Institiúid Teicneolaíochta Cheatharlach

INSTITUTE of TECHNOLOGY CARLOW

At the Heart of South Leinster

Email Spam Filter using Machine Learning

Final Report

Author: Hazel Murphy

Student ID: C00230058

Project Supervisor: James Egan

Date: Friday 30th April 2021

Abstract

The project purpose is to provide users with a secure email spam filter tool using machine learning models. Four models were implemented in this project. The project has two main components, the backend mail server which uses a scrape function to store all the data into the SQL database and then the flask application which is the front-end interface. This is the interface each user will see when viewing their emails.

Table of Contents

Introduction

The final report purpose is to describe in detail the end-product produced of the email spam filter tool using machine learning algorithms. This document has various sections about the project.

Section 1: Describes the final product in detail. It covers the tools and technologies used and how they were integrated together. I have also included a diagram of the system architecture to allow readers to visualize the project.

Section 2: This section displays each screen of the web application with a short description.

Section 3: The document contains learning outcomes from the project and project review.

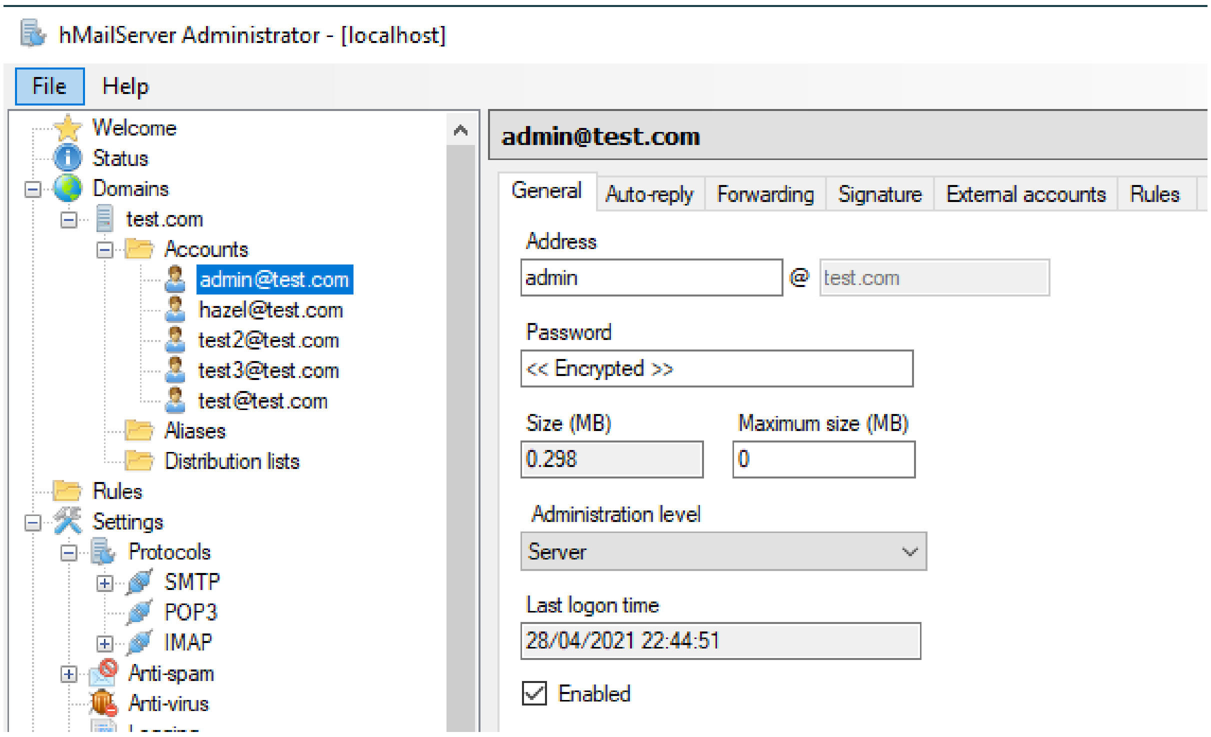
Section 4: This section covers in detail the testing which took place for my project. Testing on both and back-end and front-end was performed.

Section 5: Finally, I have included the future features for my project going forward.

Project description

The email spam filter was implemented as follows:

I installed hMailServer on my local system for the purpose of creating a local mail server. The “@test.com” domain was created in the hMailServer along with test accounts to send and receive mails between accounts.



hMailServer Administrator - [localhost]

admin@test.com

Address: admin @ test.com

Password: << Encrypted >>

Size (MB): 0.298

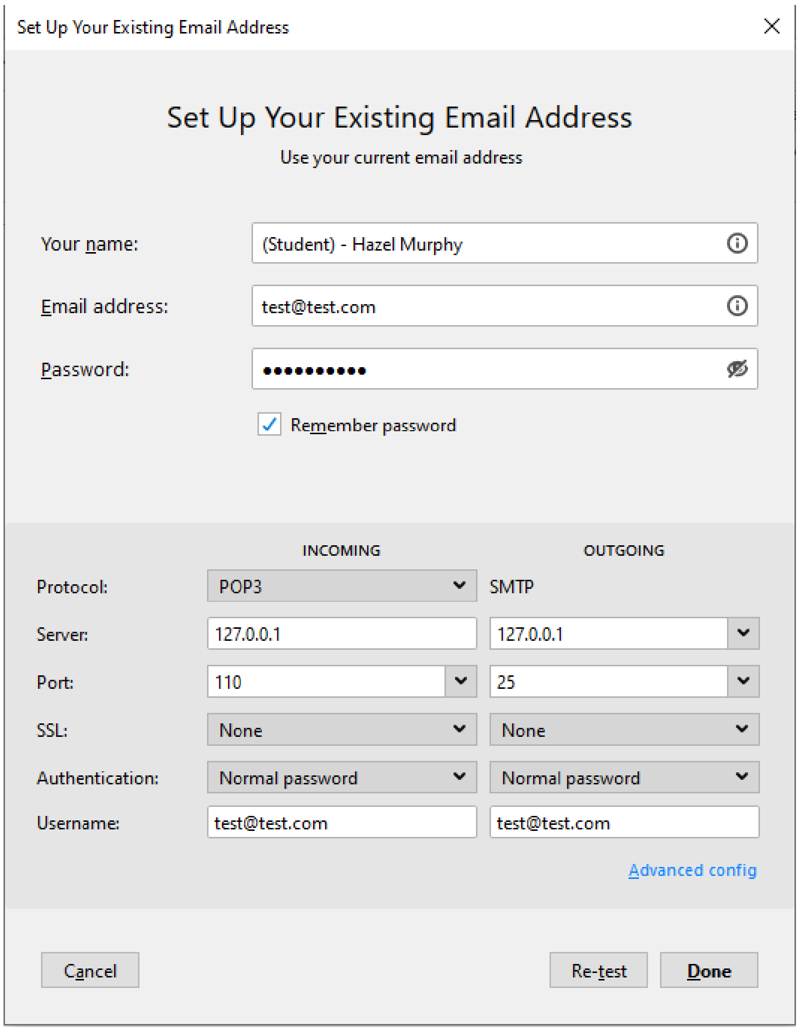
Maximum size (MB): 0

Administration level: Server

Last logon time: 28/04/2021 22:44:51

☑ Enabled

Thunderbird is an email client software application. I used this to send mails between accounts I set up on my local server. I logged into each account on the local server using the following configuration:



Set Up Your Existing Email Address

Set Up Your Existing Email Address

Use your current email address

Your name:	(Student) - Hazel Murphy
Email address:	test@test.com
Password:	••••••••••

✓ Remember password

	INCOMING	OUTGOING
Protocol:	POP3	SMTP
Server:	127.0.0.1	127.0.0.1
Port:	110	25
SSL:	None	None
Authentication:	Normal password	Normal password
Username:	test@test.com	test@test.com

Emails sent via thunderbird are stored in the hMailServer directory on the PC. For these emails to be stored in the SQL database I had to implement a scrape function. This function scans through the directories of the local mail server, and for each email which has not already been stored it transforms the emails data for storage and then calls upon each of the model's functions created and inserts the spam classification boolean data for each model into the database. The four models, Naïve Bayes, SVM, Random Forest and Logistic Regression were trained using the following dataset: https://www.kaggle.com/uciml/sms-spam-collection-dataset. I then split the data set into a training set and a testing set. The training set consist of 4457 entries where 603 are spam and 3854 are ham while the test dataset consists of 1116 entries where 146 are spam and 971 are ham. I used the scikit library to train each of the models: Naïve Bayes, SVM, Random Forest and Logistic Regression. The four models were added to pickle files to allow for quick access to the models. I used the scikit learn metrics library to obtain the confusion matrix, accuracy, f1, precision & recall and sensitivity and specificity scores which helped me decide which model to use with my front-end web application.

