Web-Presenter

2020

An app for giving online presentations

Sebastian Grman, Florian Schwarzl

4bhif | 4BHIF

Inhaltsverzeichnis

[1) Verteilung der Aufgaben 2](#_Toc42852220)

[2) Benutzeranleitung 2](#_Toc42852221)

[2.1) Projektidee 2](#_Toc42852222)

[2.2) Startanleitung 2](#_Toc42852223)

[2.3) Präsentation bearbeiten 3](#_Toc42852224)

[2.4) Präsentation starten 3](#_Toc42852225)

[2.5) Präsentation beenden 4](#_Toc42852226)

[3) Technische Beschreibung 4](#_Toc42852227)

[3.1) Serverseitiger Aufbau 4](#_Toc42852228)

[3.1.1) Program.cs &Startup.cs 4](#_Toc42852229)

[3.1.2) Presentation.cs 5](#_Toc42852230)

[3.1.3) PresentationFundamentals.cs & RunningPresentationFundamentals.cs 5](#_Toc42852231)

[3.1.4) Helper.cs 5](#_Toc42852232)

[3.2) Services 5](#_Toc42852233)

# Verteilung der Aufgaben

# Benutzeranleitung

## 2.1) Projektidee

Das Projekt soll die Möglichkeit bieten, schnell und einfach online eine Präsentation zu erstellen und diese mit anderen anzusehen. Das Prinzip dahinter ist ähnlich wie bei vielen Webseiten. Der Vortragende erstellt die Präsentation und bearbeitet sie, danach teilt er den Link mit seinen Zuschauern. Während die Präsentation läuft sehen die Zuschauer natürlich nur die jeweilige Folie, dem Ersteller hingegen stehen Steuerungsfunktionen zur Verfügung. Abgesehen von den grundlegenden Funktionen sind auch noch das Speichern der Präsentationen in einer Datenbank und eine Liste aller aktiven Präsentationen implementiert.

2.2) Startanleitung  
Der Startablauf kann aus zwei Sichten betrachtet werden. Der Vortragende erstellt zuerst eine Präsentation, indem er entweder auf der Startseite auf „Create Presentation“ klickt, oder in der Navigationsleiste am oberen Bildschirmrand auf „Create“ drückt.

Im nächsten Schritt muss der Name und der Titel der Präsentation angegeben werden. Der Name ist jenes Erkennungsmerkmal, unter dem die Präsentation gespeichert wird und mit dessen Hilfe man sie später wieder aufrufen kann. Er kann nachher nicht mehr geändert werden. Der Titel ist jener Text, den die Zuschauer während der Vorstellung als Überschrift sehen.

Nachdem man nun Beides eingegeben hat kommt man zum Herzstück des Projekts, der Bearbeitungs-/Präsentationsseite. Hier kann man seine Präsentation nach belieben bearbeiten.

Die derzeitige Implementierung ist so, dass diese Seite auch gleichzeitig die Präsentationsseite ist. Dass heißt, dass man jetzt bereits über den unten zur Verfügung gestellten Link als Zuschauer beitreten kann, die Präsentation muss nicht erst gestartet werden. Man sollte daher sicherstellen, dass man mit dem Bearbeiten fertig ist, bevor man den Link teilt. Es kann aber auch von Vorteil sein, sollte man mit mehreren Personen zusammen eine Präsentation erstellen, so können die Mitwirkenden live bei der Erstellung zusehen.

## 2.3) Präsentation bearbeiten

Die ohne Zweifel wichtigste Funktion ist das Bearbeiten einer Präsentation, bevor man sie den Zuschauern zeigt. Hierzu bietet der Web-Presenter grundsätzlich zwei verschiedene Modi: den Textmodus (Text) und den Folienmodus (Slides).

Im Textmodus erstellt man die Präsentation als Text, dass heißt, man schreibt den Content in das Eingabefeld und er wird, der gewählten Formatierung entsprechend, angezeigt. Für die Formatierung stehen simple Paragraphen, welche sich wie normaler Text verhalten, Monospace für gleichbleibende Zeichenabstände, Markdown und HTML zur Verfügung. Die letzten beiden sind komplexer als zum Beispiel „Text“, da sie seperate Formatierung und Syntaxrichtlinien haben.

