PROJECT MASSIVE

ADVANCED ARTIFICIAL INTELLIGENCE

BATCH (6)

SVARAKAMA : KuyFutsal

Oleh:

Marco Philips Sirait Politeknik Negeri Batam

Edwin Prayoga H. Universitas Nasional Karangturi

Bagas Hilmi Arib Politeknik Negeri Batam

Noneng Ismaryanti Institut Teknologi Nasional bandung

ADVANCED ARTIFICIAL INTELLIGENCE

INFINITE LEARNING

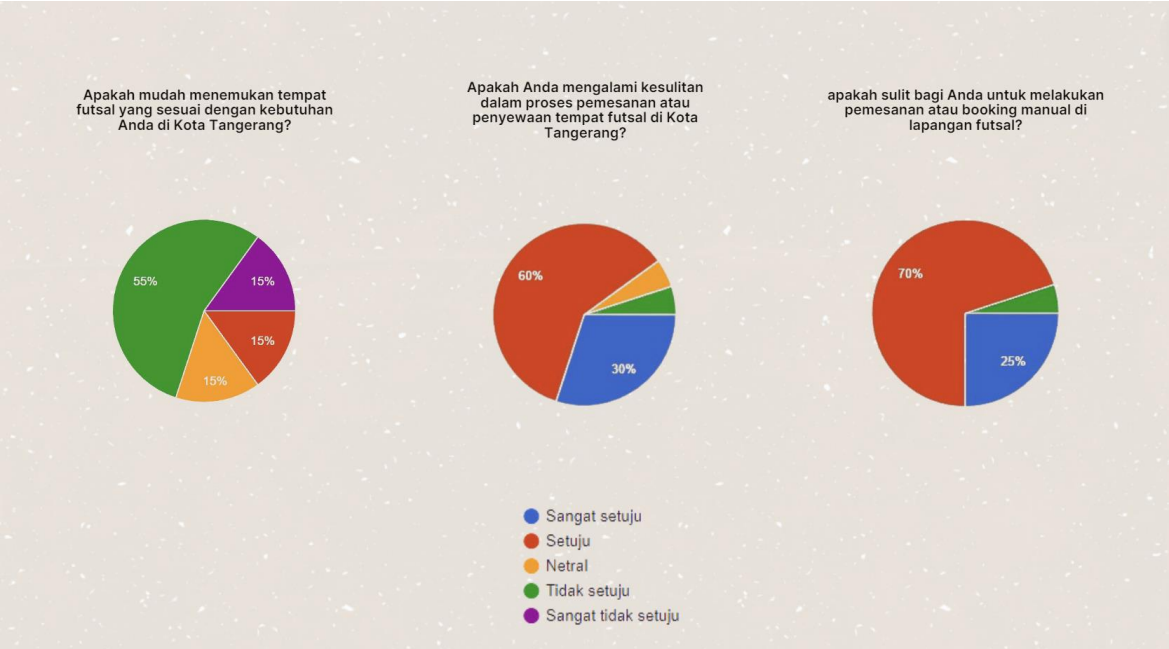
2024

A. Project Overview

1. Background

Awal mula kami dalam menentukan projek kami yaitu adalah sebuah produk digital sebagai solusi dalam permasalahan sehari-hari. Dari tujuan awal tersebut di temukanlah sebuah ide media penyewaan lapangan futsal berbasis website yang bernama “KuyFutsal”. Alasan kami dalam memilih ide tersebut karena kurangnya informasi penyedia lapangan futsal yang ada. Selain itu, kompetitor sejenis penyewaan lapangan olahraga juga tergolong sedikit sehingga membuat kami lebih yakin dalam membuat website penyewaan lapangan yang berfokus pada olahraga futsal..

Berdasarkan jurnal-jurnal, hasil analisis faktor menunjukkan bahwa terdapat 5 faktor yang biasa dijadikan pertimbangan sebelum melakukan sewa lapangan olahraga, yaitu Faktor Harga, Faktor Promosi, Faktor Fasilitas, Faktor Tersedianya Kontak Person, dan Faktor Lokasi. Faktor Harga, Faktor Promosi, Faktor Fasilitas, Faktor Tersedianya Kontak Person, dan Faktor Lokasi secara bersama-sama berpengaruh signifikan terhadap Keputusan Sewa Lapangan Olahraga dan pengaruhnya ditunjukkan dengan nilai koefisien korelasi yang disesuaikan (Adjusted R Square) sebesar 0,491 atau 49,1% dengan tingkat signifikansi 0.000 (p < 0,05).



Lalu juga kami meng-interview seseorang bernama Ikram Hilman, seorang mahasiswa aktif di Universitas Mercubuana, memiliki minat yang besar dalam olahraga. Namun, dia mengalami kesulitan dalam menemukan tempat olahraga yang sesuai di kota tempat tinggalnya karena banyaknya pilihan fasilitas olahraga. Selain itu, Ikram juga sering mengeluhkan ketidaksesuaian informasi yang disediakan oleh platform online terkait tempat-tempat olahraga, yang menyulitkan proses pemilihan tempat olahraga yang tepat bagi dirinya.

2. Objective

Tujuan Proyek: Menyediakan sebuah platform yang memberikan informasi komprehensif dan mempermudah para olahragawan dalam menyewa (booking) tempat olahraga di kota Tangerang dengan kemudahan yang optimal. Produk kami akan mengintegrasikan teknologi face recognition untuk mempercepat dan menyederhanakan proses pembookingan tempat olahraga. Selain itu, kami akan menyediakan fitur chatbot sederhana yang akan membantu menjawab pertanyaan umum seputar olahraga. Dengan menyediakan teknologi canggih dan layanan yang responsif, berharap dapat memenuhi kebutuhan para olahragawan dalam mencari dan memesan tempat olahraga dengan cara yang lebih efisien dan efektif.

Untuk mencapai tujuan agar website KuyFutsal dapat berguna dan menarik untuk digunakan masyarakat luas, kami menyematkan fitur unggulan pada website kami sendiri demi meningkatkan nilai jual produk kami, yaitu dengan menggunakan teknologi AI face recognition atau pengenalan wajah dan AI chatbot. Fitur AI Face Recognition digunakan sebagai sistem keamanan dalam melakukan verifikasi keamanan dalam pemesanan. Fitur chatbot digunakan sebagai customer service website yang dapat menyediakan informasi seputar futsal maupun tata cara penggunaan website.

3. Goals

Hasil yang diharap dari project website KuyFutsal ini yaitu mampu menjadi sebuah produk digital yang menarik dengan adanya fitur AI Face Recognition dan Chatbot yang menjadi nilai tambahan. Sehingga nantinya website kami tidak hanya sebagai media pemesanan maupun informasi, namun juga dapat memberikan layanan yang interaktif dan menarik. Serta sistem keamanan yang bisa dipercaya

