National College of Ireland

Configuration Manual

MSc Research Project
Master of Science In Cybersecurity

Vatsala Narayan
Student ID: 23201126

School of Computing
National College of Ireland

Supervisor: Liam Mccabe

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Vatsala Narayan

Student ID: 23201126

Programme: MSc. In Cybersecurity **Year:** 2024-25

Module: MSc. Research Project

Lecturer: Liam Mccabe

Submission Due Date: 11/08/2025

Project Title: AI-Powered Phishing Detection: Overcoming Gaussian RBF Kernel Limitations

Word Count: 355 **Page Count:** 6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Vatsala Narayan

Date: 11/08/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Vatsala Narayan
x23201126@student.ncirl.ie

Project Title: AI-Powered Phishing Detection: Overcoming Gaussian RBF Kernel Limitations

Table of Content

1. Introduction

This configuration manual provides step-by-step instructions for setting up and executing the proof-of-concept encrypted communication system based on AES encryption with dynamic rolling keys. The system is implemented entirely in Python and executed in a WSL (Ubuntu) environment, making it platform-independent and resource-light. It demonstrates how rolling key strategies can prevent man-in-the-middle (MitM) attacks on public networks by frequently rotating the encryption key derived from a shared secret.

2. System Requirements

Component	Minimum Requirement
Host OS	Windows 11
Python Version	Python 3.10
Hardware	Intel(R) Core (TM) Ultra 7 155H (3.80 GHz)
Packages Management	virtualenv
Category	**Tool / Version Used**
Programming Language	Python 3.10
Core Libraries	pandas=2.0.3, numpy<=1.24.3

Deep Learning	tensorflow=2.13.0, keras (bundled with TensorFlow)
Machine Learning	scikit-learn=1.3.2, joblib=1.3.2
Feature Extraction	TF-IDF (Scikit-learn), Tokenizer (Keras)
Data Handling & Preprocessing	pandas, numpy, re (Regular Expressions), nltk=3.8.1
Visualization	matplotlib=3.7.3, seaborn=0.12.2
Utilities	tqdm=4.66.1
Deployment	streamlit=1.27.2 (UI)

3. Installation Steps

Step 1: Setup of the Python Virtual Environment

Create a virtual environment: python -m venv phishing-env

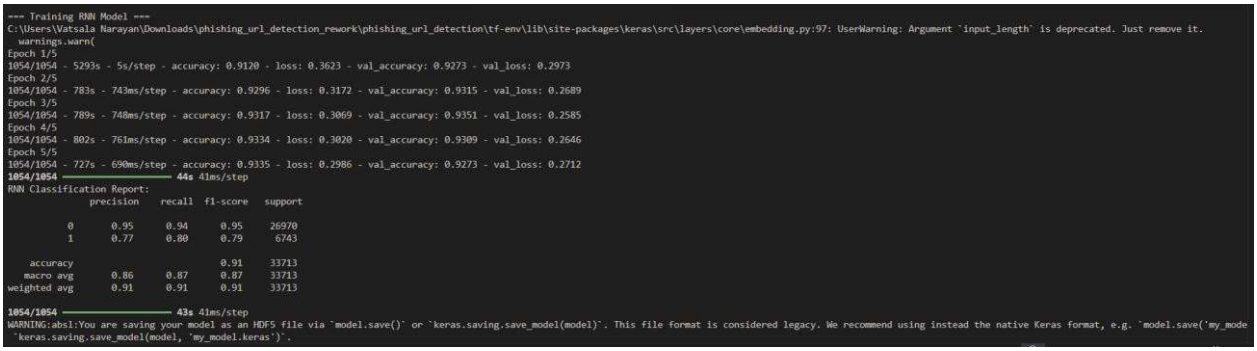
Activate the environment: phishing-env\Scripts\activate

