1) Lab Exercise: Student Management System

Objective: Implement a Student Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to log in using their roll number as both username and password, maintain session information, and perform CRUD operations on student records.

Instructions:

Database Setup:

Create a MySQL database named StudentDB.

Create a table named Student with the following schema:

```
CREATE TABLE Student (
    student_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    date_of_birth DATE,
    gender ENUM('Male', 'Female', 'Other'),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255),
    enrollment_date DATE,
    program_id INT,
    FOREIGN KEY (program_id) REFERENCES Program(program_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Create a navigation menu on the homepage (home.jsp) to perform CRUD operations.

Implement Create, Read, Update, and Delete operations for student records.

Create JSP pages (create.jsp, read.jsp, update.jsp, delete.jsp) for each operation.

Each operation should correspond to a servlet (CreateServlet, ReadServlet, UpdateServlet, DeleteServlet).

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

2) Lab Exercise: Employee Management System

Objective: Develop an Employee Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage employee records including adding, viewing, updating, and deleting employee information.

Instructions:

Database Setup:

Create a MySQL database named EmployeeDB.

Create a table named Employee with the following schema:

```
CREATE TABcodeLE Employee (
    employee_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    date_of_birth DATE,
    gender ENUM('Male', 'Female', 'Other'),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255),
    hire_date DATE,
    department_id INT,
    position VARCHAR(255),
    FOREIGN KEY (department_id) REFERENCES Department(department_id)
);
```

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for employee records.

Create JSP pages (addEmployee.jsp, viewEmployee.jsp, updateEmployee.jsp, deleteEmployee.jsp) for each operation.

Each operation should correspond to a servlet (AddEmployeeServlet, ViewEmployeeServlet, UpdateEmployeeServlet, DeleteEmployeeServlet).

User Interface:

Create HTML forms for adding and updating employee information (addEmployeeForm.html, updateEmployeeForm.html).

Display employee records in a tabular format on the view page (viewEmployee.jsp).

Provide options to edit or delete each record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

3) Lab Exercise: Customer Registration System

Objective: Develop a Customer Registration System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to register new customers, view customer details, update customer information, and delete customer records.

Instructions:

Database Setup:

Create a MySQL database named CustomerDB.

Create a table named Customer with the following schema:

```
CREATE TABLE Customer (
    customer_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    date_of_birth DATE,
    gender ENUM('Male', 'Female', 'Other'),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255),
    registration_date DATE
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for customer records.

Create JSP pages (registerCustomer.jsp, viewCustomer.jsp, updateCustomer.jsp, deleteCustomer.jsp) for each operation.

Each operation should correspond to a servlet (RegisterCustomerServlet, ViewCustomerServlet, UpdateCustomerServlet, DeleteCustomerServlet).

User Interface:

Create HTML forms for registering and updating customer information (registerCustomerForm.html, updateCustomerForm.html).

Display customer records in a tabular format on the view page (viewCustomer.jsp).

Provide options to edit or delete each record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

4) Lab Exercise: Invoice Management System

Objective: Develop an Invoice Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to create new invoices, view invoice details, update invoice information, and track payment status.

Instructions:

Database Setup:

Create a MySQL database named InvoiceDB.

Create a table named Invoice with the following schema:

sql

```
CREATE TABLE Invoice (
    invoice_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    invoice_date DATE,
    total_amount DECIMAL(10, 2),
    payment_status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for invoice records.

Create JSP pages (createInvoice.jsp, viewInvoice.jsp, updateInvoice.jsp, deleteInvoice.jsp) for each operation.

Each operation should correspond to a servlet (CreateInvoiceServlet, ViewInvoiceServlet, UpdateInvoiceServlet, DeleteInvoiceServlet).

User Interface:

Create HTML forms for creating and updating invoice information (createInvoiceForm.html, updateInvoiceForm.html).

Display invoice records in a tabular format on the view page (viewInvoice.jsp).

Provide options to edit or delete each invoice.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

5) Lab Exercise: Order Management System

Objective: Develop an Order Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to create new orders, view order details, update order information, and cancel orders.

Instructions:

Database Setup:

Create a MySQL database named OrderDB.

Create a table named Order with the following schema:

```
CREATE TABLE `Order` (
    order_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    order_date DATE,
    total_amount DECIMAL(10, 2),
    status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for order records.

Create JSP pages (createOrder.jsp, viewOrder.jsp, updateOrder.jsp, cancelOrder.jsp) for each operation.

Each operation should correspond to a servlet (CreateOrderServlet, ViewOrderServlet, UpdateOrderServlet, CancelOrderServlet).

User Interface:

Create HTML forms for creating and updating order information (createOrderForm.html, updateOrderForm.html).

Display order records in a tabular format on the view page (viewOrder.jsp).

Provide options to edit or cancel each order.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

6) Lab Exercise: Employee Attendance Management System

Objective: Develop an Employee Attendance Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to record employee attendance by logging clock in and clock out times.

Instructions:

Database Setup:

Create a MySQL database named AttendanceDB.

Create a table named EmployeeAttendance with the following schema:

```
CREATE TABLE EmployeeAttendance (
    attendance_id INT PRIMARY KEY AUTO_INCREMENT,
    employee_id INT,
    date DATE,
    clock_in_time TIME,
    clock_out_time TIME,
    FOREIGN KEY (employee_id) REFERENCES Employee(employee_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for employee attendance records.

Create JSP pages (recordAttendance.jsp, viewAttendance.jsp, updateAttendance.jsp, deleteAttendance.jsp) for each operation.

Each operation should correspond to a servlet (RecordAttendanceServlet, ViewAttendanceServlet, UpdateAttendanceServlet, DeleteAttendanceServlet).

User Interface:

Create HTML forms for recording and updating employee attendance (recordAttendanceForm.html, updateAttendanceForm.html).

Display attendance records in a tabular format on the view page (viewAttendance.jsp).

Provide options to edit or delete each attendance record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

7) Lab Exercise: Student Course Enrollment System:

Objective: Develop a Student Course Enrollment System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to enroll students in courses, view enrollment details, update enrollment information, and delete enrollments.

Instructions:

Database Setup:

Create a MySQL database named EnrollmentDB.

Create a table named StudentCourse with the following schema:

```
CREATE TABLE StudentCourse (
    enrollment_id INT PRIMARY KEY AUTO_INCREMENT,
    student_id INT,
    course_id INT,
    enrollment_date DATE,
    FOREIGN KEY (student_id) REFERENCES Student(student_id),
    FOREIGN KEY (course_id) REFERENCES Course(course_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for student course enrollments.

Create JSP pages (enrollStudent.jsp, viewEnrollment.jsp, updateEnrollment.jsp, deleteEnrollment.jsp) for each operation.

Each operation should correspond to a servlet (EnrollStudentServlet, ViewEnrollmentServlet, UpdateEnrollmentServlet, DeleteEnrollmentServlet).

User Interface:

Create HTML forms for enrolling and updating student course enrollments (enrollStudentForm.html, updateEnrollmentForm.html).

Display enrollment records in a tabular format on the view page (viewEnrollment.jsp).

Provide options to edit or delete each enrollment.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

8) Lab Exercise: Project Management System:

Objective: Develop a Project Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage projects including adding new projects, viewing project details, updating project information, and tracking project status.

Instructions:

Database Setup:

Create a MySQL database named ProjectDB.

Create a table named Project with the following schema:

```
CREATE TABLE Project (
    project_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    start_date DATE,
    end_date DATE,
    manager_id INT,
    status VARCHAR(50),
    FOREIGN KEY (manager_id) REFERENCES Employee(employee_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for project records.

Create JSP pages (addProject.jsp, viewProject.jsp, updateProject.jsp, deleteProject.jsp) for each operation.

Each operation should correspond to a servlet (AddProjectServlet, ViewProjectServlet, UpdateProjectServlet, DeleteProjectServlet).

User Interface:

Create HTML forms for adding and updating project information (addProjectForm.html, updateProjectForm.html).

Display project records in a tabular format on the view page (viewProject.jsp).

Provide options to edit or delete each project.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

9) Lab Exercise: Supplier Management System

Objective: Develop a Supplier Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage suppliers including adding new suppliers, viewing supplier details, updating supplier information, and deleting suppliers.

Instructions:

Database Setup:

Create a MySQL database named SupplierDB.

Create a table named Supplier with the following schema:

```
CREATE TABLE Supplier (
    supplier_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    contact_person VARCHAR(255),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for supplier records.

Create JSP pages (addSupplier.jsp, viewSupplier.jsp, updateSupplier.jsp, deleteSupplier.jsp) for each operation.

Each operation should correspond to a servlet (AddSupplierServlet, ViewSupplierServlet, UpdateSupplierServlet, DeleteSupplierServlet).

User Interface:

Create HTML forms for adding and updating supplier information (addSupplierForm.html, updateSupplierForm.html).

Display supplier records in a tabular format on the view page (viewSupplier.jsp).

Provide options to edit or delete each supplier.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

10) Lab Exercise: Transaction Management System

Objective: Develop a Transaction Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage transactions including recording new transactions, viewing transaction details, updating transaction information, and deleting transactions.

Instructions:

Database Setup:

Create a MySQL database named TransactionDB.

