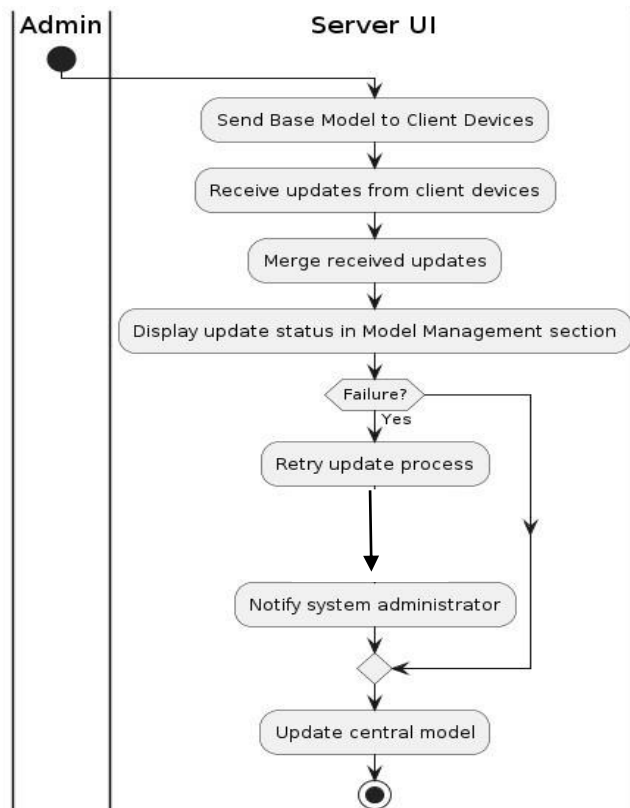
- Receive Model: The client re-attempts to receive the model (optional error-handling loop).
- Error Messages: The system displays error messages to inform the client of issues.

8. EndNode:

The workflow concludes.

5.4 SERVER MANAGEMENT

Activity Diagram:



Flow of Operations

1. **Start**
 - The process begins at the **Start Node**.
2. **Send Base Model to Client Devices**
 - The system sends the base model to the connected client devices.
3. **Receive Updates from Client Devices**
 - The system waits to receive updates from the client devices.
4. **Merge Received Updates**
 - All received updates are merged into a central system.

Parallel Execution Using split

At this point, the flow splits into **two parallel activities**:

1. Update Central Model

- The merged updates are applied to the central model.

2. Display Update Status in Model Management Section

- The status of the updates is displayed in the **Model Management** section.
- A conditional check is performed:
 - **Failure Check:** If an error or failure occurs:
 - **Retry Update Process:** The system retries the update process.
 - **Notify System Administrator:** A notification is sent to the system administrator.

