FLASHCARD O RAMA

in development product manual

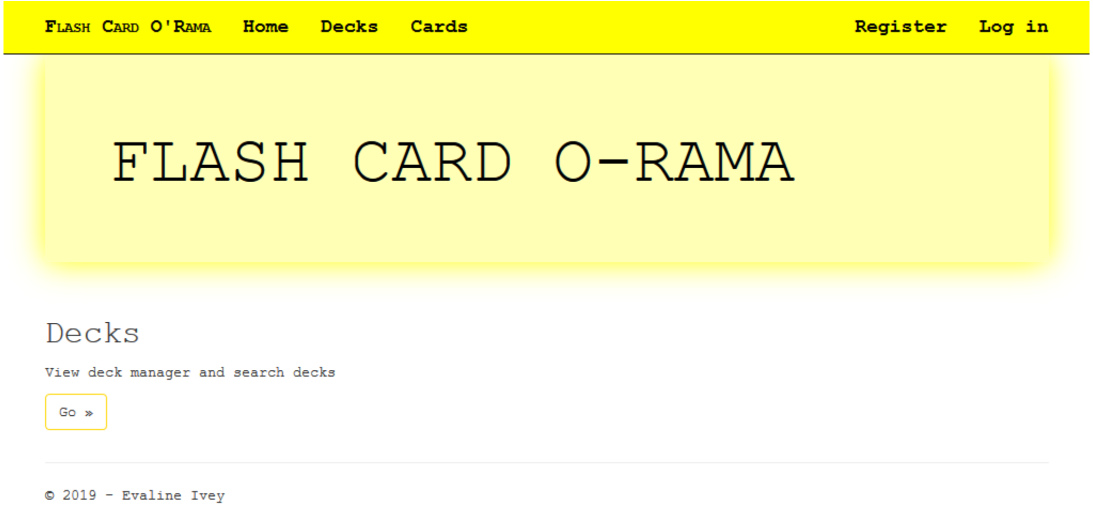
4/14/2019

Contents:

Evaline Ivey

User Manual:

The Home Page of the Application



FLASH CARD O-RAMA

Decks

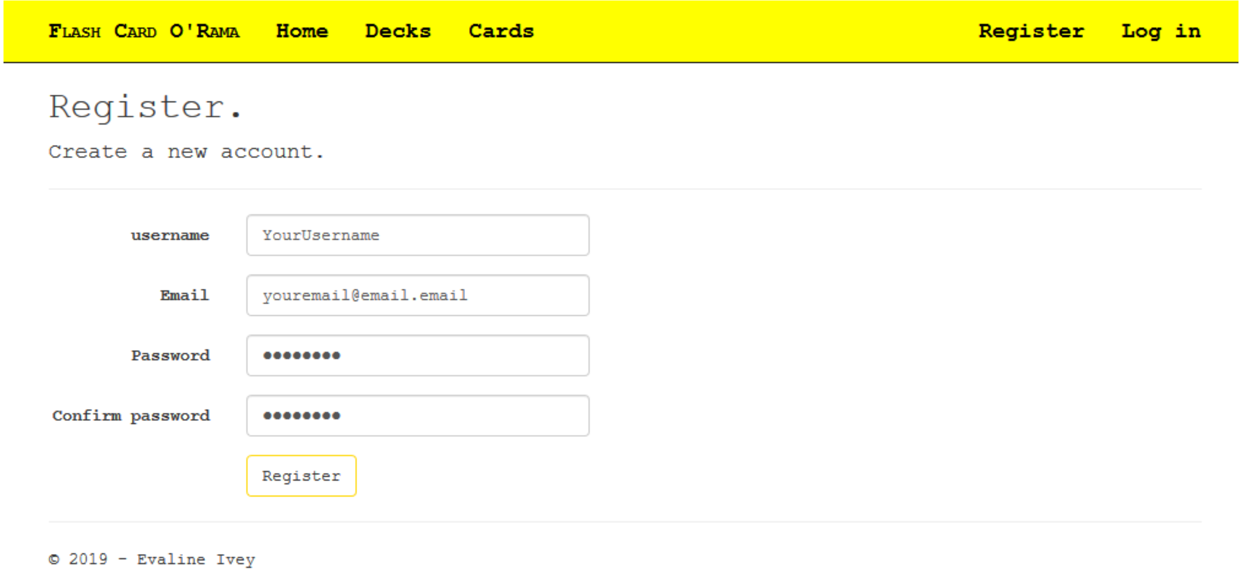
View deck manager and search decks

© 2019 - Evaline Ivey

This is the launch page of the application

Registration & Login

To register, select register at the top right corner of the page. does not currently send a confirmation email, though that ability can easily be added in the future.



