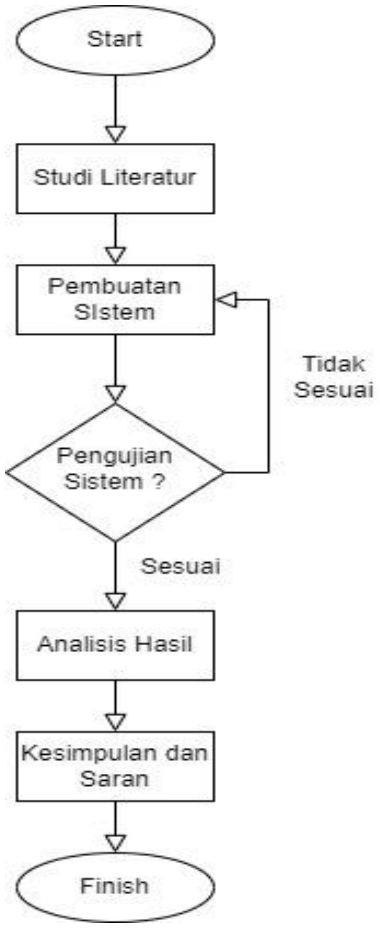
BAB 3
METODE PENELITIAN

3.1 ALUR PENELITIAN

Pada tahap ini akan dijelaskan mengenai alur penelitian pada penelitian ini. Adapan diagram perancangan sistem bisa dilihat pada Gambar 3.1.



Gambar 3.1 *Flowchart* alur penelitian

Pada alur penelitian ini memiliki beberapa tahap. Pada tahap awal, dilakukan proses pengumpulan materi dan referensi sebagai acuan serta membandingkan kajian teori yang telah ada dalam penelitian sebelumnya. Pengumpulan materi dapat dilakukan melalui sumber-sumber seperti jurnal, buku, *e-book*, dan *website*. Pada tahap perancangan dan pembuatan sistem, mulai menyusun rancangan sistem yang terdiri dari *library OpenCV* dan bahasa pemrograman *Python* dengan menggunakan *software* IDE *Python*. Setelah

menyelesaikan perancangan dan pembuatan sistem, dilakukan pengujian sistem. Pengujian sistem meliputi keakuratan dari *face recognition* dan *digital image processing* dalam mendeteksi kantuk. Perancangan perangkat dan pengujian sistem telah dilaksanakan, data yang diperoleh akan di analisis untuk dapat diambil kesimpulan dan saran dari pengujian sistem, serta mengetahui nilai kesalahan yang terjadi saat pengujian sistem.

3.2 ALAT DAN BAHAN

Pada perancangan sistem ini meliputi perangkat keras dan perangkat lunak yang digunakan untuk mendapatkan data dan akan dianalisis. Berikut alat dan bahan yang digunakan dalam penelitian ini.

3.2.1 Perangkat Keras

Perangkat keras yang digunakan pada penelitian ini yaitu:

1) Laptop digunakan untuk pembuatan sistem pendeteksi kantuk.
2) *Webcam* digunakan untuk proses pengambilan data *image* dan proses pendeteksi kantuk.

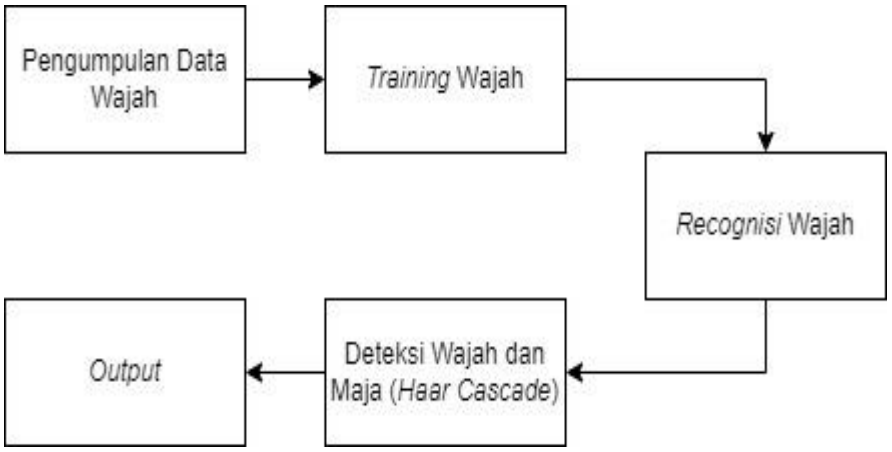
3.2.2 Perangkat Lunak

Perangkat lunak yang digunakan pada penelitian ini yaitu:

1) *Python* versi 3.10.5 merupakan bahasa pemrograman *Python* dengan versi terbaru yaitu 3.10.5.
2) *OpenCV Library* adalah sebuah aplikasi bersifat *open source* yang menyediakan perpustakaan (*library*) untuk pengembangan pemrosesan citra digital.
3) *Pycharm Community* 2022 merupakan *software* IDE bahasa pemrograman khususnya bahasa pemrograman *Python*. Digunakan untuk membuat program dengan dilengkapi berbagai fitur yang mempermudah proses pembuatan sistem.
4) *Command Prompt* adalah sebuah *command line* (baris perintah) berbasis GUI untuk mengesekusi *file*.

