

**TUGAS UJIAN TENGAH SEMESTER (UTS)
MATA KULIAH ADVANCE NLP**

**Dosen Pengampu
Dr. Sajarwo Anggai, S.ST., M.T
Dr. Ir Agung Budi Susanto., MM**



**Oleh:
Asep Ridwan Hidayat
231012050036**

**PROGRAM STUDI TEKNIK INFORMATIKA S-2
PROGRAM PASCASARJANA
UNIVERSITAS PAMULANG
TANGERANG SELATAN
2025**

DAFTAR ISI

DAFTAR ISI.....	i
TUGAS PERTEMUAN 1	3
1.1 Tugas Pertemuan 1	3
1.2 10 jurnal terkait perkembangan NLP terkini (Lima Tahun Terakhir) 3	
1.3 10 Jurnal tentang Teknologi NLP yang yang telah diadopsi oleh Industri/ Government / Perusahaan TI.....	8
PERTEMUAN 3.....	11
1.4 Tugas.....	11
1.5 10 jurnal terkait dengan perkembangan Preprocessing di NLP	11
1.6 Program untuk melakukan Preprocessing pada suatu dokumen .	13
PERTEMUAN 4.....	19
1.7 Tugas.....	19
1.8 10 Jurnal Terkait Dengan Perkembangan <i>Feature Extraction</i> Di NLP	19
1.9 Program untuk melakukan <i>Feature Extraction</i> pada corpus/dataset	21
PERTEMUAN 5.....	27
1.10 Tugas.....	27
1.11 10 jurnal terkait dengan Information Retrieval	27
1.12 Buatlah program Inverted Index untuk menampung corpus/dataset!	29

1.12.1	Tampilan Program	29
1.12.2	Menu Search Index dan output dari seach index.....	29
1.12.3	Script aplikai.....	30
PERTEMUAN 6.....		48
1.13	Tugas.....	48
1.14	10 Jurnal terkait pemanfaatan Topic Model (LDA dan turunannya/related work).....	48
1.15	Pembuatan Program Model LDa	51

TUGAS PERTEMUAN 1

1.1 Tugas Pertemuan 1

1. Cari 10 jurnal terkait perkembangan NLP terkini (5 tahun terakhir)
2. Cari 10 teknologi NLP yang telah diadopsi oleh Industri/
Government/Perusahaan TI

1.2 10 jurnal terkait perkembangan NLP terkini (Lima Tahun Terakhir)

Berikut ini adalah beberapa jurnal perkembangan *Natural Language Processing* (NLP) lima tahun terakhir :

No	Penulis	Tahun	Judul Penelitian	Model	Accuracy	Sumber
1	Muhammad Fathan Fauzan, Rahmi Imanda, Muhammad Adryan Hasbi	2025	Designing an Chatbot with NLP Technology in a Website-Based New Student Admission Information System	Chatbot, NLP	First, blackbox testing shows that the system functions well in responding to messages sent through the chatbot on the website, both from text that matches the intention and text that is abstract and does not match the pattern, with an accuracy rate of 87.5%.	Journal of Applied Informatics and Computing (JAIC)
2	Mukhlis Amien	2023	Sejarah dan Perkembangan Teknik <i>Natural Language Processing</i> (NLP) Bahasa Indonesia: Tinjauan tentang sejarah, perkembangan teknologi, dan	NLP	Studi ini juga menggali aplikasi NLP dalam industri dan penelitian bahasa Indonesia serta mengidentifikasi tantangan dan peluang dalam penelitian dan pengembangan	ELANG: Journal of Interdisciplinary Research E-ISSN: 3025-2482

No Penulis	Tahun	Judul Penelitian	Model	Accuracy	Sumber
		aplikasi NLP dalam bahasa Indonesia		NLP bahasa Indonesia.	
3 Andre Farhan Saputra, Kecitaan Harefa	2025	Penerapan Metode Natural Language Processing (Nlp) Dalam Implementasi Asisten Virtual Chatbot Dengan Memanfaatkan Api Chatgpt Dan Gradio App	Chatbot, NLP	The research results showed that from a questionnaire of 24 respondents, ChatGPT users who directly tested Chillbot provided positive feedback, with a user satisfaction rate of 87.17%.	Building of Informatics, Technology and Science (BITS)
4 Muhammad Rofiq Sudrajat, Muhammad Zakariyah	2024	Penerapan Natural Language Processing dan Machine Learning untuk Prediksi Stres Siswa SMA Berdasarkan Analisis Teks	NLP, Regression, Naive Bayes, Random Forest, dan Support Vector Machine (SVM)	Model Machine Learning yang diujicobakan antara lain Logistic Regression, Naive Bayes, Random Forest, dan Support Vector Machine (SVM). Hasil dari eksperimen menunjukkan bahwa model Naive Bayes yang menggunakan fitur Bigram mencapai akurasi tertinggi yaitu 95,6%, dengan model lainnya mencapai sekitar 93%.	MDPI,Applied Science
5 Muhammad Yusuf, Indah Purnama Sari, Virda Kristy	2024	Sistem Pakar Mencegah Stunting dengan Menentukan Gizi Anak Menggunakan Natural Language	NLP	The strength of the NLP algorithm lies in its ability to understand user queries based on context, resulting in relevant and responsive	Jurnal JTik (Jurnal Teknologi Informasi dan komunikasi) OI: https://doi.org/10.35870/jtik.v9i3.3614

No Penulis	Tahun	Judul Penelitian	Model	Accuracy	Sumber	
		Processing (NLP)		solutions. The testing results indicate a system accuracy rate of 0.9756 or 97%, achieved through valuations using a dataset of user queries under various test scenarios.		
6	Xinyu Fu	2024	Natural Language Processing in Urban Planning: A Research Agenda	NLP	The results reveal that existing research is primarily exploratory with a fragmented research landscape. Future studies should focus on sharing data, benchmarking NLP techniques, fostering collaborative research tailored to planning, and addressing ethical implications to harness NLP's full potential in planning	https://journals.sagepub.com/doi/10.117
7	Ghofrane Merhbene, Alexandre Puttick, Mascha Kurpicz-Briki	2024	Investigating machine learning and natural language processing techniques applied for detecting eating disorders: a systematic literature review	NLP	Investigasi pada jurnal ini mencakup empat area utama: (a) analisis metadata dari makalah yang diterbitkan, (b) pemeriksaan ukuran dan topik spesifik dari kumpulan data yang digunakan, (c) tinjauan penerapan teknik pembelajaran mesin dalam mendeteksi gangguan makan	https://www.frontiersin.org/journals/psychiatry/articles/10.3389/fpsy.2024.1319522/full

No Penulis	Tahun	Judul Penelitian	Model	Accuracy	Sumber
				dari teks, dan terakhir (d) evaluasi model yang digunakan, dengan fokus pada kinerja, keterbatasan, dan potensi risiko yang terkait dengan metodologi saat ini.	
8	2024	Panteleimon Krasadakis, Evangelos Sakkopoulos, Vassilios S. Verykios <i>A Survey on Challenges and Advances in Natural Language Processing with a Focus on Legal Informatics and Low-Resource Languages</i>	NLP	the jurnal conducted an extensive literature review of NLP research focused on legislative documents. We present the current state-of-the-art NLP tasks related to Law Consolidation, highlighting the challenges that arise in low-resource languages. Our goal is to outline the difficulties faced by this field and the methods that have been developed to overcome them	https://www.mdpi.com/2079-9292/13/3/648
9	2024	M Raihan <i>Dynamic Topic Modelling Menggunakan BERTOPIC Dalam Pemilihan Presiden Tahun 2019</i>	Bertopic	Peneliti mencoba menganalisis topik apa saja yang dihasilkan dari <i>tweet</i> yang diunggah oleh masyarakat menjelang Pemilu 2019 dan disertai dengan evolusi topiknya dari waktu ke waktu. Metode pemodelan topik	https://repository.uinjkt.ac.id/dspace/handle/123456789/81888

No Penulis	Tahun	Judul Penelitian	Model	Accuracy	Sumber
				yang akan digunakan kali ini adalah <i>BERTopic</i> . Metode pemodelan topik ini di dasari <i>sentence embedding</i> dengan salah satu jenis arsitektur <i>neural network</i> yaitu <i>Siamese network</i> sehingga metode ini dapat mengelompokkan kata sesuai konteksnya dalam suatu kalimat. Metode <i>BERTopic</i> ini juga dilengkapi dengan fitur <i>Dynamic Topic Modelling</i> yaitu metode pemodelan topik yang dilanjutkan dengan mengevolusi setiap topiknya dari waktu ke waktu. Dengan data <i>tweet</i> yang ada, metode <i>BERTopic</i> mampu menghasilkan topik-topik yang ada dengan baik, hal ini dapat dibuktikan dengan hasil evaluasi dari nilai koheren yang dihasilkan yaitu 0.71. Topik yang dihasilkan juga relevan dan dapat dibuat narasinya.	
10 Fitria	2025	Penerapan Metode NLP	NLP Chatbot	Integrasi dengan suara	NLP output pada Jurnal portal publikasi

No Penulis	Tahun	Judul Penelitian	Model	Accuracy	Sumber
		pada Chatbot Output Suara		chatbot, training data, evaluasi model	

1.3 10 Jurnal tentang Teknologi NLP yang telah diadopsi oleh Industri/ Government / Perusahaan TI

Berikut ini adalah 10 jurnal membahas penerapan Natural Language Prosesing (NLP) pada Perusahaan dan pemerintahan.

