Cloud Computing: Project 6: Deliverables

Building an E-commerce Microservices Application on Cloud using Docker, Kubernetes, Jenkins and Git

PES2UG21CS461	Saksham Alok
PES2UG21CS463	Samantha Susan John
PES2UG21CS915	Himanshu Nanda
PES2UG21CS917	Kushaagra Shrivastava

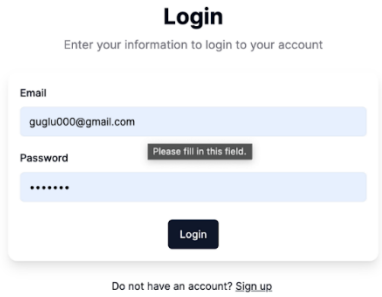
Table of Contents

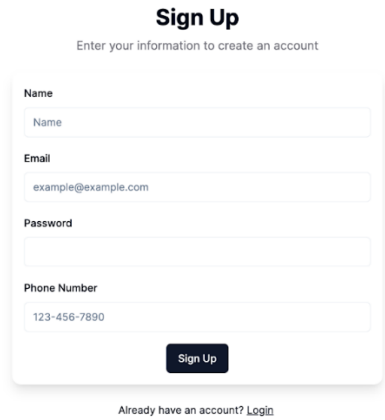
Cloud Computing Project 6 Deliverables
461-463-915-917

