Assignment 3

Start Assignment

- Due 29 Mar by 20:00
- Points 45
- Submitting a file upload
- Available 1 Feb at 0:00 - 2 Apr at 20:00

Assignment Name	Full-Stack Web Development for AI Application in Cybersecurity Scenarios
Assignment Description	In this final assignment, students will develop a full-stack web application that integrates the AI model created in Assignment 2. Using React.js for the front end and FastAPI for the back end, students will build an application that allows users to input data, processes the input using the AI model on the server side, and displays the results through interactive visualizations on the front-end. As the culminating assignment of the course, this project carries the largest weight in the final grade. It will test students’ ability to integrate front-end and back-end technologies, handle data efficiently, and deliver clear visualizations of the AI model’s output. In this assignment, students are expected to demonstrate proficiency in the following areas:: • **Full-Stack Web Development**: Implement a complete web application using React.js for the user interface and FastAPI for server-side processing. The front end must capture user inputs, and the back end must handle the AI model’s processing and return the prediction results. • **AI Model Integration**: Connect the AI model to the FastAPI back-end, enabling real-time predictions based on user input data. This will involve passing data between the front-end and back-end and efficiently processing it with the AI model. • **Data Visualization**: Use React.js along with visualization libraries like D3.js, Chart.js, or Plotly.js to present the AI model’s predictions. Ensure the visualizations are dynamic and provide clear, meaningful insights into the model’s output. At least two types of charts must be included. • **Error Handling and Input Validation**: Ensure robust error handling and input validation in both the front-end and back-end. The front end should validate user inputs before sending them to the server, and the back end must handle any errors gracefully, providing appropriate feedback to the user. Depending on the AI model’s topic, students’ applications might focus on one of the following areas: 1. **Misinformation Detection on Social Media**: Detect misinformation on social media platforms. Students will collect real-world social media data, extract useful features (like text content and user behaviour), and train models to identify false or misleading posts. The goal is to help improve online safety and address a key cybersecurity challenge in today's digital world. 2. **Spam or Malware Detection**: Detect spam messages or potential malware threats. Students will work with datasets containing emails, messages, or files, extract relevant

	features (such as text patterns or file properties), and build models to classify harmful content. The project aims to help protect users from digital threats and improve cybersecurity in communication systems. 3. **Detecting Software Vulnerabilities**: Detect security vulnerabilities in software code. Students will analyse source code to identify patterns that may lead to bugs or exploits, and train models to automatically flag risky code. The goal is to help developers find and fix vulnerabilities early, improving the overall security of software systems. 4. **Network Traffic Classification for Anomaly Detection**: Classify network traffic and detect unusual or potentially malicious activity. Students will analyse network data, extract features such as packet size and timing, and train models to distinguish normal behaviour from anomalies. The goal is to support early detection of cyberattacks and improve network security. 5. **Cybersecurity threat detection in banking**: This covers a range of cybersecurity problems in the banking sector. Students are expected to do some research to choose a suitable cybersecurity problem (e.g. fraud detection, identify theft, etc.) to solve using AI engineering. The goal is to improve the system security in this important industry sector.
Weight	45% of your total marks for the unit
Due Date	Sunday, 29/03/2026, 11:59 pm
Submission	1. **Report**: • **Size**: No more than 8 pages (excluding cover page, table of contents, bibliography, and appendix) • **Format**: PDF 2. **Source Code**: • **Format**: ZIP file. • **Contents**: ○ Front-end React.js code (excluding node_modules) ○ Back-end FastAPI code (must include the AI model code used in the web application) ○ README file (Markdown or PDF format) 3. **Video Demonstration**: • **Format**: MP4 or AVI (maximum 7 minutes). 4. **Meeting Minutes**: • **Format**: PDF. 5. **Contribution Form**: • **Format**: PDF.
Late Penalties	10% deduction of the available mark per calendar day or part thereof for up to one week. Submissions more than 7 calendar days after the due date will receive a mark of zero (0) and no assessment feedback will be provided.

1. Project Description

The project description provides an overview of the background and essential requirements for the cybersecurity-related projects.

This project involves developing a comprehensive machine-learning solution that integrates project management, design elements, and technical implementation to address real-world challenges. Students will work in teams to complete the project in three phases: creating a detailed project management plan, implementing a machine learning model, and developing a dynamic website to showcase their results.

The Machine Learning Web Application aims to deliver an interactive platform for users to engage with machine learning models and visualize data insights. The primary goal is to demonstrate practical machine learning applications in real-world scenarios, enhancing user interaction and understanding of the underlying models.