Abgesehen vom Inhalt der Präsentation können auch Notizen gemacht werden, welche auch für spätere Verwendung gespeichert werden.

Der Folienmodus (Slides) ermöglicht es, Bilder hochzuladen, welche dann als eine Folie angezeigt werden. Der Modus einer Präsentation gilt für alle Folien, dass heißt, eine Präsentation kann nur in einem der beiden Modi sein.

## 2.4) Präsentation starten

Sobald man mit dem Bearbeiten fertig ist, kann man ganz einfach mit der Präsentation starten, indem man den Link an seine Zuschauer versendet. Wie schon zuvor erwähnt können diese sofort die aktuelle Folie sehen, daher ist es wichtig, dass man alles vorbereitet hat.

Als Zuschauer kann man auch über die Liste aller aktuellen Präsentationen, welche man in der Navigationsleiste unter „List“ findet, einer Präsentation beitreten. Dazu muss man nur auf den entsprechenden Eintrag in die Liste klicken, daraufhin erscheint ein Knopf „View“, über welchen man in den Zuschauerbereich gelangt.

Um eine bereits erstellte Präsentation, welche in der Datenbank gespeichert ist, zu bearbeiten, muss man in der Navigationsleiste auf „Start“ gehen. Dort wird man aufgefordert, den Namen – nicht den Titel – der Präsentation anzugeben. Nachdem man das getan hat gelangt man über „Start editing“ wieder in den Bearbeitungsmodus (siehe 2.3). Weiters kann man auch über die Liste aller Präsentationen, eine Präsentation auswählen und starten und kommt auch in den Bearbeitungsmodus.

## 2.5) Präsentation beenden

Seine aktuelle Präsentation kann der Vortragende in der Bearbeitungssicht ganz einfach beenden, indem er am unteren Bildschirmrand auf „end“ klickt.

# Technische Beschreibung

Wir haben unser Projekt auf Basis von ASP.NET Core in C# erstellt. Für die Darstellung am Client wurde Angular gewählt. Als Datenbank dient Postgres.

## 3.1) Serverseitiger Aufbau

Aufgrund der Verwendung von C# als serverseitige Programmiersprache ist das Projekt in einige Dateien aufgeteilt, welche in den folgenden Abschnitten genauer erläutert werden.

### 3.1.1) Program.cs & Startup.cs

Diese beiden Dateien stellen das Fundament dar, auf welchem das Projekt aufgebaut ist. In ihnen befindet sich nur wenig projektbezogene Funktionalität, da sie für grundlegende Aufgaben verantwortlich sind.

**Program.cs** enthält die Main-Funktion und den HostBuilder, welcher den Webserver startet.

In **Startup.cs** befindet sich die eigentliche Konfiguration des Servers. In der Methode „ConfigureServices“ werden die sogenannten Services registriert.

Services sind, vereinfacht gesagt, Abhängigkeiten, welche über Dependency Injection eingefügt werden. Man kann sie in etwa mit Node-Modulen vergleichen. Sie stellen dem Server eine gewisse Funktionalität zur Verfügung. Ein Service kann verschiedene Lebensdauern haben, welche durch die jeweilige Funktion, mit der er zu „services“ hinzugefügt wird, festgelegt wird. In diesem Projekt kommen hauptsächlich „Singleton“ und „Transient“ vor.

Singleton entspricht, wie der Name schon sagt, dem Singleton Pattern, das heißt, der Service wird nur beim ersten Aufruf erstellt und alle späteren Aufrufe greifen auf diese Instanz zu.

Transient ist mehr oder weniger das Gegenteil von Singleton, da hier bei jedem Aufruf des Services eine neue Instanz erstellt wird.

Abgesehen von den Services werden noch die Konfigurationen für Angular und die Verbindung mit der Datenbank in Startup.cs festgelegt.

### 3.1.2) Presentation.cs

Diese Klasse beschreibt eine Präsentation. Sie enthält die Daten der Präsentation als Attribute und dementsprechende Methoden, um diese abzufragen und zu bearbeiten.

### 3.1.3) PresentationFundamentals.cs & RunningPresentationFundamentals.cs

PresentationFundamentals stellt eine Präsentation dar, wie sie aus manchen API-Funktionen ersichtlich ist, dass heißt ohne Inhalt und nur mit Name, Titel und dem Ersteller.