No	Penulis/Sum ber	Tahun	Judul Penelitian	Model	Pembahasan	Sumber
1	Eric W.T. Ngai, Ariel K.H. Lui, Brian C.W. Kei	2024	Natural language processing in government applications: A literature review and case analysis	NLP	Studi tinjauan literatur dan analisis kasus penggunaan NLP dalam aplikasi pemerintahan, termasuk efisiensi dokumen dan layanan publik.	Emerald insight Discover Journals, Books & Case Studies
2	Yunqing Jiang, Patrick Cheong-Iao Pang, Dennis Wong ,Ho Yin Kan	2023	Natural Language Processing Adoption in Governments and Future Research Directions: A Systematic Review	NLP	Tinjauan sistematis adopsi NLP di pemerintahan, membahas aplikasi, manfaat, dan tantangan implementasi NLP di sektor publik.	https://www.mdpi.com/2076-3417/13/22/12346
3	Dr. Jagreet Kaur Gill	2025	Natural Language Processing in Government: Complete Guide	Chatbot, NLP	Penjelasan peran NLP dalam transformasi pemerintahan menuju e-governance, termasuk chatbot, analisis sentimen, dan penerjemahan Bahasa	XenonStack Blog dalam (https://www.xenonstack.com/blog/nlp-in-government)

No	Penulis/Sum ber	Tahun	Judul Penelitian	Model	Pembahasan	Sumber
4	Cogent Infotech	2024	Role of NLP in the Public Sector	NLP	Aplikasi NLP untuk menjawab pertanyaan publik, sector peningkatan kepolisian, analisis media sosial, dan ekstraksi informasi penting.	https://www.cogentinfo.com/resources/nlp-in-the-public-sector
5	SAS Whitepaper	2023	Revolutionizing Government Communication with NLP Applications	NLP	Fokus pada chatbot, analisis sentimen, penerjemahan bahasa, ekstraksi informasi, dan pengenalan suara dalam komunikasi pemerintah.	https://www.sas.com/content/dam/SAS/documents/briefs/solution-brief/en/natural-language-processing-government-113025.pdf
6	INA Solutions Blog	2023	Revolutionizing Government Communication with NLP Applications	NLP, Chatbot	Fokus pada chatbot, analisis sentimen, penerjemahan bahasa, ekstraksi informasi, dan pengenalan suara dalam komunikasi pemerintah.	https://inasolutions.com/resources/2023/12/01/natural-language-processing-government-applications/
7	Deloitte Insights	2019	Natural Language Processing Examples in Government Data	NLP	Contoh konkret penggunaan NLP untuk analisis data tak terstruktur di pemerintahan AS, termasuk deteksi pola dan kepatuhan regulasi.	https://www2.deloitte.com/us/en/insights/focus/cognitive-technologies/natural-language-processing-examples-in-government-data.html
8	Emerald Insigh	2024	<i>Natural Language Processing in Government Applications</i>	NLP	Studi komprehensif mengenai penggunaan NLP di pemerintahan, termasuk pengelolaan data, analisis kebijakan, dan pelayanan publik.	https://www.emerald.com/insight/content/doi/10.1108/ims-07-2024-0711/full/html
9	IBM Research	2021	<i>NLP in IT Companies: Enhancing</i>	NLP	Aplikasi NLP dalam perusahaan IT untuk chatbot,	

No	Penulis/Sum ber	Tahun	Judul Penelitian	Model	Pembahasan	Sumber
			<i>Customer Support and Automation</i>		analisis sentimen pelanggan, dan otomatisasi proses bisnis. (berdasarkan pengetahuan umum)	
10	Microsoft Research	2022	Leveraging NLP for Enterprise Knowledge Management	NLP	Penggunaan NLP untuk ekstraksi informasi dan manajemen pengetahuan di perusahaan teknologi besar. (berdasarkan pengetahuan umum)	

PERTEMUAN 3

1.4 Tugas

1. Cari 10 jurnal terkait dengan perkembangan Preprocessing di NLP
2. Buatlah program untuk melakukan Text Preprocessing pada suatu dokumen

1.5 10 jurnal terkait dengan perkembangan Preprocessing di NLP

Berikut tabel 10 jurnal terkait dengan perkembangan *Preprocessing* pada natural language processing (NLP)

No	Penulis/Sumber	Tahun	Judul Penelitian	Model	Pembahasan	Sumber
1	Palomino, Aider	2023	Natural Language Processing: Recent Development and Applications	NLP	Pengaruh preprocessing pada akurasi analisis sentimen media sosial	https://www.mdpi.com/2076-3417/13/20/11395
2	Camacho-Collados et al.	2017	On the Role of Text Preprocessing in Neural Network Architectures		Evaluasi tokenizing, lemmatizing, lowercasing, multiword grouping pada neural classifier Dan tokenisasi sederhana cukup, namun teknik lain berpengaruh tergantung tugas	https://arxiv.org/abs/1707.01780
3	ScienceDirect (Review)	2024	Recent advancements and challenges of NLP-based sentiment analysis		Tinjauan preprocessing pada analisis sentimen (tokenization, normalization, stemming, dsb.) dan Preprocessing penting untuk hasil analisis sentimen yang lebih baik	https://www.sciencedirect.com/science/article/pii/S2949719124000074

No	Penulis/Sumber	Tahun	Judul Penelitian	Model	Pembahasan	Sumber
4	Meystre et al.	2015	Recent Advances in Clinical Natural Language Processing in Support of Semantic Analysis		Preprocessing dalam NLP klinis (normalisasi, tokenisasi, dsb.) dan Preprocessing mendukung ekstraksi informasi medis lebih akurat	https://pmc.ncbi.nlm.nih.gov/articles/PMC4587060/
5	Accenture (dalam Yroge)	2023	Global NLP Market and Preprocessing Trends	<i>Preprocessing NLP</i>	Proyeksi pasar NLP dan pentingnya preprocessing dan Preprocessing menjadi kunci pertumbuhan aplikasi NLP	https://www.linkedin.com/pulse/enhancing-nlp-accuracy-power-text-preprocessing-techniques-yroge
6	LinkedIn (Yroge)	2024	Future Trends and Innovations in Text Preprocessing		Inovasi preprocessing (deep learning, automation, integrasi NER, dsb.) dan Preprocessing makin otomatis dan kontekstual	https://www.linkedin.com/pulse/enhancing-nlp-accuracy-power-text-preprocessing-techniques-yroge
7	ScienceDirect (Review)	2024	Pre-processing methodologies in sentiment analysis		Studi berbagai teknik preprocessing untuk analisis sentimen dan Kombinasi teknik preprocessing menghasilkan hasil terbaik	https://www.sciencedirect.com/science/article/pii/S2949719124000074
8	Palomino & Aider	2017	Preprocessing for Social Media Sentiment Analysis		Perbandingan preprocessing untuk pelatihan word embeddings Preprocessing berbeda berdampak pada hasil embedding	https://arxiv.labs.arxiv.org/html/1707.01780
9	Camacho-Collados et al.	2017	Variability in Preprocessing Techniques and Their Effects		Variasi teknik preprocessing dan pengaruhnya pada model NLP dan Variasi teknik menghasilkan	https://arxiv.labs.arxiv.org/html/1707.01780

No	Penulis/Sumber	Tahun	Judul Penelitian	Model	Pembahasan	Sumber
					variabilitas performa model	
10	Meystre et al.	2015	Semantic Analysis Supported by Preprocessing in Clinical NLP		Variasi teknik preprocessing dan pengaruhnya pada model NLP dan Variasi teknik menghasilkan variabilitas performa model	https://arxiv.org/abs/1707.01780

1.6 Program untuk melakukan Preprocessing pada suatu dokumen

1. Berikut link lengkap pengerjaan menggunakan GoogleColab terkait preprocessing
<https://colab.research.google.com/drive/1z5X5RjH2SqQo8uRaPBKFlxRlBiAlrGf5?usp=sharing>
2. Berikut Scriptnya

```
# Install and import necessary libraries
import nltk
import stanza
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
import string

# Ensure required NLTK data is available
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
nltk.download('omw-1.4')
nltk.download('maxent_ne_chunker')
nltk.download('words')
nltk.download('perluniprops')
nltk.download('nonbreaking_prefixes')

# Ensure Stanza model is downloaded
stanza.download('en')
nlp = stanza.Pipeline('en')
```

```

def preprocess_text(text):
    # Normalization (Lowercasing)
    text = text.lower()

    # Punctuation Removal
    text = text.translate(str.maketrans("", "", string.punctuation))

    # Ensure punkt tokenizer is available
    try:
        tokens = word_tokenize(text)
    except LookupError:
        nltk.download('punkt')
        nltk.download('punkt_tab')
        tokens = word_tokenize(text)

    # Ensure stopwords are available
    try:
        stop_words = set(stopwords.words('english'))
    except LookupError:
        nltk.download('stopwords')
        stop_words = set(stopwords.words('english'))

    filtered_tokens = [word for word in tokens if word not in stop_words
and word.isalnum()]

    # Stemming
    stemmer = PorterStemmer()
    stemmed_tokens = [stemmer.stem(word) for word in filtered_tokens]

    # Lemmatization
    lemmatizer = WordNetLemmatizer()
    lemmatized_tokens = [lemmatizer.lemmatize(word) for word in
filtered_tokens]

    return {
        "Original Text": text,
        "Tokens": tokens,
        "Filtered Tokens": filtered_tokens,
        "Stemmed Tokens": stemmed_tokens,
        "Lemmatized Tokens": lemmatized_tokens
    }

def preprocess_text_stanza(text):
    doc = nlp(text)

```

```

tokens = [word.text for sent in doc.sentences for word in sent.words]
lemmas = [word.lemma for sent in doc.sentences for word in
sent.words]
pos_tags = [word.upos for sent in doc.sentences for word in
sent.words]

dependencies = [(word.head, word.deprel, word.id) for sent in
doc.sentences for word in sent.words]

ner_tags = [(ent.text, ent.type) for ent in doc.ents]

return {
    "Original Text": text,
    "Tokens": tokens,
    "Lemmas": lemmas,
    "POS Tags": pos_tags,
    "Dependencies": dependencies,
    "NER Tags": ner_tags
}

# Stanza MultilingualPipeline
from stanza.pipeline.multilingual import MultilingualPipeline
lang_id_config = {"langid_clean_text": True}
lang_configs = {"en": {"processors": {"ner": "conll03"}}}
nlp_multi = MultilingualPipeline(lang_id_config=lang_id_config,
lang_configs=lang_configs)

def preprocess_multilingual_text(text_list):
    docs = nlp_multi(text_list)
    results = []
    for doc in docs:
        if hasattr(doc, 'sentences'):
            for sent in doc.sentences:
                results.append({
                    "Text": sent.text,
                    "Language": doc.lang,
                    "Dependencies": sent.dependencies_string() if hasattr(sent,
'dependencies_string') else "N/A"
                })
        else:
            results.append({
                "Text": doc.text,
                "Language": doc.lang,
                "Dependencies": "N/A"
            })

```



```

    return results

# Sample text
test_text = "Natural Language Processing (NLP) is a field of AI that focuses
on the interaction between computers and humans through natural
language."

# Preprocess text using NLTK
result_nltk = preprocess_text(test_text)

# Preprocess text using Stanza
result_stanza = preprocess_text_stanza(test_text)

# Preprocess text using Stanza MultilingualPipeline
result_multilingual = preprocess_multilingual_text([test_text, "Bonjour le
monde!", "Hallo, wie geht's?"])

# Print results
print("=== NLTK Preprocessing ===")
for step, output in result_nltk.items():
    print(f"{step}:\n{output}\n")

print("=== Stanza Preprocessing ===")
for step, output in result_stanza.items():
    print(f"{step}:\n{output}\n")

print("=== Stanza Multilingual Preprocessing ===")
for res in result_multilingual:
    print(res)

```

3. Berikut Output program:

```

=== NLTK Preprocessing ===
Original Text:
natural language processing nlp is a field of ai that focuses on the
interaction between computers and humans through natural language

Tokens:
['natural', 'language', 'processing', 'nlp', 'is', 'a', 'field', 'of', 'ai', 'that',
'focuses', 'on', 'the', 'interaction', 'between', 'computers', 'and', 'humans',
'through', 'natural', 'language']

Filtered Tokens:

```

['natural', 'language', 'processing', 'nlp', 'field', 'ai', 'focuses', 'interaction', 'computers', 'humans', 'natural', 'language']

Stemmed Tokens:

['natur', 'languag', 'process', 'nlp', 'field', 'ai', 'focus', 'interact', 'comput', 'human', 'natur', 'languag']

Lemmatized Tokens:

['natural', 'language', 'processing', 'nlp', 'field', 'ai', 'focus', 'interaction', 'computer', 'human', 'natural', 'language']

=== Stanza Preprocessing ===

Original Text:

Natural Language Processing (NLP) is a field of AI that focuses on the interaction between computers and humans through natural language.