Core Functional Requirements for the whole project:

- Select a project topic and find relevant datasets suitable for the chosen topic.
- Investigate and analyze the topic, using machine learning techniques for prediction, attribution, or classification to gain a better understanding and response to the topic.
- Develop a website application that allows users to interact with trained models and provides corresponding responses based on user inputs.
- Include data visualization features to help users understand the dataset.

(Note that we have highlighted any information related to Assignment 3.)

2. Overview of the tasks

1. Front-End Development (React.js):

- Create a user input form with appropriate validation mechanisms.
- Implement a**t least two** types of data visualization charts using D3.js, Chart.js, or Plotly.js (three charts for HD-level work).
- Design a user-friendly, responsive interface.
- Ensure seamless interaction with the back-end API for data submission and retrieval.
- Apply UI/UX principles to enhance usability and aesthetics.

2. Back-End Development (FastAPI):

- Set up a FastAPI server and define necessary routes (using **at least two** HTTP methods: GET, PUT, POST, or DELETE).
- Integrate the AI model from Assignment 2 for prediction processing.
- Develop API endpoints for handling user data and returning model predictions.
- Implement robust error handling and exception management for stable operation.

3. AI Model Integration:

- Ensure efficient execution of the AI model on the server.
- Implement data preprocessing and postprocessing logic to handle inputs and outputs.

4. Documentation:

- Write a comprehensive README file with instructions for setup and running the application.

- Prepare a comprehensive project report that covers all aspects of the system, including the system architecture, implementation details, introduction, and API documentation.
  - The implementation details should provide a clear explanation of how the front-end and back-end components are developed, how the AI model is integrated, and any key design decisions made. Include descriptions of data preprocessing, postprocessing, error handling, and validation techniques.

Note: To achieve a High Distinction (HD), students must include additional features such as at least three types of charts, interactive visualizations, and at least three advanced functionalities based on their AI model outcomes. Examples include:

- **Enhanced Visualizations**: Implement at least three chart types with interactive features (e.g., zoom, filtering) using D3.js or Plotly.js.
- **Real-Time Updates**: Integrate live data feeds or enable auto-refreshing charts as new predictions are generated.
- **Custom User Input**: Provide advanced input forms with comprehensive validation and dynamic user feedback.
- **Error Handling**: Implement robust error handling with clear, informative messages on both the front-end and back-end.
- **UX Enhancements**: Include features such as export functionality for predictions and visualizations, but not limited to that.

3 Deliverables

Your submission should contain the following:

1. Project Report (PDF):

- **Project Title**: Title of your full-stack web application project (include on the cover page).
- **Group Information**: Group name and ID, student names and IDs, tutor's name (include on the cover page).
- **System Architecture**: Describe the overall architecture, including front-end, back-end, and AI model components.
- **Front-End Implementation**:
  - Briefly describe key features like the user input form and data visualization with validation.
  - Explain how data is sent to the back-end and predictions are retrieved using HTTP requests (e.g., Axios).
- **Back-End Implementation**:
  - Explain the FastAPI server setup.
  - Provide API endpoint documentation (including request type, JSON response, etc.).
- **AI Model Integration**:
  - Briefly recap the AI model from Assignment 2.
  - Explain how the model is integrated into the web application and describe any modifications for web deployment.
- **Conclusion**:
  - Explain how the solution addressed the chosen real-world issue and provided practical insights to users.
  - Identify areas for improvement and outline future enhancements like expanded functionality.
- **Bibliography**: List references in Harvard style.
- **Appendix**: Include additional files or information if necessary.

2. Source Code (ZIP file):

- **Front-End**: React.js code (exclude node_modules).

- **Back-End**: Back-end FastAPI code (include the AI model code used in the web application with any modification for integration)
- **README File**: Provide setup instructions, necessary libraries, running instructions for both front-end and back-end, and configuration steps for AI model integration.

3. Demonstration Video (Maximum 7 minutes):

- **Content**:
  - Showcase the FULL functionality of the web application.
  - Demonstrate **every aspect** of the website, including **all user interactions** (e.g., clickable buttons, input fields).
  - Ensure that **every part** of the application is shown, covering all features and components.
  - Show responsiveness on **desktop, tablet**, and **mobile** devices.
  - Walk through the **entire** workflow, from user input to data processing and result visualization.
  - Explain key technical decisions and challenges.
- **Format**: MP4 or AVI.
- **YouTube Option**: If the file size exceeds the Canvas limit, upload the video as an unlisted YouTube video and provide the link in your submission. Ensure the link remains active until grading is complete.
  - If submitting a YouTube link, please include it on the cover page of your project report and also in a separate text file named group-name-video-link.txt.
- **Important**: If the video is inaccessible at the time of grading, you will receive **zero marks for most components of this assignment**, as the video serves as the primary method for presenting and demonstrating your work. Ensure that the video is properly uploaded and accessible.

