

R-based API in ECS

Task

Deploy an R-based API in ECS for analyzing recent Binance (or any other real-time) data. The API should include at least 4 endpoints using different serializers, and these endpoints should be other than the ones we covered in the class. At least one endpoint should have at least a few parameters. Build a Docker image, push it to ECR, and deploy as service in ECS. Document the steps required to set up ECR/ECS with screenshots, then delete all services after confirming that everything works correctly.

Tech Setup

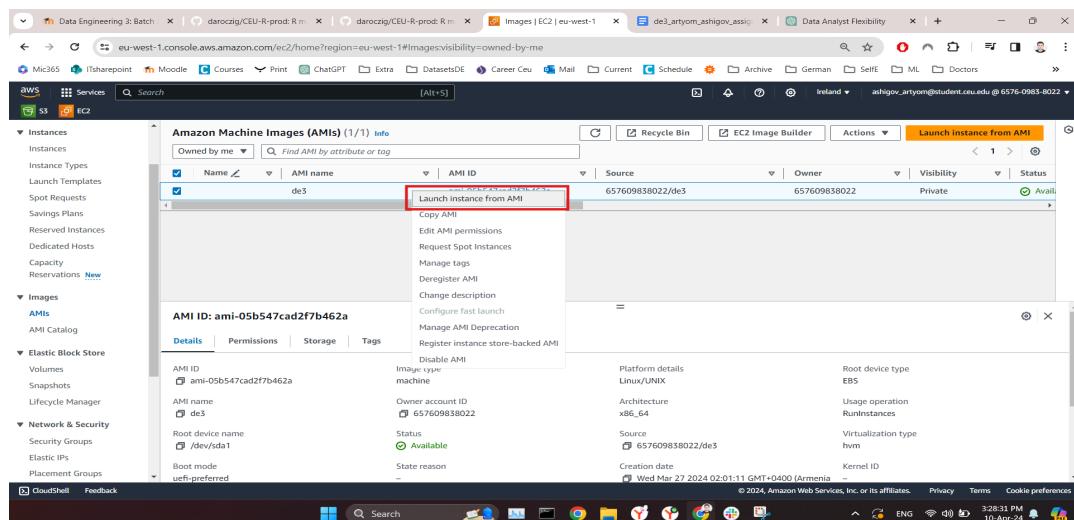
We are going to use the following preconfigured tools:

- **de3** Amazon Machine Image that you can use to spin up an EC2 node with RStudio Server, Shiny Server, Jenkins, Redis and Docker installed & pre-configured (use your AWS username and the password shared on Slack previously) along with the most often used R packages (including the ones we used for stream processing, eg botor, AWR.Kinesis and the binancer package)
- **de3** EC2 IAM role with full access to Kinesis, Dynamodb, Cloudwatch and the slack token in the Parameter Store
- **de3** security group with open ports for RStudio Server and Jenkins

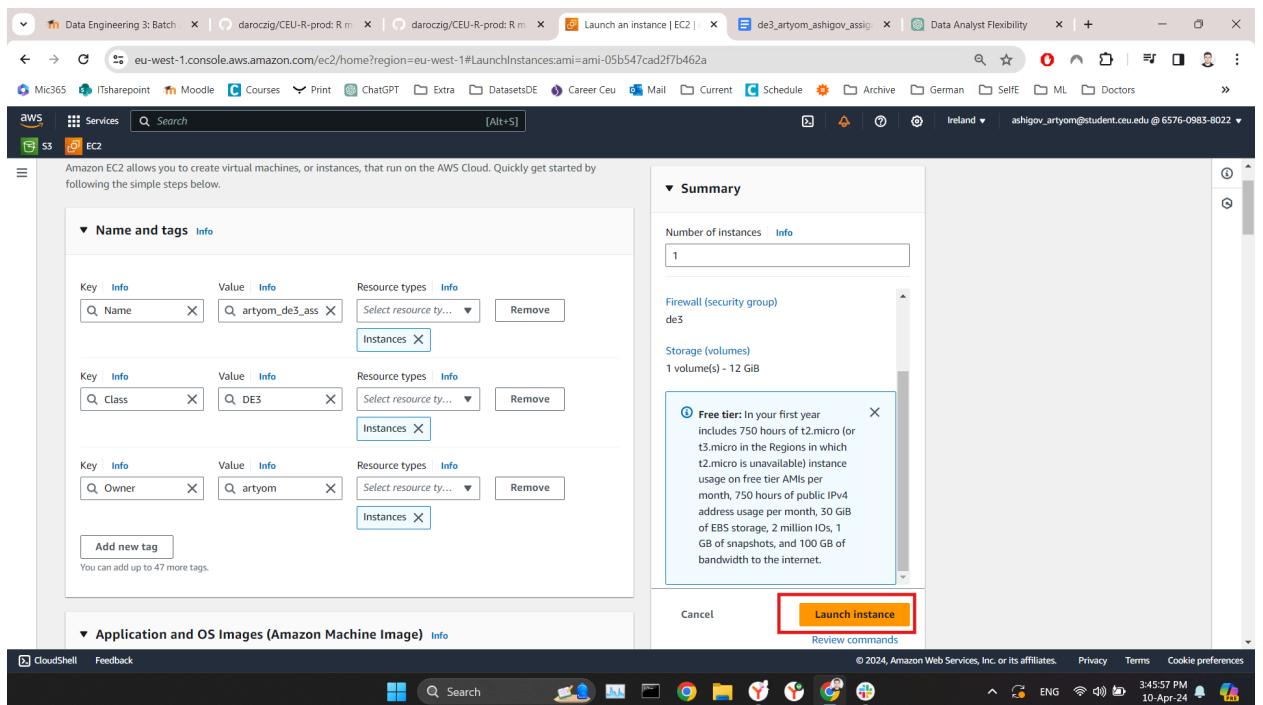
PART 1: Launching EC2 instance

We are going to launch an instance using a pre configured Amazon Machine Image (AMI) named **de3**.