Tokens:

['Natural', 'Language', 'Processing', '(', 'NLP', ')', 'is', 'a', 'field', 'of', 'AI', 'that', 'focuses', 'on', 'the', 'interaction', 'between', 'computers', 'and', 'humans', 'through', 'natural', 'language', '.']

Lemmas:

['Natural', 'language', 'processing', '(', 'NLP', ')', 'be', 'a', 'field', 'of', 'AI', 'that', 'focus', 'on', 'the', 'interaction', 'between', 'computer', 'and', 'human', 'through', 'natural', 'language', '.']

POS Tags:

['ADJ', 'NOUN', 'NOUN', 'PUNCT', 'PROPN', 'PUNCT', 'AUX', 'DET', 'NOUN', 'ADP', 'PROPN', 'PRON', 'VERB', 'ADP', 'DET', 'NOUN', 'ADP', 'NOUN', 'CCONJ', 'NOUN', 'ADP', 'ADJ', 'NOUN', 'PUNCT']

Dependencies:

[(2, 'amod', 1), (3, 'compound', 2), (9, 'nsubj', 3), (5, 'punct', 4), (3, 'appos', 5), (5, 'punct', 6), (9, 'cop', 7), (9, 'det', 8), (0, 'root', 9), (11, 'case', 10), (9, 'nmod', 11), (13, 'nsubj', 12), (11, 'acl:relcl', 13), (16, 'case', 14), (16, 'det', 15), (13, 'obl', 16), (18, 'case', 17), (16, 'nmod', 18), (20, 'cc', 19), (18, 'conj', 20), (23, 'case', 21), (23, 'amod', 22), (13, 'obl', 23), (9, 'punct', 24)]

NER Tags:

[('Natural Language Processing', 'ORG'), ('NLP', 'ORG')]

=== Stanza Multilingual Preprocessing ===

{'Text': 'Natural Language Processing (NLP) is a field of AI that focuses on the interaction between computers and humans through natural

```

language.', 'Language': 'en', 'Dependencies': "({'Natural', 2,
'amod'})\n({'Language', 3, 'compound'})\n({'Processing', 9, 'nsubj'})\n({'(', 5,
'punct'})\n({'NLP', 3, 'appos'})\n({'', 5, 'punct'})\n({'is', 9, 'cop'})\n({'a', 9,
'det'})\n({'field', 0, 'root'})\n({'of', 11, 'case'})\n({'AI', 9, 'nmod'})\n({'that', 13,
'nsubj'})\n({'focuses', 11, 'acl:relcl'})\n({'on', 16, 'case'})\n({'the', 16,
'det'})\n({'interaction', 13, 'obl'})\n({'between', 18, 'case'})\n({'computers', 16,
'nmod'})\n({'and', 20, 'cc'})\n({'humans', 18, 'conj'})\n({'through', 23,
'case'})\n({'natural', 23, 'amod'})\n({'language', 13, 'obl'})\n({'.', 9, 'punct'})"}
{'Text': 'Bonjour le monde!', 'Language': 'fr', 'Dependencies': "({'Bonjour',
3, 'discourse'})\n({'le', 3, 'det'})\n({'monde', 0, 'root'})\n({'!', 1, 'punct'})"}
{'Text': "Hallo, wie geht's?", 'Language': 'nl', 'Dependencies': "({'Hallo', 4,
\parataxis'})\n({'\,', 1, 'punct'})\n({'wie', 4, 'nsubj'})\n({'geht's', 0,
'root'})\n({'?', 4, 'punct'})"}

```

PERTEMUAN 4

1.7 Tugas

1. Cari 10 jurnal terkait dengan perkembangan Feature Extraction di NLP
2. Buatlah program untuk melakukan Feature Extraction pada corpus/dataset

1.8 10 Jurnal Terkait Dengan Perkembangan *Feature Extraction* Di NLP

Feature extraction (ekstraksi fitur) dalam Natural Language Processing (NLP) adalah proses mengubah data teks mentah menjadi representasi numerik yang dapat dipahami dan diolah oleh mesin atau algoritma machine learning. Berikut adalah jurnal terkait dengan perkembangan Feture Extraction

No	Penulis/Sumber	Tahun	Judul Penelitian	Model	Pembahasan	Sumber
1	Xiaobing Sun, Jiayi Li, Wei Lu	2023	Unraveling Feature Extraction Mechanisms in Neural Networks	LLM, NLP	Analisis teoretis mekanisme ekstraksi fitur pada neural networks, efek fungsi aktivasi, self-attention, dg kesimpulan Multiplication-based models unggul di n-gram, ReLU bias fitur, insight untuk LLM	https://aclanthology.org/2023.emnlp-main.650/
2	S. Palomino, A. Aider	2023	Natural Language Processing: Recent Development and Applications	NLP, BERT	Survei teknik NLP, termasuk ekstraksi fitur klasik dan deep learning dg kesimpulan Deep learning (Word2Vec, BERT) mengungguli teknik klasik seperti BoW, TF-IDF	https://www.mdpi.com/2076-3417/13/11/6438
3	Xiaobing Sun, Jiayi Li, Wei Lu	2021	Advances In Natural Language Processing: A Survey Of Techniques	NLP	Survei perkembangan teknik NLP, termasuk evolusi feature extraction dg kesimpulan Transformer dan contextual embedding merevolusi ekstraksi fitur	https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5064608

No	Penulis/Sum ber	Tahun	Judul Penelitian	Mod el	Pembahasan	Sumber
4	S. Palomino, A. Aider	2023	Preprocessing for Social Media Sentiment Analysis	<i>NLP</i>	Studi pengaruh preprocessing dan feature extraction pada data media sosial dg kesimpulan Preprocessing dan feature extraction meningkatkan performa klasifikasi sentimen	https://www.mdpi.com/2076-3417/13/11/6438
5	S. Palomino, A. Aider	2023	Preprocessing Impact on Sentiment Classifier Performance	<i>NLP</i>	Pengaruh preprocessing dan feature extraction pada performa klasifikasi sentimen dg kesimpulan Kombinasi preprocessing dan feature extraction signifikan meningkatkan akurasi	https://www.mdpi.com/2076-3417/13/11/6438
6	D. Supriyanto et al.	2024	Development of extraction features for Detecting Adolescent Personality with Machine Learning Algorithms	Naïv e Baye s	Kombinasi TF-IDF + N- Gram Z untuk klasifikasi kepribadian remaja dg kesimpulan TF-IDF+N- Gram Z + Naïve Bayes capai akurasi hingga 98%	https://joiv.org/index.php/joiv/article/view/3091/0
7	Chao et al.	2024	Research on Features Extraction and Classification for Images based on Transformer Learning	<i>CNN</i> , <i>ML</i>	Framework transformer- based untuk ekstraksi fitur dan klasifikasi pada data gambar dg kesimpulan Transformer-based extraction unggul dibanding CNN/ML klasik pada dataset gambar	https://proceedings.mlr.press/v245/chao24a.html
8	Y. Zhang et al.	2023	TwIdw-A Novel Method for Feature Extraction from Unstructured Text for Fake News Classification	<i>NLP</i>	Pengembangan metode TwIdw untuk ekstraksi fitur pada deteksi berita palsudg kesimpulan TwIdw efektif meningkatkan akurasi deteksi fake news	https://www.mdpi.com/2076-3417/13/11/6438
9	S. Palomino, A. Aider	2024	Pre-processing methodologies in sentiment analysis	<i>NLP</i>	Studi berbagai teknik preprocessing dan feature extraction untuk analisis sentimen dg kesimpulan Kombinasi teknik menghasilkan hasil terbaik	https://www.mdpi.com/2076-3417/13/11/6438

No	Penulis/Sum ber	Tahun	Judul Penelitian	Mod el	Pembahasan	Sumber
10	S. Palomino, A. Aider	2023	Integrating Feature Extraction and Optimal Selection to Combat Fake News	NLP	Dual-stage feature extraction dan seleksi fitur untuk deteksi fake news dg kesimpulan Seleksi fitur setelah ekstraksi meningkatkan akurasi deteksi fake news	https://journal.esrgroups.org/jes/article/view/6017

1.9 Program untuk melakukan *Feature Extraction* pada corpus/dataset

1. Berikut link GoggleColab terkait dengan *Feature Extraction*

<https://colab.research.google.com/drive/1eJrrKQScuqNCR3XhDgQblpJgkk-tZmXL?usp=sharing>

