**Uputstvo za upotrebu .NET Core**

# Šta je .NET Core?

* Modularni, open-source framework razvijen od strane Microsofta.
* Cross-platform
* Omogućava izradu različitih vrsta aplikacija (Web, Desktop, Cloud services etc.) i neke od njih su:
  + ASP.NET Core (razvoj WEB aplikacija)
  + Blazor (razvoj modernih WEB aplikacija koristeći C# i .NET umesto JS-a)
  + Xamarin (razvoj mobilnih aplikacija)
  + WPF i WinForms (razvoj desktop aplikacija)

# Osnovni koncepti

* Solution – Kontejner koji sadrži više .NET projekata
* C# - Glavni jezik za .Net
* .NET – Framework koji pruža zajedničku infrastrukturu za razvoj aplikacija
* NuGet Packages – upravljanje paketima (bibliotekama)
* CLI (Command-Line Interface) – Naredbe preko komandnih linija (dotnet new, dotnet build...)

# Razvoj aplikacije

* Dotnet new – kreiranje novog projekta
* Dotnet build – izgradnja projekta
* Dotnet run – pokretanje aplikacije
* ORM (Object-Relational Mapping) – framework za rad sa bazama podataka
* ASP.NET Core:
  + Middleware, MVC Pattern etc.

**.NET Core WEB API**

# Osnovna struktura

* Projekat
  + .NET Core WEB API je organizovana u jednom projekatu
  + Sadrži sve potrebne fajlove i direktorijume za razvoj i izgradnju aplikacije

A screenshot of a computer

Description automatically generated

* Kontroleri
  + Odgovorni za obradu HTTP zahteva
  + Svaki kontroler sadrži akcije koje se pozivaju kada se primi određeni tip HTTP zahteva
  + Uglavnom se nalazi u Console Application – API
* Modeli
  + Predstavljaju podatke koji se koriste u aplikaciji
  + Najčešće su klase koje predstavljaju entitete u bazi podataka ili DTO (Data Transfer Object) klase
  + Uglavnom se nalaze u zasebnom ClassLibrary-u
* Servisi
  + Sadrže poslovnu logiku koja se koristi u kontrolerima
  + Ovde se vrši obrada podataka, komunikacija sa bazom podataka ili drugim servisima
  + Uglavnom se nalaze u zasebnom ClassLibrary-u
* Appsettings.json
  + Konfiguracioni fajl gde se mogu postaviti različite konfiguracije aplikacije, kao što su veze sa bazom podataka, ključeni i drugi parametri
* DataAccessLayer
  + Preko DataAccessLayera vršimo komunikaciju sa bazom podataka
  + Uglavnom se nalazi u zasebnom ClassLibrary-u

# Osnovni pojmovi

* HTTP Zahtevi – Aplikacija komunicira sa svetom putem HTTP zahteva (GET, POST, PUT, DELETE)
* Routing - Određuju kako će aplikacija odgovarati na različite HTTP zahteve
* Dependency Injection (DI):
  + Omogućava ubacivanje (inject) zavisnosti u kontrolere i servise
  + Pomaže u smanjenju zavisnosti i čini kod testabilnijim
* Middleware:
  + Komponente koje se izvršavaju tokom obrade HTTP zahteva
  + Dodaju funkcionalnosti poput logovanja, autentikacija, itd..
* Model Binding – Automatski mapira podatke iz HTTP zahteva na parametre metoda kontrolera