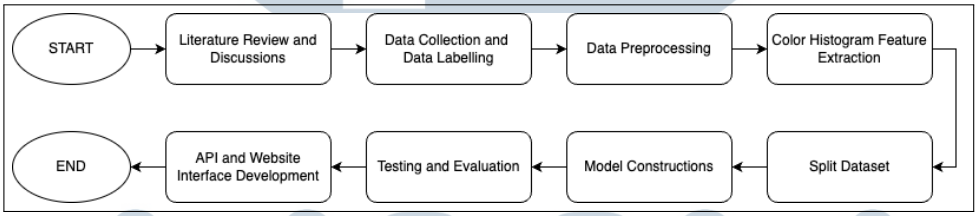
BAB 3
METODOLOGI PENELITIAN

Penelitian ini dilakukan melalui beberapa tahapan yang sistematis, seperti yang ditunjukkan pada Gambar 3.1. Proses dimulai dengan studi literatur dan diskusi untuk memahami dasar teori yang relevan. Selanjutnya, dilakukan pengumpulan dan pelabelan data sebelum memasuki tahap preprocessing, yang mencakup ekstraksi fitur histogram warna. Setelah data diproses, dataset dibagi menjadi data pelatihan dan pengujian untuk memastikan model dapat dievaluasi dengan baik.

Model CNN kemudian dikonstruksi dan disesuaikan dengan pencarian hyperparameter terbaik sebelum dilakukan pengujian serta evaluasi performa. Setelah model mencapai hasil yang optimal, penelitian dilanjutkan dengan pengembangan API dan antarmuka web untuk menerapkan model ke dalam sistem yang dapat digunakan secara praktis.



Gambar 3.1. Flowchart Metodologi Penelitian

3.1 Literature Review and Discussions

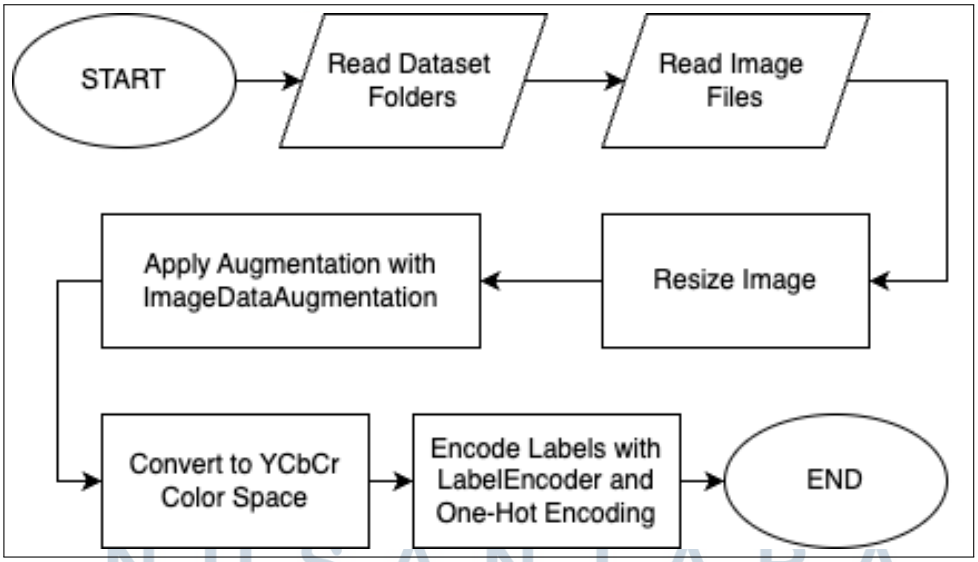
Pada tahap awal, dilakukan studi literatur terkait penelitian terkait, terutama penelitian klasifikasi Batik menggunakan *Convolutional Neural Network* (CNN). Selain melakukan studi literatur, dilakukan juga berbagai macam diskusi dengan dosen pembimbing, dosen ahli Batik, dan juga diskusi dengan pakar Batik Cirebon dari Universitas Muhammadiyah Cirebon untuk mengetahui karakteristik untuk menemukan metode penelitian yang tepat. Hasil akhir dari tahap ini adalah keputusan menggunakan *Convolutional Neural Network* (CNN) dengan tambahan ekstraksi fitur *Color Histogram* untuk mengatasi kompleksitas motif dari Batik Cirebon.

