**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM**

**Báo cáo cuối kì**

**XÂY DỰNG WEBSITE QUẢN LÍ RESORT**

**Chuyên ngành: Công nghệ thông tin**

**Giảng viên: Nguyễn Văn Vương.**

**Thành viên thực hiện:**

* **Hồ Trần Quốc Huy.**
* **Hoàng Thị Kim Phượng.**
* **Lê Huy**

**Lớp sinh hoạt: 20CNTT1**

**Đà Nẵng, năm 2022**

**MỤC LỤC**

[**MỤC LỤC** 2](#_Toc122009906)

[PHẦN 1: CƠ SỞ LÝ THUYẾT. 3](#_Toc122009907)

[**1.** **Tổng quan về dịch vụ, thương mại điện tử và ứng dụng của nó.** 3](#_Toc122009908)

[**2.** **Mục đích của website quản lí Resort.** 3](#_Toc122009909)

[PHẦN 2: THIẾT KẾ XỬ LÝ VÀ XÂY DỰNG DỮ LIỆU. 3](#_Toc122009910)

[**1.** **Giới thiệu công cụ.** 3](#_Toc122009911)

[**2.** **Xây dựng cơ sở dữ liệu.** 4](#_Toc122009912)

[**3.** **Thiết kế xử lý dữ liệu.** 11](#_Toc122009913)

[PHẦN 3: THIẾT KẾ GIAO DIỆN VÀ XỬ LÝ DỮ LIỆU. 24](#_Toc122009914)

[**1.** **Giới thiệu công cụ.** 24](#_Toc122009915)

[**2.** **Thiết kế mô hình giao diện.** 25](#_Toc122009916)

[**3.** **Thiết kế giao diện.** 34](#_Toc122009917)

[**4.** **Thiết kế xử lý dữ liệu.** 48](#_Toc122009918)

[PHẦN 4: CÀI ĐẶT VÀ KẾT QUẢ. 52](#_Toc122009919)

# PHẦN 1: CƠ SỞ LÝ THUYẾT.

## **Tổng quan về dịch vụ, thương mại điện tử và ứng dụng của nó.**

Với vị thế đang càng đi lên của Khoa học kỹ thuật, đặc biệt là Công nghệ thông tin với những ứng dụng Công nghệ thông tin vào các lĩnh vực đời sống đã góp phần to lớn cho sự nghiệp phát triển con người.

Như chúng ta đã thấy sự cạnh tranh giữa các thị trường với nhau ngày càng quyết liệt, và hầu hết những người sử dụng dịch vụ luôn cần những thứ tối ưu và tiện lợi với họ nhất.

Việt Nam nói riêng và các nước đang phát triển trên thế giới nói chung đều đang chuyển dịch cơ cấu kinh tế từ công nghiệp sang dịch vụ. Ngành dịch vụ, du lịch, lữ hành đang là ngành “hot” trong ngành kinh tế hiện tại. Bởi lẽ đó, hàng loạt dự án khách sạn cao cấp, resort 3-4-5 sao mọc lên như nấm ở bất kể vùng biển hay núi. Nhất là vùng ven với các khu du lịch sinh thái.

Thời đại công nghệ 4.0 đang phát triển từng ngày. Do đó, cả chủ doanh nghiệp và khách hàng đều đang thay đổi dần thói quen của mình trong cuộc sống. Chính vì vậy mà không chỉ riêng ngành du lịch, bất cứ ngành kinh doanh nào cũng đang ứng dụng công nghệ thông tin, phủ sóng thông tin trên khắp internet. Ngay thời điểm này, việc thiết kế website khách sạn hay thiết kế trang web resort, homestay là điều rất cần thiết.

1. **Mục đích của website quản lí Resort.**

* Xây dựng website hoàn chỉnh giúp người dùng hiểu rõ tất tần tật thông tin về resort
* Website cung cấp những tính năng cần thiết cho người khách hàng và cả admin
* Thiết kế một trang web hoàn chỉnh, bắt mắt.

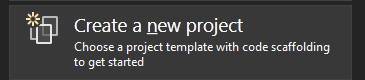
# PHẦN 2: THIẾT KẾ XỬ LÝ VÀ XÂY DỰNG DỮ LIỆU.

## **Giới thiệu công cụ.**

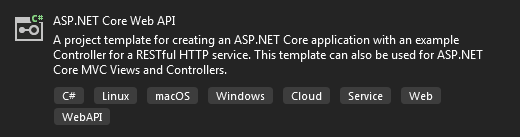
* Visual studio community 2022:
  + Đây là phiên bản miễn phí có đầy đủ tính năng, có thể mở rộng. Phiên bản dành cho sinh viên, nhà phát triển nguồn mở và cá nhân, để tạo các ứng dụng hiện đại cho Android, IOS, Windows, cũng như các ứng dụng web và dịch vụ đám mây.
  + Mục đích chính của nó là cung cấp hỗ trợ Ecosystem (hàng nghìn tiện ích mở rộng) và Language (có thể lập trình bằng C#, C++, HTML, JavaScript, Python, v.v.).
* MySQL: Là hệ quản trị cơ sở dữ liệu tự do nguồn mở phổ biến nhất thế giới và được các nhà phát triển rất ưa chuộng trong quá trình phát triển ứng dụng.
* Microsoft SQL Server: Là một hệ quản trị cơ sở dữ liệu quan hệ được phát triển bởi Microsoft. Là một máy chủ cơ sở dữ liệu, nó là một sản phẩm phần mềm có chức năng chính là lưu trữ và truy xuất dữ liệu theo yêu cầu của các ứng dụng phần mềm khác.

## **Xây dựng cơ sở dữ liệu.**

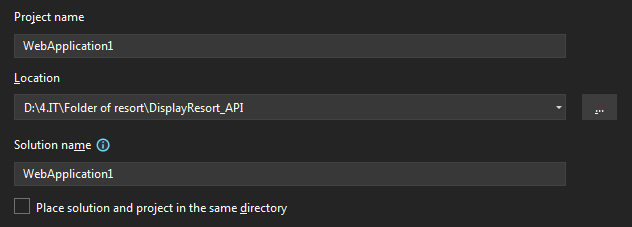
* Tạo dự án:
  + Mở visual studio: chọn Create a new project.



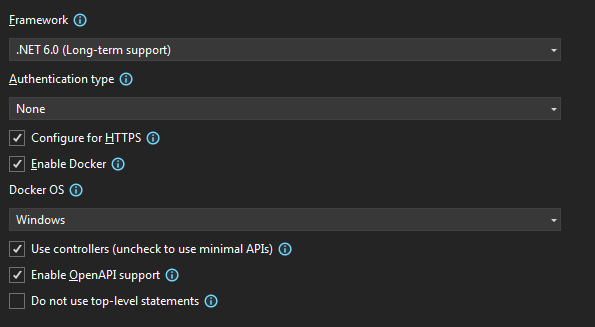
* Tìm mẫu ASP.NET Core Web API => Next.



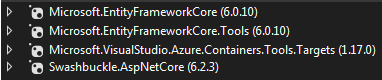
* Đổi tên dự án, tên giải pháp và đường dẫn cho dự án => Next.



* Chọn Framework và Docker => Create.



* Tải thư viện cần thiết: Denpendencies => Packages => chuột phải chọn Manage Nuget Packages… =>Tìm và tải về đúng phiên bản.
  + Các gói chung:



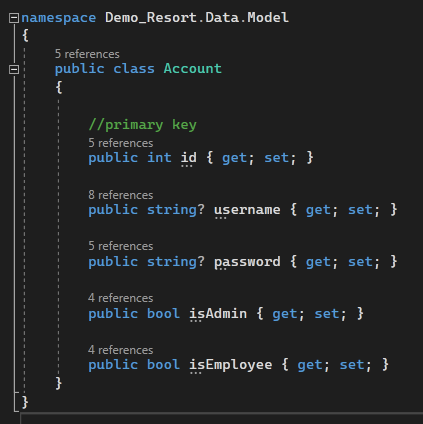
* + Sử dụng cơ sở dữ liệu MySql:



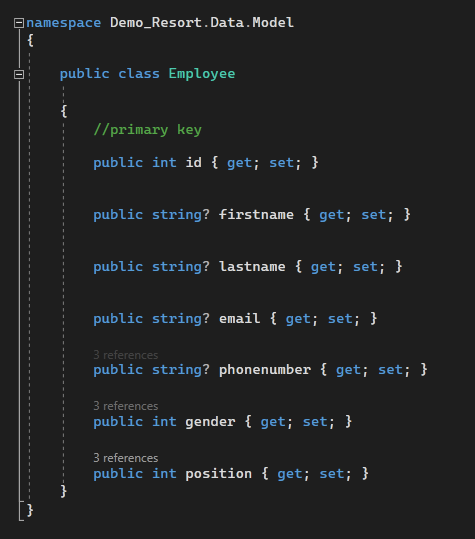
* + Sử dụng cơ sở dữ liệu SQL Server:

….

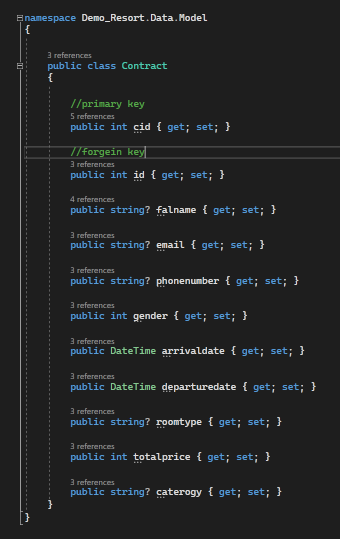
* Tạo các thư mục cần thiết: Chuột phải => Add => New Folder => Tạo 2 thư mục Data và Validate.
  + Thư mục Data: Chứa các mô hình thực thể ( tạo thư mục Model ) và xây dựng cơ sở dữ liệu.
  + Thư mục Validate: Chứa các mô hình kiểm tra dữ liệu hợp lệ hay không.
* Xây dựng cơ sở dữ liệu: Chuột phải trên thư mục Model =>Add => New Item => Class => Đổi tên => Add.
  + Trong thư mục Model tạo các class: Account.cs, Employee.cs, Contract.cs, Customer.cs để chứa thuộc tính của các bảng dữ liệu.
  + Trong thư mục Data tạo class: ResortContext.cs để xây dựng migrations => tạo database.
* Trong các class của thư mục Model:
  + Account.cs:



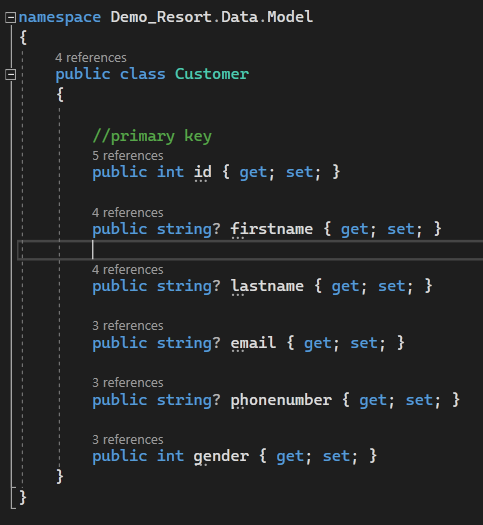
* + Employee.cs:



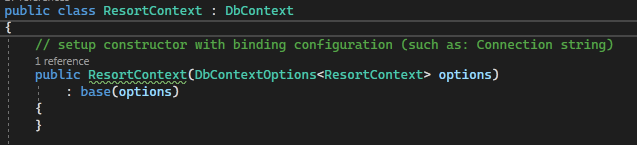
* + Contract.cs:



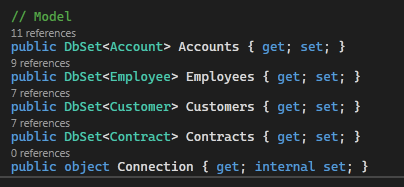
* + Customer.cs:



* Trong class ResortContext.cs:
  + Thiết lập hàm tạo với cấu hình ràng buộc (chẳng hạn như: Chuỗi kết nối):



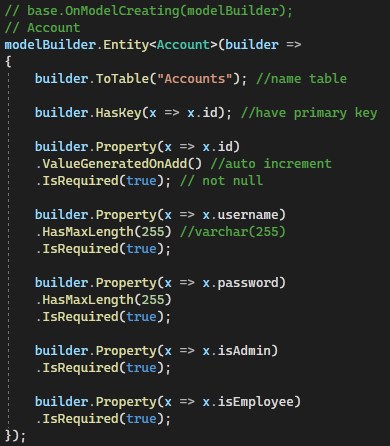
* + Thiết lập hàm các model và đối tượng kết nối:



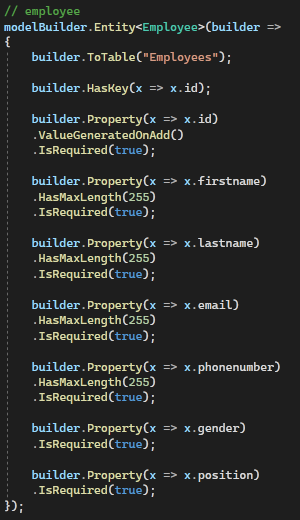
* + Thiết lập hàm cấu hình ánh xạ:



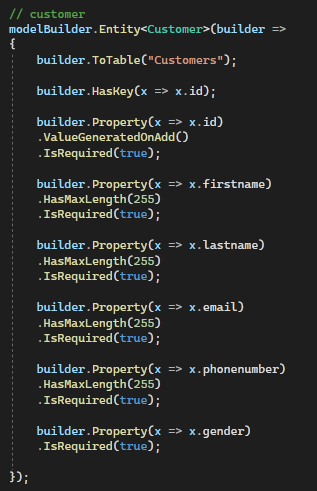
* + Trong hàm cấu hình ánh xạ tạo các bảng và thuộc tính bên trong.
    - Account



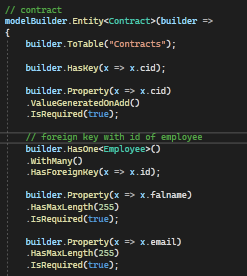
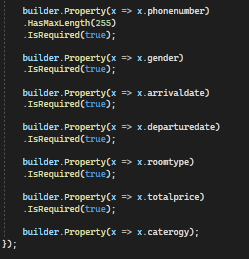
* + - Employee



* + - Customer



* + - Contract

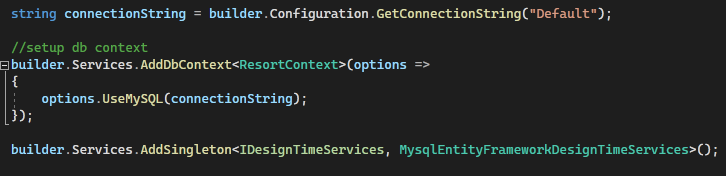
* + Thiết lập hàm tạo bối cảnh xây dựng.
    - MySQL:



* + - SQL Server:

…..

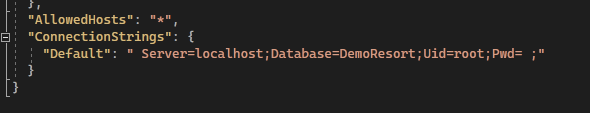
* Trong Program.cs của dự án: Thêm mô hình sau:
  + MySQL:



* + SQL Server:

……

* Trên thanh công cụ chọn Tools => NuGet Package Manager =>Package Manager Console => Gõ lệnh “ add-migration Data” => Thư mục Migrations được tạo. => Gõ lệnh “update database” để tạo database với tên, công cụ sử dụng, tài khoản và mật khẩu đăng nhập công cụ tương ứng được thiết lập trong bối cảnh xây dựng.
* Trong appsettings.json của dự án gõ thêm các lệnh như sau để thực hiện kết nối với database đã tạo ở trên trong các lần sử dụng dữ liệu sau đó. Lưu ý tên, tên server và công cụ cũng như tài khoản mật khẩu phải giống với bối cảnh được thiết lập ở hàm tạo bối cảnh xây dựng.
  + MySQL:

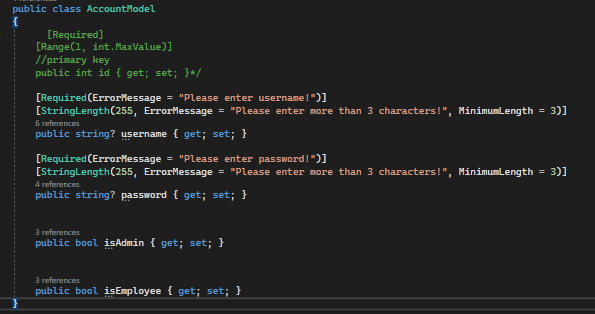


* + SQL Server:

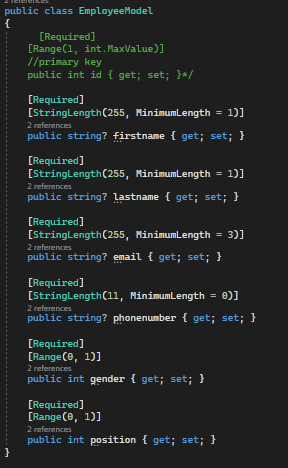
…..

1. **Thiết kế xử lý dữ liệu.**

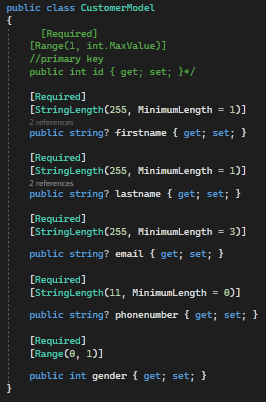
* Trong thư mục Validate tạo các class: AccountModel.cs, EmployeeModel.cs, CustomerModel.cs, ContractModel.cs.
  + AccountModel.cs:



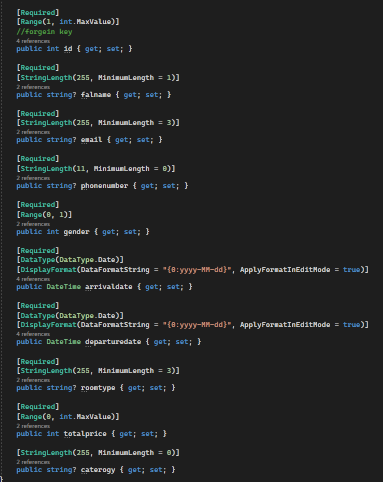
* + EmployeeModel.cs:



* + CustomerModel.cs:



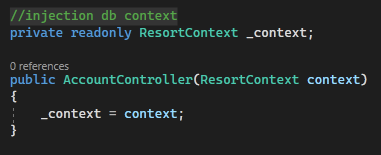
* + ContractModel.cs:



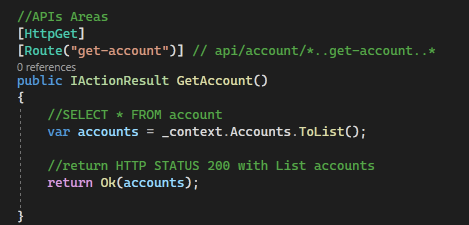
* Trong thư mục Controllers => Chuột phải chọn Add => Controller…

=> Common => API => API Controller – Empty => Đổi tên =>Add.

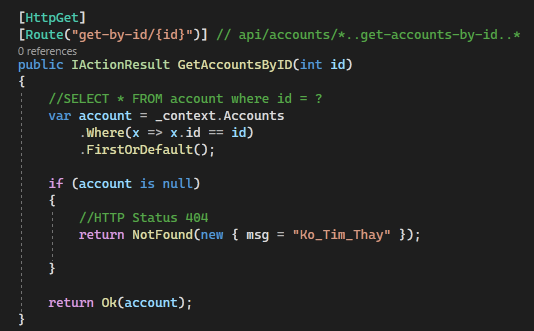
* + AccountController.
  + EmployeeController.
  + CustomerController.
  + ContractController.
* AccountController:
  + Thêm các hàm lấy bối cảnh database.



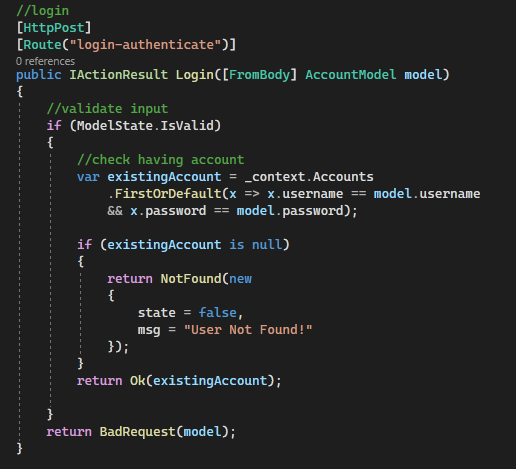
* + Thêm hàm xử lí đọc dữ liệu:



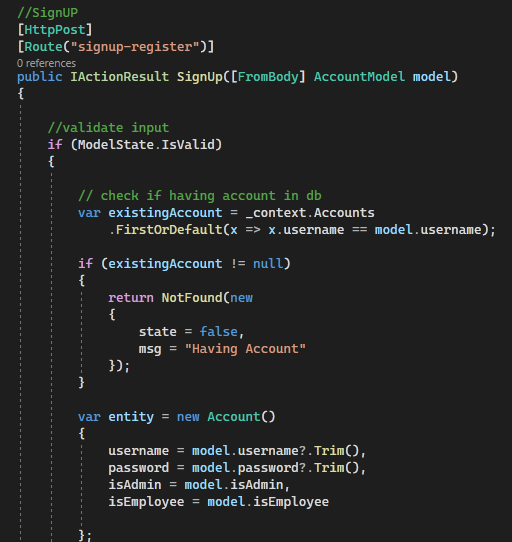
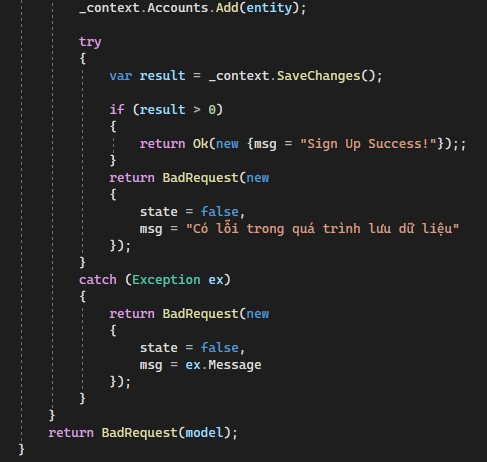
* + Thêm hàm xử lí đọc dữ liệu bằng id:



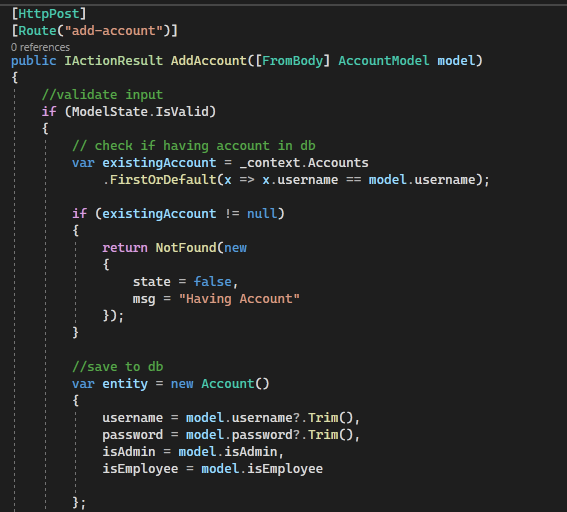
* + Thêm hàm xử lí đăng nhập:



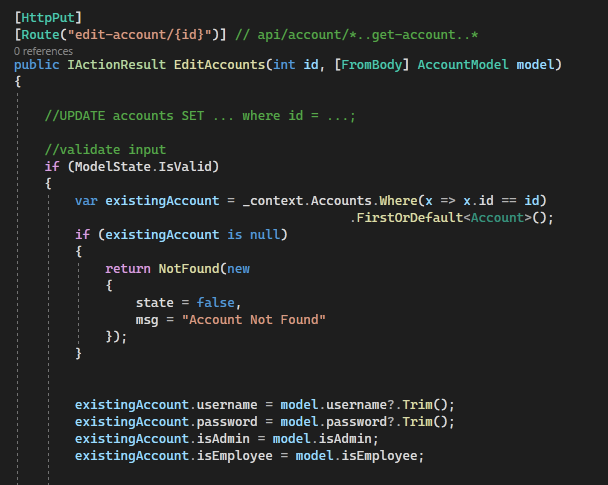
* + Thêm hàm xử lí đăng kí:

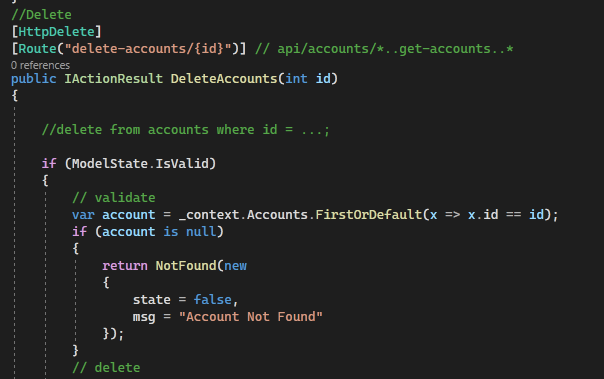
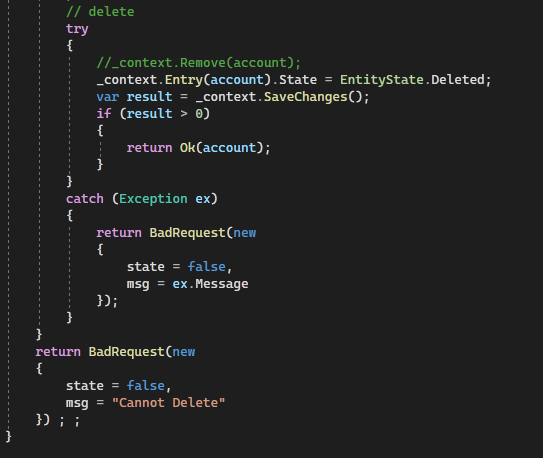
* + Thêm hàm xử lí thêm dữ liệu:

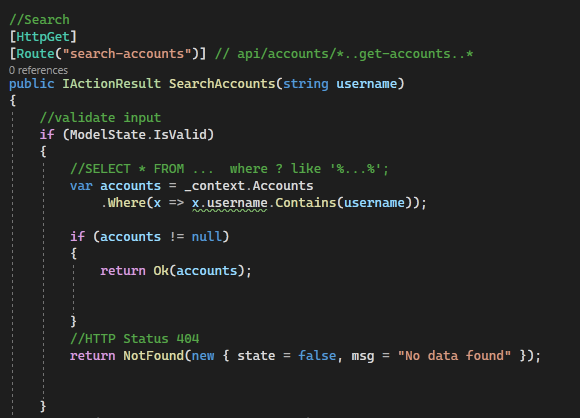
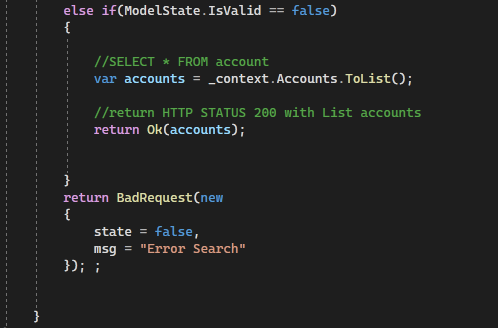
* + Thêm hàm xử lí sửa dữ liệu:

* + Thêm hàm xử lí xóa dữ liệu:

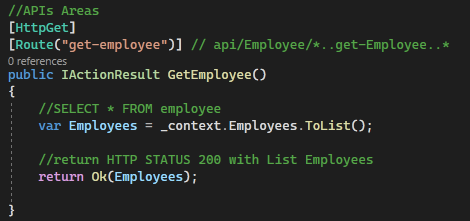
* + Thêm hàm xử lí tìm kiếm dữ liệu:

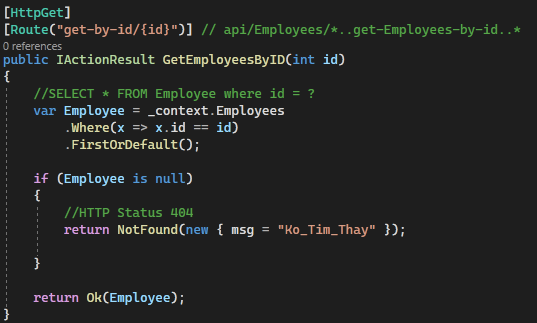
* EmployeeController:
  + Thêm các hàm lấy bối cảnh database.



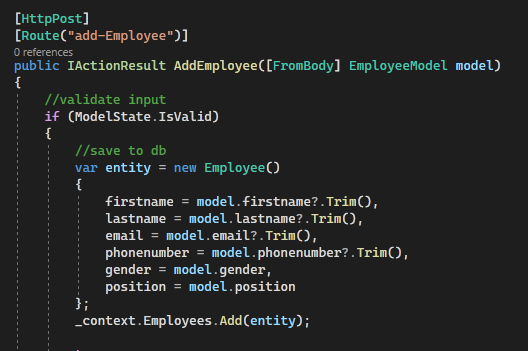
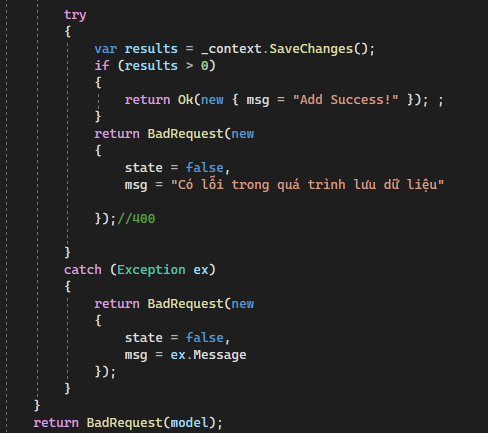
* + Thêm hàm xử lí đọc dữ liệu:



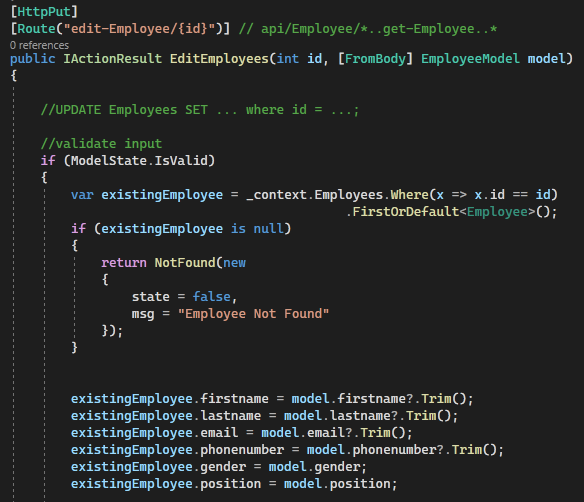
* + Thêm hàm xử lí đọc dữ liệu bằng id:



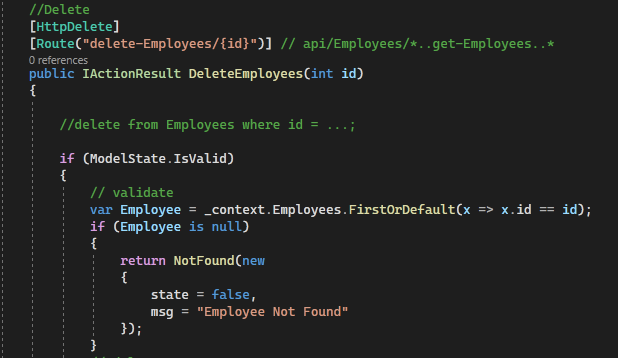
* + Thêm hàm xử lí thêm dữ liệu:

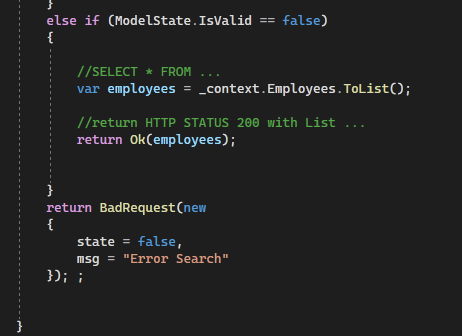
* + Thêm hàm xử lí sửa dữ liệu:

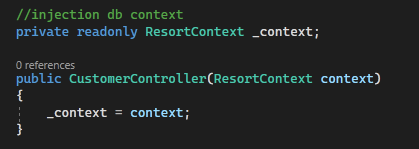
* + Thêm hàm xử lí xóa dữ liệu:

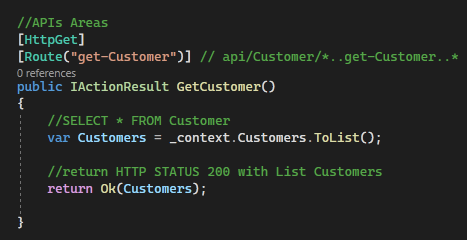
* + Thêm hàm xử lí tìm kiếm dữ liệu:

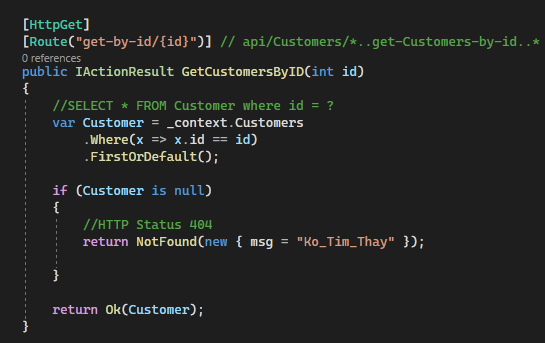
* CustomerController:
  + Thêm các hàm lấy bối cảnh database.