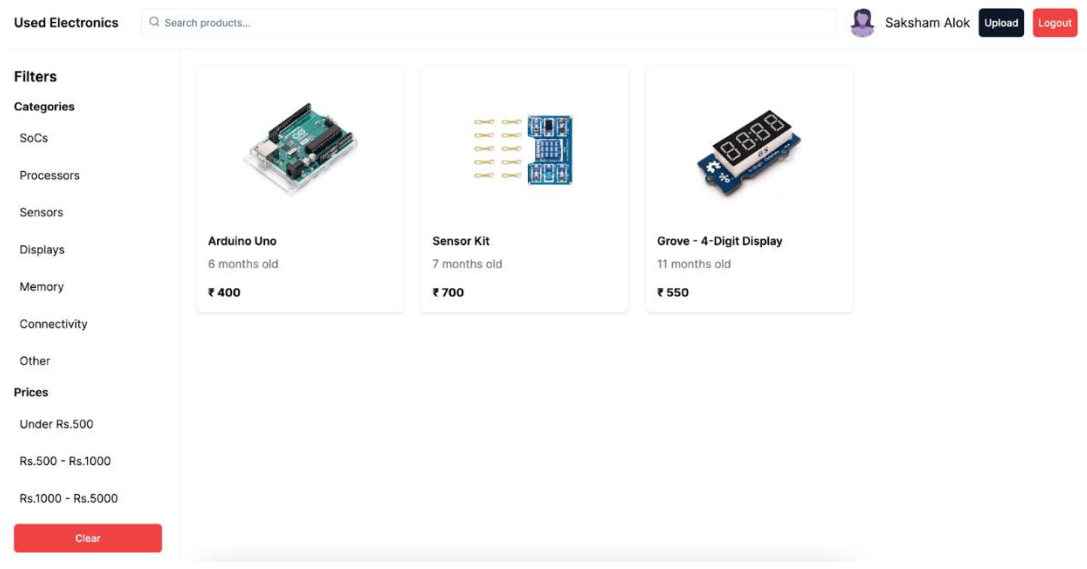
Week 1

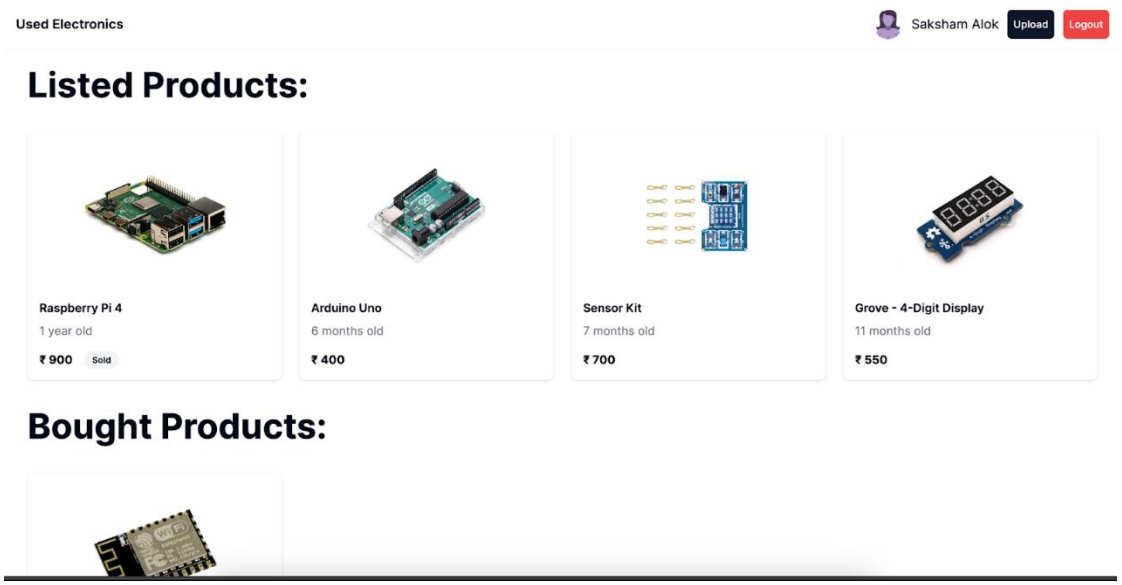
Microservices Architecture Design and Development