1. We open EC2 and under the **Images** toggle menu select AMIs.
2. From there we right click on **de3** AMI, then click on **Launch instance from AMI**



3. Provide a name **artyom_de3_assignment**. Add additional tags:
 - a. owner: **artyom**
 - b. class: **DE3**
4. From **Application and OS Images (Amazon Machine Image)** toggle menu choose **de3** AMI
5. From **Instance type** choose **t3a.medium**
6. From **Key pair** select **artyom_de3**
7. From **Network settings** select **Existing security group** and choose **de3**
8. From **Advanced details** under **IAM instance profile** select **ceudataserver**
9. Click on **Launch Instance**



10. Click on Instance ID ([i-0376a845670c1d678](#))
11. Note: to access RStudio or Jenkins we add to our IP address /rstudio or /jenkins respectively.
12. username: **ashigov_artyom**
password: **TBehRz0zrUdkmBMVFF9VBE3Xy40nsE0dmdyDU0m6D4**

PART 2: Setup an API with 4 different endpoints and serializers

To set up our API, we'll be utilizing R Plumber. We've named and saved the file that contains the API as plumber.R for ease of use. Below is the complete code that we implemented to configure the API:

```
library(binancer)
library(readr)
library(dplyr)
```

```
library(ggplot2)
```

```
#* @apiTitle Cryptocurrencies Analysis API
#* @apiDescription Provides detailed statistics and visual insights on cryptocurrencies from
Binance including average prices, market depth, and price trends.

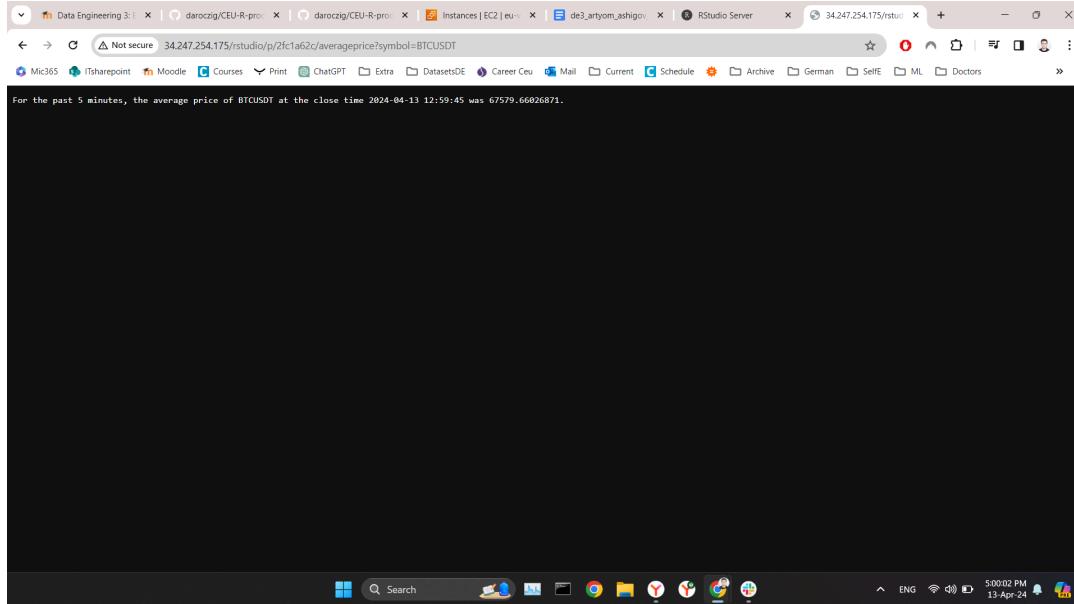
# Endpoint 1: Get the Average Price of a Cryptocurrency
#* @param symbol String: the symbol to fetch, default is 'BTCUSDT'
#* @get /averageprice
#* @serializer text
function(symbol = 'BTCUSDT') {
  # Fetch the average price from Binance
  x <- binance_avg_price(symbol)

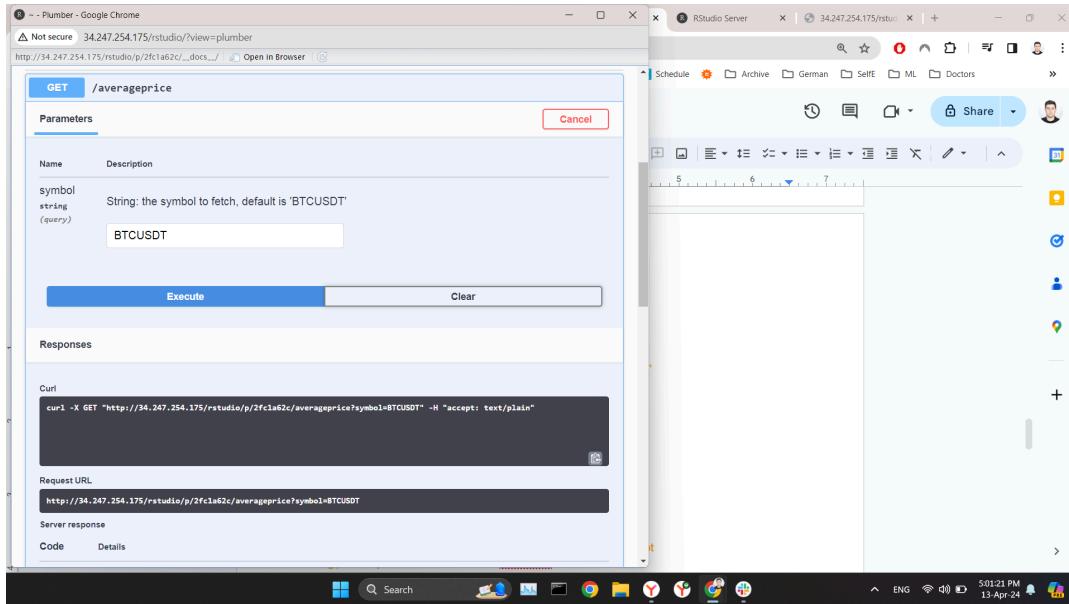
  # Format the closeTime for readability
  close_time_formatted <- format(as.POSIXct(x$closeTime / 1000, origin = "1970-01-01"),
"%Y-%m-%d %H:%M:%S")

  summary_text <- sprintf("For the past %d minutes, the average price of %s at the close time
%s was %s.",
      x$mins, symbol, close_time_formatted, x$price)

  summary_text
}
```