Flask was used to create the web application. This web app houses the functions of my system. It is an easy-to-use application. When the user successfully registered with the system they then will need to login to their account. The application will display the user’s emails once logged in. The emails are classified using classification algorithms. The four algorithms I implemented are Naive Bayes, SVM, Random Forest and Logistic Regression. The outcome of the algorithm determines whether the email is spam or not spam. However, in the end-product I chose Naïve Bayes to classify the emails as spam or ham. I choose Naïve Bayes over the other three algorithms because the accuracy score was 0.9856, this score is calculated by the number of items categorized correctly divided by the total number of items. It is the fraction of the time the classifier is correct. The f1 score helped me in making the decision. The F1 score is to measure a trade-off between precision and recall. A low F1 score means the spam filtering is pickier which may lead to fewer real spam emails being marked as spam and allow it into your inbox. The naïve bayes f1 score for ham is 0.9917 and for spam is 0.9448 while comparing to Random Forest’s f1 score is 0.9862 for ham and 0.8988 for spam. The figure which made me decide the final decision also was the false negative score. Naïve Bayes predict only 8% of emails are not spam but they are spam whereas SVM predicts 13% of emails are not spam but they are spam. This means using SVM to classify your emails will lead to more spam emails reaching your inbox and increase the risk of phishing, malspam etc.
If the email is spam it will display in the spam folder for the user otherwise the email will be in their inbox folder. The administrator is the only user who will have access to the spam reports and have access to view all the registered users for the web application. The system only has one administrator account. The flask app queries the SQL database to display the emails, generate the spam reports and display all other information on the web application.

System components

Hardware

- Windows machine hardware
- ✓ Acting as mail server running local software
- ✓ Acting as web server running local software

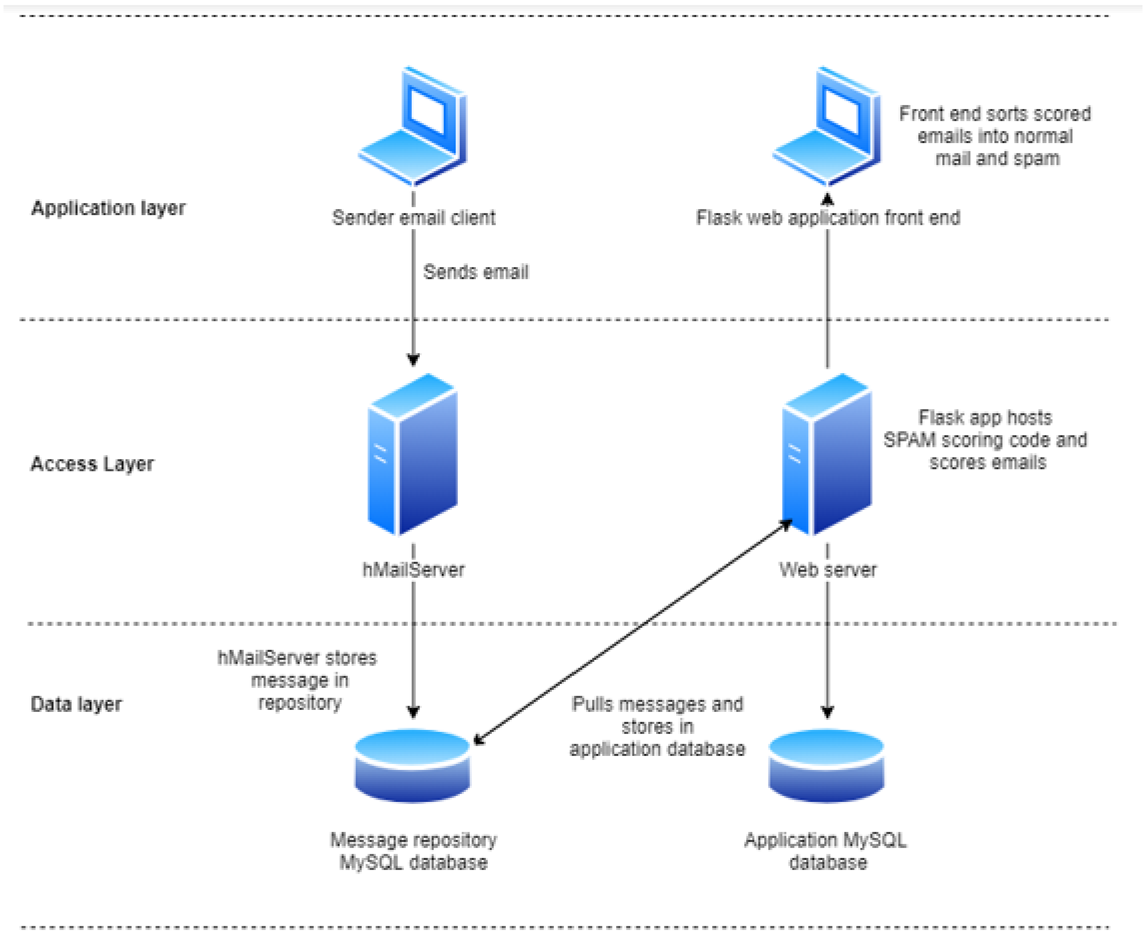
Software

- Mail server (hMailServer)
- ✓ Setup locally
- ✓ Stores emails

- Flask application
- ✓ Created to display all the emails
- ✓ The systems front-end

System architecture

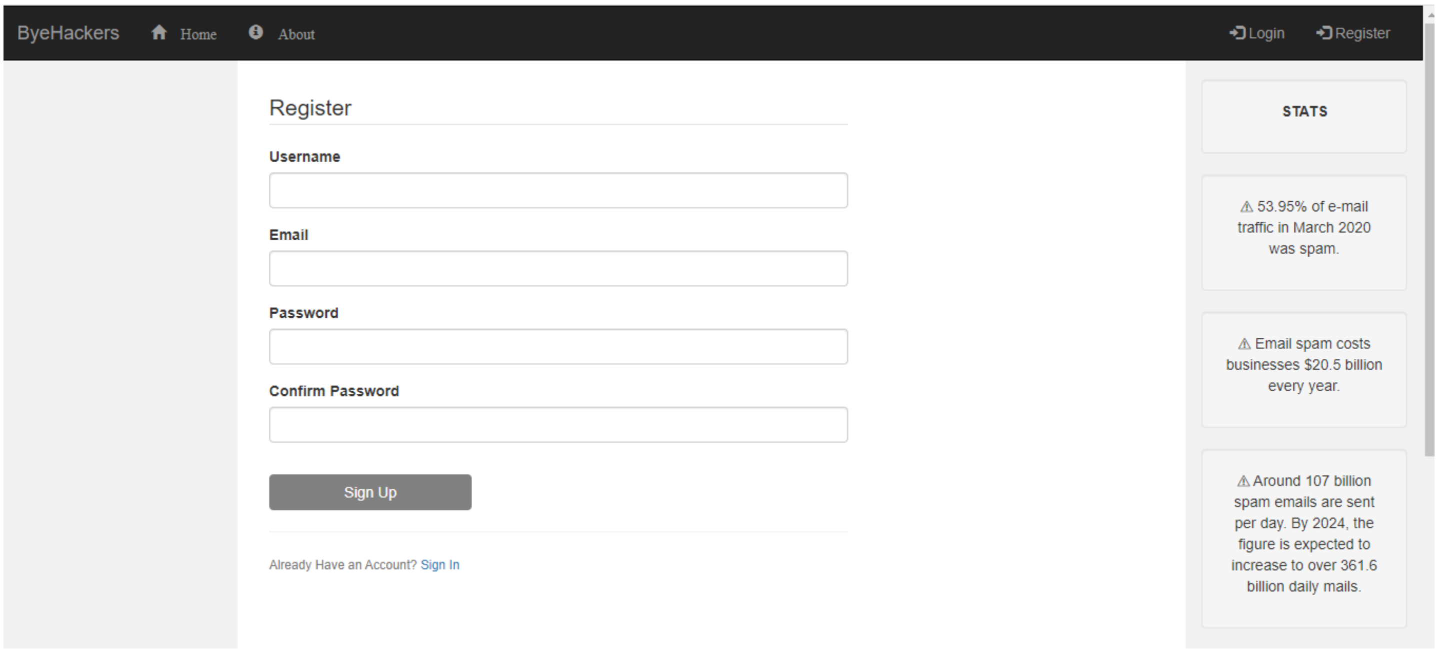
The following is a high level three tier diagram to show how this project is implemented. It shows the sender's client sending an email which reaches hMailServer which acts as an SMTP, MTA and deals with IMAP and POP3 protocols. This information is stored in a message repository in .eml format. Our web application is hosted on a local web server, which has access to a local MySQL database. The flask application hosts code to periodically check for new emails, parse the stored email messages into storable string format and stores it in the MySQL database. Spam scoring takes place in the flask application and the spam score is stored in the MySQL database. The front end of the flask application provides the user a way to log in, when logged in there many pages which separate mails like sent, inbox and spam.


Application layer
Access Layer
Data layer
Sender email client
Sends email
Flask web application front end
Front end sorts scored emails into normal mail and spam
hMailServer
Web server
Flask app hosts SPAM scoring code and scores emails
hMailServer stores message in repository
Pulls messages and stores in application database
Message repository MySQL database
Application MySQL database

Web Application User Interface

Registration

Before you can receive and view your emails you must register with the application. You must complete the registration form as seen below. You will have to provide a username, email address and password.