Create a table named Transaction with the following schema:

```
CREATE TABLE Transaction (
    transaction_id INT PRIMARY KEY AUTO_INCREMENT,
    date DATE,
    amount DECIMAL(10, 2),
    description TEXT,
    type ENUM('Income', 'Expense'),
    related_entity_id INT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for transaction records.

Create JSP pages (recordTransaction.jsp, viewTransaction.jsp, updateTransaction.jsp, deleteTransaction.jsp) for each operation.

Each operation should correspond to a servlet (RecordTransactionServlet, ViewTransactionServlet, UpdateTransactionServlet, DeleteTransactionServlet).

User Interface:

Create HTML forms for recording and updating transaction information (recordTransactionForm.html, updateTransactionForm.html).

Display transaction records in a tabular format on the view page (viewTransaction.jsp).

Provide options to edit or delete each transaction.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

11) Lab Exercise: Contact Management System

Objective: Develop a Contact Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage contacts including adding new contacts, viewing contact details, updating contact information, and deleting contacts.

Instructions:

Database Setup:

Create a MySQL database named ContactDB.

Create a table named Contact with the following schema:

```
CREATE TABLE Contact (
    contact_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for contact records.

Create JSP pages (addContact.jsp, viewContact.jsp, updateContact.jsp, deleteContact.jsp) for each operation.

Each operation should correspond to a servlet (AddContactServlet, ViewContactServlet, UpdateContactServlet, DeleteContactServlet).

User Interface:

Create HTML forms for adding and updating contact information (addContactForm.html, updateContactForm.html).

Display contact records in a tabular format on the view page (viewContact.jsp).

Provide options to edit or delete each contact.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

12) Lab Exercise: Inventory Management System

Objective: Develop an Inventory Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage inventory including adding new items, viewing inventory details, updating item information, and deleting items.

Instructions:

Database Setup:

Create a MySQL database named InventoryDB.

Create a table named Inventory with the following schema:

```
CREATE TABLE Inventory (
    inventory_id INT PRIMARY KEY AUTO_INCREMENT,
    product_id INT,
    quantity_available INT,
    location VARCHAR(255),
    FOREIGN KEY (product_id) REFERENCES Product(product_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for inventory items.

Create JSP pages (addItem.jsp, viewInventory.jsp, updateItem.jsp, deleteItem.jsp) for each operation.

Each operation should correspond to a servlet (AddItemServlet, ViewInventoryServlet, UpdateItemServlet, DeleteItemServlet).

User Interface:

Create HTML forms for adding and updating item information (addItemForm.html, updateItemForm.html).

Display inventory records in a tabular format on the view page (viewInventory.jsp).

Provide options to edit or delete each item.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

13) Lab Exercise: Reservation Management System

Objective: Develop a Reservation Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage reservations including making new reservations, viewing reservation details, updating reservation information, and canceling reservations.

Instructions:

Database Setup:

Create a MySQL database named ReservationDB.

Create a table named Reservation with the following schema:

```
CREATE TABLE Reservation (
    reservation_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    reservation_date DATE,
    start_date DATE,
    end_date DATE,
    status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for reservation records.

Create JSP pages (makeReservation.jsp, viewReservation.jsp, updateReservation.jsp, cancelReservation.jsp) for each operation.

Each operation should correspond to a servlet (MakeReservationServlet, ViewReservationServlet, UpdateReservationServlet, CancelReservationServlet).

User Interface:

Create HTML forms for making and updating reservation information (makeReservationForm.html, updateReservationForm.html).

Display reservation records in a tabular format on the view page (viewReservation.jsp).

Provide options to edit or cancel each reservation.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

14) Lab Exercise: Feedback Management System

Objective: Develop a Feedback Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow customers to submit feedback, view feedback details, update their feedback, and delete feedback entries.

Instructions:

Database Setup:

Create a MySQL database named FeedbackDB.

Create a table named Feedback with the following schema:

```
CREATE TABLE Feedback (
    feedback_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    date DATE,
    message TEXT,
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for feedback entries.

Create JSP pages (submitFeedback.jsp, viewFeedback.jsp, updateFeedback.jsp, deleteFeedback.jsp) for each operation.

Each operation should correspond to a servlet (SubmitFeedbackServlet, ViewFeedbackServlet, UpdateFeedbackServlet, DeleteFeedbackServlet).

User Interface:

Create HTML forms for submitting and updating feedback (submitFeedbackForm.html, updateFeedbackForm.html).

Display feedback entries in a tabular format on the view page (viewFeedback.jsp).

Provide options to edit or delete each feedback entry.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

15) Lab Exercise: Payment Management System

Objective: Develop a Payment Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow customers to make payments, view payment details, update payment information, and delete payment records.

Instructions:

Database Setup:

Create a MySQL database named PaymentDB.

Create a table named Payment with the following schema:

sql

Copy code

```
CREATE TABLE Payment (
    payment_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    amount DECIMAL(10, 2),
    date DATE,
    payment_method VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for payment records.

Create JSP pages (makePayment.jsp , viewPayment.jsp, updatePayment.jsp, deletePayment.jsp) for each operation.

Each operation should correspond to a servlet (MakePaymentServlet, ViewPaymentServlet, UpdatePaymentServlet, DeletePaymentServlet).

User Interface:

Create HTML forms for making and updating payments (makePaymentForm.html, updatePaymentForm.html).

Display payment records in a tabular format on the view page (viewPayment.jsp).

Provide options to edit or delete each payment record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

16) Lab Exercise: Subscription Management System

Objective: Develop a Subscription Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow customers to manage their subscriptions including starting new subscriptions, viewing subscription details, updating subscription information, and canceling subscriptions.

Instructions:

Database Setup:

Create a MySQL database named SubscriptionDB.

Create a table named Subscription with the following schema:

```
CREATE TABLE Subscription (
    subscription_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    start_date DATE,
    end_date DATE,
    status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for subscription records.

Create JSP pages (startSubscription.jsp, viewSubscription.jsp, updateSubscription.jsp, cancelSubscription.jsp) for each operation.

Each operation should correspond to a servlet (StartSubscriptionServlet, ViewSubscriptionServlet, UpdateSubscriptionServlet, CancelSubscriptionServlet).

User Interface:

Create HTML forms for starting and updating subscriptions (startSubscriptionForm.html, updateSubscriptionForm.html).

Display subscription records in a tabular format on the view page (viewSubscription.jsp).

Provide options to edit or cancel each subscription.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

17) Lab Exercise: Service Management System

Objective: Develop a Service Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage services including adding new services, viewing service details, updating service information, and deleting services.

Instructions:

Database Setup:

Create a MySQL database named ServiceDB.

Create a table named Service with the following schema:

```
CREATE TABLE Service (
    service_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    price DECIMAL(10, 2)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for service records.

Create JSP pages (addService.jsp, viewService.jsp, updateService.jsp, deleteService.jsp) for each operation.

Each operation should correspond to a servlet (AddServiceServlet, ViewServiceServlet, UpdateServiceServlet, DeleteServiceServlet).

User Interface:

Create HTML forms for adding and updating service information (addServiceForm.html, updateServiceForm.html).

Display service records in a tabular format on the view page (viewService.jsp).

Provide options to edit or delete each service.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

18) Lab Exercise: Product Review Management System

Objective: Develop a Product Review Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow customers to review products, view review details, update their reviews, and delete reviews.

Instructions:

Database Setup:

Create a MySQL database named ReviewDB.

Create a table named Review with the following schema:

```
CREATE TABLE Review (
    review_id INT PRIMARY KEY AUTO_INCREMENT,
    product_id INT,
    customer_id INT,
    rating INT,
    comment TEXT,
    date DATE,
    FOREIGN KEY (product_id) REFERENCES Product(product_id),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for review records.

Create JSP pages (addReview.jsp, viewReview.jsp, updateReview.jsp, deleteReview.jsp) for each operation.

Each operation should correspond to a servlet (AddReviewServlet, ViewReviewServlet, UpdateReviewServlet, DeleteReviewServlet).

User Interface:

Create HTML forms for adding and updating review information (addReviewForm.html, updateReviewForm.html).

Display review records in a tabular format on the view page (viewReview.jsp).

Provide options to edit or delete each review.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

19) Lab Exercise: Expense Management System

Objective: Develop an Expense Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage their expenses including adding new expenses, viewing expense details, updating expense information, and deleting expenses.

Instructions:

Database Setup:

Create a MySQL database named ExpenseDB.

Create a table named Expense with the following schema:

```
CREATE TABLE Expense (
    expense_id INT PRIMARY KEY AUTO_INCREMENT,
    date DATE,
    amount DECIMAL(10, 2),
    description TEXT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for expense records.

Create JSP pages (addExpense.jsp, viewExpense.jsp, updateExpense.jsp, deleteExpense.jsp) for each operation.

Each operation should correspond to a servlet (AddExpenseServlet, ViewExpenseServlet, UpdateExpenseServlet, DeleteExpenseServlet).

User Interface:

Create HTML forms for adding and updating expense information (addExpenseForm.html, updateExpenseForm.html).

Display expense records in a tabular format on the view page (viewExpense.jsp).

Provide options to edit or delete each expense.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

20) Lab Exercise: Asset Management System

Objective: Develop an Asset Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage assets including adding new assets, viewing asset details, updating asset information, and deleting assets.

Instructions:

Database Setup:

Create a MySQL database named AssetDB.

