# Project Report

For

**PCause:** PCOS detection system based on deep learning model using ultrasound images.

09-05-2025

Prepared by

| Specialization | SAP ID | Name |
|---|---|---|
| AIML Hons. | 500094127 | Lakshay Agarwal |
| AIML Hons. | 500090912 | Charu Gupta |

AI Cluster
School Of Computer Science
UNIVERSITY OF PETROLEUM & ENERGY STUDIES,
DEHRADUN- 248007. Uttarakhand

# CANDIDATE’S DECLARATION

We hereby certify that the project work entitled “PCause” in partial fulfillment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in Artificial Intelligence and Machine Learning and submitted to the Department of Systemics, School of Computer Science, University of Petroleum & Energy Studies, Dehradun, is an authentic record of our work carried out during a period from January 2025 to May 2025 under the supervision of Ms. Sugandha Sharma.

The matter presented in this project has not been submitted by us for the award of any other degree of this or any other University.

Lakshay Agarwal (500094127)

Charu Gupta (500090912)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: 09/05/2025

Ms. Sugandha Sharma

Project Mentor

# ACKNOWLEDGEMENT

We wish to express our deep gratitude to our guide Dr. Kamakshi Rautela, for all advice, encouragement and constant support she has given us throughout our project work. This work would not have been possible without her support and valuable suggestions. We sincerely thank our respected Dr Anil Kumar, Head Department of SOCS, for his great support in doing our project in PCause.

We are also grateful to Dean SoCS UPES for giving us the necessary facilities to carry out our project work successfully. We also thank our Course Coordinator, Mr. Kunwar Pankaj Siddharth and our Activity Coordinator Dr Vidyanand Mishra for providing timely support and information during the completion of this project.

We would like to thank all our friends for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our parents who have shown us this world and for every support they have given us.

| **SAP ID** | **Name** |
|---|---|
| 500094127 | Lakshay Agarwal |
| 500090912 | Charu Gupta |

# ABSTRACT

Polycystic Ovary Syndrome (PCOS) is a complex hormonal disorder affecting millions of women globally, often leading to infertility, metabolic issues, and long-term health complications. Early and accurate diagnosis is critical to mitigating its impact, yet traditional diagnosis via ultrasonography remains subjective, operator-dependent, and prone to inconsistency. To address these limitations, this project presents PCause, an AI-powered PCOS detection system that leverages deep learning and image processing to analyze ovarian ultrasound scans and predict the presence of PCOS with high accuracy.

The core of PCause is a deep convolutional neural network (CNN) based on DenseNet-121, a powerful architecture known for its feature reuse and computational efficiency. This model is trained on a labeled dataset of ovarian ultrasound images sourced from public repositories. The dataset includes 3,856 grayscale images categorized as either “infected” (PCOS-positive) or “not infected” (healthy). To address the relatively small dataset size and class imbalance, data augmentation techniques—such as flipping, zooming, and rotation—are employed, along with synthetic image generation using Generative Adversarial Networks (GANs). Additionally, Principal Component Analysis (PCA) is used to reduce the dimensionality of image feature vectors, ensuring the model focuses on the most significant components during training.

The images are pre-processed to a fixed resolution of 224×224 pixels and normalized for uniformity. The DenseNet-121 model, initially pretrained on ImageNet, is fine-tuned for binary classification using sigmoid activation in the final layer. Training is conducted using binary cross-entropy loss and the Adam optimizer, with techniques like dropout and early stopping to prevent overfitting. The model's performance is validated using metrics such as accuracy, precision, recall, and F1 score on unseen test data.

The trained model is then integrated into a cross-platform application developed using Flutter, enabling users to upload ultrasound images, receive predictions in real time, and view diagnostic insights via an interactive dashboard. The backend is powered by Firebase, providing secure authentication, data storage, and real-time cloud functions to process predictions. The app supports both healthcare professionals and patients, offering a streamlined user experience for PCOS screening and monitoring.

Through this system, PCause aims to automate and standardize PCOS detection, thereby reducing diagnostic variability and empowering early intervention. It serves as a decision-support tool for clinicians and a personal health assistant for users. By combining state-of-the-art AI techniques with accessible application design, this project contributes a scalable, non-invasive, and cost-effective solution to a widespread healthcare challenge.

PCause demonstrates how artificial intelligence can be effectively applied to medical imaging for the detection of PCOS. It bridges the gap between complex diagnostic procedures and practical, user-friendly tools. The system’s robust performance, usability, and scalability make it a promising innovation for improving women’s reproductive healthcare and advancing the field of AI in medicine.

# Table of Contents

General Instructions:

1. Font should be Time new Roman 12
2. Main heading should be All Capital with Times New Roman 14
3. Sub-Heading should be Times new roman 12, Underline
4. Line gap should be 1.15
5. Justified alignment should be used for all text
6. Content inside a table should be Times New Roman 10
7. Caption for both Table and Figure should be Times New Roman 11
8. Add Source for all Images used.

| | | |
|---|---|---|
| 1 | INTRODUCTION | |
| | 1.1 Purpose of the Project | Describe the scope of this project by stating and justifying the problem statement of the project. Present will clear motivation to execute the project. |
| | 1.2 Target Beneficiary | Identify the prime beneficiaries of the project. |
| | 1.3 Project Scope | Provide a short description of area of application of the software, include relevant benefits, objectives, and goals. State clearly the requirement and deliverables of the project. |
| | 1.4 References | List all documents or Web addresses to which this SRS refers. |
| 2 | PROJECT DESCRIPTION | |
| | 2.1 Reference Algorithm | State the reference algorithm for the project and identify the required data structure (**Mandatory for Minor1**) Or/Add design algorithm justifying the methodology of the project |
| | 2.2 Characteristic of Data | Present with the characteristic of the dataset used for the project. Provide the primary and secondary source of the data, along with sampling techniques. Explain the statistical method used for data processing (**if any**). |
| | 2.3 SWOT Analysis | Present with a justification to support your project. |
| | 2.4 Project Features | Summarize the major features the product contains or the significant functions that it performs or lets the user perform. (Level 2 USE Case diagram) |
| | 2.5 User Classes and Characteristics | Identify the various user classes that you anticipate will use this product. |
| | 2.6 Design and Implementation Constraints | Present hardware boundary conditions (timing requirements, memory requirements); interfaces to other applications; specific technologies, and tools to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards. |
| | 2.7 Design diagrams | Present all the required Diagram (USE –Case, Class Diagram, Activity, Sequence, Data Flow diagram and State Diagram. (Major project should include Collaboration and Deployment Diagram too) |

| | | |
|---|---|---|
| | 2.8 Assumption and Dependencies | List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. Also identify any dependencies the project has on external factors. |
| 3 | SYSTEM REQUIREMENTS | |
| | 3.1 User Interface | Define the software components for which a user interface is needed. |
| | 3.2 Software Interface | Describe the connections between modules. Describe the services needed and the nature of communications. Describe detailed application programming interface protocols. |
| | 3.3 Database Interface | Explain the Database management system used |
| | 3.4 Protocols | Describe the requirements associated with any protocol deployed in the project. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms |
| 4 | NON-FUNCTIONAL REQUIREMENTS | |
| | 4.1 Performance requirements | If there are performance requirements for the product under various circumstances, state them. Specify the timing relationships for real time systems. State performance requirements for individual functional requirements or features |
| | 4.2 Security requirements | Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define authentication, verification and validation of the system. Refer to any external policies or regulations containing security issues that affect the product. |
| | 4.3 Software Quality Attributes | Explain: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. |
| 5 | Other Requirements | Define any other requirements not covered elsewhere in the SRS. |
| Appendix A: Glossary | | Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. |
| Appendix B: Analysis Model | | Pertinent analysis models used for this project |
| Appendix C: Issues List | | This is a dynamic list of the open requirements issues. |

# INTRODUCTION

Polycystic Ovary Syndrome (PCOS) is a prevalent endocrine disorder affecting a significant percentage of women worldwide. It causes symptoms such as ovarian cysts, hormonal imbalance, irregular menstrual cycles, and increased risk of metabolic and cardiovascular issues. Early diagnosis of PCOS is crucial for managing its effects and improving long-term health outcomes. However, traditional diagnosis often relies on manual interpretation of ultrasound scans by specialists, which can be subjective and time-consuming.

Ultrasonography is one of the primary modalities for detecting the polycystic ovaries associated with PCOS, as it can reveal characteristic ovarian morphology. Nevertheless, ultrasound images are prone to noise and artifacts, and interpretation depends heavily on the operator's experience. Deep learning and computer vision techniques have shown great promise in automating medical image analysis, achieving high accuracy in tasks such as disease diagnosis from images. By leveraging deep convolutional neural networks and advanced image processing, artificial intelligence can increase the speed and consistency of medical diagnoses.

The PCause system is designed to harness these advances for PCOS detection. It uses modern image processing techniques — such as Generative Adversarial Networks (GANs) to synthetically augment the training data and Principal Component Analysis (PCA) to select the most relevant features — before feeding images into a convolutional neural network for classification. By automating PCOS detection, PCause aims to provide healthcare professionals and patients with a reliable, efficient tool that supports early intervention and personalized treatment strategies.

## 1.1 Purpose of the Project

The primary objective of the PCause project is to develop a user-friendly, robust, and intelligent application that automates the detection of Polycystic Ovary Syndrome (PCOS) from ovarian ultrasound images using advanced deep learning techniques. PCOS is a hormonal disorder that affects a significant percentage of women of reproductive age and is one of the leading causes of infertility. However, the process of diagnosing PCOS is often time-consuming, prone to human error, and varies based on the expertise of the radiologist. Traditional diagnostic methods rely heavily on manual interpretation of ultrasound images, which can lead to inconsistent results, especially in resource-limited clinical settings.

To address these issues, PCause introduces an AI-driven solution that leverages deep convolutional neural networks—specifically, the DenseNet-121 architecture—to analyze ovarian ultrasound images and detect PCOS with high precision. The system is designed to minimize subjectivity, increase diagnostic speed, and provide consistent and reproducible results, thereby supporting healthcare professionals in making informed decisions.

Beyond just a model, PCause encompasses the development of a fully functional cross-platform mobile application built using Flutter. The application allows users—both medical professionals and patients—to interact seamlessly with the underlying AI model. The user interface has been designed with simplicity and clarity in mind, offering features such as:

- **Secure User Authentication**: Ensuring patient data is protected, the system uses Firebase-based login and registration mechanisms.
- **Image Upload Interface**: Users can upload ovarian ultrasound images directly through the app for analysis.
- **Real-time Prediction Output**: Once an image is uploaded, the AI model runs inference in real-time and displays whether the scan indicates PCOS or not.
- **Historical Data Tracking**: The app stores and visualizes past prediction results, allowing users to track changes or trends over time.
- **Graphical Dashboards**: These dashboards provide insightful visual analytics such as the count of infected vs. not-infected cases, helping users understand statistical patterns.

Another major purpose of the project is to make PCOS screening accessible to a wider population, especially in regions where expert radiologists are not readily available. By democratizing the diagnostic process and embedding it into a mobile platform, PCause serves as a valuable tool for preliminary screening and self-monitoring.

Moreover, the application is designed with scalability and extendability in mind. The modular architecture allows future integration of additional features like multilingual support, doctor consultations, or integration with wearable health devices. This ensures that PCause not only serves the current need of automated PCOS detection but can also evolve into a comprehensive reproductive health management platform.

## 1.2 Target Beneficiary

The **PCause** application is designed with a wide range of stakeholders in mind, each benefiting uniquely from its features and capabilities. The system aims to address existing gaps in PCOS diagnosis and monitoring through an AI-powered, user-friendly interface. Its target beneficiaries are broadly categorized into **primary** and **secondary** user groups:

**Primary Beneficiaries**

1. **Healthcare Professionals**
    - **Gynecologists, Radiologists, and Sonographers** are among the foremost users who benefit from PCause. These professionals are often responsible for diagnosing PCOS through clinical symptoms, blood reports, and ultrasound images.
    - By using PCause, clinicians can reduce their dependency on subjective visual interpretation and instead rely on objective, AI-generated predictions. This

helps **streamline their workflow**, **reduce diagnostic variability**, and **improve clinical decision-making**.

  - The model's consistent and rapid predictions allow them to handle more cases with higher accuracy, especially in busy clinical environments.

2. **Patients**

  - Women of reproductive age—especially those showing symptoms like irregular menstrual cycles, excessive androgen levels, or fertility issues—form the core patient group that will benefit from the PCause system.
  - Through early and accurate detection of PCOS, patients can seek timely medical advice and initiate personalized treatment regimens. The application empowers them to take charge of their health with accessible and real-time insights.
  - Additionally, the historical records and prediction history provide a transparent view of the patient's journey, which is valuable for both self-monitoring and future consultations.

**Secondary Beneficiaries**

1. **Medical Researchers**

  - With access to anonymized diagnostic data and analytics, researchers can explore emerging trends and patterns in PCOS cases across different demographics.
  - This enables deeper insights into prevalence rates, co-morbidities, and the effectiveness of early intervention strategies, contributing to the broader field of women's health research.

2. **Educators and Trainees**

  - Medical educators and students in gynecology, radiology, and machine learning can use PCause as a **training tool**. The combination of image-based analysis and real-time feedback provides a practical supplement to traditional learning methods.
  - The visual dashboards and prediction logs serve as **educational assets** for teaching pattern recognition in ovarian scans and understanding the real-world applications of AI in medicine.

3. **Healthcare Institutions and NGOs**

- Clinics, diagnostic centers, and non-profit organizations working in women's health can adopt PCause as a **cost-effective solution** to scale early diagnosis programs in rural or underserved areas.
- It also aligns well with preventive healthcare initiatives and community awareness campaigns around reproductive disorders.

## 1.3 Project Scope

The **PCause** project involves the complete development, training, deployment, and packaging of an intelligent, cross-platform software system focused on the **automated detection of Polycystic Ovary Syndrome (PCOS)** using ultrasound images. It brings together deep learning, cloud services, and intuitive user interface design to create a unified solution that can support both healthcare providers and patients in clinical and remote settings.

**Core Components of the Project Scope**

1. **Cross-Platform Application Development**
    - The application has been developed using **Flutter**, enabling deployment across both **mobile (Android/iOS)** and **web** platforms. This approach ensures broad accessibility and consistency in user experience.
    - The app offers a clean, responsive interface that caters to both medical professionals and patients with varying degrees of technical expertise.
2. **Backend Infrastructure**
    - The backend is powered by **Firebase**, which provides secure and scalable solutions for **authentication**, **database management**, and **cloud storage**.
    - Firebase Authentication ensures that only authorized users can access the system, protecting sensitive health data.
    - User data, uploaded ultrasound images, and diagnostic results are stored securely in **Cloud Firestore**, while files are handled using **Firebase Storage**.
3. **Deep Learning Model Integration**
    - A pre-trained **DenseNet-121** convolutional neural network model is integrated into the application for performing **binary classification** (PCOS / Non-PCOS) based on ovarian ultrasound scans.
    - The model has been trained on a labeled dataset of ultrasound images and provides fast, reliable predictions with strong performance across various evaluation metrics (accuracy, precision, recall, F1-score).

