



DIVINE WORD COLLEGE OF CALAPAN
SCHOOL OF INFORMATION & TECHNOLOGY
PREFINAL EXAMINATION
(SYSTEM INTEGRATION AND ARCHITECTURE 1)

NAME: _____

SCORE: _____

RATING: _____

ID NUMBER: _____

COURSE: _____

HANDS-ON (50 points)

Directions:

Build a web application in Laravel that allows the school registrar to manage student records through a simple CRUD (Create, Read, Update, Delete) interface. The application must follow Laravel’s MVC (Model-View-Controller) architecture and use MySQL as the database.

Scenario:

You have been hired by a college registrar’s office to create a **Student Management System** that can help staff organize and maintain student information. The system should allow authorized users to **add, view, edit, and delete** student records in the database.

Each student record should include the following attributes:

- Student ID
- Full Name
- Course
- Email Address
- Contact Number

Application Requirements:

1. Install Laravel using **Composer(cmd)** and set up the project folder named **student_management**.
2. Configure your .env file and connect it to a local MySQL database named **student_db**.
3. Create a **Model** and **Migration** for the students table with the attributes listed above.
4. Create a **Controller** named StudentController that handles all CRUD operations.
5. Set up **Routes** in web.php for each action (index, create, store, edit, update, destroy).
6. Create **Blade views** for:
 - Displaying all students (Index page)
 - Adding a new student (Create page)
 - Editing existing student details (Edit page)
7. Use **Bootstrap** for simple UI styling and form layout.
8. Implement basic **validation** for form inputs.
9. Display success messages upon completing operations (e.g., “Student added successfully”).

SUBMISSION INSTRUCTIONS:

- Submit all deliverables (**screenshots, Laravel project folder, and database export**) in a single **PDF or ZIP file**.
- Use the filename format:
PREFINAL_SURNAME
- Upload your submission through the **Learning Management System (LMS)**.
- Ensure your submission includes:
 - Working project folder (student_management)
 - Exported database file (student_db.sql)
 - Screenshot proofs of working CRUD functions

REMINDER:

Any form of **cheating, plagiarism, or unauthorized assistance**, including the use of **AI tools** to generate code or perform exam tasks, will result in an **automatic zero** for this exam.

You are expected to complete the exam independently and demonstrate your genuine understanding of the topic.

EVALUATION CRITERIA:

| Criteria | Excellent (10) | Good (8) | Fair (6) | Poor (4) | Not Done (0) | Score |
|---------------------------------|---|--|--|---|--------------|-------|
| Method Implementation | All CRUD operations are fully functional and perform as expected. | CRUD operations mostly functional with minor issues. | CRUD operations partially functional or have issues. | CRUD not functional or incomplete. | | |
| Class and Properties | Follows MVC design and Laravel best practices. | Mostly correct with minor issues. | Some missing properties or structural issues. | Incorrect or incomplete implementation. | | |
| Code Cleanliness & Organization | Code is well-organized, readable, and follows conventions. | Minor formatting or naming issues. | Functional but unorganized or inconsistent. | Poorly structured code, missing comments. | | |
| Database Connectivity | Correct .env setup, migration works perfectly. | Database connected with minor issues. | Requires manual fixes. | Not connected or migration fails. | | |
| UI Design | Clean, intuitive, and user-friendly interface. | Functional but with minor UI flaws. | Usable but lacks design consistency. | Confusing or poorly styled UI. | | |