3.3 PERANCANGAN SISTEM

Pada tahap ini akan dijelaskan mengenai perancangan sistem pada penelitian ini. Adapan diagram perancangan sistem bisa dilihat pada Gambar 3.2.



Gambar 3.2 Diagram perancangan sistem

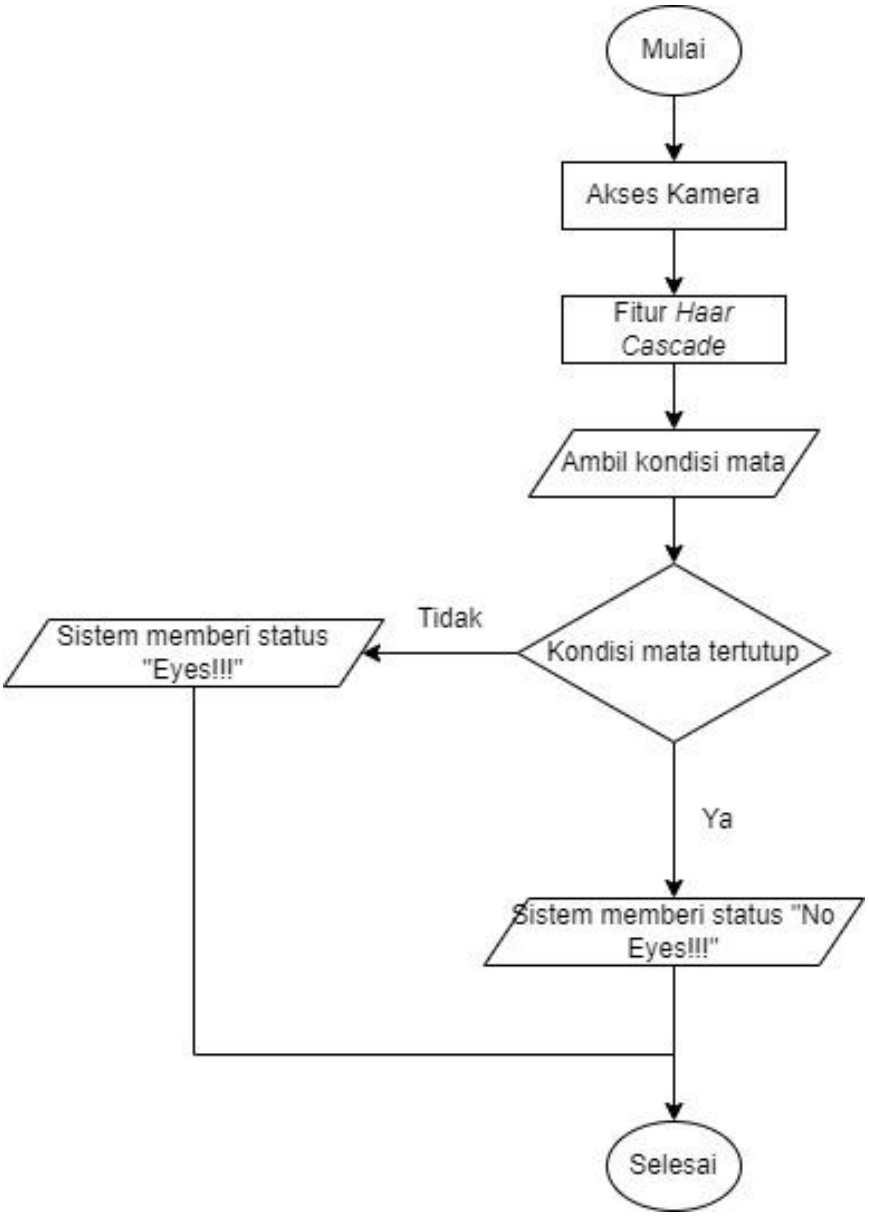
Pada Gambar 3.2 menunjukkan diagram keseluruhan dari rancangan sistem deteksi kantuk. Diagram ini menjelaskan tentang alur sistem deteksi kantuk. Pada blok pengumpulan data wajah dilakukan dengan memberi masukan (*input*) pada sistem. *Input* berupa citra (video) bagian wajah. Proses pengambilan citra dilakukan secara langsung atau *real time*, dimana sistem akan terus berjalan unutk memproses gambar. Untuk mendapatkan hasil yang optimal, jarak antara wajah dengan kamera atau *webcam* sekitar 30 hingga 40 cm. Data wajah yang akan diambil sebanyak 1.600 gambar dengan beberapa parameter. Pada blok *training* wajah, proses yang dilakukan pada blok ini yaitu membuat file *training.yml* yang akan dipanggil oleh program *wajah.py*. Pada blok rekognisi wajah dilakukan proses pendeteksian wajah. Wajah yang sesuai kriteria akan terdeteksi oleh kamera. Setelah proses pendeteksian wajah berhasil, pada blok *output* akan menampilkan status wajah tersebut mengantuk atau tidak mengantuk.