    - The model has been deployed in a serverless architecture, enabling real-time inference upon image upload.

4. **User Interface Modules**
    - The application comprises multiple, logically structured user-facing screens:
        - **Login/Signup Pages**: For secure account creation and access.
        - **Image Upload Screen**: Allows users to upload ultrasound images for analysis.
        - **Prediction Result Page**: Displays real-time output from the integrated model.
        - **User Profile and History Section**: Shows past predictions with dates and outcomes.
        - **Interactive Dashboard**: Provides visual summaries and statistical insights into prediction history and usage metrics.
5. **Visualization and Reporting**
    - Users can access a **dashboard** that presents data using interactive **charts and graphs**, enabling pattern recognition over time.
    - This visual analytics module enhances engagement by helping users understand their health trends and empowers medical professionals with historical context for better consultations.
6. **Deliverables**
    - A complete, functional **mobile and web application**.
    - The **trained PCOS classification model** ready for deployment.
    - Supporting documentation including:
        - **User Guide** for patients and doctors.
        - **Technical Notes/Developer Guide** for understanding the model integration and app architecture.

**Project Impact and Use Cases**

- By combining deep learning with a usable mobile platform, **PCause offers a significant advancement in women's health diagnostics**.

- The tool enhances clinical accuracy and efficiency, while enabling early detection and self-monitoring outside hospital settings.
- It is particularly suited for **telemedicine** setups, **low-resource environments**, and **ongoing treatment follow-up**, thus making PCOS management more proactive and inclusive.

## 1.4 References

- Choudhari, A., "PCOS detection using ultrasound images", Kaggle Dataset (2019).
- S. Aslam & T. F. Rabie, "Principal Component Analysis in Image Classification: A review," 2023.
- P. Chitra *et al.*, "Classification of Ultrasound PCOS Image using Deep Learning based Hybrid Models," ICEARS (2023).
- Suha & Islam, "Extended machine learning technique for PCOS detection," *Scientific Reports* (2022).
- Aggarwal, Mittal & Battineni, "Generative adversarial network: An overview," *IJIM Data Insights* (2021).
- Sumathi *et al.*, "Study and detection of PCOS related diseases using CNN," IOP Conf. Ser.: Materials Sci. Eng. (2021).
- Aslam, S., & Rabie, T. F. (2023, February). Principal Component Analysis in Image Classification: A review. In 2023 Advances in Science and Engineering Technology International Conferences (ASET) (pp. 1-7). IEEE.
- Chitra, P., Srilatha, K., Sumathi, M., Jayasudha, F. V., Bernatin, T., & Jagadeesh, M. (2023, March). Classification of Ultrasound PCOS Image using Deep Learning based Hybrid Models. In 2023 Second International Conference on Electronics and Renewable Systems (ICEARS) (pp. 1389-1394). IEEE.
- Suha, S. A., & Islam, M. N. (2022). An extended machine learning technique for polycystic ovary syndrome detection using ovary ultrasound image. Scientific Reports, 12(1), 17123.
- Aggarwal, A., Mittal, M., & Battineni, G. (2021). Generative adversarial network: An overview of theory and applications. International Journal of Information Management Data Insights, 1(1), 100004.
- Sumathi, M., Chitra, P., Prabha, R. S., & Srilatha, K. (2021, February). Study and detection of PCOS related diseases using CNN. In IOP Conference Series: Materials Science and Engineering (Vol. 1070, No. 1, p. 012062). IOP Publishing.

- Lin LH, Baracat MC, Gustavo AR, et al. Androgen receptor gene polymorphism and polycystic ovary syndrome. Int J Gynaecol Obstet. 2013;120:115–118. doi: 10.1016/j.ijgo.2012.08.016

- Hai Bai, Huanhuan Ding, Mingming Wang. Polycystic Ovary Syndrome (PCOS): Symptoms, Causes, and Treatment. Clin. Exp. Obstet. Gynecol. 2024, 51(5), 126.

- Chhabra, M., Kumar, R. (2022). A Smart Healthcare System Based on Classifier DenseNet 121 Model to Detect Multiple Diseases. In: Marriwala, N., Tripathi, C., Jain, S., Kumar, D. (eds) Mobile Radio Communications and 5G Networks. Lecture Notes in Networks and Systems, vol 339. Springer, Singapore.

# PROJECT DESCRIPTION

## 2.1 Reference Algorithm

The core algorithm in PCause is a deep convolutional neural network based on DenseNet-121, a model known for high accuracy and efficient parameter usage. We initialize DenseNet-121 with pretrained ImageNet weights and replace its final fully-connected layers with new layers suited for binary classification (PCOS vs. No PCOS). A Global Average Pooling layer is added to reduce overfitting, followed by a dropout layer for regularization. The output layer uses a sigmoid activation function for the final prediction.

**Data preprocessing and augmentation:** All input images are resized to 224×224 pixels and normalized. Data augmentation is applied (rotations, flips, zooms, shifts) to expand the effective training dataset. Additionally, GAN-generated synthetic images supplement the training set, improving model generalization. Principal Component Analysis (PCA) is applied to the flattened feature vectors to reduce dimensionality and highlight the most significant features before classification.

**Training procedure:** The model is compiled with a suitable optimizer (e.g., Adam) and loss function (binary cross-entropy for two-class output). Training is performed on the augmented dataset with a train-validation-test split. Early stopping is used to prevent overfitting, and performance is monitored via accuracy, precision, recall, and F1 score on the validation set.

**Deployment:** Once trained, the DenseNet-121 model is exported (e.g., via TensorFlow SavedModel) and integrated into the application backend. The app sends user-uploaded images to this model (running on a server or cloud function), which returns a prediction (PCOS or healthy) along with a confidence score.

**Algorithmic Steps:**

1. **Input:** Receive a preprocessed ultrasound image.

2. **Feature Extraction:** The image is fed through DenseNet-121 up to the Global Average Pooling layer, producing feature maps.

3. **Classification:** The feature maps pass through the new Dense layers with ReLU activation, dropout, and finally a sigmoid unit.

4. **Output:** Generate a probability score for PCOS presence; threshold this score to decide the final classification.

5. **Post-processing:** Record the prediction and confidence, and optionally compute additional statistics (e.g., feature importance via Grad-CAM).

This pipeline ensures robust analysis of input images and transparent flow of data from raw input to prediction output.

## 2.2 Characteristic of Data

The project uses a publicly available ultrasound image dataset specifically curated for PCOS research. The dataset contains **3,856** grayscale images of ovaries. Each image is labeled as either **“infected”** (indicating PCOS) or **“notinfected”** (healthy). Images come in two main folders (train and test), with roughly half the images in each. The label distribution is roughly balanced between the two classes.

Images vary in native resolution; all are converted to 224×224 pixels for model input. The grayscale intensity values are normalized. No identifying metadata (e.g., patient age or ID) is included, as the focus is purely on image data.

Given the limited size of medical datasets, data augmentation is crucial. We apply random rotations, flips, and zooms during training. Additionally, Generative Adversarial Networks (GANs) are used to synthesize realistic ovarian images, effectively expanding the dataset. This augmentation helps the model generalize to variations in real-world images.

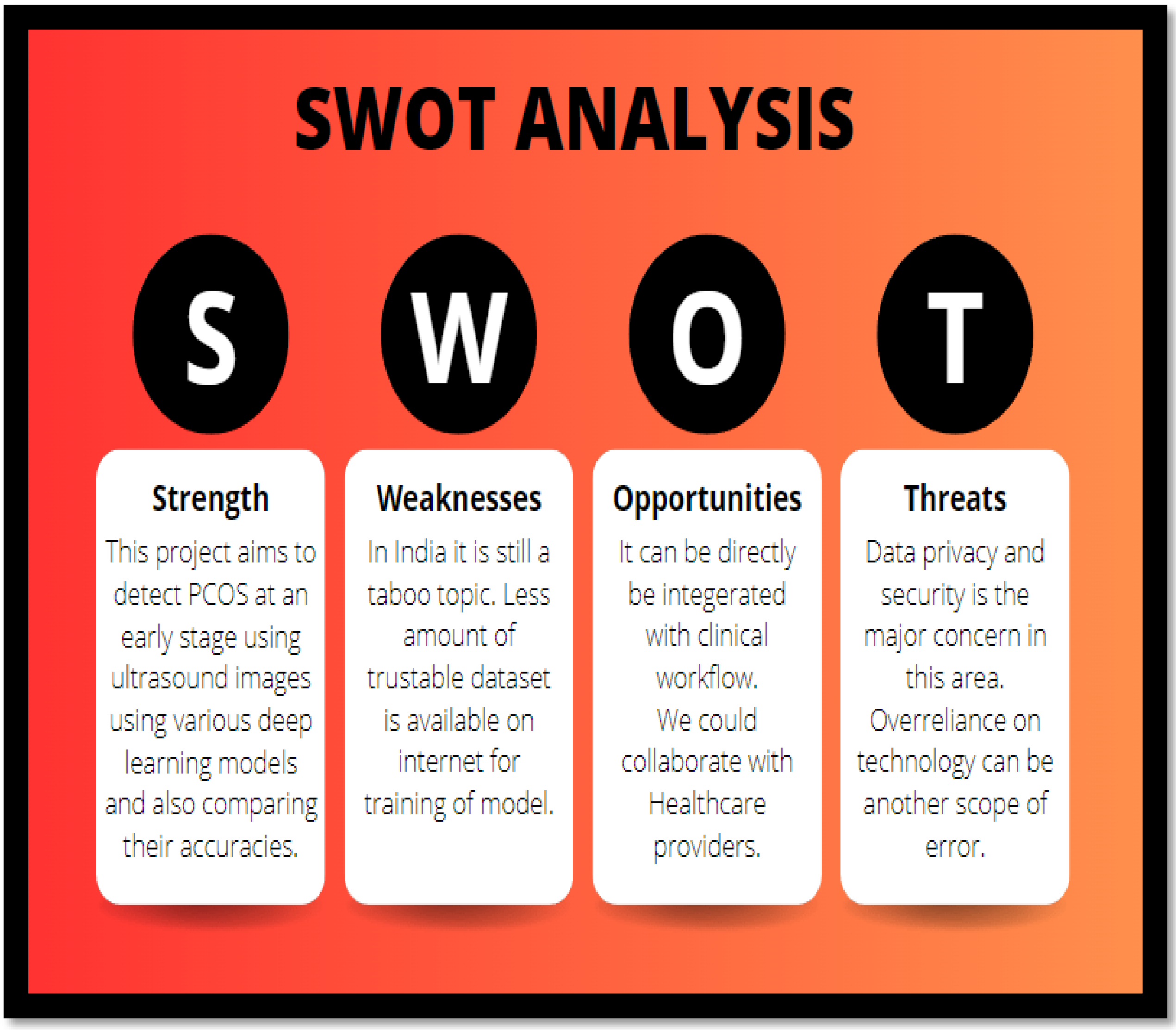
**Data Sources and Quality:** The images were sourced from the public Kaggle repository “PCOS detection using ultrasound images”. We assume the provided labels are correct. Since the data originates from multiple clinics, the image quality and acquisition settings may vary; this diversity aids robustness but may introduce noise. No personal data accompanies the images, simplifying privacy concerns.

**Dataset Usage:** For training, the data is split into training (≈1924 images) and testing (≈1932 images) sets. During development, a validation split is taken from the training set for hyperparameter tuning. All processing (augmentation, PCA, normalization) is applied consistently across sets. After training, the model’s performance is evaluated on the held-out test set to estimate real-world accuracy.

## 2.3 SWOT Analysis

- **Strengths:**
    - *Automation and Accuracy:* By leveraging deep learning, PCause provides consistent and objective PCOS diagnoses, reducing human error.
    - *Early Detection:* Automated analysis enables screening large populations quickly, supporting early intervention.
    - *User-Friendly Interface:* The mobile/web app design and interactive dashboards make advanced diagnostics accessible even to non-experts.
    - *Scalability:* Cloud-based services allow the system to scale with demand and integrate updates.

- **Weaknesses:**
    - *Data Limitations:* The model's performance is constrained by the size and diversity of the training set. Rare cases or unusual presentations of PCOS may be missed.
    - *Explainability:* As a neural network, the model's decisions are not easily interpretable by users or clinicians, which may limit trust.
    - *Dependency on Image Quality:* Poor-quality ultrasound images (e.g., blurred or low-contrast) can degrade accuracy.
- **Opportunities:**
    - *Telemedicine Integration:* Growing telehealth trends mean PCause can extend diagnostic services to remote areas.
    - *Expanded Applications:* The core AI and app framework could be adapted for diagnosing other conditions (e.g., endometriosis, ovarian tumors) from imaging data.
    - *Research and Education:* Aggregated anonymous data from the system can aid PCOS research and help train new doctors.
    - *Market Demand:* With PCOS affecting an estimated one in fifteen women globally, there is a significant need for accessible diagnostic tools.
- **Threats:**
    - *Regulatory Hurdles:* Medical software must comply with health regulations (e.g., FDA or CE certification) before clinical use, which can delay deployment.
    - *Data Privacy:* Handling medical images involves strict privacy laws (HIPAA/GDPR); any breach could have legal consequences.
    - *Technical Competition:* Similar AI diagnostic tools are being developed; standing out requires demonstrable accuracy and reliability.
    - *User Acceptance:* Clinicians may be hesitant to rely on AI without clear understanding; overcoming scepticism is a challenge.


SWOT ANALYSIS

S W O T

**Strength**
This project aims to detect PCOS at an early stage using ultrasound images using various deep learning models and also comparing their accuracies.

**Weaknesses**
In India it is still a taboo topic. Less amount of trustable dataset is available on internet for training of model.

**Opportunities**
It can be directly be integerated with clinical workflow. We could collaborate with Healthcare providers.

**Threats**
Data privacy and security is the major concern in this area. Overreliance on technology can be another scope of error.

Figure 1: SWOT Diagram

## 2.4 Project Features

- **Secure User Authentication:** Users (patients and providers) must create an account and log in to access the system. We implement email/password signup and login with validation and error handling (e.g., feedback on invalid credentials).
- **Image Upload and Prediction:** The core feature allows users to upload ovarian ultrasound images. The app supports selecting images from device storage or capturing via camera. Once an image is uploaded, the deep learning model processes it and returns a result indicating the presence or absence of PCOS, along with a confidence score. The result screen displays this information clearly.
- **Interactive Dashboard:** The app includes a dashboard where users can visualize their prediction history. Charts (e.g., pie charts or timelines) show trends such as the

proportion of positive diagnoses or changes in confidence over time. This helps users spot patterns and progress.

- **History/Profile Screen:** Every prediction (with date/time, image thumbnail, and result) is saved in the user's profile. Users can review past results, compare them, and export a summary report to share with their doctor. The profile also stores user details (name, email) and allows password management.

