**This document is used to describe the classes and functions**

## Command

**RelayCommand.cs:** sử dụng để triển khai **ICommand**, hỗ trợ việc thực hiện các hành động từ ViewModel thay vì thực hiện các hành động ở code-behind. Với RelayCommand chúng ta có thể bind trực tiếp các hành động từ ViewModel vào các điều khiển trong xaml

**Converters**

**BoolToVisibilityConverter**: sử dụng để thay đổi trạng thái hiển thị, giúp kiểm soát hiển thị các phần tử trong giao diện người dùng (UI)

* Nếu true thì phương thức trả về là *Visibility.Visible*
* Nếu false thì phương thức trả về là *Visibility.Collapsed*

## Model

**DatabaseManager**: quản lý việc kết nối đến cơ sở dữ liệu Mysql trong ứng dụng

* Phương thức **GetConnection**: tạo một kết nối tới db

Các lớp*: Event, User* dùng để lưu trữ dữ liệu được truy vấn từ cơ sở dữ liệu

## Services

**GenSalt**: dùng để tạo ra một chuỗi ngẩu nhiên để thêm vào dữ liệu trước khi thực hiện hashing, dùng để đảm bảo dữ liệu khi hashing là duy nhất dù cho dữ liệu ban đầu có giống nhau.

**Hasher**: hashing một chuỗi dữ liệu đầu vào sử dụng thuật toán MD5 và trả về giá trị dưới dạng chuỗi hexadecimal

## ViewModel

**LoginViewModel**:

**ExecuteLogin**: khi người dùng click vào loginButton, thì lúc này sẽ tiến hành kiểm tra xem email mà người dùng nhập vô có tồn tại hay không, nếu không trả về lỗi “*UserName or Password is incorrect*!”, nếu tồn tại thì tiến hành lấy mật khẩu mà người dùng nhập + salt từ dữ liệu truy vấn và sử dụng phương thức hash trong Hasher để hash và tiến hành kiểm tra với mật khẩu được lưu trữ trong db, nếu giống nhau thì đăng nhập thành công, còn nếu sau sẽ in ra lỗi “*UserName or Password is incorrect!”*

**RegisterViewModel**:

**ExecuteRegister:** đầu tiên sẽ tiến hành kiểm tra xem email mà người dùng nhập vô có tồn tại trong db chưa nếu tồn tại sẽ in ra lỗi "Account already exists", nếu chưa tồn tại thì bắt đầu genSalt, sau đó hashing mất khẩu + salt và ghi xuống db

**EventSchedulingViewModel**

**ExecuteEvent**: khi người dùng clieck vào saveButton, thì lúc này sẽ tiến hành ghi thông tin event xuống db

**LoadOrders**: Xóa danh sách các đơn hàng hiện tại (Orders.Clear()).

Kết nối cơ sở dữ liệu và thực hiện truy vấn để lấy tất cả các sự kiện có status='0'.

Tạo các Grid cho mỗi sự kiện và thêm chúng vào danh sách đơn hàng (Orders).

**CreateOrderGrid**(DataRow row):

Tạo một Grid cho mỗi sự kiện, bao gồm thông tin sự kiện như tên, ngày, giờ và người đặt.

Thêm bb nút "Complete" để cập nhật trạng thái sự kiện khi nhấn.

Trả về Grid chứa thông tin sự kiện và nút "Complete".

**ResetForm**(object parameter):

Đặt lại tất cả các giá trị trong EventParams (như tên, số điện thoại, email, ghi chú, số bàn, v.v.) về giá trị mặc định (rỗng hoặc 0).

**OnStackPanelPressed**(object sender, PointerRoutedEventArgs e):

Khi nhấn vào một StackPanel, chuyển đến chế độ xem sự kiện.

Lấy thông tin chi tiết của sự kiện từ cơ sở dữ liệu và gán chúng vào các tham số (InfoParams) để hiển thị thông tin sự kiện.

**ReturnEventSchedulingPage**(object parameter):

Khi chuyển về trang lịch trình sự kiện, ẩn chế độ xem sự kiện và hiển thị lại chế độ xem mặc định (IsEventViewVisible và IsDefaultViewVisible).