ByeHackers Home About Login Register

Register

Username

Email

Password

Confirm Password

Sign Up

Already Have an Account? Sign In

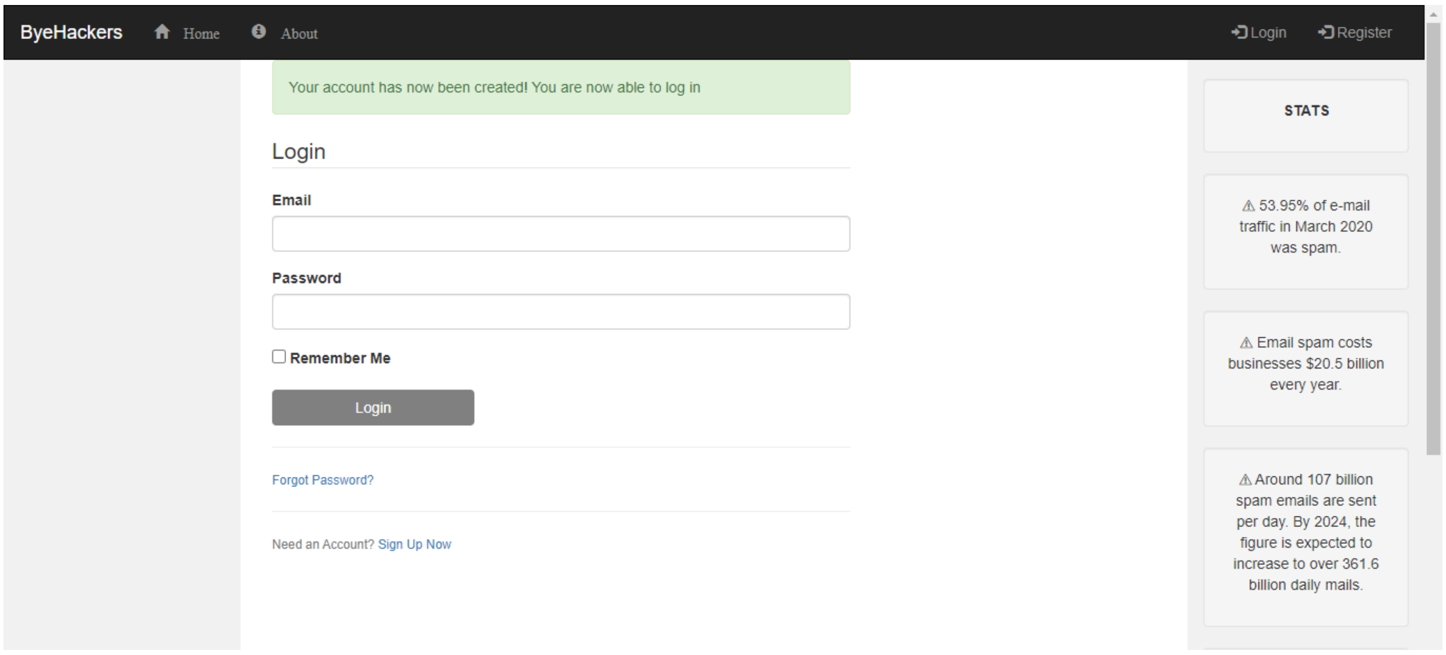
STATS

⚠ 53.95% of e-mail traffic in March 2020 was spam.

⚠ Email spam costs businesses $20.5 billion every year.

⚠ Around 107 billion spam emails are sent per day. By 2024, the figure is expected to increase to over 361.6 billion daily mails.

If all details entered are authenticated, a success message will appear to the user as seen below.


ByeHackers Home About Login Register

Your account has now been created! You are now able to log in

Login

Email

Password

Remember Me

Login

Forgot Password?

Need an Account? Sign Up Now

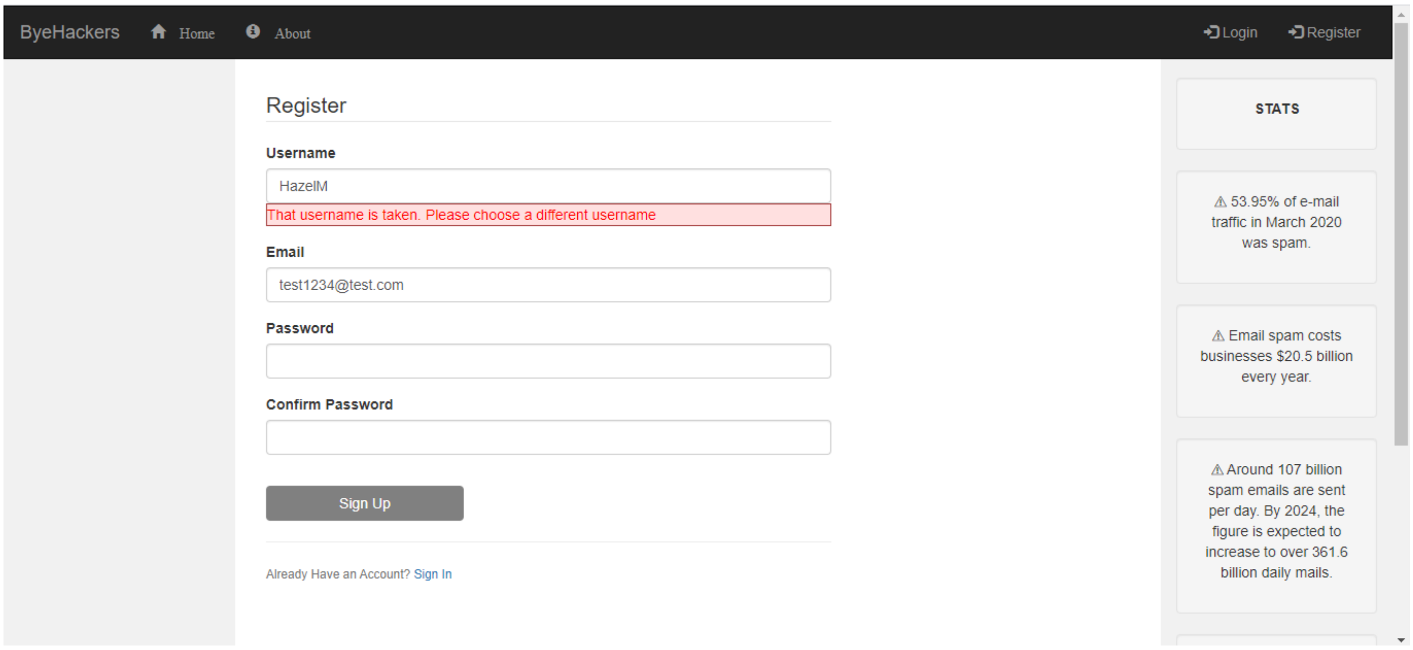
STATS

⚠ 53.95% of e-mail traffic in March 2020 was spam.

⚠ Email spam costs businesses $20.5 billion every year.

⚠ Around 107 billion spam emails are sent per day. By 2024, the figure is expected to increase to over 361.6 billion daily mails.

If the username and/or email address is already in use an error message will display to notify the user.