Create a table named Asset with the following schema:

```
CREATE TABLE Asset (
    asset_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    value DECIMAL(10, 2)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for asset records.

Create JSP pages (addAsset.jsp, viewAsset.jsp, updateAsset.jsp, deleteAsset.jsp) for each operation.

Each operation should correspond to a servlet (AddAssetServlet, ViewAssetServlet, UpdateAssetServlet, DeleteAssetServlet).

User Interface:

Create HTML forms for adding and updating asset information (addAssetForm.html, updateAssetForm.html).

Display asset records in a tabular format on the view page (viewAsset.jsp).

Provide options to edit or delete each asset.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

21) Lab Exercise: Issue Tracking System

Objective: Develop an Issue Tracking System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage issues including adding new issues, viewing issue details, updating issue information, and deleting issues.

Instructions:

Database Setup:

Create a MySQL database named IssueDB.

Create a table named Issue with the following schema:

```
CREATE TABLE Issue (
    issue_id INT PRIMARY KEY AUTO_INCREMENT,
    description TEXT,
    status VARCHAR(50),
    priority VARCHAR(50),
    assigned_to INT,
    FOREIGN KEY (assigned_to) REFERENCES Employee(employee_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for issue records.

Create JSP pages (addIssue.jsp, viewIssue.jsp, updateIssue.jsp, deleteIssue.jsp) for each operation.

Each operation should correspond to a servlet (AddIssueServlet, ViewIssueServlet, UpdateIssueServlet, DeleteIssueServlet).

User Interface:

Create HTML forms for adding and updating issue information (addIssueForm.html, updateIssueForm.html).

Display issue records in a tabular format on the view page (viewIssue.jsp).

Provide options to edit or delete each issue.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

22) Lab Exercise: Policy Management System

Objective: Develop a Policy Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage insurance policies including adding new policies, viewing policy details, updating policy information, and deleting policies.

Instructions:

Database Setup:

Create a MySQL database named PolicyDB.

Create a table named Policy with the following schema:

```
CREATE TABLE Policy (
    policy_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    coverage_amount DECIMAL(10, 2),
    premium_amount DECIMAL(10, 2)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for policy records.

Create JSP pages (addPolicy.jsp, viewPolicy.jsp, updatePolicy.jsp, deletePolicy.jsp) for each operation.

Each operation should correspond to a servlet (AddPolicyServlet, ViewPolicyServlet, UpdatePolicyServlet, DeletePolicyServlet).

User Interface:

Create HTML forms for adding and updating policy information (addPolicyForm.html, updatePolicyForm.html).

Display policy records in a tabular format on the view page (viewPolicy.jsp).

Provide options to edit or delete each policy.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

23) Lab Exercise: Ticket Management System

Objective: Develop a Ticket Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage tickets including adding new tickets, viewing ticket details, updating ticket information, and deleting tickets.

Instructions:

Database Setup:

Create a MySQL database named TicketDB.

Create a table named Ticket with the following schema:

```
CREATE TABLE Ticket (
    ticket_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    priority VARCHAR(50),
    status VARCHAR(50),
    assigned_to INT,
    FOREIGN KEY (assigned_to) REFERENCES Employee(employee_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for ticket records.

Create JSP pages (addTicket.jsp, viewTicket.jsp, updateTicket.jsp, deleteTicket.jsp) for each operation.

Each operation should correspond to a servlet (AddTicketServlet, ViewTicketServlet, UpdateTicketServlet, DeleteTicketServlet).

User Interface:

Create HTML forms for adding and updating ticket information (addTicketForm.html, updateTicketForm.html).

Display ticket records in a tabular format on the view page (viewTicket.jsp).

Provide options to edit or delete each ticket.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

24) Lab Exercise: Membership Management System

Objective: Develop a Membership Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage memberships including adding new memberships, viewing membership details, updating membership information, and deleting memberships.

Instructions:

Database Setup:

Create a MySQL database named MembershipDB.

Create a table named Membership with the following schema:

```
CREATE TABLE Membership (
    membership_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    start_date DATE,
    end_date DATE,
    type VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for membership records.

Create JSP pages (addMembership.jsp, viewMembership.jsp, updateMembership.jsp, deleteMembership.jsp) for each operation.

Each operation should correspond to a servlet (AddMembershipServlet, ViewMembershipServlet, UpdateMembershipServlet, DeleteMembershipServlet).

User Interface:

Create HTML forms for adding and updating membership information (addMembershipForm.html, updateMembershipForm.html).

Display membership records in a tabular format on the view page (viewMembership.jsp).

Provide options to edit or delete each membership.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

25) Lab Exercise: Account Management System

Objective: Develop an Account Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage accounts including adding new accounts, viewing account details, updating account information, and deleting accounts.

Instructions:

Database Setup:

Create a MySQL database named AccountDB.

Create a table named Account with the following schema:

```
CREATE TABLE Account (
    account_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    account_number VARCHAR(255),
    balance DECIMAL(10, 2),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for account records.

Create JSP pages (addAccount.jsp, viewAccount.jsp, updateAccount.jsp, deleteAccount.jsp) for each operation.

Each operation should correspond to a servlet (AddAccountServlet, ViewAccountServlet, UpdateAccountServlet, DeleteAccountServlet).

User Interface:

Create HTML forms for adding and updating account information (addAccountForm.html, updateAccountForm.html).

Display account records in a tabular format on the view page (viewAccount.jsp).

Provide options to edit or delete each account.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

26) Lab Exercise: Messaging System

Objective: Develop a Messaging System using HTML, JSP, and Servlets to facilitate communication between employees and customers. The system should allow users to send and receive messages, view message details, and manage their inbox.

Instructions:

Database Setup:

Create a MySQL database named MessagingDB.

Create a table named Message with the following schema:

```
CREATE TABLE Message (
    message_id INT PRIMARY KEY AUTO_INCREMENT,
    sender_id INT,
    receiver_id INT,
    subject VARCHAR(255),
    content TEXT,
    date DATETIME,
    FOREIGN KEY (sender_id) REFERENCES Employee(employee_id),
    FOREIGN KEY (receiver_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for messages.

Create JSP pages (sendMessage.jsp, inbox.jsp, viewMessage.jsp, deleteMessage.jsp) for each operation.

Each operation should correspond to a servlet (SendMessageServlet, InboxServlet, ViewMessageServlet, DeleteMessageServlet).

User Interface:

Create HTML forms for sending messages (sendMessageForm.html).

Display received messages in a list format on the inbox page (inbox.jsp).

Provide options to view and delete each message.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access messaging features.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

27) Lab Exercise: Recipe Management System

Objective: Develop a Recipe Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage recipes including adding new recipes, viewing recipe details, updating recipe information, and deleting recipes.

Instructions:

Database Setup:

Create a MySQL database named RecipeDB.

Create a table named Recipe with the following schema:

```
CREATE TABLE Recipe (
    recipe_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    ingredients TEXT,
    instructions TEXT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for recipe records.

Create JSP pages (addRecipe.jsp, viewRecipe.jsp, updateRecipe.jsp, deleteRecipe.jsp) for each operation.

Each operation should correspond to a servlet (AddRecipeServlet, ViewRecipeServlet, UpdateRecipeServlet, DeleteRecipeServlet).

User Interface:

Create HTML forms for adding and updating recipe information (addRecipeForm.html, updateRecipeForm.html).

Display recipe records in a tabular format on the view page (viewRecipe.jsp).

Provide options to edit or delete each recipe.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

28) Lab Exercise: Blog Management System

Objective: Develop a Blog Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage blog posts including adding new posts, viewing post details, updating post information, and deleting posts.

Instructions:

Database Setup:

Create a MySQL database named BlogDB.

Create a table named BlogPost with the following schema:

```
CREATE TABLE BlogPost (
    post_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(255) NOT NULL,
    content TEXT,
    author_id INT,
    date DATE,
    FOREIGN KEY (author_id) REFERENCES Employee(employee_id) ON DELETE CASCADE,
    FOREIGN KEY (author_id) REFERENCES Customer(customer_id) ON DELETE CASCADE
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for blog post records.

Create JSP pages (addPost.jsp, viewPost.jsp, updatePost.jsp, deletePost.jsp) for each operation.

Each operation should correspond to a servlet (AddPostServlet, ViewPostServlet, UpdatePostServlet, DeletePostServlet).

User Interface:

Create HTML forms for adding and updating blog post information (addPostForm.html, updatePostForm.html).

Display blog post records in a list format on the view page (viewPost.jsp).

Provide options to view, edit, or delete each post.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

29) Lab Exercise: Medical Record Management System

Objective: Develop a Medical Record Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage medical records including adding new records, viewing record details, updating record information, and deleting records.

Instructions:

Database Setup:

Create a MySQL database named MedicalRecordDB.

Create a table named MedicalRecord with the following schema:

```
CREATE TABLE MedicalRecord (
    record_id INT PRIMARY KEY AUTO_INCREMENT,
    patient_id INT,
    doctor_id INT,
    date DATE,
    diagnosis TEXT,
    treatment TEXT,
    FOREIGN KEY (patient_id) REFERENCES Patient(patient_id),
    FOREIGN KEY (doctor_id) REFERENCES Doctor(doctor_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for medical record entries.

Create JSP pages (addRecord.jsp, viewRecord.jsp, updateRecord.jsp, deleteRecord.jsp) for each operation.

Each operation should correspond to a servlet (AddRecordServlet, ViewRecordServlet, UpdateRecordServlet, DeleteRecordServlet).

User Interface:

Create HTML forms for adding and updating medical record information (addRecordForm.html, updateRecordForm.html).

Display medical record entries in a tabular format on the view page (viewRecord.jsp).

Provide options to edit or delete each record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

30) Lab Exercise: Patient Management System

Objective: Develop a Patient Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage patient records including adding new patients, viewing patient details, updating patient information, and deleting patients.

Instructions:

Database Setup:

Create a MySQL database named PatientDB.