3.2 Data Collection and Labelling

Penelitian ini menggunakan dataset berupa gambar-gambar batik Cirebon yang mencakup empat jenis motif. Dataset tersebut diperoleh dari berbagai sumber di internet, juga dari pusat batik di Cirebon. Keempat motif batik yang digunakan adalah Mega Mendung, Paksi Naga Liman, Singa Barong, dan Sawat Pengantin. Keempat motif ini dimasukkan ke dalam setiap folder yang mewakili nama setiap motif batik untuk dilakukannya *training* dengan label.

3.3 Data Preprocessing

Setelah dilakukannya *data collection* dan *labelling*, dilakukan tahap *data preprocessing* bertujuan untuk meningkatkan kualitas data dengan melakukan augmentasi serta normalisasi sehingga model dapat belajar secara lebih efektif. Tahapan preprocessing yang diterapkan dalam penelitian ini dapat dilihat pada Gambar 3.2. Setiap langkah yang dipilih memiliki tujuan yang jelas dan telah terbukti efektif dalam penelitian sebelumnya [5] [21] [22]. Proses ini juga tidak menambahkan langkah yang tidak perlu (misalnya, filtering, equalization, atau edge detection) yang tidak relevan untuk tugas klasifikasi citra ini agar tetap menjaga efisiensi komputasi tanpa mengorbankan akurasi.



Gambar 3.2. Flowchart Pre-processing

Proses preprocessing dalam metode ini dimulai dengan inisialisasi variabel data untuk menyimpan fitur-fitur hasil ekstraksi dan labels untuk menyimpan

label kelas dari setiap gambar. Selanjutnya, dilakukan konfigurasi augmentasi data menggunakan ImageDataGenerator, yang melakukan transformasi seperti rotasi, translasi, zoom, flipping horizontal, dan shear. Augmentasi hanya mencakup transformasi dasar seperti rotasi, translasi, zoom, flipping horizontal, dan shear karena transformasi ini cukup meningkatkan generalisasi model tanpa menyebabkan distorsi berlebihan yang dapat merusak informasi penting dalam gambar [21].

Program kemudian membaca folder dataset berdasarkan kelas-kelas yang diberikan, dan untuk setiap folder kelas, dilakukan iterasi terhadap file gambar. Gambar yang berhasil dimuat akan di-resize ke ukuran seragam (default: 128x128 piksel) menggunakan cv2.resize dan di-expand dimensinya agar kompatibel untuk augmentasi. Pemilihan ukuran 128×128 piksel didasarkan pada pertimbangan bahwa ukuran ini cukup kecil untuk menjaga efisiensi komputasi, namun tetap cukup besar untuk mempertahankan fitur penting dalam gambar. Seperti yang dikemukakan oleh [22], ukuran 128×128 merupakan nilai optimal dalam pelatihan model berbasis citra, karena tidak menyebabkan kehilangan informasi yang signifikan sekaligus mempertahankan akurasi klasifikasi yang tinggi [23].

Gambar yang telah di-resize selanjutnya melalui proses augmentasi sesuai konfigurasi sebelumnya. Hasil augmentasi kemudian diubah ke ruang warna YCbCr menggunakan cv2.cvtColor, di mana komponen Y (intensitas), Cb (chrominance biru), dan Cr (chrominance merah) diekstrak. Konversi ke YCbCr dipilih karena lebih efektif dalam mengekstrak informasi warna dibandingkan ruang warna RGB, yang sering memiliki korelasi tinggi antar-kanal. Hal ini memungkinkan model untuk menangkap perbedaan fitur tekstur yang lebih baik [5].

