# Erfan

# DataHandlingLibrary

Data Handling Library är ett class library-projekt som innehåller all logik för att hantera data

## File Post

* Inom **Post-namespace** finns all kod som krävs för att deserialisera JSON-data till c# object
* **PostJsonDeserialize** är en klass för att deserialisera en lista av JSON-inlägg.
* **Post** är en klass för att deserialisera endast en post

## File Comment

* Inom **CommentNamespace** finns all kod som krävs för att deserialisera JSON-data till c# object
* **CommentsJsonDeserialize**är en klass för att deserialisera en lista av JSON-inlägg.
* **Comment** är en klass för att deserialisera endast en comment
* **CommentsUsers** är en klass som används för att identifiera användaren för kommentarerna.

## File Product

* Inom **Product-namespace** finns all kod som krävs för att deserialisera JSON-data till c# object
* **ProductsJsonDeserialize**är en klass för att deserialisera en lista av JSON-inlägg.
* **Product** är en klass för att deserialisera endast en Product

## File User

* Inom **UserNamespace** finns all kod som krävs för att deserialisera JSON-data till c# object
* **UsersJsonDeserialize** är en klass för att deserialisera en lista av JSON-inlägg.
* **User** är en klass för att deserialisera endast en User

# CacheProvider

Det är en klass som ärver från ICacheProvider (interface). I ICacheProvider finns det funktioner som fungerar som hjälpmetoder för att spara data som erhållits från en endPoint i cache-minnet. Syftet med att använda cache-minnet är att programmet inte behöver hämta data från API varje gång; istället kan det använda data från cache-minnet, vilket resulterar i betydligt snabbare åtkomst till informationen.

# EndPoinDeserializer

Syftet med den här klassen är att deserialisera en 'AllEndPoints.json'-fil, vilket möjliggör åtkomst till olika endpoints.

# GetEndPoints

Denna klass öppnar filen AllEndPoints.json och deserialiserar datan som finns där med hjälp av klassen EndPointDeserializer. Syftet med den här klassen är att tillhandahålla funktioner som underlättar skapandet av olika endpoints. Alla endpoints kommer från en enda plats, vilket gör det möjligt att enkelt ändra dem senare om så önskas.

Några av de funktioner som klassen innehåller är utformade för att göra det lättare att skapa endpoints med olika parametrar, inklusive:

* selectData: specifierar vilka dataattribut som ska inkluderas i resultatuppsättningen.
* id: identifierar användaren, kommentaren eller inlägget.
* limit: sätter det maximala antalet objekt att hämta eller behandla.
* skip: bestämmer startpunkten genom att hoppa över ett angivet antal objekt.

Genom att tillhandahålla dessa funktioner gör klassen det enkelt att dynamiskt skapa endpoints med olika parametrar och behov

# ResponseHandler

I den här klassen finns det funktioner för att utföra GET- (för att hämta data) och PATCH- (för att uppdatera en del av datat) förfrågningar, både asynkront och synkront. Båda metoderna stöder att ange en URL eller selectData. Dessutom finns det en funktion som heter DeserializeJsonData, som gör det möjligt att deserialisera olika typer av JSON-data beroende på vilken typ (klass) som anges till funktionerna.Funktionerna är utformade för att vara användarvänliga och ge flexibilitet genom att stödja både asynkrona och synkrona anrop, samt möjligheten att hantera olika JSON-strukturer genom deserialisering.

# Responses

Den här klassen ärver från ResponseHandler.

I den här klassen finns det funktioner för att utföra Get förfrågningar med den skillnaden att de här funktionen (använder funktioner av ResponseHandler) skickar get requst som innehåller specificerar vilka dataattribu och deserialiserar de och lämnar tillbaka , c# object

E-TrackPro

# CommentCell

Visar kommentarer , Informationen hämtas från API , *titel , kropp , bild*

# ListItemCell

Visar en Produkter , Informationen hämtas från API , *titel , pris, bild, betyg, barnd*

# ListPostComentCell

Visar en post från en användare samt de olika kommentarer som andra användare har skrivit under posten. Informationen hämtas från API och inkluderar *titel, kropp, bild och taggar.*

# ListUserCell

Visar anävndre som Informationen hämtas från API ,Användarnamn, *namen , efternamen , ålder* , sex , bild

# AddCommentPage

Används för att användare kan skriva kommentarer under produkter Genom att ange ett random Id , och kommenterar

# PriceMonitoringPage

På den här sidan finns det diagram som visar hur olika kategorier har förändrat pris över tid. Det finns också ett annat diagram som visar lagerpris. Användaren kan välja olika kategorier genom en ComboBox.

# ProductsReviewPage

Visar olika produkter med hjälp av **ListItemCell**.

# SocialMediaDashboardPage

Visar olika post och och kommentarer som hör till dem.

# UpdatePage

På denna sida kan användaren uppdatera olika egenskaper för användare genom att ange deras ID. Efter att ha skickat en PATCH-förfrågan får användaren ett meddelande om det var möjligt att uppdatera (värdet ändras inte alltid)