# BAB III ANALISIS DAN PERANCANGAN

## Analisis

### Identifikasi Masalah

Deteksi penyakit kulit adalah penentuan dari suatu kondisi yang terjadi pada lapisan luar tubuh manusia dengan masalah seperti iritasi atau peradangan, yang sering menyebabkan gatal, bersisik, nyeri, kemerahan, bahkan mati rasa dan cacat yang dilakukan oleh dokter spesialis atau ahli. Dokter spesialis atau ahli sebagian besar hanya dapat ditemui di Rumah Sakit tertentu. Jumlah kasus penyakit kulit di Indonesia belum memiliki nilai pasti pada keseluruhan penyakit kulit. Tetapi angka kasus berdasarkan jenis penyakit kulit dapat menjadi gambaran masalah penyakit kulit. Contohnya penyakit kulit scabies, di tahun 2008 jumlah kasus scabies di seluruh Puskesmas di Indonesia berkisar antara 5,6% hingga 12,9%. Sebaran kasus terbesar penderita penyakit kulit didaerah pesisir sungai karena keterbatasan air bersih dan daerah yang padat penduduk (Kurniawan & Ling, 2020). Dimana yang terjadi pada masyarakat seringkali penyakit kulit tersebut tidak diobati dengan baik dan menimbulkan masalah serius pada kulit seperti cacat, tidak dapat disembuhkan, kanker kulit, atau bahkan kematian. Hal ini karena tidak banyak orang yang mampu atau sadar untuk melakukan periksa penyakit kulit ke Rumah Sakit..

Berdasarkan identifikasi masalah tersebut maka peneliti menggambarkan masalah secara umum pada tabel berikut :

**Tabel 3.1** Identifikasi Masalah

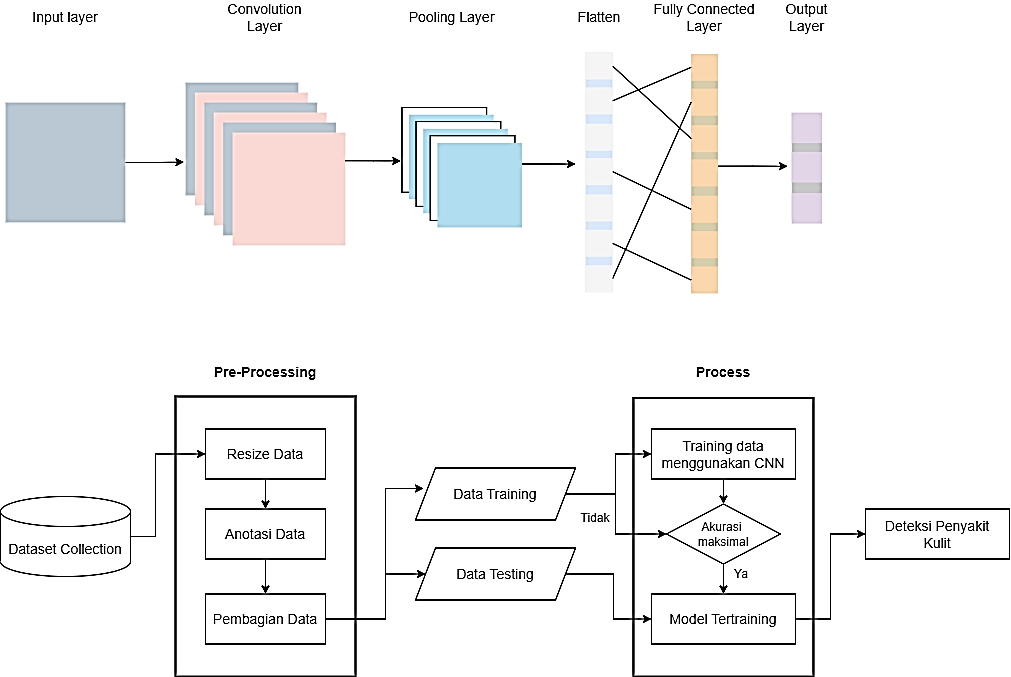
|  |  |  |
| --- | --- | --- |
| **Akar Masalah** | **Penyebab** | **Dampak** |
| Identifikasi dilakukan manual dengan menggunakan objek visual mata oleh dokter spesialis atau ahli | Tidak banyak orang yang mampu hanya untuk periksa penyakit kulit yang diderita ke dokter spesialis atau ahli di Rumah Sakit | Telat penanganan penyakit kulit yang diderita oleh masyarakat |
| Pengujian deteksi penyakit kulit menggunakan metode jaringan syaraf tiruan dan SVM masih belum bisa menghasilkan deteksi dengan akurasi yang maksimal. | terdapat data yang memiliki gejala sama tetapi hasil diagnosis berbeda, kurannya dataset yang digunakan | Pengujian pada data tidak akurat |

### Pemecahan Masalah

Berdasarkan uraian diatas, maka pemecahan masalah yang dapat dilakukan adalah dengan membuat sistem deteksi penyakit kulit yang dapat diakses oleh masyarakat dengan mudah dan cepat. Sistem deteksi kulit menggunakan metode CNN karena terbukti telah menghasilkan akurasi deteksi gambar yang maksimal. Maka deteksi dapat dilakukan hanya melalui foto atau gambar saja.