Results 1





```
# Endpoint 2: Retrieve Historical Market Depth in CSV Format
#* @param symbol String: the symbol to fetch, default is 'ETHUSDT'
#* @param limit Integer: example 1000
#* @get /bids-asks-csv
#* @serializer csv
function(symbol='ETHUSDT', limit = 1000) {
  depth <- binance_depth(symbol, limit = limit)
  bids_asks <- rbind(setNames(depth$bids, c("price", "quantity")),
                     setNames(depth$asks, c("price", "quantity")))
  bids_asks
}
```

Results 2

```

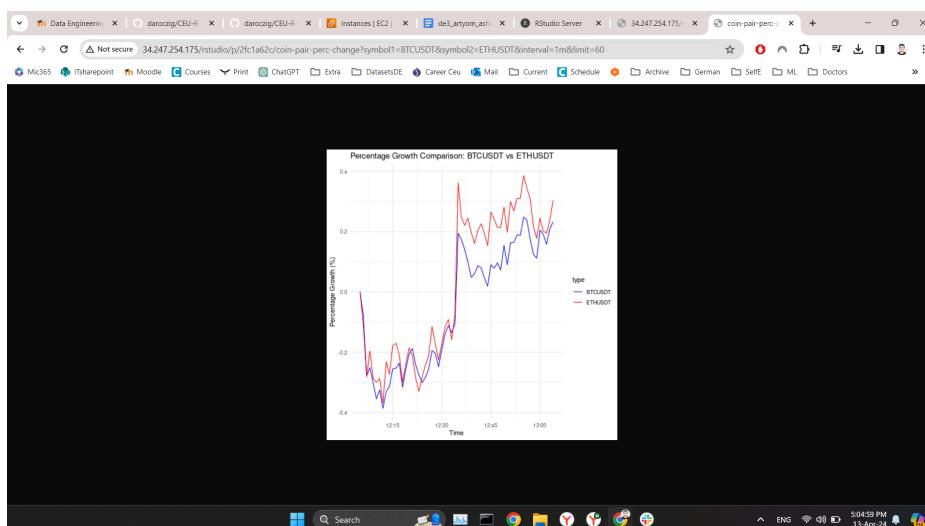
# Endpoint 3: RGenerate a Real-time Price Change Comparison Plot
#* @param symbol1 coin1: default is 'BTCUSDT'
#* @param symbol2 coin2: default is 'ETHUSDT'
#* @param interval Integer: example 1m
#* @param limit Integer: example 1000
#* @get /coin-pair-perc-change
#* @serializer png
function(symbol1 = 'BTCUSDT', symbol2 = 'ETHUSDT', interval = '1m', limit = 60L) {
  data1 <- binance_klines(symbol1, interval = interval, limit = as.integer(limit))
  data2 <- binance_klines(symbol2, interval = interval, limit = as.integer(limit))

  data1 <- data1 %>% mutate(percentage_growth = 100 * (close - first(close)) / first(close))
  data2 <- data2 %>% mutate(percentage_growth = 100 * (close - first(close)) / first(close))

  # Combine the datasets
  data1$type <- symbol1
  data2$type <- symbol2
  combined_data <- rbind(data1[, c("open_time", "percentage_growth", "type")],
                           data2[, c("open_time", "percentage_growth", "type")])
  # Generate the plot
  p <- ggplot(combined_data, aes(x = open_time, y = percentage_growth, color = type)) +
    geom_line() +
    theme_minimal() +
    labs(title = paste('Percentage Growth Comparison:', symbol1, 'vs', symbol2),
         x = 'Time', y = 'Percentage Growth (%)') +
    scale_color_manual(values = setNames(c('blue', 'red'), c(symbol1, symbol2)))
  # Output the plot
  print(p)
}

```

Results 3



```

# Endpoint 4: Get Historical Prices in HTML Format
#* @get /historical-prices-html
#* @serializer html
#* @param String: the symbol to fetch, default is 'BTCUSDT'
#* @param interval Integer: example 1m, 1d
#* @param limit Integer: example 100
function(symbol = 'BTCUSDT', interval = '1d', limit = 30) {
  prices <- binance_klines(symbol, interval = interval, limit = as.integer(limit))

  prices_html <- prices %>%
    knitr::kable("html") %>%
    as.character()

  html_output <- paste0("<html><body>", prices_html, "</body></html>")
  html_output
}

}

