**BAB V**

**IMPLEMENTASI DAN PENGUJIAN**

**5.1 Spesifikasi Kebutuhan Implementasi Sistem**

Tahap implementasi dan pengujian sistem adalah tahap dimana sistem siap diuji dan dipakai. Tahap ini merupakan tahap lanjutan dari tahap peracangan,analisis dan desain sistem yang bertujuan untuk menguji coba sistem yang telah di buat apakah sesuai dengan tujuan penelitian ini. Dalam tahap ini terdapat pendefinisian spesifikasi terhadap apa saja yang di butuhkan agar sistem dapat berjalan.

Implementasi sistem meliputi :

1. Perangkat Keras (*Hardware*)
2. Perangkat Lunak (*Software*)
3. Pengguna (User)

**5.1.1 Spesifikasi Perangkat Keras (*Hardware*)**

Berikut tabel perangkat keras yang memenuhi spesifikasi :

Tabel 5.1 Spesifikasi Perangkat Keras (*Hardware*)

|  |  |  |
| --- | --- | --- |
| **No.** | **Perangkat Keras** | **Spesifikasi** |
| 1. | Laptop | HP 14-AF115AU AMD Quad Core A6 RAM 4GB 14 Inch |

|  |  |  |
| --- | --- | --- |
| 2. | Mikrokontroler | NodeMCU LUA WIFI V3 4MB 32MBITS,  Sensor Ultrasonik HC-SR04, Motor Servo SG90 dan Kabel jumper |
| 3. | Kabel USB | Micro USB |

**5.1.2 Spesifikasi Perangkat Lunak (*Software*)**

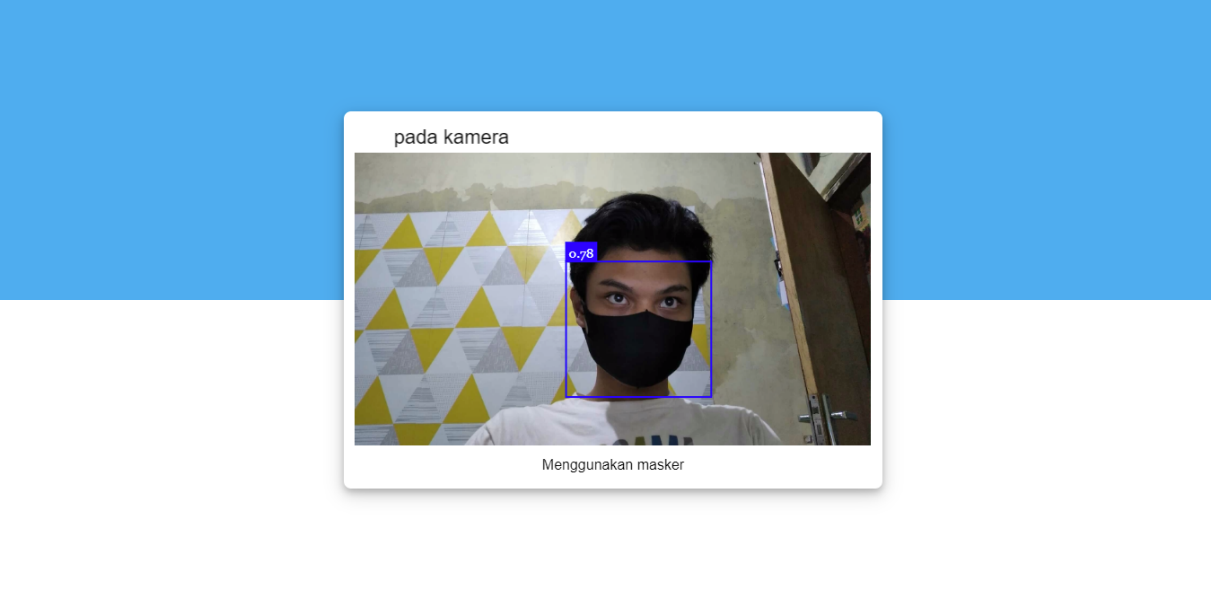
Perangkat lunak yang memenuhi spesifikasi tertera dalam tabel berikut :

