CoinSpy

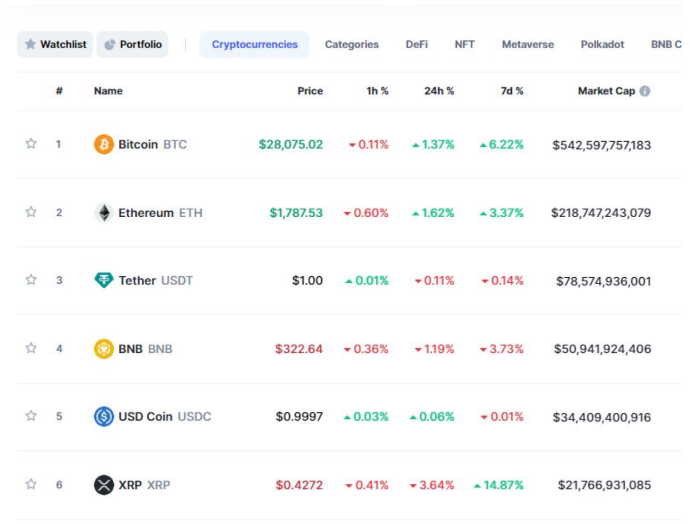
Final Project

Vlad Borungel

The IDEA

-With a large amount of cryptocurrencies various blockchains it could be a hassle to track your investments.

-I will be creating a way for people to get live coin prices and track their assets



Watchlist | Portfolio | Cryptocurrencies | Categories | DeFi | NFT | Metaverse | Polkadot | BNB C

#	Name	Price	1h %	24h %	7d %	Market Cap
1	Bitcoin BTC	$28,075.02	▼0.11%	▲1.37%	▲6.22%	$542,597,757,183
2	Ethereum ETH	$1,787.53	▼0.60%	▲1.62%	▲3.37%	$218,747,243,079
3	Tether USDT	$1.00	▲0.01%	▼0.11%	▼0.14%	$78,574,936,001
4	BNB BNB	$322.64	▼0.36%	▼1.19%	▼3.73%	$50,941,924,406
5	USD Coin USDC	$0.9997	▲0.03%	▲0.06%	▼0.01%	$34,409,400,916
6	XRP XRP	$0.4272	▼0.41%	▼3.64%	▲14.87%	$21,766,931,085

CoinSpy

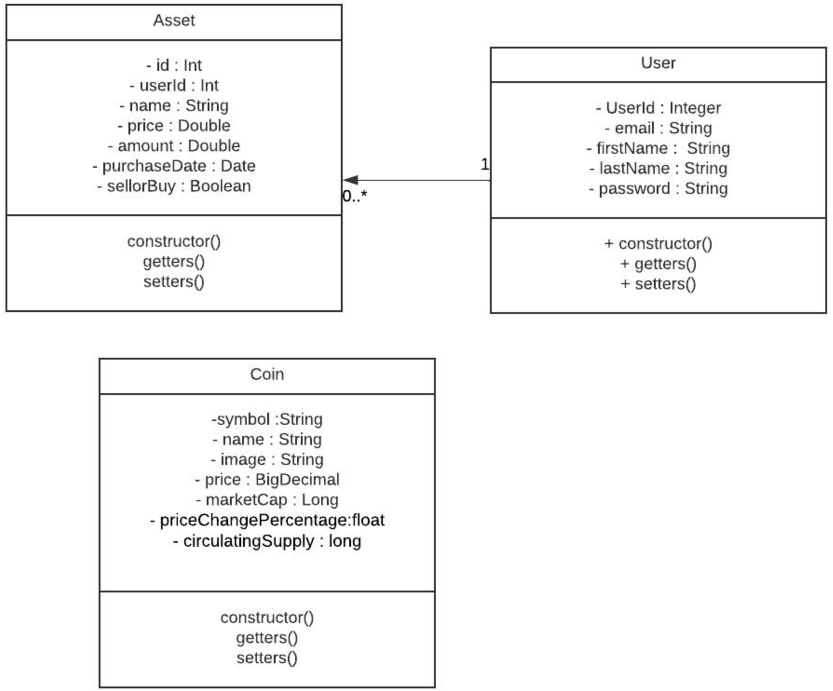
UML diagram

3 model classes:

User - login information

Asset - user to store the Users cryptocurrency

Coin - used to display live cryptocurrency prices

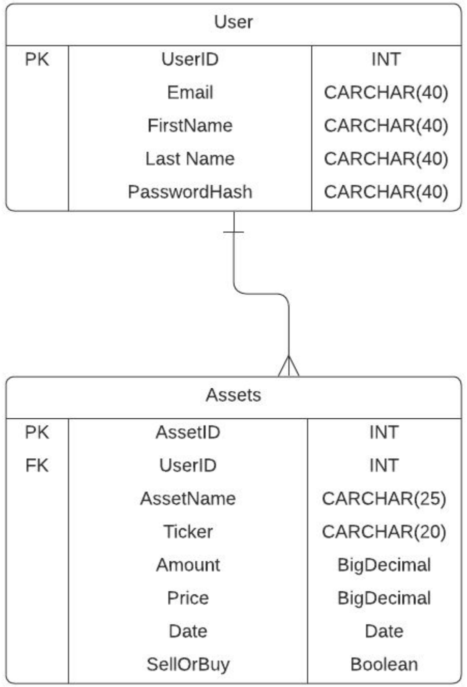


CoinSpy

Database

-The Coin objects do not need to be stored in the database as the information is retrieved from the Coin Gecko API

-Issue with accessing the current user principal, to get around this email has to be stored in Assets table



User		
PK	UserID	INT
	Email	CARCHAR(40)
	FirstName	CARCHAR(40)
	Last Name	CARCHAR(40)
	PasswordHash	CARCHAR(40)

Assets		
PK	AssetID	INT
FK	UserID	INT
	AssetName	CARCHAR(25)
	Ticker	CARCHAR(20)
	Amount	BigDecimal
	Price	BigDecimal
	Date	Date
	SellOrBuy	Boolean

Technology

Frontend : HTML ,CSS , BootStrap, Ajax , Thymeleaf

Backend: Java, Maven , JDBC

Database : MySQL

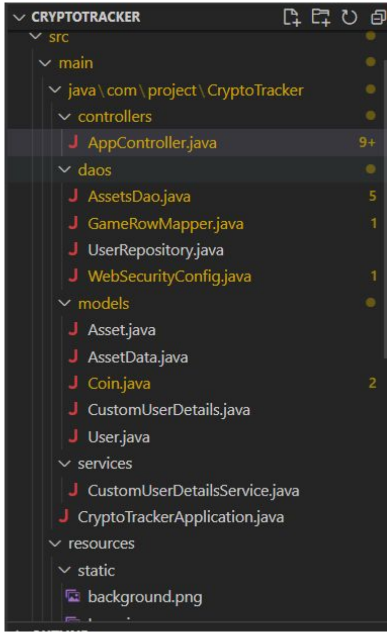
Backups : Git

Structure and testing

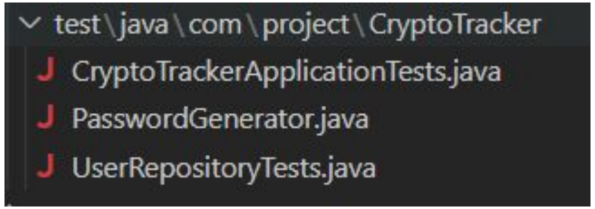
-Implemented a Register/Login system (outsourced from CodeJava) using Spring Security

-Follows MVC structure

-Unit testing was conducted to check login system and software functionality



```
CRYPTOTRACKER
  src
    main
      java\com\project\CryptoTracker
        controllers
          AppController.java                9+
        daos
          AssetsDao.java                    5
          GameRowMapper.java                1
          UserRepository.java
          WebSecurityConfig.java            1
        models
          Asset.java
          AssetData.java
          Coin.java                         2
          CustomUserDetails.java
          User.java
        services
          CustomUserDetailsService.java
        CryptoTrackerApplication.java
      resources
        static
          background.png
```



