**Project Design Phase**

**Solution Architecture**

|  |  |
| --- | --- |
| Date | 27 June 2025 |
| Team ID | LTVIP2025TMID34533 |
| Project Name | Pollen Profiling |
| Maximum Marks | 4 Marks |

**Solution Architecture:**

The solution architecture for the *Pollen’s Profiling* project is designed to automate the classification of pollen grain images using a deep learning pipeline deployed through a web interface. The architecture bridges the gap between the domain problems of manual pollen identification and technology-driven automation using AI.

**🔹 Architecture Objectives**

* To automate image-based classification of pollen grains with high accuracy.
* To simplify the workflow for end-users like researchers and doctors using an intuitive UI.
* To ensure scalability and integration across healthcare, environmental, and agricultural domains.

| **Component** | **Description** |
| --- | --- |
| **User Interface** | A web-based UI (using Flask or Streamlit) for users to upload pollen images and view classification results. |
| **Image Preprocessing Module** | Performs resizing, normalization, and transformation of input images to match model requirements. |
| **Deep Learning Model** | A Convolutional Neural Network (CNN) trained on labeled pollen image data for multi-class classification. |
| **Training Data Pipeline** | Used to train and validate the model using labeled datasets. |
| **Prediction Engine** | Receives user-submitted images, processes them, and returns class predictions. |
| **Results Storage (Optional)** | Stores output predictions and metadata for future queries or research analytics. |

**Architecture Flow**

1. User uploads pollen image via UI.
2. Image is preprocessed for quality, size, and format.
3. The trained DL model classifies the image and returns a label.
4. Prediction results are displayed on the interface.

**Technology Stack**

* **Frontend**: HTML, CSS, Streamlit or Flask
* **Backend**: Python, TensorFlow/Keras
* **Data Storage**: Firebase / SQLite (for image results and metadata)
* **Model**: CNN architecture trained on custom pollen image dataset
* **Deployment**: Streamlit Cloud / AWS EC2 / Heroku

**Development Phases**

1. Data Collection & Preprocessing
2. Model Building & Training
3. Model Evaluation & Optimization
4. Frontend-Backend Integration
5. Deployment &Testing

**Scalability Considerations**

* Modular design for adding new pollen classes or other allergens.
* Can integrate with APIs for real-time environmental data.
* Scalable cloud deployment ensures availability for a broader user base.

**Solution Architecture Diagram:** 