2. Script

```
import string
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk import pos_tag

# Pastikan nltk resources sudah diunduh
import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
nltk.download('punkt_tab')
nltk.download('averaged_perceptron_tagger_eng')

def preprocess_text(text):
    text = text.lower()
    text = text.translate(text.maketrans("", "", string.punctuation))
    tokens = word_tokenize(text)
    stop_words = set(stopwords.words('english'))
    filtered_tokens = [word for word in tokens if word not in stop_words and
                      word.isalnum()]

    ps = PorterStemmer()
    stemmed_tokens = [ps.stem(word) for word in filtered_tokens]
    lemmatizer = WordNetLemmatizer()
```

```

    lemmatized_tokens = [lemmatizer.lemmatize(word) for word in
filtered_tokens]
    pos_tagged_tokens1 = pos_tag(stemmed_tokens)
    pos_tagged_tokens2 = pos_tag(lemmatized_tokens)
    return {
        "original_text": text,
        "tokens": tokens,
        "filtered_tokens": filtered_tokens,
        "stemmed_tokens": stemmed_tokens,
        "lemmatized_tokens": lemmatized_tokens,
        "pos_tagged_tokens1": pos_tagged_tokens1,
        "pos_tagged_tokens2": pos_tagged_tokens2,
    }

```

```

sentences = [
    "Saya suka belajar data science.",
    "Python adalah bahasa pemrograman yang populer.",
    "Saya menggunakan Python untuk analisis data.",
    "Analisis data membantu dalam pengambilan keputusan.",
    "Machine learning adalah cabang dari kecerdasan buatan.",
    "Algoritma machine learning bisa memprediksi hasil.",
    "Saya tertarik pada teknologi baru.",
    "Kecerdasan buatan memiliki banyak aplikasi.",
    "Belajar data science sangat menarik.",
    "Saya mengikuti kursus online tentang machine learning."
]

```

```

for i, sentence in enumerate(sentences):
    result = preprocess_text(sentence)
    print(f"Hasil preprocessing kalimat {i+1}:\n")
    for key, value in result.items():
        print(f"{key}:\n{value}\n")
    print("-" * 50)

```

```

import re
def preprocess_text(text):
    text = text.lower()
    text = text.translate(text.maketrans("", "", string.punctuation))
    text = re.sub(r'\W', ' ', text)
    words = word_tokenize(text)
    stop_words = set(stopwords.words('indonesian'))
    words = [word for word in words if word not in stop_words]
    stemmer = PorterStemmer()
    words = [stemmer.stem(word) for word in words]
    return ' '.join(words)

```

```

preprocessed_sentences = [preprocess_text(sentence) for sentence in
sentences]

from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
X_bow = vectorizer.fit_transform(preprocessed_sentences)

print("Vocabulary:\n", vectorizer.get_feature_names_out())
print('\nVector vocabulary:\n',vectorizer.vocabulary_)
print("\nBag of Words:\n", X_bow.toarray())

from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer()
X_tfidf = tfidf_vectorizer.fit_transform(preprocessed_sentences)

print("Vocabulary:\n", tfidf_vectorizer.get_feature_names_out())
print('\nVector vocabulary:\n',tfidf_vectorizer.vocabulary_)
print("\nTF-IDF:\n", X_tfidf.toarray())

```

3. Output dari aplikasi

Output dari pemograman diatas:
 Hasil preprocessing kalimat 1:

original_text:

saya suka belajar data science

tokens:

['saya', 'suka', 'belajar', 'data', 'science']

filtered_tokens:

['saya', 'suka', 'belajar', 'data', 'science']

stemmed_tokens:

['saya', 'suka', 'belajar', 'data', 'scienc']

lemmatized_tokens:

['saya', 'suka', 'belajar', 'data', 'science']

pos_tagged_tokens1:

[('saya', 'NN'), ('suka', 'NN'), ('belajar', 'NN'), ('data', 'NNS'), ('scienc', 'NN')]

pos_tagged_tokens2:

[('saya', 'NN'), ('suka', 'NN'), ('belajar', 'NN'), ('data', 'NNS'), ('science', 'NN')]

Vocabulary:

['algoritma' 'analisi' 'aplikasi' 'bahasa' 'belajar' 'buatan' 'cabang'
'data' 'hasil' 'kecerdasan' 'keputusan' 'kursu' 'learn' 'machin'
'membantu' 'memiliki' 'memprediksi' 'menarik' 'mengikuti' 'onlin'
'pemrograman' 'pengambilan' 'popul' 'python' 'scienc' 'suka' 'teknolog'
'tertarik']

Vector vocabulary:

{ 'suka': 25, 'belajar': 4, 'data': 7, 'scienc': 24, 'python': 23, 'bahasa': 3, 'pemrograman': 20, 'popul': 22, 'analisi': 1, 'membantu': 14, 'pengambilan': 21, 'keputusan': 10, 'machin': 13, 'learn': 12, 'cabang': 6, 'kecerdasan': 9, 'buatan': 5, 'algoritma': 0, 'memprediksi': 16, 'hasil': 8, 'tertarik': 27, 'teknolog': 26, 'memiliki': 15, 'aplikasi': 2, 'menarik': 17, 'mengikuti': 18, 'kursu': 11, 'onlin': 19 }

Bag of Words:

[[0 0 0 0 1 0 0 1 0 1 1 0 0]
[0 0 0 1 0 1 0 1 1 0 0 0 0]
[0 1 0 0 0 0 0 1 0 1 0 0 0 0]
[0 1 0 0 0 0 0 1 0 0 1 0 0 0 1 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 1 0 0 1 0 0 1 1 0]
[1 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 1 1]
[0 0 1 0 0 1 0 0 0 1 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 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 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 1 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0]]

Vocabulary:

['algoritma' 'analisi' 'aplikasi' 'bahasa' 'belajar' 'buatan' 'cabang'
'data' 'hasil' 'kecerdasan' 'keputusan' 'kursu' 'learn' 'machin'
'membantu' 'memiliki' 'memprediksi' 'menarik' 'mengikuti' 'onlin'
'pemrograman' 'pengambilan' 'popul' 'python' 'scienc' 'suka' 'teknolog'
'tertarik']

TF-IDF:

```

[[0.    0.    0.    0.    0.5007009 0.
  0.    0.3894615 0.    0.    0.    0.
  0.    0.    0.    0.    0.    0.
  0.    0.    0.    0.    0.    0.
  0.5007009 0.58899657 0.    0.    ]
[0.    0.    0.    0.5182909 0.    0.
  0.    0.    0.    0.    0.    0.
  0.    0.    0.    0.    0.    0.
  0.    0.    0.5182909 0.    0.5182909 0.44059462
  0.    0.    0.    0.    ]
[0.    0.6195754 0.    0.    0.    0.
  0.    0.48192597 0.    0.    0.    0.
  0.    0.    0.    0.    0.    0.
  0.    0.    0.    0.    0.    0.6195754
  0.    0.    0.    0.    ]
[0.    0.41679765 0.    0.    0.    0.
  0.    0.32419882 0.    0.    0.49029748 0.
  0.    0.    0.49029748 0.    0.    0.
  0.    0.    0.    0.49029748 0.    0.
  0.    0.    0.    0.    ]
[0.    0.    0.    0.    0.    0.45108142
  0.53062699 0.    0.    0.45108142 0.    0.
  0.39464294 0.39464294 0.    0.    0.    0.
  0.    0.    0.    0.    0.    0.
  0.    0.    0.    0.    ]
[0.49348778 0.    0.    0.    0.    0.
  0.    0.    0.49348778 0.    0.    0.
  0.36702141 0.36702141 0.    0.    0.49348778 0.
  0.    0.    0.    0.    0.    0.
  0.    0.    0.    0.    ]
[0.    0.    0.    0.    0.    0.

```

0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.70710678	0.70710678]		
[0.	0.	0.53874817	0.	0.	0.45798516
0.	0.	0.	0.45798516	0.	0.
0.	0.	0.	0.53874817	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.]	
[0.	0.	0.	0.	0.5007009	0.
0.	0.3894615	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.58899657
0.	0.	0.	0.	0.	0.
0.5007009	0.	0.	0.]	
[0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.49348778
0.36702141	0.36702141	0.	0.	0.	0.
0.49348778	0.49348778	0.	0.	0.	0.
0.	0.	0.	0.]]	

PERTEMUAN 5

1.10 Tugas

1. Cari 10 jurnal terkait dengan Information Retrieval!
2. Buatlah program Inverted Index untuk menampung corpus/dataset!

1.11 10 jurnal terkait dengan Information Retrieval

No	Penulis/Sumber	Tahun	Judul Penelitian	Pembahasan	Sumber
1	Hope Nabankema	2024	Evaluation of Natural Language Processing Techniques for Information Retrieval	Evaluasi teknik NLP (tokenisasi, NER, semantic parsing, word embeddings) untuk IR di berbagai domain dg kesimpulan NLP signifikan meningkatkan akurasi dan efisiensi IR, rekomendasi integrasi teknik	hybrid https://doi.org/10.47941/ejkm.1752
2	Chuang, Jackson & Jensen	2012	Topic Modeling for Enhanced Information Retrieval	Integrasi topic modeling dalam algoritma pencarian dengan kesimpulan Memperluas eksplorasi konten dan hasil pencarian tematis	https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3400142/
3	Luong, Pham & Manning	2013	Word Embeddings and Semantic Search for Information Retrieval	Penerapan embeddings untuk semantic search dan query expansion dengan kesimpulan Semantic search dengan embeddings meningkatkan relevansi hasil pencarian	https://aclanthology.org/D13-1170/
4	Nivre et al.	2020	Dependency Parsing in Information Retrieval	Analisis struktur sintaksis untuk meningkatkan pemahaman query dan ranking dokumen dengan kesimpulan Dependency parsing memperbaiki pencocokan query-dokumen kompleks	https://aclanthology.org/2020.lrec-1.348/
5	Le, Mikolov	2014	Distributed Representations	Doc2vec untuk representasi dokumen dalam IR dengan	https://arxiv.org/abs/1405.4053

