

**KLASIFIKASI JENIS IKAN HIU MENGGUNAKAN METODE  
CONVOLUTIONAL NEURAL NETWORK**

**SKRIPSI**

Sebagai Salah Satu Syarat untuk Meraih  
Gelar Sarjana Komputer

Oleh :

Davin Christian Juan  
32180045



Fakultas Teknik dan Desain  
Program Studi Informatika  
Universitas Bunda Mulia  
Tangerang  
2022

**UNIVERSITAS BUNDA MULIA  
FAKULTAS TEKNOLOGI DAN DESAIN  
PROGRAM STUDI INFORMATIKA**

## Pernyataan Kesiapan Ujian Pendadaran Skripsi

Saya Davin Christian Juan, dengan ini menyatakan bahwa Skripsi yang berjudul  
:

## **“KLASIFIKASI JENIS IKAN HIU MENGGUNAKAN METODE**

## CONVOLUTIONAL NEURAL NETWORK ”

merupakan hasil karya saya dan belum pernah diajukan sebagai karya ilmiah, sebagian atau seluruhnya, atas nama saya atau pihak lain.

---

Davin Christian Juan  
32180045

Disetujui oleh Pembimbing,  
Kami setuju Skripsi tersebut diajukan untuk Ujian Pendadarhan

---

Dionisia Bhisetya Rarasati, S.Kom., M.T.I

(20 Desember 2022)

Disetujui oleh Ketua Program Studi,

---

Dr. Fransiskus Adikara, S.Kom., MM.

(20 Desember 2022)

**UNIVERSITAS BUNDA MULIA  
FAKULTAS TEKNOLOGI DAN DESAIN  
PROGRAM STUDI INFORMATIKA**

**Persetujuan Skripsi**

Yang bertandatangan di bawah ini, menyatakan bahwa Skripsi dengan judul  
**“KLASIFIKASI JENIS IKAN HIU MENGGUNAKAN METODE  
CONVOLUTIONAL NEURAL NETWORK”**

Disusun oleh :

Davin Christian Juan  
32180045

Telah disetujui dan diterima sebagai salah satu karya ilmiah mahasiswa yang bersangkutan pada Fakultas Teknologi dan Desain – Program Studi Informatika Universitas Bunda Mulia.

Tangerang, 20 Desember 2022

Mengetahui,  
Ketua Program Studi,

Dosen Pembimbing,

Dr. Fransiskus Adikara, S.Kom., Dionisia Bhisetya Rarasati, S.Kom.,  
MM. M. T.I

## **PERNYATAAN**

Saya menyatakan bahwa Skripsi yang berjudul "**KLASIFIKASI JENIS IKAN HIU MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK**" sepenuhnya karya saya sendiri. Tidak ada bagian di dalamnya yang merupakan plagiat dari karya orang lain dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan. Atas pernyataan ini, saya siap menanggung resiko/sanksi yang dijatuhkan kepada saya, atau ada klaim dari pihak lain terhadap keaslian karya saya ini.

Tangerang, 20 Desember 2022

Yang membuat pernyataan,

*Materai Rp10.000,-  
ttd*

Davin Christian Juan

## ABSTRAK

Hiu adalah ikan yang termasuk kedalam kelompok ikan bertulang rawan, atau ikan hiu dapat disebut sebagai *chondrichthyes*. Ikan hiu sendiri masuk ke dalam satu kelompok dengan ikan pari dan hiu hantu. Penelitian tentang klasifikasi jenis ikan hiu ini menggunakan metode *Convolutional Neural Network* dan berbasis web.

Penelitian ini melakukan proses klasifikasi gambar dari masing-masing ikan hiu. Proses klasifikasi gambar menggunakan metode *Convolutional Neural Network*. CNN termasuk ke dalam Supervised Learning dikarenakan setiap jenis ikan dimasukkan ke dalam kelas atau adanya pelabelan terhadap jenis-jenis ikan hiu tersebut. Setelah dilakukan klasifikasi gambar, maka akan dilakukan perhitungan akurasi dengan menggunakan *confusion matrix*.

Tujuan dari penelitian ini yaitu untuk melihat seberapa baik model yang digunakan untuk melakukan pengklasifikasian terhadap jenis-jenis ikan hiu. Sebelum dimulainya klasifikasi gambar dilakukan sebuah pengumpulan data sampel yang berasal dari Kaggle. Dari data sampel tadi akan dijadikan menjadi data latih (train) dan data uji (test). Pengujian aplikasi ini digunakan 60 Data Test untuk melakukan pengujian. Hasil pengujian dari 60 data yang diujikan mendapatkan hasil akurasi sebesar 73,3% yang berarti pengujian aplikasi ini memiliki tingkat akurasi yang tinggi.

Penelitian ini dapat disimpulkan bahwa metode *Convolutional Neural Network* dapat digunakan untuk melakukan klasifikasi jenis ikan hiu dengan hasil yang cukup baik. Untuk penelitian selanjutnya, diharapkan untuk memperbanyak Data Set untuk mendapatkan tingkat akurasi yang lebih tinggi.

### Kata Kunci:

Hiu, *Confusion Matrix*, *Convolution Neural Network*, Klasifikasi Gambar.

## **ABSTRACT**

*Sharks are fish that belong to the cartilaginous fish group, or sharks can be referred to as Chondrichthyes. Sharks themselves enter into one group with stingrays and ghost sharks. Research on the classification of shark species uses the Convolutional Neural Network method and is web-based.*

*This study carried out the process of classifying images of each shark. The image classification process uses the Convolutional Neural Network method. CNN is included in Supervised Learning because each type of fish is included in the class or there is a label for the type of shark. After classifying the images, the accuracy calculation will be carried out using the confusion matrix.*

*The purpose of this research is to see how well the model is used to classify shark species. Prior to the start of image classification, a sample data collection was carried out from Kaggle. The sample data will be used as training data (train) and test data (test). Testing this application uses 60 Test Data to do the testing. The test results of the 60 tested data obtained an accuracy of 73.3%, which means that this application test has a high level of accuracy.*

*It can be concluded from this study that the Convolutional Neural Network method can be used to classify shark species with fairly good results. For further research, it is expected to reproduce the Data Set to obtain a higher level of accuracy.*

**Keywords:**

*Shark, Confusion Matrix, Convolutional Neural Network, Image Classification.*

## **PRAKATA**

Puji dan syukur kepada TME karena atas berkatnya sehingga penulis dapat menyelesaikan skripsi yang berjudul “Klasifikasi Jenis Ikan Hiu Menggunakan Metode Convolutional Neural Network” dengan tepat waktu. Skripsi ini dilakukan untuk memenuhi persyaratan dalam memperoleh gelar sarjana komputer.

Penulis ingin mengucapkan terimakasih kepada berbagai pihak yang telah membantu dalam memberikan masukan dan bimbingan, yaitu kepada:

1. Bapak Doddy Surja Bajudji, S.E., M.B.A., selaku Rektor Universitas Bunda Mulia.
2. Bapak Howard S.Giam, S.E., Ak., M.B.A., selaku Pelaksana Harian Rektor Universitas Bunda Mulia.
3. Ibu Kandi Sofia Senastri Dahlan, S.E., M.B.A., Ph.D., selaku Wakil Rektor Bidang Akademik Universitas Bunda Mulia.
4. Ibu Shanty Sudarji, S.Psi., M.Psi., Psikolog, selaku Direktor Universitas Bunda Mulia.
5. Bapak Dr.Fransiskus Adikara, S.Kom., MM., selaku Dekan Fakultas Teknologi dan Desain Universitas Bunda Mulia.
6. Ibu Henny Hartono, S.Kom., M.M., selaku Head of Akademik 1 Universitas Bunda Mulia.
7. Bapak Dr. Fransiskus Adikara, S.Kom., MM., selaku Ketua Program Studi Teknik Informatika.
8. Bapak Ignatius Adrian Mastan, S.E., S.Kom., S.A.B., M.M., M.Eng., selaku Manager Akademik 1 Universitas Bunda Mulia.
9. Ibu Dionisia Bhisetya Rarasati, S.Kom., M.T.I., selaku Dosen Pembimbing yang telah membimbing, dan memberikan masukan dari awal penyusunan skripsi.

10. Seluruh Dosen Universitas Bunda Mulia yang memberikan ilmu kepada penulis selama masa perkuliahan.
11. Keluarga serta teman - teman yang tidak dapat disebutkan satu persatu, yang telah mendukung penulis dalam menyelesaikan skripsi ini hingga selesai.
12. Jennie Ruby Jane, sebagai pacar yang selalu mendukung dalam pembuatan skripsi penulis.
13. Lalisa Manoban, sebagai sahabat yang selalu mendukung dalam pembuatan skripsi penulis.

Penulis menyadari bahwa laporan skripsi ini memiliki kekurangan, maka dari itu penulis membutuhkan kritik dan saran, dan penulis berharap laporan penelitian ini dapat bermanfaat.

Tangerang, 20 Desember 2022

Davin Christian Juan

## DAFTAR ISI

ABSTRAK .....	i
<i>ABSTRACT</i> .....	ii
PRAKATA .....	iii
DAFTAR ISI .....	v
DAFTAR TABEL .....	vii
DAFTAR GAMBAR .....	viii
BAB 1 PENDAHULUAN .....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	2
1.3 Tujuan & Manfaat .....	3
1.4 Ruang Lingkup .....	3
1.5 Metodologi Penelitian.....	4
1.5.1 Teknik Pengumpulan Data.....	4
1.6 Sistematika Penelitian.....	4
BAB 2 LANDASAN TEORI .....	6
2.1 Hiu .....	6
2.2 <i>Citra</i> .....	10
2.3 <i>Citra Digital</i> .....	10
2.4 Python .....	11
2.5 <i>Image Classification</i> .....	11
2.6 <i>Machine Learning</i> .....	11
2.7 <i>Convolutional Neural Network</i> .....	12
2.8 UML ( <i>Unified Modeling Language</i> ) .....	16
2.9 <i>Flowchart</i> .....	16
2.10 <i>Use Case Diagram</i> .....	19
2.11 <i>Activity Diagram</i> .....	20
2.12 <i>Sequence Diagram</i> .....	22
2.13 <i>Blok Diagram</i> .....	23
2.14 <i>System Development Life Cycle (SDLC)</i> .....	23
2.15 <i>Confusion Matrix</i> .....	25
2.16 Penelitian Terdahulu .....	26
BAB 3 METODE PENELITIAN .....	29
3.1 Kerangka Berpikir .....	29
3.2 Pemilihan Algoritma .....	30
3.3 Perancangan Proses .....	30
3.3.1 Pengumpulan Data.....	32
3.3.2 Pre-processing .....	32
3.4 Perancangan Sistem.....	32
3.4.1 <i>Use Case Diagram</i> .....	32
3.4.2 <i>Sequence Diagram</i> .....	34

3.4.3	Blok Diagram .....	35
3.5	Perancangan Tampilan.....	36
3.5.1	Tampilan Masukan (Input).....	36
3.5.2	Tampilan Keluaran (Output) .....	36
3.6	<i>Input Layer</i> .....	37
3.7	<i>Convolution Layer</i> .....	38
3.8	<i>Activation Function</i> .....	39
3.9	<i>Pooling Layer</i> .....	40
3.10	<i>Flatten Layer</i> .....	41
3.11	<i>Fully Connected Layer</i> .....	42
3.12	Perencanaan Pengujian .....	44
3.14	Simulasi Perhitungan CNN .....	44
3.15	Jadwal Pengerjaan .....	48
<b>BAB 4</b>	<b>IMPLEMENTASI.....</b>	<b>49</b>
4.1	Kebutuhan Perangkat.....	49
4.1.1	Perangkat Keras ( <i>Hardware</i> ).....	49
4.1.2	Perangkat Lunak ( <i>Software</i> ).....	49
4.2	Implementasi Antarmuka Pengguna.....	50
4.2.1	Tampilan Halaman Utama .....	50
4.2.2	Tampilan Peringatan pada Halaman Utama.....	51
4.2.3	Tampilan Hasil Klasifikasi.....	52
4.3	Implementasi Metode atau Algoritma .....	52
4.3.1	<i>Preprocessing Image</i> .....	53
4.3.2	Arsitektur <i>Convolutional Neural Network</i> .....	54
4.3.3	Membagi Dataset .....	55
4.4	<b>Perhitungan Akurasi di Aplikasi</b> .....	70
4.5	<b>Evaluation Method</b> .....	71
<b>BAB 5</b>	<b>PENUTUP.....</b>	<b>72</b>
5.1	Simpulan .....	72
5.2	Saran .....	72
5.3	Keterbatasan Penelitian .....	72
5.4	Kontribusi Penelitian .....	72
<b>DAFTAR REFERENSI</b>	.....	<b>73</b>
<b>RIWAYAT HIDUP</b>	.....	<b>77</b>
<b>LAMPIRAN</b>		

## **DAFTAR TABEL**

	Hal
Tabel 2.1 Simbol dalam Flowchart .....	17
Tabel 2.2 Simbol dalam <i>Use Case</i> .....	19
Tabel 2.3 Simbol dalam <i>Activity Diagram</i> .....	21
Tabel 2.4 Simbol dalam <i>Sequence Diagram</i> .....	22
Tabel 2.5 Penelitian Terdahulu .....	26
Tabel 3.1 Jadwal Pengerjaan.....	48
Tabel 4.1 <i>Hardware</i> .....	49
Tabel 4.2 <i>Software</i> .....	50
Tabel 4.3 Hasil pembagian Dataset .....	56

## DAFTAR GAMBAR

	Hal
Gambar 2.1 Convolution Layer .....	14
Gambar 2.2 Pooling Layer .....	15
Gambar 2.3 Fully Connected Layer .....	16
Gambar 2.4 <i>System Development Life Cycle</i> .....	24
Gambar 3.1 Kerangka Berpikir .....	29
Gambar 3.2 <i>Flowchart</i> Perancangan Proses .....	31
Gambar 3.3 Pre-processing .....	32
Gambar 3.4 <i>Use Case Diagram</i> .....	33
Gambar 3.5 <i>Sequence Diagram</i> .....	34
Gambar 3.6 Blok Diagram .....	35
Gambar 3.7 Halaman Awal .....	36
Gambar 3.8 Halaman Akhir .....	36
Gambar 3.9 <i>Input Layer</i> .....	37
Gambar 3.10 <i>Convolution Layer</i> .....	39
Gambar 3.11 <i>Activation Function</i> .....	40
Gambar 3.12 Simulasi <i>Pooling Layer</i> .....	41
Gambar 3.13 Simulasi <i>Flatten Layer</i> .....	42
Gambar 3.14 Simulasi <i>Fully Connected Layer</i> .....	43
Gambar 3.15 Perencanaan Pengujian .....	44
Gambar 3.16 Simulasi Perhitungan CNN .....	45
Gambar 4.1 Halaman Utama .....	50
Gambar 4.2 Tampilan Peringatan .....	51
Gambar 4.3 Hasil Klasifikasi .....	52
Gambar 4.4 Preprocessing Image .....	53
Gambar 4.5 Arsitektur CNN .....	54
Gambar 4.6 Hasil Model Summary .....	55
Gambar 4.7 Membagi Dataset .....	55
Gambar 4.8 Grafik Akurasi data latih dan validasi .....	69
Gambar 4.9 Grafik Loss data latih dan validasi .....	69
Gambar 4.10 Keberhasilan Metode .....	70

# BAB 1

## PENDAHULUAN

### 1.1 Latar Belakang

Hiu adalah ikan yang termasuk kedalam kelompok ikan bertulang rawan, atau ikan hiu dapat disebut sebagai *chondrichthyes*. Ikan hiu sendiri masuk ke dalam satu kelompok dengan ikan pari dan hiu hantu. Hiu digolongkan sebagai kelompok ikan fosil hidup karena kelompok ikan ini belum terlalu berubah secara morfologi jika membandingkannya dengan leluhurnya yang ada jutaan tahun yang lalu [1].

*Computer Vision* adalah salah satu cabang ilmu komputer yang membahas bagaimana komputer memungkinkan untuk melihat, mempelajari, dan memproses gambar menggunakan analisis yang sama seperti mata manusia, setelah itu informasi yang diperlukan diekstraksi. *Computer vision* memiliki konsep bagaimana menanamkan kecerdasan manusia ke dalam komputer. Bagian paling penting dalam *computer vision* yaitu proses awal dari *computer vision* itu sendiri [2].

Gambar adalah representasi objek dua dimensi dari dunia visual, yang mencakup berbagai disiplin ilmu seperti seni, penglihatan manusia, astronomi, teknologi, dll. Citra juga merupakan sebuah kumpulan *pixel-pixel* atau yang biasa disebut dengan titik-titik yang berwarna dan berbentuk dua dimensi [3].

Klasifikasi terhadap citra adalah sebuah proses di mana pengelompokan terhadap sebuah *pixel* pada sebuah citra ke dalam sebuah kelompok yang menyebabkan terjadi diinterpretasikannya sebagai suatu *property* yang spesifik.

Terdapat dua proses dalam klasifikasi citra, antara lain klasifikasi tak terbimbing dan klasifikasi terbimbing [4].

Metode CNN adalah pengklasifikasi citra yang diawali dengan melakukan pelatihan terhadap data yang digunakan. Proses melatih data ini berfungsi agar data yang digunakan dapat mengenali objek yang akan diuji. Setelah data tersebut di latih agar dapat mengenali objek lalu data tersebut diuji agar mengetahui apakah objek tersebut termasuk ke dalam label yang mana. Setelah data diuji maka akan diperoleh tingkat akurasi dari objek tersebut dengan label [5].

Sebelum dimulainya klasifikasi gambar dilakukan sebuah pengumpulan data sampel yang berasal dari Kaggle. Dari data sampel tadi akan dijadikan sebuah data latih (*train*) dan data uji (*test*). Dengan begitu dapat dilakukan klasifikasi pada gambar yang diinginkan [6].

Berdasarkan kasus ini, bahwa klasifikasi gambar jenis ikan hiu merupakan topik utama untuk menentukan tingkat akurasi dari jenis ikan hiu yang berada pada situs Kaggle agar mengetahui jenis-jenis ikan hiu dengan cara menentukan tingkat akurasi tersebut. Maka penulis membuat sebuah aplikasi tentang *image classification* dengan metode CNN (*Convolutional Neural Network*).

## 1.2 Rumusan Masalah

Berdasarkan uraian latar belakang terdapat sebuah rumusan masalah sebagai berikut:

1. Bagaimana algoritma CNN dalam klasifikasi jenis-jenis ikan hiu?
2. Bagaimana tingkat akurasi menggunakan confusion matrix terhadap klasifikasi jenis-jenis ikan hiu?

### 1.3 Tujuan & Manfaat

#### 1.3.1 Tujuan

Adapun tujuan dari klasifikasi jenis-jenis ikan hiu menggunakan *image classification* dengan metode CNN adalah untuk melihat seberapa baik model yang telah dibuat dalam melakukan klasifikasi jenis-jenis ikan hiu.

#### 1.3.2 Manfaat

Adapun manfaat dari klasifikasi jenis-jenis ikan hiu menggunakan *image classification* dengan metode CNN adalah sebagai berikut:

1. Menjadi acuan pada penelitian yang sejenis.
2. Menambah pengetahuan mengenai jenis-jenis ikan hiu.

### 1.4 Ruang Lingkup

Adapun batasan dari klasifikasi jenis-jenis ikan hiu menggunakan *image classification* dengan metode Convolutional Neural Network adalah sebagai berikut:

1. Image Classification berbasis website.
2. Menggunakan bahasa pemrograman Python.
3. Menggunakan PyCharm sebagai teks editor.
4. Objek yang digunakan dalam penelitian ini yaitu jenis-jenis ikan hiu.
5. Menggunakan 5 jenis ikan hiu yaitu Hiu Paus, Hiu Sirip Putih, Hiu Putih, Hiu Martil, dan Hiu Tikus.
6. Menampilkan tingkat akurasi dari jenis-jenis ikan hiu.

## 1.5 Metodologi Penelitian

### 1.5.1 Teknik Pengumpulan Data

Teknik pengumpulan data yang akan digunakan untuk penelitian yaitu dengan menggunakan studi pustaka, yaitu mengumpulkan penelitian, artikel, contoh *project* maupun jurnal penelitian sebelumnya yang berhubungan dengan *image classification* untuk dijadikan sebagai bahan referensi dan sebagai landasan teori dari penelitian ini.

## 1.6 Sistematika Penelitian

Penelitian ini akan ditulis dengan sistematika sebagai berikut:

### 1. BAB 1 PENDAHULUAN

Pada bab ini akan menjelaskan mengenai latar belakang, tujuan dan manfaat penelitian, rumusan masalah, batasan masalah, metodologi penelitian, dan sistematika penelitian.

### 2. BAB 2 LANDASAN TEORI

Pada bab ini, akan dijelaskan materi yang digunakan sebagai landasan teori pada perancangan dan penyusunan penelitian.

### 3. BAB 3 ANALISIS DAN PERANCANGAN

Pada bab ini menguraikan tentang jenis penelitian, analisa perancangan aplikasi dan metode-metode yang digunakan dalam menunjang perancangan aplikasi.

### 4. BAB 4 IMPLEMENTASI

Pada bab ini membahas mengenai informasi yang dihasilkan dari aplikasi yang dirancang dengan metode yang telah diterapkan oleh penulis.

### 5. BAB 5 SARAN DAN KESIMPULAN

Pada bab ini merupakan penutup yang telah didapatkan dari hasil pembahasan serta saran-saran yang dapat menjadi masukan dan manfaat sebagai bahan pertimbangan bagi pembaca.

## **BAB 2**

### **LANDASAN TEORI**

Landasan Teori menyajikan teori-teori yang relevan, dan sejalan dengan penelitian penulis. Teori-teori yang dikemukakan oleh penulis berasal dari jurnal-jurnal dari hasil penelitian terdahulu. Berikut teori-teori yang digunakan dalam penelitian penulis:

#### **2.1 Hiu**

Berdasarkan karakteristik biologis hiu, hiu cenderung tumbuh lambat, mungkin memiliki masa hidup yang panjang, dan hiu juga mencapai kematangan seksual dan siklus reproduksinya dengan lambat. Selain karakteristik biologis ini, hiu umumnya bereproduksi melalui kelahiran, sehingga betina cenderung menghasilkan lebih sedikit anak hiu selama siklus reproduksi. Kelompok hiu rentan terhadap *overfishing* dan *overfishing* karena pada dasarnya hiu hanya melahirkan anak tidak lebih dari 10 ekor, sehingga ketika populasi hiu terganggu di alam, untuk rentan kembali pulih memerlukan kurun waktu yang cukup lama [1]. Berikut beberapa penjelasan mengenai ikan hiu yang digunakan oleh penulis:

##### **1. Hiu Paus (*Rhincodon typus*)**

Hiu paus adalah jenis hiu terbesar di dunia dan terbesar dari jenis hiu lainnya. Panjang tubuh hiu paus dapat mencapai 12 meter, hingga mencapai 18 meter. Selain hiu paus mempunyai ukuran badan yang sangat panjang, hiu paus juga mempunyai ciri-ciri seperti struktur kepala yang lebar dan rata serta mulut yang agak besar di bagian depan. Tubuh hiu paus sendiri ditutupi oleh kulit yang tebal dengar garis-garis

yang menonjol di sepanjang sisi tubuhnya, dan memiliki corak warna abu-abu dengan totol-totol berwarna putih atau kekuningan. Cotak totol-totol ini terkadang menjadi suatu masalah ketika saat ingin mengidentifikasi bagian tubuhnya seperti sirip. Sirip hiu paus sekilas mirip dengan sirip pari kupu-kupu atau *Rhina aencylostoma*, karena sama-sama memiliki warna kelabu dan totol-totol putih, akan tetapi keduanya dapat dibedakan berdasarkan ukurannya. Tinggi sirip pari kupu-kupu pada umumnya hanya berukuran kurang dari 30 cm, sedangkan sirip hiu paus pada umumnya lebih rapi membentuk garis dan terdapat jarak, sedangkan totol-totol pada pari kupu-kupu relatif tidak beraturan dan lebih rapat [1].

## 2. Hiu Sirip Putih (*Triaenodon obesus*)

Hiu sirip putih atau *Triaenodon obesus* merupakan jenis hiu yang berkeliaran dekat terumbu karang, dan dikenal dengan nama *Whitetip Reef Shark*. Hiu sirip putih memiliki ciri umum yaitu ujung sirip punggung dan ujung sirip ekor berwarna putih, panjang jenis ikan hiu ini dapat mencapai 200 meter pada jantannya, dan untuk betinanya dapat mencapai ukuran 105-120 meter. Jenis ikan hiu ini dapat dijumpai diseluruh perairan Indo-Pasifik, hidup di daerah pantai atau dasar perairan yang bercelah atau berlubang dan di daerah terumbu karang yang berair jernih, dengan kedalaman antara 1-40 meter. Menurut data konservasi internasional yaitu data IUCN atau *International Union for Conservation Nature*, spesies ini berada di kategori *near threatened* atau hampir punah [7].

### 3. Hiu Putih (*Carcharodon Carcharias*)

Hiu putih atau yang sering dikenal dengan *great white shark* merupakan sebuah ikon ikan hiu sebagai ikan pembunuh manusia, karena terdapat film-film *Hollywood* yang menyesatkan seperti *Jaws*, *Shark Night*, *Shark Attack*, *Red Water*, *Deep Blue Sea*, dan lain sebagainya. Namun dengan adanya film tersebut, ikan hiu jadi lebih dikenal dan diidentifikasi. Hiu putih juga termasuk kedalam kelompok hiu *Lamniformes* yang memiliki lunas pada pangkal ekornya. Lunas pada hiu putih menandakan bahwa hiu putih merupakan hiu yang dapat berenang dengan cepat. Hiu putih merupakan ikan hiu terbesar yang bukan pemakan plankton atau dalam artian kata merupakan sebuah predator sejati. Hiu putih sendiri memiliki ukuran tubuh mencapai 7,2 meter, tetapi pada umumnya sering ditemukan dengan ukuran maksimal antara 5-6 meter. Bentuk tubuh hiu putih sangat mudah diidentifikasi terutama karena ukurannya yang relatif besar dari jenis-jenis hiu lainnya. Bentuk sirip punggungnya segitiga tegak dengan ujung meruncing, sama halnya dengan bagian sirip dadanya. Sedangkan gigi dari hiu putih sering kali dijadikan aksesoris oleh manusia karena bentuk giginya segitiga yang hampir simetris dan relatif besar [1].

### 4. Hiu Martil (*Sphyrna spp.*)

Hiu martil merupakan jenis hiu yang paling mudah dikenali apabila masih dalam kondisi utuh, karena bentuk kepalanya yang sangat khas, yaitu pipih dan berbentuk seperti kepala matril. Hiu martil jenis *Sphyrna Lewini*, merupakan jenis yang paling umum ditemukan oleh kita di

Indonesia, dikenali dengan bentuk ujung kepala yang sedikit melengkung dengan adanya lekukan dibagian tengahnya, dan sisi samping di belakang mata jenis hiu ini berbentuk cekung. Hiu jenis martis besar (*Sphyraena Mokarran*) memiliki ujung kepala yang relatif rata dengan sedikit lekukan di tengahnya, dan bagian sisi samping di belakang mata terlihat relatif lurus. Jenis hiu martil halus (*Sphyraena Zygaena*) memiliki bentuk ujung kepala yang melengkung tanpa adanya lekukan dibagian tengahnya, sedangkan bagian sisi samping di belakang mata terlihat melengkung ke belakang. Sirip punggung hiu martil dapat dibedakan dengan sirip dada dengan melihat kedua bagian sisinya. Warna sirip punggung pada kedua sisi siripnya sama, sedangkan sirip dada memiliki warna yang berlainan. Sirip punggung hiu martil dapat dibedakan dengan sirip hiu jenis lainnya dengan mengukur tinggi sirip dari sisi depannya hingga ke ujung atas sirip, kemudian mengukur lebar sirip punggung [1].

##### 5. Hiu Tikus (*Alopias spp.*)

Hiu tikus atau *Alopias spp.* merupakan kelompok hiu yang berekor panjang yang hidup di perairan paparan benua hingga oseanik. Warna sirip dari hiu *Alopias* cenderung lebih gelap dibandingkan dengan hiu dari marga *Carcharhinus*, dalam bentuk keringnya biasanya ditemukan warna kelabu gelap atau kehitaman [1].

## 2.2 *Citra*

Citra adalah sebuah pantulan (*image*), kesamaan, atau tiruan dari suatu objek. Citra sendiri terbagi menjadi 2, citra yang memiliki sifat identik dan citra yang memiliki sifat nomorik. Citra identik yaitu sebuah gambar yang bersifat berkelanjutan, contohnya gambar di monitor televisi, foto sinar X, dll. Sedangkan citra nomorik yaitu citra yang diperoleh dari komputer [8].

## 2.3 *Citra Digital*

Citra digital mewakili gambar yang ditangkap oleh mesin menggunakan pendekatan berbasis sampling dan kuantisasi. Pengambilan sampel menunjukkan ukuran kotak di letakkan di kolom dan baris. Dengan kata lain, pola pada gambar menunjukkan ukuran *pixel* (titik) yang sedikit di sebuah gambar, dan kuantisasi mewakili ukuran tingkat kecerahan, yang disebut sebagai nilai keabuan atau yang sering dikenal dengan *greyscale* sebesar banyaknya bit biner yang dipakai oleh mesin tersebut, dapat diartikan kuantisasi pada gambar menyatakan banyaknya warna terdapat di gambar tersebut. Citra memiliki 3 jenis yaitu citra biner, citra warna dan citra *grayscale*. Citra biner atau monokrom mempunyai dua warna, hitam dan putih, untuk menyimpan warna hitam dan putih didalam memori memerlukan 1 *bit*. Citra *Grayscale* atau skala keabuan di beberapa bit di dalam memori mempunyai beragam untuk memenuhi persyaratan warna tersebut, c gambar 2 bit mewakilkan 4 warna, gambar 3 bit mewakilkan 8 warna, dll. Jenis citra yang terakhir yaitu citra berwarna atau *true color*, dimana piksel yang ada pada citra berwarna tersebut merepresentasikan warna gabungan dari ketiga warna primer (RGB), setiap warna primer menggunakan penyimpanan 8 bit = 1 *byte*, yang memiliki tujuan agar satuan *pixel* mempunya kombinasi warna sebesar  $28 \times 28 \times 28 = 224 = 16$  juta warna lebih [8].

## 2.4 Python

Python merupakan salah satu bahasa pemrograman yang banyak dipergunakan dan memiliki kelebihan sebagai berikut [9]:

1. Sangat gampang penggunaanya dalam mengembangkan sebuah perangkat lunak, perangkat keras, *IOT*, aplikasi berbasis web, dan juga *video game*.
2. Python mempunyai tingkat pemahaman kode yang bagus sehingga membuat kode mudah dipahami, sehingga *python* mempunyai banyak *library* untuk digunakan.

## 2.5 *Image Classification*

Gambar atau yang sering disebut dengan gambar merupakan citra dalam bidang Drimatika (dua dimensi). Melalui sudut pandang matematis, gambar adalah fungsi kontinu dari intensitas cahaya dalam bidang dua dimensi. Sumber cahaya menerangi objek, setelah itu objek membiaskan kembali sebagian cahaya. Pantulan cahaya inilah yang ditangkap oleh alat optik seperti mata manusia, kamera, *scanner*, dan lain-lain untuk merekam bayangan suatu benda, yang disebut dengan citra [4].

## 2.6 *Machine Learning*

*Machine learning* merupakan aplikasi komputer dan algoritma matematika yang belajar dari data dan membuat prediksi mengenai masa depan. Pembelajaran yang terlibat adalah upaya untuk memperoleh kecerdasan dalam dua fase, pelatihan dan pengujian [10].

Sederhananya, *machine learning* adalah algoritme yang dirancang untuk memungkinkan program komputer mempelajari instruksi dan melakukan tugas

tanpa campur tangan *user*. Algoritme memiliki cara kerja dengan membuat model dimulai dengan masukan sehingga dapat mengambil keputusan berdasarkan data yang digunakannya. [8].

Pada *maching learning* mempunyai 3 komponen paling penting, yaitu [8]:

### 1. *Supervised Learning*

Penggunaan data disertai dengan pengidentifikasi/kategori yang membeberitahukan di mana klasifikasi atau kelompok tempat data tersebut berada. Model yang dihasilkan merupakan model prediktif dari data yang diberikan pada kelas tersebut.

### 2. *Unsupervised Learning*

Data yang digunakan tidak memiliki pengenal/kategori, jadi Anda harus mencari struktur data yang digunakan dan mengelompokkannya sesuai dengan data yang diperoleh.

### 3. *Reinforced Learning*

Pelajari yang semestinya dilakukan dan bagaimana menerapkan situasi untuk memanfaatkannya sebaik mungkin. Peserta didik tidak mengetahui tindakan mana yang seharusnya dipilih, tetapi peserta didik diberi tahu tindakan mana yang akan menghasilkan hadiah terbesar saat diselesaikan.