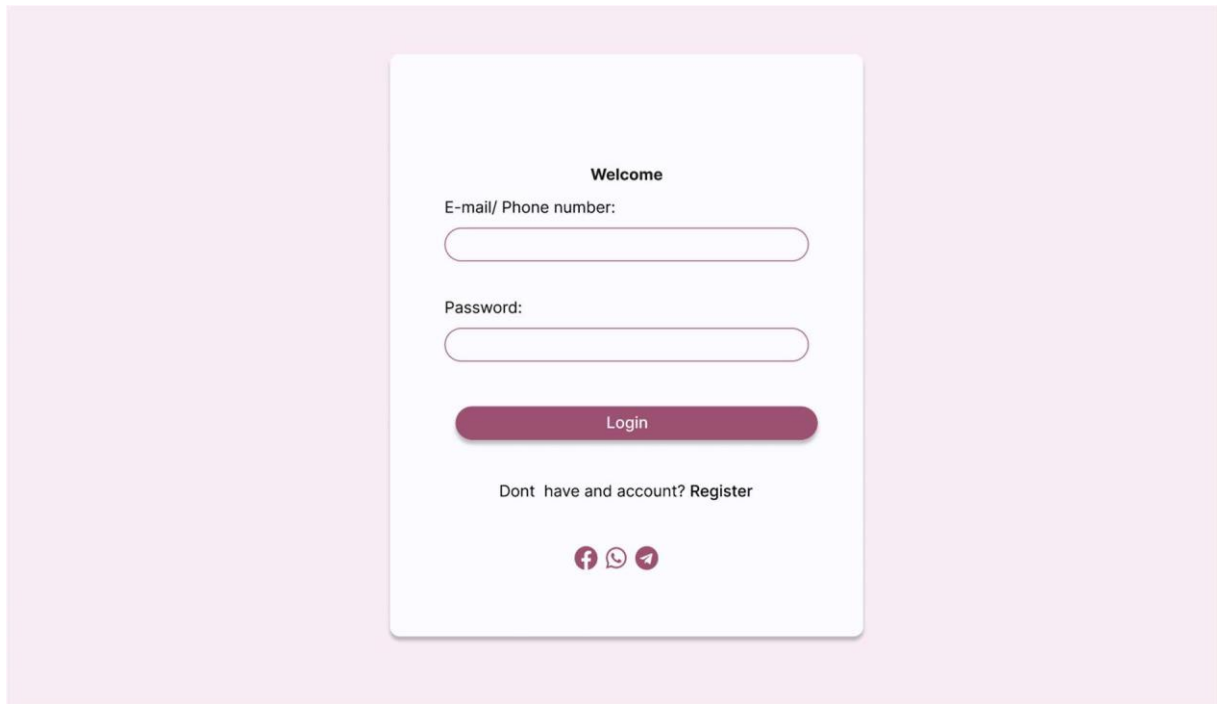
End of Process

- After both parallel activities are completed, the flow merges back.
- The **Stop Node** indicates the end of the process.

6. User Interface Design

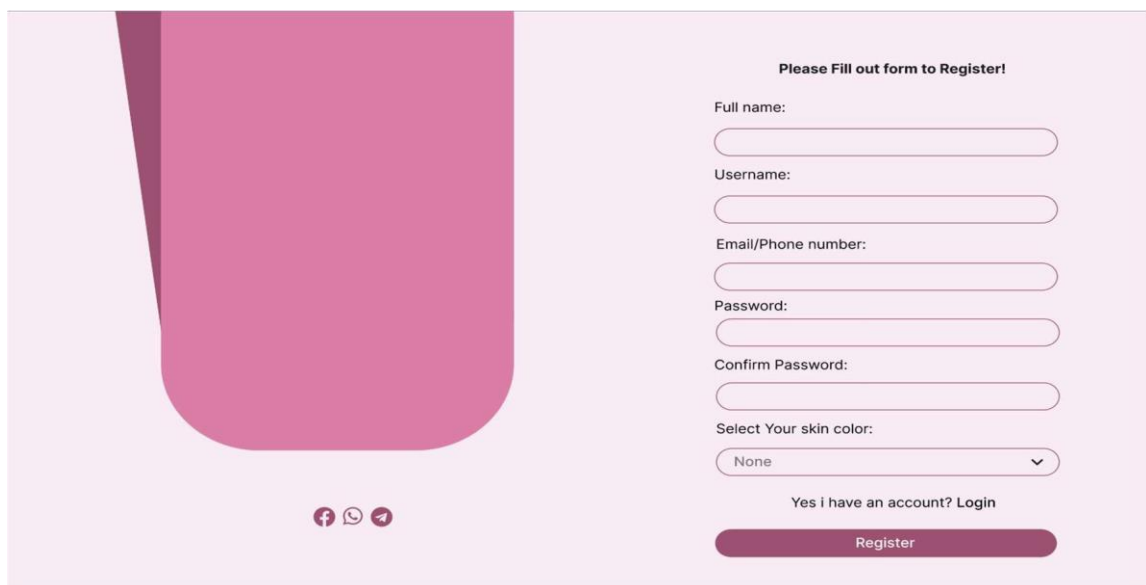
6.1 Login Section:

The user enters their email or phone number and password, then clicks the "Login" button. The backend validates the credentials against the database. If correct, the user is redirected to the homepage; otherwise, an error message appears. homepage; otherwise, an error message appears.



The image shows a login form centered on a light pink background. The form is a white rounded rectangle with a subtle shadow. At the top, it says "Welcome". Below that is a label "E-mail/ Phone number:" followed by a white rounded input field. Underneath is a label "Password:" followed by another white rounded input field. A dark pink rounded button with the text "Login" is positioned below the password field. Below the button, the text "Dont have and account? Register" is displayed. At the bottom of the form, there are three small circular icons for Facebook, WhatsApp, and Telegram.