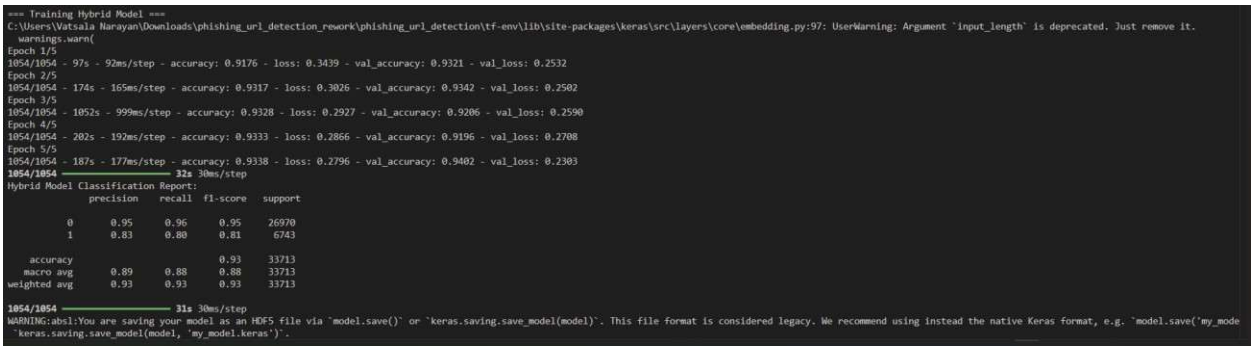
Step 2: Install Required Dependencies
Install all required libraries from the requirements.txt file: pip install –r requirements.txt

Step 3: Train all the Models
The file main.py is for **model (SVM, CNN, RNN, Hybrid) training and evaluation** using command **python main.py**

```
(tf-env) PS C:\Users\Vatsala Narayan\Downloads\phishing_url_detection_rework\phishing_url_detection> python main.py
2025-07-28 17:41:19.552838: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results
due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`
.
2025-07-28 17:41:22.397881: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results
due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`
.

=== Training SVM Model ===
SVM Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.99      0.95     26970
           1       0.95      0.59      0.73      6743

    accuracy                           0.91     33713
   macro avg       0.93      0.79      0.84     33713
weighted avg       0.92      0.91      0.90     33713

SVM Accuracy: 0.9119330821938125
```

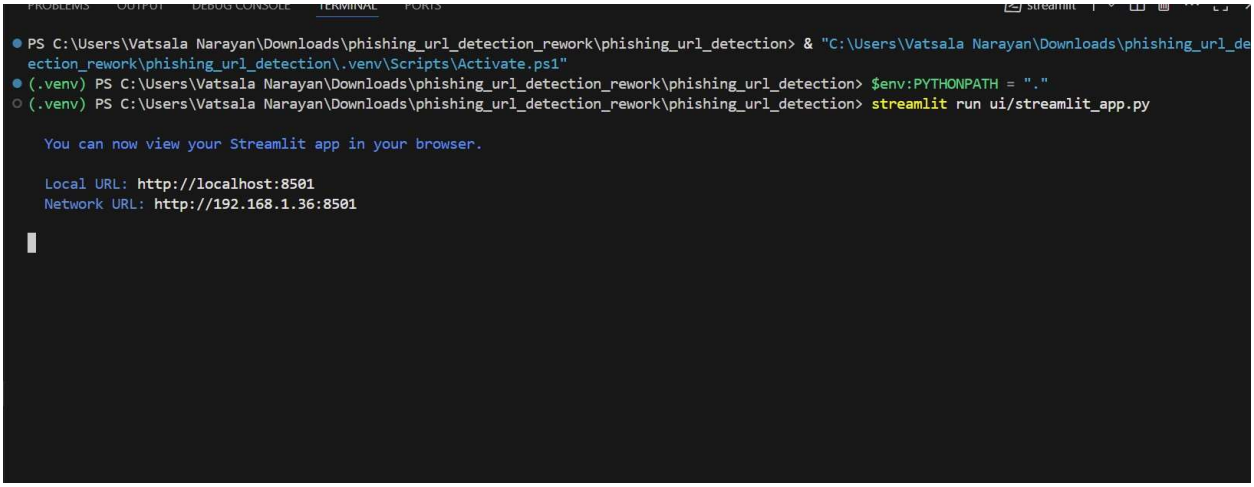
```
=== Training CNN Model ===
C:\Users\Vatsala Narayan\Downloads\phishing_url_detection_rework\phishing_url_detection\tf-env\lib\site-packages\keras\src\layers\core\embedding.py:97: UserWarning: Argument `input_length` is deprecated. Just remove it.
  warnings.warn(
2025-07-28 17:41:59.348243: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
Epoch 1/5
1054/1054 - 68s - 64ms/step - accuracy: 0.9064 - loss: 0.3415 - val_accuracy: 0.9274 - val_loss: 0.2775
Epoch 2/5
1054/1054 - 72s - 68ms/step - accuracy: 0.9276 - loss: 0.3064 - val_accuracy: 0.9288 - val_loss: 0.2632
Epoch 3/5
1054/1054 - 71s - 67ms/step - accuracy: 0.9290 - loss: 0.2956 - val_accuracy: 0.9327 - val_loss: 0.2536
Epoch 4/5
1054/1054 - 72s - 69ms/step - accuracy: 0.9302 - loss: 0.2881 - val_accuracy: 0.9373 - val_loss: 0.2305
Epoch 5/5
1054/1054 - 72s - 68ms/step - accuracy: 0.9310 - loss: 0.2831 - val_accuracy: 0.9323 - val_loss: 0.2486
1054/1054 ━━━━━━━━━━━━━━━━━━━━ 7s 7ms/step
CNN Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.94      0.95     26970
           1       0.78      0.81      0.79      6743

    accuracy                           0.92     33713
   macro avg       0.86      0.87      0.87     33713
weighted avg       0.92      0.92      0.92     33713

1054/1054 ━━━━━━━━━━━━━━━━━━━━ 8s 7ms/step
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('
```