Create a table named Patient with the following schema:

```
CREATE TABLE Patient (
    patient_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    date_of_birth DATE,
    gender VARCHAR(10),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address TEXT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for patient records.

Create JSP pages (addPatient.jsp, viewPatient.jsp, updatePatient.jsp, deletePatient.jsp) for each operation.

Each operation should correspond to a servlet (AddPatientServlet, ViewPatientServlet, UpdatePatientServlet, DeletePatientServlet).

User Interface:

Create HTML forms for adding and updating patient information (addPatientForm.html, updatePatientForm.html).

Display patient records in a tabular format on the view page (viewPatient.jsp).

Provide options to edit or delete each patient record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

31) Lab Exercise: Flight Management System

Objective: Develop a Flight Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage flight information including adding new flights, viewing flight details, updating flight information, and deleting flights.

Instructions:

Database Setup:

Create a MySQL database named FlightDB.

Create a table named Flight with the following schema:

```
CREATE TABLE Flight (
    flight_id INT PRIMARY KEY AUTO_INCREMENT,
    airline VARCHAR(255) NOT NULL,
    flight_number VARCHAR(20) NOT NULL,
    departure_airport VARCHAR(255) NOT NULL,
    arrival_airport VARCHAR(255) NOT NULL,
    departure_date DATETIME NOT NULL,
    arrival_date DATETIME NOT NULL
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for flight records.

Create JSP pages (addFlight.jsp, viewFlight.jsp, updateFlight.jsp, deleteFlight.jsp) for each operation.

Each operation should correspond to a servlet (AddFlightServlet, ViewFlightServlet, UpdateFlightServlet, DeleteFlightServlet).

User Interface:

Create HTML forms for adding and updating flight information (addFlightForm.html, updateFlightForm.html).

Display flight records in a tabular format on the view page (viewFlight.jsp).

Provide options to edit or delete each flight record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

32) Lab Exercise: Weather Monitoring System

Objective: Develop a Weather Monitoring System using HTML, JSP, and Servlets to track weather conditions for different locations. The system should allow users to manage weather data including adding new records, viewing weather details, updating weather information, and deleting records.

Instructions:

Database Setup:

Create a MySQL database named WeatherDB.

Create a table named Weather with the following schema:

```
CREATE TABLE Weather (
    weather_id INT PRIMARY KEY AUTO_INCREMENT,
    location_id INT,
    temperature DECIMAL(5, 2),
    humidity DECIMAL(5, 2),
    wind_speed DECIMAL(5, 2),
    condition VARCHAR(255),
    observation_time DATETIME,
    FOREIGN KEY (location_id) REFERENCES Location(location_id)
);
```

CRUD Operations:

Implement Create, Read, Update, and Delete operations for weather records.

Create JSP pages (addWeather.jsp, viewWeather.jsp, updateWeather.jsp, deleteWeather.jsp) for each operation.

Each operation should correspond to a servlet (AddWeatherServlet, ViewWeatherServlet, UpdateWeatherServlet, DeleteWeatherServlet).

User Interface:

Create HTML forms for adding and updating weather information (addWeatherForm.html, updateWeatherForm.html).

Display weather records in a tabular format on the view page (viewWeather.jsp).

Provide options to edit or delete each weather record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

33) Lab Exercise: Room Booking System

Objective: Develop a Room Booking System using HTML, JSP, and Servlets to manage room reservations in a hotel. The system should allow users to add new rooms, view room details, update room information, and delete rooms.

Instructions:

Database Setup:

Create a MySQL database named HotelDB.

Create a table named Room with the following schema:

```
CREATE TABLE Room (
    room_id INT PRIMARY KEY AUTO_INCREMENT,
    room_number VARCHAR(10) NOT NULL,
    type VARCHAR(50),
    rate_per_night DECIMAL(8, 2),
    availability_status VARCHAR(20)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for room records.

Create JSP pages (addRoom.jsp, viewRoom.jsp, updateRoom.jsp, deleteRoom.jsp) for each operation.

Each operation should correspond to a servlet (AddRoomServlet, ViewRoomServlet, UpdateRoomServlet, DeleteRoomServlet).

User Interface:

Create HTML forms for adding and updating room information (addRoomForm.html, updateRoomForm.html).

Display room records in a tabular format on the view page (viewRoom.jsp).

Provide options to edit or delete each room record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

34) Lab Exercise: Car Rental Management System

Objective: Develop a Car Rental Management System using HTML, JSP, and Servlets to manage car rentals for a rental agency. The system should allow users to add new cars, view car details, update car information, and delete cars.

Instructions:

Database Setup:

Create a MySQL database named CarRentalDB.

Create a table named Car with the following schema:

```
CREATE TABLE Car (
    car_id INT PRIMARY KEY AUTO_INCREMENT,
    make VARCHAR(50) NOT NULL,
    model VARCHAR(50) NOT NULL,
    year INT NOT NULL,
    rental_rate_per_day DECIMAL(8, 2) NOT NULL,
    availability_status VARCHAR(20) NOT NULL
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for car records.

Create JSP pages (addCar.jsp, viewCar.jsp, updateCar.jsp, deleteCar.jsp) for each operation.

Each operation should correspond to a servlet (AddCarServlet, ViewCarServlet, UpdateCarServlet, DeleteCarServlet).

User Interface:

Create HTML forms for adding and updating car information (addCarForm.html, updateCarForm.html).

Display car records in a tabular format on the view page (viewCar.jsp).

Provide options to edit or delete each car record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet)(navigation menu should be visible in every page by using include servlet).

35) Lab Exercise: Bike Rental Management System

Objective: Develop a Bike Rental Management System using HTML, JSP, and Servlets to manage bike rentals for a rental agency. The system should allow users to add new bikes, view bike details, update bike information, and delete bikes.

Instructions:

Database Setup:

Create a MySQL database named BikeRentalDB.

Create a table named Bike with the following schema:

```
CREATE TABLE Bike (
    bike_id INT PRIMARY KEY AUTO_INCREMENT,
    make VARCHAR(50) NOT NULL,
    model VARCHAR(50) NOT NULL,
    year INT NOT NULL,
    rental_rate_per_day DECIMAL(8, 2) NOT NULL,
    availability_status VARCHAR(20) NOT NULL
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for bike records.

Create JSP pages (addBike.jsp, viewBike.jsp, updateBike.jsp, deleteBike.jsp) for each operation.

Each operation should correspond to a servlet (AddBikeServlet, ViewBikeServlet, UpdateBikeServlet, DeleteBikeServlet).

User Interface:

Create HTML forms for adding and updating bike information (addBikeForm.html, updateBikeForm.html).

Display bike records in a tabular format on the view page (viewBike.jsp).

Provide options to edit or delete each bike record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet)(navigation menu should be visible in every page by using include servlet).

36) Lab Exercise: Faculty Management System

Objective: Develop a Faculty Management System using HTML, JSP, and Servlets to manage faculty information for an educational institution. The system should allow administrators to add new faculty members, view faculty details, update faculty information, and delete faculty records.

Instructions:

Database Setup:

Create a MySQL database named FacultyDB.

Create a table named Faculty with the following schema:

```
CREATE TABLE Faculty (
    faculty_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    designation VARCHAR(100) NOT NULL,
    department VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    phone_number VARCHAR(20),
    joining_date DATE NOT NULL,
    qualification TEXT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for faculty records.

Create JSP pages (addFaculty.jsp, viewFaculty.jsp, updateFaculty.jsp, deleteFaculty.jsp) for each operation.

Each operation should correspond to a servlet (AddFacultyServlet, ViewFacultyServlet, UpdateFacultyServlet, DeleteFacultyServlet).

User Interface:

Design user-friendly HTML forms for adding and updating faculty information (addFacultyForm.html, updateFacultyForm.html).

Display faculty records in a tabular format on the view page (viewFaculty.jsp).

Provide options to edit or delete each faculty record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet)(navigation menu should be visible in every page by using include servlet).

37) Lab Exercise: Outsourcing Staff Management System

Objective: Develop an Outsourcing Staff Management System using HTML, JSP, and Servlets to manage information about outsourced staff members for a company. The system should allow administrators to add new staff members, view staff details, update staff information, and delete staff records.

Instructions:

Database Setup:

Create a MySQL database named OutsourcingStaffDB.

Create a table named OutsourcingStaff with the following schema:

```
CREATE TABLE OutsourcingStaff (
    staff_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    role VARCHAR(100) NOT NULL,
    department VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    phone_number VARCHAR(20),
    contract_start_date DATE NOT NULL,
    contract_end_date DATE NOT NULL,
    hourly_rate DECIMAL(8, 2) NOT NULL
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for staff records.

Create JSP pages (addStaff.jsp, viewStaff.jsp, updateStaff.jsp, deleteStaff.jsp) for each operation.

Each operation should correspond to a servlet (AddStaffServlet, ViewStaffServlet, UpdateStaffServlet, DeleteStaffServlet).

User Interface:

Design user-friendly HTML forms for adding and updating staff information (addStaffForm.html, updateStaffForm.html).

Display staff records in a tabular format on the view page (viewStaff.jsp).

Provide options to edit or delete each staff record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet)(navigation menu should be visible in every page by using include servlet).

1) Lab Exercise: Student Management System

Objective: Implement a Student Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to log in using their roll number as both username and password, maintain session information, and perform CRUD operations on student records.

Instructions:

Database Setup:

Create a MySQL database named StudentDB.

Create a table named Student with the following schema:

```
CREATE TABLE Student (
    student_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    date_of_birth DATE,
    gender ENUM('Male', 'Female', 'Other'),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255),
    enrollment_date DATE,
    program_id INT,
    FOREIGN KEY (program_id) REFERENCES Program(program_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Create a navigation menu on the homepage (home.jsp) to perform CRUD operations.

