APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 6 No. 1 (2025) 1165-1184
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

The Development of DSG Micro-credential Course Registration System using Laravel Framework

Amirul Hafiz Zulazli[1], Nur Ariffin Mohd Zin[1]*

[1] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: ariffin@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2025.06.01.064

Article Info

Received: 14 July 2024
Accepted: 18 June 2025
Available online: 30 June 2025

Keywords

DSG Micro-credential Course Registration System, Web Technology, Prototyping model, DSG.

Abstract

The term "micro" signifies something small or on a smaller scale, and inthe context of a micro-credential, it describes a brief, targeted program focused on specific skills or knowledge. The Doctorate Support Group (DSG) on Facebook, with its 107,000 members, is a vital support hub for master's and PhD students. However, DSG faces significant challenges with user frustration and inefficient data management due to a lack of standardized registration processes, impacting both user experience and organizational efficiency. Moreover, the absence of a dedicated system for monitoring user engagement and progress hampers data-driven decision-making and program assessment, whilethe resource-heavy manual registration process limits scalability and efficiency, making it hard to meet the increasing demand for micro- credential programs. To resolve these issues, the DSG Micro-credentialCourse Registration System integrates web technology to overhaul theoutdated methods. Using PHP with the Laravel framework and MySQL as the database, this system adopts an object-oriented approach to deliver a streamlined, web-based registration process. It includes modules for User Registration, Administrative Reports, Course Management, User Management, Payments, and more. Following rigorous alpha and beta testing, User Acceptance Testing (UAT) indicates that over 80% of respondents are satisfied with the system. In conclusion, the DSG Micro-credential Course Registration System notonly addresses these critical challenges but also sets new standards forefficient educational support, fostering a better environment for academic success.

1. Introduction

In response to the dynamic landscape of doctoral education, the Doctorate Support Group (DSG) has recognized the need for transformative support, leading to the conceptualization of the DSG Micro-credential Course Registration System. Micro-credentials, characterized by their focused and concise nature, offer a solution to the evolving demands of academia [1]. With a substantial membership of 103,045, DSG serves as a central hub for students, emphasizing the interdisciplinary journey of pursuing a doctorate in today's rapidly changing academic environment [2]. However, the organization faces critical challenges, including an unstandardized registration system, an inability to track registered users effectively, and reliance on a manual registration process. This research project aims to address these issues by proposing a comprehensive web-based registration system, utilizing an object-oriented approach. The anticipated outcomes encompass a standardized and user-friendly

registration process, enhanced data-driven decision-making, and improved scalability for DSG's micro-credential program, ultimately contributing to the holistic development of future scholars and the advancement of doctoral education.

2. Related Work

This section provides a comprehensive overview of the proposed Doctorate Support Group (DSG) Micro-credential Course Registration System, addressing the evolving landscape of education with a focus on micro-credentials. Micro-credentials are explored as concise acknowledgments of individual expertise, emphasizing targeted skills in an efficient and practical manner [4].

The section delves into the Course Registration System, detailing its multifaceted features, including user registration, administrative controls, course selection, payment processing, and communication tools. The adoption of the Laravel framework is justified for its compatibility with the system's design objectives. A critical analysis of DSG's current manual processes highlights the inefficiencies, setting the stage for the proposed system's transformative impact [3].

The study of existing systems, such as Help University Micro-credential, UTAR Massive Open Online Course, and INTI Micro-credential Program, provides valuable insights and potential enhancements for the DSG system. A systematic comparison of features reveals the unique strengths of the DSG Micro-credential Course Registration System, particularly in user registration, administrative reporting, the classroom system allowing Zoom meetings, and the "My Courses" section that includes learning materials, assignments, and marks. The chapter concludes by emphasizing the exclusive attributes of the proposed system and sets the stage for the subsequent methodology chapter.

Table 1 *Comparison with the Existing Systems*

Features/System	**Help University Micro-credential**	**UTAR Massive Open Online Course (MOOC)**	**INTI Micro-Credential Program**	**DSG Micro-credential Course Registration System**
User Registration	X	X	X	√ Users have to register and login
Administrative Report	X	X	X	√ Provide administrators with access to administrative reports, including total students, total instructors, total income, a graph depicting students' gender distribution, and a table of popular courses.
Classroom System	X	X	X	√ Students and instructors can join and create meetings respectively
Courses Registration	√	√	√	√ Course fees, user details
My Courses	X	X	X	√ Students and instructors can view their assigned courses
Payment System	X	X	X	√ Payment gateway using Stripe

In conclusion, this section provides a comprehensive overview of studies conducted by other researchers, serving as references, sources of information, and guidance for the development of this system. First, micro-credentials are explained; these are certifications that recognize skills and competencies acquired through short, focused learning experiences. Second, the study of the course registration system addresses the issues identified with micro-credentials and proposes a solution. Third, the Laravel framework is discussed as the technology chosen for the system. Additionally, the examination of existing systems highlights how their functionalities can be improved. With these guidelines, references, and insights, this system aims to meet all the goals and objectives set, ensuring acceptance by its users—administrators, instructors, and students.

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 1165-1184

3. Methodology

3.1 Prototyping Model

The Prototyping Model, characterized by its iterative development approach, involves the continuous refinement of an initial system prototype through rapid cycles of user feedback and subsequent adjustments. As articulated by Sommerville (2011), this methodology emphasizes active end-user involvement, fostering a collaborative environment where the development team works closely with users to iteratively build a functional prototype. The model's efficacy lies in its capacity to address evolving or ambiguous requirements, allowing for early identification and mitigation of potential risks. It promotes adaptability and a design philosophy centered on user needs, ensuring the final product closely aligns with user expectations.

For the DSG Micro-credential Course Registration System, the development process involves five phases: Planning, Analysis, Design, Implementation, and Prototyping. The planning phase defines tasks, outlines the problem, scope, and objectives, and utilizes Gantt Chart techniques for effective management. The analysis phase gathers requirements, scrutinizes existing systems, and defines functional and non-functional aspects using UML diagrams. The design phase focuses on creating architecture, database, and interface designs, detailing hardware, software, network infrastructure, and user interactions. The implementation phase utilizes Visual Studio Code and Xampp, employing PHP and SQL for development, with testing supported by Selenium IDE. The testing phase involves iterative cycles, starting with an initial prototype, followed by evaluation and feedback from the system's end-users, including administrators, instructors, and students. Reanalysis, redesign, and reimplementation cycles continue until a consensus is reached on a fully functional prototype ready for deployment and use within the user community. This approach ensures the system is developed in alignment with user needs and expectations, effectively addressing evolving requirements and potential risks through continuous user feedback and iterative refinement.

Table 2: *Software development activities and their task*

Phase	Task	Output
Planning	• Proposed the project • Determine the project schedule, activities, and output	• Project proposal • Develop Gantt chart
Analysis	• Collect and analyse the collected information	• Functional and non-functional requirements • User requirement analysis • Software and hardware requirements • UML diagrams • Requirement traceability matrix
Design	• Design a logical of the structure of the system	• Architecture Design • Database Design • Interface Design
Implementation	• Develop the system • Test the system	• Program code • Test code
Prototype	• Repeat task from planning phase until implementation phase and detect errors again on the system and repair the existing system	• System prototype • Final system

3.2 System requirements

Requirement analysis determines the requirements that the developed system must meet or the users' expectations from the proposed system. Furthermore, these requirements are critical in defining the system's overall utility and quality because they influence both. The requirements analysis, including functional and non-functional requirements, and user requirements, will be discussed in further detail in the following sections.

