# General description

MyStocs.com is an online application for tracking stock market portfolio, built using AWS Amplify Gen 2. It has 3 main entities: Stocks, Transactions, Wallets.

# Stocks

* Stocks represent equities like stocks, ETF or crypto.
* The user manages the stocks in the Portfolio page (app/(authed)/stocks-listing/page.tsx).
* Each stock has the following fields, stored in model PortfolioStock:
  + **symbol**. The Yahoo Finance symbol of the stock. Provided by the user.
  + **stockType**. The stock type. Provided by the user from the options in stockTypeEnum.
  + **region**. The region where the stock is from. Provided by the user from the options in regionEnum.
  + **pdp.** Price Drop Percentage (PDP). Used to calculate target prices for buying (LBD) or selling (TP).
  + **plr.** Profit Loss Ratio (PLR). Used to calculate the target price for selling (TP). Provided by the user.
  + **swingHoldRatio**. Swing-Hold Ratio (SHR). A number between 0 and 100. Used to calculate the investment allocation for the Swing or Hold strategies. Provided by the user.
  + **budget.** Used for reporting. Provided by the user.
* The stocks can be added/edited (app/(authed)/add-stocks/page.tsx) or hidden app/(authed)/stocks-listing/page.tsx. When hidden, they are excluded from any calculations or reporting.
* The latest price for the stocks can be pulled via API (amplify/functions/getYfinanceData/handler.ts) from Yahoo Finance by the user clicking the “Fetch Prices” button in the nav bar (app/components/NavBar.tsx).

# Transactions

* The user performs Buy and Sell transactions on a stock, in the Wallets page (app/(authed)/wallets/[stockId]/page.tsx).
* A Buy transaction has the following fields, stored in model Transaction:
  + **date.** The date of the transaction. Provided by the user.
  + **action.** This is set to “Buy” by default. Pulled from the options in txnActionEnum.
  + **txnType**. Transaction type. Decides which type of wallets will be created by the Buy transaction. Options are a “Swing” wallet, a “Hold” wallet or both if “Split” is selected. Provided by the user.
  + **signal**. Label indicating the reason for the Buy. Provided by the user from the options in txnSignalEnum.
  + **price**. The price at which the stock is being bought. Provided by the user.
  + **investment**. The investment for the transaction. Provided by the user.
* The Buy transaction can be added via a modal (app/components/TransactionForm.tsx) by clicking the “Add Transaction” button in the Wallets page.
* When a Buy transaction is created, the application calculates and stores the following values in the Transactions model:
  + **holdShares.** The number of Hold shares is calculated based on the value of tnxType. If tnxType is “Hold”, holdShares = investment / price. If tnxType is “Split”, then holdShares = (investment \* (1 - (swingHoldRatio/100))) / price. If tnxType is “Swing”, holdShares = 0. The logic is in app/components/TransactionForm.tsx.
  + **swingShares.** The number of Swing shares is calculated based on the value of tnxType. If tnxType is “Swing”, then swingShares = investment / price. If tnxType is “Split”, then swingShares = (investment \* (swingHoldRatio/100)) / price. If tnxType is “Hold”, then swingShares = 0. The logic is in app/components/TransactionForm.tsx.
  + **lbd.** Last Buy Drop (LBD). Represents the stock price at which the user might want to make another buy, because the current price has dropped by a percentage equal to PDP, from the price at which the stock was last bought. lbd = price - (price \* (pdp / 100)). E.g.: Buy Price = $100 and PDP = 5%, then LBD = $95. The logic is in app/components/TransactionForm.tsx.
  + **quantity.** Quantity. Calculated as Investment / Buy Price. E.g.: Buy Price = $100, Investment = $1,000, then Quantity = 10. The logic is in app/components/TransactionForm.tsx.
  + **tp.** Take Profit (TP). The price at which the user might want to sell the stock, for a profit. tp = price + (price \* (pdp \* plr / 100)). E.g: Buy Price = $100, PDP = 5, PLR = 2, then TP = $110. The logic is in app/components/TransactionForm.tsx.
* The transaction is then displayed in the Wallets page, in the Transactions table..
* The transaction can be deleted or edited with certain restrictions.
* Sell transactions can be performed from a Wallet, as explained in the Wallets section below.
* A Sell transaction has the following fields, stored in model Transaction:
  + **date.** The date of the transaction. Provided by the user.
  + **price**. The price at which the stock is being sold. Provided by the user.
  + **quantity**. The quantity of shares sold. Provided by the user.
  + **signal**. Label indicating the reason for the Sell. Provided by the user from the options in txnSignalEnum.
* The Sell transaction can be entered via a modal (app/components/TransactionForm.tsx) by clicking the “Sell from Wallet” icon that each wallet has.
* When a Sell transaction is created, the application calculates and stores the following values in model Transaction:
  + **action**. This is set to “Sell” by default. Pulled from the options in txnActionEnum.
  + **txnType**. A Sell can be “Swing” or “Hold”, depending on the wallet type where the Sell has been made from.
  + **txnProfit**. Profit / Loss. txnProfit = (sellPrice - buyPrice) \* quantity. The logic is in app/utils/financialCalculations.ts.
  + **txnProfitPercent**. Profit / Loss, as percentage. The logic is in app/utils/financialCalculations.ts.
  + **completedTxnId**. The Id of the wallet where the Sell is originating.

