



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



NAME: Marjovic P. Alejado

Schedule: 7:00AM-10:00AM

Score: _____

SUBJECT: **WEB SYSTEMS AND TECHNOLOGIES** INSTRUCTOR: Jim S. Jamero

DATE: 10-7-2025

LABORATORY EXERCISE 6 COURSE ENROLLMENT SYSTEM

Learning Objectives

By the end of this laboratory exercise, students should be able to:

- Design and create a new database table to manage relationships between users and courses.
- Implement server-side logic for handling course enrollments.
- Display user-specific data (enrolled courses) in a dashboard.
- Utilize jQuery and AJAX to create a dynamic, seamless user experience without page reloads.
- Understand and implement basic foreign key relationships in a web application.

Prerequisite student experiences and knowledge

Before starting this exercise, students should have:

- ❖ Completed Laboratory Exercise 5 (Admin and Student Dashboards).
- ❖ A solid understanding of the MVC architecture in CodeIgniter.
- ❖ Proficiency in writing database queries using CodeIgniter's Query Builder.
- ❖ Basic knowledge of SQL relationships (one-to-many).
- ❖ Familiarity with jQuery syntax and the concept of AJAX.
- ❖ Ability to create and style front-end components with Bootstrap.

Background

A core feature of any Learning Management System (LMS) is the ability for students to enroll in available courses. This involves creating a relationship between the **users** table (students) and the **courses** table. This relationship is typically stored in a pivot table. To enhance user experience, the enrollment process should be dynamic, allowing students to join courses without refreshing the page. This is achieved using jQuery AJAX to send a request to the server in the background, providing immediate feedback to the user.

Materials/Resources

- Personal Computer with Internet Access
- XAMPP/WAMP/LAMP server installed
- CodeIgniter Framework (latest version)
- Visual Studio Code or any code editor
- Git and GitHub Account
- Web Browser (Chrome, Firefox, etc.)

Laboratory Activity

Step 1: Create a Database Migration for the Enrollments Table

1. Create a new migration file for the **enrollments** table.
Run: `php spark make:migration CreateEnrollmentsTable`



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



NAME: Marjovic P. Alejado

Schedule: 7:00AM-10:00AM

Score: _____

SUBJECT: **WEB SYSTEMS AND TECHNOLOGIES** INSTRUCTOR: Jim S. Jamero

DATE: 10-7-2025

2. Open the newly created file in app/Database/Migrations/.
3. In the up() method, define the table with the following fields:
 - ✓ id (primary key, auto-increment)
 - ✓ user_id (int, foreign key to **users** table)
 - ✓ course_id (int, foreign key to **courses** table)
 - ✓ enrollment_date (datetime)
4. In the down() method, define how to drop the table.
5. Run the migration: php spark migrate.

Step 2: Create the Enrollment Model

1. Navigate to app/Models/ and create a file named EnrollmentModel.php.
2. Create a model class with methods to:
 - ✓ enrollUser(\$data): Insert a new enrollment record.
 - ✓ getUserEnrollments(\$user_id): Fetch all courses a user is enrolled in.
 - ✓ isAlreadyEnrolled(\$user_id, \$course_id): Check if a user is already enrolled in a specific course to prevent duplicates.

Step 3: Modify the Course Controller

1. Open your Course.php controller (or create it if it doesn't exist).
2. Add a new method, enroll(), to handle the AJAX request.
 - ✓ This method should:
 - ✓ Check if the user is logged in.
 - ✓ Receive the **course_id** from the POST request.
 - ✓ Check if the user is already enrolled.
 - ✓ If not, insert the new enrollment record with the current timestamp.
 - ✓ Return a JSON response indicating success or failure.

Step 4: Update Student Dashboard View

1. Open/Check the student dashboard view file.
2. Create a section to **Display Enrolled Courses**. Use a Bootstrap list group or cards to iterate over and display the courses returned by **EnrollmentModel::getUserEnrollments()**.
3. Create another section for **Available Courses**. Display a list of courses with an **Enroll** button next to each.

Step 5: Implement AJAX Enrollment

1. In the **Available Courses** section of the dashboard, add a **data_course_id** attribute to each **Enroll** button containing the specific course ID.
2. Include the jQuery library in your view if it's not already included.
3. Write a jQuery script that:
 - ✓ Listens for a click on the **Enroll** button.
 - ✓ Prevents the default form submission behavior.



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



NAME: Marjovic P. Alejado

Schedule: 7:00AM-10:00AM

Score: _____

SUBJECT: **WEB SYSTEMS AND TECHNOLOGIES**

INSTRUCTOR: Jim S. Jamero

DATE: 10-7-2025

- ✓ Uses `$.post()` to send the **course_id** to the `/course/enroll` URL.
- ✓ On a successful response from the server:
- ✓ Displays a Bootstrap alert message.
- ✓ Hides or disables the **Enroll** button for that course.
- ✓ Updates the **Enrolled Courses** list dynamically without reloading the page.

