```
require("data.table");require("stringr");require("pbapply");require("httr")
require("rvest");require("dplyr")

# ************************************************
# function to get Future Chains/Quotes/tickers
# ************************************************
getFutQuotes = function(ticker){
  # get the list of contracts available
  # page url
  pg <- html_session(paste0("https://www.barchart.com/futures/quotes/",ticker,"/futures-prices"))
  # save page cookies
  cookies <- pg$response$cookies
  # Use a named character vector for unquote splicing with !!!
  token <- URLdecode(dplyr::recode("XSRF-TOKEN", !!!setNames(cookies$value,
                                                             cookies$name)))
  shortN = str_sub(ticker,1,-4)
  # get data by passing in url and cookies
  pg <-
    pg %>% rvest:::request_GET(
      paste0("https://www.barchart.com/proxies/core-api/v1/quotes/get?fields=",
             "symbol%2CcontractSymbol%2ClastPrice%2CpriceChange%2CopenPrice%2",
             "ChighPrice%2ClowPrice%2CpreviousPrice%2Cvolume%2CopenInterest%2",
             "CtradeTime%2CsymbolCode%2CsymbolType%2ChasOptions&list=futures.",
             "contractInRoot&root=",shortN,"&meta=field.shortName%2Cfield.type%2Cfield.",
             "description&hasOptions=true&page=1&limit=100&raw=1")
      ,
      config = httr::add_headers(`x-xsrf-token` = token)
    )

  # raw data
  data_raw <- httr::content(pg$response)

  # convert into a data table
  futs = lapply(as.list(1:length(data_raw$data)), function(ii){
    as.data.frame(do.call(cbind,data_raw$data[[ii]]$raw))
  })
  futs = as.data.frame(rbindlist(futs,use.names = TRUE,fill = TRUE))
  futs$lastPrice = sapply(futs$lastPrice, as.numeric)
  futs$priceChange = sapply(futs$priceChange, as.numeric)
  futs$openPrice = sapply(futs$openPrice, as.numeric)
  futs$highPrice = sapply(futs$highPrice, as.numeric)
  futs$lowPrice = sapply(futs$lowPrice, as.numeric)
  futs$previousPrice = sapply(futs$previousPrice, as.numeric)
  futs$openInterest = sapply(futs$openInterest, as.numeric)
  futs$volume = sapply(futs$volume, as.numeric)
  futs$tradeTime = as.POSIXct(as.numeric(futs$tradeTime),origin="1970-01-01")
  futs$pctChange = round(futs$lastPrice/futs$previousPrice-1,4)
  # asssign Quotes
  assign("futs",futs,envir = .GlobalEnv)
  # extract futures contracts
  futNames = futs$symbol
  # exclude cash futures
  futNames = futNames[!str_detect(futNames,"00")]
  # return Future Quotes
  futNames
}
# ************************************************
# get Futures Historical Data
# ************************************************
getFuturesHistorical = function(ticker){
# get the list of contracts available
# page url
basePG = paste0("https://www.barchart.com/futures/quotes/",ticker,"/price-history/historical")
pg <- html_session(basePG)
```

```
# save page cookies
cookies <- pg$response$cookies
# Use a named character vector for unquote splicing with !!!
token <- URLdecode(dplyr::recode("XSRF-TOKEN", !!!setNames(cookies$value,
                                                           cookies$name)))
# get data by passing in url and cookies
pg <-
  pg %>% rvest:::request_GET(
    paste0("https://www.barchart.com/proxies/core-api/v1/historical/get?symbol=",ticker,
           "&fields=tradeTime.format(m%2Fd%2FY)%2CopenPrice%2ChighPrice%2ClowPrice",
           "%2ClastPrice%2CpriceChange%2CpercentChange%2Cvolume%2CopenInterest",
           "%2CsymbolCode%2CsymbolType&type=eod&orderBy=tradeTime&orderDir=desc",
           "&limit=10000&meta=field.shortName%2Cfield.type%2Cfield.description&raw=1")
    ,
    config = httr::add_headers(`x-xsrf-token` = token)
  )

# raw data
data_raw <- httr::content(pg$response)

# convert into a data table
data = lapply(as.list(1:length(data_raw$data)), function(ii){
  as.data.frame(do.call(cbind,data_raw$data[[ii]]$raw))
})
# rowbind
data = rbindlist(data,use.names = TRUE,fill = TRUE)
# add ticker to Column
data$LongName = ticker
# return table
data
}
# ************************************************


# ticker to start
ticker = "BTH21"

# get Quotes & latest contracts
futNames = getFutQuotes(ticker=ticker)

# get Historical Data for all Futures Contracts Available
ALL = lapply(as.list(futNames),function(x){
  Sys.sleep(5)
  tmp = try(getFuturesHistorical(ticker=x))
  if(!inherits(tmp,'try-error'))
    tmp
})
# remove empty lists
ALL = ALL[lapply(ALL,length)>0]

#combine data
ALL = rbindlist(ALL,use.names = TRUE, fill = TRUE)

# call unique Futures names in dataset
unique(ALL$LongName)
```