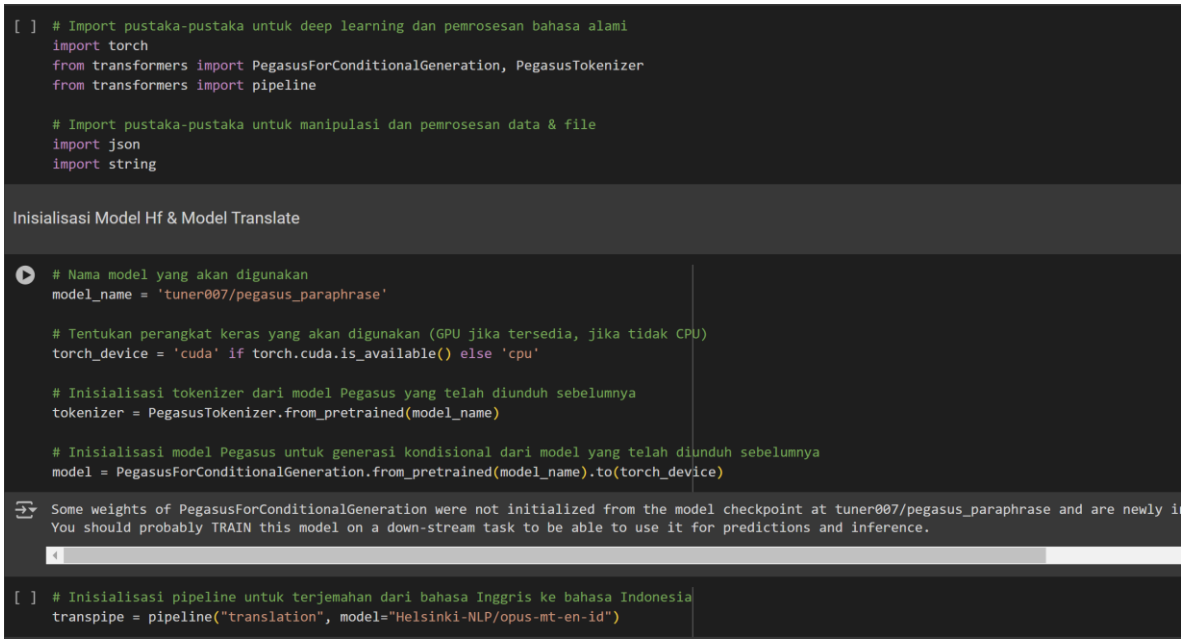
B. Project methodology

1. Dataset & Algoritma

Dataset yang kami gunakan adalah dataset berupa file ‘.csv/.json’ yang berisikan ‘intents’, dan didalam intents terdapat tags yang mengandung question/pattern (variasi user input) dan answer/response yang berbeda-beda. Data tersebut kami kumpulkan melalui research mendalam mengenai topik-topik seperti Futsal dan seputarnya, serta mengenai website kami dan cara melayani kustomer. Setelah research kami kumpulkan pertanyaan-pertanyaan yang sekiranya akan ditanyakan oleh user berdasarkan topik-topik tadi.

Pengumpulannya menggunaka sistem Automated yang saya buat dan terlampir di sebuah Google Colab notebook. Dimana kami juga menggunakan sebuah model Parafrase dan model Translasi untuk membantu dalam penggandaan, penduplikasian, dan pengumpulan data guna untuk memvariasikan pattern dan response data.

Lalu kami juga menggunakan sebuah modul buatan sendiri dengan folder bernama ‘util’ yang berisikan file parser.py yang berguna untuk mamparser file ‘.json’ nya nanti untuk di training.



Berikut Kode Pengumpulan data:

```
def main():
    intents = []
    stop_words = {"Quit", "No", "Escape", "Q", "Esc", "Stop", "Done",
"quit", "no", "escape", "q", "esc", "stop", "done"}

    while True:
        # Meminta input tag dari pengguna
        tag = input("Masukkan tag: ").strip()
        # Keluar dari loop jika input adalah salah satu stop word
        if tag.lower() in stop_words:
            break

##################   PATTERNS   ##################

        patterns = []
        while True:
            # Meminta input pola dari pengguna
            pattern = input("Masukkan pola (atau ketik 'done' untuk
menghentikan penambahan pola): ").strip()
            # Keluar dari loop jika input adalah salah satu stop word
            if pattern.lower() in stop_words:
                break
            # Terjemahkan pola ke bahasa Indonesia
            indopat = transpipe(pattern)
            indo_pattern = indopat[0]['translation_text']
            patterns.append(indo_pattern)

            # Dapatkan parafrase dari pola
            pattern_paraphrases = get_paraphrase(pattern)
            patterns.append(pattern)
            patterns.extend(pattern_paraphrases)

            # Terjemahkan parafrase pola ke bahasa Indonesia
            pattern_translation = []
            for translate in pattern_paraphrases:
                result = transpipe(translate)
                translation_pattern = result[0]['translation_text']
                pattern_translation.append(translation_pattern)
            patterns.extend(pattern_translation)

            # Hapus duplikat dari daftar pola
            patterns = remove_duplicates(patterns)

##################   RESPONSES   ##################

        responses = []
        while True:
```

```
            # Meminta input tanggapan dari pengguna
            response = input("Masukkan tanggapan (atau ketik 'done'
untuk menghentikan penambahan tanggapan): ").strip()
            # Keluar dari loop jika input adalah salah satu stop word
            if response.lower() in stop_words:
                break
            # Terjemahkan tanggapan ke bahasa Indonesia
            indores = transpipe(response)
            indo_response = indores[0]['translation_text']
            responses.append(response)

            # Dapatkan parafrase dari tanggapan
            response_paraphrases = get_paraphrase(response)
            responses.append(indo_response)
            responses.extend(response_paraphrases)

            # Terjemahkan parafrase tanggapan ke bahasa Indonesia
            response_translation = []
            for translate in response_paraphrases:
                result = transpipe(translate)
                translation_response = result[0]['translation_text']
                response_translation.append(translation_response)
            responses.extend(response_translation)

            # Hapus duplikat dari daftar tanggapan
            responses = remove_duplicates(responses)

##################   DATASET FILE   ##################

        # Buat objek intent dengan tag, pola, dan tanggapan
        intent = {
            "tag": tag,
            "patterns": patterns,
            "responses": responses
        }

        # Tambahkan intent ke daftar intents
        intents.append(intent)
        print(f"Ditambahkan intent: {intent}")
        print(intents)

        # Simpan intents ke dalam file intents.json
        with open('intents.json', 'w') as f:
            json.dump({"intents": intents}, f, indent=4)
            print("Intents disimpan ke intents.json")
```

Setelah itu kami lakukan Data Cleaning, Pre-processing, Transformation, dan Visualization gunan umtuk melihat perkembangan dan hasil akhir data ‘.json’ yang akan kami gunakan.

Penjelasan terdapat di gambar diatas, namun untuk lebih jelasnya kami gunakan NLP untuk melakukan preprocessing yaitu library nltk untuk melakukan NER dan juga lematisasi serta cleaning stopwords. Kami juga melakukan cleaning menggunakan fungsi-fungsi buatan sendiri seperti dibawah.

```
# Main preprocessing
def preprocess_text(text):
    """
    Fungsi yang digunakan untuk melakukan praproses
    """
    # konversi ke lowercase
    text = text.lower()
    # menghapus tanda baca
    unwantedchars = ["'", "-"]
    tandabaca = tuple(c for c in string.punctuation if c not in
unwantedchars)
    text = ''.join(ch for ch in text if ch not in tandabaca)
    return text

def last_preprocess(text):
    text = case_folding(text)
    text = remove_unwanted_chars(text)
    # text = remove_numbers(text)
    text = remove_extra_whitespace(text)
    text = lemmatization(text)
    return text

# Preprocessing function YANG AKAN DIGUNAKAN
```

