**Design Doc**

**-Project Galleria**

**Saroj, Pranay, Ravanpreet, Pranav, Rohan**

**OVERVIEW:**

Galleria is an attractive yet simple photo book web- application catering the picture lovers to conceal their memories and even share them on the very same platform. Galleria is not just a usual image gallery application, it’s an application with minimal design themes and maximum engagement thoughts. Galleria provides the users to edit, display, share and relive the memories once again.

It brings together the attractive yet simple UI helping even the non-tech background people to come up and use it with their daily needs. A picture gallery, a photo editor, a social media app, and memory storage all under one roof, “GALLERIA.”

**Context:**

A gallery of images is not just a picture storage or viewing application, it’s an application helping you to relive your memories once again and experience the same emotional experience from it. Galleria raises the bar and takes it to the next level. It’s providing you with the power to store, edit, share and relive your past once again.

It’s open to all applications but believes to be a gem among photogenic people. Our team wants to introduce you to the initial version of this gem in the next 12 days but promises to deliver a full-fledged application with more fun and attractive functions in the future.

**TECH SPEC**

* HTML, CSS, JavaScript (frontend)
* Django framework (server-side execution)
* Cropper.js (for cropping images)
* PhotoEditorSDK (for doodling over images)
* Thumbnails-PIL library (for storing thumbnails for images)

**IMAGE UPLOAD**

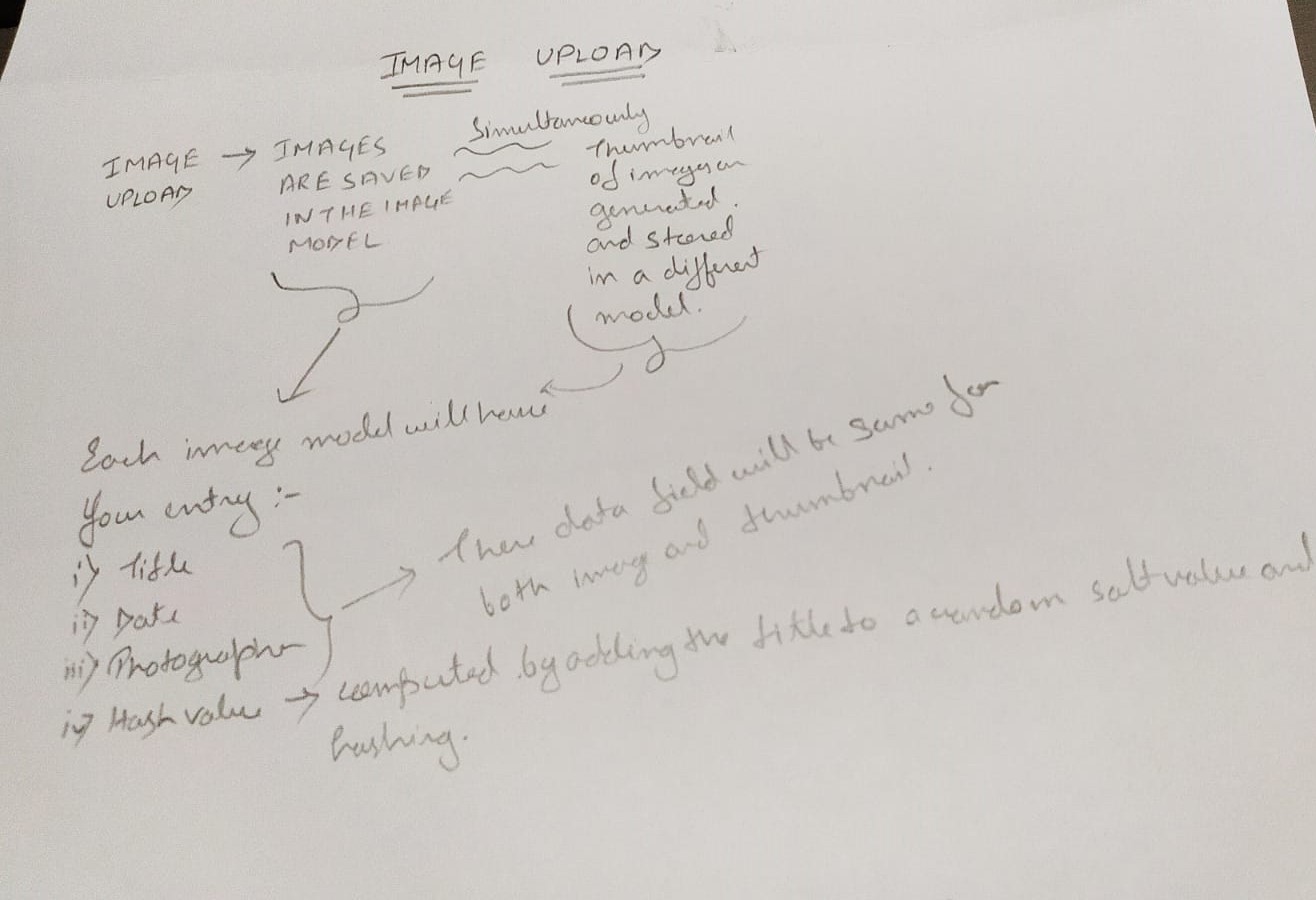
* The images will be uploaded by the user on our web application.
* All those images will get saved in the image model through the post request which the server will get from the client side.
  + The image model will have respective entries:
    - Title of the image
    - Date of upload
    - Photographer
    - Hash Values
      * The hash values will be computed by adding the title with random the salt value and hashing which will create a unique value to distinguish the different images.
* During this process only the thumbnails of the images will be generated and stored in a different model.
  + The thumbnails generated will also have same entries and will have the same metadata as of the image model.