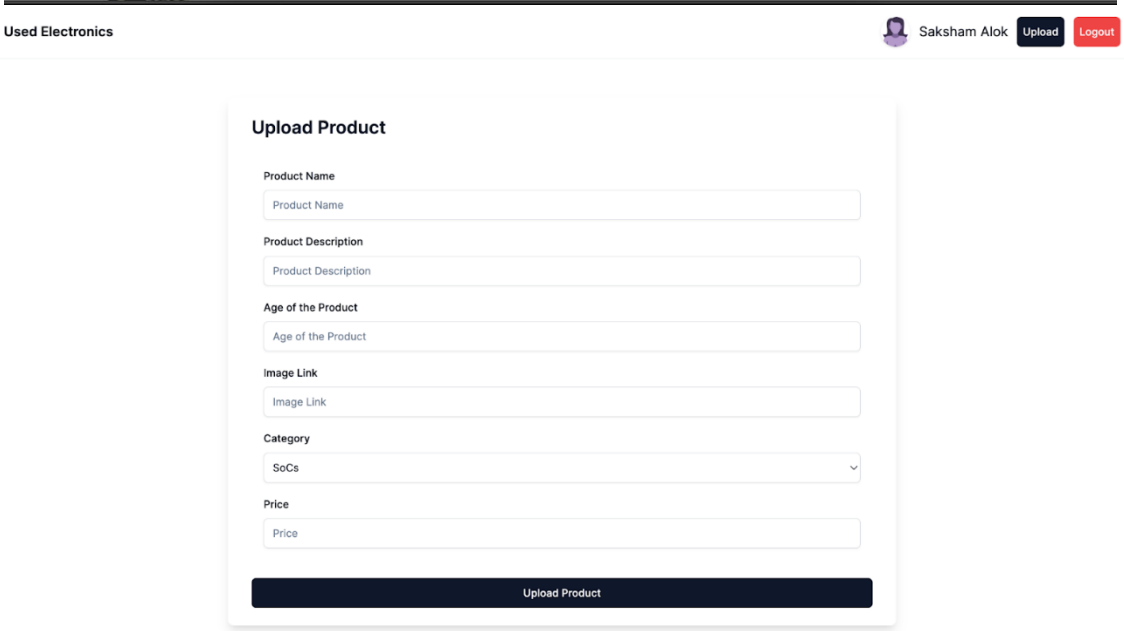
Screenshots

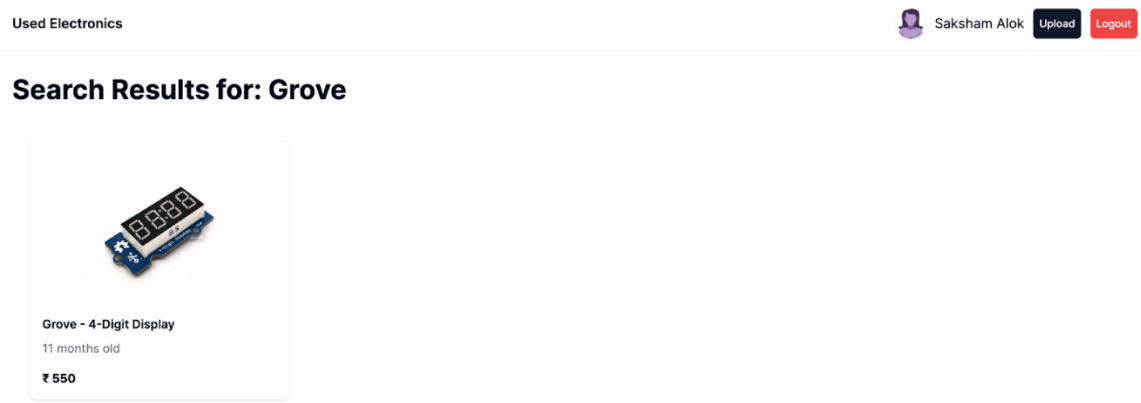












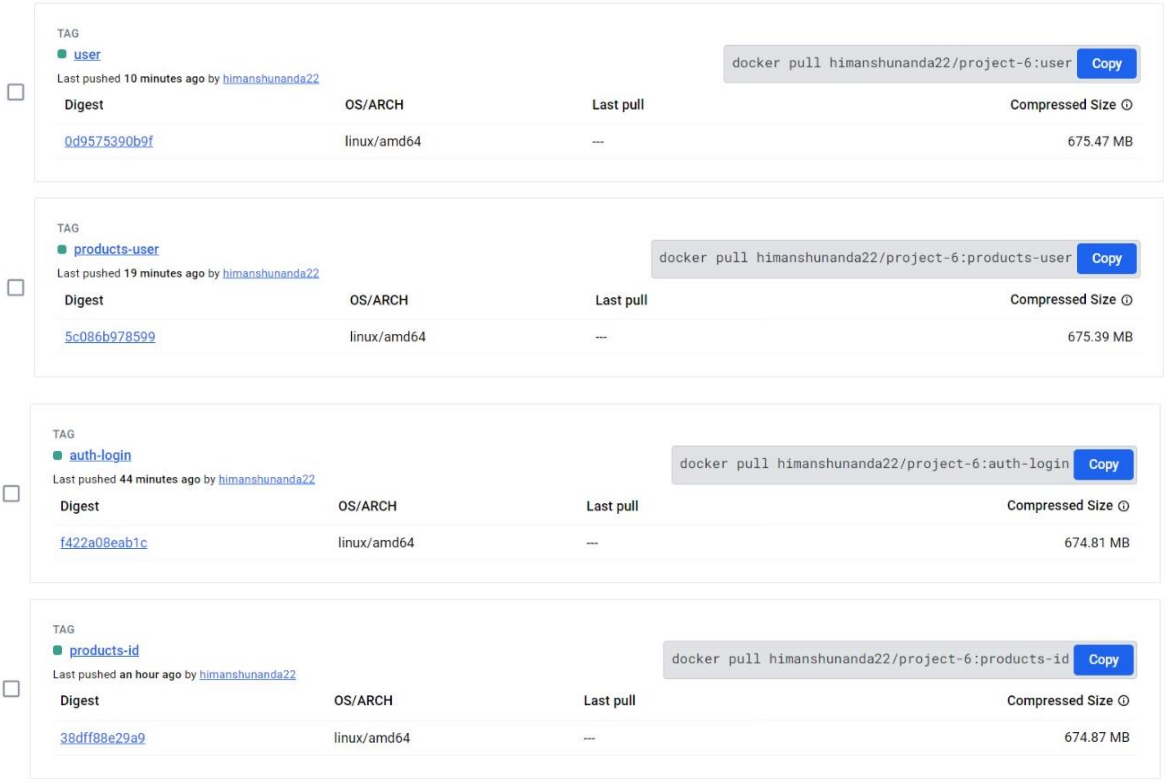
GitHub Repository Link
https://github.com/KushShriv/Monolithic-to-Microservices-CC.git