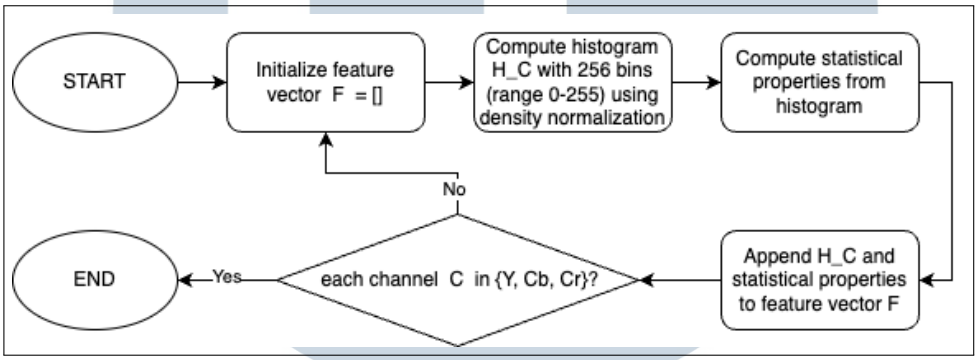
Setelah semua gambar dalam dataset diproses, label kemudian dikodekan menjadi angka menggunakan LabelEncoder dan diubah ke dalam format one-hot vector menggunakan to_categorical. Proses encoding menggunakan LabelEncoder dan one-hot vector memastikan bahwa model dapat memproses label dengan baik dalam bentuk numerik yang dapat diterima oleh algoritma klasifikasi berbasis deep learning [24].

3.4 Ekstraksi Fitur Color Histogram

Tahapan untuk melakukan ekstraksi fitur pada *Color Histogram* dapat dilihat pada Gambar 3.3. Berdasarkan penelitian Sergyan (2008), pola tekstur yang kompleks dan berbasis warna, seperti batik, lebih sesuai untuk dianalisis

menggunakan Color Histogram karena mampu menangkap distribusi warna secara efektif tanpa kehilangan informasi tekstur yang signifikan. [5].

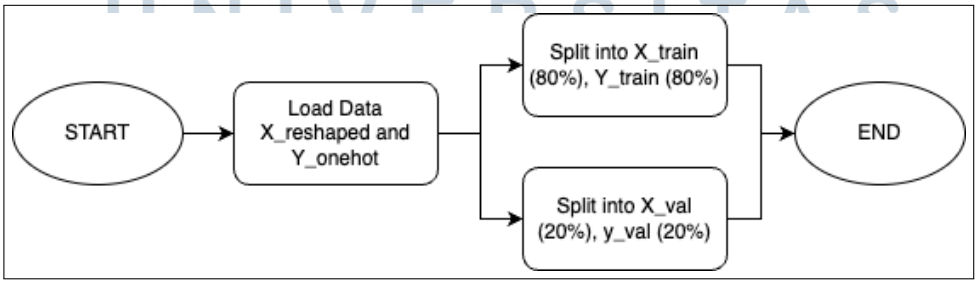
Untuk setiap channel warna pada YcbCr ini, histogram intensitas dihitung menggunakan np.histogram dengan 256 bins, dan fitur-fitur statistik seperti *mean*, *standard deviation*, *skewness*, *energy*, dan *entropy* dihitung melalui fungsi *histogram_properties* sesuai alur pada Gambar 3.3. Fitur-fitur ini dirangkum menjadi satu list yang kemudian ditambahkan ke dalam variabel data digabungkan dengan labels yang telah diproses untuk dilakukan *feature reshape* menjadi input model.



