WEB PROGRAMMING & APPLICATIONS - 503073

SEMESTER 2 – ACADEMIC YEAR 2024 – 2025

Version 1.0, last updated on Dec 24, 2024 by **Mai Van Manh**

The essay (midterm) must be completed within 8 weeks. Some groups (5-8 groups) will be selected to present their report in class (registration is required at the beginning of the semester, and PowerPoint slides should be prepared for each member to present in weeks 9th and 10th of the theory classes).

There are no limits to the development of the demonstration website; the more you do and the better you do it, the more highly it will be evaluated. The instructor has not set specific constraints, allowing groups to freely express their ideas and apply the knowledge they have acquired in the demo. It is important to note that the demo should not be too simplistic. For example, illustrating real-time features with a chat website will always be rated higher than a basic page that simply updates the temperature of a city in real-time.

I. LIST OF TOPICS

ID	TOPC NAME	DESCRIPTION This is a suggestion, not a strict requirement. Completing all criteria doesn't guarantee a perfect score. Students are encouraged to complete it as thoroughly as possible
1	**Account Management**	Account management involves designing systems to securely handle user authentication, profile management, and recovery processes. Key Points for the Essay (you should explore more for a higher score): • Features: Registration, login, password recovery, two-factor authentication (2FA).

		• Best Practices: Password hashing (e.g., bcrypt), token-based email verification, and session security. • Security: Discuss 2FA implementation, such as Time-based OTP or authenticator apps. • Examples: Include cases of account compromise and how robust design can prevent such incidents. Some demo suggestions (you should explore more for a higher score): • Implement account management features (registration, login, password recovery). • Integrate 2FA and demonstrate enabling/disabling it. • Ensure security by using hashed passwords and one-time recovery links with timeouts.
2	**Front-End Frameworks: React with Material-UI/Ant Design**	React is a popular library for building interactive user interfaces. Coupled with Material-UI or Ant Design, it enables the development of elegant and responsive web applications. Key Points for the Essay (you should explore more for a higher score): • React Concepts: Components, state management, and props. • UI Libraries: Explore Material-UI and Ant Design for creating reusable and modern components. • Comparison: React vs. other frameworks like Angular or Vue.js. Some demo suggestions (you should explore more for a higher score): • Build a responsive web interface (e.g., a dashboard or e-commerce layout). • Use Material-UI or Ant Design for styling and interactivity.
3	**Restful Web Service**	REST (Representational State Transfer) is an architectural style for designing networked applications, leveraging standard HTTP methods for communication. Key Points for the Essay (you should explore more for a higher score): • REST Principles: Stateless communication, resource representation, and uniform interface. • HTTP Methods: Explain GET, POST, PUT, DELETE, and their roles in CRUD operations.

4		• Security: JWT (JSON Web Token), OAuth, and other authentication methods. • Comparison: Highlight the differences between RESTful APIs and other styles like GraphQL or SOAP. Some demo suggestions (you should explore more for a higher score): • Create a REST API with endpoints for managing a dataset (e.g., a library or inventory system). • Secure the API using JWT authentication. • Demonstrate how to consume the API with a front-end or a tool like Postman.
4	**AJAX: Loading Content Without Page Refresh**	AJAX (Asynchronous JavaScript and XML) allows web applications to send and retrieve data from a server asynchronously without refreshing the web page. This creates smoother and more interactive user experiences. Key Points for the Essay (you should explore more for a higher score): • Principles: Discuss how AJAX leverages JavaScript, XML/JSON, and HTTP requests to fetch data dynamically. Explain the roles of XMLHttpRequest and Fetch API. • Advantages: Reduced server load, faster user interactions, and improved user experience. • Challenges: Handling asynchronous code, debugging issues, and ensuring security. • Comparison: Analyze how AJAX differs from WebSockets and Server-Sent Events in use cases and implementation. Some demo suggestions (you should explore more for a higher score) (including but not limited to): • Build a web page that dynamically updates data (e.g., a news feed or live comment system). • Implement Fetch API for retrieving server data. • Create an interactive interface to demonstrate the seamless experience AJAX offers.
5	**Progressive Web Apps (PWAs)**	PWAs combine the capabilities of web and native apps, offering offline functionality, push notifications, and hardware access.

