Final Case Study

ITEW6 – Web Development Framework

NAME: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Score: \_\_\_\_\_\_\_\_\_\_\_\_

STUDENT NUMBER: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_\_\_\_\_

NAME OF FACULTY: JOSEPH D. CARTAGENAS

**Use Case: Hospital Management System**

**Objective:**

To create a Hospital Management System where students can practice developing a web application using Laravel for the backend API and Vue.js for the frontend. This system will manage patients, doctors, appointments, and medical records.

**Use Case Description:**

**Title:** Hospital Management System

**Actors:**

* **Admin:** Manages the system, doctors, and patients.
* **Doctor:** Views patient records and manages appointments.
* **Patient:** Books appointments and views their medical records.

**Functional Requirements:**

1. **User Authentication:**
   * **Admin:** Can add, edit, and delete doctors and patients.
   * **Doctor:** Can view and update patient records, and manage their own appointments.
   * **Patient:** Can register, book appointments, and view their own medical records.
2. **Patient Management:**
   * **Admin:** Can add, edit, and delete patient records.
   * **Doctor:** Can view and update patient records.
   * **Patient:** Can view their own records.
3. **Doctor Management:**
   * **Admin:** Can add, edit, and delete doctor records.
   * **Doctor:** Can update their own profile and view their appointments.
4. **Appointment Management:**
   * **Admin:** Can view all appointments.
   * **Doctor:** Can manage their own appointments.
   * **Patient:** Can book, view, and cancel their appointments.
5. **Medical Records:**
   * **Admin:** Can view all medical records.
   * **Doctor:** Can update patient medical records.
   * **Patient:** Can view their own medical records.

**Technology Stack:**

* **Backend:** Laravel
* **Frontend:** Vue.js
* **Database:** MySQL
* **Authentication:** built in Laravel

**Implementation Steps:**

**1. Setting Up Laravel API:**

* **Step 1:** Install Laravel.
* **Step 2:** Set up database connection.
* **Step 3:** Create models and migrations for Users, Patients, Doctors, Appointments, and Medical Records.
* **Step 4:** Implement authentication using Laravel.
* **Step 5:** Create API routes and controllers for managing patients, doctors, appointments, and medical records.
* **Step 6:** Set up appropriate relationships between models (e.g., a doctor has many appointments).

**2. Setting Up Vue.js Frontend:**

* **Step 1:** Create a new Vue.js project.
* **Step 2:** Install Vue Router and Vuex for state management.
* **Step 3:** Create views and components for login, dashboard, patient management, doctor management, and appointment management.
* **Step 4:** Implement API calls using Axios to communicate with the Laravel backend.
* **Step 5:** Set up authentication handling in Vue.js.

**3. Implementing User Authentication:**

* **Admin:** Create an interface for managing doctors and patients.
* **Doctor:** Create a dashboard to view and manage their appointments and patient records.
* **Patient:** Create a user interface for booking appointments and viewing medical records.

**Example API Endpoints:**

* **Authentication:**
  + POST /api/register - Register a new user
  + POST /api/login - Login a user
  + POST /api/logout - Logout a user
* **Patients:**
  + GET /api/patients - Get all patients
  + POST /api/patients - Create a new patient
  + GET /api/patients/{id} - Get a single patient
  + PUT /api/patients/{id} - Update a patient
  + DELETE /api/patients/{id} - Delete a patient
* **Doctors:**
  + GET /api/doctors - Get all doctors
  + POST /api/doctors - Create a new doctor
  + GET /api/doctors/{id} - Get a single doctor
  + PUT /api/doctors/{id} - Update a doctor
  + DELETE /api/doctors/{id} - Delete a doctor
* **Appointments:**
  + GET /api/appointments - Get all appointments
  + POST /api/appointments - Create a new appointment
  + GET /api/appointments/{id} - Get a single appointment
  + PUT /api/appointments/{id} - Update an appointment
  + DELETE /api/appointments/{id} - Cancel an appointment
* **Medical Records:**
  + GET /api/records - Get all medical records
  + POST /api/records - Create a new medical record
  + GET /api/records/{id} - Get a single medical record
  + PUT /api/records/{id} - Update a medical record
  + DELETE /api/records/{id} - Delete a medical record

**Example Vue.js Components:**

* **Login.vue:** Component for user login.
* **Dashboard.vue:** Dashboard component for admin, doctors, and patients.
* **Patients.vue:** Component to manage patients.
* **Doctors.vue:** Component to manage doctors.
* **Appointments.vue:** Component to manage appointments.
* **MedicalRecords.vue:** Component to manage medical records.