Project Title: CDP Assignment: Laravel

Overview:

Build a Beautiful CRUD App with the help of APIs setup focuses on creating a secure and functional web application.

Objectives:

- Showcase Advanced Framework Expertise
- Develop and Implement Authorization and Access Control
- Create and Implement Authentication API
- Demonstrate how secure you can develop.
- Build CRUD API for one table/Entity
- Design and Implement a CRUD Application with Data Visualization
- Develop and Implement User Interface (UI) with Forms and Validation
- Write Comprehensive Unit Test Cases
- Apply Best Practices in Code Quality, Performance, and Security

Tools & Technologies:

- Latest >= Laravel 10
- Python + MySQL
- POSTMAN
- JavaScript, jQuery, etc.
- CSS
- XHTML5
- Git for version control
- Composer

Assignment Breakdown:

1. Application Environment Setup and Design (16 hours)

- Raise IT ticket and obtain manager approval (optional).
- Configure and set up the working environment after approval.
- Install framework and set up the database.

2. DB setup and schemas design (2 Hours)

- Design secure user table schemas.

Author: Rajesh Ganjeer
(Sr. Team Lead)

- Determine table names and columns for CRUD operations (e.g., category, product).
- Utilize framework tools for schema generation and relationship mapping.
- Insert sufficient seed data for UI presentation.

3. Authentication (2 hours)

- Implement a secure user login API using methods like JWT, OAuth, or Bearer Token.
- Restrict access to protected resources to authorized users only.
- Handle login failures and credential verification.
- Generate and manage expiring tokens (e.g., UUID-based or secure tokens) after successful authentication.

5. CRUD pages and Data Visualization (8 hours)

- Create RESTful, secure web APIs with proper authentication.
- Follow API URL naming conventions and ensure input/output consistency.
- Develop CRUD pages for users and other tables.
- Encrypt sensitive data, like passwords, in the database.
- Use CSRF tokens for create/update forms.
- Prevent SQL injection.
- Implement bearer tokens or JWT for authentication.
- Confirm record deletions.
- Add filters and pagination.
- Ensure proper exception handling and error messaging.
- Use status codes for responses.
- Optimize data fetching and rendering for performance.
- Apply OOP principles for clean, reusable code.

6. Performance Optimization (2 hours)

- Implement and explain performance optimizations.
- Use memorization techniques where relevant.

7. Testing, Quality Assurance and Final Touches (4 hours)

- Write unit tests for all APIs using PHP Unit.
- Perform integration testing to ensure components work together as expected.
- Review and refactor code to ensure it meets high-quality standards.
- Refactor the codebase to ensure consistency and readability.

Author: Rajesh Ganjeer
(Sr. Team Lead)

8. Deliverables:

- Source Code: Complete assignment code in a Git repository.
- Documentation: Detailed README file.
- Deployed Application: Live link for testing.
- Test Coverage: High coverage with unit tests.

9. Assessment Criteria:

- Code Quality & Security: Well-documented, maintainable code.
- Functionality: Meets specified requirements.
- Performance Optimization: Efficient resource use.
- UI/UX Design: User-friendly and appealing.
- Testing: Thoroughly tested and passed.
- Version Control: Effective Git uses with clear commits.

10. JOSS_CDP_OSS: GIT Repository Link:

- https://gitlab.happiestminds.com/root/INTPDJ01_3.git

Author: Rajesh Ganjeer
(Sr. Team Lead)