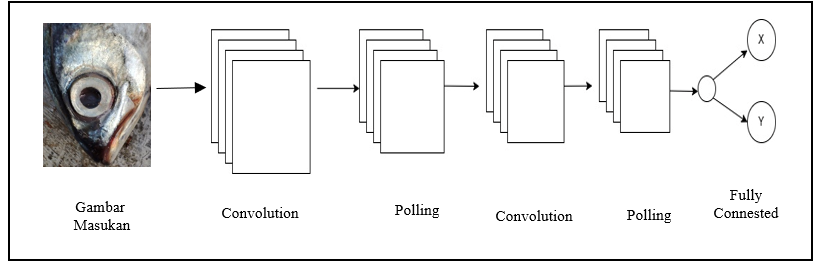
Komposisi web

1. **Daftar Tools**

| **No.** | **Tools/Library** | **Fungsi dalam Proyek** |
| --- | --- | --- |
| 1. | **Python** | Bahasa pemrograman untuk backend dan menjalankan model klasifikasi CNN. |
| 2. | **TensorFlow (Keras)** | Library untuk melatih dan menjalankan model CNN (klasifikasi gambar ikan). |
| 3. | **Flask** / **FastAPI** | Framework Python untuk membangun REST API, menghubungkan frontend dan model prediksi. |
| 4. | **API (Application Programming Interface)** | Penghubung komunikasi antara frontend (React) dan backend (Python). |
| 5. | **React.js** | Framework JavaScript untuk membangun tampilan web interaktif (frontend). |
| 6. | **JavaScript Web API** | Fitur native dari browser seperti pengelolaan file upload, pop-up, atau preview gambar. |
| 7. | **MediaDevices API** | Untuk akses kamera dari browser (HP/laptop), sehingga user bisa ambil gambar langsung. |
| 8. | **Axios** | Library JavaScript untuk mengirim data (gambar) ke backend API dari React. |
| 9. | **Vercel** | Layanan hosting untuk **men-deploy frontend React** agar bisa diakses secara online. |
| 10. | **Google Colab** | Platform cloud untuk melatih model CNN tanpa perlu komputer spesifikasi tinggi. |
| 11. | **VS Code (Visual Studio Code)** | Code editor utama untuk menulis kode frontend (React), backend (Flask), dan model ML. |
| 12. | **Ngrok (opsional)** | Untuk menghubungkan backend lokal ke internet (misalnya saat tes di HP). |
| 13. | **SweetAlert2 (opsional)** | Untuk menampilkan pop-up hasil klasifikasi (misalnya “Segar 85%”) yang menarik. |
| 14. | **HTML5 File Input** | Komponen input file di frontend untuk upload gambar ikan. |
| 15. | **CSS (Tailwind, Bootstrap, atau native)** | Untuk mempercantik tampilan web agar lebih modern dan user-friendly. |
|  |  |  |

1. **Arsitektur sistemnya**

Arsitektur ini terdiri dari beberapa komponen utama yang saling terhubung dan berfungsi secara berurutan:

**1. Gambar Masukan (Input Image)**

Proses dimulai dengan memasukkan gambar sebagai data awal. Citra ini akan diproses lebih lanjut untuk mengenali pola atau ciri tertentu yang dibutuhkan dalam proses klasifikasi.

**2. Lapisan Konvolusi (Convolution Layer)**

Lapisan ini bertugas melakukan ekstraksi ciri dari gambar dengan menggunakan filter (*kernel*). Filter ini bergerak menyapu gambar dan menghasilkan peta fitur (*feature map*) yang menggambarkan karakteristik penting, seperti bentuk tepi atau tekstur.

**3. Lapisan Pooling (Pooling Layer)**

Setelah konvolusi, hasilnya akan diproses pada lapisan *pooling*. Fungsi utama dari lapisan ini adalah mengurangi ukuran data tanpa menghilangkan informasi penting. Salah satu metode yang sering digunakan adalah max pooling, yang mengambil nilai tertinggi dari area tertentu dalam peta fitur.