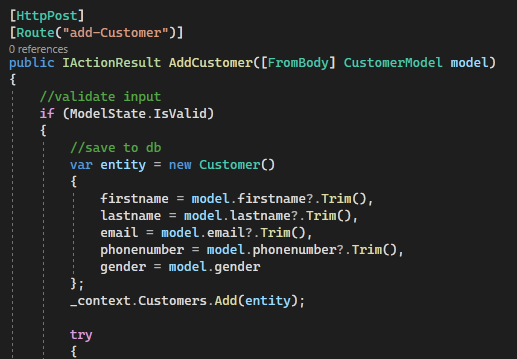
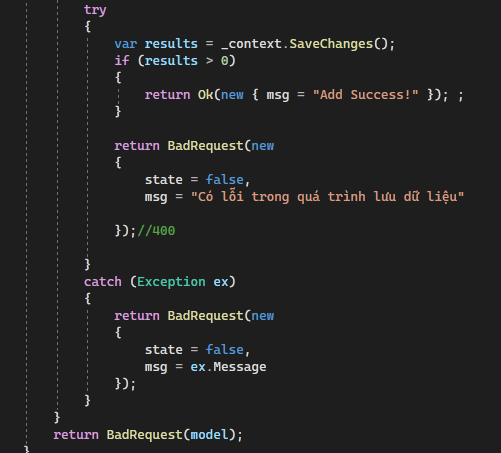
* + Thêm hàm xử lí đọc dữ liệu:



* + Thêm hàm xử lí đọc dữ liệu bằng id:



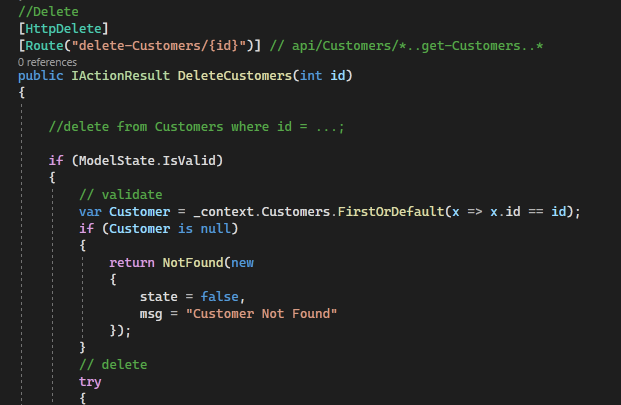
* + Thêm hàm xử lí thêm dữ liệu:

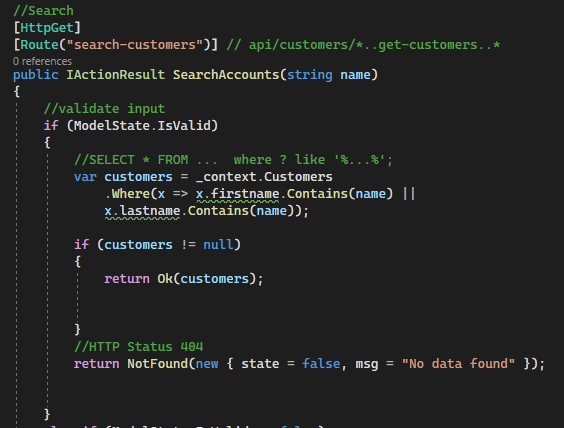
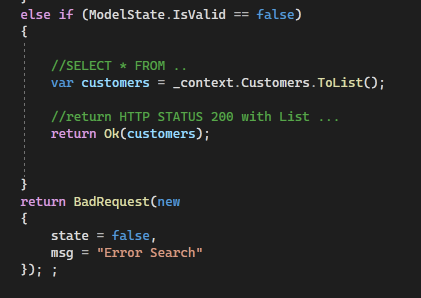
* + Thêm hàm xử lí sửa dữ liệu:

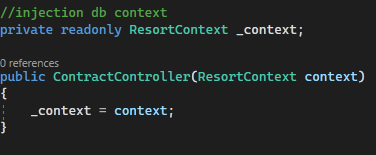
* + Thêm hàm xử lí xóa dữ liệu:

* + Thêm hàm xử lí tìm kiếm dữ liệu:

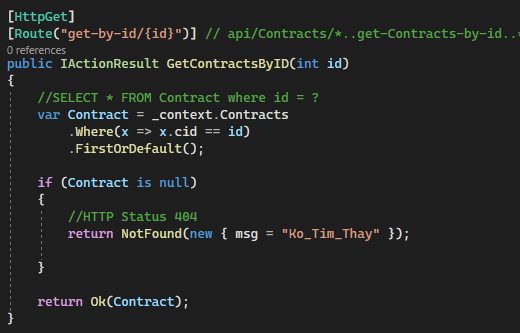
* ContractController:
  + Thêm các hàm lấy bối cảnh database.



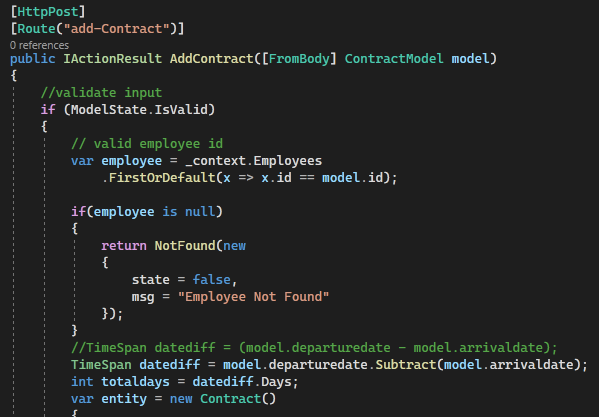
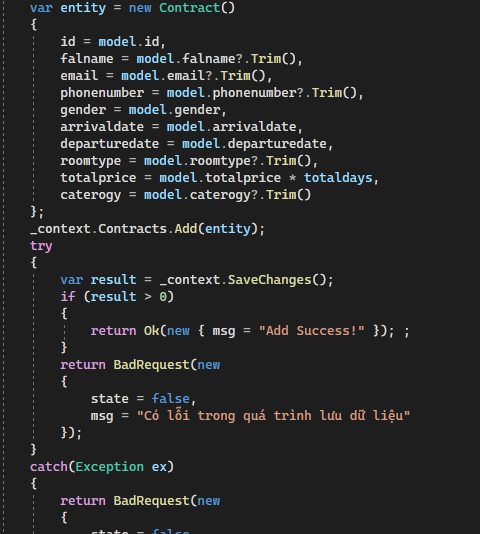
* + Thêm hàm xử lí đọc dữ liệu:



* + Thêm hàm xử lí đọc dữ liệu bằng id:



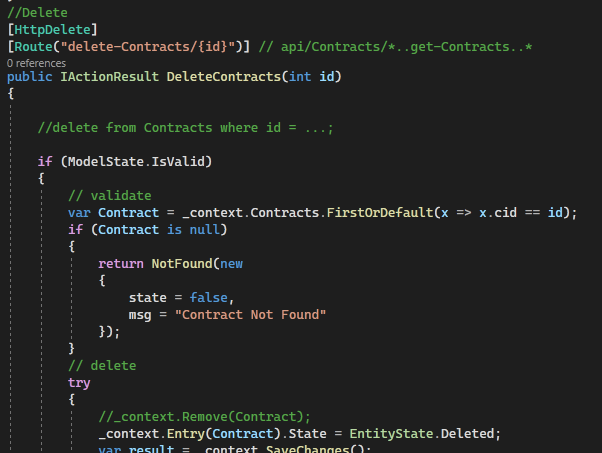
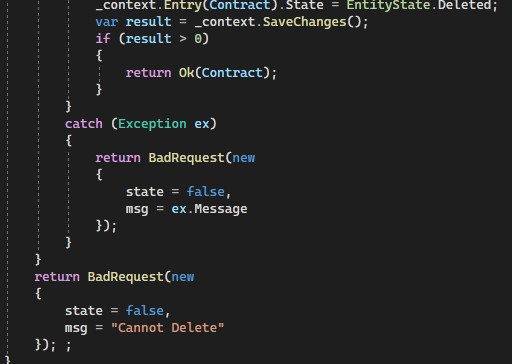
* + Thêm hàm xử lí thêm dữ liệu:

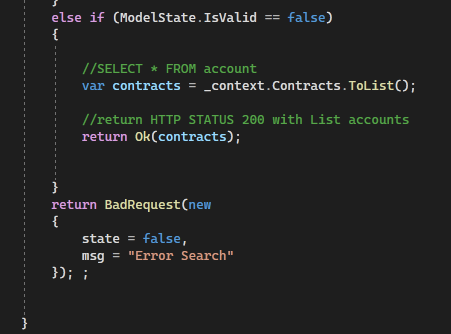
* + Thêm hàm xử lí sửa dữ liệu:

* + Thêm hàm xử lí xóa dữ liệu:

* + Thêm hàm xử lí tìm kiếm dữ liệu:

# PHẦN 3: THIẾT KẾ GIAO DIỆN VÀ XỬ LÝ DỮ LIỆU.

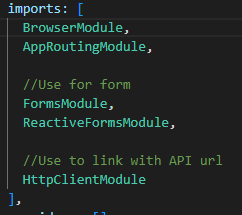
## **Giới thiệu công cụ.**

* Visual studio code: Là một trình soạn thảo mã nguồn được phát triển bởi Microsoft dành cho Windows, Linux và macOS. Nó hỗ trợ chức năng debug, đi kèm với Git, có chức năng nổi bật cú pháp, tự hoàn thành mã thông minh, snippets, và cải tiến mã nguồn.
* Angular: Là một khung ứng dụng web nguồn mở, miễn phí và dựa trên TypeScript do Nhóm Angular tại Google và một cộng đồng gồm các cá nhân và tập đoàn đứng đầu. Angular được viết lại hoàn toàn từ cùng một nhóm đã xây dựng AngularJS.
* NodeJS: Là một hệ thống phần mềm được thiết kế để viết các ứng dụng internet có khả năng mở rộng, đặc biệt là máy chủ web. Chương trình được viết bằng JavaScript, sử dụng kỹ thuật điều khiển theo sự kiện, nhập/xuất không đồng bộ để tối thiểu tổng chi phí và tối đa khả năng mở rộng.

