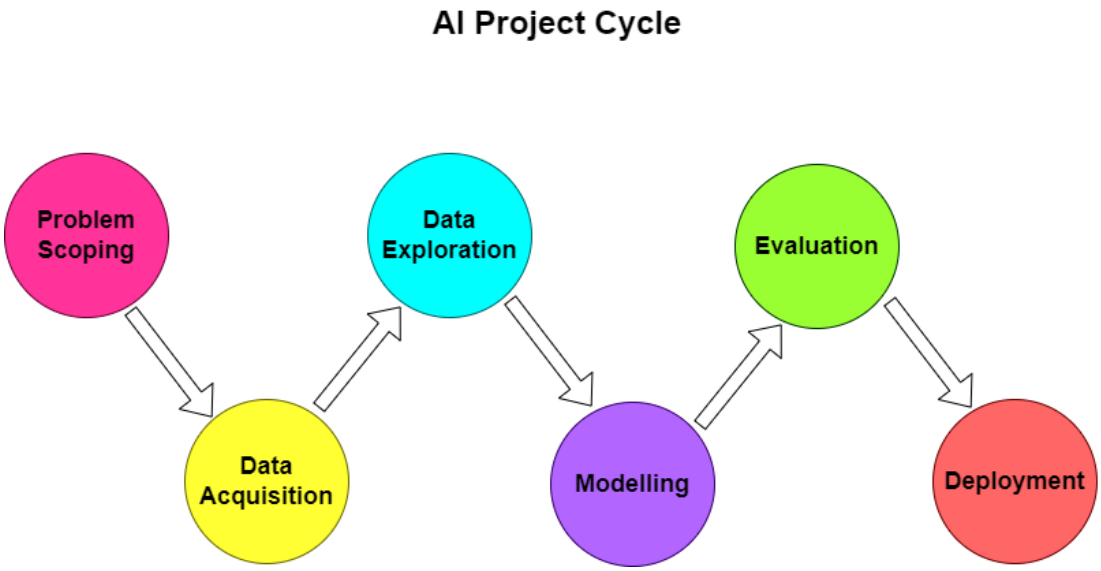
BAB III

METODE PENELITIAN

3.1. Metode dan Desain Penelitian

Metode dan desain penelitian ini menggunakan *framework AI Project Cycle.* Tahapan yang terkandung dalam *framework AI Project Cycle* terdiri dari enam tahapan, yaitu *Problem Scoping, Data Acquisition, Data Exploration, Modelling, Evaluation,* dan *Deployment* (Widodo et al., 2022). Tahapan tersebut dapat dilihat pada Gambar 3.1.



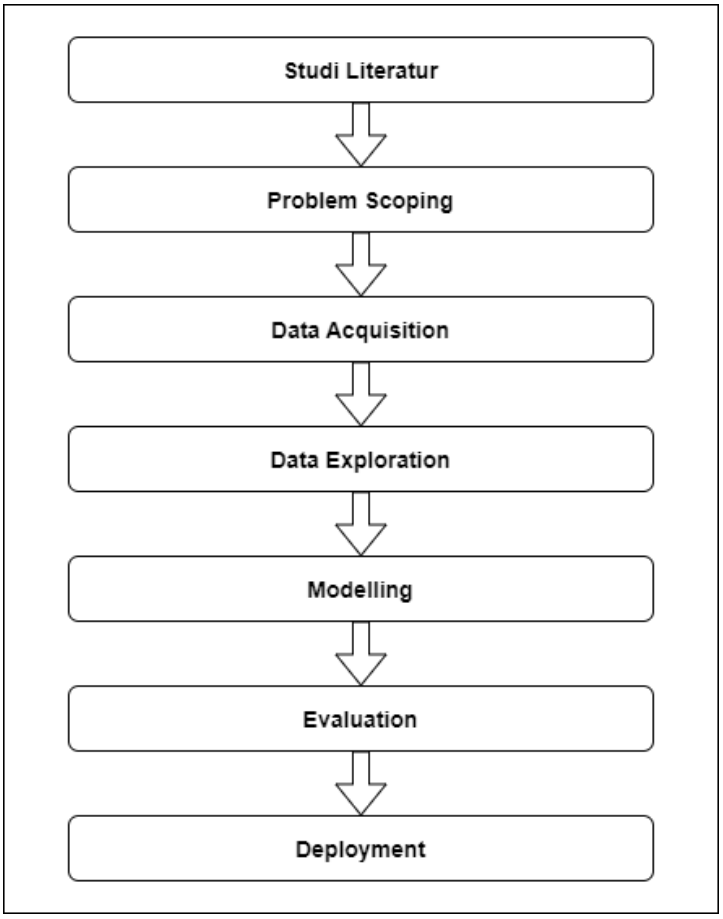
Gambar 3. 1 *Framework AI Project Cycle*