Tabel 5.2 Spesifikasi Perangkat Lunak (*Software*)

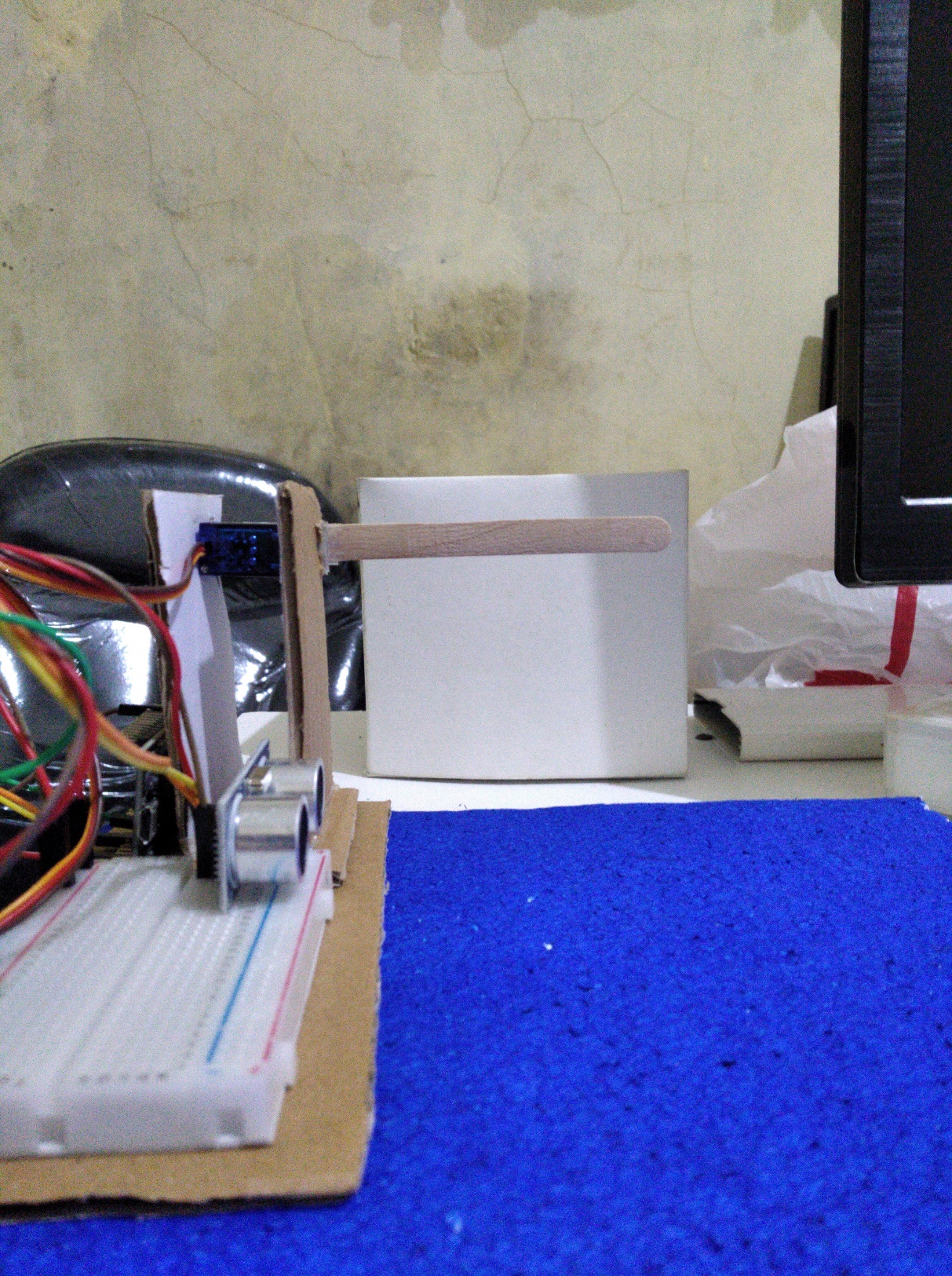
|  |  |  |
| --- | --- | --- |
| **No.** | **Perangkat Keras** | **Spesifikasi** |
| 1. | Sistem Operasi | Windows 10 |
| 2. | Web Server | Apache 2.0 |
| 3. | Code Editor | Visual Studio Code dan Arduino IDE |
| 4. | Browser | Chrome |
| 5. | Bahasa Pemrograman | Node JS dan C |
| 6. | StarUML | Versi 3.0 |

