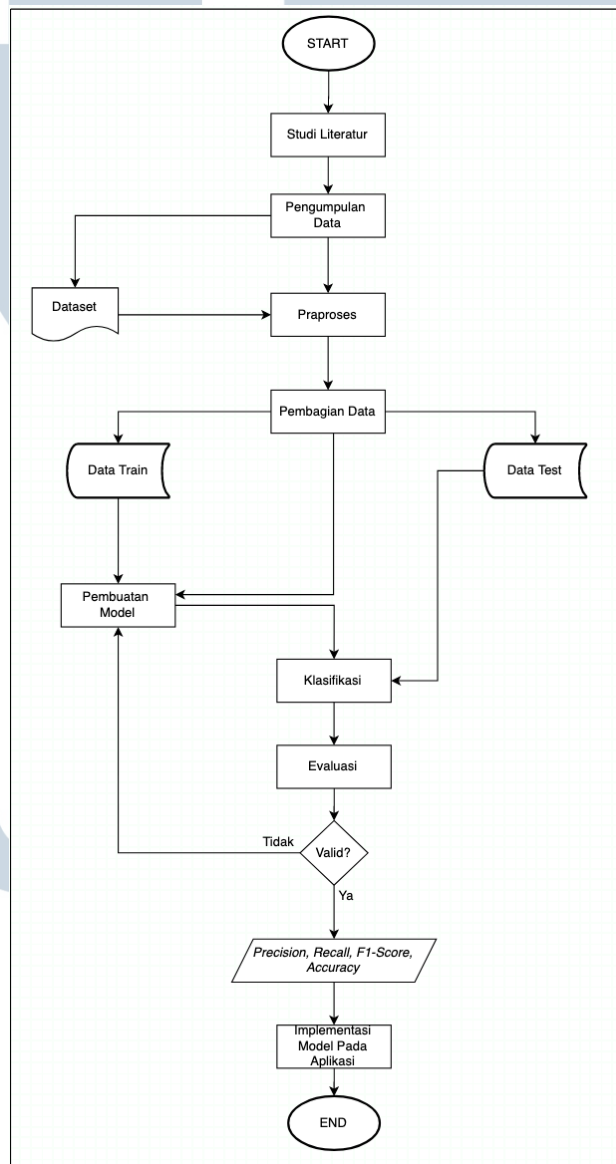


### BAB 3 METODOLOGI PENELITIAN

Berikut merupakan beberapa tahapan yang dilakukan, yaitu, Studi Literatur, Pengumpulan Data, Pembagian Data, Pembuatan Model, Evaluasi Model, dan Implementasi Model Pada Aplikasi. Gambar 3.1 merupakan Flowchart dari metodologi penelitian.