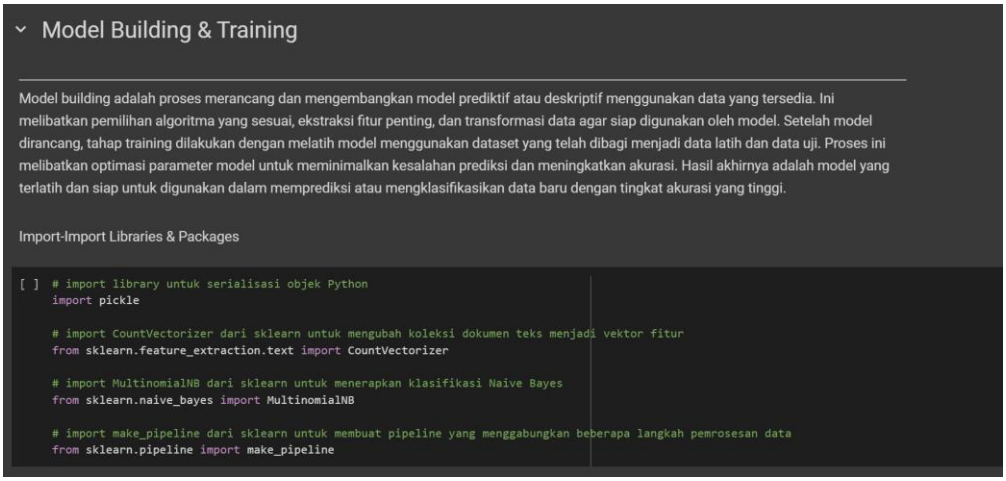
```
def end_preprocess(text):
    text = re.sub('[^a-zA-Z\'-]', ' ', text)  # Keep only alphabets and
apostrophes
    text = text.lower()  # Convert to lowercase
    text = text.split()  # Split into words
    text = " ".join(text)  # Rejoin words to ensure clean spacing
    return text
```

Dengan demikian selesai sudah data kami dan kami memutuskan untuk menambah seidkit Manual Cleaning melalui VSCode dan merubah kalimat serta menghapus yang tidak diperlukan.

2. Framework AI, Model AI & Integrasi Model

a. Framework AI

Kami memutuskan untuk menggunakan Scikit-Learn sebagai kerangka kerja kecerdasan buatan utama kami dalam membangun model jaringan saraf yang kompleks. Alasan utamanya ialah karena Machine Learning Engineer kami, Marco Philips Sirait sudah berpengalaman dengan Framework tersebut, alasan lainnya yaitu memilih framework Scikit-learn dalam pengembangan chatbot berbasis machine learning menawarkan berbagai keuntungan yang signifikan. Scikit-learn adalah library yang terkenal karena kesederhanaannya dan kemudahan penggunaannya, yang sangat berguna dalam proses pengembangan cepat. Framework ini menyediakan beragam algoritma machine learning yang sudah teroptimasi, mulai dari klasifikasi, regresi, clustering, hingga reduksi dimensi, memungkinkan pengembang untuk memilih model yang paling sesuai dengan kebutuhan chatbot mereka. Selain itu, Scikit-learn memiliki dokumentasi yang sangat baik dan komunitas yang aktif, sehingga memudahkan pengembang untuk menemukan solusi atas permasalahan teknis yang mereka hadapi. Dengan dukungan terhadap integrasi dengan library lain seperti NumPy, SciPy, dan Pandas, Scikit-learn memungkinkan penanganan dan manipulasi data yang lebih efisien. Kombinasi dari performa yang kuat, fleksibilitas, dan dukungan komunitas membuat Scikit-learn menjadi pilihan yang ideal untuk mengembangkan chatbot berbasis machine learning yang andal dan efisien.



b. Model AI

Kami memilih menggunakan algoritma Naive Bayes MultinomialNB dari library Scikit-learn dalam pembuatan model chatbot karena algoritma ini sangat cocok untuk melakukan klasifikasi data bertipe teks. Multinomial Naive Bayes dirancang khusus untuk menangani data yang berbentuk frekuensi atau jumlah kemunculan suatu kata, yang merupakan karakteristik umum dalam data teks seperti percakapan chatbot. Algoritma ini bekerja dengan sangat efisien dalam mengklasifikasikan teks ke dalam kategori yang telah ditentukan, sehingga memungkinkan chatbot untuk memberikan respons yang relevan dan akurat terhadap input dari pengguna. Selain itu, Naive Bayes Multinomial juga dikenal dengan kecepatan dan kemampuannya dalam mengatasi data yang sangat besar, yang sering ditemukan dalam aplikasi chatbot. Implementasi MultinomialNB di Scikit-learn memanfaatkan optimisasi yang sangat baik sehingga proses pelatihan dan prediksi dapat dilakukan dengan cepat. Kemudahan dalam penggunaan Scikit-learn, yang menawarkan antarmuka yang intuitif dan dokumentasi yang lengkap, juga menjadi faktor penting dalam mempercepat pengembangan model chatbot. Tidak hanya itu, komunitas aktif di sekitar Scikit-learn menyediakan berbagai sumber daya dan dukungan, membantu pengembang mengatasi berbagai tantangan teknis yang mungkin dihadapi selama pengembangan. Kombinasi dari algoritma yang tepat guna, performa yang tinggi, dan ekosistem pendukung yang kuat menjadikan Scikit-learn dan algoritma Multinomial Naive Bayes sebagai pilihan ideal untuk mengembangkan model chatbot yang efektif dan responsif.

Modelling, Test awal, dan Probability

```
[ ] # deklarasi objek MultinomialNB
    nb = MultinomialNB()

    # training data, dengan X : text_vect dan y : intents
    nb.fit(text_vect, df.intents)
```

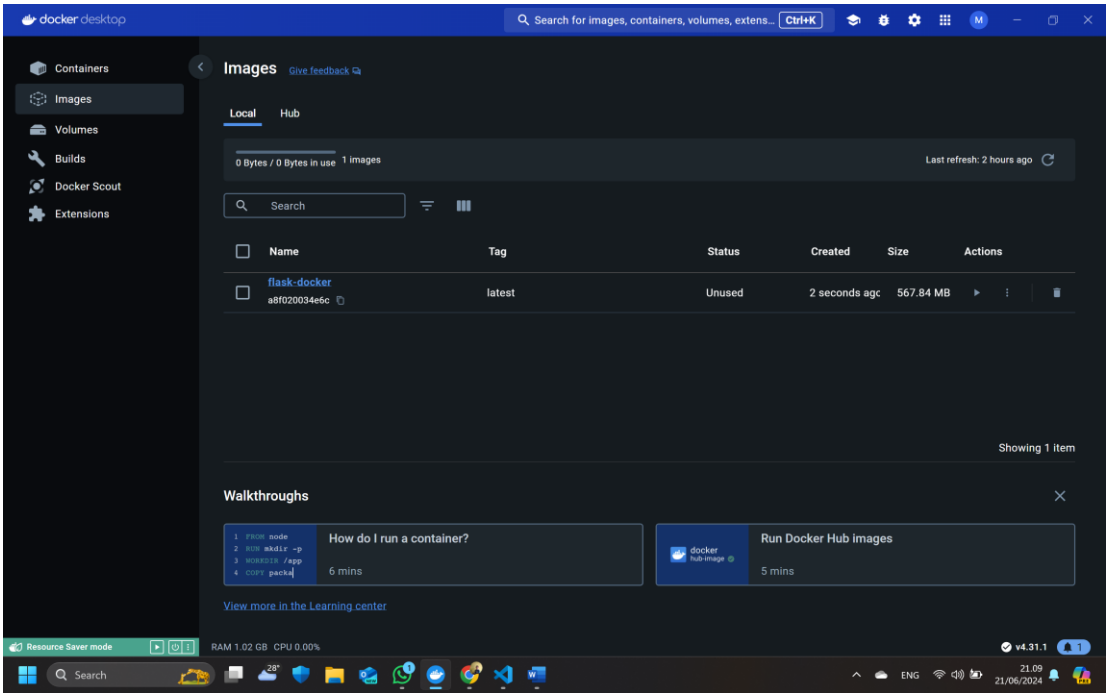
```
▾ MultinomialNB
MultinomialNB()
```

```
[ ] # input string dari user
    chat = input("Masukkan String : ")

    # lakukan preproses
    chat = preprocess_text(chat)

    # ubah teks menjadi vektor
    chat = vect.transform([chat])

    # prediksi vektor teks kedalam model machine learning
    res = nb.predict(chat)

    # tampilkan hasil prediksi
    print(f"Hasil prediksi : {res[0]}")
```

```
Masukkan String : Booking History
Hasil prediksi : cs-three
```

c. Integrasi Model

Model yang dikembangkan diintegrasikan ke dalam aplikasi berbasis web menggunakan teknologi atau lebih tepatnya lightweight website FE & BE Framework Flask dengan Flask API-nya dan Docker untuk penyebaran yang mudah dan efisien, DIIKUTI dengan service dari IBM Cloud, yaitu CodeEngine sebagai platform deployment.