Table 3 Functional Requirements

No.	Module	Functional requirements
1.	User Registration	• The system shall allow users to log in using a registered username and password. • The system should alert users for any invalid input.
2.	Administrative Report	• The system shall allow administrators to view and download administrative reports.
3.	Manage Courses	• The system shall allow administrators to view, add, edit, and delete courses.
4.	Manage Users	• The system shall allow administrators to view, edit, and delete user information.
5.	Manage Payments	• The system shall allow administrators to view and approve course fee payments.
6.	Courses Registration	• The system shall allow students to view available courses and enroll in them.
7.	My Courses	• The system shall allow users to view enrolled courses and access course details.
8.	Payment System	• The system shall process course fee payments for students, using Stripe.
9.	Learning Materials	• The system shall allow students to view and download learning materials. • The system shall allow instructors to add, edit, and delete learning materials.
10.	Assignments	• The system shall allow instructors to add, edit, and delete assignments. • The system shall allow students to view and upload assignments before due dates.
11.	Marks	• The system shall allow instructors to assign, edit, and delete marks. • The system shall allow students to view their marks.
12.	Classroom System	• The system shall facilitate scheduling and joining Zoom meetings for instructors and students respectively.
13.	Manage Profile	• The system shall allow users to view and edit their profile information.

Table 4 *Non-functional requirements*

No.	Module	Non-functional Requirements
1	Performance	• The system should respond to user interactions (e.g., registration, course selection) within 3 seconds. • Scalability testing should be performed to ensure smooth operation under increased load. • Scheduled maintenance should occur during off-peak hours.
2	Usability	• The user interface should be intuitive and accessible. • The chat function should have a user-friendly design.
3	Reliability	• The system should be able to recover gracefully from unexpected errors. • Clear and user-friendly error messages should be displayed to users.
4	Security	• All sensitive user data, including passwords and payment information, should be encrypted during transmission and storage.
5	Compatibility	• The system should be compatible with the latest versions of popular web browsers (e.g., Chrome, Firefox, Safari). • The user interface should be responsive and functional on various devices, including smartphones and tablets.

3.3 Use Case

Appendix A presents the use case diagram for administrators, students, and instructors. The visual representation encapsulates the system's activities.

For Administrator users, the use cases include logging in, managing profiles, managing courses, managing users, managing payments, downloading administrative reports, adding courses, editing courses, deleting courses, viewing users, editing users, deleting users, and approving payments. For Student users, the use cases encompass logging in, managing profiles, enrolling in courses, paying course fees, managing classrooms, accessing "My Courses," submitting assignments, viewing marks, and downloading learning materials. For Instructor users, the use cases cover logging in, managing profiles, managing classrooms, managing learning materials, managing assignments, managing marks, creating Zoom meetings, adding learning materials, editing learning materials, deleting learning materials, adding assignments, editing assignments, deleting assignments, assigning marks, editing marks, and deleting marks.

3.4 Class Diagram

Appendix B presents the class diagram for the DSG Micro-credential Course Registration System, outlining thirteen classes: Administrator, Student, Instructor, User Registration, Administrative Report, Manage Courses, Manage Users, Manage Payments, Course Registration, My Courses, Payment System, Learning Materials, Assignments, Marks, Classroom System, and Manage Profile. The Administrator class oversees User Registration, Manage Users, Approve Payments, View Administrative Reports, Manage Courses, and Manage Profile. The Student class is connected to User Registration, Course Registration, My Courses, Payment System, Learning Materials, Assignments, Marks, Classroom System, and Manage Profile. The Instructor class is linked to User Registration, My Courses, Learning Materials, Assignments, Marks, Classroom System, and Manage Profile. This class diagram serves as a foundational guide for the system's executable code and overall functionality.

3.5 Relationship Schema

The database schema extracted from the Class Diagram is listed below:

i. Users (UserID, name, email, country_code, contact_number, gender, profile_picture, is_active, role, password, created_at, updated_at)
ii. Courses (CourseID, instructor_ID, name, category, fee, description, duration, created_at, updated_at, start_date, end_date, zoom_meeting_link)
iii. Learning_Materials (MaterialsID, CourseID, title, file_path, created_at, updated_at)
iv. Assignments (AssignmentID, CourseID, title, description, due_date, created_at, updated_at)
v. Marks (MarksID, UserID, AssignmentID, CourseID, mark, created_at, updated_at)
vi. Payments (PaymentID, UserID, CourseID, amount, status, created_at, updated_at, payment_date)

3.6 To-Be Model

The DSG Micro-credential Course Registration System is a system that allows students to enroll in micro-credential courses. The system involves three main actors: Administrator, Student, and Instructor. The Administrator's role includes managing users, managing courses, and managing payments. Students can enroll in courses by viewing the list of available courses and registering. Instructors' role includes managing courses and managing profile. The To-Be Model for the DSG Micro-credential Course Registration System can be found in **Appendix C**.

4. Results and Discussion

This section is mainly about user interface, code segment of the DSG Micro-credential Course Registration System. Result of test cases have been included in this section as well.

4.1 Implementation

The system was executed as planned, using Laravel as the programming framework and MySQL for database management. Visual Studio Code was used to write the source code. The user interface includes modules for User Registration, Administrative Reports, Classroom System, Course Registration, Payment System, and the "My Courses" module. Each user has access to different modules based on their role.

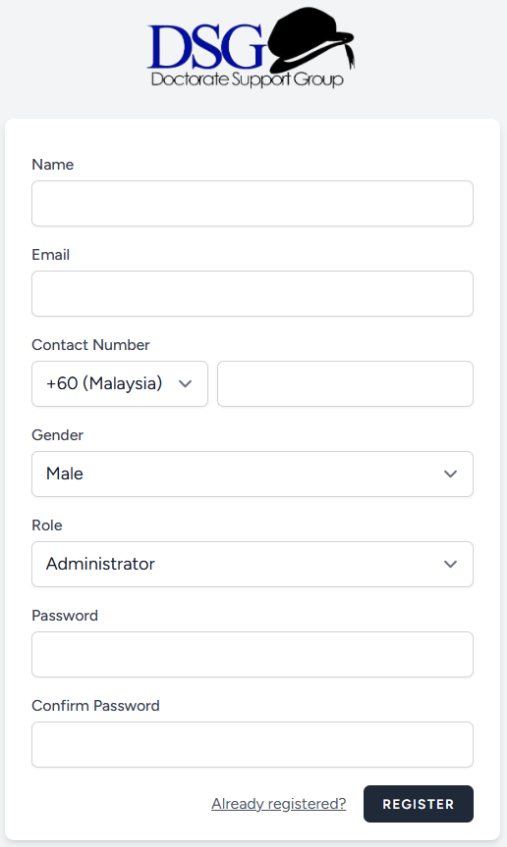


Figure 4(a)

Figure 4(b)

