Project Zero

Daniel Marabito,

Jean D King

Introduction

Goals

The goal of this project is to make a fan site for fans of the anime series Code Geass.
It is meant to increase the availability of information for fans of the show.
Many similar sites have their site structured as a wiki, this has its benefits and its drawbacks. The main drawback and feature of a wiki is to have users be able to update the content. This might lead to inaccuracies. So what we bring to the table is a more curated experience.

Target Audience

User Personas

We shall explore our target audience by presenting to the following User Personas as fictionalized versions of potential customers.
Peter is a recent highschool graduate, he is a 19 years old male currently going to college.he is a big fan of Code Geass and feels like running a Table-top campaign inspired by the world of Code Geass. This means he needs a good source of information for translation.

User Stories

As a Code Geass fan Peter engages with our site on occasion when the mood strikes him. He will occasionally look at it when he has a question about the world of Code Geass.

Functional Requirements

Screens

The first screen we will explore is Lelouch's Character page. The body of the page is primarily a text about lelouch. This is sourced from the about section of the character data model, and other properties. There is a clickable menu on the top right hand of the page that displays a sidebar with navigation links.

On it is a link to the All Characters View, it takes the user straight to the list of all characters in the Database. Some of these have associated links given that lead to the corresponding character's view, others just list them and indicate that they are not currently ready. This feature was chosen to help keep a level of curation to the content.

The Create a new character link in the sidebar takes a an individual to a page where they are prompted to enter a form about the Character: Name, About, and Powers associated with the character. And there is a submission button. At a later point security features should be added to restrict the creation of new characters. There is an Organization Page, which will redirect the user to the Organization page.

Help page is just the information about the page.

Messaging

Communication on a subject is something fans love. So we will have a space on some of our views in which the content can be discussed. This is done with a view component that displays comments related to the current subject and allows users to post their own comments. We will allow for relative anonymity in the comments by allowing users to post using a nickname of their choosing with the freedom of changing their name with each post. This will make moderation difficult but can allow unique interactions such as roleplay as cannon characters by simply setting their name to the character's name.

Controllers

In this first version we will have three controllers. The first one is the home controller which is in charge of the home page/ The second one is used to direct to organization pages for code geass. The third directs to character pages. For character pages there will be get and post actions for the page.

For the Create page, there is a Controller with an IActionResult, which will prompt the user to submit a form about the new character. If the user tries to submit without any information entered, the message will pop up as a warning, all the prompts must be filled before submission.

Data Models

In this first version we will be focused on just two data models.

Our first model is a character model. This model stores basic characteristics about a character including Name, Aliases, eye color, about and powers.

Our second model is an organization model. This model stores the name of the organization, a description of the organization, and a character that is the leader.

Views

We have several views for our minimum viable product. What follows is a table with the view name, the standard route to get to it and a brief description of it.

View Name	Standard Route	Description
Home	/Home/Home	The landing page acts as a starting place with links.
Help	/Home/Help	Gives a place to post grievances and contact info.
Lelouch	/Characters/Lelouch	Covers the character Lelouch Vi Britannia, the main character of Code Geass.
Create	/Characters/Create	This enables us to create additional characters that are added to our database.
All (In characters folder)	/Characters/All	Lists all the characters currently with data in our database and provides links to those that are currently supported.
Black Knights	/Organizations/Black_Knights	Is about the organization the Black Knights, which is led by Lelouch.
All (in organization folder)	/Organization/All	Lists all the organizations currently in our database. Provides links to organizations if it is supported.

In addition to the views mentioned in the table above there is also the Shared Component Messenger which supports messaging. See the messaging section for more details.

Persistant data storage

To manage our database we will be using an entity framework to serve as a sql database.

Routes

So our main routing scheme is /controller/action. This means we will have a /Character/Lelouch as one of our routes, We will also Have /Organization/Black_Knights.

We have also, Characters/create

And characters/help.

We will also provide some additional routing through a custom middleware that will update the path based on Aliases of characters. Thus /Character/Zero will lead to /character/Lelouch.

