Programme Bourse

Node modules : xlscsv csv2json

Parse dans un array

Modélise graphe avec chart.js

But/sell algo

Let data = /\* datas \*/

/\* Somme investie \*/

Let invest = /\*uservalue\*/

/\* Quantité achetée \*/

Let amount = 0;

For(let i=0, i<tab.length, i++) {

      If(i>0 && isNaN(data[i].alma)===false && isNaN(data[i].ema50)===false && isNaN(data[i].ema100)===false ){

         /\* Buy conditions \*/

         If(prevALMA > prevEMA100 &&              prevEMA100 > prevEMA50 && amount =0){

         amount = invest/data[i].price

         invest = 0

         }

         /\* Sell conditions \*/

         If(prevEMA100 > prevALMA && prevEMA50 > prevEMA100 && invest = 0) {

         Invest = amount×data[i].price

         amount = 0

         }

      }

      let prevALMA = data[i].alma

      let prevEMA50 = data[i].ema50

      let prevEMA100 = data[i].ema100

  }

Si ALMA(n-1)>EMA100(n-1) et EMA100(n-1)>EMA50(n-1) et opération précédente n'est pas un achat, alors achat à prix(n)

Si EMA100(n-1)>ALMA(n-1) et EMA50(n-1)>EMA100(n-1) et opération précédente n'est pas une vente, alors vente à prix(n)

Utilisation de script PowerShell pour pouvoir utiliser comme programme (node binaries, voir app stage) et trouver comment transformer PS1 en exe.

Fonctionnalités :

User doit pouvoir uploader un ou plusieurs jeux de données ( input type file multiple)

-> identification du type de fichier, si xls conversion en csv puis json, si csv conversion en json, si json, pas de conversion.

-> Si plusieurs jeux de données concaténation des données en un seul fichier

Chaque fichier parsé en array

If file>1 =>

Boucles imbriquées, si file1[i-1].date == file2[j-1].date {

   Si file1[i].date == file2[j].date {

      Si i>j { file1.splice(i-2, file1.length-(i-2)}

      Else { ....}

OU concaténer les diffeu files en un file + trouver algo tri par date + virer doublons

-> Pendant traitement, voir afficher barre type loading.

User doit pouvoir voir la représentation graphique des données (via chart.js)

User doit pouvoir entrer le montant de son investissement

User doit pouvoir voir ses gains, pertes, en montant et en pourcentage

Let InvestEvolution

If(amount == 0){

/\* Retourne invest et calcule pourcentage \*/

InvestEvolution = invest / baseInvest \*100

} Else {

/\* Retourne amount \* lastPrice et calcule le pourcentage \*/

Invest évolution = invest / ( amount\*lastPrice) \*100

}

User doit voir la courbe d'évolution en fonction des transactions (chart.js) + peut être tableau

-> ajouter a l'algo de transaction le stockage de la date, prix, montant lors de chaque opération dans un second tableau transactionData.

Let transactionData = []

transactionData.push({data[i].date, data[i].Price,  amount ou (invest/data[i].price) si opération de vente})

-> stockage sous forme de json du tableau

User doit pouvoir télécharger cet historique des opérations au format csv (ou xls).