		Key Points for the Essay (you should explore more for a higher score): • Core Concepts: Explain Service Workers, Web App Manifest, and the App Shell model. • Benefits: Reduced development costs, offline access, and app-like user experiences. • Comparison: PWAs vs. native mobile apps, including performance and limitations. Some demo suggestions (you should explore more for a higher score): • Develop a PWA that includes offline functionality via a Service Worker. • Implement push notifications (e.g., updates or alerts). • Include hardware interaction, such as capturing photos using the device's camera.
6	**Common Website Attacks and Countermeasures**	Understanding security threats is crucial to safeguarding web applications. Key Points for the Essay (you should explore more for a higher score): • Attack Types: XSS, SQL injection, CSRF, and phishing. • Examples: Illustrate real-world scenarios where attacks caused damage. • Countermeasures: Input validation, escaping, and security headers. Some demo suggestions (you should explore more for a higher score): • Develop a secure web app with protections against XSS and SQL injection. • Simulate a controlled attack scenario to show vulnerabilities and fixes.
7	**WebAssembly (Wasm)**	WebAssembly is a binary instruction format that enables high-performance applications on web platforms by running compiled code from languages like C, C++, or Rust. Key Points for the Essay (you should explore more for a higher score): • Purpose: Discuss why WebAssembly is essential for computationally intensive tasks on the web.

		• Workflow: Explore how source code (e.g., C) is compiled to WebAssembly and integrated into JavaScript. • Use Cases: Games, video editing, and machine learning on the browser. • Comparison: Contrast WebAssembly with JavaScript in terms of performance and limitations. Some demo suggestions (you should explore more for a higher score): • Compile a C program (e.g., video processing with FFmpeg) into WebAssembly. • Create a web page where the program processes video files in real time.
8	**Machine Learning on the Web**	Browser-based machine learning enables the use of pre-trained models and in-browser computation for tasks like face detection and object recognition. Key Points for the Essay (you should explore more for a higher score): • Technologies: Focus on TensorFlow.js and other JavaScript libraries for machine learning. • Capabilities: Discuss how ML models can run directly in the browser without server dependencies. • Challenges: Resource limitations and security concerns when processing sensitive data. Some demo suggestions (you should explore more for a higher score): • Build a browser-based application for real-time face detection using TensorFlow.js. • Explain the architecture, including model loading and prediction logic.
9	**Methods for Publishing Websites During Development**	Exposing development projects to the internet is essential for testing and collaboration. Key Points for the Essay (you should explore more for a higher score): • Tools: LocalTunnel, Ngrok, and alternatives. • Networking: Explain port forwarding and firewalls. • Comparison: Highlight the pros and cons of each approach, including ease of use and security.

		Some demo suggestions (you should explore more for a higher score): • Set up a local web server and expose it using at least two tools. • Demonstrate how external collaborators can access the application.
10	**Deployment of Web Applications**	Deployment involves taking a web app from development to production, ensuring reliability and scalability. Key Points for the Essay (you should explore more for a higher score): • Steps: Server setup, database configuration, and domain/HTTPS integration. • HTTPS: Explain SSL certificates and how to configure them. • Hosting Options: Compare services like AWS, Heroku, and shared hosting. Some demo suggestions (you should explore more for a higher score): • Deploy a web application on a production server. • Configure HTTPS and integrate a database.
11	**Containers in Web Development and Deployment (Docker)**	Containers simplify application deployment by bundling software and dependencies into isolated environments. Key Points for the Essay (you should explore more for a higher score): • Concepts: Docker basics, container orchestration, and scalability. • Advantages: Portability, consistency across environments, and resource efficiency. Some demo suggestions (you should explore more for a higher score): • Containerize a web app and database. • Demonstrate deployment using Docker Compose or Kubernetes.

12	**Web Crawling**	Web crawling involves automated data retrieval from web pages, useful for data analysis and aggregation. Key Points for the Essay (you should explore more for a higher score): • How It Works: Crawling logic, parsing HTML, and handling robots.txt. • Ethics: Discuss the ethical implications and legal considerations. • Use Cases: Content aggregation, SEO, and research. Some demo suggestions (you should explore more for a higher score): • Build a web crawler to scrape content from a site (e.g., blogs, files). • Implement filters to extract and save specific data (e.g., titles, links, or images).
13	**Optimizing Image and Video Loading**	Explore modern techniques to improve performance and user experience when loading images and videos on the web. Key Points for the Essay (you should explore more for a higher score): • Lazy Loading: Load images/videos only as they enter the viewport using HTML attributes or Intersection Observer API. • Thumbnails and Progressive Loading: Serve low-resolution placeholders or use progressive formats like JPEG Progressive or AVIF for smoother image rendering. • Responsive Media: Use <picture> and modern formats like WebP for adaptive media delivery across devices. • HLS for Videos: Implement HTTP Live Streaming for adaptive bitrate video playback, improving performance on varied network conditions. • Skeleton Screens: Utilize placeholders to mimic interface elements while content is loading. • Additional Techniques: Incorporate CDNs for faster delivery and optimize media formats using efficient codecs (e.g., H.265, VP9).