## Perancangan

### Perancangan Sistem CNN

**Gambar 3.1** Diagram blok alur sistem

Pada proses perancangan terdapat 3 proses yaitu *pre-processing* data, *process* data dan *post-processing* data. Alur proses perancangan diawali dengan pengumpulan dataset kemudian data masuk pada *pre-processing* data. Pada proses ini data diubah ukurannya untuk mempercepat waktu komputasi. Lalu data di anotasi yaitu setiap data dilabeli pada objek posisi penyakit. Kemudian data dibagi menjadi 2 yaitu data *training* dan data *testing*. Data *training* dilatih menggunakan metode *convolution neural network* hingga mendapat akurasi yang maksimal. Lalu dengan data *testing* diuji menggunakan model yang telah ter*training*. Kemudian mendapatkan hasil deteksi penyakit kulit.

1. **Dataset *Collection***

Dataset collection adalah proses pengumpulan data yang akan digunakan dalam penelitian ini untuk mengidentifikasi berbagai jenis penyakit kulit (Rozaqi dkk., 2021). Pada penelitian ini dataset yang digunakan yaitu dataset yang diambil dari situs [www.kaggle.com](http://www.kaggle.com) dengan nama “*Dermnet*”. Dengan banyaknya penyakit kulit yang ada pada dataset, penelitian ini fokus pada tiga penyakit kulit yaitu eksim, psoriasis dan scabies.

**Tabel 3.2** Jumlah Dataset

|  |  |
| --- | --- |
| **Jenis Penyakit** | **Jumlah Dataset** |
| Eksim | 1000 |
| Psoriasis | 1000 |
| Scabies | 560 |
| **Total** | 2560 |

Dataset tersebut digunakan untuk proses *training* dan *testing*. Berikut beberapa contoh dataset dari tiga penyakit kulit.

**Gambar 3.2** (A) Eksim (B) Psoriasis (C) Scabies

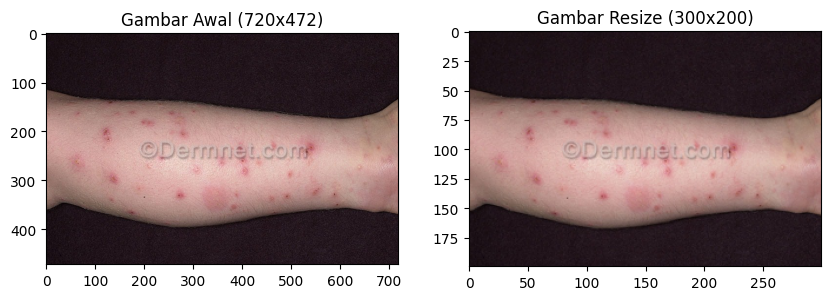
A

C

B

1. ***Resize* Data**

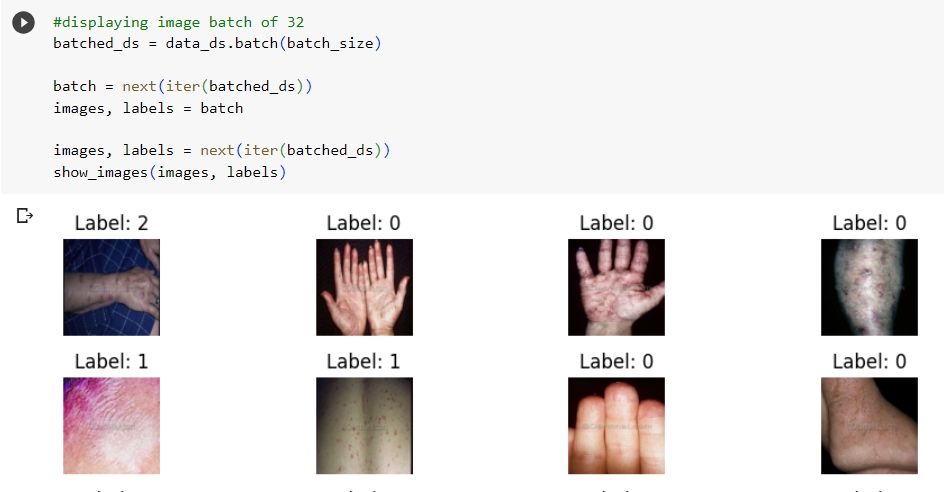
*Resize* data adalah proses pengubahan ukuran gambar dari gambar asal menjadi ukuran yang akan diubah. Pada penelitian ini proses *resize* data dilakukan dengan mengubah ukuran gambar menjadi lebar 300 pixel dan tinggi 200 pixel. Proses *resize* tersebut dilakukan untuk untuk mempercepat waktu komputasi dan membantu mengatasi batasan kapasitas memori (Ho dkk., 2018).