```

Results 4

open	time	high	low	close	volume	close_time	quote_asset	volume	trades	taker_buy_base_asset	volume	taker_buy_quote_asset	volume	symbol
65000.00	2024-03-15 09:00:00	65100.00	64900.00	64949.83	10331.00	2024-03-15 23:59:59			49844.05	360938570.00	360938570.00	BTCUSDT		
65000.00	2024-03-16 09:00:00	67000.00	65300.00	55924.69	2024-03-16 23:59:59				27240.19	184671512.00	184671512.00	BTCUSDT		
55924.69	2024-03-17 09:00:00	64000.00	63300.00	49743.22	2024-03-17 23:59:59				25328.294	169207961.00	169207961.00	BTCUSDT		
49743.22	2024-03-18 09:00:00	68000.00	66500.00	56691.08	2024-03-18 23:59:59				28559.672	1930736566.00	1930736566.00	BTCUSDT		
56691.08	2024-03-19 09:00:00	63124.11	61555.00	61937.40	2024-03-19 23:59:59				43206.284	309325080.00	309325080.00	BTCUSDT		
61937.40	2024-03-20 09:00:00	67840.41	65000.00	67840.41	2024-03-20 23:59:59				45523.843	290722206.00	290722206.00	BTCUSDT		
67840.41	2024-03-21 09:00:00	65450.12	53374.50	53374.50	2024-03-21 23:59:59				26660.098	177843851.00	177843851.00	BTCUSDT		
53374.50	2024-03-22 09:00:00	67064.6	5163240.41	5163240.41	2024-03-22 23:59:59				24578.880	1580940164.00	1580940164.00	BTCUSDT		
5163240.41	2024-03-23 09:00:00	66649.6	645900.00	645900.00	2024-03-23 23:59:59				17249.008	823892847.00	823892847.00	BTCUSDT		
645900.00	2024-03-24 09:00:00	67620.80	67327.72	67209.99	2024-03-24 23:59:59				16689.456	1095820626.00	1095820626.00	BTCUSDT		
67209.99	2024-03-25 09:00:00	67610.00	67110.00	67110.00	2024-03-25 23:59:59				26322.020	27815.588	27815.588	BTCUSDT		
67110.00	2024-03-26 09:00:00	67680.00	67156.00	67156.00	2024-03-26 23:59:59				19498.857	1915614174.00	1915614174.00	BTCUSDT		
67156.00	2024-03-27 09:00:00	676987.99	676987.99	676987.99	2024-03-27 23:59:59				137170830.00	137170830.00	137170830.00	BTCUSDT		
676987.99	2024-03-28 09:00:00	676989.62	676989.62	676989.62	2024-03-28 23:59:59				24902.335	173732591.00	173732591.00	BTCUSDT		
676989.62	2024-03-29 09:00:00	67800.60	65435.99	65435.99	2024-03-29 23:59:59				28007.269	1270852447.00	1270852447.00	BTCUSDT		
65435.99	2024-03-30 09:00:00	670780.60	670780.60	670780.60	2024-03-30 23:59:59				11839.093	82039932.00	82039932.00	BTCUSDT		
670780.60	2024-03-31 09:00:00	69850.54	25445.05	25445.05	2024-03-31 23:59:59					136441690.00	136441690.00	BTCUSDT		
25445.05	2024-04-01 09:00:00	69580.53	69580.53	69580.53	2024-04-01 23:59:59					6423.894	944936190.00	944936190.00	BTCUSDT	
69580.53	2024-04-02 09:00:00	69580.56	69580.56	69580.56	2024-04-02 23:59:59					10142.480	715151369.00	715151369.00	BTCUSDT	
69580.56	2024-04-03 09:00:00	69674.81	69674.81	69674.81	2024-04-03 23:59:59					20211.150	1402849945.00	1402849945.00	BTCUSDT	
69674.81	2024-04-04 09:00:00	69649.81	69649.81	69649.81	2024-04-04 23:59:59					35520.174	234263612.00	234263612.00	BTCUSDT	
69649.81	2024-04-05 09:00:00	69693.63	64493.07	64493.07	2024-04-05 23:59:59					20618.587	136036351.00	136036351.00	BTCUSDT	
64493.07	2024-04-06 09:00:00	65456.27	63930.99	63930.99	2024-04-06 23:59:59					21013.322	141569952.00	141569952.00	BTCUSDT	
63930.99	2024-04-07 09:00:00	68487.50	68487.50	68487.50	2024-04-07 23:59:59					255458953.00	1273220223.00	1273220223.00	BTCUSDT	
68487.50	2024-04-08 09:00:00	67747.83	68896.00	68896.00	2024-04-08 23:59:59					1973131968.00	1973131968.00	1973131968.00	BTCUSDT	
68896.00	2024-04-09 09:00:00	70326.29	69360.39	69360.39	2024-04-09 23:59:59					1496213373.00	1269040.00	1269040.00	BTCUSDT	
69360.39	2024-04-10 09:00:00	71758.19	71758.19	71758.19	2024-04-10 23:59:59					11449.304	77496775.00	77496775.00	BTCUSDT	
71758.19	2024-04-11 09:00:00	70631.00	70631.00	70631.00	2024-04-11 23:59:59					2370853310.00	1811209902.00	1811209902.00	BTCUSDT	
70631.00	2024-04-12 09:00:00	22.1227	76006.22	76006.22	2024-04-12 23:59:59					2749073275.00	1321865585.00	1321865585.00	BTCUSDT	
22.1227	2024-04-13 09:00:00	65792.00	65731.12	65731.12	2024-04-13 23:59:59					15908.491	1120249941.00	1120249941.00	BTCUSDT	
65731.12	2024-04-14 09:00:00	67603.99	17948.27	17948.27	2024-04-14 23:59:59					384351585.00	177780849.00	177780849.00	BTCUSDT	
17948.27	2024-04-15 09:00:00	67603.52	65487.79	65487.79	2024-04-15 23:59:59					1203383218.00	537316318.00	537316318.00	BTCUSDT	

PART 3: Build a Docker Image

To build a Docker Image first we need to create a Dockerfile with all necessary settings:

1. Create a text file
2. Add the content:

```
FROM rstudio/plumber
```

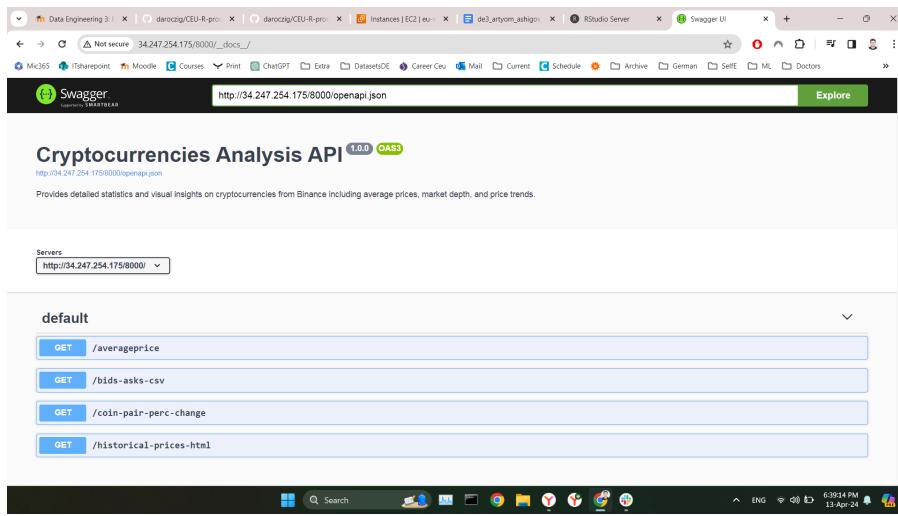
```
RUN apt-get update && apt-get install -y pandoc && apt-get clean && rm -rf /var/lib/apt/lists/
RUN install2.r ggplot2 dplyr readr
RUN installGithub.r daroczig/binancer
```

