Subject: Advanced Web Development-BET1CE15321

Type of course: Professional Core

Prerequisite: Basic knowledge of HTML, CSS, JavaScript, PHP, Object-Oriented Programming, database concepts, and web technologies with logical problem-solving skills is required.

Rationale:

In the era of digitization, the demand for Internet-based applications is surging, driving the need for skilled developers' adept in both front-end and back-end design. This comprehensive course aims to immerse students in the intricacies of web development, empowering them to navigate the dynamic landscape of the Internet-driven world with confidence. By focusing on both front-end and back-end design principles, learners will gain a holistic understanding of the development process, from creating visually engaging user interfaces to implementing robust server-side functionalities.

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks		Total Marks
CI	T	P	C	SEE	CCE	
2	0	4	4	100	50	150

Legends: CI-Class Room Instructions; T – Tutorial; P - Practical; C – Credit; SEE - Semester End Evaluation; CCE-Continuous and Comprehensive Evaluation.

Course Content:

Sr. No	Course Content	Hrs.	% Weightage
1	**OOP in PHP** ***Theory Topics:*** Classes, Objects, Constructors, Destructors, Inheritance, Method Overriding, Abstract Classes & Interfaces, Traits & Multiple Inheritance, Namespaces & Autoloading, Magic Methods (__construct, __destruct, etc.), Exception Handling.	18	20%

Practical:

1. Create a user class with properties (name, email) and display data using a PHP webpage.
2. Create a class with constructor & destructor and show output in browser.
3. Build a student class and display student details in HTML table.
4. Create Person (parent) and Student (child) class and display data using a PHP form.
5. Override a method to display different dashboard messages for Admin and User roles.
6. Create an abstract Payment class and implement methods for UPI and Card payment.
7. Create an interface Logger and store logs into a file using PHP.
8. Create a trait for database connection and reuse it in multiple classes.
9. Use multiple traits for validation and logging in a form system.
10. Create a folder structure (models, controllers) and implement namespace-based class usage.
11. Implement autoloading using spl_autoload_register for automatic class loading.
12. Use __get() and __set() magic methods in a dynamic form handler.
13. Use __toString() method to display object data in the browser.
14. Create a custom exception for database connection failure.
15. Handle form errors using try-catch and display user-friendly messages.
16. Create a Logger class to store errors into a text file.
17. Store user activity logs (login, logout) using OOP-based implementation.
18. Develop an OOP-based Student Management System using PHP and MySQL with CRUD operations.

Examination Method:

Sr. No.	Evolution Methods	SEE	CCE
1	**Debugging and Implementation of OOP-Based PHP Application with Exception Handling:** Students will debug and complete a PHP application by applying classes, objects, constructors, inheritance, HTML output formatting, and custom exception handling for email validation..	20	-
2	**Optimization of Procedural PHP Code Using Object-Oriented Concepts, Traits, and Magic Methods:** Students will convert procedural PHP code into an optimized OOP-based structure using classes, traits, and magic methods to improve code readability, reusability, and browser output presentation.	-	10
	Total	20	10

Examination Style:

1. **Debugging and Implementation of OOP-Based PHP Application with Exception Handling (20 Marks):**
   - Students will be given a partially developed PHP program based on Object-Oriented Programming concepts for managing user and student data. They are required to identify and correct logical and structural errors by implementing class and object concepts, constructor initialization, and inheritance between the User and Student classes. Students must also add a method to display details in proper HTML format and implement exception handling to validate email input using a custom exception. The final output should display either the correct result or an appropriate error message using the try-catch block.

2. **Optimization of Procedural PHP Code Using Object-Oriented Concepts, Traits, and Magic Methods. (10 Marks):**
   - Students will be given a procedural PHP script for handling user data. They are required to convert and optimize the script into a proper Object-Oriented structure by creating suitable classes and improving code reusability. Students must incorporate at least one trait, such as validation or logging, and use any one magic method to enhance functionality. The final output should be cleanly displayed in the browser with improved code readability, structure, and maintainability.