**Gambar 3.3** *Resize* data

1. **Anotasi Data**

Anotasi data digunakan untuk pelabelan objek penyakit pada gambar untuk mendeteksi penyakit. Anotasi dilakukan untuk mempermudah sistem melakukan *training* dalam mengenali pola penyakit. Anotasi data ini biasanya dilakukan dengan melibatkan manusia dengan memberikan label secara manual. Tetapi proses anotasi data pada penelitian ini dilakukan dengan membagi data menjadi *batch-batch* dengan ukuran batch tertentu kemudian mengambil *batch* pertama dari data yang telah dibagi. Lalu gambar dan label dipisahkan dari *batch* tersebut kedalam variabel ‘images’ dan ‘labels’. Kemudian ditampilkan dengan variabel label 0 untuk penyakit eksim, label 1 untuk penyakit psoriasis, dan label 2 untuk penyakit scabies.



**Gambar 3.4** Anotasi Data

1. **Pembagian data**

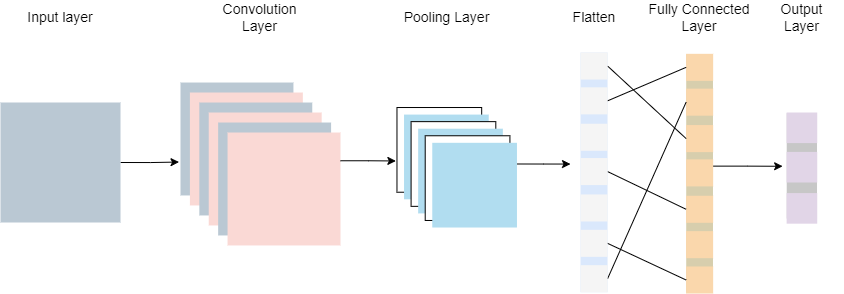
Tahap ini merupakan tahap persiapan untuk pengolahan data dan klasifikasi data (Rozaqi dkk., 2021). Pembagian setiap data pada jenis penyakit dibagi menjadi data *training* dan data *testing*. Data dibagi dengan perbandingan 75% untuk data *training* dan 25% untuk data *testing*. Data *training* digunakan untuk menjalakan *Convolution Neural Network* (CNN) dalam pencarian model yang terbaik, sedangkan data *testing* digunakan untuk menguji model yang telah dihasilkan pada proses *training*.

**Tabel 3.3** Pembagian Data

|  |  |  |
| --- | --- | --- |
| **Jenis Data** | **Data *Training* (75%)** | **Data *Testing* (25%)** |
| Eksim | 750 | 250 |
| Psoriasi | 750 | 250 |
| Scabies | 420 | 140 |
| **Total** | 1920 | 640 |

1. **Proses *Training* Data**

Proses *training* data dilakukan menggunakan metode *convolution neural network*. Pada penelitian ini terdiri dari beberapa layer yaitu *input layer*, *convolution layer*, *pooling layer*, *flatten*, *fully connected layer* dan *output layer*.

****

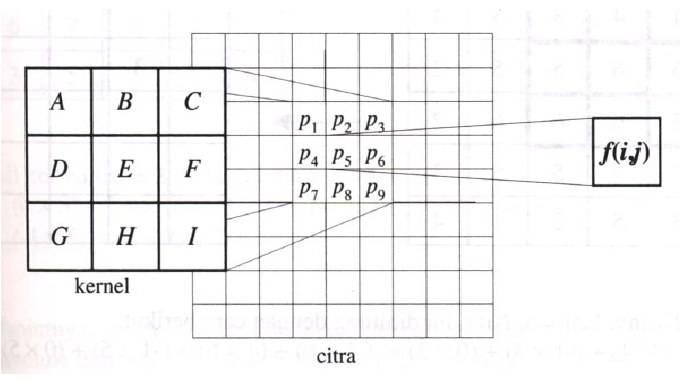
**Gambar 3.5** Arsitektur *Convolution Neural Network*