4. Meeting Minutes (PDF):

- Students are required to hold at least one meeting each week since the team's inception and submit all meeting minutes, along with other deliverables, as part of the assignment. You can use the attached meeting minutes as they are or as a reference to create your own. **Meeting Minutes Example.docx (https://swinburne.instructure.com/courses/71633/files/40836565?wrap=1)**

5. Contribution Form (PDF):

- A form includes sections for the personal information of each team member, details of the contribution, and other additional information. You can download the form **Group Assessment Contribution Form.docx (https://swinburne.instructure.com/courses/71633/files/40836872?wrap=1) (https://swinburne.instructure.com/courses/71633/files/40836872/download?download_frd=1)**

4. Submission

You must submit your assignment via the “Assignment 3 Submission” link on the Canvas site by the deadline (Sunday 11:59 pm, 09th November 2025, Week 13). No hard copy submission is required. Only one student per group needs to submit all the required documents.

File Naming: All files should be named using your group name, e.g., “session-xx-group-1-AssignmentName.pdf”. Marks may be deducted if naming and file requirements are not strictly followed. Submissions via email will not be accepted.

Detailed Submission Requirements:

1. Project Report:

- **Length**: No more than 8 pages (excluding cover page, table of contents, bibliography, and appendix).
- **Format**: PDF.
- **Note**: If the API documentation is too long to fit, include the most important APIs. Document at least 4 APIs in proper format, and mentioning that not all APIs are shown due to page limitations.

2. Source Code (ZIP File):

- **Structure**: Both front-end (React.js) and back-end (FastAPI) code, organized in separate folders.
- **AI Model**: The back-end folder must include the packaged AI model used in the application.
- **README File:** Markdown (.md) or PDF (.pdf) format.

3. Demonstration Video:

- **Length**: Maximum 7 minutes.
- **Format**: MP4 or AVI.
- **YouTube Option**: If the file size exceeds Canvas limits, upload the video as an unlisted YouTube video and provide the link. Ensure the link remains active until grading is complete.

4. Meeting Minutes:

- **Format**: PDF

5. Contribution Form:

- **Format**: PDF

5. Marking Criteria

You must acknowledge all statements and information taken from other sources and adhere to the guidelines published regarding plagiarism. All ideas and material taken from references must be cited within the report itself and a full reference list and bibliography (if appropriate) must be provided at the end of the report. Diagrams and/or tables may be used if you think this will strengthen your arguments. Remember that diagrams and tables adapted from other sources must be cited (***Harvard*** style) as well.

- **Note**: If the video is inaccessible at the time of grading, you will receive **zero marks for most components of this assignment**, as the video serves as the primary method for presenting and demonstrating your work. Ensure that the video is properly uploaded and accessible.

Important Notes:

- Do not publicly post any part of your work or reasoning on Canvas discussions to avoid violating plagiarism/collusion rules.
- Generative AI tools must not be used to generate content for this assignment.
  - However, their use for learning, idea generation, or debugging support is permitted. You are responsible for understanding, writing, and being able to explain all core logic in your application. Copying entire functions or components without full comprehension constitutes plagiarism. All use of generative AI must be acknowledged and documented in your report.
- Late video submissions or modifications after the deadline will result in penalties.
- Ensure the video clearly demonstrates all aspects of your application for grading.