3.3.1 *Flowchart* Sistem

Pada tahap ini, data input yang merupakan citra wajah akan melalui beberapa proses pengklasifikasian untuk mendapatkan informasi mengenai kelas citra tersebut. Berikut beberapa *flowchart* yang digunakan dalam perancangan sistem deteksi kantuk.

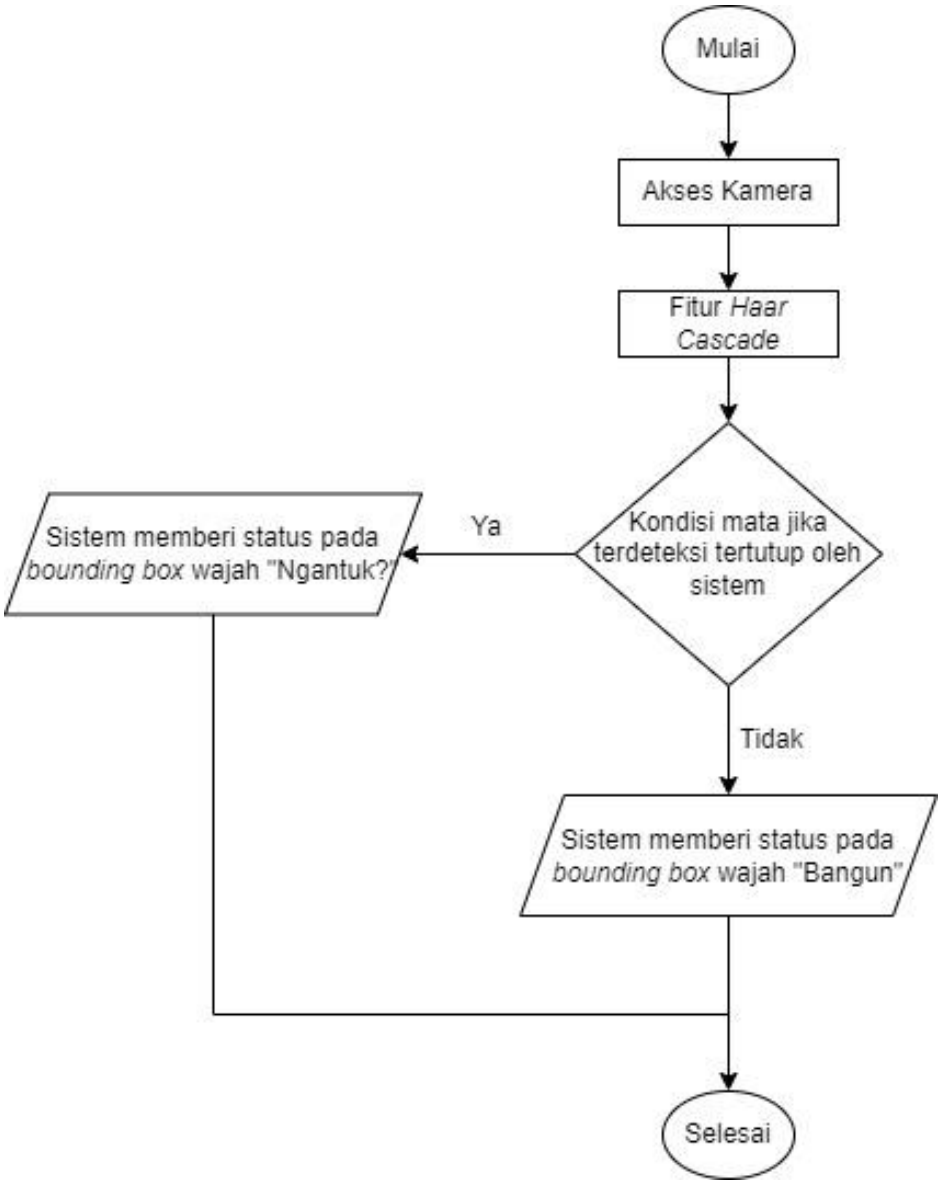
Gambar 3.3 *Flowchart* pengumpulan dataset wajah dan fitur *Haar Cascade*