- **Data Export and Sharing:** Users can generate a PDF report of their results or share them securely with healthcare professionals. This feature supports continuity of care, as doctors can review the AI's findings alongside traditional tests.

- **Responsive Design:** The application uses a responsive UI layout, ensuring that it works well on different devices and orientations. All main functionalities are accessible through an intuitive navigation (tabs or menu) system.

These features collectively ensure that the application is not only technically powerful but also practical and user-centric.

## 2.5 User Classes and Characteristics

- **Patients:** Women experiencing symptoms of PCOS (irregular periods, infertility concerns, etc.) who want a quick screening tool. They are assumed to have basic smartphone literacy and expect a simple, non-technical interface. They care primarily about the accuracy and clarity of results, and the privacy of their medical data.

- **Clinicians (Sonographers/Gynecologists):** Medical professionals who perform or analyze ultrasound exams. They use the system to validate their findings, obtain second opinions, or educate patients. They are accustomed to medical software and require more detailed information (e.g., confidence scores, image overlays). They value data security and compliance with medical standards.

- **Administrators/Developers:** Technical users responsible for maintaining the system (e.g., updating the model, managing users). They need administrative interfaces (not part of core app) for monitoring system health and user feedback.

- **Researchers/Educators:** Academics or medical educators who might use the system's data insights. They require exportable data and may interface with the system in a research mode.

Each user class has distinct expectations. The UI must cater to both non-expert patients (simple language, visual cues) and expert clinicians (detailed analytics). Access control ensures that, for example, patients cannot see each other's data.

## 2.6 Design and Implementation Constraints

- **Hardware Constraints:** The target devices are typical smartphones and tablets. These devices have limited CPU, GPU, memory, and battery life. The deep learning inference must either run on a server or use a lightweight model to accommodate these constraints. The design must ensure the app runs smoothly on both low-end and high-end devices.

- **Software/Platform Constraints:** The application is built with Flutter (Dart) for cross-platform compatibility (Android and iOS). Backend services rely on Firebase (Authentication, Firestore, Storage). This choice constrains us to use technologies supported by these platforms. For example, the ML model must be deployable as a web service or via Firebase Cloud Functions. Third-party libraries must be compatible with these environments.

- **Performance Constraints:** Real-time usability imposes constraints on inference time and data transfer. The model should produce a result within a few seconds after image upload. Large image files should be compressed to reduce upload time. The system also has to handle at least modest concurrency if multiple users use it simultaneously.

- **Scalability Considerations:** While the initial user base may be small, the architecture should allow scaling (e.g., Firebase can handle large numbers of users and data). Database and storage choices must accommodate growing data (thousands of images and records) without performance degradation.

- **Regulatory and Security Constraints:** Because the app deals with medical data, it must comply with security and privacy regulations (e.g., encryption, audit trails). Any third-party services used (like Firebase) must meet industry standards.

- **Data Constraints:** The project depends on the availability of quality ultrasound images. If the input images are too noisy or vary widely, additional preprocessing or data filtering must be implemented. Also, since we rely on a public dataset, any limitations or biases in that data affect the system.

- **Implementation Constraints:** The development follows an iterative model. Dependencies include specific versions of software (e.g., Flutter SDK, Python ML libraries). These must be noted so that other developers can reproduce the environment. We assume continuous internet connectivity for cloud operations, although minimal offline functionality (like viewing history) could be supported via Firestore's offline mode.

These constraints guide the design decisions. For example, knowing that device memory is limited informs the choice to perform heavy computation on cloud servers rather than entirely on-device.

## 2.7 Design Diagrams

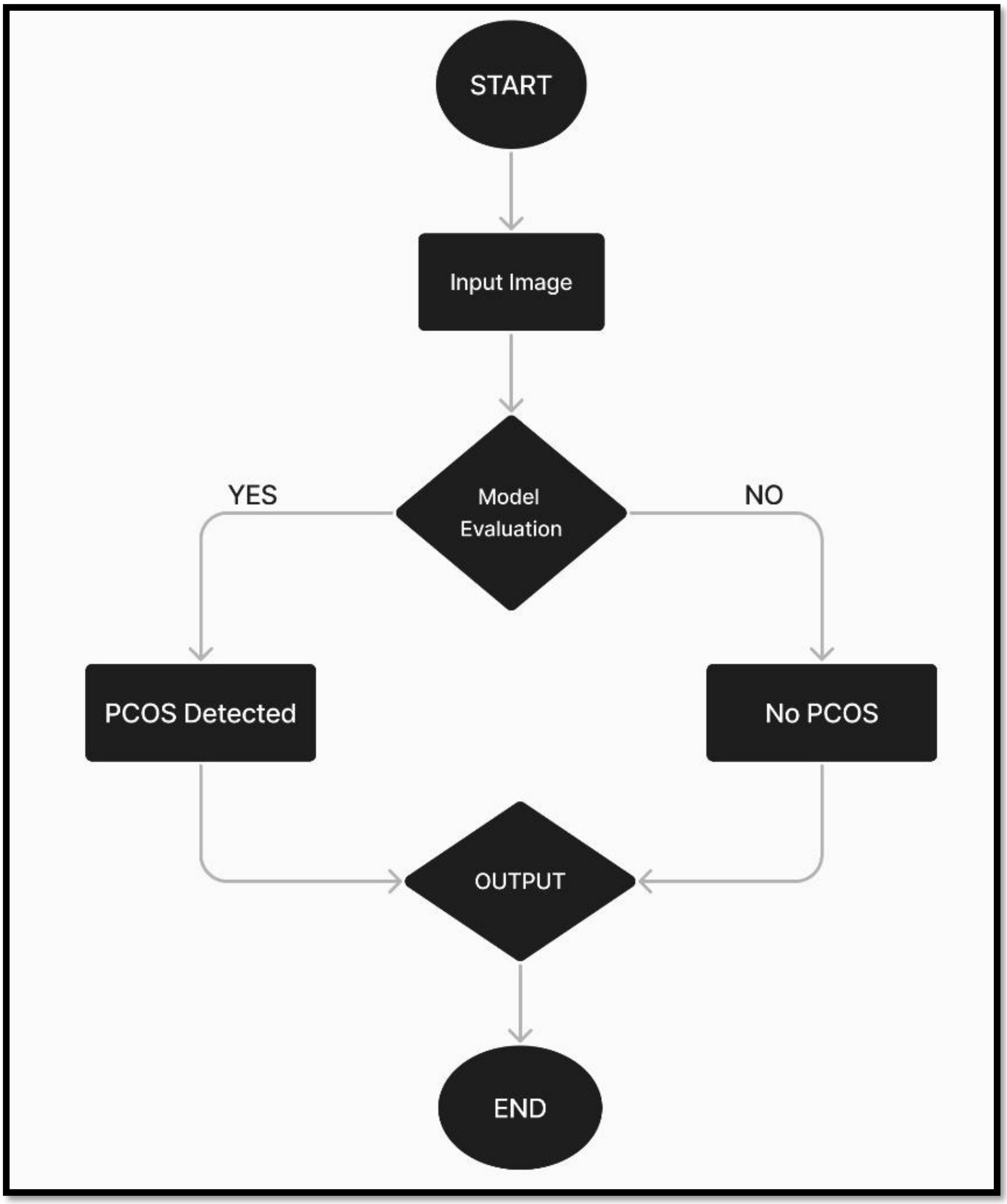


Figure 2: Flow Chart

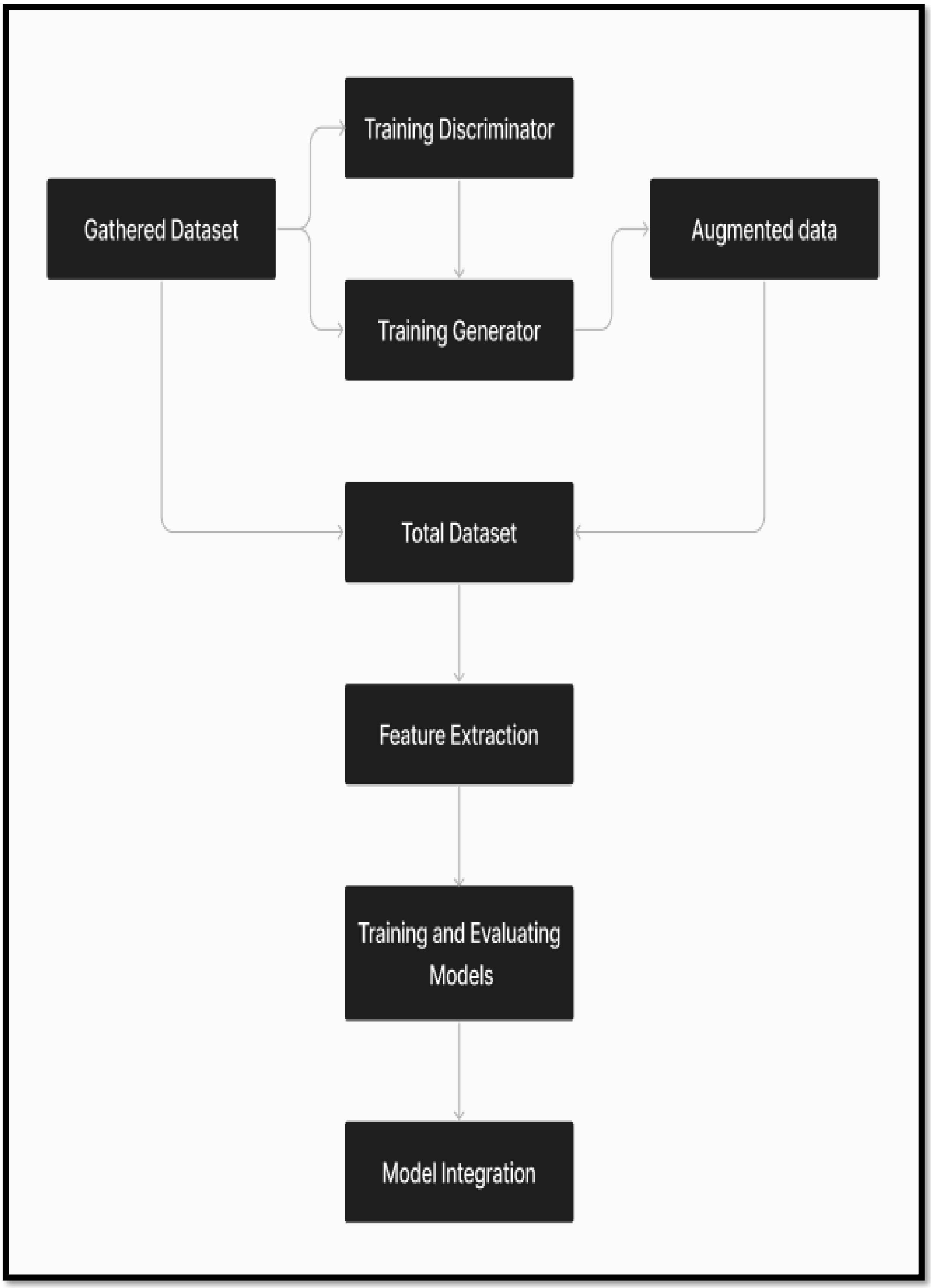


Figure 3: Working Module

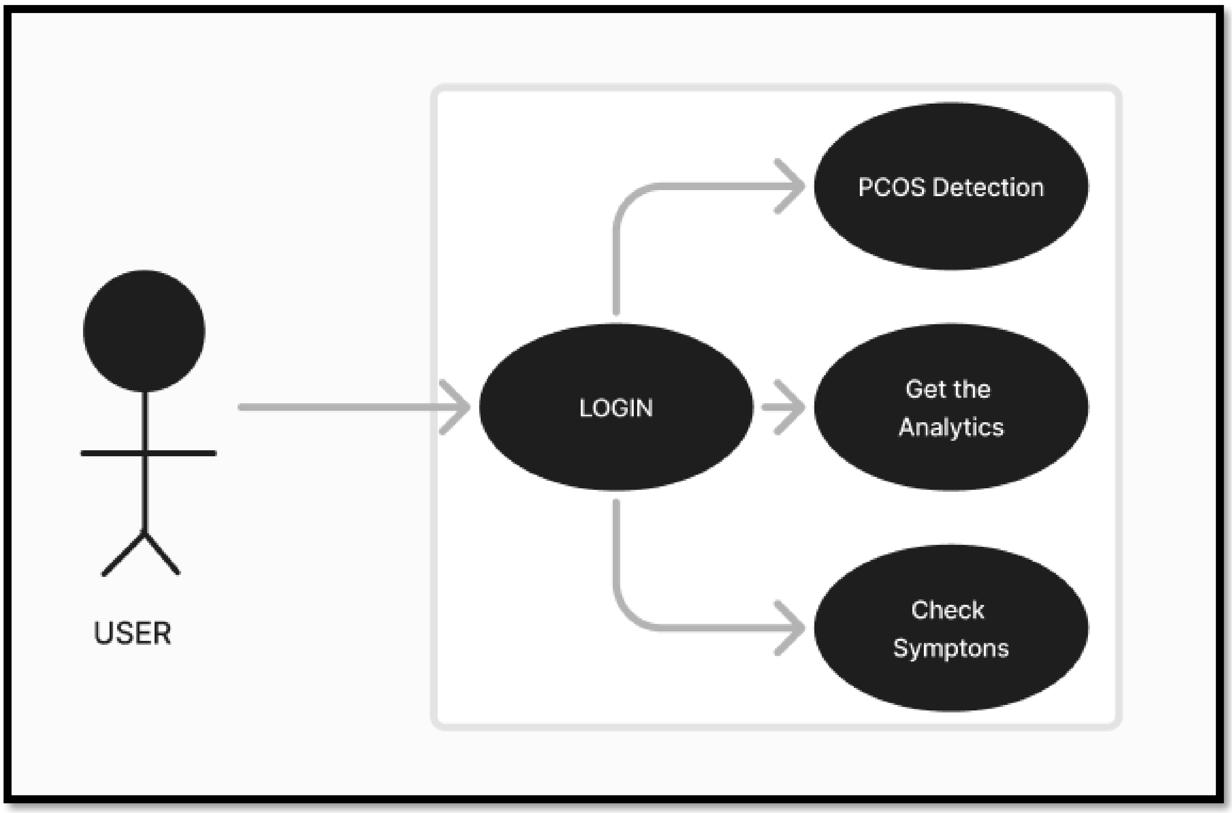


Figure 4: UML Diagram

## 2.8 Assumptions and Dependencies

**Assumptions:**

- All provided ultrasound images are correctly labeled as PCOS or healthy, ensuring the training data's integrity.
- Input images are of sufficient quality (resolution and clarity) to allow meaningful analysis.
- Users have access to modern smartphones or tablets with internet connectivity. The app will not function properly without network access, as it relies on cloud services.
- Adequate computational resources (GPUs) are available for model training. We assume access to a cloud GPU or local high-performance machine for this purpose.
- Medical regulations (data privacy, device certification) are considered but are assumed not to change drastically during development.