Figure 4(a) shows the registration form interface for the DSG system, which includes fields for Name, Email, Contact Number, Gender, Role, Password, and Confirm Password. This interface allows users to input their information for registration. Figure 4(b) displays the corresponding HTML code for the registration form. The code illustrates the form structure, including input fields with labels, validation messages, and selection options for the country code and gender, ensuring proper data capture and validation during the registration process.

```
<div class="container mt-4">
    <h1 class="h3 mb-4 text-center">Administrator's Dashboard</h1>
    <div class="row mb-4">
        <div class="col-md-4">
            <div class="card text-white bg-primary mb-3 shadow-sm">
                <div class="card-body">
                    <h5 class="card-title">Students</h5>
                    <p class="card-text">
                        <span class="display-4">{{ $studentsCount }}</span><br>
                    </p>
                </div>
            </div>
        </div>
        <div class="col-md-4">
            <div class="card text-white bg-success mb-3 shadow-sm">
                <div class="card-body">
                    <h5 class="card-title">Instructors</h5>
                    <p class="card-text">
                        <span class="display-4">{{ $instructorsCount }}</span><br>
                    </p>
                </div>
            </div>
        </div>
        <div class="col-md-4">
            <div class="card text-white bg-danger mb-3 shadow-sm">
                <div class="card-body">
                    <h5 class="card-title">Total Income</h5>
                    <p class="card-text">
                        RM <span class="display-4">{{ number_format($totalIncome, 2) }}</span><br>
                    </p>
                </div>
            </div>
        </div>
    </div>
    <div class="text-left">
        <button id="downloadReport" class="btn btn-primary mt-3">Download Report as PDF</button>
    </div>
```

Figure 5(a) **Figure 5(b)**

Figure 5(a) displays the Administrator's Dashboard interface, which provides a visual summary of key metrics such as the number of students, instructors, and total income. It also includes a student gender distribution chart and a table of popular courses with the number of enrolled students. Figure 5(b) shows the HTML code for this dashboard, outlining the layout and styling of the dashboard elements. The code includes sections for displaying student and instructor counts, total income with formatting, and a button for downloading the report as a PDF. This setup helps administrators quickly access and manage critical data.

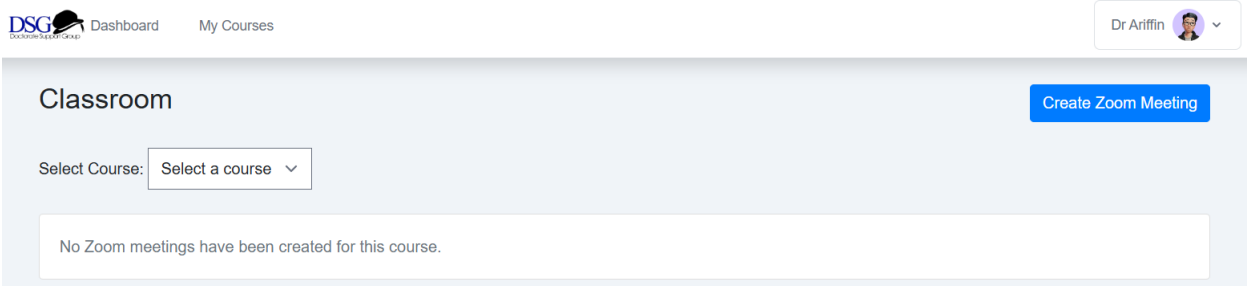


Figure 6(a)

Figure 6(b)

```
<div class="container mt-4">
    <div class="d-flex justify-content-between align-items-center mb-4">
        <h1 class="h3">Classroom</h1>
        @if(Auth::user() && Auth::user()->role === 'instructor')
            <a href="{{ route('classroom.create') }}" class="btn btn-primary">Create Zoom Meeting</a>
        @endif
    </div>

    <div class="mb-4">
        <label for="courseSelect" class="form-label">Select Course:</label>
        <select id="courseSelect" class="form-select" onchange="location = this.value;">
            <option value="">Select a course</option>
            @foreach ($courses as $c)
                <option value="{{ route('classroom.index', ['course' => $c->id]) }}" {{ $c->id == optional($selectedCourse)->id ? 'selected' : '' }}>{{ $c->name }}</option>
            @endforeach
        </select>
    </div>

    @if(session('success'))
        <div class="alert alert-success">
            {{ session('success') }}
        </div>
    @endif

    <div class="card">
        <div class="card-body">
            @if($classrooms->isEmpty())
                <p class="text-muted">No Zoom meetings have been created for this course.</p>
            @else
                <table class="table table-bordered table-striped text-center">
                    <thead>
                        <tr>
                            <th scope="col">Title</th>
                            <th scope="col">Date</th>
                            <th scope="col">Start Time</th>
                            <th scope="col">End Time</th>
                            <th scope="col">Zoom URL</th>
```

Figure 6(c)

Figure 6(a) shows the Classroom page interface for the DSG system, where instructors can select a course and see if any Zoom meetings have been created for it. There's a button to create a new Zoom meeting. Figure 6(b) depicts the interface for creating a new Zoom meeting, allowing the instructor to input details like course, title, date, time, and Zoom URL. Figure 6(c) displays the HTML code for the Classroom page, which includes logic to display the "Create Zoom Meeting" button for instructors, a dropdown for selecting a course, and a table that lists any existing Zoom meetings for the selected course.

Figure 7(a)

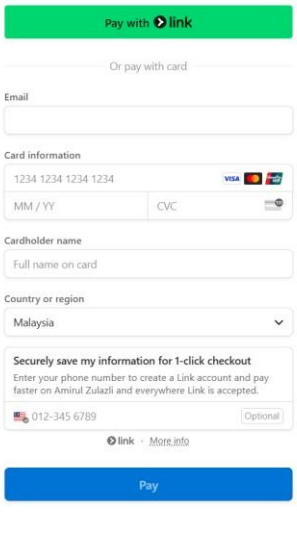
```
<div class="container mt-4">
    <p>Welcome to the course registration page! Below is the list of available courses for you to enroll in.</p>

    <div class="mb-3">
        @if(Auth::user() && Auth::user()->role === 'administrator')
            <a href="{{ route('course.create') }}" class="btn btn-success">Add New Course</a>
        @endif
    </div>

    <table class="table table-striped">
        <thead>
            <tr>
                <th scope="col">Course Name</th>
                <th scope="col">Instructor Name</th>
                <th scope="col">Category</th>
                <th scope="col">Fee (RM)</th>
                <th scope="col">Duration (hours)</th>
                <th scope="col">Description</th>
                @if(Auth::user() && Auth::user()->role === 'administrator')
                    <th scope="col">Action</th>
                @else
                    <th scope="col">Enroll</th>
                @endif
            </tr>
        </thead>
        <tbody>
            @foreach($courses as $course)
                <tr>
                    <td>{{ $course->name }}</td>
                    <td>{{ $course->instructor->name }}</td>
                    <td>{{ $course->category }}</td>
                    <td>{{ number_format($course->fee, 2) }}</td>
                    <td>{{ $course->duration }}</td>
                    <td>{{ $course->description }}</td>
                    @if(Auth::user() && Auth::user()->role === 'administrator')
                        <td>
                            <div class="btn-group" role="group">
```

Figure 7(b)