Gambar 3.3. Flowchart Ekstraksi Fitur Color Histogram

3.5 Split Dataset

Pembagian dataset merupakan langkah penting dalam proses pelatihan model, karena memastikan bahwa model tidak hanya menghafal data pelatihan tetapi juga mampu menggeneralisasi pada data yang belum pernah dilihat sebelumnya. Pada penelitian ini, dataset dibagi menjadi dua bagian utama, yaitu data pelatihan dan data pengujian. Proses pembagian dataset ini dapat dilihat pada Gambar 3.4.



Gambar 3.4. Flowchart Split Dataset

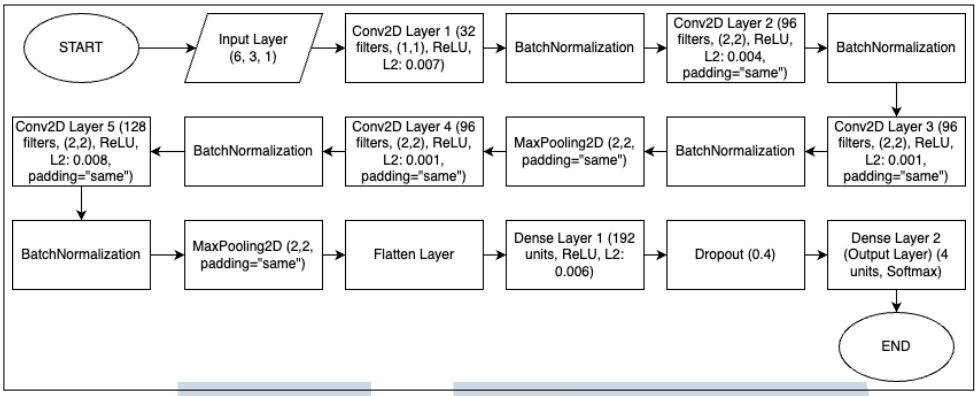
Pada langkah ini, dataset dibagi menjadi dua bagian, yaitu data pelatihan dan data pengujian, menggunakan fungsi train_test_split dari *library* sklearn. Data fitur yang telah diproses sebelumnya (X_reshaped) dan label yang telah diubah ke dalam format one-hot encoding (y_onehot) digunakan sebagai input. Sebanyak 80% data dialokasikan untuk pelatihan model, sementara 20% sisanya digunakan untuk menguji performa model. Parameter test_size=0.2 menentukan proporsi data yang digunakan untuk pengujian, sedangkan random_state=42 memastikan bahwa pembagian data dilakukan secara konsisten setiap kali kode dijalankan, sehingga hasilnya dapat direproduksi. Output dari proses ini adalah empat bagian: X_train dan y_train untuk data pelatihan, serta X_test dan y_test untuk data pengujian. Pembagian dataset ini penting untuk mengevaluasi kemampuan model dalam menggeneralisasi data baru yang belum pernah dilihat sebelumnya.

3.6 Model Constructions

Dalam tahap ini, dilakukan finalisasi model, di mana eksperimen dengan berbagai skenario arsitektur CNN untuk menemukan kombinasi terbaik. Percobaan ini mencakup variasi jumlah lapisan konvolusional, yaitu satu, dua, tiga, dan lima lapisan, serta eksplorasi jumlah filter pada setiap lapisan konvolusi, penggunaan dan ukuran pooling layer, berbagai tingkat dropout untuk mengurangi overfitting, serta penerapan regularisasi L2 guna meningkatkan generalisasi model.

Model Convolutional Neural Network (CNN) ini dikembangkan untuk mengklasifikasikan motif Batik Cirebon berdasarkan fitur yang telah diekstraksi. Untuk meningkatkan performa model, dilakukan proses tuning hyperparameter menggunakan Keras Tuner. Diagram alur dari proses pembangunan dan tuning model CNN ini dapat dilihat pada Gambar 3.5.

Pemilihan model dilakukan berdasarkan eksperimen dengan berbagai konfigurasi jumlah lapisan konvolusional. Dari beberapa percobaan, arsitektur dengan lima lapisan konvolusi memberikan hasil paling optimal. Flatten Layer, MaxPooling, Dense Layer, dan Dropout Layer digunakan berdasarkan penelitian yang dilakukan oleh Arya et al., yang telah membuktikan efektivitas metode ini dengan kedalaman 6 layer dalam klasifikasi batik menggunakan CNN, dengan akurasi yang tinggi [25].