## 2.7 *Convolutional Neural Network*

*Convolutional Neural Network* (CNN) ialah algoritme pembelajaran mendalam yang digunakan dalam peristiwa ini visi komputer seperti klasifikasi gambar atau video dan deteksi objek pada gambar atau bahkan area gambar. CNN adalah lapisan dengan kumpulan neuron 3D (lebar, tinggi, dan kedalaman). Lebar

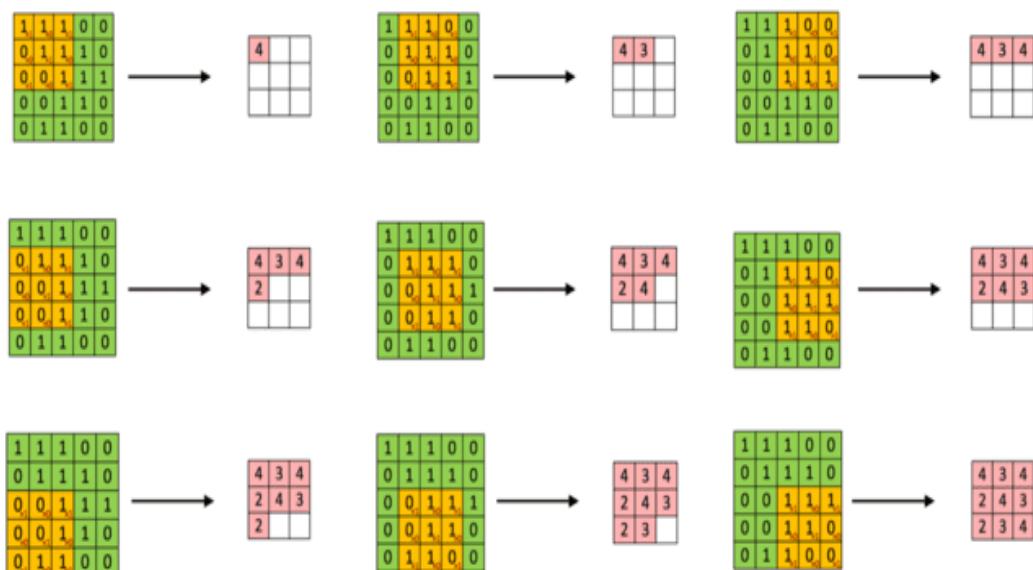
dan tinggi merupakan dimensi lapisan, sedangkan kedalaman mengacu pada jumlah lapisan [11]. Menurut [12] hasil akurasi CNN dikatakan baik apabila memiliki hasil di atas 50%. CNN termasuk kedalam golongan *supervised learning* karena dalam klasifikasi gambar dibutuhkannya sebuah pemberian label pada suatu citra [10].

*Convolutional Neural Network* (CNN) menggambarkan pengembangan lebih lanjut dari *Multilayer Perception* (MLP) yang diperuntukkan mengolah data dua dimensi. CNN juga merupakan algoritma yang masuk kedalam *Deep Neural Network* dikarenakan CNN memiliki kedalaman jaringan yang tinggi dan banyak digunakan dalam pencitraan. Kunihiko Fukushima, seorang peneliti di NHK Broadcasting Science Laboratories, mengembangkan CNN pertama yang disebut *NeoCognitron*. CNN salah satu algoritma dari *Deep Learning*, evolusi dari MLP yang diperuntukkan membuat sebuah data menjadi berbentuk *grid*, scontohnya adalah gambar dua dimensi seperti gambar atau suara. *Convolutional Neural Network* memiliki beberapa arsitektur, sebagai berikut [8]:

1. Convolution Layer

*Convolution layer* adalah sebuah operasi konvolusi yang terjadi di output dari lapisan sebelumnya. Layer merupakan sebuah langkah pertama yang mendasar dalam CNN. *Convolution layer* menjadi lapisan yang utama untuk digunakan. Konvolusi adalah sebuah maksud matematis yang dalam pengolahan gambar memiliki arti mengimplementasikan satu kernel (kotak berwarna kuning) pada gambar disemua *offset* yang memungkinkan seperti yang ditunjukkan oleh Gambar 2.1, sedangkan kotak yang memiliki warna hijau secara keseluruhan adalah gambar yang nantinya akan dibahas nanti.

Kernel atau kotak yang berwarna kuning bergerak dari pojok kiri atas ke pojok kanan bawah, menghasilkan konvolusi dari sebuah gambar muncul di sebelah kanan gambar. Konvolusi sendiri memiliki sebuah tujuan yaitu untuk mengekstraksi fitur dari gambar masukan. Konvolusi tersebut akan mengeluarkan hasil perubahan linier dari data inputan sesuai dengan informasi spasial dari data tersebut.

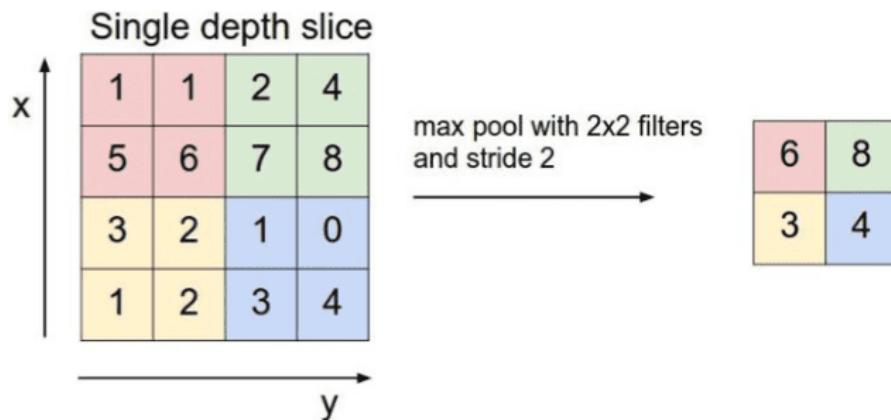


**Gambar 2.1 Convolution Layer**

## 2. Pooling Layer

*Pooling layer* adalah lapisan yang menggunakan fungsi *feature map* sebagai masukan dan mengerjakannya dengan berbagai macam operasi *statistic* berdasarkan nilai piksel yang paling dekat. *Pooling layer* yang dimasukkan di antara lapisan konvolusi secara berurutan dalam arsitektur model CNN dapat secara progresif mengurangi ukuran dari volume *output* pada *feature map*, yang menyebabkan sejumlah parameter dan perhitungan pada jaringan

berkurang, serta untuk mengendalikan *overfitting*. *Pooling layer* digunakan untuk mengambil nilai maksimal (*max-pooling*) atau nilai rata-rata (*average pooling*) dari bagian-bagian piksel dalam citra. Metode pooling yang paling sering digunakan pada CNN adalah metode *max-pooling*. *Max-pooling* membagi *output* dari *convolution layer* menjadi beberapa *grid* kecil kemudian akan mengambil nilai maksimal dari setiap *grid* untuk menyusun sebuah matriks citra yang sudah direduksi seperti pada Gambar 2.2.

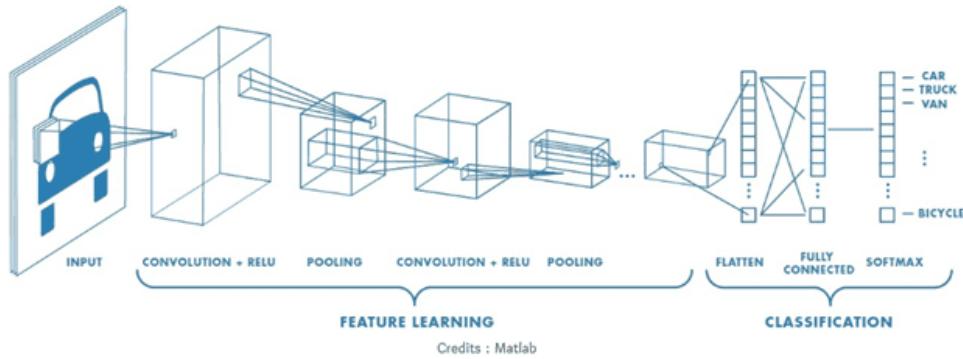


**Gambar 2.2 Pooling Layer**

### 3. Fully Connected Layer

*Fully-connected layer* adalah lapisan dimana semua neuron aktivasi dari lapisan sebelumnya yang terhubung dengan neuron di lapisan selanjutnya sama halnya dengan jaringan saraf tiruan biasa. Dalam setiap aktivasi dari lapisan sebelumnya harus diubah menjadi sebuah data satu dimensi sebelum dihubungkan pada semua neuron pada *fully-connected layer*. Lapisan ini biasa digunakan pada metode MLP (*Multilayer Perceptron*) yang bertujuan untuk mengolah data sehingga data tersebut dapat diklasifikasikan. Dapat

dilihat pada Gambar 2.3 tentang jaringan arsitektur pada *Convolutional Neural Network*.



**Gambar 2.3 Fully Connected Layer**

## 2.8 UML (*Unified Modeling Language*)

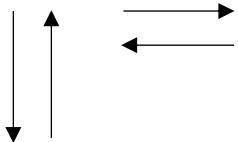
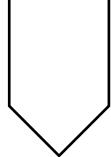
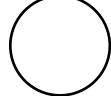
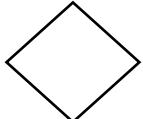
*Unified Modeling Language* (UML) merupakan bahasa spesifikasi standar yang gunakan untuk mendokumentasikan, menentukan dan membuat perangkat lunak. UML adalah metodologi untuk pengembangan sistem bertema objek dan sebagai alat untuk membantu dalam pengembangan sebuah sistem. *Unified Modeling Language* (UML) sendiri juga memfasilitasi penulisan *blueprint* yang standar, yang mencakup teori rancangan proses usaha, suatu pengelompokan kelas dalam bahasa pemrograman, skema *database*, dan komponen yang diperlukan dalam sebuah sistem *software* [13].

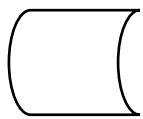
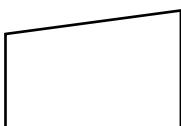
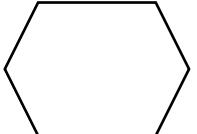
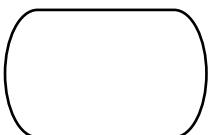
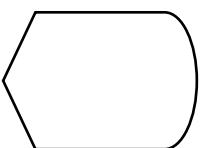
## 2.9 Flowchart

*Flowchart* adalah penggambaran grafis mulai langkah-langkah dan urutan kegiatan dalam sebuah program. Biasanya mengacu pada pemecahan masalah yang memerlukan penyelidikan dan evaluasi lebih lanjut. *Flowchart* dapat berfungsi untuk menunjukkan operasi manual, operasi pemrosesan, atau keduanya.

*Flowchart* itu sendiri merupakan sekumpulan simbol yang digunakan untuk membuatnya [14]. Simbol yang digunakan untuk menjelaskan *flowchart* dapat dilihat pada tabel 2.1.

**Tabel 2.1 Simbol dalam Flowchart**

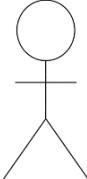
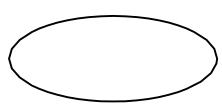
No	Lambang	Keterangan
1		<i>Flow Diagram Symbol</i> adalah simbol yang digunakan untuk menghubungkan satu simbol yang satu dengan simbol lainnya. Bisa juga disebut sebagai <i>connecting line</i> .
2		<i>Conncector Symbol</i> adalah ikon yang memiliki kegunaan masuk keluarannya atau menggabungkan proses di lembaran/halaman yang berbeda.
3		<i>Conncector Symbol</i> adalah ikon untuk masuk keluarannya atau menggabungkan proses di lembaran/halaman yang sama.
4		<i>Processing Symbol</i> adalah ikon yang memberitahukan pemrosesan yang sedang dikerjakan oleh komputer.
5		<i>Decission Symbol</i> adalah ikon untuk mengambil keputusan sebuah proses berdasarkan kondisi yang ada.

6		<i>Disk and On-line Storage Symbol</i> adalah ikon yang menandakan masukan yang berasal dari <i>disk</i> atau disimpan ke <i>disk</i> .
7		<i>Input – Output Symbol</i> adalah ikon yang menampilkan proses masukan dan keluaran terlepas dari tipe perangkatnya.
8		<i>Manual Input Symbol</i> adalah ikon berfungsi pemasukkan data secara manual <i>on-line keyboard</i> .
9		<i>Manual Operation Symbol</i> adalah ikon yang memperlihatkan pemrosesan yang tidak terlihat oleh komputer.
10		<i>Predefine proses symbol</i> adalah ikon yang berfungsi implementasi suatu bagian ( <i>subprogram</i> ) atau <i>prosедure</i> .
11		<i>Preparation symbol</i> adalah ikon yang digunakan untuk membuat ruang penyimpanan yang dipakai menjadi lokasi pemrosesan di <i>storage</i> .
12		<i>Terminator Symbol</i> adalah ikon untuk awal ( <i>start</i> ) dan akhir ( <i>stop</i> ) dari suatu eksekusi.
13		<i>Display symbol</i> adalah ikon yang menunjukkan barang <i>output</i> yang digunakan seperti monitor, plotter, printer, dll.

## 2.10 Use Case Diagram

*Use case diagram* merupakan diagram yang bekerja dengan menginterpretasikan hubungan tipikal antara pengguna sistem dan sistem itu sendiri dengan menjelaskan cara penggunaan sistem. Diagram *use case* terdiri dari aktor dan interaksinya. Aktor ini bisa berupa orang, perangkat, sistem lain atau mereka yang memiliki hubungan dalam menggunakan sistem. *Use case* juga dapat disebut sebagai dokumen naratif yang menggambarkan peristiwa atau insiden daripada menggunakan sistem untuk menyelesaikan suatu proses [15]. Simbol yang digunakan untuk menggambarkan *use case* dapat dilihat pada tabel 2.2.

**Tabel 2.2 Simbol dalam Use Case**

Simbol	Nama	Keterangan
	<i>Actor</i> (Aktor)	Menggambarkan pengguna atau sistem yang menjalankan atau menggunakan fungsi dari sistem yang dibangun. Perlu digaris bawahi bahwa aktor memiliki interaksi dengan <i>Use Case</i> , tetapi mereka tidak dapat memengaruhinya.
	<i>Use Case</i>	Menjelaskan fungsi yang disediakan sistem antara

		entitas dan aktor seperti entitas yang bertukar pesan, diekspresikan dengan kata kerja.
_____	<i>Association</i> (Asosiasi)	Jalur komunikasi antara <i>aktor</i> dengan <i>use case</i> .
«extend» ----->	<i>Extend</i> (Menjangkau)	Tambahkan perilaku ke <i>use case</i> dasar.
----->	<i>Generalization</i> (Generalisasi)	Menunjukkan sebuah hubungan antara <i>use case</i> yang bersifat umum dengan <i>use case</i> lain yang bersifat lebih spesifik.
«include» ----->	<i>Include</i> (Termasuk)	Menambahkan perbuatan ke suatu <i>use case</i> dasar yang menjelaskan penambahan.
	<i>System Boundary</i>	Menggambarkan batasan proses fungsionalitas yang ada dalam sistem dan bagian luar sistem.

## 2.11 *Activity Diagram*

*Activity diagram* atau diagram aktivitas merepresentasikan alur kerja atau fungsi dari suatu sistem atau suatu jalannya bisnis atau menu yang terdapat pada

sebuah perangkat lunak (*software*). Fokus *activity diagram* adalah penggambaran fungsi sistem atau fungsi yang dapat dijalankan sistem, bukan aktor [16]. Simbol yang digunakan untuk menggambarkan *activity diagram* dapat dilihat pada tabel 2.3.

**Tabel 2.3 Simbol dalam *Activity Diagram***

Simbol	Nama	Keterangan
	<i>Start Point</i>	Titik awal dimulainya aktivitas.
	<i>Activities</i>	Menggambarkan suatu proses / kegiatan bisnis.
	<i>Decision Points</i>	Menggambarkan pilihan untuk mengambil keputusan, benar atau salah.
	<i>Fork / Percabangan</i>	Digunakan untuk menunjukkan operasi perpecahan satu jalur menjadi dua/bercabang.
	<i>Join</i> (Penggabungan) atau <i>rake</i>	Digunakan untuk menunjukkan dari dua jalur menjadi satu jalur.
	<i>End Points</i>	Merupakan titik akhir aktivitas.
	<i>Swimlane</i>	Pembagian <i>activity diagram</i> berfungsi memperlihatkan apa yang

			sedang dikerjakan.
--	--	--	--------------------

## 2.12 Sequence Diagram

*Sequence diagram* atau sequens diagram mengilustrasikan interaksi antara gambar di dalam dan di sekitar sistem dalam hal pesan yang dijelaskan dari waktu ke waktu. Sebuah *sequence diagram* terdiri berdasarkan dimensi vertikal (waktu) & horizontal (objek terkait) [15]. Simbol yang digunakan untuk menggambarkan *sequence diagram* dapat dilihat pada tabel 2.4.

**Tabel 2.4 Simbol dalam Sequence Diagram**

Simbol	Nama	Keterangan
	<i>Entity Class</i>	Bagian dari sistem yang berisikan gabungan kelas dalam bentuk entitas. Entitas membuat gambaran awal dari sebuah sistem dan menjadi dasar pembuatan database.
	<i>Boundary Class</i>	Berisi kumpulan kelas-kelas yang menjadi tampilan utama atau korelasi antara satu bahkan lebih aktor dengan sistem, contohnya tampilan formular pendaftaran dan formulir cetak.
	<i>Control Class</i>	Objek yang berisi logika aplikasi yang tidak bertanggung jawab atas entitas, contohnya adalah perhitungan dan aturan bisnis yang berisi banyak objek.
	<i>Message</i>	Merupakan symbol yang

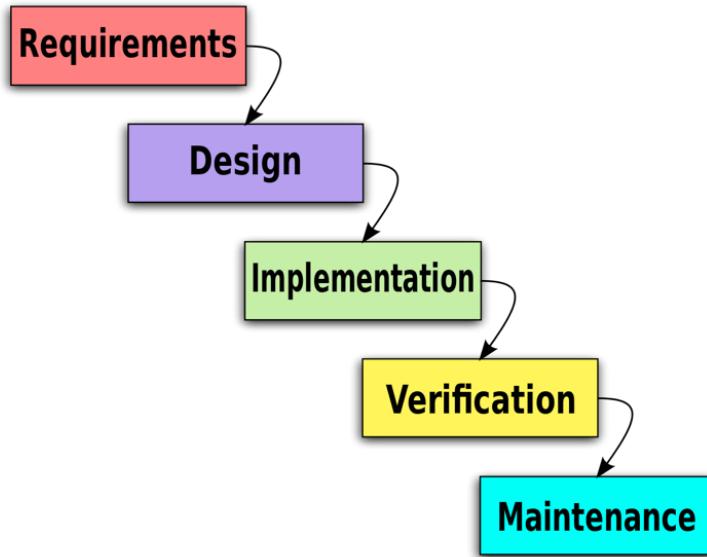
		menggambarkan pengiriman pesan antar <i>class</i> .
	<i>Activation</i>	Panjang kotak ini yang mewakili pelaksanaan operasi oleh objek bertolak belakang dengan lamanya aktivasi operasi.
	<i>Lifetime</i>	Serangkaian titik yang terhubung ke objek, aktivasi terjadi di sepanjang garis kehidupan.

### 2.13 *Blok Diagram*

*Blok Diagram* adalah ekspresi hubungan sekuensial dari satu atau lebih komponen yang memiliki persamaan kerjanya masing-masing, dan setiap blok komponen memengaruhi komponen lainnya. Diagram blok adalah salah suatu cara termudah untuk menyatakan cara kerja dari suatu sistem [17].

### 2.14 *System Development Life Cycle (SDLC)*

*System Development Life Cycle (SDLC)* adalah jenis format yang menggambarkan fase proses pengembangan sistem. SDLC sendiri ialah model klasik yang bersifat sistematis dan pada pembuatan perangkat lunak akan dibuat secara berurut. Tugas utama SDLC adalah memenuhi kebutuhan aktor dalam hal pengembangan sistem. Persyaratan pembaharuan sistem dapat berupa perubahan atau pembuatan aplikasi baru. Selain itu, pengembang dapat menggunakan SDLC untuk memperkirakan usia perangkat lunak yang dibuat atau digunakan [18]. Berikut tahapan-tahapan yang ada pada *System Development Life Cycle (SDLC)* dapat dilihat pada gambar 2.4:



**Gambar 2.4 System Development Life Cycle**

Penelitian tentang SDLC (*System Development Life Cycle*) ini dikembangkan dengan beberapa tahap sebagai berikut [19]:

1. *Requirements* (Analisis Kebutuhan Software)

Analisa dan pengumpulan kebutuhan sistem, berisikan tentang informasi yang akan diteliti dilakukan pada tahap ini.

2. *Design*

Pada tahap desain, dilakukan dengan cara merumuskan *flowchart* sesuai dengan apa yang akan dikembangkan.

3. *Implementation*

Tahap *implementation* berguna untuk mengimplementasikan sebuah desain menggunakan bahasa pemrograman ke dalam sebuah *system*. Perolehan dari bagian ini merupakan program komputer berjalan sama dengan desain yang telah dirancang.

4. *Verification*

Setelah tahap *implementation* akan dilakukan tahap *verification*, dimana bagian ini merupakan bagian pengujian terhadap program perangkat lunak yang telah dibuat. Pengujian sistem ini sangatlah penting untuk memberikan bukti bahwa kualitas dan fungsi dari program *software* ini beroperasi dengan benar.

### 5. *Maintenance*

Setelah perangkat lunak dapat dijalankan dengan baik, maka tahap akhir dari SDLC ini adalah melakukan *maintenance*. Pada tahap *maintenance* memiliki beberapa prosedur yang harus dilakukan, antara lain:

- a. *Corrective Maintenance*: mengoreksi atau revisi program yang dikembangkan karena adanya *error* yang terdeteksi dalam program.
- b. *Adaptive Maintenance*: penyesuaian penerapan program yang sudah beredar dengan target dalam program.
- c. *Perfective Maintenance*: Apabila program berhasil digunakan oleh pengguna dengan baik. *Maintenance* bertujuan untuk meningkatkan kemampuan dari program seperti menambahkan beberapa fungsi, meningkatkan kinerja dari program tersebut, dan lainnya.

### **2.15 Confusion Matrix**

*Confusion matrix* adalah sebuah metode yang digunakan untuk menilai suatu kinerja dari sebuah metode klasifikasi. Pada intinya, *confusion matrix* berisi informasi yang mengukur hasil klasifikasi yang dihasilkan sistem dengan hasil klasifikasi yang sebenarnya. Saat mengukur kinerja dengan matriks konfusi, ada 4 sebutan yang mewakili hasil dari proses pengklasifikasian. Istilah ini terdiri dari *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN).

Nilai *True Negative* (TN) merupakan total data negatif yang terdeteksi dengan benar, sedangkan nilai *False Positive* (FP) merupakan nilai yang negatif tetapi dia terdeteksi sebagai data yang positif [20].

Berikut rumus untuk menghitung *accuracy*, *precision*, dan *recall* [21]:

$$\text{Accuracy} = \frac{TP+TN}{Total}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

## 2.16 Penelitian Terdahulu

**Tabel 2.5 Penelitian Terdahulu**

No	Judul Penelitian	Persamaan Penelitian	Perbedaan Penelitian	Kesimpulan
1.	Convolutional Neural Network Untuk Kalasifikasi Penggunaan Masker	Menggunakan Algoritma yang sama yaitu CNN (Convolutional Neural Network)	Pada penelitian ini menggunakan algoritma CNN sebagai algoritma klasifikasi dalam penggunaan masker, sedangkan penulis	Pada penelitian ini memberikan tingkat akurasi yang cukup baik yaitu sebesar 73,3%.

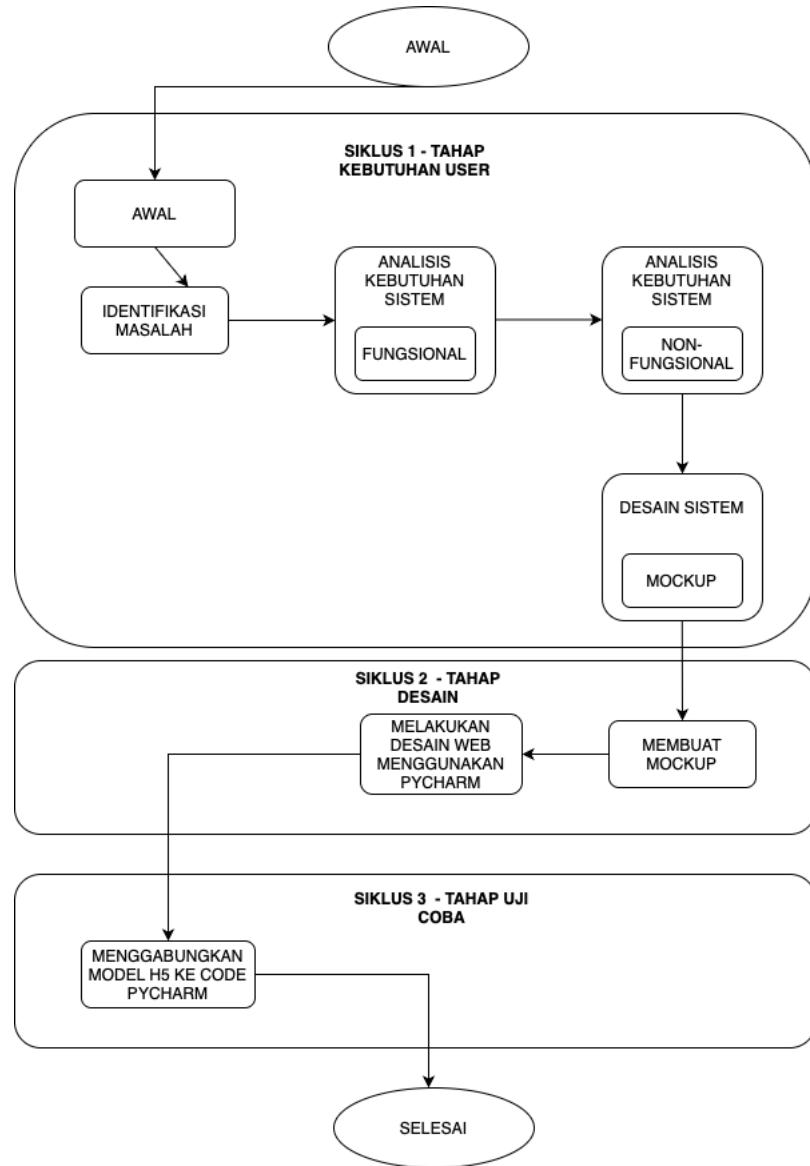
			menggunakan algoritma CNN sebagai klasifikasi jenis ikan hiu.	
2.	Implementasi Deep Learning Untuk Image Classification Menggunakan Algoritma Covolutional Neural Network (CNN) Pada Citra Kebun dan Sawah	Menggunakan algoritma yang sama yaitu CNN (Convolutional Neural Network)	Pada penelitian ini menggunakan citra kebun dan sawah, sedangkan penulis menggunakan citra jenis ikan hiu.	Pada penelitian ini 60 objek yang diuji, 44 diantara nya dapat terdeteksi sesuai dengan kategorinya.
3.	Klasifikasi Menggunakan Convolutional Neural Network (CNN) Pada Tangkapan Layar Halaman Instafram	Menggunakan algoritma yang sama yaitu CNN (Convolutional Neural Network)	Pada penelitian ini menggunakan algoritma CNN untuk digunakan pada tangkapan layar halaman instgram.	Pada penelitian ini memiliki tingkat akurasi 91%, precision 93%, dan recall 90%.
4.	Klasifikasi Citra Menggunakan Convolutional Neural Network dan K Fold Cross Validation	Menggunakan algoritma CNN (Convolutional Neural	Penelitian ini menggunakan algoritma yang digunakan untuk klasifikasi bunga	Sistem pada penelitian ini dapat memprediksi dengan rata-

		Network) sama seperti penelitian ini	mawar, tulip dan matahari.	rata prediksi tertinggi 50,31% dan nilai prediksi tertinggi yaitu sebesar 65,47%.
5.	Implementasi Metode CNN dalam Klasifikasi Gambar Jamur pada Analisis Image Processing	Menggunakan algoritma CNN (Convolutional Neural Network) sama seperti penelitian ini	Menggunakan algoritma CNN untuk mengklasifikasi gambar jamur.	Memperoleh hasil akurasi untuk mengklasifikasi gambar jamur menggunakan CNN sebesar 62%, dengan perbandingan data train 80% dan validation 20%.

## BAB 3

### METODE PENELITIAN

#### 3.1 Kerangka Berpikir



Gambar 3.1 Kerangka Berpikir

Pada gambar 3.1 di atas, dijelaskan bahwa pada tahap awal penulis mengidentifikasi masalah yang dibutuhkan oleh *user*, kemudian menganalisis kebutuhan sistem fungsional dan juga non-fungsional, setelah menganalisis

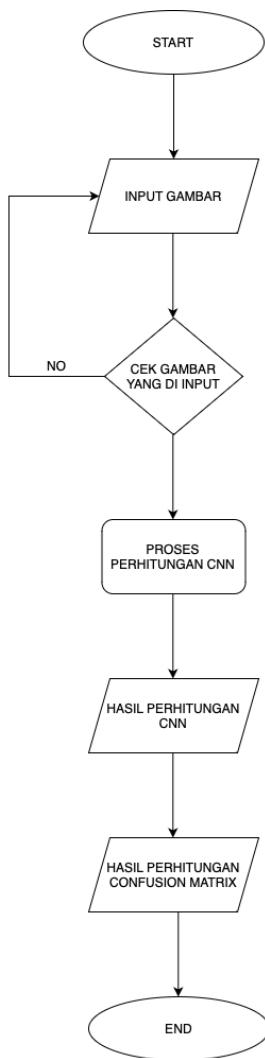
kebutuhan sistem fungsional dan juga non-fungsional, penulis lalu akan masuk ke mockup desain sistem. Setelah membuat mockup desain, akan dilanjutkan dengan membuat desain tersebut, desain dari mockup tersebut akan diimplementasikan menggunakan aplikasi yang bernama Phycarm. Setelah selesai melakukan desain terhadap website tersebut, penulis akan memasukan model h5 ke dalam codingan Pycharm, yang berguna untuk menampilkan hasil dari CNN di website yang dibuat oleh penulis.

### **3.2 Pemilihan Algoritma**

Pada penelitian ini digunakan Algoritme *Convolutional Neural Network* (CNN) yang digunakan penulis saat dilakukan klasifikasi terhadap gambar atau sebuah citra.

### **3.3 Perancangan Proses**

Perancangan proses yang berfungsi untuk menunjukkan alur dari program yang dibuat oleh penulis dapat dilihat pada gambar 3.2.



**Gambar 3.2 Flowchart Perancangan Proses**

Pada gambar 3.2 dijelaskan tentang alur dari aplikasi yang akan dibuat oleh penulis. Saat mengakses aplikasi ini, user harus memasukan atau menginput sebuah gambar sebelum dilakukannya perhitungan. Setelah *user* memasukan gambar maka sistem akan melihat format dari gambar yang sudah di *input* oleh *user*, apakah format tersebut sesuai dengan kriteria format gambar atau tidak, jika ya maka sistem akan melanjutkan ke proses perhitungan menggunakan algoritma CNN, jika format gambar tidak sesuai dengan kriteria format gambar maka sistem akan menyuruh user untuk memasukan kembali gambar dengan format gambar yang benar. Setelah

melakukan cek gambar dan melakukan proses perhitungan dengan algoritma CNN maka sistem akan memberitahu hasil akhir dari perhitungan tersebut dan *user* dapat melihat hasil dari perhitungan tersebut.

### **3.3.1 Pengumpulan Data**

Data yang digunakan merupakan sebuah gambar yang diunduh dari sebuah website yaitu Kaggle atau dapat diakses melalui link kaggle.com. Kaggle sendiri merupakan tempat pencarian dataset yang akan digunakan oleh penulis dengan komposisi 70% untuk data latih, 20% untuk data validasi, dan 10% untuk data test. Dataset yang diperoleh dari Kaggle digunakan oleh penulis karena merupakan dataset yang dapat digunakan secara bebas.

### **3.3.2 Pre-processing**

Tahap ini dilakukan untuk memberikan sebuah label pada citra yang sudah disiapkan. Setelah citra diberikan label maka akan dilakukan *resizing* gambar agar semua citra yang ada di dalam dataset memiliki ukuran yang sama, dan akan menetapkan banyaknya label kategori yang ada pada dataset ini. Kategori label pada citra jenis ikan hiu dapat dilihat pada gambar 3.3.

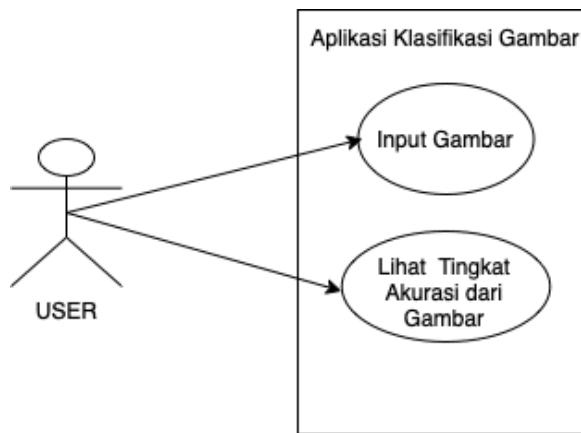
```
[ 'Lainnya', 'Hiu Martil', 'Hiu Paus', 'Hiu Putih', 'Hiu Sirip Putih', 'Hiu Tikus' ]
```

**Gambar 3.3 Pre-processing**

## **3.4 Perancangan Sistem**

### **3.4.1 Use Case Diagram**

*Use Case Diagram* aplikasi klasifikasi gambar menunjukkan hal-hal yang pengguna dapat lakukan dalam aplikasi ini. *Use Case Diagram* pada sistem ini dapat dilihat pada gambar 3.4.