Figure 7(a) displays the Course Registration page interface, listing available courses with details such as course name, instructor name, category, fee, duration, and description, along with an "Enroll" button for students. Administrators see an additional button to add new courses. Figure 7(b) shows the HTML code for this page, outlining the table structure and dynamically displaying courses using a foreach loop. Conditional statements handle the display of the "Add New Course" button for administrators and the "Enroll" button for students.




```
<div class="container mt-5">
    <h2 class="text-center">Stripe Payment</h2>
    <button id="checkout-button" class="btn btn-primary">Pay Now</button>
</div>
@endsection

@section('scripts')
<script src="https://js.stripe.com/v3/"></script>
<script>
    var stripe = Stripe('{{ env('STRIPE_KEY') }}');

    var checkoutButton = document.getElementById('checkout-button');

    checkoutButton.addEventListener('click', function () {
        fetch("{{ route('create.checkout.session') }}", {
            method: "POST",
            headers: {
                "Content-Type": "application/json",
                "X-CSRF-TOKEN": "{{ csrf_token() }}"
            },
            body: JSON.stringify({
                amount: {{ $course->fee }},
                course_name: "{{ $course->name }}"
            })
        })
        .then(function (response) {
            return response.json();
        })
        .then(function (session) {
            return stripe.redirectToCheckout({ sessionId: session.id });
        })
        .then(function (result) {
            if (result.error) {
                alert(result.error.message);
            }
        })
        .catch(function (error) {
```

Figure 8(b) **Figure 8(a)**

Figure 8(a) shows the Stripe payment interface for the DSG system, where users can pay for a course, such as TESL, priced at MYR 900.00. The form includes fields for email, card information, cardholder name, and country, with an option to save information for future payments. Figure 8(b) displays the corresponding HTML and JavaScript code for the payment process. The code sets up a button to initiate the Stripe checkout process, capturing the course fee and name. When clicked, it sends a POST request to create a checkout session and redirects the user to the Stripe checkout page.

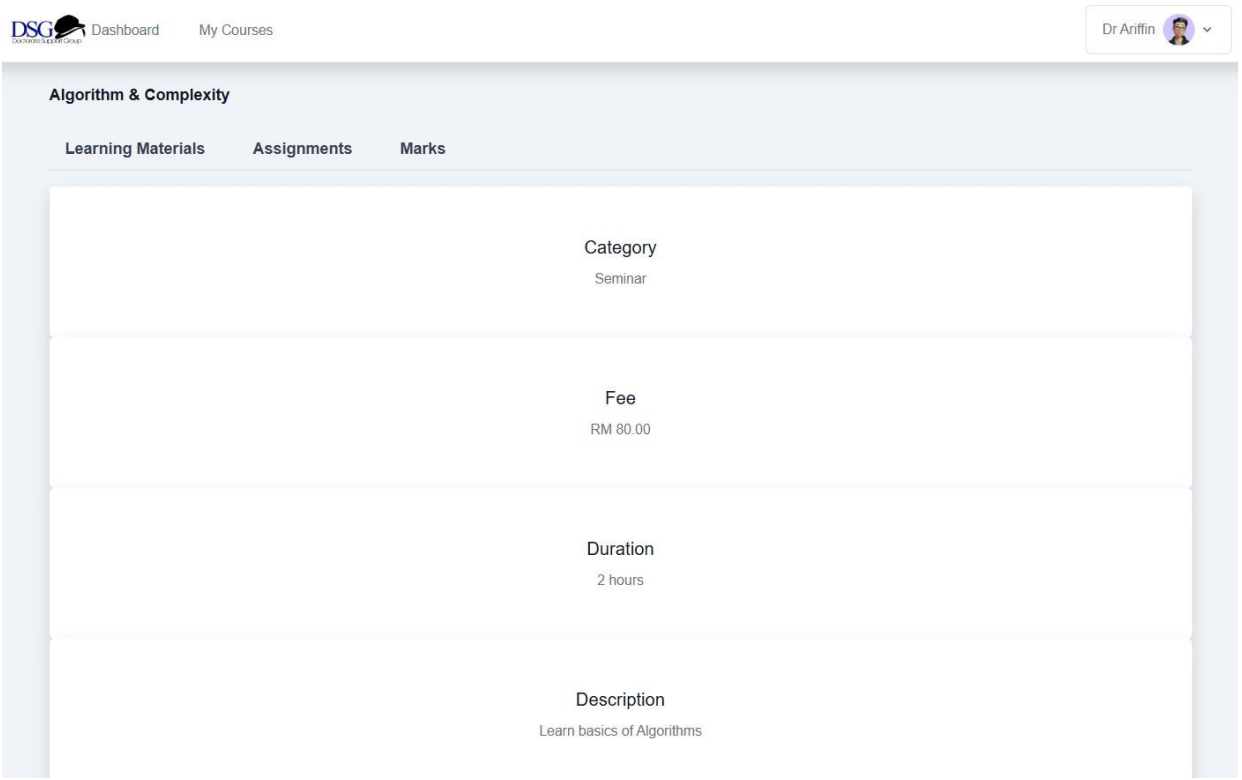


Figure 9(a)

```
@section('content')
<div class="container mx-auto py-6 px-4 sm:px-6 lg:px-8">
    <h1 class="text-3xl font-semibold text-gray-900 mb-6">{{ $course->name }}</h1>
    <div class="tabs">
        <a class="tab {{ request()->is('courses/'.$course->id.'/learning-materials') ? 'tab-active' : '' }}" href="{{ route('learning_materials.index', $course->id) }}"
        >Learning Materials</a>
        <a class="tab {{ request()->is('courses/'.$course->id.'/assignments') ? 'tab-active' : '' }}" href="{{ route('assignments.index', $course->id) }}">Assignments</a>
        <a class="tab {{ request()->is('courses/'.$course->id.'/marks') ? 'tab-active' : '' }}" href="{{ route('marks.index', $course->id) }}">Marks</a>
    </div>
    <div class="grid grid-cols-1 gap-5 sm:grid-cols-2 lg:grid-cols-4 mb-6">
        <div class="bg-white overflow-hidden shadow rounded-lg">
            <div class="p-5 text-center">
                <div class="text-lg font-medium text-gray-900">Category</div>
                <div class="mt-2 text-sm text-gray-500">{{ $course->category }}</div>
            </div>
        </div>
        <div class="bg-white overflow-hidden shadow rounded-lg">
            <div class="p-5 text-center">
                <div class="text-lg font-medium text-gray-900">Fee</div>
                <div class="mt-2 text-sm text-gray-500">RM {{ number_format($course->fee, 2) }}</div>
            </div>
        </div>
        <div class="bg-white overflow-hidden shadow rounded-lg">
            <div class="p-5 text-center">
                <div class="text-lg font-medium text-gray-900">Duration</div>
                <div class="mt-2 text-sm text-gray-500">{{ $course->duration }} hours</div>
            </div>
        </div>
        <div class="bg-white overflow-hidden shadow rounded-lg col-span-1 sm:col-span-2 lg:col-span-4">
            <div class="p-5 text-center">
                <div class="text-lg font-medium text-gray-900">Description</div>
                <div class="mt-2 text-sm text-gray-500">{{ $course->description }}</div>
            </div>
        </div>
    </div>
```

Figure 9(b)

Figure 9(a) shows the course details page for "Algorithm & Complexity" in the DSG system, providing tabs for Learning Materials, Assignments, and Marks. The page displays the course category, fee, duration, and description. Figure 9(b) contains the HTML code for this page, detailing the layout and structure. It includes navigation tabs with conditional active states for different sections (Learning Materials, Assignments, Marks) and uses grid classes to display course information. The code ensures the dynamic display of course details, enhancing the user experience with a well-organized interface.

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 1165-1184

4.2 Testing Phase

User Acceptance Testing is a crucial stage in software development where end-users or stakeholders assess the system's usability, functionality, and overall suitability for their needs. It is conducted to verify that the software meets the requirements and expectations of the intended users. The functionality testing has been done as shown in Table 11.

