Project Development Report By group 20

Introduction:

This report details the development of the flask-based web application, called Reeler, that allows users to input an email they receive and have it evaluated by a LLM for its likeliness to be a phishing email. The LLM uses advanced machine learning models to process the user inputted emails and output an evaluation. The project integrates a phishing detection LLM model based of the DisilBERT architecture. This allows our users to receive an intelligent evaluation quickly and seamlessly which will help enhance the user experience and encourage them to come back.

Project requirements and objectives:

Requirements:

User Interface: It must be easy to understand and use so that users don’t have trouble interacting with the UI.

Email Evaluation: It must provide an accurate and prompt evaluation of the inputted email in order to provide an efficient and effective service in order to enhance the user experience.

Log-in Service: It must provide the user with the ability to make an account and log-in. This is done so that the user can feel a sense of belonging with the application and thus an increase in loyalty.

Session Management: It must ensure that only authorized users can access protected resources through the use of encryption technologies like bcrypt and that it effectively maintains the user state.

Security: We must ensure that the database that contains the user’s login in detail remains as secure as possible and that the contents must be encrypted using the bcrypt technology.

Objectives:

Create an application that allows for text inputs of an email and then utilises pre-trained machine learning models to provide evaluation as to the likelihood of it being a phishing email. Provide accurate and timely evaluations to enhance user experience. Build UI that is easy to understand and use whilst being responsive.

Technical specifications:

Front-end:

Technologies: HTML5, CSS3

Framework: Bootstrap 4.5 (for a fast and responsive design)

Back-end:

Programming Languages: Python 3, SQLite3

Framework: Flask (web framework known by the whole team)

Session Management: Flask-Session

Machine Learning Models:

Email Evaluation Model: phishing-email-detection-distilbert\_v2.4.1

Libraries and Tools:

Pytorch: To integrate deep learning models

Werkzeug: To create a secure WSGI

Bcrypt: For secure encryptions

Hashlib: For secure hashing

Ngrok: To put the application into a website

Project Plan:

Project Manager: Paulius

Front-end Developer: Samuel

Back-end Developer: Emily

Machine Learning Engineer: Henry

Quality Assurance Engineer: Paulius

Timeline and Milestones:

A colorful squares on a white surface

Description automatically generated

Wireframe:

A screenshot of a computer error message

Description automatically generated

A screenshot of a computer screen

Description automatically generated

Product Explanation:

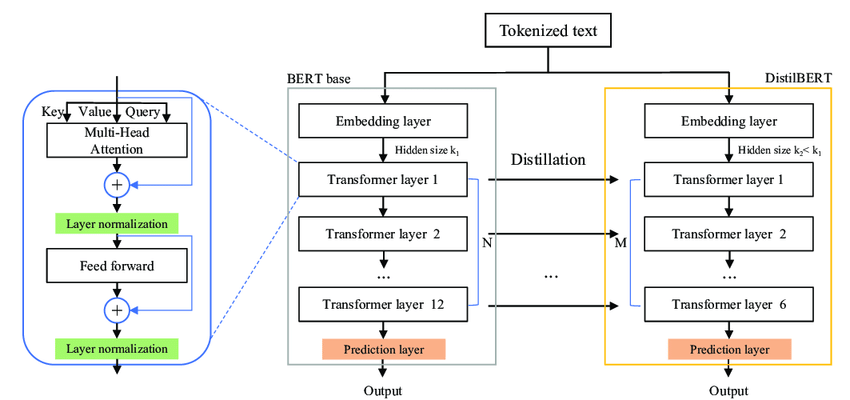
Features:

Likelihood of Phishing Evaluation Tool: This is a tool that allow for users to input an email that they have received that they would like to verify the legitimacy of. This tool uses an LLM to ensure that the evaluation is accurate.

Account Management System: This is a feature that allows user to create an account and log into it. This is done in order to give the user a sense of belonging and thus an increased sense of loyalty.