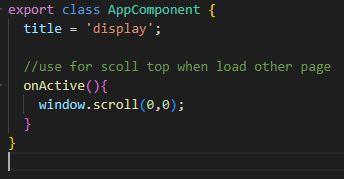
## **Thiết kế mô hình giao diện.**

* Tạo dự án: Mở Visual Studio Code => trên thanh công cụ Terminal => New Terminal => gõ lệnh ng new display để tạo dự án angular. (Lưu ý: Đã cài đặt angular và nodejs cũng như cho phép visual sử dụng powershell)
* Tạo các component, guard và service tương ứng: Gõ lệnh các lệnh sau trong terminal:
  + Để tạo giao diện dùng chung:
    - ng g c layouts/default
    - ng g c layouts/logsign-layout
  + Để tạo giao diện các trang chính:
    - ng g c pages/about
    - ng g c pages/contact
    - ng g c pages/error
    - ng g c pages/home
    - ng g c pages/login
    - ng g c pages/room
    - ng g c pages/service
    - ng g c pages/signup
  + Để tạo giao diện các trang danh sách :
    - //Trang account
    - ng g c pages/lists/account/add-acc
    - ng g c pages/lists/account/edit-acc
    - ng g c pages/lists/account/read-acc
    - //Trang employee
    - ng g c pages/lists/employee/add-emp
    - ng g c pages/lists/employee/edit-emp
    - ng g c pages/lists/employee/read-emp
    - //Trang customer
    - ng g c pages/lists/customer/add-cus
    - ng g c pages/lists/customer/edit-cus
    - ng g c pages/lists/customer/read-cus
    - //Trang contract
    - ng g c pages/lists/contract/add-con
    - ng g c pages/lists/contract/edit-con
    - ng g c pages/lists/contract/read-con
  + Để tạo các dịch vụ: Các trang liên kết với API thông qua url.
    - //Trang authenticate
    - ng g s services/auth –skip-tests
    - //Các trang ngăn chặn thực thi khi không hợp lệ
    - //Trang xác thực
    - ng g g services/guard/authen --skip-tests
    - //Trang kiểm tra quyền hạn
    - ng g g services/guard/author --skip-tests
    - ng g g services/guard/author2 --skip-tests
    - //Các trang xử lí dữ liệu liên kết với API thông qua url
    - ng g s services/lists-pages/account --skip-tests
    - ng g s services/lists-pages/contract --skip-tests
    - ng g s services/lists-pages/account --skip-tests
    - ng g s services/lists-pages/contract --skip-tests
* Trong app.module.ts:
  + Gõ lệnh sau để import thư viện HttpClientModule:





* Trong app.component.ts



* Trong app.component.html

<router-outlet (activate)= "onActive()"></router-outlet>

* Trong app-routing.module.ts

const routes: Routes = [

  { path: "", component: DefaultComponent,

    children: [

      { path: "", component: HomeComponent },

      { path: "about", component: AboutComponent },

      { path: "service", component: ServiceComponent },

      { path: "room", component: RoomComponent },

      //Path lists of pages

      //path account page

      { path: "account", component: ReadACCComponent, canActivate: [AuthenGuard, AuthorGuard] },

      { path: "addACC", component: AddACCComponent, canActivate: [AuthenGuard, AuthorGuard] },

      { path: "editACC/:id", component: EditACCComponent, canActivate: [AuthenGuard, AuthorGuard] },

      //path employee page

      { path: "employee", component: ReadEMPComponent, canActivate: [AuthenGuard] },

      { path: "addEMP", component: AddEMPComponent, canActivate: [AuthenGuard, AuthorGuard, Author2Guard] },

      { path: "editEMP/:id", component: EditEMPComponent, canActivate: [AuthenGuard, AuthorGuard, Author2Guard] },

      //path customer page

      { path: "customer", component: ReadCUSComponent, canActivate: [AuthenGuard] },

      { path: "addCUS", component: AddCUSComponent, canActivate: [AuthenGuard, Author2Guard] },

      { path: "editCUS/:id", component: EditCUSComponent, canActivate: [AuthenGuard, Author2Guard] },

      //path contract page

      { path: "contract", component: ReadCONComponent, canActivate: [AuthenGuard] },

      { path: "addCON", component: AddCONComponent, canActivate: [AuthenGuard] },

      { path: "editCON/:id", component: EditCONComponent, canActivate: [AuthenGuard, Author2Guard] },

      { path: "contact", component: ContactComponent, canActivate: [AuthenGuard] },

    ]

  },

  { path: "", component: LogsignLayoutComponent,

    children: [

      { path: "login", component: LoginComponent },

      { path: "signup", component: SignupComponent }

    ]

  },

  { path: "404", component: ErrorComponent }

];

* Trong auth.service.ts:

export class AuthService {

  //link with API url

  private baseURL:string="http://localhost:43825/api/Account/"

  constructor(

    private http : HttpClient

  ) { }

  //take link login

  login(loginObj:any){

    return this.http.post<any>(`${this.baseURL}login-authenticate`,loginObj)

  }

  //check authenticate

  isAuthenticated():boolean{

    const token = localStorage.getItem(USER\_TOKEN\_KEY)

    if(!token || token === '' || token === undefined){

      return false

    }

    return true

  }

  //check authorize

  isAdmin():boolean{

    const token = localStorage.getItem('AdminRole')

    if(token === 'true'){

      return true

    }

    return false

  }

  //check authorize

  isEmployee():boolean{

    const token = localStorage.getItem('EmployeeRole')

    if(token === 'true'){

      return true

    }

    return false

  }

  logout(){

    // localStorage.clear()

    localStorage.removeItem(USER\_TOKEN\_KEY)

    localStorage.removeItem('AdminRole')

    localStorage.removeItem('EmployeeRole')

  }

  getUsername():string{

    return localStorage.getItem(USER\_TOKEN\_KEY) ?? ''

  }

  //take link signUp

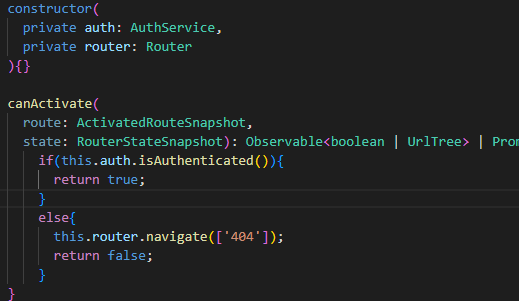
  signUp(userObj:any){

    return this.http.post<any>(`${this.baseURL}signup-register`,userObj)

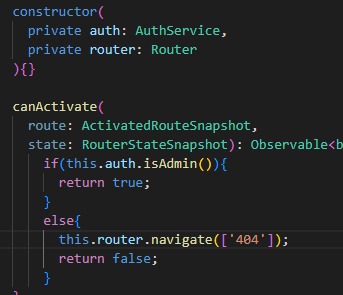
  }

}

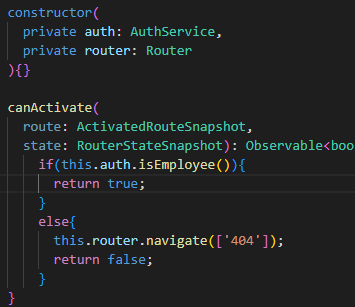
* Trong authen.guard.ts



* Trong author.guard.ts



* Trong author2.guard.ts



* Trong account.service.ts:

  private baseURL:string="http://localhost:43825/api/Account/"

  constructor(

    private http : HttpClient

  ) { }

  //for read

  getList(){

    return this.http.get(`${this.baseURL}get-account`)

  }

  //for update

  getById(id:number){

    return this.http.get(`${this.baseURL}get-by-id/` + id)

  }

  Update(id:number, updateObj:any){

    return this.http.put<any>(`${this.baseURL}edit-account/` + id, updateObj)

  }

  //for delete

  Del(id:number){

    return this.http.delete(`${this.baseURL}delete-accounts/` + id)

  }

  //for add

  Add(addObj:any){

    return this.http.post<any>(`${this.baseURL}add-account`,addObj)

  }

  //for search

  Srch(username:string){

    return this.http.get<any>(`${this.baseURL}search-accounts?username=${username}`)

  }

}

* Trong employee.service.ts:

  private baseURL:string="http://localhost:43825/api/Employee/"

  constructor(

    private http : HttpClient

  ) { }

  //for read

  getList(){

    return this.http.get(`${this.baseURL}get-employee`)

  }

  //for update

  getById(id:number){

    return this.http.get(`${this.baseURL}get-by-id/` + id)

  }

  Update(id:number, updateObj:any){

    return this.http.put<any>(`${this.baseURL}edit-employee/` + id, updateObj)

  }

  //for delete

  Del(id:number){

    return this.http.delete(`${this.baseURL}delete-employees/` + id)

  }

  //for add

  Add(addObj:any){

    return this.http.post<any>(`${this.baseURL}add-employee`,addObj)

  }

  //for search

  Srch(name:string){

    return this.http.get<any>(`${this.baseURL}search-employees?name=${name}`)

  }

}

* Trong customer.service.ts:

  private baseURL:string="http://localhost:43825/api/Customer/"

  constructor(

    private http : HttpClient

  ) { }

  //for read

  getList(){

    return this.http.get(`${this.baseURL}get-customer`)

  }

  //for update

  getById(id:number){

    return this.http.get(`${this.baseURL}get-by-id/` + id)

  }

  Update(id:number, updateObj:any){

    return this.http.put<any>(`${this.baseURL}edit-customer/` + id, updateObj)

  }

  //for delete

  Del(id:number){

    return this.http.delete(`${this.baseURL}delete-customers/` + id)

  }

  //for add

  Add(addObj:any){

    return this.http.post<any>(`${this.baseURL}add-customer`,addObj)

  }

  //for search

  Srch(name:string){

    return this.http.get<any>(`${this.baseURL}search-customers?name=${name}`)

  }

}

* Trong contract.service.ts:

  private baseURL:string="http://localhost:43825/api/Contract/"

  constructor(

    private http : HttpClient

  ) { }

  //for read

  getList(){

    return this.http.get(`${this.baseURL}get-contract`)

  }

  //for update

  getById(id:number){

    return this.http.get(`${this.baseURL}get-by-id/` + id)

  }

  Update(id:number, updateObj:any){

    return this.http.put<any>(`${this.baseURL}edit-contract/` + id, updateObj)

  }

  //for delete

  Del(id:number){

    return this.http.delete(`${this.baseURL}delete-contracts/` + id)

  }

  //for add

  Add(addObj:any){

    return this.http.post<any>(`${this.baseURL}add-contract`,addObj)

  }

  //for search

  Srch(name:string){

    return this.http.get<any>(`${this.baseURL}search-contracts?name=${name}`)

  }

}

1. **Thiết kế giao diện.**

Các trang dùng chung:

* Default
  + Default.conponent.ts:
* constructor(
* private auth: AuthService
* ) { }
* ngOnInit(): void {
* }
* onActive(){
* window.scroll(0,0);
* }
* isAuthenticated():boolean{
* return this.auth.isAuthenticated()
* }
* isAdmin():boolean{
* return this.auth.isAdmin()
* }
* logout(){
* this.auth.logout()
* }
* get username (){
* return this.auth.getUsername()
* }
  + Default.component.html
* <!--Header-->
* <!--navbar-->
* <nav class="navbar navbar-default navbar-fixed-top">
* <div class="container">
* <div class="navbar-header">
* <button type="button" class="navbar-toggle" data-toggle="collapse"
* data-target="#myNavbar">
* <span class="icon-bar"></span> <span class="icon-bar"></span> <span
* class="icon-bar"></span>
* </button>
* <a class="navbar-brand " [routerLink]="''"  >H.G.R Resort</a>
* </div>
* <div class="collapse navbar-collapse" id="myNavbar">
* <ul class="nav navbar-nav navbar-right">
* <li class="nav-item"><a class="nav-link" [routerLink]="''">Home</a></li>
* <li class="nav-item"><a class="nav-link" [routerLink]="'/about'"   >About</a></li>
* <li class="nav-item"><a class="nav-link" [routerLink]="'/service'"  >Service</a></li>
* <li class="nav-item"><a class="nav-link" [routerLink]="'/room'"  >Room</a></li>
* <li \*ngIf="isAuthenticated()" class="dropdown"><a class="dropdown-toggle"
* data-toggle="dropdown" href="#" >List<span class="caret"></span>
* </a>
* <ul class="dropdown-menu">
* <li \*ngIf="isAdmin()" class="nav-item"><a class="nav-link"
* [routerLink]="'/account'">Account</a></li>
* <li class="nav-item"><a class="nav-link"
* [routerLink]="'/employee'">Employee</a></li>
* <li class="nav-item"><a class="nav-link"
* [routerLink]="'/customer'">Customer</a></li>
* <li class="nav-item"><a class="nav-link"
* [routerLink]="'/contract'" >Contract</a></li>
* </ul></li>
* <li \*ngIf="isAuthenticated()" class="nav-item"><a class="nav-link" [routerLink]="'/contact'"
* >Contact</a></li>
* <li \*ngIf="isAuthenticated()" class="dropdown"><a class="dropdown-toggle"
* data-toggle="dropdown" href="#"><i class="fas fa-user"></i>{{username}}<span
* class="caret"></span> </a>
* <ul class="dropdown-menu">
* <li class="nav-item"><a (click)="logout()" class="nav-link"
* [routerLink]="'/login'">Logout</a></li>
* </ul></li>
* <li \*ngIf="!isAuthenticated()" class="nav-item"><a class="nav-link" [routerLink]="'/login'">Login</a></li>
* </ul>
* </div>
* </div>
* </nav>
* <!--nav-->
* <!--Header-->
* <router-outlet (activate)= "onActive()"></router-outlet>
* <!--Footer-->
* <div class="container-fluid bg-grey">
* <div class="row" style="font-family: serif; font-size: 150%;">
* <div class="col-sm-3">
* <p>About Us</p>
* <p>Terms &amp; Conditions</p>
* <p>Privacy Policy</p>
* <p>Rooms</p>
* </div>
* <div class="col-sm-3">
* <p>The Rooms &amp; Suites</p>
* <p>About Us</p>
* <p>Contact Us</p>
* <p>Services</p>
* </div>
* <div class="col-sm-3">
* <p>
* <span class="d-block"><span class="glyphicon glyphicon-map-marker"> </span>Address:</span> <span>
* 123 Go to sky, <br> Da Nang, VN
* </span>
* </p>
* <p>
* <span class="d-block"><span class="fas fa-phone"> </span>Phone:</span>
* <span> (+83) 123 1234 12</span>
* </p>
* <p>
* <span class="d-block"><span class="fas fa-envelope"> </span>Email:</span>
* <span> myEmail@gmail.com</span>
* </p>
* </div>
* <div class="col-sm-3">
* <p>Sign up for our newsletter</p>
* <form action="#" class="footer-newsletter">
* <div class="form-group">
* <input type="email" class="form-control" placeholder="Email..." readonly>
* </div>
* <i class="fas fa-paper-plane"></i>
* </form>

* </div>
* </div>
* </div>
* <footer class="panelfooter container-fluid2 bg-black text-center">
* <a href="#myPage" title="To Top"> <span class="glyphicon glyphicon-chevron-up"></span>
* </a>
* <p style="color: white;">
* Copyright by <a [routerLink]="''"  >H.G.R Resort</a>
* </p>
* </footer>
* Logsign-layout:
  + Logsign-layout.component.html
* <router-outlet></router-outlet>

Các trang dùng chính: ….component.html

* Index.html
* <!doctype html>
* <html>
* <head>
* <meta charset="utf-8">
* <title>Display</title>
* <base href="/">
* <meta name="viewport" content="width=device-width, initial-scale=1">
* <link rel="icon" type="image/x-icon" href="favicon.ico">
* <!-- bootstrap -->
* <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
* <link href="https://fonts.googleapis.com/css?family=Montserrat" rel="stylesheet" type="text/css">
* <link href="https://fonts.googleapis.com/css?family=Lato" rel="stylesheet" type="text/css">
* <link rel="stylesheet"
* href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-datetimepicker/4.17.47/css/bootstrap-datetimepicker.css">
* <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css" />
* <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.0/css/all.min.css" />
* <!--Home Style-->
* <style>
* .jumbotron {
* background-image: url('assets/image/image2.png');
* background-size: cover;
* color: #ffffff;
* padding: 100px 25px;
* margin-bottom: 0;
* }
* .bg-grey {
* background-color: #f6f6f6;
* }
* .bg-black {
* background-color: black;
* }
* .bg-litleblack {
* background-color: #282834;
* }
* .container-fluid {
* padding: 60px 50px;
* }
* .container-fluid2 {
* padding: 6px 50px;
* }
* .container-fluid3 {
* padding: 0px 100px;
* }
* .panelbook {
* border: 2px solid #ae9548;
* border-radius: 0;
* transition: box-shadow 0.5s;
* float: left;
* }
* .logo {
* font-size: 300px;
* color: green;
* }
* @media screen and (max-width: 768px) {
* .col-sm-4 {
* text-align: center;
* margin: 25px 0;
* }
* }
* .thumbnail {
* padding: 0 0 15px 0;
* border: none;
* border-radius: 0;
* }
* .bg-img {
* background-image: url('assets/image/imagenavbar.jpg');
* }
* .panel {
* border: 1px solid green;
* border-radius: 0;
* transition: box-shadow 0.5s;
* }
* .panelfooter {
* border: 4px solid green;
* border-radius: 0;
* }
* .panel:hover {
* box-shadow: 5px 0px 40px rgba(0, 0, 0, .2);
* }
* .panel-footer .btn:hover {
* border: 1px solid green;
* background-color: #fff !important;
* color: green;
* }
* .panel-heading {
* color: #fff !important;
* background-color: green !important;
* padding: 25px;
* border-bottom: 1px solid transparent;
* border-top-left-radius: 0px;
* border-top-right-radius: 0px;
* border-bottom-left-radius: 0px;
* border-bottom-right-radius: 0px;
* }
* .panel-footer {
* background-color: #fff !important;
* }
* .panel-footer h3 {
* font-size: 32px;
* }
* .panel-footer h4 {
* color: #aaa;
* font-size: 14px;
* }
* .navbar {
* margin-bottom: 0;
* background-color: black;
* z-index: 9999;
* border: 0;
* font-size: 12px !important;
* line-height: 1.42857143 !important;
* letter-spacing: 4px;
* border-radius: 0;
* }
* .navbar li a,
* .navbar .navbar-brand {
* color: #fff !important;
* }
* .navbar-nav li a:hover,
* .navbar-nav li.active a {
* color: black !important;
* background-color: #fff !important;
* }
* .navbar-default .navbar-toggle {
* border-color: transparent;
* color: #fff !important;
* }
* .dropdown-menu {
* background-color: #000;
* }
* /\*--book--\*/
* .gridbook {
* display: grid;
* grid-template-columns: repeat(5, 1fr);
* grid-gap: 20px;
* }
* input {
* outline: none;
* border: none;
* width: 100%;
* }
* .book {
* padding: 40px 0;
* background: #282834;
* color: #fff;
* }
* .gridbook .text {
* color: black;
* }
* h1 {
* font-family: serif;
* }
* .book h1 {
* font-family: serif;
* font-size: 28px;
* }
* .book h1 span {
* color: #7fc142;
* }
* .book input {
* padding: 20px;
* }
* .book input:nth-last-child(1) {
* background: #7fc142;
* color: #fff;
* }
* .visit input:nth-last-child(1) {
* background: #7fc142;
* color: #fff;
* }
* /\*--book--\*/
* </style>
* <!--About Style-->
* <style>
* .about {
* background-image: url('assets/image/about.jpg');
* background-size: cover;
* color: #ffffff;
* padding: 100px 25px;
* margin-bottom: 0;
* }
* .fa-tv {
* color: blue;
* }
* .fa-wifi {
* color: blue;
* }
* .fa-wind {
* color: blue;
* }
* .fa-location-dot {
* color: blue;
* }
* .fa-person-swimming {
* color: blue;
* }
* </style>
* <!--Service Style-->
* <style>
* .service{
* background-image: url('assets/image/service.jpg');
* background-size: cover;
* color: #ffffff;
* padding: 100px 25px;
* margin-bottom: 0;
* }
* .textfeatures{
* font-family: cursive;
* text-align: center;
* margin-top: 30px;
* }
* .textfeatures i {
* color: #ae9548;
* display: inline-block;
* font-size: 200%;
* }
* </style>
* <!--Room Style-->
* <style>
* .room {
* background-image: url('assets/image/room.jpg');
* background-size: cover;
* color: #ffffff;
* padding: 100px 25px;
* margin-bottom: 0;
* }
* .textfeatures i {
* color: #ae9548;
* display: inline-block;
* font-size: 200%;
* }
* .panelroom {
* border: 2px solid #ae9548;
* border-radius: 0;
* transition: box-shadow 0.5s;
* margin-top: 100px;
* margin-bottom: 100px;
* box-shadow: 5px 0px 40px rgba(0, 0, 0, .5);
* }
* .panelroom:hover {
* box-shadow: 5px 0px 40px rgba(0, 0, 0, 1);
* }
* .slide {
* border: 4px solid #282834;;
* border-radius: 0;
* transition: box-shadow 0.5s;
* margin-top: 20px;
* margin-bottom: 20px;
* }
* .room-text .room-title {
* overflow: hidden;
* margin-bottom: 18px;
* padding-right: 50px;
* }
* .room-text .room-title h2 {
* float: left;
* font-size: 36px;
* color: #2d220f;
* }
* .room-text .room-title .room-price span {
* position: absolute;
* left: -50px;
* bottom: 7px;
* font-size: 20px;
* font-family: serif;
* font-weight: 300;
* line-height: 25px;
* color: #2d220f;
* }
* .room-text .room-title .room-price sub {
* position: absolute;
* right: -45px;
* bottom: 7px;
* color: #2d220f;
* font-family: serif;
* font-weight: 300;
* font-size: 18px;
* line-height: 25px;
* }
* .room-text .room-title .room-price {
* float: right;
* position: relative;
* }
* .room-desc{
* font-size: 25px;
* font-family: fangsong;
* }
* .button{
* color: #ae9548;
* }
* </style>
* <!--Contact Style-->
* <style>
* .contact {
* background-image: url('assets/image/contact.jpg');
* background-size: cover;
* color: #ffffff;
* padding: 100px 25px;
* margin-bottom: 0;
* }
* .panelcontact {
* border: 3px solid #ae9548;
* border-radius: 0;
* transition: box-shadow 0.5s;
* box-shadow: 5px 0px 40px rgba(0, 0, 0, .5);
* }
* .contact-icons li {
* margin-bottom:30px;
* font-size:250%;
* }
* .contact-icons p {
* font-size: 20px;
* font-style: italic;
* }
* .contact-icons i {
* color: green;
* }
* .fa-2x {
* font-size: 1em;
* margin-right: 9px;
* }
* </style>
* </head>
* <body id="myPage" data-spy="scroll" data-target=".navbar" data-offset="60">
* <app-root></app-root>
* <!-- js -->
* <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
* <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
* <script src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.24.0/moment.min.js"></script>
* <script
* src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-datetimepicker/4.17.47/js/bootstrap-datetimepicker.min.js"></script>
* <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta1/dist/js/bootstrap.bundle.min.js"></script>
* <script>
* $(document).ready(function () {
* // Add smooth scrolling to all links in navbar + footer link
* $(".navbar a, footer a[href='#myPage']").on('click', function (event) {
* // Make sure this.hash has a value before overriding default behavior
* if (this.hash !== "") {
* // Prevent default anchor click behavior
* event.preventDefault();
* // Store hash
* var hash = this.hash;
* // Using jQuery's animate() method to add smooth page scroll
* // The optional number (900) specifies the number of milliseconds it takes to scroll to the specified area
* $('html, body').animate({
* scrollTop: $(hash).offset().top
* }, 900, function () {
* // Add hash (#) to URL when done scrolling (default click behavior)
* window.location.hash = hash;
* });
* } // End if
* });
* })
* </script>
* </body>
* </html>
* Các trang giao diện vì quá dài nên không thể thêm hết được.