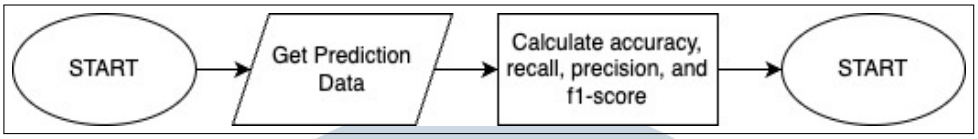
Gambar 3.5. Flowchart Model CNN setelah Hyperparameter Tuning

Pada langkah ini, dilakukan pembangunan model CNN (*Convolutional Neural Network*) untuk klasifikasi gambar menggunakan *library* Keras Tuner untuk pencarian *hyperparameter* terbaik. Model awal dirancang menggunakan fungsi build_model, yang mendefinisikan arsitektur CNN dengan parameter seperti jumlah filter, ukuran kernel, ukuran pooling, jumlah unit dalam lapisan dense, dan jenis optimizer. Parameter ini diatur agar dapat diuji berbagai kombinasi nilai melalui hp.Choice dan hp.Int. Model terdiri dari beberapa lapisan, termasuk lapisan convolutional dengan fungsi aktivasi ReLU, lapisan pooling untuk pengurangan dimensi, lapisan dense untuk pengambilan keputusan, dan lapisan output dengan fungsi aktivasi softmax untuk klasifikasi multikelas. Model dikompilasi dengan fungsi loss categorical_crossentropy dan metrik accuracy.

Proses tuning dilakukan menggunakan kt.RandomSearch, yang menguji hingga 10 kombinasi hyperparameter dengan dua eksekusi per kombinasi untuk mencari model dengan akurasi validasi terbaik (val_accuracy). Selama pencarian, callback EarlyStopping diterapkan untuk menghentikan pelatihan jika tidak ada peningkatan signifikan pada loss validasi setelah 10 epoch berturut-turut. Setelah pencarian selesai, hyperparameter optimal ditemukan, seperti jumlah filter (96), ukuran kernel (3, 1), ukuran pooling (2, 1), jumlah unit dalam lapisan dense (192), dan optimizer adam. Hyperparameter ini digunakan untuk melatih model terbaik.

3.7 Testing and Evaluation

Pada tahap evaluasi, performa model diuji menggunakan data pengujian untuk menilai efektivitas prediksi yang telah dilakukan. Diagram alur evaluasi model ditampilkan pada Gambar 3.6, yang menggambarkan tahapan dari prediksi hingga perhitungan metrik evaluasi.