Register

Username

HazelM

That username is taken. Please choose a different username

Email

test1234@test.com

Password

Confirm Password

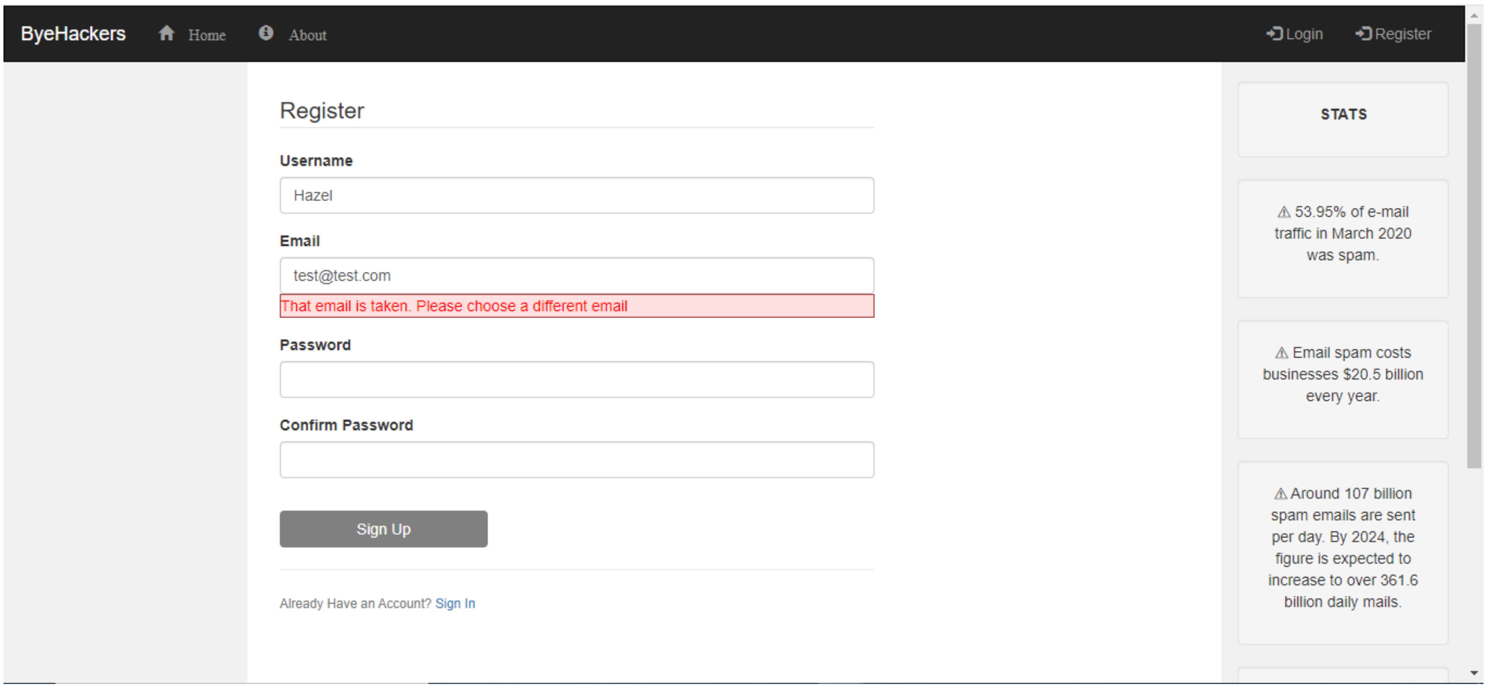
Already Have an Account? Sign In

STATS

⚠ 53.95% of e-mail traffic in March 2020 was spam.

⚠ Email spam costs businesses $20.5 billion every year.

⚠ Around 107 billion spam emails are sent per day. By 2024, the figure is expected to increase to over 361.6 billion daily mails.



Register

Username

Hazel

Email

test@test.com

That email is taken. Please choose a different email

Password

Confirm Password

Already Have an Account? Sign In

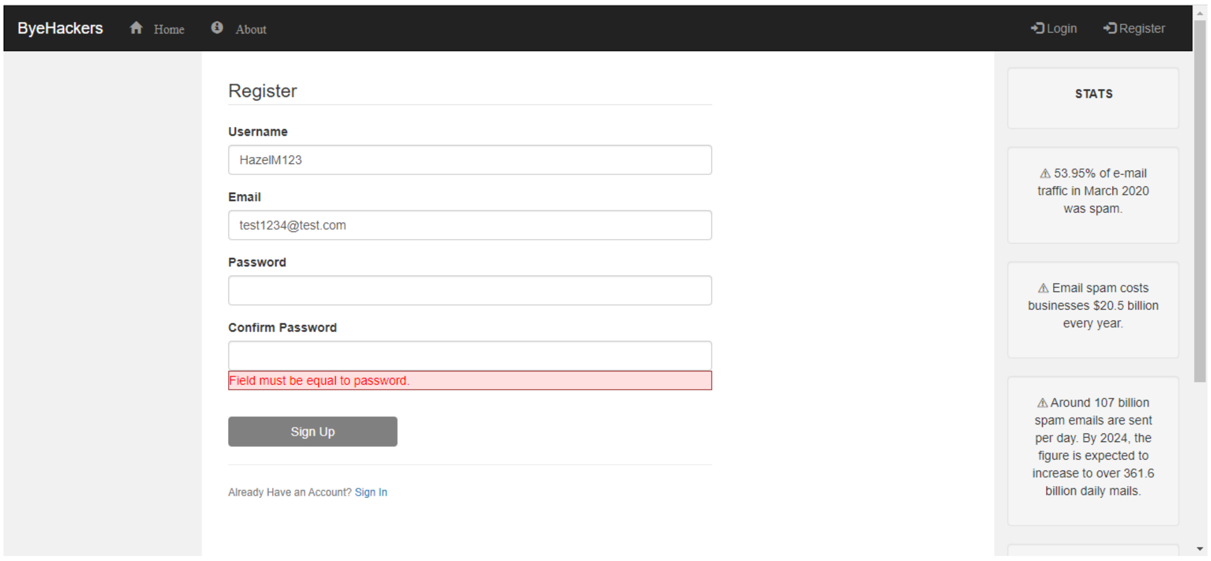
STATS

⚠ 53.95% of e-mail traffic in March 2020 was spam.

⚠ Email spam costs businesses $20.5 billion every year.

⚠ Around 107 billion spam emails are sent per day. By 2024, the figure is expected to increase to over 361.6 billion daily mails.

If the password field and the confirm password field do not match an error message will be displayed to the user.



Register

Username

HazelM123

Email

test1234@test.com

Password

Confirm Password

Field must be equal to password.

Already Have an Account? Sign In

STATS

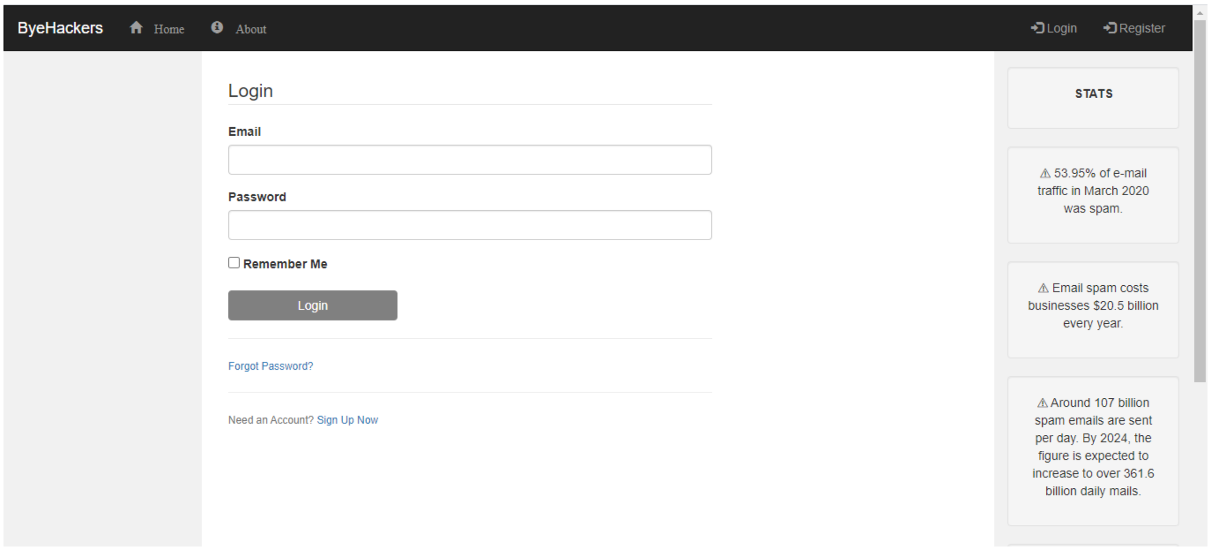
⚠ 53.95% of e-mail traffic in March 2020 was spam.

⚠ Email spam costs businesses $20.5 billion every year.

⚠ Around 107 billion spam emails are sent per day. By 2024, the figure is expected to increase to over 361.6 billion daily mails.

Login

Once you have registered successfully, you can then login to the application. To login you will need to provide the email address and password used when registering.



Login

Email

Password

☐ Remember Me

Forgot Password?

Need an Account? Sign Up Now

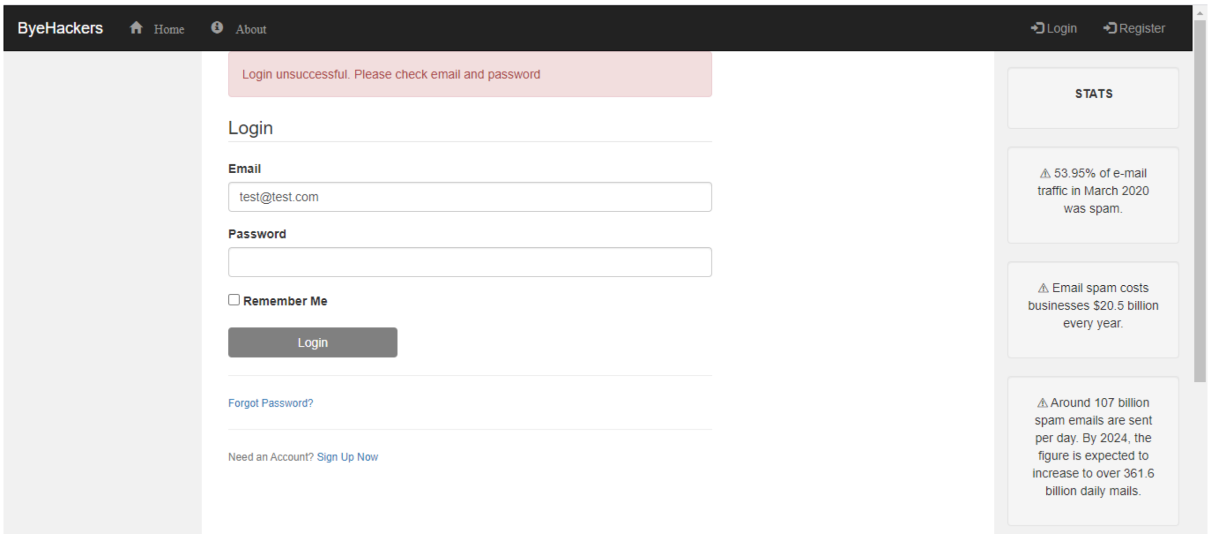
STATS

⚠ 53.95% of e-mail traffic in March 2020 was spam.