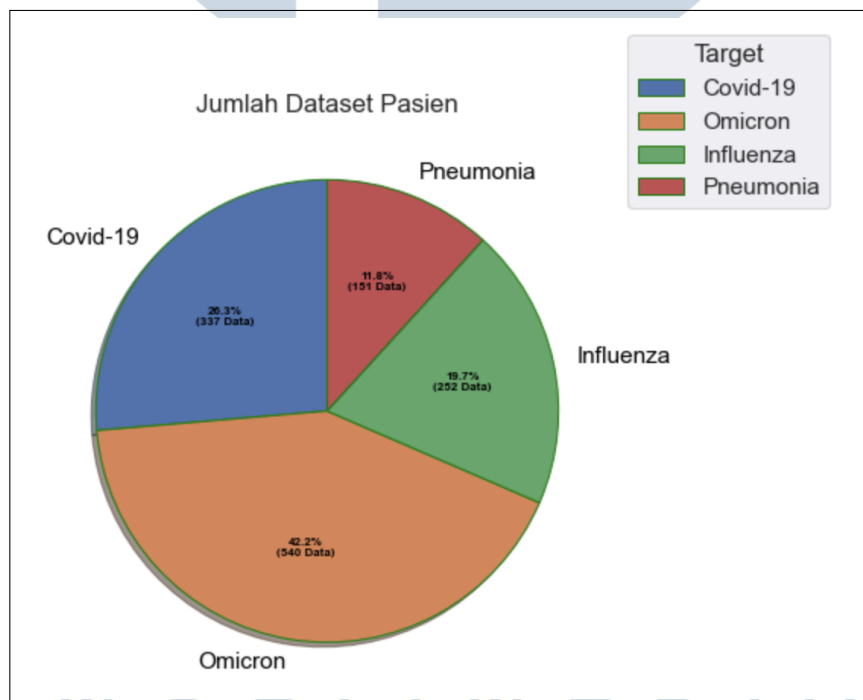
Gambar 3.1. Tahapan Penelitian

### 3.1 Studi Literatur

Studi literatur dilakukan dengan cara membaca, mencari, serta mempelajari ilmu yang berkaitan dengan topik permasalahan, yang didapatkan melalui buku, karya tulis ilmiah, ataupun jurnal. Studi Literatur dilakukan agar supaya mendapatkan referensi serta teori-teori pendukung yang berkaitan dengan topik permasalahan yang dapat mendukung penelitian ini.

### 3.2 Pengumpulan Data

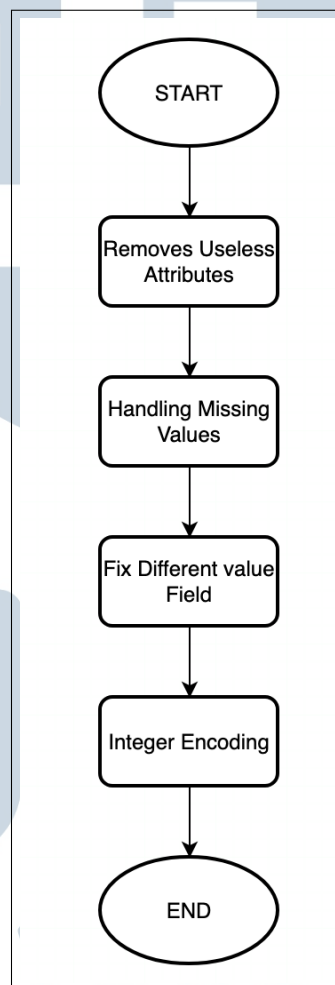
Data yang digunakan dalam penelitian ini didapatkan dari RSUP (Rumah Sakit Umum Pusat) Prof. Dr. R. D. Kandou Manado dan RS Siloam Manado. Dataset diinput oleh Dokter yang bertugas di RS tersebut, pengisian dataset dilakukan dengan 2 cara yaitu observasi, dan juga interview. Dataset yang terkumpul adalah sebanyak 1284 data, dan diisi dalam *file* Excel yang telah dibuat. Gambar 3.2 merupakan representasi dataset yang digunakan.



Gambar 3.2. Persentase Dataset

### 3.3 Praproses

Praproses merupakan tahapan yang dilakukan guna menormalisasi data dari dataset yang telah terkumpul. Tahapan ini dilakukan agar supaya data menjadi lebih mudah di olah, dan lebih sesuai dengan yang dibutuhkan dalam pembuatan model dengan tujuan untuk membuat meningkatkan efektifitas dan juga mendapatkan nilai yang lebih akurat. Gambar 3.3 merupakan *Flowchart* praproses data yang dilakukan pada penelitian ini.



Gambar 3.3. Praproses Data

- *Remove Useless Attributes*

*Removes Useless Attributes* adalah langkah yang dilakukan untuk menghapus atribut-atribut yang tidak diperlukan dalam pembuatan model. Dalam dataset terdapat 1 atribut yaitu Inisial, yang tidak diperlukan dalam pembuatan

model, sehingga data tersebut dihilangkan. Setelah melakukan penghapusan atribut, terdapat sisa 8 atribut yang digunakan dalam pembuatan model.

- *Handling Missing Values*

*Handling Missing Values* adalah langkah yang dilakukan untuk mencegah terdapatnya data yang tidak memiliki *value* atau nilai, dari atribut-atribut yang digunakan dalam pembuatan model. Sebab data yang tidak memiliki nilai tersebut dapat mempengaruhi keakuratan model.