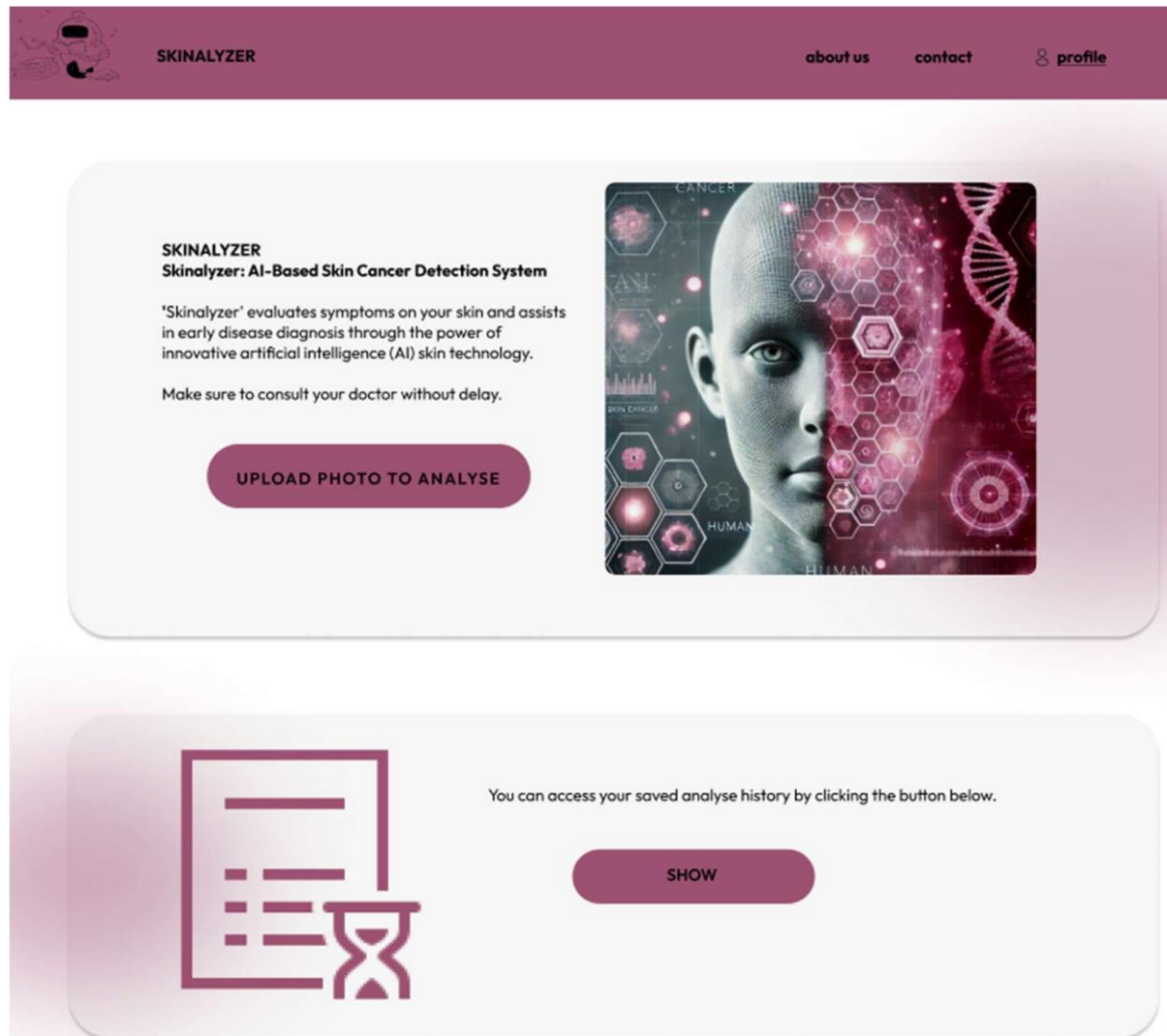
Register Section: The user provides their first name, last name, password, and skin tone, selecting the latter for skin cancer awareness. Upon clicking "Register," the backend checks if the email/phone number is already registered and verifies the password match. If valid, the user is added to the database and redirected to log in; otherwise, an error message



The image shows a registration form on a light pink background. On the left, there is a large, abstract pink graphic. The form is on the right, titled "Please Fill out form to Register!". It contains several input fields: "Full name:", "Username:", "Email/Phone number:", "Password:", "Confirm Password:", and "Select Your skin color:". The skin color field is a dropdown menu currently showing "None". Below these fields is a dark pink rounded button labeled "Register". Above the button, the text "Yes i have an account? Login" is visible. At the bottom left of the form area, there are three small circular icons for Facebook, WhatsApp, and Telegram.

appears.

Homepage: After accessing our homepage, you can view your past results, access your profile and information about us, or upload a new photo to receive the closest diagnosis and explanation.



Upload Photo Button: By clicking this button, you can drag and drop your photo into the pop-up that appears.

SKINALYZER

Skinalyzer: AI-Based Skin Cancer Detection System

'Skinalyzer' evaluates symptoms on your skin and assists in early disease diagnosis through the power of innovative artificial intelligence (AI) skin technology.

Make sure to consult your doctor without delay.




UPLOAD
PHOTO

x



New Result View: You can view the uploaded photo, the AI's predicted diagnosis along with its percentage, and the description of the diagnosis. If you wish, you can click the "Show" button below to view all your results.

 SKINALYZER

[about us](#) [contact](#) [profile](#)

Photo that you uploaaded:



As a result of AI analysis, the uploaded photo has been identified as potentially being melanoma with a 94.5% probability.