⚠ Email spam costs businesses $20.5 billion every year.

⚠ Around 107 billion spam emails are sent per day. By 2024, the figure is expected to increase to over 361.6 billion daily mails.

If your credentials do not match an error message will display for the user.



Login unsuccessful. Please check email and password

Login

Email: test@test.com

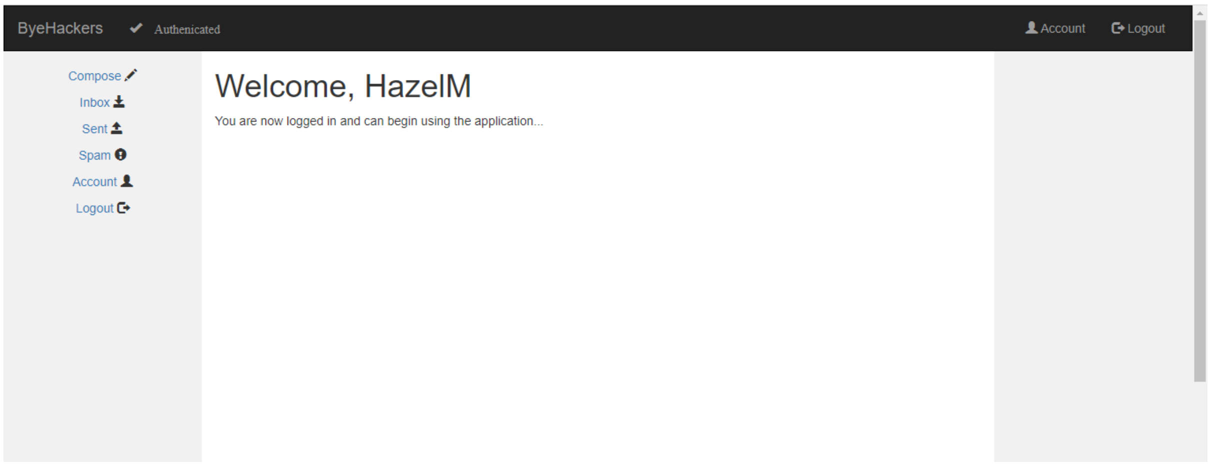
Need an Account? Sign Up Now

⚠ 53.95% of e-mail traffic in March 2020 was spam.

⚠ Email spam costs businesses $20.5 billion every year.

⚠ Around 107 billion spam emails are sent per day. By 2024, the figure is expected to increase to over 361.6 billion daily mails.

Once you have logged in successfully you will be greeted with the below screen.



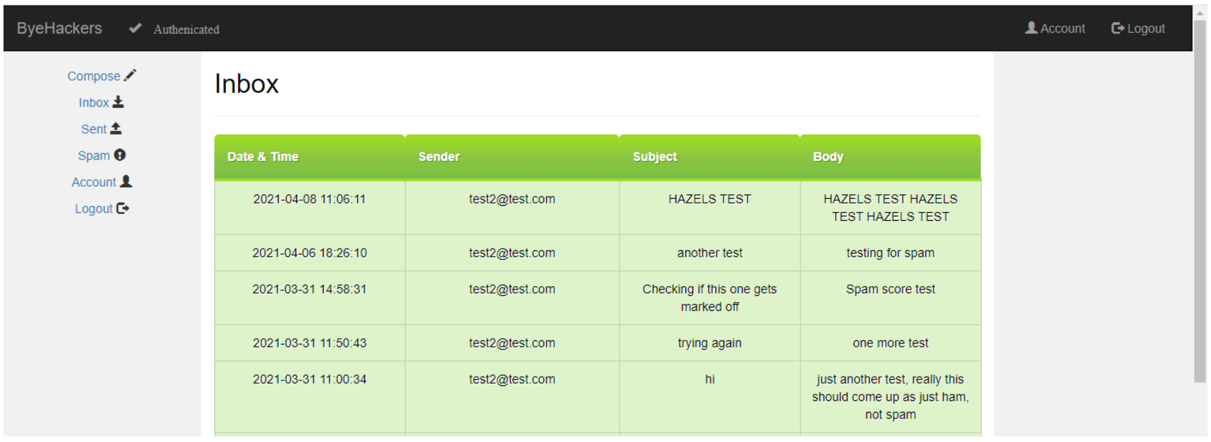
Welcome, HazelM

You are now logged in and can begin using the application...

General Usage

Once you have registered and logged into the application you have access to your emails. You can view your inbox, sent, spam mails and account information. However, if you have logged into the application as the administrator you have additional features such as view spam reports and view all the registered users of the application.

Inbox mail

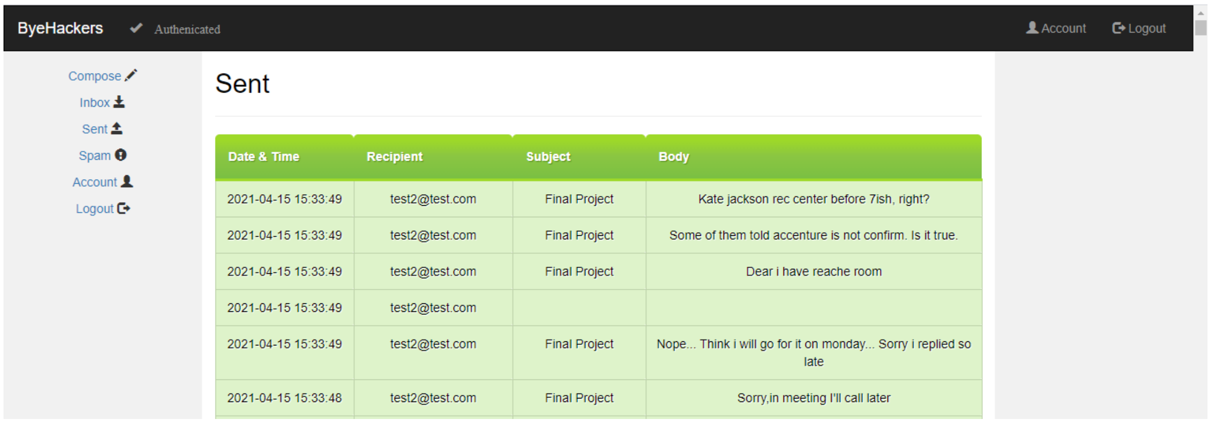

ByeHackers ✔ Authenicated | Account | Logout

Compose | Inbox | Sent | Spam | Account | Logout

Inbox

Date & Time	Sender	Subject	Body
2021-04-08 11:06:11	test2@test.com	HAZELS TEST	HAZELS TEST HAZELS TEST HAZELS TEST
2021-04-06 18:26:10	test2@test.com	another test	testing for spam
2021-03-31 14:58:31	test2@test.com	Checking if this one gets marked off	Spam score test
2021-03-31 11:50:43	test2@test.com	trying again	one more test
2021-03-31 11:00:34	test2@test.com	hi	just another test, really this should come up as just ham, not spam

Sent mail

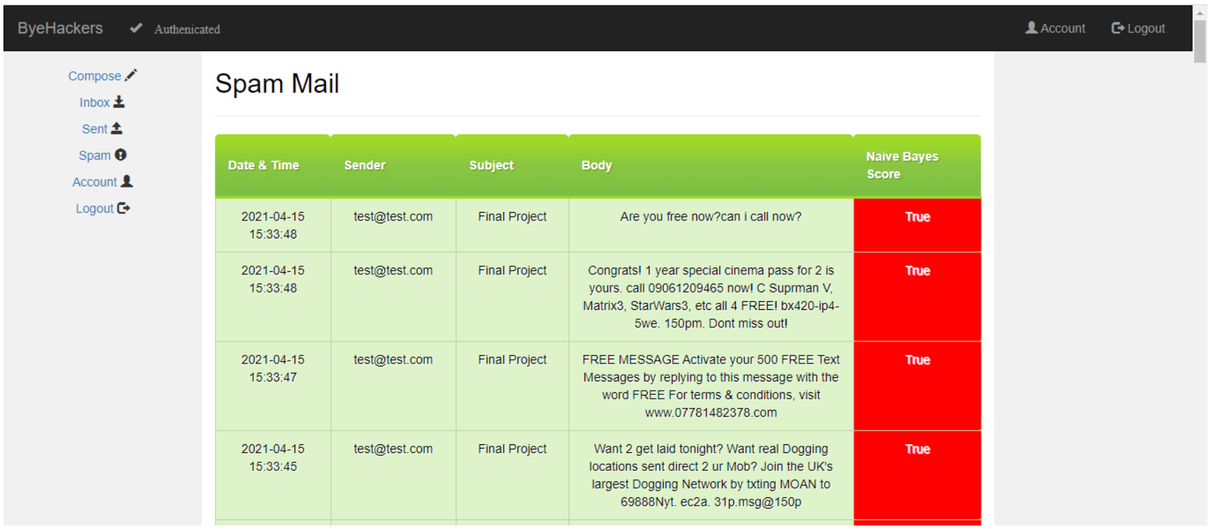

ByeHackers ✔ Authenicated | Account | Logout

Compose | Inbox | Sent | Spam | Account | Logout

Sent

Date & Time	Recipient	Subject	Body
2021-04-15 15:33:49	test2@test.com	Final Project	Kate jackson rec center before 7ish, right?
2021-04-15 15:33:49	test2@test.com	Final Project	Some of them told accenture is not confirm. Is it true.
2021-04-15 15:33:49	test2@test.com	Final Project	Dear i have reache room
2021-04-15 15:33:49	test2@test.com		
2021-04-15 15:33:49	test2@test.com	Final Project	Nope... Think i will go for it on monday... Sorry i replied so late
2021-04-15 15:33:48	test2@test.com	Final Project	Sorry,in meeting I'll call later

