**Al-Zaytoonah University of Jordan**

**Faculty of Science and Information Technology**

**Department of COMPUTER SCINECE**



**An Online Real Estate Platform for Property Sales (Generations of Tomorrow`s Website)**

**Prepard By**

**Mohammad Darweesh Mousa Mohammad (202020824)**

**Leena Osama Mostafa Al-toubah (202020521)**

**Maryam Issam Khames Abu Ruqah (202010146)**

**Supervised by**

**Dr. Ali Al-Daoud**

**A GRADUATION PROJECT SUBMITTED IN PARTIAL FULFILMENT OF HE REQUIREMENTS FOR THE DEGREE OF BSC IN COMPUTER SCINECE**

## Abstract:

## The Alhada Estate website project aims to facilitate online real estate transactions, allowing buyers to preview and gather comprehensive details about properties before contacting the owners. The website enables users to navigate virtually and view all available options, empowering them to choose a property based on location, size, and price. The standout feature of this project lies in providing buyers the comfort to make purchase decisions without the need to visit real estate sites or be influenced by property brokers. Buyers can select their property from the convenience of their own location, aiding them in making optimal choices without feeling psychological or physical pressure.

## Project Objectives:

## The project aims to simplify and facilitate real estate purchasing processes through the use of technology and the

## internet, making the entire process more convenient and seamless for buyers.

## Property Preview Capability: The project also aims to provide a means for previewing properties online, allowing

## users to browse and view properties with their details and images before making a purchase decision.

## Target Audience:

## Property Buyers: This category includes individuals searching to purchase residential or commercial properties,

## whether they are real estate investors or individuals looking to own a home or property for residential or

## investment purposes.

## Individuals Wanting to Showcase Their Properties for Sale: The project also targets property owners who

## want to showcase their properties for sale, whether they are individuals with personal properties or companies

## looking to sell commercial real estate.

## Acknowledgment

I think my college trip is already over today, after a long period of exhaustion and hardship. Today, I'm finalizing my graduation research at full capacity In me all appreciation and gratitude to everyone who was credited with my career and helped me. To those who paved the way for knowledge of all my distinguished teachers.

* Especially thanks and appreciation to Dr. Ali AL Dahoud, Thank you for the assistance you provided and for your belief in my project.
* Special thanks to Dean Ayesh alhroub for providing a suitable environment that enables us to succeed and finish the requirements of this project.

Thanks To Everyone….

**Table of Contents**

[Abstract 2](#_bookmark0)

[Acknowledgment 3](#_bookmark1)

[Table of Contents 4](#_bookmark2)

[List of Figures 6](#_TOC_250006)

[List of Tables 7](#_TOC_250005)

1. [CHAPTER ONE: INTRODUCTION 8](#_bookmark3)
   1. [Background 8](#_bookmark4)
   2. [Business Model 10](#_bookmark6)
      1. [Business Needs 10](#_bookmark7)
      2. [Business Environment 10](#_bookmark8)
      3. [Stakeholder Analysis 12](#_bookmark9)
      4. [System Vision Document 12](#_bookmark10)
      5. [Project Management 13](#_bookmark11)
      6. [Development Environment 14](#_bookmark12)
2. [CHAPTER TWO: REQUIREMENTS 17](#_bookmark13)
   1. [Gathering Requirements 17](#_bookmark14)
   2. [System Problem Statement 17](#_bookmark15)
   3. [Functional, Non-Functional Requirements and System Constraints 17](#_bookmark16)
   4. [System Models 20](#_bookmark17)
      1. [Use Case Model 20](#_TOC_250004)
      2. [Use Cases Descriptions "Scenarios" 21](#_TOC_250003)
      3. [Activity & System Sequence Models 24](#_TOC_250002)
      4. [Domain Class Model 28](#_TOC_250001)
3. [CHAPTER THREE: DESIGN 29](#_bookmark18)
   1. [Architecture and Deployment environment Design 29](#_bookmark19)
   2. [Software Architecture Design 29](#_bookmark20)
   3. [Design the database 30](#_bookmark21)
      1. [Design Entity-Relationship Model 30](#_bookmark22)
   4. Design the system and user interfaces………………………………………………….31
4. [CHAPTER FIVE: Analysis And TESTING 76](#_bookmark27)
   1. [Unit Testing 76](#_bookmark28)
      1. [Set cases test 76](#_bookmark29)
   2. [Integration Testing 76](#_bookmark30)
   3. [System Testing 76](#_TOC_250000)
5. [CONCLUSIONS AND FUTURE WORK 77](#_bookmark31)
   1. [Conclusion 77](#_bookmark32)
   2. [Future work 77](#_bookmark33)

### **List of Figures:**

|  |  |  |
| --- | --- | --- |
| Figure1:Softwer Context Diagram0 | ………………………………………………………………………………………. | 11 |
| Figure2: Softwer Context Diagram1 | ………………………………………………………………………………………. | 11 |
| Figure-3:Difference between Web and MVC | ………………………………………………………………………………………. | 15 |
| Figure -4:what the MVC Patter | ………………………………………………………………………………………. | 16 |
| Figure-5:Why Acp.net core(MVC) | ………………………………………………………………………………………. | 16 |
| Figure-6:Use Case (Admin) | ………………………………………………………………………………………. | 20 |
| Figure-7: Use Case (User) | ………………………………………………………………………………………. | 20 |
| Figure-8: Use Case (EMP) | ………………………………………………………………………………………. | 21 |
| Figure-9:Activty Admin Login | ………………………………………………………………………………………. | 24 |
| Figure-10:Sequence Admin Login | ………………………………………………………………………………………. | 24 |
| Figure-11: Activty Admin Add State | ………………………………………………………………………………………. | 25 |
| Figure-12:Sequence Add State Admin | ………………………………………………………………………………………. | 25 |
| Figure-13:Acticty Employee Login | ………………………………………………………………………………………. | 26 |
| Figure-14:Sequence Employee Login | ………………………………………………………………………………………. | 26 |
| Figure-15:Activty Add State Employee | ………………………………………………………………………………………. | 27 |
| Figure-16:Sequence Add State Employee | ………………………………………………………………………………………. | 27 |
| Figure-17:Domane Class Model | ………………………………………………………………………………………. | 28 |
| Figure-18: Deployment environment Design | ………………………………………………………………………………………. | 29 |
| Figure-19:Architecture Design | ………………………………………………………………………………………. | 29 |
| Figure-20:Design Entity-Relationship Model | ………………………………………………………………………………………. | 30 |
| Figure-21:Home Page | ………………………………………………………………………………………. | 31 |
| Figure-22:Login Page | ………………………………………………………………………………………. | 31 |
| Figure-23:Add New Land Page(Admin) | ………………………………………………………………………………………. | 32 |
| Figure-24:Add New E-state Page(admin) | ………………………………………………………………………………………. | 32 |
| Figure-25:Add New customer (Admin) | ………………………………………………………………………………………. | 33 |
| Figure-26:Search Page(Admin) | ………………………………………………………………………………………. | 33 |
| Figure-27:Add New Estate (Employee) | ………………………………………………………………………………………. | 34 |
| Figure-28:Add New Customer (Employee) | ………………………………………………………………………………………. | 34 |
| Figure-29:My Customer (Employee) | ………………………………………………………………………………………. | 35 |
| Figure-30:Users(Admin) | ………………………………………………………………………………………. | 35 |
| Figure-31:Add New User(Admin) | ………………………………………………………………………………………. | 36 |
| Figure-32:My Client(Admin) | ………………………………………………………………………………………. | 36 |
| Figure-33:My Profile | ………………………………………………………………………………………. | 37 |
| Figure-34:Inbox (Admin) | ………………………………………………………………………………………. | 37 |
| Figure-35:Contact US(User) | ………………………………………………………………………………………. | 38 |
| Figure-36:Search Land | ………………………………………………………………………………………. | 38 |
| Figure-37:Design the system Security | ………………………………………………………………………………………. | 39 |

### **List of Tables:**

|  |  |  |
| --- | --- | --- |
| Table1: Stakeholder Analysis | …………………………………………………………………………………………. | 12 |
| Table2: Development Tools | …………………………………………………………………………………………. | 14 |
| Table3: use Cases Descriptions(Admin) | …………………………………………………………………………………………. | 22 |
| Table4: use Cases Descriptions(emp) | …………………………………………………………………………………………. | 22 |
| Table5: use Cases Descriptions(user) | …………………………………………………………………………………………. | 23 |
| Table-6: Entity-Relationship model(emp) | …………………………………………………………………………………………. | 73 |
| Table-7: Entity-Relationship model(client) | …………………………………………………………………………………………. | 73 |
| Table8: Entity-Relations(stateImage) | …………………………………………………………………………………………. | 74 |
| Table9: Entity-Relationship model(state) | …………………………………………………………………………………………. | 74 |
| Table10: Entity-Relationship model(client Details) | …………………………………………………………………………………………. | 75 |
| Table11: Entity-Relationship model(client) | …………………………………………………………………………………………. | 75 |
| Table12: Set cases test | …………………………………………………………………………………………. | 76 |

# **CHAPTER ONE: INTRODUCTION**

### **Background**

The evolution of the internet and modern technology has brought about a significant transformation in the real

estate industry, allowing individuals and companies to showcase and purchase properties through dedicated

websites for online real estate sales. Here is an introduction to the benefits of these online real estate platforms:

* + 1. **Wider Access:**

Transitioning to online real estate platforms enables numerous users to access a vast and diverse array of

properties in various locations, cities, and countries. This provides property seekers with extensive opportunities

for selection and comparison among the available options.

* + 1. **Comprehensive Information:**

Online real estate sales platforms offer comprehensive information about listed properties, including

high-quality images, detailed descriptions, specifications, geographic locations, and prices. This assists

potential buyers in making informed decisions without the need to personally visit each property.

* + 1. **Time and Effort Savings:**

Thanks to detailed information and clear images available online, potential buyers can browse properties and

choose suitable options without spending excessive time on searching or traveling to physically visit properties.

* + 1. **Comparison Capability:**

The internet facilitates easy comparison of properties, allowing users to compare prices, specifications, and

locations among various available options, aiding them in making the best choice.

* + 1. **Secure Transactions**:

Online real estate sales platforms provide a secure environment to complete sales transactions, including legal

and financial procedures that ensure the safety of the deal for all involved parties.

* + 1. **Easy Communication:**

These platforms offer convenient communication channels with sellers or real estate agents, making

negotiation, information exchange, and inquiries more straightforward.

* + 1. **Continuous Updates:**

Online real estate sales platforms are regularly updated, meaning users receive new information and updated

listings periodically.

Summary, online real estate sales platforms offer a convenient and comprehensive platform

that combines the advantages of wide access and detailed information, making the process

of purchasing properties easier and more efficient for individuals and companies alike.

### **Business Model**

#### **Business Needs**

1. The real estate website should have a user-friendly interface that allows users to easily browse properties

and access information quickly and effortlessly.

1. The website should offer interactive tools such as virtual property tours or multimedia content like images

andvideos to showcase properties in an engaging and clear manner.

1. There should be tools available on the website for data analysis and reporting, providing statistics on sales

, real estate demands, and customer feedback to facilitate decision-making processes.

1. The website should provide tools to track the progress of property transactions, including information on

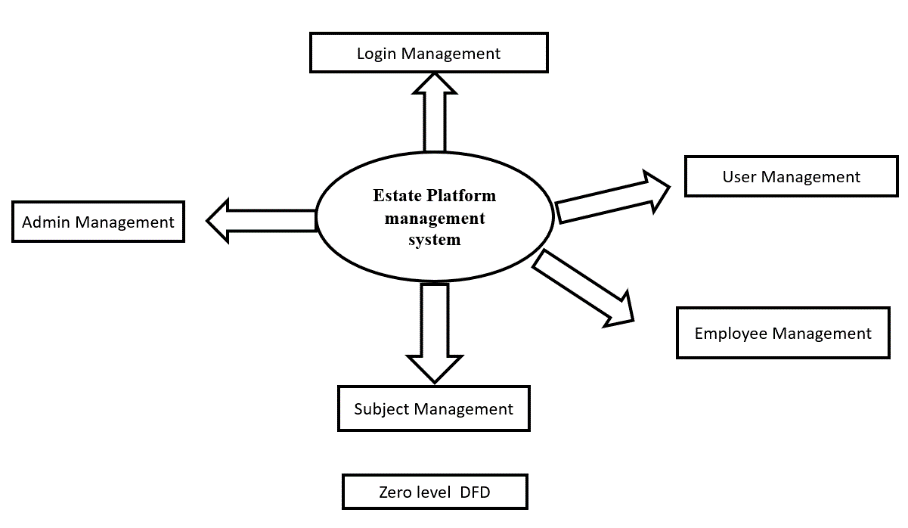
sale status, current listings, and customer/property records.

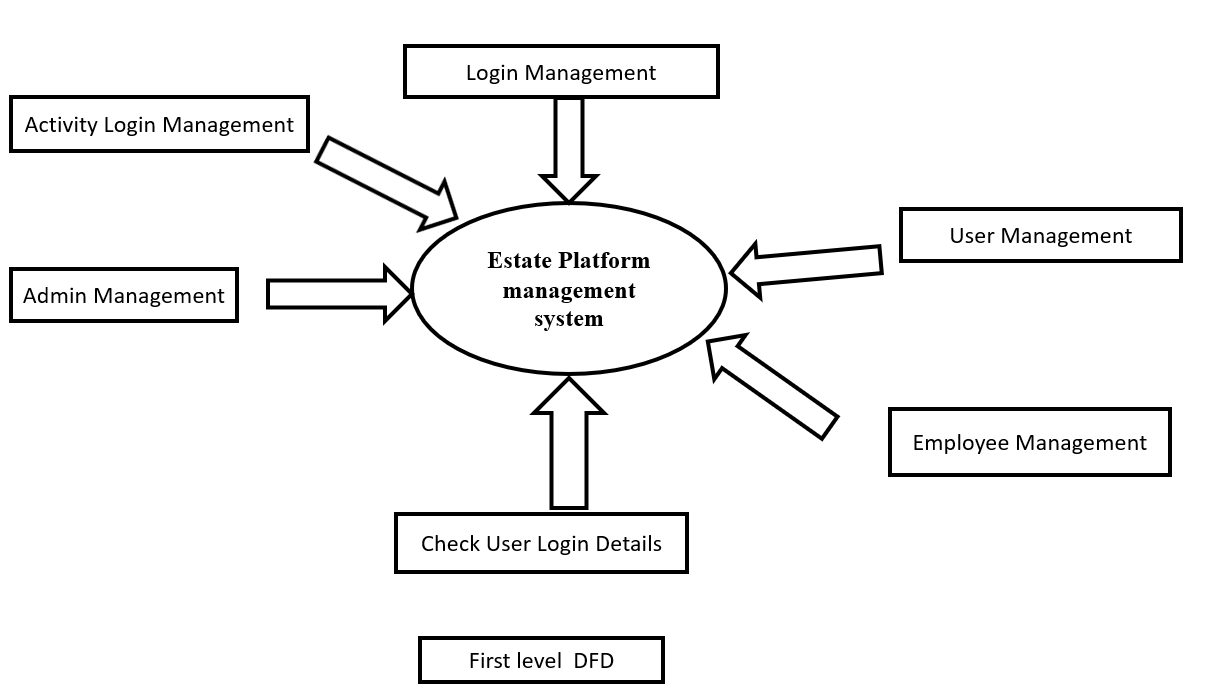
5- The website should offer customization options for users, enabling them to specify property search requirements and customize viewing schedules based on their preferences.

