**Laboratory Management and Booking System**

Here is a **System Specification Document** for a **Laboratory Management and Booking System** at the **University of Tripoli**.

**📘 System Title:**

**University of Tripoli Laboratory Management and Booking System**

**🧭 System Objective:**

To provide a centralized, accessible platform for managing and booking laboratories and their instruments across all campuses of the University of Tripoli. The system aims to:

* Maintain accurate records of all labs and instruments/tools.
* Allow students and staff to view lab resources and book instruments.
* Enable administrators to manage instruments and view usage reports.

**🏗️ System Modules:**

**1. Laboratory Management**

Each laboratory record should include:

* Campus Number (e.g., 1, 2, 3...)
* Building Name (e.g., Engineering Block A)
* Room Number (e.g., B101)
* Description (textual description of the lab and its purpose)

**2. Instrument/Tool Inventory**

Each lab can contain multiple instruments/tools, with the following information stored:

* Instrument Name
* Purpose
* Description
* Serial Number
* Model
* Types of Experiments it supports
* Type of Analysis it can perform
* Operational Status (e.g., Active, Under Maintenance, Out of Order)
* Required Materials (e.g., Chemicals, Samples)
* Assigned Responsible Person (Name of the lab manager/technician)

**3. Booking Management**

All authenticated **students and staff** can:

* View lab details and instrument listings.
* View real-time booking availability (booking calendar/schedule).
* Book a tool/instrument by selecting a time slot and providing the purpose/experiment details.
* Receive notifications (email/system alert) confirming booking status.

Constraints:

* No overlapping bookings.
* Responsible person approval may be required for certain instruments.

**4. User Roles and Permissions**

| **Role** | **Permissions** |
| --- | --- |
| **Student/Staff** | - View labs and instruments- Book instruments- View booking calendar |
| **Admin** | - Add/Edit/Delete labs and instruments- Move instruments between labs- View usage reports- Approve special bookings (if needed) |

**5. Reporting**

Admin should be able to generate/view reports such as:

* List of all labs by campus/building
* Instrument usage statistics (most/least used)
* Bookings by user or by date range
* Instruments needing maintenance
* Inventory of instruments by lab

**🧑‍💼 Actors**

* **University Admins**
* **Lab Technicians/Managers**
* **Academic Staff**
* **Students**

**🖥️ System Features Summary**

| **Feature** | **Description** |
| --- | --- |
| Lab Directory | Search and browse all labs by campus, building, or room |
| Instrument Listing | View instruments and their specs per lab |
| Booking System | Calendar-based booking interface for users |
| Booking Schedule View | See current and future reservations for each instrument |
| Admin Dashboard | Instrument/lab management, movement, and reporting |
| Notifications | Confirmation emails and alerts for bookings and changes |

**💻 Suggested Technology Stack**

* **Backend:** Laravel (PHP Framework)
* **Frontend:** Blade Templates with Metronic UI
* **Database:** MySQL
* **Authentication:** Laravel Breeze / Jetstream (for role-based access)
* **Booking Calendar:** FullCalendar.js (integrated in dashboard)
* **Reports:** Laravel Excel, Charts (e.g., Chart.js)

**🔐 Security Considerations**

* Role-based access control to prevent unauthorized changes
* Input validation and CSRF protection
* Logging admin changes for audit trail

**🌐 Multilingual Support**

Support for Arabic and English user interface and data input, especially for lab and instrument descriptions.

Here's a **Database Schema (ERD)** and corresponding **Laravel migration snippets** for the Laboratory Management and Booking System at the University of Tripoli.

## 📘 **Entity Relationship Diagram (ERD)**

[Campuses]

- id

- name (optional)

[Buildings]

- id

- campus\_id (FK)

- name

[Labs]

- id

- building\_id (FK)

- room\_number

- description

[Instruments]

- id

- lab\_id (FK)

- name

- purpose

- description

- serial\_number

- model

- experiment\_types

- analysis\_types

- status

- required\_materials

- responsible\_person

[Users]

- id

- name

- email

- role (admin, staff, student)

- password

[Bookings]

- id

- user\_id (FK)

- instrument\_id (FK)

- start\_time

- end\_time

- experiment\_description

- status (pending, approved, cancelled)

## 📦 **Laravel Migration Snippets**

### 1. ****create\_campuses\_table****

Schema::create('campuses', function (Blueprint $table) {

$table->id();

$table->string('name')->nullable();

$table->timestamps();

});

### 2. ****create\_buildings\_table****

Schema::create('buildings', function (Blueprint $table) {

$table->id();