Pada Gambar 3.3 yaitu *flowchart* pengumpulan dataset wajah dan fitur *Haar Cascade*. Pada *flowchart* ini, data *image* yang sesuai kriteria akan diberikan fitur persegi (*Haar Cascade*). Proses awal yaitu sistem akan mengakses *webcam*, setelah berhasil *webcam* akan mulai mengambil data wajah dengan sesuai kriteria. Data wajah yang digunakan sebanyak 1.600 gambar. Setelah data wajah berhasil didapat, proses selanjutnya yaitu proses *training* wajah, seperti yang dijelaskan sebelumnya. setelah data wajah di-*training*, *webcam* akan merekognisi wajah, apakah objek yang berada di depan *webcam* tersebut merupakan wajah atau bukan. Setelah webcam mengetahui objek tersebut merupakan wajah, maka fitur persegi (*Haar Cascade*) akan muncul untuk mengindikasikan objek tersebut adalah sebuah wajah.



Gambar 3.4 *Flowchart* pengujian kondisi mata

Pada Gambar 3.4 yaitu *flowchart* pengujian kondisi mata. Pada *flowchart* ini, sistem akan mengambil kondisi mata seseorang dengan indikator mata terbuka dan tertutup. Jika mata tertutup terdeteksi oleh sistem, maka sistem akan memberi *output* "*No Eye*" dan sebaliknya. Pada pengujian ini menggunakan metode *Haar Cascade* dan *Eye Aspect Ratio*. Dalam pengujian kondisi mata, fitur *Haar Cascade* dan *Eye Aspect Ratio* akan fokus mendeteksi *frame* wajah dan mata agar sistem dapat mendeteksi secara optimal Nantinya data mata ini akan digunakan pada program deteksi kantuk.



Gambar 3.5 *Flowchart* deteksi kantuk

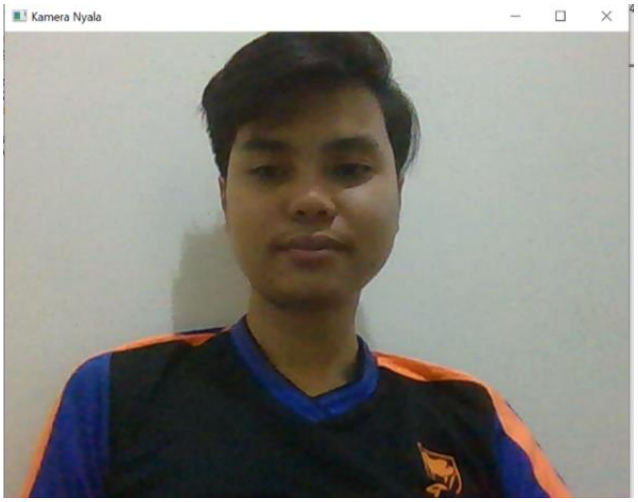
Pada Gambar 3.5 yaitu *flowchart* deteksi kantuk. Pada *flowchart* ini, setelah data wajah berhasil diindikasi oleh *webcam* dengan fitur persegi *Haar Cascade*, selanjutnya sistem akan fokus mendeteksi *Eye Aspect Ratio* (EAR) pada mata. Jika mata tertutup, akan diberikan *output* "Bangun", dan jika mata terbuka akan diberikan *output* "Ngantuk?". Setelah proses pendeteksian berhasil, maka sistem yang dibuat sudah benar. Pada proses pendeteksian kantuk, jarak antara wajah dan kamera sekitar 30 hingga 40 cm agar sistem dapat mendeteksi wajah dan mata dan pastikan kondisi lingkungan saat proses deteksi kantuk memiliki pencahayaan yang cukup agar proses dan hasil deteksi sistem dapat optimal

3.4 KOMPONEN PENGUJIAN

Pada bagian ini, terdapat beberapa komponen pengujian yang digunakan dalam penelitian ini, yaitu :

3.4.1 Pengujian *Webcam*

Proses pengujian *webcam* dilakukan dengan mengakses *webcam* menggunakan bahasa pemrograman *Python*. Sebelum proses pengujian kamera atau *webcam*, pastikan bahasa pemrograman Pyhton sudah ter-*install* pada laptop atau PC serta laptop atau PC terhubung ke *internet* agar *webcam* bisa diakses dan dapat mengambil gambar. Proses pengujian *webcam* sangat penting, demi kelancaran proses sistem pendeteksi kantuk ini. Contoh hasil akses *webcam* dengan *Python* bisa dilihat pada Gambar 3.4.