1. *Problem Scoping* merupakan tahap merinci suatu cara untuk menyelesaikan suatu masalah, dimulai dari pemahaman dan analisis masalah, hingga penentuan tujuan atau target sasaran yang akhirnya akan menghasilkan pemecahan masalah dengan baik.
2. *Data Acquisition* merupakan tahap mengukur, mengumpulkan, dan memvalidasi data yang diperlukan untuk menyelesaikan suatu masalah.
3. *Data Exploration* merupakan tahap memahami karakteristik data dan mengolah data sehingga siap digunakan.
4. *Modelling* merupakan tahap pemilihan algoritma dan pengembangan algoritma model menggunakan data pelatihan dan data uji.

5. *Evaluation* merupakan melibatkan penilaian kinerja algoritma model menggunakan matrik dan teknik evaluasi yang relevan yang telah dikembangkan selama tahap pemodelan. Pada tahap ini dapat mengukur sejauh mana algoritma model memenuhi tujuan analisis dan memberikan solusi yang efektif. Selain itu, pada tahap ini melakukan pengujian secara manual terhadap nilai akurasi model menggunakan URL baru yang belum dilihat oleh model selama proses pelatihan.
6. *Deployment* merupakan tahap implementasi model terbaik yang telah telah dibuat dan dievaluasi (Widodo et al., 2022).

3.2. Prosedur Penelitian

Penelitian ini diawali dengan studi literatur, kemudian dilanjutkan dengan tahapan pertama pada *framework AI Project Cycle* sesuai dengan desain penelitian yaitu, perencanaan dan menganalisis masalah beserta solusinya, pengumpulan data, mengembangkan algoritma prediksi *link phishing*, evaluasi algoritma pelatihan model, dan pengembangan model dalam bentuk *website*. Prosedur atau alur penelitian dapat dilihat pada Gambar 3.2. Berikut merupakan penjelasan pada setiap langkah prosedur penelitian yang dilakukan.



Gambar 3. 2 Prosedur atau Alur Penelitian

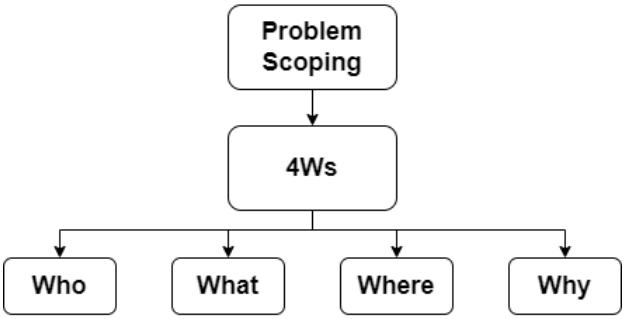
3.3.1 Studi Literatur

Pada langkah awal penelitian, peneliti melakukan studi literatur dengan mengumpulkan dan menganalisis literatur terkait topik penelitian. Studi literatur dilakukan untuk memahami konteks, teori, dan temuan sebelumnya. Studi literatur membantu peneliti dalam membangun landasan teoritis, mengidentifikasi kesenjangan pengetahuan, merumuskan masalah penelitian, dan menemukan ciri khas URL sebagai fitur URL *based* untuk melatih algoritma *logistic regression*. Selain itu, studi literatur juga memberikan wawasan mengenai metode yang telah digunakan oleh peneliti lain dalam penelitian serupa.

3.3.2 *Problem Scoping*

Pada tahap *problem scoping*, peneliti menggunakan metode 4Ws (*Who, What, Where, Why*) untuk memahami masalah dengan memetakan secara sistematis apa, siapa, di mana, dan mengapa masalah perlu diselesaikan sehingga memudahkan

menemukan solusi penyelesaian masalah dengan baik. Penggunaan metode 4Ws dalam tahap *problem scoping* diilustrasikan pada Gambar 3.3.