ADD plumber.R /app/plumber.R

EXPOSE 8000

WORKDIR /app

3. Save the file as Dockerfile
4. Run the command in the terminal: sudo docker build -t crypto-api .
5. Check the images: sudo docker images
6. Run the container based on the image: sudo docker run -p 8000:8000 -ti crypto-api plumber.R
7. Access the address: we need to add /8000/_docs_ to our ip address
8000 is our port

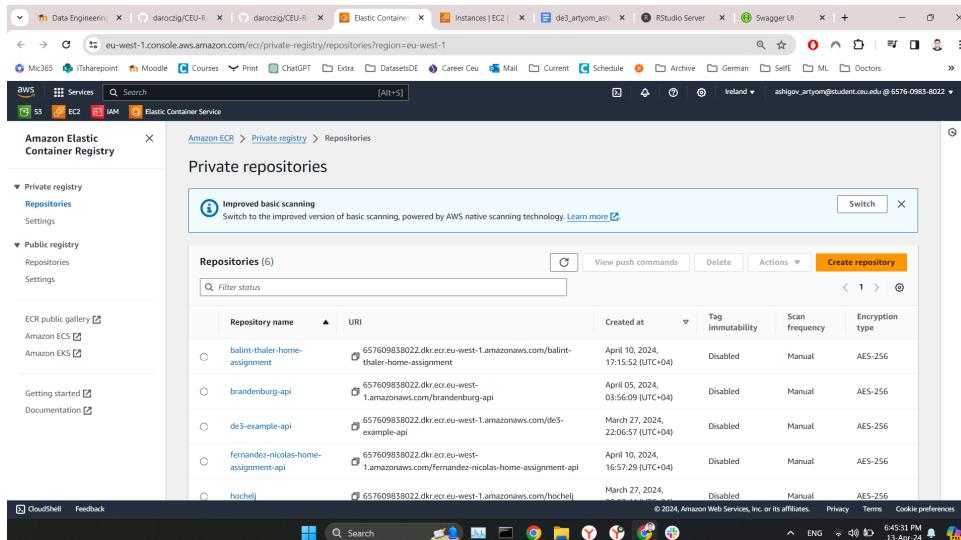


PART 4: Push the image to ECR

Next, we'll deploy the Docker image that was created and tested above by uploading it to the Elastic Container Registry (ECR) to make it accessible outside of the RStudio Server.

We will do the following steps:

1. Create a private repository



Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type
balint-thaler-home-assignment	657609838022.dkr.ecr.eu-west-1.amazonaws.com/balint-thaler-home-assignment	April 10, 2024, 17:15:52 (UTC-04)	Disabled	Manual	AES-256
brandenburg-api	657609838022.dkr.ecr.eu-west-1.amazonaws.com/brandenburg-api	April 05, 2024, 03:56:09 (UTC-04)	Disabled	Manual	AES-256
de3-example-api	657609838022.dkr.ecr.eu-west-1.amazonaws.com/de3-example-api	March 27, 2024, 22:06:57 (UTC-04)	Disabled	Manual	AES-256
fernandez-nicolas-home-assignment-api	657609838022.dkr.ecr.eu-west-1.amazonaws.com/fernandez-nicolas-home-assignment-api	April 05, 2024, 16:57:29 (UTC-04)	Disabled	Manual	AES-256
hochelj	657609838022.dkr.ecr.eu-west-1.amazonaws.com/hochelj	March 27, 2024, 22:06:57 (UTC-04)	Disabled	Manual	AES-256

2. Provide a name `artyom-assignment`

3. Click on **Create Repository**

4. To login to ECR we run this command in the terminal:

```
aws ecr get-login-password --region eu-west-1 | sudo docker
login --username AWS --password-stdin
657609838022.dkr.ecr.eu-west-1.amazonaws.com
```

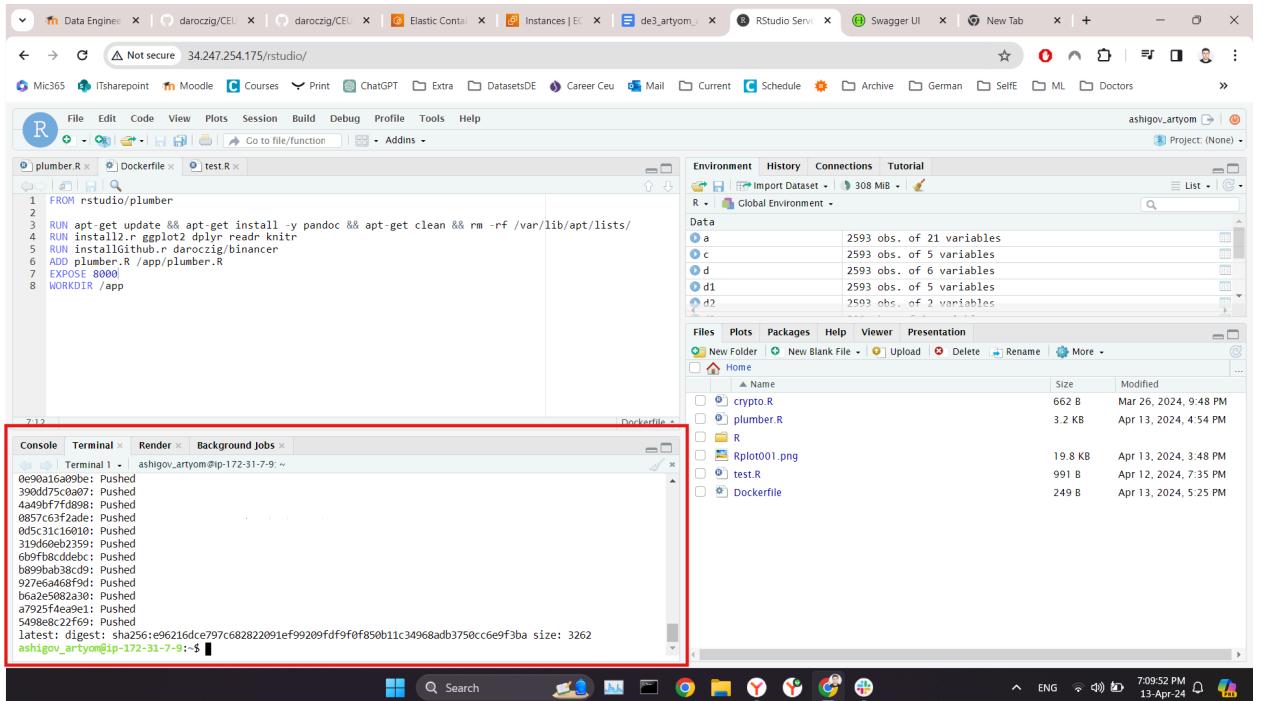
5. Check docker images again: `sudo docker images`