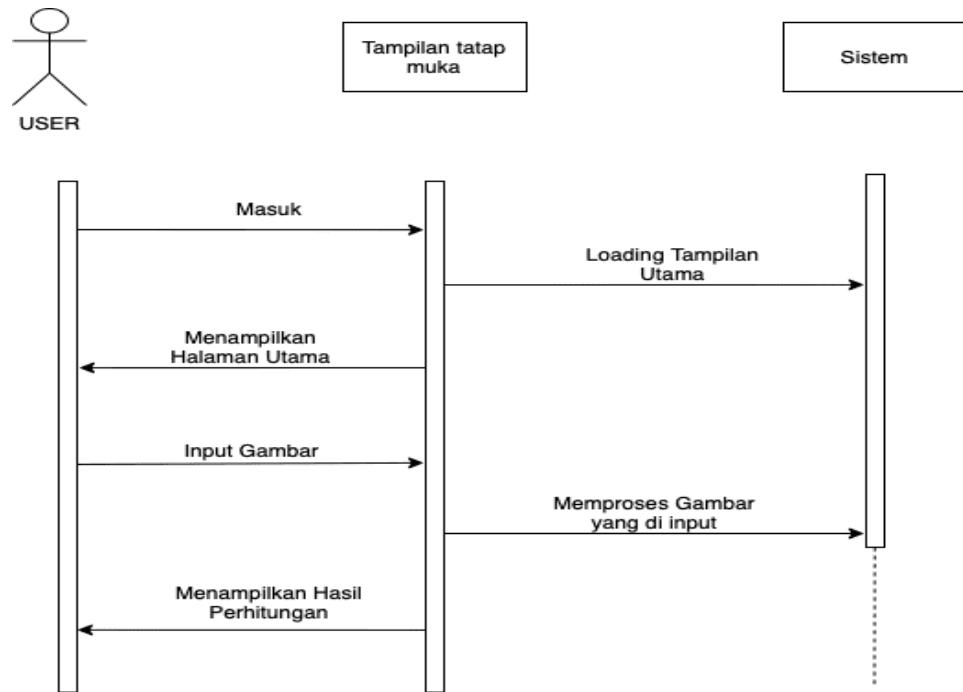
**Gambar 3.4 Use Case Diagram**

Pada gambar 3.4 menggambarkan interaksi antara seorang *user* & aplikasi yang dapat dijelaskan sebagai berikut:

1. Setelah *user* mengakses *website* ini, maka hal yang harus dilakukan pertama kali oleh *user* yaitu memasukkan gambar dengan format yang sudah ditentukan yaitu dengan format ekstensi gambar .jpg, .jpeg, jika *user* tidak memasukkan gambar dengan format ekstensi gambar yang telah ditentukan maka *user* akan diminta untuk memasukan kembali gambar dengan format yang sesuai.
2. Kemudian, *user* akan mengsubmit gambar yang sudah di *input* di awal saat mengakses *website*.
3. *User* atau pengguna dapat melihat gambar yang sudah di *input* beserta nilai akurasi dari gambar tersebut yang telah melewati perhitungan dari algoritma CNN.

### 3.4.2 Sequence Diagram

*Sequence diagram* aplikasi klasifikasi gambar menjelaskan tentang interaksi yang dilakukan oleh seorang *user* dengan sistem dan dapat dilihat pada gambar 3.5.

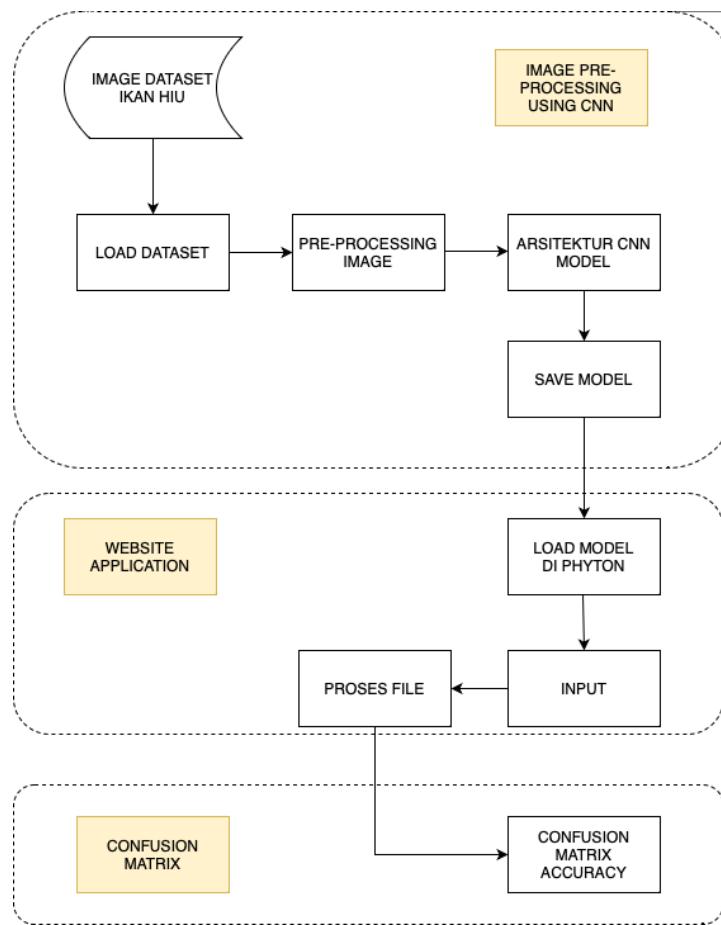


**Gambar 3.5 Sequence Diagram**

*Sequence diagram* pada gambar 3.5 menampilkan proses aplikasi yang dilakukan oleh *user* dimulai dari masuk ke halaman utama dari *program*. Pada halaman utama akan muncul *button* untuk meng gambar sesuai dengan ekstensi yang sudah ditentukan. Setelah *user* melakukan *input* gambar sesuai dengan ekstensi gambar yang sudah ditentukan, sistem akan menampilkan hasil perhitungan akurasi terhadap gambar yang sudah di *input* sebelumnya.

### 3.4.3 Blok Diagram

Blok diagram berfungsi untuk memperoleh informasi inti dari perancangan sebuah sistem aplikasi yang akan dibuat oleh penulis. Blok diagram tersebut dapat dilihat pada gambar 3.6.



**Gambar 3.6 Blok Diagram**

Pada gambar 3.6 menjelaskan tentang dari awal hingga akhir pembuatan sistem aplikasi yang mencakup Dataset, algoritma CNN, sistem aplikasi itu sendiri dan pengujian akurasi yang menggunakan *Confusion Matrix*.

### 3.5 Perancangan Tampilan

#### 3.5.1 Tampilan Masukan (Input)

Tampilan masukan merupakan tampilan awal saat program dijalankan, pada halaman ini terdapat *button* untuk melakukan ungguh *file* dengan ekstensi PNG atau JPG saja yang dapat diunggah, tampilan awal dapat dilihat pada gambar 3.7.



Apakah Anda Melihat Hiu?

**Gambar 3.7 Halaman Awal**

#### 3.5.2 Tampilan Keluaran (Output)

Tampilan keluaran merupakan halaman akhir dari program yang dijalankan. Pada halaman ini terdapat hasil prediksi dan akurasi dari gambar yang sudah di input sebelumnya di awal halaman. Halaman akhir dapat dilihat pada gambar 3.8.

Uploaded Image

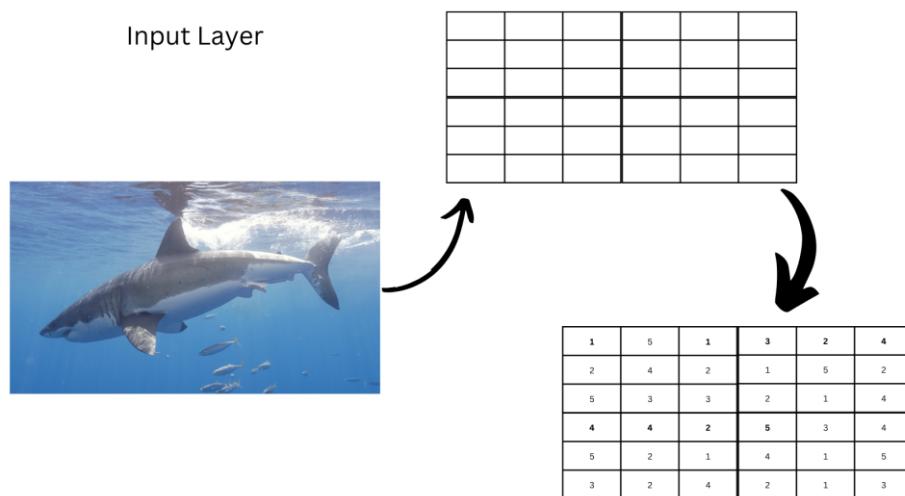


Nama Label dan  
Tingkat Akurasi

**Gambar 3.8 Halaman Akhir**

### 3.6 Input Layer

*Input Layer* merupakan *layer* pertama dalam sebuah arsitektur CNN, pada *layer* ini pada akhirnya semua citra akan di-*input* dan akan diproses ke *layer* berikutnya. Dengan mata manusia maka akan terlihat sebuah gambar, berbeda dengan komputer, jika komputer melihat sebuah gambar maka komputer akan merepresentasikan gambar tersebut sebagai sekumpulan *pixel*, yang nantinya *input layer* ini akan memiliki *image size* sebesar  $100 \times 100$  *pixel*. Pada simulasi ini, *input layer* akan memiliki *size* sebesar  $6 \times 6$ , sehingga nantinya seluruh gambar (citra) yang masuk ke dalam arsitektur untuk dilakukan proses *size*-nya akan di *convert* menjadi ukuran  $6 \times 6$  seperti yang ditunjukkan pada gambar 3.9, untuk semua *size* harus dibuat menjadi sama sehingga tidak ada data yang memiliki *pixel* di luar nilai *pixel* yang telah ditentukan. Jika terdapat perbedaan ukuran nilai *pixel* maka nilai *pixel* gambar 1 dengan gambar yang lainnya tidak dapat dibandingkan.



**Gambar 3.9 Input Layer**

### 3.7 Convolution Layer

Menggunakan *convolution layer* dapat membuat citra dapat di ekstrak fiturnya untuk diambil pola bentuk dan warnanya, dengan begitu komputer dapat membedakan antara ikan hiu Martil, Paus, Putih, Sirip Putih, dan Tikus dengan melihat bentuk tubuh dan warna dari setiap jenis ikan dari *layer* sebelumnya. Penggunaan *filter size* yaitu  $3 \times 3$ , digunakan *filter size*  $3 \times 3$  dikarenakan *input layer* tidak lebih dari  $200 \times 200$  pixel. Untuk jumlah *convolution layer* terdiri dari 3 buah, karena jika menggunakan di bawah/di atas 3 *layer* akan mengurangi nilai akurasinya, untuk jumlah *filter* baru dengan nilai 18, 32, 64. Pemilihan besarnya nilai *filter* baru didasari dengan pengembangan dari jurnal terdahulu, yang dimana jika menggunakan *layer* sesuai dengan *layer* pada jurnal terdahulu akan membuat nilai akurasi menjadi kurang baik, serta penggunaan *padding same* dilakukan agar mengurangi data yang terbuang. Angka *filter* tidak ditentukan sebelumnya, karena tidak ditentukan di awal sehingga isi dari semua *pixel* pada *filter* otomatis sesuai dengan *fitur* apa yang nantinya akan di ekstrak pada *fully connected layer*, seperti deteksi tepi, dan lain-lain. Contoh simulasi perhitungan *convolution* dengan *layer* inputan dan *filter* yang berukuran  $3 \times 3$  sebelumnya karena *padding same* maka inputan *image* akan menjadi  $8 \times 8$  dengan *padding 0* pada akhirnya. Serta *image* berukuran  $6 \times 6$  menjadi  $8 \times 8$  lalu dikali dengan *filter* baru. Dapat dilihat pada gambar 3.10.

0	0	0	0	0	0	0	0
0	5	3	6	9	5	1	0
0	6	1	1	7	2	6	0
0	0	2	3	4	9	2	0
0	8	5	4	0	3	4	0
0	9	7	8	2	1	3	0
0	3	4	9	8	7	2	0
0	0	0	0	0	0	0	0

X

0	1	0
1	1	-1
1	-1	-1

0	3	0	0	5	0
6	1	-1	7	2	-6
0	-2	-3	4	-9	-2
0	5	0	0	3	0
9	7	-8	2	1	-3
3	-4	-9	8	-7	-2

**Gambar 3.10 Convolution Layer**

### 3.8 Activation Function

Fungsi dari *activation* salah satunya adalah untuk membuat semua nilai yang memiliki nilai *negative* menjadi nilai *positive* dengan menggunakan aktifasi *Relu*, sehingga nantinya semua nilai yang bernilai *negative* akan dibuat 0, karena angka *negative* hanya akan memperlambat komputasi komputer dalam melakukan *train* sebuah model dikarenakan nilai *filter* yang *negative* tidak dapat diekstrak nilainya. Dapat dilihat pada gambar 3.11.

0	3	0	0	5	0
6	1	0	7	2	0
0	0	0	4	0	0
0	5	0	0	3	0
9	7	0	2	1	0
3	0	0	8	0	0

**Gambar 3.11 Activation Function**

### 3.9 Pooling Layer

*Pooling layer* memiliki fungsi untuk mengurangi komputasi komputer, sehingga hanya mengambil nilai *pixel* terbesar dari setiap *pool size*, karena nilai yang lebih kecil kurang berguna untuk mengekstrak fitur-fiturnya, yang menyebabkan nantinya hanya akan menambah komputasi komputer. Memiliki *pool size* sebesar  $2 \times 2$  dikarenakan untuk lebih sedikit data yang terbuang. Dapat dilihat pada gambar 3.12.

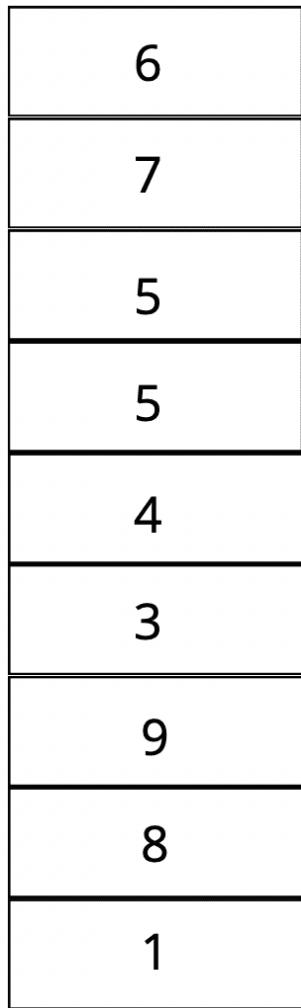
0	3	0	0	5	0
6	1	0	7	2	0
0	0	0	4	0	0
0	5	0	0	3	0
9	7	0	2	1	0
3	0	0	8	0	0

6	7	5
5	4	3
9	8	1

**Gambar 3.12 Simulasi Pooling Layer**

### 3.10 *Flatten Layer*

*Flatten layer* berfungsi menjadi inputan dari *layer* berikutnya untuk dikumpulkan dan diekstrak fiturnya menjadi satu bagian, memiliki bentuk 1 dimensi karena seluruh nilai yang ada pada *flatten* akan masuk kedalam *fully connected layer* yang memiliki bentuk yang sama yaitu 1 dimensi, contohnya seperti *pixel* pada *layer* sebelumnya, maka dari *pixel*  $4 \times 4$  akan dijadikan menjadi 1 dimensi, yang akan dilakukan dari kiri atas ke bawah kanan. Dapat dilihat simulasi pada gambar 3.13.

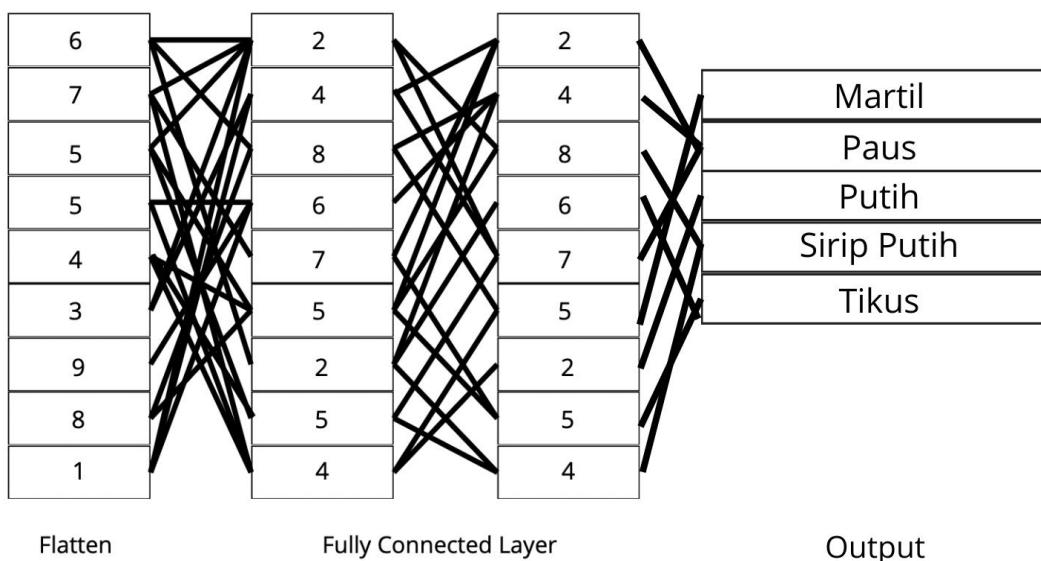


Gambar 3.13 Simulasi *Flatten Layer*

### 3.11 *Fully Connected Layer*

*Fully connected layer* berisi kumpulan dari seluruh fitur sebuah gambar dimulai dari *edge detection*, pola bentuk dari setiap *pixel*, warna, dan lain-lain, serta setiap *layer* yang ada pada *fully connected layer* akan saling berhubungan dengan yang lainnya sehingga membentuk suatu *neuron*, yang membuat output-nya menghasilkan sebuah kelas ikan hiu Martil, Paus, Putih, Sirip Putih, dan Tikus. Dapat dilihat simulasi *fully connected layer* pada gambar 3.14.

Pada bagian kiri berisikan *flatten* yang merupakan sebuah *input-an* dari *layer-layer* sebelumnya, pada *fully connected layer* berisi ekstrak dari sebuah gambar yang telah dipelajari sebelumnya, yang menyebabkan *flatten layer*, *fully connected layer*, dan *output layer* akan saling terhubung dan komputer dapat memgeneralisasikan sebuah kelas yang didapatkan dari keterhubungan antar *layer* tersebut.



**Gambar 3.14 Simulasi Fully Connected Layer**

### 3.12 Perencanaan Pengujian

		True Class				
		Martil	Paus	Putih	Sirip Putih	Tikus
Predicted Class	Martil	8	0	0	1	1
	Paus	1	8	1	0	0
	Putih	0	0	8	1	1
	Sirip Putih	1	0	1	8	0
	Tikus	1	1	0	0	8

**Gambar 3.15 Perencanaan Pengujian**

Pada gambar 3.15 merupakan hasil pengujian dari jenis-jenis ikan hiu, dimana pada gambar tersebut terdapat 10 gambar yang tidak sesuai dengan yang diprediksi yaitu ikan hiu jenis Martil 2 gambar, Paus 2 gambar, Putih 2 gambar, Sirip Putih 2 gambar dan Tikus 2 gambar. Dengan 10 gambar yang tidak sesuai dengan yang diprediksi maka tingkat akurasi dari program yang dibuat oleh penulis adalah sebagai berikut:

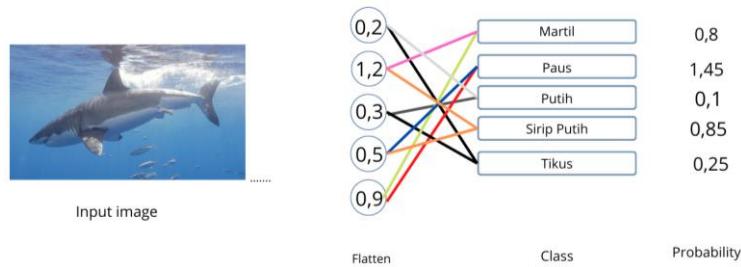
$$\text{Akurasi} = \frac{TP}{Total} \times 100\%$$

$$\text{Akurasi} = \frac{37}{60} = 61,6\%$$

### 3.14 Simulasi Perhitungan CNN

Melakukan *Input image* untuk melakukan perhitungan akurasi CNN, kemudian *image* tersebut akan melewati beberapa tahap arsitektur model CNN seperti *convolutioin layer*, *pooling layer*, dan lain-lain. Untuk nilai *flatten layer*

berasal dari nilai terakhir dari *pooling layer* yang *kernel size*-nya dibuat menjadi 1 dimensi. Kemudian untuk mendapatkan nilai *probability* akan dilakukan perhitungan rata-rata dari jumlah *pixel* yang terhubung dengan kelas yang ada. Dapat dilihat pada gambar 3.16 terdapat sebuah *input-an* gambar yang telah melewati beberapa tahap yaitu *convolution layer* dan *pooling layer*, kemudian hasil setelah melewati dua tahap tersebut akan masuk ke dalam *flatten layer*, dan *flatten layer* ini akan membuat sebuah jaringan-jaringan yang nantinya dapat menentukan kelas kemiripan. Pada gambar 3.16, terdapat contoh dengan *pixel* 0,2 kemudian akan dilihat kemiripan atau kecocokan dengan kelas Martil dan kelas lainnya, sehingga pada saat semua *flatten layer* terhubung dengan sebuah kelas maka nilai *probability* dapat disimulasikan dengan menghitung rata-rata terbesar pada setiap kelasnya, untuk pada kasus ini nilai terbesar berada pada kelas Paus, sehingga pada akhirnya AI akan menampilkan kelas Paus dengan *probability* sebesar 1,45.



**Gambar 3.16 Simulasi Perhitungan CNN**

### 3.15 Percobaan Layer pada CNN

```
from keras.models import Sequential
from keras.layers import Dense, Activation, Flatten, Dropout
from keras.layers import Conv2D, MaxPooling2D

model=Sequential()

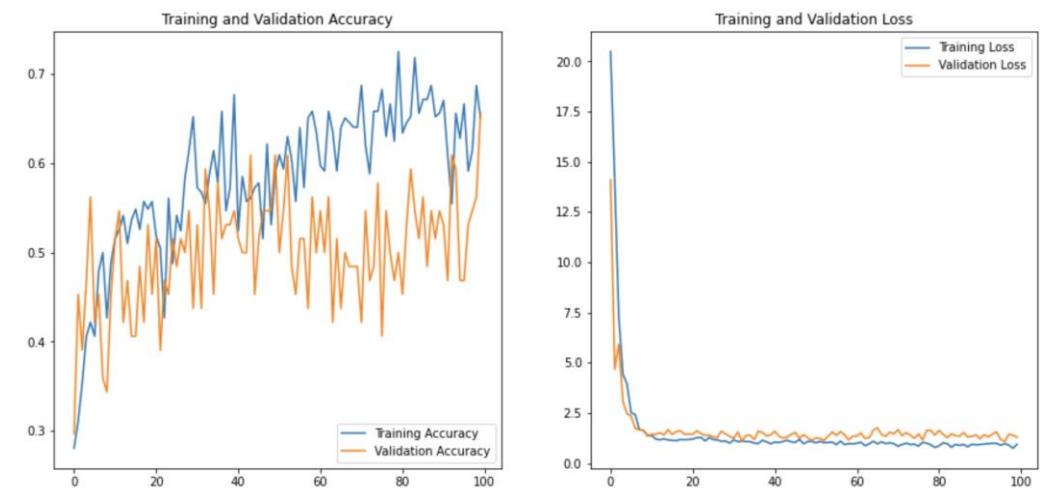
model.add(Conv2D(filters=32, kernel_size=(3,3), input_shape=(100, 100, 3), padding='SAME'))
model.add(MaxPooling2D(pool_size=(2,2), padding='SAME'))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(6, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

**Gambar 3.17 Percobaan 1 Layer**

Pada gambar 3.17, penulis menggunakan layer 1 untuk mencari nilai akurasi yang terbaik untuk sebuah model yang akan digunakan. Pada pada 1 layer ini digunakan filter sebesar 32, dan menghasilkan diagram yang dapat dilihat pada gambar 3.18.



**Gambar 3.18 Hasil Diagram 1 Layer**

Hasil dari diagram 1 layer yang dapat dilihat pada gambar 3.18 memiliki validation akurasi yang sangat tidak baik, yang membuat tingkat akurasi dapat berkurang.

```
from keras.models import Sequential
from keras.layers import Dense, Activation, Flatten, Dropout
from keras.layers import Conv2D, MaxPooling2D

model=Sequential()

model.add(Conv2D(filters=32, kernel_size=(3,3), input_shape=(100, 100, 3), padding='SAME'))
model.add(MaxPooling2D(pool_size=(2,2), padding='SAME'))

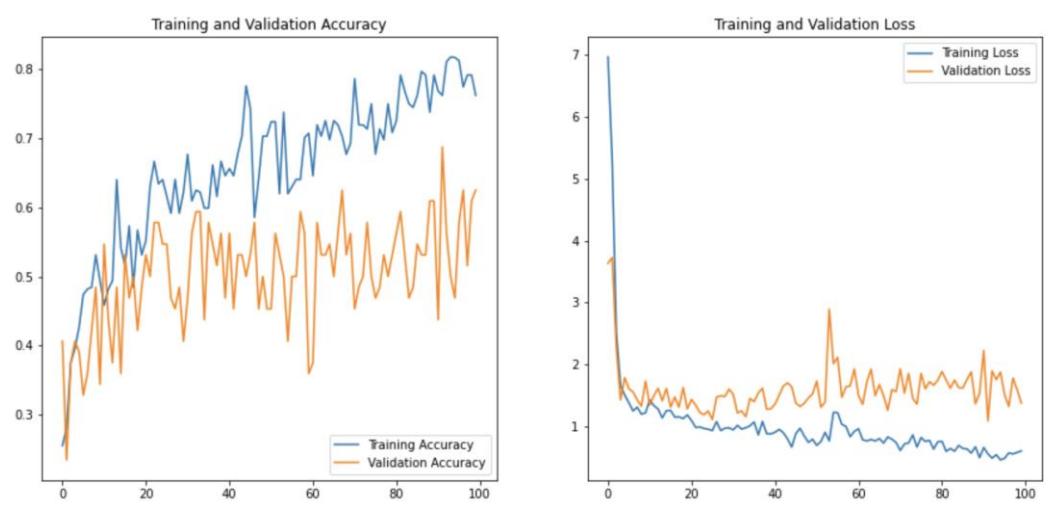
model.add(Conv2D(filters=64, kernel_size=(3,3), padding='SAME'))
model.add(MaxPooling2D(pool_size=(2,2), padding='SAME'))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(6, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

**Gambar 3.19 Percobaan 2 Layer**

Percobaan menggunakan 2 layer dapat dilihat pada gambar 3.19, dimana filter yang digunakan untuk kedua layer adalah 32 dan 64. Hasil dari penggunaan 2 layer dapat dilihat pada gambar 3.20.



**Gambar 3.20 Hasil Diagram 2 Layer**

Pada gambar 3.20, dapat dilihat bahwa menghasilkan akurasi yang tinggi, tetapi memiliki validasi yang rendah, pada tabel loss nilai validasi juga memiliki nilai yang tinggi.

### **3.15 Jadwal Pengerjaan**

### Tabel 3.1 Jadwal Pengerjaan

## **BAB 4**

### **IMPLEMENTASI**

#### **4.1 Kebutuhan Perangkat**

Dalam mendukung pembuatan aplikasi yang dilakukan oleh penulis dibutuhkan sebuah perangkat yang dapat memudahkan penulis dalam pengembangan aplikasi yang akan dikembangkan, diantara nya kebutuhan perangkat keras (*Hardware*) dan kebutuhan perangkat lunak (*Software*).

##### **4.1.1 Perangkat Keras (*Hardware*)**

Dapat dilihat pada Tabel 4.1 yang merupakan spesifikasi perangkat keras yang dibutuhkan untuk pembuatan aplikasi.

**Tabel 4.1 *Hardware***

No	Spesifikasi
1.	Processor <i>Intel® Core™ i5-4200 CPU</i>
2.	RAM 4GB
3.	SSD 500GB
4.	Mouse dan Keyboard

##### **4.1.2 Perangkat Lunak (*Software*)**

Dapat dilihat pada Tabel 4.2 yang merupakan spesifikasi perangkat lunak yang dibutuhkan untuk pembuatan aplikasi.

**Tabel 4.2 Software**

No	Spesifikasi
1.	Sistem Operasi Windows 10
2.	Pycharm
3	Google Chrome

## 4.2 Implementasi Antarmuka Pengguna

Implementasi antarmuka dari aplikasi klasifikasi jenis ikan hiu menggunakan metode *convolutional neural network* sebagai berikut.

### 4.2.1 Tampilan Halaman Utama

**Gambar 4.1 Halaman Utama**

Pada gambar 4.1, merupakan halaman *index* atau halaman utama ketika website dijalankan untuk yang pertama kali. Pada halaman ini terdapat dua buah *button*, dimana *user* akan diminta untuk memasukan sebuah gambar dengan format

yang sudah ditentukan lalu gambar yang sudah di masukkan akan di eksekusi dan *user* akan dibawa ke tampilan berikutnya.

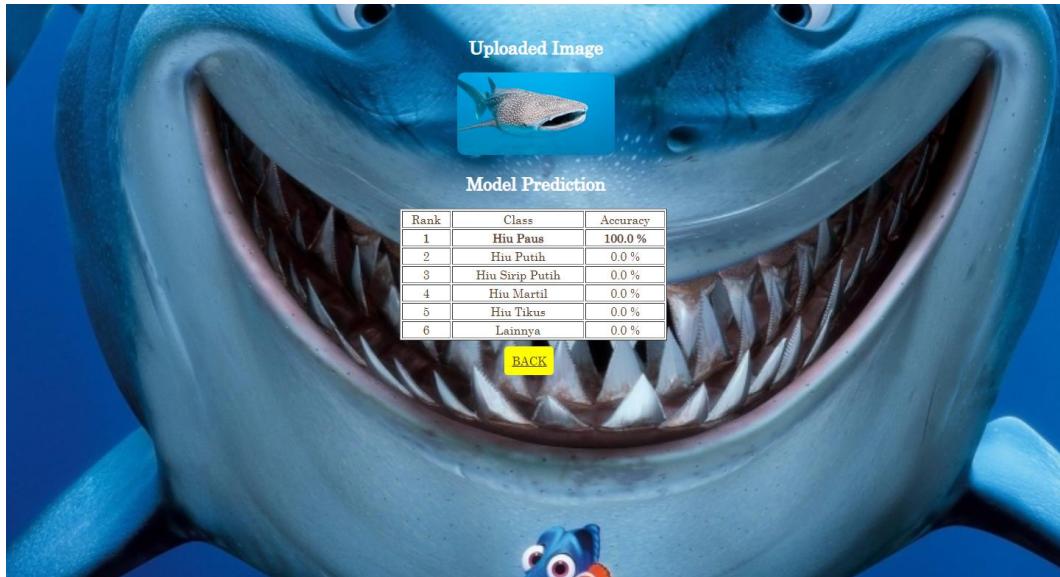
#### 4.2.2 Tampilan Peringatan pada Halaman Utama



**Gambar 4.2 Tampilan Peringatan**

Apabila gambar yang di *input* atau di masukkan oleh *user* tidak sesuai dengan yang sudah ditentukan, maka program akan mengeluarkan sebuah peringatan dan mengatakan pesan yang dapat dilihat pada gambar 4.2. Dengan pesan tersebut *user* diminta untuk memasukkan gambar dengan format gambar .jpg, .png, atau .jpeg.

#### 4.2.3 Tampilan Hasil Klasifikasi



**Gambar 4.3 Hasil Klasifikasi**

Pada gambar 4.3, merupakan tampilan hasil klasifikasi dari gambar yang sudah di *input* atau di masukkan oleh *user*. Gambar yang di *input* merupakan ikan hiu jenis Paus dan setelah di eksekusi jenis ikan hiu tersebut memiliki angka 100%.

#### 4.3 Implementasi Metode atau Algoritma

Pada penelitian ini penulis menggunakan Algoritma *Convolutional Neural Network* (CNN) untuk dilakukan klasifikasi terhadap suatu gambar jenis-jenis ikan hiu. Terdapat 3 tahapan yaitu *preprocessing image*, arsiteksur *Convolutional Neural Network*, dan membagi Dataset.

### 4.3.1 Preprocessing Image

```
[1] from tensorflow.keras.preprocessing.image import ImageDataGenerator

latih_datagen = ImageDataGenerator(
    rescale= 1./255,
    horizontal_flip=True,
    rotation_range=35,
    shear_range = 20.0,
    fill_mode = 'nearest')

validasi_datagen = ImageDataGenerator(
    rescale= 1./255,
    vertical_flip=True,
    rotation_range=35,
    shear_range = 20.0,
    fill_mode = 'nearest')

[2] latih_generator = latih_datagen.flow_from_directory(
    latih_dir, # direktori data latih
    target_size=(100, 100), # mengubah resolusi seluruh gambar menjadi 100x100 piksel
    class_mode='categorical')

validasi_generator = validasi_datagen.flow_from_directory(
    validasi_dir, # direktori data validasi
    target_size=(100, 100), # mengubah resolusi seluruh gambar menjadi 100x100 piksel
    class_mode='categorical')
```

**Gambar 4.4 Preprocessing Image**

Pada gambar 4.4 dijelaskan bahwa program akan melihat ada berapa banyak kelas yang terdapat dalam folder Data\_Latih dan Data\_Validasi. Kelas yang digunakan sebanyak 6 kelas, diantaranya Hiu Martil, Hiu Paus, Hiu Putih, Hiu Sirip Putih, Hiu Tikus, Lainnya. Setelah mendapatkan banyaknya gambar yang terdapat disetiap kelasnya, program akan melakukan *rescale* terhadap gambar tersebut menjadi 100x100.

### 4.3.2 Arsitektur *Convolutional Neural Network*

```
[1] from keras.models import Sequential
[2] from keras.layers import Dense, Activation, Flatten, Dropout
[3] from keras.layers import Conv2D, MaxPooling2D

[4] model=Sequential()

[5] model.add(Conv2D(filters=32, kernel_size=(3,3), input_shape=(100, 100, 3), padding='SAME'))
[6] model.add(MaxPooling2D(pool_size=(2,2), padding='SAME'))

[7] model.add(Conv2D(filters=64, kernel_size=(3,3), padding='SAME'))
[8] model.add(MaxPooling2D(pool_size=(2,2), padding='SAME'))

[9] model.add(Conv2D(filters=128, kernel_size=(3,3), padding='SAME'))
[10] model.add(MaxPooling2D(pool_size=(2,2), padding='SAME'))

[11] model.add(Flatten())
[12] model.add(Dense(512,activation='relu'))
[13] model.add(Dropout(0.5))
[14] model.add(Dense(6,activation='softmax'))

[15] model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

**Gambar 4.5 Arsitektur CNN**

Pada gambar 4.5, dijelaskan bahwa terdapat 4 layer dalam pengolahan gambar dimana penggunaan filter pada disetiap layer memiliki angka yang berbeda sesuai yang dibutuhkan saat ingin melakukan pelatihan model. Setelah model yang dibuat telah dibangun lalu akan menjalankan perintah `model.summary()` yang berfungsi untuk melihat rangkuman dari model yang telah dibangun dan akan menampilkan *layer-layer* yang sudah dibuat. Hasil dari `model.summary()` dapat dilihat pada gambar 4.6.