- *Fix Different Value Field*

*Fix Different Value Field* adalah langkah yang dilakukan untuk menyamakan nilai data pada suatu fitur dan target, dimana perbedaan nilai data terjadi sebab adanya kesalahan pengetikan atau kesalahan *input* data. Langkah untuk menyamakan *value* atau nilai pada setiap fitur dan target, sangatlah penting sebelum melakukan proses *Integer Encoding*, juga *Fix Different Value Field* mempengaruhi keakuratan model RF yang dibuat.

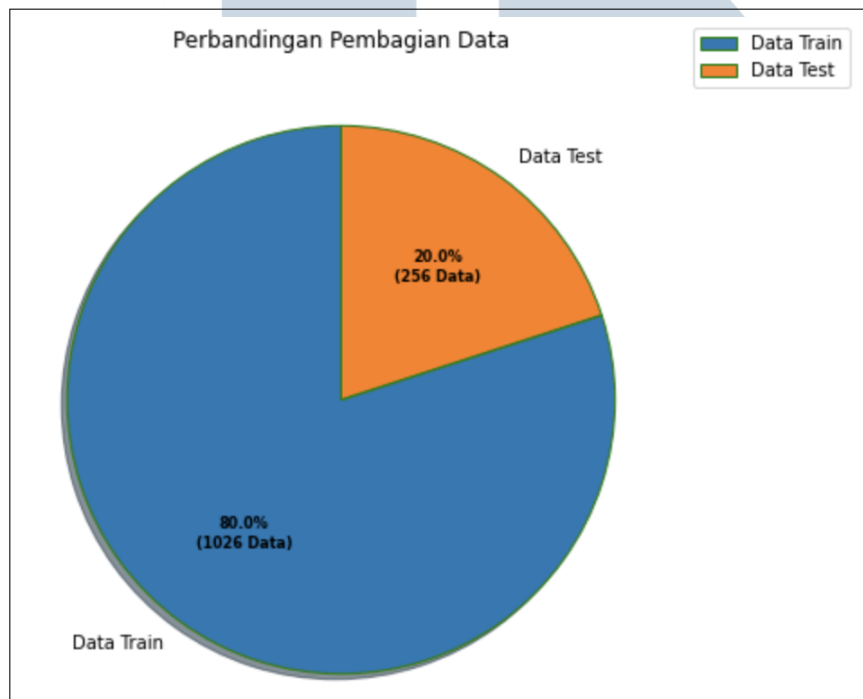
- *Integer Encoding*

*Integer Encoding* adalah langkah yang dilakukan untuk merubah nilai suatu data menjadi *Numerical*. Seluruh atribut dan target dalam dataset yang digunakan pada penelitian ini merupakan *Categorical* data, dimana *value* atau nilai dari suatu data berupa label. Hal tersebut dilakukan sebab terdapat algoritma ML yang tidak bisa memproses *Categorical* data [27]. Terdapat 2 metode yang dilakukan untuk melakukan *Integer Encoding*, yaitu, metode pertama adalah Hot Encoding serta metode kedua adalah label encoder dari *sklearn.preprocessing*. Dalam penelitian ini digunakan metode kedua yaitu Label Encoder, sebab, dalam menggunakan metode Hot Encoding terjadi perubahan *shape* pada dataset.