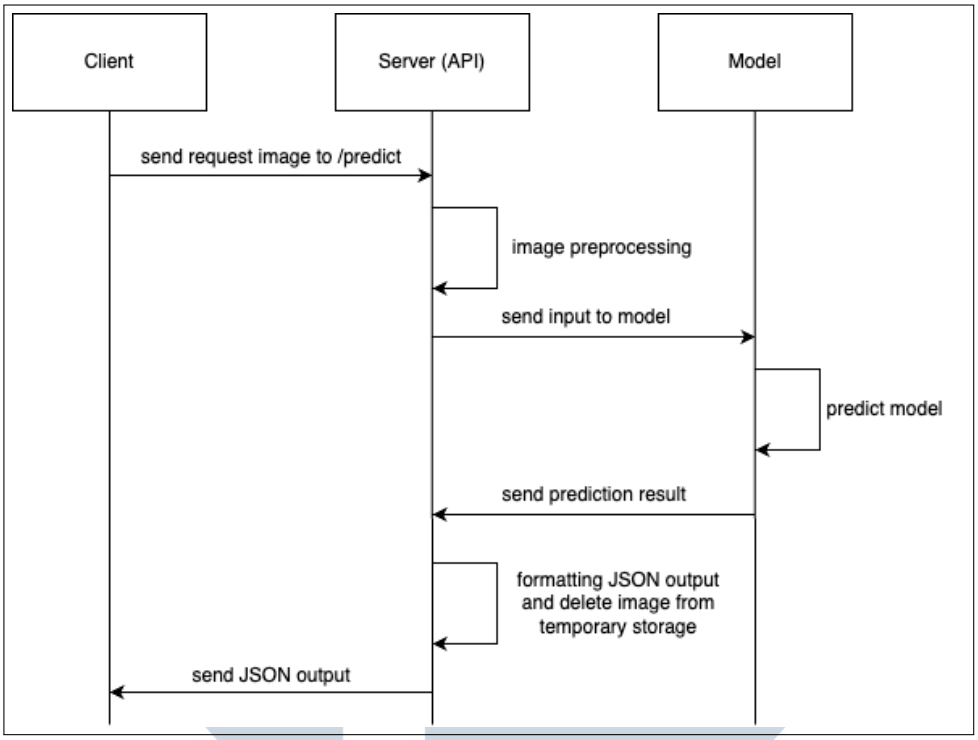
Gambar 3.6. Flowchart Evaluation

Pada langkah ini, dilakukan evaluasi model terhadap data pengujian untuk mengukur performa prediksi. Pertama, model terbaik yang telah dilatih (best_model) digunakan untuk memprediksi label dari data fitur pengujian (X_test) menggunakan metode .predict. Hasil prediksi (y_pred) berupa probabilitas untuk setiap kelas. Probabilitas ini kemudian diubah menjadi label kelas prediksi dengan menggunakan fungsi np.argmax, yang memilih indeks dengan probabilitas tertinggi pada setiap prediksi (dalam hal ini, sepanjang sumbu axis=1 untuk data multi-kelas). Hal serupa dilakukan pada label data pengujian asli (y_test), yang juga diubah dari format one-hot encoding menjadi label kelas sebenarnya menggunakan np.argmax.

Setelah itu, hasil prediksi (y_pred_classes) dibandingkan dengan label asli (y_test_classes) untuk menghitung berbagai metrik evaluasi, seperti presisi, recall, f1-score, dan akurasi, menggunakan fungsi classification_report dari scikit-learn. Metrik ini disajikan untuk setiap kelas dalam dataset, dengan nama kelas diberikan melalui parameter target_names yang berisi daftar nama kelas (selected_classes). Proses ini memberikan gambaran menyeluruh tentang seberapa baik model mengklasifikasikan data pengujian, baik secara keseluruhan maupun untuk masing-masing kelas.