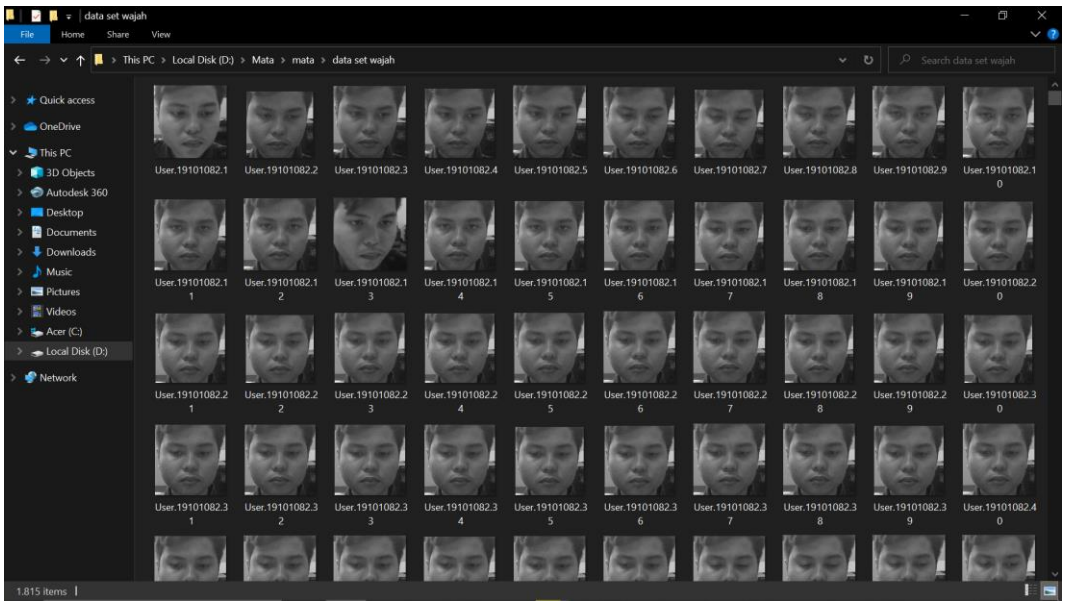


Gambar 3.4 Akses *webcam* dengan *Python*

3.4.2 Pengambilan *Dataset* Wajah

Pada proses ini dilakukan proses memberi masukan (*input*) data wajah pada sistem. Proses pengambilan *dataset* wajah dilakukan secara *real time*, dimana sistem akan terus berjalan untuk memproses gambar. Untuk mendapatkan hasil yang optimal, jarak antara wajah dengan kamera atau webcam sekitar 30 hingga 40 cm. Data dikamera kemudian di-*capture* dan disimpan sehingga menjadi data wajah. Kriteria *dataset* wajah yang digunakan pada penelitian ini yaitu wajah manusia secara *full face*. Jumlah data wajah sebanyak 1.600 gambar wajah dengan kondisi mata sebagai berikut : satu mata tertutup sebanyak 400

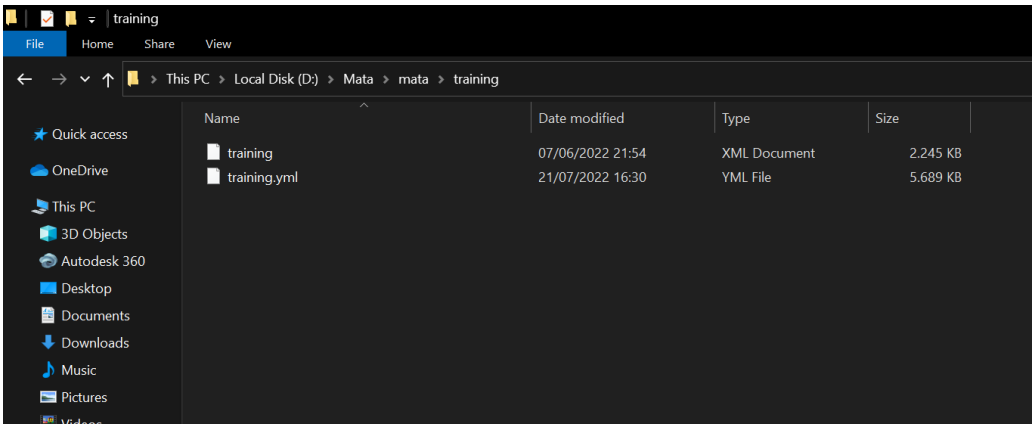
gambar, dua mata terbuka sebanyak 400 gambar, dua mata tertutup sebanyak 400 gambar, dan dua mata setengah tertutup sebanyak 400 gambar. Contoh *dataset* wajah bisa dilihat pada Gambar 3.5.