No	Penulis/Sumber	Tahun	Judul Penelitian	Pembahasan	Sumber
			of Sentences and Documents	kesimpulan Meningkatkan akurasi pencarian dokumen berbasis semantik	
6	Blei, D.M.	2003	Topic Modeling and Information Retrieval	Penggunaan topic modeling untuk identifikasi tema dokumen dan peningkatan IR dengan kesimpulan Topic modeling meningkatkan clustering dan relevansi hasil pencarian	https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf
7	Mikolov et al.	2013	Word Embeddings for Information Retrieval	Penggunaan word2vec/GloVe untuk representasi semantik kata pada IR dengan kesimpulan Word embeddings meningkatkan relevansi ranking dan query expansion	https://arxiv.org/abs/1301.3781
8	Bast & Haussmann	2015	Semantic Parsing for Information Retrieval	Penggunaan semantic parsing untuk pemahaman query dan pencarian informasi pada data terstruktur/semi-terstruktur dengan kesimpulan Semantic parsing meningkatkan presisi dan recall IR	https://www.semanticscholar.org/paper/Parsing-as-Reduction-Bast-Haussmann/6e5e0e8b7b2c1b1b
9	Kim & Lee	2020	Meta-Analysis of NLP Techniques for Information Retrieval	Meta-analisis teknik NLP (tokenization, NER, semantic parsing) untuk peningkatan akurasi IR dengan kesimpulan Hybrid NLP methods (word embeddings + topic modeling) unggul dalam IR	https://carijournals.org/journals/index.php/EJIKM/article/view/1752
10	MDPI (Special Issue Editors)	2023	Natural Language Processing and Information Retrieval (Special Issue)	Kumpulan makalah terbaru tentang teori, model, dan aplikasi NLP untuk IR dengan kesimpulan NLP memperluas aplikasi IR di berbagai domain dan meningkatkan semantic search	https://www.mdpi.com/journal/electronics/special_issues/natural_language_information_retrieval

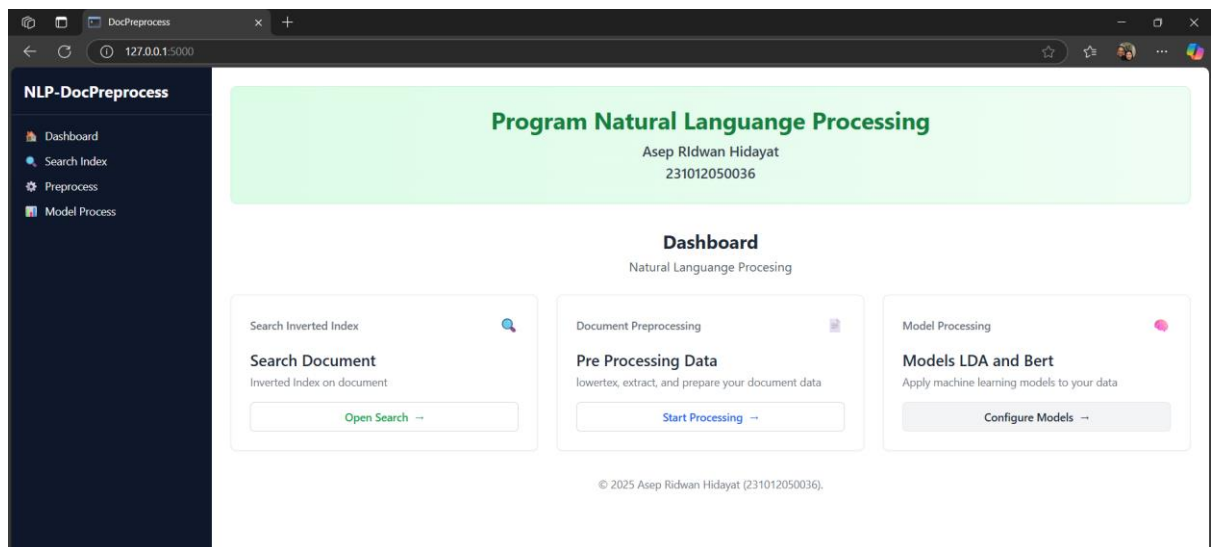
1.12 Buatlah program Inverted Index untuk menampung corpus/dataset!

Link Aplikasi di GoogleDrive:

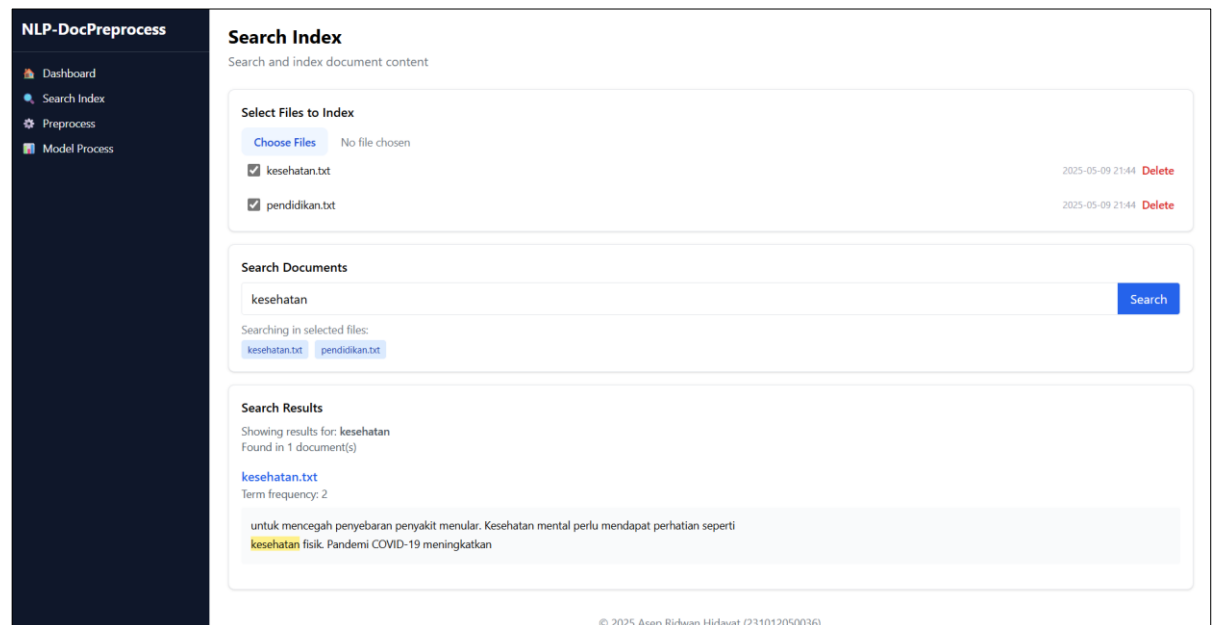
<https://drive.google.com/file/d/1s7bs4lmRdZfChUXvOZUDU3RbnLBRMUvi/view?usp=sharing>

1.12.1 Tampilan Program

Dashboaard



1.12.2 Menu Search Index dan output dari seach index



1.12.3 Script aplikasi

Back end App.py dengan flask

```
from flask import Flask, render_template, request, redirect, url_for, flash, jsonify
import os
from werkzeug.utils import secure_filename
from datetime import datetime
import re
from collections import defaultdict
from docx import Document
import pandas as pd
from PyPDF2 import PdfReader
import odf.opendocument
from odf import text

app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = 'uploads'
app.config['SECRET_KEY'] = 'your-secret-key'
app.config['MAX_CONTENT_LENGTH'] = 50 * 1024 * 1024 # 50MB
# Ensure upload folder exists
os.makedirs(app.config['UPLOAD_FOLDER'], exist_ok=True)

# Inverted index storage
inverted_index = defaultdict(dict)

def index_file(filename):
    """Index a file and add to inverted index"""
    filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)

    try:
        text = ""
        # Handle PDF
        if filename.lower().endswith('.pdf'):
            reader = PdfReader(filepath)
            for page in reader.pages:
                text += page.extract_text()
            return text
        # Handle other file types
        else:
            doc = Document(filepath)
            return doc.text
    except Exception as e:
        print(f"Error indexing file {filename}: {e}")
        return ""
```

```

with open(filepath, 'rb') as f:
    pdf = PdfReader(f)
    text = " ".join([page.extract_text() for page in pdf.pages])

# Handle Word DOCX
elif filename.lower().endswith('.docx'):
    doc = Document(filepath)
    text = " ".join([para.text for para in doc.paragraphs])

# Handle Legacy Word DOC (membutuhkan antiword)
elif filename.lower().endswith('.doc'):
    try:
        text = os.popen(f'antiword "{filepath}"').read()
    except:
        text = ""

# Handle Excel
elif filename.lower().endswith(('xlsx', 'xls')):
    df = pd.read_excel(filepath, sheet_name=None)
    text = " ".join(
        [str(cell) for sheet in df.values()
         for row in sheet.values
         for cell in row]
    )

# Handle OpenDocument (ODT)
elif filename.lower().endswith('.odt'):
    doc = odf.opendocument.load(filepath)
    text = " ".join(
        [text.Text(e).toString()
         for e in doc.getElementsByType(text.P)]
    )

# Handle plain text

```



```

elif filename.lower().endswith('.txt'):
    with open(filepath, 'r', encoding='utf-8') as f:
        text = f.read()

    # Bersihkan teks dari karakter khusus
    text = re.sub(r'\s+', ' ', text).strip()

    if not text:
        app.logger.warning(f"No text extracted from {filename}")
        return

    # Proses teks yang sudah diekstrak
    words = re.findall(r'\w+', text.lower())

    # Update inverted index
    for word in set(words):
        term_frequency = words.count(word)
        inverted_index[word][filename] = term_frequency

except Exception as e:
    app.logger.error(f"Error processing {filename}: {str(e)}")
    flash(f"Error processing {filename}", "error")

def search_in_index(query, selected_files=None):
    """Search the inverted index for the query"""
    results = []
    query_terms = re.findall(r'\w+', query.lower())

    for term in query_terms:
        if term in inverted_index:
            for filename, term_frequency in inverted_index[term].items():
                # If specific files are selected, only search in those
                if selected_files and filename not in selected_files:
                    continue

```

```

        # Get snippets of text around the search term
        snippets = get_snippets(filename, term)

        # Add to results
        results.append({
            'filename': filename,
            'term': term,
            'term_frequency': term_frequency,
            'snippets': snippets
        })

    return results

def get_snippets(filename, term):
    """Get snippets of text around the search term (support multi-format)"""
    filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)
    text = ""

    try:
        # Replikasi logika ekstraksi teks dari index_file()
        if filename.lower().endswith('.pdf'):
            with open(filepath, 'rb') as f:
                pdf = PdfReader(f)
                text = " ".join([page.extract_text() for page in pdf.pages])

        elif filename.lower().endswith('.docx'):
            doc = Document(filepath)
            text = " ".join([para.text for para in doc.paragraphs])

        elif filename.lower().endswith('.xlsx'):
            df = pd.read_excel(filepath, sheet_name=None)
            text = " ".join([str(cell) for sheet in df.values() for row in sheet.values for
                             cell in row])