6. We need to rename our docker image for upload:

```
sudo docker tag crypto-api:latest
657609838022.dkr.ecr.eu-west-1.amazonaws.com/artyom-assignme
nt:latest
```

7. Push the image:

```
sudo docker push
657609838022.dkr.ecr.eu-west-1.amazonaws.com/artyom-assignme
nt:latest
```



8. Check our repository

https://eu-west-1.console.aws.amazon.com/ecr/repositories/private/657609838022/artyo_m-assignment?region=eu-west-1

PART 5: Deploy as service in ECS

We will utilize Amazon Elastic Container Service (ECS) to deploy it as a service. Here are the steps we will follow:

1. We open ECS

<https://eu-west-1.console.aws.amazon.com/ecs/v2/clusters?region=eu-west-1>

2. Click on Task definitions

Cluster	Services	Tasks	Container instances	CloudWatch monitoring	Capacity provider strategy
KOKAKALE_API	1	0 Open... 1 Run...	0 EC2	Default	No default found
FERNANDEZ_NICOLAS_API	0	No tasks running	0 EC2	Default	No default found
BTC_API_Balint-Thaler	0	No tasks running	0 EC2	Default	No default found

3. Provide a name under **Task definition family**: artyom-assignment

4. Under **Container - 1** part fill

a. Name - **api**

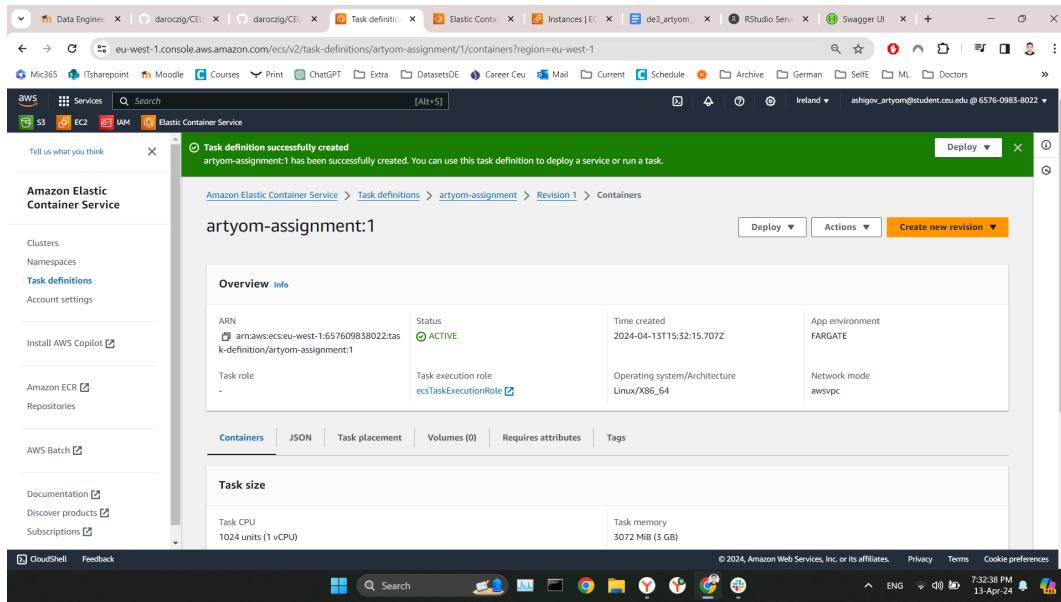
b. Image URI -

657609838022.dkr.ecr.eu-west-1.amazonaws.com/artyom-assignment

c. Container port: **8000**

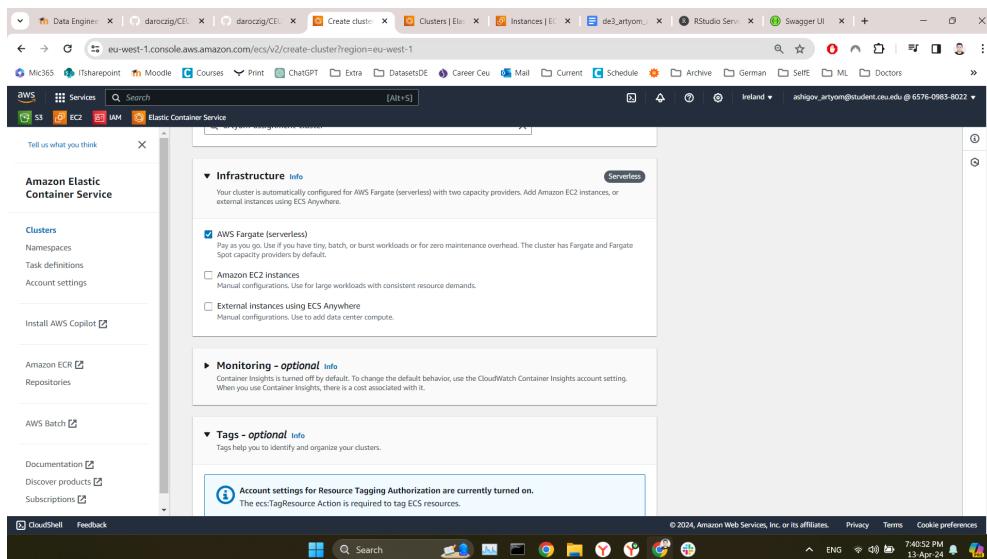
5. From Docker configuration under Command field put **plumber.R**

6. Click on **Create Result**



Now we need to create a cluster.