```

Model: "sequential_2"
-----  

Layer (type)          Output Shape       Param #
-----  

conv2d_6 (Conv2D)      (None, 100, 100, 32)    896  

max_pooling2d_6 (MaxPooling 2D)        (None, 50, 50, 32)    0  

conv2d_7 (Conv2D)      (None, 50, 50, 64)     18496  

max_pooling2d_7 (MaxPooling 2D)        (None, 25, 25, 64)    0  

conv2d_8 (Conv2D)      (None, 25, 25, 128)    73856  

max_pooling2d_8 (MaxPooling 2D)        (None, 13, 13, 128)    0  

conv2d_9 (Conv2D)      (None, 13, 13, 128)    147584  

max_pooling2d_9 (MaxPooling 2D)        (None, 7, 7, 128)     0  

flatten_2 (Flatten)    (None, 6272)           0  

dense_4 (Dense)        (None, 512)            3211776  

dropout_2 (Dropout)    (None, 512)            0  

dense_5 (Dense)        (None, 6)              3078
-----  

Total params: 3,455,686  

Trainable params: 3,455,686  

Non-trainable params: 0

```

**Gambar 4.6 Hasil Model Summary**

#### 4.3.3 Membagi Dataset

```

history=model.fit(latih_generator,
                  steps_per_epoch=6,
                  epochs=100,
                  validation_data=validasi_generator, # menampilkan akurasi pengujian data validasi
                  validation_steps=2,
                  verbose=2)

```

**Gambar 4.7 Membagi Dataset**

Pada gambar 4.7 dijelaskan bahwa terdapat 600 gambar akan dibagi setiap *batch*-nya menjadi 6 batch. Epoch yang digunakan berada di angka 100 yang akan

digunakan untuk mengolah semua dataset dalam penelitian ini. Hasil pembagian dataset dapat dilihat pada tabel 4.3.

**Tabel 4.3 Hasil pembagian Dataset**

Epoch 1/100
6/6 - 10s - loss: 2.1169 - accuracy: 0.2188 - val_loss: 1.5360 - val_accuracy: 0.4219 - 10s/epoch - 2s/step
Epoch 2/100
6/6 - 7s - loss: 1.5484 - accuracy: 0.4024 - val_loss: 1.6954 - val_accuracy: 0.3438 - 7s/epoch - 1s/step
Epoch 3/100
6/6 - 11s - loss: 1.4670 - accuracy: 0.3646 - val_loss: 1.5309 - val_accuracy: 0.3750 - 11s/epoch - 2s/step
Epoch 4/100
6/6 - 8s - loss: 1.3948 - accuracy: 0.4115 - val_loss: 1.3809 - val_accuracy: 0.4219 - 8s/epoch - 1s/step
Epoch 5/100
6/6 - 9s - loss: 1.2978 - accuracy: 0.4792 - val_loss: 1.2960 - val_accuracy: 0.5000 - 9s/epoch - 1s/step
Epoch 6/100
6/6 - 10s - loss: 1.3321 - accuracy: 0.4740 - val_loss: 1.5722 - val_accuracy: 0.3906 - 10s/epoch - 2s/step
Epoch 7/100

6/6 - 10s - loss: 1.4738 - accuracy: 0.4695 - val\_loss: 1.6335 - val\_accuracy:  
0.3906 - 10s/epoch - 2s/step

Epoch 8/100

6/6 - 8s - loss: 1.2641 - accuracy: 0.5208 - val\_loss: 1.6258 - val\_accuracy:  
0.4219 - 8s/epoch - 1s/step

Epoch 9/100

6/6 - 9s - loss: 1.2557 - accuracy: 0.5104 - val\_loss: 1.2236 - val\_accuracy:  
0.4844 - 9s/epoch - 1s/step

Epoch 10/100

6/6 - 9s - loss: 1.1966 - accuracy: 0.5365 - val\_loss: 1.4388 - val\_accuracy:  
0.5469 - 9s/epoch - 2s/step

Epoch 11/100

6/6 - 9s - loss: 1.1401 - accuracy: 0.5677 - val\_loss: 1.2914 - val\_accuracy:  
0.5312 - 9s/epoch - 1s/step

Epoch 12/100

6/6 - 9s - loss: 1.3178 - accuracy: 0.5244 - val\_loss: 1.8920 - val\_accuracy:  
0.3281 - 9s/epoch - 1s/step

Epoch 13/100

6/6 - 7s - loss: 1.3154 - accuracy: 0.4583 - val\_loss: 1.4071 - val\_accuracy:  
0.4844 - 7s/epoch - 1s/step

Epoch 14/100

6/6 - 9s - loss: 1.0915 - accuracy: 0.5915 - val\_loss: 1.5361 - val\_accuracy:  
0.4531 - 9s/epoch - 1s/step

Epoch 15/100

6/6 - 9s - loss: 1.1574 - accuracy: 0.5729 - val\_loss: 1.5576 - val\_accuracy:  
0.3750 - 9s/epoch - 1s/step

Epoch 16/100

6/6 - 8s - loss: 1.1099 - accuracy: 0.5488 - val\_loss: 1.5437 - val\_accuracy:  
0.4375 - 8s/epoch - 1s/step

Epoch 17/100

6/6 - 8s - loss: 1.1136 - accuracy: 0.6037 - val\_loss: 1.7139 - val\_accuracy:  
0.5312 - 8s/epoch - 1s/step

Epoch 18/100

6/6 - 9s - loss: 1.0505 - accuracy: 0.5573 - val\_loss: 1.4721 - val\_accuracy:  
0.3594 - 9s/epoch - 1s/step

Epoch 19/100

6/6 - 8s - loss: 0.9461 - accuracy: 0.6458 - val\_loss: 1.5159 - val\_accuracy:  
0.5312 - 8s/epoch - 1s/step

Epoch 20/100

6/6 - 10s - loss: 0.9840 - accuracy: 0.6250 - val\_loss: 1.3316 - val\_accuracy:  
0.5156 - 10s/epoch - 2s/step

Epoch 21/100

6/6 - 8s - loss: 1.1006 - accuracy: 0.5938 - val\_loss: 1.0827 - val\_accuracy:  
0.5781 - 8s/epoch - 1s/step

Epoch 22/100

6/6 - 10s - loss: 1.0408 - accuracy: 0.5885 - val\_loss: 1.2450 - val\_accuracy:  
0.5156 - 10s/epoch - 2s/step

Epoch 23/100

6/6 - 9s - loss: 0.9553 - accuracy: 0.6198 - val\_loss: 1.4178 - val\_accuracy: 0.4844 - 9s/epoch - 1s/step  
Epoch 24/100  
6/6 - 9s - loss: 1.0270 - accuracy: 0.5915 - val\_loss: 1.5115 - val\_accuracy: 0.4062 - 9s/epoch - 2s/step  
Epoch 25/100  
6/6 - 7s - loss: 1.0775 - accuracy: 0.6402 - val\_loss: 1.5012 - val\_accuracy: 0.4844 - 7s/epoch - 1s/step  
Epoch 26/100  
6/6 - 9s - loss: 1.0972 - accuracy: 0.5260 - val\_loss: 1.3464 - val\_accuracy: 0.5000 - 9s/epoch - 1s/step  
Epoch 27/100  
6/6 - 7s - loss: 0.9642 - accuracy: 0.6585 - val\_loss: 1.5271 - val\_accuracy: 0.5312 - 7s/epoch - 1s/step  
Epoch 28/100  
6/6 - 8s - loss: 1.0272 - accuracy: 0.5976 - val\_loss: 1.1704 - val\_accuracy: 0.5469 - 8s/epoch - 1s/step  
Epoch 29/100  
6/6 - 7s - loss: 1.1414 - accuracy: 0.5305 - val\_loss: 1.5226 - val\_accuracy: 0.5000 - 7s/epoch - 1s/step  
Epoch 30/100  
6/6 - 9s - loss: 0.9299 - accuracy: 0.6198 - val\_loss: 1.6022 - val\_accuracy: 0.4844 - 9s/epoch - 1s/step  
Epoch 31/100

6/6 - 8s - loss: 0.9472 - accuracy: 0.6524 - val\_loss: 1.4351 - val\_accuracy: 0.5781 - 8s/epoch - 1s/step  
Epoch 32/100  
6/6 - 8s - loss: 0.8814 - accuracy: 0.6875 - val\_loss: 1.4442 - val\_accuracy: 0.5000 - 8s/epoch - 1s/step  
Epoch 33/100  
6/6 - 9s - loss: 0.8572 - accuracy: 0.6951 - val\_loss: 1.3626 - val\_accuracy: 0.5625 - 9s/epoch - 2s/step  
Epoch 34/100  
6/6 - 8s - loss: 0.9311 - accuracy: 0.6646 - val\_loss: 1.7811 - val\_accuracy: 0.4531 - 8s/epoch - 1s/step  
Epoch 35/100  
6/6 - 7s - loss: 0.9559 - accuracy: 0.6402 - val\_loss: 1.9487 - val\_accuracy: 0.5156 - 7s/epoch - 1s/step  
Epoch 36/100  
6/6 - 8s - loss: 1.1410 - accuracy: 0.6280 - val\_loss: 1.2212 - val\_accuracy: 0.4688 - 8s/epoch - 1s/step  
Epoch 37/100  
6/6 - 9s - loss: 1.1246 - accuracy: 0.5610 - val\_loss: 1.3895 - val\_accuracy: 0.4844 - 9s/epoch - 2s/step  
Epoch 38/100  
6/6 - 9s - loss: 0.9762 - accuracy: 0.6406 - val\_loss: 1.4030 - val\_accuracy: 0.4531 - 9s/epoch - 2s/step  
Epoch 39/100

6/6 - 7s - loss: 0.9305 - accuracy: 0.6524 - val\_loss: 1.2268 - val\_accuracy: 0.5156 - 7s/epoch - 1s/step  
Epoch 40/100  
6/6 - 7s - loss: 0.9387 - accuracy: 0.6615 - val\_loss: 1.2533 - val\_accuracy: 0.6250 - 7s/epoch - 1s/step  
Epoch 41/100  
6/6 - 10s - loss: 0.9997 - accuracy: 0.5781 - val\_loss: 1.1704 - val\_accuracy: 0.5938 - 10s/epoch - 2s/step  
Epoch 42/100  
6/6 - 7s - loss: 0.9389 - accuracy: 0.6463 - val\_loss: 1.5340 - val\_accuracy: 0.5625 - 7s/epoch - 1s/step  
Epoch 43/100  
6/6 - 8s - loss: 0.7575 - accuracy: 0.7240 - val\_loss: 1.1797 - val\_accuracy: 0.6094 - 8s/epoch - 1s/step  
Epoch 44/100  
6/6 - 9s - loss: 0.8119 - accuracy: 0.6951 - val\_loss: 1.3589 - val\_accuracy: 0.5781 - 9s/epoch - 1s/step  
Epoch 45/100  
6/6 - 8s - loss: 0.8939 - accuracy: 0.6463 - val\_loss: 1.5333 - val\_accuracy: 0.5625 - 8s/epoch - 1s/step  
Epoch 46/100  
6/6 - 9s - loss: 0.7922 - accuracy: 0.7292 - val\_loss: 1.0914 - val\_accuracy: 0.6250 - 9s/epoch - 2s/step  
Epoch 47/100

6/6 - 6s - loss: 0.7069 - accuracy: 0.7927 - val\_loss: 1.1739 - val\_accuracy: 0.6406 - 6s/epoch - 1s/step  
Epoch 48/100  
6/6 - 7s - loss: 0.8137 - accuracy: 0.7134 - val\_loss: 1.2772 - val\_accuracy: 0.5469 - 7s/epoch - 1s/step  
Epoch 49/100  
6/6 - 8s - loss: 0.8896 - accuracy: 0.7083 - val\_loss: 1.3851 - val\_accuracy: 0.5312 - 8s/epoch - 1s/step  
Epoch 50/100  
6/6 - 8s - loss: 0.6243 - accuracy: 0.7604 - val\_loss: 1.2129 - val\_accuracy: 0.5938 - 8s/epoch - 1s/step  
Epoch 51/100  
6/6 - 9s - loss: 0.7048 - accuracy: 0.7240 - val\_loss: 1.0384 - val\_accuracy: 0.6250 - 9s/epoch - 1s/step  
Epoch 52/100  
6/6 - 8s - loss: 0.7497 - accuracy: 0.7135 - val\_loss: 1.2013 - val\_accuracy: 0.5625 - 8s/epoch - 1s/step  
Epoch 53/100  
6/6 - 8s - loss: 0.7014 - accuracy: 0.7396 - val\_loss: 1.1726 - val\_accuracy: 0.6406 - 8s/epoch - 1s/step  
Epoch 54/100  
6/6 - 9s - loss: 0.6951 - accuracy: 0.7083 - val\_loss: 1.4488 - val\_accuracy: 0.5312 - 9s/epoch - 2s/step  
Epoch 55/100

6/6 - 9s - loss: 0.5759 - accuracy: 0.7865 - val\_loss: 1.5027 - val\_accuracy: 0.5156 - 9s/epoch - 1s/step  
Epoch 56/100  
6/6 - 9s - loss: 0.7469 - accuracy: 0.7292 - val\_loss: 1.4315 - val\_accuracy: 0.5938 - 9s/epoch - 1s/step  
Epoch 57/100  
6/6 - 8s - loss: 0.7595 - accuracy: 0.7012 - val\_loss: 1.7692 - val\_accuracy: 0.5625 - 8s/epoch - 1s/step  
Epoch 58/100  
6/6 - 9s - loss: 0.7691 - accuracy: 0.7195 - val\_loss: 1.7279 - val\_accuracy: 0.5781 - 9s/epoch - 2s/step  
Epoch 59/100  
6/6 - 8s - loss: 0.7616 - accuracy: 0.7344 - val\_loss: 1.2897 - val\_accuracy: 0.6406 - 8s/epoch - 1s/step  
Epoch 60/100  
6/6 - 8s - loss: 0.7272 - accuracy: 0.7439 - val\_loss: 1.4875 - val\_accuracy: 0.5312 - 8s/epoch - 1s/step  
Epoch 61/100  
6/6 - 9s - loss: 0.7257 - accuracy: 0.7396 - val\_loss: 1.3096 - val\_accuracy: 0.5312 - 9s/epoch - 1s/step  
Epoch 62/100  
6/6 - 8s - loss: 0.5755 - accuracy: 0.7812 - val\_loss: 1.3589 - val\_accuracy: 0.6094 - 8s/epoch - 1s/step  
Epoch 63/100

6/6 - 9s - loss: 0.6375 - accuracy: 0.7683 - val\_loss: 1.3619 - val\_accuracy: 0.5000 - 9s/epoch - 1s/step  
Epoch 64/100  
6/6 - 9s - loss: 0.8166 - accuracy: 0.7073 - val\_loss: 0.9632 - val\_accuracy: 0.7031 - 9s/epoch - 1s/step  
Epoch 65/100  
6/6 - 8s - loss: 0.8075 - accuracy: 0.7195 - val\_loss: 2.0972 - val\_accuracy: 0.4844 - 8s/epoch - 1s/step  
Epoch 66/100  
6/6 - 8s - loss: 0.7364 - accuracy: 0.7031 - val\_loss: 1.2529 - val\_accuracy: 0.6562 - 8s/epoch - 1s/step  
Epoch 67/100  
6/6 - 8s - loss: 0.6714 - accuracy: 0.7656 - val\_loss: 1.4898 - val\_accuracy: 0.5312 - 8s/epoch - 1s/step  
Epoch 68/100  
6/6 - 9s - loss: 0.6287 - accuracy: 0.7604 - val\_loss: 1.3535 - val\_accuracy: 0.6094 - 9s/epoch - 2s/step  
Epoch 69/100  
6/6 - 8s - loss: 0.5337 - accuracy: 0.8177 - val\_loss: 1.3175 - val\_accuracy: 0.5938 - 8s/epoch - 1s/step  
Epoch 70/100  
6/6 - 8s - loss: 0.4972 - accuracy: 0.8177 - val\_loss: 1.3062 - val\_accuracy: 0.6719 - 8s/epoch - 1s/step  
Epoch 71/100

6/6 - 9s - loss: 0.6271 - accuracy: 0.7500 - val\_loss: 1.7824 - val\_accuracy: 0.5312 - 9s/epoch - 2s/step

Epoch 72/100

6/6 - 9s - loss: 0.5709 - accuracy: 0.7812 - val\_loss: 1.6498 - val\_accuracy: 0.5469 - 9s/epoch - 1s/step

Epoch 73/100

6/6 - 9s - loss: 0.5750 - accuracy: 0.7917 - val\_loss: 1.8854 - val\_accuracy: 0.5781 - 9s/epoch - 1s/step

Epoch 74/100

6/6 - 7s - loss: 0.5597 - accuracy: 0.7561 - val\_loss: 1.5395 - val\_accuracy: 0.4844 - 7s/epoch - 1s/step

Epoch 75/100

6/6 - 7s - loss: 0.6633 - accuracy: 0.7500 - val\_loss: 1.1842 - val\_accuracy: 0.6094 - 7s/epoch - 1s/step

Epoch 76/100

6/6 - 10s - loss: 0.6681 - accuracy: 0.7683 - val\_loss: 1.6813 - val\_accuracy: 0.5781 - 10s/epoch - 2s/step

Epoch 77/100

6/6 - 8s - loss: 0.6350 - accuracy: 0.7708 - val\_loss: 1.4322 - val\_accuracy: 0.5938 - 8s/epoch - 1s/step

Epoch 78/100

6/6 - 7s - loss: 0.5687 - accuracy: 0.8110 - val\_loss: 1.5957 - val\_accuracy: 0.5781 - 7s/epoch - 1s/step

Epoch 79/100

6/6 - 8s - loss: 0.5900 - accuracy: 0.7969 - val\_loss: 1.2933 - val\_accuracy: 0.5312 - 8s/epoch - 1s/step

Epoch 80/100

6/6 - 8s - loss: 0.5204 - accuracy: 0.8125 - val\_loss: 1.0196 - val\_accuracy: 0.5625 - 8s/epoch - 1s/step

Epoch 81/100

6/6 - 8s - loss: 0.4752 - accuracy: 0.8073 - val\_loss: 1.5473 - val\_accuracy: 0.5625 - 8s/epoch - 1s/step

Epoch 82/100

6/6 - 7s - loss: 0.4907 - accuracy: 0.8415 - val\_loss: 1.7444 - val\_accuracy: 0.4688 - 7s/epoch - 1s/step

Epoch 83/100

6/6 - 7s - loss: 0.5457 - accuracy: 0.8073 - val\_loss: 2.0736 - val\_accuracy: 0.5938 - 7s/epoch - 1s/step

Epoch 84/100

6/6 - 7s - loss: 0.4797 - accuracy: 0.8073 - val\_loss: 1.5080 - val\_accuracy: 0.5469 - 7s/epoch - 1s/step

Epoch 85/100

6/6 - 8s - loss: 0.4657 - accuracy: 0.7988 - val\_loss: 1.7500 - val\_accuracy: 0.5781 - 8s/epoch - 1s/step

Epoch 86/100

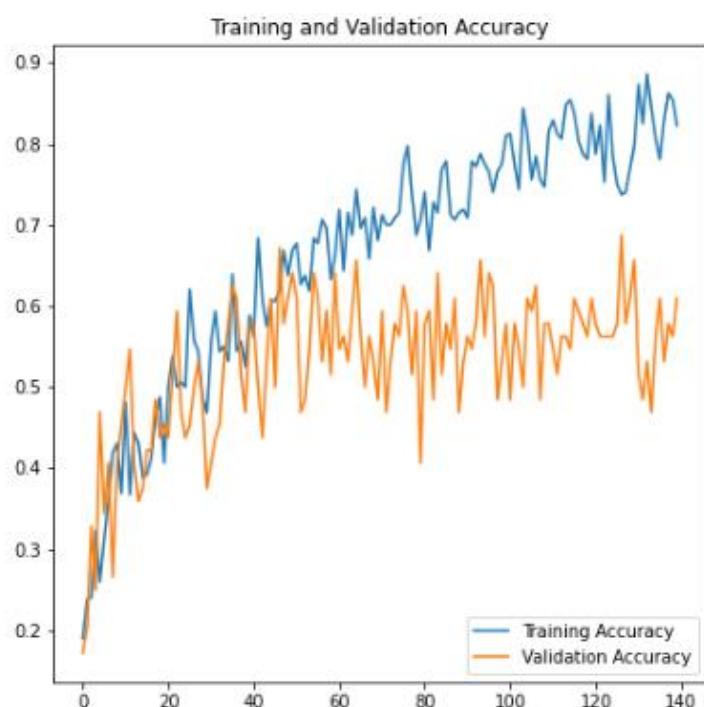
6/6 - 7s - loss: 0.4795 - accuracy: 0.8293 - val\_loss: 1.4733 - val\_accuracy: 0.5781 - 7s/epoch - 1s/step

Epoch 87/100

6/6 - 8s - loss: 0.4471 - accuracy: 0.8177 - val\_loss: 1.4211 - val\_accuracy: 0.6094 - 8s/epoch - 1s/step  
Epoch 88/100  
6/6 - 10s - loss: 0.3804 - accuracy: 0.8750 - val\_loss: 1.6821 - val\_accuracy: 0.5156 - 10s/epoch - 2s/step  
Epoch 89/100  
6/6 - 8s - loss: 0.4310 - accuracy: 0.8594 - val\_loss: 1.2455 - val\_accuracy: 0.6250 - 8s/epoch - 1s/step  
Epoch 90/100  
6/6 - 9s - loss: 0.4852 - accuracy: 0.8293 - val\_loss: 1.6570 - val\_accuracy: 0.5625 - 9s/epoch - 1s/step  
Epoch 91/100  
6/6 - 9s - loss: 0.4380 - accuracy: 0.8333 - val\_loss: 2.0807 - val\_accuracy: 0.5000 - 9s/epoch - 2s/step  
Epoch 92/100  
6/6 - 8s - loss: 0.2963 - accuracy: 0.8906 - val\_loss: 1.4767 - val\_accuracy: 0.6406 - 8s/epoch - 1s/step  
Epoch 93/100  
6/6 - 9s - loss: 0.4076 - accuracy: 0.8542 - val\_loss: 1.7279 - val\_accuracy: 0.5625 - 9s/epoch - 1s/step  
Epoch 94/100  
6/6 - 9s - loss: 0.3203 - accuracy: 0.9062 - val\_loss: 2.0066 - val\_accuracy: 0.5938 - 9s/epoch - 2s/step  
Epoch 95/100

```
6/6 - 9s - loss: 0.3503 - accuracy: 0.8854 - val_loss: 1.0145 - val_accuracy:  
0.7500 - 9s/epoch - 1s/step  
Epoch 96/100  
  
6/6 - 9s - loss: 0.2952 - accuracy: 0.9010 - val_loss: 1.9425 - val_accuracy:  
0.5781 - 9s/epoch - 1s/step  
Epoch 97/100  
  
6/6 - 6s - loss: 0.4163 - accuracy: 0.8659 - val_loss: 1.7870 - val_accuracy:  
0.5938 - 6s/epoch - 1s/step  
Epoch 98/100  
  
6/6 - 8s - loss: 0.4258 - accuracy: 0.8415 - val_loss: 1.3288 - val_accuracy:  
0.6406 - 8s/epoch - 1s/step  
Epoch 99/100  
  
6/6 - 7s - loss: 0.4890 - accuracy: 0.8125 - val_loss: 2.1171 - val_accuracy:  
0.5469 - 7s/epoch - 1s/step  
Epoch 100/100  
  
6/6 - 7s - loss: 0.4257 - accuracy: 0.8415 - val_loss: 2.2863 - val_accuracy:  
0.5312 - 7s/epoch - 1s/step
```

Dari hasil yang ditampilkan pada **tabel 4.3** dapat digambarkan dengan gambar grafik untuk melihat hasil pembagian dataset agar mudah dilihat. Grafik akurasi dari data latih dan validasi dapat dilihat pada gambar 4.8. Grafik loss dari data latih dan data validasi dapat dilihat pada gambar 4.9.

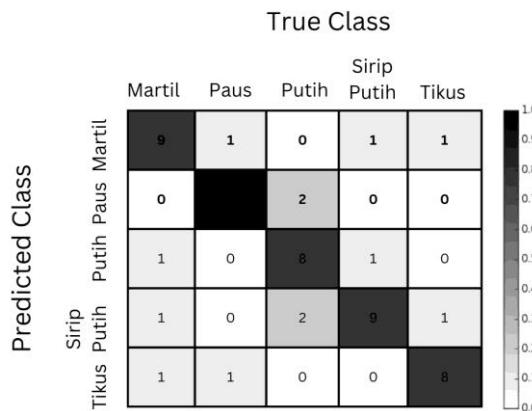


**Gambar 4.8 Grafik Akurasi data latih dan validasi**



**Gambar 4.9 Grafik Loss data latih dan validasi**

#### 4.4 Perhitungan Akurasi di Aplikasi



**Gambar 4.10 Perhitungan Akurasi di Aplikasi**

Pada gambar 4.10 telah dilakukan perhitungan akurasi di aplikasi dalam penelitian ini, hasil akurasi yang didapatkan dari keberhasilan pengujian adalah sebagai berikut:

$$\text{Akurasi} = \frac{\text{Jumlah data yang benar}}{\text{Total}} \times 100\%$$

$$\text{Akurasi} = \frac{44}{60} \times 100\% = 73,3\%$$

Penulis menggunakan 60 data test kemudian memasukan hasil ke dalam confusion matrix dimana hanya 44 data test yang benar dalam pengujian yang menyebabkan nilai akurasi menjadi 73,3%.

#### 4.5 Evaluation Method

Berdasarkan pada gambar 4.10, maka akan dilakukan sebuah evaluation method dengan menggunakan *recall* dan *precision*, hasil dari *recall* dan *precision* dapat dilihat pada gambar 4.11.

		precision	recall
Martil		0.750	0.750
Paus		0.833	0.833
Putih		0.800	0.666
Sirip Putih		0.692	0.818
Tikus		0.800	0.800

**Gambar 4.11 Evaluation Method**

## BAB 5

### PENUTUP

#### 5.1 Simpulan

Dengan menggunakan *Convolution Neural Network* (CNN) pada penelitian ini, penulis memperoleh tingkat akurasi sebesar 73.3%, dengan total data test 44 gambar yang dapat mengklasifikasi gambar sesuai dengan kelasnya, dan 16 gambar tidak dapat mengklasifikasi gambar sesuai dengan kelasnya. Setelah mendapatkan tingkat akurasi yang baik, penulis menyimpulkan bahwa algoritma CNN dapat digunakan dalam mengklasifikasi jenis ikan hiu karena tingkat akurasinya yang baik.

#### 5.2 Saran

Setelah mendapatkan simpulan seperti itu, penulis memiliki saran untuk penelitian kedepannya, agar menambahkan Data Set untuk menambah tingkat akurasi dari yang sekarang penulis teliti.

#### 5.3 Keterbatasan Penelitian

Penulis mempunyai keterbatasan dalam pengumpulan Data Set yang menyebabkan Data Set yang digunakan pada penelitian ini sedikit.

#### 5.4 Kontribusi Penelitian

Penelitian ini mendapatkan tingkat akurasi yang lebih tinggi dibandingkan penelitian terdahulu.

## **DAFTAR REFERENSI**

- [1] Fahmi, “MENGENAL JENIS HIU APENDIKS II CITES,” vol. XLIII, pp. 1–17, 2018, [Online]. Available: [www.cites.org/eng/cop/index](http://www.cites.org/eng/cop/index).
- [2] M. R. Rasyid, Z. Tahir, and Syafaruddin, “Digital Image Processing for Detecting Industrial Machine Work Failure with Quantization Vector Learning Method,” *Journal Pekommas*, vol. 4, no. 2, p. 131, Oct. 2019, doi: 10.30818/jpkm.2019.2040203.
- [3] J. Jumadi and D. Sartika, “PENGOLAHAN CITRA DIGITAL UNTUK IDENTIFIKASI OBJEK MENGGUNAKAN METODE HIERARCHICAL AGGLOMERATIVE CLUSTERING,” vol. 10, no. 2, 2021.
- [4] M. R. Effendi, “SISTEM DETEKSI WAJAH JENIS KUCING DENGAN IMAGE CLASSIFICATION MENGGUNAKAN OPENCV,” 2018.
- [5] N. Fadlia and R. Kosasih, “KLASIFIKASI JENIS KENDARAAN MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN),” *Jurnal Ilmiah Teknologi dan Rekayasa*, vol. 24, no. 3, pp. 207–215, 2019, doi: 10.35760/tr.2019.v24i3.2397.
- [6] R. W. Pratiwi and Y. S. Nugroho, “PREDIKSI RATING FILM MENGGUNAKAN METODE NAÏVE BAYES.” [Online]. Available: <https://www.kaggle.com/>.
- [7] A. Wehantouw, E. Like Ginting, and S. Wullur, “IDENTIFIKASI SIRIP IKAN HIU YANG DIDAPAT DARI PENGUMPUL DI MINAHASA

- TENGGARA MENGGUNAKAN DNA BARCODE,” 2017. [Online]. Available: <http://www.ncbi.nlm.nih.gov>.
- [8] M. Isyatan Mardiyah, “IMPLEMENTASI DEEP LEARNING UNTUK IMAGE CLASSIFICATION MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) PADA CITRA KEBUN DAN SAWAH,” 2020.
- [9] T. M. Kadarina and M. H. Ibnu Hajar, “PENGENALAN BAHASA PEMROGRAMAN PYTHON MENGGUNAKAN APLIKASI GAMES UNTUK SISWA/I DI WILAYAH KEMBANGAN UTARA,” 2019. [Online]. Available: <https://codecombat.com/>.
- [10] A. Roihan, P. Abas Sunarya, and A. S. Rafika, “Pemanfaatan Machine Learning dalam Berbagai Bidang: Review paper,” 2019.
- [11] A. Rahim, Kusrini, and E. Taufiq Luthfi, “Convolutional Neural Network untuk Kalasifikasi Penggunaan Masker,” 2020.
- [12] O. N. Putri, “IMPLEMENTASI METODE CNN DALAM KLASIFIKASI GAMBAR JAMUR PADA ANALISIS IMAGE PROCESSING,” 2020.
- [13] H. N. Putra, “Implementasi Diagram UML (Unified Modelling Language) dalam Perancangan Aplikasi Data Pasien Rawat Inap pada Puskesmas Lubuk Buaya,” vol. 2, 2018.
- [14] I. Budiman, S. Saori, R. Nurul Anwar, Fitriani, and M. Yuga Pangestu, “ANALISIS PENGENDALIAN MUTU DI BIDANG INDUSTRI

MAKANAN (Studi Kasus: UMKM Mochi Kaswari Lampion Kota Sukabumi)," *JIP (Jurnal Inovasi Penelitian)*, vol. 1, no. 10, 2021.

- [15] T. Bayu Kurniawan, "PERANCANGAN SISTEM APLIKASI PEMESANAN MAKANAN DAN MINUMAN PADA CAFETARIA NO CAFFE DI TANJUNG BALAI KARIMUN MENGGUNAKAN BAHASA PEMOGRAMAN PHP DAN MYSQL," *TIKAR*, vol. 1, no. 2, 2020.
- [16] J. Simatupang and S. Sianturi, "PERANCANGAN SISTEM INFORMASI PEMESANAN TIKET BUS PADA PO. HANDOYO BERBASIS ONLINE," *Intra-Tech*, vol. 3, 2019.
- [17] I. Zulfa, H. Syahputra, and A. Faisal, "RANCANG BANGUN SYSTEM KONTROL ALAT-ALAT LISTRIK MENGGUNAKAN BLUETOOTH BERBASIS MIKROKONTROLER," vol. 14, no. 1, pp. 188–199, 2021, [Online]. Available: <http://journal.stekom.ac.id/index.php/elkom>■page188
- [18] P. D. P. Silitonga and D. E. R. Purba, "IMPLEMENTASI SYSTEM DEVELOPMENT LIFE CYCLE PADA RANCANG BANGUN SISTEM PENDAFTARAN PASIEN BERBASIS WEB," *Jurnal Sistem Informasi Kaputama (JSIK)*, vol. 5, no. 2, 2021.
- [19] A. P. Setiany, D. Noviyanto, M. Irfansyahfalaloh, S. Aisah, A. Saifudin, and I. Kusyadi, "Jurnal Teknologi Sistem Informasi dan Aplikasi Penggunaan Metode System Development Life Cycle ( SDLC) dalam Analasis dan Perancangan Sistem Informasi Penerimaan Kas Sekolah," vol. 4, no. 3, pp. 179–186, 2021, doi: 10.32493/jtsi.v4i3.11992.

- [20] Karsito and S. Susanti, “KLASIFIKASI KELAYAKAN PESERTA PENGAJUAN KREDIT RUMAH DENGAN ALGORITMA NAÏVE BAYES DI PERUMAHAN AZZURA RESIDENCIA,” vol. 9, 2019.
- [21] D. Normawati and S. A. Prayogi, “Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitter,” 2021.

## **RIWAYAT HIDUP**

Nama : Davin Christian Juan  
NIM : 32180045  
Tempat/tanggal lahir : Serang/10 Juni 2000  
Jenis Kelamin : Laki-Laki  
Alamat : Batu Ceper Permai Blok J No.4, Tangerang, 15122  
No. Telp : 087809669492



### **Riwayat Pendidikan**

Tahun 2006 s/d 2012 Sekolah Dasar, SD Mardi Yuana. Serang  
Tahun 2012 s/d 2015 Sekolah Menengah Pertama, Maria Immaculata.  
Tangerang  
Tahun 2015 s/d 2018 Sekolah Menengah Kejuruan, Mutiara Bangsa 1.  
Tangerang  
Tahun 2018 s/d 2023 Sarjana, Universitas Bunda Mulia. Tangerang

### **Pengalaman Kerja**

Tahun 2017 Departemen Teknik, PT. Surya Rengo Container  
Tahun 2022 Translator MLBB ENG-ID, Moonton  
Tahun 2022 IT Staff Internship, Bunda Mulia University

## LAMPIRAN

Coding Test.py