**4. Konvolusi dan Pooling Tambahan**

CNN umumnya terdiri dari beberapa kombinasi lapisan konvolusi dan pooling secara berurutan. Setiap tahap bertujuan untuk menangkap pola yang lebih kompleks dan mendalam dari gambar input.

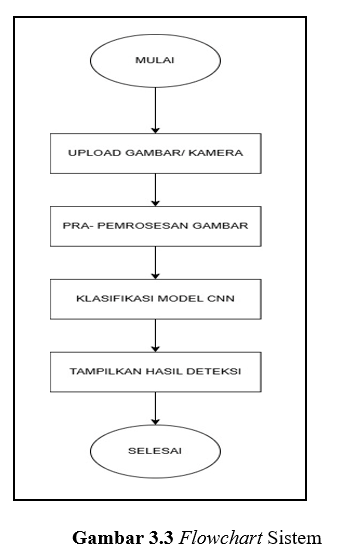
**5. Lapisan Fully Connected (Fully Connected Layer)**

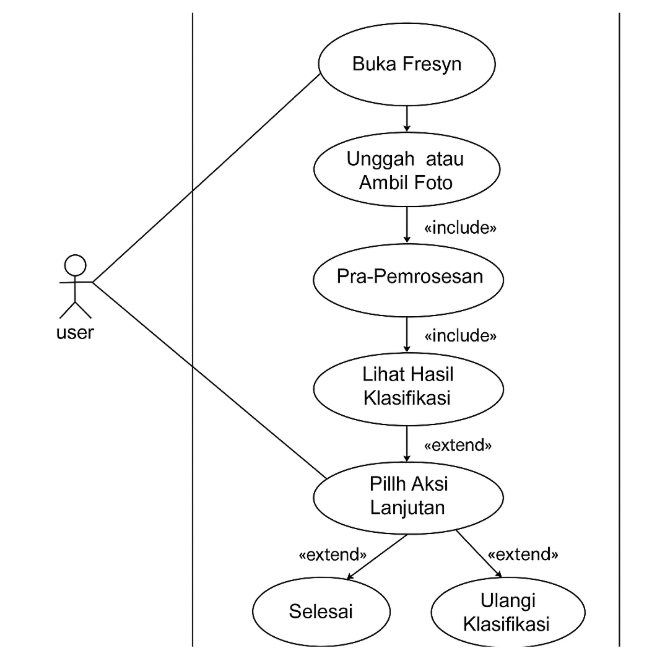
Pada tahap akhir, semua fitur yang telah diperoleh diratakan (*flatten*) dan diproses oleh lapisan fully connected. Lapisan ini berfungsi untuk menentukan hasil klasifikasi berdasarkan fitur-fitur yang telah dianalisis sebelumnya. Sebagai contoh, output-nya bisa berupa dua kelas, seperti "Segar" dan "Tidak Segar".