		Some demo suggestions (you should explore more for a higher score): • Create a web page showcasing lazy loading and responsive media. • Implement progressive image loading and HLS video streaming. • Add a skeleton screen for an enhanced user experience while loading content.

II. GENERAL REQUIREMENTS

Research and Comprehension:

- Conduct a thorough survey of the assigned topic, including its history, core principles, technical specifications, and best practices. Utilize reliable sources such as documentation, research papers, tutorials, and official websites.
- Demonstrate a clear understanding of the topic by providing accurate and informative explanations in your essay. Include interesting facts, figures, and examples to enhance your writing.
- Analyze the strengths and weaknesses of the technology, comparing it to similar alternatives where applicable. Identify potential challenges and discuss solutions or mitigation strategies.

Demonstration Website:

- Develop a functional website that showcases your understanding of the assigned technology. Choose a project scope that allows you to implement key concepts and functionalities effectively.
- Within the essay, provide a detailed introduction to your demonstration website. Explain its purpose, target audience, and core features.
- Describe the architecture, components, and technologies used to develop your website. Explain how you implemented the assigned technology within your project, highlighting specific functionalities and techniques.
- Include screenshots or visuals whenever possible to illustrate your website's design and features. Briefly discuss the overall effectiveness of your implementation and any challenges you encountered.

Writing and Presentation:

- Write a well-structured essay that clearly presents your research and demonstrates your understanding of the topic. Utilize proper grammar, spelling, and academic citation format.

- Maintain a professional and engaging writing style, showcasing your ability to effectively communicate technical concepts to a diverse audience.
- Organize your essay logically, including:
  - Introduction: Briefly introduce the chosen topic and provide an overview of your essay's content.
  - Body: Discuss the various aspects of the topic comprehensively, integrating insights from your research and website development experience.
  - Conclusion: Summarize your key points and provide a final reflection on the topic's significance and potential future developments.
- Pay attention to presentation and formatting, ensuring your essay is visually appealing and easy to read. Utilize headings, subheadings, and clear formatting to guide readers through your content.

III. REQUIREMENTS FOR OUTPUT

1. **Complete Report (Word and PDF Format):** The report must be written according to the Faculty's template and must be professionally structured and free of grammatical or formatting errors. Ensure that you include appropriate references to external sources used in your research.
2. **Source Code and Libraries**
   - Complete source code: Provide all the source code and related files necessary to run your demo project. The code should be clean, well-documented, and adhere to coding best practices.
   - Dependency management: Include all necessary libraries or dependencies that are required to run the project. Make sure to specify these dependencies in a pubspec.yaml file.
   - Your submission must be organized into folders and include a README file (see 3.4) for clear instructions on how to set up and run the project.
3. **Product Presentation Video**
   - Duration: Maximum 20 minutes.
   - Content: This video must include a presentation of the theoretical aspects, demo project, and its architecture, as well as an overview of how your project works in practice.

- Each team member must participate in the presentation.
- You can use Google Meet, Zoom, or any other tool to record your presentation.
- The video should demonstrate the demo project, showing how it works, the technologies used, and explaining key parts of the code or architecture.
- Make sure that the video is clear, concise, and well-organized. The grader should easily understand what your group has done and how it reflects the chosen topic.

4. **Readme.txt** - Instructions for Use and Testing
   - Setup instructions: Provide a step-by-step guide on how to set up the environment and run your project. Make sure the grader can follow these instructions easily and run your demo on their computer.
   - Include any system requirements (e.g., OS, Flutter version, additional dependencies).
   - Provide necessary commands for installing dependencies, running the project, and testing specific features.
   - Ensure that all setup instructions have been thoroughly tested.
   - The README should be clear, well-structured, and easy to follow.

IV. DETAILED SCORING

CRITERIA	POINT	0	25%	50% - 75%	Full Score
REPORT	**6.0**				
Theoretical Understanding	2.0	No relevant theoretical content provided.	Superficial understanding, missing key concepts, lacks detail and depth.	Covers basic theoretical aspects but lacks clarity or detail in certain sections.	Thorough and in-depth explanation of all key theoretical aspects, demonstrating a strong understanding.