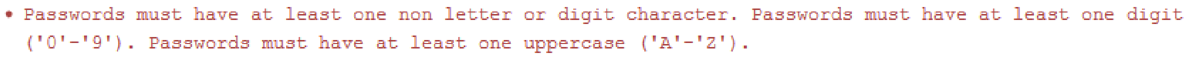
Register.

Create a new account.

username	YourUsername
Email	youremail@email.email
Password	••••••••
Confirm password	••••••••

© 2019 - Evaline Ivey

If the requirements for a secure password are not met, or you try to create a username that already exists. the form will provide feedback.

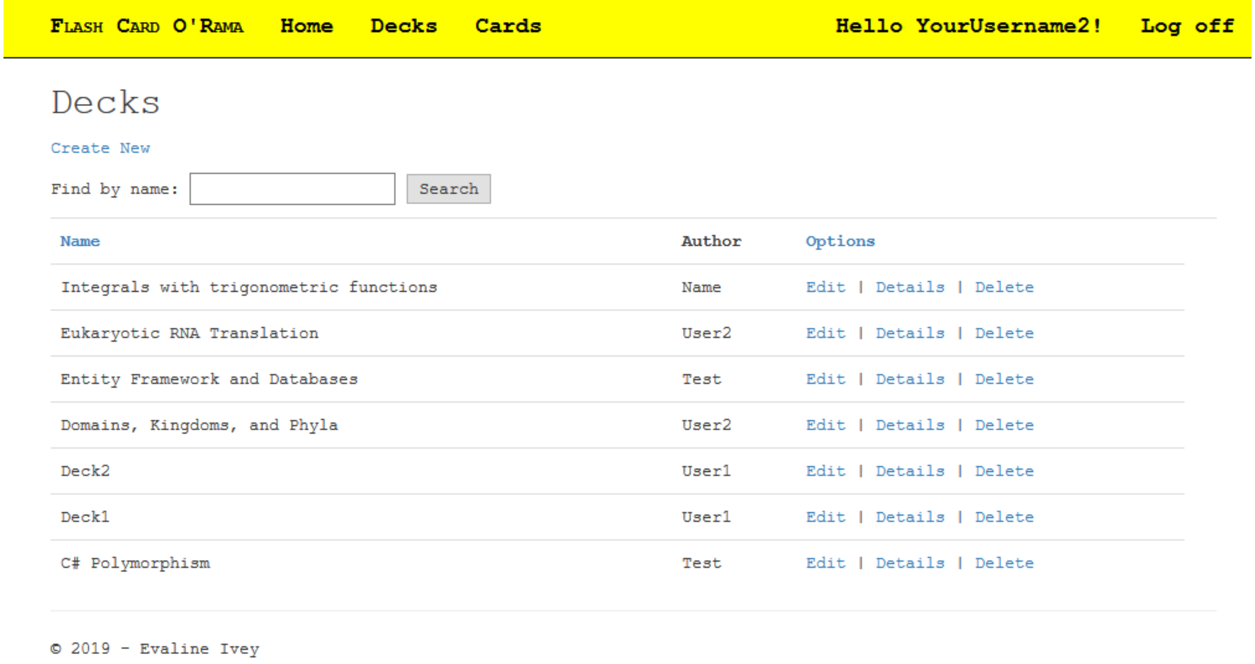


- Passwords must have at least one non letter or digit character. Passwords must have at least one digit ('0'-'9'). Passwords must have at least one uppercase ('A'-'Z').

Once you have created an account, you will be redirected to the home page, and your username should be displayed in the top right corner with a greeting message and a log off option.

Decks

The Deck management page is not complete, but it does contain a search option.