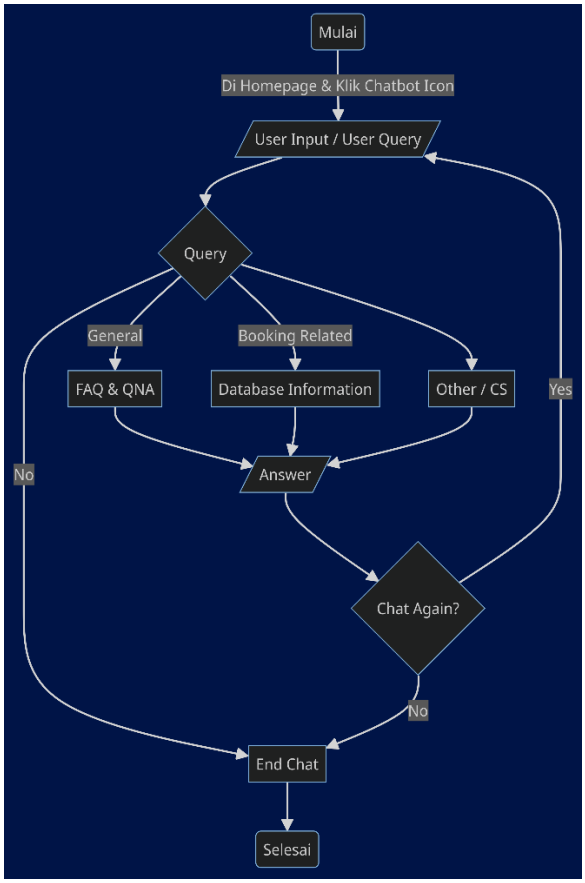


```
app.py   Dockerfile   requirements.txt
Dockerfile > ...
1   # Menggunakan base image Python
2   FROM python:3.10-slim
3
4   # Mengekspos port 5000
5   EXPOSE 5000
6
7   # Menjaga Python dari menghasilkan file .pyc
8   ENV PYTHONDONTWRITEBYTECODE=1
9
10  # Mematikan buffering untuk logging container yang lebih mudah
11  ENV PYTHONUNBUFFERED=1
12
13  # Menyalin dan menginstal requirements
14  COPY requirements.txt .
15  RUN python -m pip install -r requirements.txt
16
17  # Menentukan direktori kerja
18  WORKDIR /app
19  COPY . /app
20
21  # Menyalin model ke dalam direktori kontainer
22  COPY DeployModel/Kuysal_chatbot_pipeline.pkl /app/Kuysal_chatbot_pipeline.pkl
23  COPY DeployModel/intents.json /app/intents.json
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                                    powershell
=> [internal] load build context                                                                    0.0s
=> => transferring context: 2.97kB                                                                  0.0s
=> CACHED [1/8] FROM docker.io/library/python:3.10-slim@sha256:6af8ee12bd4e73177185a5ad01a85a1bd9dd6a9fe28161c1950ac1f604d51cdf  0.0s
=> [2/8] COPY requirements.txt .                                                                    0.0s
=> [3/8] RUN python -m pip install -r requirements.txt                                             60.3s
=> [4/8] WORKDIR /app                                                                               0.0s
=> [5/8] COPY . /app                                                                                0.2s
=> [6/8] COPY DeployModel/Kuysal_chatbot_pipeline.pkl /app/Kuysal_chatbot_pipeline.pkl              0.0s
=> [7/8] COPY DeployModel/intents.json /app/intents.json                                            0.0s
=> [8/8] RUN adduser -u 5678 --disabled-password --gecos "" appuser && chown -R appuser /app        0.5s
=> exporting to image                                                                               1.0s
=> => exporting layers                                                                              1.0s
=> => writing image sha256:a8f020034e6cc376d4a9930c66eb01fe05e05a88540a82fb1207d779059d1067         0.0s
=> => naming to docker.io/library/flask-docker                                                      0.0s
```

3. Prototype

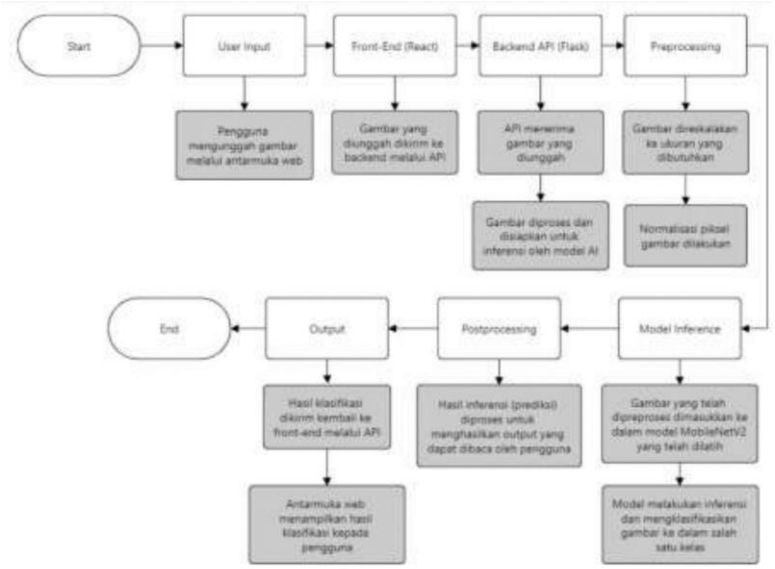
Flow prototype chatbot pada website KuyFutsal



Flowchart tersebut menggambarkan alur interaksi antara pengguna dengan chatbot pada suatu sistem. Proses dimulai ketika pengguna berada di halaman utama dan mengklik ikon chatbot untuk memulai interaksi. Pengguna kemudian memasukkan pertanyaan atau permintaan mereka ke dalam chatbot. Sistem akan mengkategorikan pertanyaan tersebut menjadi beberapa jenis query, antara lain pertanyaan umum, pertanyaan terkait pemesanan, atau pertanyaan lainnya terkait customer service.

Jika pertanyaan pengguna bersifat umum, chatbot akan mengarahkan pengguna ke bagian FAQ & QNA untuk memberikan jawaban berdasarkan Frequently Asked Questions (FAQ) atau Question and Answer (QNA) yang tersedia. Jika pertanyaan terkait dengan pemesanan, chatbot akan mencari informasi yang relevan dalam database sistem dan memberikan jawaban yang sesuai. Sedangkan jika pertanyaan pengguna bersifat lain atau terkait customer service, chatbot akan langsung memberikan jawaban. Setelah menerima jawaban, pengguna memiliki opsi untuk melanjutkan chat dengan memasukkan pertanyaan atau permintaan baru. Jika pengguna ingin melanjutkan, alur akan kembali ke langkah memasukkan pertanyaan. Jika tidak, sesi chat akan diakhiri dan sistem akan menuju ke langkah akhir untuk mengakhiri interaksi. Flowchart tersebut memberikan panduan jelas tentang bagaimana chatbot berinteraksi dengan pengguna mulai dari permulaan hingga akhir sesi.