$table->foreignId('campus\_id')->constrained()->onDelete('cascade');

$table->string('name');

$table->timestamps();

});

### 3. ****create\_labs\_table****

Schema::create('labs', function (Blueprint $table) {

$table->id();

$table->foreignId('building\_id')->constrained()->onDelete('cascade');

$table->string('room\_number');

$table->text('description');

$table->timestamps();

});

### 4. ****create\_instruments\_table****

Schema::create('instruments', function (Blueprint $table) {

$table->id();

$table->foreignId('lab\_id')->constrained()->onDelete('cascade');

$table->string('name');

$table->string('purpose');

$table->text('description');

$table->string('serial\_number')->unique();

$table->string('model')->nullable();

$table->text('experiment\_types')->nullable();

$table->text('analysis\_types')->nullable();

$table->enum('status', ['active', 'maintenance', 'out\_of\_order'])->default('active');

$table->text('required\_materials')->nullable();

$table->string('responsible\_person')->nullable();

$table->timestamps();

});

### 5. ****create\_users\_table**** (modified for roles)

Schema::create('users', function (Blueprint $table) {

$table->id();

$table->string('name');

$table->string('email')->unique();

$table->enum('role', ['admin', 'staff', 'student'])->default('student');

$table->string('password');

$table->rememberToken();

$table->timestamps();

});

### 6. ****create\_bookings\_table****

Schema::create('bookings', function (Blueprint $table) {

$table->id();

$table->foreignId('user\_id')->constrained()->onDelete('cascade');

$table->foreignId('instrument\_id')->constrained()->onDelete('cascade');

$table->timestamp('start\_time');

$table->timestamp('end\_time');

$table->text('experiment\_description')->nullable();

$table->enum('status', ['pending', 'approved', 'cancelled'])->default('pending');

$table->timestamps();

});

## 🧩 Relationships Summary

* A **Campus** has many **Buildings**
* A **Building** has many **Labs**
* A **Lab** has many **Instruments**
* An **Instrument** can be booked by many **Users**
* A **User** can make many **Bookings**

Would you like me to generate the **seeders**, **factories**, or **controllers** next?

Certainly! Below are **six practical CRUD-based test tasks** derived from the Laboratory Management and Booking System specification. These tasks are designed for a Laravel test using the **Metronic interface**.

**🧪 Practical Laravel CRUD Tasks for Students**

**Context**: University of Tripoli - Laboratory Management and Booking System  
**Requirement**: Use Laravel (preferably Laravel Breeze or Jetstream) with Metronic Blade templates. Bootstrap and jQuery are allowed.

**✅ Task 1: Lab Management CRUD**

**Question**:

Build a CRUD interface to manage university labs.  
Each lab must include the following fields:

* Campus Number
* Building Name
* Room Number
* Lab Description

Display the lab records in a table using Metronic's datatable. Include **Add**, **Edit**, and **Delete** buttons for each row.

**✅ Task 2: Instrument Inventory CRUD**

**Question**:

Create a CRUD system to manage instruments assigned to labs. Each instrument should include:

* Name
* Serial Number
* Model
* Status (active, maintenance, out\_of\_order)
* Associated Lab

Add a dropdown to assign the instrument to a specific lab from existing labs. Use Metronic form controls and modals for editing.

**✅ Task 3: Instrument Details Page**

**Question**:

Build a **view page** for a specific instrument.  
It should display:

* Name, Purpose, Description
* Experiment types and analysis types
* Responsible person

Style the view using Metronic cards and badges for status.

**✅ Task 4: Booking Interface**

**Question**:

Create a form for students to **book an instrument**.  
The form should include:

* Instrument (dropdown)
* Start Time
* End Time
* Purpose of experiment

Use a calendar widget or datetime picker from Metronic. Save bookings in the database. Ensure time slots do **not overlap**.

**✅ Task 5: Booking Calendar View**

**Question**:

Create a **read-only schedule page** showing bookings for all instruments using a calendar view (e.g., FullCalendar.js).

* Allow filtering by instrument
* Bookings should show student name and status (pending, approved)

Use Metronic's calendar component for integration.

**✅ Task 6: Instrument Movement**

**Question**:

Create a function in the instrument edit form to **move an instrument** from one lab to another.

* Display current lab info
* Allow the admin to select a new lab from a dropdown

Upon saving, update the instrument’s lab association and show a success message using a Metronic alert.

Would you like these formatted as a printable PDF assignment, or do you want Arabic versions too?