Step 6: Configure Routes

1. Update `app/Config/Routes.php` to include a route for the enrollment action.
`$routes->post('/course/enroll', 'Course::enroll');`

Step 7: Test the Application Thoroughly

1. Log in as a student.
2. Navigate to the student dashboard.
3. Click the **Enroll** button on an available course and verify:
 - The page does not reload.
 - A success message appears.
 - The button becomes disabled or disappears.
 - The course appears in the **Enrolled Courses** list.

Step 8: Push to GitHub

1. Commit your changes with a descriptive message.
2. Push your changes to your GitHub repository.

Step 9: Vulnerable Checking

1. Test for Authorization Bypass
 - ❖ Log out of the application and attempt to directly access the enrollment endpoint via Postman or browser console by sending a POST request to `/course/enroll` with a `course_id` parameter.
 - ❖ Verify that the server returns an unauthorized error instead of processing the enrollment.
2. Test for SQL Injection
 - ❖ While logged in, use browser developer tools to modify the AJAX request and change the `course_id` value to `1 OR 1=1`.
 - ❖ Check if the application properly validates the input and prevents SQL injection attacks.
3. Test for CSRF (Cross-Site Request Forgery)
 - ❖ Check if your enrollment form includes CSRF protection tokens.
 - ❖ Verify that CodeIgniter's CSRF protection is enabled in `app/Config/Security.php`.
 - ❖ Attempt to make an enrollment request without a valid CSRF token and confirm it is rejected.
4. Test for Data Tampering



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



NAME: Marjovic P. Alejado

Schedule: 7:00AM-10:00AM

Score: _____

SUBJECT: **WEB SYSTEMS AND TECHNOLOGIES**

INSTRUCTOR: Jim S. Jamero

DATE: 10-7-2025

-
- ❖ As a student, try to enroll another user in a course by modifying the user ID in the request.
 - ❖ Verify that the server-side code uses the logged-in user's session ID rather than trusting client-supplied user IDs.
5. Test for Input Validation
- ❖ Attempt to enroll in non-existent courses by sending invalid course_id values.
 - ❖ Verify that the application properly validates that the course exists before creating an enrollment.

Output / Results

- ✓ Screenshot of your database's **enrollments** table structure (phpMyAdmin or equivalent).
- ✓ A screenshot of the student dashboard showing the **Available** and **Enrolled Courses** sections is attached.
- ✓ A screenshot of the browser's developer tools (Network tab) shows the successful AJAX POST request and response when enrolling in a course.
- ✓ A screenshot of the GitHub repository with the latest commit for this exercise.



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



NAME: Marjovic P. Alejado

Schedule: 7:00AM-10:00AM

Score: _____

SUBJECT: **WEB SYSTEMS AND TECHNOLOGIES** INSTRUCTOR: Jim S. Jamero

DATE: 10-7-2025

QUESTIONS:

1. What is the purpose of the **enrollments** table? Why is it necessary, instead of just adding a **course_id** column to the **users** table?

The purpose of the enrollments table is to connect students with the courses they are taking by storing the relationship between users and courses. It is necessary because one student can take many courses, and one course can have many students enrolled in it.

2. Explain the role of the **isAlreadyEnrolled()** method in the Model. What potential issue does it prevent?

The role of the **isAlreadyEnrolled()** method in the Model is by checking if a student is already enrolled in a specific course before allowing a new enrollment. Potential issue it does prevent is duplicate enrollments because it stops a student from being enrolled in the same course multiple times by mistake.

3. Describe the client-side and server-side steps when students click the **Enroll** button until they receive confirmation.

Steps in Client Side Scripting

When I click the **Enroll** button, the JavaScript immediately disables the button, shows a loading spinner, and prepares my enrollment request with the **course_id** and **CSRF token** from the page. The browser then sends a **POST** request to the **/course/enroll** endpoint with headers including **X-Requested-With: XMLHttpRequest** and **X-CSRF-TOKEN** to verify it's AJAX requested by me/the user.

Steps in Server Side Scripting

The server receives my POST request at the **/course/enroll endpoint** and checks if I'm logged in as a student while validating all the security tokens and headers I sent. If everything is correct and I'm not already enrolled, the server adds my enrollment to the database and sends back a JSON response with my enrollment details.



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



Output/Results

phpMyAdmin Enrollments Table

Dashboard | GitHub Dashboard | localhost / 127.0.0.1 / lms_alejo | Student Dashboard - MGOD L | AJAX REQUEST IN POSTMAN

localhost/phpmyadmin/index.php?route=/sql&pos=0&db=lms_alejado&table=enrollments

Server: 127.0.0.1 » Database: lms_alejado » Table: enrollments

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Tracking | Triggers

Showing rows 0 - 10 (11 total. Query took 0.0004 seconds.)