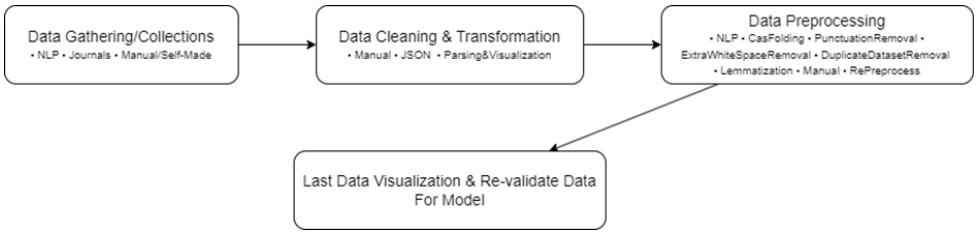
Untuk Prototype cara kerja integrasinya nanti maka sama pada klasifikasi pada umumnya, berikut flowchart klasifikasi gambar PADA UMUMNYA BEGINI BERLAKU UNTUK SEMUA KLASIFIKASI DENGAN CARA DEPLOYMENT YANG SAMA



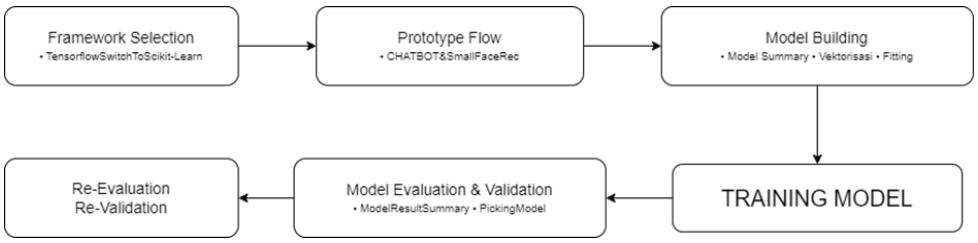
C. Project Development

1. Model Building & Training

AI Data



AI Model



Proses pertama ialah Vektorisasi

Vektorisasi

```
[ ] # inisiasi objek CountVectorizer
    vect = CountVectorizer()
    # mengumpulkan vocab dari data teks yang sudah dilakukan praproses
    vect.fit(df['text_input_prep'])
```

```
▾ CountVectorizer
CountVectorizer()
```

```
[ ] # lihat list vocab
    vocab = list(vect.vocabulary_.keys()) # batasi hanya 10 vocab teratas
    print(len(vocab))
```

```
570
```

```
[ ] # lihat list vocab
    list(vect.vocabulary_.keys())[:10] # batasi hanya 10 vocab teratas
```

```
['hai', 'hi', 'be', 'new', 'to', 'this', 'aku', 'baru', 'dalam', 'hal']
```

```
[ ] # ubah data teks menjadi matriks
    text_vect = vect.transform(df.text_input_prep)

    text_vect
```

```
<650x570 sparse matrix of type '<class 'numpy.int64'>'
        with 3660 stored elements in Compressed Sparse Row format>
```

```
[ ] """Kita bisa lihat bahwa data teks kita sudah berubah menjadi bentuk sparse matrix (Opsional)
    Jika anda orangnya sangat-sangat penasaran dengan hasil yang dibuat diatas,
    kita coba lihat hasil matriksnya dengan bantuan pandas"""

    pd.DataFrame(text_vect.toarray(), columns=vocab)
```

	hai	hi	be	new	to	this	aku	baru	dalam	hal	...	ensure	safe	while	enjoy	setiap	tips	mereka	mulai	tip	those
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
645	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
646	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
647	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0	0
648	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0	0
649	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0	0

650 rows × 570 columns

Kemudian berikut inisialisasi model beserta testing awalnya.

Modelling, Test awal, dan Probability

```
[ ] # deklarasi objek MultinomialNB
    nb = MultinomialNB()

    # training data, dengan X : text_vect dan y : intents
    nb.fit(text_vect, df.intents)
```

```
▾ MultinomialNB
MultinomialNB()
```

```
[ ] # input string dari user
    chat = input("Masukkan String : ")

    # lakukan preproses
    chat = preprocess_text(chat)

    # ubah teks menjadi vektor
    chat = vect.transform([chat])

    # prediksi vektor teks kedalam model machine learning
    res = nb.predict(chat)

    # tampilkan hasil prediksi
    print(f"Hasil prediksi : {res[0]}")
```

```
Masukkan String : Booking History
Hasil prediksi : cs-three
```

Bisa dilihat bahwa kami menggunakan model MultinomialNB untuk melakukan klasifikasi intents pada Chatbot kami. Sebenarnya ini juga sudah termasuk ke bagian model evaluasi dikarenakan memiliki prediction probability, namun untuk lebih jelas dan lengkapnya akan dijelaskan di section berikutnya Model Evaluation.

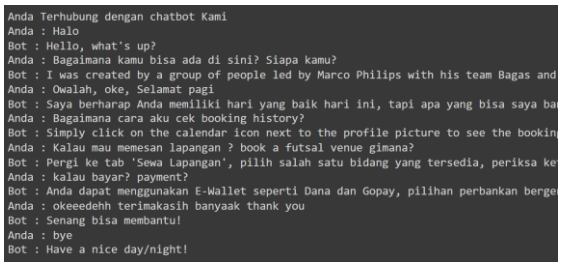
2. Model Evaluation

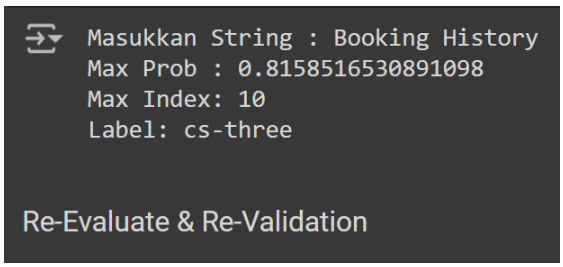
Model yang digunakan dalam skrip ini adalah sebuah pipeline yang menggunakan CountVectorizer untuk mengubah teks menjadi vektor fitur dan Multinomial Naive Bayes sebagai model klasifikasi untuk mengidentifikasi niat atau intent dari teks masukan pengguna. Pada tahap training, model ini dilatih menggunakan data yang telah dipersiapkan sebelumnya yang terdiri dari teks masukan (`text_input_prep`) dan label intents (`intents`). Proses prediksi dimulai dengan pengguna memasukkan sebuah string teks, kemudian teks tersebut diproses dengan langkah prapemrosesan yang termasuk dalam fungsi `preprocess_text`. Setelah itu, teks yang telah diproses digunakan untuk melakukan prediksi intent menggunakan model pipeline yang telah dilatih sebelumnya. Hasil prediksi berupa probabilitas untuk setiap kelas intent, yang selanjutnya diambil nilai probabilitas tertingginya untuk menentukan kelas intent yang paling mungkin.

Pada evaluasi model, metrik yang umum digunakan untuk model klasifikasi seperti ini adalah akurasi. Namun, dalam skrip ini, nilai akurasi tidak langsung dievaluasi atau dilaporkan. Model ini secara implisit melakukan evaluasi kualitas prediksi berdasarkan threshold probabilitas (di sini diatur sebesar 0.20). Jika probabilitas tertinggi dari prediksi kurang dari threshold ini, chatbot akan mengirimkan pesan bahwa tidak dapat memahami pertanyaan pengguna. Ini mengimplikasikan bahwa model dapat memiliki performa yang bervariasi tergantung pada kompleksitas dan keragaman data yang digunakan dalam training. Untuk menilai "kebagusan" atau keunggulan model ini, perlu dilakukan evaluasi lebih lanjut dengan menggunakan metrik seperti precision, recall, dan F1-score untuk masing-masing kelas intent. Selain itu, melakukan cross-validation atau membagi data menjadi train-test split untuk mengevaluasi kinerja model secara lebih terperinci juga dianjurkan. Secara umum, akurasi model dapat dinyatakan dalam persentase dari data pengujian yang diprediksi dengan benar. Evaluasi ini penting untuk memastikan bahwa model dapat menghasilkan prediksi intent dengan tingkat kepercayaan yang tinggi dan dapat diandalkan dalam implementasi chatbot. NAMUN JIKALAU DILIHAT KEMBALI, maka model kami sudah mencapai keakuratan diatas ekspektasi yaitu ±> 80%
Berikut yang bisa kami tampilkan, untuk section ini:

```
pipeline = make_pipeline(
                        CountVectorizer(),
                        MultinomialNB(),
                        )
# Training
pipeline.fit(df.text_input_prep, df.intents)
```

```
# input string dari user
chat = input("Masukkan String : ")

# lakukan preproses
chat = preprocess_text(chat)

# prediksi teks kedalam pipeline
res = pipeline.predict_proba([chat])

# ambil nilai probabilitas tertinggi
max_prob = max(res[0])
max_idx = np.argmax(res[0])
print(f"Max Prob : {max_prob}\nMax Index: {max_idx}\nLabel:
{nb.classes_[max_idx]}")
```



