Advanced Web Application Development Assignment

Objective:

- Deepen understanding of Laravel, MVC, user authentication, and authorization.
- Develop hands-on experience by building a secure web application.
- Demonstrate theoretical knowledge application in a practical scenario.

Course Outcomes

In this assessment, you are assessed based on the following **course outcomes**:

CO1	Build interactive and database-driven web applications using a web application frame- work and model-view-controller (MVC) software architecture.
CO2	Build web applications with user authentication and authorization using cookies, session and role-based access control (RBAC).

Assessment Contribution

This assignment contributes **20%** to the overall assessment for this course.

Membership

This is a **group** assignment: maximum **Four (4)** students per group.

Deadline

The deadline for this assignment is **Week 12**.

Part 1 (40%)

Instructions:

For this part, you will investigate and report on some issues related web development frameworks, software design patterns, focusing on the following areas:

1. **Web Development Frameworks**
   - **Tasks:**

     (i) Compare **FIVE (5)** popular web development frameworks (e.g., Laravel, Django, Ruby on Rails, Spring Boot, Express.js). For projecting a summary, build a table and extract suitable criteria including at least the following to highlight the contrasts.

     - Advantages and disadvantages.
     - Suitability for different project types.
     - Community support and ecosystem.
     - Development experience.

     (ii) Discuss the key capabilities of a modern web application. Argue if AI can play a role in developing web applications and how the future of web applications with the introduction of generative AI will be.

     (iii)Explain your preference and describe your personal perspective with respect to a particular web development framework. For this task, you need to do analysis based on logical statements and use facts to support your opinion.

2. **Software Design Patterns**
   - **Tasks:**

     (i) Describe **FIVE (5)** software design patterns including the key characteristics and sample open-source projects.

(ii) Discuss benefits and drawbacks of using MVC architecture and how the MVC architecture separates and organizes concerns for clean, maintainable code.

(iii) Explain practical scenarios where different design patterns might be preferable.

Writing guidelines:

- Ensure the report adheres to the required format and appropriate academic writing styled. Use font of Times New Roman, 12 font-sized. 1.5 spacing. Criteria of evaluation include consistent format and style used, grammar, spelling, correct use of punctuation, appropriate sectioning, sources cited using consistent citation format, and all the components required in the assignment are discussed.
- You must appropriately reference all the sources of information that you use in your report. Harvard referencing format should be used. Also, do remember to include in-text citation in your report.
- Incorporate visuals (e.g., diagrams, charts) to enhance understanding where appropriate.

Part 2 (60%)

Instructions:

Search and select an open-source **PHP** project repository. Add/modify at least the following features for the selected project based on the **Laravel 8/9** framework, implement the components and develop the web application.

A: MVC

1. **View/Controller/Model**
    - **Tasks:**
        - Create well-designed views for the project interfaces.
        - Create controllers for organizing the application logic.
        - Create models to handle database operations.
2. **Database migration**
    - **Tasks:**
        - Create appropriate migrations for the application database.

B: RBAC

3. **User Authentication**
    - **Tasks:**
        - Enable user registration and login using email/password authentication with at least **TWO (2)** user categories (e.g. admin, user, …).
        - Implement secure password hashing and storage (e.g., bcrypt).
        - Handle session management and user verification upon login.
4. **User Roles and Permissions (authorization)**
    - **Tasks:**

        - Define at least **TWO (2)** roles (e.g., author, editor, administrator) with appropriate permissions.
        - Restrict access to create, edit, and delete blog posts based on user roles.
        - Employ a suitable approach for role-based access control (e.g., Laravel gates or policies, middleware).

Writing guidelines:

- This part contains:
    - **Title** and **description** of your web application, explaining the open source project, its characteristics number of users ,….
    - Sample screens:
        - Code listing of all database migrations
        - Code listing of your route file
        - Code listing of all controller classes
        - Code listing of all model classes.
        - Code listing of all authentication validation.
        - Code listing of all gates and/or policies.
        - Code listing of all cookies/session implementations.

The final report submission:

The final report should be only a **single PDF** file **comprising parts 1 and 2**. Each member needs to have his/her work recognized in the report, e.g. in a table at the beginning of the report to specify the tasks done by him/her. The file should be named using your group number (e.g. UECS3294_Assign_GroupNumber.pdf)

Assessment Criteria

Part 1 (40%):

COs	Criteria	Poor (0)	Needs Improvement (1-20)	Satisfactory (21-60)	Good (61-80)	Excellent (81-100)
CO1	Analogy of Web Frameworks (15 marks)	No comparison or irrelevant information.	Inaccurate or incomplete comparison, lacks understanding of strengths/weaknesses.	Basic comparison, limited strengths/weaknesses, no personal perspective.	Clear comparison, reasonable strengths/weaknesses, some personal opinion.	Comprehensive comparison of frameworks, clear strengths/weaknesses, insightful personal perspective.
CO1	Software Design Patterns (15 marks)	No discription of software design patterns or irrelevant information.	Misunderstanding of patterns, inaccurate benefits/drawbacks, or irrelevant information.	Basic explanation of patterns, some understanding of benefits/drawbacks, no discussion of alternatives.	Good explanation of patterns, reasonable benefits/drawbacks, limited discussion of alternatives.	In-depth explanation of patterns, clear benefits/drawbacks, discussion of alternatives.
CO1	Quality of report writing (5 marks)	Difficult to understand, disorganized, significant errors, inaccurate or irrelevant information.	Unclear, disorganized, numerous errors, incorrect technical terms.	Organized, some errors, inaccuracies in technical terms.	Clear and concise, few errors, mostly accurate technical terms.	Well-organized document, free of errors, uses technical terms correctly.
CO1	Visuals and Citations (5 marks)	Irrelevant or misleading visuals, incorrect or missing citations.	No visuals or citations.	Few or irrelevant visuals, inconsistent or missing citations.	Some visuals or citations used, inconsistent or unclear.	Relevant and informative visuals enhance understanding, consistent citation style.

Part 2 (60%):

	COs	Criteria	Poor (0)	Needs Improvement (1-20)	Satisfactory (21-60)	Good (61-80)	Excellent (81-100)
A	CO1	View/Controller /Model (20 marks)	No or non-View/Controller /Model.	Incomplete view/controller/model or missing key features.	Basic view/controller/model, limited organization.	Well-defined view/controller/model, following some best practices.	Well-defined view/controller/model, following best practices.
	CO1	Database migration (5 marks)	No database migration.	Incomplete database migration or missing key features.	Basic database migration, limited organization.	Well-defined database migration, following some best practices.	Well-defined database migration, following best practices.
B	CO2	User Authentication (10 marks)	No authentication.	Missing authentication features or insecure authentication mechanisms.	Limited authentication, insecure authentication, or unclear implementation.	Multiple user types, basic authentication, some authentication implementation.	Distinct user types, appropriate authentication, secure authentication mechanism.
	CO2	User Roles and Permissions (15 marks)	No roles or permissions.	Missing role-based features or insecure access control mechanisms.	Limited roles or permissions, insecure access control, or unclear implementation.	Multiple roles, basic permissions, some access control implementation.	Distinct roles, appropriate permissions, secure access control using policies/middleware.
	CO2	Code Quality and Documentation (10 marks)	Unusable code or no documentation.	Poorly written code, difficult to understand, missing or inaccurate documentation.	Unorganized code, limited readability, incomplete documentation.	Mostly readable code, some organization issues, basic documentation information with presentation of app screens.	Clean, well-organized code, clear documentation with presentation of app screens.