Spam mail

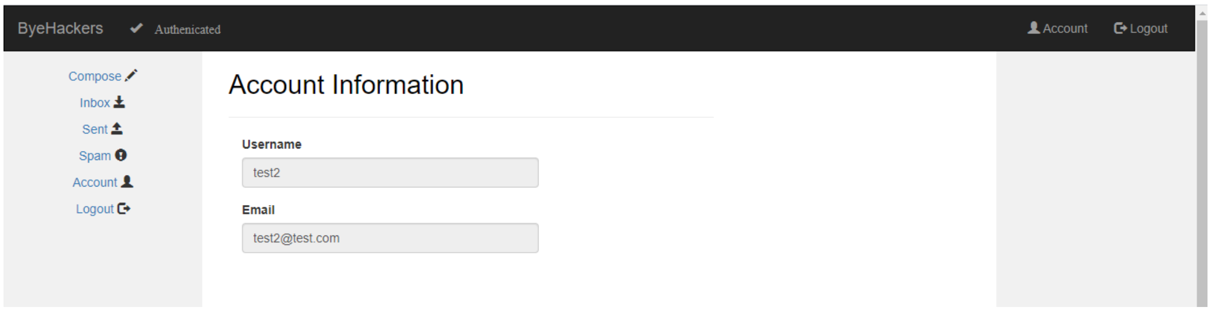

ByeHackers ✔ Authenicated | Account | Logout

Compose | Inbox | Sent | Spam | Account | Logout

Spam Mail

Date & Time	Sender	Subject	Body	Naive Bayes Score
2021-04-15 15:33:48	test@test.com	Final Project	Are you free now?can i call now?	True
2021-04-15 15:33:48	test@test.com	Final Project	Congrats! 1 year special cinema pass for 2 is yours. call 09061209465 now! C Suprman V, Matrix3, StarWars3, etc all 4 FREE! bx420-ip4-5we. 150pm. Dont miss out!	True
2021-04-15 15:33:47	test@test.com	Final Project	FREE MESSAGE Activate your 500 FREE Text Messages by replying to this message with the word FREE For terms & conditions, visit www.07781482378.com	True
2021-04-15 15:33:45	test@test.com	Final Project	Want 2 get laid tonight? Want real Dogging locations sent direct 2 ur Mob? Join the UK's largest Dogging Network by txting MOAN to 69888Nyt. ec2a. 31p.msg@150p	True

Account information



Account Information

Username

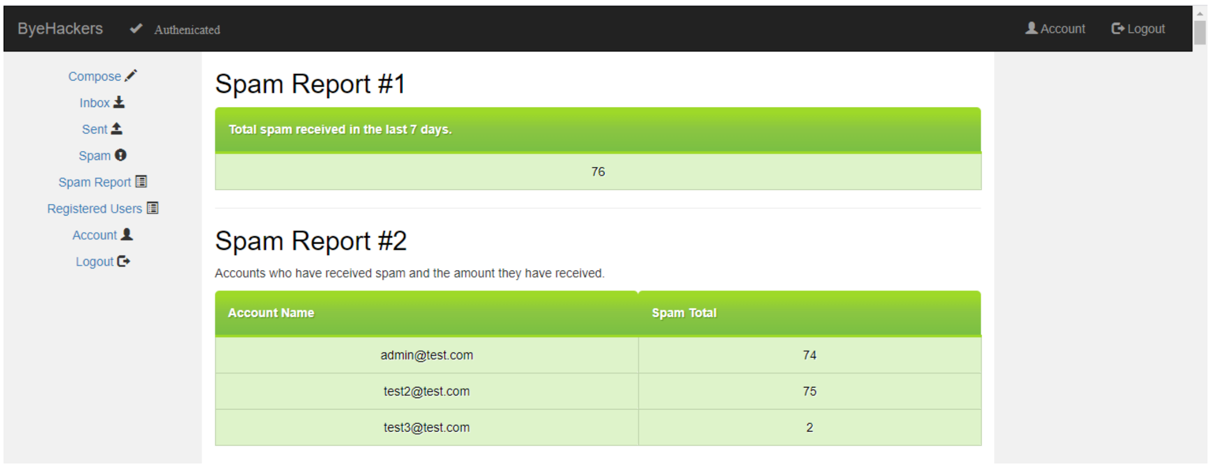
test2

Email

test2@test.com

Spam report

The admin user only has access to this page.



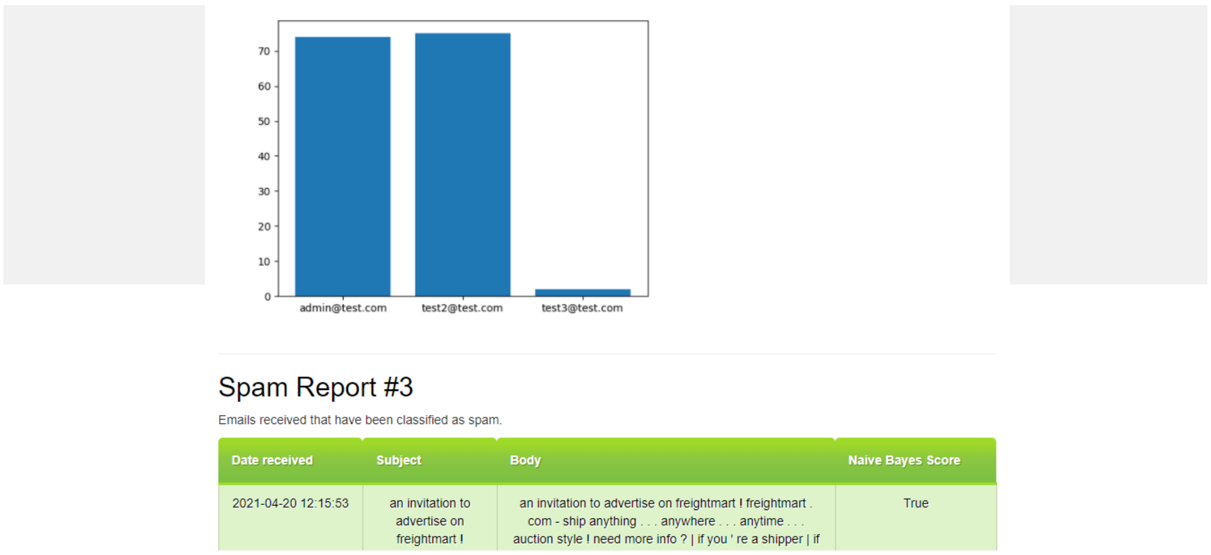
Spam Report #1

Total spam received in the last 7 days.
76

Spam Report #2

Accounts who have received spam and the amount they have received.

Account Name	Spam Total
admin@test.com	74
test2@test.com	75
test3@test.com	2



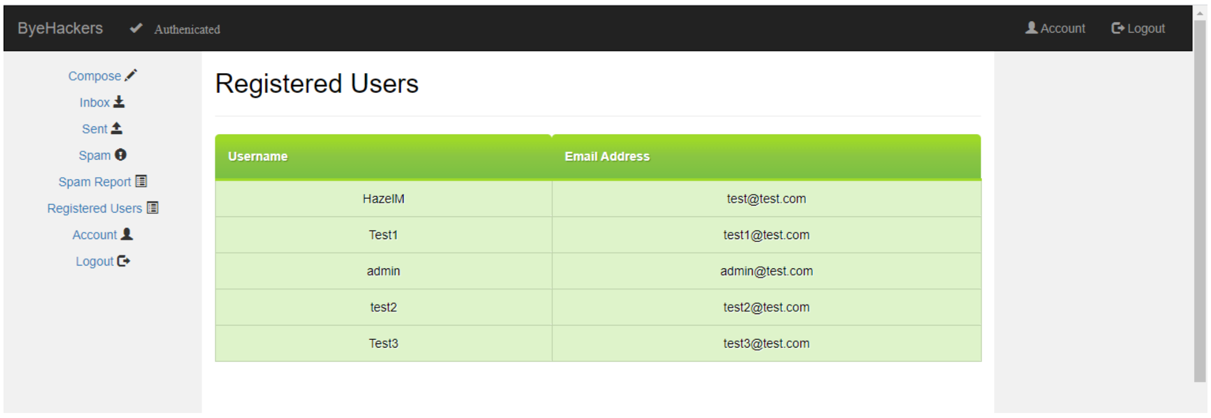
Spam Report #3

Emails received that have been classified as spam.