Week 2

Containerization and Orchestration

Docker

Containers



Auth-login

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker build -t auth-login:latest -f app/api/auth/login/D
ockerfile .
[+] Building 0.0s (0/0)  docker:default
[+] Building 28.6s (12/12) FINISHED                                                                                   docker:default
 => [internal] load build definition from Dockerfile                                                                            0.0s
 => => transferring dockerfile: 624B                                                                                            0.0s
 => [internal] load metadata for docker.io/library/node:20                                                                      1.2s
 => [internal] load .dockerignore                                                                                               0.0s
 => => transferring context: 2B                                                                                                 0.0s
 => [internal] load build context                                                                                               0.0s
 => => transferring context: 5.27kB                                                                                             0.0s
 => [1/7] FROM docker.io/library/node:20@sha256:844b41cf784f66d7920fd673f7af54ca7b81e289985edc6cd864e7d05e0d133c                0.0s
 => CACHED [2/7] WORKDIR /app                                                                                                   0.0s
 => CACHED [3/7] COPY package*.json ./                                                                                          0.0s
 => CACHED [4/7] RUN npm install                                                                                                0.0s
 => [5/7] COPY . .                                                                                                              0.1s
 => [6/7] RUN npx prisma generate                                                                                               3.1s
 => [7/7] RUN npm run build                                                                                                    23.5s
 => exporting to image                                                                                                          0.4s
 => => exporting layers                                                                                                         0.4s
 => => writing image sha256:7ef3cad56257d67031ef4e6e3e963401b9c102357b749a897f40513a9f3a7dfd                                    0.0s
 => => naming to docker.io/library/auth-login:latest                                                                            0.0s

View build details: docker-desktop://dashboard/build/default/default/6tymnjgdpfpmncxa5yho2u259

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker push himanshunanda22/project-6:auth-login
The push refers to repository [docker.io/himanshunanda22/project-6]
9223bcf8d8b4: Layer already exists
40afb7066e00: Layer already exists
b1f8280ea5b9: Layer already exists
44fa5c5c9bba: Layer already exists
14c2694f88e3: Layer already exists
72702b05020d: Layer already exists
123194e8ca81: Layer already exists
a28e61f75fa0: Layer already exists
41c2d1f0a1d3: Layer already exists
3e81cc85b636: Layer already exists
893507f6057f: Layer already exists
2353f7120e0e: Layer already exists
51a9318e6edf: Layer already exists
c5bb35826823: Layer already exists
auth-login: digest: sha256:f422a08eab1c84b6899d5bf7bfa6ea2d8a23ad0912c1edacb0e3e2da3515e43e size: 3265
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```dockerfile
# Use an appropriate base image for Node.js applications
FROM node:20

# Set the working directory inside the container
WORKDIR /app

# Copy package.json and package-lock.json to the working directory
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code to the working directory
COPY . .