```

import os
import uuid
import urllib
from tensorflow.keras.models import load_model
from flask import Flask, render_template, request, send_file
from tensorflow.keras.preprocessing.image import load_img, img_to_array

app = Flask(__name__)
BASE_DIR = os.path.dirname(os.path.abspath(__file__))
model = load_model(os.path.join(BASE_DIR, 'ModelDataTest10affhiyh.h5'))
ALLOWED_EXT = set(['jpg', 'png'])

def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1] in ALLOWED_EXT

classes = ['Hiu Martil', 'Hiu Paus', 'Hiu Putih', 'Hiu Sirip Putih', 'Hiu Tikus', 'Lainnya']

def predict(filename, model):
    img = load_img(filename, target_size=(100, 100))
    img = img_to_array(img)
    img = img.reshape(1, 100, 100, 3)
    img = img.astype('float32')
    img = img / 255.0
    result = model.predict(img)
    dict_result = {}
    for i in range(6):
        dict_result[result[0][i]] = classes[i]
    res = result[0]
    res.sort()
    res = res[::-1]
    acc = res[:6]

    acc_result = []
    class_result = []
    for i in range(6):
        acc_result.append((acc[i] * 100).round(3))
        class_result.append(dict_result[acc[i]])
    return class_result, acc_result

@app.route('/')
def home():
    return render_template("index.html")

```

```

@app.route('/success', methods=['GET', 'POST'])
def success():
    error = ""
    target_img = os.path.join(os.getcwd(), 'static/images')
    if request.method == 'POST':
        if (request.form):
            link = request.form.get('link')
            try:
                resource = urllib.request.urlopen(link)
                unique_filename = str(uuid.uuid4())
                filename = unique_filename + ".jpg"
                img_path = os.path.join(target_img, filename)
                output = open(img_path, "wb")
                output.write(resource.read())
                output.close()
                img = filename
                class_result, acc_result = predict(img_path, model)
                predictions = {
                    "class1": class_result[0],
                    "class2": class_result[1],
                    "class3": class_result[2],
                    "class4": class_result[3],
                    "class5": class_result[4],
                    "class6": class_result[5],
                    "acc1": acc_result[0],
                    "acc2": acc_result[1],
                    "acc3": acc_result[2],
                    "acc4": acc_result[3],
                    "acc5": acc_result[4],
                    "acc6": acc_result[5],
                }
            except Exception as e:
                print(str(e))
                error = 'This image from this site is not accesible or inappropriate input'
            if (len(error) == 0):
                return render_template('success.html', img=img,
predictions=predictions)
            else:
                return render_template('index.html', error=error)

        elif (request.files):
            file = request.files['file']
            if file and allowed_file(file.filename):
                file.save(os.path.join(target_img, file.filename))
                img_path = os.path.join(target_img, file.filename)
                img = file.filename

```

```

class_result, acc_result = predict(img_path, model)
predictions = {
    "class1": class_result[0],
    "class2": class_result[1],
    "class3": class_result[2],
    "class4": class_result[3],
    "class5": class_result[4],
    "class6": class_result[5],
    "acc1": acc_result[0],
    "acc2": acc_result[1],
    "acc3": acc_result[2],
    "acc4": acc_result[3],
    "acc5": acc_result[4],
    "acc6": acc_result[5],
}
else:
    error = "JPG and PNG extension file only"
if (len(error) == 0):
    return render_template('success.html', img=img,
predictions=predictions)
else:
    return render_template('index.html', error=error)
else:
    return render_template('index.html')

if __name__ == "__main__":
    app.run(debug=True)

```

## Coding Success.html

```

<!DOCTYPE html>
<html lang="eng">
<head>
    <title> Hasil Akurasi </title>
    <link rel="stylesheet"
        href="{{ url_for('static', filename='style2.css') }}"
        type="text/css">
</head>
<body><br>
    <h2 class="upload">Uploaded Image</h2>
    Model Prediction</h2>
    <table border = "1" class="tbl">
        <tr>

```

```

<td>Rank</td>
<td>Class</td>
<td>Accuracy</td>
</tr>
<tr>
    <th>1</th>
    <th>{ predictions.class1 }</th>
    <th>{ predictions.acc1 } %</th>
</tr>
<tr>
    <td>2</td>
    <td>{ predictions.class2 }</td>
    <td>{ predictions.acc2 } %</td>
</tr>
<tr>
    <td>3</td>
    <td>{ predictions.class3 }</td>
    <td>{ predictions.acc3 } %</td>
</tr>
<tr>
    <td>4</td>
    <td>{ predictions.class4 }</td>
    <td>{ predictions.acc4 } %</td>
</tr>
<tr>
    <td>5</td>
    <td>{ predictions.class5 }</td>
    <td>{ predictions.acc5 } %</td>
</tr>
<tr>
    <td>6</td>
    <td>{ predictions.class6 }</td>
    <td>{ predictions.acc6 } %</td>
</tr>
</table><br>
<a href ="/" class = "btnb">BACK</a>
</body>
</html>

```

## Coding Index.html

```

<!DOCTYPE html>
<html lang="eng">
<head>
    <title> Klasifikasi Jenis-Jenis Ikan Hiu </title>

```

```
<link rel="stylesheet"
      href="{{ url_for('static', filename='style.css') }}"
      type="text/css">
</head>

<body>
  <h1 class = home >Klasifikasi Jenis-Jenis Ikan Hiu </h1>
  <form class = "brdhome file-form" action="/success" method="post",
enctype="multipart/form-data">
    <input class="btnc file-form-input" type="file" , name = "file"/>
  <br><br>
    <button class="btnp btn-success btn-lg">Eksekusi</button>
  </form>
  <p class = "error">{{ error }}</p>
  <p class = "predict">Hanya dapat mengeksekusi gambar dengan
ekstensi JPG, dan PNG <br>Klasifikasi Jenis Ikan Hiu: <br>Hiu Martil, Hiu
Paus, Hiu Putih, Hiu Sirip Putih, dan Hiu Tikus</p>
</body>
</html>
```

### Coding Style.css

```
body {

background-image: url("bg hiu.jpg");
background-size:cover;
background-repeat: no-repeat;
background-attachment: fixed;
background-position:center;
}

.home {
color : yellow;
text-align:center;
```

```
font-size:75px;  
font-family:"Century"  
margin-top:3%;  
width : 100%;  
padding-top : 5%;  
}  
  
.btnc {  
margin-left:23%;  
}  
  
.btnp {  
margin-left:23%;  
width : 52%;  
}  
  
.brdhome {  
background-color : yellow;  
border-radius: 15px;  
opacity : 80%;  
width : 25%;  
margin-left: 34%;  
padding :50px;  
font-family:"Century";
```

```
}
```

```
.show {
```

```
    margin-left: 42.5%;
```

```
    border-radius: 10px;
```

```
    width:15%;
```

```
    height:15%;
```

```
}
```

```
.error {
```

```
    text-align: center;
```

```
    margin-top: 15px;
```

```
    color: #8B0000;
```

```
    font-family:"Century";
```

```
}
```

```
.predict {
```

```
    text-align: center;
```

```
    color:yellow;
```

```
    font-size:35px;
```

```
    font-family:"Century";
```

```
}
```

```
.btnb {
```

```
color : #594128;  
  
justify-content: center;  
  
font-family:"Century";  
  
background-color : white;  
  
border-radius: 5px;  
  
margin-left: 47%;  
  
padding :10px;  
  
}  
  
  
.upload {  
  
font-family:"Century";  
  
text-align: center;  
  
color:white;  
  
}  
  
  
.tbl {  
  
font-family:"Century";  
  
text-align: center;  
  
margin-left: 37%;  
  
width:25.5%;  
  
background-color : white;  
  
color:#594128;  
  
}
```

```
.tblsucess {  
    padding: 3%;  
    opacity : 75%;  
}
```

### Coding Style2.css

```
body {  
background-image: url("bg hiu.jpg");  
background-size:cover;  
background-repeat: no-repeat;  
background-attachment: fixed;  
background-position:center;  
}
```

```
.home {  
color : white;  
text-align:center;  
font-size:35px;  
font-family:"Century"  
margin-top:3%;  
width : 100%;  
padding-top : 5%;  
}
```

```
.btnc {  
margin-left:23%;  
}  
  
.btnp {  
margin-left:23%;  
width : 52%;  
}  
  
.brdhome {  
background-color : white;  
border-radius: 15px;  
opacity : 80%;  
width : 25%;  
margin-left: 34%;  
padding :50px;  
font-family:"Century";  
}  
  
.show {  
margin-left: 42.5%;  
border-radius: 10px;  
width:15%;
```

```
height:15%;  
}  
  
.error {  
text-align: center;  
margin-top: 15px;  
color: #8B0000;  
font-family:"Century";  
}  
  
.predict {  
text-align: center;  
color:orange;  
font-family:"Century";  
}  
  
.btnb {  
color : #594128;  
justify-content: center;  
font-family:"Century";  
background-color : yellow ;  
border-radius: 5px;  
margin-left: 47%;  
padding :10px;
```

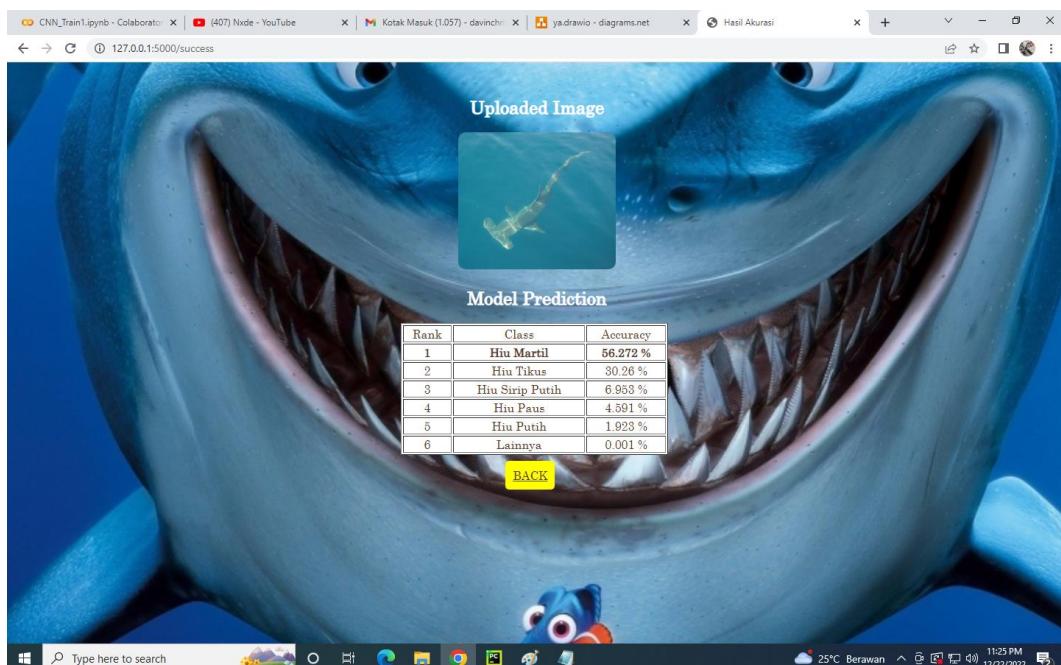
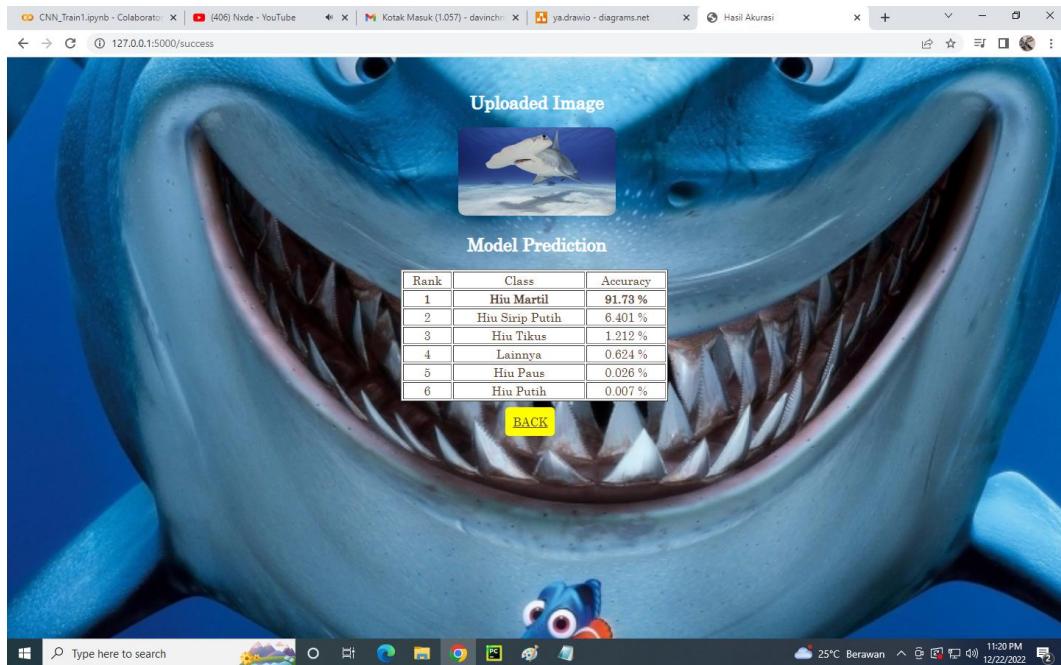
```
}
```

```
.upload {  
font-family:"Century";  
text-align: center;  
color:white;  
}
```

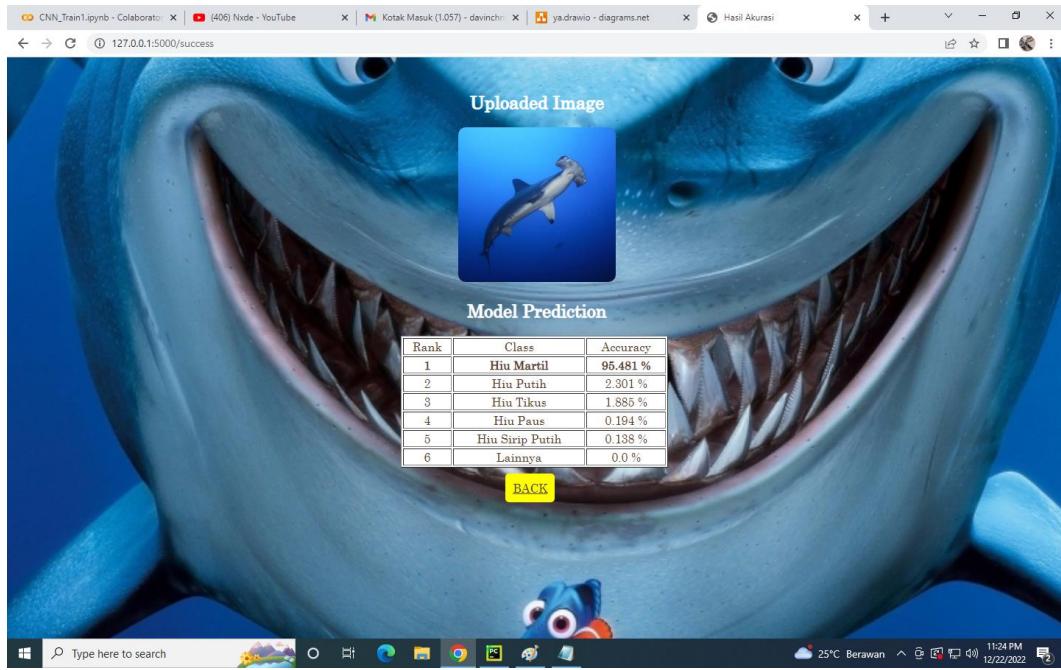
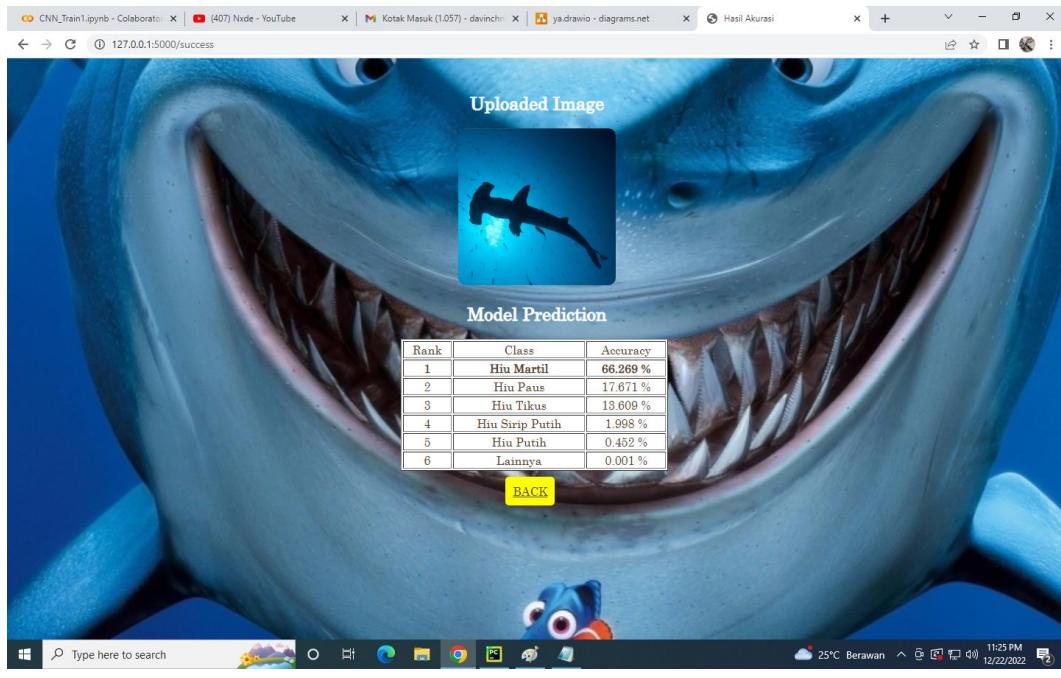
```
.tbl {  
font-family:"Century";  
text-align: center;  
margin-left: 37%;  
width:25.5%;  
background-color : white;  
color:#594128;  
}
```

