Photo Editor Technical Design Document

1. Tech Stack

The photo editor is a web application which runs on Django REST API framework. There are various technologies that will be used for the entire application as listed below.

1.1 Frontend :

React js is used to build the front end UI interface. It allows us to use reusable components and efficiently update DOM. HTML and CSS is used to style the UI pages and create the visual layout of the web pages. Javascript is used with the above entities and performs operations on the frontend and communicates with the APIs. Bootstrap is also used to make the website responsive. This provides the best pre-defined UI designs for components such as buttons, forms, navigation bars etc.

1.2 Backend :

Python is used for the entire backend development for both features development and API building. Postgres will be used as a backend database. Apart from that there are various tools which will be used to develop the features are OpenCv, numpy, Pillow, Matplotlib, Pandas etc.

There are no specific system requirements for the application. Since it's a web application we just need a browser and a proper internet connection.

2. Accounts and Infrastructure

Since it's an web application open to all the users there won't be any account to develop and test. But for deployment will be using kubernetes cluster so there will be a production server details to deploy the product.

2.1 Development

For development we will be using our basic local machines for both frontend and backend. And for testing will be using a development server and production server. We will be deploying on our internal kubernetes cluster.

We will use kubernetes to package our application into containers. Define the desired state of our application using manifests. Push our code to GITHub. Use a CI/CD tool to automate. Expose the application.Lastly,monitor and manage our application.

2.2 Production

Production environment is a kubernetes cluster. Details Unknown .

3.Data Sources, Models, Timing

Since this is an open web application there won't be any database for storing the user data. Here the input will be images which will be used to process the features on it selected by the user. The further data input, output and the expiry will be in the following details.

3.1 Data Sources

The data source in this application is images. The images won't be stored in a database since it is a privacy data we are not allowed to store. We will take the input from the user and will process according to the user requirements and will store a static folder for 24 hrs. It will be auto deleted after that.
Instead of storing the user data we will be storing the user activity and the frequency of features used by the user. It is used to show the statistics of the application used by the various users and the most used features etc.

3.2 Data Models and Structure

The Database contains only one table to store the user activity. The Database **Activity relation** contains the following attributes. **ID, Features, Count.**

For the API Input and output will be in JSON format.

Sample Input :

```
{
   "Image" : "Input Image Url",
   "Features" : "predefined keys for each feature"
}
```

Sample Output :

Successful Message

```
{
   "Output Image" : "Downloadable processed image url",
   "Message" : "Processed successfully ",
   "Status" : "200 ok"
}
```

Error Message :