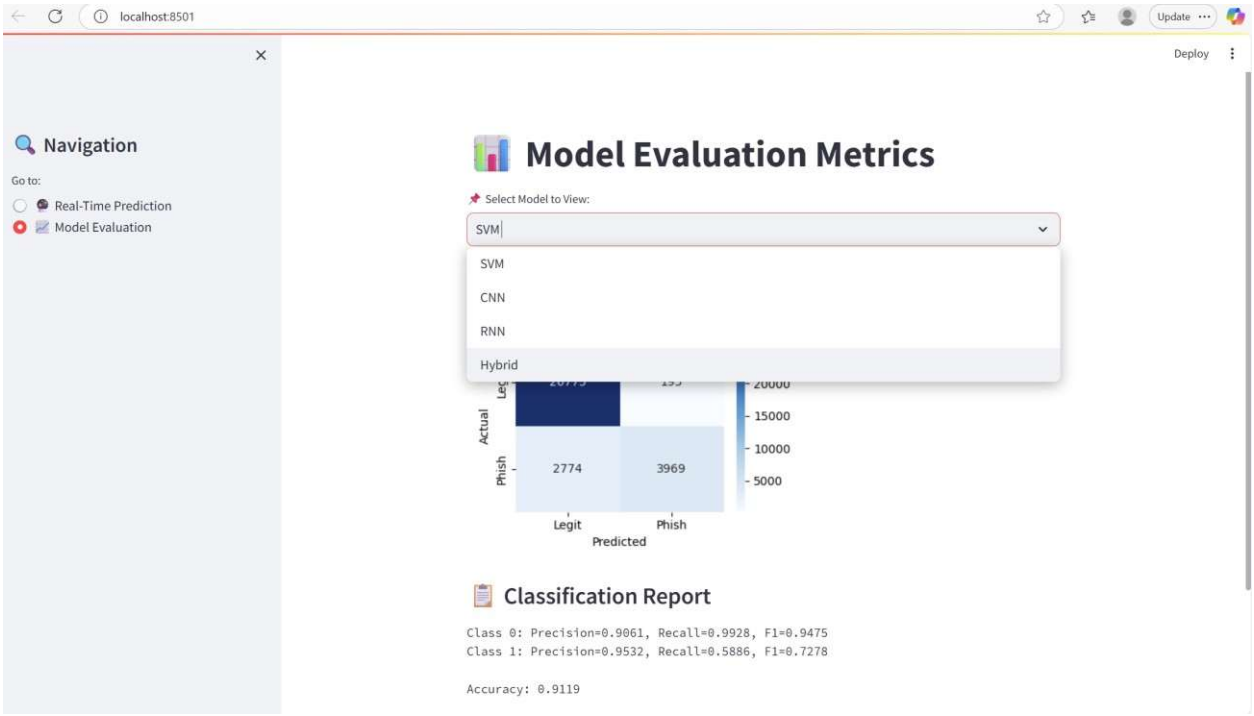


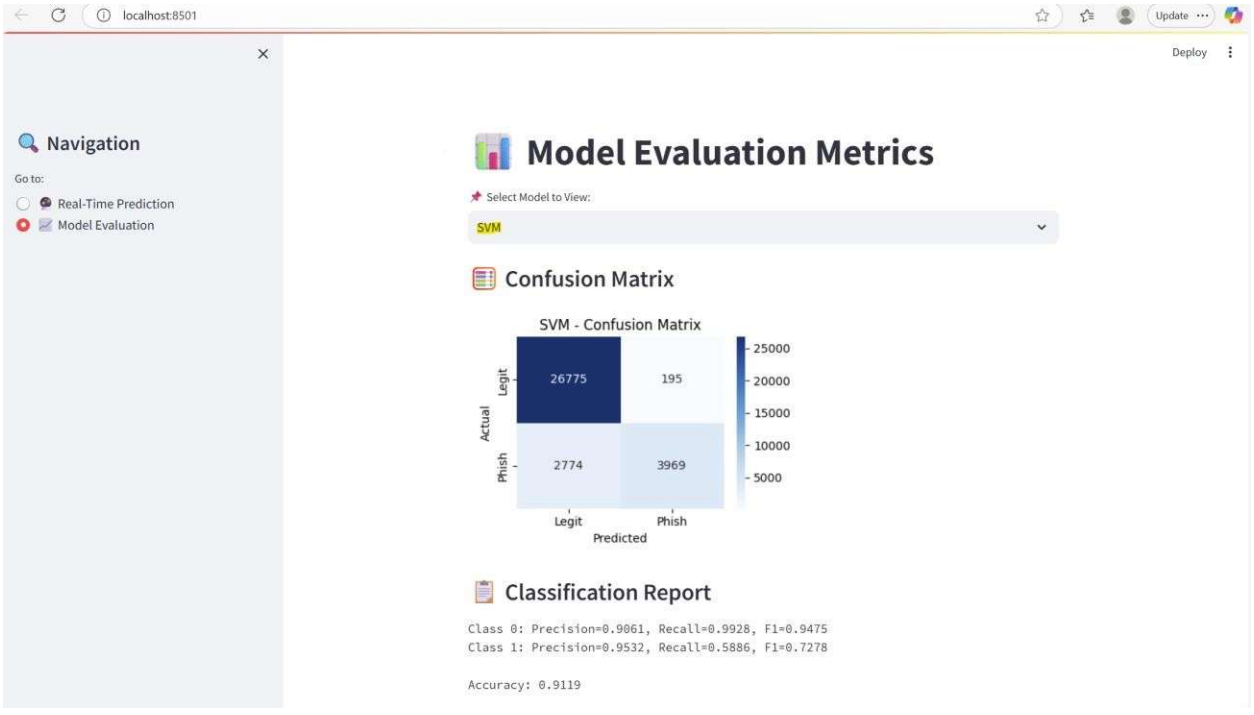
Step 4: Set the environment path using the command **$env:PYTHONPATH = "."**