#### **1.2.2Business Environment**

Environment of computer platforms and the Internet in which adding states and lands take place which include synchronous and asynchronous courses. Estate platform that allows sharing of Estates and Lands promotions to all users who want to view or buy via the web.

##### **Software Context Diagram**



**Figure-1:Manag****ement System**

**Figure-2:Management System 1**

##### **System Scope**

The e-learning platform will solve flexibility, where you can complete your learning at the right time and place for you.

#### **Stakeholder Analysis**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Id** | **Position** | **Internal/External** | **Supports** | **Knowledge** | **Power** |
| 1 | Web Developer | Internal | Neutral | High | High |
| 2 | ASP.NET Developer | Internal | Neutral | High | High |
| 3 | Database Developer | Internal | Neutral | High | High |
| 4 | User | External | Neutral | Low | Low |
| 5 | Employee | External | Neutral | Low | Low |
| 6 | Administration | Internal | Neutral | High | High |

#### Table-1: Stakeholder Analysis

#### **System Vision Document**

##### **Objectives**

The main objective of an Estate website is to provide users with all estate and lands declared on website in order to save time and cost for clients and users also to keep data from all employee’s data (estates and lands)

##### **Benefits**

1. Keeping data Saving the employees transaction. This is done by retrieving all estates and lands that is found by employee and save employee’s commission without any problem
2. Admin controls all data from system by showing all transactions (adding estate or lands) in website with capability to add more employees .

from lost because they don’t require2d to use paper for saving data

##### **Capabilities**

1. An easy-to-use interface is easy to navigate, and users can easily find the information they need.
2. The website has a estates management system that allows administrators to know all states and lands in his systems.
3. Support of multiple content types (text, audio, images, video)
4. Structured storage of estate materials.

#### **Project Management**

Regardless of the size of the project, there are specific basic elements that a project manager must follow that fails that will frustrate his project with many problems and issues. Whether the intent of the project is to manage change, improve services, or implement systems, a successful project must include such basic considerations as strategic planning. Planning defines and understands the needs of the project and what it wants to achieve. Through this process, project managers are able to set specific, measurable and achievable goals.

##### **Project Iteration and Schedule**

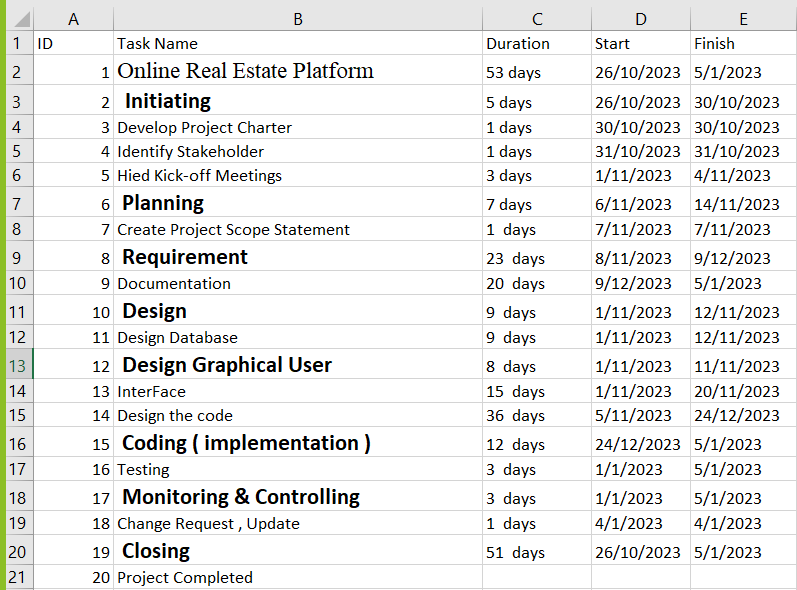
##### **Project Financial Feasibility**

Determining the financial feasibility of creating an e-state website, we need to consider the costs associated with the project as well as the potential revenue it can generate. The costs associated with establishing an e-learning site include:

Technology infrastructure: The first cost associated with creating an e-state site is the technology infrastructure required to host the site. This includes the cost of servers, website development and maintenance.

#### **Development Environment**

##### **Development Tools and Support Services**



**Table-2:** **Development Tools**

##### **Visual Studio**: an integrated development environment from Microsoft. It is used to develop computer

##### programs including websites, web apps, web services. Programming languages used ***(***ASp&Html &JavaScript & Css).

##### **SQL Server Management Studio (SSMS):** SQL Server Management Studio (SSMS) is a database

##### management tool compatible with the SQL Server system, developed by Microsoft. SSMS is an integrated and

##### powerful environment that allows developers and administrators to efficiently manage databases. Through SSMS, users

##### can create, edit, delete, and query databases, manage security and permissions, and perform maintenance and backup

##### of databases conveniently and effectively. This tool provides a user-friendly graphical interface that facilitates data

##### management and analysis in the SQL Server environment.

##### Asp.Net Core(MVC): .NET Core is a new version of .NET Framework .which is a free, open-source,

##### general-purpose development platform. maintained by Microsoft. It is a cross-platform framework that runs on

##### Windows, macOS, and Linux operating systems.

##### NET Core Framework can be used to build different types of applications such as mobile, desktop, web, cloud, IoT,

##### machine learning, microservices, game, etc.

##### .NET Core is written from scratch to make it modular, lightweight, fast, and cross-platform Framework.

##### Other features are provided as NuGet packages, which you can add it in your application as needed.

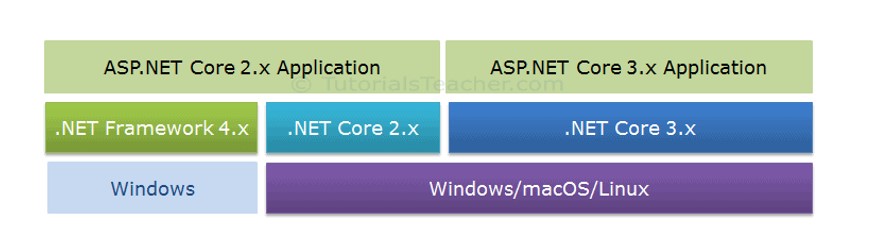
##### In this way, the .NET Core application speed up the performance, reduce the memory footprint and becomes easy to maintain.

##### -**Difference between Web Form and MVC Design**

**Figure-3:Difference between Web and MVC**

##### **System Development Process**

**Figure-4:What The MVC Pattern**

**figure-5:Why Asp.net Core(MVC)**

# **CHAPTER TWO: REQUIREMENTS**

### **Gathering Requirements**

The techniques we have used to gather requirements:

1. User observations.
2. Brainstorming
3. Prototyping.

### **System Problem Statement**

The Estate website aims to provide an alternative solution by providing a view to search to any state or lands from anywhere and at any time, with a wide range of states or lands in Jordan available. However, the design and implementation of an effective e-state website requires addressing a variety of technical and educational challenges, including platform design, adding estate or lands , Employees’ commission , evaluation and support.

### **Functional, Non-Functional Requirements and System Constraints**

**Functional Requirements**

In this section we will discuss different functional requirement of different user:

**1-Administration:**

* Adding And Manage View Estate
* Adding And Manage View Land.
* Ability to Logout.
* Ability to Contact US.
* Ability to Modify Estate.
* Ability to Modify Land.
* Ability to My Estate.
* Ability to MyLand.
* Ability to Modify Client.
* Ability to Edit Profile.
* Ability to View Users.
* Ability to Modify User

1. **User:**
   * Ability to View Estate.
   * Ability to Get Estate By(ID).
   * Ability to View Land.
   * Ability to Get Land By(ID)
   * Ability to Contant US.
   * Ability to Login.
   * Ability to Contact US.
2. **Employee:**

* Ability to View Estate.
* Ability View Land.
* Ability to Logout.
* Ability to Contact US.
* Ability to Modify EState.
* Ability to Modify Land.
* Ability to My Estate.
* Ability to MyLand.
* Ability to Modify Client.
* Ability to Edit Profile.

### Non-Functional Requirements

**Usability:**

* Easy to use and navigate, with clear and concise instructions and easy to use interface.
* No technical skills shall be required to use the system.

**Performance:**

The site should be quick and responsive, with minimal downtime, to ensure users have access to the site for services.

**Security:**

* The system should be secure, with measures in place to protect employee and administrator data, prevent hacking, and ensure that only authorized users have access to the system.
* The Email and password shall be required, only authorized people can access to their accounts.

**Efficiency:**

The system should respond in 3 seconds or less.

**Scalability:**

The system should be scalable, able to accommodate increasing numbers of users and data without compromising performance or speed.

**Reliability:**

The website should be reliable, with minimal errors or bugs, to ensure that users can complete their courses without interruption.

**Responsiveness:**

For every invalid input from the user, the system shall display a meaningful error message explaining the input format expected.

**Maintainability:**

The system components shall be able to change without breaking the whole system.

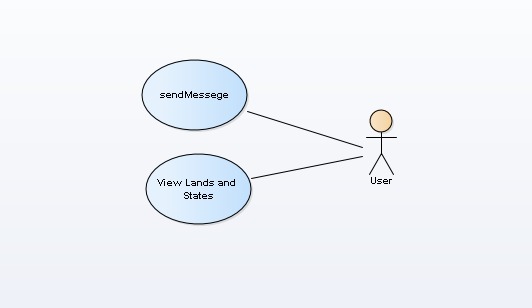
## Constraints

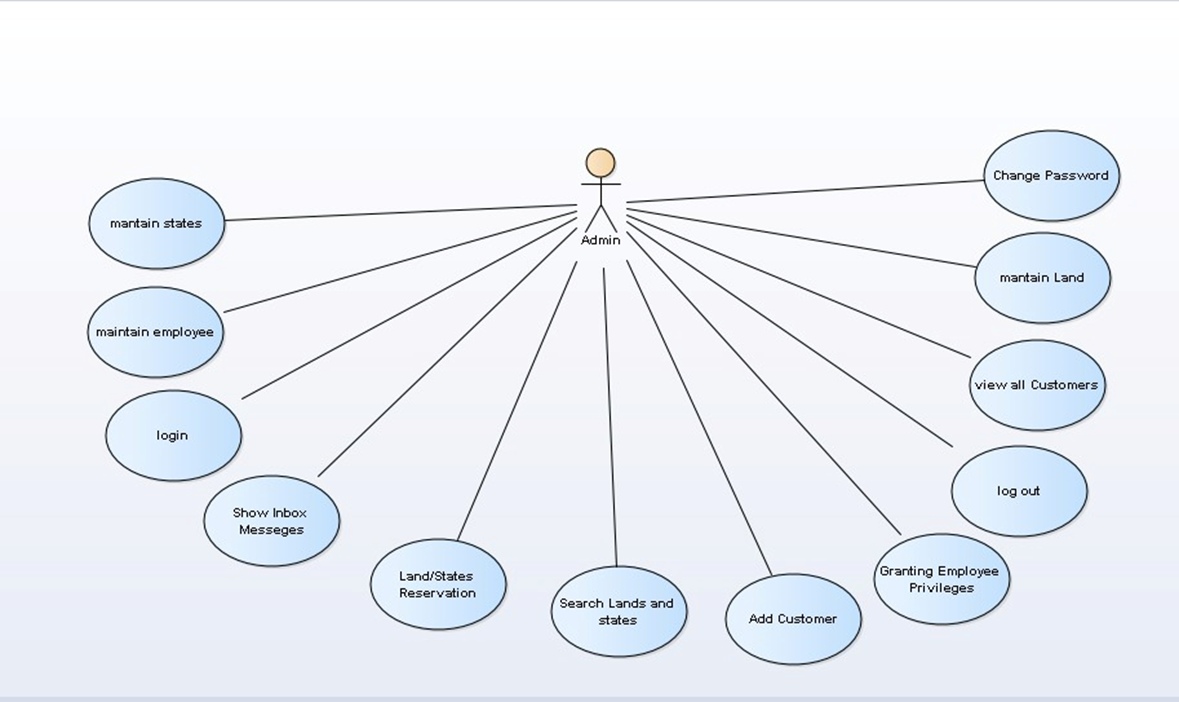
**Cost-effectiveness:**

The system should be cost-effective, with affordable pricing models and a clear return on investment for schools and educational institutions.

