**ASP.NET MVC Core Grundlagen**

**Teil 1: Theoretische Einführung**

**1.1 Was bedeutet MVC?**

**MVC steht für Model–View–Controller** und ist ein Architekturpattern zur Trennung von Verantwortlichkeiten in einer Anwendung.

| **Komponente** | **Aufgabe** | **Beispiel** |
| --- | --- | --- |
| **Model** | Stellt die Daten und Geschäftslogik dar | class Produkt { int Id; string Name; } |
| **View** | Präsentation der Daten (HTML-Seite) | .cshtml-Datei mit Razor-Syntax |
| **Controller** | Vermittelt zwischen Model und View, reagiert auf HTTP-Requests | public class ProduktController : Controller { ... } |

Ziel: **Trennung von Logik, Darstellung und Daten**  
→ Erleichtert Wartung, Testbarkeit und Wiederverwendbarkeit.

**Ablauf eines Requests in MVC**

1. Ein **HTTP-Request** trifft auf den Server (z. B. /Produkt/Details/5).
2. Das **Routing-System** sucht den passenden **Controller** (ProduktController) und die passende **Action-Methode** (Details(int id)).
3. Die Action ruft evtl. Daten aus einem **Model** oder der Datenbank ab.
4. Die Action gibt ein **ViewResult** zurück → Razor-View (Views/Produkt/Details.cshtml) wird gerendert.
5. Der **HTTP-Response** mit der gerenderten HTML-Seite wird an den Browser zurückgesendet.

**1.3 Aufbau eines ASP.NET Core MVC-Projekts**

Typischer Projektbaum:

Projektname/

├── Controllers/

│ └── HomeController.cs

├── Models/

│ └── Produkt.cs

├── Views/

│ ├── Home/

│ │ └── Index.cshtml

│ └── Shared/

│ └── \_Layout.cshtml

├── wwwroot/

│ ├── css/

│ ├── js/

│ └── images/

├── Program.cs

├── appsettings.json

Wichtige Dateien:

* **Program.cs:** Startpunkt der Anwendung (Konfiguration, Services)
* **Views/\_Layout.cshtml:** Master-Layout (gemeinsames Design)
* **appsettings.json:** Konfiguration (z. B. DB-Verbindung)
* **wwwroot:** Statische Dateien (CSS, JS, Bilder)

**Teil 2: Praktischer Einstieg**

**2.1 Neues MVC-Projekt erstellen**

In Visual Studio:

**Datei → Neues Projekt → ASP.NET Core Web App (Model-View-Controller)**

* Framework: .NET 8.0 (oder aktueller)
* Authentifizierung: „Keine“

**Der erste Controller**

using Microsoft.AspNetCore.Mvc;

namespace MeineMvcApp.Controllers

{

public class ProduktController : Controller

{

public IActionResult Index()

{

return View();

}

}

}

ASP.NET MVC sucht automatisch nach einer View mit dem Namen **Views/Produkt/Index.cshtml**.

**2.3 Die erste View (Razor)**

**Views/Produkt/Index.cshtml**

@{

ViewData["Title"] = "Produktübersicht";

}

<h1>@ViewData["Title"]</h1>

<p>Willkommen in der Produktverwaltung!</p>

**2.4 Model erstellen**

namespace MeineMvcApp.Models

{

public class Produkt

{

public int Id { get; set; }

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

public decimal Preis { get; set; }

}

}

**2.5 Daten an die View übergeben**

public IActionResult Index()

{

var produkte = new List<Produkt>

{

new() { Id = 1, Name = "Apfel", Preis = 0.80m },

new() { Id = 2, Name = "Banane", Preis = 1.20m }

};

return View(produkte);

}

**View: Index.cshtml**

@model IEnumerable<MeineMvcApp.Models.Produkt>

<h2>Produkte</h2>

<table class="table">

<thead>

<tr>

<th>ID</th>

<th>Name</th>

<th>Preis</th>

</tr>

</thead>

<tbody>

@foreach (var p in Model)

{

<tr>

<td>@p.Id</td>

<td>@p.Name</td>

<td>@p.Preis.ToString("C")</td>

</tr>

}

</tbody>

</table>

**Teil 3: Routing und Parameter**

**3.1 Action mit Parameter**

public IActionResult Details(int id)

{

var produkt = new Produkt { Id = id, Name = "Testprodukt", Preis = 9.99m };

return View(produkt);

}

**View: Details.cshtml**

@model MeineMvcApp.Models.Produkt

<h2>Details</h2>

<p>ID: @Model.Id</p>

<p>Name: @Model.Name</p>

<p>Preis: @Model.Preis €</p>

📍 URL: /Produkt/Details/5 → zeigt Produkt mit ID 5.

**Teil 4: Layout und Navigation**

**4.1 Gemeinsames Layout (\_Layout.cshtml)**

Views/Shared/\_Layout.cshtml enthält die Grundstruktur aller Seiten (z. B. Menü, Footer).

Ein Beispiel:

<!DOCTYPE html>

<html>

<head>

<title>@ViewData["Title"] - MeineMvcApp</title>

<link rel="stylesheet" href="~/css/site.css" />

</head>

<body>

<header>

<nav>

<a href="/">Start</a> |

<a href="/Produkt">Produkte</a>

</nav>

</header>

<main>

@RenderBody()

</main>

<footer>&copy; @DateTime.Now.Year - MeineMvcApp</footer>

</body>

</html>

**🧠 Teil 5: Vertiefung und Übung**

**Übung 1: Controller und Views**

Erstellen Sie einen neuen Controller **KundeController** mit einer Action Index(), die eine Liste von Kunden an die View übergibt.  
Zeigen Sie die Kunden in einer Tabelle an.

**Übung 2: Detailseite**

Fügen Sie eine Action Details(int id) hinzu, die einen einzelnen Kunden anzeigt.

**Übung 3: Navigation**

Ergänzen Sie im Layout-Menü einen Link zu /Kunde.

**💬 Teil 6: Erweiterung (optional)**

* **Formulare und Model Binding:** Eingabeformulare mit HttpPost-Action (z. B. neues Produkt anlegen)
* **Validierung mit DataAnnotations**
* **Entity Framework Core** zur Datenanbindung
* **Dependency Injection** für Services oder Repositories

**Fazit**

ASP.NET Core MVC trennt klar zwischen:

* **Model (Daten)**,
* **View (Darstellung)**,
* **Controller (Logik und Ablauf)**.

Diese Struktur erlaubt **saubere, skalierbare Webanwendungen** – und ist die Basis vieler moderner .NET-Projekte.