1. Kebutuhan fungsional

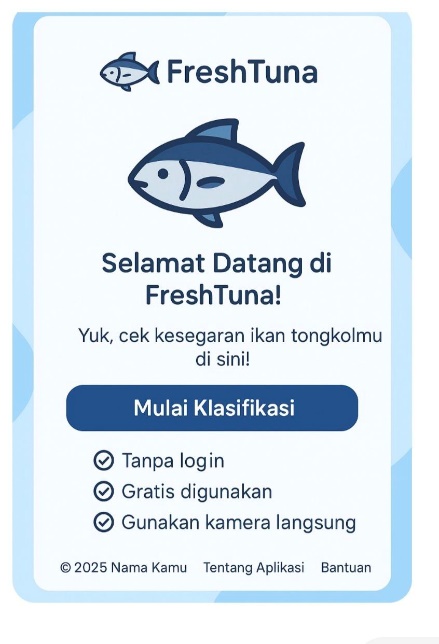
|  |  |  |
| --- | --- | --- |
| **No** | **Kebutuhan Fungsional** | **Deskripsi** |
| 1 | Upload Gambar | Pengguna bisa mengunggah gambar mata ikan dari perangkat mereka ke dalam sistem. |
| 2 | Akses Kamera Real-Time | Sistem menyediakan fitur untuk memotret langsung menggunakan kamera perangkat , sehingga pengguna bisa langsung ambil gambar tanpa harus menyimpan terlebih dahulu. |
| 3 | Deteksi Kesegaran | Setelah gambar dimasukkan, system akan menjalankan proses klasifikasi untuk menentukan apakah ikan tersebut segar atau tidak. |
| 4 | Menampilkan Hasil Deteksi | Hasil dari proses deteksi ditampilkan langsung kepada pnegguna dalam bentuk teks yang mudah dimengerti, seperti “Segar” atau “Tidak Segar”. |

1. **Lowchart system**

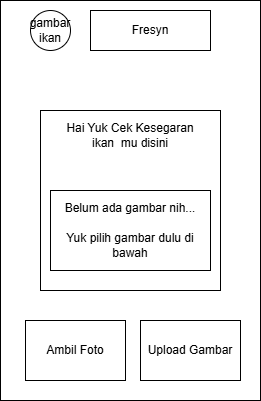


1. **Usecase rancangan sistem**

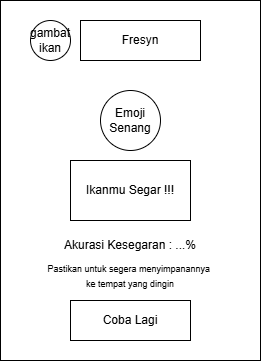
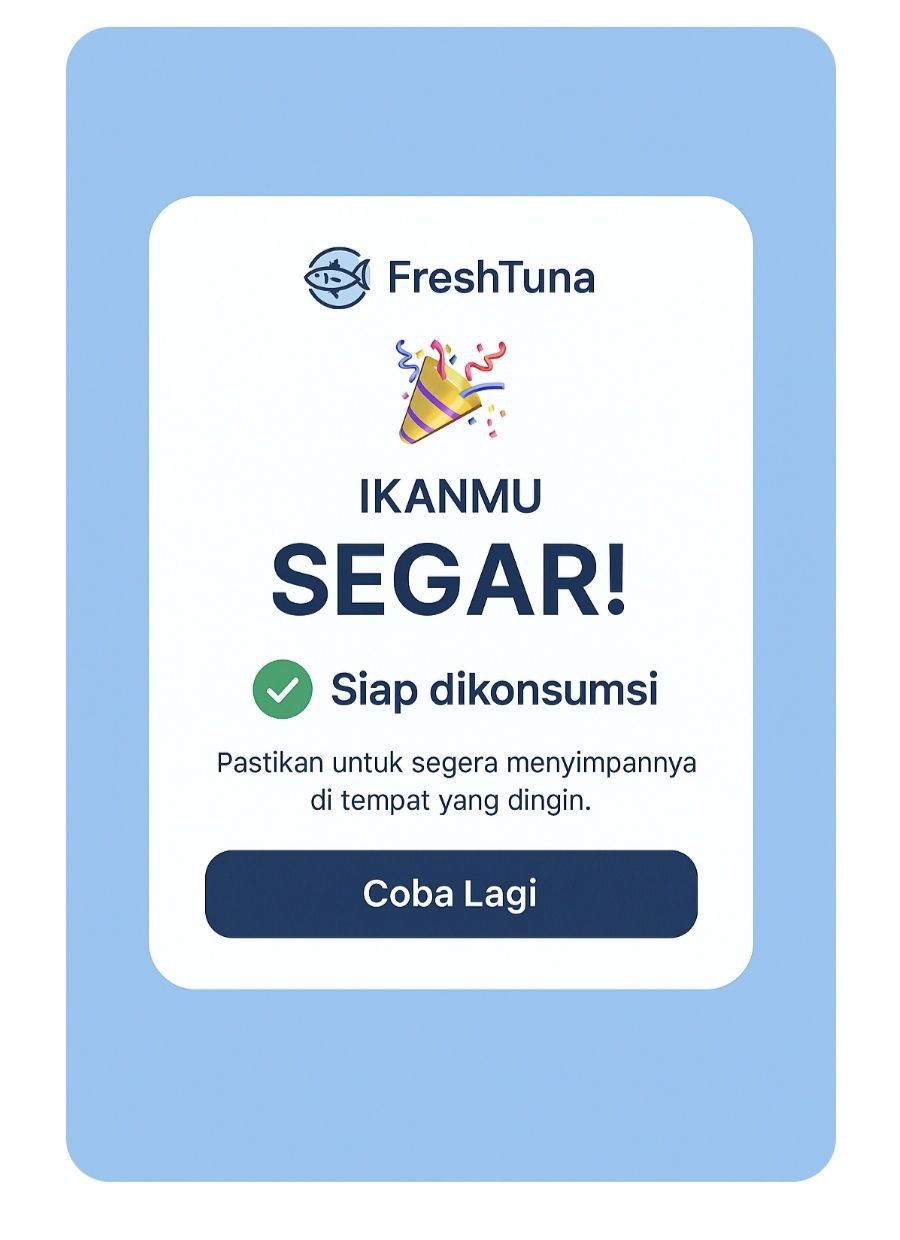
6. Wireframe