```

```

else: # Untuk format teks biasa
    with open(filepath, 'r', encoding='utf-8') as f:
        text = f.read()

    # Cari snippet dengan konteks
    pattern = re.compile(r'(\b\w+\W+){0,5}\b' + re.escape(term) +
r'\b(\W+\w+\b){0,5}', re.IGNORECASE)
    matches = pattern.finditer(text)

    snippets = []
    for match in matches:
        snippet = match.group(0)
        highlighted = snippet.replace(term, f'<span class="bg-yellow-
200">{term}</span>')
        snippets.append(highlighted)
        if len(snippets) >= 3:
            break

    return snippets

except Exception as e:
    app.logger.error(f"Error getting snippets from {filename}: {str(e)}")
    return []

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

@app.route('/preprocess')
def preprocess():
    return render_template("preprocessing.html")

@app.route('/model')
def model():

```

```

upload_dir = os.path.join(app.config['UPLOAD_FOLDER'])
filenames = [f for f in os.listdir(upload_dir) if f.endswith('.txt')]
return render_template("model.html", filenames=filenames)

@app.route('/search_menu', methods=['GET', 'POST'])
def search():
    # Handle search request
    if request.method == 'POST':
        query = request.form.get('query', "").strip()
        selected_files = request.form.get('selected_files', "")
        selected_files = selected_files.split(',') if selected_files else None

        app.logger.debug(f"selected_files: {selected_files}")

        if not query :
            flash('Please enter a search term', 'error')
            return redirect(url_for('search'))

        # Validasi untuk selected_files
        if not selected_files or all(file == " for file in selected_files):
            flash('At least one file must be selected', 'error')
            return redirect(url_for('search'))

        search_results = search_in_index(query, selected_files)
    else:
        search_results = None
        query = None

    # Get uploaded files list
    uploaded_files = []
    for filename in os.listdir(app.config['UPLOAD_FOLDER']):
        file_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
        if os.path.isfile(file_path):
            mtime = os.path.getmtime(file_path)

```

```

        uploaded_files.append({
            'name': filename,
            'date': datetime.fromtimestamp(mtime).strftime('%Y-%m-%d %H:%M')
        })

    # Sort by upload time descending
    uploaded_files.sort(key=lambda x: x['date'], reverse=True)

    return render_template(
        'search.html',
        uploaded_files=uploaded_files,
        search_results=search_results,
        search_query=query
    )

@app.route('/upload', methods=['POST'])
def upload_file():
    if 'files' in request.files:
        files = request.files.getlist('files')
        uploaded_count = 0

        for file in files:
            if file.filename != "":
                filename = secure_filename(file.filename)
                # Filter ekstensi file
                if not filename.lower().endswith(('.pdf', '.doc', '.docx', '.txt', '.xls', '.xlsx')):
                    continue

                filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)
                file.save(filepath)
                index_file(filename)
                uploaded_count += 1

        if uploaded_count > 0:
            flash(f'{uploaded_count} file(s) uploaded and indexed', 'success')

```

```

else:
    flash('No valid files uploaded', 'error')

return jsonify({'status': 'success'}), 200

@app.route('/delete/<filename>', methods=['POST'])
def delete_file(filename):
    # safe_filename = secure_filename(filename)
    file_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)

    app.logger.debug(f"Mencoba menghapus file: {file_path}")
    app.logger.debug(f"Filename asli: {filename}")
    app.logger.debug(f"Filename aman: {filename}")

    try:
        if os.path.exists(file_path):
            os.remove(file_path)
            # Remove from inverted index
            for term in list(inverted_index.keys()):
                if filename in inverted_index[term]:
                    del inverted_index[term][filename]
                    if not inverted_index[term]: # Remove term if no more documents
                        del inverted_index[term]
            app.logger.debug(f"File {file_path} berhasil dihapus")
            flash('File berhasil dihapus', 'success')
        else:
            app.logger.warning(f"File {file_path} tidak ditemukan")
            flash('File tidak ditemukan', 'error')
    except Exception as e:
        app.logger.error(f"Gagal menghapus file: {str(e)}", exc_info=True)
        flash(f"Gagal menghapus file: {str(e)}", 'error')

    return redirect(url_for('search'))

```

```

from flask import render_template
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
from bertopic import BERTopic

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
from bertopic import BERTopic

@app.route('/run_model', methods=['POST'])
def run_model():
    model_type = request.form.get('model_type')
    selected_files = request.form.getlist('selected_files')

    texts = []
    filenames = []

    for filename in selected_files:
        path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
        if os.path.isfile(path):
            try:
                with open(path, 'r', encoding='utf-8') as f:
                    content = f.read().strip()
                    if len(content) > 20:
                        texts.append(content)
                        filenames.append(filename)
            except Exception as e:
                app.logger.error(f"Error reading {filename}: {e}")

    if not texts:
        flash("Tidak ada dokumen valid yang dipilih atau isinya kosong.", "error")
        return redirect(url_for('model'))

    try:

```

```

doc_topics = []

if model_type == 'lda':
    vectorizer = CountVectorizer(stop_words='english')
    dtm = vectorizer.fit_transform(texts)
    lda = LatentDirichletAllocation(n_components=5, random_state=42)
    lda.fit(dtm)
    for i, doc in enumerate(dtm):
        topic_distribution = lda.transform(doc)
        dominant_topic = topic_distribution[0].argmax()
        probability = round(topic_distribution[0][dominant_topic], 2)
        doc_topics.append({
            'document': filenames[i],
            'topic': f"Topic {dominant_topic+1}",
            'probability': probability
        })

    topics = []
    for idx, topic in enumerate(lda.components_):
        keywords = [vectorizer.get_feature_names_out()[i] for i in
topic.argsort()[-5:]]
        topics.append({'topic': f"Topic {idx+1}", 'keywords': keywords})

    return render_template("model.html", topics=topics, doc_topics=doc_topics,
model_type='lda')

elif model_type == 'bertopic':
    model = BERTopic()
    topics, probs = model.fit_transform(texts)
    topic_info = model.get_topic_info()

    for i, (topic, prob) in enumerate(zip(topics, probs)):
        doc_topics.append({
            'document': filenames[i],

```



```

        'topic': f"Topic {topic}",
        'probability': round(prob if prob is not None else 0.0, 2)
    })

    return render_template("model.html", topics=topic_info.to_dict('records'),
doc_topics=doc_topics, model_type='bertopic')

except Exception as e:
    app.logger.error(f"Model error: {e}", exc_info=True)
    flash(f"Terjadi error saat memproses model: {str(e)}", "error")
    return redirect(url_for('model'))

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

```

Front end template layout menu search menggunakan Html dan Tailwind (css)

```

{% extends "base.html" %}

{% block content %}

<h1 class="text-2xl font-bold mb-1">Search Index</h1>
<p class="text-gray-500 mb-6">Search and index document content</p>

<!-- upload file -->
<div class="bg-white border rounded-lg p-4 shadow">
    <h2 class="font-semibold mb-2">Select Files to Index</h2>

    <!-- Form Upload File -->
    <form method="POST" action="/upload" enctype="multipart/form-data" id="auto-
upload-form">
        <div class="flex items-center">

```

```

<input type="file"
      name="files"
      id="file-upload"
      multiple
      class="block w-full text-sm text-gray-500
            file:mr-4 file:py-2 file:px-4
            file:rounded-lg file:border-0
            file:text-sm file:font-semibold
            file:bg-blue-50 file:text-blue-700
            hover:file:bg-blue-100">

</div>
</form>

<!-- <form method="POST" action="/upload" enctype="multipart/form-data"
class="mb-4" id="auto-upload-form">
  <div class="flex items-center gap-2">
    <input type="file"
          name="files"
          id="file-upload"
          multiple
          class="block w-full text-sm text-gray-500
                file:mr-4 file:py-2 file:px-4
                file:rounded-lg file:border-0
                file:text-sm file:font-semibold
                file:bg-blue-50 file:text-blue-700
                hover:file:bg-blue-100">

    <button type="submit"
            class="bg-blue-600 text-white px-4 py-2 rounded-lg hover:bg-blue-700
            transition-colors">
      Upload
    </button>
  </div>
</form> -->

<!-- File List -->

```

```

<ul class="space-y-2 text-sm max-h-64 overflow-y-auto">
  {% if uploaded_files %}
    {% for file in uploaded_files %}
      <li class="flex items-center justify-between hover:bg-gray-50 p-2 rounded">
        <label class="flex items-center space-x-2 w-full">
          <input type="checkbox"
            name="selected_files"
            value="{{ file.name }}"
            class="h-4 w-4 file-checkbox"
            data-filename="{{ file.name }}">
          <span class="truncate">{{ file.name }}</span>
        </label>
        <div class="flex items-center gap-2">
          <span class="text-gray-400 text-xs whitespace-nowrap">{{ file.date
}}</span>
          <form action="{{ url_for('delete_file', filename=file.name) }}"
method="POST">
            <button type="submit"
              class="text-red-600 hover:text-red-800 text-sm font-medium"
              onclick="return confirm('Are you sure you want to delete this file?')">
              Delete
            </button>
          </form>
        </div>
      </li>
    {% endfor %}
  {% else %}
    <li class="text-gray-400 text-center py-4">No files uploaded yet</li>
  {% endif %}
</ul>
</div>

<!-- search file -->
<div class="bg-white border rounded-lg p-4 shadow mt-4">

```

```

        <label for="search" class="block font-semibold mb-2">Search
Documents</label>
        <form id="search-form" method="POST" action="{{ url_for('search') }}">
        <div class="flex items-center border rounded overflow-hidden">
            <input type="text"
                id="search"
                name="query"
                placeholder="Search indexed documents..."
                class="flex-1 px-3 py-2 outline-none"
                required>
            <button type="submit" class="bg-blue-600 text-white px-4 py-2 hover:bg-blue-
700">Search</button>
        </div>
        <div id="selected-files-container" class="mt-2 hidden">
            <p class="text-sm text-gray-500 mb-1">Searching in selected files:</p>
            <div id="selected-files-list" class="flex flex-wrap gap-2"></div>
            <input type="hidden" id="selected-files-input" name="selected_files">
        </div>
        </form>
    </div>
    <!-- end search file -->

    <!-- <div class="grid grid-cols-1 md:grid-cols-2 gap-4"> -->

    <!-- result file -->
    <div class="bg-white border rounded-lg p-4 shadow mt-4">
        <h2 class="font-semibold mb-2">Search Results</h2>
        {% if search_results %}
        <div class="mb-4">
            <p class="text-sm text-gray-600">
                Showing results for: <span class="font-semibold">{{ search_query }}</span>
            </p>