2	**Database connection using PDO** ***Theory Topics:*** Database connection using PDO, Prepared Statements, SQL Injection Prevention, Transactions, Error Handling in PDO. ***Practical:*** 19. Connect PHP with MySQL using PDO. 20. Create database & table using PHP. 21. Insert data using prepared statement. 22. Fetch data using fetch (). 23. Fetch all records using fetchAll(). 24. Update records using PDO. 25. Delete records using PDO. 26. Implement search functionality. 27. Prevent SQL injection using prepared statements. 28. Use transactions (commit & rollback). 29. Create login system using PDO. 30. Display data in HTML table. 31. Pagination with database. 32. Error handling using try-catch in PDO. 33. Mini project: Secure CRUD Application. **Examination Method:**	18	20%

Sr. No.	Evolution Methods	SEE	CCE
1	**Optimization of PDO-Based Database Operations with Transactions and Pagination:** Students will optimize a PDO-based PHP script by implementing transactions, commit and rollback operations, pagination, improved error handling, and clean browser output.	20	-
2	**Active Learning Activity (ALA): Development and Debugging of Secure PDO-Based CRUD Application:** Students will debug and develop a secure PDO-based CRUD application using prepared statements, proper database connection, table creation, insert, fetch, update, delete operations, exception handling, and HTML table output.	-	10
	Total	20	10

	Examination Style: **1. Development and Debugging of Secure PDO-Based CRUD Application (20 Marks):** • Students will be given a partially implemented PHP application for managing user records using a database. They are required to identify and correct errors by establishing a proper database connection using PDO, creating the required tables, and implementing secure CRUD operations such as insert, fetch, update, and delete. Students must use prepared statements to prevent SQL injection, apply exception handling using the try-catch block, and display records in a proper HTML table. The final output should demonstrate a fully functional, secure, and database-driven PHP application. **2. Active Learning Activity (ALA): Optimization of PDO-Based Database Operations with Transactions and Pagination (10 Marks)** • Students will be given a basic GMIU Web Portal module developed using PHP and PDO, but the module lacks optimization and advanced database handling features. They are required to improve the module by implementing transactions with commit and rollback operations for safe data handling, adding pagination for efficient record display, and enhancing error handling. Students must prepare a short PDF report including the objective, code changes, screenshots, output, and conclusion, and upload it on the GMIU Web Portal.		
3	**Web Security and Session Management** ***Theory Topics:*** Form Handling & Validation, Input Sanitization, File Upload Security, Sessions & Cookies, Session Hijacking Prevention, CAPTCHA Integration. ***Practical:*** 34. Create HTML form with validation. 35. Create HTML form with validation. 36. Validate form using PHP. 37. Sanitize user input. 38. Implement login form with session. 39. Logout functionality. 40. Store user data in session. 41. Create cookie and retrieve it. 42. Delete cookie. 43. Secure file upload system. 44. Restrict file types and size. 45. Prevent directory traversal.	18	20%

46. Implement CAPTCHA (Google reCAPTCHA).
47. Create registration form with validation.
48. Password hashing using password hash().
49. Mini project: Secure Login & Registration System.

Examination Method:

Sr. No.	Evolution Methods	SEE	CCE
1	**Development of Secure PHP-Based Login and Registration System with Validation and Session Management:** Students will develop a secure PHP login and registration system by implementing validation, input sanitization, password hashing, session handling, cookies, CAPTCHA, secure file upload, and proper success or error messages.	20	-
2	**Enhancement of PHP Form Handling with Input Sanitization, Cookies, and Security Practices:** Students will enhance a basic PHP form handling script by applying input sanitization, improved validation, cookie management, security practices, and user-friendly browser output.	-	10
	Total	20	10

Examination Style:

1. **Development of Secure PHP-Based Login and Registration System with Validation and Session Management (20 Marks):**
   - Students will be required to develop a secure PHP-based web application that includes user registration and login functionality. They must implement proper form handling with validation and input sanitization. Students are required to store user credentials securely using password hashing and manage login and logout functionality using sessions. They must also maintain required user data using sessions and cookies, integrate CAPTCHA for additional security, and implement secure file upload with restrictions on file type and file size while preventing directory traversal attacks. The final application should ensure data security and display appropriate success or error messages.

2. **Enhancement of PHP Form Handling with Input Sanitization, Cookies, and Security Practices. (10 Marks):**
   - Students will be given a basic PHP form handling script that lacks proper validation and security practices. They are required to enhance the script by implementing input sanitization, improving validation, managing cookies effectively, and applying

<table>
<tr><td></td><td>secure programming practices. The final output should be user-friendly, secure, properly structured, and should follow secure coding standards.</td><td></td><td></td></tr>
<tr><td>4</td><td>AJAX and JSON Integration
<u>Theory Topics:</u>
AJAX concepts (sync vs async), Fetch API JSON encoding/decoding, API requests, Client-side & server-side error handling.

<u>Practical:</u>

50. Create simple AJAX request.
51. Fetch data without page reload.
52. Use Fetch API (GET request).
53. Use Fetch API (POST request).
54. Send form data via AJAX.
55. Display JSON data using JavaScript.
56. Convert PHP array to JSON.
57. Decode JSON in PHP.
58. Create live search using AJAX.
59. Create auto-suggestion system.
60. Handle AJAX errors.
61. Build dynamic dropdown.
62. Load content dynamically.
63. Validate form using AJAX.
64. Mini project: AJAX-based Dashboard.