**Dependencies:**

- **Frameworks and Libraries:** Flutter SDK for the app, Dart language libraries, and Firebase SDKs (Auth, Firestore, Storage). On the backend, TensorFlow or PyTorch for model hosting.
- **Cloud Services:** Google Firebase (Authentication, Firestore, Storage) and optionally Google Cloud or another server for hosting the ML inference engine.
- **External Data:** The Kaggle ultrasound dataset is assumed to remain available for reference. The model's accuracy depends on the quality of this data.
- **Third-party APIs:** Any third-party APIs (e.g., image processing libraries) must be compatible with Flutter or be callable via backend services.
- **Regulatory Standards:** Compliance with medical software standards (such as IEC 62304 for medical device software) is assumed for future deployment, though full certification is beyond the current project scope.

These assumptions and dependencies form the context in which the system will be built and used. Any changes (e.g., loss of internet, or updated privacy laws) would require revisiting the requirements.

## SYSTEM REQUIREMENTS

### 3.1 User Interface

**Authentication Screen:** The login/signup screen is the entry point. It contains tabs or toggles to switch between *Login* and *Sign Up*. In the *Login* form, users enter their email and password; validation ensures a correct email format and non-empty fields. Error messages appear for incorrect credentials. The *Sign Up* form collects email, password (with strength checker), and optional profile details (e.g., name). Upon successful login or registration, the user is directed to the Dashboard.

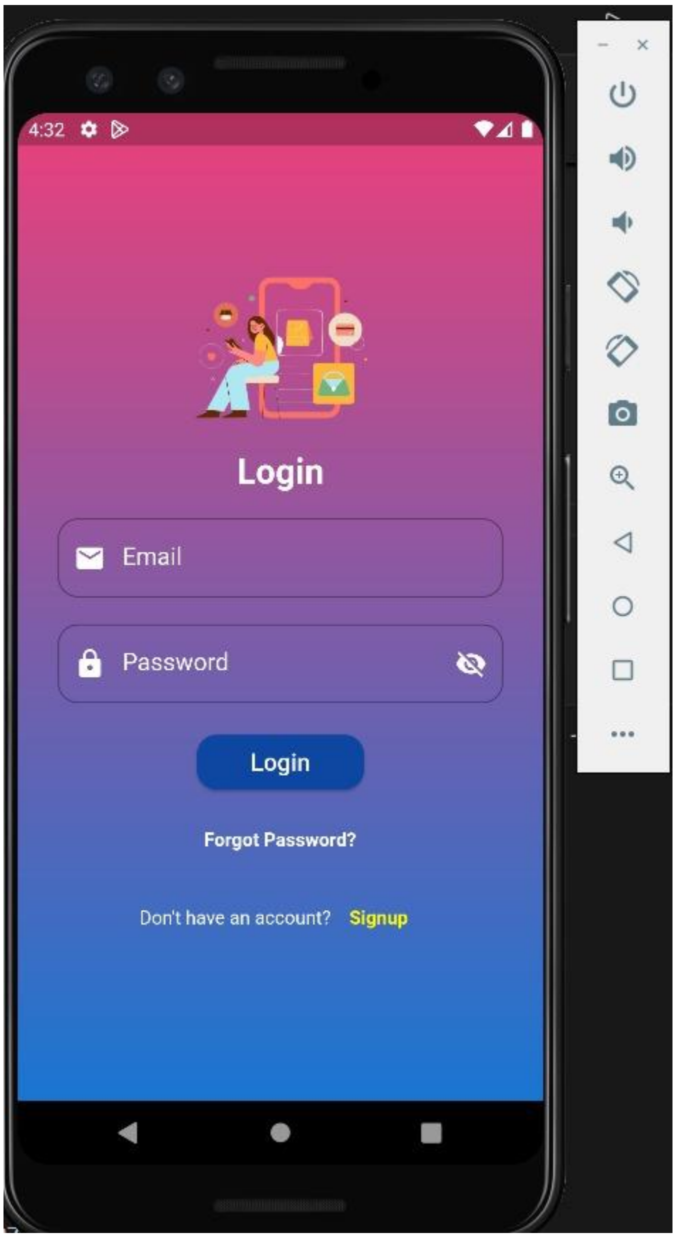


Figure 5: Login Screen

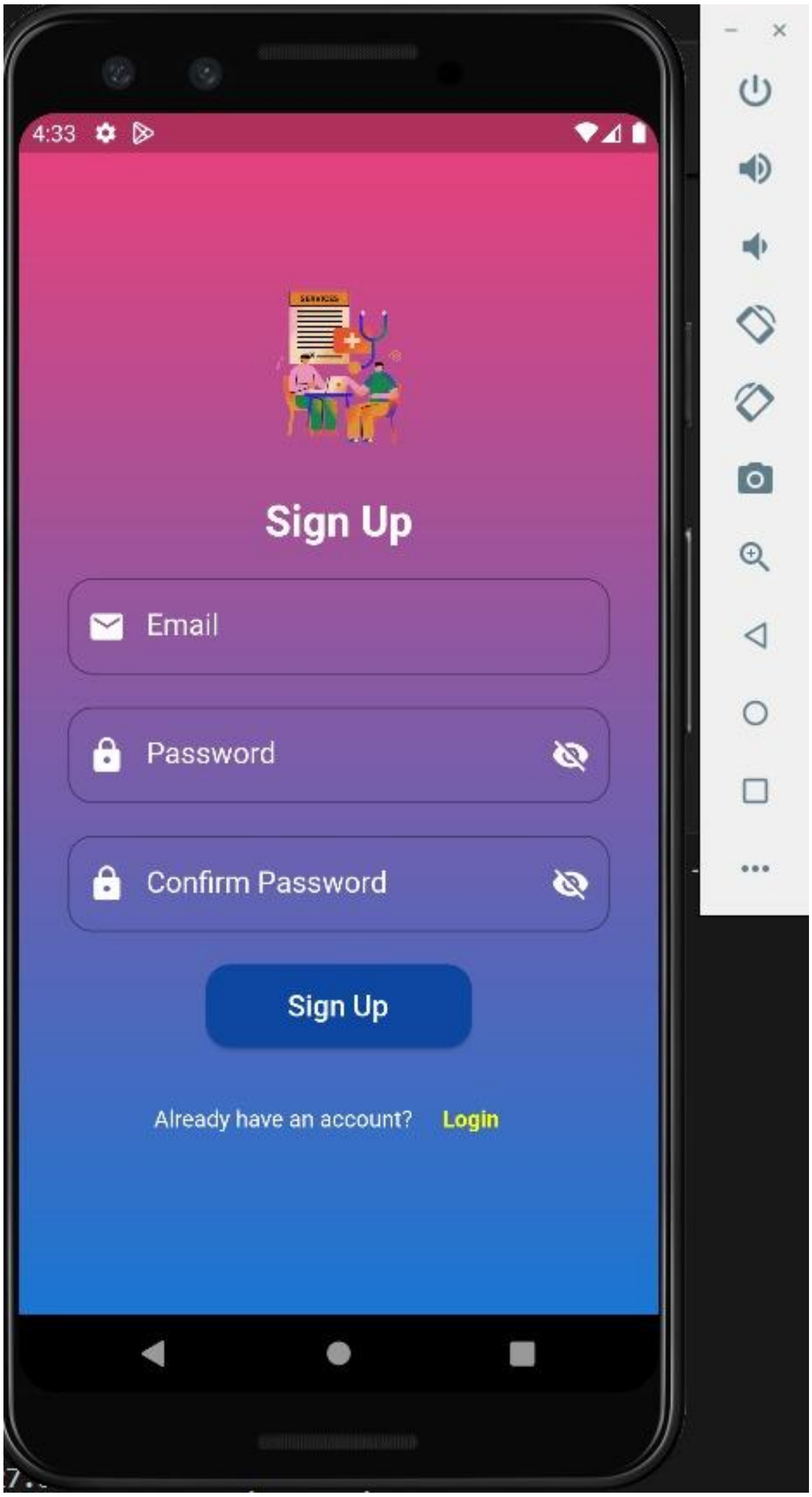


Figure 6: Sign Up Screen

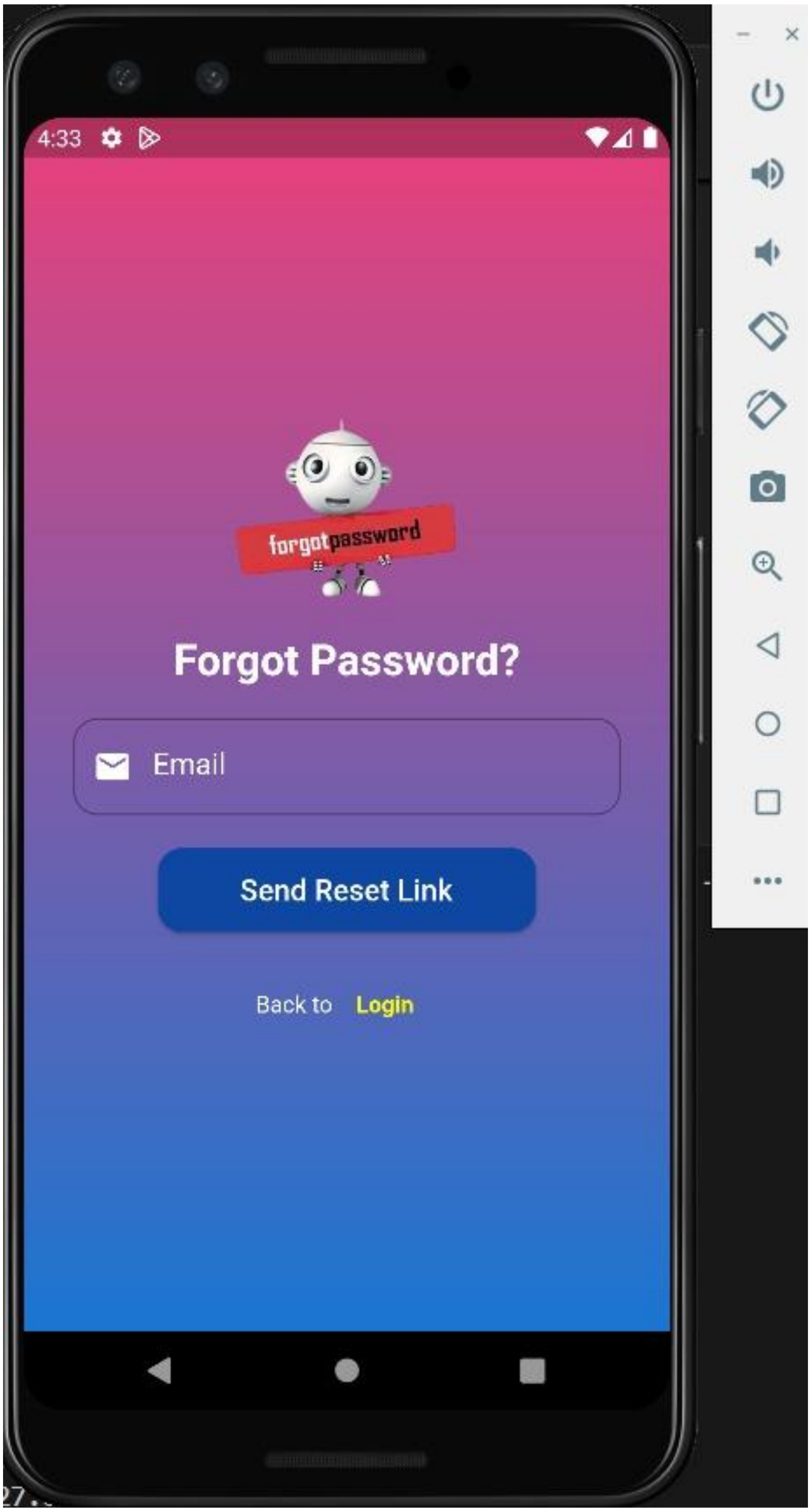
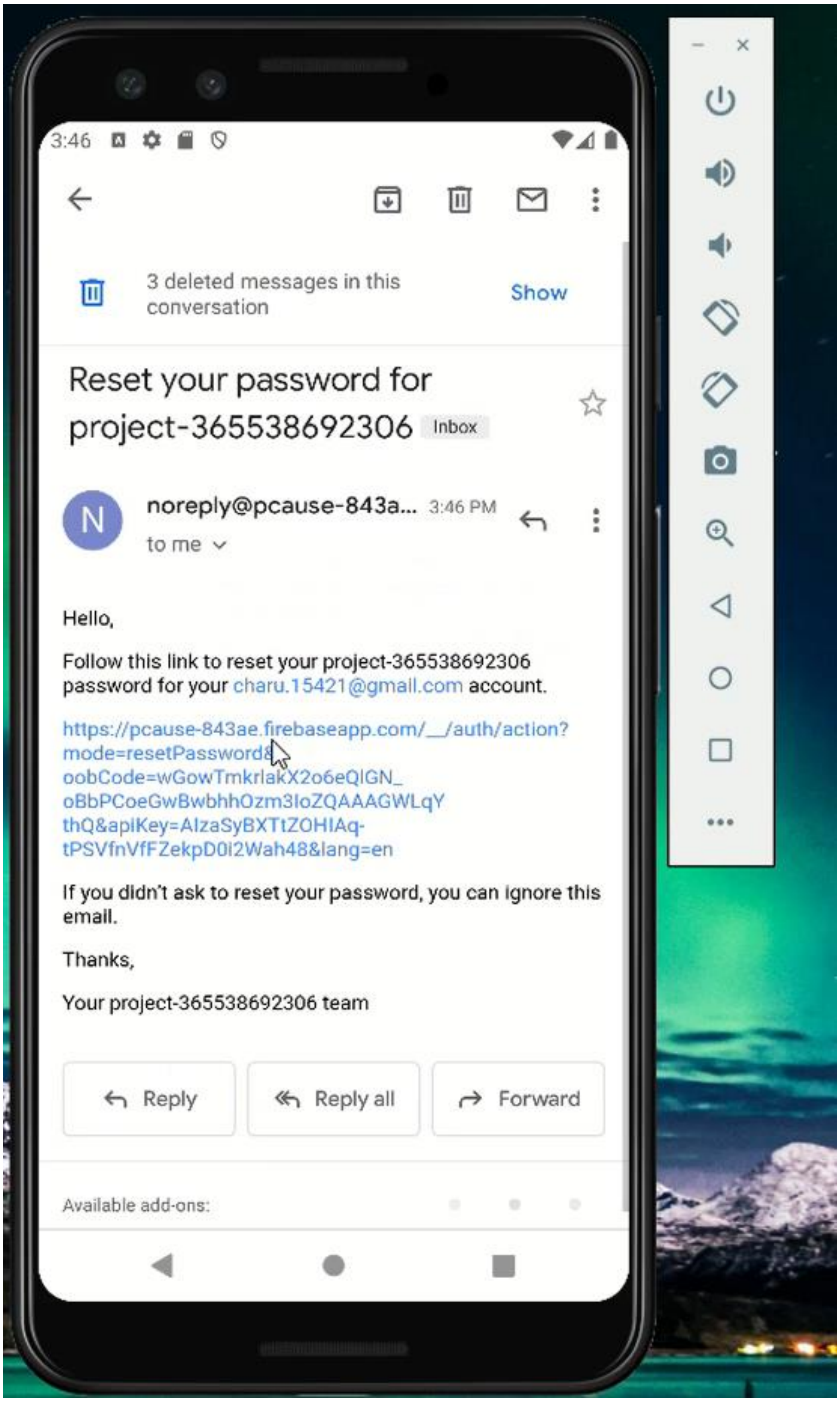


Figure 7: Forgot Password Screen



Reset your password for project-365538692306

noreply@pcause-843a...

