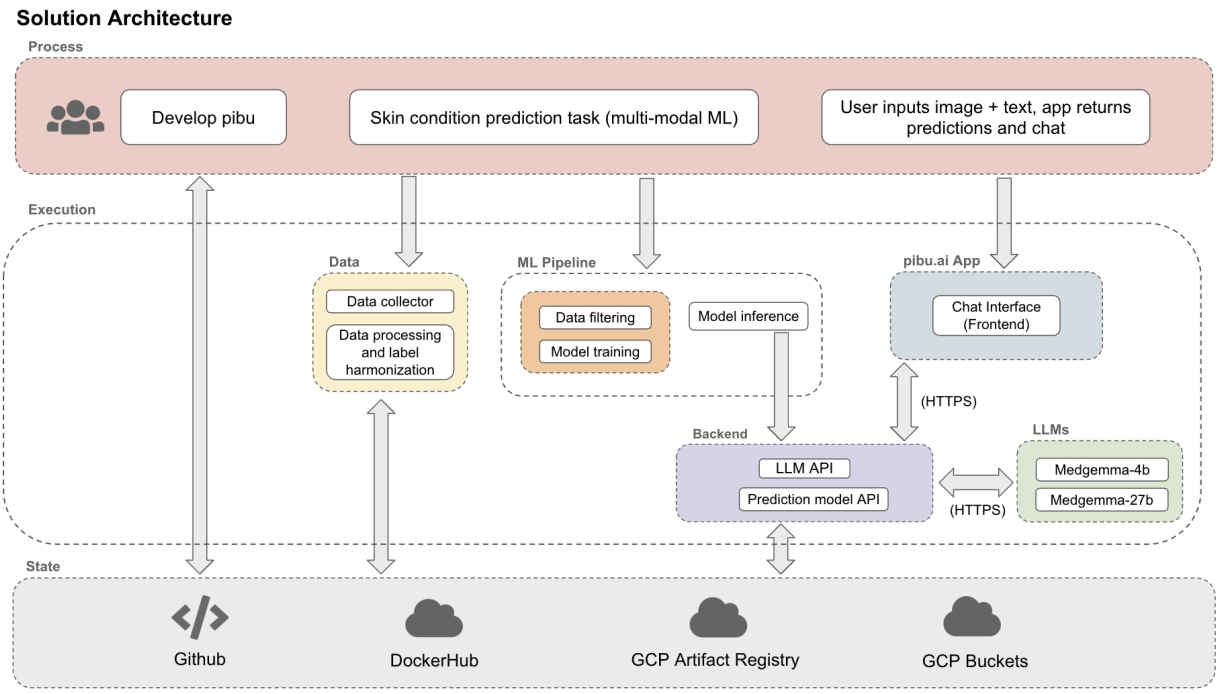
pibu.ai — Solution & Technical Architecture

1. Solution Architecture

pibu.ai integrates multimodal ML models, backend orchestration, and a chat-based LLM interface for privacy preserving and safe, useful dermatology guidance.



1.1 Components Overview

Data
The dataset contains ~190k images across 6 datasets and 96 harmonized skin-condition classes. Roughly all of the images include associated text metadata (symptoms, demographics, notes). All data is stored in a **GCS bucket** with unified label mapping.

ML Pipeline
Each training run filters data based on the experiment configuration (via YAML config). The pipeline trains:

- A **vision encoder** and **vision MLP head**
- A **text MLP head**
- A **multimodal classifier head**

The model outputs a predicted skin condition + confidence from both image embeddings and user-provided text.

Backend
The backend exposes a unified inference and LLM interface through an **APIManager** class. Responsibilities include:

- Receiving user inputs (image embeddings + text)
- Calling the ML inference container for predictions
- Passing predictions + user metadata to the LLM
- Handling ongoing chat history for follow-up questions
- Returning structured final and interactive responses

LLM Layer
We use MedGemma (27B) for medical-oriented reasoning. The backend provides predictions, metadata, and conversation history; MedGemma generates:

- The initial explanation of the predicted condition
- Follow-up answers in chat mode, grounded in the ML prediction and prior turns

Frontend
The frontend provides a unified dashboard for users to track and manage their dermatological cases. Users can:

- View active and historical cases with progress over time
- Select any past case to review updates or ask new questions
- Create a new case by uploading an image and entering symptoms

The frontend communicates with the backend to:

- Send image embeddings and text inputs for ML inference
- Fetch predictions and LLM responses
- Stream chat-like follow-up responses from Pibu, the dermatology assistant

1.2 APIs

LLM API
1. **Initial Response Endpoint:** Takes ML predictions + user text and returns the first clinical explanation.
2. **Chat Endpoint:** Accepts user questions and conversation history and returns LLM answers.