```

```

<p class="text-sm text-gray-600">
  Found in {{ search_results|length }} document(s)
</p>
</div>

<div class="space-y-4">
  {% for result in search_results %}
    <div class="border-b pb-4 last:border-b-0">
      <h3 class="font-medium text-blue-600">{{ result.filename }}</h3>
      <p class="text-sm text-gray-600 mb-2">Term frequency: {{
result.term_frequency }}</p>
      <div class="text-sm bg-gray-50 p-3 rounded">
        {% for line in result.snippets %}
          <p class="mb-1">{{ line|safe }}</p>
        {% endfor %}
      </div>
    </div>
    {% endif %}
  {% endfor %}
</div>

{% else %}
  {% if search_query %}
    <p class="text-gray-500">No results found for "{{ search_query }}"</p>
  {% else %}
    <p class="text-gray-500">Enter a search term to find content in indexed
documents</p>
  {% endif %}
{% endif %}
</div>

<!--end result file -->
<!-- </div> -->

<div class="space-y-4">

```

```

    {% with messages = get_flashed_messages(with_categories=true) %}
    {% if messages %}
        {% for category, message in messages %}
            <div class="mb-4 p-3 rounded-lg bg-{{ category }}--100 text-{{ category }}-800">
                <h3 class="font-medium text-blue-600"> {{ message }}</h3>
            </div>
        {% endfor %}
    {% endif %}
    {% endwith %}
</div>

<script>
document.addEventListener('DOMContentLoaded', function() {
    const checkboxes = document.querySelectorAll('.file-checkbox');
    const selectedFilesContainer = document.getElementById('selected-files-container');
    const selectedFilesList = document.getElementById('selected-files-list');
    const selectedFilesInput = document.getElementById('selected-files-input');
    const fileUpload = document.getElementById('file-upload');
    const uploadForm = document.getElementById('auto-upload-form');

    checkboxes.forEach(checkbox => {
        checkbox.addEventListener('change', updateSelectedFiles);
    });

    function updateSelectedFiles() {
        const selectedFiles = Array.from(document.querySelectorAll('.file-checkbox:checked'))
            .map(checkbox => checkbox.dataset.filename);

        if (selectedFiles.length > 0) {
            selectedFilesContainer.classList.remove('hidden');
            selectedFilesList.innerHTML = "";

```

```

selectedFiles.forEach(file => {
  const fileTag = document.createElement('span');
  fileTag.className = 'bg-blue-100 text-blue-800 text-xs px-2 py-1 rounded';
  fileTag.textContent = file;
  selectedFilesList.appendChild(fileTag);
});
selectedFilesInput.value = selectedFiles.join(',');
} else {
  selectedFilesContainer.classList.add('hidden');
  selectedFilesInput.value = "";
}
}

fileUpload.addEventListener('change', function() {
  if(this.files.length > 0) {
    const formData = new FormData(uploadForm);

    fetch('/upload', {
      method: 'POST',
      body: formData
    })
    .then(response => {
      if(response.ok) {
        window.location.reload(); // Reload untuk update list file
      } else {
        alert('Upload failed');
      }
    })
    .catch(error => {
      console.error('Error:', error);
      alert('Upload error');
    });
  }
});

```

```
});  
</script>  
  
{% endblock %}
```


PERTEMUAN 6

1.13 Tugas

1. Buatlah Model LDA (Dataset bahasa Indonesia&Inggris)
2. Cari 10 Jurnal terkait pemanfaatan Topic Model (LDA dan turunannya/related work)
3. Buat Interface program

1.14 10 Jurnal terkait pemanfaatan Topic Model (LDA dan turunannya/related work)

No	Penulis/Sum ber	Tahun	Judul Penelitian	Mod el	Pembahasan	Sumber
1	Maarten Grootendorst et al.	2025	AI-powered topic modeling: comparing LDA and BERTopic		Perbandingan LDA (MALLET) dan BERTopic (BioBERT, UMAP, HDBSCAN) dengan integrasi AI (ChatGPT-4-Turbo) pada data medis dengan kesimpulan BERTopic lebih baik dalam koherensi semantik dan interpretabilitas; LDA butuh preprocessing manual; AI meningkatkan hasil	https://www.ebm-journal.org/journals/experimental-biology-and-medicine/articles/10.3389/ebm.2025.10389/full
2	Xuefeng Zhu et al.	2024	Investigating topic modeling techniques through evaluation of short texts		Evaluasi LDA dan NMF pada dataset teks pendek (SemEval 2016, BBC News) dengan clustering dan silhouette analysis Model evaluasi baru mengungguli LDA dan NMF; penemuan topik baru	https://www.nature.com/articles/s41598-024-61738-4
3	Maarten Grootendorst et al.	2025	AI-powered topic modeling: comparing LDA and BERTopic		Sama seperti jurnal SEBM, fokus pada analisis topik terkait	https://pmc.ncbi.nlm.nih.gov/articles/PMC11906279/

No	Penulis/Sum ber	Tahun	Judul Penelitian	Mod el	Pembahasan	Sumber
					<p>risiko kardiovaskular opioid pada wanita</p> <p>BERTopic menghasilkan cluster lebih kompak dan interpretasi otomatis dengan AI</p>	
4	Abdullah Al Mamun et al.	2024	Evaluating the latest trends of Industry 4.0 based on LDA topic model		<p>LDA diterapkan untuk mengidentifikasi pola tersembunyi dalam penelitian Industry 4.0</p> <p>LDA efektif mengungkap tren riset Industry 4.0</p>	https://dl.acm.org/doi/10.1007/s11227-024-06247-x
5	S. S. S. R. K. Prasad et al.	2025	A Performance- Driven Exploration of Combining Topic Modeling and Deep Learning		<p>Kombinasi LDA dan deep learning untuk peningkatan kualitas topic modeling</p> <p>Kombinasi metode meningkatkan akurasi dan interpretabilitas topik</p>	https://www.temjournal.com/content/141/TEMJournalFebruary2025_511_527.pdf
6	S. Wang et al.	2024	A topic modeling approach for analyzing and categorizing electronic health records		<p>Topic modeling untuk analisis dan kategorisasi data rekam medis elektronik</p> <p>Model topic membantu klasifikasi dan ekstraksi informasi medis</p>	https://www.nature.com/articles/s41598-024-83743-3
7	Md. Tanjim Hossain et al.	2024	Combining BERT with LDA: Improved Topic Modeling in Bengali Language	Integrasi embeddings LDA untuk modeling Bengali	BERT dengan topic bahasa	https://www.iaeng.org/IJCS/issues_v52/issue_2/IJCS_52_2_11.pdf

No	Penulis/Sum ber	Tahun	Judul Penelitian	Mod el	Pembahasan	Sumber
					Peningkatan kualitas topik dan interpretasi dibanding LDA tradisional	
8	Y. Zhang et al.	2024	Application of structural topic modeling in a literature review of air pollution research		Structural Topic Modeling (STM) diterapkan untuk review literatur polusi udara STM efektif mengidentifikasi tema dan tren riset polusi udara	https://www.sciencedirect.com/science/article/abs/pii/S096969972400173X
9	Feliciaa Muhammad Rizky Pribadi	2024	<i>Analisis Interaksi Pengguna Sosial Media Sekolah di Palembang Berdasarkan Topik dengan hLDA dan SVM</i>	SVM , LDA	Pada proses klasifikasi dataset dibagi menjadi 70% untuk training dan 30% untuk testing, dengan evaluasi berdasarkan F1-Score. Hasil terbaik diperoleh oleh SVM-SMOTE, dengan nilai F1-Score terbaik dari Dataset hLDA 3 Level (13 label), mencapai 95.68% dan nilai terendah dari Dataset hLDA 5 Level (8 label), mencapai 79.43%. Dataset yang memiliki lebih banyak topik memberikan hasil klasifikasi yang lebih baik. Berdasarkan jumlah like setiap topik Dataset hLDA 3 Level, yang paling diminati adalah topik 11 yang meliputi fasilitas sekolah, seragam murid, dan event hiburan. Informasi ini dapat membantu sekolah untuk mengembangkan lebih lanjut topik yang paling diminati serta meningkatkan topik yang kurang diminati.	https://jurnal.unprimdn.ac.id/index.php/JUTIKOMP/article/view/5536

No	Penulis/Sum ber	Tahun	Judul Penelitian	Mod el	Pembahasan	Sumber
10	Springer, Cham	2024	Comparison of LDA, NMF and BERTopic Topic Modeling Techniques on Amazon Product Review Dataset: A Case Study	LDA ,NMF, BERT	Dengan algoritma pemodelan topik, keluhan pengguna dapat dikelompokkan dan dibaca dalam kelompok. Dalam penelitian ini, LDA (Latent Dirichlet allocation), NMF (Non-Negative Matrix Factorization) dan algoritma BERTopic yang diuji pada kumpulan data ulasan produk Amazon dibandingkan. Menurut hasil yang diperoleh, semua 3 algoritma berhasil dan berguna. Algoritma BERTopic menghasilkan hasil yang lebih bermakna daripada algoritma lain sesuai dengan metrik perhitungan konsistensi.	https://link.springer.com/chapter/10.1007/978-3-031-53717-2_3

