**PROJECT FPT BOOKING**

(Web module)

1. **Tổng quan:**

* Ngôn ngữ lập trình chính: C#, .NET Core 3.1, ASP.NET Core 3.1
* Hệ quản trị database: SQL Server 2016
* Front-end framework: HTML/CSS/JS, Jquery, Bootstrap
* Kiến trúc tổng quát:
  + Web API: dành cho ứng dụng mobile sử dụng
  + Web Admin: kiến trúc Razor Page (gần giống MVVM), sử dụng API internal của web để tương tác với database
  + Sử dụng kiến trúc 3-tiers và 3-layers cho project
  + Database sử dụng Code first model (update code trước rồi từ đó update database)
  + Sử dụng Firebase service cho Authentication (đăng nhập bằng mail trường) và Notification (thông báo)

1. **Yêu cầu:**

* Hệ thống development:
  + .NET Core SDK 3.1
  + Visual Studio 2019
  + SQL Server 2016
* Hệ thống runtime ở server:
  + .NET Core Hosting Bundle 3.1:

<https://dotnet.microsoft.com/download/dotnet-core/thank-you/runtime-aspnetcore-3.1.8-windows-hosting-bundle-installer>

* + SQL Server 2016
  + Có cài đặt IIS features
* Kĩ năng cần thiết:
  + Lập trình C#, .NET, ASP.NET Core, Razor Page và Web API
  + Tìm hiểu về RESTFUL standard, HTTP, JSON
  + Cập nhật Database migration bằng Code first model sử dụng EF Core
  + Hiểu về kiến trúc 3-tiers và 3-layers
  + Biết sử dụng Cloud service: Firebase, Google …

1. **Cấu trúc thư mục:**

Bắt đầu từ thư mục gốc khi clone trên Git về máy:

* *FPTBooking/FPTBooking.sln:* File Visual StudioSolution chung của project
* *FPTBooking/FPTBooking.WebApi:* project Web API
* *FPTBooking/FPTBooking.WebAdmin:* project Web Admin
* *FPTBooking/FPTBooking.Data:* tầng Data của ứng dụng. Tầng này chứa các class Entity ứng với các table trong database và các thông tin Migration.
* *FPTBooking/FPTBooking.Business:* tầng Business của ứng dụng. Tầng này chứa các class Service làm việc với các Entity để hiện thực các nghiệp vụ chính của ứng dụng.
* *Còn lại:* Các project helpers

1. **Một số lưu ý đối với Backend module:**
   1. **Project *FPTBooking.Data*:**
   * Sử dụng Code first model:
     + Tìm hiểu cách sử dụng Code first của framework EF Core:

<https://entityframeworkcore.com/approach-code-first>

* + - Tìm hiểu cách thực hiện Migration sao cho hợp lí để tránh gây mất mát dữ liệu khi update database
    - Tóm tắt các bước:

1. Cài đặt các thư viện cần thiết
2. Cấu hình class DbContext và lớp DbContextFactory
3. Update code trong class Entity
4. Chạy lệnh Add-Migration [Migration-Name]
5. Chạy lệnh Update-Database
6. Kiểm tra thay đổi dưới database
   * Sử dụng ASP.NET Core Identity framework cho User management (hạn chế chỉnh sửa phần này hoặc tìm hiểu kĩ hơn về Identity framework)

Vd: các class AppUser, AppRole, …

* + Các constant được đặt tập trung ở file *Constants.cs*
  + Các cấu hình chung của tầng Data đặt ở file *Global.cs*. Vd: Dependency Injection, …
  1. **Project *FPTBooking.Business*:**
  + File *Settings.cs* dùng để binding cấu hình từ file sang class cho dễ sử dụng. (binding được thực hiện ở các project web)
  + Folder *Clients* chứa các class dùng để làm việc với các service bên ngoài hệ thống. Vd: FAP service
  + Khi sử dụng FAP client cần có service key, key này nằm trong phần cấu hình ở project Web. Nếu key hết hạn hoặc gặp trục trặc, liên hệ bộ phận IT để cấp key mới.
  + Folder *Models* chứa các class model dùng để binding dữ liệu từ client gửi lên vào một C# object hoặc để đóng gói thông tin từ server trả về cho người dùng.
  + Folder *Queries* chứa các file có các hàm Extension (xem thêm extension method) của các Entity tương ứng 🡪 tiện hơn trong việc build câu query database.