Implement Create, Read, Update, and Delete operations for student records.

Create JSP pages (create.jsp, read.jsp, update.jsp, delete.jsp) for each operation.

Each operation should correspond to a servlet (CreateServlet, ReadServlet, UpdateServlet, DeleteServlet).

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

2) Lab Exercise: Employee Management System

Objective: Develop an Employee Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage employee records including adding, viewing, updating, and deleting employee information.

Instructions:

Database Setup:

Create a MySQL database named EmployeeDB.

Create a table named Employee with the following schema:

```
CREATE TABcodeLE Employee (
    employee_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    date_of_birth DATE,
    gender ENUM('Male', 'Female', 'Other'),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255),
    hire_date DATE,
    department_id INT,
    position VARCHAR(255),
    FOREIGN KEY (department_id) REFERENCES Department(department_id)
);
```

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for employee records.

Create JSP pages (addEmployee.jsp, viewEmployee.jsp, updateEmployee.jsp, deleteEmployee.jsp) for each operation.

Each operation should correspond to a servlet (AddEmployeeServlet, ViewEmployeeServlet, UpdateEmployeeServlet, DeleteEmployeeServlet).

User Interface:

Create HTML forms for adding and updating employee information (addEmployeeForm.html, updateEmployeeForm.html).

Display employee records in a tabular format on the view page (viewEmployee.jsp).

Provide options to edit or delete each record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

3) Lab Exercise: Customer Registration System

Objective: Develop a Customer Registration System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to register new customers, view customer details, update customer information, and delete customer records.

Instructions:

Database Setup:

Create a MySQL database named CustomerDB.

Create a table named Customer with the following schema:

```
CREATE TABLE Customer (
    customer_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    date_of_birth DATE,
    gender ENUM('Male', 'Female', 'Other'),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255),
    registration_date DATE
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for customer records.

Create JSP pages (registerCustomer.jsp, viewCustomer.jsp, updateCustomer.jsp, deleteCustomer.jsp) for each operation.

Each operation should correspond to a servlet (RegisterCustomerServlet, ViewCustomerServlet, UpdateCustomerServlet, DeleteCustomerServlet).

User Interface:

Create HTML forms for registering and updating customer information (registerCustomerForm.html, updateCustomerForm.html).

Display customer records in a tabular format on the view page (viewCustomer.jsp).

Provide options to edit or delete each record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

4) Lab Exercise: Invoice Management System

Objective: Develop an Invoice Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to create new invoices, view invoice details, update invoice information, and track payment status.

Instructions:

Database Setup:

Create a MySQL database named InvoiceDB.

Create a table named Invoice with the following schema:

sql

```
CREATE TABLE Invoice (
    invoice_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    invoice_date DATE,
    total_amount DECIMAL(10, 2),
    payment_status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for invoice records.

Create JSP pages (createInvoice.jsp, viewInvoice.jsp, updateInvoice.jsp, deleteInvoice.jsp) for each operation.

Each operation should correspond to a servlet (CreateInvoiceServlet, ViewInvoiceServlet, UpdateInvoiceServlet, DeleteInvoiceServlet).

User Interface:

Create HTML forms for creating and updating invoice information (createInvoiceForm.html, updateInvoiceForm.html).

Display invoice records in a tabular format on the view page (viewInvoice.jsp).

Provide options to edit or delete each invoice.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

5) Lab Exercise: Order Management System

Objective: Develop an Order Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to create new orders, view order details, update order information, and cancel orders.

Instructions:

Database Setup:

Create a MySQL database named OrderDB.

Create a table named Order with the following schema:

```
CREATE TABLE `Order` (
    order_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    order_date DATE,
    total_amount DECIMAL(10, 2),
    status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for order records.

Create JSP pages (createOrder.jsp, viewOrder.jsp, updateOrder.jsp, cancelOrder.jsp) for each operation.

Each operation should correspond to a servlet (CreateOrderServlet, ViewOrderServlet, UpdateOrderServlet, CancelOrderServlet).

User Interface:

Create HTML forms for creating and updating order information (createOrderForm.html, updateOrderForm.html).

Display order records in a tabular format on the view page (viewOrder.jsp).

Provide options to edit or cancel each order.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

6) Lab Exercise: Employee Attendance Management System

Objective: Develop an Employee Attendance Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to record employee attendance by logging clock in and clock out times.

Instructions:

Database Setup:

Create a MySQL database named AttendanceDB.

Create a table named EmployeeAttendance with the following schema:

```
CREATE TABLE EmployeeAttendance (
    attendance_id INT PRIMARY KEY AUTO_INCREMENT,
    employee_id INT,
    date DATE,
    clock_in_time TIME,
    clock_out_time TIME,
    FOREIGN KEY (employee_id) REFERENCES Employee(employee_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for employee attendance records.

Create JSP pages (recordAttendance.jsp, viewAttendance.jsp, updateAttendance.jsp, deleteAttendance.jsp) for each operation.

Each operation should correspond to a servlet (RecordAttendanceServlet, ViewAttendanceServlet, UpdateAttendanceServlet, DeleteAttendanceServlet).

User Interface:

Create HTML forms for recording and updating employee attendance (recordAttendanceForm.html, updateAttendanceForm.html).

Display attendance records in a tabular format on the view page (viewAttendance.jsp).

Provide options to edit or delete each attendance record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

7) Lab Exercise: Student Course Enrollment System:

Objective: Develop a Student Course Enrollment System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to enroll students in courses, view enrollment details, update enrollment information, and delete enrollments.

Instructions:

Database Setup:

Create a MySQL database named EnrollmentDB.

Create a table named StudentCourse with the following schema:

```
CREATE TABLE StudentCourse (
    enrollment_id INT PRIMARY KEY AUTO_INCREMENT,
    student_id INT,
    course_id INT,
    enrollment_date DATE,
    FOREIGN KEY (student_id) REFERENCES Student(student_id),
    FOREIGN KEY (course_id) REFERENCES Course(course_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for student course enrollments.

Create JSP pages (enrollStudent.jsp, viewEnrollment.jsp, updateEnrollment.jsp, deleteEnrollment.jsp) for each operation.

Each operation should correspond to a servlet (EnrollStudentServlet, ViewEnrollmentServlet, UpdateEnrollmentServlet, DeleteEnrollmentServlet).

User Interface:

Create HTML forms for enrolling and updating student course enrollments (enrollStudentForm.html, updateEnrollmentForm.html).

Display enrollment records in a tabular format on the view page (viewEnrollment.jsp).

Provide options to edit or delete each enrollment.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

8) Lab Exercise: Project Management System:

Objective: Develop a Project Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage projects including adding new projects, viewing project details, updating project information, and tracking project status.

Instructions:

Database Setup:

Create a MySQL database named ProjectDB.

Create a table named Project with the following schema:

```
CREATE TABLE Project (
    project_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    start_date DATE,
    end_date DATE,
    manager_id INT,
    status VARCHAR(50),
    FOREIGN KEY (manager_id) REFERENCES Employee(employee_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for project records.

Create JSP pages (addProject.jsp, viewProject.jsp, updateProject.jsp, deleteProject.jsp) for each operation.

Each operation should correspond to a servlet (AddProjectServlet, ViewProjectServlet, UpdateProjectServlet, DeleteProjectServlet).

User Interface:

Create HTML forms for adding and updating project information (addProjectForm.html, updateProjectForm.html).

Display project records in a tabular format on the view page (viewProject.jsp).

Provide options to edit or delete each project.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

9) Lab Exercise: Supplier Management System

Objective: Develop a Supplier Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage suppliers including adding new suppliers, viewing supplier details, updating supplier information, and deleting suppliers.

Instructions:

Database Setup:

Create a MySQL database named SupplierDB.

Create a table named Supplier with the following schema:

```
CREATE TABLE Supplier (
    supplier_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    contact_person VARCHAR(255),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for supplier records.

Create JSP pages (addSupplier.jsp, viewSupplier.jsp, updateSupplier.jsp, deleteSupplier.jsp) for each operation.

Each operation should correspond to a servlet (AddSupplierServlet, ViewSupplierServlet, UpdateSupplierServlet, DeleteSupplierServlet).

User Interface:

Create HTML forms for adding and updating supplier information (addSupplierForm.html, updateSupplierForm.html).

Display supplier records in a tabular format on the view page (viewSupplier.jsp).

Provide options to edit or delete each supplier.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

10) Lab Exercise: Transaction Management System

Objective: Develop a Transaction Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage transactions including recording new transactions, viewing transaction details, updating transaction information, and deleting transactions.

Instructions:

Database Setup:

Create a MySQL database named TransactionDB.

Create a table named Transaction with the following schema:

```
CREATE TABLE Transaction (
    transaction_id INT PRIMARY KEY AUTO_INCREMENT,
    date DATE,
    amount DECIMAL(10, 2),
    description TEXT,
    type ENUM('Income', 'Expense'),
    related_entity_id INT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for transaction records.

Create JSP pages (recordTransaction.jsp, viewTransaction.jsp, updateTransaction.jsp, deleteTransaction.jsp) for each operation.

Each operation should correspond to a servlet (RecordTransactionServlet, ViewTransactionServlet, UpdateTransactionServlet, DeleteTransactionServlet).