Gambar 3. 3 Metode 4Ws pada *Problem Scoping*

Berdasarkan Gambar 3.3 Metode 4Ws memiliki beberapa indikator. *Who* adalah indikator yang digunakan untuk menerangkan subjek yang terlibat atau mengalami masalah. *What* yaitu indikator yang digunakan untuk memetakan masalah apa yang ditemukan. *Where* yaitu indikator yang digunakan untuk memetakan kapan atau saat apa masalah ditemukan. *Why* adalah indikator yang digunakan untuk memetakan alasan mengapa masalah tersebut perlu diselesaikan dan bagaimana solusinya.

3.3.3 *Data Acquisition*

Setelah menentukan lingkup masalah, peneliti melakukan riset dan pengumpulan data yang relevan terkait masalah yang telah diidentifikasi. Pada tahap *data acquisition* mencakup pencarian dan pemilihan *dataset* yang sesuai serta memastikan validitas data yang diperoleh. Peneliti mengumpulkan data melalui *platform open data source Kaggle*, UCI *Machine Learning Repository*, The Moz untuk URL *non-phishing* di Indonesia, dan *PhishTank* untuk URL *phishing* di Indonesia. Terdapat 8 *dataset* yang akan digunakan untuk melatih algoritma model, diantaranya:

1. *Phishing Site URL* yang diperoleh dari *Kaggle*

   Phishing Site URLs Dataset berupa format .csv yang terdiri dari kolom URL dan label serta terdiri dari 549.346 baris data. Kolom label terdiri dari dua kategori yaitu *good* dan *bad* dengan perbandingan proporsi masing-masing label URL adalah sekitar 70:30.

2. *Dataset* URL yang diperoleh dari *Kaggle*

Dataset ini berupa format .csv yang terdiri dari 12 kolom serta terdiri dari 1.400 baris data. Namun, peneliti akan menggunakan 2 kolom, yaitu URL dan *Result*. Kolom *result* terdiri dari dua kategori yaitu -1 untuk label URL *non-phishing* dan 1 untuk label URL *phishing* dengan perbandingan proporsi masing-masing label URL adalah sekitar 65:35.

3. *Dataset* URL *phishing* yang diperoleh dari *Kaggle*

Dataset ini berupa format .csv yang terdiri dari 8 kolom serta terdiri dari 12.489 baris data. Namun, peneliti akan menggunakan 2 kolom, yaitu URL dan label. Kolom label hanya terdiri dari satu kategori yaitu *yes* untuk label URL *phishing.*

4. *Dataset* URL *non-phishing* di Indonesia yang diperoleh dari The Moz

Dataset ini berupa format .htm yang terdiri dari 5 kolom serta terdiri dari 500 baris data. Namun, peneliti hanya mengambil kolom URL.

5. *Dataset for Link Phishing Detection* yang diperoleh dari *Kaggle*

Dataset ini berupa format .csv yang terdiri dari 87 kolom serta terdiri dari 19.431 baris data. Namun, peneliti akan menggunakan 2 kolom, yaitu url dan *Status*. Kolom *Status* terdiri dari dua kategori yaitu *legitimate* untuk label URL *non-phishing* dan *phishing* untuk label URL *phishing* dengan perbandingan proporsi masing-masing label URL adalah 50:50.

6. *Malicious URLs Dataset* yang diperoleh dari *Kaggle*

Dataset ini berupa format .csv yang terdiri dari kolom URL dan label serta terdiri dari 651.191 baris data. Kolom label terdiri dari empat kategori, yaitu *phishing*, *benign*, *defacement*, dan *malware.* Namun, peneliti akan menggunakan URL yang memiliki label *phishing* saja dari *dataset* ini, yaitu sebanyak 94.086 data URL.

7. *Dataset* URL *phishing* di Indonesia yang diperoleh dari PhishTank

Dataset ini berupa format .csv yang terdiri dari 8 kolom serta terdiri dari 52.697 baris data. Namun, peneliti akan menggunakan 2 kolom, yaitu URL dan label. Kolom label hanya terdiri dari satu kategori yaitu *yes* untuk label URL *phishing.*

8. *Phishing* URL *Website Dataset* yang diperoleh dari UCI *Machine Learning Repository*

Dataset ini berupa format .csv yang terdiri dari 56 kolom serta terdiri dari 235.795 baris data. Namun, peneliti akan menggunakan 2 kolom, yaitu URL dan label. Kolom label terdiri dari dua kategori yaitu 0 untuk label URL *non-phishing* dan 1 untuk label URL *phishing* dengan perbandingan proporsi masing-masing label URL adalah 40:60.

