M151\_CodeSnipper

Inhalt

[Ziel von CodeSnipper 2](#_Toc105347962)

[Architektur 3](#_Toc105347963)

[4-Tier Architektur 3](#_Toc105347964)

[Datenbank 4](#_Toc105347965)

[Datensicherheit 4](#_Toc105347966)

[Sessions 4](#_Toc105347967)

[Transaktionen 4](#_Toc105347968)

[SSL/TLS 4](#_Toc105347969)

[Arbeitsjournal 4](#_Toc105347970)

[Sicherheit 4](#_Toc105347971)

# Ziel von CodeSnipper

CodeSnipper ist eine Plattform, welche das einfache Speichern und Veröffentlichen von CodeSnippets ermöglicht.

Als Entwickler gibt es vielmals Code, welchen man wiederholt schreiben muss. Um diese lästige Arbeit zu vernichten, können solche Ausschnitte in CodeSnipper notiert werden.

Die Software bietet die Möglichkeit, Snippets als privat oder öffentlich abzuspeichern, womit andere ebenfalls profitieren können. Zusätzlich kann man einem Snippet einen Titel und Kategorien zuweisen, wobei das Snippet schnell gefunden werden kann.  
CodeSnipper verzichtet bewusst auf eine Beschreibung, da die Ansicht so simpel wie möglich gehalten wird, und das Snippet so schnell wie möglich kopiert werden kann.

Beispielsweise können ConnectionStrings, die Methode "RaisePropertyChanged" und weitere nützliche Funktion mit CodeSnipper kopiert werden.

# Architektur

In diesem Kapitel wird erläutert, wie genau die Web-Applikation gebaut wurde. Es werden Dinge über die Architektur beschrieben, Themen, welche mit Datenbanken zu tun haben, aufgegriffen, sowie manche Dinge zu Sicherheitsaspekten, wobei das meisten im Kapitel [Sicherheit](#_Sicherheit) dokumentiert ist.

## Allgemeines

Um unsere Vision zu realisieren, benutzten wir gewisse Frameworks, welche hier kurz erwähnt werden:

* C# als Haupt-Programmiersprache
* .NET 6.0
* ASP.NET MVC
* Entity Framework
* Standard-Login von ASP.NET MVC

## 4-Tier Architektur

Unsere Web-Applikation hat eine 4-Tier Architektur implementiert. Dies bedeutet, dass es 4 Teilbereiche gibt, welche zusammen harmonieren. Unsere verschiedenen Layers sind:

* Presentation Layer
* Business Layer
* Data Access Layer
* Data Layer

Diese werden in Unterkapiteln erläutert.

### Presentation Layer

Im Presentation Layer geht es um die Darstellung von Inhalt. Der Layer interagiert mit dem Business Layer, von dem es seine Daten erhält.

Das MVC Design Pattern wird innerhalb des Layers angewendet. Die Daten, welche von dem Business Layer geholt werden, werden in Models (ViewModels) abgefüllt, mit denen dynamisch HTML erzeugt werden. Das erzeugte HTML wird mit weiteren, statischen Ressourcen dem Client zugesendet.  
Das Layer ist somit zuständig für die Präsentation des Inhaltes.

### Business Layer

Im Business Layer wird Funktionalität definiert, welche genutzt werden kann. Der Layer interagiert mit dem Presentation und dem Data Access Layer.

**[Beispiele in CodeSnipper]**

### Data Access Layer

Der Data Access Layer vereinfacht den Zugriff auf den Data Layer. Der Layer interagiert mit dem Data Layer und dem Business Layer.

Hierbei spielt in unserem Projekt Entity Framework eine grosse Rolle. Es regelt den Zugriff auf die Datenbank.  
Zusätzlich gibt es ein C#-Projekt, indem die Models liegen und die Verbindung mit der Datenbank definiert ist.

### Data Layer

Im Data Layer geht es hauptsächlich um abgespeicherte Daten. Genauer gesagt ist dies die Datenbank. Die Datenbank, welche CodeSnipper benutzt ist eine MSSQL-Datenbank.

CodeSnipper speichert die Snippets und die Benutzer persistent in die Datenbank.

## Datenbank

In der Datenbank sind die Daten abgespeichert, welche CodeSnipper benötigt. Sie ist eine MSSQL-Datenbank.  
CodeSnipper speichert die eingetragenen CodeSnippets und die erstellten Benutzer in die Datenbank ein.

Die Daten werden mithilfe des Entity Frameworks ausgelesen und verarbeitet.

## Sessions

CodeSnipper verwendet Sessions zur Anmeldung von Benutzern. Damit diese Session nicht sofort verfällt, werden Cookies benutzt.  
Für das Anmelden von Benutzern wird das Standard-Login von ASP.NET verwendet. Da dies bereits sicher implementiert wurde und es in unserem Beispiel keinen Sinn ergibt, etwas Eigenes zu machen.

## Transaktionen

Gibt es hier etwas zu erwähnen?

## SSL/TLS

Gibt es hier etwas zu erwähnen?

# Arbeitsjournal

Das Arbeitsjournal wurde in Workjournal.md geführt. Zu dieser Datei steht auch eine PDF-Version zur Verfügung.

# Sicherheit

Um die Datensicherheit zu steigern, benutzen wir Entity Framework mit LINQ. So kann beispielsweise SQL-Injection vermieden werden.