Date received	Subject	Body	Naive Bayes Score
2021-04-20 12:15:53	an invitation to advertise on freightmart !	an invitation to advertise on freightmart ! freightmart . com - ship anything . . . anywhere . . . anytime . . . auction style ! need more info ? | if you ' re a shipper | if	True

Registered Users

The admin user only has access to this page. This page displayed all the registered users of the application.



Registered Users

Username	Email Address
HazelM	test@test.com
Test1	test1@test.com
admin	admin@test.com
test2	test2@test.com
Test3	test3@test.com

Learning outcomes

From completing this project, I have learnt end-less amounts over the past 6 months. I have been learning since day one of researching. I took on this project to push myself and put myself out of my comfort zone. Prior to working on this project, I had zero experience with python and machine learning. Because of this research for the project took me a lot longer than originally planned. However, as the research was completed the implementation started smoothly and at a steady pace. This project allowed me to learn about new tools and technologies such as hMailServer, machine learning algorithms, flask, and python. Also, this was my first time creating and implementing an end-to-end web application and understanding how the database interacts. My supervisor guided me with great examples and guidance whenever needed throughout the project duration.

Technical and Personal achievements

As stated above, I had zero experience with python and machine learning when I took on this project. From extensive research and learning the new tools and technologies was a steep learning curve than I had imagined.

- **HTML**

I used HTML within the flask application to create a professional and positive look for my web application. I have a basic knowledge of the HMTL language as it was covered in the

Web Development module, I studied in 2nd year of my degree. By implemented this language in greater details than thought in 2nd year I feel will be of very high value to me in future careers.

- **Bootstrap**

The CSS I implemented was bootstrap. I used bootstrap to design how my web application would look like to users. I had never used bootstrap in the past this means it was a completely new framework for me to use. Bootstrap is a popular framework and use quite often which means it great experience to gain for entering the workforce.

- **MySQL**

Incorporating MySQL with databases was covered at a high level in the Web development module in 2nd year of my degree. I used MySQL frequently in my project which has allowed me to further my skills in this area.

- **Python**

I implemented my project using python as my programming language. This language was never thought in my degree which meant I had zero experience starting the project. However, it was easier than I had thought it would be. It was a great learning aspect. I used an online course to help me with the python language. The course can be found at the following link: https://www.codecademy.com/learn/learn-python. Due to timing constraints, I didn’t finish the online course, however, the course enabled me to implement the aspects needed throughout my project.

- **Flask**

I implemented flask as my front-end. Like python, flask took me a lot longer than planned to grasp. One of the main reasons I used flask was because it is specifically for python and python was the core language used for the project. I followed many tutorials online to help guide me through setting up the flask application. One online tutorial I would highly recommend is the following by Corey Schafer:
https://www.youtube.com/watch?v=MwZwr5Tvyxo

Project Review

This section of the document discusses aspects achieved and not achieved, aspects that went wrong during implementation, thing I would change if starting again and finally any problems I encountered throughout.

Positive aspects/aspects achieved

The development of the email spam filter tool was successful. The tool achieved all the functions initialized at the very start of this project which means users can safely send and receive emails and any emails containing spam will be placed in a separate folder for the user. This project has ended at a very good stage. The end-product is presentable and a usable tool.

The table below is the metric table from the functional specification document. Every initialize goal was successful.

Criteria	**Description**
Spam detection	In the process of selecting and evaluating the machine learning model the use of confusion matrix to compare the results of the models against a dataset where we already predict the answer. For some scoring models like logistic regression which was also tested the 0.5 score is likely a threshold that gives a classifier with reasonably good accuracy. The spam filter has a specificity of 0.9448, which means that it marks about 0.5662% of non-spam emails as spam.
Spam detection	In the process of selecting and evaluating the machine learning model we are seeking a low level of Type I error (false positive), to stop important emails being classified as spam which are not spam. Because the precision score is 0.997 this means 0.1% of emails maybe wrongly classified.
Spam detection	In the process of selecting and evaluating the machine learning model we are seeking a low level of Type II error (false negatives), to stop spam being classified as a normal mail. Because the accuracy score is 0.9856 means the Naïve Bayes models is very accurate at classifying the emails correctly. The False Negative result from the confusion matrix is 8.
Spam detection	Ensure the Recall (True positive rate) of the machine learning model to ensure the correct proportion of actual positives was identified correctly. The recall score for the Naïve Bayes model I implemented is 0.99174. I have high recall and precision is emphasized over recall. This is appropriate for a spam filter, because it is more important to not lose non-spam email than it is to

	filter every single piece of spam out of our inbox. The true positive rate is 962.
Spam detection	Ensure the Precision of the machine learning model is sufficient. This score should be at least 0.91. The precision score for the Naïve Bayes models I implemented is average 0.9682. The higher the precision means less emails are incorrectly classified.
Security	The web application is not vulnerable to SQL injection attacks and uses an object relational mapper.
Security	The user's passwords are stored using a hashing key to encrypt them.
Errors	Low level of application errors.
Usability	Users are prompted appropriately at all stages of user input
Usability	The system is easy to use and intuitive in design
Usability	The system makes use of labels for accessibility
Reliability	The system works and does not throw exceptions during standard usage

Aspects not achieved

Due to time constraints, not all functions for the flask application were achieved. I would implement the function of allowing a user to send mails from the web app, request to change their password and finally allow the admin to remove accounts if requested by a user. However, the focus on this project was spam classification, as a result, I am very happy in what was achieved over the six months.

Aspects gone wrong

When testing the spam classification using the mail merge technique the subject lines were too long which was causing issues with the scrape function and the emails were not being stored in the database, so I had to delete every email in the hMailServer directory for that user and reduce the subject line length and re-start the testing.

While coding the web application the jinja templates were picking up python variables when they were inside the html comments. Also, they were not html this python code was displaying and as a result hindered me from getting data displayed.

When I was storing the models to the pickle files, I was not storing the count vectorizer as well as the models. So, I would initiate a new count vectorizer every time and the models would not work as the vectors for the words did not match what the model had stored. I solved this issue by also storing the count vectorizer in a pickle file.

Things I would change

If I were given the option to begin this project again from the start, I would change the following to improve the project:

- Conduct extra research on flask

I feel if I had spent more time researching flask, I could have produced a more professional design for my application.

- Time management

For my final year of college, managing time was critical as there was extensive amounts of assignments due throughout the year. However, at some points throughout the year this project was pushed back due to other modules exams and reports due date approaching and taking me longer than expected to complete. This meant any lunch breaks or spare time in the evenings/weekends was spent as wisely as possible on the project. I learnt a large amount of time should be spent on researching the required tools and technologies needed and how they work before trying to implement them without any knowledge. Also, research in-dept the best fitted language to use to implement with the tools and technologies. I feel in the long-term this strategy is the key to success.

- Development inconsistency

I would finish one task at a time if I were starting the project again. At the beginning I started another task before the task I was on was finished. This became very confusing and more difficult to track the progress of my project. As some tasks I was working on were backend tasks such as implementing the machine learning models while also working on the GUI for the web app development.

Problems encountered

I thought that the functions for the four models (Naïve Bayes, SVM, Random Forest and logistic Regression) had scoring functions I could use with them. But in fact, I had to use a combination of using confusion matrix values and metric functions from the sklearn kit.

(sklearn.metrics.recall_score — scikit-learn 0.24.2 documentation, 2021)
(sklearn.metrics.precision_score — scikit-learn 0.24.2 documentation, 2021)
(sklearn.metrics.f1_score — scikit-learn 0.24.2 documentation, 2021)
(sklearn.metrics.confusion_matrix — scikit-learn 0.24.2 documentation, 2021) (204.4.2 Calculating Sensitivity and Specificity in Python | Statinfer, 2021)