Gambar 3.5 Contoh *dataset* wajah

3.4.3 Pengujian *Training* Wajah

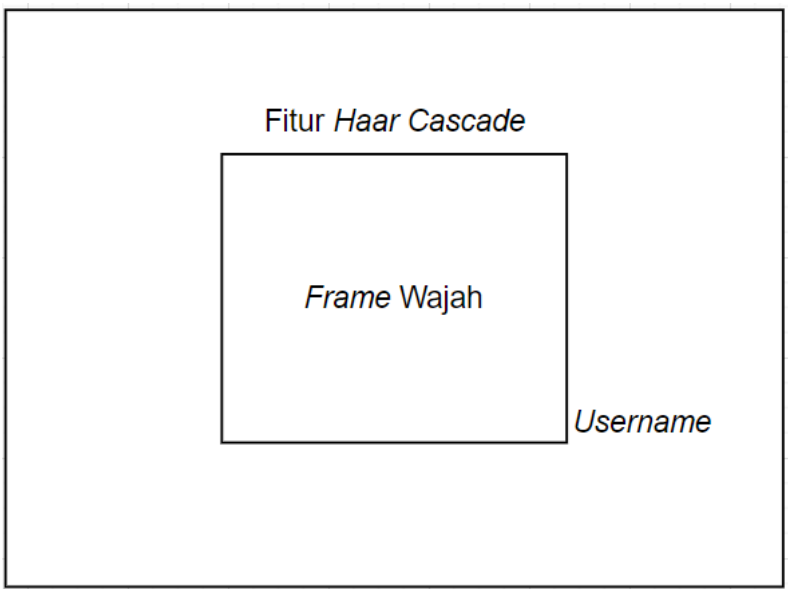
Proses pengujian training wajah bertujuan untuk mengetahui apakah *webcam* dapat mendeteksi wajah dan membuat *file training*.yml yang akan dipanggil oleh program *wajah.py* untuk mendeteksi wajah. Data wajah yang sudah didapat dimasukkan kedalam sistem untuk di-*training.* Data wajah yang digunakan sebanyak 1.600 gambar. Hasil dari *training* ini adalah kemampuan sistem untuk mengenali wajah dan pembuatan *file* training.yml. Nantinya *file* training.yml akan dipanggil oleh sistem dalam tahap pengujian selanjutnya agar hasil proses deteksi mengantuk dapat optimal. Contoh pembuatan *file training*.yml bisa dilihat pada Gambar 3.6.



Gambar 3.6 Contoh *file training.yml*

3.4.4 Pengujian Rekognisi Wajah

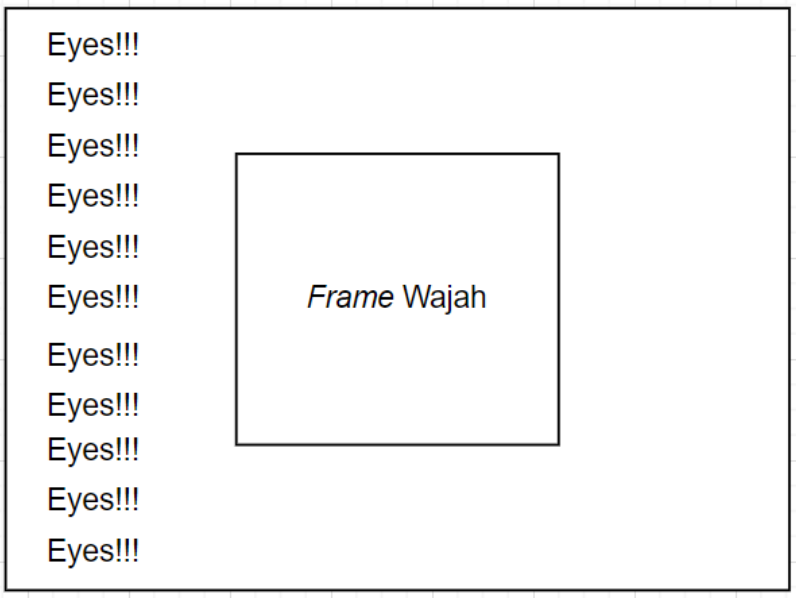
Proses pengujian rekognisi wajah bertujuan untuk mengetahui apakah *webcam* dapat mendeteksi sebuah gambar wajah secara benar. Pengujian ini dilakukan dengan memberikan data wajah sebanyak 1.600 gambar. Pada pengujian ini, data wajah yang sudah dimasukkan akan dideteksi sistem apakah objek yang terdeteksi oleh *webcam* adalah wajah atau bukan wajah. Pada pengujian rekognisi wajah, pastikan wajah mendapatkan cahaya yang cukup agar sistem dapat mendeteksinya. Perancangan pengujian rekognisi wajah bisa dilihat pada Gambar 3.7.