Hello,

Follow this link to reset your project-365538692306 password for your charu.15421@gmail.com account.

https://pcause-843ae.firebaseapp.com/__/auth/action?mode=resetPassword&oobCode=wGowTmkrlakX2o6eQIGN_oBbPCoeGwBwbhhOzm3IoZQAAAGWLqYthQ&apiKey=AIzaSyBXTtZOHIAq-tPSVfnVfFZekpD0i2Wah48&lang=en

If you didn't ask to reset your password, you can ignore this email.

Thanks,

Your project-365538692306 team

Figure 8: Reset password

**Dashboard Screen:** After logging in, users see the Dashboard, which provides an overview of application features and their data. It displays recent prediction history (e.g., a scrollable list or cards with date, result, and thumbnail). Statistical widgets (charts or summary cards) might show metrics like the number of total scans, positive diagnoses, and confidence averages. The Dashboard also includes navigation buttons or a menu to the other screens (Upload Image, View History, Profile, Logout). It may display the user's name/email and a link to profile settings.

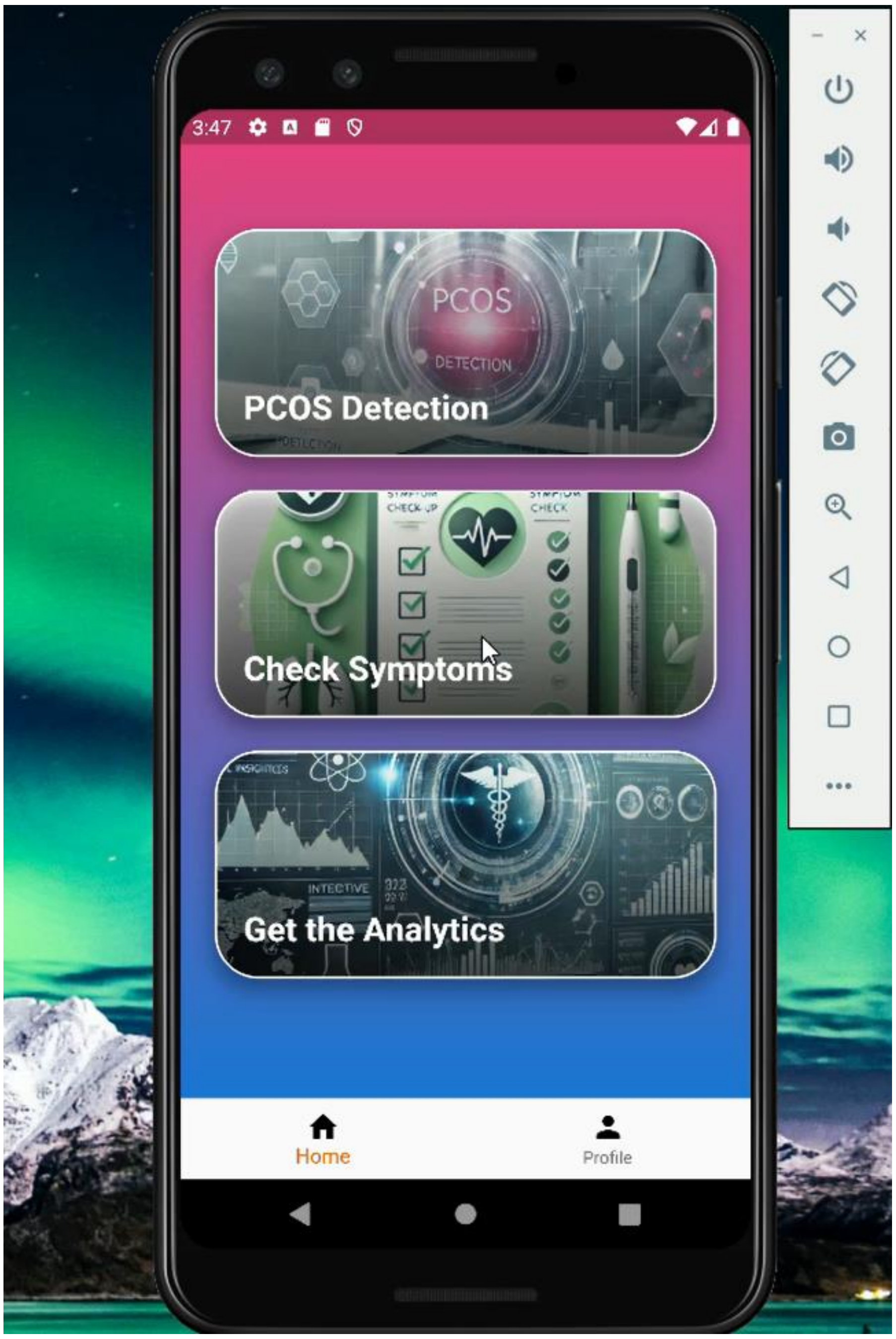


Figure 9: Home Screen

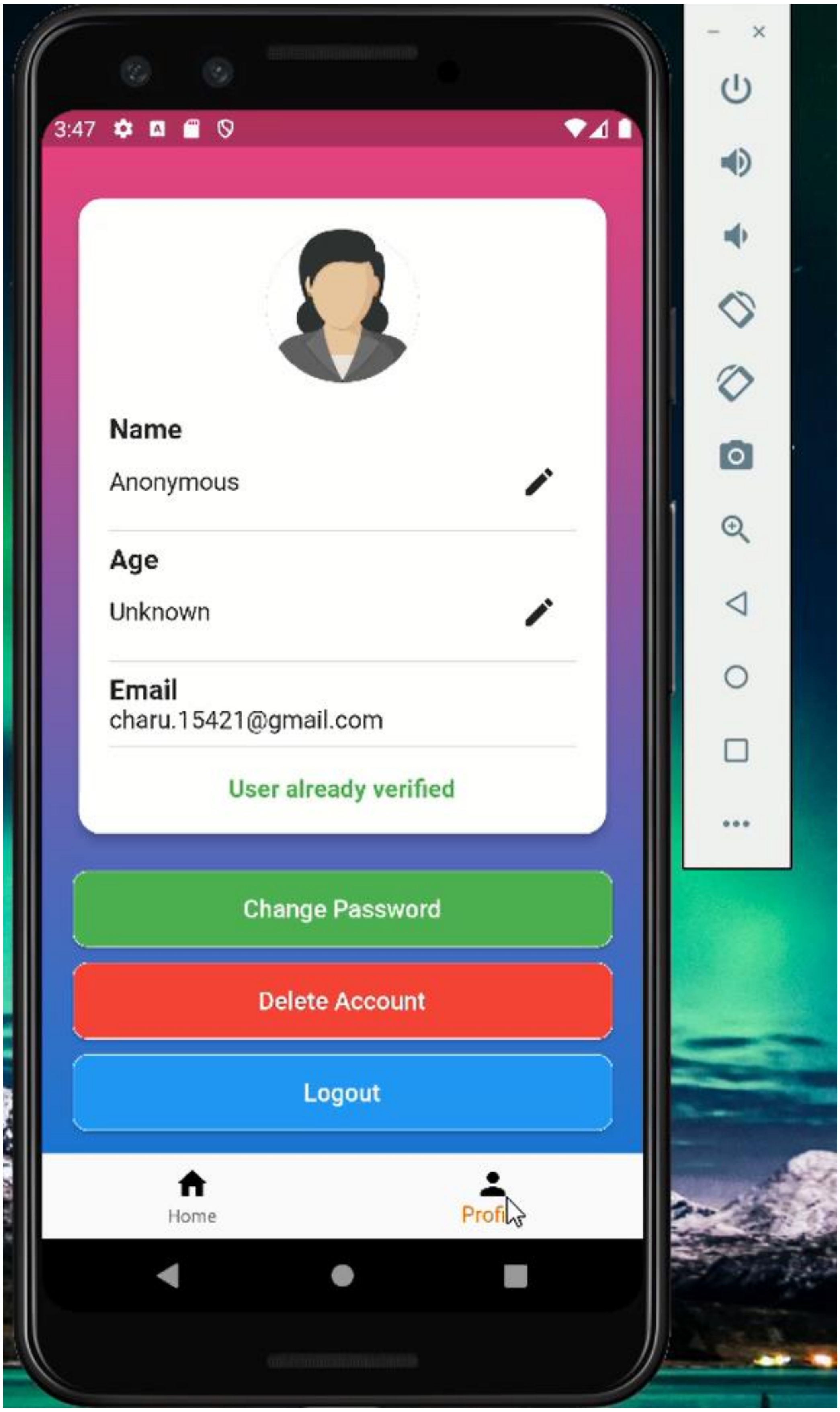


Figure 10: Profile Screen

**Image Upload Screen:** This screen allows users to select an ultrasound image for analysis. It offers options/buttons to *Choose from Gallery* or *Take Photo*. The chosen image is previewed on-screen so the user can confirm or reselect before submission. A button labeled *Analyze* or *Upload* submits the image. Basic checks prevent non-image files (using file type filters). The screen also shows instructions or a sample image to guide users on proper framing. Progress indicators (like a spinner) appear while uploading.

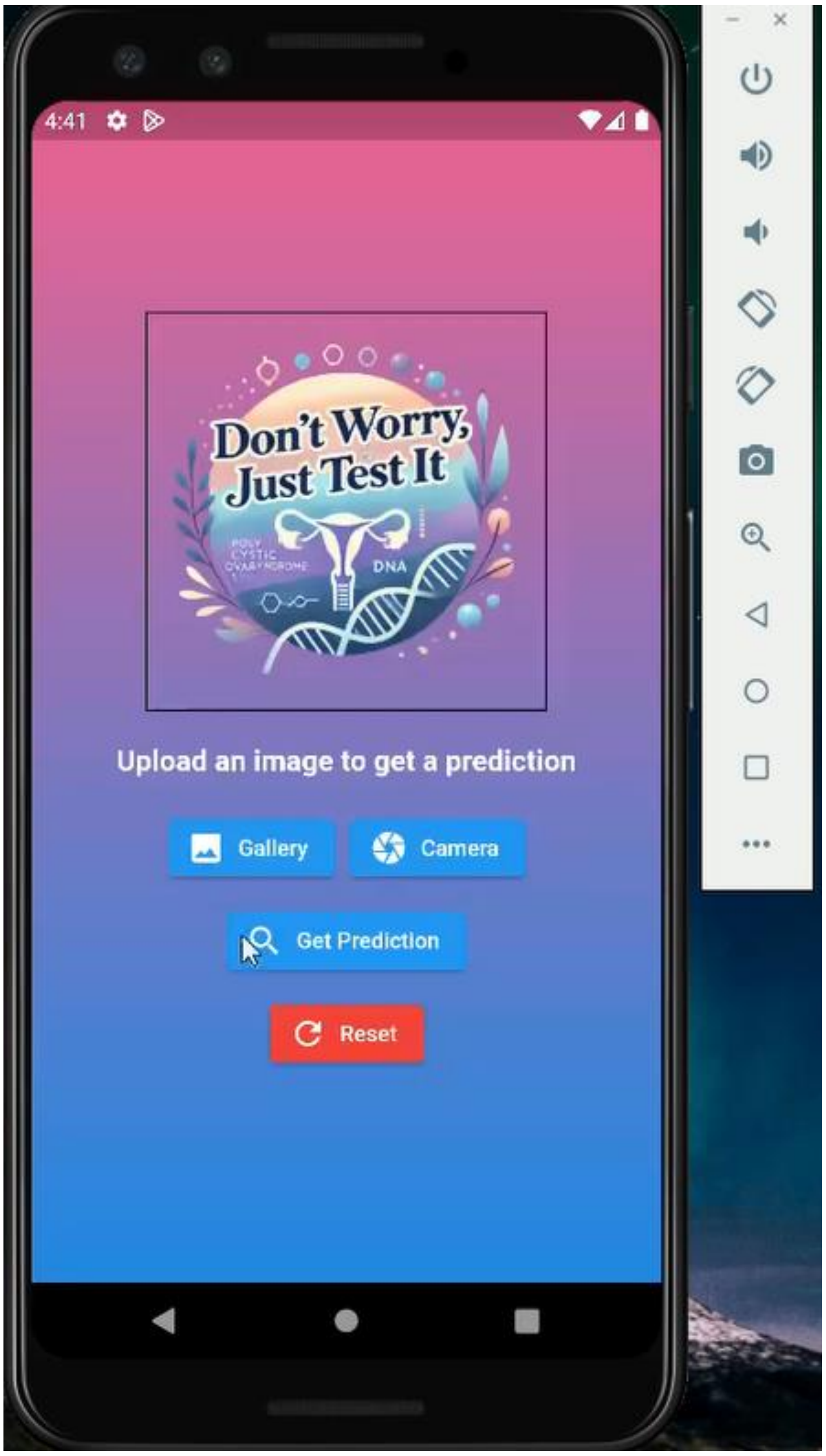

4:41
Don't Worry, Just Test It
POLY CYSTIC OVARY SYNDROME
DNA
Upload an image to get a prediction
Gallery
Camera
Get Prediction
Reset

Figure 11: Image Upload Screen

**Prediction Result Screen:** After processing, this screen shows the result of the PCOS analysis. The main content is a clear message (e.g., "PCOS Detected" or "No PCOS Detected") along with a confidence percentage or probability score. For example, "PCOS Detected (87% confidence)". To aid understanding, a simple icon or chart (e.g., a bar showing confidence level) can be shown. Additional information might include brief advice (e.g., recommending consulting a doctor if positive). Buttons for *Save Result*, *Share Report*, or *Return to Dashboard* are provided. If the model allows multi-class, the screen could list each output label with scores.