1.15 Pembuatan Program Model LDA

1. Interface Form model Process

NLP-DocPreprocess

- Dashboard
- Search Index
- Preprocess
- Model Process

Model Process

Module Configuration
Results

Text Modeling Configuration

Pilih Dokumen yang Akan Diproses

☒ kesehatan.txt ☒ pendidikan.txt

Number of Topics

☒ 5 topics

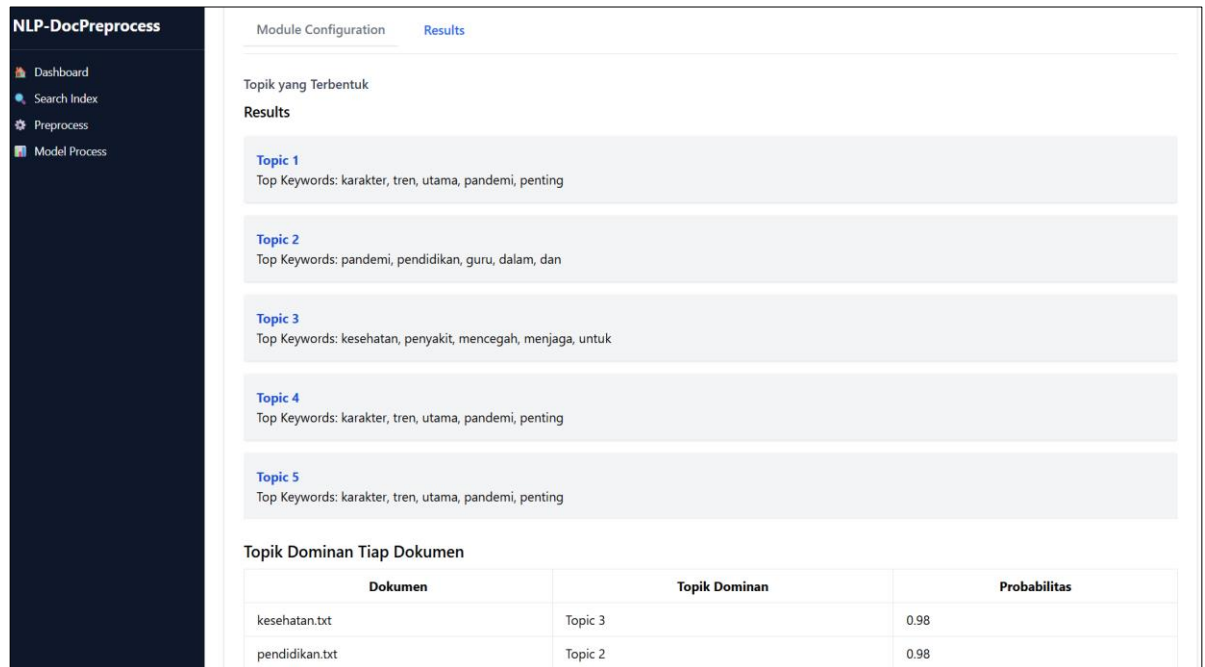
Select Model Type

LDA (Latent Dirichlet Allocation)

Run Model

© 2025 Asep Ridwan Hidayat (231012050036).

2. Interface Form hasil output proses



The screenshot displays the NLP-DocPreprocess web application. On the left is a dark sidebar with navigation links: Dashboard, Search Index, Preprocess, and Model Process. The main content area has two tabs: 'Module Configuration' and 'Results', with 'Results' being the active tab. Under the 'Results' tab, the section 'Topik yang Terbentuk' shows five topics with their top keywords:

- Topic 1**: Top Keywords: karakter, tren, utama, pandemi, penting
- Topic 2**: Top Keywords: pandemi, pendidikan, guru, dalam, dan
- Topic 3**: Top Keywords: kesehatan, penyakit, mencegah, menjaga, untuk
- Topic 4**: Top Keywords: karakter, tren, utama, pandemi, penting
- Topic 5**: Top Keywords: karakter, tren, utama, pandemi, penting

Below this, the 'Topik Dominan Tiap Dokumen' section contains a table:

Dokumen	Topik Dominan	Probabilitas
kesehatan.txt	Topic 3	0.98
pendidikan.txt	Topic 2	0.98

3. Script sama dengan program pertemuan 5 Cuma beda template layout menu model

```
{% extends "base.html" %}

{% block content %}

<!-- Header -->
<h1 class="text-2xl font-bold text-gray-800 mb-6">Model
Process</h1>

<!-- Model Configuration Section -->
<div class="bg-white rounded-lg shadow p-6 border border-gray-
200">

  <!-- Tab Buttons -->
  <div class="flex space-x-4 mb-6 border-b pb-2">
    <button
      id="moduleTab"
      class="px-4 py-2 font-medium text-blue-600 border-b-2
border-blue-600 focus:outline-none"
      onclick="switchTab('module', 'result')"
```

```

>
    Module Configuration
</button>
<button
    id="resultTab"
    class="px-4 py-2 font-medium text-gray-500 hover:text-
blue-600 focus:outline-none"
    onclick="switchTab('result', 'module')"
>
    Results
</button>
</div>

<!-- Tab Contents -->
<div class="tab-content">
    <!-- Module Configuration Content -->
    <div id="moduleContent" class="block">
        <h2 class="text-lg font-semibold text-gray-800 mb-4">Text
Modeling Configuration</h2>

        <!-- Model Selection -->
        <form method="POST" action="{{ url_for('run_model')
}}">

            <!-- File Selection -->
            <div class="mb-6">
                <h3 class="font-medium text-gray-700 mb-2">Pilih
Dokumen yang Akan Diproses</h3>
                <div class="ml-4 space-y-2">
                    {% for filename in filenames %}
                        <label class="inline-flex items-center">
                            <input type="checkbox" name="selected_files"
value="{{ filename }}" class="w-4 h-4 text-blue-600">
                            <span class="ml-2 text-gray-700">{{ filename
}}</span>
                        </label>
                    {% else %}
                        <p class="text-sm text-gray-500">Tidak ada file
di folder uploads.</p>
                    {% endfor %}
                </div>
            </div>

            <!-- Model Selection -->
            <!-- <div class="mb-6">
                <h3 class="font-medium mb-2 text-gray-700">Pilih
Model</h3>

```

```

        <select name="model_type" class="border rounded p-
2">
            <option value="lda">LDA (Latent Dirichlet
Allocation)</option>
            <option value="bertopic">BERTopic</option>
        </select>
    </div> -->

    <!-- Submit -->
    <!-- <button type="submit" class="bg-blue-500 text-white
px-4 py-2 rounded hover:bg-blue-600">
        Jalankan Model
    </button>
</form> -->

<!-- Number of Topics -->
<div class="mb-6">
    <h3 class="font-medium text-gray-700 mb-2">Number of
Topics</h3>
    <div class="ml-4">
        <label class="inline-flex items-center">
            <input type="radio" name="numTopics" class="w-4 h-4
text-blue-600" checked>
            <span class="ml-2 text-gray-700">5 topics</span>
        </label>
    </div>
</div>

<!-- Run Button -->
<div class="mt-6">
    <form method="POST" action="{{ url_for('run_model') }}">
        <div class="mb-6">
            <h3 class="font-medium mb-2">Select Model
Type</h3>
            <select name="model_type" class="border rounded
p-2">
                <option value="lda">LDA (Latent Dirichlet
Allocation)</option>
                <option value="bertopic">BERTopic</option>
            </select>
        </div>
        <button type="submit" class="bg-blue-500 text-white
px-4 py-2 rounded hover:bg-blue-600">
            Run Model
        </button>
    </form>
</div>

```

```

        </form>
        {% with messages =
get_flashed_messages(with_categories=true) %}
        {% if messages %}
            <ul class="mb-4">
                {% for category, message in messages %}
                <li class="text-sm text-{{ 'red' if category ==
'error' else 'green' }}-600">{{ message }}</li>
                {% endfor %}
            </ul>
        {% endif %}
        {% endwith %}
    </div>
</div>

<!-- Results Content
<div id="resultContent" class="hidden">
    <h2 class="text-lg font-semibold text-gray-800 mb-
4">Results</h2> -->
    <!-- Results Content -->
    <div id="resultContent" class="hidden">
        <!-- <h2 class="text-lg font-semibold text-gray-800 mb-
4">Model Results</h2> -->

        <!-- Topik yang Terbentuk -->
        <div class="mb-6">
            <h3 class="font-medium text-gray-700 mb-2">Topik
yang Terbentuk</h3>
            <div class="overflow-x-auto">
                <!-- Results Content -->
                <div id="resultContent" class="block">
                    <h2 class="text-lg font-semibold mb-
4">Results</h2>
                    {% if topics %}
                        <div class="space-y-4">
                            {% for topic in topics %}
                                <div class="bg-gray-100 p-4
rounded shadow">
                                    <h3 class="font-bold text-
blue-700">{{ topic.topic or topic.Topic }}</h3>
                                    <p>
                                        {% if topic.keywords %}
                                            Top Keywords: {{
topic.keywords | join(', ') }}
                                        {% elif topic.Name %}
                                            {{ topic.Name }}

```



```

                                {% endif %}
                            </p>
                        </div>
                    {% endfor %}
                </div>
            {% else %}
                <p class="text-gray-600">No results yet.
Please run the model.</p>
            {% endif %}
        </div>
    </div>
</div>

<!-- Distribusi Topik per Dokumen -->
{% if doc_topics %}
    <h3 class="text-xl font-semibold mt-6">Topik Dominan
Tiap Dokumen</h3>
    <table class="table-auto w-full border mt-2">
        <thead>
            <tr>
                <th class="border px-4 py-2">Dokumen</th>
                <th class="border px-4 py-2">Topik Dominan</th>
                <th class="border px-4 py-2">Probabilitas</th>
            </tr>
        </thead>
        <tbody>
            {% for doc in doc_topics %}
                <tr>
                    <td class="border px-4 py-2">{{ doc.document }}</td>
                    <td class="border px-4 py-2">{{ doc.topic }}</td>
                    <td class="border px-4 py-2">{{ doc.probability
}}</td>
                </tr>
            {% endfor %}
        </tbody>
    </table>
    {% endif %}

    <div class="mt-6">
        <h3 class="font-medium text-gray-700 mb-
2">Visualisasi Topik Dominan per Dokumen</h3>
        <canvas id="topicChart" height="100"></canvas>
    </div>

<!-- </div> -->

```

```

    </div>
</div>

<!-- Tab Switching Script -->
<script>
function switchTab(showId, hideId) {
    document.getElementById(`${showId}Content`).classList.remove('hidden');
    document.getElementById(`${hideId}Content`).classList.add('hidden');
    document.getElementById(`${showId}Tab`).classList.add('text-blue-600', 'border-blue-600');
    document.getElementById(`${hideId}Tab`).classList.remove('text-blue-600', 'border-blue-600');
    document.getElementById(`${showId}Tab`).classList.remove('text-gray-500');
    document.getElementById(`${hideId}Tab`).classList.add('text-gray-500');
}
</script>
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

<!-- <script>
    const ctx =
document.getElementById('topicChart').getContext('2d');
    const topicChart = new Chart(ctx, {
        type: 'bar',
        data: {
            labels: {{ doc_topics | map(attribute='document') | list | tojson | safe }},
            datasets: [{
                label: 'Topik Dominan per Dokumen',
                data: {{ doc_topics | map(attribute='probability') | list | tojson | safe }},
                backgroundColor: 'rgba(75, 192, 192, 0.2)',
                borderColor: 'rgba(75, 192, 192, 1)',
                borderWidth: 1
            }]
        },
        options: {
            scales: {
                y: {
                    beginAtZero: true
                }
            }
        }
    })
-->

```

```
});  
</script> -->  
  
{% endblock %}
```

Alhamdulillah terimakasih