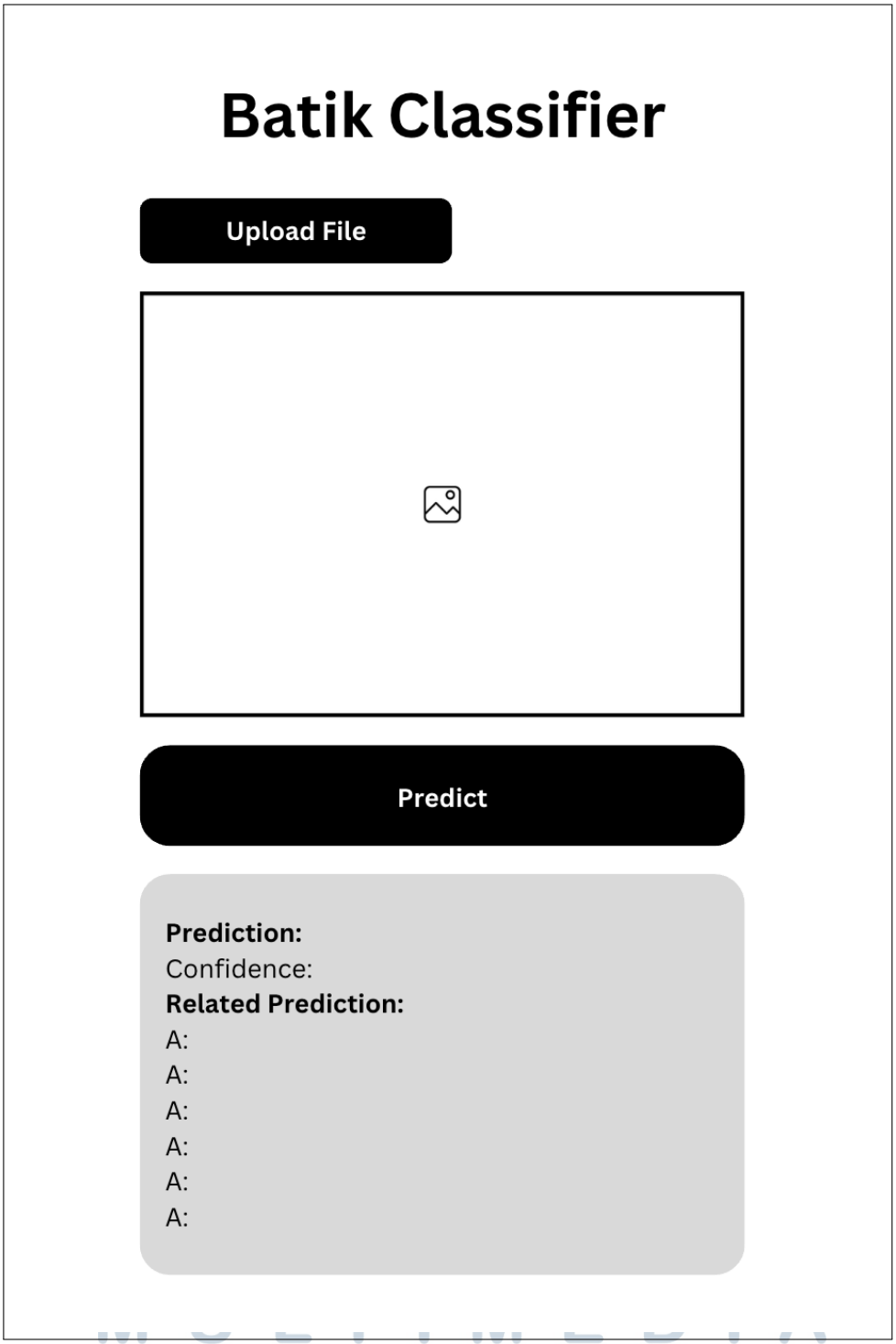
3.8 API and Website Interface Development

Pada tahap ini, dilakukan pembentukan API (*Application Programming Interface*) menggunakan framework Flask untuk menyediakan layanan prediksi gambar batik berdasarkan model CNN yang telah dilatih sebelumnya menggunakan Python. Gambar 3.7 adalah diagram urutan yang mendeskripsikan alur sistem, terutama pada bagian API. Ketika *client*, melakukan *request* POST ke *endpoint* ”/predict”, maka server API akan melakukan *preprocessing* sebelum melakukan predict ke dalam model. Setelah mendapatkan hasil dari model, server API akan melakukan *formatting*, di mana output JSON akan dimodifikasi agar dapat menampilkan prediksi utama dan prediksi lainnya beserta tingkat *confidence* dari masing-masing kelas.



Gambar 3.7. Sequence Diagram Sistem Klasifikasi Batik Cirebon

Setelah pembentukan API selesai, akan dilakukan pembangunan *website interface* menggunakan React sederhana agar model dapat digunakan secara umum untuk mengklasifikasikan gambar Batik Cirebon lainnya. Gambar 3.8 adalah *wireframe* desain website yang akan dibangun.



Gambar 3.8. Wireframe Website Klasifikasi Batik Cirebon