Step 5: Launch the Real-Time Detection App **streamlit run ui/streamlit_app.py**

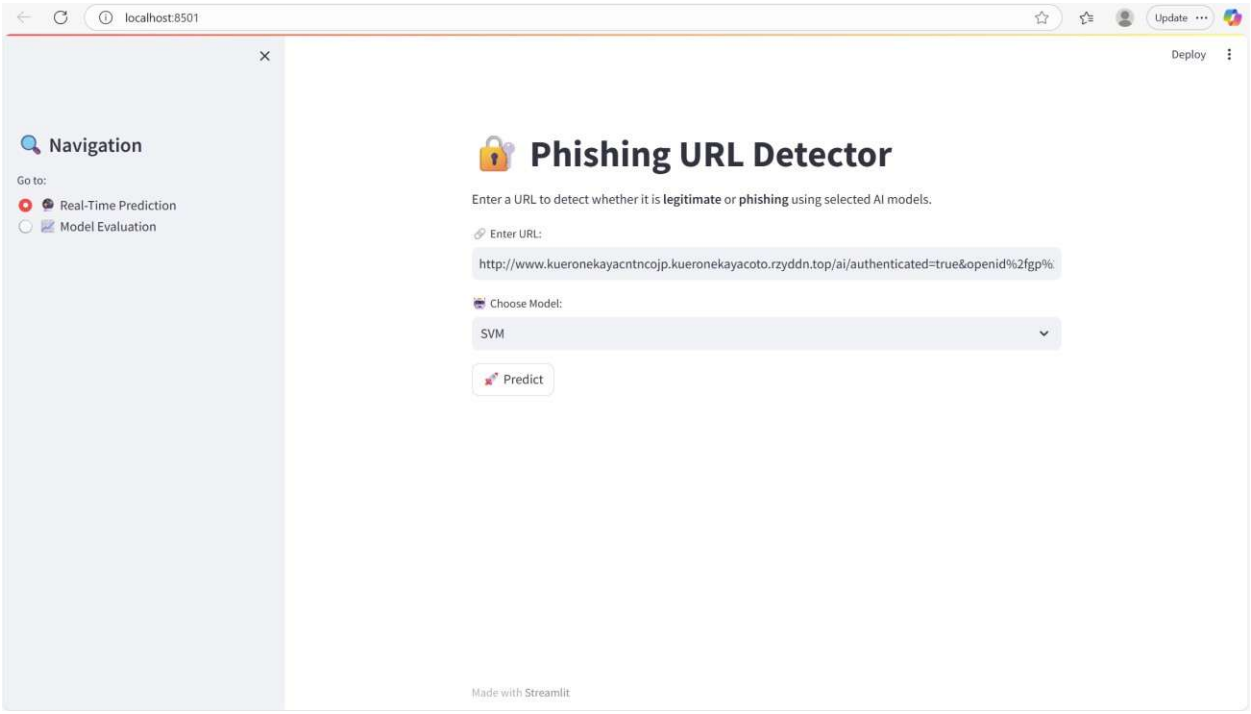


Step 6: Go to the Model Evaluation module on UI.
- Select the Model name to view the respective confusion matrix and other metric like **Accurcy, recall, precison and F1 scores.** For example, in the screenshot below the SVM Model is selected.