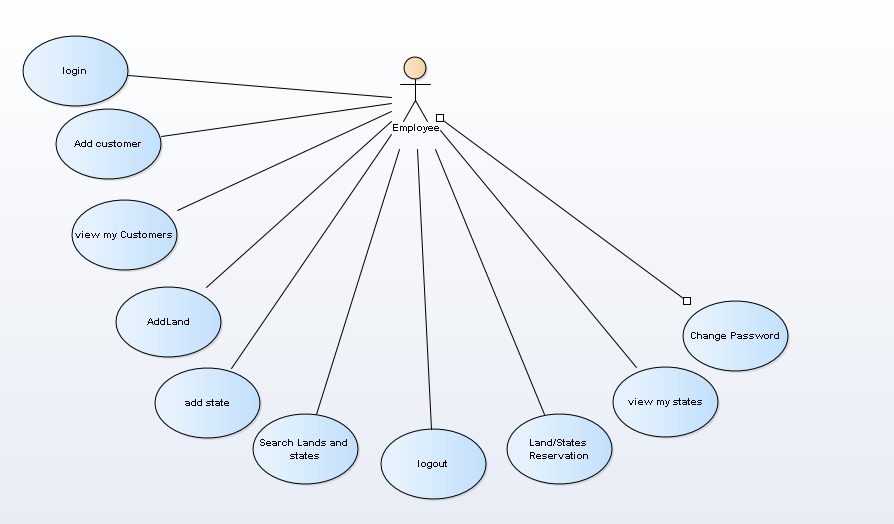
### **System Models**

#### **Use Case Model**



**Figure-6: Use Case Admin**

**Figure-7:Use Case User**



**Figure-8:Use Case Employee**

**use Cases Descriptions "Scenarios"**

|  |  |  |
| --- | --- | --- |
| **Actor Name** | **Use Case** | **Description** |
| **Administration** | **Login** | To Entering the Website |
| **Change Password** | The possibility of changing a  Admin password |
| **Add/Manage Teachers** | Ability to add and manage  Teachers |
| **Add/Manage Students** | Ability to add and manage  Students |
| **Add Class** | Ability to add classrooms |
| **Add Subject** | Ability to add topics |
| **Course Student/Teacher** | Ability to add Courses for  Teachers and Students |
| **LogOut** | Log-out the Website |

Table-3: use Cases Descriptions(Admin)

|  |  |  |
| --- | --- | --- |
| **Actor Name** | **Use Case** | **Description** |
| **Employee** | **Login** | To Entering the Website |
| **View Personal**  **Information** | Show Teacher information |
| **View Student**  **Information** | Show Students information |
| **Change Password** | The possibility of changing a teacher  password |
| **View Schedule** | Teachers can view the courses  schedule |
| **Add Exams** | Addition of exams to the student |
| **Add Lessons** | Add lessons to the student |
| **Show/hidden Marks** | Showing or hidden student Marks |
| **Add HomeWorks** | Adding homework to the student |
| **Competition** | Addition of student competition |
| **Add Marks** | Addition of course marks for the  student |
| **Add Meeting** | Creating a classroom for students |
| **Attendance** | Student attendance registration |
| **LogOut** | Log-out the Website |

Table4:useCasesDescriptions(emp)

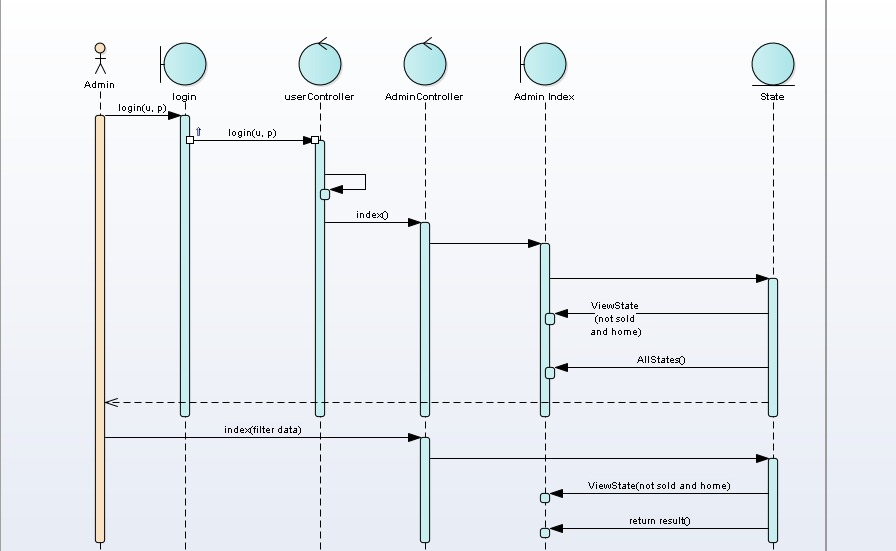
|  |  |  |
| --- | --- | --- |
| **Actor Name** | **Use Case** | **Description** |
| **User** | **Login** | To Entering the Website |
| **View Information** | Show student information |
| **Change Password** | The possibility of changing a student  password |
| **View Schedule** | Students can view the courses  Schedule |
| **Exams** | The student can Download the exam  and upload it on the Website |
| **Lessons** | View the lessons required from the  student |
| **Notifications** | Send notifications to the student |
| **HomeWorks** | Solving the homework required of the  student |
| **View Competion** | Student participation in competions |
| **View Marks** | Showing marks of courses to the  student |
| **Join Meeting** | Student entrance to the class |
| **View Attendence** | Showing the attendance of the  student |
| **View Total Marks** | Showing student total marks |
| **LogOut** | Log-out the Website |

**Table-5:** use Cases Descriptions(user)

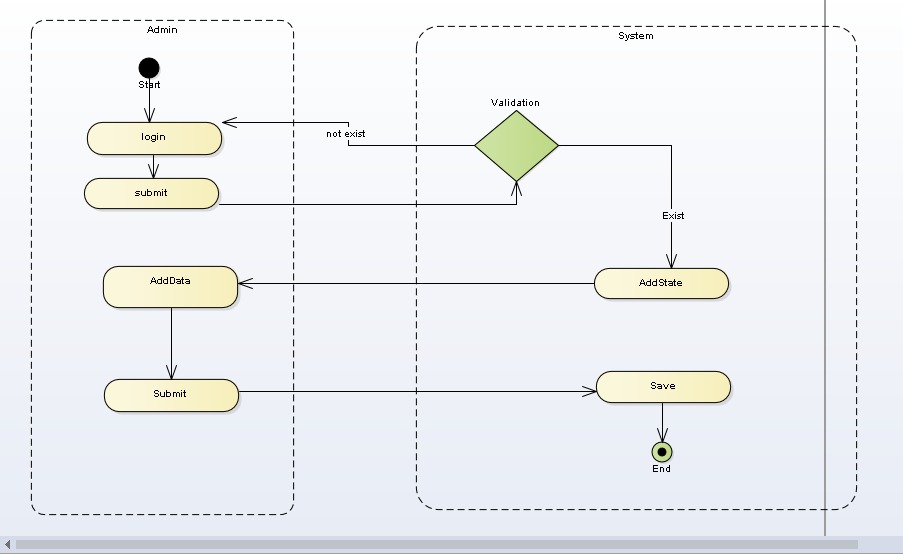
#### **Activity & System Sequence Models**

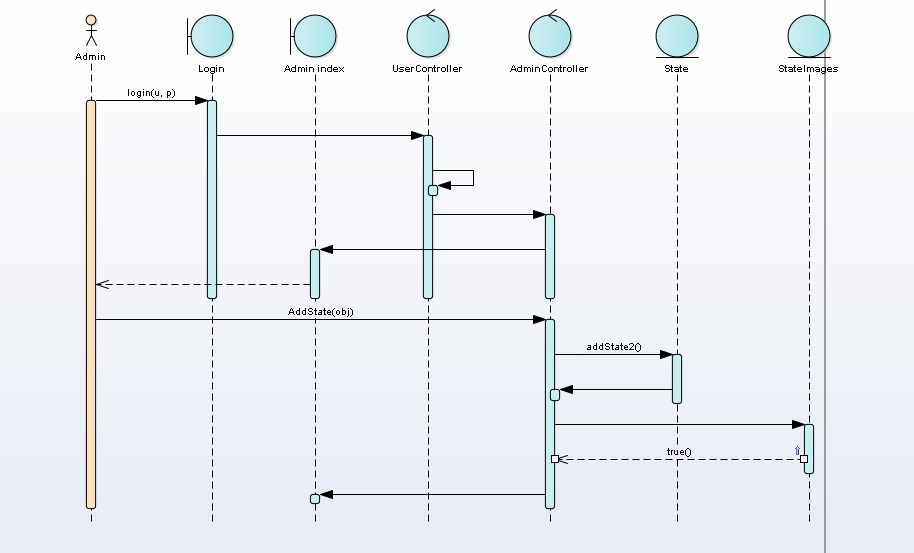
#### **-admin login**:

#### **-**

 **Figure-9:Activity Admin Login**

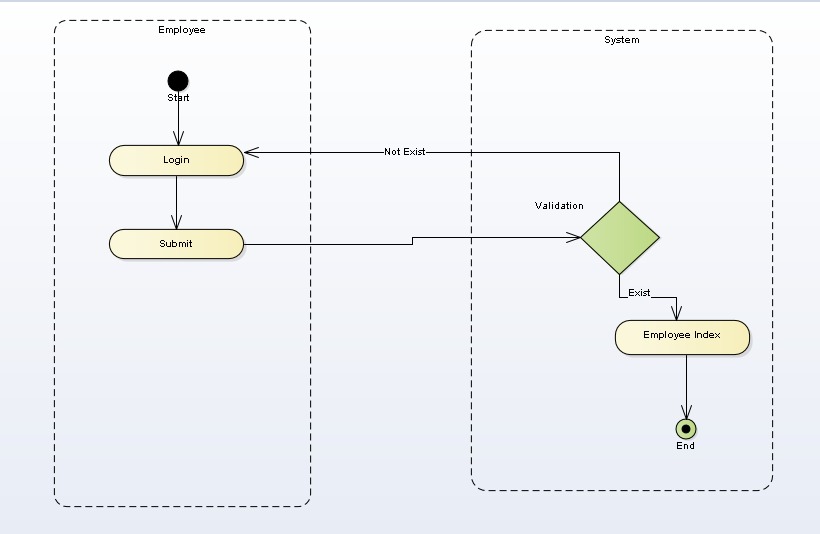
**Figure-10:** **Sequence Admin Login**

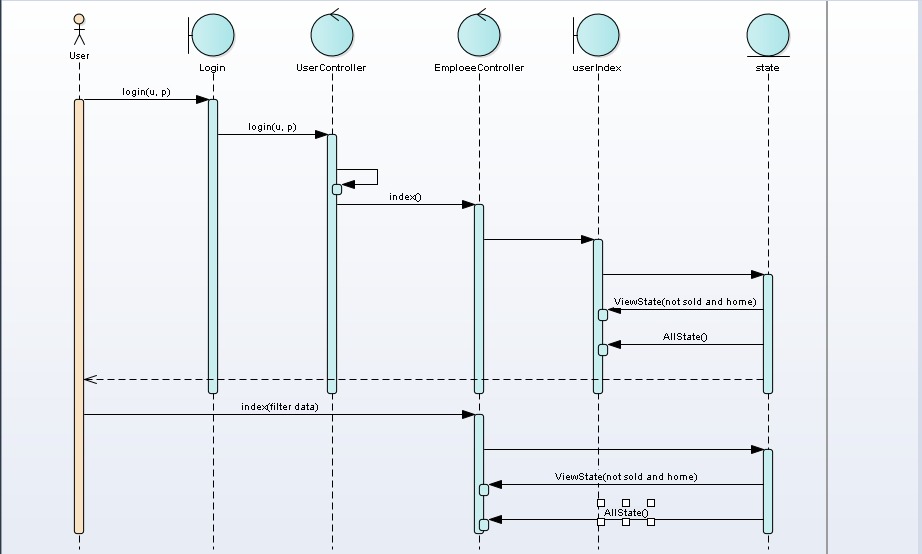
**-Admin Add State:**

**Figure-11:Activity Add State Admin**

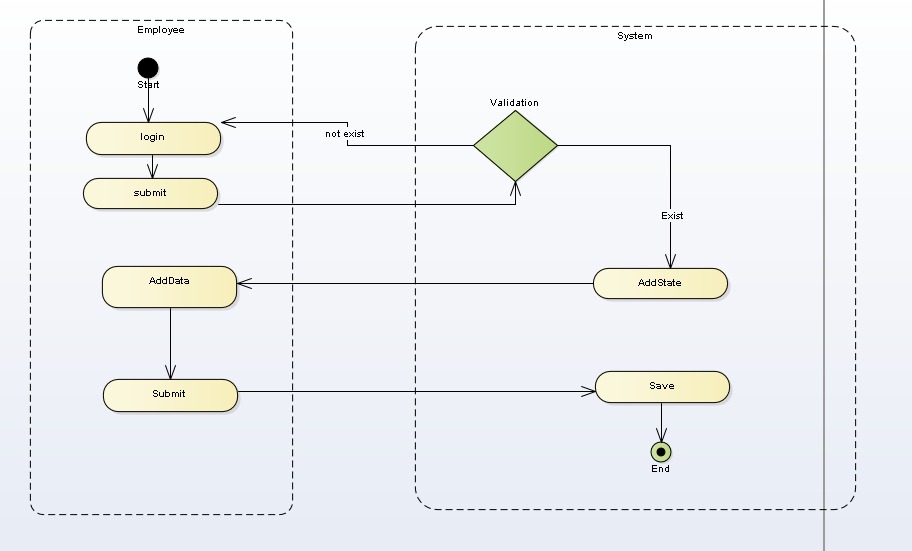
**Figure-12: Sequence Add State Admin**

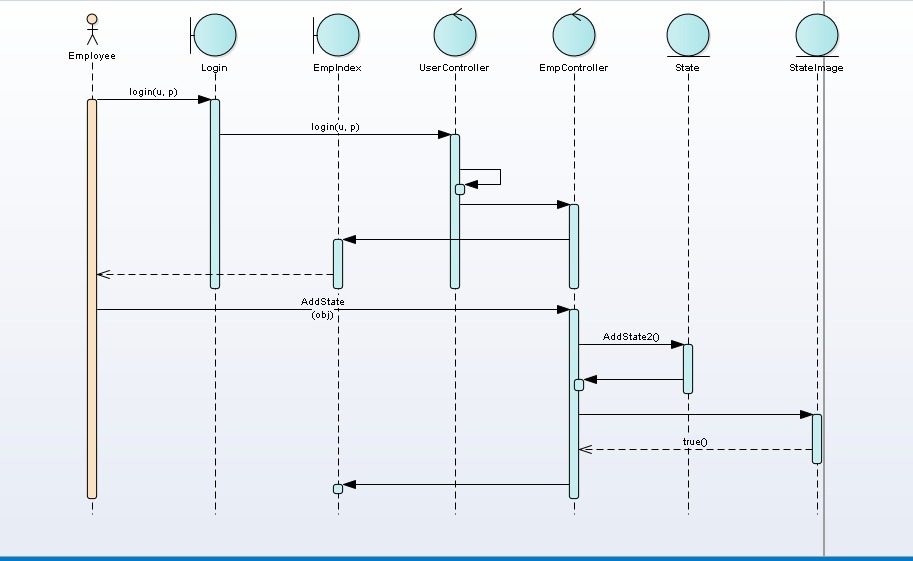
**-Emp Login:**



**Figure-13:Activity Login Employee**

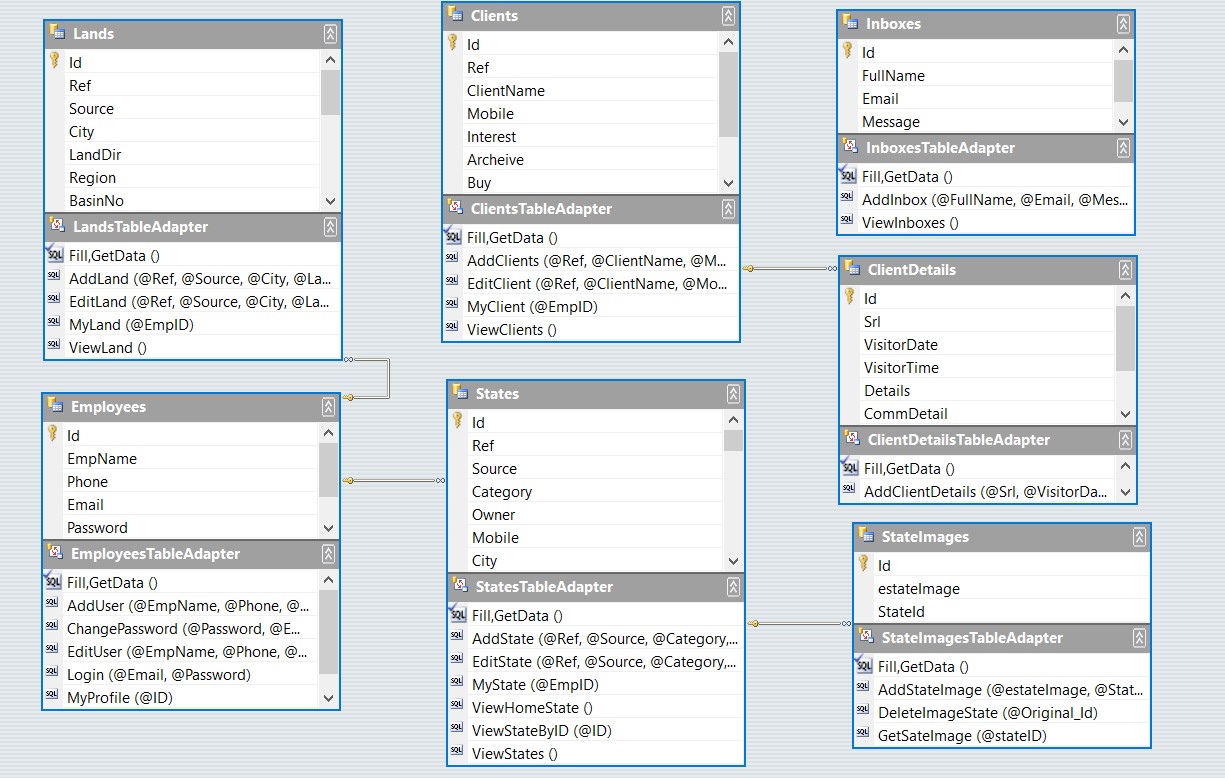
**Figure-14: Sequence Login Employee**

**-Emp Add State:**

**Figure-15:Activity Add State Employee**

**Figure-16: Sequence AddState Employee**

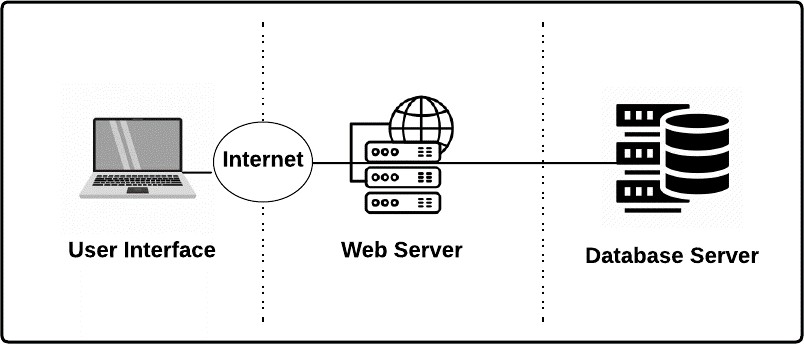
#### Domain Class Model



**Figure-17:Domain Class Model**

# **CHAPTER THREE: DESIGN**

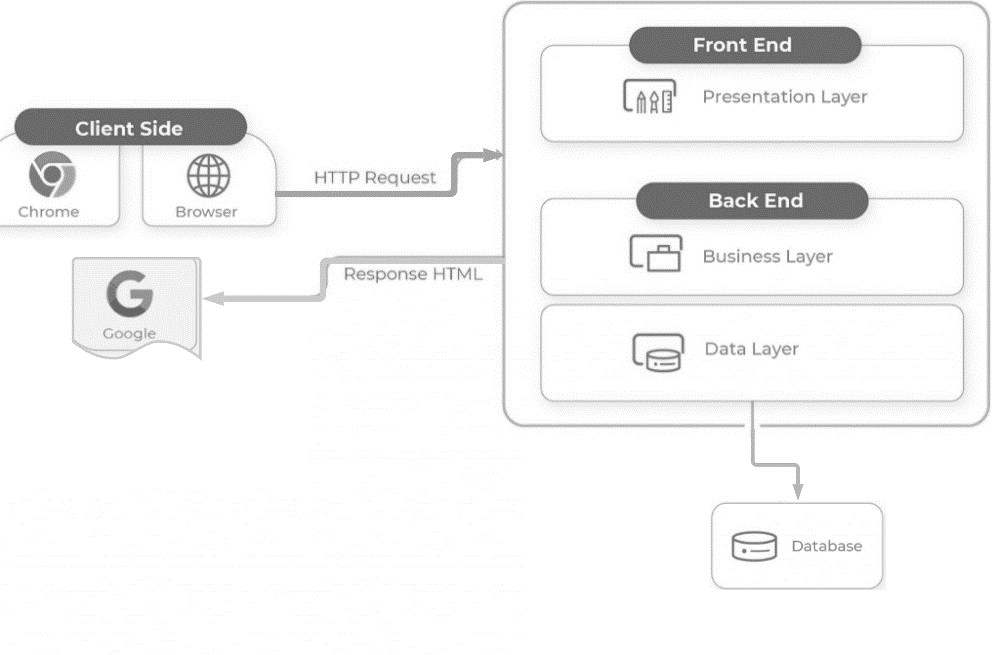
### **Architecture and Deployment environment Design**



**Figure-18:** Deployment environment Design

Design your system, you need to understand the physical and logical aspects of your current environment. From a physical standpoint, your design depends on the type and integrity of your network infrastructure.

### **Software Architecture Design**



**Figure-19:Architecture Design**

**Design the database**

#### **Design Entity-Relationship Model**

****

**Figure-20:Design Entity-Relationship Model**

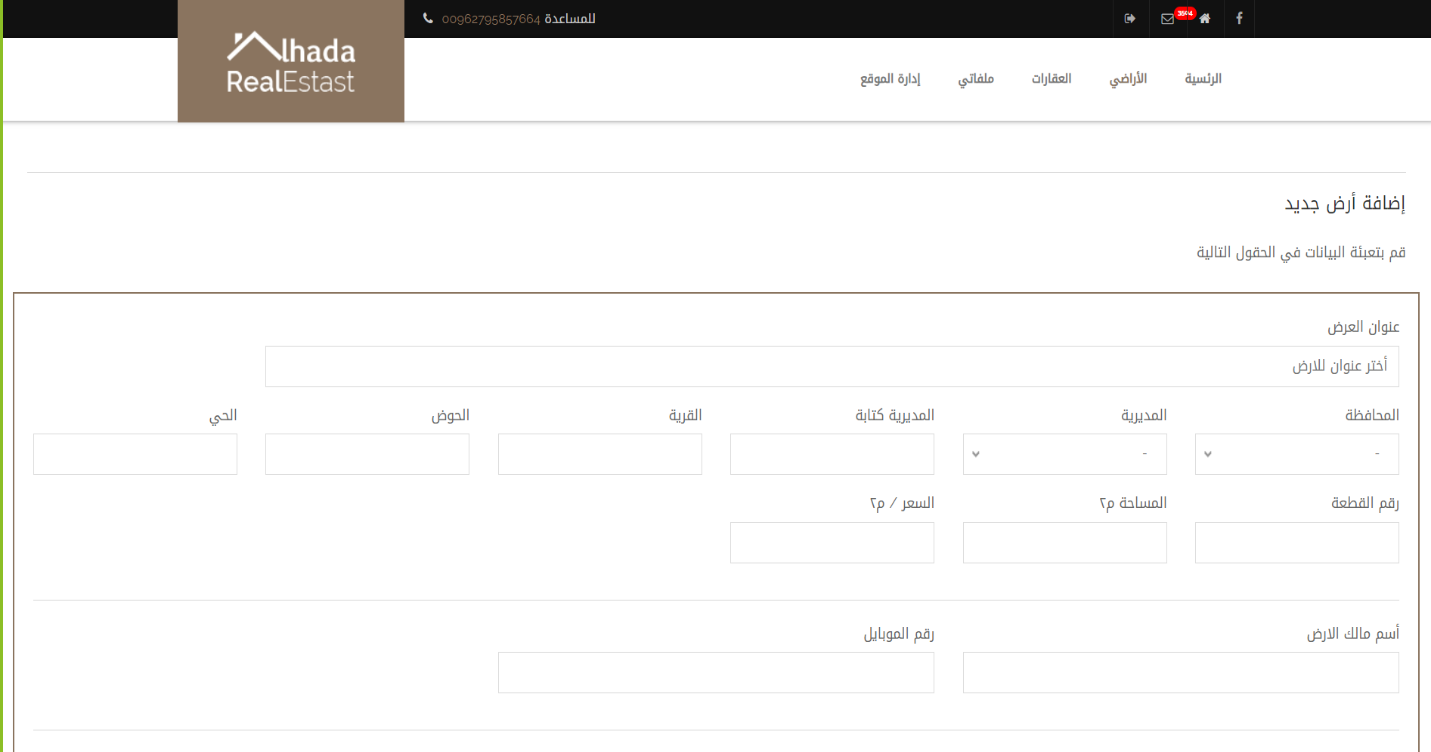
### **Design the system and user interfaces**

#### **Home Page:**

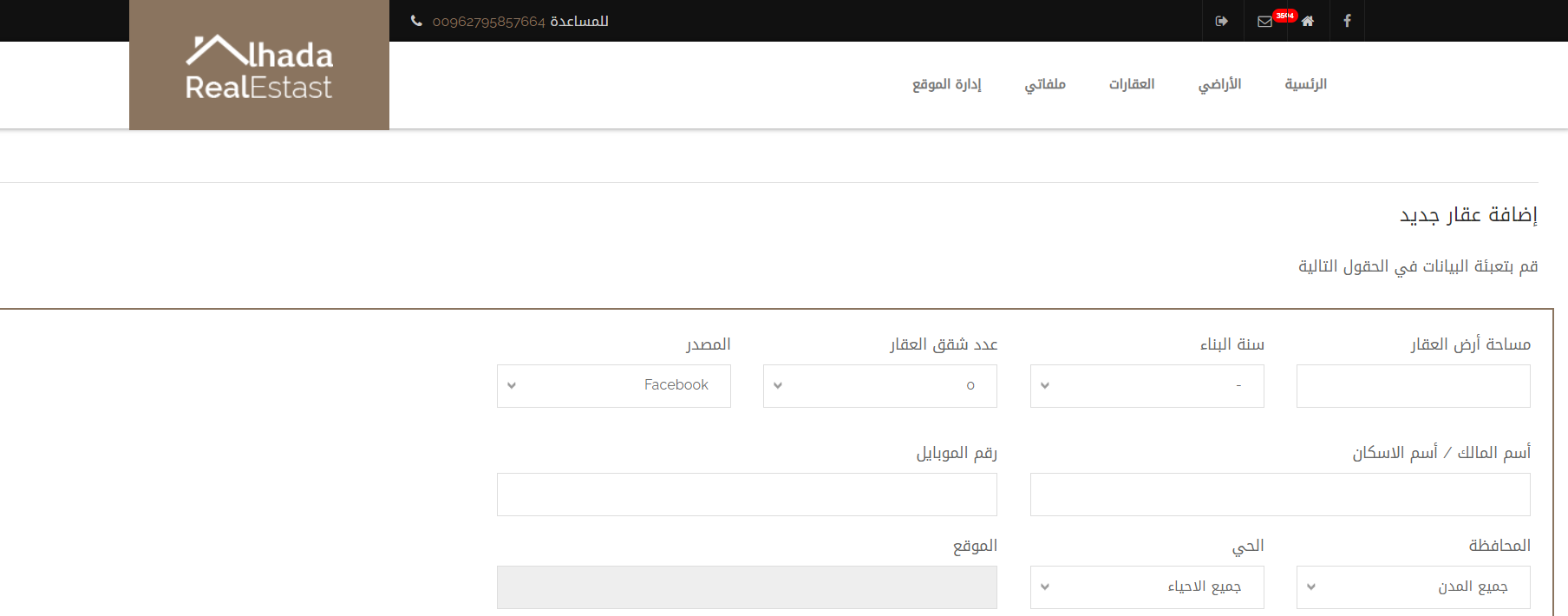
#### Figure-21:Home Page

 **login page:**

#### Figure-22:Login Page

**Add New Land Page(admin):**

**Figure-23:Add New Land Page(Admin)**

**Add New E-state page(admin):**

**Figure-24:Add New E-state Page(admin)**

#### **Add New Customer Page (Admin):**

#### 

**Figure-25:Add New customer (Admin)**

#### **Search Page (Admin):**

****

**Figure-26:Search Page(Admin)**

#### **Add New Estate Page (Employee):**

#### 

**Figure-27:Add New Estate (Employee)**

#### **Add new Customer Page (Employee):**

#### 

**Figure-28:Add New Customer (Employee)**

#### My Customer Page (Employee):

#### 

**Figure-29:My Customer (Employee)**

Users (Admin):

