**Contributions and Technical Artefacts**

**Frontend Design**

The frontend design was vital since it connected users to the AI-powered vulnerability detection system. To improve user experience, I created responsive, user-friendly, and attractive web pages.

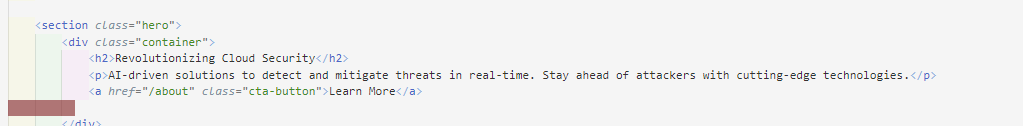


Figure Hero Section

A screenshot of a computer

Description automatically generated

Figure Features Section

A screenshot of a computer code

Description automatically generated

Figure Navigator and Footer

A close-up of a computer screen

Description automatically generated

Figure Footer design

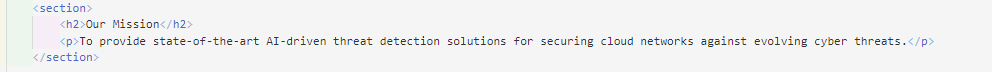


Figure Mission Section

A close-up of text

Description automatically generated

Figure Team Section

A computer code on a white background

Description automatically generated

Figure Threat Detection Section



Figure Displaying predicted result

1. **Development of Responsive Web Pages:**
   * I created four main web pages—Home, About and Admin —using HTML and CSS, ensuring a cohesive design across the application (Figures 1–8).
   * **Home Page:**
     + HTML and CSS were used to create four web pages (Home, About, and Admin) for a consistent application design (Figures 1-8).
     + Designing a homepage hero section with a stunning background and strong call-to-action button (Figure 1).
     + In a grid format, the system's benefits were clear (Figure 2).
     + Added navigation links to header and footer for smooth page transitions (Figures 3 and 4).
   * **About Page:**
     + The About Page describes the project's goals.
     + Team details, including photos and roles, were arranged for readability (Figures 5 and 6).
   * **Admin Page:**
     + On the Admin Page, a dynamic form was created for threat detection, allowing users to input textual data for analysis (refer to Figure 7).
     + Dynamic prediction results were shown underneath the form to promote engagement (Figure 8).
2. **Ensuring Cross-Browser Compatibility:**
   * To ensure consistent functionality and design across browsers, I tested all web pages on Chrome, Firefox, Edge, and Safari.
   * Changed CSS attributes and vendor prefixes to fix browser-specific rendering issues.
3. **Responsiveness Across Devices:**
   * CSS media queries optimise layout for computers, tablets, and phones.
   * Make buttons and forms touch-friendly for smaller screens.
4. **Challenges and Solutions:**
   * **Challenge:** Aesthetics, usability, and fast loading.
   * **Solution:** Reduced heavy elements like high-resolution images and used efficient CSS to improve performance.

**Model Evaluation and Comparison**

Evaluation and comparison of machine learning models were needed to identify the best deployment algorithm. My contributions were model performance metrics analysis and presentation.

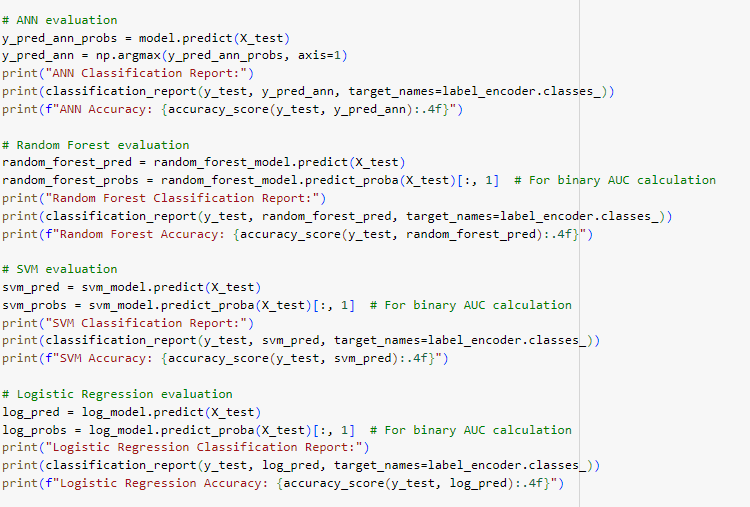


Figure Model Evaluation

A screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure Model Evaluation Results