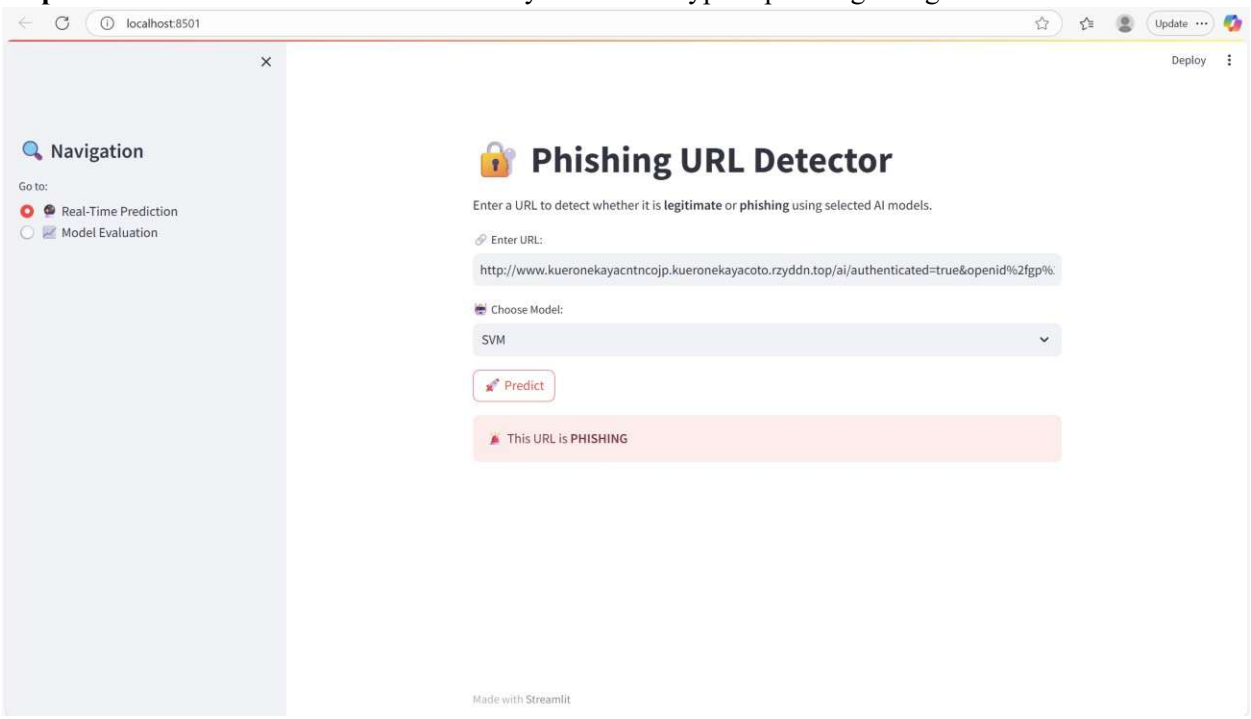

localhost:8501
Update
Deploy
Navigation
Go to:
Real-Time Prediction
Model Evaluation
Model Evaluation Metrics
Select Model to View:
SVM
SVM
CNN
RNN
Hybrid
Actual
Phish
2774
3969
Legit
Phish
Predicted
15000
10000
5000
Classification Report
Class 0: Precision=0.9061, Recall=0.9928, F1=0.9475
Class 1: Precision=0.9532, Recall=0.5886, F1=0.7278
Accuracy: 0.9119


localhost:8501
Update
Deploy
Navigation
Go to:
Real-Time Prediction
Model Evaluation
Model Evaluation Metrics
Select Model to View:
SVM
Confusion Matrix
SVM - Confusion Matrix
Actual
Legit
Phish
26775
195
2774
3969
Legit
Phish
Predicted
25000
20000
15000
10000
5000
Classification Report
Class 0: Precision=0.9061, Recall=0.9928, F1=0.9475
Class 1: Precision=0.9532, Recall=0.5886, F1=0.7278
Accuracy: 0.9119

Step 6: Go to the module Real-Time Prediction on UI and enter the URL



Step 7: Click on the Predict button to identify if the URL type is phishing or legitimate.



Sample Test Data's Used are:

- **Unknown Phishing URL** - https://tweurp.kcrjdrxxs.es/KVdw9@SuEPo2g
- **Unknown Legitimate URL** - https://www.google.com/
- **Known Phishing URL -** http://www.kueronekayacntncojp.kueronekayacoto.rzyddn.top/ai/authenticated=true&openid%2fgp%2fsignin%2fx%26i%3da%26oauth%3dm%26i%3fie%3dutf8%26ref_%3drhf_custrec_signin127763c3de8b2310c4b3bb96ddcd6822ac65e2ab
- **Known Legitimate URL** - https://www.stardog.com

4. Project File Structure