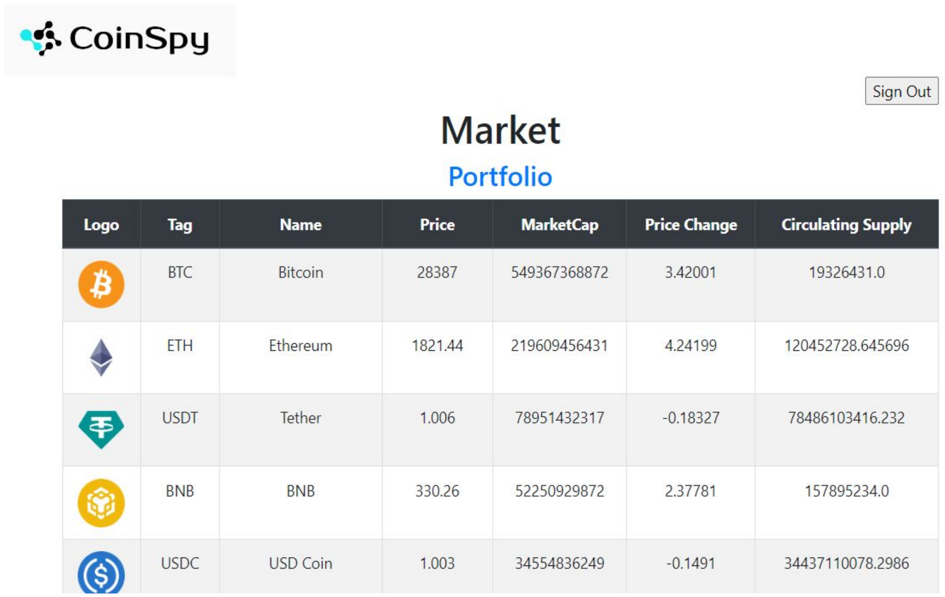
```
test\java\com\project\CryptoTracker
  CryptoTrackerApplicationTests.java
  PasswordGenerator.java
  UserRepositoryTests.java
```

CoinSpy

Market

Using coin gecko API I have retrieved information for over 11,000 coins.

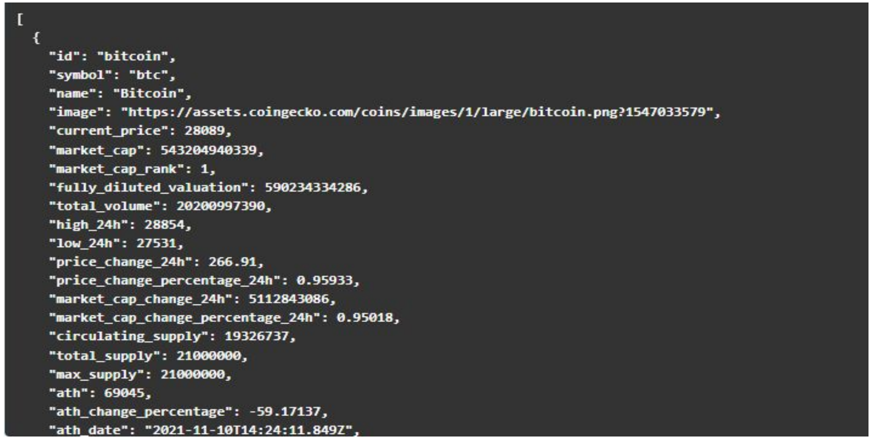
Sent a HTTP request and parsed the data using JSON.



Market

Portfolio

Logo	Tag	Name	Price	MarketCap	Price Change	Circulating Supply
	BTC	Bitcoin	28387	549367368872	3.42001	19326431.0
	ETH	Ethereum	1821.44	219609456431	4.24199	120452728.645696
	USDT	Tether	1.006	78951432317	-0.18327	78486103416.232
	BNB	BNB	330.26	52250929872	2.37781	157895234.0
	USDC	USD Coin	1.003	34554836249	-0.1491	34437110078.2986