3.3.4 *Data Exploration*

Setelah *dataset* terkumpul, selanjutnya adalah memahami karakteristik data dan melakukan *data preprocessing* untuk membersihkan dan memastikan kualitas data, seperti mengecek adanya duplikasi dan *missing value*, konversi *value* atau kategori label pada kolom variabel target, serta melakukan kombinasi pada beberapa *dataset* yang telah diperoleh. Proses ini bertujuan untuk memastikan bahwa data yang akan digunakan dalam pelatihan model adalah bersih, konsisten, dan representatif. Selanjutnya, pada tahap ini terdapat dua eksperimen pada tahap eksplorasi data. Eksperimen pertama yaitu memproses secara langsung kolom URL yang berupa data teks URL pada *dataset* dengan melakukan beberapa praproses teks seperti berikut.

1. Tokenisasi pada penelitian ini adalah untuk memecah URL menjadi *token* atau bagian-bagian kecil berupa kata. Peneliti menggunakan *library* pada *Python,* yaitu 'RegexpTokenizer' pada proses tokenisasi dalam memisahkan URL menjadi *token* seperti kode pada Gambar 3.4 berikut.

```
from nltk.tokenize import RegexpTokenizer
tokenizer = RegexpTokenizer(r'[A-Za-z]+')

data['text_tokenized'] = data.URL.map(lambda t:
tokenizer.tokenize(t))
```

Gambar 3. 4 *Source Code* Tokenisasi

Dengan demikian, 'RegexpTokenizer' akan memisahkan URL berdasarkan urutan huruf dalam URL dan mengabaikan atau menghapus karakter lain selain huruf.

2. *Stemming* pada penelitian ini adalah untuk mengubah bentuk kata yang mengandung imbuhan menjadi bentuk kata dasarnya. Peneliti menggunakan

library pada *Python*, yaitu 'SnowballStemmer' pada proses *stemming* dalam mengubah kata yang berimbuhan pada URL menjadi menjadi kata dasar seperti kode pada Gambar 3.5 berikut.

```
from nltk.stem import SnowballStemmer
stemmer = SnowballStemmer("english")

data['text_stemmed'] = data['text_tokenized'].map(lambda
l: [stemmer.stem(word) for word in l])

data['text_sent'] = data['text_stemmed'].map(lambda l: '
'.join(l))
```

Gambar 3. 5 *Source Code Stemming*

3. Representasi teks pada penelitian ini adalah untuk melakukan vektorisasi fitur dengan mengubah URL yang berupa data teks menjadi vektor numerik atau matriks agar dapat dimengerti dan diproses oleh algoritma *machine learning*. Peneliti menggunakan *library* pada *Python* yaitu 'CountVectorizer' pada proses representasi URL menjadi matriks seperti kode pada Gambar 3.6 berikut.

```
from sklearn.feature_extraction.text import
CountVectorizer
cv = CountVectorizer()

feature = cv.fit_transform(data.text_sent)
```

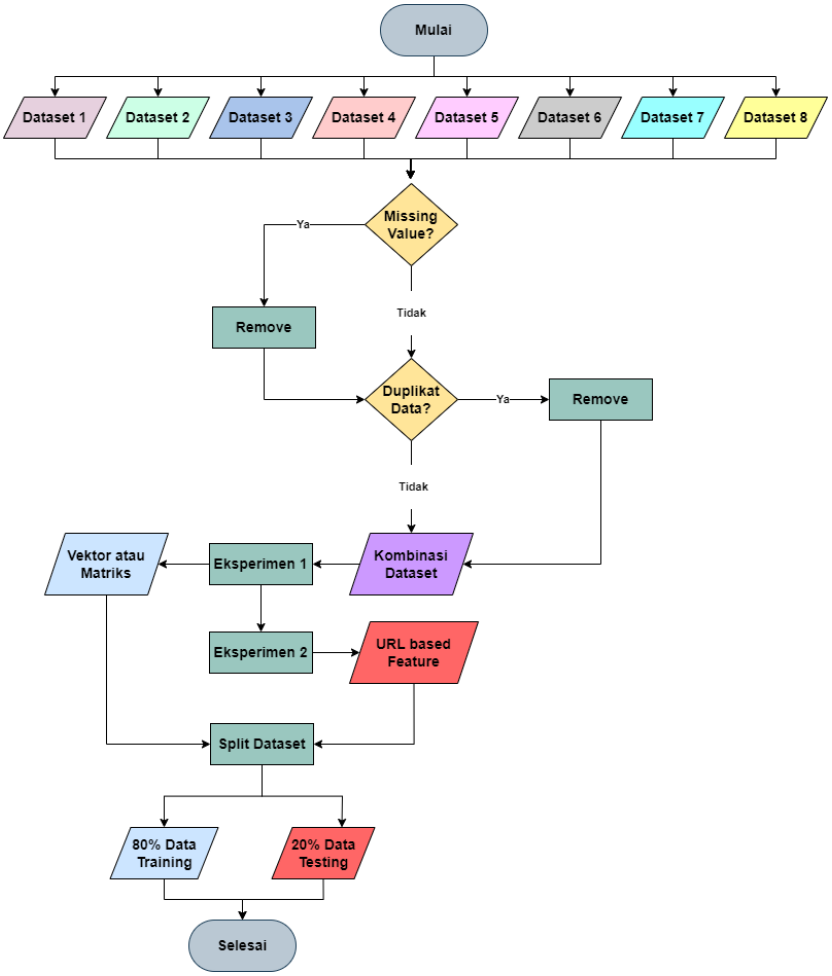
Gambar 3. 6 *Source Code* Representasi Teks