RunningPresentationFundamentals hat einen ähnlichen Zweck, es ist jedoch wie der Name schon sagt, für aktive Präsentationen zuständig und hat daher eine ID als Attribut.

### 3.1.4) Helper.cs

Diese Klasse ist eine Hilfsklasse, welche praktische Funktionalitäten, hauptsächlich im Bezug auf die Verarbeitung der hochgeladenen Bilder, enthält.

### 3.1.5) DbConnectionString.cs

Enthält die Informationen für die Verbindung mit der Datenbank.

## 3.2) Services

Die Services sind wie schon zuvor beschrieben Dependency Injections, welche unterschiedliche Aufgaben erfüllen.

### 3.2.1) StorageService.cs

Der StorageService ist eine simple Implementierung eines Dictionarys, welche zur Speicherung der Präsentationen und Gruppen verwendet wird.

### 3.2.2) PresentationsService.cs

Dieser Service ist für die Bereitstellung der Präsentationen zuständig. Er bietet verschiedene Methoden, um Präsentationen abzurufen, zu starten und zu beenden. Die meisten dieser Methoden greifen auf die Klassen Presentations.cs oder PresentationFundamentals.cs zu, um die Präsentationen bereitzustellen.

### 3.2.3) PresentationDataService.cs

PresentationDataService stellt die Schnittstelle zur Datenbank dar. Hier werden die Präsentationen aus der Datenbank geladen und den anderen Services bereitgestellt. Weiters werden auch neue Präsentationen in der Datenbank gespeichert.

### 3.2.4) GroupManager.cs

GroupManager hat, im Gegensatz zu den anderen Services, nichts mit den Präsentationen zu tun. Er ist für die separaten „Räume“, in denen die Präsentationen abgehalten werden, verantwortlich. Diese Funktionalität wird von SignalR bereitgestellt.

## 3.3) Models

Models sind die Darstellungen der Daten aus der Datenbank. Sie repräsentieren die Tabellen in der Datenbank und sind für den einfachen Zugriff notwendig.

### 3.3.1) PresentationData.cs

Dieses Model stellt eine Präsentation dar, wie sie in der Datenbank gespeichert wird.

### 3.3.2) Users.cs

Users stellt einen Benutzer, also jemanden, der Präsentationen anlegt, dar.

## 3.4) Hubs

### 3.4.1) PresentationsHub.cs

Das Hub ist das Herzstück der Verbindung zwischen Server und Client, es stellt jene Methoden zur Verfügung, welche der Client über die Websocket-Verbindung aufrufen kann. Diese Funktionalität ist ein grundlegender Teil von SignalR.

Die bereitgestellten Methoden sind für alle „Aktionen“ verantwortlich, die der Benutzer machen kann. Dazu gehören zum Beispiel das Setzten des Titels/Namen/Präsentationsmodus oder der Notizen. Auch das Wechseln zwischen den Folien wird hier geregelt.

Kurz zusammengefasst kann der Client alle Methoden, welche im Hub als „public“ deklariert sind, aufrufen.

## 3.5) Data

In diesem Ordner befindet sich nur der WebPresenterContext, welcher für die Abfrage der Daten über die Models zuständig ist. Er bietet eine einheitliche Schnittstelle für den Zugriff auf die Präsentationen und bedient sich dabei der Klassen PresentationData und Users.

## 3.6) Controllers

Die Controller stellen API Schnittstellen dar und sind für die Datenübertragung zum Client verantwortlich. Dies ist nicht zu verwechseln mit dem Hub, welcher für „Aktionen“, also zum Beispiel das Ändern der Folie, zuständig ist. Controller sind für die Übertragung von zum Beispiel den Bildern der Folie zum Client verantwortlich.

Ihr Aufbau ist recht simpel, in eckigen Klammern steht vor der Funktion die http-Methode, auf welche die jeweilige Funktion reagiert. In runden Klammern steht danach noch gegebenenfalls der Pfad, auf welchem die Schnittstelle erreichbar ist. Danach folgt die Funktion, welche die Daten zurückliefert oder empfängt.

### 3.6.1) PresentationsController.cs