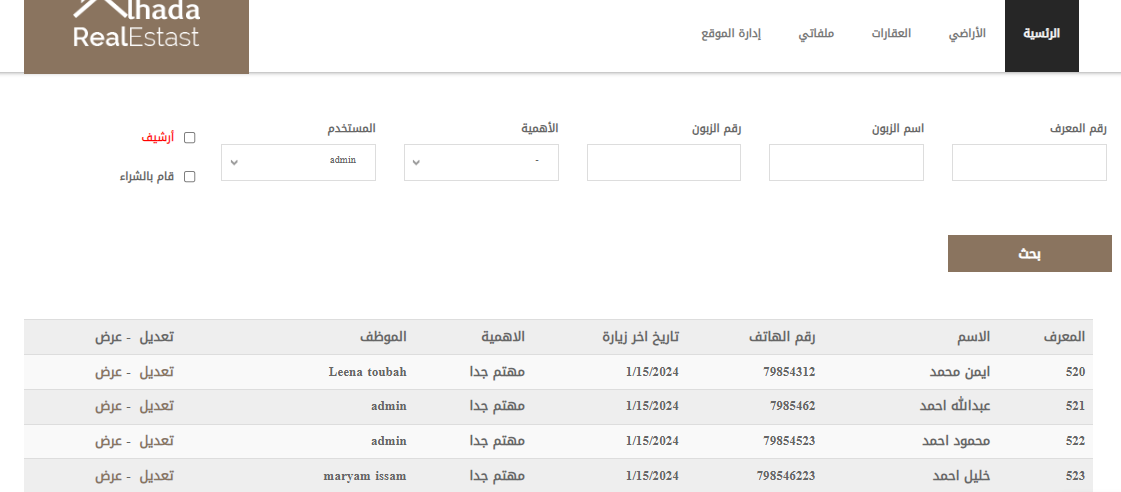
**Figure-30:Users(Admin)**

#### **Add New User:**

**Figure-31:Add New User(Admin)**

#### **My E-state(Admin)**

**Figure-31:MyE-State(Admin)**

** My Client(Admin)**

**Figure-32:My Client(Admin)**

** My Profile:**

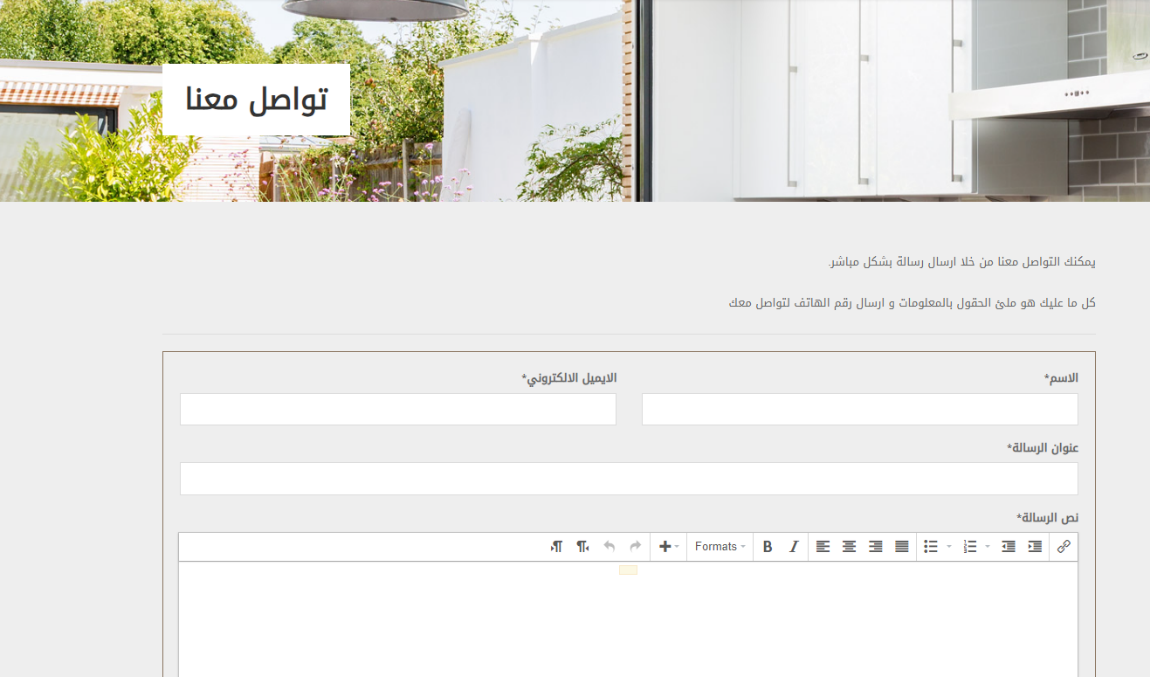
**Figure-33:My Profile**

**Inbox (Admin):**



**Figure-34:Inbox (Admin)**

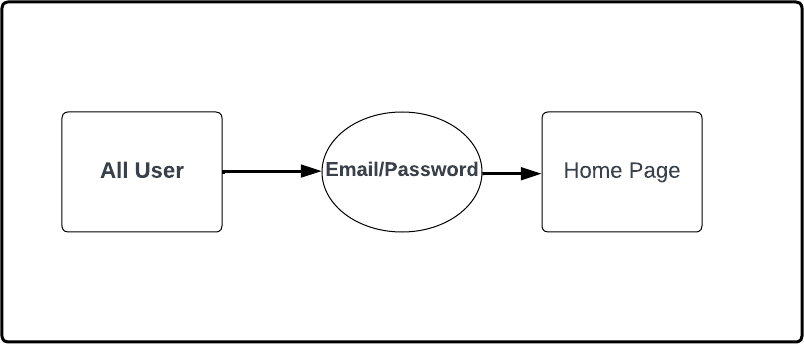
**Contact Us(user):**

**Figure-35:Contact US(User)**

### **Search Land(User):**

**Figure-36:Search Land**

### **3-8 Design the system security**



**Figure-37:Design the system Security**

# CHAPTER FOUR: IMPLEMENTATION

**4.1.1Model Code:**

1. **Client Model:**

using System.ComponentModel.DataAnnotations.Schema;

namespace alhadaProject.Models

{

public class Client

{

public int Id { get; set; }

public int Ref { get; set; } // رقم المعرف

public string? ClientName { get; set; } // اسم العميل

public int Mobile { get; set; } // رقم الموبايل

public string? Interest { get; set; } // درجة الأهمية

public bool Archeive { get; set; } = false; // ارشيف

public bool Buy { get; set; } = false; // قام بالشراء

public string? details { get; set; } // تفاصيل الزبون

public string? AddedDate { get; set; } // تاريخ الاضافة

public int EmpNo { get; set; } // رقم الموظف

}

}

1. **Client Details Model:**

using System.ComponentModel.DataAnnotations.Schema;

namespace alhadaProject.Models

{

public class ClientDetail

{

public int Id { get; set; }

public string? Srl { get; set; }

public DateTime VisitorDate { get; set; }

public DateTime VisitorTime { get; set; }

public string? Details { get; set; }

public string? CommDetail { get; set; }

public string? VisitType { get; set; }

//relationship

public int ClientNo { get; set; }

}

}

1. **Client View Model:**

using System.ComponentModel.DataAnnotations.Schema;

namespace alhadaProject.Models

{

public class ClientDetail

{

public int Id { get; set; }

public string? Srl {get; set; }

public DateTime VisitorDate { get; set; }

public DateTime VisitorTime { get; set; }

public string? Details { get; set; }

public string? CommDetail { get; set; }

public string? VisitType { get; set; }

//relationship

public int ClientNo { get; set; }

}

}

1. **Employee Model:**

namespace alhadaProject.Models

{

public class Employee

{

public int Id { get; set; }

public string? EmpName { get; set; }

public int? Phone { get; set; }

public string? Email { get; set; }

public string? Password { get; set; }

public string? UserType { get; set; }

public string? Notes { get; set; }

//Relationship

}

}

1. **Employee View Model:**

namespace alhadaproject.Models

{

public class EmployeeViewModel

{

public int Id { get; set; }

public string? EmpName { get; set; }

}

}

1. **Inbox Model:**

namespace alhadaproject.Models

{

public class Inbox

{

public int Id { get; set; }

public string? FullName { get; set; }

public string? Email { get; set; }

public string? Message { get; set; }

public string? Title { get; set; }

}

}

1. **Lands Model:**

using System.ComponentModel.DataAnnotations.Schema;

namespace alhadaProject.Models

{

public class Land

{

public int Id { get; set; }

public int Ref { get; set; } // رقم المرجع

public string? Source { get; set; } //

public string? City { get; set; } // المحاقظة

public string? LandDir { get; set; } // المديرية

public string? Region { get; set; } // القرية

public int BasinNo { get; set; } // رقم الحوض

public string? BasinName { get; set; } // اسم الحوض

public string? Owner { get; set; } // اسم المالك

public string? Mobile { get; set; } // الهاتف

public int LandNo { get; set; } // رقم القطعة

public double TotalArea { get; set; } = 0; // المساحة

public double Price { get; set; } // السعر

public string? Location { get; set; } // الموقع

public int StreetNo { get; set; } = 0; // عدد الشوارع

public bool Sold { get; set; } = false; // بيعت

public string? Notes { get; set; } // ملاحظات

//Relationship

public int EmpNo { get; set; }

}

}

1. **Messages Model:**

namespace alhadaproject.Models

{

public class Messages

{

public int Id { get; set; }

public string? FullName { get; set; }

public string? Email { get; set; }

public string? Message { get; set; }

public string? Title { get; set; }

}

}

1. **State Model:**

using System.ComponentModel.DataAnnotations.Schema;

namespace alhadaProject.Models

{

public class State

{

public int Id { get; set; }

public int Ref { get; set; } // 1 المرجع

public string? Source { get; set; } // المصدر 1

public string? Category { get; set; } // تصنيف العقار

public string? Owner { get; set; } // 1 المالك

public string? Mobile { get; set; } // 1 الموبايل

public string? City { get; set; } // 1 المحافظة

public string? Region { get; set; } // 1 الحي

public string? Location { get; set; } // 1 الموقع الجغرافي

public string? Address { get; set; } // 1العنوان بالتفصيل

public double LandArea { get; set; } // 1 مساحة الأرض

public int BuildYear { get; set; } // 1سنة البناء

public int ApartNo { get; set; } // 1عدد شقق العقار

public string? stateType { get; set; } // نوع العقار

public double StateArea { get; set; } // مساحة البناء

public string? StateStatus { get; set; } // جالة العقار

public double Price { get; set; } // سعر العقار

public string? Floor { get; set; } // الطابق

public int RoomNo { get; set; } // عدد الغرف

public int MasterNo { get; set; } // عدد غرف الماستر

public int BalconyNo { get; set; } // عدد البلاكونات

public int PathNo { get; set; } // عدد الجمامات

public int KitchNo { get; set; } // عدد المطابخ

public int LiveRoomNo { get; set; } // عدد غرف المعيشة

public int SaloonNo { get; set; } // عدد الصالونات

public bool LaundryRoom { get; set; } = false; // غرفة غسيل

public bool Guard { get; set; } = false; // حارس

public bool Furnished { get; set; } = false; // مفروشة

public bool SolarHeater { get; set; } = false; // سخان شمسي

public bool ElectricBlind { get; set; } = false; // اباجورات كهرباء

public bool MaidRoom { get; set; } = false; // غرفة خادمة

public bool PrivateGarage { get; set; } = false; // كراج خاص

public bool Haunted { get; set; } = false; // شقة مسكونة

public bool Jacozzi { get; set; } = false; // جاكوزي

public bool Parquet { get; set; } = false; // باركيه

public bool Heating { get; set; } = false; // تدفئة

public bool Garage { get; set; } = false; // كراج

public bool NeverLive { get; set; } = false; // جديدة لم تسكن

public bool ShowerBox { get; set; } = false; // شوربوكس

public bool Viewed { get; set; } = false; // اطلالة

public bool EstablishHeat { get; set; } = false; // تأسيس تدفئة

public bool Garden { get; set; } = false; // حديقة

public bool Geezer { get; set; } = false; // جيزر

public bool DoubleGlass { get; set; } = false; // دبل جلاس

public bool Installment { get; set; } = false; // أقساط

public bool Conditioning { get; set; } = false; // تكييف

public bool Taras { get; set; } = false; // ترس

public bool FirePlace { get; set; } = false; // فيربليس

public bool Ceramic { get; set; } = false; // سيرامك

public bool Elevator { get; set; } = false; // مصعد

public bool Entrance { get; set; } = false; // مدخل خاص

public bool SwimmingPool { get; set; } = false; // مفروشة

public bool Marble { get; set; } = false; // رخام

public bool StateView { get; set; } = false; // الاعلان شوهد

public bool trading { get; set; } = false; // تجاري

public bool NeedDate { get; set; } = false; // تحتاج لموعد للمشاهدة

public string? GuardName { get; set; } // اسم الحارس

public string? GradMobile { get; set; } // رقم موبايل الحارس

public double LandTarasArea { get; set; } // مساحة الارض و الترس

public double TarasArea { get; set; } // مساحة الترس

public double GardenArea { get; set; } // مساحة الجديقة

public string? Commision { get; set; } // العمولة

public bool Sold { get; set; } // بيعت

public bool HomePage { get; set; } // الصفحة الرئيسة

public string? Photo { get; set; } // صور الاعلان

public string? Notes { get; set; } // الملاحظات

public string? PromoNotes { get; set; } // اعلانات للعقار تظهر على الصفحة الرئيسية

//relationship

public int EmpNo { get; set; }

}

}

1. **State Image Model:**

using System.ComponentModel.DataAnnotations.Schema;

namespace alhadaProject.Models

{

public class StateImage

{

public int Id { get; set; }

public string estateImage { get; set; }

public int StateId { get; set; }

}

}

**11-State View Model:**

using alhadaProject.Models;

namespace alhadaproject.Models

{

public class StateViewModel

{

public State? state { get; set; }

public List<StateImage>? stateImages { get; set; }

}

}

* + 1. **Controller Code:**

**1-User Controller:**

using alhadaproject.Models;

using alhadaProject.Models;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

namespace alhadaProject.Controllers

{

public class UserController : Controller

{

private readonly ApplicationDbContext \_context;

public UserController(ApplicationDbContext context)

{

\_context = context;

}

public IActionResult Index()

{

HttpContext.Session.Clear();

HomeViewModel model = new HomeViewModel();

model.carousalState = \_context.States.Where(tbl => tbl.Sold == false && tbl.HomePage == true).AsNoTracking().ToList();

model.states = \_context.States.Where(tbl => tbl.Sold == false).ToList();

return View(model);

}

[HttpPost]

public IActionResult Index(string reference, string category, string city, string region, int roomNo, double fromArea, double toArea, double fromPrice, double toPrice, int fromYear, int toYear, string floor)

{

string sql = "select \* from states where sold =0 ";

if (reference != null)

sql += " and Ref = " + reference + "";

if (category != "0")

sql += " and Category = N'" + category + "'";

if (city != "0")

sql += " and City = N'" + city + "'";

if (region != "0")

sql += " and City = N'" + city + "'";

if (roomNo > 0)

sql += " and RoomNo = " + roomNo + "";

if (fromArea > 0 && toArea > 0)

sql += " and LandArea between " + fromArea + " and " + toArea + "";

if (fromPrice > 0 && toPrice > 0)

sql += " and Price between " + fromPrice + " and " + toPrice + "";

if (fromYear > 0 && toYear > 0)

sql += " and Price between " + fromPrice + " and " + toPrice + "";

if (floor != "0")

sql += " and Floor = N'" + floor + "'";