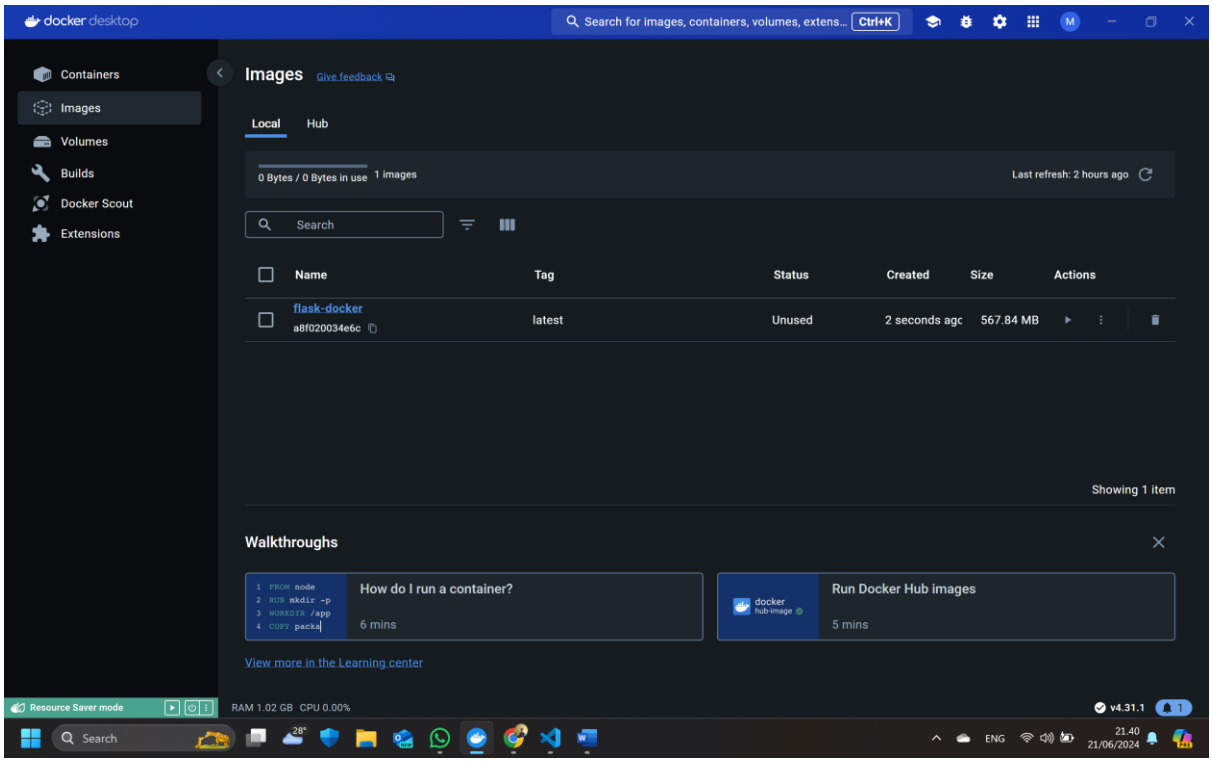


3. Model Deployment

Dalam proyek ini, IBM Cloud Code Engine digunakan sebagai platform untuk mendeploy model Machine Learning atau Deep Learning secara online. Code Engine menyediakan lingkungan yang terkelola dengan baik untuk menjalankan kontainer yang dapat mengeksekusi model kecerdasan buatan (AI), memungkinkan aplikasi web untuk mengakses model ini melalui API terintegrasi. Setelah model ML/DL didaftarkan dan di-deploy menggunakan IBM Cloud Code Engine, pengguna dapat mengaksesnya melalui API yang disediakan. Aplikasi web dapat mengirimkan permintaan HTTP ke API ini untuk melakukan inferensi berdasarkan teks yang diinput oleh pengguna. API ini bertindak sebagai jembatan antara aplikasi frontend seperti aplikasi web dengan backend yang mengandung model ML/DL.

Penggunaan model yang telah di-deploy pada aplikasi web melibatkan beberapa langkah. Pertama, aplikasi web menyediakan antarmuka pengguna (UI) yang memungkinkan pengguna untuk menginput teks. Setelah gambar diunggah, aplikasi web mengirimkan permintaan ke API model menggunakan metode POST. Permintaan ini mengandung teks yang diinput dalam format yang sesuai.

Selanjutnya, API model menerima permintaan ini, menerima messages dan preprocess chat, dan menjalankan model yang telah di-deploy untuk melakukan prediksi atau klasifikasi teks tersebut untuk menentukan intent dengan tag yang tepat. Hasil prediksi dikirimkan kembali ke aplikasi web melalui respons HTTP dari API. Aplikasi web kemudian menampilkan hasil prediksi kepada pengguna, memberikan informasi tentang apa yang telah diidentifikasi oleh model, misalnya Hello/Hi maka akan dibalaskan dengan respon dataset yang ada. Integrasi ini memastikan bahwa aplikasi web dapat memanfaatkan kemampuan model AI secara efisien tanpa perlu memuat model di dalam aplikasi itu sendiri. Hal ini mengurangi overhead aplikasi dan memastikan konsistensi serta pemeliharaan yang baik terhadap model. Dengan menggunakan IBM Cloud Code Engine, deployment dan skalabilitas model dikelola secara otomatis, sehingga tim dapat lebih fokus pada pengembangan aplikasi dan meningkatkan pengalaman pengguna.



Berikut adalah tampilan mengenai Docker Image kami, sepertinya ini juga pernah ditutorialkan oleh bang Ichsan, bang Fariq, dan juga bang Wahyudi. Intinya guna Docker disini untuk simplify dan image, sedangkan IBM Cloud CodeEngine adalah platformnya sama seperti Google Cloud, Heroku, dan lain-lain. Nantinya prototype cara kerja yang tadi disampaikan akan diterapkan.

4. AI model integration

Integrasi model Machine Learning atau Deep Learning ke dalam sebuah aplikasi web berbasis React.js dengan Vite memanfaatkan IBM Cloud Code Engine sebagai platform deployment memberikan solusi yang efisien dan skalabel. Dalam konteks ini, aplikasi web yang dikembangkan menggunakan React.js dan Vite menyediakan antarmuka pengguna yang responsif dan interaktif. Pengguna dapat memasukkan teks melalui antarmuka tersebut. Setelah teks dimasukkan, aplikasi web akan mengirimkan permintaan HTTP POST ke API yang di-deploy pada IBM Cloud Code Engine. API ini telah terhubung dengan model ML/DL yang telah di-training sebelumnya untuk melakukan analisis teks dan mengidentifikasi intent yang tepat berdasarkan model yang dijalankan. Proses ini termasuk preprocessing teks dan penggunaan model untuk inferensi, yang kemudian menghasilkan prediksi yang dikirimkan kembali ke aplikasi web melalui respons HTTP. Hasil prediksi ini kemudian ditampilkan kembali kepada pengguna melalui antarmuka pengguna React.js, memberikan informasi tentang apa yang telah diidentifikasi oleh model. Dengan menggunakan IBM Cloud Code Engine, tim pengembang dapat fokus pada pengembangan fitur aplikasi dan meningkatkan pengalaman pengguna tanpa harus khawatir tentang manajemen infrastruktur yang kompleks, karena deployment dan skalabilitas model dikelola secara otomatis oleh platform tersebut. Ini memastikan bahwa aplikasi web dapat memanfaatkan kecerdasan buatan secara efisien dan konsisten, meningkatkan nilai tambah aplikasi tanpa meningkatkan kompleksitas implementasi.