A screen shot of a graph

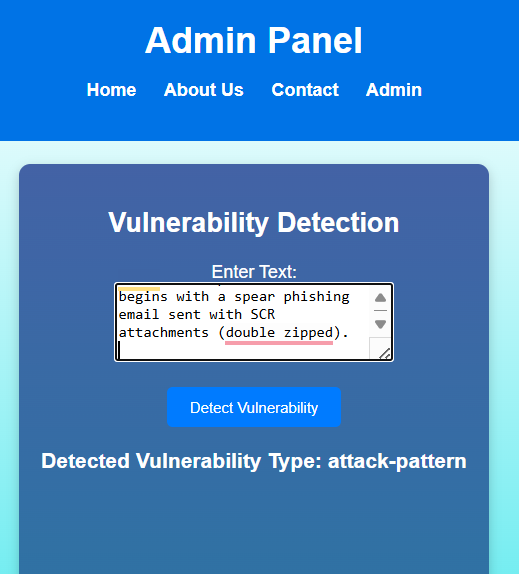
Description automatically generated

Figure Comparison of Performance Metrics

1. **Comprehensive Evaluation of Models:**
   * Assess Random Forest, ANN, SVM, and Logistic Regression using accuracy, precision, recall, F1-score, and AUC (Figures 9-11).
   * **Steps in Evaluation:**
   * Created detailed categorisation reports for each model, exhibiting performance across classes.
   * Used AUC values to evaluate each model's binary classification performance.
   * Analysed accuracy, recall, and F1-scores to find models that effectively handled imbalanced datasets for rare and frequent classes.
2. **Comparison of Model Performance:**
   * Comparing Model Performance: Bar charts simplify performance comparisons (Figure 11).
   * - While ANN models performed well, Random Forest was more interpretable and balanced.
   * Explained dataset complexity and class imbalance limits of Logistic Regression and SVM.
3. **Impact of Evaluation:**
   * The team chose the Random Forest model based on my evaluation, ensuring accuracy and efficiency.
4. **Challenges and Solutions:**
   * **Challenge:** Assessing unbalanced datasets fairly.
   * **Solution:** Prioritised minority class indicators like recall and F1-score over accuracy, which can be misleading in unbalanced situations.

**Testing and Deployment**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure Back End Testing

Testing and deployment were necessary to verify the app's real-world functioning and reliability.

1. **Functional Testing:**
   * End-to-end web page testing to ensure functionality (Figure 12).
   * **Verified the following:**
     + Users navigated to right pages without errors using navigation links.
     + Forms, including Admin forms, processed inputs and displayed proper validation signals.
     + Users could dynamically examine prediction results on the Admin page after entering data.
2. **Backend Testing:**
   * Examined Flask-machine learning model integration.
   * Simulated various input scenarios, including:
     + Simulations had valid inputs and reliable predictions for well-formed data.
     + Resilience assessment for handling unexpected or faulty inputs.
3. **Load Testing:**
   * Simulated heavy traffic with multiple people accessing the Admin page and submitting forms.
   * Made Flask installations more load-tolerant without compromising performance.
4. **Bug Identification and Resolution:**
   * Bug Identification and Resolution:
   * **Input validation:** Addressed input validation issues: Sanitised special characters and formats before processing.
   * **Backend errors:** Server faults are now handled by the backend, which shows user-friendly messages.
5. **Real-World Testing and Scalability:**
   * Real-World Testing and Scalability: Recorded application performance when deployed.
   * Scaled the application for user and data growth.
6. **Challenges and Solutions:**
   * **Challenge:** High-traffic scalability.
   * **Solution:** Stress testing and improving server settings like thread pooling and asynchronous request handling solved the problem.

**Technical Artefacts**

Key objects from my contributions helped the project succeed:

1. **HTML/CSS Templates:**

* Made modular, reusable templates for all web pages.
* Features include:
  + Modern, clean designs for enhanced user engagement.
  + Employed responsive layouts for different screen sizes and devices.
* **Impact:** These templates made the software look professional, enhancing user trust and satisfaction.

1. **Evaluation Reports:**

* Model classification reports with precision, recall, F1-score, and accuracy.
* Bar charts were used to compare model performance and choose the best algorithm.
* Data-driven model selection was enabled by reports.

1. **Testing Framework:**

* Created a comprehensive functional testing framework to verify application component usability and accuracy.
* Conducted backend testing to ensure smooth integration of machine learning model and Flask application.
* Load testing assessed system scalability and robustness in high-traffic areas.
* **Impact:** This framework made the app reliable, scalable, and deployable.

**Challenges and Impact**

I overcame project challenges:

1. **Balancing Aesthetics and Functionality:**

* **Challenge:** Creating a beautiful frontend without sacrificing performance or usability.
* **Solution:** The solution balances aesthetics and speed with lightweight CSS and optimal assets.

1. **Handling Edge-Case Inputs:**

* **Challenge:** Handling edge-case inputs requires the application to handle unexpected or incorrect user inputs.
* **Solution:** Improved input validation and preparation for data sanitisation.

1. **Scalability:**

* **Challenge:** Managing concurrent users without performance degradation.
* **Solution:** Optimising Flask configurations and testing in high-traffic scenarios solved the problem.

**Impact of Contributions**

* My work resulted in a polished, user-friendly, and stable app: • Improved User Interface: **Engaging User Interface:** Responsive web pages enhanced program accessibility and intuitiveness.
* **Data-Driven Insights:** Evaluation reports provided relevant model performance insights to help the team choose a deployment choice.
* **Robust Application:** The thorough testing infrastructure made the robust application stable and scalable, meeting all project goals.