### 3.4 Pembagian data

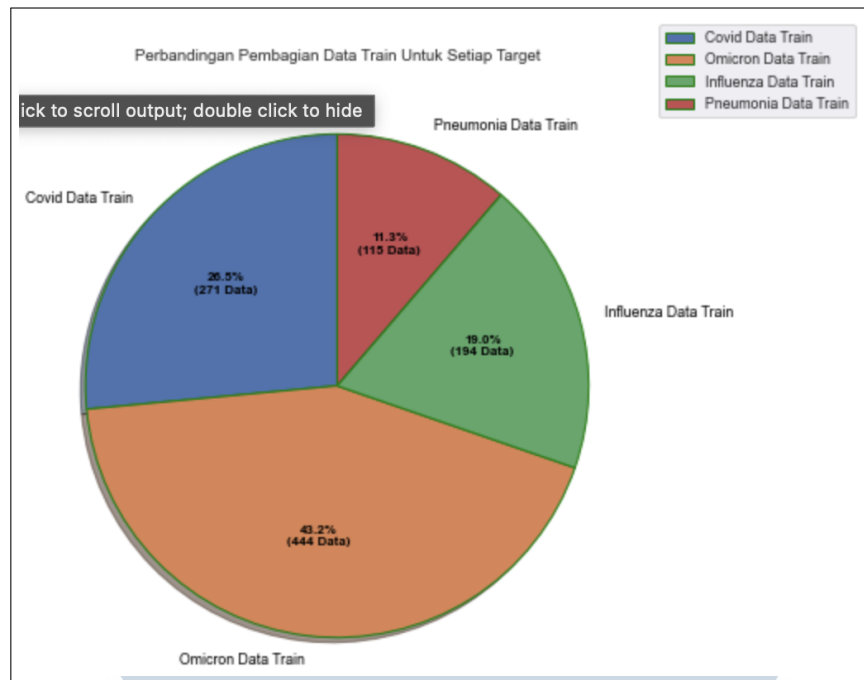
Untuk melakukan prediksi dengan *Machine Learning*, pembagian data dibutuhkan, dengan tujuan untuk mendapatkan data latih atau *data train*, dan data uji coba atau *data test*. *Data train* dibutuhkan untuk melatih model dengan menggunakan parameter yang ditetapkan, sedangkan *data test* dibutuhkan untuk untuk pengujian model serta evaluasi model. Adapun rasio perbandingan pembagian data yaitu 80:20 yaitu 80% untuk data train, dan 20% untuk data test

seperti pada Gambar 4.31. Sedangkan untuk pembagian target yang terdapat pada data train yang berjumlah 1206 Data adalah seperti pada Gambar 3.5. Dari Gambar 3.5 untuk Covid terdapat 271 data (26.5%), Omicron terdapat 444 data (43.2%), Influenza terdapat 194 data (29.0%), dan Pneumonia terdapat 115 data (11.3%).



Gambar 3.4. Rasio Perbandingan Data Latih dan Data Uji





Gambar 3.5. Perbandingan Target Pada Data Latih

Untuk melakukan pembagian data *function* yang digunakan adalah `train_test_split` dari `sklearn.model_selection`. *function* tersebut akan membagi *data train* dan *data test* sesuai dengan parameter yang ditetapkan. Adapun parameter yang ditetapkan dalam *function* tersebut terdapat dalam implementasi.

### 3.5 Pembuatan Model

Model adalah fungsi dari pembelajaran yang kemudian diimplementasikan pada aplikasi yang dibuat. Model dibuat berdasarkan *data train* yang telah dibagi pada proses Pembagian Data, model yang dibuat adalah dengan menggunakan algoritma *random forest classifier*. Adapun terdapat parameter-parameter yang ditetapkan dalam pembuatan model dengan *random forest classifier*, yaitu:

#### 1. N\_Estimators

N\_estimator merupakan *function* yang digunakan agar dapat menentukan jumlah *tree* yang terdapat didalam model RF, semakin banyak jumlah *tree* yang terdapat didalam model RF, maka semakin lama juga waktu yang dibutuhkan untuk melakukan pelatihan model. Jumlah *tree* yang digunakan didalam model RF akan berpengaruh pada kinerja model RF, untuk mendapatkan kinerja model RF terbaik maka membutuhkan sejumlah besar

*tree*, akan tetapi pada titik tertentu besaran dari *tree* yang mempengaruhi kinerja model akan mencapai batas maksimum [12]. Penetapan parameter *n\_estimators* bertujuan untuk mendapatkan kinerja terbaik dari model RF yang dibuat.

## 2. Criterion

Criterion merupakan *function* yang berfungsi untuk mengukur kualitas dari split. Pengukuran kualitas dilakukan dengan cara mengukur nilai *Gini impurity* atau dengan mengukur nilai *information gain*. Untuk mengukur nilai Gini Impurity maka parameter diinput dengan “gini” sedangkan untuk mendapatkan nilai Information Gain diukur parameter diinput dengan “entropy”.