HomeViewModel model = new HomeViewModel();

model.carousalState = \_context.States.Where(tbl => tbl.Sold == false && tbl.HomePage == true).AsNoTracking().ToList();

model.states = \_context.States.FromSqlRaw(sql).ToList();

return View(model);

}

public IActionResult StateDetails(int id)

{

StateViewModel = new StateViewModel();

stateViewModel.state = \_context.States.Where(tbl => tbl.Id == id).FirstOrDefault();

stateViewModel.stateImages = \_context.StateImages.Where(tbl => tbl.StateId == id).ToList();

return View(stateViewModel);

}

public IActionResult ViewEstate()

{

return View();

}

public IActionResult ContactUs()

{

return View();

}

[HttpPost]

public IActionResult ContactUs(string FullName , string Email , string Title , string Message )

{

Inbox = new Inbox();

inbox.Email = Email;

inbox.Title = Title;

inbox.Message = Message;

inbox.FullName = FullName;

\_context.Add(inbox);

\_context.SaveChanges();

return View();

}

[HttpGet]

public IActionResult SearchLand()

{

var rec = \_context.Lands.ToList();

return View(rec);

}

[HttpPost]

public IActionResult SearchLand(string refs, string owner, string mobile, bool sold, string city,

string landDir, string region, string basinName, int landNo, double fromArea, double toArea, double fromPrice, double toPrice)

{

string sql = "select \* from Lands where 1=1 ";

if (refs != null)

sql = sql + " and ref =" + refs + "";

if (owner != null)

sql = sql + " and owner like N'%" + owner + "%'";

if (mobile != null)

sql = sql + " and mobile =N'" + mobile + "'";

if (sold == true)

sql = sql + " and sold =1";

if (city != "0")

sql = sql + " and city =N'" + city + "'";

if (landDir != "0")

sql = sql + " and landDir =N'" + landDir + "'";

if (region != null)

sql = sql + " and region like N'%" + region + "%'";

if (basinName != null)

sql = sql + " and basinName like N'%" + basinName + "%'";

if (landNo > 0)

sql = sql + " and landNo =" + landNo + "";

if (fromArea > 0 && toArea > 0)

sql = sql + " and TotalArea between " + fromArea + " and " + toArea + "";

if (fromPrice > 0 && toPrice > 0)

sql = sql + " and price between " + fromPrice + " and " + toPrice + "";

List<Land> recs = \_context.Lands.FromSqlRaw(sql).ToList();

return View(recs);

}

public IActionResult LandDetails(int id)

{

var rec = \_context.Lands.Find(id);

return View(rec);

}

public IActionResult LoginUser()

{

return View();

}

[HttpPost]

public IActionResult LoginUser(string email , string password)

{

var rec = \_context.Employees.Where(tbl=> tbl.Email == email && tbl.Password== password).FirstOrDefault();

if (rec != null)

{

HttpContext.Session.SetString("id", rec.Id.ToString());

HttpContext.Session.SetString("name", rec.EmpName.ToString());

if (rec.UserType =="Admin")

{

return RedirectToAction("index", "Admin");

}

else

{

return RedirectToAction("index", "Emp");

}

}

else

{

ViewBag.error = "المستخدم غير مخول لدخول النظام";

return View();

}

}

}

}

**2-Admin Controller:**

using alhadaproject.Models;

using alhadaProject.Models;

using MessagePack.Formatters;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Microsoft.EntityFrameworkCore;

namespace alhadaProject.Controllers

{

public class AdminController : Controller

{

private readonly ApplicationDbContext \_context;

public AdminController(ApplicationDbContext context)

{

\_context = context;

}

public IActionResult Index()

{

HomeViewModel model = new HomeViewModel();

model.carousalState = \_context.States.Where(tbl => tbl.Sold == false && tbl.HomePage == true).AsNoTracking().ToList();

model.states = \_context.States.Where(tbl => tbl.Sold == false).ToList() ;

return View(model);

}

[HttpPost]

public IActionResult Index(string reference, string category , string city , string region , int roomNo , double fromArea , double toArea, double fromPrice, double toPrice , int fromYear , int toYear , string floor)

{

string sql = "select \* from states where sold =0 ";

if (reference!=null)

sql += " and Ref = " + reference + "";

if (category!="0")

sql += " and Category = N'" + category + "'";

if (city != "0")

sql += " and City = N'" + city + "'";

if (region != "0")

sql += " and City = N'" + city + "'";

if (roomNo >0)

sql += " and RoomNo = " + roomNo + "";

if (fromArea > 0 && toArea > 0)

sql += " and LandArea between " + fromArea + " and " + toArea + "";

if (fromPrice > 0 && toPrice > 0)

sql += " and Price between " + fromPrice + " and " + toPrice + "";

if (fromYear > 0 && toYear > 0)

sql += " and Price between " + fromPrice + " and " + toPrice + "";

if (floor != "0")

sql += " and Floor = N'" + floor + "'";

HomeViewModel model = new HomeViewModel();

model.carousalState = \_context.States.Where(tbl => tbl.Sold == false && tbl.HomePage == true).AsNoTracking().ToList();

model.states = \_context.States.FromSqlRaw(sql).ToList();

return View(model);

}

#region "States"

public IActionResult AddEstate1()

{

return View();

}

[HttpGet]

public IActionResult AddEstate2()

{

List<string> year = new List<string>();

for (int i =1980;i<=2024; i++)

{

year.Add(i.ToString());

}

List<string> num = new List<string>();

for (int i = 0; i <= 20; i++)

{

num.Add(i.ToString());

}

ViewBag.year = year;

ViewBag.num = num;

ViewBag.source = new List<string> { "فيسبوك" , "السوق المفتوح" , "مواقع عقارية" , "غير ذلك" };

ViewBag.city = new List<string> { "عمان" };

ViewBag.region = new List<string> { "خلدا", "عرقوب خلدا", "مرج الحمام", "صويلح" };

ViewBag.category = new List<string> { "شقة", "شقة مع روف", "بيت مستقل", "شقة طابقية" ,"دوبلكس" , "شبه فيلا" , "شقة مفروشة" };

ViewBag.stateType = new List<string> { "بيع", "ايجار", "بيع أو ايجار" };

ViewBag.StateStatus = new List<string> { "فارغ", "مفروش" };

ViewBag.floor = new List<string> { "أرضي", "أول" ,"ثاني" , "ثالث", "رابع", "تسوية" };

return View();

}

[HttpPost]

public IActionResult AddEstate2(State state)

{

string mainPath = "no\_image.jpg";

int id = int.Parse(HttpContext.Session.GetString("id"));

state.EmpNo = id;

state.PromoNotes = state.Notes;

var file = HttpContext.Request.Form.Files;

if (file.Count > 0)

{

mainPath = Guid.NewGuid().ToString() + Path.GetExtension(file[0].FileName);

var fileStream = new FileStream(Path.Combine("wwwroot/", "uploads", mainPath), FileMode.Create);

file[0].CopyTo(fileStream);

state.Photo = mainPath;

}

else if (state.Photo == null && state.Id == 0)

{

state.Photo = "no\_image.jpg";

}

else

{

state.Photo = state.Photo;

}

if (ModelState.IsValid)

{

\_context.Add(state);

\_context.SaveChanges();

int stateID = \_context.States.AsNoTracking().Max(tbl => tbl.Id);

if (file.Count >0)

{

for (int i = 0; i < file.Count; i++)

{

mainPath = Guid.NewGuid().ToString() + Path.GetExtension(file[i].FileName);

var fileStream = new FileStream(Path.Combine("wwwroot/", "uploads", mainPath), FileMode.Create);

file[i].CopyTo(fileStream);

state.Photo = mainPath;

StateImage model = new StateImage();

model.estateImage = mainPath;

model.StateId = stateID;

\_context.Add(model);

\_context.SaveChanges();

}

}

return RedirectToAction("Index");

}

else

{

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

List<string> year = new List<string>();

for (int i = 1980; i <= 2024; i++)

{

year.Add(i.ToString());

}

List<string> num = new List<string>();

for (int i = 0; i <= 20; i++)

{

num.Add(i.ToString());

}

ViewBag.year = year;

ViewBag.num = num;

ViewBag.source = new List<string> { "فيسبوك", "السوق المفتوح", "مواقع عقارية", "غير ذلك" };

ViewBag.city = new List<string> { "عمان" };

ViewBag.region = new List<string> { "خلدا", "عرقوب خلدا", "مرج الحمام", "صويلح" };

ViewBag.category = new List<string> { "شقة", "شقة مع روف", "بيت مستقل", "شقة طابقية", "دوبلكس", "شبه فيلا", "شقة مفروشة" };

ViewBag.stateType = new List<string> { "بيع", "ايجار", "بيع أو ايجار" };

ViewBag.StateStatus = new List<string> { "فارغ", "مفروش" };

ViewBag.floor = new List<string> { "أرضي", "أول", "ثاني", "ثالث", "رابع", "تسوية" };

return View(state);

}

}

[HttpGet]

public IActionResult EditState(int id)

{

List<string> year = new List<string>();

for (int i = 1980; i <= 2024; i++)

{

year.Add(i.ToString());

}

List<string> num = new List<string>();

for (int i = 0; i <= 20; i++)

{

num.Add(i.ToString());

}

ViewBag.year = year;

ViewBag.num = num;

ViewBag.source = new List<string> { "فيسبوك", "السوق المفتوح", "مواقع عقارية", "غير ذلك" };

ViewBag.city = new List<string> { "عمان" };

ViewBag.region = new List<string> { "خلدا", "عرقوب خلدا", "مرج الحمام", "صويلح" };

ViewBag.category = new List<string> { "شقة", "شقة مع روف", "بيت مستقل", "شقة طابقية", "دوبلكس", "شبه فيلا", "شقة مفروشة" };

ViewBag.stateType = new List<string> { "بيع", "ايجار", "بيع أو ايجار" };

ViewBag.StateStatus = new List<string> { "فارغ", "مفروش" };

ViewBag.floor = new List<string> { "أرضي", "أول", "ثاني", "ثالث", "رابع", "تسوية" };

ViewBag.images = \_context.StateImages.Where(tbl => tbl.StateId == id).ToList();

var rec = \_context.States.Find(id);

return View(rec);

}

[HttpPost]

public IActionResult EditState( State state)

{

var rec = \_context.States.Where(tbl => tbl.Id == state.Id).AsNoTracking().FirstOrDefault();

state.Address = rec.Address;

state.PromoNotes = rec.PromoNotes;

state.Photo = rec.Photo;

if (ModelState.IsValid)

{

\_context.Update(state);

\_context.SaveChanges();

return RedirectToAction("index");

}

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

List<string> year = new List<string>();

for (int i = 1980; i <= 2024; i++)

{

year.Add(i.ToString());

}

List<string> num = new List<string>();

for (int i = 0; i <= 20; i++)

{

num.Add(i.ToString());

}

ViewBag.year = year;

ViewBag.num = num;

ViewBag.source = new List<string> { "فيسبوك", "السوق المفتوح", "مواقع عقارية", "غير ذلك" };

ViewBag.city = new List<string> { "عمان"};

ViewBag.region = new List<string> { "خلدا", "عرقوب خلدا", "مرج الحمام", "صويلح" };

ViewBag.category = new List<string> { "شقة", "شقة مع روف", "بيت مستقل", "شقة طابقية", "دوبلكس", "شبه فيلا", "شقة مفروشة" };

ViewBag.stateType = new List<string> { "بيع", "ايجار", "بيع أو ايجار" };

ViewBag.StateStatus = new List<string> { "فارغ", "مفروش" };

ViewBag.floor = new List<string> { "أرضي", "أول", "ثاني", "ثالث", "رابع", "تسوية" };

return View(state);

}

[HttpGet]

public IActionResult StateDetails(int id)

{

StateViewModel = new StateViewModel();

stateViewModel.state = \_context.States.Where(tbl => tbl.Id == id).FirstOrDefault();

stateViewModel.stateImages = \_context.StateImages.Where(tbl=>tbl.StateId == id).ToList();

return View(stateViewModel);

}

[HttpGet]

public IActionResult MyEstate()

{

int id = int.Parse(HttpContext.Session.GetString("id"));

var rec = \_context.States.Where(tbl => tbl.EmpNo == id).ToList();

return View(rec);

}

#endregion

#region "Lands"

[HttpGet]

public IActionResult AddLand()

{

ViewBag.city = new List<string> { "عمان" };

ViewBag.landDir = new List<string> { "اراضي شرق عمان", "اراضي غرب عمان", "اراضي شمال عمان", "اراضي جنوب عمان" };

return View();

}

[HttpPost]

public IActionResult AddLand(Land land)

{

string mainPath = "no\_image.jpg";

int id = int.Parse(HttpContext.Session.GetString("id"));

land.EmpNo = id;

land.StreetNo = 0;

var file = HttpContext.Request.Form.Files;

if (file.Count > 0)

{

mainPath = Guid.NewGuid().ToString() + Path.GetExtension(file[0].FileName);

var fileStream = new FileStream(Path.Combine("wwwroot/", "uploads", mainPath), FileMode.Create);

file[0].CopyTo(fileStream);

land.Source = mainPath;

}

else if (land.Source == null && land.Id == 0)

{

land.Source = "no\_image.jpg";

}

else

{

land.Source = land.Source;

}

if (ModelState.IsValid)

{

\_context.Add(land);

\_context.SaveChanges();

return RedirectToAction("Index");

}

else

{

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

return View(land);

}

}

[HttpGet]

public IActionResult SearchLand()

{

var rec = \_context.Lands.ToList();

return View(rec);

}

[HttpPost]

public IActionResult SearchLand(string refs,string owner , string mobile , bool sold ,string city ,

string landDir , string region , string basinName, int landNo , double fromArea , double toArea , double fromPrice , double toPrice)

{

string sql = "select \* from Lands where 1=1 ";

if (refs != null)

sql = sql + " and ref =" + refs +"";

if (owner != null)

sql = sql + " and owner like N'%" + owner + "%'";

if (mobile != null)

sql = sql + " and mobile =N'" + mobile + "'";

if (sold==true)

sql = sql + " and sold =1";

if (city != "0")

sql = sql + " and city =N'" + city + "'";

if (landDir != "0")

sql = sql + " and landDir =N'" + landDir + "'";

if (region != null)

sql = sql + " and region like N'%" + region + "%'";

if (basinName != null)

sql = sql + " and basinName like N'%" + basinName + "%'";

if (landNo >0)

sql = sql + " and landNo =" + landNo + "";

if (fromArea >0 && toArea >0)

sql = sql + " and TotalArea between "+ fromArea+" and "+ toArea +"";

if (fromPrice > 0 && toPrice > 0)

sql = sql + " and price between " + fromPrice + " and " + toPrice + "";