Place all these files into a directory named Phishing_URL_Detection.

Files	Description
phishing_url_dataset.csv	Datasets contain phishing and legitimate URLs with labels for the model training and evaluation.
data_analysis.py	Script to explore the dataset.
cnn_model.py	Script to run Convolutional Neural Network training models' script.
hybrid_model.py	Script to run hybrid training models' script.
rnn_model.py	Implements a Recurrent Neural Networks model for phishing detection.
svm_model.py	Training of SVM (Support Vector Machine) model for phishing detection.
clean_data.py	Preprocessing script for noise removal from the dataset before its feature extraction.
feature_extraction.py	Extracts feature from URLs (length, presence of special chars, TF-IDF of tokens).
cnn_model.keras	Saved CNN model file in keras format after training.
hybrid_model.keras	Saved Hybrid model file in keras format.
rnn_model.keras	Saved Neural Network model file in keras format.
svm_model.pkl	Pickled trained SVM model is saved int this format.
tfidf_vectorizer.pkl	Vectorizer to extract features from the text-based URL.
streamlit_app.py	Streamlit web application script for real-time phishing URL detection.
config.py	Contains the configuration variables of project.
helper.py	Contains the functions to load models, preprocess URLs
main.py	The main file is to execute the code for training and evaluation.
README.md	The execution steps are written
requirements.txt	Python packages to setup the environment