20. November 2021

Scherzer Tobias

HTL ZWETTL

Zwettl, Nö

Skript – Technologien Backend

Diplomarbeit

Inhalt

[Modelle in JSON POSTen 2](#_Toc95287303)

[Andere Variante zum POSTen 3](#_Toc95287304)

[Funktionaler POST 4](#_Toc95287305)

[Extras 4](#_Toc95287306)

[Nugets 4](#_Toc95287307)

[Modell für Produkte 5](#_Toc95287308)

[Interfaces 6](#_Toc95287309)

[IConfigId 6](#_Toc95287310)

[IDescribable 6](#_Toc95287311)

[IImageable 6](#_Toc95287312)

[IIndexable 7](#_Toc95287313)

[INameable 7](#_Toc95287314)

[Enumerators 7](#_Toc95287315)

[EStatus 7](#_Toc95287316)

[EValueMode 7](#_Toc95287317)

[Wrapper 7](#_Toc95287318)

[ConfigurationDeleteWrapper 7](#_Toc95287319)

[ConfiguratorPost 7](#_Toc95287320)

[SavedConfigWrapper 7](#_Toc95287321)

[SavedNameWrapper 8](#_Toc95287322)

[Modell 8](#_Toc95287323)

[Account 8](#_Toc95287324)

[Configurator 8](#_Toc95287325)

[ConfiguratorBase 8](#_Toc95287326)

[ConfiguratorSlim 8](#_Toc95287327)

[ConfiguredProduct 8](#_Toc95287328)

[ModelType 9](#_Toc95287329)

[Option 9](#_Toc95287330)

[OptionGroup 9](#_Toc95287331)

[OptionGroupExtended 9](#_Toc95287332)

[OptionSection 9](#_Toc95287333)

[ProductSave 10](#_Toc95287334)

[ProductSaveExtended 10](#_Toc95287335)

[ProductSaveSlim 10](#_Toc95287336)

[Rules 10](#_Toc95287337)

[RulesExtended 10](#_Toc95287338)

[Emails senden 11](#_Toc95287339)

[Schritt 1 (Installation): 11](#_Toc95287340)

[Installation 11](#_Toc95287341)

[Schritt 2 (Aufbau): 11](#_Toc95287342)

[Emailaufbau + Senden 11](#_Toc95287343)

[Empfangene Email 12](#_Toc95287344)

[Schritt 3 (Aufbereitung): 12](#_Toc95287345)

[Projektdatei bearbeiten 12](#_Toc95287346)

[Email mit Aufbereitung 13](#_Toc95287347)

[PDF generieren 14](#_Toc95287348)

[Schritt 1 (Installieren): 14](#_Toc95287349)

[Schritt 2 (Ausgabe): 14](#_Toc95287350)

[JSON Web Token (JWT) 14](#_Toc95287351)

[Schritt 1 (Installieren): 14](#_Toc95287352)

[Schritt 2 (Implementieren): 15](#_Toc95287353)

[Schritt 3 (Ausführen): 15](#_Toc95287354)

# Modelle in JSON POSTen

<https://andrewlock.net/model-binding-json-posts-in-asp-net-core/>

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# Andere Variante zum POSTen

<https://www.stevejgordon.co.uk/sending-and-receiving-json-using-httpclient-with-system-net-http-json>

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEin Bild, das Text enthält.

Automatisch generierte Beschreibung

# Funktionaler POST

Ein Bild, das Text, drinnen, Bildschirm, schließen enthält.

Automatisch generierte Beschreibung





## Extras

### Nugets

Newtonsoft.JSON installieren in Nugets

<https://www.educative.io/edpresso/how-to-create-a-json-string-in-c-sharp>

# Modell für Produkte

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

## Interfaces

### IConfigId

namespace Model.Interfaces

{

public interface IConfigId

{

public string ConfigId { get; set; }

}

}

### IDescribable

namespace Model.Interfaces

{

public interface IDescribable

{

string Description { get; set; }

}

}

### IImageable

namespace Model.Interfaces

{

public interface IImageable

{

List<string> Images { get; set; }

}

}

### IIndexable

namespace Model.Interfaces

{

public interface IIndexable

{

string Id { get; set; }

}

}

### INameable

namespace Model.Interfaces

{

public interface INameable

{

string Name { get; set; }

}

}

## Enumerators

### EStatus

namespace Model.Enumerators

{

public enum EStatus

{

ordered = 0, saved

}

}

### EValueMode

namespace Model.Enumerators

{

public enum EValueMode

{

TestValues, DatabaseValues

}

}

## Wrapper

### ConfigurationDeleteWrapper

namespace Model.Wrapper

{

public class ConfigurationDeleteWrapper : IConfigId

{

public string ConfigId { get; set; }

}

}

### ConfiguratorPost

namespace Model

{

public class ConfiguratorPost : ConfiguratorBase

{

public Rules Rules { get; set; }

public List<OptionGroupExtended> OptionGroups { get; set; }

}

}

### SavedConfigWrapper

namespace Model.Wrapper

{

public class SavedConfigWrapper : IConfigId

{

public string ConfigId { get; set; }

public string SavedName { get; set; }

}

}

### SavedNameWrapper

namespace Model.Wrapper

{

public class SavedNameWrapper

{

public string SavedName { get; set; }

}

}

## Modell

### Account

namespace Model

{

public class Account

{

public string UserName { get; set; }

public string UserEmail { get; set; }

public bool IsSameUser(Account temp)