Inference API
Runs the multimodal model using image embeddings + symptom text and returns predictions.

1.3 Data Flow (End-to-End)

1. **User Inputs:** User uploads an image + short text description.
2. **Local Processing:** The device computes **image embeddings** locally for privacy and speed.
3. **Backend Inference:** Embeddings + user text are sent to the backend; the ML model outputs a ranked list of predicted skin conditions.
4. **LLM Reasoning:** The backend supplies predictions + metadata to MedGemma; the LLM produces safe, helpful guidance.
5. **Interactive Chat:** Additional user questions are processed through the chat endpoint while maintaining context.

1.4 Multimodal Model

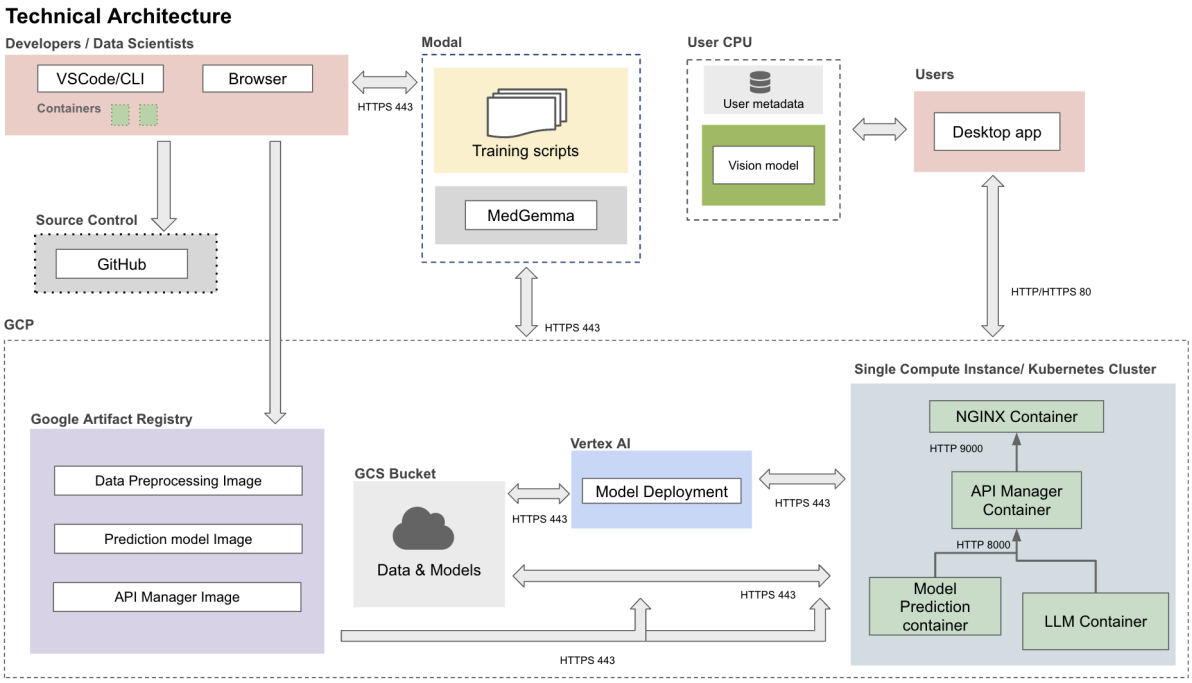
The model contains:

- **Vision encoder** → image embedding
- **Text MLP head** → symptom/metadata embedding
- **Classifier head** → joint embedding → prediction over 96 classes

The model is fully configurable and reproducible through YAML-based experiment configs.

---

2. Technical Architecture



2.1 Development Environment

VSCode / CLI / Browser support rapid development, testing, and inspection of logs. They allow flexible local iteration before pushing to cloud infrastructure.

GitHub provides version control, code collaboration, and CI for linting, formatting, and basic checks.

2.2 Training & Inference Infrastructure

Modal provides high-end GPUs (e.g., H200s) needed to train multimodal models efficiently at scale. We also run **MedGemma inference** on Modal because A100 GPUs are required, which are not available on GCP.

2.3 Containerization & Deployment

- **Google Artifact Registry** stores all container images for preprocessing, inference, and API services.
- **Vertex AI** will be used for scalable, managed deployment of inference services.
- **Kubernetes** provides flexible scaling, rolling updates, and separation of services (ML inference, LLM routing, frontend API, etc.).

---

3. Summary

The system integrates multimodal ML, medical-tuned LLM reasoning, and a unified backend orchestration layer. Data flows cleanly from user → embeddings → prediction → LLM → chat output. Cloud-native infrastructure (Modal, Vertex AI, Kubernetes) ensures scalable and reliable model deployment. The design supports secure handling of medical images while providing fast, accurate, and interpretable dermatology-oriented guidance.