**MARKET WATCH**

Classes: -

***1.Stock***

Properties: -

a). stock Id

b). stock Name

c). starting Price

d). price Difference

***2.WishList***

Properties: -

a). Wishlist id

b). List<Stock>

Methods: -

1.AddStockToWishList (userid, stockId).

-when user clicks on add button from the main grid, this method will add that stock to user’s wish list based on user id and stock id.

Microservices: -

External Service – For Interaction with external API

API: -

**1.GetStartingPrices ()**

-Interact with external API and get all the stocks with their names and opening prices.

-Store this data in SQL DB/IN MEMORY DB, so that we can retrieve this data later.

-We can schedule this API call for every morning.

**2.GetCurrentPrice (**stockId**)**

-Interact with external API and get Increment/Decrement for every stock in wish list.

Market Service – For UI Interaction

API: -

**1.GetAllStocks ().**

-This method will fetch all the stocks from DB with their starting prices to show on the main grid.

**2.UpdateWishList (stockId, userId).**

-Once the user selects any stock from the main grid, this method will add that stock to Wishlist and to corresponding table in DB.

**3.RefreshWishList(userId)**

**-**get all the stocks from the wish list.

-for every stock in the wish list we can call GetCurrenPrice(stockId) to fetch the current price.

-we need to schedule this call to happen at some frequency in every 1 minute or external API should give some indication every time a transaction happens on stock

Data Base Schema: -

Tables Required: -

**User** (to store different users) 🡪 userId, username, password.

**Stock** (to store different stocks) 🡪 stock id, stock name, starting price.

**UserWishlist** (to store different Wishlist from different users) 🡪 user id, wish list id.

**Wishlist** (to store stock and Wishlist relationship, a Wishlist can have many stocks and a stock can belong to many Wishlist) 🡪 Wishlist Id, stockId (Many to Many Mappings table)

Technologies can be used: -

1. UI: - .Net Core MVC (5+) or Angular (8+)
2. API: - .Net Core 5+
3. DB: - SQL or IN MEMORY
4. LINQ – To interact with DB.
5. EF Core – To interact with DB.

Tools can be used: -

1. VS 2022 – To develop and unit test APIs.
2. VS code 2022 – To develop and unit test UI.
3. SSMS / In Memory – To maintain DB.