# Run prisma generate
RUN npx prisma generate

# Build the Next.js application
RUN npm run build

# Expose the port the app runs on (assuming it's 3000)
EXPOSE 3000

# Command to run the application
CMD ["npm", "start"]
```

Product-id

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker build -t products-id:latest -f app/api/products/[id]/Dockerfile .
[+] Building 0.0s (0/0)  docker:default
[+] Building 31.5s (13/13) FINISHED                                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                                          0.0s
 => => transferring dockerfile: 800B                                                                                          0.0s
 => [internal] load metadata for docker.io/library/node:20                                                                    4.5s
 => [auth] library/node:pull token for registry-1.docker.io                                                                   0.0s
 => [internal] load .dockerignore                                                                                             0.0s
 => => transferring context: 2B                                                                                               0.0s
 => [1/7] FROM docker.io/library/node:20@sha256:844b41cf784f66d7920fd673f7af54ca7b81e289985edc6cd864e7d05e0d133c               0.0s
 => [internal] load build context                                                                                             0.1s
 => => transferring context: 389.08kB                                                                                         0.0s
 => CACHED [2/7] WORKDIR /app                                                                                                 0.0s
 => CACHED [3/7] COPY package*.json ./                                                                                        0.0s
 => CACHED [4/7] RUN npm install                                                                                              0.0s
 => [5/7] COPY . .                                                                                                            0.1s
 => [6/7] RUN npx prisma generate                                                                                             3.8s
 => [7/7] RUN npm run build                                                                                                  22.5s
 => exporting to image                                                                                                        0.2s
 => => exporting layers                                                                                                       0.2s
 => => writing image sha256:abcad492d9e2f10f00de3b8c75f0f575528cfdc5246d1b26887f04ade3a2cab0                                  0.0s
 => => naming to docker.io/library/products-id:latest                                                                         0.0s

View build details: docker-desktop://dashboard/build/default/default/c6nn4y0rhc4x1a22ky42x7ekd

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker tag products-id himanshunanda22/project-6:products-id
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker push himanshunanda22/project-6:products-id
The push refers to repository [docker.io/himanshunanda22/project-6]
ccf2a298688f: Pushed
52b87d3e6685: Pushed
0536088158a5: Pushed
44fa5c5c9bba: Layer already exists
14c2694f88e3: Layer already exists
72702b05020d: Layer already exists
123194e8ca81: Layer already exists
a28e61f75fa0: Layer already exists
41c2d1f0a1d3: Layer already exists
3e81cc85b636: Layer already exists
893507f6057f: Layer already exists
2353f7120e0e: Layer already exists
51a9318e6edf: Layer already exists
c5bb35826823: Layer already exists
products-id: digest: sha256:38dff88e29a9e82f143f0b6318df288906eeca14be5d861417e4f76a1142853b size: 3266
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```dockerfile
# Use an appropriate base image for Node.js applications
FROM node:20

# Set the working directory inside the container
WORKDIR /app

# Copy package.json and package-lock.json to the working directory
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code to the working directory
COPY . .

# Run prisma generate (if you are using Prisma in this microservice)
# If you are not using Prisma, you can remove this line
RUN npx prisma generate

# Build the application (if there is a build step, adjust the command accordingly)
RUN npm run build

# Expose the port the app runs on (assuming it's 3000, customize as needed)
EXPOSE 3000

# Command to run the application
CMD ["npm", "start"]
```

Product-user

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker build -t products-user:latest -f app/api/products/
user/Dockerfile .
[+] Building 0.0s (0/0)  docker:default
[+] Building 29.8s (13/13) FINISHED                                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                                             0.0s
 => => transferring dockerfile: 800B                                                                                             0.0s
 => [internal] load metadata for docker.io/library/node:20                                                                       3.7s
 => [auth] library/node:pull token for registry-1.docker.io                                                                      0.0s
 => [internal] load .dockerignore                                                                                                0.0s
 => => transferring context: 2B                                                                                                  0.0s
 => [1/7] FROM docker.io/library/node:20@sha256:844b41cf784f66d7920fd673f7af54ca7b81e289985edc6cd864e7d05e0d133c                 0.0s
 => [internal] load build context                                                                                                0.1s
 => => transferring context: 903.80kB                                                                                            0.1s
 => CACHED [2/7] WORKDIR /app                                                                                                    0.0s
 => CACHED [3/7] COPY package*.json ./                                                                                           0.0s
 => CACHED [4/7] RUN npm install                                                                                                 0.0s
 => [5/7] COPY . .                                                                                                               0.1s
 => [6/7] RUN npx prisma generate                                                                                                3.3s
 => [7/7] RUN npm run build                                                                                                     22.2s
 => exporting to image                                                                                                           0.2s
 => => exporting layers                                                                                                          0.2s
 => => writing image sha256:a23127238b2929d56c2105c8e7d6bc240d2f1e0e39c18600a87a72068150a6b9                                     0.0s
 => => naming to docker.io/library/products-user:latest                                                                          0.0s