Below is a chart of planned Aliases

User entered Path	Updated path
/	/home/home
/home	/home/home
/Help	/home/Help

/all	/Characters/all
/Character/Zero	/Character/Lelouch
/Zero	/Character/Lelouch
/Create	/Characters/Create
/Organizations/Black-Knights	/Organizations/Black_Knights
/Black-Knights	/Organizations/Black_Knights
/help	/Characters/help

Non-Functional Requirements

Performance Considerations

A key consideration for performance is our caching policy. It is important for us to consider when a page changes.

Security Considerations

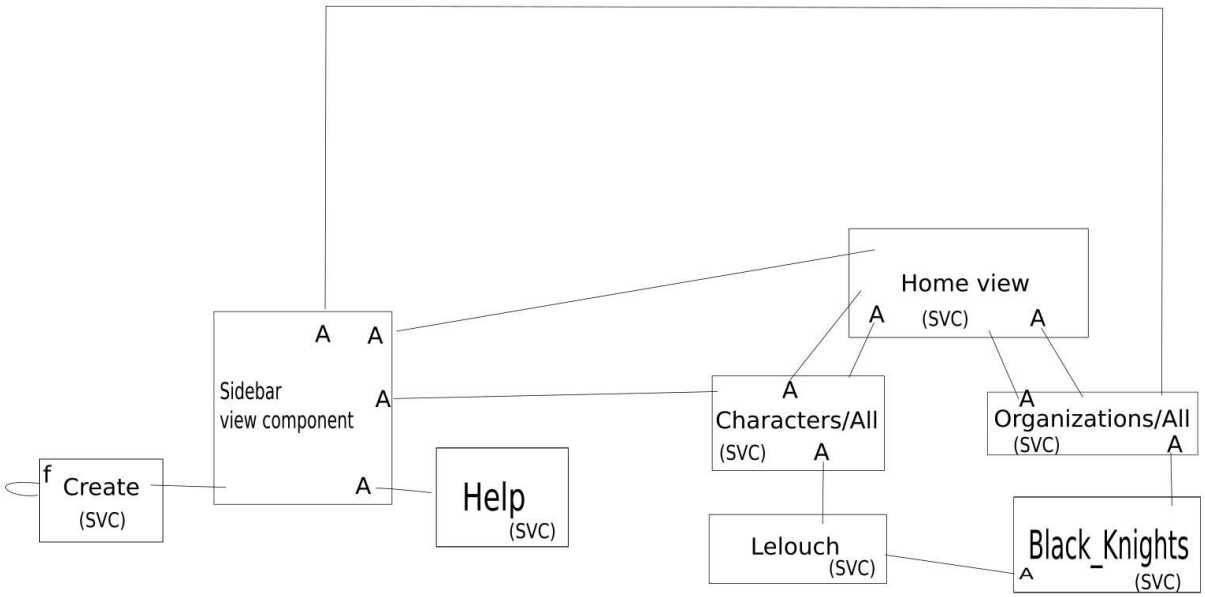
We have to be aware of the possibility of various types of attacks. We shall be focusing our attention on security in a later round of development. During which our first concern is with handling bad data such as sql injection attempts. With that in mind we have to make sure we sanitize any input we receive from the user via forms, query strings, segment variables and any other source of user input.

An additional layer of security for our forms we will also focus on avoiding Cross-Site Scripting attacks. To help mitigate this we shall include a hidden input on our forms that includes a CRSF token.

Another security detail that should be applied is access to characters/create should be restricted to limit who can create and edit them.

Navigation Structure

Below is our Site map. On it you will see an “A” on a view indicating there is a link going to where the line goes. You will also see the (SVC) that is short for the sidebar view component which means each of those pages with (SVC) has access to all the links that the Sidebar view Component has. The f in create means that it connects to itself via a form. There is a bit of obfuscation with that since it does go to a controller action that redirects back to the create but I thought it would get alot more complicated if I include controllers on the site map.



Other Resources