1. *Convolution Layer*

*Convolution layer* adalah *layer* yang digunakan untuk melakukan operasi konvolusi pada *input layer*. *Layer* ini yang pertama kali menerima gambar yang di*input*kan ke dalam arsitektur. *Convolution layer* terdiri dari *filter-filter* yang dipelajari secara acak untuk melakukan operasi konvolusi dengan tujuan mengerkstraksi fitur untuk mempelajari representasi fitur dari gambar yang di*input*kan. Bentuk *layer* ini berupa sebuah *filter* dengan panjang (*pixel*), lebar (*pixel*) dan tebal sesuai dengan *channel image* data yang di*input*kan. Konvolusi melibatkan penjumlahan hasil perkalian dari setiap elemen kernel dengan setiap titik pada fungsi masukan. Langkah awal dalam konvolusi pada citra adalah mengubahnya menjadi matriks dengan tingkat keabuan (0-255), di mana setiap titiknya memiliki nilai intensitas. Matriks citra tersebut kemudian dikalikan dengan matriks kernel. Pada pengolahan citra, konvolusi dilakukan secara dua dimensi untuk sebuah citra (Affifah & Permanasari, 2022). Maka digunakanlah persamaan berikut :

|  |  |  |
| --- | --- | --- |
|  |  | (3.1) |

Untuk melakukan konvolusi, perlu ditentukan *Stride*, yaitu parameter yang menentukan seberapa besar jarak pergeseran kernel pada saat melakukan konvolusi. Selain itu, dilakukan juga proses Padding, yaitu penambahan piksel tambahan dengan nilai tertentu di sekitar gambar untuk menghindari perubahan ukuran setelah konvolusi. Hasil konvolusi adalah *Feature Map*, yang merupakan representasi hasil fitur dari citra setelah melewati proses konvolusi dengan kernel tertentu. Lapisan konvolusi secara signifikan meningkatkan kompleksitas model melalui optimalisasi outputnya (Perdananto, 2019). Berikut adalah contoh ilustrasi dari proses konvolusi antara matriks citra digital dan matriks kernel :

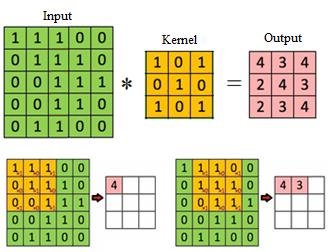


**Gambar 3.6** Proses Konvolusi Citra (Affifah & Permanasari, 2022)

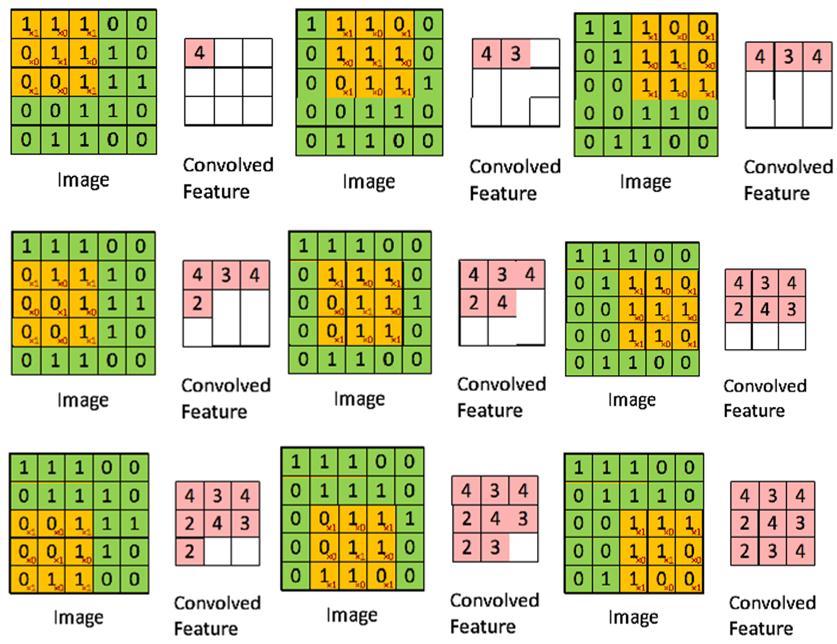
Pada gambar 2.4 menggambarkan *f(i,j)* hasil penjumlahan dari perkalian setiap titik citra dan setiap elemen kernel, seperti persamaan berikut :

|  |  |
| --- | --- |
|  | (3.2) |

Berikut contoh perhitungan konvolusi citra.