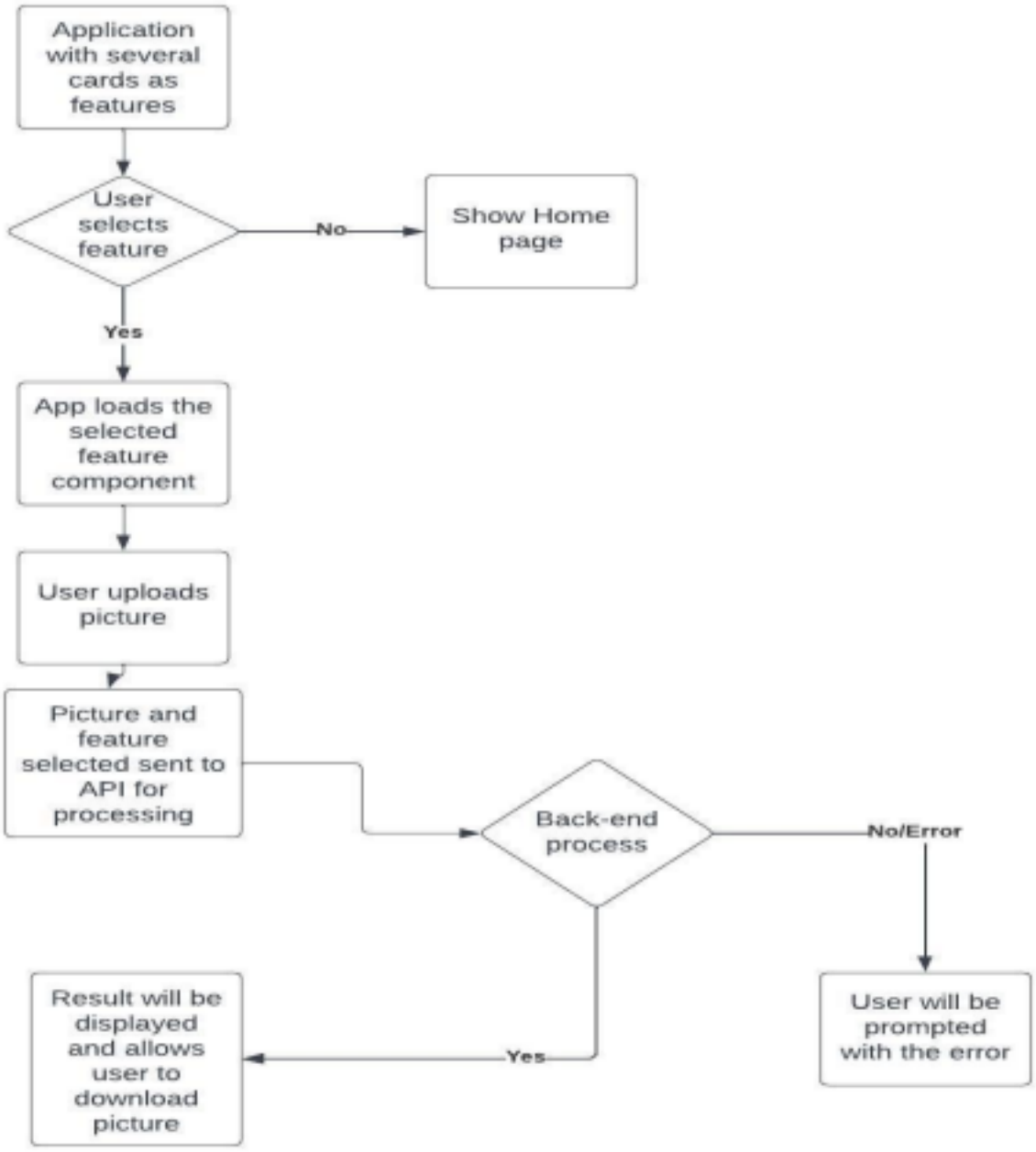
```
{
   "Error Message" : "Proper error message ",
   "Status" : "404"
}
```

3.3 Timing

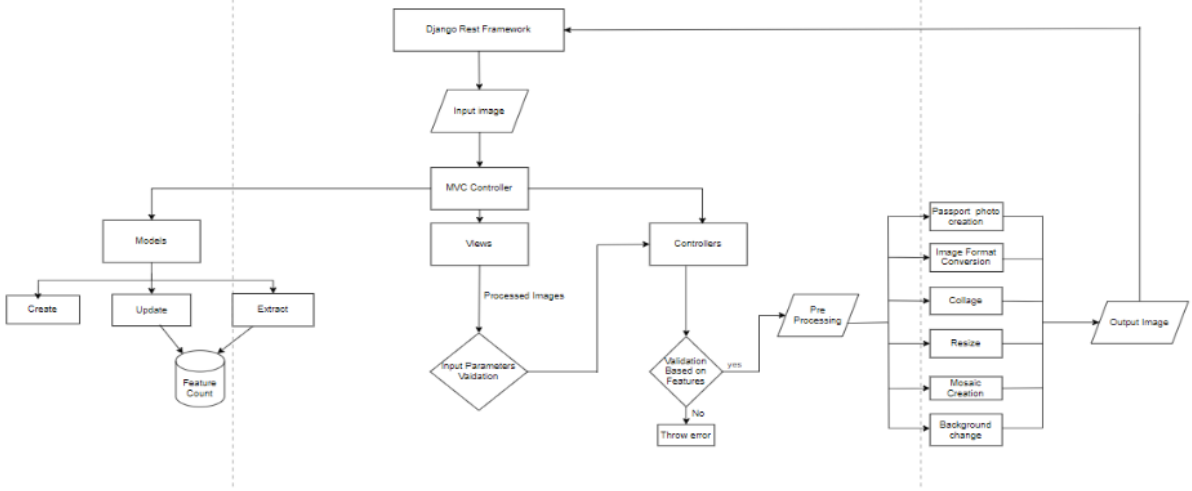
There won't be any data stored in the system or database related to the users, but the photos which the user has used to process some operation on it will be stored for 24 hrs and then it will be auto deleted since there is a option for user to download we can't remove the photo immediately after processing, till the time the photo will be in static folder in the backend REST API. Later on it will be deleted automatically.

System Architecture Diagram

Frontend Workflow



Backend Workflow



Deployment Methodology

The final web application will be deployed in the kubernetes cluster; either the entire code or a Docker image will be used for deployment.