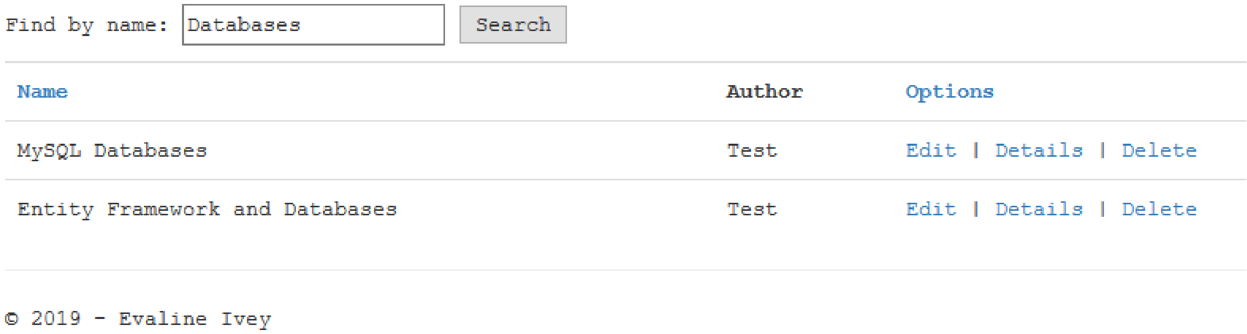
Decks

Find by name:

Name	Author	Options
Integrals with trigonometric functions	Name	Edit | Details | Delete
Eukaryotic RNA Translation	User2	Edit | Details | Delete
Entity Framework and Databases	Test	Edit | Details | Delete
Domains, Kingdoms, and Phyla	User2	Edit | Details | Delete
Deck2	User1	Edit | Details | Delete
Deck1	User1	Edit | Details | Delete
C# Polymorphism	Test	Edit | Details | Delete

© 2019 - Evaline Ivey

Searching for a word will result in all decks with that word in the title.



Find by name: Databases

Name	Author	Options
MySQL Databases	Test	Edit | Details | Delete
Entity Framework and Databases	Test	Edit | Details | Delete

© 2019 - Evaline Ivey

Currently, this application contains user *authentication,* but not *authorization.* In the future, users will only be able to edit and delete a deck that they have created.