Responsive and easy to understand interface: This allows for the webpage to provide a smooth and seamless feel to it when traversing it. And the use of a professional font without serifs such as century gothic allows for it to be easy to read and understand.

Architecture Overview:



(Mabrouk, 2022)

This above diagram shows the DistilBERT architecture which we are using compared with its teacher model BERT. They both take a tokenized text and output a predictive result. However, what separates DistilBERT from BERT is that it only uses 6 layers whereas the teacher model uses 12. Whilst this would make it less accurate than BERT, it would make it far more lightweight and faster. This is why we chose to go with DistilBERT since we believed it would fit our needs better. Our version of DistilBERT used databases filled with phishing emails in order to train itself to recognise key signs of phishing and use that to form an evaluation.

Technologies used:

Pytorch: In order to integrate the LLM into the application smoothly and efficiently.

Phishing-email-detection-distilbert\_v2.4.1: The LLM that will produce the evaluations for our Likelihood of Phishing Evaluation Tool in a manner that is quick and high quality.

Flask: Because it’s a lightweight web framework that the whole team is familiar with so theres no need to learn a new framework.

Flask-Session: An extension onto the Flask technology that allows for us to provide good quality session management that is already integrated with our web framework.

Werkzeug: A python library for web development that allows us to easily set up a WSGI and provide secure file handling to ensure the users data safety.

Bootstrap 4.5: It means that our web page is very responsive and seamless to navigate.

Testing:

Tests Conducted:

Likelihood of Phishing Evaluation Tool:  
Verified that the evaluation tool handled invalid, incorrectly formatted or corrupted files properly.

Verified that we received the evaluation result in an acceptable time frame.

Verified that the LLM’s evaluation method was accurate.

Log In/Account Creation Tests:

Verified that it wouldn’t let me set my account details to anything inappropriate.

Verified that it wouldn’t let me access someone else’s account through the use of bugs.

Security Test:

Made sure that the SUUID was properly randomized and encrypted.

Tested for vulnerabilities in file handling.

Verified that sessions are managed securely.

Cross-Browser Compatibility:

Tested the application on major browsers (Edge, Firefox, Chrome, Safari).

Results:

|  |  |  |
| --- | --- | --- |
| Functionality | Performance | Usability |
| Likelihood of Phishing Evaluation Tool | The evaluation occurred within acceptable time frames and to a good degree of accuracy. | Users found the tool helpful and seamless. |
| Account Management System | Account creation as well as logging in work exactly as expected and there were no exploitable bugs to allow you to access another user’s account illicitly. | Users found the system easy to use and navigate. |
| User Database | Was tested for vulnerabilities in file handling as well as user input processing and the strength of their encryption. Everything held up well except for the encryption. This problem was rectified by introducing hashing. | Users appreciated the fact that their data was kept securely. |
| User Interface | Was tested for the speed of swapping pages as well as for any bugs that may cause an error with the UI. No such errors occurred, and the speed of page swapping was found to be acceptable. | Users found the interface easy to use and understand and also found it responsive. |

Conclusion:

The development of the Reeler web application successfully achieved the projects objectives by providing a sleek and responsive UI that allows for user input of an email and outputs an accurate and timely evaluation. The integration of the phishing-email-detection-distilbert\_v2.4.1 LLM allowed for this evaluation to be so accurate whilst still running on such a lightweight LLM. This helped us keep the UI nice and responsive as well as being easy to understand thanks to the use of serifless fonts and a standardised UI layout. This was all possible thanks to rigorous testing and a well-organized development plan, The application is robust, user-friendly, efficient, and scalable.

References:

* Mabrouk, A. (2022) *The Distilbert model architecture and components. | download scientific diagram*, *ResearchGate*. Available at: https://www.researchgate.net/figure/The-DistilBERT-model-architecture-and-components\_fig2\_358239462 (Accessed: 05 December 2024).