SELECT * FROM `enrollments`

Profiling [Edit Inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

				id	user_id	course_id	enrollment_date
<input type="checkbox"/>	Edit	Copy	Delete	1	4	1	2025-10-07 13:48:58
<input type="checkbox"/>	Edit	Copy	Delete	2	4	2	2025-10-07 13:50:08
<input type="checkbox"/>	Edit	Copy	Delete	3	4	3	2025-10-07 13:50:11
<input type="checkbox"/>	Edit	Copy	Delete	4	5	1	2025-10-07 13:53:30
<input type="checkbox"/>	Edit	Copy	Delete	5	5	2	2025-10-07 14:00:13
<input type="checkbox"/>	Edit	Copy	Delete	6	5	3	2025-10-07 14:00:21
<input type="checkbox"/>	Edit	Copy	Delete	7	6	1	2025-10-07 15:02:51
<input type="checkbox"/>	Edit	Copy	Delete	8	6	2	2025-10-07 15:05:59
<input type="checkbox"/>	Edit	Copy	Delete	9	6	3	2025-10-07 15:14:02
<input type="checkbox"/>	Edit	Copy	Delete	10	7	1	2025-10-07 15:19:50
<input type="checkbox"/>	Edit	Copy	Delete	11	7	3	2025-10-07 23:28:51

Check all | With selected: Edit | Copy | Delete | Export

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Query results operations

Console



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



Student Dashboard

Dashboard

GitHub Dashboard

localhost / 127.0.0.1 / lms_alejo

Student Dashboard - MGOD LMS

localhost/ITE311-ALEJADO/dashboard#

MGOD LMS

Student

Dashboard

My Courses

Assignments

Grades

Schedule

M Marjovic Alejado

Student Dashboard

Welcome back, Marjovic Alejado! Continue your learning journey and achieve your goals.

2

Enrolled Courses

Active learning paths

0

Completed

Assignments finished

0

Pending

Awaiting completion

0%

Average Grade

Overall performance

My Enrolled Courses

Continue your learning journey

Database Design

Enrolled

DB201 • Marjovic Prato Alejado

Learn relational database concepts, SQL, and database optimization....

Progress0%

Enrolled: Oct 7, 2025

Continue

Introduction to Programming

Enrolled

CS101 • Marjovic Prato Alejado

Learn the fundamentals of programming with hands-on examples and projects....

Progress0%

Enrolled: Oct 7, 2025

Continue

Available Courses

Discover new learning opportunities

Web Development Basics

Active

WEB101 • Marjovic Prato Alejado

Master HTML, CSS, and JavaScript to build modern web applications....

Credits4

Duration12w

Students25

Feb 1, 2024 - Apr 30, 2024

Enroll in Course

Upcoming Deadlines

Don't miss these important dates

Assignment deadlines and important dates will appear here.

Recent Grades & Feedback

Your latest academic performance

Your grades and teacher feedback will appear here.

Search

12:03 AM 10/8/2025



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



Browser's Developer Tools AJAX POST

The screenshot shows a web browser window with the URL `localhost/ITE311-ALEJADO/dashboard`. A modal dialog titled "Enrollment Successful!" is displayed in the center. The dialog contains a green checkmark icon, the text "Welcome to Database Design!", and a message: "You have been successfully enrolled in this course. You can now access course materials and start learning." Below this, it shows the "Enrollment Date: Oct 7, 2025 11:28PM" and a "Refresh Page" button. A "Close" button is at the bottom right of the dialog.

The background of the browser shows a dashboard with sections for "Upcoming Deadlines" and "Recent Grades & Feedback".

The right side of the image shows the browser's Developer Tools, specifically the Network tab. It displays a list of requests, with the selected request being an "enroll" request. The request details are as follows:

Name	Method	Status	Response
enroll	POST	201 Created	application/json; charset=UTF-8

The request headers are:

- Request URL: `http://localhost/ITE311-ALEJADO/course/enroll`
- Request Method: POST
- Status Code: 201 Created
- Remote Address: `[::1]:80`
- Referrer Policy: `strict-origin-when-cross-origin`

The response headers are:

- Cache-Control: `no-store, no-cache, must-revalidate`
- Connection: `Keep-Alive`
- Content-Length: `258`
- Content-Type: `application/json; charset=UTF-8`
- Date: `Tue, 07 Oct 2025 15:28:51 GMT`
- Expires: `Thu, 19 Nov 1981 08:52:00 GMT`
- Keep-Alive: `timeout=5, max=100`
- Pragma: `no-cache`
- Server: `Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.2.12`
- Set-Cookie: `ci_session=e2979a7fac714fba199d02747c8a74db; expires=Tue, 07 Oct 2025 17:28:51 GMT; Max-Age=7200; path=/; HttpOnly; SameSite=Lax`
- X-Powered-By: `PHP/8.2.12`