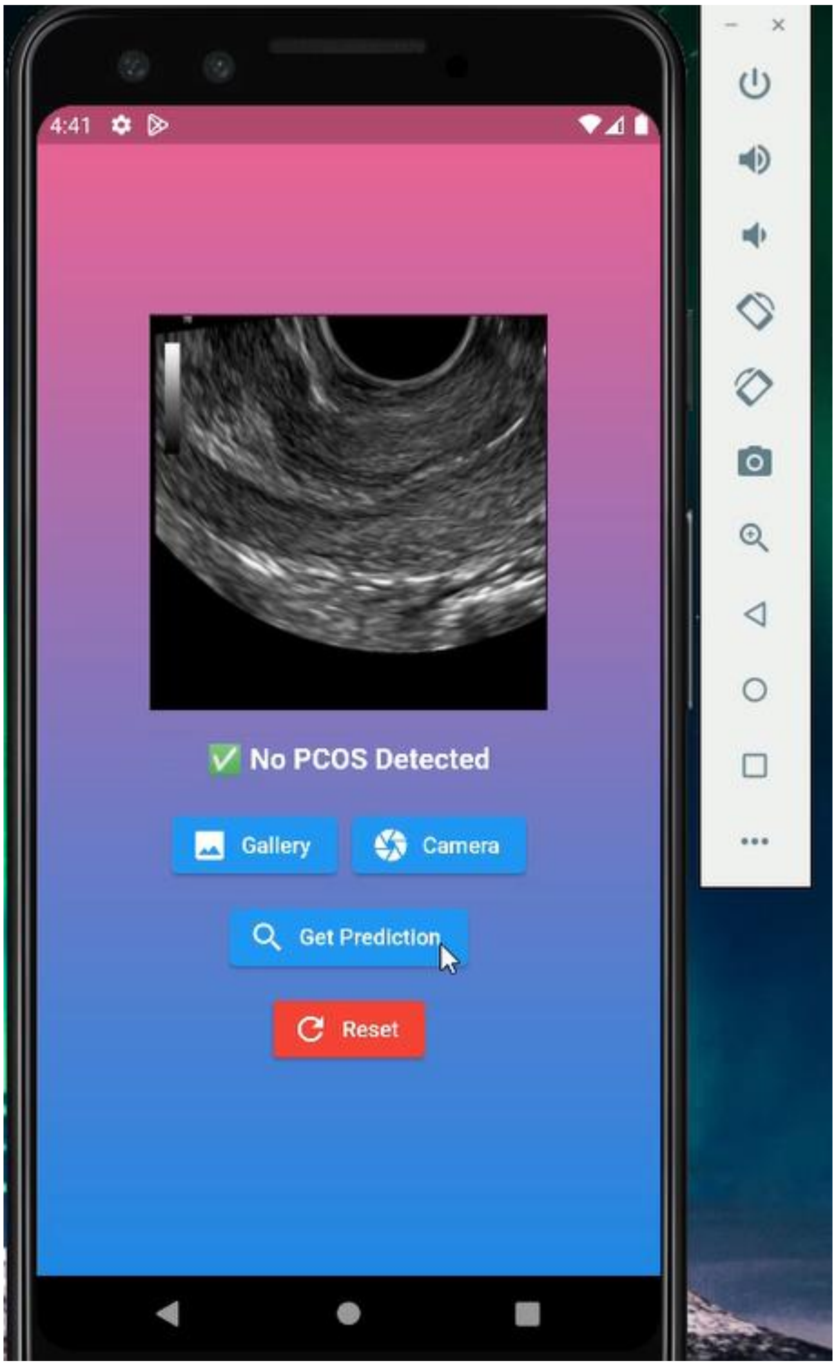


Figure 12: Prediction Screen

**History/Profile Screen:** This combined screen lets users review past activity and manage their account. One section lists all previous image analyses in reverse chronological order: each entry shows the image thumbnail, date/time, and result. Users can tap an entry to see details or delete it. Another section displays user profile information (name, email) and provides options like *Change Password* or *Log Out*. This screen may include a *Statistics* tab, where users view aggregated data (e.g., trends of positive vs. negative results). The layout ensures clear separation between history and account settings.

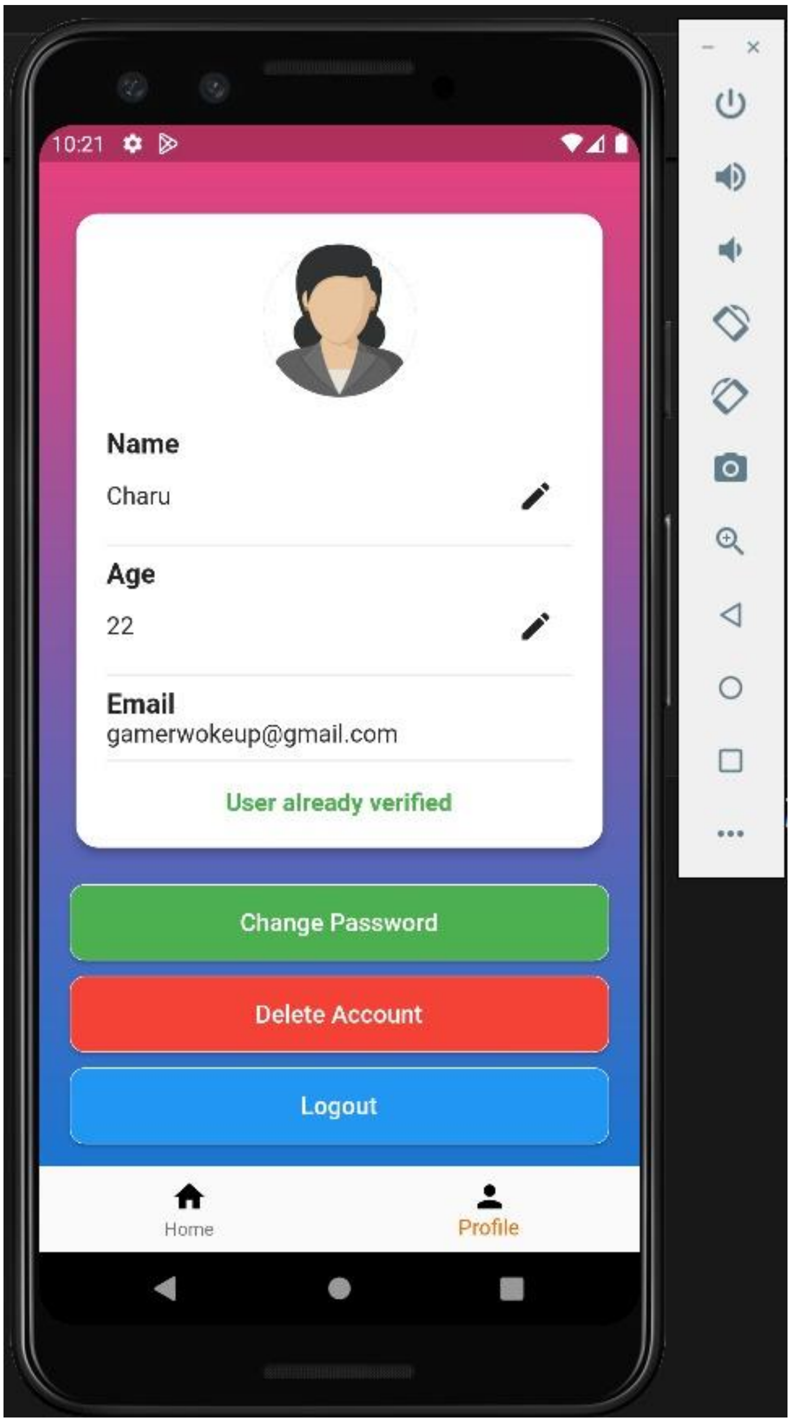
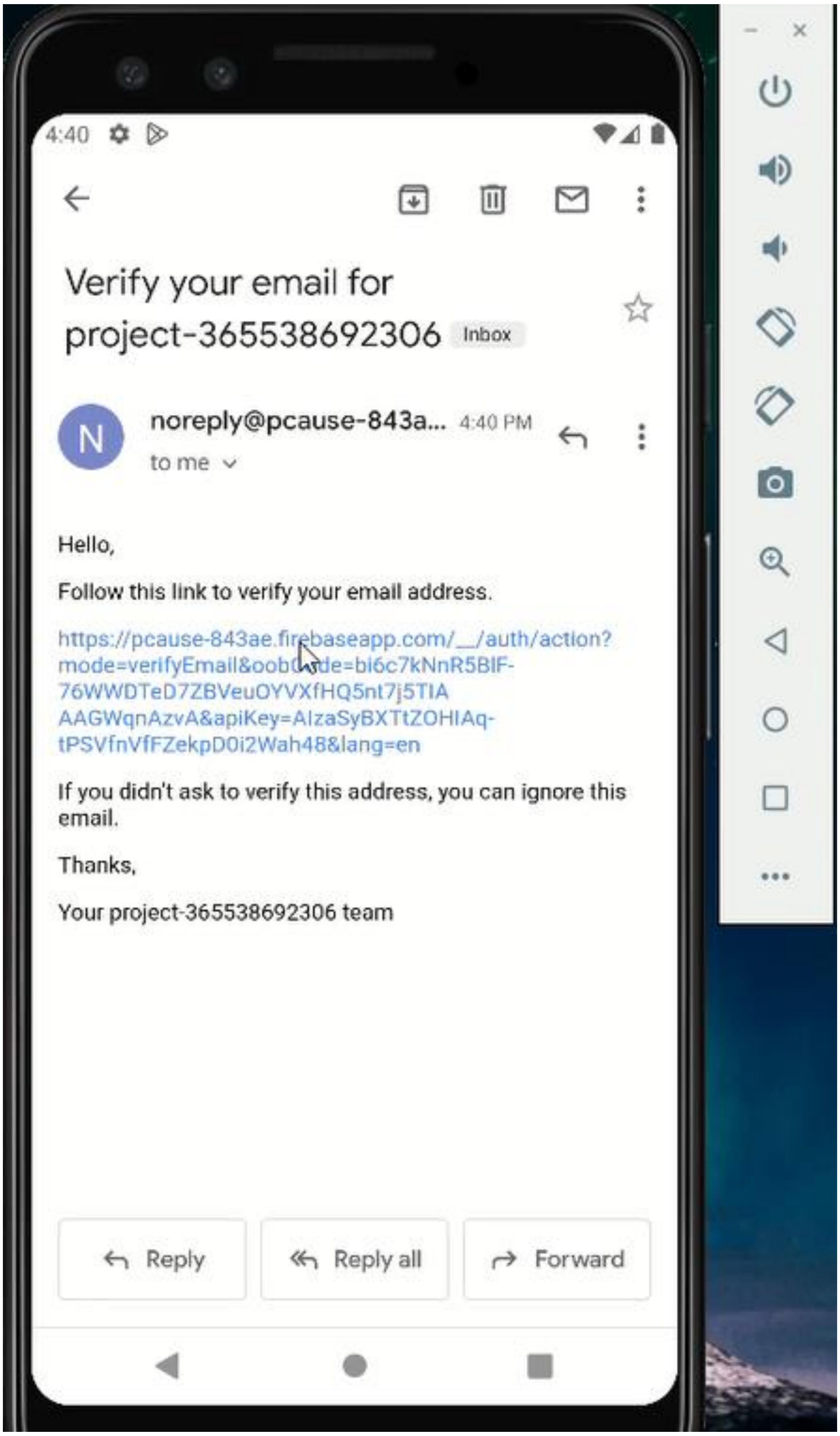


Figure 13: Profile Screen



4:40

Verify your email for project-365538692306 Inbox

noreply@pcause-843a... 4:40 PM
to me

Hello,

Follow this link to verify your email address.

https://pcause-843ae.firebaseapp.com/__/auth/action?mode=verifyEmail&oobCode=bi6c7kNnR5BlF-76WWDTeD7ZBVeuOYVXfHQ5nt7j5TIAAAGWqnAzvA&apiKey=AIzaSyBXTtZOHIAq-tPSVfnVfFZekpD0i2Wah48&lang=en

If you didn't ask to verify this address, you can ignore this email.

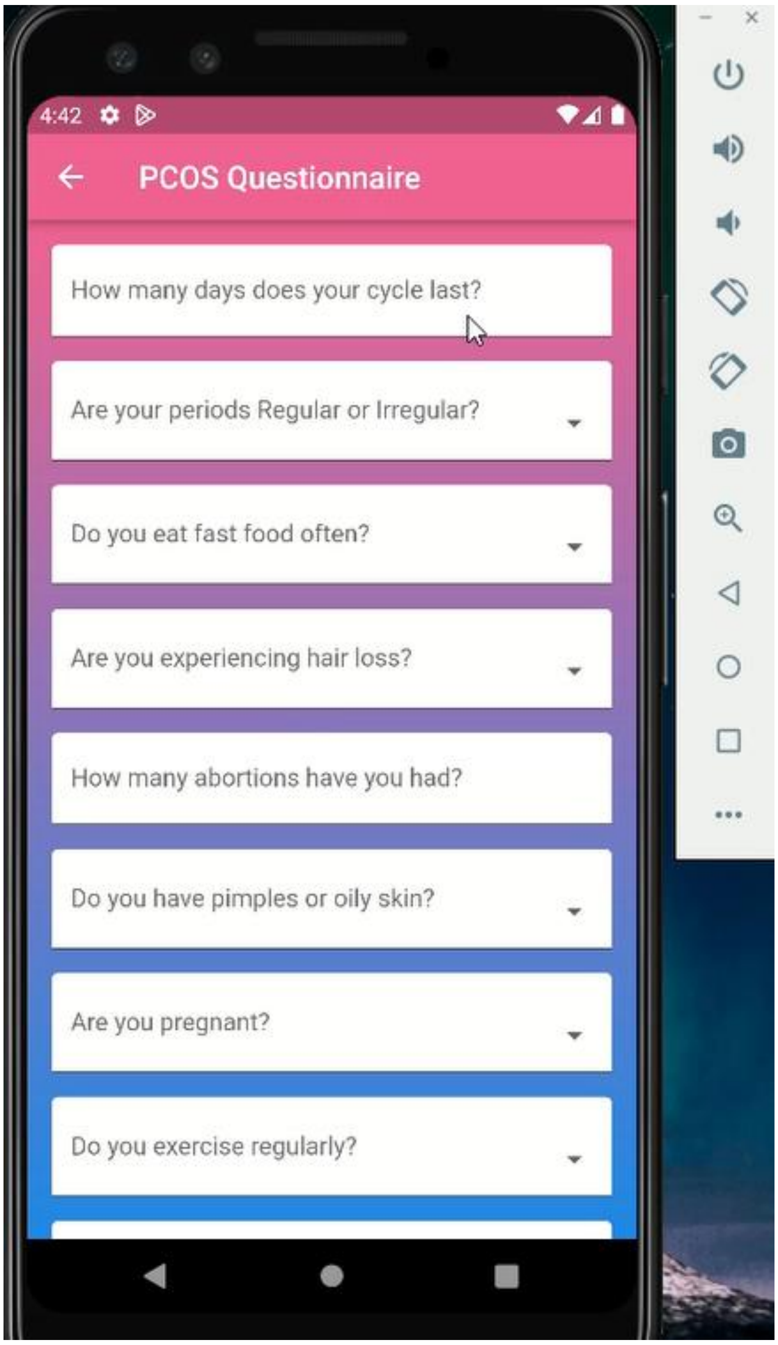
Thanks,

Your project-365538692306 team

Figure 14: Verification Email

**Check Symptoms Screen:** This screen allows users to **enter and check their symptoms** using a predefined symptom checker. The key features are:

- Input fields for users to select or type in symptoms.
- The backend uses a **trained model or logic** to process symptoms and predict a potential diagnosis or recommend the next steps.
- Helps in **early identification** and **preliminary analysis** before consulting a doctor.