{

if(this.UserName == temp.UserName)

if(this.UserEmail == temp.UserEmail)

return true;

return false;

}

}

}

### Configurator

namespace Model

{

public class Configurator : ConfiguratorBase

{

public RulesExtended Rules { get; set; }

public List<OptionGroup> OptionGroups { get; set; }

}

}

### ConfiguratorBase

namespace Model

{

public class ConfiguratorBase : ConfiguratorSlim

{

public List<OptionSection> OptionSections { get; set; }

public List<Option> Options { get; set; }

}

}

### ConfiguratorSlim

namespace Model

{

public class ConfiguratorSlim : IConfigId, INameable, IDescribable, IImageable

{

public string ConfigId { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public List<string> Images { get; set; }

}

}

### ConfiguredProduct

namespace Model

{

public class ConfiguredProduct

{

public string ConfigurationName { get; set; } = "";

public List<string> Options { get; set; } = new List<string>();

public float Price { get; set; }

public string Model { get; set; }

}

}

### ModelType

namespace Model

{

public class ModelType : INameable, IDescribable

{

public string Name { get; set; }

public string Description { get; set; }

public List<string> Options { get; set; } = new List<string>();

}

}

### Option

namespace Model

{

public class Option : IIndexable, INameable, IDescribable

{

public string Id { get; set; }

public string Name { get; set; }

public string Description { get; set; }

}

}

### OptionGroup

namespace Model

{

public class OptionGroup : IIndexable, INameable, IDescribable

{

public string Id { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public List<string> OptionIds { get; set; } = new List<string>();

public bool Required { get; set; }

}

}

### OptionGroupExtended

namespace Model

{

public class OptionGroupExtended : OptionGroup

{

public bool Replacement { get; set; }

}

}

### OptionSection

namespace Model

{

public class OptionSection : IIndexable, INameable

{

public string Id { get; set; }

public string Name { get; set; }

public List<string> OptionGroupIds { get; set; } = new List<string>();

}

}

### ProductSave

namespace Model

{

public class ProductSave : ProductSaveSlim, INameable, IDescribable, IConfigId

{

public string ConfigId { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public string Status { get; set; }

public DateTime Date { get; set; }

}

}

### ProductSaveExtended

namespace Model

{

public class ProductSaveExtended : ProductSave

{

public Account User { get; set; }

}

}

### ProductSaveSlim

namespace Model

{

public class ProductSaveSlim

{

public string SavedName { get; set; }

public List<string> Options { get; set; } = new List<string>();

}

}

### Rules

namespace Model

{

public class Rules

{

public float BasePrice { get; set; }

public string DefaultModel { get; set; }

public List<ModelType> Models { get; set; } = new List<ModelType>();

public Dictionary<string, List<string>> Requirements { get; set; } = new Dictionary<string, List<string>>();

public Dictionary<string, List<string>> Incompatibilities { get; set; } = new Dictionary<string, List<string>>();

public Dictionary<string, List<string>> GroupRequirements { get; set; } = new Dictionary<string, List<string>>();

public Dictionary<string, float> PriceList { get; set; } = new Dictionary<string, float>();

}

}

### RulesExtended

namespace Model

{

public class RulesExtended : Rules

{

public Dictionary<string, List<string>> ReplacementGroups { get; set; } = new Dictionary<string, List<string>>();

}

}

# Emails senden

(Quelle P0015K)

## Schritt 1 (Installation):

### Installation

Ein Bild, das Text, Screenshot, Bildschirm, schwarz enthält.

Automatisch generierte Beschreibung

## Schritt 2 (Aufbau):

### Emailaufbau + Senden

Ein Bild, das Text, Screenshot, Bildschirm enthält.

Automatisch generierte Beschreibung

### Empfangene Email

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

## Schritt 3 (Aufbereitung):

### Projektdatei bearbeiten

Ein Bild, das Text, Monitor, Screenshot, Bildschirm enthält.

Automatisch generierte Beschreibung

(PreserveCompilationContext)

### Email mit Aufbereitung

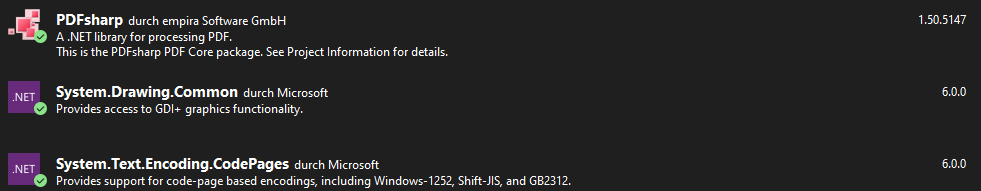
Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEin Bild, das Text enthält.

Automatisch generierte Beschreibung

# PDF generieren

## Schritt 1 (Installieren):



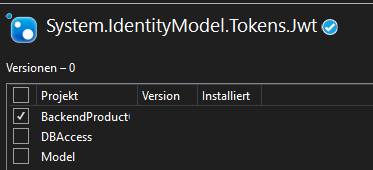
## Schritt 2 (Ausgabe):

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# JSON Web Token (JWT)

## Schritt 1 (Installieren):



## Schritt 2 (Implementieren):

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

P0019K

## Schritt 3 (Ausführen):

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Sollte im GET von den gespeicherten Produkten passieren