**5.2 Implementasi Antarmuka Sistem**

Implementasi Antarmuka merupakan pemaparan mengenai tampilan aplikasi. Untuk memperjelas bentuk dari implementasi antarmuka, berikut pemaparan dan fungsi dari setiap tampilan yang telah dibuat.

1. Halaman Pendeteksi Masker

Gambar 5.1 Antarmuka Halaman Pendeteksi Masker

1. Simulator Palang Pintu

Gambar 5.2 Antarmuka Simulator Palang Pintu

**5.3 Pengujian perangkat lunak**

Pengujian dilakukan untuk menjamin bahwa perangkat lunak telah berfungsi dengan baik dan sesuai dengan tujuan dari penelitian ini. Pada tahap pengujian penulis menggunakan metode *black box testing*.

**5.3.1 *Black Box Testing***

*Black box testing* befokus pada spesifikasi fungsional dari perangkat lunak, kumpulan kondisi *input* dan melakukan pengetesan pada fungsional program (Mustaqbal, 2015:34).

Tabel 5.3 *Black Box Testing*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Nama Fungsi** | **Kriteria** | **Hasil Test** | **Keterangan** |
| 1. | Mendeteksi masker pada wajah | Berhasil mendeteksi | Berhasil |  |
| 2. | Membuka palang pintu otomatis | Berhasil membuka | Berhasil |  |

**5.3.2 White Box Testing**

*White Box Testing* yaitu menguji perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi-fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi kebutuhan (Rosa dan Shalahuddin, 2013:276).

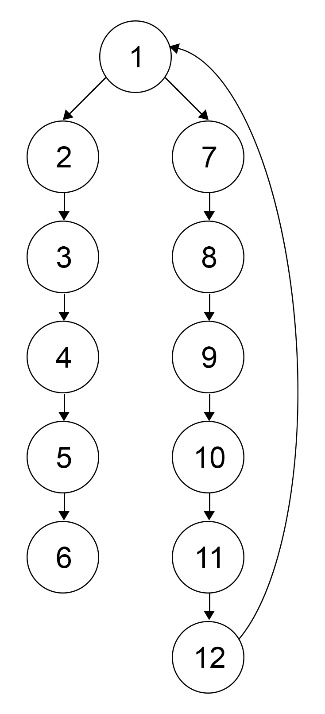
1. Halaman pendeteksi masker
2. *Pseudocode*

**// Jika terdeteksi masker**

1. If (
2. const maskImages = document.querySelectorAll('.mask-img');
3. maskImages.forEach(img => {
4. const tfImg = tf.browser.fromPixels(img);
5. const logits = mobilenetModule.infer(tfImg, 'conv\_preds');
6. classifier.addExample(logits, 0); }));
7. Else (
8. const noMaskImages = document.querySelectorAll('.no-mask-img');
9. noMaskImages.forEach(img => {
10. const tfImg = tf.browser.fromPixels(img);
11. const logits = mobilenetModule.infer(tfImg, 'conv\_preds');
12. classifier.addExample(logits, 1); }));

**// Hasil deteksi**

1. return classifier;
2. *Flow Graph*



Gambar 5.3 *Flow Graph*

1. *Cyclomatic Complexity V(G)*

V(G) = E - N + 2

= 12 - 12 + 2

= 2

1. *Independen Path*

Tabel 5.4 *Independen Path*

|  |  |
| --- | --- |
| **No.** | **Path** |
| 1. | 1, 2, 3, 4, 5, 6 |
| 2 | 1, 7, 8, 9, 10, 11, 12, 1 |