Eksperimen lain pada tahap eksplorasi data yang peneliti lakukan yaitu memproses kolom URL pada *dataset* dengan tahapan sebagai berikut.

1. *Feature extraction* pada penelitian ini adalah untuk mendapatkan ciri-ciri unik dari URL dan dari beberapa bagian struktur dasar yang ada pada URL seperti *protokol*, *hostname*, *domain*, *subdomain*, dan *path*. Ciri-ciri unik tersebut diperoleh dengan cara mengurai data URL. Peneliti menggunakan fungsi atau *rule* untuk menghasilkan fitur yang lebih banyak atau variabel-variabel prediktor dengan *value* yang telah dikonversi menjadi data numerik agar dapat digunakan oleh algoritma *machine learning* untuk memahami URL dengan baik sehingga dapat membuat prediksi. Dengan demikian, data URL dapat diwakilkan atau direpresentasikan oleh fitur-fitur yang dihasilkan pada proses *feature extraction* yang dapat digunakan lebih lanjut untuk pemodelan.

2. *Feature Selection* pada penelitian ini adalah untuk memilih fitur-fitur yang dihasilkan pada proses *feature extraction* berdasarkan hasil analisa korelasi yang dapat menghitung seberapa kuat hubungan fitur atau variabel prediktor dengan variabel target. Fitur yang memiliki hubungan yang kuat dengan variabel target adalah fitur yang memiliki nilai korelasi mendekati 1 dan -1. Peneliti menggunakan '*heatmap*' pada *library Python* dalam melakukan analisis korelasi.

Setelah melakukan representasi teks dan *feature selection* pada eksperimen satu dan dua, selanjutnya adalah melakukan pembagian pada kombinasi *dataset* menjadi 80% data *training* dan 20% data *testing.* Pembagian persentase data *training* dan data *testing* tersebut merupakan proporsi pembagian yang paling umum digunakan oleh beberapa penelitian terdahulu dalam pelatihan algoritma model *machine learning* dan terbukti menghasilkan model dengan performa yang baik, yaitu penelitian Yuda & Hendra Suputra (2021), Fatkhurohman & Pujastuti (2019), Green Arter. S, dkk (2018), P. Amba Bhavani, dkk (2022). Rangkaian proses awal hingga akhir pada data eksploration ini direpresentasikan dalam bentuk *flowchart* seperti pada Gambar 3.7 berikut.