Examination Style:
<table><tr><th>Sr. No.</th><th>Evolution Methods</th><th>SEE</th><th>CCE</th></tr><tr><td>1</td><td>Development of AJAX-Based Dynamic Web Application Using Fetch API and JSON Handling:
Students will develop a dynamic PHP-based web application using AJAX and Fetch API by implementing GET and POST requests, JSON data handling, live search or auto-suggestion, dynamic content loading, and proper error messages.</td><td>20</td><td>-</td></tr><tr><td>2</td><td>Active Learning Activity (ALA): Optimization of AJAX-Based Application with JSON Processing and Error Handling:
Students will optimize a basic AJAX-based application by improving Fetch API request handling, JSON encoding and decoding, client-side error handling, dynamic output display, and overall user experience.</td><td>-</td><td>10</td></tr><tr><td colspan="2">Total</td><td>20</td><td>10</td></tr></table></td><td>18</td><td>20%</td></tr>
</table>

	Examination Style: 1. **Development of AJAX-Based Dynamic Web Application Using Fetch API and JSON Handling (20 Marks):** • Students will be required to develop a dynamic PHP-based web application using AJAX and Fetch API to perform asynchronous operations without page reload. They must implement both GET and POST requests to send and retrieve data from the server. Students are required to convert PHP data into JSON format and display it using JavaScript. They must also create features such as live search or auto-suggestion and dynamically load content into the webpage. The final application should handle client-side and server-side errors properly and display appropriate messages in the interface. 2. **Optimization of AJAX-Based Application with JSON Processing and Error Handling (10 Marks):** • Students will be given a basic AJAX-based application that performs data fetching but lacks proper structure and error handling. They are required to optimize the application by improving JSON encoding and decoding, enhancing request handling using Fetch API, and implementing effective client-side error handling. The final output should be dynamic, user-friendly, properly structured, and should follow best practices. Students must prepare a short PDF report including the objective, code changes, screenshots, output, and conclusion, and upload it on the GMIU Web Portal.		
5	**RESTful API Development** ***Theory Topics:*** REST API architecture, CRUD operations via API, JSON & XML responses, JWT Authentication, OAuth Basics, Third-party API integration ***Practical:*** 65. Create basic REST API in PHP. 66. Create GET API. 67. Create POST API. 68. Create PUT API. 69. Create DELETE API. 70. Connect API with database. 71. Return JSON response. 72. Parse JSON in frontend. 73. Create XML response. 74. Implement JWT authentication. 75. Secure API endpoints.	18	20%

76. Test API using Postman.
77. Integrate third-party API (Weather API).
78. Handle API errors.
79. Mini project: Full CRUD REST API System.

Examination Style:

Sr. No.	Evolution Methods	SEE	CCE
1	**Development of Secure RESTful API with CRUD Operations, JWT Authentication, and JSON/XML Response Handling:** Students will develop a secure RESTful API in PHP by implementing GET, POST, PUT, and DELETE operations, database connectivity, JSON/XML response handling, JWT-based authentication, endpoint security, error handling, and API testing using Postman.	20	-
2	**Active Learning Activity (ALA): Optimization of REST API with Third-Party Integration and Error Handling:** Students will optimize a basic REST API by improving response structure, implementing proper error handling, integrating a third-party API such as Weather API, and ensuring secure, meaningful, and clean API responses.	-	10
	Total	20	10

Examination Style:

1. **Development of Secure RESTful API with CRUD Operations, JWT Authentication, and JSON/XML Response Handling (20 Marks):**
   - Students will be required to develop a secure RESTful API in PHP that performs complete CRUD operations, including GET, POST, PUT, and DELETE, by connecting to a database. They must return API responses in JSON format and may optionally support XML response handling. Students are required to implement JWT-based authentication to secure API endpoints and ensure authorized access only. They must also integrate proper error handling for API requests and demonstrate endpoint testing using tools such as Postman. The final output should represent a fully functional, secure, and well-structured API system.
2. **Optimization of R EST API with Third-Party Integration and Error Handling (10 Marks):**
   - Students will be given a basic REST API script that lacks proper structure, advanced features, and effective error handling. They

		are required to optimize the API by improving response handling, implementing proper error management, and integrating a third-party API, such as a Weather API. The final API should be well-structured, secure, and capable of returning clean, meaningful, and user-friendly responses. Students must prepare a short PDF report including the objective, code changes, screenshots, output, testing evidence, and conclusion, and upload it on the GMIU Web Portal.		