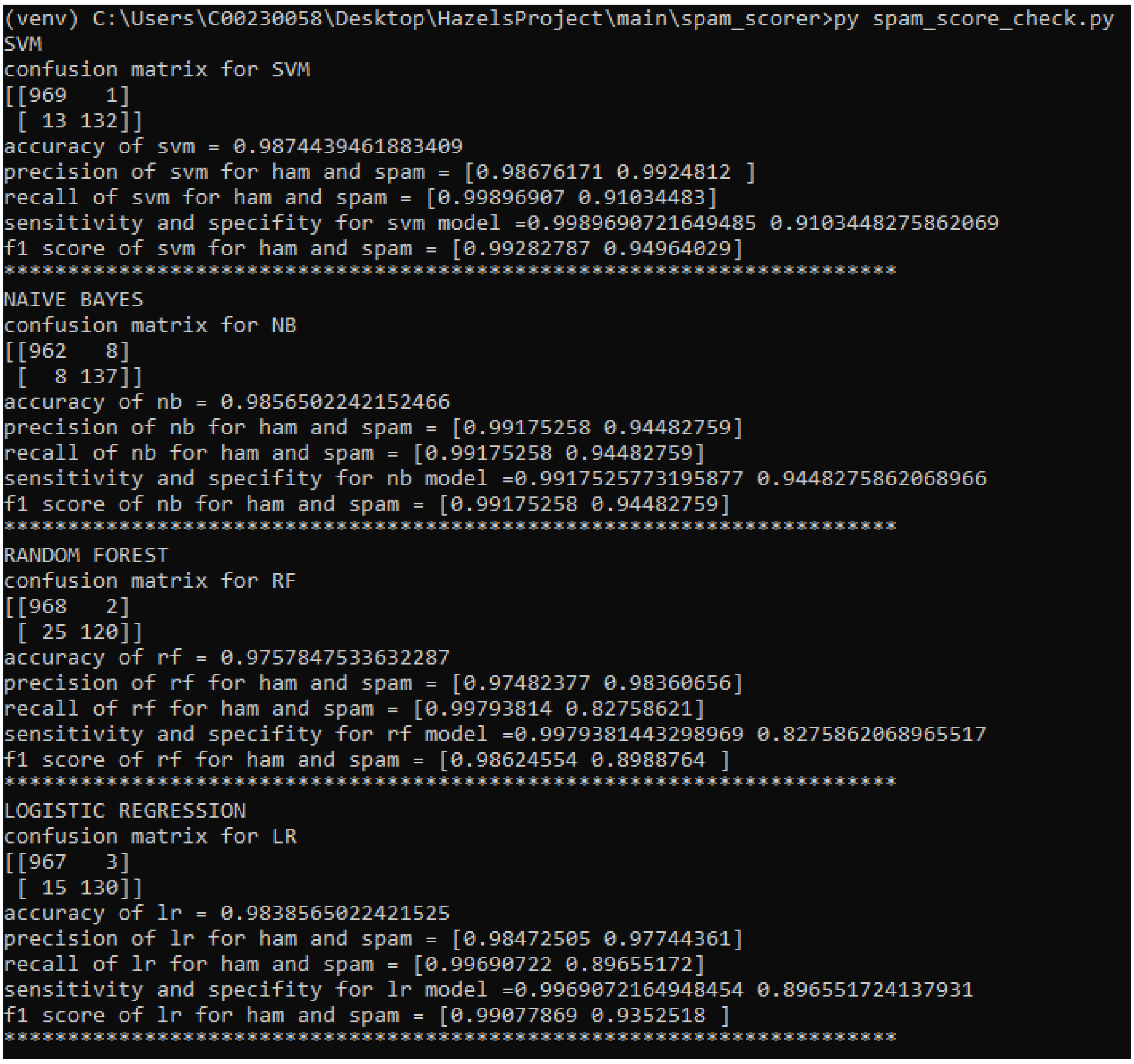
Testing

Machine learning models test

After conducting the below tests I have decided to implement the Naïve Bayes machine learning model on the web application for the following reasons:

- The accuracy score was 0.9856, this score is calculated by the number of items categorized correctly divided by the total number of items. It is the fraction of the time the classifier is correct.
- The F1 score is to measure a trade-off between precision and recall. A low F1 score means the spam filtering is pickier which may lead to fewer real spam emails being marked as non-spam and allow it into your inbox. The naïve bayes f1 score for ham is 0.9917 and for spam is 0.9448 while comparing to Random Forest F1 score is 0.9862 for ham and 0.8988 for spam.
- The spam filter has a specificity of 0.9448, which means that it marks 0.5662% of non-spam email as spam.
- The recall score for the Naïve Bayes model I implemented is 0.99174. I have high recall and precision is emphasized over recall. This is appropriate for a spam filter, because it is more important to not lose non-spam email than it is to filter every single piece of spam out of our inbox.
- The precision score for the Naïve Bayes models I implemented is 0.9917. The higher the precision means less emails are incorrectly classified.
- Naïve Bayes predict only 8% of emails are not spam but they are spam whereas SVM predicts 13% of emails are not spam but they are spam. This means using SVM to

classify your emails will lead to more spam emails reaching your inbox and increase the risk of phishing, malspam etc.



```
(venv) C:\Users\C00230058\Desktop\HazelsProject\main\spam_scorer>py spam_score_check.py
SVM
confusion matrix for SVM
[[969   1]
 [ 13 132]]
accuracy of svm = 0.9874439461883409
precision of svm for ham and spam = [0.98676171 0.9924812 ]
recall of svm for ham and spam = [0.99896907 0.91034483]
sensitivity and specifity for svm model =0.9989690721649485 0.9103448275862069
f1 score of svm for ham and spam = [0.99282787 0.94964029]
**********************************************************************
NAIVE BAYES
confusion matrix for NB
[[962   8]
 [  8 137]]
accuracy of nb = 0.9856502242152466
precision of nb for ham and spam = [0.99175258 0.94482759]
recall of nb for ham and spam = [0.99175258 0.94482759]
sensitivity and specifity for nb model =0.9917525773195877 0.9448275862068966
f1 score of nb for ham and spam = [0.99175258 0.94482759]
**********************************************************************
RANDOM FOREST
confusion matrix for RF
[[968   2]
 [ 25 120]]
accuracy of rf = 0.9757847533632287
precision of rf for ham and spam = [0.97482377 0.98360656]
recall of rf for ham and spam = [0.99793814 0.82758621]
sensitivity and specifity for rf model =0.9979381443298969 0.8275862068965517
f1 score of rf for ham and spam = [0.98624554 0.8988764 ]
**********************************************************************
LOGISTIC REGRESSION
confusion matrix for LR
[[967   3]
 [ 15 130]]
accuracy of lr = 0.9838565022421525
precision of lr for ham and spam = [0.98472505 0.97744361]
recall of lr for ham and spam = [0.99690722 0.89655172]
sensitivity and specifity for lr model =0.9969072164948454 0.896551724137931
f1 score of lr for ham and spam = [0.99077869 0.9352518 ]
**********************************************************************
```

Functionality testing

- ✓ **Spam Report**

To test the spam report, a batch of 1000 emails was sent to each registered user inbox. The emails sent had documented predicted spam classification values. The spam report was then compared to the predicted values and any discrepancies mitigated.

Exploratory testing

Using exploratory testing students were asked to use the system, to test functionalities such as login, logout, view spam, view inbox etc. Any issues found were noted and resolved before submission.

Comparison to Original Design and Specification

During development, the email spam filter tool did not change significantly. All the core functions have been implemented in the project as specified at the start. However, certain functions on the web-app were not completed due to prioritising the backend processes over the front-end because the back end of this project is the core feature. If the models are not trained and tested correctly this will cause issues when classifying the emails as spam or ham and could lead to security risks for the users, therefore this area in the project needed a significant amount of attention.

Future features

Due to time constraints, some features would not have been implemented within the deadline. Therefore, there are also future features in my project that could be implemented. I would like to implement a fully functioning web application to allow users to send emails from the web application instead of using hMailServer and allowing them to deal with their emails for example, delete an email, forward an email etc. The spam classification was only the tip of the iceberg with my interest in how it all worked.

I would also like to make a mobile application version for this project as many people use their mobile to send and check emails.

Acknowledgments

There are many people I would like to thank and acknowledge for all their kindness and help throughout the year. Completing my final year in IT Carlow during a pandemic was a challenge but I would not have achieved this without the help from all my lectures and classmates. The support from my classmates this year will be certainly unforgettable. I would also like to thank my close friends Katie Brophy and Sine Dohney. These two girls have been extremely supportive since I met them at the start of second year and every year since. All the IT Carlow staff are approachable and love sharing their knowledge with every student. All my lectures from first year have always guided me and helped me solved any issues I faced along the way especially Richard Butler, Keara Barrett, and James Egan.

Finally, I would like to thank my project supervisor, James Egan. James supported me from day one with my project and set me on the right track. He always provided me with guidance, support, and advice. James answered any questions I had promptly and always had solutions. Thank you very much James for all your time, effort and help you have put into my project to get the project completed.

Plagiarism Declaration

I declare all submitted work is my own work. I have cited using the institutes standards, any sources of quotations, paraphrases, tables, diagrams, or other material where intellectual property rights may reside. Bibliography is provided in each document where needed. I understand it is serious offence if I fail to obey with the Institute’s regulations governing plagiarism.

Student Name: Hazel Murphy

Student Number: C00230058

Signature: Hazel Murphy

Date: 30th April 2021

Bibliography

Scikit-learn.org. 2021. *sklearn.metrics.recall_score — scikit-learn 0.24.2 documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html> [Accessed 12 April 2021].

Scikit-learn.org. 2021. *sklearn.metrics.precision_score — scikit-learn 0.24.2 documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html> [Accessed 12 April 2021].

Scikit-learn.org. 2021. *sklearn.metrics.f1_score — scikit-learn 0.24.2 documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html> [Accessed 12 April 2021].

Scikit-learn.org. 2021. *sklearn.metrics.confusion_matrix — scikit-learn 0.24.2 documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html> [Accessed 12 April 2021].

Statinfer | Data Science starts here. 2021. *204.4.2 Calculating Sensitivity and Specificity in Python | Statinfer*. [online] Available at: <https://statinfer.com/204-4-2-calculating-sensitivity-and-specificity-in-python/> [Accessed 12 April 2021].