The request headers also show:

- Accept: `*/*`
- Accept-Encoding: `gzip, deflate, br, zstd`



RAMON MAGSAYSAY MEMORIAL COLLEGES
Information Technology Education Program
1st SEMESTER: AY: 2025 - 2026



GitHub Link: <https://github.com/Marjovic/WebSystem-ITE311>

The screenshot shows the GitHub repository page for 'WebSystem-ITE311' by user 'Marjovic'. The repository is public and has 0 forks and 0 stars. The main branch is 'main'. The repository contains several files and folders: 'app', 'public', 'system', 'tests', 'writable', '.htaccess', 'LICENSE', 'README.md', 'composer.json', 'env', and 'index.php'. The most recent commit is 'Add CSRF tokens to the dashboard for security purposes' by Marjovic, committed 1 minute ago. The repository also has a 'Readme' file, a 'MIT license', and a 'Languages' section showing PHP at 98.0% and JavaScript at 1.1%.

The screenshot shows the 'Commits' section of the GitHub repository for 'WebSystem-ITE311'. The repository is public and has 0 forks and 0 stars. The main branch is 'main'. The repository contains several files and folders: 'app', 'public', 'system', 'tests', 'writable', '.htaccess', 'LICENSE', 'README.md', 'composer.json', 'env', and 'index.php'. The most recent commit is 'Add CSRF tokens to the dashboard for security purposes' by Marjovic, committed 1 minute ago. The repository also has a 'Readme' file, a 'MIT license', and a 'Languages' section showing PHP at 98.0% and JavaScript at 1.1%.

Commit Message	Author	Committed	SHA-1
Enhance student dashboard to display enrolled and available courses with progress tracking	Marjovic	29 minutes ago	7688d9
Add Course controller and seeder for course enrollment functionality	Marjovic	30 minutes ago	d6ebbcc
Add EnrollmentModel for managing user enrollments and course registrations	Marjovic	30 minutes ago	c146dc4
Enhance enrollment and submission tables to replace 'student_id' with 'user_id'	Marjovic	31 minutes ago	eb4a2ef
Refactor dashboard view and auth controller for role based implementation	Marjovic	4 hours ago	7af8d3d
Improves Vulnerability and added Manage Users Functionality	Marjovic	5 hours ago	3f14e27



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



Vulnerability Checking

The screenshot shows the Postman interface with a workspace named "LMS". The "Auth" collection is expanded, showing a "POST Login" request. The request is a POST to `http://localhost/ITE311-ALEJADO/login` with the following parameters:

Key	Value	Description
email	maryjoy.alejado@student.lms.com	
password	student123	

The response is a 200 OK status with a response time of 143 ms and a body size of 33.47 KB. The response body is a JSON object representing a student dashboard:

```
{  "status": "success",  "message": "Welcome back, Mary Joy Prato Alejado! Continue your learning journey and achieve your goals.",  "data": {    "user": {      "id": 1,      "name": "Mary Joy Prato Alejado",      "email": "maryjoy.alejado@student.lms.com",      "role": "Student"    },    "courses": [      {        "id": 1,        "name": "Introduction to IT",        "status": "enrolled"      },      {        "id": 2,        "name": "Web Development",        "status": "not_enrolled"      },      {        "id": 3,        "name": "Database Management",        "status": "not_enrolled"      },      {        "id": 4,        "name": "Networking Fundamentals",        "status": "not_enrolled"      },      {        "id": 5,        "name": "Cybersecurity Basics",        "status": "not_enrolled"      }    ]  } }
```

The screenshot shows the Postman interface with a workspace named "LMS". The "Vulnerability Check" collection is expanded, showing a "POST 5. Test for Input Validation" request. The request is a POST to `http://localhost/ITE311-ALEJADO/course/enroll` with the following parameters:

Key	Value	Description
course_id	1	
csrf_token_name	c413d2fe59979b582063949b8aaabdc1	

The response is a 409 Conflict status with a response time of 149 ms and a body size of 669 B. The response body is a JSON object representing an error message:

```
{  "success": false,  "message": "You are already enrolled in this course.",  "error_code": "ALREADY_ENROLLED",  "csrf_hash": "c413d2fe59979b582063949b8aaabdc1" }
```



RAMON MAGSAYSAY MEMORIAL COLLEGES

Information Technology Education Program

1st SEMESTER: AY: 2025 - 2026



Conclusion

In conclusion, thanks to this Laboratory Exercise 5, I learned how to create course enrollment system by designing database tables with proper relationships and implementing secure server-side logic to handle student enrollments.

In addition, I also learned how to use jQuery and AJAX to create a interactive user experience where students can enroll in courses without refreshing the page.