1. März 2024

Elijah & Amar

Bibliotheke\_dokumentation

Unsere Dokumentation über unser Bibliothekverwaltungssystem in ASP.NET

M320 Projektdokumentation: Bibliotheke

# 1. Einführung

In diesem Projekt haben wir ein Bibliothekverwaltungssystem gemacht, wo der User alle Bücher verwalten kann. Er kann die Bücher erstellen, lesen, bearbeiten und löschen.

# 2. Projekteinrichtung

Wir haben uns zuerst überlegt, was wir genau in dem Projekt haben möchten. Wir haben einen Plan erstellt und noch die Klasse für die Bücher wo der User dann benutzten kann, um seine Bücher zu erstellen, lesen, bearbeiten und löschen.

Wir haben uns für ASP.NET entschieden, da es am besten ist eine CRUD zu erstellen. Das Ganze wurde auch in Visual Studio 2022 gemacht, da es für uns die beste Programmierplattform ist zum das Projekt starten, vor allem mit den Funktionen von GitHub Copilot.

Amar hat mir seine Codes via Discord geschickt. Er hat mir die Search Funktion gemacht im BooksController, den Home/Index & Privacy. Er hat mir noch geholfen mit dem Layout von dem CRUD sowie auch Design Ideen mit dem Bootstrap und JQuery.

# 3. Konfiguration

Die Konfiguration der Anwendung wird über die 'appsettings.json' und 'appsettings.Development.json' Dateien gesteuert. Diese Dateien enthalten wichtige Einstellungen wie Datenbankverbindungszeichenfolgen und Anwendungsparameter, die für die Entwicklung und Produktion angepasst werden können.

# 4. Architekturübersicht

Die Anwendung folgt dem MVC (Model-View-Controller) Architekturmuster, um eine klare Trennung von Anwendungslogik, Benutzeroberfläche und Datenzugriff zu gewährleisten. Dieser Abschnitt bietet einen Überblick über die Hauptkomponenten der Anwendung und ihre Interaktion.

* Wir haben als Authentifizierungstyp die «Einzelne Konten» benutzt damit wir noch User Accounts leicht erstellen können, dass hilft uns nachher die Berechtigungen und Bearbeitungen von den Accounts zu managen.
* Wir haben ein «Bücher» Model mit allen wichtigen Eigenschaften, um im Programm die Bücher mit Ihren wichtigen Eigenschaften einzuspeichern.
* Wir haben Rollen erstellt «Admin & User», der Admin kann die Bücher erstellen, lesen, bearbeiten und löschen, und der User kann die Bücher nur lesen. Der User ist der Kunde und der Admin ist der Arbeiter.
* Wir haben ein Search-Feature hinzugefügt, wo nach dem Namen des Buches sucht, damit es dem Kunde und dem Arbeiter die Suche nach den Büchern erleichtert.
* Wir haben Bootstrap und JQuery benutzt, um eine schöne Oberfläche zu generieren
* Wir haben Authorizations gemacht, damit der User sich zuerst einloggen oder registrieren muss, um die Bücher zu lesen. Wir haben es auch gemacht damit der User nur die Bücher lesen kann und der Admin hat zugriff auf Alles. Wir wollen nicht, dass der User Zugriff auf die Datenbank hat sowie auch andere sensible Daten, wo er dann angreifen kann.

# 5. Detaillierte Komponenten

Im Folgenden werden die Schlüsselkomponenten des Projekts detailliert beschrieben:

* Controllers: Verarbeitet Anfragen von Benutzern (HomeController.cs, BooksController.cs)
* Models: Definieren die Datenstruktur. (ErrorViewModel.cs, Books.cs)
* Views: Verantwortlich für die Darstellung der Benutzeroberfläche. (Books, Home, Shared, \_Viewimports.cshtml and \_ViewStart.cshtml)
* Data: Enthält den Datenzugriff und die Datenbankmodelle. (CreateIdentity, initialsetup)
* wwwroot: Beinhaltet statische Ressourcen wie CSS und JavaScript-Dateien.

# 6. Entwicklungsrichtlinien

Um die Konsistenz und Wartbarkeit des Codes zu gewährleisten, sollten Entwickler folgende Richtlinien beachten:  
  
- Einhalten von Namenskonventionen und Coding Standards.  
- Kommentierung und Dokumentation des Codes, wo notwendig.  
- Regelmässige Code-Reviews und Design-Reviews.