```
.tblsuccess {  
padding: 3%;  
opacity : 75%;  
}
```

## Hasil Akurasi



## Lampiran



## Lampiran

The screenshot shows a web browser window with a background image of a shark's mouth. At the top, it says "Uploaded Image" and displays a small thumbnail of a hammerhead shark. Below that, it says "Model Prediction" and shows a table of results:

Rank	Class	Accuracy
1	Hiu Tikus	97.904 %
2	Hiu Martil	1.98 %
3	Hiu Putih	0.054 %
4	Hiu Sirip Putih	0.032 %
5	Hiu Paus	0.029 %
6	Lainnya	0.0 %

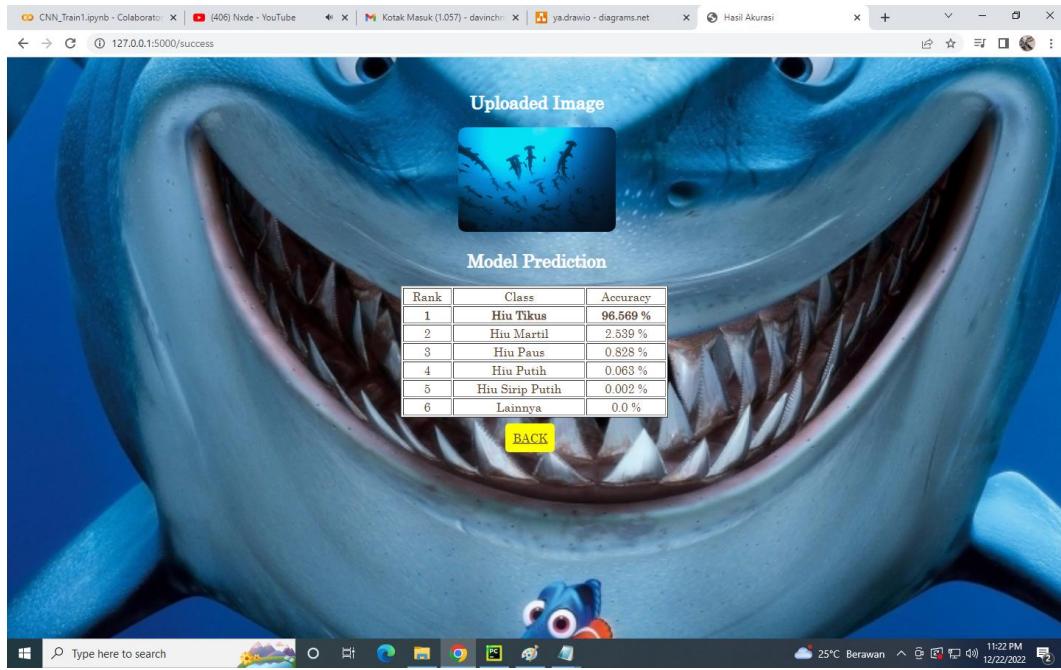
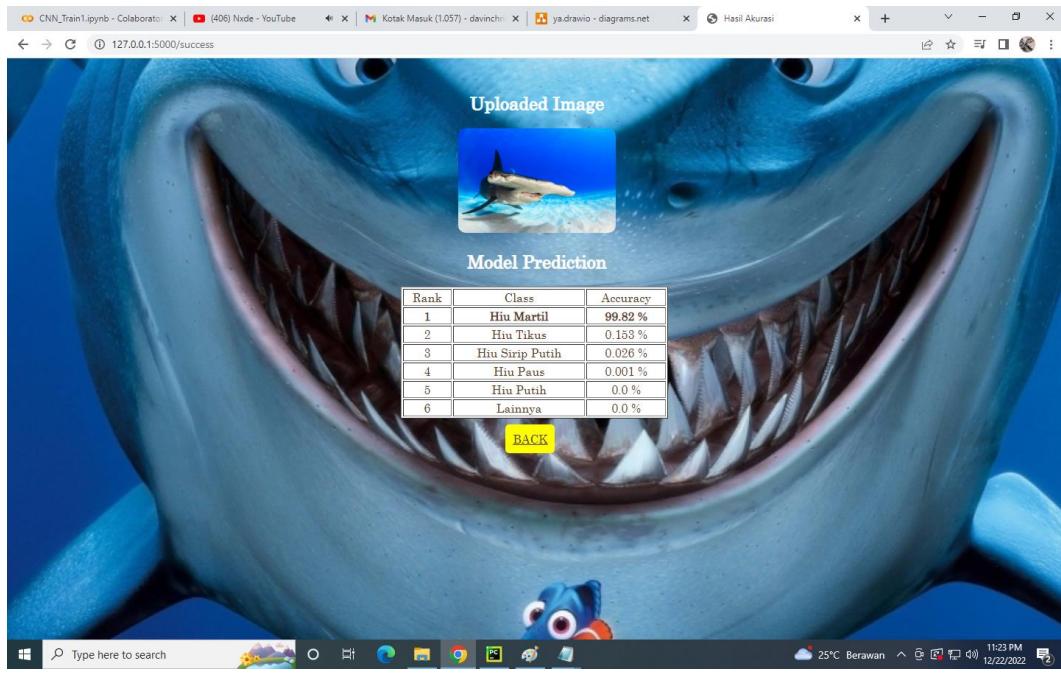
A yellow "BACK" button is located at the bottom of the prediction table.

The screenshot shows a web browser window with a background image of a shark's mouth. At the top, it says "Uploaded Image" and displays a small thumbnail of a hammerhead shark. Below that, it says "Model Prediction" and shows a table of results:

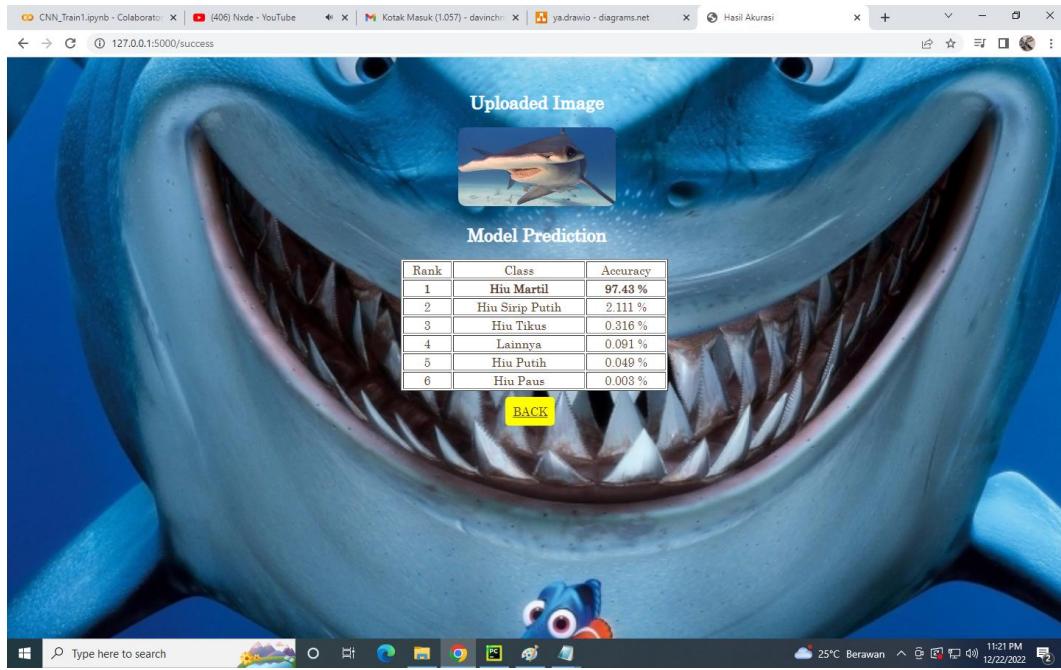
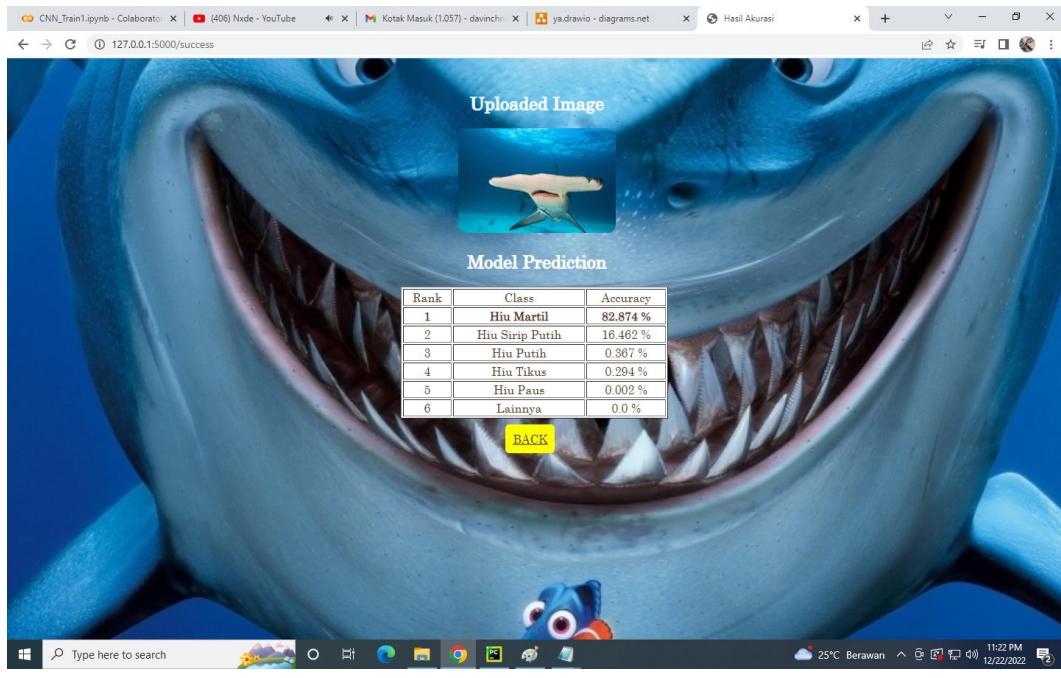
Rank	Class	Accuracy
1	Hiu Sirip Putih	62.672 %
2	Hiu Tikus	20.704 %
3	Lainnya	12.508 %
4	Hiu Martil	3.813 %
5	Hiu Putih	0.298 %
6	Hiu Paus	0.005 %

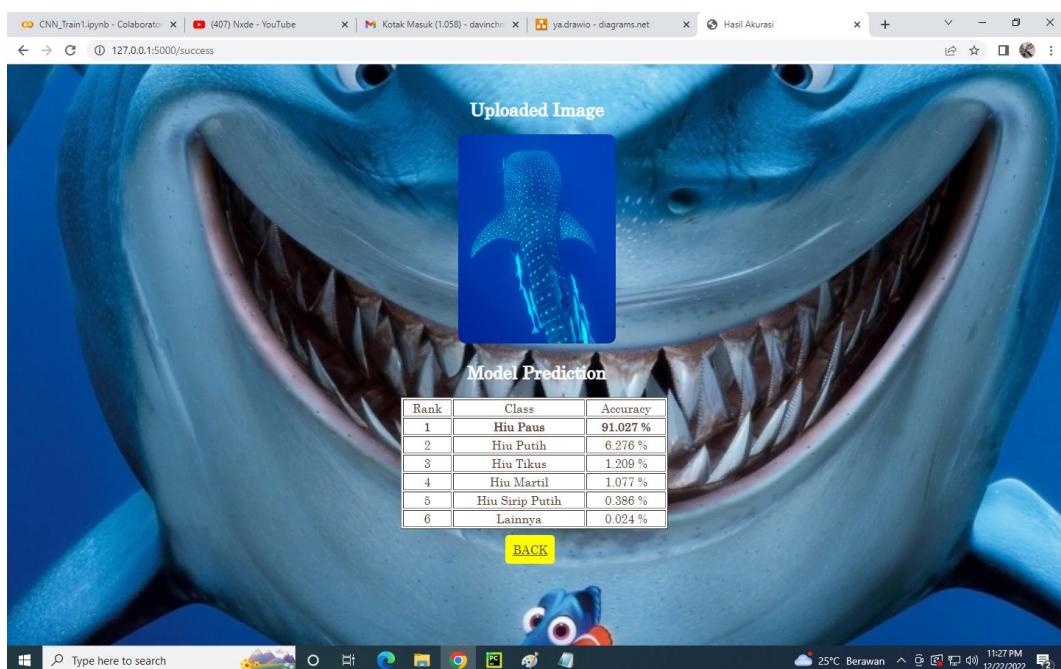