How many days does your cycle last?

Are your periods Regular or Irregular?

Do you eat fast food often?

Are you experiencing hair loss?

How many abortions have you had?

Do you have pimples or oily skin?

Are you pregnant?

Do you exercise regularly?

Figure 15: Questionnaire screen

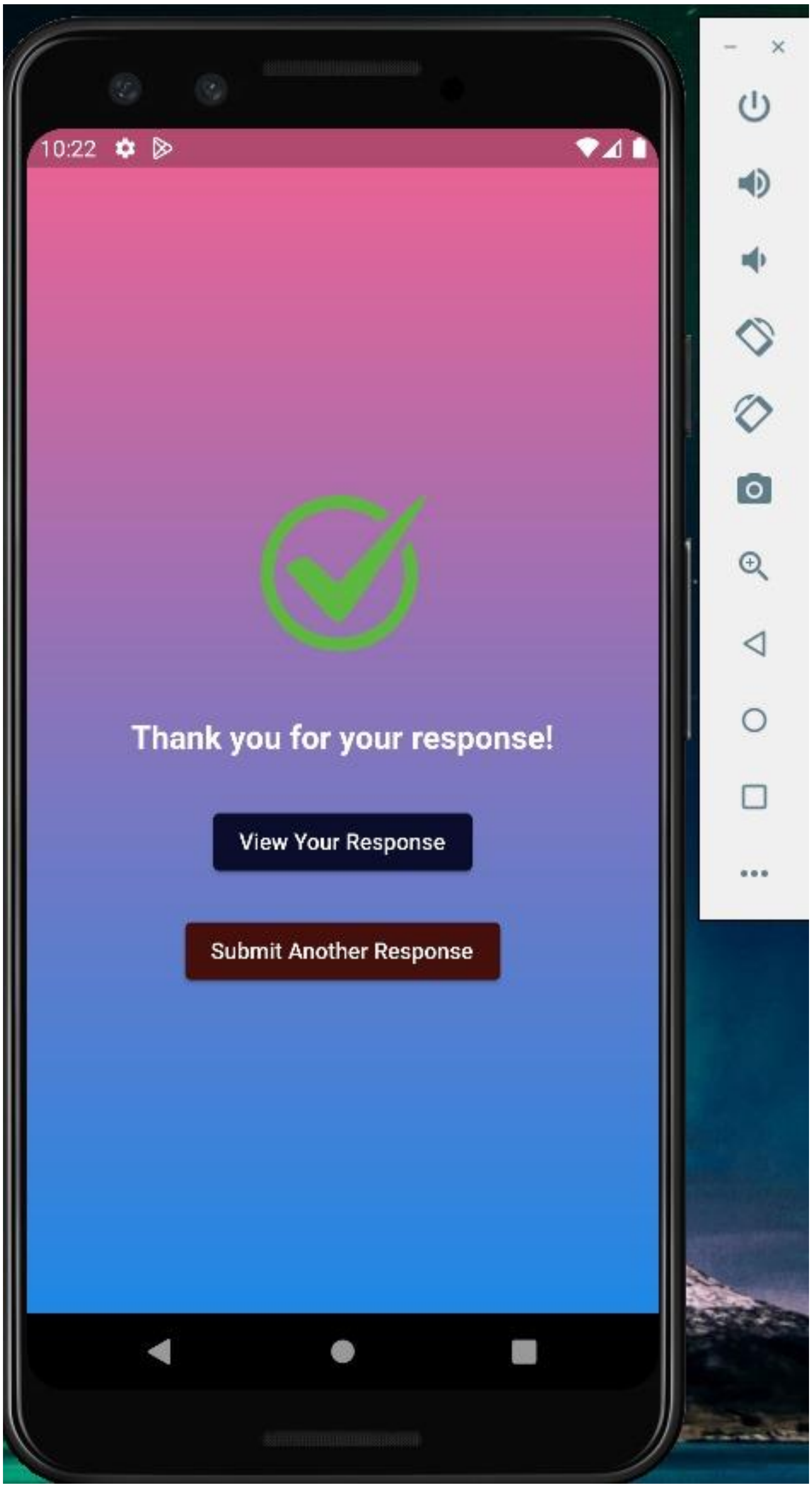
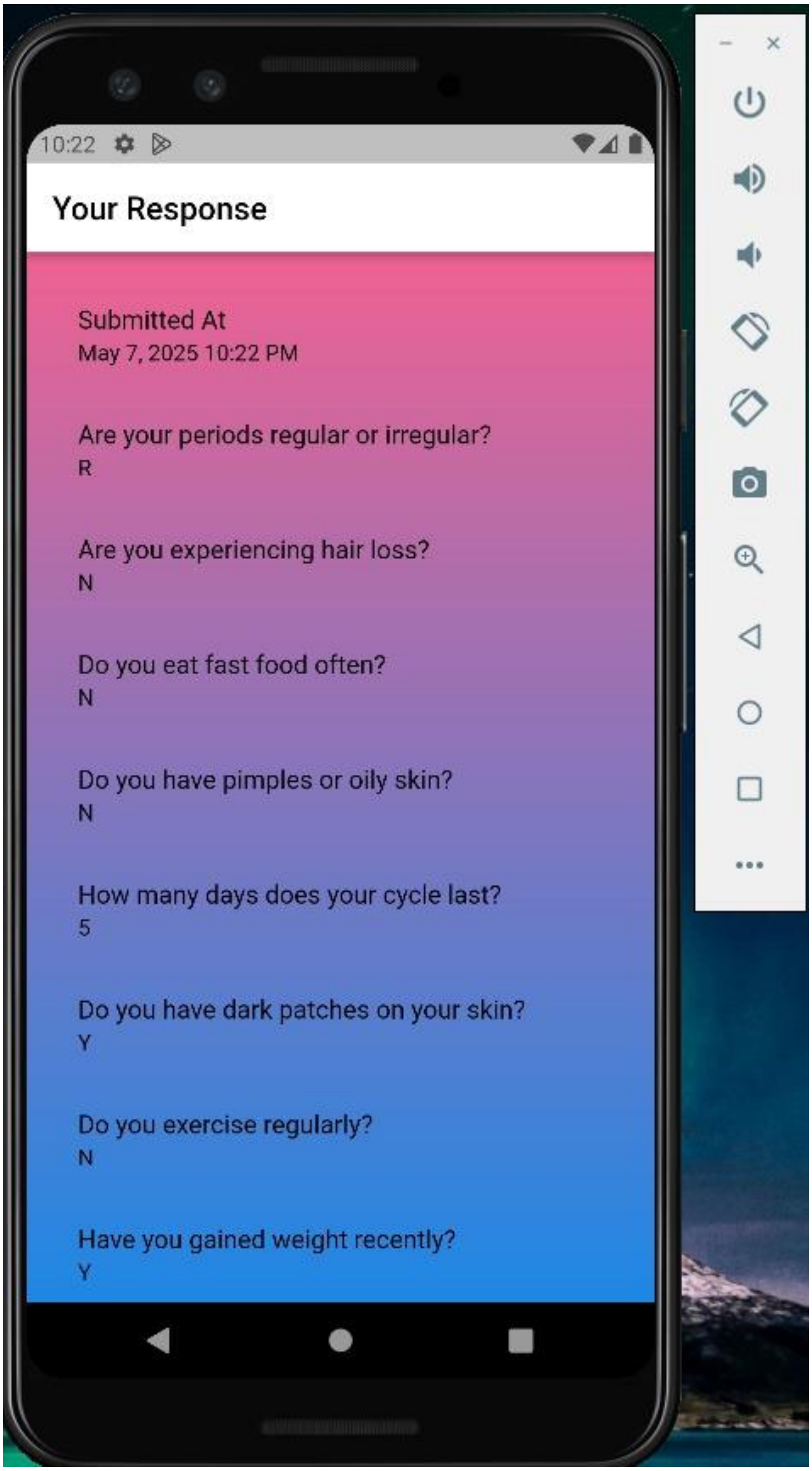


Figure 16: Response confirmation screen



Your Response

Submitted At
May 7, 2025 10:22 PM

Are your periods regular or irregular?
R

Are you experiencing hair loss?
N

Do you eat fast food often?
N

Do you have pimples or oily skin?
N

How many days does your cycle last?
5

Do you have dark patches on your skin?
Y

Do you exercise regularly?
N

Have you gained weight recently?
Y

Figure 17: View your response screen

All UI components are designed with consistency (same color theme, typography) and intuitive layout. Instructions and feedback messages ensure that users understand each action. For example, after a successful upload and analysis, the app confirms “Analysis complete – view your results”. All text is justified for neat presentation, and interactive elements have sufficient spacing for touch input.

**3.2 Software Interface**

The system’s software components communicate primarily through standard network APIs. The mobile frontend (Flutter app) uses the Firebase SDKs to interface with cloud services. For example:

- **Authentication Service:** The app calls Firebase Authentication’s REST API to register and log in users over HTTPS. Tokens are managed by the SDK.
- **Backend Model Service:** When a user requests a prediction, the app sends the image (or its storage URL) to a serverless function or REST endpoint. This is done via an HTTPS POST request. The request payload may be JSON containing the image reference. The backend service runs the DenseNet-121 model, then returns the classification result as a JSON response.
- **Firestore Database:** All data exchanges with Firestore use its secure API. The app reads/writes documents (in JSON format) to collections like “users” and “predictions”. For example, after a prediction, a new document with fields (userID, imageURL, result, confidence, timestamp) is created. These operations use HTTPS under the hood.
- **Cloud Storage:** Image files are uploaded to Firebase Storage using an HTTPS POST, and download URLs are generated. The app uses these URLs when updating Firestore records.

Inter-module connections are thus:

- The app UI calls *AuthModule* for login/signup, then *DatabaseModule* for storing user info.
- For predictions, the *ImageService* uploads to storage, then triggers *ModelService* which processes the image.
- Once the model returns a result, *ResultService* updates the UI and writes to the database.

All inter-component communication happens over HTTPS (SSL/TLS). Data formats are JSON or plain images as appropriate. The system does not require any proprietary protocols; it relies on standard RESTful or gRPC calls provided by the services. No direct database access from the UI occurs without going through the authorized APIs.

## 3.3 Database Interface

The PCause application utilizes **Firebase Firestore** as its primary database solution. Firestore is a scalable, flexible, cloud-hosted **NoSQL document database** provided by Google, designed to support real-time syncing and offline access for mobile and web applications. Its structure, performance, and ease of integration with Flutter make it ideal for the demands of a healthcare-oriented AI system like PCause.

### Data Structure and Schema

The database is structured around the concept of **collections and documents**, rather than rigid tables and rows. The key collections used in PCause include:

- **Users Collection**: Each user of the application is represented as a document in the Users collection. The document ID corresponds to the unique userUID assigned during authentication. Each document stores:
    - email: User's registered email address.
    - name: Full name of the user.
    - profile_settings: Any personal configurations or preferences.
- **Predictions Subcollection**: Each user document contains a nested Predictions subcollection, which stores all prediction-related entries. Each document in this subcollection includes:
    - imageURL: A reference to the uploaded ultrasound image (stored in Firebase Storage).
    - resultLabel: The result of the PCOS prediction (e.g., Positive, Negative).
    - confidence: The model's confidence percentage in its prediction.
    - timestamp: A server-generated time value indicating when the prediction was made.

This structure enables efficient storage and retrieval of historical medical predictions per user, enhancing personalized analysis and tracking over time.

### Data Flow and Synchronization

When a user submits an ultrasound image for analysis:

1. The raw image file is first uploaded to **Firebase Storage**, a cloud object storage service built on Google Cloud Platform.

2. A new prediction entry is simultaneously created in Firestore under the respective user's Predictions subcollection.
3. Firestore transactions ensure atomicity, meaning both the image upload and metadata creation are coordinated as a single logical operation.

To support robustness and availability:

- **Offline persistence** is enabled, allowing the app to function without an internet connection by queueing write operations locally.
- Once connectivity is restored, Firestore automatically syncs the pending operations with the cloud database.

**Security and Access Control**

- Firebase provides fine-grained **security rules**, ensuring that users can only **read and write** their own data.
- For example, a user cannot view another user's predictions or image files.
- Images stored in Firebase Storage are accessed via secure URLs, which are referenced inside the Firestore documents.

**Why Firestore?**

Firestore was chosen for several reasons:

- **Real-time updates**: Changes in user data or prediction history are reflected instantly in the app UI.
- **Scalability**: The platform scales automatically with the user base, making it suitable for future growth.
- **NoSQL Flexibility**: Allows schema evolution (e.g., adding new analytics fields) without disrupting existing data.
- **Flutter Integration**: Firestore's official Flutter plugin ensures smooth development and reliable app performance.
- **Powerful querying**: Developers can retrieve historical predictions efficiently using indexed fields like timestamp.

## 3.4 Protocols

To ensure secure, efficient, and reliable communication between the PCause application and its backend services, a set of standardized, well-supported network protocols is employed. These protocols cover authentication, data transmission, and file handling — all critical to the functioning of a healthcare-related AI platform that deals with sensitive user data and medical predictions.

**Secure Communication Protocols**

All network communication in PCause is conducted over **HTTPS (HyperText Transfer Protocol Secure)**, which layers **SSL/TLS encryption** on top of standard HTTP. This ensures that all data transmitted between the client application (mobile or web) and backend services (Firebase, cloud functions, storage) is **encrypted in transit**, mitigating the risk of man-in-the-middle attacks or eavesdropping.

- **Authentication**: PCause uses Firebase Authentication, which operates over HTTPS and leverages **OAuth 2.0** under the hood. OAuth tokens are securely issued by Firebase upon login/signup and are automatically refreshed during a user session by the Firebase SDK.
- **Image Upload/Download**: Ultrasound images are transmitted as **binary files** (image/jpeg or image/png) via HTTPS, typically using multipart upload forms or direct file upload endpoints provided by **Firebase Storage**.
- **API Calls**: All interactions with Firestore (reading/writing user data, predictions, etc.) and Firebase Functions (if used for model inference) are performed using HTTPS requests with valid authentication tokens in the request headers.

**Data Formats and Interactions**

- **JSON** (JavaScript Object Notation) is the primary format for structured data exchanged between the app and backend.
  - Firestore documents are serialized as JSON objects.
  - Model prediction results (e.g., label, confidence, timestamp) are also returned and parsed as JSON.
- **Binary MIME types** are used for image files. These are typically not embedded in JSON but referenced via URLs to Firebase Storage.

This division of concerns — structured metadata via JSON and raw images via binary upload — ensures both performance and clarity in communication between app modules.

**Communication Patterns**

- **RESTful architecture** is used for most interactions, such as fetching prediction history or submitting new data.
- **gRPC** may optionally be used internally by Firebase services for efficiency, but this is abstracted away from the developer and user.

- The app follows a **request/response cycle**, avoiding the need for persistent socket connections or polling. Prediction uploads, authentication, and dashboard data retrieval are all initiated by explicit user actions.