COS30049 Assignment 3 Rubric

Criteria	Ratings			Pts
Report:1) Formatting & Referencing	**1 Pts** - Clear and professional project title, group name, student names on the cover page. Well-organized formatting with consistent layout, font, and spacing throughout. All references are properly cited and listed in Harvard style.	**0.5 Pts** - Project title and group information are provided, but there are several formatting issues or inconsistencies. References are cited but with significant errors in Harvard style.	**0 Pts** - Missing or incomplete project title, group information, and poor formatting. References are missing or not in Harvard style.	1 pts
Report:2)System Architecture & Implementation (Front-End, Back-End, API Integration)	**2 Pts** - Provides a detailed and clear explanation of the system architecture, including front-end, back-end, and AI integretion. The implementation details are thoroughly described, with justifications for technology choices. The explanation includes clear diagrams or flowcharts illustrating the system structure.	**1 Pts** - Presents a basic description of the system architecture. Some key components may be missing or poorly explained. Implementation details are vague, and technology choices are not justified. Diagrams or flowcharts are basic or may be missing.	**0 Pts** - The system architecture is poorly described or missing entirely. Implementation details are vague or incorrect. There's little to no explanation of technology choices, and no supporting diagrams or flowcharts are provided.	2 pts
Report:3)API Endpoint Documentation	**2 Pts** - All API endpoints are clearly and thoroughly documented, including request types (GET, POST, etc.), parameters, JSON request/response structures, and examples. Documentation is well-organized and easy to follow.	**1 Pts** - Some API endpoints are documented, but key details like parameters, request/response structure, or examples are unclear or missing. Documentation lacks organization.	**0 Pts** - API documentation is missing, incomplete, or contains critical errors, making it difficult to understand the API usage.	2 pts
Source code execution 1) Execution	**2 Pts** - The code executes flawlessly without any errors or warnings. All features of the web application run smoothly, including frontend, backend, and AI model integration. The application starts up quickly and responds to all user interactions without any	**1 Pts** - The code executes, but with several non-critical errors or warnings. Some features of the web application may not work as expected, but the core functionality is still operational. There might be noticeable delays in application startup or response times. Some	**0 Pts** - The code has significant execution problems. There are multiple critical errors that prevent core features from working correctly. The application may fail to start, crash frequently, or have major functionality issues. Serious dependency conflicts or missing	2 pts

Criteria	Ratings			Pts
	issues. Dependencies are correctly managed and there are no conflicts.	dependency issues may be present, requiring minor troubleshooting.	dependencies may be present, requiring substantial effort to resolve.	
Source code execution 2) Comments: Proper code comments inside the file	**1 Pts** - The source code is thoroughly commented on. All significant sections of the code have clear and informative comments that explain the purpose and functionality of the code. The comments enhance the readability and maintainability of the code, making it easy for others to understand and modify.	**0.5 Pts** - Comments are present but often brief or vague, leaving some sections unclear. Key functionality is not always explained, making the code harder to understand and modify.	**0 Pts** - The code has minimal comments, with many sections lacking explanations. The comments provided may be too brief, vague, or inconsistent, leading to difficulties in understanding the code's functionality.	1 pts
Source code execution 3) Code Structure: Clean and well structured source code and program files	**2 Pts** - The source code is clean, well-organized, and follows best practices for code structure. The program files are logically organized, with a clear hierarchy, and are easy to navigate. The code is modular, with functions or classes clearly separated, making it maintainable and scalable.	**1 Pts** - The source code is generally organized but may have some issues in structure or organization. The program files are mostly well-structured but might have some redundancies or lack clear separation between different functions or modules.	**0 Pts** - The source code is poorly structured, with a disorganized layout and lack of clear organization. The program files may be difficult to navigate, with unclear or inconsistent structuring, making the code hard to maintain or understand.	2 pts
Functionality 1) AI Model Integration	**3 Pts** - AI model is seamlessly integrated, provides real time predictions, and handles various input types effectively.	**1.5 Pts** - The AI model is integrated, but many issues exist with data flow or prediction. The integration functions correctly but may have occasional performance or latency issues.	**0 Pts** - AI model integration is non-functional or missing.	3 pts
Functionality 2) Front-end Communication with backend	**4 Pts** - The communication between the front-end and back-end is smooth and error-free. Data is transferred efficiently using proper HTTP methods (e.g., GET, POST), and the application responds	**2 Pts** - Communication between the front-end and back-end works, but there are occasional delays or significant errors. Data transfer functions, but some interactions may need	**0 Pts** - Communication between the front-end and back-end is non-functional or severely broken. Data is not transferred correctly, and critical features relying on data exchange do not work.	4 pts