## 3. Max.Features

Max.Features adalah *function* yang digunakan untuk menentukan jumlah dari fitur/atribut yang digunakan dalam pembuatan model RF. Parameter *max\_feature* ditetapkan menjadi 8 berdasarkan atribut yang ada pada dataset yang telah melalui tahap *preprocessing*.

## 4. Max.Depth

Max.Depth adalah *function* yang digunakan untuk menentukan kedalaman maksimum dari *tree*. Nilai *max\_depth* yang digunakan bertujuan untuk mengurangi variasi yang terbentuk dari fitur yang digunakan dalam pembuatan model RF, semakin tinggi tingkat kedalam sebuah *tree*, maka semakin kompleks pohon yang dihasilkan [12].

### 3.6 Evaluasi Model

Evaluasi model merupakan tahap untuk mengukur seberapa baik kinerja model RF yang telah dibuat, serta untuk mendapatkan nilai terbaik dari kinerja model RF yang telah dibuat, yang nantinya dari kinerja model terbaik kemudian diimplementasikan pada aplikasi. Adapun nilai-nilai yang diukur untuk mengukur seberapa baik kinerja model RF, yaitu *Precision*, *Recall*, *F1-Score*, serta *Accuracy*. Dalam mengevaluasi model RF yang telah dibuat digunakan *function*; *accuracy\_score*, *confusion\_matrix*, *classification\_report* dari *sklearn.metrics*. *Function classification\_report* merupakan *function* untuk mendapatkan nilai *Precision*, *Recall*, *F1-Score* seperti pada Gambar 3.6,



	precision	recall	f1-score	support
0	0.87	0.81	0.84	59
1	0.89	0.81	0.85	124
2	0.71	0.92	0.80	51
3	0.81	0.93	0.86	27
micro avg	0.83	0.85	0.84	261
macro avg	0.82	0.87	0.84	261
weighted avg	0.84	0.85	0.84	261
samples avg	0.84	0.84	0.84	261