**** **1. tampilan Homepage**

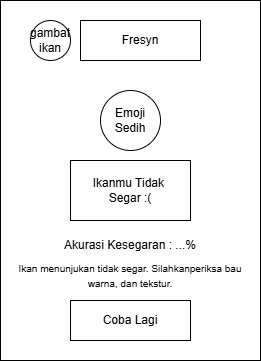
Halaman utama aplikasi Fresyn dibuat dengan desain antarmuka yang simpel dan mudah digunakan, sehingga mempermudah proses klasifikasi kesegaran ikan tongkol. Navigasi yang intuitif lewat tombol utama dan panduan langkah yang jelas memudahkan pengguna dalam menjalankan aplikasi dengan efisien.

**2.tampilan Main Menu**

Halaman wireframe ini menggambarkan tampilan halaman kedua dari aplikasi web *Fresyn* yang didesain untuk mempermudah pengguna dalam memulai proses klasifikasi tingkat kesegaran ikan tongkol. Pengguna dapat memilih salah satu dari dua opsi input gambar, yaitu mengambil foto langsung menggunakan kamera atau mengunggah gambar dari perangkat. Tersedia juga area pratinjau untuk menampilkan gambar yang telah dipilih sebelum diproses lebih lanjut. Dengan antarmuka yang sederhana dan mudah dipahami, halaman ini menjadi tahapan awal sebelum sistem melakukan klasifikasi menggunakan model CNN.

**3.Tampilan Pop Up Segar**

Wireframe ini menggambarkan tampilan website *Fresyn* ketika ikan yang dianalisis terdeteksi dalam kondisi “segar”. Sistem menampilkan informasi berupa teks hasil klasifikasi, emoji ekspresif (senang), tingkat akurasi prediksi, serta saran penyimpanan ikan. Disediakan pula tombol untuk melakukan analisis ulang.

**4. Tampilan Pop Up Tidak Segar**



Wireframe ini memperlihatkan tampilan aplikasi Fresyn saat hasil analisis menunjukkan bahwa ikan berada dalam kondisi tidak segar. Informasi yang ditampilkan meliputi teks hasil klasifikasi, emoji ekspresif (sedih atau netral), tingkat akurasi prediksi, serta pesan peringatan atau saran penanganan lebih lanjut. Tersedia juga tombol untuk melakukan analisis ulang.