Melanoma is a serious type of skin cancer that starts in melanocytes, the cells responsible for producing melanin, which gives skin its color. It often appears as a new or unusual mole, or changes in an existing one. Melanomas are typically asymmetrical with irregular borders, multiple colors, and can grow larger than 6mm. If detected early, melanoma is treatable, but it can spread to other parts of the body, making early diagnosis crucial. Regular skin checks and attention to any changes are key to prevention and early treatment.




You can access your saved analyse history by clicking the button below.

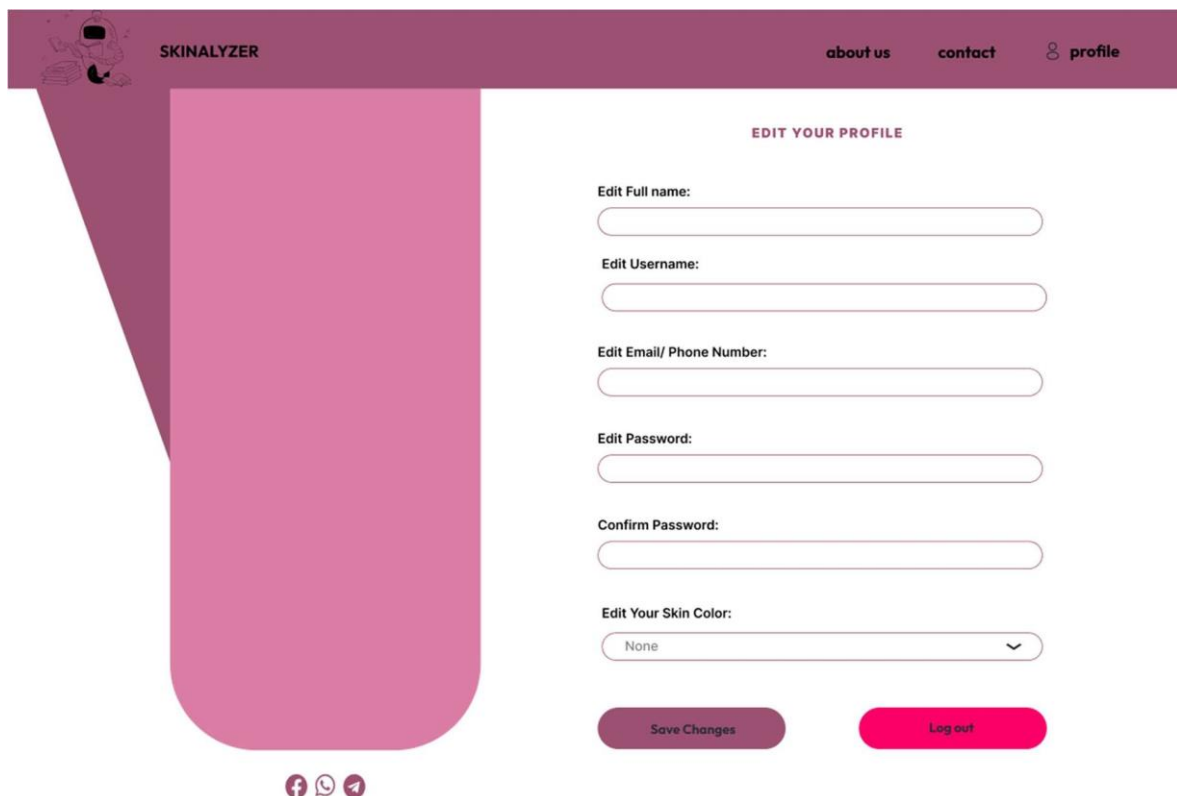
SHOW

Show All Result: On this page, you can find every photo you previously uploaded, along with its diagnosis, percentage, description, and the date it was uploaded.



Your Photo:	Diagnosis:	Definition:	Date:
	Melonama	Melanoma is a serious type of skin cancer that starts in melanocytes, the cells responsible for producing melanin, which gives skin its color.	11.10.2023

Profile: Finally, you can access your profile to update the information you provided during registration or log out.

The image shows the 'EDIT YOUR PROFILE' page in the SKINALYZER application. The header bar is identical to the previous image. On the left side, there is a large, vertical, rounded pink rectangle intended for a profile picture. To the right of this rectangle is a form titled 'EDIT YOUR PROFILE'. The form contains several input fields: 'Edit Full name:', 'Edit Username:', 'Edit Email/ Phone Number:', 'Edit Password:', 'Confirm Password:', and 'Edit Your Skin Color:'. The 'Edit Your Skin Color:' field is a dropdown menu currently showing 'None'. At the bottom of the form are two buttons: 'Save Changes' (dark purple) and 'Log out' (pink). Below the pink placeholder rectangle, there are three small social media icons: Facebook, WhatsApp, and Telegram.