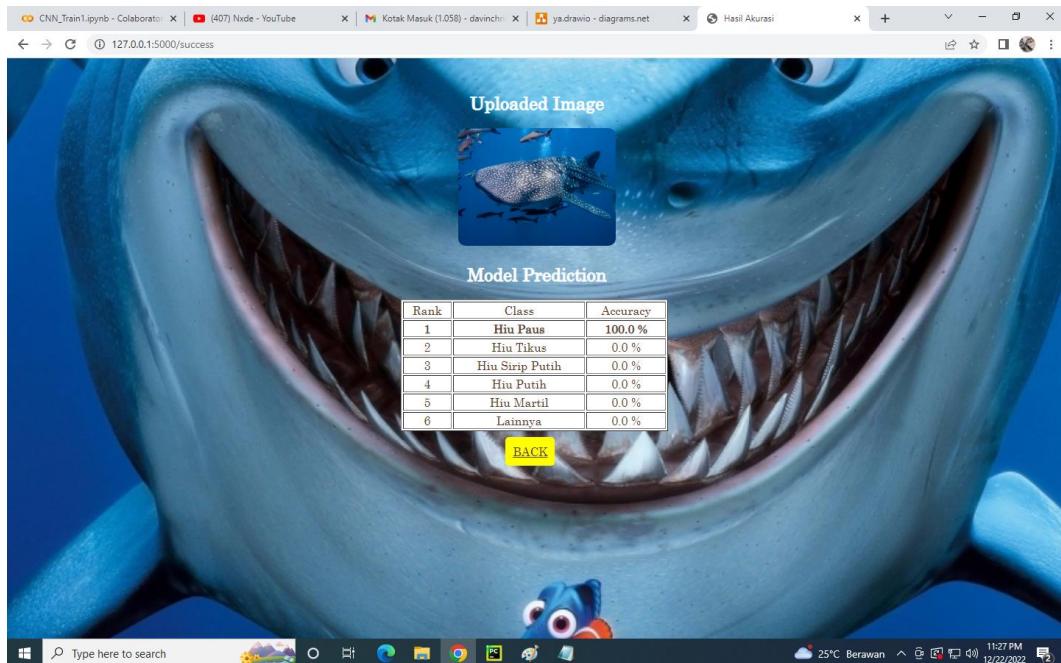
A yellow "BACK" button is located at the bottom of the prediction table.

## Lampiran

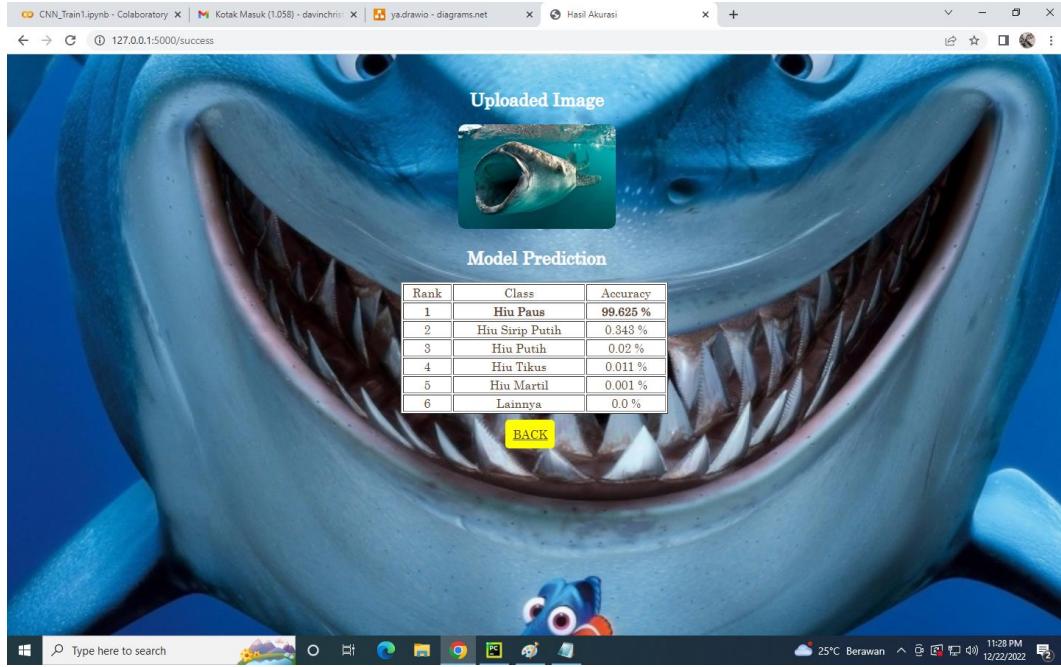
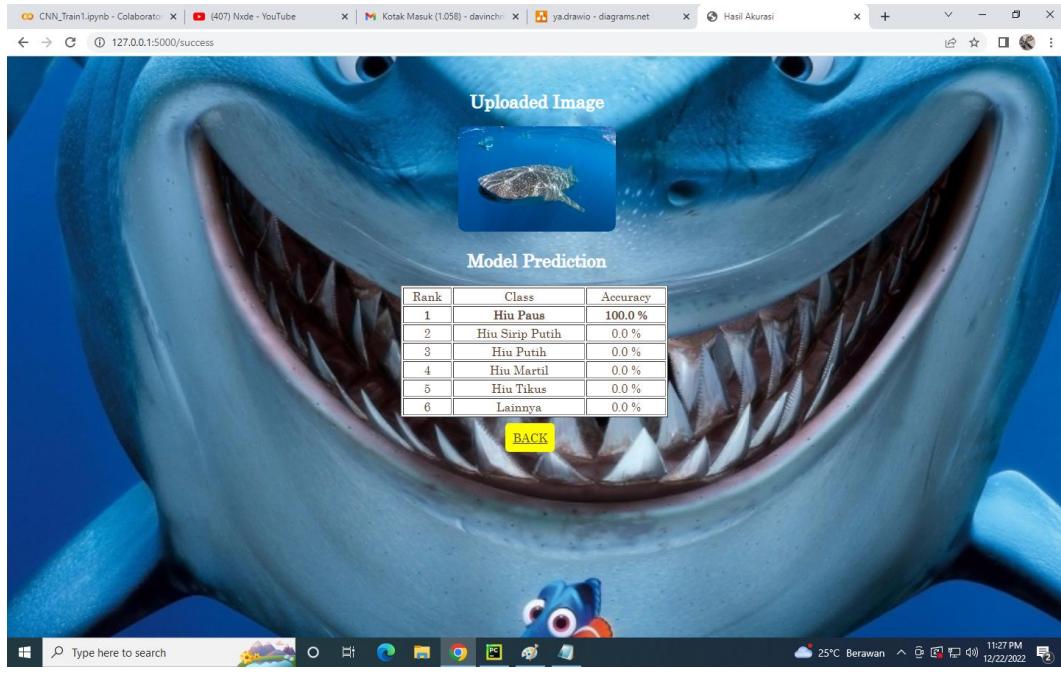


## Lampiran

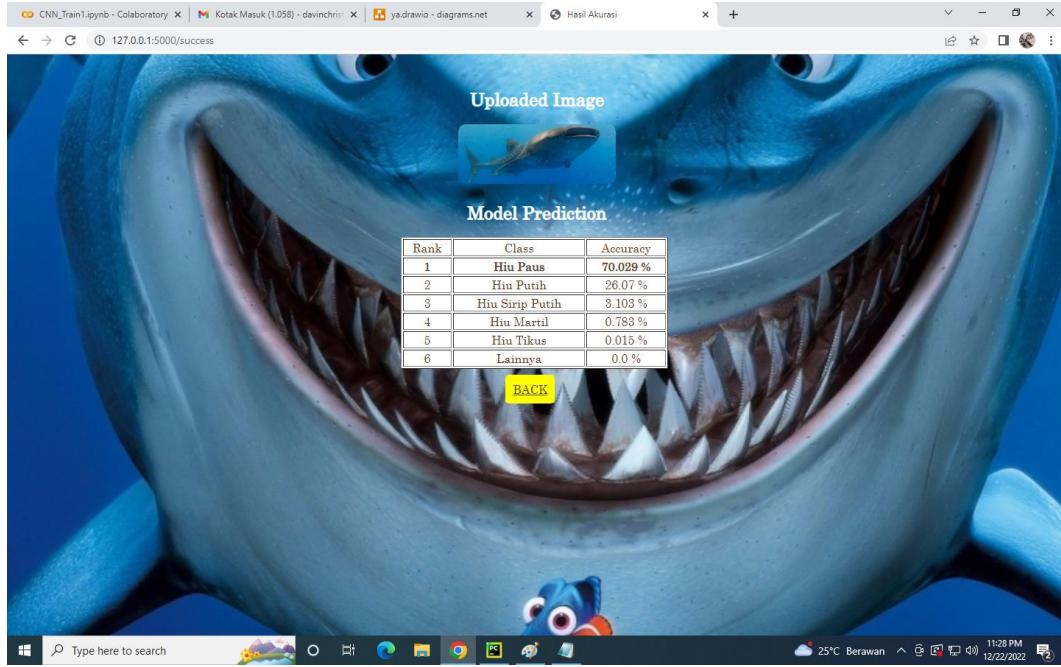
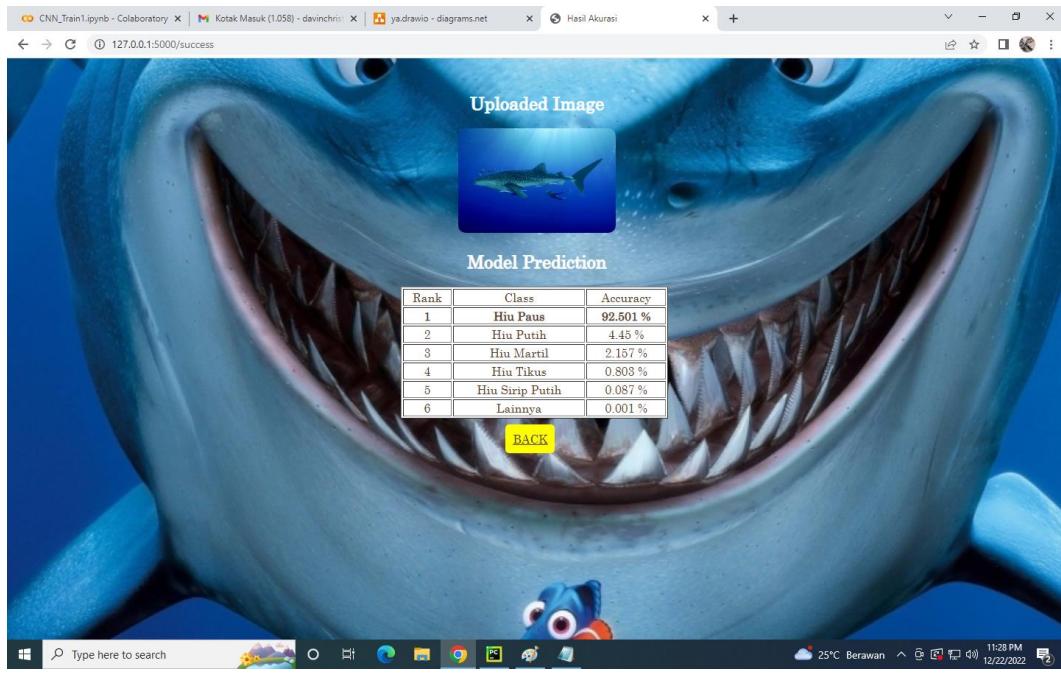




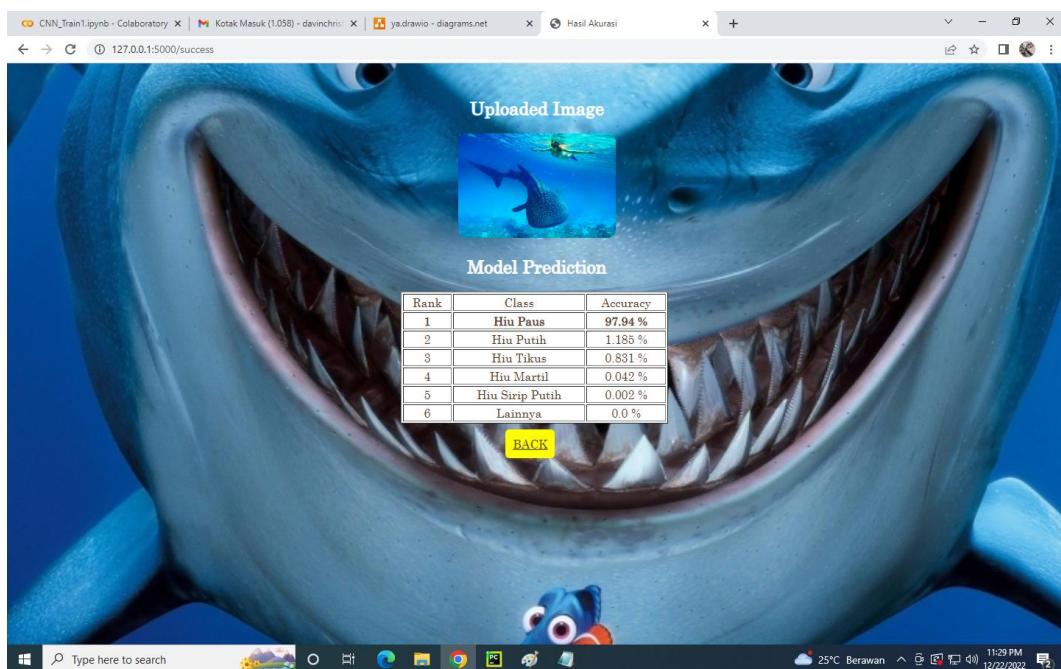
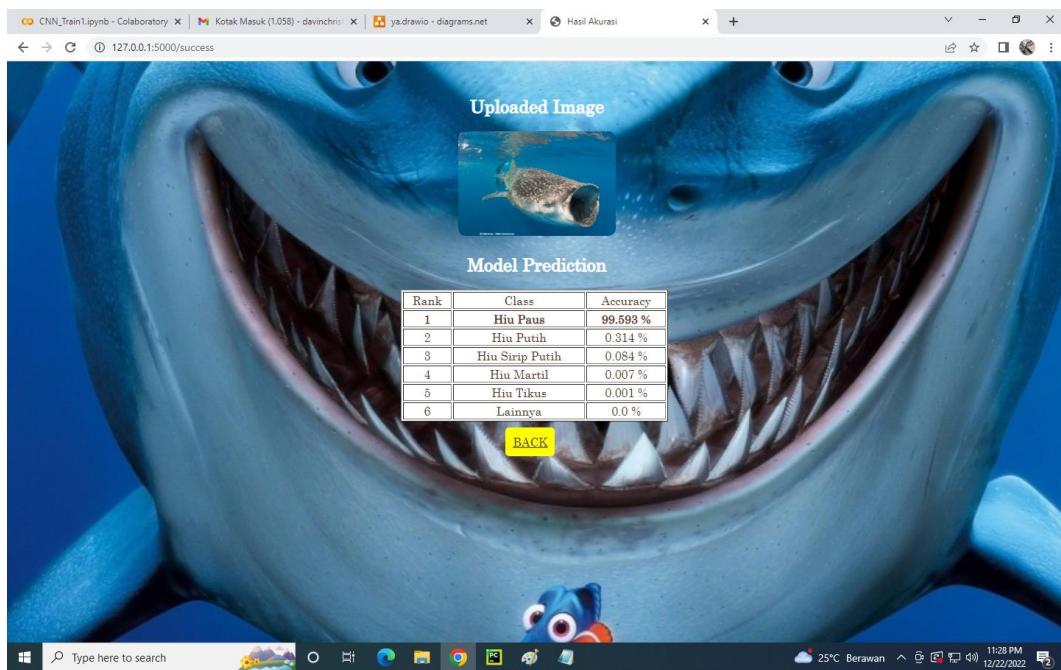
## Lampiran



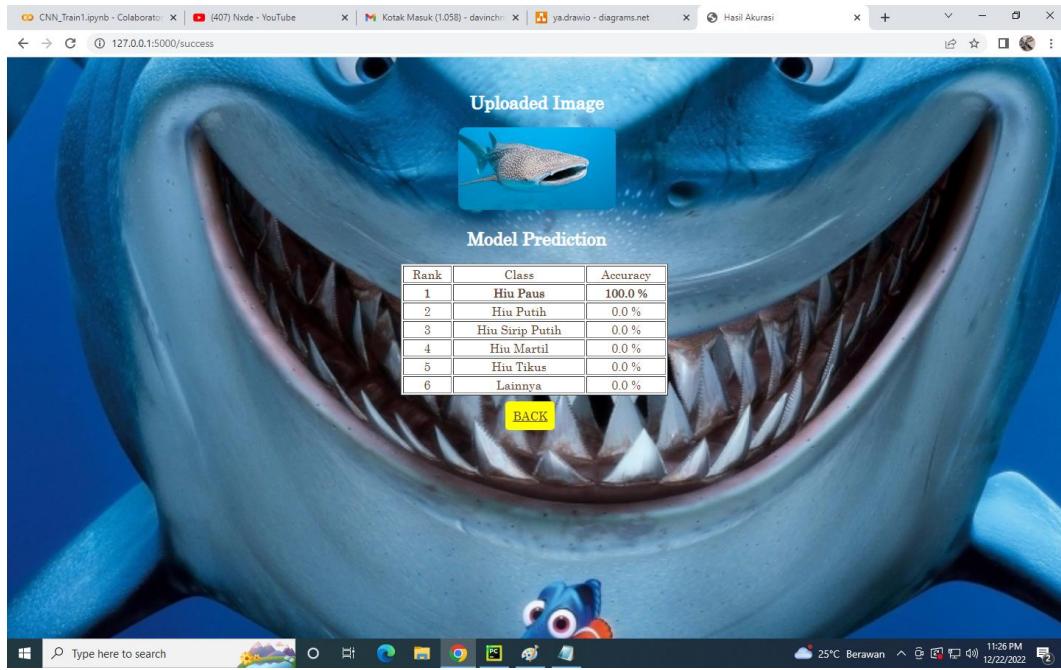
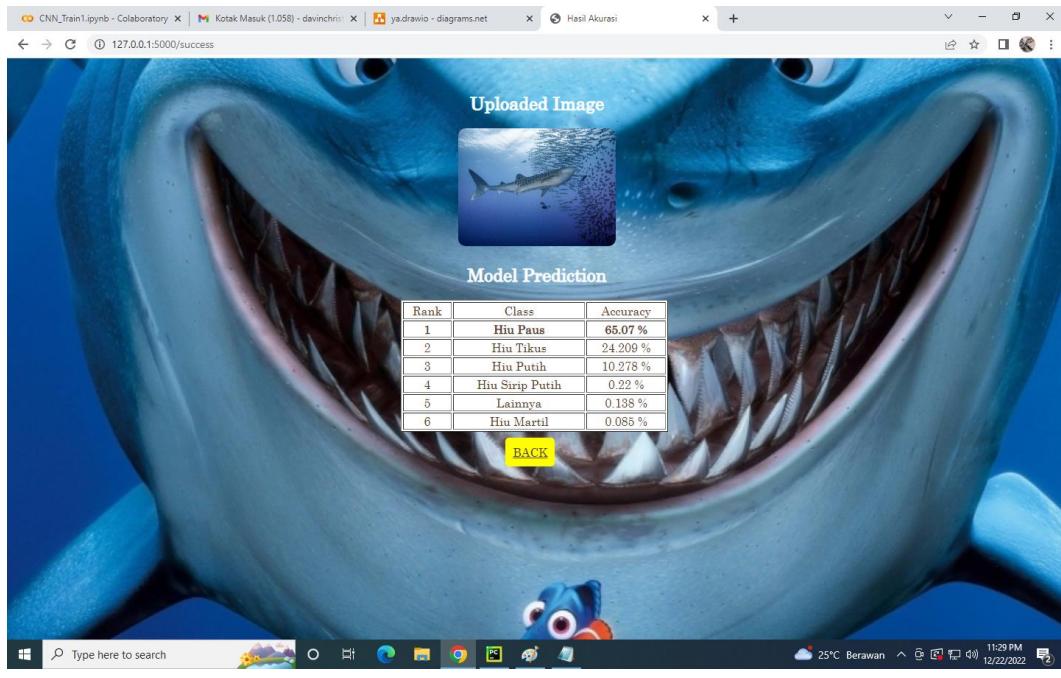
## Lampiran



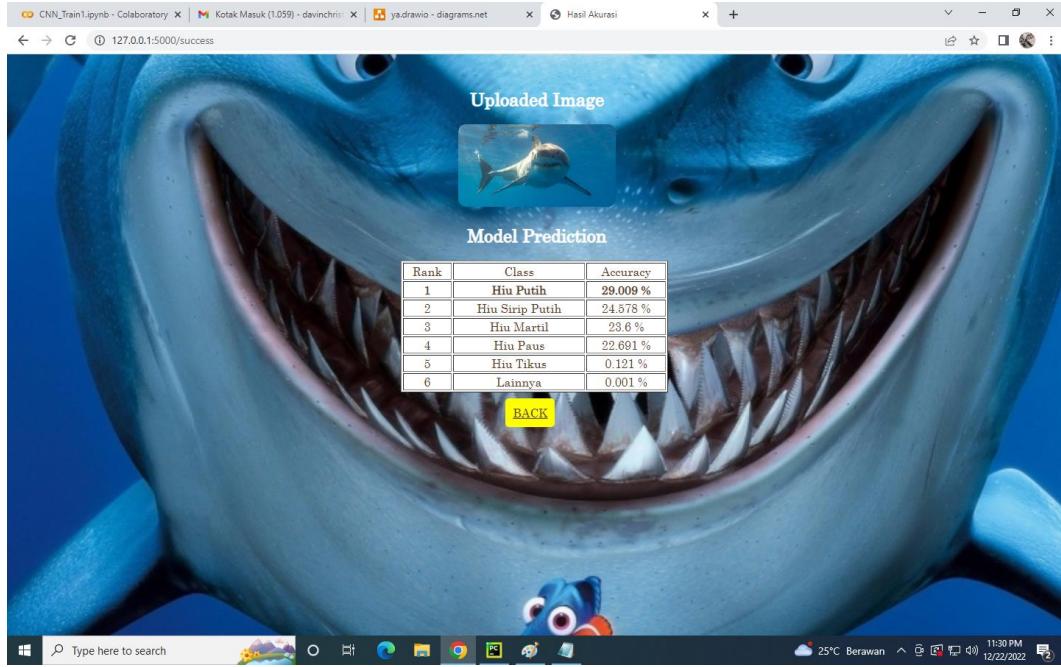
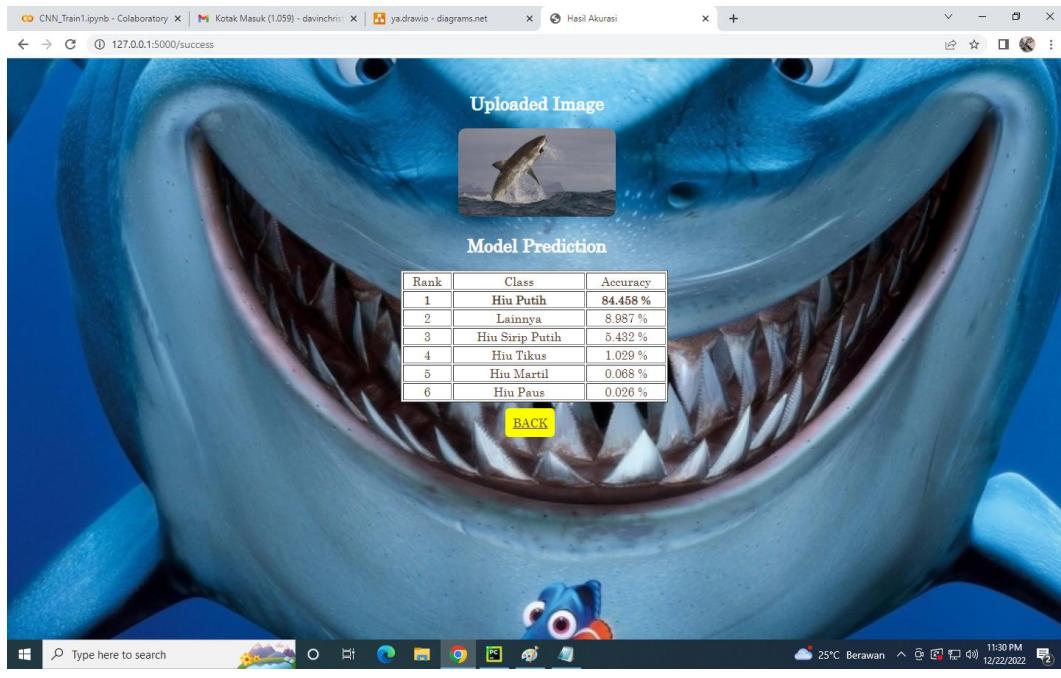
## Lampiran



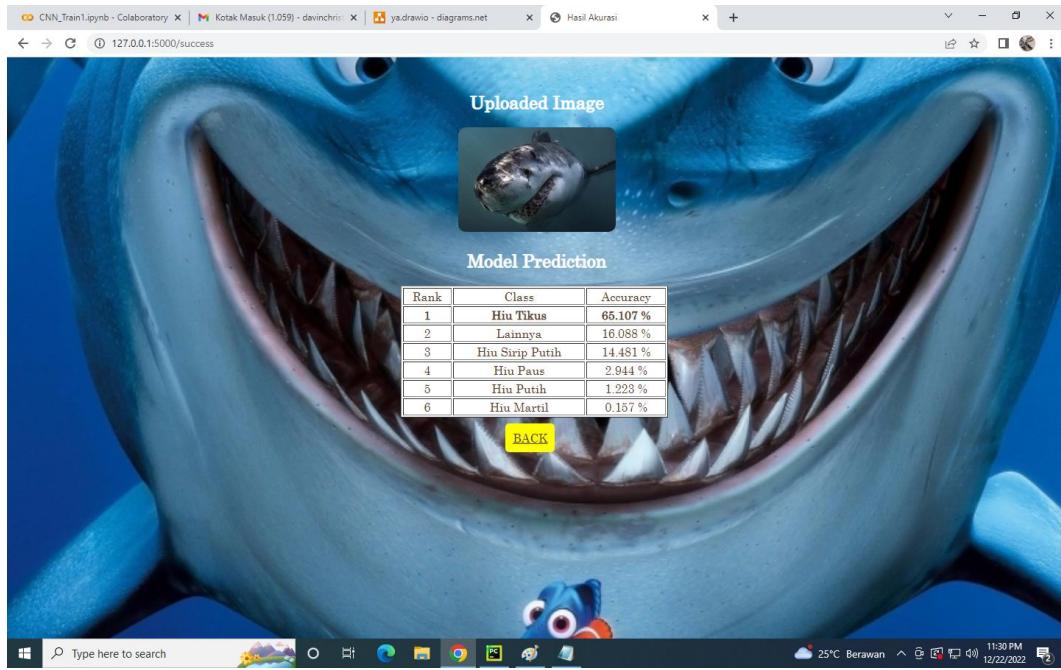
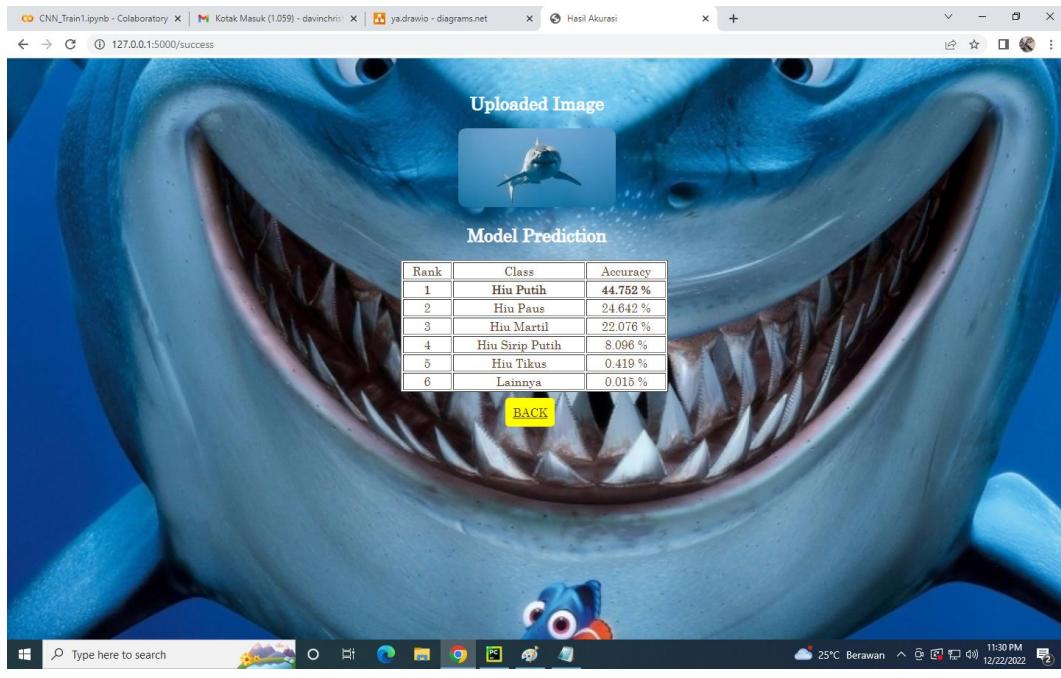
## Lampiran



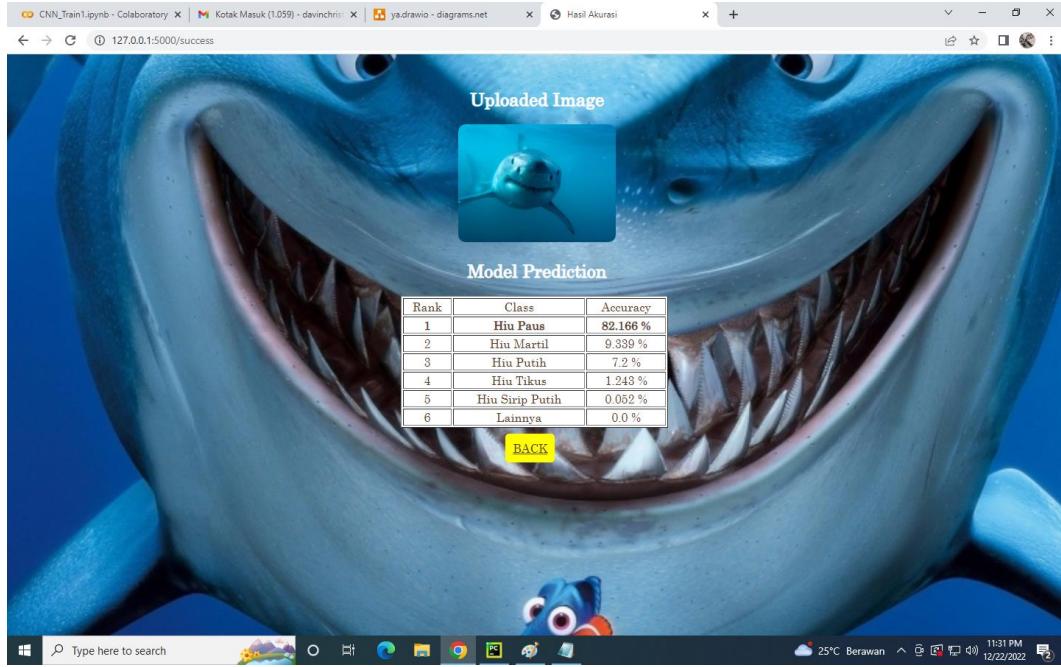
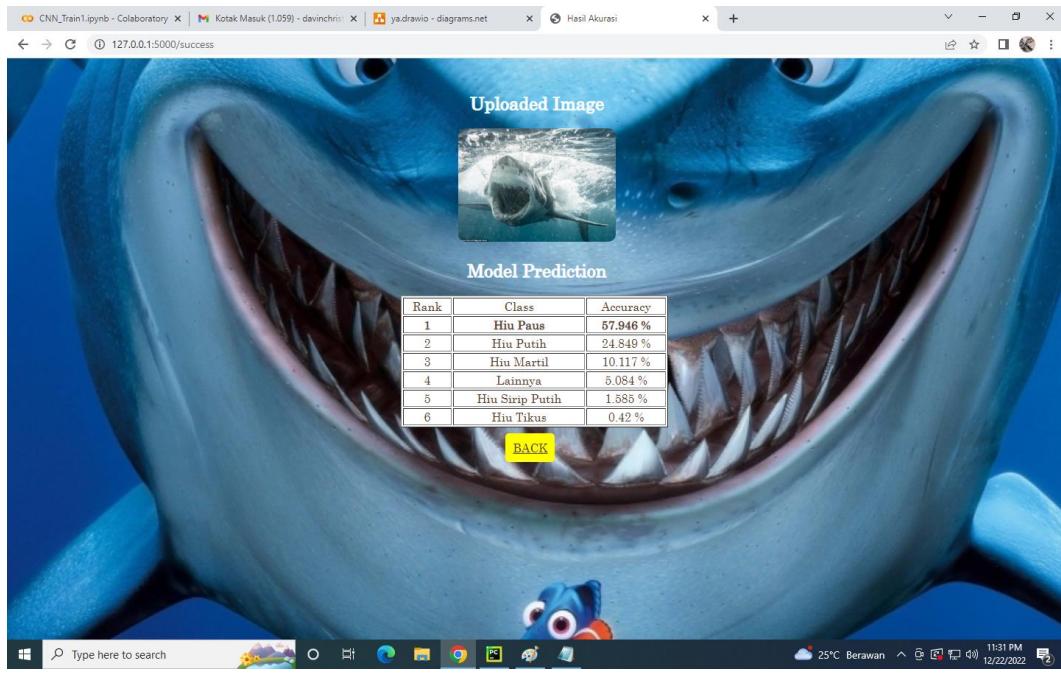
## Lampiran



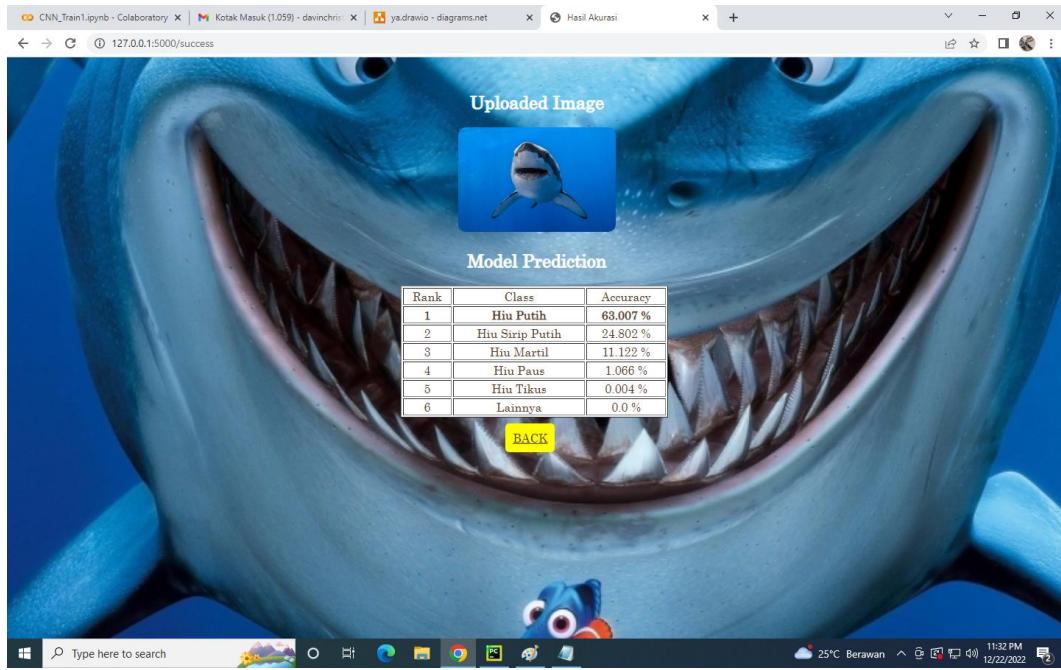
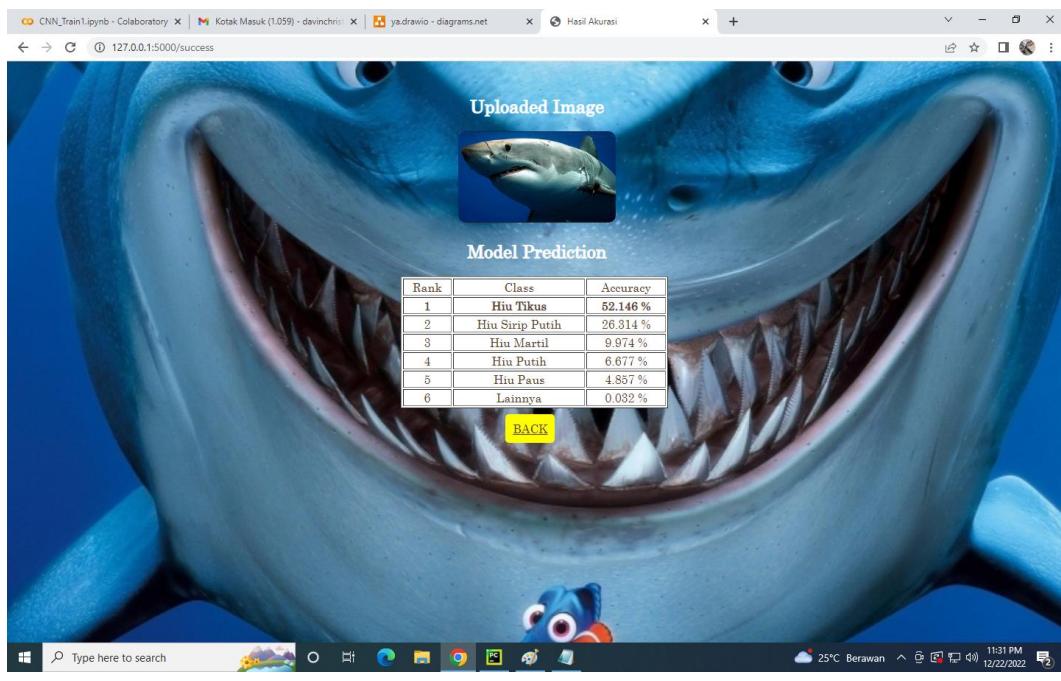
## Lampiran



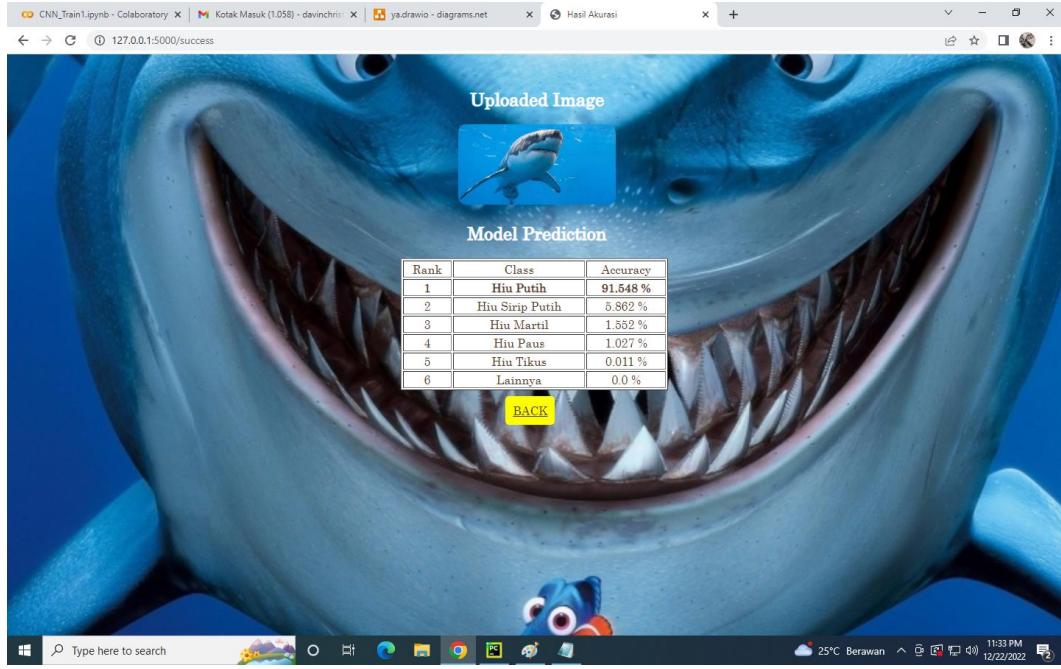
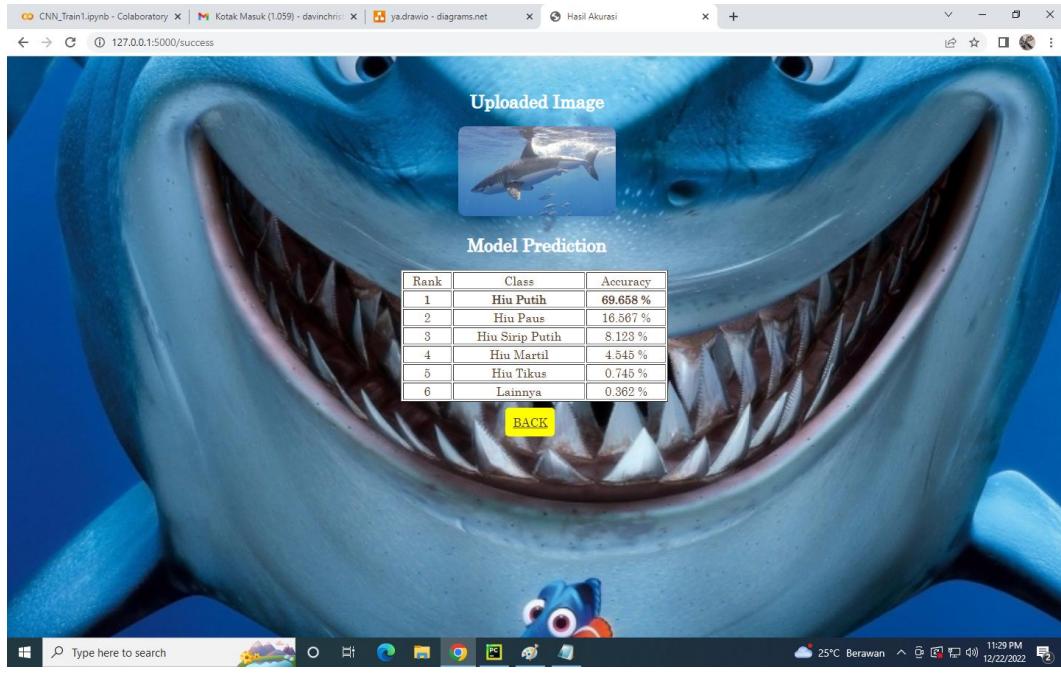
## Lampiran

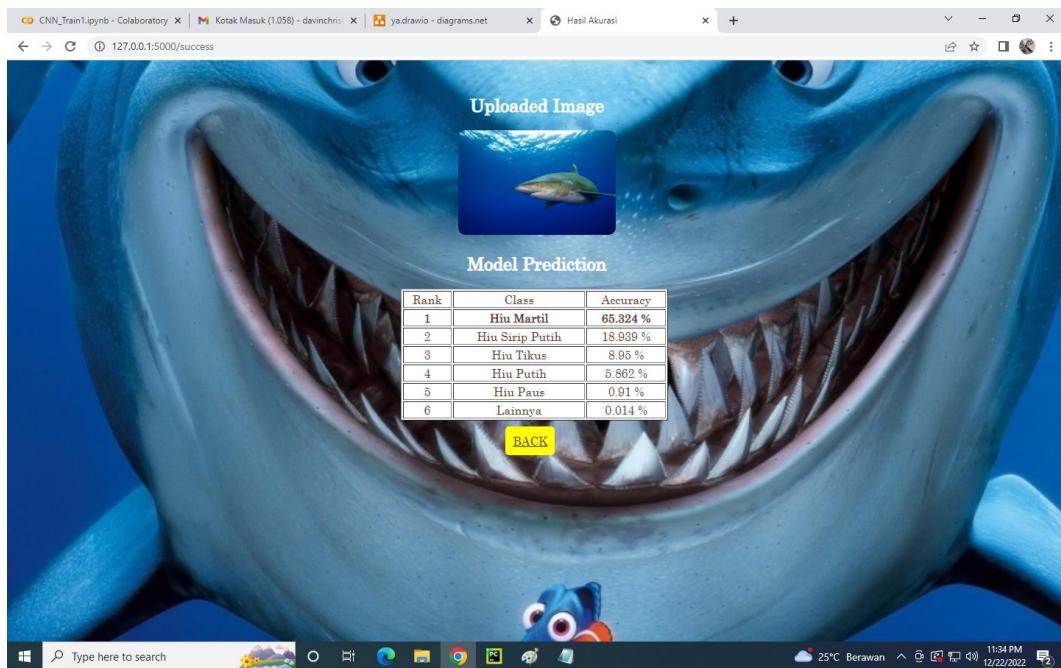
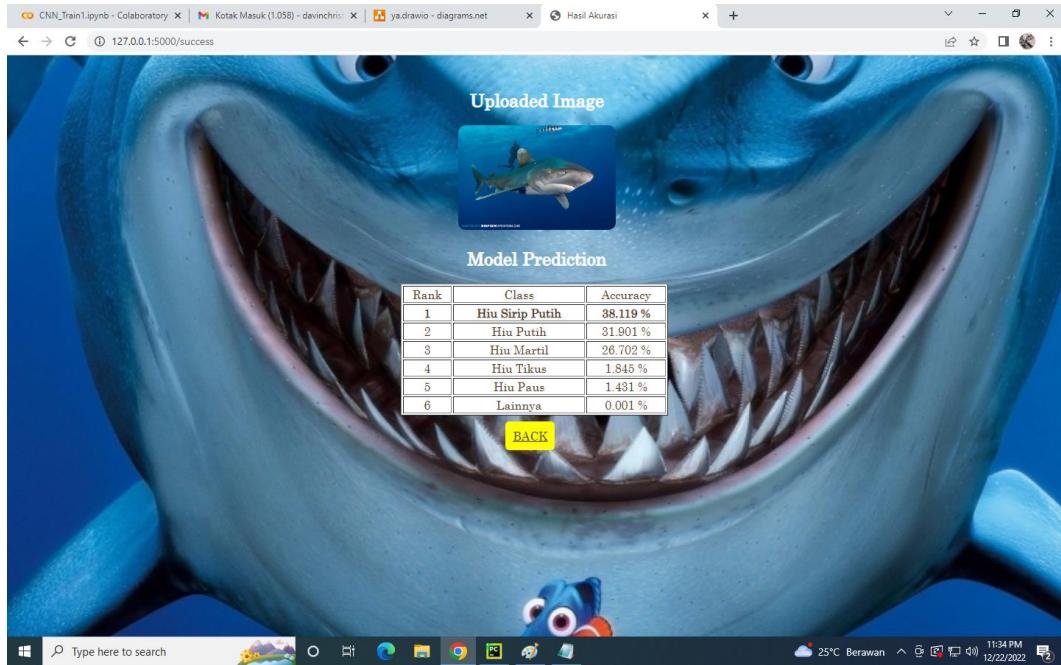


## Lampiran

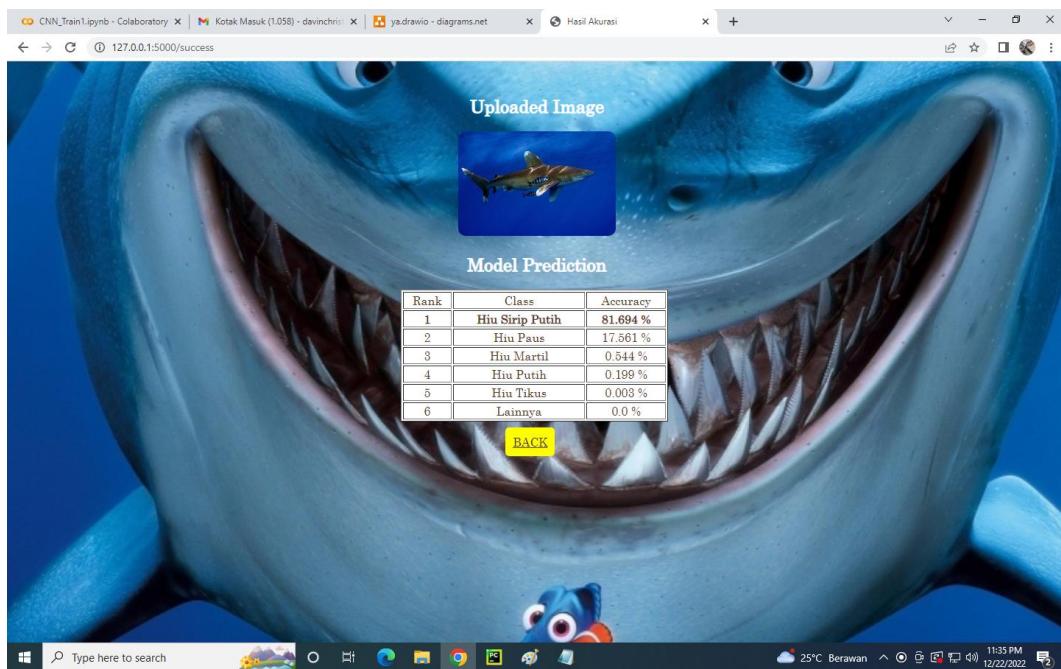
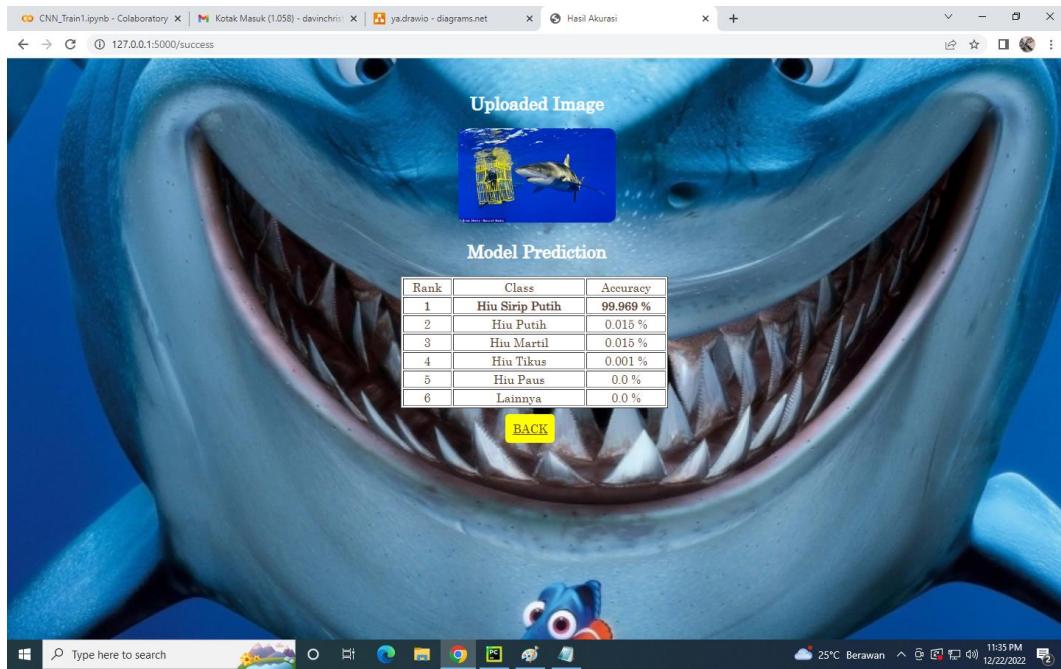


