**Detailed Design**

**Architecture:**

We will be using the MVC (Model View Controller) architecture:

**Model:**

* Manages user-uploaded eye photos, ML model predictions, and user data.
* Handles data storage, retrieval, and processing of eye images and shape classifications.

**View:**

* A user-friendly GUI created using Flutter (or another frontend framework if needed).
* Accepts user input.
* Displays prediction results and additional analytics.

**Controller:**

* Handles user actions (like uploading images of eyes) and sends data to the Model for processing.
* Updates the View with results from the Model.

**Data Storage:**

Types of stored data:

1. **Eye images:**

* Stored in Google Cloud Storage
* Images are saved with unique identifiers for retrieval.

1. **Predictions:**

* Stored in a relational database (like MySQL) with the following metadata:
* User ID
* Image URL
* Predicted eye shape
* Confidence score
* Timestamp

**Graphic Description:**

תמונה שמכילה טקסט, צילום מסך, קו, גופן

התיאור נוצר באופן אוטומטי

**Data Description:**

Stored metadata:

1. **Eye images metadata:**

* Unique image identifier
* File path or URL
* Upload timestamp

1. **Prediction Results:**

* Eye shape category (almond, round, hooded)
* Confidence score

**API Specification:**

1. The system displays a list of user operations.
2. The user selects the 'take photo' option.
3. The user takes photo of client's eyes.
4. The system takes the user's photo and analyzes the image.
5. The system presents the best fitting silicone shield for the client.

**Interface Design:**

**App Structure:**

1. Input**:**

* Uploads eyes photo (via camera or from photos library).

1. Processing:

* Sends photo to the backend through REST API.

1. Output:

* Displays predicted eye shape and confidence score.

**Programming Languages and Tools:**

1. Frontend:

* Flutter (for a cross-platform mobile application)

1. Backend:

* Python
* TensorFlow or PyTorch for ML model.

1. Database:

* MySQL (relational database for structured data).

1. Cloud Service:

* Google Cloud Storage for images storage.
* Deployment on Google Cloud Run.

1. ML Libraries:

* TensorFlow or PyTorch.
* OpenCV for image preprocessing.

**Algorithms Description:**

1. Detect eye and key facial landmarks:

* Use a single algorithm, like Mediapipe Face Mesh or Dlib Facial Landmark Detection, to detect the eye and its surrounding region.
* There tools provide landmarks for the eye contour, eyelids, and surrounding areas, which can be used for both eye shape classification and eyelash angle calculation.

1. Crop and analyze the eye region:

* Use the detected eye landmarks to crop the eye region.
* Extract features for eye shape classification (curvature, aspect, ratio) and locate points along the lash line.

1. Calculate eyelash growth angle:

* From the lash points detected in step 1:
* Identify base and tip points of eyelashes.
* Calculate the angle of growth using geometric formulas.

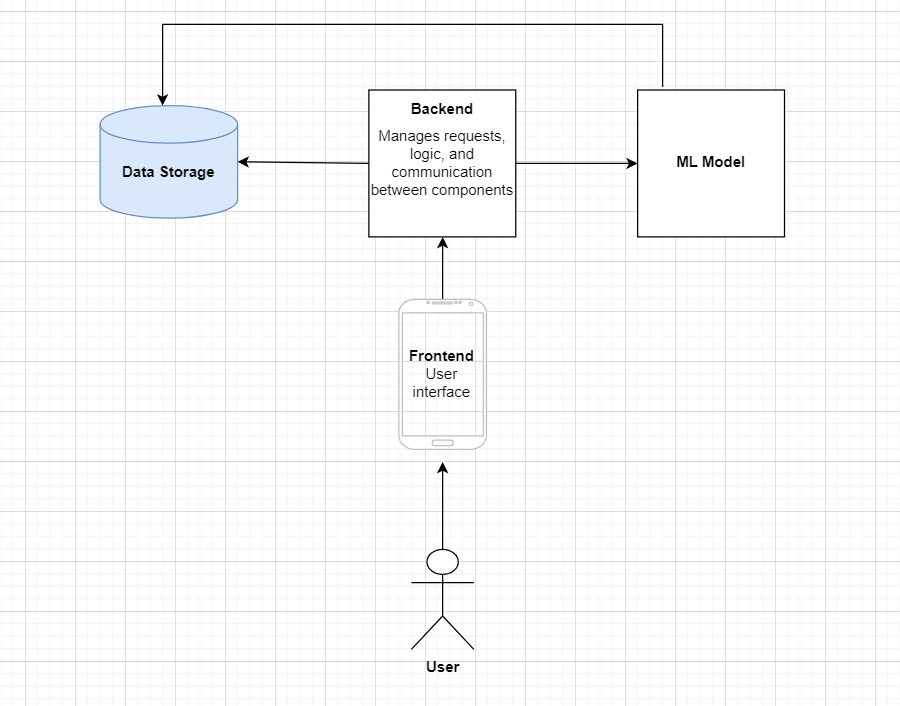
1. Classify eye shape and calculate lash angle:

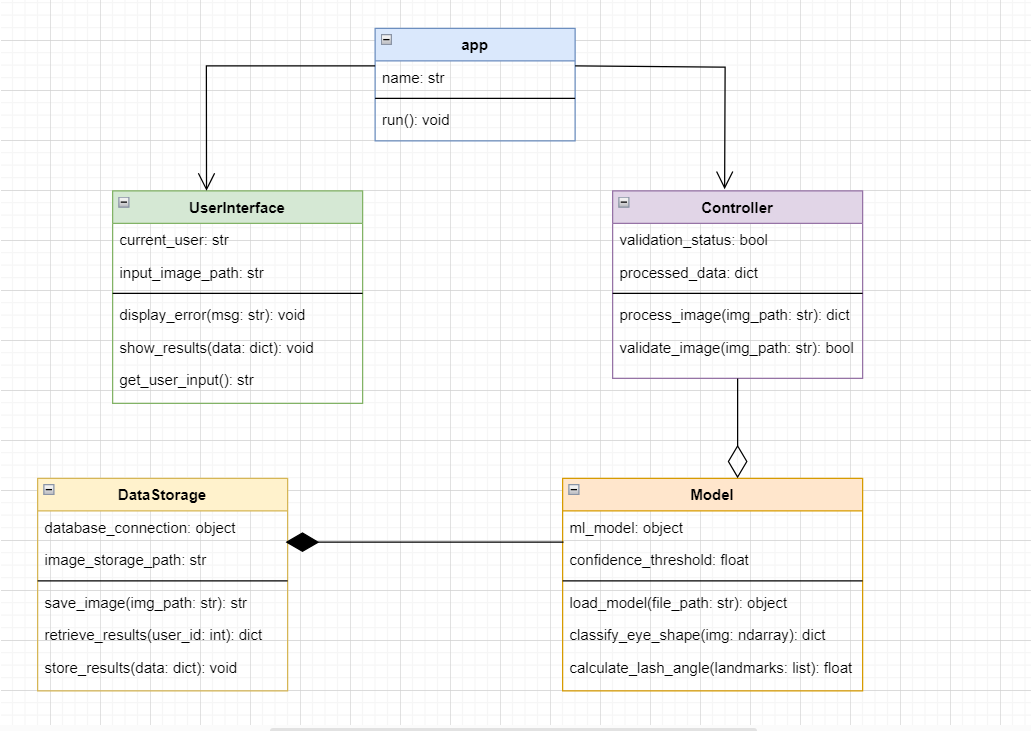
* Send the cropped eye region into the ML model for eye shape classification.
* Combine this result with eyelash growth angle calculation to create an analysis.

1. Return result:

* Output both the predicted eye shape and the calculated eyelash growth angle in a single result.
* Output the best silicone rod option for lashes.

**Architecture Diagram:**



 **UML Class Diagram:**