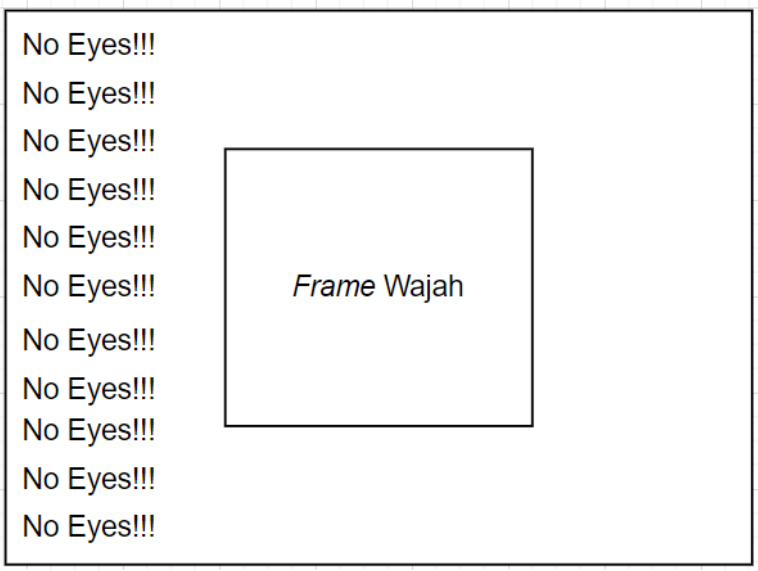
Gambar 3.7 Perancangan pengujian rekognisi wajah

3.4.5 Pengujian Kondisi Mata

Proses pengujian mata terbuka dan tertutup bertujuan sebagai indikator sistem untuk mengetahui apakah seseorang tersebut mengantuk atau tidak. Dalam pengujian ini, *webcam* secara *real time* akan mengambil kondisi mata seseorang. Jika mata terbuka, sistem akan memberikan *output* "*Eye*", dan jika mata tertutup, sistem akan memberikan *output* "*No Eye*". Pada penelititan ini, kondisi mata menjadi indikator seseorang tersebut dalam keadaan mengantuk atau tidak mengantuk. Perancangan pengujian kondisi mata terbuka dan tertutup dapat dilihat pada Gambar 3.8 dan 3.9.



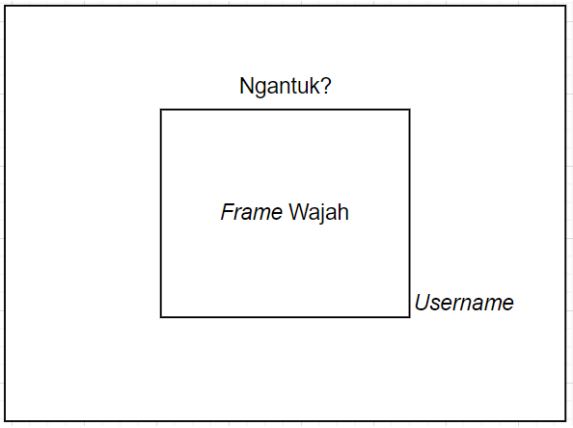
Gambar 3.8 Perancangan pengujian kondisi mata terbuka



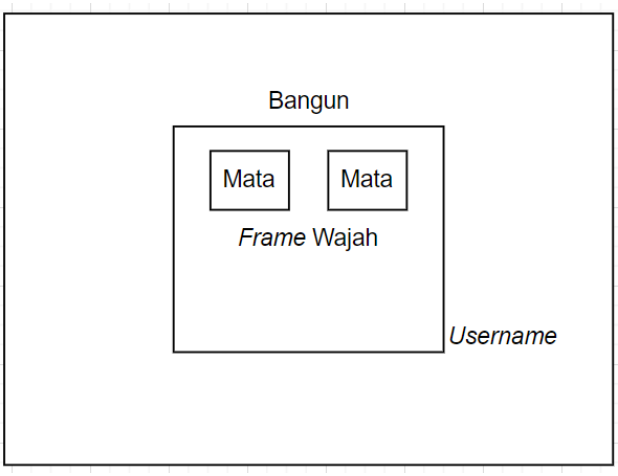
Gambar 3.9 Perancangan pengujian kondisi mata tertutup