1. Click on **Clusters** from left-side menu
2. Click on **Create Cluster**
3. Put the **Cluster name: artyom-assignment-cluster**
4. Choose **AWS Fargate** under **Infrastructure**



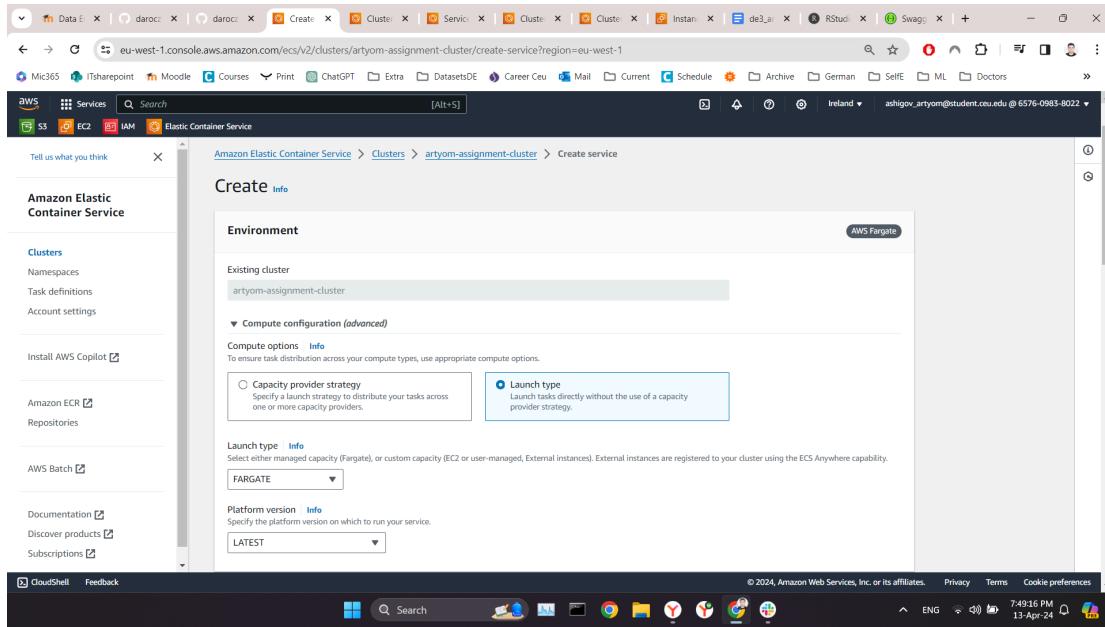
5. Add a tag: Class - DE3
6. Click on **Create**

Now we need to create a service in our cluster.

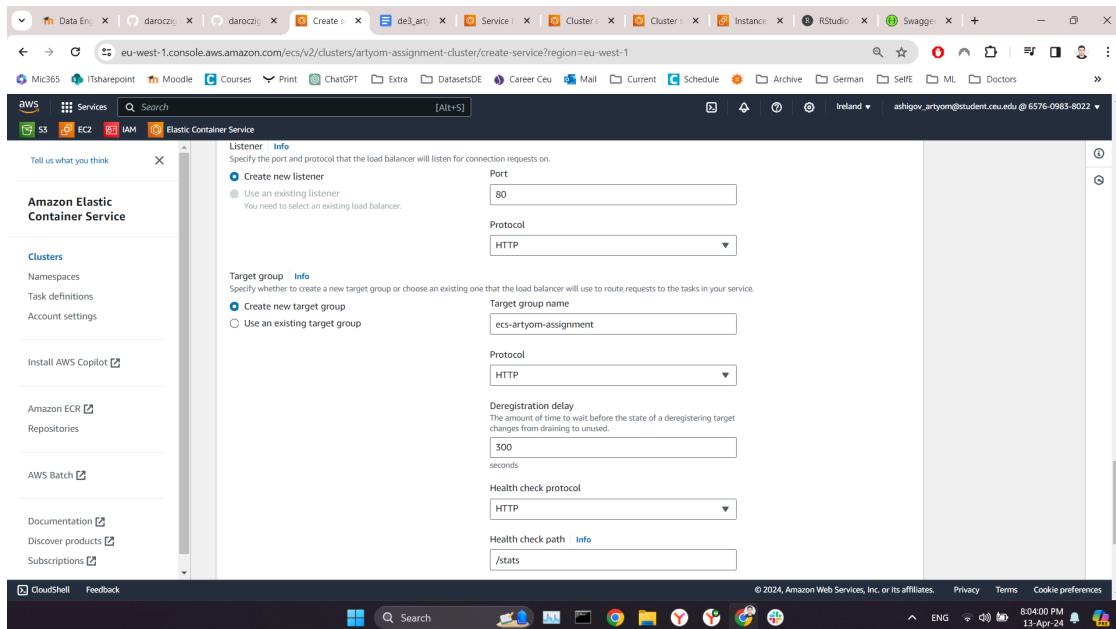
1. Click on our cluster:

<https://eu-west-1.console.aws.amazon.com/ecs/v2/clusters/artyom-assignment-cluster/services?region=eu-west-1>

2. Click on **Create** under Services



3. Choose **Launch type** under **Compute options**
4. Under the **Deployment configuration**, select **artyom-assignment** from the Family options.
5. Put the same under **Service name**
6. Under **Networking** choose our security group: **de3**
7. Scroll and under **Load balancing** from **Load balance type** choose **Application Load Balancer**, give it a name **artyom-assignment-balancer**
8. Put 10 under **Health check grace period**
9. Create a listener on port 80 using **/coin-pair-perc-change** (or any other endpoints) as the **Health check path** and click on **Create** afterwards