User Interface:

Create HTML forms for recording and updating transaction information (recordTransactionForm.html, updateTransactionForm.html).

Display transaction records in a tabular format on the view page (viewTransaction.jsp).

Provide options to edit or delete each transaction.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

11) Lab Exercise: Contact Management System

Objective: Develop a Contact Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage contacts including adding new contacts, viewing contact details, updating contact information, and deleting contacts.

Instructions:

Database Setup:

Create a MySQL database named ContactDB.

Create a table named Contact with the following schema:

```
CREATE TABLE Contact (
    contact_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for contact records.

Create JSP pages (addContact.jsp, viewContact.jsp, updateContact.jsp, deleteContact.jsp) for each operation.

Each operation should correspond to a servlet (AddContactServlet, ViewContactServlet, UpdateContactServlet, DeleteContactServlet).

User Interface:

Create HTML forms for adding and updating contact information (addContactForm.html, updateContactForm.html).

Display contact records in a tabular format on the view page (viewContact.jsp).

Provide options to edit or delete each contact.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

12) Lab Exercise: Inventory Management System

Objective: Develop an Inventory Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage inventory including adding new items, viewing inventory details, updating item information, and deleting items.

Instructions:

Database Setup:

Create a MySQL database named InventoryDB.

Create a table named Inventory with the following schema:

```
CREATE TABLE Inventory (
    inventory_id INT PRIMARY KEY AUTO_INCREMENT,
    product_id INT,
    quantity_available INT,
    location VARCHAR(255),
    FOREIGN KEY (product_id) REFERENCES Product(product_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for inventory items.

Create JSP pages (addItem.jsp, viewInventory.jsp, updateItem.jsp, deleteItem.jsp) for each operation.

Each operation should correspond to a servlet (AddItemServlet, ViewInventoryServlet, UpdateItemServlet, DeleteItemServlet).

User Interface:

Create HTML forms for adding and updating item information (addItemForm.html, updateItemForm.html).

Display inventory records in a tabular format on the view page (viewInventory.jsp).

Provide options to edit or delete each item.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

13) Lab Exercise: Reservation Management System

Objective: Develop a Reservation Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage reservations including making new reservations, viewing reservation details, updating reservation information, and canceling reservations.

Instructions:

Database Setup:

Create a MySQL database named ReservationDB.

Create a table named Reservation with the following schema:

```
CREATE TABLE Reservation (
    reservation_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    reservation_date DATE,
    start_date DATE,
    end_date DATE,
    status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for reservation records.

Create JSP pages (makeReservation.jsp, viewReservation.jsp, updateReservation.jsp, cancelReservation.jsp) for each operation.

Each operation should correspond to a servlet (MakeReservationServlet, ViewReservationServlet, UpdateReservationServlet, CancelReservationServlet).

User Interface:

Create HTML forms for making and updating reservation information (makeReservationForm.html, updateReservationForm.html).

Display reservation records in a tabular format on the view page (viewReservation.jsp).

Provide options to edit or cancel each reservation.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

14) Lab Exercise: Feedback Management System

Objective: Develop a Feedback Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow customers to submit feedback, view feedback details, update their feedback, and delete feedback entries.

Instructions:

Database Setup:

Create a MySQL database named FeedbackDB.

Create a table named Feedback with the following schema:

```
CREATE TABLE Feedback (
    feedback_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    date DATE,
    message TEXT,
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for feedback entries.

Create JSP pages (submitFeedback.jsp, viewFeedback.jsp, updateFeedback.jsp, deleteFeedback.jsp) for each operation.

Each operation should correspond to a servlet (SubmitFeedbackServlet, ViewFeedbackServlet, UpdateFeedbackServlet, DeleteFeedbackServlet).

User Interface:

Create HTML forms for submitting and updating feedback (submitFeedbackForm.html, updateFeedbackForm.html).

Display feedback entries in a tabular format on the view page (viewFeedback.jsp).

Provide options to edit or delete each feedback entry.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

15) Lab Exercise: Payment Management System

Objective: Develop a Payment Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow customers to make payments, view payment details, update payment information, and delete payment records.

Instructions:

Database Setup:

Create a MySQL database named PaymentDB.

Create a table named Payment with the following schema:

sql

Copy code

```
CREATE TABLE Payment (
    payment_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    amount DECIMAL(10, 2),
    date DATE,
    payment_method VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for payment records.

Create JSP pages (makePayment.jsp , viewPayment.jsp, updatePayment.jsp, deletePayment.jsp) for each operation.

Each operation should correspond to a servlet (MakePaymentServlet, ViewPaymentServlet, UpdatePaymentServlet, DeletePaymentServlet).

User Interface:

Create HTML forms for making and updating payments (makePaymentForm.html, updatePaymentForm.html).

Display payment records in a tabular format on the view page (viewPayment.jsp).

Provide options to edit or delete each payment record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

16) Lab Exercise: Subscription Management System

Objective: Develop a Subscription Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow customers to manage their subscriptions including starting new subscriptions, viewing subscription details, updating subscription information, and canceling subscriptions.

Instructions:

Database Setup:

Create a MySQL database named SubscriptionDB.

Create a table named Subscription with the following schema:

```
CREATE TABLE Subscription (
    subscription_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    start_date DATE,
    end_date DATE,
    status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for subscription records.

Create JSP pages (startSubscription.jsp, viewSubscription.jsp, updateSubscription.jsp, cancelSubscription.jsp) for each operation.

Each operation should correspond to a servlet (StartSubscriptionServlet, ViewSubscriptionServlet, UpdateSubscriptionServlet, CancelSubscriptionServlet).

User Interface:

Create HTML forms for starting and updating subscriptions (startSubscriptionForm.html, updateSubscriptionForm.html).

Display subscription records in a tabular format on the view page (viewSubscription.jsp).

Provide options to edit or cancel each subscription.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

17) Lab Exercise: Service Management System

Objective: Develop a Service Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage services including adding new services, viewing service details, updating service information, and deleting services.

Instructions:

Database Setup:

Create a MySQL database named ServiceDB.

Create a table named Service with the following schema:

```
CREATE TABLE Service (
    service_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    price DECIMAL(10, 2)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for service records.

Create JSP pages (addService.jsp, viewService.jsp, updateService.jsp, deleteService.jsp) for each operation.

Each operation should correspond to a servlet (AddServiceServlet, ViewServiceServlet, UpdateServiceServlet, DeleteServiceServlet).

User Interface:

Create HTML forms for adding and updating service information (addServiceForm.html, updateServiceForm.html).

Display service records in a tabular format on the view page (viewService.jsp).

Provide options to edit or delete each service.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

18) Lab Exercise: Product Review Management System

Objective: Develop a Product Review Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow customers to review products, view review details, update their reviews, and delete reviews.

Instructions:

Database Setup:

Create a MySQL database named ReviewDB.

Create a table named Review with the following schema:

```
CREATE TABLE Review (
    review_id INT PRIMARY KEY AUTO_INCREMENT,
    product_id INT,
    customer_id INT,
    rating INT,
    comment TEXT,
    date DATE,
    FOREIGN KEY (product_id) REFERENCES Product(product_id),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for review records.

Create JSP pages (addReview.jsp, viewReview.jsp, updateReview.jsp, deleteReview.jsp) for each operation.

Each operation should correspond to a servlet (AddReviewServlet, ViewReviewServlet, UpdateReviewServlet, DeleteReviewServlet).

User Interface:

Create HTML forms for adding and updating review information (addReviewForm.html, updateReviewForm.html).

Display review records in a tabular format on the view page (viewReview.jsp).

Provide options to edit or delete each review.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

19) Lab Exercise: Expense Management System

Objective: Develop an Expense Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage their expenses including adding new expenses, viewing expense details, updating expense information, and deleting expenses.

Instructions:

Database Setup:

Create a MySQL database named ExpenseDB.

Create a table named Expense with the following schema:

```
CREATE TABLE Expense (
    expense_id INT PRIMARY KEY AUTO_INCREMENT,
    date DATE,
    amount DECIMAL(10, 2),
    description TEXT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for expense records.

Create JSP pages (addExpense.jsp, viewExpense.jsp, updateExpense.jsp, deleteExpense.jsp) for each operation.

Each operation should correspond to a servlet (AddExpenseServlet, ViewExpenseServlet, UpdateExpenseServlet, DeleteExpenseServlet).

User Interface:

Create HTML forms for adding and updating expense information (addExpenseForm.html, updateExpenseForm.html).

Display expense records in a tabular format on the view page (viewExpense.jsp).

Provide options to edit or delete each expense.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

20) Lab Exercise: Asset Management System

Objective: Develop an Asset Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage assets including adding new assets, viewing asset details, updating asset information, and deleting assets.

Instructions:

Database Setup:

Create a MySQL database named AssetDB.

Create a table named Asset with the following schema:

```
CREATE TABLE Asset (
    asset_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    value DECIMAL(10, 2)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for asset records.

Create JSP pages (addAsset.jsp, viewAsset.jsp, updateAsset.jsp, deleteAsset.jsp) for each operation.

Each operation should correspond to a servlet (AddAssetServlet, ViewAssetServlet, UpdateAssetServlet, DeleteAssetServlet).

User Interface:

Create HTML forms for adding and updating asset information (addAssetForm.html, updateAssetForm.html).

Display asset records in a tabular format on the view page (viewAsset.jsp).

Provide options to edit or delete each asset.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

21) Lab Exercise: Issue Tracking System

Objective: Develop an Issue Tracking System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage issues including adding new issues, viewing issue details, updating issue information, and deleting issues.