3.4.6 Pengujian Deteksi Wajah Mengantuk

Proses deteksi wajah mengantuk bertujuan untuk mengetahui apakah *webcam* atau kamera dapat mendeteksi wajah dalam kondisi mengantuk dan tidak mengantuk. Setelah pengujian rekognisi wajah berhasil, selanjutnya dilakukan pengujian deteksi kantuk. Dalam pengujian ini, *webcam* atau kamera secara *real time* akan mendeteksi apakah wajah mengantuk atau tidak mengantuk dengan melihat kondisi mata terbuka atau tertutup. Jika maka terbuka, *output* yang akan diberikan *webcam* yaitu "Bangun", dan jika mata tertutup, *output* yang akan diberikan *webcam* yaitu "Mengantuk". Pada pengujian ini, agar sistem dapat mendeteksi secara optimal, jarak antara wajah dengan kamera sekitar 30 hingga 40 cm. Perancangan pengujian deteksi wajah mengantuk dan tidak mengantuk bisa dilihat pada Gambar 3.10 dan 3.11.



Gambar 3.10 Perancangan pengujian deteksi wajah mengantuk



Gambar 3.11 Perancangan pengujian deteksi wajah tidak mengantuk/bangun

3.4.7 Pengujian Sistem Keseluruhan

Pengujian sistem secara keseluruhan dimulai dari pengujian *webcam* sampai pengujian deteksi kantuk. Hasil dari pengujian ini adalah pemberitahuan wajah mengantuk dan tidak mengantuk. Pengujian dilakukan sebanyak 200 kali pengujian dari parameter kondisi mata. Keberhasilan dari pengujian ini dimasukkan kedalam *confusion matrix* dan *performance matrix*.

a. *Confusion Matrix*

Confusion matrix adalah metode pengukuran untuk mengevaluasi kinerja dan tingkat keakuratan suatu proses klasifikasi. Berisi tabel yang menyajikan informasi jumlah data yang diklasifikasikan secara benar dan salah.

1. *True Positive* (TP)

   True Positive (TP) merupakan jumlah data benar yang dideteksi benar oleh sistem. Dalam sistem ini, misalnya wajah yang mengantuk dan dideteksi mengantuk oleh sistem.

2. *True Negative* (TN)

   True Negative (TN) merupakan jumlah data salah yang dideteksi salah oleh sistem. Dalam sistem ini, misalnya wajah yang tidak mengantuk dideteksi tidak mengantuk oleh sistem.

3. *False Positive* (FP)

   False Positive (FP) merupakan jumlah data salah yang dideteksi benar oleh sistem. Dalam sistem ini, misalnya wajah tidak mengantuk dideteksi wajah mengantuk oleh sistem.

4. *False Negative* (FN)

   False Negative (FN) merupakan jumlah data benar yang dideteksi salah oleh sistem. Dalam sistem ini, misalnya wajah mengantuk dideteksi wajah tidak mengantuk oleh sistem.

b. *Performance Matrix*

Performance Matrix adalah suatu alat ukur yang digunakan untuk mengukur kinerja dari sistem. Beberapa bagian dari *performance matrix* yaitu:

1. *Accuracy*

   Accuracy menjelaskan seberapa akurat model atau sistem dapat mengklasifikasikan data dengan benar. Perhitungan *accuracy* dapat dilihat

pada persamaan 3.1.

$$Accuracy = \frac{\text{TP}+TN}{\text{TP+TN+FP+FN}}\ x\ 100\% \quad \textbf{(3.1)}$$

2. *Precision*

   Precision menghitung berapa banyak data positif yang diprediksi benar dari semua data yang diprediksi sebagai data positif. Perhitungan *precision* dapat dilihat pada persamaan 3.2.

$$Precision = \frac{\text{TP}}{\text{TP+FP}}\ x\ 100\% \quad \textbf{(3.2)}$$

3. *Recall*

   Recall menggambarkan sejauh mana keberhasilan model atau sistem dalam mengidentifikasikan kembali data positif. Perhitungan *recall* dapat dilihat pada persamaan 3.3.

$$Recall = \frac{\text{TP}}{\text{TP+FN}}\ x\ 100\% \quad \textbf{(3.3)}$$