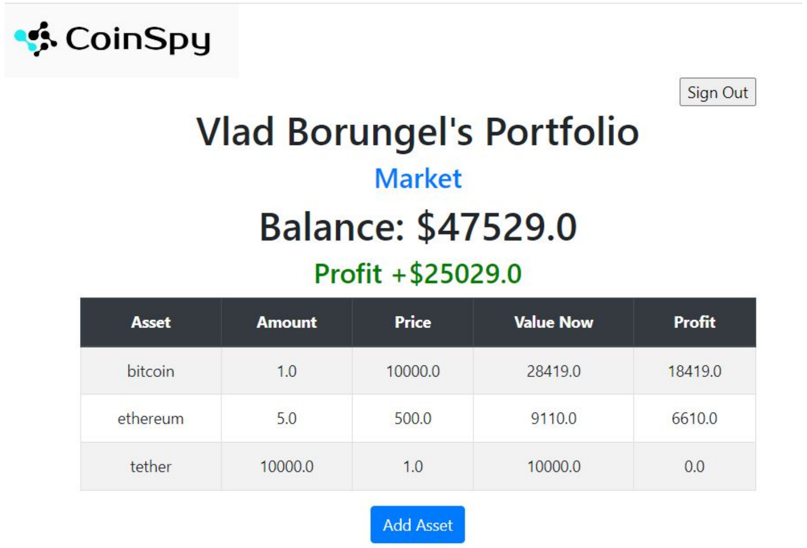
```
[
  {
    "id": "bitcoin",
    "symbol": "btc",
    "name": "Bitcoin",
    "image": "https://assets.coingecko.com/coins/images/1/large/bitcoin.png?1547033579",
    "current_price": 28089,
    "market_cap": 543204940339,
    "market_cap_rank": 1,
    "fully_diluted_valuation": 590234334286,
    "total_volume": 20200997390,
    "high_24h": 28854,
    "low_24h": 27531,
    "price_change_24h": 266.91,
    "price_change_percentage_24h": 0.95933,
    "market_cap_change_24h": 5112843086,
    "market_cap_change_percentage_24h": 0.95018,
    "circulating_supply": 19326737,
    "total_supply": 21000000,
    "max_supply": 21000000,
    "ath": 69045,
    "ath_change_percentage": -59.17137,
    "ath_date": "2021-11-10T14:24:11.849Z",
```

Portfolio

-The user is able to add assets into their portfolio using a modal.

-The portfolio displays the assets and the current price and profit.

-Used dropdown for coin validation to limit user input mistake

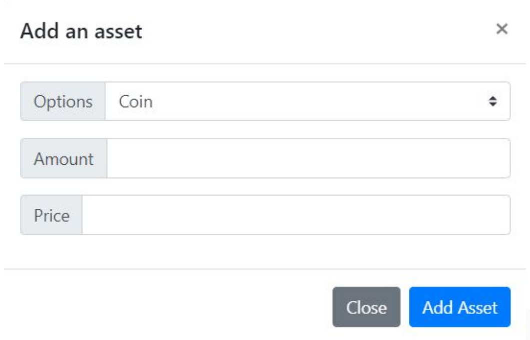


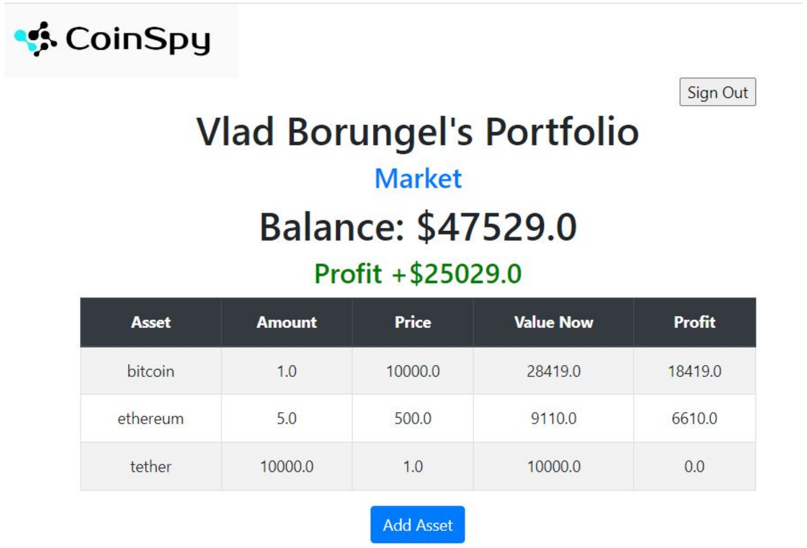
Vlad Borungel's Portfolio

Balance: $47529.0

Profit +$25029.0

Asset	Amount	Price	Value Now	Profit
bitcoin	1.0	10000.0	28419.0	18419.0
ethereum	5.0	500.0	9110.0	6610.0
tether	10000.0	1.0	10000.0	0.0

