# Verwendete Technologien & Programme

Datenbank: MySQL + Workbench + Xampp

Webanwendung: Blazor + C# + Visual Studio, Librarys (Dapper, MySQL.Data)

**Verwendete MariaDB Version: 10.4.17**

**MySQL Workbench 8.0.23 CE:** Die Workbench ermöglicht hier das schnelle und

übersichtliche Erstellen von ER-Modellen und zusätzlich eine Oberfläche zur

Erstellung von Queries mit den jeweiligen Ausgaben. Wir haben dieses Tool bereits in der Berufsschule für jegliche Anwendungsbereiche verwendet und somit war ich schon dementsprechend mit diesem Tool vertraut.

**Xampp:** Dieses Tool installiert und betreibt die nötigen Server / Komponenten für

die Umgebung. Einen Webserver (Apache), und einen DB (MariaDB)

Server. Es reduziert den Installations / Konfigurationsaufwand enorm und

eignet sich ideal zur Entwicklung auf dem eigenen Rechner (localhost)  
Diese Software wurde ebenfalls bereits in der Berufsschule für die obengenannten Einsatzbereiche verwendet.

**Blazor:** Ist ein Open Source Web Framework, welches die Entwicklung von Web Anwendungen mit Hilfe von C# und HTML ermöglicht. In diesem Fall handelt es sich um eine Blazor Server App.  
Diese Anwendungen werden auf einem ASP.NET Server gehostet.

**C#:** C# ist eine objektorientierte und typsichere Programmiersprache. Sie integriert sich in .NET-Plattformen, die Cross-Development für Anwendungen für Windows, macOS, iOS, Android und Linux ermöglichen.

**HTML:** Die Hypertext Markup Language ist eine textbasierte Auszeichnungssprache zur Strukturierung elektronischer Dokumente wie Texte mit Hyperlinks, Bildern und anderen Inhalten. HTML-Dokumente sind die Grundlage des World Wide Web und werden von Webbrowsern dargestellt.

**Visual Studio Community 2019:** Ist eine von Microsoft herausgebrachte Entwicklungsumgebung für verschiedene Programmiersprachen wie z.B.: C#, Javascript, HTML, CSS, C++.   
Mit ihr ist es möglich unter anderem Webanwendungen, dynamische Webseiten bzw. Webservices, Apps für Windows, iOS, macOS und Android zu entwickeln

**Dapper:** Dapper ist ein objektrelationales Zuordnungsprodukt für die Microsoft .NET-Plattform: Es bietet ein Framework für die Zuordnung eines objektorientierten Domänenmodells zu einer herkömmlichen relationalen Datenbank.

**MySQL.Data:** Wird für den Verbindungsaufbau verwendet. Damit man in C# eine Verbindung zu MySQL herstellen kann.

**NETStandard.Library:** Untergerüst für Blazor

# Programmaufbau

Program.cs: Wenn man die Anwendung startet wird als allererstes public static void Main aufgerufen. Diese nutzt Startup.cs um Konfigurationen vorzunehmen. (Standard/automatisiert bei der Projekterstellung erstellt)

Startup.cs: Bei public void ConfigureServices: werden verschiedene Services hinzugefügt, unter anderem auch DataAccess, damit die einzelnen Pages später darauf zugreifen können.

Public void Configure: nimmt Einstellungen vor wie sich die Webseite verhält z.B.: unterschiedliche Einstellungen für Development oder Production und Umleitung von http auf https und der Verwendung von Routing in der App.

appsettings.json: Hier gibt man die Verbindung zur MySQL Datenbank an, auf die die Anwendung später zugreifen soll.

DataLibrary: Diese erstellte Bibliothek ermöglicht einen generischen Zugriff auf Datenbankbefehle.

* IDataAccess.cs: Interface, das festlegt welche Aktionen, für Datenbank Zugriffe möglich sein sollen.
* DataAccess.cs: Implementierung des Interfaces. Hier werden die Methoden erstellt, welche es ermöglichen Daten speichern (Update, Delete, Insert) und laden (Select) zu können.

Shared:

* MainLayout.razor: Erstellt das Grundgerüst der Webseite, bestehend aus dem fixen Teil links sowie oben (Navbar, Header) und dem variablen Teil (body), der je nach Navigation andere Informationen anzeigt.
* NavMenu.razor: Hier werden alle Menüpunkte (Pages), durch die Route (z.B.: /impfanmeldung, /schema) in das Navigationsmenüs eingebunden. Jede Page hat eine eigene Route, die in der jeweiligen Razor Page (ganz oben) definiert wird. Dadurch kann im Navmenu darauf verwiesen werden.

Models:

* ForeignKeyConstraintModel.cs
* ImpfAdressModel.cs
* ImpfModel.cs
* SchemaModel.cs
* TabellenModel.cs
* TabellenStrukturModel.cs

Pages:

* Terminbuchung.razor
* Impfanmeldung.razor
* Index.razor
* Personenübersicht.razor
* Schema.razor
* Struktur.razor
* Tabellen.razor