Gambar 3.6. Contoh *Classification Report*

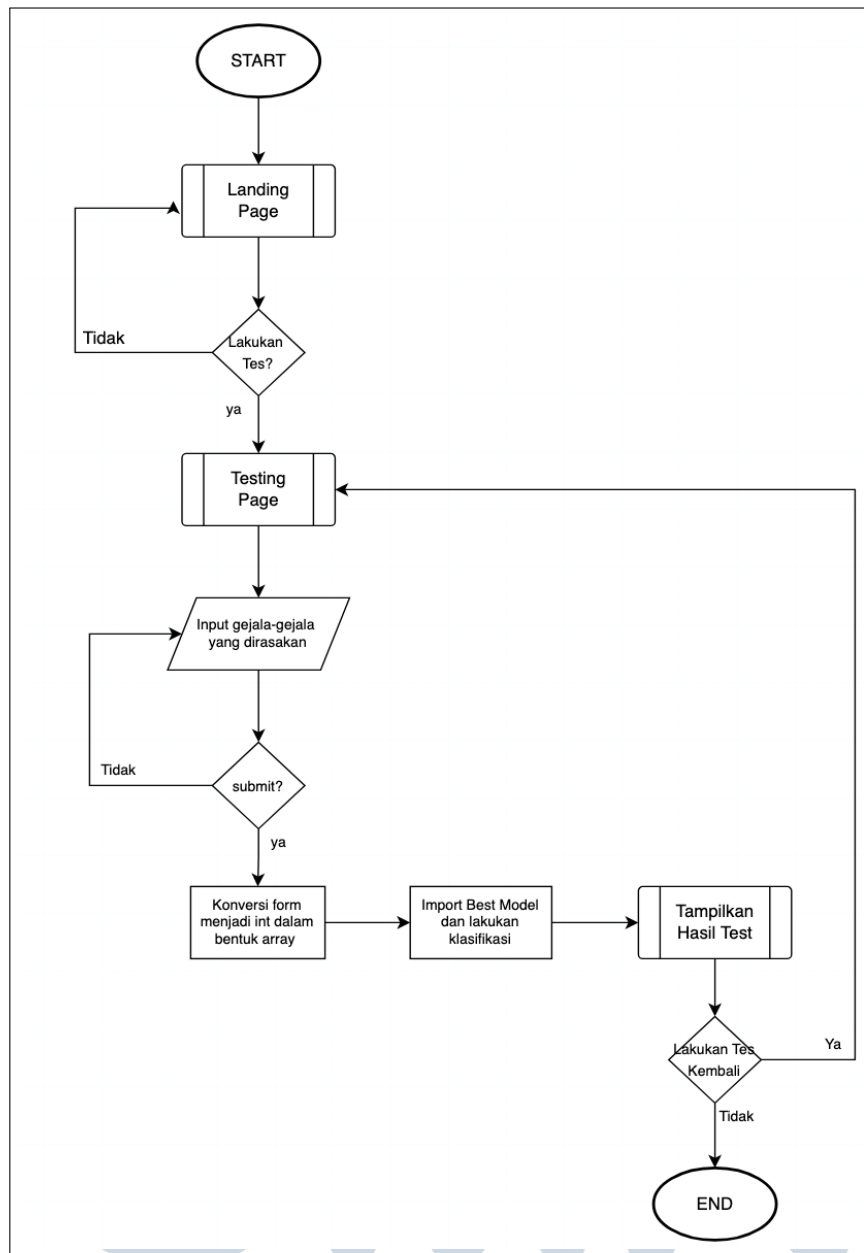
### 3.7 Perancangan Aplikasi

Setelah membuat dan mendapatkan model dengan nilai *Precision*, *Recall*, *F1-Score*, dan *Accuracy* yang maksimal, maka selanjutnya adalah mengimplementasikan model yang telah dibangun pada aplikasi yang dibuat. Tujuan pembuatan aplikasi ini agar supaya dapat melakukan klasifikasi penyakit (Covid-19, Omicron, Influenza, Pneumonia) yang diderita, berdasarkan gejala-gejala yang dirasakan.

#### 3.7.1 Flowchart Aplikasi

Gambar 3.7 merupakan flowchart dari aplikasi yang dibuat. Dari Gambar 3.7, saat pengguna masuk ke dalam Aplikasi maka pengguna di *direct* ke Halaman Awal atau *Landing Page*, setelah itu pengguna bisa memilih untuk melakukan tes atau tidak. Jika pengguna memilih melakukan tes, maka pengguna di *direct* ke Halaman Tes atau *Testing Page*, *testing page* berisi gejala-gejala yang dirasakan yang berdasarkan 8 fitur pembuatan model yaitu: "Batuk", "Pilek", "Sakit Tenggorokan", "Sesak Nafas", "Demam", "Sakit Kepala", "Lemas" dan "Anosmia". Kemudian pengguna diharuskan memasukan gejala-gejala yang dirasakan. Selanjutnya data yang telah dimasukan oleh pengguna, kemudian di konversi menjadi integer (0, 1) kemudian diubah bentuknya menjadi 2D (Dimensional) array. Setelah itu, dilakukan import model terbaik yang telah disimpan dalam *pickle*, kemudian masukan hasil konversi data untuk dilakukan klasifikasi. Setelah mendapatkan hasil tes, pengguna dapat memilih untuk melakukan hasil tes lagi, atau tidak. Jika pengguna memilih untuk melakukan tes lagi, maka pengguna di *redirect* kembali ke halaman *Testing Page* dan mengulangi kembali dari proses yang sama.