View build details: docker-desktop://dashboard/build/default/default/xy88x32vdta9rohekw837phzi

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker tag products-user himanshunanda22/project-6:produc
ts-user
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker push himanshunanda22/project-6:products-user
The push refers to repository [docker.io/himanshunanda22/project-6]
b03e73944ffc: Pushed
471ea366c789: Pushed
426cfa323b08: Pushed
44fa5c5c9bba: Layer already exists
14c2694f88e3: Layer already exists
72702b05020d: Layer already exists
123194e8ca81: Layer already exists
a28e61f75fa0: Pushed
41c2d1f0a1d3: Pushed
3e81cc85b636: Layer already exists
893507f6057f: Layer already exists
2353f7120e0e: Layer already exists
51a9318e6edf: Layer already exists
c5bb35826823: Layer already exists
products-user: digest: sha256:5c086b97859927b878e7436c7024f46c15150e3421f914c72e0eae28b5bd93ec size: 3266
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```
# Use an appropriate base image for Node.js applications
FROM node:20

# Set the working directory inside the container
WORKDIR /app

# Copy package.json and package-lock.json to the working directory
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code to the working directory
COPY . .

# Run prisma generate (if you are using Prisma in this microservice)
# If you are not using Prisma, you can remove this line
RUN npx prisma generate

# Build the application (if there is a build step, adjust the command accordingly)
RUN npm run build

# Expose the port the app runs on (assuming it's 3000, customize as needed)
EXPOSE 3000

# Command to run the application
CMD ["npm", "start"]
```

User

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker build -t user:latest -f app/api/User/[id]/Dockerfi
le .
[+] Building 36.1s (13/13) FINISHED                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                                   0.1s
 => => transferring dockerfile: 800B                                                                                   0.0s
 => [internal] load metadata for docker.io/library/node:20                                                             2.1s
 => [auth] library/node:pull token for registry-1.docker.io                                                            0.0s
 => [internal] load .dockerignore                                                                                      0.0s
 => => transferring context: 2B                                                                                        0.0s
 => [1/7] FROM docker.io/library/node:20@sha256:844b41cf784f66d7920fd673f7af54ca7b81e289985edc6cd864e7d05e0d133c      0.0s
 => [internal] load build context                                                                                      0.1s
 => => transferring context: 701.53kB                                                                                  0.1s
 => CACHED [2/7] WORKDIR /app                                                                                          0.0s
 => CACHED [3/7] COPY package*.json ./                                                                                 0.0s
 => CACHED [4/7] RUN npm install                                                                                       0.0s
 => [5/7] COPY . .                                                                                                     0.2s
 => [6/7] RUN npx prisma generate                                                                                      3.0s
 => [7/7] RUN npm run build                                                                                           30.1s
 => exporting to image                                                                                                 0.4s
 => => exporting layers                                                                                                0.4s
 => => writing image sha256:cdb5637ba5ace1f6b0ea014605435fe979b4f082e7bc8c76f134caf212d2c18a                           0.0s
 => => naming to docker.io/library/user:latest                                                                         0.0s

View build details: docker-desktop://dashboard/build/default/default/nskjiqwvk4e0m68cs7q1ph8zb
```

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker tag user himanshunanda22/project-6:user
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> docker push himanshunanda22/project-6:user
The push refers to repository [docker.io/himanshunanda22/project-6]
c1fda07a4cc6: Pushed
2ba7517200a3: Pushed
b31b21c8b3b1: Pushed
44fa5c5c9bba: Layer already exists
14c2694f88e3: Layer already exists
72702b05020d: Layer already exists
123194e8ca81: Layer already exists
a28e61f75fa0: Layer already exists
41c2d1f0a1d3: Layer already exists
3e81cc85b636: Layer already exists
893507f6057f: Layer already exists
2353f7120e0e: Layer already exists
51a9318e6edf: Layer already exists
c5bb35826823: Layer already exists
user: digest: sha256:0d9575390b9f5f0102882e9870fc992048e3ab2806e0a5adf2cf55d60ab138fc size: 3266
```

```dockerfile
# Use an appropriate base image for Node.js applications
FROM node:20

# Set the working directory inside the container
WORKDIR /app

# Copy package.json and package-lock.json to the working directory
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code to the working directory
COPY . .