Cách này sử dụng Builder pattern. Vd:

var emptyRooms = rooms.Empty().NameContains(“A101”).ToList();

nghĩa là query các phòng trống tên có chứa chuỗi “A101”

* + Folder *Services* chứa các class Service chính của project.

Mỗi service xử lí một số business nhất định liên quan tới một hoặc nhiều Entity (bảng).

Service sẽ thực hiện các câu query, CRUD dữ liệu để đáp ứng nghiệp vụ của app thông qua Entity framework.

* 1. **Project *FPTBooking.WebApi*:**
  + QUAN TRỌNG: Khi clone project về phải xin 2 file secret *firebase.json* và *fap-secret.txt* ở trên server về
  + File *Startup.cs*: là file cấu hình tập trung của project web ASP.NET Core.

Vd: Dependency Injection, EF Core, Database connection, Config binding, Identity framework, Firebase …

* + Sử dụng NLog làm logging framework. Xem thêm NLog để biết thêm chi tiết.
  + *appsettings.json*: là file runtime setting tập trung của project web. Mỗi section có thể binding vào các class *Settings.cs* của project Data, Business để dễ sử dụng.
    - Ở mục ConnectionStrings 🡪 DataContext: điều chỉnh đường dẫn database tại đây bằng cách thay đổi server ip, database name (giữ nguyên {envConfig}, đổi User Id và Password
    - BusinessSettings 🡪 FapApiUrl: ứng với server của FAP hiện tại
    - BusinessSettings 🡪 StudentAllowed (true/false): cho phép sinh viên login vào app hay không. Khi chỉnh cần phải reset lại Web API (có thể chỉnh trực tiếp trên Web Admin không cần reset)
    - FirebaseCredential/FapSecretFile: chỉ đến các secret file tương ứng
  + Folder *Secrets* chứa các file secret key của project. Vd: FAP key, Firebase credential…

Các file này không nên đưa lên git (đã nằm sẵn trong gitignore)

* + Folder *Controllers* chứa các Controller class, ứng với các api endpoint mà client thấy.

Thiết kế theo RESTFUL API Standard. Vd: GET {base\_url}/api/rooms

* + Controllers sẽ sử dụng các class Service để thao tác nghiệp vụ.
  1. **Project *FPTBooking.WebAdmin*:**
  + QUAN TRỌNG: Khi clone project về phải xin 2 file secret *firebase.json* và *fap-secret.txt* ở trên server về
  + Cấu trúc khá tương tự project WebAPI
  + Sử dụng kiến trúc Razor Page (tìm hiểu thêm):
    - 1 razor page gồm có trang .cshtml và file code behind (Model) để xử lí logic.
    - Phần frontend bên .cshtml lấy dữ liệu được đẩy qua từ code behind.
  + *bundleconfig.json* và *libman.json*: quản lí lib bên frontend (hạn chế chỉnh sửa)
  + Folder *wwwroot*: chứa các static files như hình ảnh, script, html, media …
  + Folder *Views*: chứa các Partial View (hạn chế chỉnh sửa)
  + Folder *Pages*: chứa các page chính của web admin
    - Folder *Shared* chứa layout của web
    - Các folder khác chứa các module tương ứng
  + *appsettings.json*: là file runtime setting tập trung của project web. Mỗi section có thể binding vào các class *Settings.cs* của project Data, Business để dễ sử dụng.
    - Ở mục ConnectionStrings 🡪 DataContext: điều chỉnh đường dẫn database tại đây bằng cách thay đổi server ip, database name (giữ nguyên {envConfig}, đổi User Id và Password
    - BusinessSettings 🡪 FapApiUrl: ứng với server của FAP hiện tại
    - BusinessSettings 🡪 StudentAllowed (true/false): cho phép sinh viên login vào app hay không. Khi chỉnh cần phải reset lại Web API (có thể chỉnh trực tiếp trên Web Admin không cần reset)
    - WebAdminSettings 🡪 ApiUrl: đường dẫn của WebApi đã deploy (dành cho tính năng cấu hình WebAPI thông qua WebAdmin)
    - FirebaseCredential/FapSecretFile: chỉ đến các secret file tương ứng

**HẾT**