List<Land> recs = \_context.Lands.FromSqlRaw(sql).ToList();

return View(recs);

}

public IActionResult LandDetails(int id)

{

var rec = \_context.Lands.Find(id);

return View(rec);

}

#endregion

#region "Clients"

[HttpGet]

public IActionResult MyClient()

{

// int id = int.Parse(HttpContext.Session.GetString("id"));

var r = (from Client in \_context.Clients

join Employee in \_context.Employees on Client.EmpNo equals Employee.Id

select new

{

Client.Id,

Client.Ref,

Client.Buy,

Client.ClientName,

Client.Mobile,

Client.Archeive,

Client.Interest,

Client.AddedDate,

Employee.EmpName,

}).ToList();

List<ClientViewModel> list = new List<ClientViewModel>();

for (int i = 0; i < r.Count; i++)

{

list.Add(new ClientViewModel { Id = r[i].Id , Ref = r[i].Ref , AddedDate = r[i].AddedDate, Archeive = r[i].Archeive , Buy = r[i].Archeive

, ClientName = r[i].ClientName, Mobile = r[i].Mobile ,

EmpName = r[i].EmpName ,

Interest = r[i].Interest });

}

List<Employee> employees = \_context.Employees.ToList();

List<SelectListItem> selectList = new List<SelectListItem>();

foreach (var item in employees)

{

selectList.Add(new SelectListItem { Text = item.EmpName, Value = Convert.ToString( item.Id) } );

}

ViewBag.items = selectList;

return View(list);

}

[HttpPost]

public IActionResult MyClient(int refs , string cname , string mobile, string interested , string user\_id , bool is\_archived , bool buy)

{

string sql = "select \* from Clients where 1=1";

if (refs >0)

sql+=" and ref = "+ refs +"";

if (cname != null)

sql += " and ClientName like N'%"+ cname +"%'";

if (mobile != null)

sql+=" and mobile = "+ mobile +"";

if (interested !="0")

sql += " and Interest like N'%" + cname + "%'";

if (user\_id != null)

sql += " and EmpNo = " + user\_id + "";

if (is\_archived == true)

sql += " and Archeive =1 ";

if (buy == true)

sql += " and buy =1 ";

List<Client> recs = \_context.Clients.FromSqlRaw(sql).ToList();

var r = (from Client in recs

join Employee in \_context.Employees on Client.EmpNo equals Employee.Id

select new

{

Client.Id,

Client.Ref,

Client.Buy,

Client.ClientName,

Client.Mobile,

Client.Archeive,

Client.Interest,

Client.AddedDate,

Employee.EmpName,

}).ToList();

List<ClientViewModel> list = new List<ClientViewModel>();

for (int i = 0; i < r.Count; i++)

{

list.Add(new ClientViewModel

{

Id = r[i].Id,

Ref = r[i].Ref,

AddedDate = r[i].AddedDate,

Archeive = r[i].Archeive,

Buy = r[i].Archeive

,

ClientName = r[i].ClientName,

Mobile = r[i].Mobile,

EmpName = r[i].EmpName,

Interest = r[i].Interest

});

}

List<Employee> employees = \_context.Employees.ToList();

List<SelectListItem> selectList = new List<SelectListItem>();

foreach (var item in employees)

{

selectList.Add(new SelectListItem { Text = item.EmpName, Value = Convert.ToString(item.Id) });

}

ViewBag.items = selectList;

return View(list);

}

public IActionResult AddClient()

{

ViewBag.items = new List<string>() { "مهتم جدا", "مهتم", "غير مهتم" };

ViewBag.id = HttpContext.Session.GetString("id");

return View();

}

[HttpPost]

public IActionResult AddClient(Client client)

{

int id = int.Parse(HttpContext.Session.GetString("id"));

client.EmpNo = id;

client.AddedDate = DateTime.Now.ToShortDateString();

if (ModelState.IsValid)

{

\_context.Add(client);

\_context.SaveChanges();

return RedirectToAction("MyClient");

}

else

{

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

return View(client);

}

return View();

}

public ActionResult EditClient(int id)

{

ViewBag.items = new List<string>() { "مهتم جدا", "مهتم", "غير مهتم" };

var rec = \_context.Clients.Find(id);

return View(rec);

}

[HttpPost]

public ActionResult EditClient(int id , Client client)

{

var rec = \_context.Clients.Where(tbl => tbl.Id == id).AsNoTracking().FirstOrDefault();

client.EmpNo = rec.EmpNo;

client.AddedDate = rec.AddedDate;

if (ModelState.IsValid)

{

\_context.Update(client);

\_context.SaveChanges();

return RedirectToAction("MyClient");

}

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

ViewBag.items = new List<string>() { "مهتم جدا", "مهتم", "غير مهتم" };

return View(client);

}

public IActionResult ViewClient()

{

return View();

}

#endregion

#region "Users"

public IActionResult Users()

{

var recs = \_context.Employees.ToList();

return View(recs);

}

public IActionResult AddUsers()

{

ViewBag.items = new List<string> { "Admin" , "Emp" };

return View();

}

[HttpPost]

public IActionResult AddUsers(Employee employee)

{

var rec = \_context.Employees.Where(tbl=> tbl.Email == employee.Email).AsNoTracking().FirstOrDefault();

if (rec != null)

{

ViewBag.error = "البريد الالكتروني موجود مسبقا";

ViewBag.items = new List<string> { "Admin", "Emp" };

return View(employee);

}

if (ModelState.IsValid)

{

\_context.Add(employee);

\_context.SaveChanges();

return RedirectToAction("Users");

}

else

{

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

ViewBag.items = new List<string> { "Admin", "Emp" };

return View(employee);

}

}

[HttpGet]

public IActionResult EditUser(int id)

{

ViewBag.items = new List<string> { "Admin", "Emp" };

var rec = \_context.Employees.Find(id);

return View(rec);

}

[HttpPost]

public IActionResult EditUser(int id , Employee employee)

{

var rec = \_context.Employees.Where(tbl=>tbl.Id== id).AsNoTracking().FirstOrDefault();

employee.Password = rec.Password;

if (ModelState.IsValid)

{

\_context.Update(employee);

\_context.SaveChanges();

return RedirectToAction("Users");

}

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

ViewBag.items = new List<string> { "Admin", "Emp" };

return View(employee);

}

public IActionResult MyProfile()

{

int id = int.Parse(HttpContext.Session.GetString("id"));

var rec = \_context.Employees.Find(id);

return View(rec);

}

public IActionResult ChangePassword()

{

return View();

}

#endregion

#region "Inbox"

public IActionResult Inbox()

{

var recs =\_context.Inboxes.ToList();

ViewBag.msg = recs;

return View();

}

public IActionResult readInbox(int id)

{

var recs = \_context.Inboxes.Where(tbl=>tbl.Id==id).ToList().FirstOrDefault();

ViewBag.msg = recs;

return View();

}

#endregion

}

}

**3-Employee Controller:**

using alhadaproject.Models;

using alhadaProject.Models;

using MessagePack.Formatters;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Microsoft.EntityFrameworkCore;

namespace alhadaProject.Controllers

{

public class EmpController : Controller

{

private readonly ApplicationDbContext \_context;

public EmpController(ApplicationDbContext context)

{

\_context = context;

}

public IActionResult MyProfile()

{

int id = int.Parse(HttpContext.Session.GetString("id"));

var rec = \_context.Employees.Find(id);

return View(rec);

}

public IActionResult ChangePassword()

{

return View();

}

public IActionResult Index()

{

HomeViewModel model = new HomeViewModel();

model.carousalState = \_context.States.Where(tbl => tbl.Sold == false && tbl.HomePage == true).AsNoTracking().ToList();

model.states = \_context.States.Where(tbl => tbl.Sold == false).ToList();

return View(model);

}

[HttpPost]

public IActionResult Index(string reference, string category, string city, string region, int roomNo, double fromArea, double toArea, double fromPrice, double toPrice, int fromYear, int toYear, string floor)

{

string sql = "select \* from states where sold =0 ";

if (reference != null)

sql += " and Ref = " + reference + "";

if (category != "0")

sql += " and Category = N'" + category + "'";

if (city != "0")

sql += " and City = N'" + city + "'";

if (region != "0")

sql += " and City = N'" + city + "'";

if (roomNo > 0)

sql += " and RoomNo = " + roomNo + "";

if (fromArea > 0 && toArea > 0)

sql += " and LandArea between " + fromArea + " and " + toArea + "";

if (fromPrice > 0 && toPrice > 0)

sql += " and Price between " + fromPrice + " and " + toPrice + "";

if (fromYear > 0 && toYear > 0)

sql += " and Price between " + fromPrice + " and " + toPrice + "";

if (floor != "0")

sql += " and Floor = N'" + floor + "'";

HomeViewModel model = new HomeViewModel();

model.carousalState = \_context.States.Where(tbl => tbl.Sold == false && tbl.HomePage == true).AsNoTracking().ToList();

model.states = \_context.States.FromSqlRaw(sql).ToList();

return View(model);

}

#region "States"

public IActionResult AddEstate1()

{

return View();

}

[HttpGet]

public IActionResult AddEstate2()

{

List<string> year = new List<string>();

for (int i = 1980; i <= 2024; i++)

{

year.Add(i.ToString());

}

List<string> num = new List<string>();

for (int i = 0; i <= 20; i++)

{

num.Add(i.ToString());

}

ViewBag.year = year;

ViewBag.num = num;

ViewBag.source = new List<string> { "فيسبوك", "السوق المفتوح", "مواقع عقارية", "غير ذلك" };

ViewBag.city = new List<string> { "عمان" };

ViewBag.region = new List<string> { "خلدا", "عرقوب خلدا", "مرج الحمام", "صويلح" };

ViewBag.category = new List<string> { "شقة", "شقة مع روف", "بيت مستقل", "شقة طابقية", "دوبلكس", "شبه فيلا", "شقة مفروشة" };

ViewBag.stateType = new List<string> { "بيع", "ايجار", "بيع أو ايجار" };

ViewBag.StateStatus = new List<string> { "فارغ", "مفروش" };

ViewBag.floor = new List<string> { "أرضي", "أول", "ثاني", "ثالث", "رابع", "تسوية" };

return View();

}

[HttpPost]

public IActionResult AddEstate2(State state)

{

string mainPath = "no\_image.jpg";

int id = int.Parse(HttpContext.Session.GetString("id"));

state.EmpNo = id;

state.PromoNotes = state.Notes;

var file = HttpContext.Request.Form.Files;

if (file.Count > 0)

{

mainPath = Guid.NewGuid().ToString() + Path.GetExtension(file[0].FileName);

var fileStream = new FileStream(Path.Combine("wwwroot/", "uploads", mainPath), FileMode.Create);

file[0].CopyTo(fileStream);

state.Photo = mainPath;

}

else if (state.Photo == null && state.Id == 0)

{

state.Photo = "no\_image.jpg";

}

else

{

state.Photo = state.Photo;

}

if (ModelState.IsValid)

{

\_context.Add(state);

\_context.SaveChanges();

int stateID = \_context.States.AsNoTracking().Max(tbl => tbl.Id);

if (file.Count > 0)

{

for (int i = 0; i < file.Count; i++)

{

mainPath = Guid.NewGuid().ToString() + Path.GetExtension(file[i].FileName);

var fileStream = new FileStream(Path.Combine("wwwroot/", "uploads", mainPath), FileMode.Create);

file[i].CopyTo(fileStream);

state.Photo = mainPath;

StateImage model = new StateImage();

model.estateImage = mainPath;

model.StateId = stateID;

\_context.Add(model);

\_context.SaveChanges();

}

}

return RedirectToAction("Index");

}

else

{

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

List<string> year = new List<string>();

for (int i = 1980; i <= 2024; i++)

{

year.Add(i.ToString());

}

List<string> num = new List<string>();

for (int i = 0; i <= 20; i++)

{

num.Add(i.ToString());

}

ViewBag.year = year;

ViewBag.num = num;

ViewBag.source = new List<string> { "فيسبوك", "السوق المفتوح", "مواقع عقارية", "غير ذلك" };

ViewBag.city = new List<string> { "عمان" };

ViewBag.region = new List<string> { "خلدا", "عرقوب خلدا", "مرج الحمام", "صويلح" };

ViewBag.category = new List<string> { "شقة", "شقة مع روف", "بيت مستقل", "شقة طابقية", "دوبلكس", "شبه فيلا", "شقة مفروشة" };

ViewBag.stateType = new List<string> { "بيع", "ايجار", "بيع أو ايجار" };

ViewBag.StateStatus = new List<string> { "فارغ", "مفروش" };

ViewBag.floor = new List<string> { "أرضي", "أول", "ثاني", "ثالث", "رابع", "تسوية" };

return View(state);

}

}

[HttpGet]

public IActionResult EditState(int id)

{

List<string> year = new List<string>();

for (int i = 1980; i <= 2024; i++)

{

year.Add(i.ToString());

}

List<string> num = new List<string>();

for (int i = 0; i <= 20; i++)

{

num.Add(i.ToString());

}

ViewBag.year = year;

ViewBag.num = num;

ViewBag.source = new List<string> { "فيسبوك", "السوق المفتوح", "مواقع عقارية", "غير ذلك" };

ViewBag.city = new List<string> { "عمان" };

ViewBag.region = new List<string> { "خلدا", "عرقوب خلدا", "مرج الحمام", "صويلح" };

ViewBag.category = new List<string> { "شقة", "شقة مع روف", "بيت مستقل", "شقة طابقية", "دوبلكس", "شبه فيلا", "شقة مفروشة" };

ViewBag.stateType = new List<string> { "بيع", "ايجار", "بيع أو ايجار" };

ViewBag.StateStatus = new List<string> { "فارغ", "مفروش" };

ViewBag.floor = new List<string> { "أرضي", "أول", "ثاني", "ثالث", "رابع", "تسوية" };

ViewBag.images = \_context.StateImages.Where(tbl => tbl.StateId == id).ToList();

var rec = \_context.States.Find(id);

return View(rec);

}

[HttpPost]

public IActionResult EditState(State state)

{

var rec = \_context.States.Where(tbl => tbl.Id == state.Id).AsNoTracking().FirstOrDefault();

state.Address = rec.Address;

state.PromoNotes = rec.PromoNotes;

state.Photo = rec.Photo;

if (ModelState.IsValid)

{

\_context.Update(state);

\_context.SaveChanges();

return RedirectToAction("index");

}

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

List<string> year = new List<string>();

for (int i = 1980; i <= 2024; i++)

{

year.Add(i.ToString());

}

List<string> num = new List<string>();

for (int i = 0; i <= 20; i++)