Suggested Specification table with Marks (Theory):100

Distribution of Theory Marks (Revised Bloom's Taxonomy)						
Level	Remembrance (R)	Understanding (U)	Application (A)	Analyze (N)	Evaluate (E)	Create (C)
Weightage %	10%	15%	30%	15%	10%	20%

Note: This specification table shall be treated as a general guideline for students and teachers. The actual distribution of marks in the question paper may vary slightly from above table.

Course Outcome:

After learning the course, the students should be able to:	
CO1	Apply PHP OOP concepts to develop modular web applications.
CO2	Develop secure database-driven applications using PDO and transactions.
CO3	Design secure web forms using validation, sessions, CAPTCHA, and file security.
CO4	Implement dynamic web applications using AJAX, Fetch API, and JSON.
CO5	Develop secure RESTful APIs with CRUD, JWT, and third-party integration.

Instructional Method:

The course delivery method will depend upon the nature of the content and practical requirements of the students. The instructor may use conventional teaching methods along with demonstration, live coding, debugging sessions, brainstorming, quizzes, project-based learning, flipped classroom approach, MOOCs, and hands-on laboratory practice.

Students will use supplementary learning resources such as online tutorials, NPTEL/SWAYAM courses, technical documentation, virtual labs, GitHub repositories, and API testing tools.

The internal evaluation will be carried out through active learning assignments, mini projects, debugging activities, code optimization tasks, and practical performance. Practical and viva examinations will be conducted at the end of the semester to evaluate implementation skills and conceptual understanding

Reference Books:

[1] *PHP and MySQL Web Development — Luke Welling & Laura Thomson, Addison-Wesley.*
[2] *Modern PHP: New Features and Good Practices — Josh Lockhart, O'Reilly Media.*
[3] *JavaScript and AJAX for the Web — Tom Negrino & Dori Smith, Peachpit Press.*
[4] *RESTful Web APIs — Leonard Richardson & Mike Amundsen, O'Reilly Media.*
[5] *Web Application Security — Andrew Hoffman, O'Reilly Media.*

Suggested Rubrics:

Module-1: OOP in PHP

OOP-Based PHP Application Development (20 Marks)		
Criteria	**Description**	**Marks**
Problem Understanding & Class Design	Correct identification of the requirement and design of classes, objects, and constructors	05
OOP Implementation (Inheritance, Interfaces & Traits)	Proper implementation of inheritance, method overriding, abstract classes/interfaces, and traits with correct logic	05
Exception Handling & Input Validation	Clear implementation of exception handling and validation for robust, error-free execution	05
Output & Result Verification	Accurate final output with correct verification of class behavior and functionality	05
	Total	**20**

Module-2: Database Connectivity using PDO

PDO-Based Database Operations (20 Marks)		
Criteria	**Description**	**Marks**
Problem Understanding & Database Design	Correct identification of the requirement and design of database/table structure	05
PDO Implementation & CRUD Operations	Proper implementation of PDO connection with correct CRUD logic using prepared statements	05
Transaction & Security Handling	Clear demonstration of transactions (commit/rollback), SQL injection prevention, and error handling	05
Output & Result Verification	Accurate final output with correct verification of database operations	05
	Total	**20**

Module-3: Web Security and Session Management

Secure Form Handling and Session Management (20 Marks)		
Criteria	**Description**	**Marks**
Problem Understanding & Form Design	Correct identification of the security requirement and form/session design approach	05
Validation & Sanitization Implementation	Proper implementation of input validation, sanitization, and secure coding logic	05
Session, Cookie & File Security Handling	Clear demonstration of session/cookie management, CAPTCHA integration, and secure file upload	05
Output & Result Verification	Accurate final output with correct verification of the security measures applied	05
	Total	**20**

Module-4: AJAX and JSON Integration

AJAX-Based Dynamic Web Application (20 Marks)		
Criteria	**Description**	**Marks**
Problem Understanding & Request Design	Correct identification of the asynchronous requirement and request/response design	05
AJAX / Fetch API Implementation	Proper implementation of AJAX or Fetch API with correct client-server communication logic	05
JSON Processing & Execution Trace	Clear demonstration of JSON encoding/decoding and step-by-step data handling	05
Output & Result Verification	Accurate final output with correct verification of dynamic functionality	05
	Total	**20**

Module-5: RESTful API Development

Secure REST API with CRUD Operations (20 Marks)		
Criteria	**Description**	**Marks**
Problem Understanding & API Design	Correct identification of the API requirement and endpoint design approach	05
CRUD & Endpoint Implementation	Proper implementation of REST endpoints (GET/POST/PUT/DELETE) with correct logic	05
Authentication & Security Handling	Clear demonstration of JWT authentication and secure API practices	05
Output & Result Verification	Accurate final output with correct verification and testing of API responses	05
	Total	**20**