Instructions:

Database Setup:

Create a MySQL database named IssueDB.

Create a table named Issue with the following schema:

```
CREATE TABLE Issue (
    issue_id INT PRIMARY KEY AUTO_INCREMENT,
    description TEXT,
    status VARCHAR(50),
    priority VARCHAR(50),
    assigned_to INT,
    FOREIGN KEY (assigned_to) REFERENCES Employee(employee_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for issue records.

Create JSP pages (addIssue.jsp, viewIssue.jsp, updateIssue.jsp, deleteIssue.jsp) for each operation.

Each operation should correspond to a servlet (AddIssueServlet, ViewIssueServlet, UpdateIssueServlet, DeleteIssueServlet).

User Interface:

Create HTML forms for adding and updating issue information (addIssueForm.html, updateIssueForm.html).

Display issue records in a tabular format on the view page (viewIssue.jsp).

Provide options to edit or delete each issue.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

22) Lab Exercise: Policy Management System

Objective: Develop a Policy Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage insurance policies including adding new policies, viewing policy details, updating policy information, and deleting policies.

Instructions:

Database Setup:

Create a MySQL database named PolicyDB.

Create a table named Policy with the following schema:

```
CREATE TABLE Policy (
    policy_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    coverage_amount DECIMAL(10, 2),
    premium_amount DECIMAL(10, 2)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for policy records.

Create JSP pages (addPolicy.jsp, viewPolicy.jsp, updatePolicy.jsp, deletePolicy.jsp) for each operation.

Each operation should correspond to a servlet (AddPolicyServlet, ViewPolicyServlet, UpdatePolicyServlet, DeletePolicyServlet).

User Interface:

Create HTML forms for adding and updating policy information (addPolicyForm.html, updatePolicyForm.html).

Display policy records in a tabular format on the view page (viewPolicy.jsp).

Provide options to edit or delete each policy.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

23) Lab Exercise: Ticket Management System

Objective: Develop a Ticket Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage tickets including adding new tickets, viewing ticket details, updating ticket information, and deleting tickets.

Instructions:

Database Setup:

Create a MySQL database named TicketDB.

Create a table named Ticket with the following schema:

```
CREATE TABLE Ticket (
    ticket_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    priority VARCHAR(50),
    status VARCHAR(50),
    assigned_to INT,
    FOREIGN KEY (assigned_to) REFERENCES Employee(employee_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for ticket records.

Create JSP pages (addTicket.jsp, viewTicket.jsp, updateTicket.jsp, deleteTicket.jsp) for each operation.

Each operation should correspond to a servlet (AddTicketServlet, ViewTicketServlet, UpdateTicketServlet, DeleteTicketServlet).

User Interface:

Create HTML forms for adding and updating ticket information (addTicketForm.html, updateTicketForm.html).

Display ticket records in a tabular format on the view page (viewTicket.jsp).

Provide options to edit or delete each ticket.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

24) Lab Exercise: Membership Management System

Objective: Develop a Membership Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage memberships including adding new memberships, viewing membership details, updating membership information, and deleting memberships.

Instructions:

Database Setup:

Create a MySQL database named MembershipDB.

Create a table named Membership with the following schema:

```
CREATE TABLE Membership (
    membership_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    start_date DATE,
    end_date DATE,
    type VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for membership records.

Create JSP pages (addMembership.jsp, viewMembership.jsp, updateMembership.jsp, deleteMembership.jsp) for each operation.

Each operation should correspond to a servlet (AddMembershipServlet, ViewMembershipServlet, UpdateMembershipServlet, DeleteMembershipServlet).

User Interface:

Create HTML forms for adding and updating membership information (addMembershipForm.html, updateMembershipForm.html).

Display membership records in a tabular format on the view page (viewMembership.jsp).

Provide options to edit or delete each membership.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

25) Lab Exercise: Account Management System

Objective: Develop an Account Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage accounts including adding new accounts, viewing account details, updating account information, and deleting accounts.

Instructions:

Database Setup:

Create a MySQL database named AccountDB.

Create a table named Account with the following schema:

```
CREATE TABLE Account (
    account_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    account_number VARCHAR(255),
    balance DECIMAL(10, 2),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for account records.

Create JSP pages (addAccount.jsp, viewAccount.jsp, updateAccount.jsp, deleteAccount.jsp) for each operation.

Each operation should correspond to a servlet (AddAccountServlet, ViewAccountServlet, UpdateAccountServlet, DeleteAccountServlet).

User Interface:

Create HTML forms for adding and updating account information (addAccountForm.html, updateAccountForm.html).

Display account records in a tabular format on the view page (viewAccount.jsp).

Provide options to edit or delete each account.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

26) Lab Exercise: Messaging System

Objective: Develop a Messaging System using HTML, JSP, and Servlets to facilitate communication between employees and customers. The system should allow users to send and receive messages, view message details, and manage their inbox.

Instructions:

Database Setup:

Create a MySQL database named MessagingDB.

Create a table named Message with the following schema:

```
CREATE TABLE Message (
    message_id INT PRIMARY KEY AUTO_INCREMENT,
    sender_id INT,
    receiver_id INT,
    subject VARCHAR(255),
    content TEXT,
    date DATETIME,
    FOREIGN KEY (sender_id) REFERENCES Employee(employee_id),
    FOREIGN KEY (receiver_id) REFERENCES Customer(customer_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for messages.

Create JSP pages (sendMessage.jsp, inbox.jsp, viewMessage.jsp, deleteMessage.jsp) for each operation.

Each operation should correspond to a servlet (SendMessageServlet, InboxServlet, ViewMessageServlet, DeleteMessageServlet).

User Interface:

Create HTML forms for sending messages (sendMessageForm.html).

Display received messages in a list format on the inbox page (inbox.jsp).

Provide options to view and delete each message.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access messaging features.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

27) Lab Exercise: Recipe Management System

Objective: Develop a Recipe Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage recipes including adding new recipes, viewing recipe details, updating recipe information, and deleting recipes.

Instructions:

Database Setup:

Create a MySQL database named RecipeDB.

Create a table named Recipe with the following schema:

```
CREATE TABLE Recipe (
    recipe_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    ingredients TEXT,
    instructions TEXT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for recipe records.

Create JSP pages (addRecipe.jsp, viewRecipe.jsp, updateRecipe.jsp, deleteRecipe.jsp) for each operation.

Each operation should correspond to a servlet (AddRecipeServlet, ViewRecipeServlet, UpdateRecipeServlet, DeleteRecipeServlet).

User Interface:

Create HTML forms for adding and updating recipe information (addRecipeForm.html, updateRecipeForm.html).

Display recipe records in a tabular format on the view page (viewRecipe.jsp).

Provide options to edit or delete each recipe.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

28) Lab Exercise: Blog Management System

Objective: Develop a Blog Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage blog posts including adding new posts, viewing post details, updating post information, and deleting posts.

Instructions:

Database Setup:

Create a MySQL database named BlogDB.

Create a table named BlogPost with the following schema:

```
CREATE TABLE BlogPost (
    post_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(255) NOT NULL,
    content TEXT,
    author_id INT,
    date DATE,
    FOREIGN KEY (author_id) REFERENCES Employee(employee_id) ON DELETE CASCADE,
    FOREIGN KEY (author_id) REFERENCES Customer(customer_id) ON DELETE CASCADE
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for blog post records.

Create JSP pages (addPost.jsp, viewPost.jsp, updatePost.jsp, deletePost.jsp) for each operation.

Each operation should correspond to a servlet (AddPostServlet, ViewPostServlet, UpdatePostServlet, DeletePostServlet).

User Interface:

Create HTML forms for adding and updating blog post information (addPostForm.html, updatePostForm.html).

Display blog post records in a list format on the view page (viewPost.jsp).

Provide options to view, edit, or delete each post.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

29) Lab Exercise: Medical Record Management System

Objective: Develop a Medical Record Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage medical records including adding new records, viewing record details, updating record information, and deleting records.

Instructions:

Database Setup:

Create a MySQL database named MedicalRecordDB.

Create a table named MedicalRecord with the following schema:

```
CREATE TABLE MedicalRecord (
    record_id INT PRIMARY KEY AUTO_INCREMENT,
    patient_id INT,
    doctor_id INT,
    date DATE,
    diagnosis TEXT,
    treatment TEXT,
    FOREIGN KEY (patient_id) REFERENCES Patient(patient_id),
    FOREIGN KEY (doctor_id) REFERENCES Doctor(doctor_id)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for medical record entries.

Create JSP pages (addRecord.jsp, viewRecord.jsp, updateRecord.jsp, deleteRecord.jsp) for each operation.

Each operation should correspond to a servlet (AddRecordServlet, ViewRecordServlet, UpdateRecordServlet, DeleteRecordServlet).

User Interface:

Create HTML forms for adding and updating medical record information (addRecordForm.html, updateRecordForm.html).

Display medical record entries in a tabular format on the view page (viewRecord.jsp).

Provide options to edit or delete each record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

30) Lab Exercise: Patient Management System

Objective: Develop a Patient Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage patient records including adding new patients, viewing patient details, updating patient information, and deleting patients.

Instructions:

Database Setup:

Create a MySQL database named PatientDB.

Create a table named Patient with the following schema:

```
CREATE TABLE Patient (
    patient_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    date_of_birth DATE,
    gender VARCHAR(10),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address TEXT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for patient records.

Create JSP pages (addPatient.jsp, viewPatient.jsp, updatePatient.jsp, deletePatient.jsp) for each operation.

