20. November 2021

Scherzer Tobias

HTL ZWETTL

Zwettl, Nö

Skript – Technologien Backend

Diplomarbeit

Inhalt

[Modelle in JSON POSTen 2](#_Toc95221017)

[Andere Variante zum POSTen 3](#_Toc95221018)

[Funktionaler POST 4](#_Toc95221019)

[Extras 4](#_Toc95221020)

[Nugets 4](#_Toc95221021)

[Model für Produkte 5](#_Toc95221022)

[Interfaces 5](#_Toc95221023)

[IConfigId 5](#_Toc95221024)

[IDescribable 5](#_Toc95221025)

[IImageable 5](#_Toc95221026)

[IIndexable 6](#_Toc95221027)

[INameable 6](#_Toc95221028)

[Enumerators 6](#_Toc95221029)

[EStatus 6](#_Toc95221030)

[EValueMode 6](#_Toc95221031)

[Wrapper 6](#_Toc95221032)

[ConfigurationDeleteWrapper 6](#_Toc95221033)

[ConfiguratorPost 6](#_Toc95221034)

[SavedConfigWrapper 6](#_Toc95221035)

[SavedNameWrapper 7](#_Toc95221036)

[Modell 7](#_Toc95221037)

[Account 7](#_Toc95221038)

[Configurator 7](#_Toc95221039)

[ConfiguratorBase 7](#_Toc95221040)

[ConfiguratorSlim 7](#_Toc95221041)

[ConfiguredProduct 8](#_Toc95221042)

[ModelType 8](#_Toc95221043)

[Option 8](#_Toc95221044)

[OptionGroup 8](#_Toc95221045)

[OptionGroupExtended 8](#_Toc95221046)

[OptionSection 8](#_Toc95221047)

[ProductSave 9](#_Toc95221048)

[ProductSaveExtended 9](#_Toc95221049)

[ProductSaveSlim 9](#_Toc95221050)

[Rules 9](#_Toc95221051)

[RulesExtended 9](#_Toc95221052)

[Emails senden 10](#_Toc95221053)

[Schritt 1 (Installation): 10](#_Toc95221054)

[Installation 10](#_Toc95221055)

[Schritt 2 (Aufbau): 10](#_Toc95221056)

[Emailaufbau + Senden 10](#_Toc95221057)

[Empfangene Email 11](#_Toc95221058)

[Schritt 3 (Aufbereitung): 11](#_Toc95221059)

[Projektdatei bearbeiten 11](#_Toc95221060)

[Email mit Aufbereitung 12](#_Toc95221061)

[PDF generieren 13](#_Toc95221062)

[Schritt 1 (Installieren): 13](#_Toc95221063)

[Schritt 2 (Ausgabe): 13](#_Toc95221064)

[JSON Web Token (JWT) 13](#_Toc95221065)

[Schritt 1 (Installieren): 13](#_Toc95221066)

[Schritt 2 (Implementieren): 14](#_Toc95221067)

[Schritt 3 (Ausführen): 14](#_Toc95221068)

# 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>

# Model 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

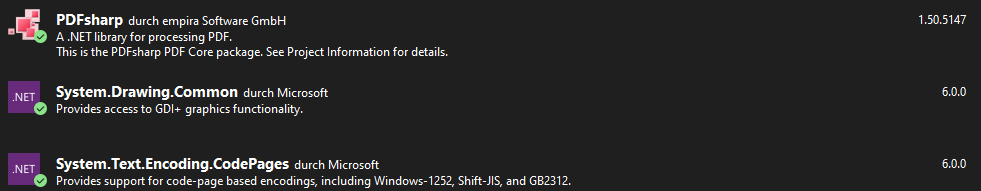
Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEin Bild, das Text enthält.

Automatisch generierte Beschreibung

# PDF generieren

## Schritt 1 (Installieren):



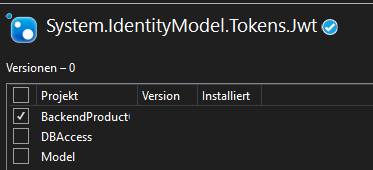
## Schritt 2 (Ausgabe):

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