10. Test the deployed service behind the load balancer using DNS name and adding an endpoint:

Examples

<http://artyom-assignment-balancer-1502977940.eu-west-1.elb.amazonaws.com/coin-pair-perc-change>



or

<http://artyom-assignment-balancer-1502977940.eu-west-1.elb.amazonaws.com/historical-prices-html>

open_time	open	high	low	close	volume	close_time	quote_asset	volume	trades	taker_buy_base_asset	taker_buy_quote_asset	volume	symbol
2024-03-15 71388.94	72419.71	65600.00	69499.85	103334.04	2024-03-15 23:59:59	7065279918 3904445		49844.665	3409385750	BTCUSDT			
2024-03-16 69499.84	70043.00	64780.00	65300.63	55926.95	2024-03-16 23:59:59	3788544203 2729019		27240.199	1846731512	BTCUSDT			
2024-03-17 65300.64	68904.40	64533.00	68393.49	49742.22	2024-03-17 23:59:59	3322963634 2449156		25328.294	1692079618	BTCUSDT			
2024-03-18 68393.47	68956.00	66565.20	67609.99	55691.08	2024-03-18 23:59:59	37682277912 2464515		28529.672	1930726566	BTCUSDT			
2024-03-19 67610.00	68124.11	61555.00	61937.40	101005.32	2024-03-19 23:59:59	6484805949 3593832		48205.284	3095258882	BTCUSDT			
2024-03-20 61937.41	68100.00	60775.00	67840.51	90420.59	2024-03-20 23:59:59	5769770052 3549793		45523.843	2907222064	BTCUSDT			
2024-03-21 67840.51	68240.47	64529.01	65501.27	53357.48	2024-03-21 23:59:59	3551223143 2388390		26650.098	1774833851	BTCUSDT			
2024-03-22 65501.28	66649.62	62260.00	63796.61	51482.38	2024-03-22 23:59:59	3310481978 2492881		24578.880	1580940164	BTCUSDT			
2024-03-23 63796.64	65999.00	63000.00	63990.01	26410.11	2024-03-23 23:59:59	1706152173 1753566		12749.008	823892847	BTCUSDT			
2024-03-24 63990.02	67628.69	63772.29	67209.99	31395.75	2024-03-24 23:59:59	2059312949 1880774		16689.656	1095820626	BTCUSDT			
2024-03-25 67210.00	71150.00	66385.06	69880.01	53431.14	2024-03-25 23:59:59	3679964074 2632220		27815.588	1915614174	BTCUSDT			
2024-03-26 69880.00	71561.10	69280.00	69988.00	38934.38	2024-03-26 23:59:59	2738537345 2176807		19498.857	1371708207	BTCUSDT			
2024-03-27 69988.00	71769.10	68339.18	69469.99	49119.36	2024-03-27 23:59:59	3426458582 2453366		24902.335	1737325916	BTCUSDT			
2024-03-28 69469.99	71552.06	68903.62	70780.61	35439.03	2024-03-28 23:59:59	2500571208 1799897		18007.269	1270852447	BTCUSDT			
2024-03-29 70780.60	70916.16	69009.00	69850.54	25445.08	2024-03-29 23:59:59	1779608477 1522607		11839.093	828039937	BTCUSDT			
2024-03-30 69850.53	70321.10	69540.00	69582.18	13644.61	2024-03-30 23:59:59	95455920 1110488		6423.894	449436190	BTCUSDT			
2024-03-31 69582.17	71366.00	69562.99	71280.01	19396.34	2024-03-31 23:59:59	13675237078 1181926		10142.480	715151369	BTCUSDT			
2024-04-01 71280.00	71288.23	68062.86	69649.89	41445.32	2024-04-01 23:59:59	2876486469 1893438		20211.150	1402849945	BTCUSDT			
2024-04-02 69649.81	69674.23	64550.00	65463.99	71799.83	2024-04-02 23:59:59	4750030718 2662410		35520.174	2348263612	BTCUSDT			
2024-04-03 65463.99	66903.63	64493.07	65963.28	39887.22	2024-04-03 23:59:59	2630556596 2172096		20618.587	1360363531	BTCUSDT			
2024-04-04 65963.27	69309.91	65064.52	68487.79	41510.48	2024-04-04 23:59:59	2795469451 2000945		21013.322	1415699592	BTCUSDT			
2024-04-05 68487.80	68756.67	65952.56	67820.61	37915.23	2024-04-05 23:59:59	255548935 1901980		18889.985	1273220223	BTCUSDT			
2024-04-06 67820.63	69692.00	67447.83	68896.00	20134.29	2024-04-06 23:59:59	1373131968 1171460		9973.864	680800856	BTCUSDT			
2024-04-07 68896.00	70326.29	68824.00	69360.39	21534.74	2024-04-07 23:59:59	1496213373 1269048		11149.904	774906775	BTCUSDT			
2024-04-08 69360.38	72797.99	69043.24	71620.00	45723.88	2024-04-08 23:59:59	3270853310 2037025		25309.905	1811209902	BTCUSDT			
2024-04-09 71620.00	71758.10	68210.00	69146.00	39293.90	2024-04-09 23:59:59	2749007422 1804157		18896.538	1321865585	BTCUSDT			
2024-04-10 69146.00	71172.08	67518.00	70631.08	42006.02	2024-04-10 23:59:59	2901564971 1852170		20269.348	1401023004	BTCUSDT			
2024-04-11 70631.08	71305.89	69567.21	70006.23	31917.26	2024-04-11 23:59:59	224743275 1658995		15908.491	1120249991	BTCUSDT			
2024-04-12 70006.22	71227.46	65086.86	67116.52	56072.86	2024-04-12 23:59:59	3843551585 2496617		25935.865	1777805491	BTCUSDT			
2024-04-13 67116.52	67929.00	60660.57	62162.90	49504.08	2024-04-13 23:59:59	3231937714 2542164		22105.114	1442835760	BTCUSDT			