Comparative Analysis	1.0	No comparison of different approaches.	Comparison is mentioned but is vague or incorrect, with little insight into differences.	Comparison covers different approaches but lacks depth or is only partially correct.	Detailed and insightful comparison of different approaches, highlighting their strengths and weaknesses effectively.
Application and Implementation	1.0	No explanation of how concepts were applied in the demo.	Brief explanation of demo, lacks details on architecture or functionality.	Clear explanation of how concepts were applied in demo with architecture details.	Detailed application of concepts with clear explanation of demo structure, components, and results.
Report Structure and Presentation*	2.0	No report submitted, or contains critical errors such as a distorted university logo, incorrect colors, inaccurate information of university, faculty, instructor, team member, project title, or course details.	Report is basic, lacks structure, or has significant formatting or content issues.	Mostly complete but contains some formatting errors, lacks detail in certain sections, or has inconsistent clarity.	Well-organized and professionally formatted report, follows required structure, and provides clear explanations and logical flow throughout.
DEMO	**4.0**	This section won't be evaluated unless it includes a **demo video with sound** and comprehensive **source code with instructions**			
Scope and Detail of the Demo	2.0	No demo or completely non-functional.	Basic demo with some aspects missing or insufficiently detailed.	Most key aspects of the topic addressed with reasonable complexity.	Detailed and comprehensive demo, addressing all key aspects with full functionality.
Demonstration and Clarity	1.0	No demo or demo is unclear, missing explanations or visuals.	Demo somewhat clear but lacks sufficient explanation or visuals.	Clear demo with relevant explanations and visuals. Some parts could be more detailed.	Demo is clear and well-structured with detailed explanations and effective visuals.

Video and Presentation Skills	1.0	No demo presentation or explanation.	Demo is presented poorly with little clarity, missing important details.	Demo is explained but some parts are unclear, or there are missing elements.	Demo is presented clearly, with all parts explained thoroughly, demonstrating a solid understanding of the topic and how the demo was implemented.

* Refer to the attached file Report templates and guidelines 2024.zip for the regulations and essential guidelines to craft a well-written report. A well-written report should avoid the following mistakes (the list is not exhaustive):

- The cover page was presented carelessly, with serious errors such as: using a substandard logo, incorrect instructor information, incorrect group member information, incorrect topic name.
- Missing necessary appendix pages as required. Not having enough chapters (or contents) as required such as: theoretical survey, requirement analysis, system design, presentation of results with necessary analysis/comparison, conclusion chapter of the report.
- Spelling errors; inconsistent font, color, and font size; failure to use appropriate margins and line spacing; overuse of bullet points in presentation.
- The report has a lot of text but lacks images, diagrams, tables, and charts to illustrate and make the content easier for readers to understand. This also shows the possibility of abusing AI tools to write reports.
- Lack of investment in presenting images, tables, and diagrams: for example, images that are too large or too small, fonts/background colors that are difficult to read, screenshots of redundant content, leading to a poorly presented report.
- Too much mention of source code while lacking description or explanation of the system, the operating principles of the components, or how the team implemented and deployed the system, makes it difficult for readers to understand what the team is working on, how they did it, what the results were, or what good things can be learned from this topic.

Late Submission Penalty: For every day that an assignment is submitted late, one point will be deducted from the total possible score. Please note that even one second late will be considered as one full day late.

Teamworking Policy

To ensure effective collaboration and equitable distribution of work, the following guidelines apply to team-based projects:

- **Team Size**: Each team must consist of 2-3 members.
- **Single-Member Teams:** Teams with only one member will receive **a 0.5-point deduction**. This policy encourages teamwork and reflects the expectations for group collaboration..
- **Exception:** If a team member drops out of the course or is unable to continue working on the project due to extenuating circumstances, the remaining member(s) should notify the teacher immediately. In such cases, the point deduction and score cap may be waived at the instructor's discretion, depending on the situation.

Points may also be deducted (0.5 to 1.0 points per instance) in the following cases:

- The report or video presentation is not in English.
- Not all team members are present in the video presentation, unless previously approved.
- The demo video has poor audio or video quality, making it difficult to understand or follow.
- There is an uneven distribution of work among team members.
- The submission does not adhere to the specified format or is missing required information, such as:
    - Missing or incomplete team member details.
    - Insufficient descriptive information, such as failing to include instructions for running the source code or not providing the necessary admin account credentials to access the application.