## Lampiran

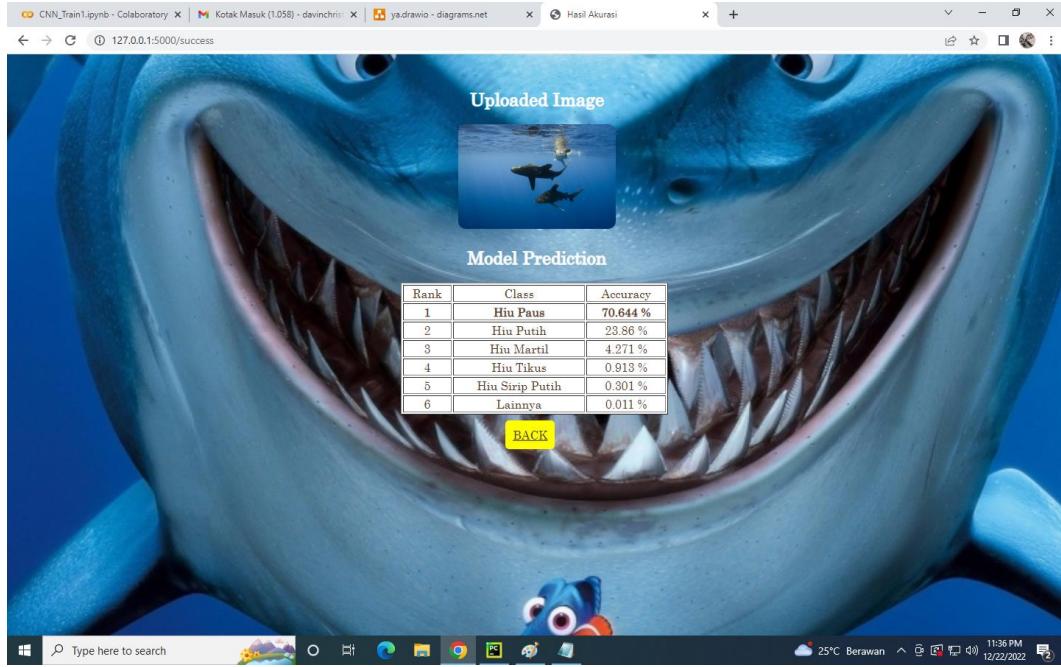
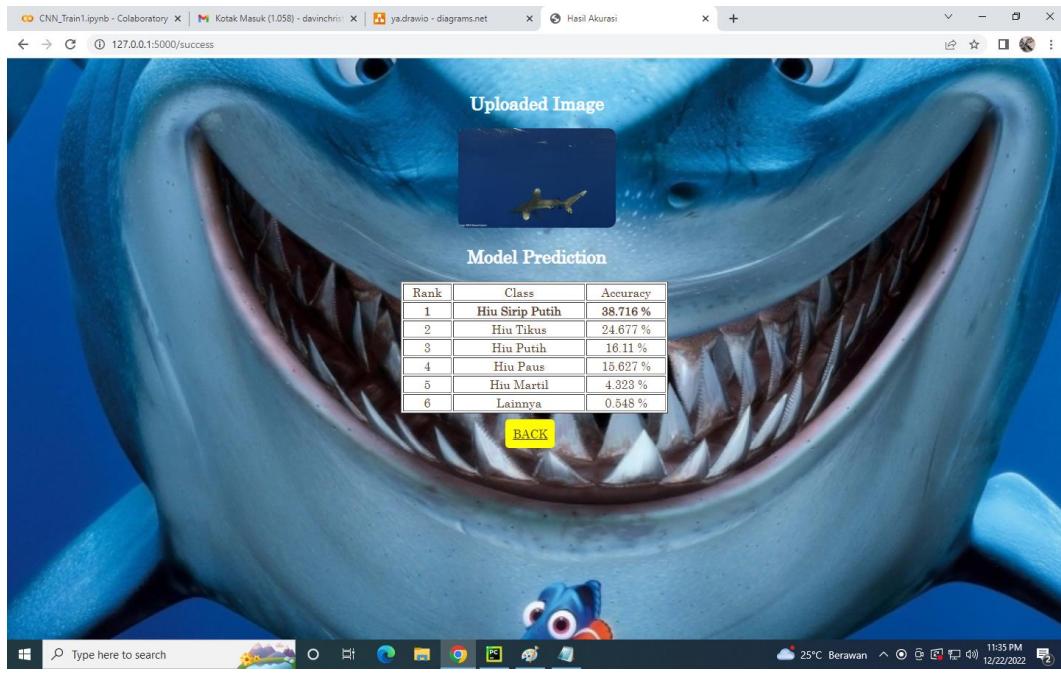




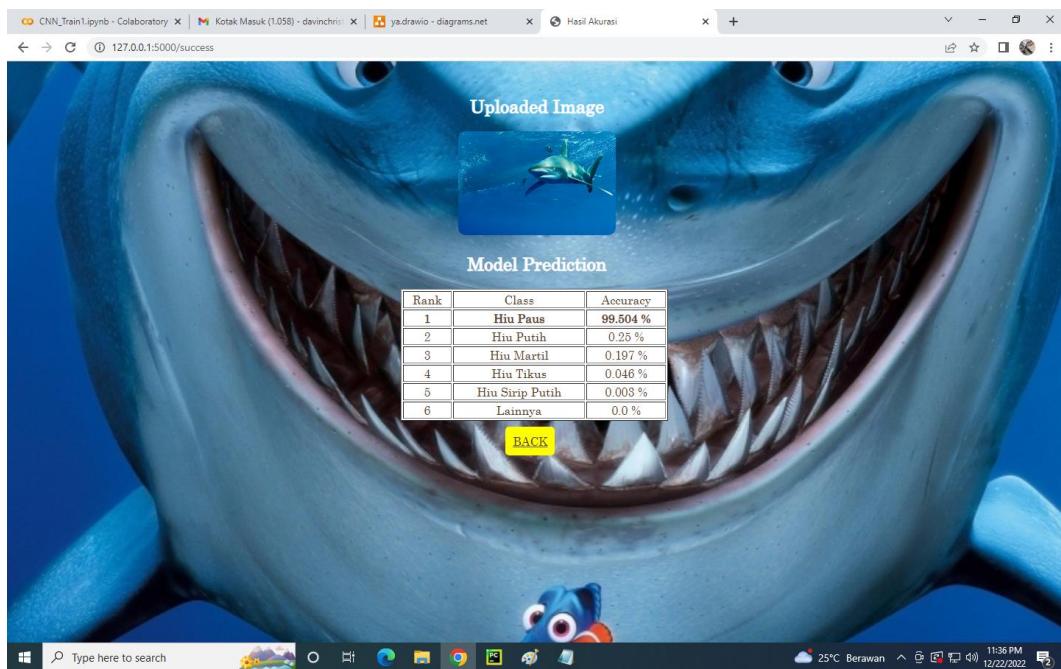
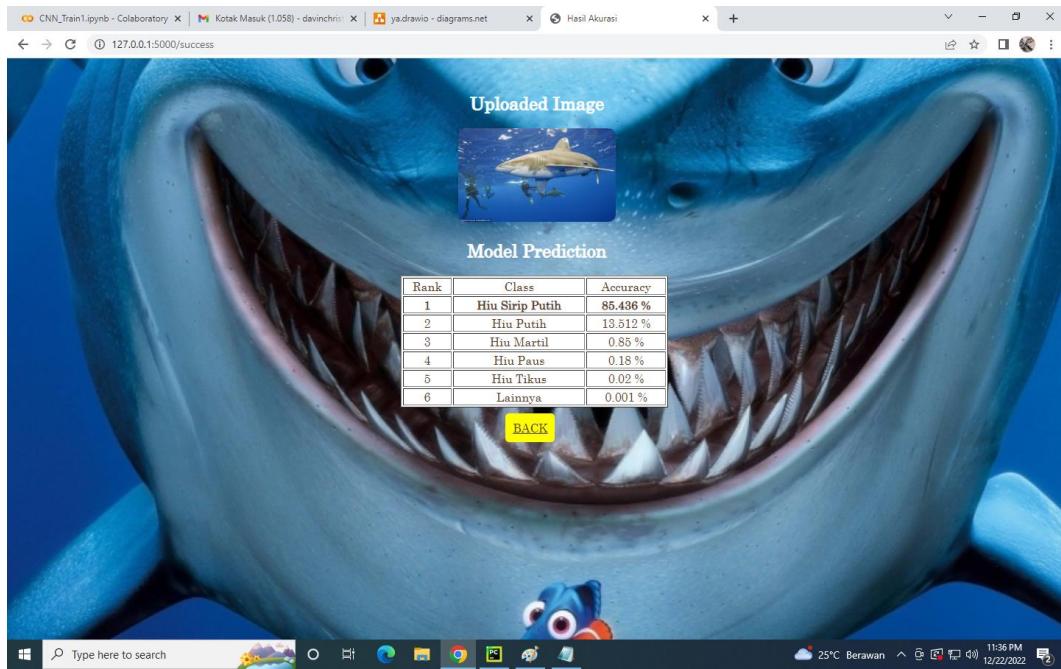
## Lampiran



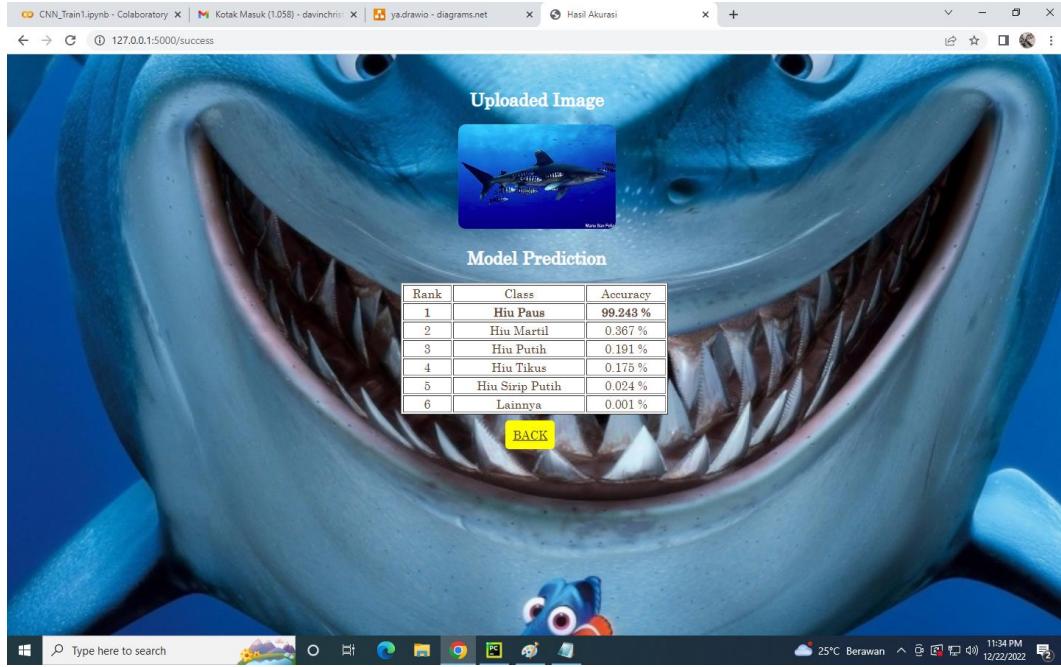
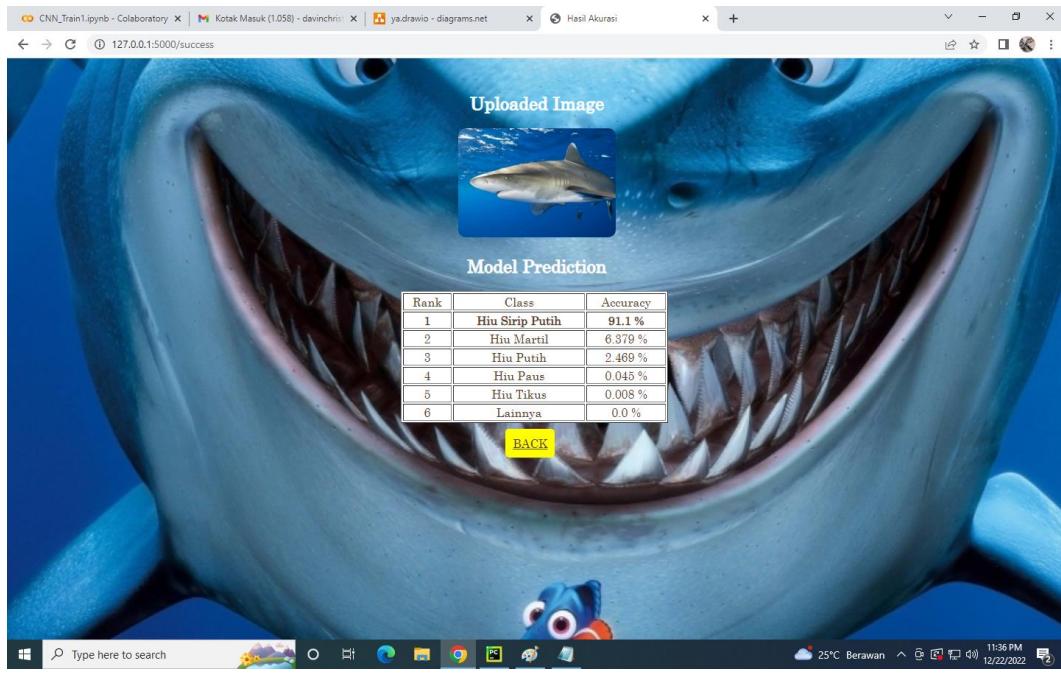
## Lampiran



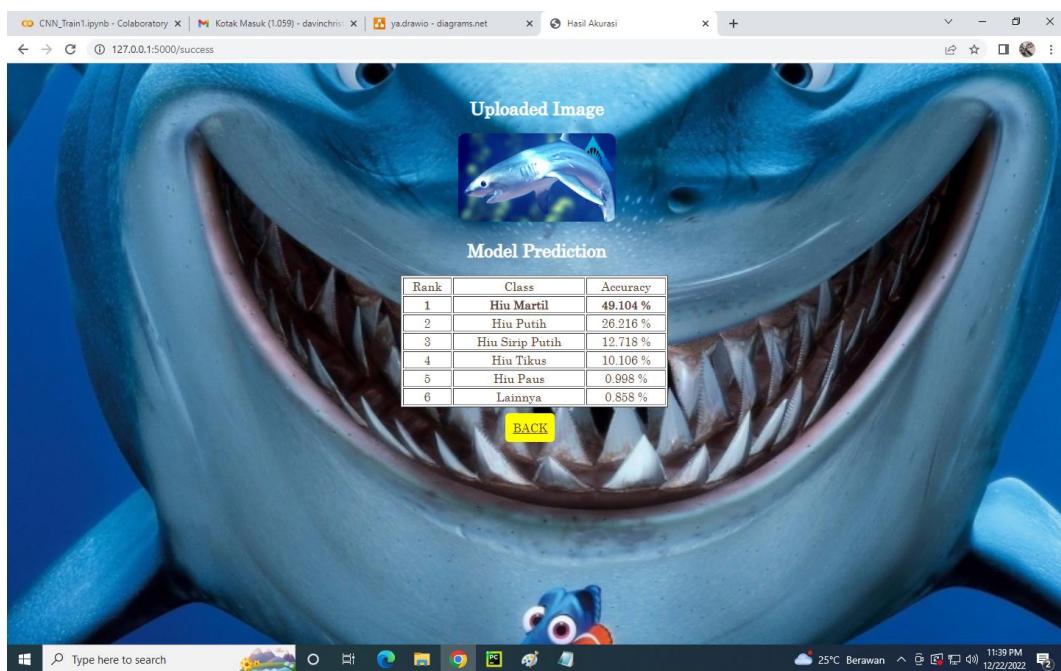
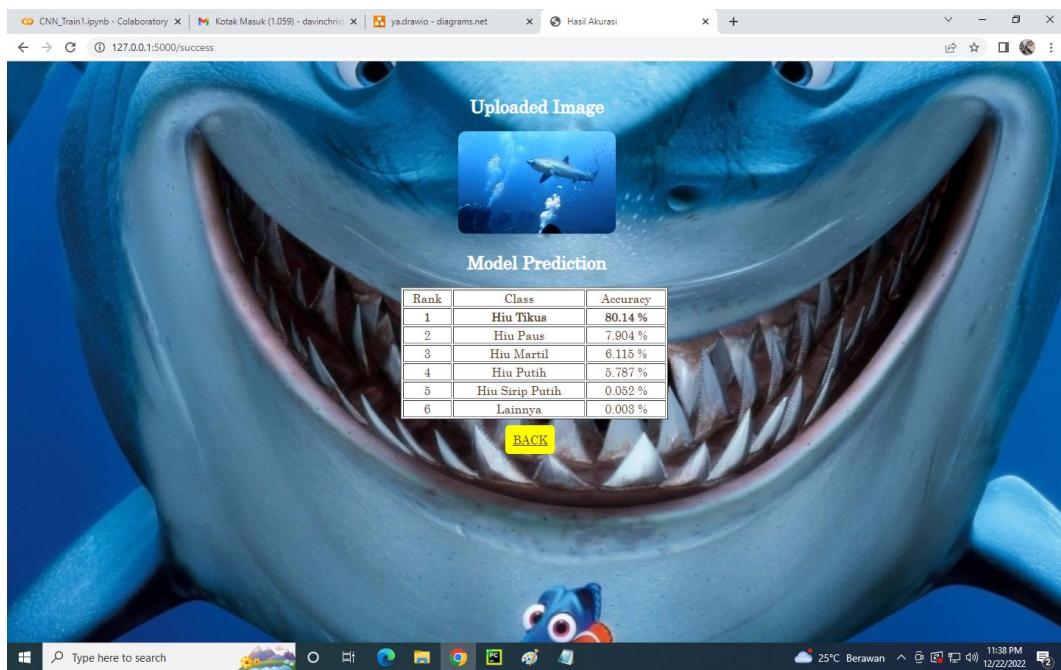
## Lampiran



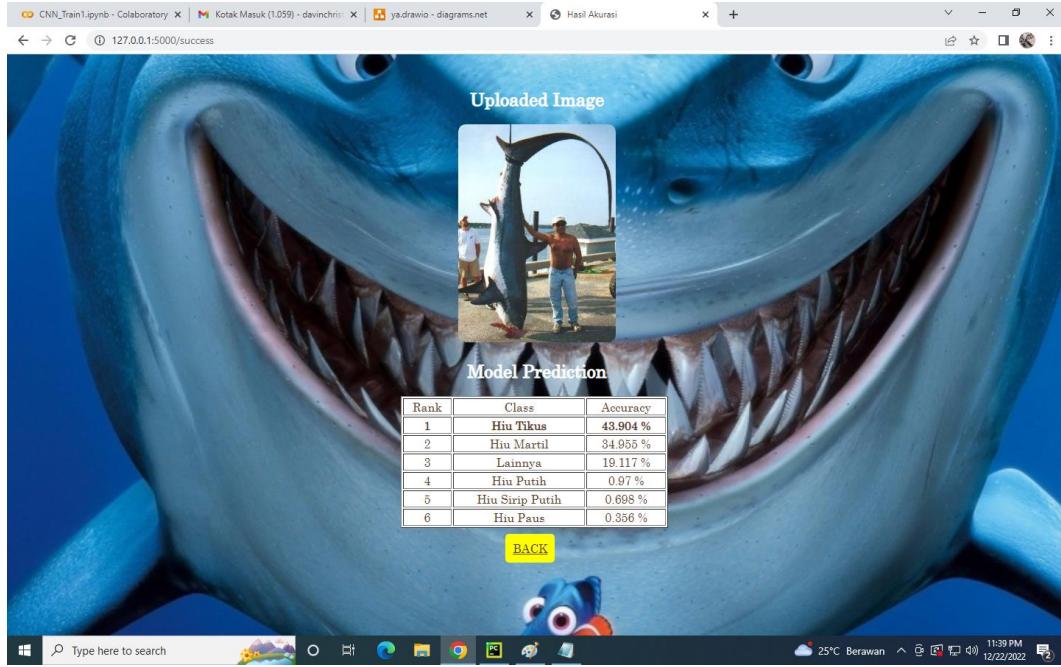
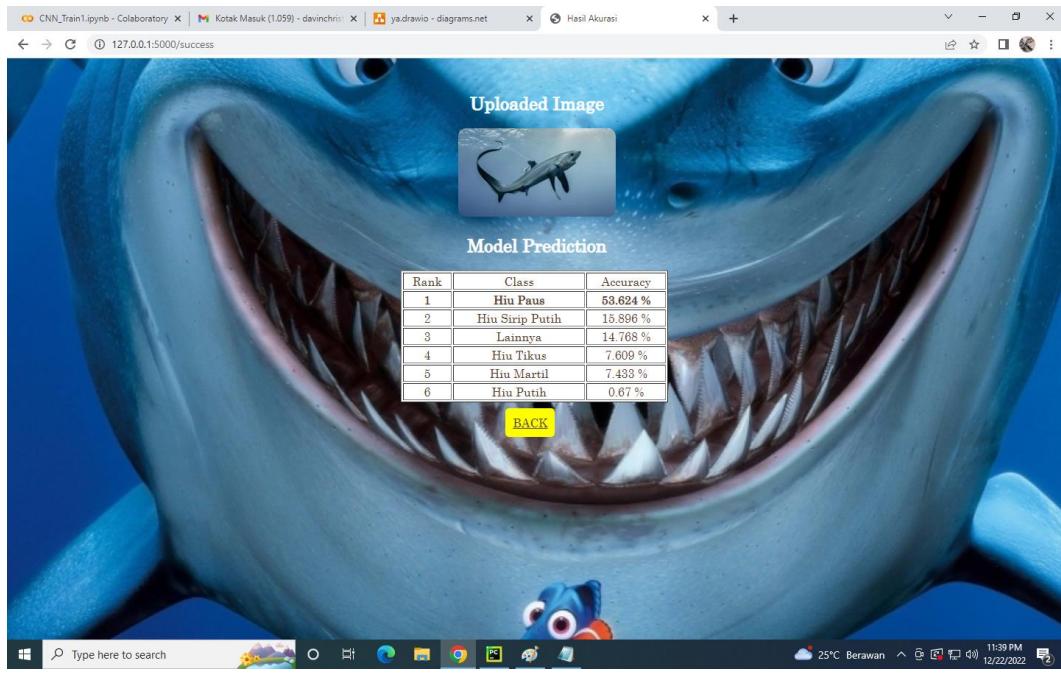
## Lampiran



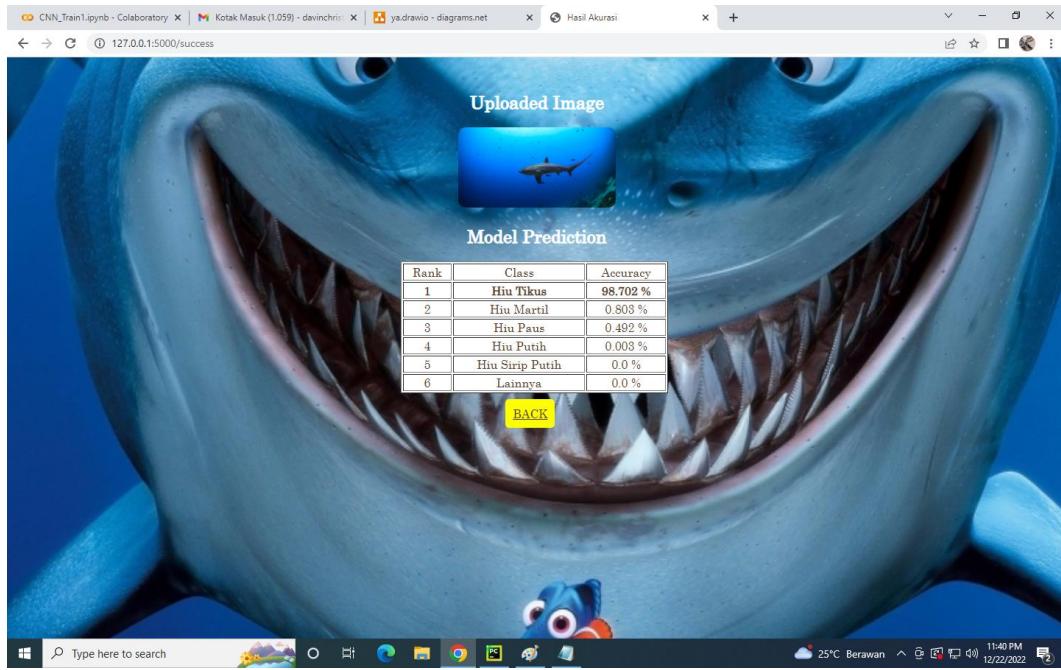
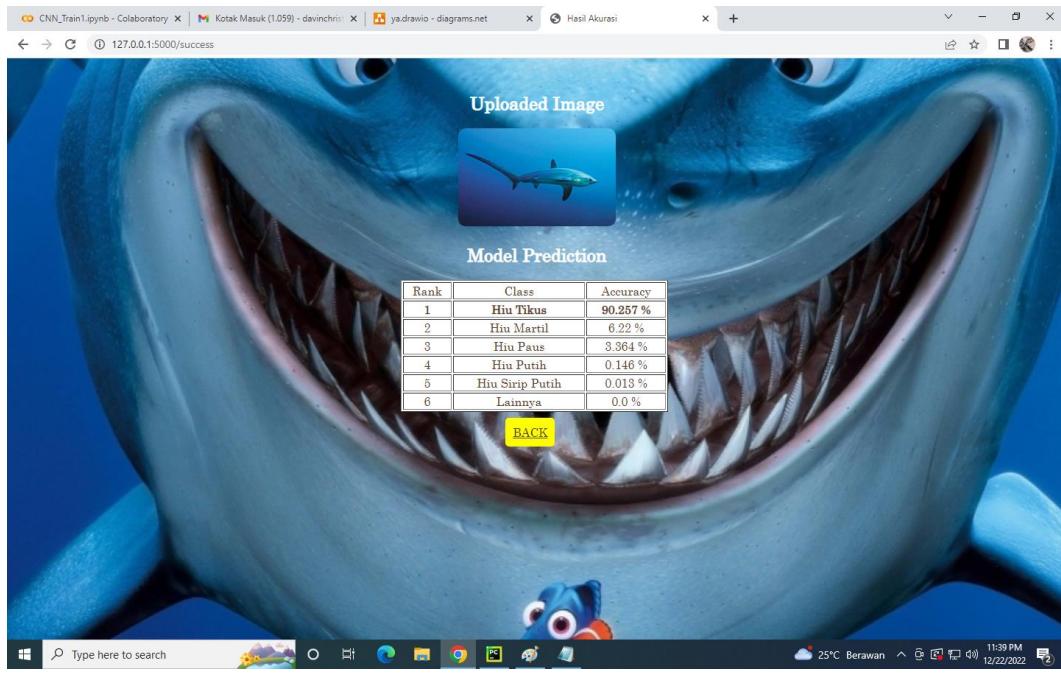
## Lampiran



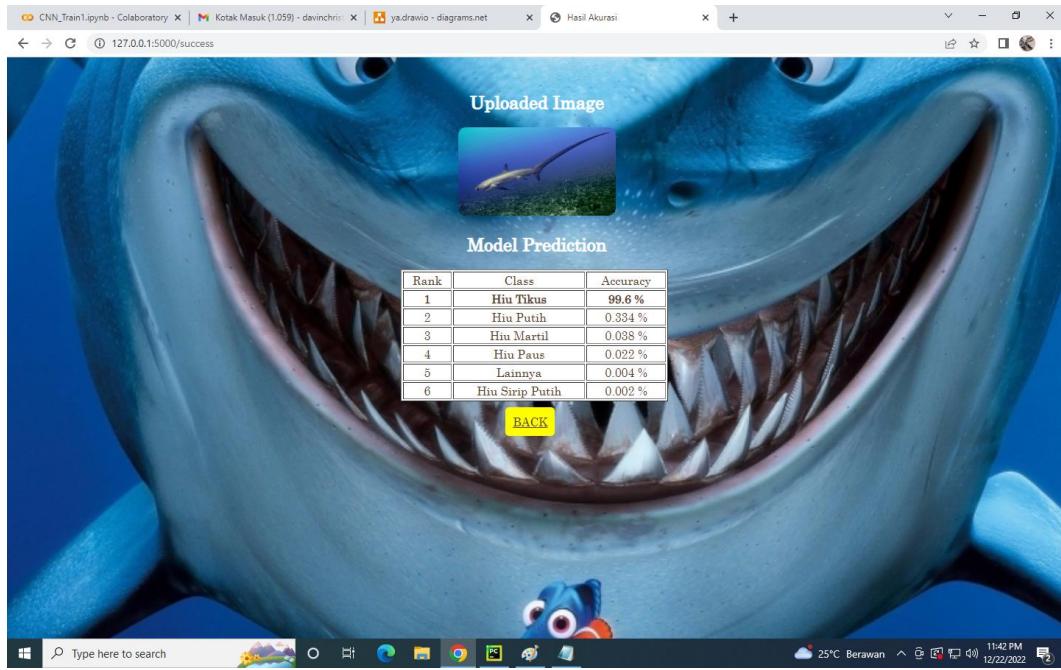
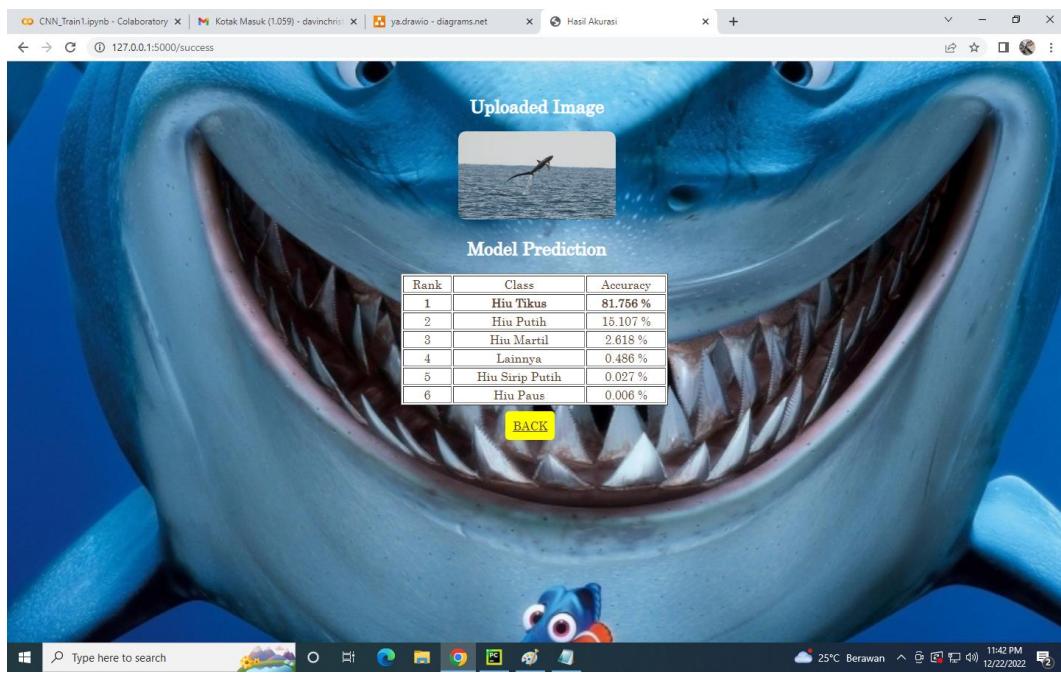
## Lampiran



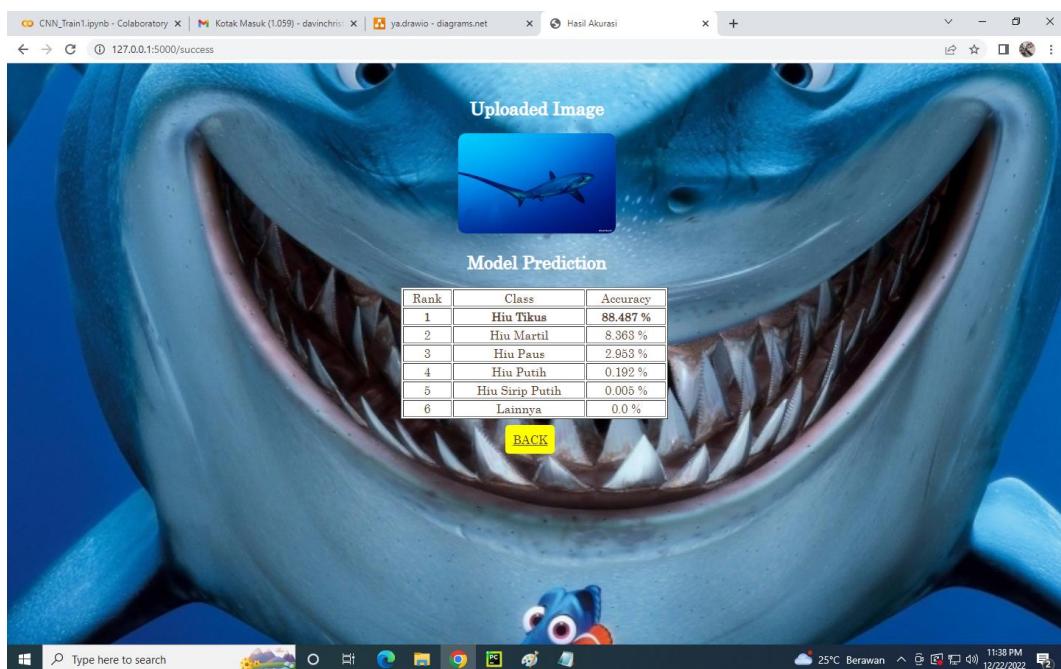
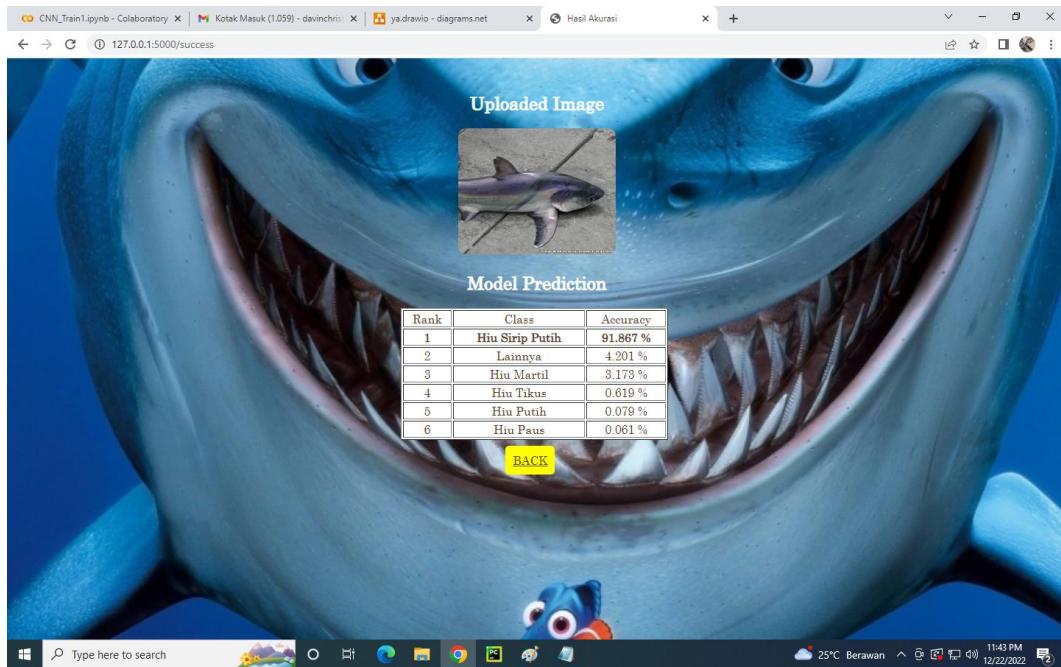
## Lampiran

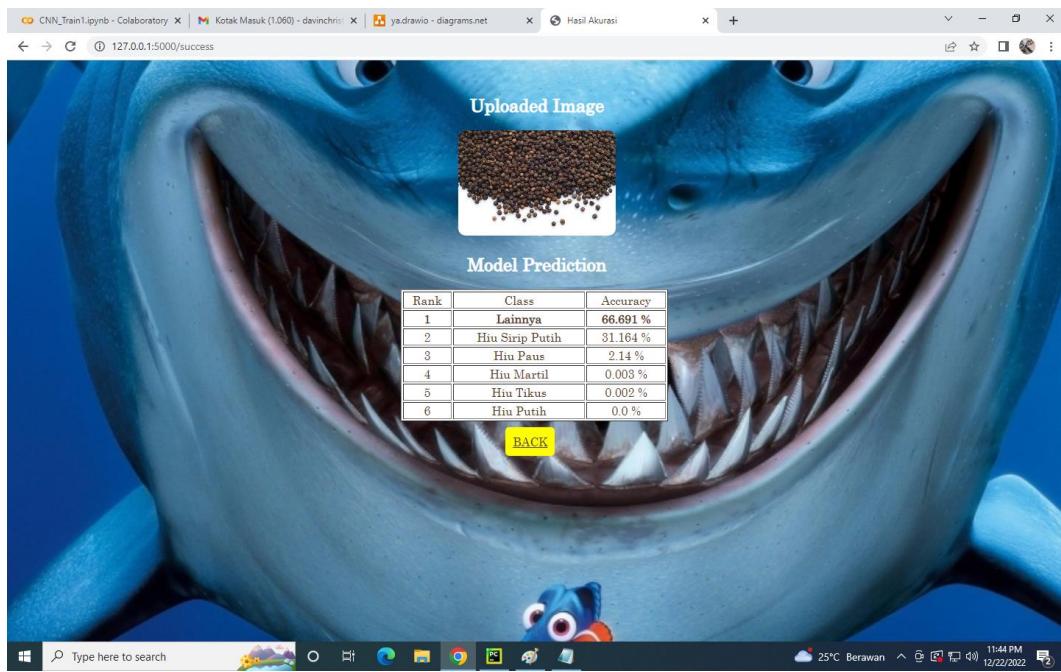
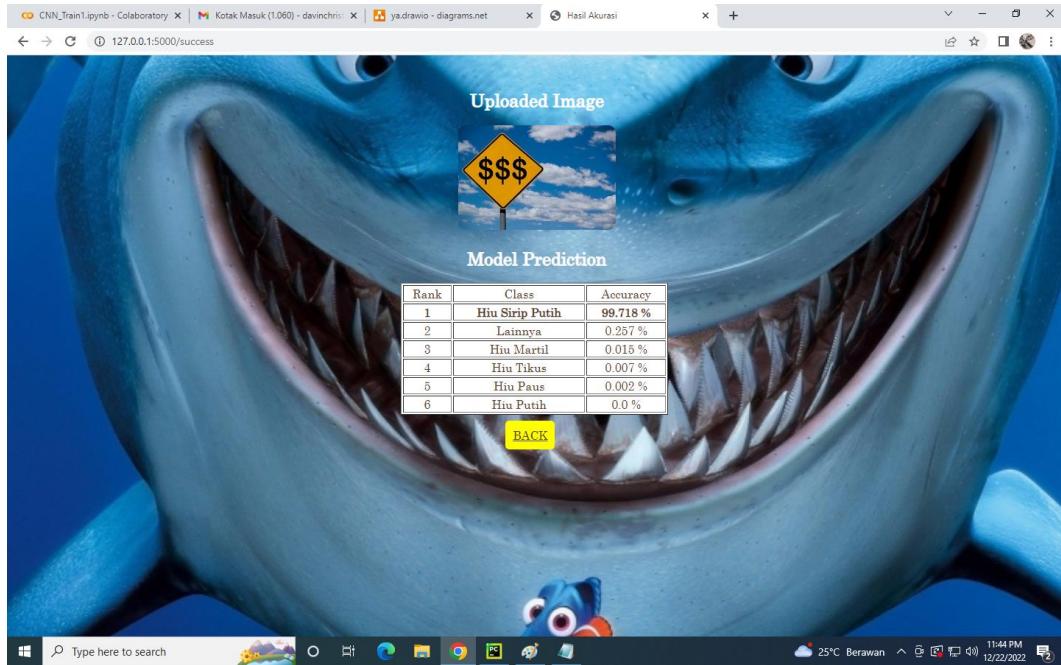


## Lampiran

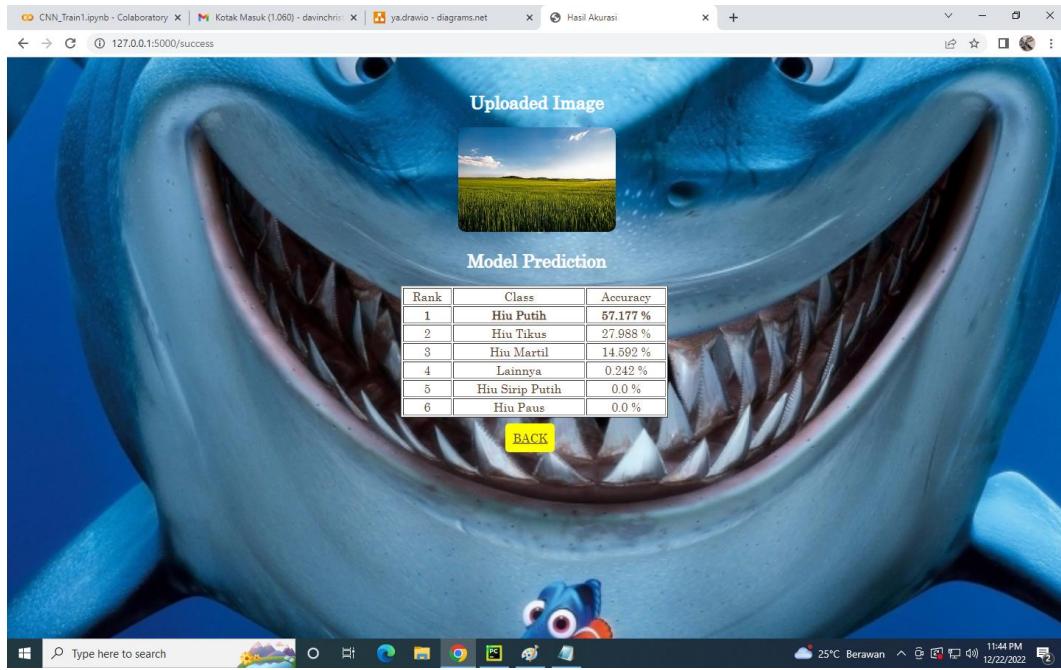
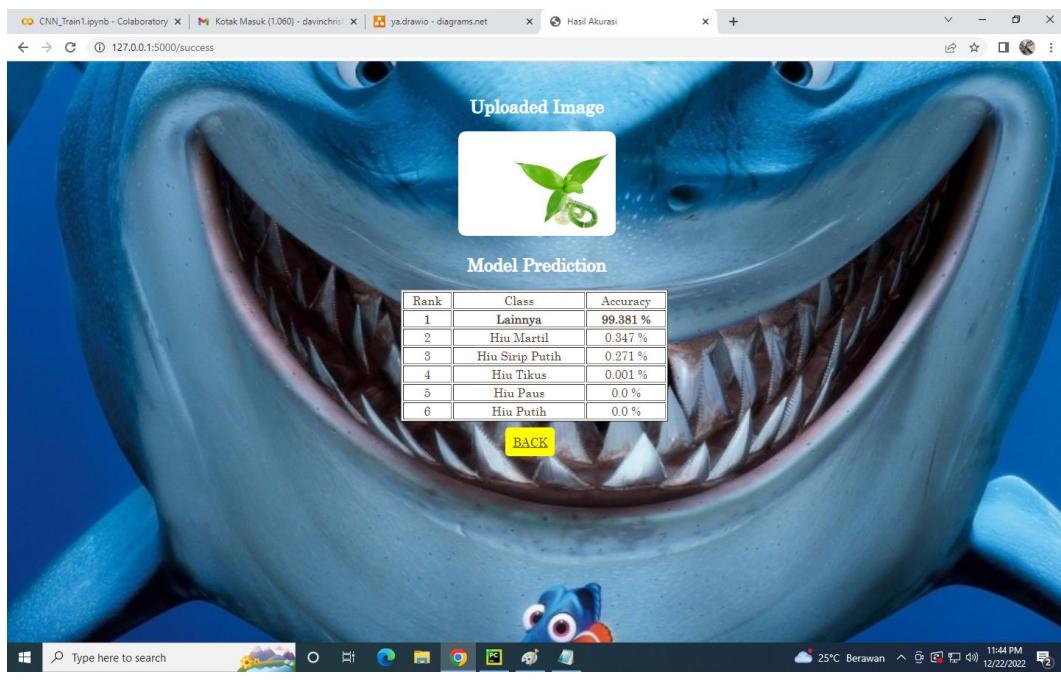


## Lampiran

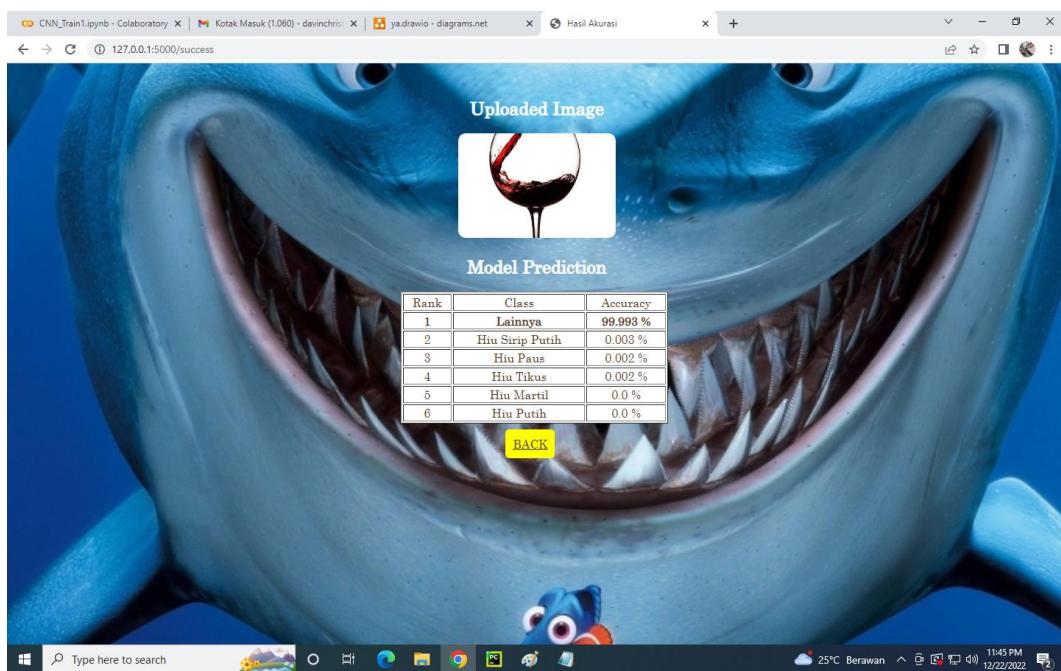
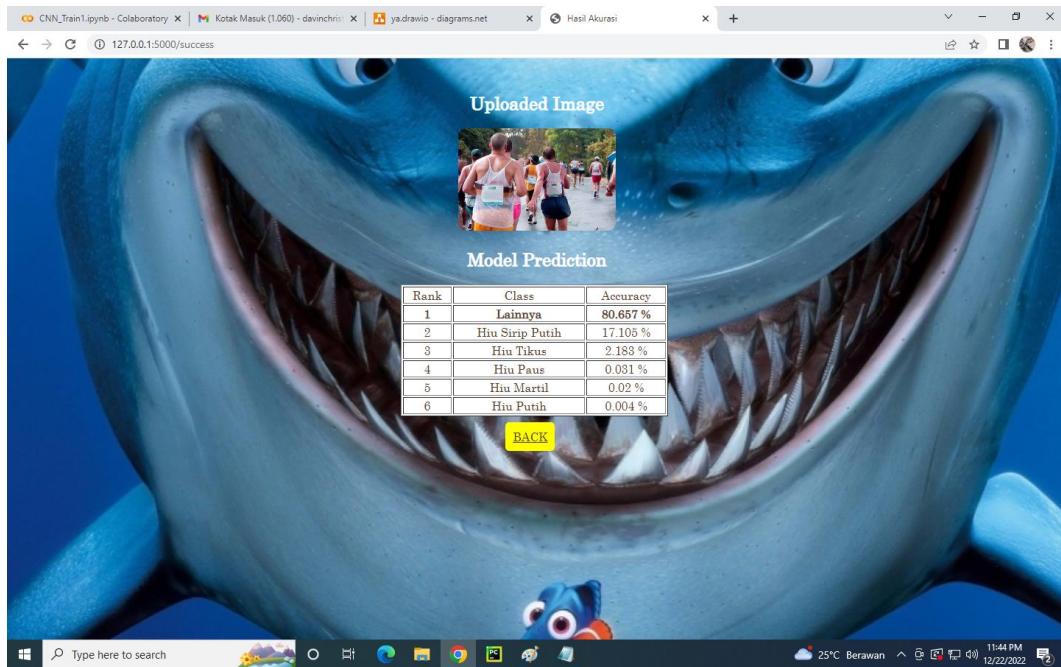




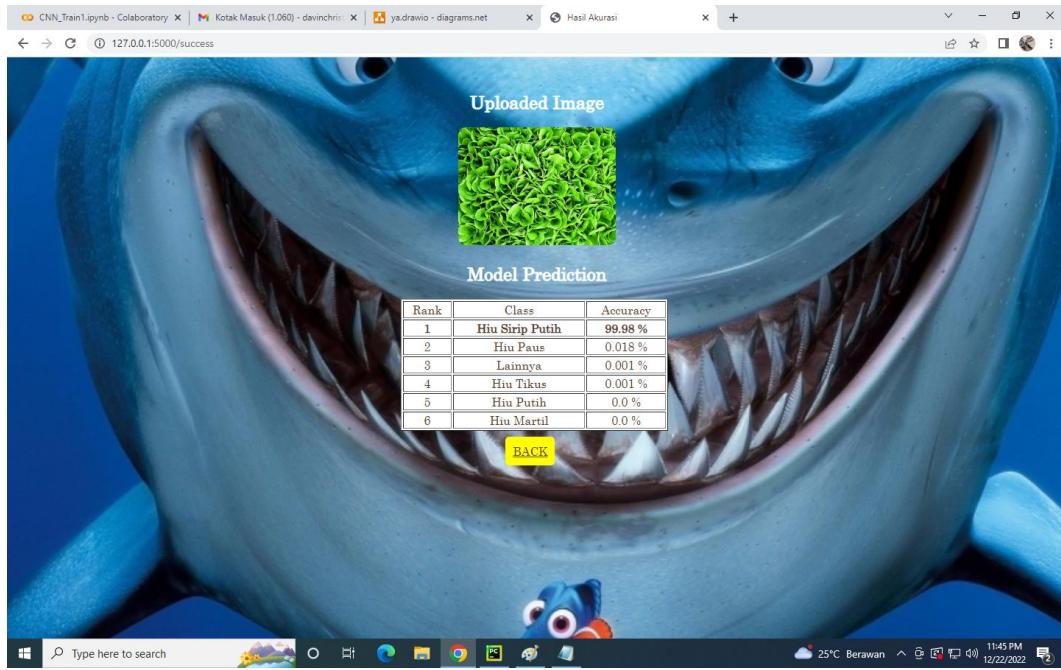
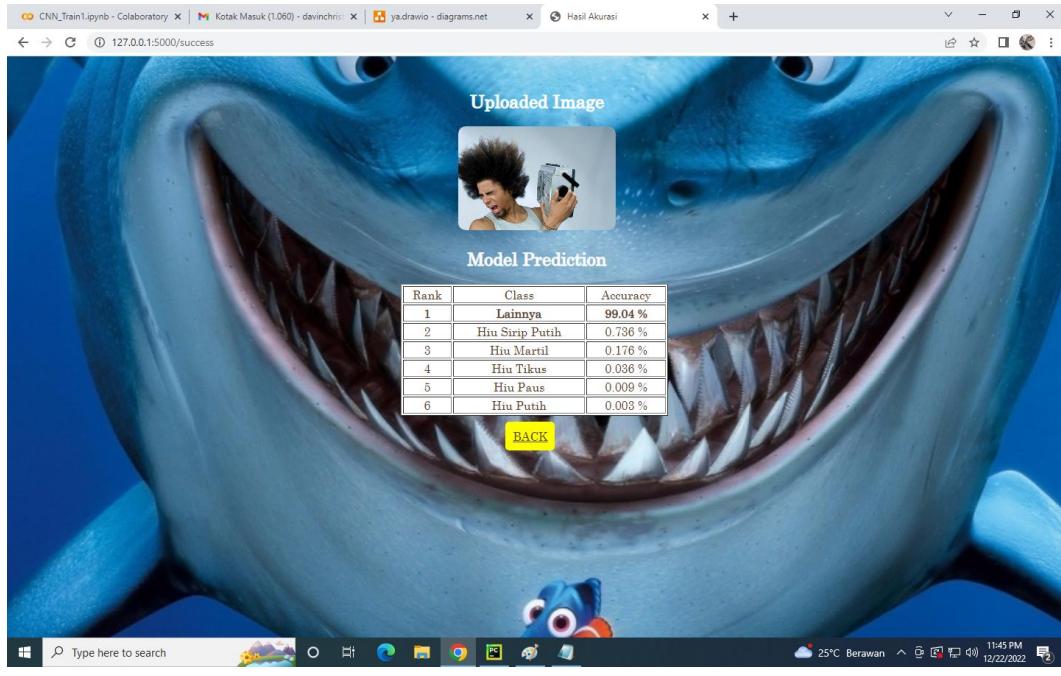
## Lampiran



## Lampiran



## Lampiran



## Lampiran