Table 11 List of Test Cases

No.	Test Cases	Description	Expected Results	Status
	TEST_100	**User Registration**		
1.	TEST_100_001	Admin clicks the signup button after entering all required information	The application should prompt a registration successful message on the screen	PASS
2.	TEST_100_002	Admin registers using an email address that already exists	The application should prompt an invalid email message on the screen	PASS
3.	TEST_100_003	Student clicks the signup button after entering all required information	The application should prompt a registration successful message on the screen	PASS
4.	TEST_100_004	Student registers using an email address that already exists	The system should prompt an invalid email message on the screen	PASS
5.	TEST_100_005	Instructor clicks the signup button after entering all required information	The system should prompt a registration successful message on the screen	PASS
6.	TEST_100_006	Instructor registers using an email address that already exists	The system should prompt an invalid email message on the screen	PASS
	TEST_200	**User Login**		
7.	TEST_200_001	Admin logs in with correct credentials	The system should redirect to the admin dashboard	PASS
8.	TEST_200_002	Student logs in with correct credentials	The system should redirect to the student dashboard	PASS
9.	TEST_200_003	Instructor logs in with correct credentials	The system should redirect to the instructor dashboard	PASS
	TEST_300	**Administrative Report**		
10.	TEST_300_001	Admin views and downloads the administrative report	The system should generate and download the report as a PDF	PASS
	TEST_400	**Manage Courses**		
11.	TEST_400_001	Admin views list of courses	The system should display all available courses	PASS
12.	TEST_400_002	Admin adds a new course	The system should display the new course in the list of courses	PASS
13.	TEST_400_003	Admin edits an existing course	The system should update the course details	PASS
14.	TEST_400_004	Admin deletes a course	The system should remove the course from the list of courses	PASS
	TEST_500	**Manage Users**		
15.	TEST_500_001	Admin views list of users	The system should display all registered users	PASS
16.	TEST_500_002	Admin edits user information	The system should update the user's information	PASS
17.	TEST_500_003	Admin deletes a user	The system should remove the user from the list	PASS
	TEST_600	**Manage Payments**		
18.	TEST_600_001	Admin views list of payments	The system should display all course fee payments	PASS
19.	TEST_600_002	Admin approves a payment	The system should mark the payment as approved	PASS
	TEST_700	**Course Registration**		

20.	TEST_700_001	Student views list of available courses	The system should display all available courses	PASS
21.	TEST_700_002	Student enrolls in a course	The system should register the student for the course and prompt payment	PASS
	TEST_800	**My Courses**		
22.	TEST_800_001	Student views list of enrolled courses	The system should display all courses the student is enrolled in	PASS
23.	TEST_800_002	Student views course details	The system should display details of the selected course	PASS
	TEST_900	**Payment System**		
24.	TEST_900_001	Student makes a payment	The system should process the payment and update the status	PASS
	TEST_1000	**Learning Materials**		
25.	TEST_1000_001	Instructor views list of learning materials	The system should display all learning materials for the course	PASS
26.	TEST_1000_002	Instructor adds a new learning material	The system should display the new learning material in the list	PASS
27.	TEST_1000_003	Instructor edits an existing learning material	The system should update the learning material details	PASS
28.	TEST_1000_004	Instructor deletes a learning material	The system should remove the learning material from the list	PASS
29.	TEST_1000_005	Student views list of learning materials	The system should display all learning materials for the course	PASS
30.	TEST_1000_006	Student downloads learning material	The system should download the selected learning material	PASS
	TEST_1100	**Assignments**		
31.	TEST_1100_001	Instructor adds a new assignment	The system should display the new assignment in the list	PASS
32.	TEST_1100_002	Instructor edits an existing assignment	The system should update the assignment details	PASS
33.	TEST_1100_003	Instructor deletes an assignment	The system should remove the assignment from the list	PASS
34.	TEST_1100_004	Student views list of assignments	The system should display all assignments for the course	PASS
35.	TEST_1100_005	Student uploads an assignment before the due date	The system should upload the assignment and confirm successful submission	PASS
	TEST_1200	**Marks**		
36.	TEST_1200_001	Instructor assigns marks	The system should record and display the marks	PASS
37.	TEST_1200_002	Instructor edits marks	The system should update the marks	PASS
38.	TEST_1200_003	Instructor deletes marks	The system should remove the marks	PASS
39.	TEST_1200_004	Student views marks	The system should display the marks for the student	PASS
	TEST_1300	**Classroom System**		
40.	TEST_1300_001	Student joins a Zoom meeting	The system should connect the student to the Zoom meeting	PASS
41.	TEST_1300_002	Instructor creates a Zoom meeting	The system should schedule and provide the meeting link	PASS
	TEST_1400	**Manage Profile**		
42.	TEST_1400_001	Admin views and edits profile	The system should display and update the admin's profile information	PASS
43.	TEST_1400_002	Student views and edits profile	The system should display and update the student's profile information	PASS

44.	TEST_1400_003	Instructor views and edits profile	The system should display and update the instructor's profile information	PASS

4.3 User Acceptance Test (students and instructors)

User Acceptance Testing (UAT) is an essential phase in the software development process. It involves end users or their representatives conducting hands-on testing to evaluate usability, functionality, and the overall user experience. UAT confirms that the software aligns with the expectations and needs of its intended audience. This stage helps identify any potential issues and fosters stakeholder confidence, serving as the final validation before the software is deployed.

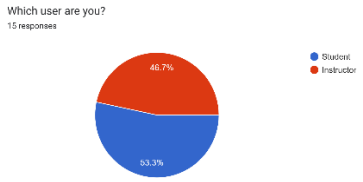


Figure 10(a)

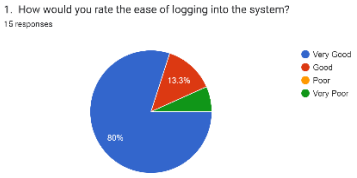


Figure 10(b)

In Figure 10(a), most respondents are students (53.3%) while the rest are instructors (46.7%). In Figure 10(b), most users rate the ease of logging into the system as "Very Good" (80%), with a smaller percentage rating it as "Good" (13.3%), and a minority finding it "Poor" (6.7%).

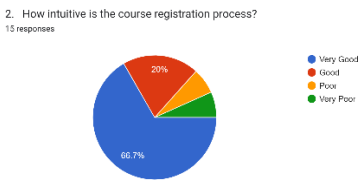


Figure 10(c)

Figure 10(d)

In Figure 10(c), a significant majority of respondents find the course registration process to be very intuitive, with 66.7% rating it as "Very Good" and 20% as "Good". In Figure 10(d), over half of the respondents (53.3%) rate the "My Courses" tab as "Good" in helping manage courses, while 40% rate it as "Very Good".

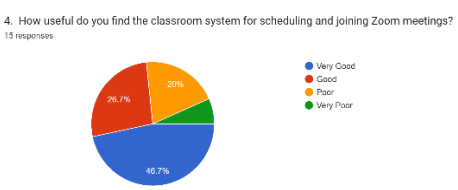


Figure 10(e)

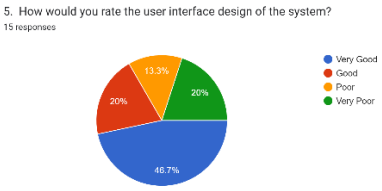


Figure 10(f)

In Figure 10(e), nearly half of the respondents (46.7%) find the classroom system very useful for scheduling and joining Zoom meetings, with 26.7% rating it as good. In Figure 10(f), the user interface design of the system is rated as very good by 46.7% of respondents, and 20% consider it good.