Each operation should correspond to a servlet (AddPatientServlet, ViewPatientServlet, UpdatePatientServlet, DeletePatientServlet).

User Interface:

Create HTML forms for adding and updating patient information (addPatientForm.html, updatePatientForm.html).

Display patient records in a tabular format on the view page (viewPatient.jsp).

Provide options to edit or delete each patient record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

31) Lab Exercise: Flight Management System

Objective: Develop a Flight Management System using HTML, JSP, and Servlets with CRUD operations. The system should allow users to manage flight information including adding new flights, viewing flight details, updating flight information, and deleting flights.

Instructions:

Database Setup:

Create a MySQL database named FlightDB.

Create a table named Flight with the following schema:

```
CREATE TABLE Flight (
    flight_id INT PRIMARY KEY AUTO_INCREMENT,
    airline VARCHAR(255) NOT NULL,
    flight_number VARCHAR(20) NOT NULL,
    departure_airport VARCHAR(255) NOT NULL,
    arrival_airport VARCHAR(255) NOT NULL,
    departure_date DATETIME NOT NULL,
    arrival_date DATETIME NOT NULL
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for flight records.

Create JSP pages (addFlight.jsp, viewFlight.jsp, updateFlight.jsp, deleteFlight.jsp) for each operation.

Each operation should correspond to a servlet (AddFlightServlet, ViewFlightServlet, UpdateFlightServlet, DeleteFlightServlet).

User Interface:

Create HTML forms for adding and updating flight information (addFlightForm.html, updateFlightForm.html).

Display flight records in a tabular format on the view page (viewFlight.jsp).

Provide options to edit or delete each flight record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

32) Lab Exercise: Weather Monitoring System

Objective: Develop a Weather Monitoring System using HTML, JSP, and Servlets to track weather conditions for different locations. The system should allow users to manage weather data including adding new records, viewing weather details, updating weather information, and deleting records.

Instructions:

Database Setup:

Create a MySQL database named WeatherDB.

Create a table named Weather with the following schema:

```
CREATE TABLE Weather (
    weather_id INT PRIMARY KEY AUTO_INCREMENT,
    location_id INT,
    temperature DECIMAL(5, 2),
    humidity DECIMAL(5, 2),
    wind_speed DECIMAL(5, 2),
    condition VARCHAR(255),
    observation_time DATETIME,
    FOREIGN KEY (location_id) REFERENCES Location(location_id)
);
```

CRUD Operations:

Implement Create, Read, Update, and Delete operations for weather records.

Create JSP pages (addWeather.jsp, viewWeather.jsp, updateWeather.jsp, deleteWeather.jsp) for each operation.

Each operation should correspond to a servlet (AddWeatherServlet, ViewWeatherServlet, UpdateWeatherServlet, DeleteWeatherServlet).

User Interface:

Create HTML forms for adding and updating weather information (addWeatherForm.html, updateWeatherForm.html).

Display weather records in a tabular format on the view page (viewWeather.jsp).

Provide options to edit or delete each weather record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

33) Lab Exercise: Room Booking System

Objective: Develop a Room Booking System using HTML, JSP, and Servlets to manage room reservations in a hotel. The system should allow users to add new rooms, view room details, update room information, and delete rooms.

Instructions:

Database Setup:

Create a MySQL database named HotelDB.

Create a table named Room with the following schema:

```
CREATE TABLE Room (
    room_id INT PRIMARY KEY AUTO_INCREMENT,
    room_number VARCHAR(10) NOT NULL,
    type VARCHAR(50),
    rate_per_night DECIMAL(8, 2),
    availability_status VARCHAR(20)
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for room records.

Create JSP pages (addRoom.jsp, viewRoom.jsp, updateRoom.jsp, deleteRoom.jsp) for each operation.

Each operation should correspond to a servlet (AddRoomServlet, ViewRoomServlet, UpdateRoomServlet, DeleteRoomServlet).

User Interface:

Create HTML forms for adding and updating room information (addRoomForm.html, updateRoomForm.html).

Display room records in a tabular format on the view page (viewRoom.jsp).

Provide options to edit or delete each room record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet).

34) Lab Exercise: Car Rental Management System

Objective: Develop a Car Rental Management System using HTML, JSP, and Servlets to manage car rentals for a rental agency. The system should allow users to add new cars, view car details, update car information, and delete cars.

Instructions:

Database Setup:

Create a MySQL database named CarRentalDB.

Create a table named Car with the following schema:

```
CREATE TABLE Car (
    car_id INT PRIMARY KEY AUTO_INCREMENT,
    make VARCHAR(50) NOT NULL,
    model VARCHAR(50) NOT NULL,
    year INT NOT NULL,
    rental_rate_per_day DECIMAL(8, 2) NOT NULL,
    availability_status VARCHAR(20) NOT NULL
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for car records.

Create JSP pages (addCar.jsp, viewCar.jsp, updateCar.jsp, deleteCar.jsp) for each operation.

Each operation should correspond to a servlet (AddCarServlet, ViewCarServlet, UpdateCarServlet, DeleteCarServlet).

User Interface:

Create HTML forms for adding and updating car information (addCarForm.html, updateCarForm.html).

Display car records in a tabular format on the view page (viewCar.jsp).

Provide options to edit or delete each car record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet)(navigation menu should be visible in every page by using include servlet).

35) Lab Exercise: Bike Rental Management System

Objective: Develop a Bike Rental Management System using HTML, JSP, and Servlets to manage bike rentals for a rental agency. The system should allow users to add new bikes, view bike details, update bike information, and delete bikes.

Instructions:

Database Setup:

Create a MySQL database named BikeRentalDB.

Create a table named Bike with the following schema:

```
CREATE TABLE Bike (
    bike_id INT PRIMARY KEY AUTO_INCREMENT,
    make VARCHAR(50) NOT NULL,
    model VARCHAR(50) NOT NULL,
    year INT NOT NULL,
    rental_rate_per_day DECIMAL(8, 2) NOT NULL,
    availability_status VARCHAR(20) NOT NULL
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for bike records.

Create JSP pages (addBike.jsp, viewBike.jsp, updateBike.jsp, deleteBike.jsp) for each operation.

Each operation should correspond to a servlet (AddBikeServlet, ViewBikeServlet, UpdateBikeServlet, DeleteBikeServlet).

User Interface:

Create HTML forms for adding and updating bike information (addBikeForm.html, updateBikeForm.html).

Display bike records in a tabular format on the view page (viewBike.jsp).

Provide options to edit or delete each bike record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet)(navigation menu should be visible in every page by using include servlet).

36) Lab Exercise: Faculty Management System

Objective: Develop a Faculty Management System using HTML, JSP, and Servlets to manage faculty information for an educational institution. The system should allow administrators to add new faculty members, view faculty details, update faculty information, and delete faculty records.

Instructions:

Database Setup:

Create a MySQL database named FacultyDB.

Create a table named Faculty with the following schema:

```
CREATE TABLE Faculty (
    faculty_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    designation VARCHAR(100) NOT NULL,
    department VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    phone_number VARCHAR(20),
    joining_date DATE NOT NULL,
    qualification TEXT
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for faculty records.

Create JSP pages (addFaculty.jsp, viewFaculty.jsp, updateFaculty.jsp, deleteFaculty.jsp) for each operation.

Each operation should correspond to a servlet (AddFacultyServlet, ViewFacultyServlet, UpdateFacultyServlet, DeleteFacultyServlet).

User Interface:

Design user-friendly HTML forms for adding and updating faculty information (addFacultyForm.html, updateFacultyForm.html).

Display faculty records in a tabular format on the view page (viewFaculty.jsp).

Provide options to edit or delete each faculty record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet)(navigation menu should be visible in every page by using include servlet).

37) Lab Exercise: Outsourcing Staff Management System

Objective: Develop an Outsourcing Staff Management System using HTML, JSP, and Servlets to manage information about outsourced staff members for a company. The system should allow administrators to add new staff members, view staff details, update staff information, and delete staff records.

Instructions:

Database Setup:

Create a MySQL database named OutsourcingStaffDB.

Create a table named OutsourcingStaff with the following schema:

```
CREATE TABLE OutsourcingStaff (
    staff_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    role VARCHAR(100) NOT NULL,
    department VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    phone_number VARCHAR(20),
    contract_start_date DATE NOT NULL,
    contract_end_date DATE NOT NULL,
    hourly_rate DECIMAL(8, 2) NOT NULL
);
```

Login Page:

Create an HTML login page (login.html) with fields for roll number and password.

Validate the credentials username and password is your roll number. (Servlet: LoginServlet)

Upon successful login, store user information in session or cookies for the entire session.

CRUD Operations:

Implement Create, Read, Update, and Delete operations for staff records.

Create JSP pages (addStaff.jsp, viewStaff.jsp, updateStaff.jsp, deleteStaff.jsp) for each operation.

Each operation should correspond to a servlet (AddStaffServlet, ViewStaffServlet, UpdateStaffServlet, DeleteStaffServlet).

User Interface:

Design user-friendly HTML forms for adding and updating staff information (addStaffForm.html, updateStaffForm.html).

Display staff records in a tabular format on the view page (viewStaff.jsp).

Provide options to edit or delete each staff record.

Navigation:

Create a navigation menu on the homepage (home.jsp) to access CRUD operations.

Each menu option should link to the corresponding JSP page(navigation menu should be visible in every page by using include servlet)(navigation menu should be visible in every page by using include servlet).