Although PCause does not implement high-frequency communication protocols such as TCP sockets or real-time streaming, **Firebase's built-in real-time capabilities** (e.g., Firestore snapshots) are leveraged to reflect data changes instantly within the UI — useful for scenarios like showing updated prediction history or dashboard analytics without requiring a full refresh.

**Security Enhancements and Best Practices**

- **Encryption at rest** is handled natively by Firebase, meaning data stored in Firestore and Firebase Storage is automatically encrypted using Google-managed keys.
- **Session Management**: Short-lived access tokens and automatic token refresh mechanisms help prevent unauthorized long-term access.
- **Re-authentication** is required for sensitive actions (e.g., deleting history or account), helping to mitigate risks from stolen sessions or device compromise.
- **Minimal Latency Design**: REST endpoints are chosen based on response efficiency, and caching is selectively applied on the client side to reduce redundant API calls.

By relying entirely on **modern, proven protocols** like HTTPS, OAuth 2.0, and JSON over REST, PCause maintains strong **data security**, **compliance with privacy standards**, and **smooth user experience**. These choices align with the industry's best practices for mobile health (mHealth) applications, especially those involving AI-driven medical diagnostics.

## NON-FUNCTIONAL REQUIREMENTS

### 4.1 Performance Requirements

The PCause application is designed for responsiveness and efficiency:

- **Latency:** User actions (tapping buttons, navigating screens) should produce a response within 100–200 milliseconds, ensuring a fluid UI experience.
- **Prediction Time:** Once an image is submitted, the end-to-end prediction (upload, inference, download) should complete within a few seconds (ideally under 5 seconds). This depends on image size and network speed. We will optimize image compression and use efficient server hardware (GPUs) to meet this goal.
- **Throughput:** While we do not expect massive concurrent load, the system should gracefully support tens of simultaneous users during peak times (e.g., a small clinic). The cloud backend will autoscale the prediction service if needed.
- **Scalability:** Firestore and Firebase Storage can handle large data volumes. As usage grows (more users and images), read/write times should remain under 500 milliseconds.
- **Resource Usage:** The mobile app's memory footprint should stay moderate (e.g., under 100 MB) and should not drain the battery (no continuous background processing). Heavy computation (model inference) is offloaded to the server to avoid burdening the device.
- **Correctness:** The model's accuracy on test data should meet or exceed clinical expectations (for example, target >85% accuracy). Performance metrics (accuracy, precision, recall, F1-score) will be documented, although these are technically functional metrics.

These requirements ensure that users experience minimal wait times and that the app remains usable and efficient on typical hardware.

### 4.2 Security Requirements

Security and privacy are paramount since PCause handles personal health information:

- **Authentication & Authorization:** All user access is gated by Firebase Authentication. Only authenticated users can reach the prediction and history features. Each user's data is isolated; Firestore security rules prevent one user from accessing another's records.
- **Data Encryption:** All data in transit is encrypted via HTTPS. Sensitive data at rest (user credentials, images) are encrypted by Firebase's managed keys. We do not store unencrypted medical data on the device.

- **Password Security:** User passwords are hashed and salted by Firebase Auth (we do not handle raw passwords). We enforce a minimum password strength (e.g., at least 8 characters, including letters and numbers) to protect accounts.

- **Privacy Compliance:** The system adheres to privacy regulations. Personal identifiers (emails) are stored only as needed; images and results are treated as protected health information (PHI). Data collection is limited to what is necessary for the system's functionality. Users must consent to data usage. We use Firebase's secure infrastructure (which is compliant with HIPAA and GDPR when configured properly).

- **Input Validation:** All user inputs are validated to prevent injection attacks or malformed data. Since we rely on managed services, standard OWASP risks (like SQL injection) are mitigated by design. However, we sanitize any text input and guard against malformed JSON.

- **Logging and Monitoring:** The system logs critical events (logins, failed attempts, prediction errors) in a secure log service. Logs do not contain sensitive data (e.g., no raw images). Monitoring alerts are set up for unusual activity (e.g., spike in failed logins).

- **Third-Party Security:** Any third-party libraries or services (Flutter packages, Firebase, cloud hosting) are vetted for security. We plan to use a BAA (Business Associate Agreement) with Firebase if deploying in a regulated environment.

Overall, security requirements aim to protect user data confidentiality and system integrity, building trust in the application's use for medical purposes.

### 4.3 Software Quality Attributes

- **Adaptability:** The system is built with modular components. Using Flutter ensures that the same codebase adapts to both Android and iOS. The machine learning model can be retrained or replaced without altering the app's core logic, allowing updates for new medical insights.

- **Availability:** Cloud services (Firebase) provide high uptime (typically 99.9%). The app will handle occasional outages gracefully by notifying users and queuing actions. Key functions (login, view history) are built to work briefly in offline mode using Firestore's local cache, enhancing availability during transient network issues.

- **Correctness:** We implement thorough unit and integration testing. The model's predictions are evaluated against a test set, and the system includes logic checks (e.g., ensuring an image was actually processed). Any inconsistency triggers an error message rather than silent failure.

- **Flexibility:** The design separates UI, backend, and model. This separation of concerns makes it easier to add new features (like alternative analysis models or report formats) with minimal changes to existing code.

- **Interoperability:** The application can integrate with other systems. For example, patient result data could be exported in CSV or FHIR format for inclusion in electronic health records. The RESTful API could allow other apps to use the model. Standard protocols (HTTPS, JSON) ensure compatibility.

- **Maintainability:** Code is written following best practices and is well-documented. Continuous Integration (CI) pipelines will run automated tests for each update. The use of popular frameworks (Flutter, Firebase) means a large developer community and well-supported libraries, simplifying future maintenance.

- **Portability:** Because the UI is built in Flutter, the same code is portable across multiple platforms (mobile and possibly web). The backend components (Firebase rules, cloud functions) can be deployed to other Google Cloud projects or ported to different cloud providers if needed.

- **Reliability:** Error handling is comprehensive: network failures retry automatically, invalid inputs prompt user correction, and uncaught exceptions are minimized. Regular backups (Firestore provides automated backups) ensure data can be recovered in case of failures. The system aims for robust operation under expected conditions.

- **Reusability:** Key modules (e.g., authentication module, image-processing service) are designed as standalone components that could be reused in other health applications. For instance, the image upload and storage logic is generic and could be repurposed.

- **Robustness:** The application validates all user interactions. For example, if the model service returns an unexpected response, the app handles it gracefully with a user-friendly error. Input files are checked for correct format before sending. These checks prevent crashes and ensure consistent behavior even with bad inputs or partial failures.

- **Testability:** Unit tests cover individual functions (e.g., form validation, data parsing). Integration tests simulate end-to-end scenarios (login → upload → prediction). The model's performance is tested on a hold-out dataset. The logging of intermediate states aids debugging.

- **Usability:** The interface follows standard mobile design guidelines: buttons are clearly labeled, navigation is straightforward, and fonts/icons are legible. We aim for minimal learning curve so that a new user can complete an analysis with no instructions. User feedback (e.g., loading spinners, success confirmations) makes the system transparent. Color schemes have sufficient contrast, and the layout is tested for accessibility.

## OTHER REQUIREMENTS

- **Development Environment:** The system is developed using open-source tools. The code repository includes a README with setup instructions. All dependencies (Flutter, Dart, Python libraries) are specified with version numbers. We use Git for version control and semantic versioning for releases.
- **Documentation:** A user manual will be provided, describing how to install and use the app. Developer documentation will cover architecture, API endpoints, and procedures to retrain or update the model.
- **Data Backup and Recovery:** Automated daily backups of the Firestore database will be scheduled. The backup process includes exporting collections to cloud storage. In case of data corruption, backups can be restored.
- **Logging and Monitoring:** We will integrate a logging service (e.g., Firebase Crashlytics) to capture runtime errors and usage metrics. This helps in ongoing maintenance.
- **Licensing:** All third-party components are chosen to be license-compatible (permissive licenses). The Kaggle dataset license (if any) is noted, and we ensure compliance by using the data for research/education.
- **Legal Compliance:** The final deployment must consider medical device regulations. Though out of scope for initial development, the architecture allows for future certification (e.g., code signing, audit trails).
- **Internationalization:** The app is initially in English. The design (Flutter localization) allows adding other languages in the future without major redesign.
- **Scalability Beyond MVP:** While the minimum viable product meets the functional needs, the system is built so that new features (like direct integration with ultrasound machines, or support for other image formats) could be added later.

These additional requirements ensure the solution is robust in practical terms and lays groundwork for long-term use and compliance.

## Appendix A: Glossary

- **PCOS (Polycystic Ovary Syndrome):** A hormonal disorder in women of reproductive age, characterized by irregular menstrual periods, excess androgen levels, and ovarian cysts.
- **AI (Artificial Intelligence):** The simulation of human intelligence by machines, especially in learning and problem-solving tasks.
- **CNN (Convolutional Neural Network):** A type of deep learning model particularly effective for analyzing visual imagery. CNNs use convolutional layers to automatically and adaptively learn spatial features from input images.
- **GAN (Generative Adversarial Network):** A machine learning model consisting of two neural networks (Generator and Discriminator) that compete to produce realistic synthetic data. GANs are used here for data augmentation of ultrasound images.
- **PCA (Principal Component Analysis):** A statistical technique for dimensionality reduction. It transforms data into a new coordinate system where the greatest variance lies on the first coordinates (principal components), used here to reduce image feature complexity.
- **DenseNet-121:** A 121-layer CNN architecture with dense connections between layers, known for efficient training and strong performance on image classification tasks. In PCause, it is the base model for classifying ultrasound images.
- **Flutter:** An open-source UI software development toolkit by Google for building natively compiled applications for mobile, web, and desktop from a single codebase, using the Dart programming language.
- **Firebase:** A cloud platform by Google providing backend services (authentication, database, storage) for mobile and web applications. We use Firebase Auth, Firestore DB, and Storage.
- **Authentication:** The process of verifying a user's identity (e.g., via email and password) before granting access to the system.
- **Firestore:** A scalable NoSQL cloud database by Firebase that stores data as documents in collections, supports real-time updates and offline mode.
- **UI (User Interface):** The part of the application with which the user interacts (screens, buttons, forms, etc.).
- **API (Application Programming Interface):** A set of rules and protocols for building and interacting with software applications. In PCause, the APIs include Firebase services and any custom HTTP endpoints.

- **HIPAA/GDPR:** Legal frameworks (in the US and EU, respectively) governing the privacy and security of personal health data.

- **PDF (Portable Document Format):** A file format used to present documents independently of software or hardware, used for exporting reports in the app.

- **SDK (Software Development Kit):** A collection of software tools and libraries that developers use to create applications for specific platforms (e.g., Flutter SDK for mobile development).

- **CSV (Comma-Separated Values):** A simple text format for tabular data, used for exporting datasets.

- **CI (Continuous Integration):** A development practice where code changes are automatically tested and merged, ensuring early detection of integration issues.

**Appendix B: Analysis Model**

This section summarizes the key design models and diagrams that guide the PCause system architecture. Each model translates parts of the requirements into structured representations:

- **Use Case Model:** Actors (Patient, Clinician) and their use cases are identified. For example, a Patient actor has the use cases *Register/Login*, *Upload Image*, *View Prediction*, and *View History*. A Clinician may additionally use *Review Reports* or *Download Data*. This model clarifies who interacts with the system and what functions they perform.

- **Class Model:** Main classes include:

    - **User:** Attributes such as userID, email, passwordHash. Methods for authentication and profile management.

    - **ImageRecord:** Attributes imageURL, timestamp, userID. Represents an uploaded ultrasound image.

    - **Prediction:** Attributes label (PCOS/Healthy), confidence, and relatedImageID. Associated with an ImageRecord.

    - **AuthService, DatabaseService, ModelService:** These backend classes handle logic for authentication, data storage, and image classification, respectively.

Associations: A User has many ImageRecords. Each ImageRecord has one Prediction.

- **Activity Model:** Example workflow "Perform Prediction":

1. *Start* → User taps "Upload" → picks image → system uploads image to storage.
2. After upload, the image URL is sent to the model service → prediction is computed.
3. The result is saved to the database → user sees the result screen. *End.*

Another activity is "User Authentication": User enters credentials → system validates → success or failure.

- **Sequence Model:** For a prediction request, the sequence is:

    - The mobile App calls Firebase Auth to get a token (if not already logged in).

    - App uploads image to Firebase Storage, obtaining an image URL.

    - App sends a request to a Cloud Function (or other API) with the image URL and user token.

    - The Model Service downloads the image from Storage, runs the DenseNet-121 model, and returns a JSON result.
    - The App receives the result and calls Firestore to save the prediction. Finally, the App updates the UI with the result.

- **State Model:** A user session can be in states: *LoggedOut*, *LoggingIn*, *LoggedIn*. Within *LoggedIn*, substates include *Idle* (viewing dashboard), *Uploading* (during image selection), and *ViewingResult*. Transitions occur on events like "user submits login", "image uploaded", or "logout".
- **Data Flow Model:** Shows data movement: Input image data flows from the App to the Storage/Model, then result data flows to Firestore and back to App. It highlights that user credentials flow to Auth, images to Storage, metadata to DB, and results to both DB and UI.
- **Deployment Model:** Illustrates physical components: The mobile app (client) runs on user devices. Backend components (Auth service, database, model inference) run on cloud servers. All components connect via the internet (HTTPS). For example, the ML model may be deployed on a Google Cloud Compute instance or Cloud Function, separate from Firebase services.

These models form the blueprint for implementation. In practice, the diagrams would be drawn using UML conventions (e.g., Figure B.1 – Use Case Diagram; Figure B.2 – Class Diagram, etc.).

## Appendix C: Issues List

1. **Limited Dataset Diversity:** The ultrasound images come from specific sources and may not cover all demographic groups or ultrasound machine settings. This could limit the model's accuracy for unseen cases.

2. **Explainability of Results:** The AI model's decisions are not transparent. Users may need explanation tools (like heatmaps) to understand why a decision was made, which is not implemented in the current version.

3. **Device Compatibility:** Older or low-memory devices may struggle with the app's performance, especially if image files are large. Testing is needed across a range of devices.

4. **Regulatory Approval:** To use this app in a clinical setting, it may require certification as a medical device. This process is outside the current development scope and is noted as a potential future requirement.

5. **Internet Dependency:** The app requires network connectivity for authentication, data storage, and model inference. Offline functionality is minimal. Users in areas with poor internet may experience limited utility.

6. **Data Privacy Concerns:** Storing medical images and predictions in the cloud raises privacy issues. While we use secure services, obtaining explicit consent and ensuring full compliance will be an ongoing concern.

7. **Maintenance of Model Accuracy:** The model's accuracy may degrade over time if new patterns of PCOS (e.g., due to improved imaging) emerge. A process for retraining the model on new data will be needed.

8. **User Training:** Although the interface is designed to be intuitive, some training or guidance may be necessary for non-technical users (especially patients) to take useful images and interpret results correctly.