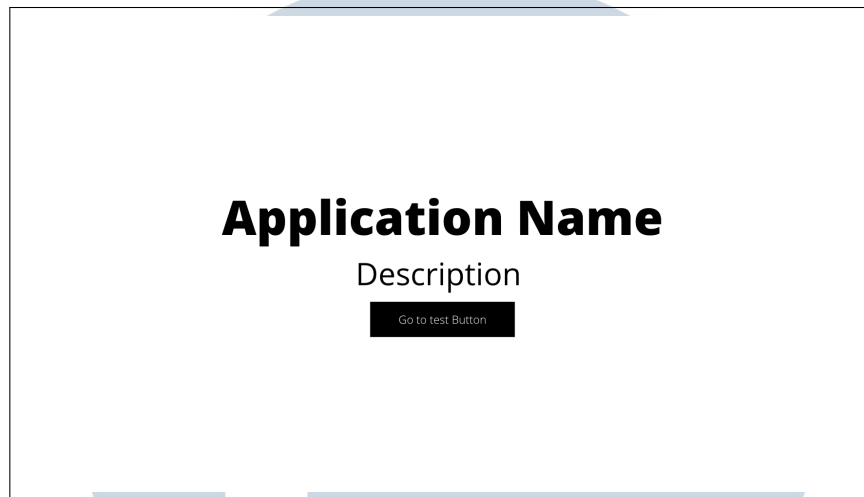


Gambar 3.7. Flowchart Aplikasi

### 3.7.2 Mockup Aplikasi

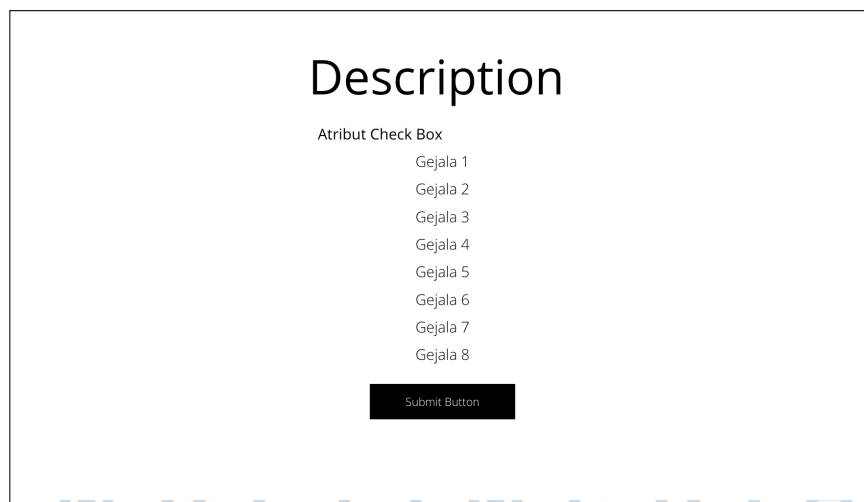
Mockup aplikasi yang dibuat adalah untuk 3 halaman, yaitu halaman *landing page* atau *home page*, yaitu halaman yang dilihat oleh pengguna pertama kali. Kemudian halaman *testing*, yaitu halaman untuk pengguna mengisi tes berupa inputan gejala-gejala yang dirasakan, dan halaman terakhir, halaman untuk menampilkan hasil tes yang didapat oleh pengguna berdasarkan gejala-gejala yang

diinput oleh pengguna. Gambar 3.8 merupakan *mockup* antarmuka untuk halaman *home*, yaitu halaman ketika pengguna pertama kali masuk ke website.



Gambar 3.8. *Mockup* Halaman *Home*

Dari Gambar 3.8, terdapat *Text* untuk menampilkan nama aplikasi dan kegunaan aplikasi, serta terdapat *Button* untuk pindah ke halaman *test*. Gambar 3.9 merupakan *mockup* antarmuka untuk halaman *test*, yaitu halaman ketika pengguna melakukan tes yaitu dengan pengisian gejala-gejala yang dirasakan oleh pengguna.



Gambar 3.9. *Mockup* Halaman *Test*

Dari Gambar 3.9, terdapat *Text* untuk mendeskripsikan pengisian tes, kemudian terdapat *Checkbox* untuk memilih gejala-gejala yang dirasakan oleh pengguna, serta terdapat *Button* untuk pindah ke halaman *result*. Gambar

3.10 merupakan *mockup* antarmuka untuk halaman *result*, yaitu halaman ketika pengguna selesai mengisi tes dan melihat diagnosa penyakit berdasarkan gejala-gejala yang diinput.



Gambar 3.10. *Mockup* Halaman *Result*

Dari Gambar 3.10, terdapat *Text* untuk mendeskripsikan pengisian tes dan hasil test yang didapat berdasarkan gejala-gejala yang diinput *result*, serta terdapat dua *Button*, *Button* yang pertama adalah *Button* untuk pindah ke halaman *home* dan *Button* yang kedua untuk pindah ke halaman *test* atau melakukan tes lagi.