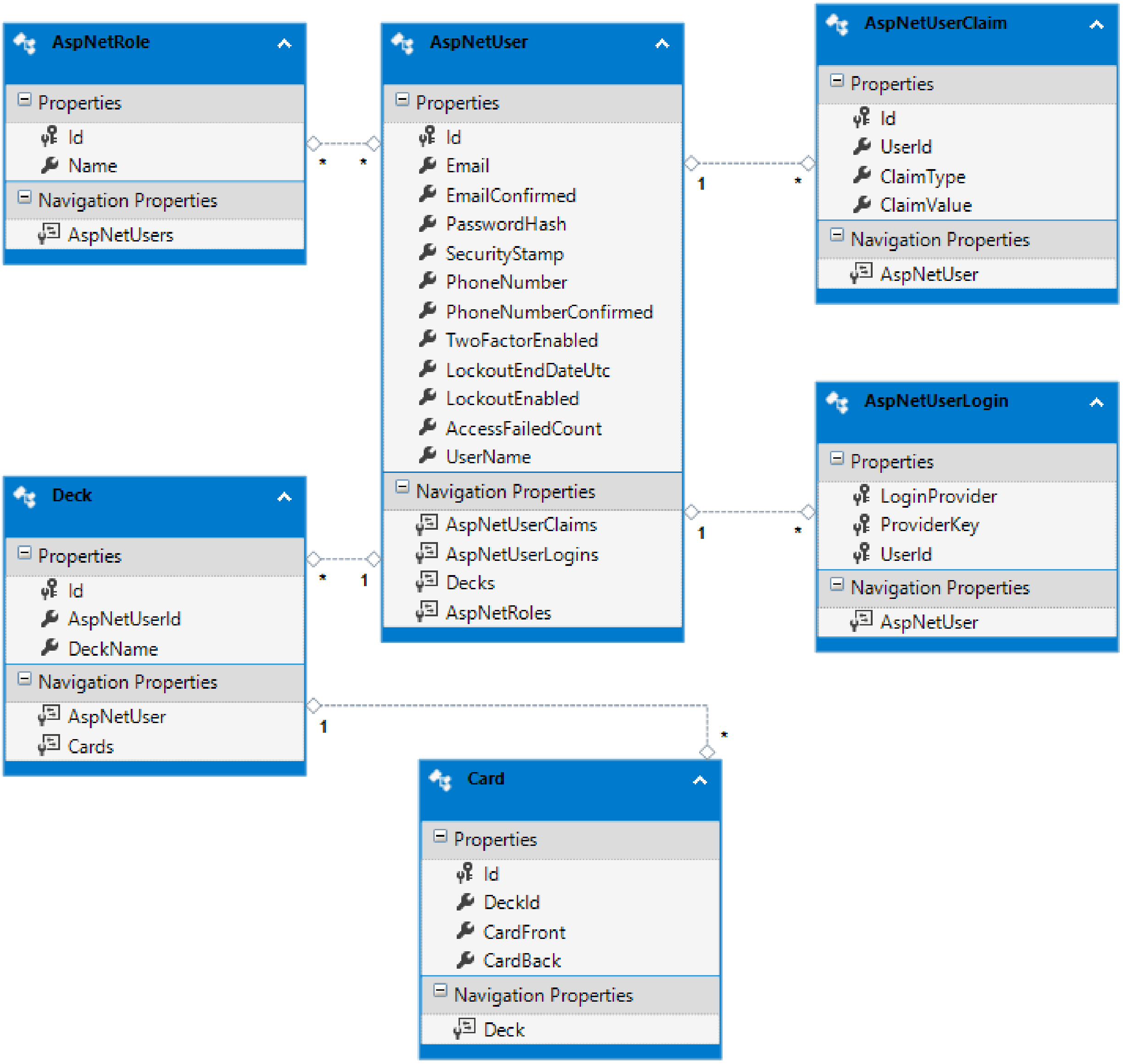
Technical Overview

This application is written in C# using ASP.NET, and uses just MVC. It uses bootstrap with some extra CSS for some minor style changes.

This application's user authentication uses the pre-built system for individual user accounts that can be selected when first creating a MVC web application in Visual Studio 2017.

Database

This application has seven tables. One is for keeping track of the application’s migration history and is not pictured in the UML diagram below. All of these tables but Decks and Cards were part of the authorization template, however, modification was needed to get the main user table, AspNetUser, to have a one to many association with Deck.



Tables

Default AspNet authentication tables:

-- AspNetUser

This table contains the user's main information. The things related to two factor authetification are unlikely to be used for a simple flash card web application and may be removed in the future.

-- AspNetRole and AspNetUserClaim

These databases are for use in different kinds of authorization. Roles based authorization and Claims based authorization.

In role-based authentication, a user is not allowed to perform an action if the action they are trying to perform declares a role that they do not have. For example, putting `[Authorize(Roles = "Administrator")],` above a method that performs a POST operation means that no one without that role associated to their user can perform that action. In this database Roles and Users have a many to many relationship, so that a role can be shared by many different users, and a user can have more than one role.

In claim-based authorization, a claim is some value associated with the user, and is specific to the user. They can be things like phone numbers, birth dates or emails, or they can also be randomly generated IDs.

This application does not currently use any authorization.

-- AspNetUserLogin

This is a database to contain login values.

Application Specific tables:

-- Deck

The deck table currently contains only a userID, the creator of the deck, and a deck name. A deck has a many to one relationship with a user, where a user can have many decks, and a deck can have one user.

Adding this relationship to users was more difficult than it should have been, due to "Generate Database from Model" on the diagram page not correctly generating associations, along with many other related issues. Eventually these problems were solved by simply writing SLQ queries to create the tables with associations, and to only update the model from the database

-- Card

A card has a many to one relationship with a deck, where a deck can have many cards.

Previously, there was an issue where a deck could not be deleted, but that was solved by adding `ON DELETE CASCADE` to the DeckCard foreign key relationship on the Card table, because that caused a decks associated card to delete when a deck was deleted.