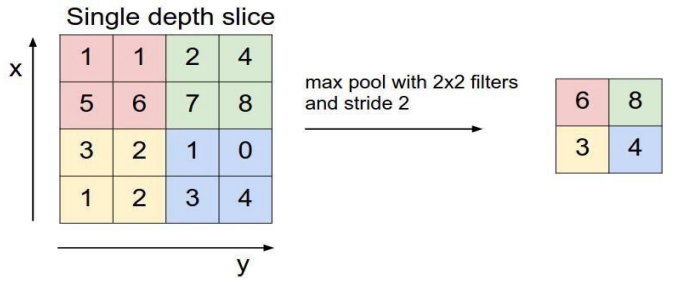
**Gambar 3.7** Contoh Perhitungan Konvolusi Citra (Rozaqi dkk., 2021)



**Gambar 3.8** Contoh Lanjutan Perhitungan Konvolusi Citra (Rozaqi dkk., 2021)

1. *Pooling Layer*

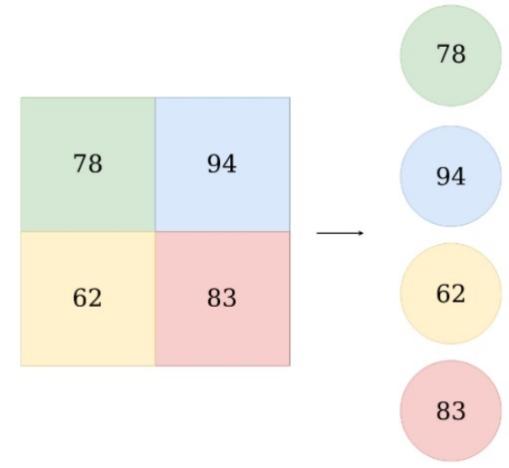
*Pooling layer* terdiri dari sebuah *filter* dengan ukuran dan pergeseran (*stride*) tertentu yang akan bergeser ke seluruh area *activation map* atau *feature map* (Perdananto, 2019). *Pooling layer* sering disisipkan secara teratur setelah beberapa *convolution layer*. Penyisipan *Pooling layer* di antara *convolution layer* berturut-turut dalam arsitektur CNN bertujuan untuk secara progresif mengurangi ukuran *volume output* dari *feature map*, sehingga dapat mengurangi jumlah parameter dan perhitungan di jaringan, serta membantu mengendalikan *overfitting*. *Pooling layer* memiliki beberapa jenis operasi, salah satunya menggunakan *max pooling* 2x2 dengan 2 pergeseran (*stride*) pada setiap filternya.



**Gambar 3.9** Proses *Max Pooling* (Perdananto, 2019)

1. *Flatten*

Hasil dari proses sebelumnya, *feature map* yang dihasilkan masih dalam bentuk data *multidimensional array*. Sedangkan jika digunakan untuk proses *fully connected layer* harus menggunakan data dalam bentuk *vector*. Oleh karena itu, digunakanlah *flatten* untuk mengubah data yang berbentuk *multidimensional array* ke data yang berbentuk *vector* (Felix dkk., 2020).



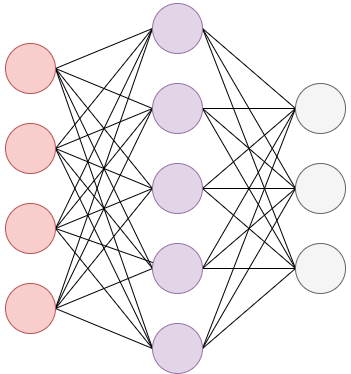
**Gambar 3.10** Proses *Flatten*

1. *Fully Connected Layer*

*Fully connected layer* adalah *layer* yang neuron dari *layer-layer* sebelumnya terhubung semua dengan neuron di *layer* selanjutnya. Oleh karena itu, dibutuhkan *flatten* untuk dapat terhubung dengan *fully connected layer* (Perdananto, 2019). Layer ini digunakan pada akhir jaringan untuk melakukan klasifikasi atau regresi berdasarkan input yang telah diproses sebelumnya. Setelah proses *fully connected layer* maka didapatkan hasil yaitu *output* *layer*. Gambar 2.9 menunjukan proses *fully connected layer*.

Fully Connected Layer

Input Layer



Output Layer

**Gambar 3.11** Proses *Fully Connected Layer*

1. **Model Ter*training***

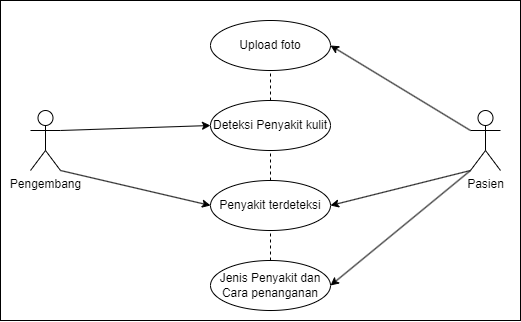
Setelah dilakukan proses *training* data dilatih oleh algoritma CNN untuk menghasilkan model yang siap untuk proses *testing*. Jika model tersebut belum mencapai akurasi yang maksimal maka sistem akan men*training* ulang model dengan batasan *epoch* tertentu sampai mendapat akurasi yang maksimal.