# Wallets

* Wallets are “wallets” that are unique by price, by only holding shares that have the same price (app/(authed)/wallets/[stockId]/page.tsx). Everytime we add a Buy transaction, it checks if a previous wallet at the Buy price exists. If it does, then it adds the new shares to it. If it doesn’t exist, then it creates a new wallet for that Buy price. E.g: if we buy stock at $10, and there are no wallets for that stock, a new $10 wallet is created. If we then buy stock at $12, a new $12 wallet will be created. If we buy stock again at $10, then the new shares will be added to the existing $10 wallet.
* A stock can have only wallets type Swing, only Hold or both Swing and Hold. The stock can have an unlimited number of wallets.
* Wallets can be sold from, by creating a Sell transaction, as long as they have shares left.
* Wallets can only be Swing or Hold, decided by the Transaction type that has created the wallet (app/components/TransactionForm.tsx). The Wallets table has 2 tabs, one for all the Swing wallets and the other one for all the Hold wallets.
* The idea behind “Swing” and “Hold” wallets is that the user wants to manage their stocks based on two distinct strategies. Swing stocks are to be sold within days or weeks, when the stock hits Take Profit (TP), or anytime the user wants. Hold stocks are to be held long term until the user decides to sell them.
* When a stock is defined, it has the swingHoldRatio field which defines how the investment should be allocated between “Swing” and “Hold”, in case the user decides to pursue both strategies. That is done by the user selecting txnType “Split” in the Buy modal.
* When adding a Buy, the user can choose the Buy to be “Swing” only, “Hold” only, or “Split” (auto).
  + If “Split” is chosen, then the system will automatically calculate the Swing / Hold wallets based on swingHoldRatio. E.g: swingHoldRatio = 60% that means that 60% of the investment will be Swing, and 40% Hold.
  + If the user selects either “Swing” or “Hold”, then the entire investment will go towards a Swing wallet or a Hold wallet.
* Each wallet has the following fields, stored in model StockWallet:
  + **portfolioStockId**. The id of the stock that has this wallet.
  + **walletType**. “Swing” or “Hold”. This is decided by the user’s selection of txnType when the Buy action is created.
  + **buyPrice**. The price at which the stock was bought. This makes the wallet unique, as there are no other wallets of the same type, with the same price. This is the price from the originating Buy transaction.
  + **totalSharesQty**. Total shares of this stock, ever bought at this price, from one or more Buy transactions at the same price.
  + **totalInvestment**. Total investment ever for this wallet, from one or more Buy transactions at the same price.
  + **sharesSold**. Shares sold from this wallet, through one or more Sell transactions.
  + remainingShares. sharesSold = totalSharesQty - sharesSold. This is updated when we have Sell transactions.
  + **realizedPl**. Accumulated P/L $ from sales from this wallet.
  + **tpValue**. Calculated TP Price ($) based on Buy price. tpValue = price + (price \* (pdp \* plr / 100)).
  + **tpPercent**. tpValue as a percent.
  + **sellTxnCount**. How many sales in total did we have from this wallet.
* When a wallet is being sold from, the Sell transaction shows up in the Transactions table.