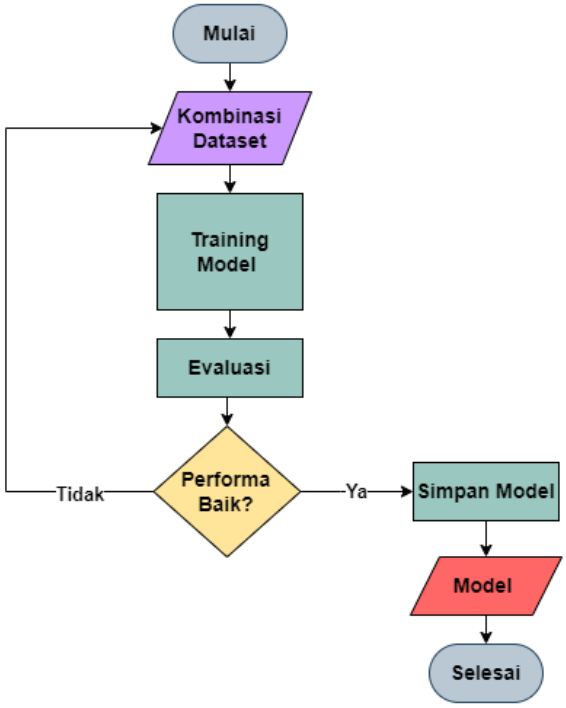
Gambar 3. 7 Proses Eksplorasi Data

3.3.5 *Modelling*

Setelah memahami karakteristik data, peneliti melakukan pengembangan algoritma model *machine learning* yang mencakup proses pemilihan dan pelatihan algoritma model. Algoritma yang dipilih untuk membuat model deteksi *link phishing* pada penelitian ini adalah algoritma *logistic regression.* Pada tahap pemodelan, peneliti melakukan beberapa eksperimen yang terdiri dari dua jenis eksperimen model. Pada jenis eksperimen model pertama, peneliti melatih model hanya menggunakan satu variabel prediktor (kolom URL). Sedangkan jenis eksperimen model lain pada tahap pelatihan model yang peneliti lakukan yaitu melatih model menggunakan lebih dari satu fitur atau variabel prediktor yang merupakan hasil dari proses *feature extraction*.

3.3.6 *Evaluation*

Setelah melakukan pelatihan model, tahap selanjutnya adalah *evaluation* model. Tahap evaluasi bertujuan untuk memahami keandalan model dalam melakukan prediksi. Peneliti melakukan evaluasi model menggunakan matriks evaluasi seperti *confusion matrix*, akurasi, presisi, recall, dan skor f1. Selain itu, peneliti menggunakan teknik evaluasi k-*fold cross validation* agar mendapatkan akurasi yang lebih akurat. Setelah mendapatkan gambaran performa model berdasarkan matriks evaluasi, kemudian dilakukan pengujian secara manual menggunakan URL baru yang belum pernah dilihat oleh model dalam proses pelatihan model. Apabila model yang dihasilkan masih kurang bagus, maka perlu memeriksa kembali tahapan sebelumnya. Sedangkan model yang memiliki performa terbaik akan disimpan dan digunakan untuk dilakukan *deployment.* Proses evaluasi yang dilakukan peneliti direpresentasikan pada Gambar 3.8 berikut.



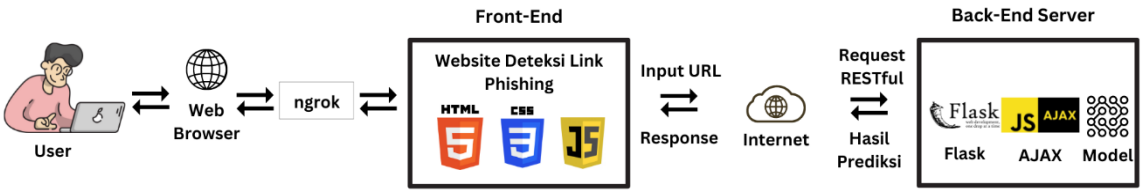
Gambar 3. 8 Proses Evaluasi

Pada Gambar 3.8, beberapa kombinasi *dataset* digunakan untuk melatih model. Pelatihan model dilakukan dengan dua jenis eksperimen yaitu menggunakan satu variabel prediktor dan menggunakan lebih dari satu fitur atau variabel prediktor. Selanjutnya, dilakukan evaluasi pada model. Evaluasi dilakukan pada kedua jenis eksperimen model. Pada evaluasi model, setelah mendapatkan gambaran performa

model berdasarkan matriks evaluasi, kemudian dilakukan pengujian menggunakan URL baru yang belum pernah dilihat oleh model dalam proses pelatihan model. Apabila performa model yang telah dievaluasi masih belum bagus maka perlu memeriksa kembali tahapan sebelumnya. Sedangkan model yang memiliki performa terbaik akan disimpan dan digunakan untuk dilakukan *deployment.*

3.3.7 *Deployment*

Tahap ini mencakup integrasi model yang telah dilatih ke dalam bentuk aplikasi *website* deteksi *link phishing*. Pada tahap *deployment,* melibatkan penggunaan *tools* pengembangan *web front-end* seperti HTML, CSS, dan *JavaScript*. Sedangkan, pengembangan *web back-end* menggunakan AJAX dan Flask *framework* menggunakan bahasa *Python*. Selanjutnya, peneliti menggunakan ngrok untuk membuat *server* lokal agar *website* dapat diakses melalui *internet* publik. Adapun arsitektur sistem dalam proses *deployment* pada penelitian ini diilustrasikan pada Gambar 3.9 berikut.