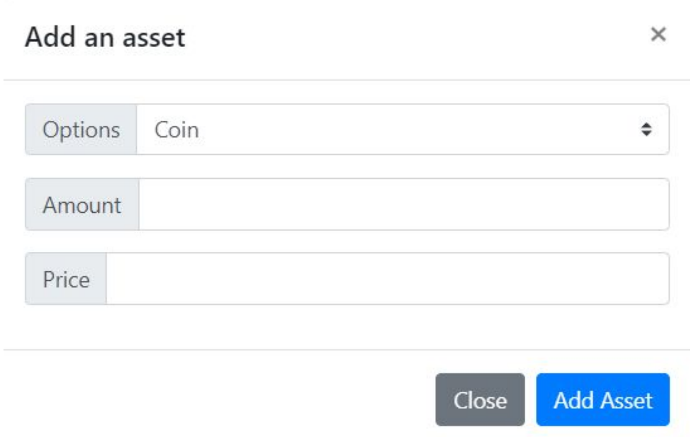


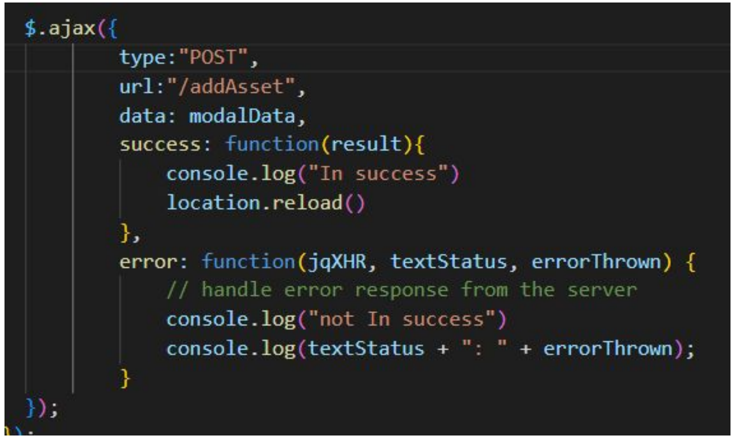


Ajax

Ajax is used to exchange data and is a form of communication between client and server. Practically, it is used to retrieve data and update a page without having to refresh everything.

This is used to add an asset to the portfolio and refresh the page.



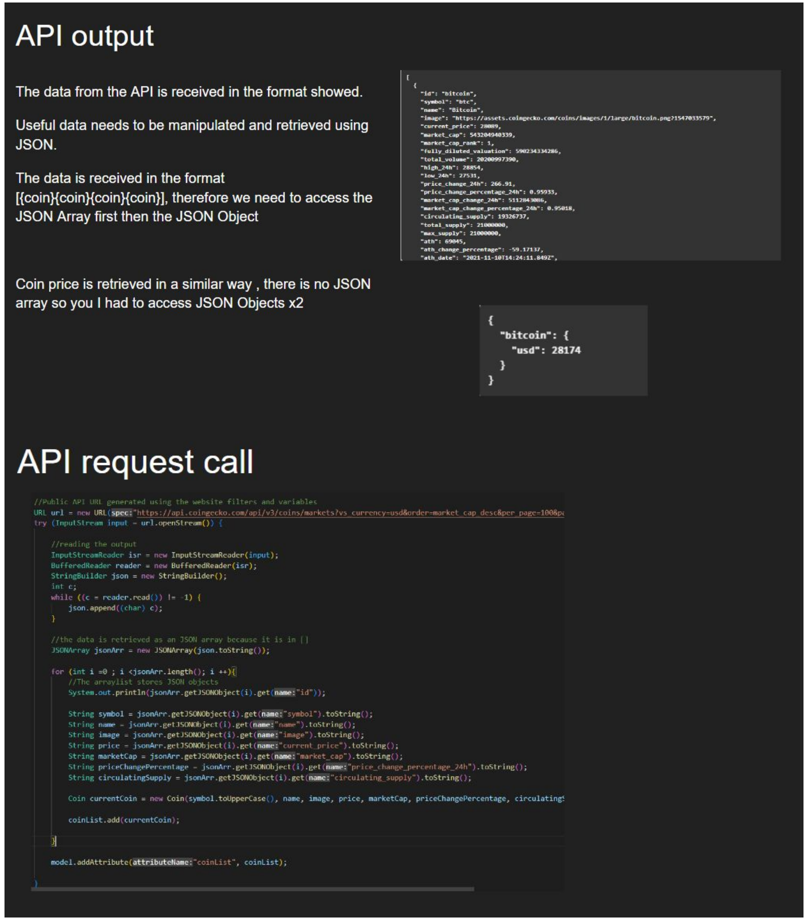


```
$.ajax({
        type:"POST",
        url:"/addAsset",
        data: modalData,
        success: function(result){
            console.log("In success")
            location.reload()
        },
        error: function(jqXHR, textStatus, errorThrown) {
            // handle error response from the server
            console.log("not In success")
            console.log(textStatus + ": " + errorThrown);
        }
});
```

Documentation

-Code structure,diagrams and wireframes

-Code documented for features like API call and Ajax https requests



API output

The data from the API is received in the format showed.

Useful data needs to be manipulated and retrieved using JSON.

The data is received in the format [{coin}{coin}{coin}{coin}], therefore we need to access the JSON Array first then the JSON Object

```
[
  {
    "id": "bitcoin",
    "symbol": "btc",
    "name": "Bitcoin",
    "image": "https://assets.coingecko.com/coins/images/1/large/bitcoin.png?1547033579",
    "current_price": 28089,
    "market_cap": 543204940339,
    "market_cap_rank": 1,
    "fully_diluted_valuation": 590234334286,
    "total_volume": 20200997390,
    "high_24h": 28854,
    "low_24h": 27531,
    "price_change_24h": 266.91,
    "price_change_percentage_24h": 0.95933,
    "market_cap_change_24h": 5112843086,
    "market_cap_change_percentage_24h": 0.95018,
    "circulating_supply": 19326737,
    "total_supply": 21000000,
    "max_supply": 21000000,
    "ath": 69045,
    "ath_change_percentage": -59.17137,
    "ath_date": "2021-11-10T14:24:11.849Z",
```

Coin price is retrieved in a similar way , there is no JSON array so you I had to access JSON Objects x2

```
{
  "bitcoin": {
    "usd": 28174
  }
}
```

API request call

```
//Public API URL generated using the website filters and variables
URL url = new URL(spec:"https://api.coingecko.com/api/v3/coins/markets?vs_currency=usd&order=market_cap_desc&per_page=100&pa
try (InputStream input = url.openStream()) {

    //reading the output
    InputStreamReader isr = new InputStreamReader(input);
    BufferedReader reader = new BufferedReader(isr);
    StringBuilder json = new StringBuilder();
    int c;
    while ((c = reader.read()) != -1) {
        json.append((char) c);
    }

    //the data is retrieved as an JSON array because it is in []
    JSONArray jsonArr = new JSONArray(json.toString());

    for (int i =0 ; i <jsonArr.length(); i ++){
        //The arraylist stores JSON objects
        System.out.println(jsonArr.getJSONObject(i).get(name:"id"));

        String symbol = jsonArr.getJSONObject(i).get(name:"symbol").toString();
        String name = jsonArr.getJSONObject(i).get(name:"name").toString();
        String image = jsonArr.getJSONObject(i).get(name:"image").toString();
        String price = jsonArr.getJSONObject(i).get(name:"current_price").toString();
        String marketCap = jsonArr.getJSONObject(i).get(name:"market_cap").toString();
        String priceChangePercentage = jsonArr.getJSONObject(i).get(name:"price_change_percentage_24h").toString();
        String circulatingSupply = jsonArr.getJSONObject(i).get(name:"circulating_supply").toString();

        Coin currentCoin = new Coin(symbol.toUpperCase(), name, image, price, marketCap, priceChangePercentage, circulating

        coinList.add(currentCoin);

    }

    model.addAttribute(attributeName:"coinList", coinList);

}
```

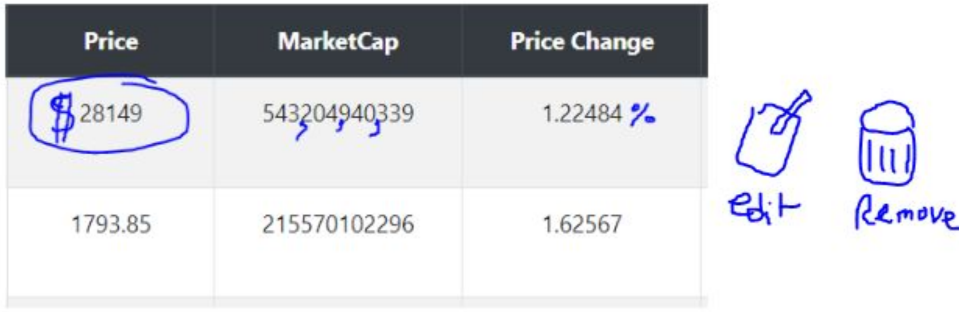
Extension

-Formatting of data presented

-Allow functionality to delete/edit an asset

-more CSS templates

Lessons Learned - Ajax is annoying



Price	MarketCap	Price Change
28149	543204940339	1.22484
1793.85	215570102296	1.62567