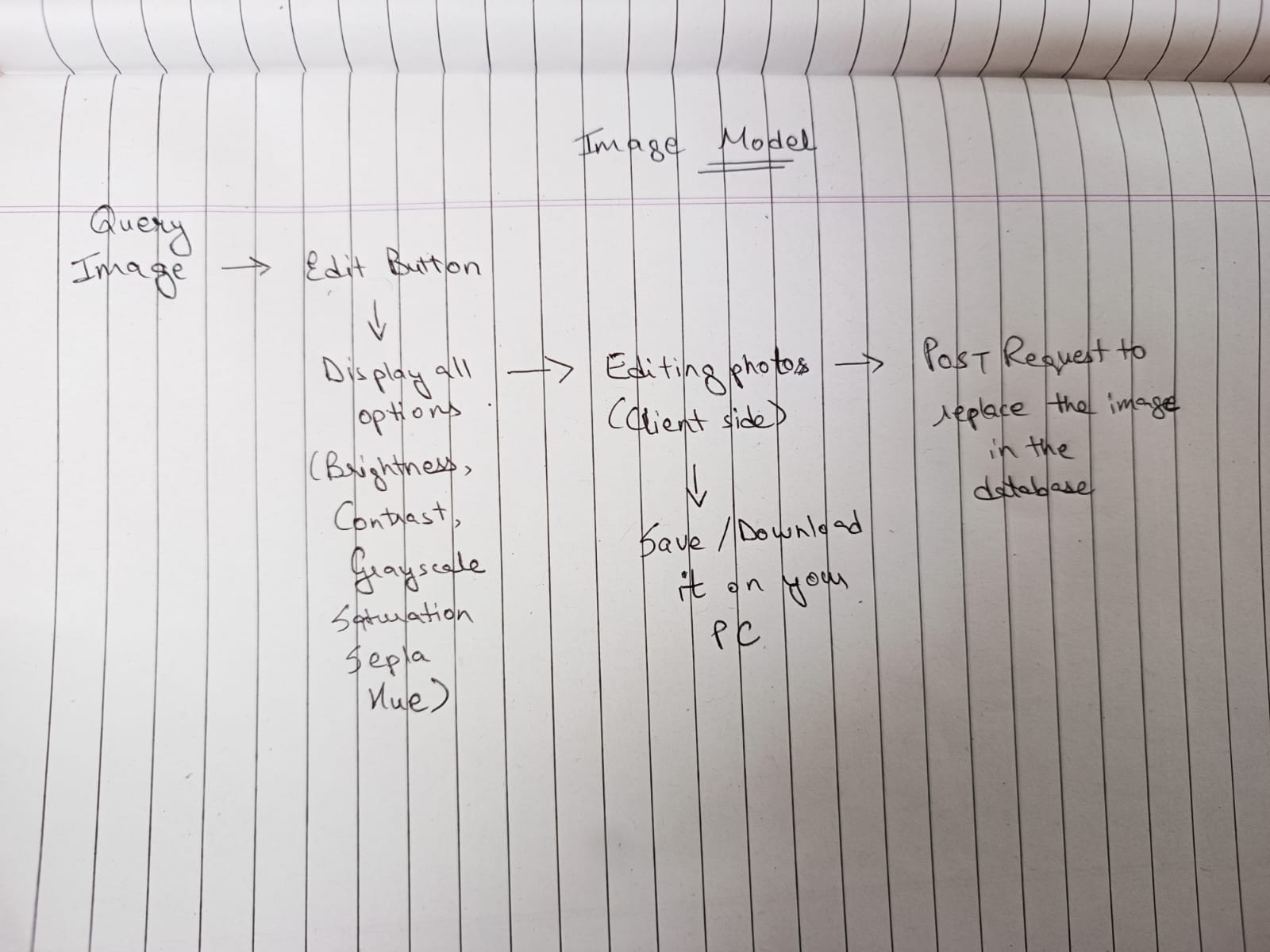
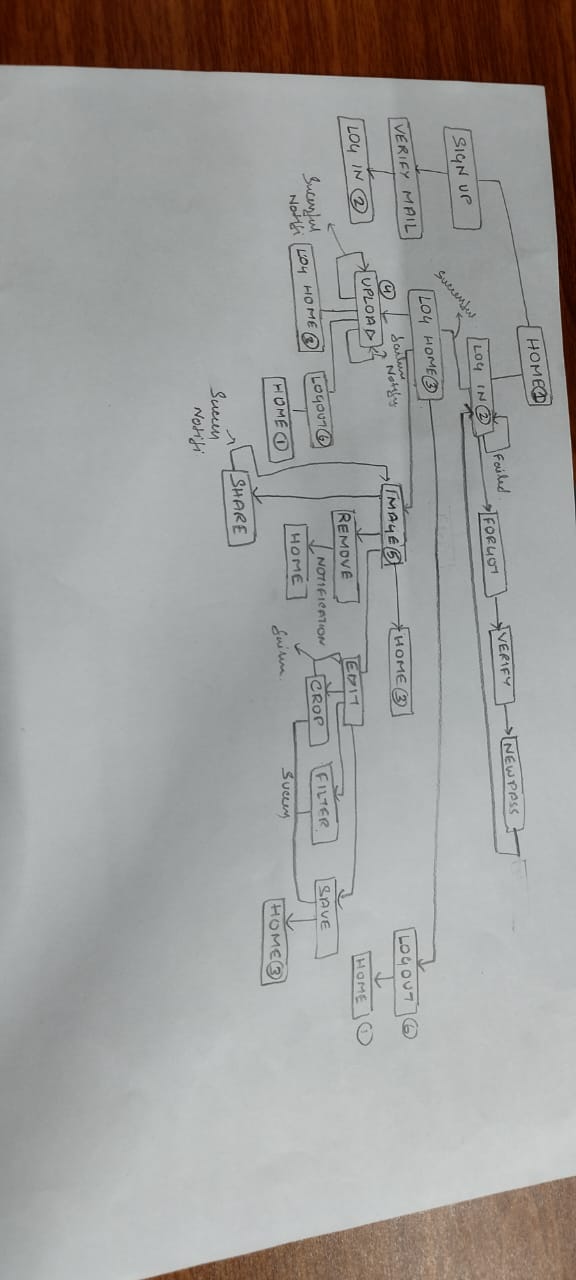
**IMAGE EDITTING**

* When the user adjusts the slider for brightness or contrast, the designated function gets called on the backend which selects the image by its id and adds the brightness or contrast filter with the adjusted slider value.
* User willing to crop an image can select the cropping option and select the area of interest in it. The Cropper.js is the pure JavaScript version of the jQuery Image Cropper plugin which provides the feature-rich image cropping functionality on any image.

**HOME PAGE**

* Device will search the image on the server.
* The server will display the images from the thumbnail model and will display all the images which the user had uploaded.
* **What Happens when the user clicks on the image.**
  + The hash values of the images which the image model has created while uploading it, will be matched with the main image which the user has clicked and if it matches the image gets displayed.
* **What happens when the user saves the image after an edit.**
  + A post request will be given by the client to the server.
  + Then the image sends all the metadata and the image is replaced by the modified one in the image model.
  + Again, the thumbnail generating process begins and the same values are given to both the images.
* **What happens if the user removes the images.**
  + The hash value of the image to be deleted is matched with the thumbnail.
  + If it matches the image gets deleted along with its thumbnail and hash values.





.