{

num.Add(i.ToString());

}

ViewBag.year = year;

ViewBag.num = num;

ViewBag.source = new List<string> { "فيسبوك", "السوق المفتوح", "مواقع عقارية", "غير ذلك" };

ViewBag.city = new List<string> { "عمان", "البلقاء", "السلط", "اربد" };

ViewBag.region = new List<string> { "خلدا", "عرقوب خلدا", "مرج الحمام", "صويلح" };

ViewBag.category = new List<string> { "شقة", "شقة مع روف", "بيت مستقل", "شقة طابقية", "دوبلكس", "شبه فيلا", "شقة مفروشة" };

ViewBag.stateType = new List<string> { "بيع", "ايجار", "بيع أو ايجار" };

ViewBag.StateStatus = new List<string> { "فارغ", "مفروش" };

ViewBag.floor = new List<string> { "أرضي", "أول", "ثاني", "ثالث", "رابع", "تسوية" };

return View(state);

}

[HttpGet]

public IActionResult StateDetails(int id)

{

StateViewModel = new StateViewModel();

stateViewModel.state = \_context.States.Where(tbl => tbl.Id == id).FirstOrDefault();

stateViewModel.stateImages = \_context.StateImages.Where(tbl => tbl.StateId == id).ToList();

return View(stateViewModel);

}

[HttpGet]

public IActionResult MyEstate()

{

int id = int.Parse(HttpContext.Session.GetString("id"));

var rec = \_context.States.Where(tbl => tbl.EmpNo == id).ToList();

return View(rec);

}

#endregion

#region "Lands"

[HttpGet]

public IActionResult AddLand()

{

ViewBag.city = new List<string> { "عمان", "البلقاء", "السلط", "اربد" };

ViewBag.landDir = new List<string> { "اراضي شرق عمان", "اراضي غرب عمان", "اراضي شمال عمان", "اراضي جنوب عمان" };

return View();

}

[HttpPost]

public IActionResult AddLand(Land land)

{

string mainPath = "no\_image.jpg";

int id = int.Parse(HttpContext.Session.GetString("id"));

land.EmpNo = id;

land.StreetNo = 0;

var file = HttpContext.Request.Form.Files;

if (file.Count > 0)

{

mainPath = Guid.NewGuid().ToString() + Path.GetExtension(file[0].FileName);

var fileStream = new FileStream(Path.Combine("wwwroot/", "uploads", mainPath), FileMode.Create);

file[0].CopyTo(fileStream);

land.Source = mainPath;

}

else if (land.Source == null && land.Id == 0)

{

land.Source = "no\_image.jpg";

}

else

{

land.Source = land.Source;

}

if (ModelState.IsValid)

{

\_context.Add(land);

\_context.SaveChanges();

return RedirectToAction("Index");

}

else

{

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

return View(land);

}

}

[HttpGet]

public IActionResult SearchLand()

{

var rec = \_context.Lands.ToList();

return View(rec);

}

[HttpPost]

public IActionResult SearchLand(string refs, string owner, string mobile, bool sold, string city,

string landDir, string region, string basinName, int landNo, double fromArea, double toArea, double fromPrice, double toPrice)

{

string sql = "select \* from Lands where 1=1 ";

if (refs != null)

sql = sql + " and ref =" + refs + "";

if (owner != null)

sql = sql + " and owner like N'%" + owner + "%'";

if (mobile != null)

sql = sql + " and mobile =N'" + mobile + "'";

if (sold == true)

sql = sql + " and sold =1";

if (city != "0")

sql = sql + " and city =N'" + city + "'";

if (landDir != "0")

sql = sql + " and landDir =N'" + landDir + "'";

if (region != null)

sql = sql + " and region like N'%" + region + "%'";

if (basinName != null)

sql = sql + " and basinName like N'%" + basinName + "%'";

if (landNo > 0)

sql = sql + " and landNo =" + landNo + "";

if (fromArea > 0 && toArea > 0)

sql = sql + " and TotalArea between " + fromArea + " and " + toArea + "";

if (fromPrice > 0 && toPrice > 0)

sql = sql + " and price between " + fromPrice + " and " + toPrice + "";

List<Land> recs = \_context.Lands.FromSqlRaw(sql).ToList();

return View(recs);

}

public IActionResult LandDetails(int id)

{

var rec = \_context.Lands.Find(id);

return View(rec);

}

#endregion

#region "Clients"

[HttpGet]

public IActionResult MyClient()

{

int id = int.Parse(HttpContext.Session.GetString("id"));

var r = (from Client in \_context.Clients.Where(tbl=>tbl.EmpNo==id).ToList()

join Employee in \_context.Employees on Client.EmpNo equals Employee.Id

select new

{

Client.Id,

Client.Ref,

Client.Buy,

Client.ClientName,

Client.Mobile,

Client.Archeive,

Client.Interest,

Client.AddedDate,

Employee.EmpName,

}).ToList();

List<ClientViewModel> list = new List<ClientViewModel>();

for (int i = 0; i < r.Count; i++)

{

list.Add(new ClientViewModel

{

Id = r[i].Id,

Ref = r[i].Ref,

AddedDate = r[i].AddedDate,

Archeive = r[i].Archeive,

Buy = r[i].Archeive

,

ClientName = r[i].ClientName,

Mobile = r[i].Mobile,

EmpName = r[i].EmpName,

Interest = r[i].Interest

});

}

return View(list);

}

[HttpPost]

public IActionResult MyClient(int refs, string cname, string mobile, string interested, string user\_id, bool is\_archived, bool buy)

{

int id = int.Parse(HttpContext.Session.GetString("id"));

string sql = "select \* from Clients where EmpNo ="+ id +"";

if (refs > 0)

sql += " and ref = " + refs + "";

if (cname != null)

sql += " and ClientName like N'%" + cname + "%'";

if (mobile != null)

sql += " and mobile = " + mobile + "";

if (interested != "0")

sql += " and Interest like N'%" + cname + "%'";

if (is\_archived == true)

sql += " and Archeive =1 ";

if (buy == true)

sql += " and buy =1 ";

List<Client> recs = \_context.Clients.FromSqlRaw(sql).ToList();

var r = (from Client in recs

join Employee in \_context.Employees on Client.EmpNo equals Employee.Id

select new

{

Client.Id,

Client.Ref,

Client.Buy,

Client.ClientName,

Client.Mobile,

Client.Archeive,

Client.Interest,

Client.AddedDate,

Employee.EmpName,

}).ToList();

List<ClientViewModel> list = new List<ClientViewModel>();

for (int i = 0; i < r.Count; i++)

{

list.Add(new ClientViewModel

{

Id = r[i].Id,

Ref = r[i].Ref,

AddedDate = r[i].AddedDate,

Archeive = r[i].Archeive,

Buy = r[i].Archeive

,

ClientName = r[i].ClientName,

Mobile = r[i].Mobile,

EmpName = r[i].EmpName,

Interest = r[i].Interest

});

}

return View(list);

}

public IActionResult AddClient()

{

ViewBag.items = new List<string>() { "مهتم جدا", "مهتم", "غير مهتم" };

ViewBag.id = HttpContext.Session.GetString("id");

return View();

}

[HttpPost]

public IActionResult AddClient(Client client)

{

int id = int.Parse(HttpContext.Session.GetString("id"));

client.EmpNo = id;

client.AddedDate = DateTime.Now.ToShortDateString();

if (ModelState.IsValid)

{

\_context.Add(client);

\_context.SaveChanges();

return RedirectToAction("MyClient");

}

else

{

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

return View(client);

}

return View();

}

public ActionResult EditClient(int id)

{

ViewBag.items = new List<string>() { "مهتم جدا", "مهتم", "غير مهتم" };

var rec = \_context.Clients.Find(id);

return View(rec);

}

[HttpPost]

public ActionResult EditClient(int id, Client client)

{

var rec = \_context.Clients.Where(tbl => tbl.Id == id).AsNoTracking().FirstOrDefault();

client.EmpNo = rec.EmpNo;

client.AddedDate = rec.AddedDate;

if (ModelState.IsValid)

{

\_context.Update(client);

\_context.SaveChanges();

return RedirectToAction("MyClient");

}

var errors = ModelState.Where(x => x.Value.Errors.Count > 0).Select(x => new { x.Key, x.Value.Errors })

.ToArray();

ViewBag.error = errors.ToString();

ViewBag.items = new List<string>() { "مهتم جدا", "مهتم", "غير مهتم" };

return View(client);

}

public IActionResult ViewClient()

{

return View();

}

#endregion

}

}

**3-Home Controller:**

using alhadaproject.Models;

using Microsoft.AspNetCore.Mvc;

using System.Diagnostics;

namespace alhadaproject.Controllers

{

public class HomeController : Controller

{

private readonly ILogger<HomeController> \_logger;

public HomeController(ILogger<HomeController> logger)

{

\_logger = logger;

}

public IActionResult Index()

{

return View();

}

public IActionResult Privacy()

{

return View();

}

[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]

public IActionResult Error()

{

return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });

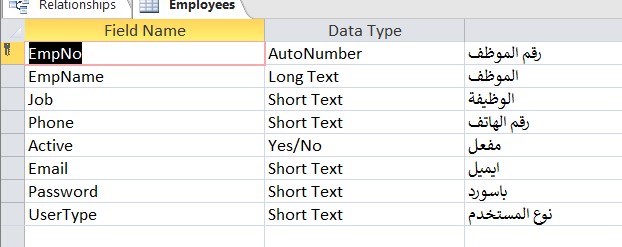
}

}

}

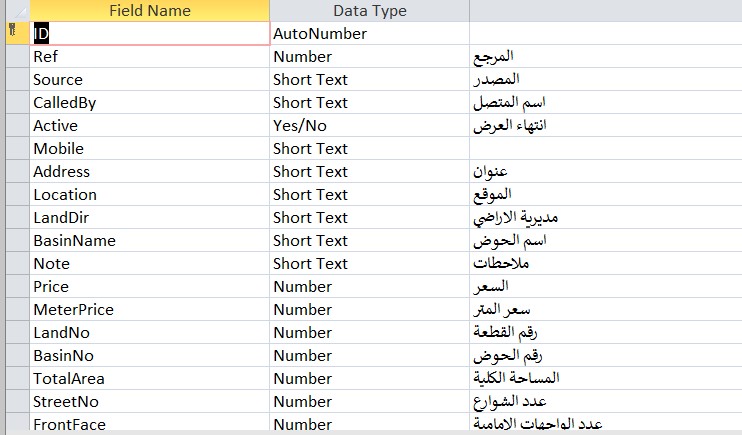
### Generate Entity-Relationship model to SQL script

## Employee:



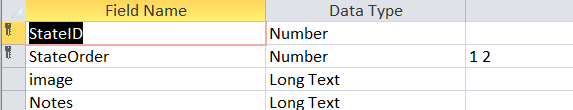
#### **Client:**

**Table-6:** Entity-Relationship model(emp)



**Table-7:** Entity-Relationship model(client)

#### **StateImage:**

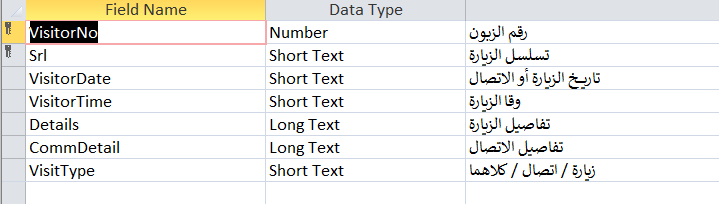


#### **Table-8: Entity-Relations(stateImage)**

#### **States:**

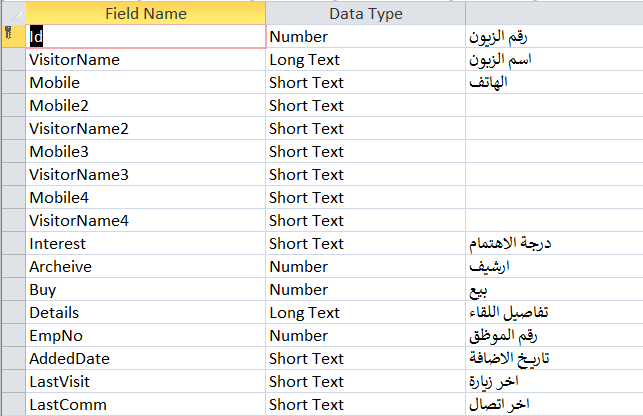
Table-9: Entity-Relationship model(state)

**ClientDetails**



**Table-10:** Entity-Relationship model(client Details)

**Client:**



**Table-11:** Entity-Relationship model(client)

# **CHAPTER FIVE: TESTING**

### **Unit Testing**

Unit testing is one of the software testing types which includes the initial testing phase where the smallest components or the modules of a software are tested individually. both testers and developers can isolate each module, identify and fix the system defects at a very early stage of the software development lifecycle (SDLC). E.g. Test every single major function such as login.

#### **Set cases test**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Step** | **Variable** | **Test Case 1** | **Test Case 2** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| **1** | **Login to System** | **Enter valid Username & Password** | ----- | **Entering to System “Displaying Home Page”** | ----- | **Pass** |
| **2** | **Login to System** | ----- | **Enter invalid Username**  **& Password** | **Displaying Message (Username or password incorrect)**  “**Login Page**” | ----- | **Pass** |

**Table-12:Set cases test**

### **Integration Testing**

Integration testing can be performed at different levels of granularity, ranging from testing the integration of individual functions or units of code to testing the integration of entire subsystems or even the entire system. The testing process may involve the use of automated testing tools or manual testing methods, depending on the complexity and size of the system being tested.

### **5.3 System Testing**

Level of testing that validates the complete and fully integrated software product. At this Level, we test the system as "one unit" to check if it encompasses all the main Functions

# CONCLUSIONS AND FUTURE WORK

### Conclusion

In conclusion Online Education Website For Schools offer an innovative and flexible approach to education that can provide numerous benefits for students of all ages and backgrounds. With the ability to learn from anywhere with an internet connection and access to a wealth of resources, eLearning provides a unique opportunity for students to customize their learning experience and achieve their educational goals on their own terms The eLearning system is designed to accommodate future upgrading and development without the need for building a new system to fit with the growing needs and demands of the system Having this system hosted online means the ability of both technicians and administrator to track and respond to demands of students at any time which add one more advantage to replacing.

### Future Work

Future Work in E-learning Website:

1. The possibility of launching the E-learning platform on mobile phones.
2. addition a chat between student and teacher.
3. addition a online exams.
4. Access to remaining student fees.
5. possibility of adding Simple Mail Transfer Protocol.
6. Support other languages.