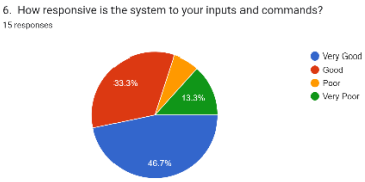


Figure 10(g)

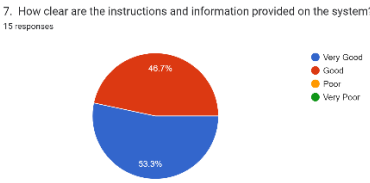


Figure 10(h)

In Figure 10(g), nearly half of the respondents (46.7%) rate the system's responsiveness to inputs and commands as very good, while 33.3% consider it good. In Figure 10(h), a majority of respondents (53.3%) find the instructions and information provided on the system very clear, with 46.7% rating them as good.

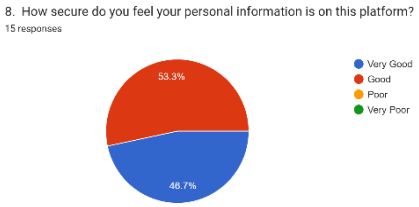


Figure 10(i)

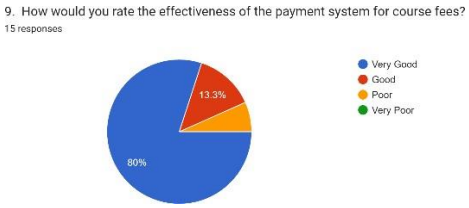


Figure 10(j)

In Figure 10(i), most respondents (53.3%) feel that their personal information is secure on the platform, rating it as good, while 46.7% rate it as very good. In Figure 10(j), a substantial majority (80%) rate the effectiveness of the payment system for course fees as very good, with 13.3% considering it good.

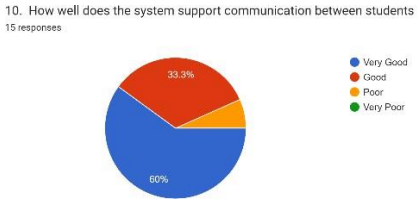


Figure 10(k)

In Figure 10(k), a majority of respondents (60%) rate the system's support for communication between students and instructors as very good, while 33.3% consider it good.

4.4 User Acceptance Test (administrator)

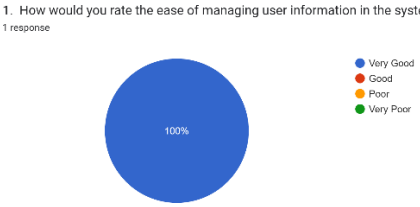


Figure 11(a)

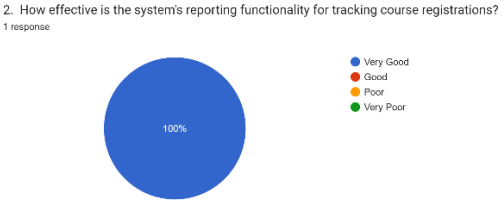


Figure 11(b)

In Figure 11(a), the single respondent rates the ease of managing user information in the system as very good (100%). Similarly, in Figure 11(b), the same respondent rates the effectiveness of the system's reporting functionality for tracking course registrations as very good (100%).

Figure 11(c)

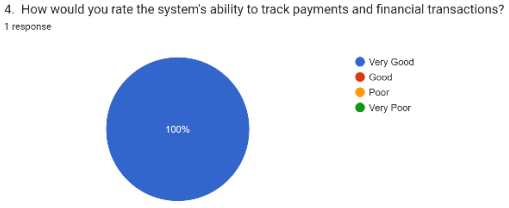


Figure 11(d)

In Figure 11(c), the single respondent rates the intuitiveness of the interface for managing courses and their details as very good (100%). Similarly, in Figure 11(d), the same respondent rates the system's ability to track payments and financial transactions as very good (100%).

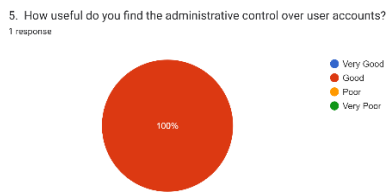


Figure 11(e)

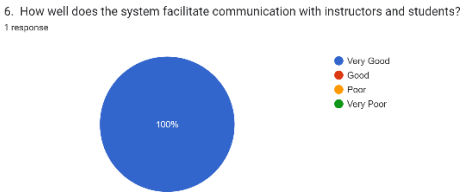


Figure 11(f)

In Figure 11(e), the single respondent finds the administrative control over user accounts to be very useful (100%). Similarly, in Figure 11(f), the same respondent rates the system's facilitation of communication with instructors and students as very good (100%).

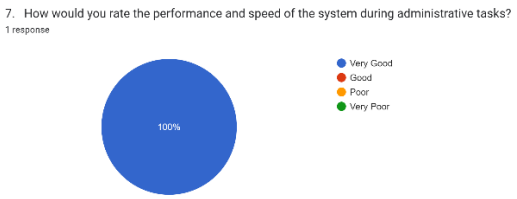


Figure 11(g)

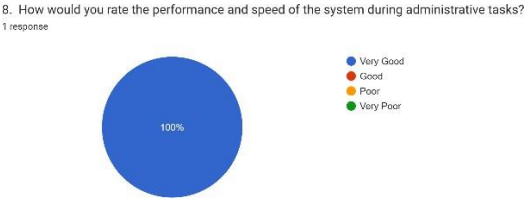


Figure 11(h)

In Figure 11(e), the single respondent finds the administrative control over user accounts to be very useful (100%). Similarly, in Figure 11(f), the same respondent rates the system's facilitation of communication with instructors and students as very good (100%).

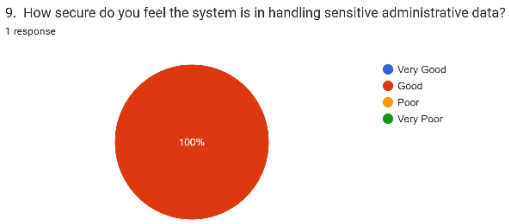


Figure 11(i)

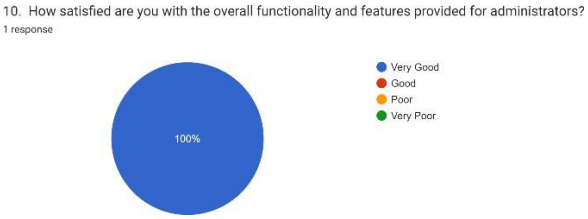


Figure 11(j)

In Figure 11(i), the single respondent feels that the system is very secure in handling sensitive administrative data (100%). Similarly, in Figure 11(j), the same respondent is very satisfied with the overall functionality and features provided for administrators (100%).

5. Conclusion

In conclusion, the DSG Micro-credential Course Registration System project successfully addressed the critical challenges faced by the Doctorate Support Group (DSG), including the lack of a standardized registration system, difficulties in tracking registered users, and inefficiencies of the manual registration process. The project achieved its primary objectives by delivering a robust, web-based platform that standardizes the registration process, provides efficient user tracking, and automates the registration workflow. The system offers several advantages, such as a consistent and user-friendly registration process, effective user tracking, and reduced administrative burdens through automation. However, it also has some limitations, including a learning curve for users, ongoing maintenance requirements, and significant initial setup costs. To further enhance the system, recommendations include integrating user feedback, developing a mobile application, and implementing advanced security measures. Overall, the project has significantly improved the efficiency and user experience of DSG's micro-credential programs, laying a strong foundation for future growth and success.