Criteria	Ratings			Pts
	promptly. All interactions are fully functional with no issues in data retrieval or submission.	improvement in speed or reliability.		
Functionality 3) Error Handling and Input Validation	**3 Pts** - Comprehensive error handling and input validation are implemented across both the front-end and back-end. All input fields are validated, and errors are handled gracefully with clear, informative messages provided to users. The system prevents invalid data from being processed.	**1.5 Pts** - Basic error handling and input validation are present, but many edge cases may not be covered. Error messages are displayed, but they may not be clear. Some invalid inputs can pass through the system.	**0 Pts** - Minimal or no error handling and input validation. Many inputs are not validated, and the system allows invalid data to be processed. Errors are either not caught or handled poorly, with no informative feedback for users.	3 pts
Functionality 4) Backend API Design and Implementation	**4 Pts** - The API is well-designed, following RESTful principles with clear, efficient endpoints. At least two HTTP methods are used appropriately (e.g., GET, POST, PUT, DELETE), and the API efficiently handles data requests and responses. The API is well-documented and performs reliably.	**2 Pts** - The API is functional but has noticeable design issues or inefficiencies. Some HTTP methods may be used incorrectly, or the endpoints may not fully adhere to RESTful principles. Documentation is minimal or lacking in detail.	**0 Pts** - The API is poorly designed, with incorrect or missing endpoints. It fails to follow RESTful principles and does not perform as required. Documentation is incomplete or missing.	4 pts
Functionality 5) Real-time Updates and Responsiveness	**4 Pts** - The application provides smooth, real-time updates with immediate responsiveness to user inputs and changes. All interactive elements and data visualizations update dynamically without delay, maintaining performance across different devices.	**2 Pts** - Real-time updates and responsiveness are functional but inconsistent. Delays are noticeable, and some elements may not respond or update smoothly. Performance issues are evident, particularly on different devices.	**0 Pts** - The application lacks real-time updates and is unresponsive. User actions result in significant delays, crashes, or elements that do not update at all. The feature is essentially non-functional or severely broken.	4 pts
Visualization 1) Chart Diversity and Relevance	**3 Pts** - The application includes multiple (at least three)	**1.5 Pts** - The application includes some variety of charts, but	**0 Pts** - The application lacks diversity in chart types, or	3 pts

Criteria	Ratings			Pts
	diverse chart types that are highly relevant to the data being presented. The charts effectively communicate the AI model's predictions and insights in a clear and meaningful way.	they may lack diversity or relevance to the data being presented. The charts communicate the data, but not as effectively or clearly as they could.	the charts are irrelevant or poorly suited to the data. They fail to effectively communicate the AI model's predictions or insights.	
Visualization 2) Interactivity of Visualizations	**4 Pts** - The visualizations are highly interactive, allowing users to engage with data through features like zoom, filtering, tooltips, or real-time updates. The interactions are smooth and enhance user experience.	**2 Pts** - The visualizations are visually appealing but offer limited interactivity. While not fully interactive, the charts are well-presented and visually clear, still providing a good user experience.	**0 Pts** - The visualizations lack interactivity and offer no user engagement options such as zoom, filtering, or tooltips. The charts are static, visually unappealing, and significantly reduce the overall effectiveness and user experience.	4 pts
Visualization 3) Data Representation Clarity	**2 Pts** - The data is represented clearly and accurately, with visualizations making the AI model's predictions easy to understand. The charts effectively highlight key insights and are well-labeled with appropriate legends, titles, and units.	**1 Pts** - The data is generally clear, but some aspects of the visualizations may be confusing or lacking in detail. Labels, titles, or legends may be present but not fully informative, making some insights harder to grasp.	**0 Pts** - The data representation is unclear, with confusing or poorly labeled visualizations. Charts are difficult to interpret, and key insights from the AI model's predictions are lost or misrepresented.	2 pts
Visualization 4) Visualization Performance and Loading	**2 Pts** - Visualizations load quickly and perform smoothly, even with large datasets.	**1 Pts** - Acceptable loading times and performance with occasional issues.	**0 Pts** - Slow loading times or poor performance of visualizations	2 pts
Usability 1) User Interface (UI)	**3 Pts** - The UI is highly polished with advanced design features such as responsive layouts, intuitive navigation, and visually appealing elements. Advanced UI components (e.g., modals, dynamic forms, or custom animations) enhance the overall user experience.	**1.5 Pts** - The UI is functional and visually satisfactory but lacks advanced design features. Navigation is intuitive, but some areas may be plain or not as visually refined. The design is consistent, but basic.	**0 Pts** - The UI is poorly designed or non-functional. It is visually unappealing, difficult to navigate, or lacks consistency. The design does not contribute positively to the user experience.	3 pts

Criteria	Ratings			Pts
Usability 2) User Experience (UX)	**3 Pts** **-** The UX is highly refined, offering a seamless and intuitive experience. Advanced features such as user feedback mechanisms, personalized interactions, or smooth transitions enhance the overall user engagement and make the web application easy to use.	**1.5 Pts** **-** The UX is generally good, with most user interactions being straightforward and functional. However, the experience may feel basic or lack features that enhance engagement or usability beyond standard expectations.	**0 Pts** **-** The UX is poor, with confusing navigation, unintuitive interactions, and a lack of consideration for user needs. The web application is difficult to use and does not provide a positive experience.	3 pts
				Total points: 45