Gambar 3. 9 Arsitektur Sistem

3.4 Lingkungan Komputasi

Dalam melakukan penelitian, peneliti menggunakan alat pendukung baik *hardware* maupun *software* dalam pembuatan sistem deteksi *link phishing*. Berikut merupakan lingkungan komputasi yang digunakan peneliti selama penelitian.

1. *Hardware*

Selama penelitian, *hardware* pendukung yang digunakan adalah *laptop* memiliki spesifikasi sebagai berikut.

a) *Windows* 11 *Pro* 64-bit
b) CPU *Intel Core* i.5−1.0 GHz
c) RAM 4GB
d) SSD 239GB

2. *Software*

Selama penelitian, *software* pendukung yang digunakan sebagai berikut.

a) Bahasa pemrograman *Python* Versi 3.10.9
b) Anaconda3 (64-bit)
c) *Jupyter Notebook* Versi 6.5.2
d) *Google Colaboratory*
e) *Visual Studio Code*
f) Ngrok

4.5 Teknik Pengumpulan Data

Data yang digunakan pada penelitian ini merupakan data sekunder, yaitu dengan mencari dan mengunduh *dataset* yang tersedia pada beberapa *platform*. *Dataset* URL yang akan digunakan untuk melatih algoritma model prediksi *link phishing* yang diambil dari *public data source Kaggle*, UCI *Machine Learning Repository*, *The Moz Top* 500 untuk URL *non-phishing* di Indonesia, dan *PhishTank* untuk URL *phishing* di Indonesia.

4.6 Teknik Analisis Data

Penelitian ini menganalisis atau mengevaluasi performa model *supervised learning*, yaitu algoritma *logistic regression.* Setelah model dilatih dan diuji, model akan dievaluasi menggunakan *confusion matrix*, akurasi, presisi, *recall*, dan skor f1 untuk mengetahui efektivitas dan kualitas kinerja sebuah model dalam mengklasifikasikan data yang baru. Evaluasi model ini dilakukan menggunakan *Python* untuk mengetahui *performance* model dari algoritma model *logistic regression*, yaitu berupa skor *accuracy*, *recall*, *precision,* dan F1-*score*. Metode *confusion matrix* merepresentasikan hasil evaluasi model dengan menggunakan tabel matriks, jika *dataset* terdiri dari dua kelas, kelas pertama dianggap positif dan kelas kedua dianggap negatif (Salmu & Solichin, 2017). Tabel *confusion matrix* direpresentasikan pada Tabel 3.1.

Tabel 3. 1 *Confusion Matrix*

Kelas Sebenarnya	**Kelas Prediksi**	
	Positif (+)	**Negatif (-)**
Positif (+)	*True Positive* (TP)	*False Negative* (FN)
Negatif (-)	*False Positive* (FP)	*True Negative* (TN)
Jumlah	N	P

True Positives (TP) merupakan kelas yang diprediksi positif dan sesuai dengan kelas sebenarnya (positif). *False Positives* (FP) merupakan kelas yang diprediksi positif, namun tidak sesuai dengan kelas sebenarnya (negatif). *True Negative* (FN) merupakan kelas yang diprediksi negatif dan sesuai dengan kelas sebenarnya (negatif). *False Negative* (FN) merupakan kelas yang diprediksi negatif namun tidak sesuai dengan kelas sebenarnya (positif). Perhitungan *accuracy*, *recall*, *precision,* dan F1-*score* dinyatakan dalam persamaan [6], [7], [8], dan [9] seperti berikut (Assaidi & Amin, 2022):