Acknowledgement

The authors express their gratitude to the Faculty of Computer Science and Information Technology at Universiti Tun Hussein Onn Malaysia for the support and encouragement extended to them.

Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

The authors acknowledge their contributions to the paper as follows: study conception and design: Amirul Hafiz Bin Zulazli, Nur Ariffin bin Mohd Zin; data collection: Amirul Hafiz Bin Zulazli; analysis and interpretation of results: Amirul Hafiz Bin Zulazli, Nur Ariffin bin Mohd Zin; draft manuscript preparation: Amirul Hafiz Bin Zulazli. All authors participated in the review of the results and approved the final version of the manuscript.

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 1165-1184

Appendix A

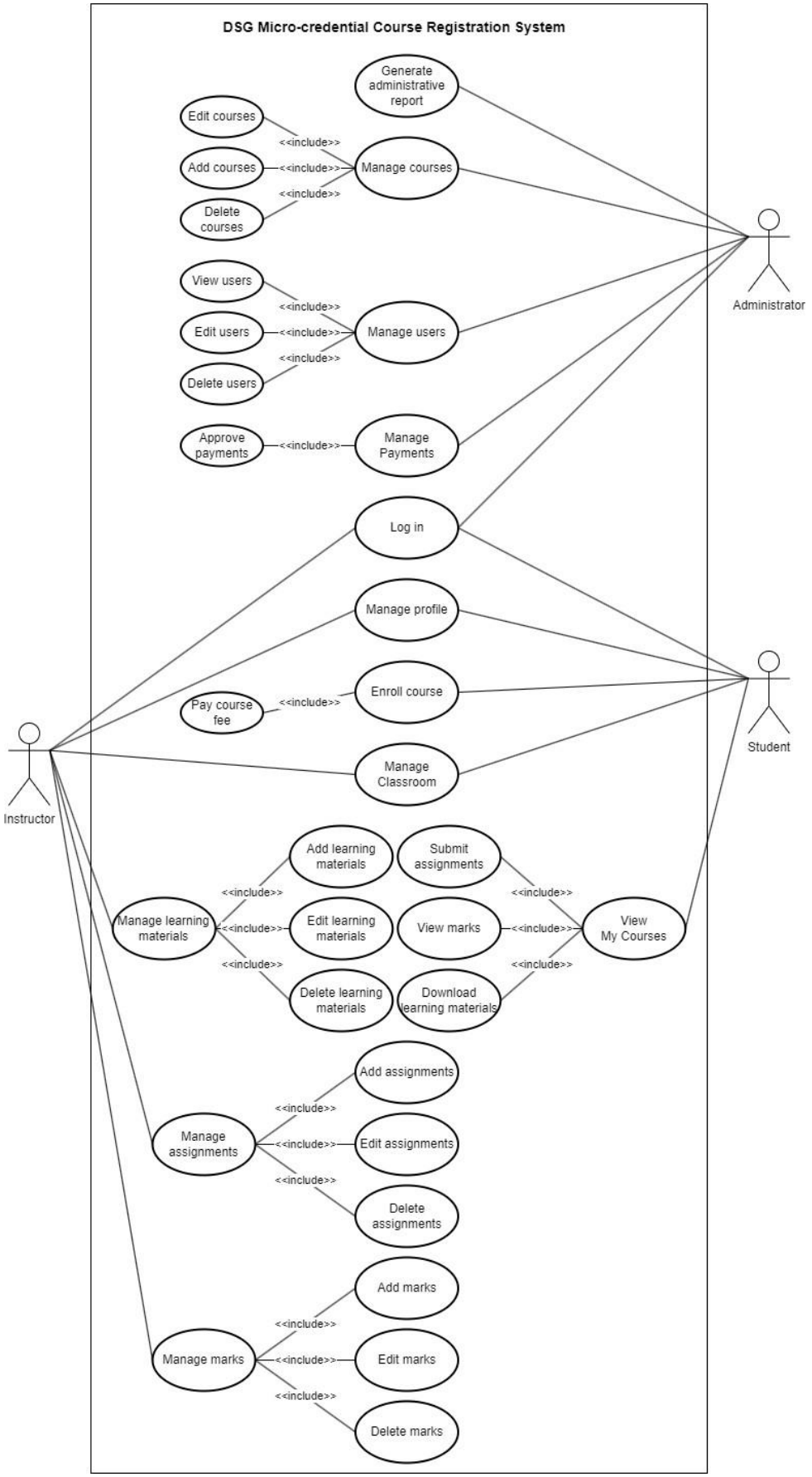


Figure 2 Use Case diagram for DSG Micro-credential Course Registration System

Appendix B

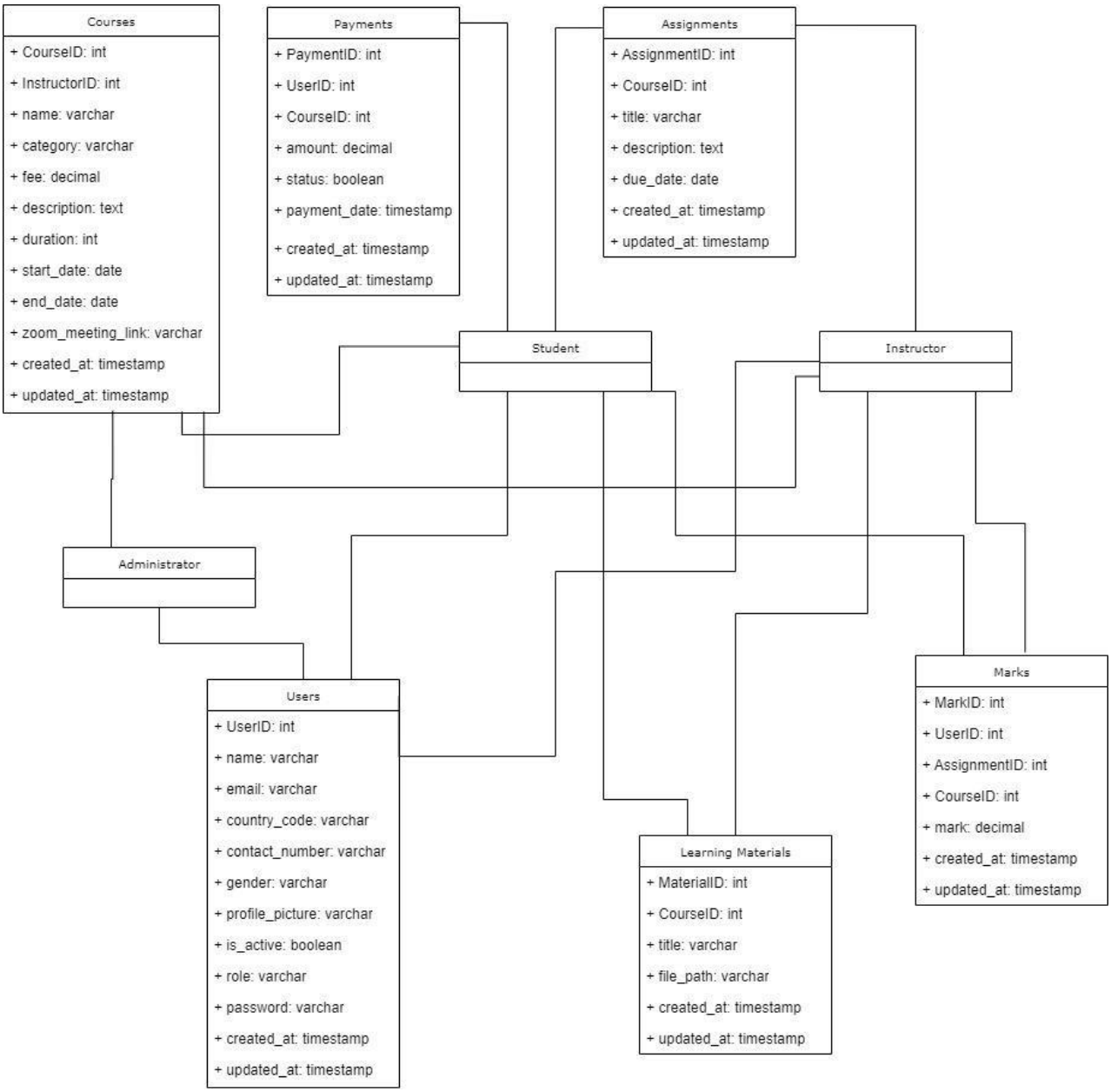


Figure 3 Class diagram for DSG Micro-credential Course Registration System

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 1165-1184

Appendix C

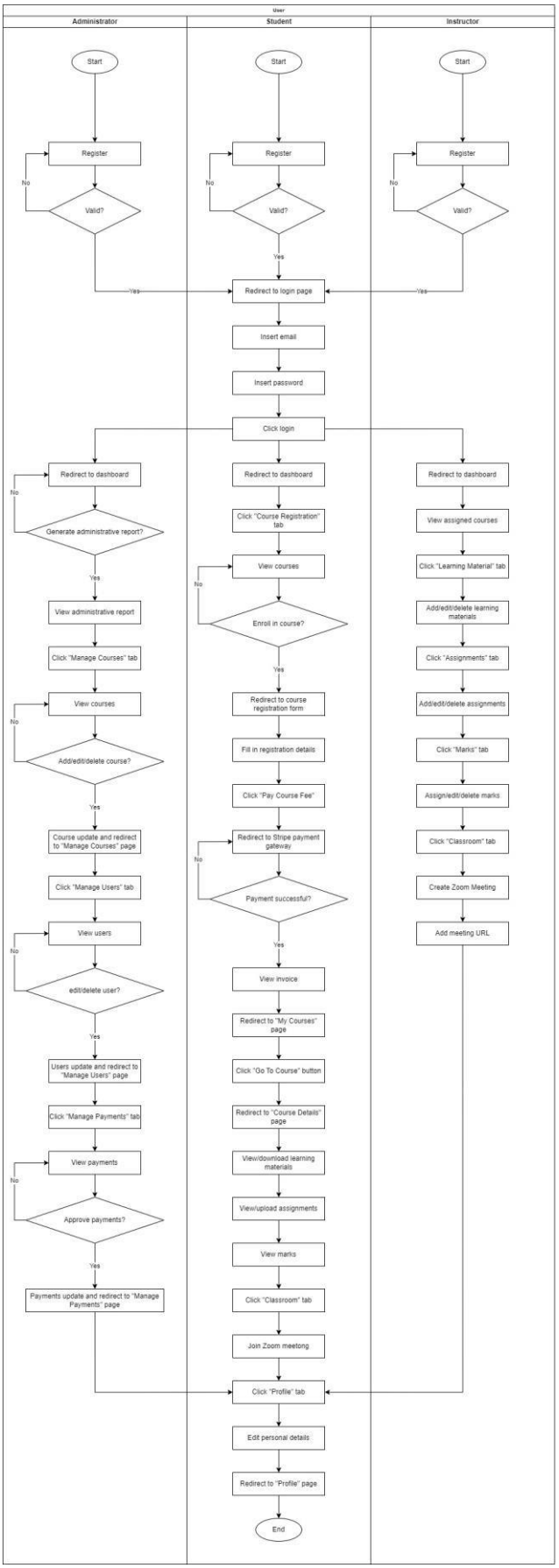


Figure 4 Class diagram for DSG Micro-credential Course Registration System

References

[1] Acree, L., "Seven lessons learned from implementing micro-credentials," Friday Institute for Educational Innovation at the NC State University College of Education, 2016.

[2] Ala'a, M., "Online registration system," *International Journal of Computer Science and Security (IJCSS)*, vol. 4, no. 3, pp. 331, 2010.

[3] Brown, M., Nic Giolla Mhichíl, M., Beirne, E., and Mac Lochlainn, C., "The global micro-credential landscape: Charting a new credential ecology for lifelong learning," 2021.

[4] Ehlers, U. D., "Higher creduation–Degree or education? The rise of Microcredentials and its consequences for the university of the future," in *European Distance and E-Learning Network (EDEN) Conference Proceedings*, no. 1, pp. 456-465, European Distance and E-Learning Network, 2018.