1. **Thiết kế xử lý dữ liệu.**

* add-acc.component.ts
* form : FormGroup
* constructor(
* private fb : FormBuilder,
* private titleService: Title,
* private account: AccountService,
* private router: Router
* ) {
* //title
* titleService.setTitle('Add | Account Page')
* //form
* this.form = this.fb.group({
* username   : ['', [Validators.required,  Validators.minLength(3)]],
* password   : ['', [ Validators.required,  Validators.minLength(3)]],
* isAdmin    : ['',[ Validators.required]],
* isEmployee : ['',[ Validators.required]]
* })
* }
* ngOnInit(): void {
* }
* onAdd(){
* if(this.form.valid){
* console.log(this.form.value)
* //send the obj to db
* this.account.Add(this.form.value).subscribe(
* {
* next:(res =>{
* alert(res.msg);
* this.form.reset();
* }),
* error:(err=>{
* alert(err?.error.msg)
* })
* }
* )
* }
* else{
* alert('Invalid some value!')
* }
* }
* edit-acc.component.ts
* form : FormGroup
* id: number = 0
* // accounts: any;
* constructor(
* private fb : FormBuilder,
* private titleService: Title,
* private account: AccountService,
* private router: Router,
* private \_router: ActivatedRoute
* ) {
* //title
* titleService.setTitle('Edit | Account Page')
* //form
* this.form = this.fb.group({
* id         : ['', Validators.required],
* username   : ['', [Validators.required,  Validators.minLength(3)]],
* password   : ['', [ Validators.required,  Validators.minLength(3)]],
* isAdmin    : ['',[ Validators.required]],
* isEmployee : ['',[ Validators.required]]
* })
* }
* ngOnInit(): void {
* this.id = this.\_router.snapshot.params.id;
* this.account.getById(this.id).subscribe((result :any) =>{
* this.form = this.fb.group({
* id         : [result['id'], Validators.required],
* username   : [result['username'], [Validators.required,  Validators.minLength(3)]],
* password   : [result['password'], [ Validators.required,  Validators.minLength(3)]],
* isAdmin    : [result['isAdmin'],[ Validators.required]],
* isEmployee : [result['isEmployee'],[ Validators.required]]
* })
* })
* }
* onUpdate(){
* if(this.form.valid){
* console.log(this.form.value)
* //send the obj to db
* this.account.Update(this.id, this.form.value).subscribe(
* {
* next:(res =>{
* alert(res.msg);
* }),
* error:(err=>{
* alert(err?.error.msg)
* })
* }
* )
* }
* else{
* alert('Invalid some value!')
* }
* }

-read-acc.component.ts

  accounts: any;

  form:FormGroup

  constructor(

    private fb: FormBuilder,

    private account: AccountService,

    private titleService: Title

  ) {

    //title

    titleService.setTitle('List | Account Page')

    this.form = this.fb.group({

      username : ['']

    })

  }

  ngOnInit(): void {

    this.getList();

  }

  getList() {

    this.account.getList().subscribe((res) => {

      this.accounts = res;

    })

  }

  onDel(id: number) {

    var option = confirm('Are you sure to delete this account!!!');

    if (option == true) {

      this.account.Del(id).subscribe(res => {

        this.getList();

      })

    }

  }

  searchByName(){

      //send the obj to db

      this.account.Srch(this.form.value.username).subscribe(

        {

          next:(res)=>{

            console.log(res);

            this.accounts = res;

          },

          error:(err)=>{

            console.log(err);

            this.getList();

          }

        }

      )

  }

- Các trang còn lại xử lý tương tự

# PHẦN 4: CÀI ĐẶT VÀ KẾT QUẢ.

Cài đặt:

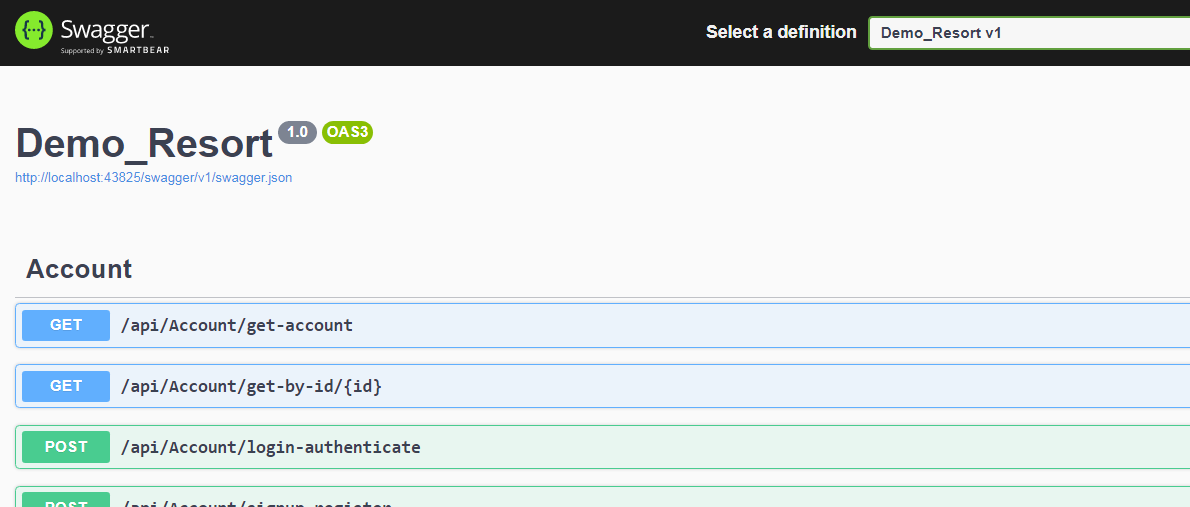
* Visual studio:
  + Trong Program.cs: thêm các dòng lệnh sau:
* builder.Services.AddCors(option =>
* {
* option.AddPolicy("MyPolicy", builder =>
* {
* builder.AllowAnyOrigin()
* .AllowAnyMethod()
* .AllowAnyHeader();
* });
* });

- app.UseCors("MyPolicy");

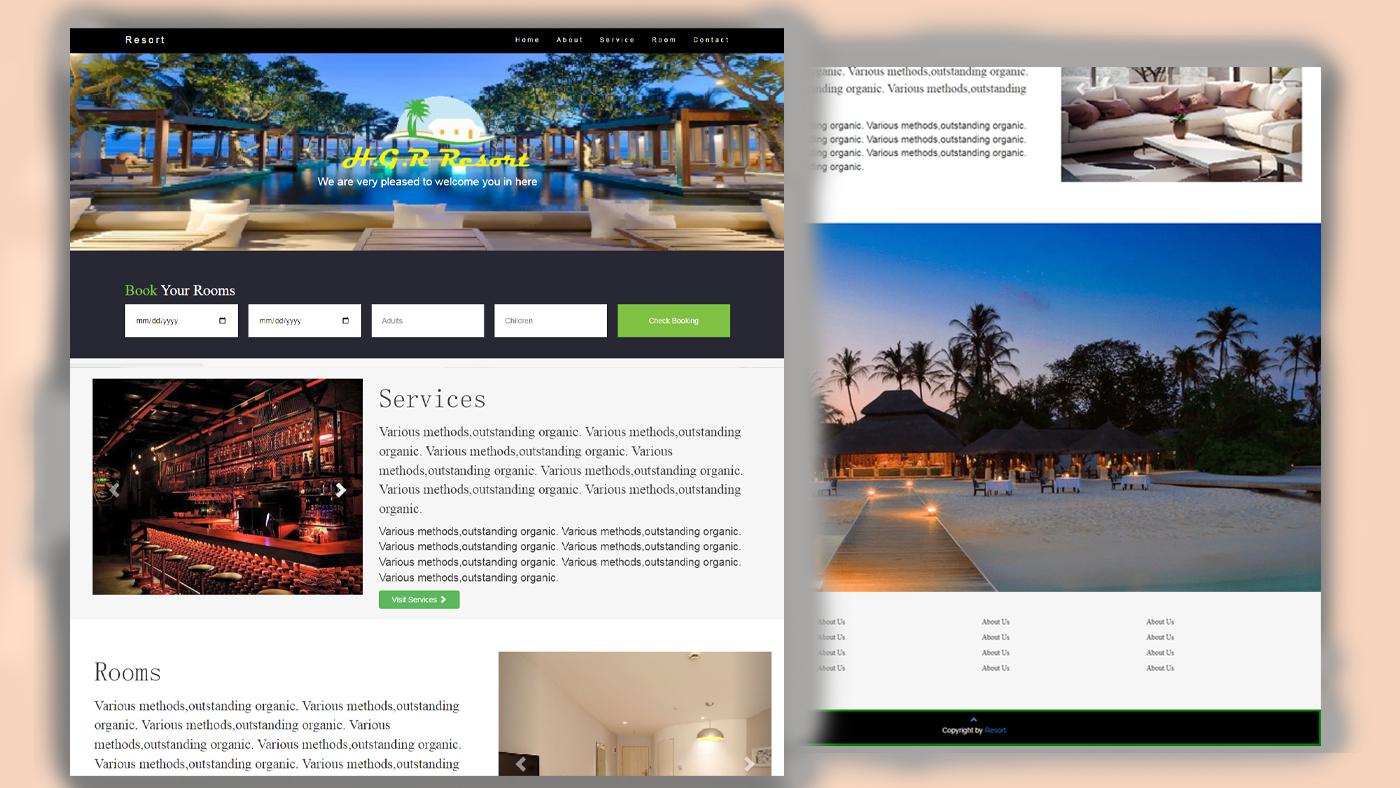
* app.UseAuthentication();
  + Chạy  để sử dụng url.
* Trong visual studio code: mở terminal gõ lệnh ng s

Kết quả:

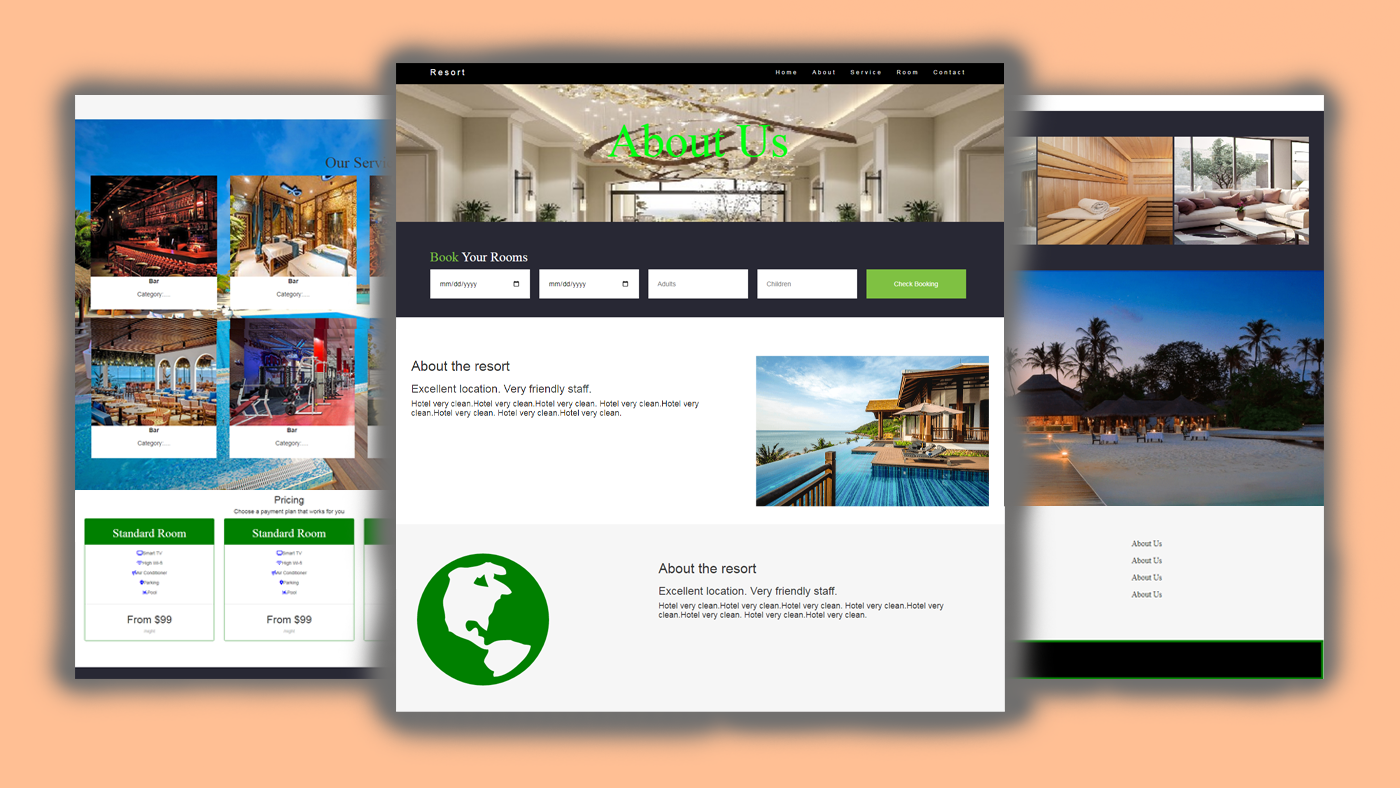
* Trang API:



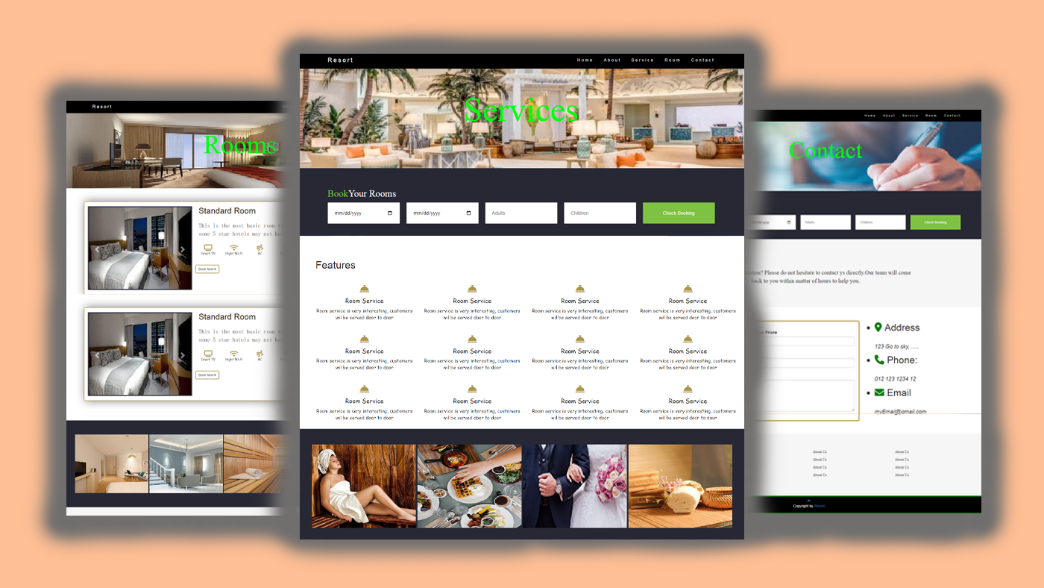
* Trang Home:



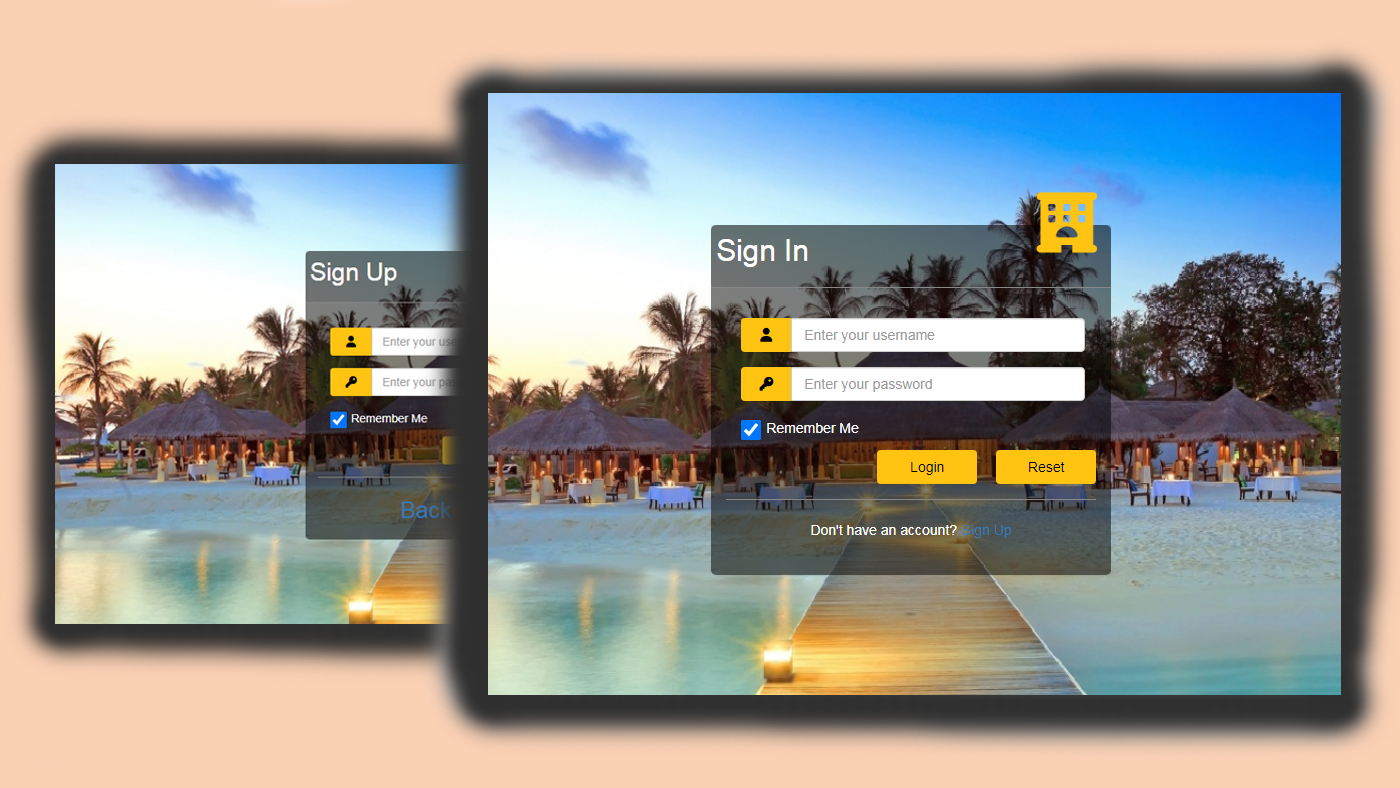
- Trang About:



- Các trang Service, Contact, Room:

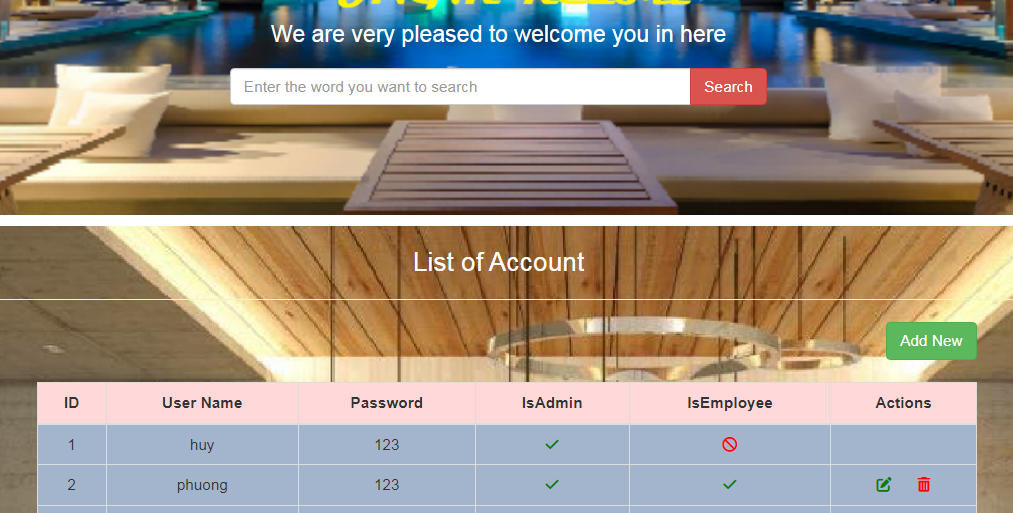


- Trang login và signUp

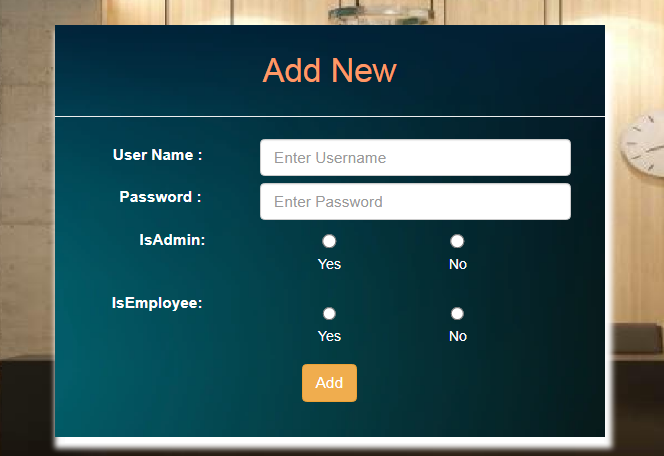


-Trang danh sách:

+ Hiển thị:



+Thêm:



+Sửa:

