Build a Bitcoin Purchasing system in Golang deployable on AWS.

API:
The server will have 5 endpoints, */signup*, */login, /validate*, */buy,* and */sell*

When the user signs up, store their username and password in a database of your choice.

On sign up, we grant the user $500,000 of virtual money.

When the user logs in, we will generate and return a valid JWT token containing their username.

When the user hits the /validate endpoint, they will pass in the JWT Token which will be decoded to verify that it hasn't been modified or is expired.

When the user hits the /buy endpoint, we pass in the current price for bitcoin in USD as well as the amount of bitcoin the user wants to purchase. Validate that the user has enough money in their wallet and if they do, increment their bitcoin value and amount. Decrement the balance in their wallet to represent their new balance. Return the wallet balance and bitcoin holdings in the response.

When the user hits the /sell endpoint, we pass in the current price for bitcoin in USD as well as the amount of bitcoin the user wants to sell. Validate that the user has enough bitcoin in their account and if they do, decrement their bitcoin value and amount. Increment the balance in their wallet to represent their new balance. Return the wallet balance and bitcoin holdings in the response.

Deployment:
Once the Golang server is complete, please create a Cloudformation or Terraform script that would enable us to deploy this API as a Docker container in AWS. The script should create a VPC with two public subnets, an internet gateway, and a load balancer. The Docker image should be running in each of the two public subnets and the load balancer should direct traffic appropriately.

Other Notes:
For Bitcoin data, refer to this API:
https://api.coindesk.com/v1/bpi/currentprice.json

This exercise is open ended in the sense that you have total creative freedom to implement this as you see fit. You can add more functionality if you think it's important, you can keep it simple and clean, or you can do both. We hope you are able to learn some new things while doing the necessary research and implementing this project.

Good luck!

Endpoints:

/signup
Request Body:
{
  “username”: “user1”,
  “password”: “test123!!!”
}

/login
Request Body:
{
  “username”: “user1”,
  “password”: “test123!!!”
}

/buy
Headers: Authorization Token
Request Body:
{
  “bitcoin”: 1.4,
  “value”: 80,000
}

/sell
Headers: Authorization Token
Request Body:
{
  “bitcoin”: 1.4,
  “value”: 80,000
}

/validate
Request Body:
{
“token”:”eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c”
}

User data model
{

```
  “username”: “user1”,
  “password”: “w4rsdfqw4rasf34r234rfwef”
  “wallet”: {
    “value”: 420,000
  }
  “bitcoin”: {
    “amount”: 1.543
    “value”: 80,000
  }
}
```