[5] Estevez, R., Rankin, S., and Silva, R., "A model for web-based course registration systems," *International Journal of Web Information Systems*, vol. 10, no. 1, pp. 51-64, 2014.

[6] Fischer, T., Oppl, S., and Stabauer, M., "Micro-credential development: tools, methods and concepts supporting the European approach," 2022.

[7] Gish-Lieberman, J. J., Tawfik, A., and Gatewood, J., "Micro-credentials and badges in education: A historical overview," *TechTrends*, vol. 65, pp. 5-7, 2021.

[8] Lemoine, P. A., and Richardson, M. D., "Micro-credentials, nano degrees, and digital badges: New credentials for global higher education," *International Journal of Technology and Educational Marketing (IJTEM)*, vol. 5, no. 1, pp. 36-49, 2015.

[9] Li Y., "Add-on Course Registration System," Doctoral dissertation, Western Oregon University, 2017.

[10] Moore, R. L., "Introducing mesocredentials: Connecting MOOC achievement with academic credit," *Distance Education*, vol. 43, no. 2, pp. 271-289, 2022.

[11] Peisachovich, E. H., Dubrowski, A., Da Silva, C., Kapralos, B., Klein, J. E., and Rahmanov, Z., "Using simulation-based methods to support demonstration of competencies required by micro-credential courses," *Cureus*, vol. 13, no. 8, 2021.

[12] Peng, Y., Liu, N., Li, Y., and Shao, Z., "Design and implementation of the online course registration system at Tsinghua University," in *2012 International Conference on Systems and Informatics (ICSAI2012) *, pp. 1179-1182, IEEE, May 2012.

[13] van der Hijden, P., and Martin, M., "Short courses, micro-credentials, and flexible learning pathways: A blueprint for policy development and action," *International Institute for Educational Planning*, 2023.

[14] Wheelahan, L., and Moodie, G., "Analysing micro-credentials in higher education: a Bernsteinian analysis," *Journal of Curriculum Studies*, vol. 53, no. 2, pp. 212-228, 2021.

[15] Zou, H., Ullah, A., Qazi, Z., Naeem, A., and Rehan, S., "Impact of micro-credential learning on students' perceived employability: the mediating role of human capital," *International Journal of Educational Management*, 2023.