1. **Deteksi Penyakit**

Proses ini juga bisa disebut dengan *testing* model. Tahapan terakhir sistem ini yaitu dapat mendeteksi data *testing* dengan menggunakan model yang telah dilatih pada tahapan sebelumnya. Model yang berjalan pada *testing* ini dapat mengidentifikasi data akan diujikan. Pada proses ini menggunakan data *testing* yang telah terbagi pada pembagian data. Proses *testing* ini hanya melakukan pengujian untuk mengevaluasi nilai bobot dengan pengujian *confusion matrix*.

### Perancangan Sistem Website

1. **Perancangan Use Case**



**Gambar 3.12** *Use case website*

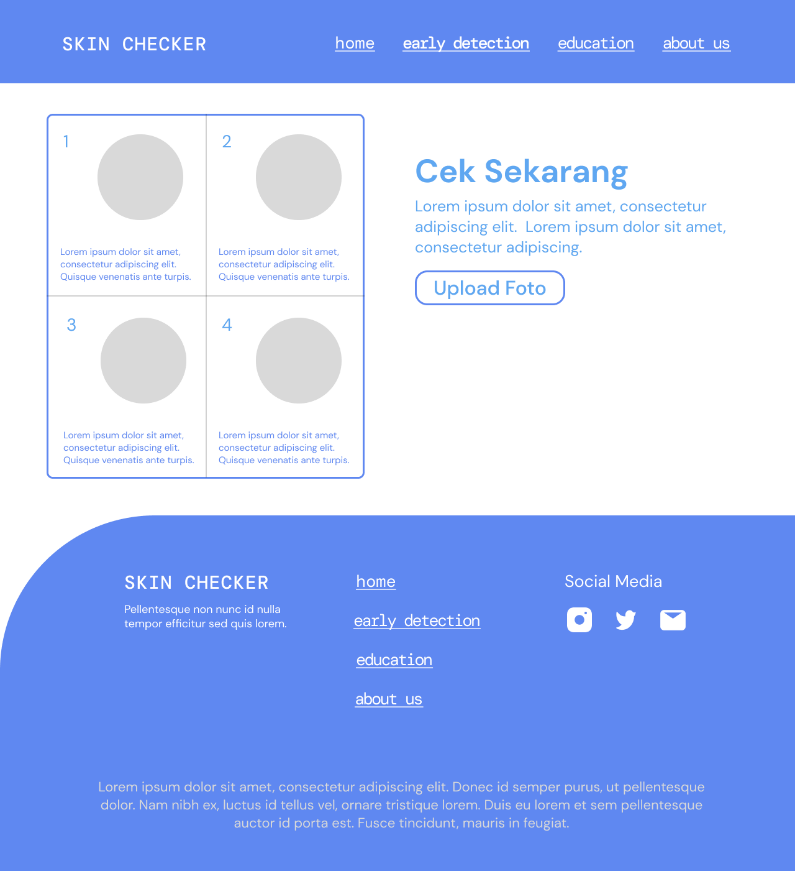
Pada sistem yang dibuat, pengguna hanya dapat mengupload foto atau gambar penyakit. Lalu proses deteksi dilakukan oleh sistem dan hasil akan muncul setelah terdeteksi. Hasil yang ditampilkan setelah proses deteksi adalah jenis penyakit kulit yang diderita dan cara penanganan penyakit tersebut.

