KFC Single Sign-On Portal

Registration

Design Document

CECS 491B, Sec 11

March 13, 2019

Julian Poyourow
013466646

Revision History

Date	Version	Description
3/12/19	1.0	First draft.

Table of Contents

1. Introduction

The purpose of this document is to illustrate the flow of registration functionality within the SSO for KFC.

2. General Design

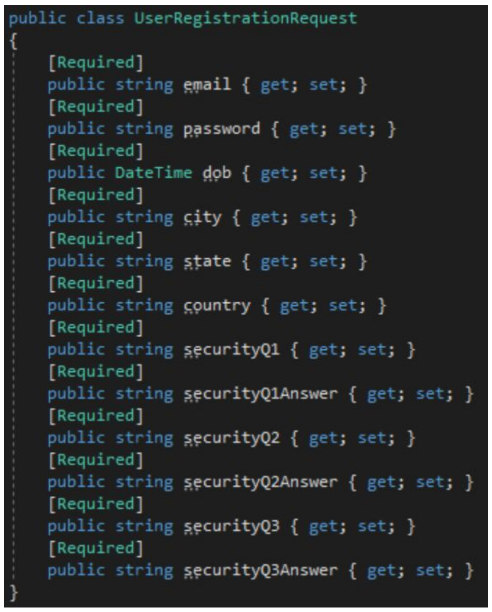
The controller layer is in charge of creating the database context, since that context will be shared across several manager calls. The controller is also in charge of managing formatting and interpretation, passing that data to the individual manager classes for processing. Lastly, the controller layer is in charge of handling any errors thrown by sub-layers, and returning an appropriate HTTP REST status code.

The manager layer accepts a database context, as well as the appropriate data. It then calls services, and orchestrates the processing of that data. The manager is responsible for throwing errors when unacceptable data or conditions are encountered.

An exceptions class has been created within the ServiceLayer to handle custom exception conditions. This class follows .NET practices as documented on the Microsoft Developer Documentation.

3. API

The route for registration is **POST /api/users/register** and is handled by the registration controller. This route expects a POST request with the following fields in the body, formatted as application/json:



```
public class UserRegistrationRequest
{
    [Required]
    public string email { get; set; }
    [Required]
    public string password { get; set; }
    [Required]
    public DateTime dob { get; set; }
    [Required]
    public string city { get; set; }
    [Required]
    public string state { get; set; }
    [Required]
    public string country { get; set; }
    [Required]
    public string securityQ1 { get; set; }
    [Required]
    public string securityQ1Answer { get; set; }
    [Required]
    public string securityQ2 { get; set; }
    [Required]
    public string securityQ2Answer { get; set; }
    [Required]
    public string securityQ3 { get; set; }
    [Required]
    public string securityQ3Answer { get; set; }
}
```

4. Additional Errors

- If any of the required fields, or the body itself, evaluates to null, the client will receive an HTTP 400 response.
- Any database errors will result in a 500 error, and all relevant data will be rolled back.