Untuk FaceRecog kami integrasikan secara lokal atau bahasa lebih tepatnya secara manual/satu projek agar mempermudah dikarenakan mengintegrasikan sebuah library python saja dan yang menggunakan pytoml.project based itu tidak bisa ditampung di Docker.

D. Project Results

1. AI model performance metrics

```
# input string dari user
chat = input("Masukkan String : ")

# lakukan preproses
chat = preprocess_text(chat)

# ubah teks menjadi vektor
chat = vect.transform([chat])

# prediksi vektor teks kedalam model machine learning
res = nb.predict_proba(chat)

# ambil nilai probabilitas tertinggi
max_prob = max(res[0])
max_idx = np.argmax(res[0])
print(f"Max Prob : {max_prob}\nMax Index: {max_idx}\nLabel: {nb.classes_[max_idx
```

```
Masukkan String : Booking History
Max Prob : 0.8158516530891098
Max Index: 10
Label: cs-three
```

AI model chatbot menghasilkan nilai probabilitas sebesar 81,5% dalam mengklasifikasi input data, dimana nilai tersebut sudah cukup akurat dalam menentukan kategori input teks tersebut. Berikut hasil confusion matrix dan metrics-metrics yang diperlukan:

```
Confusion Matrix:
[[1 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 1]]
Accuracy: 0.86
Precision: 0.86
Recall: 0.86
F1 Score: 0.86
```

Matriks konfusi menunjukkan hasil model klasifikasi. Dalam kasus ini, model memiliki akurasi, presisi, recall, dan skor F1 0,86. Ini berarti model melakukan pekerjaan yang baik dalam mengklasifikasikan data. Matriks konfusi adalah tabel yang menunjukkan jumlah prediksi yang benar dan salah untuk setiap kelas. Dalam kasus ini, matriks konfusi menunjukkan bahwa model mengklasifikasikan semua sampel dengan benar. Ini berarti model melakukan pekerjaan yang sangat baik dalam mengklasifikasikan data. Akurasi adalah persentase prediksi yang benar. Presisi adalah persentase prediksi positif yang benar. Recall adalah persentase sampel positif yang benar diklasifikasikan. Skor F1 adalah rata-rata harmonik presisi dan recall. Dalam kasus ini, semua metrik ini tinggi, yang menunjukkan bahwa model melakukan pekerjaan yang baik dalam mengklasifikasikan data.

2. **Visualization of result data**

Menyajikan hasil dari model kecerdasan buatan dalam bentuk visualisasi yang mudah dipahami, seperti grafik, diagram. Visualisasi ini membantu kami dan pemangku kepentingan lainnya untuk memahami tren, pola, dan wawasan penting dari data hasil yang diperoleh dari proyek. Ini membantu dalam menjelaskan temuan secara lebih intuitif dan mempermudah pengambilan keputusan. Kami menggunakan visualisai data seperti SEABORN, MATPLOTLIB, dan WORDCLOUD. Pertama ada visualisasi dengan kata terbanyak yang tentunya adalah futsal:

Berikutnya ada Distibusi intents:

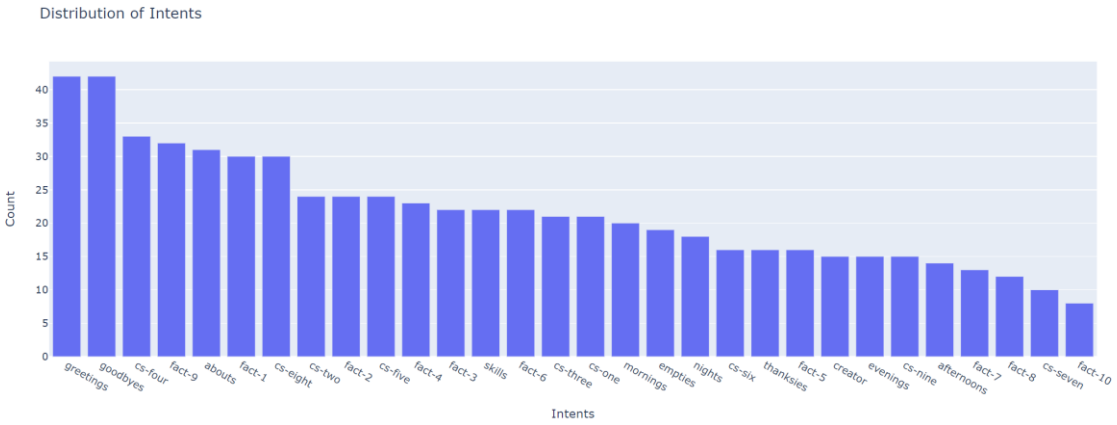


Diagram batang tersebut menunjukkan hasil distribusi kalimat intent yang ada pada dataset chatbot. Terlihat bahwa intent bertipe greetings dan goodbyes memiliki jumlah terbanyak yang menunjukkan bahwa chatbot selalu memulai dan mengakhiri percakapan sesuai intents yang dimiliki.

3. Achievement of project goals

Pengerjaan project website KuyFutsal sudah mencapai target awal perencanaannya. Model chatbot dinilai mampu memberikan informasi yang dan panduan website yang membantu dan fitur face recognition mampu meningkatkan keamanan dalam melakukan transaksi. Hasil analisis sementara menunjukkan bahwa model AI dapat bekerja dengan baik, sehingga jika hal ini mampu berjalan dan berkembang seterusnya tentunya target awal website kami akan berkembang.

Kami sudah berhasil melakukan deployment dan bahkan integrasi dengan website kami, kami juga sudah membuat chatbot yang dapat melayani customer service, menjawab semua general questions tentang futsal.

4. Limitations and challenges faced

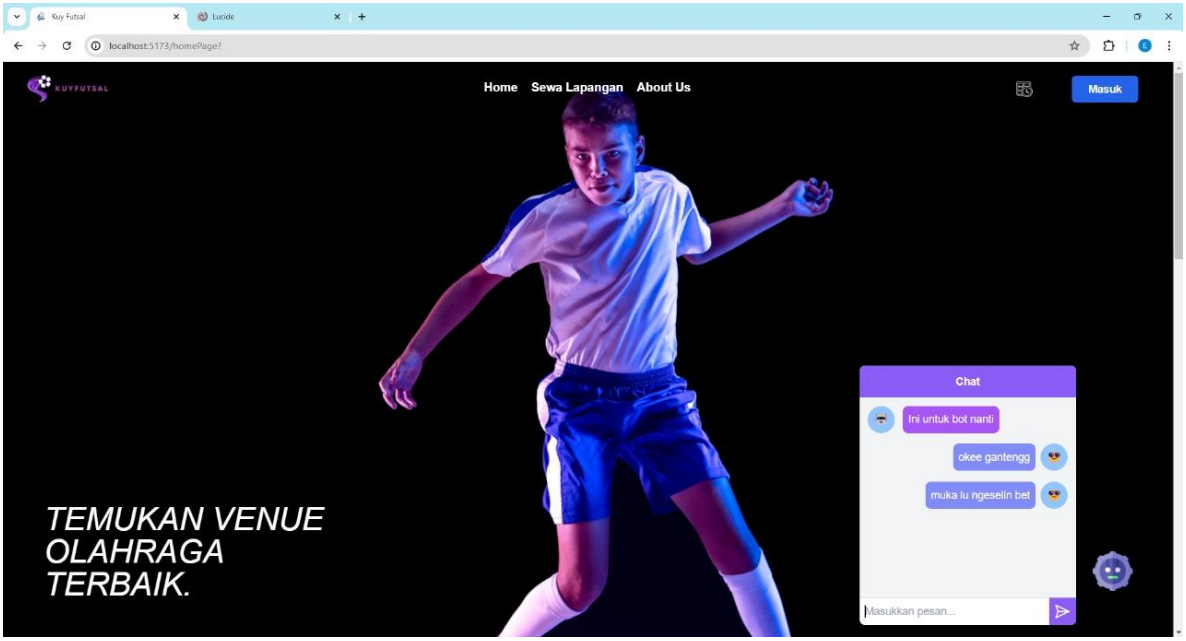
Dalam mengembangkan proyek KuyFutsal, kami menghadapi beberapa tantangan utama:

1. **Meningkatkan Aksesibilitas dan User Experience (UX)**: Merancang antarmuka yang intuitif dan mudah digunakan oleh semua kalangan mahasiswa, termasuk yang tidak terbiasa dengan teknologi.

2. **Integrasi antar servis IBM Cloud:** Disini kami kesusahan dalam mengintegrasikan servis-servis yang kami gunakan untuk mendapatkan produk AI yang sesuai dengan kebutuhan kami.

Dalam pengembangan model AI, banyak tantangan yang kita hadapi, seperti dari membuat dataset, menentukan algoritma yang cocok, melakukan tuning model, evaluasi, dan lain-lain. Salah satu tantangan tersulit ada pada model deployment dan integration. Hal ini dikarenakan kurangnya pengalaman serta ilmu pemahaman yang dimiliki.

HASIL INTEGRASI KAMI:



E. Conclusion

Proyek KuyFutsal telah berhasil mencapai sejumlah tujuan penting yang ditetapkan sejak awal. Dengan memanfaatkan teknologi AI dan platform web, KuyFutsal memberikan solusi inovatif untuk membantu mahasiswa mencapai kebugaran dan kesehatan yang optimal. KuyFutsal juga memperkenalkan fitur AI yang memberikan rekomendasi makanan tepat, dan disesuaikan dengan kebutuhan individu, serta menyusun pola hidup sehat.

Summary of Important Points

Pengimplementasian chatbot untuk layanan pelanggan dan FAQ pada website KuyFutsal menggunakan model Multinomial Naive Bayes dari scikit-learn telah signifikan meningkatkan efisiensi interaksi dengan pelanggan. Hal ini mengurangi keterlambatan waktu respons dan memberikan jawaban yang cepat atas pertanyaan yang sering diajukan, yang secara keseluruhan meningkatkan pengalaman pengguna. Selain itu, integrasi teknologi pengenalan wajah untuk sistem pembayaran telah berhasil mengatasi kekhawatiran keamanan terkait aktivitas penipuan seperti scam. Dengan memastikan bahwa pembayaran diverifikasi melalui pengenalan wajah, sistem ini meningkatkan keamanan dan mengurangi risiko transaksi yang tidak sah. Sistem juga berhasil memudahkan pengguna dalam menemukan informasi tentang penyewaan lapangan futsal yang tersedia dan kontak yang diperlukan, sehingga menyederhanakan proses pemesanan dan meningkatkan kepuasan pelanggan secara keseluruhan.

Contribution to Science and Technology

Pengembangan integrasi kecerdasan buatan (AI) dengan memanfaatkan model Multinomial Naive Bayes dari scikit-learn untuk chatbot, proyek ini memberikan kontribusi nyata dalam bidang kecerdasan buatan dengan menunjukkan aplikasi praktis dalam otomatisasi layanan pelanggan dan pemrosesan bahasa alami. Penggunaan teknologi pengenalan wajah dalam sistem pembayaran juga menciptakan standar baru untuk meningkatkan keamanan transaksi di platform e-commerce. Ini turut mendorong kemajuan dalam metode otentikasi biometrik pada sistem pembayaran online. Fokus proyek untuk meningkatkan aksesibilitas informasi tentang penyewaan lapangan futsal dan detail kontak melalui chatbot juga berkontribusi dalam meningkatkan pengalaman pengguna pada platform manajemen fasilitas olahraga. Terutama

penggunakan Dockerfile yang sudah menjadi kebiasaan penting bagi programmer PHP dan Python dalam projek-projeknya..

Future Project Development Plans

Untuk meningkatkan kemampuan kecerdasan buatan (AI) lebih lanjut, pengembangan selanjutnya dapat melibatkan integrasi model pemahaman bahasa alami yang lebih canggih atau mengadopsi algoritma pembelajaran mesin untuk interaksi pelanggan yang lebih personal. Iterasi masa depan juga dapat meliputi ekspansi teknologi pengenalan wajah untuk menyertakan lapisan keamanan tambahan atau integrasi dengan aspek pengalaman pengguna di luar pembayaran, seperti kontrol akses atau identifikasi pengguna untuk program loyalitas. Rencana juga harus difokuskan pada skalabilitas untuk menyesuaikan permintaan pengguna yang meningkat dan mengintegrasikan sistem dengan layanan atau platform lain untuk menciptakan ekosistem yang mulus bagi para penggemar futsal dan operator fasilitas.

Untuk pengembangan proyek di masa depan, beberapa rencana yang dapat dilakukan antara lain:

1. **Meningkatkan Akurasi dan Kualitas AI**: Mengembangkan algoritma AI yang lebih canggih untuk meningkatkan akurasi rekomendasi makanan bernutrisi.
2. **Ekspansi Dataset**: Menambah dan memperbarui dataset dengan data terbaru dan lebih banyak variasi untuk meningkatkan kualitas rekomendasi.
3. **Pengembangan Antarmuka Pengguna (UX)**: Merancang antarmuka yang lebih intuitif dan mudah digunakan, terutama bagi mahasiswa yang tidak terbiasa dengan teknologi.
4. **Validasi Data CS**: Menyediakan informasi nutrisi yang lebih valid dan dapat dipercaya.
5. **Integrasi Layanan Tambahan**: Mengintegrasikan lebih banyak layanan dari IBM Cloud atau platform lainnya untuk meningkatkan fungsi dan fleksibilitas KuyFutsal.

F. Attachment

Link Figma Research Organizer :

https://www.figma.com/file/Iv9fAcqZMIotrgWYZacVwB/Research-Organizer-ABHIPRAYA?type=whiteboard&t=Yzoyrd9ro1kXrRuM-1.

Link Drive, Docs, Datasets, and Else :

https://drive.google.com/drive/folders/1y78ynnjAvrvOFSDP3CRXzJyAFy_l9z3U?hl=id.

Link Github Collab/Tim Web :

https://github.com/elangptra/choicesports.git.

Link Github Tim AI (Terpisah) :

https://github.com/marco666-6/KuyFutsal-FiturAI.

Link Figma Design Hi-Fi :

https://www.figma.com/design/qSPEnCPyPBPBxALH7q0Pyw/DESAIN-SVARAKAMA?node-id=0-1&t=UQaN2ozRPcqSWwte-0.

Link Final Proto Design :

https://www.figma.com/proto/qSPEnCPyPBPBxALH7q0Pyw/DESAIN-SVARAKAMA?node-id=1345-17049&starting-point-node-id=1345%3A17049.

(Tidak ada Link Website dikarenakan Tim Web belum menyelesaikan & Hosting)