### Perancangan User Interface / *Mock-up* aplikasi



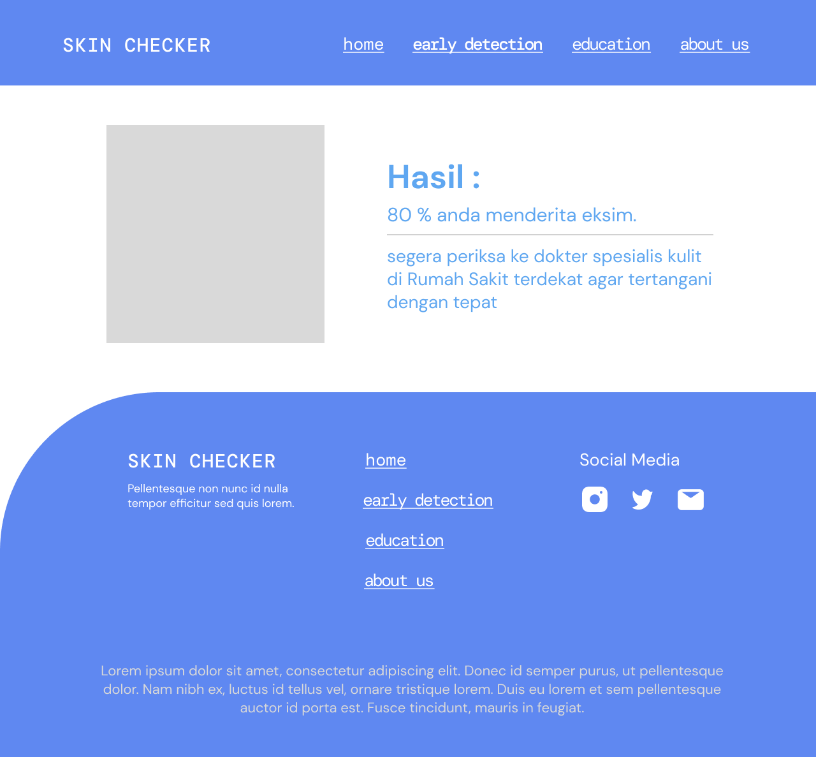
**Gambar 3.13** *Mockup* halaman *home*

Gambar 3.2 merupakan tampilan beranda yang memuat beberapa informasi mengenai website. Terdapat menu untuk mengakses halaman deteksi, halaman informasi dan halaman tentang kami. Juga menampilkan beberapa informasi sekilas mengenai penyakit dan cara penanganan.



**Gambar 3.14** *Mockup* halaman deteksi

Gambar 3.3 merupakan halaman yang digunakan untuk deteksi penyakit. Pada halaman ini menampilkan tata cara unggah foto untuk dideteksi dan button upload untuk mengunggah foto yang akan dideteksi.



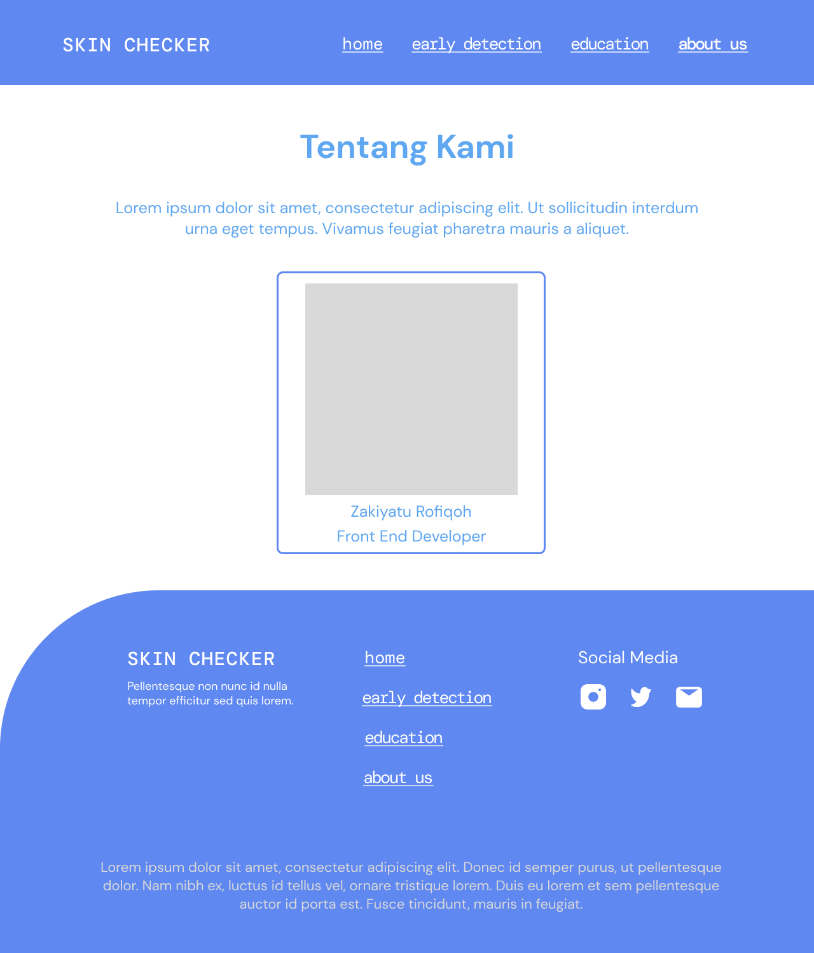
**Gambar 3.15** *Mockup* halaman hasil deteksi

Gambar 3.4 merupakan halaman hasil foto yang telah dideteksi. Halaman ini menampilkan hasil presentase deteksi dan anjuran untuk berobat ke dokter spesialis atau ahli.



**Gambar 3.16** *Mockup* halaman informasi

Gambar 3.5 merupakan halaman informasi yang memuat informasi mengenai penyakit tentang ciri-ciri, cara penanganan, cara pencegahan, berserta foto mengenai penyakit.



**Gambar 3.17** *Mockup* halaman *about*

Gambar 3.6 merupakan halaman about us yang memuat identitas pembuat website dan informasi tentang website.

## 3.3 Rancangan Pengujian

Rancangan pengujian sistem dilakukan untuk mengetahui apakah sistem yang telah dibuat dapat berjalan dengan baik dan sesuai dengan fungsinya atau tidak. Dalam penelitian ini digunakan metode *blackbox testing* dan *confusion matrix* untuk menguji data-datanya.