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{P+N}} \quad \textbf{[6]}$$

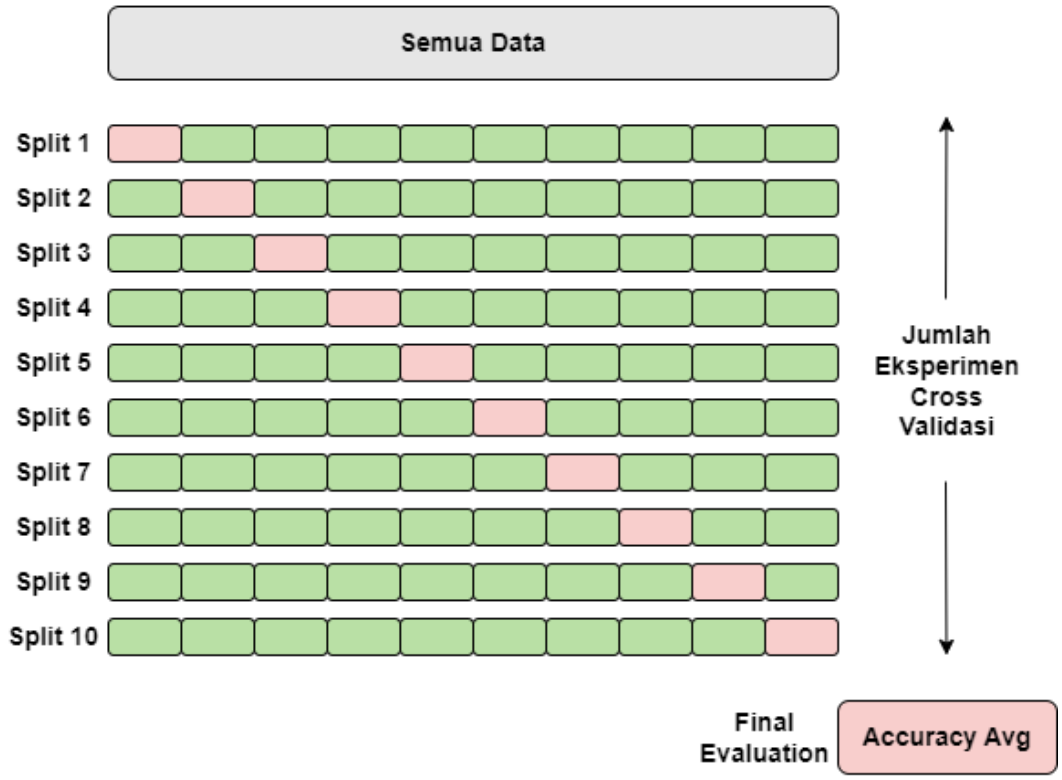
$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}} \quad \textbf{[7]}$$

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \quad \textbf{[8]}$$

$$\textbf{F1-}\boldsymbol{score} = \mathbf{2} * \frac{\boldsymbol{Precision} * \boldsymbol{Recall}}{\boldsymbol{Precision+Recall}} \quad \textbf{[9]}$$

Akurasi dihitung dengan membagi jumlah prediksi yang benar dengan total jumlah prediksi. Presisi mengacu pada rasio antara *true positive* (prediksi positif yang benar) dengan total prediksi positif. *Recall* mengacu pada rasio antara *true positive* dengan total *instance* yang seharusnya positif. F1 *score* adalah matrik gabungan yang menggabungkan presisi dan *recall* (Sandag et al., 2018).

Selain menggunakan *confusion matrix,* peneliti menggunakan teknik evaluasi k-*fold cross validation* dengan nilai k adalah 10 untuk melakukan validasi tingkat keakuratan suatu algoritma model klasifikasi sehingga mendapatkan nilai akurasi yang lebih akurat seperti yang telah dilakukan oleh penelitian Green Arter. S, dkk (2018) dan Agus Fatkhurohman dan Eli Pujastuti (2019). Evaluasi 10-*fold cross validation* dilakukan untuk melakukan uji validitas pada akurasi yang didapatkan. Teknik evaluasi 10-*fold cross validation* pada penelitian ini diilustrasikan pada Gambar 3.10 berikut.



Gambar 3. 10 10-*Fold Cross Validation*

Pada Gambar 3.10, *dataset* dibagi menjadi 10-*fold* untuk digunakan pada pelatihan model. *Cross* validasi dilakukan sebanyak nilai k yaitu 10 yang mana pada setiap iterasinya menggunakan salah satu *fold* sebagai data uji dan *folds* lainnya sebagai data latih secara bergantian. Pelatihan model pada setiap iterasi akan dihitung akurasinya berdasarkan hasil prediksi pada data *testing.* Selanjutnya, performa model dari masing-masing iterasi akan dihitung rata-ratanya untuk menentukan performa model secara keseluruhan.