### 3.3.1 Pengujian *Black Box*

Pengujian *black box* merupakan metode perancangan sistem berdasarkan spesifikasi perangkat lunak. Fokus pengujian adalah pada *output* yang dihasilkan ketika *input* yang dipilih dan kondisi eksekusi terpenuhi. Pengujian ini digunakan untuk menguji kelayakan pada sistem yang telah dibangun.

### 3.3.2 Pengujian Jumlah data dan Jumlah *Epoch*

Rancangan pengujian pada proses *training* dilakukan pengujian jumlah *epoch* dan jumlah data. Pada proses ini dilakukan pembatasan perbandingan dengan jumlah *epoch* 10 dan kelipatan hingga *epoch* 30. Lalu perbandingan jumlah data per penyakit 200, 300, 500, 800 yang dipadukan secara acak.

**Tabel 3.4** Parameter Nilai *Training*

|  |  |  |
| --- | --- | --- |
| **No.** | **Parameter** | **Nilai Parameter** |
| 1 | *Epoch* | 10, 20, 30 |
| 2 | Jumlah data per penyakit | 200, 300, 500, 800 |

### 3.3.3 Pengujian Arsitektur Model

Pengujian model merupakan lanjutan dari pengujian *epoch* dan jumlah data. Setelah didapatkan jumlah *epoch* dan jumlah data yang optimal maka dilakukan pengujian model dari arsitektur CNN. Pada penelitian ini pengujian model dari arsitektur CNN menggunakan 3 model yaitu InceptionV3, ResNet-125 V2, dan Xception. Dengan dilakukannya pengujian ini bertujuan untuk menambahkan hasil akurasi yang optimal.

### 3.3.4 Pengujian *Confusion Matrix*

*Confusion matrix* merupakan teknik yang digunakan untuk mengukur tingkat kebenaran atau keberhasilan dari model yang digunakan untuk deteksi dini penyakit kulit. *Confusion matrix* adalah tabel yang terdiri dari jumlah baris data uji yang diprediksi oleh model klasifikasi benar atau salah. Tabel ini diperlukan untuk mengukur kinerja model klasifikasi (Liliana Swastina, 2013). Pada pengujian *confusion matrix* memiliki empat kombinasi nilai prediksi dan nilai aktual yang berbeda, seperti berikut :

**Tabel 3.5** Perhitungan *Confusion Matrix*

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Aktual** | |
| **Terdeteksi Penyakit Kulit** | **Tidak Terdeteksi Penyakit Kulit** |
| **Prediksi** | **Terdeteksi Penyakit Kulit** | TP | FP |
| **Tidak Terdeteksi Penyakit Kulit** | FN | TN |

Keterangan :

1. TP : *True Positive*

Merupakan data positif yang terprediksi benar. Pada penelitian ini, gambar atau foto penyakit kulit dan dari model yang dibuat gambar atau foto tersebut terprediksi penyakit kulit.

1. FP : *False Positive*

Merupakan data negatif yang terprediksi sebagai data positif. Pada penelitian ini, gambar atau foto bukan penyakit kulit tetapi dari model yang dibuat gambar atau foto tersebut terprediksi penyakit kulit.

1. FN : *False Negative*

Merupakan data positif yang terprediksi sebagai data negatif. Pada penelitian ini, gambar atau foto penyakit kulit tetapi dari model yang dibuat gambar atau foto tersebut tidak terprediksi penyakit kulit.

1. TN : *True Negative*

Merupakan data negatif yang terprediksi benar. Pada penelitian ini, gambar atau foto bukan penyakit kulit dan dari model yang dibuat gambar atau foto tersebut tidak terprediksi penyakit kulit.

Setelah mendapat hasil dari *confusion matrix* maka dapat dihitung akurasinya. Menurut (Saifullah, 2019) akurasi menggambarkan seberapa akurat sistem mengklasifikasikan data dengan benar. Dengan kata lain, nilai akurasi merupakan perbandingan antara data yang terprediksi dengan benar dengan seluruh data. Nilai akurasi dapat diperoleh dengan persamaan berikut

|  |  |  |
| --- | --- | --- |
|  |  | (3.3) |

Lalu menurut (Poerbaningtyas dkk., 2021) untuk membuktikan keberhasilan model dalam memprediksi kebenaran positif dibandingkan keseluruhan data positif dapat diperhitungkan dengan uji sensitifitas.

|  |  |  |
| --- | --- | --- |
|  |  | (3.4) |

Dan untuk membuktikan keberhasilan model dalam memprediksi kebenaran negatif dibandingkan keseluruhan data negatif dapat diperhitungkan dengan uji spesifisitas.

|  |  |  |
| --- | --- | --- |
|  |  | (3.5) |