# Run prisma generate (if you are using Prisma in this microservice)
# If you are not using Prisma, you can remove this line
RUN npx prisma generate

# Build the application (if there is a build step, adjust the command accordingly)
RUN npm run build

# Expose the port the app runs on (assuming it's 3000, customize as needed)
EXPOSE 3000

# Command to run the application
CMD ["npm", "start"]
```

Kubernetes

Orchestration

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl apply -f kubernetes/auth/login/auth-login-deployment.yaml
deployment.apps/auth-login-deployment created
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl apply -f kubernetes/auth/login/auth-login-service.yaml
service/auth-login-service created
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get pods
NAME                                    READY   STATUS    RESTARTS   AGE
auth-login-deployment-7d44fdb7d9-4hkzf   1/1     Running   0          30s
auth-login-deployment-7d44fdb7d9-gh55s   1/1     Running   0          30s
auth-login-deployment-7d44fdb7d9-klrw2   1/1     Running   0          30s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get deployments
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
auth-login-deployment   3/3     3            3           34s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get services
NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
auth-login-service   ClusterIP   10.104.82.233   <none>        80/TCP    32s
kubernetes           ClusterIP   10.96.0.1       <none>        443/TCP   68d
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl apply -f kubernetes/products/[id]/products-id-deployment.yaml
deployment.apps/products-id-deployment created
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl apply -f kubernetes/products/[id]/products-id-service.yaml
service/products-id-service created
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get pods
NAME                                    READY   STATUS    RESTARTS   AGE
auth-login-deployment-7d44fdb7d9-4hkzf   1/1     Running   0          14m
auth-login-deployment-7d44fdb7d9-gh55s   1/1     Running   0          14m
auth-login-deployment-7d44fdb7d9-klrw2   1/1     Running   0          14m
products-id-deployment-6f96dcbb47-j2x9c  1/1     Running   0          11s
products-id-deployment-6f96dcbb47-mxmkc  1/1     Running   0          11s
products-id-deployment-6f96dcbb47-t4lp6  1/1     Running   0          11s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get deployments
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
auth-login-deployment   3/3     3            3           14m
products-id-deployment  3/3     3            3           16s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get services
NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
auth-login-service   ClusterIP   10.104.82.233   <none>        80/TCP    14m
kubernetes           ClusterIP   10.96.0.1       <none>        443/TCP   68d
products-id-service  ClusterIP   10.100.243.55   <none>        80/TCP    14s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl apply -f kubernetes/products/user/products-user-deployment.yaml
deployment.apps/products-user-deployment configured
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl apply -f kubernetes/products/user/products-user-service.yaml
service/products-user-service unchanged
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get pods
NAME                                    READY   STATUS    RESTARTS   AGE
auth-login-deployment-7d44fdb7d9-4hkzf   1/1     Running   0          38m
auth-login-deployment-7d44fdb7d9-gh55s   1/1     Running   0          38m
auth-login-deployment-7d44fdb7d9-klrw2   1/1     Running   0          38m
products-id-deployment-6f96dcbb47-j2x9c  1/1     Running   0          24m
products-id-deployment-6f96dcbb47-mxmkc  1/1     Running   0          24m
products-id-deployment-6f96dcbb47-t4lp6  1/1     Running   0          24m
products-user-deployment-5c7ddb8f8f-cn45k  1/1     Running   0          8s
products-user-deployment-5c7ddb8f8f-jzglp  1/1     Running   0          9s
products-user-deployment-5c7ddb8f8f-vjnhf  1/1     Running   0          10s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get services
NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
auth-login-service   ClusterIP   10.104.82.233   <none>        80/TCP    38m
kubernetes           ClusterIP   10.96.0.1       <none>        443/TCP   68d
products-id-service  ClusterIP   10.100.243.55   <none>        80/TCP    24m
products-user-service  ClusterIP   10.108.30.127   <none>        80/TCP    2m
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get deployments
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
auth-login-deployment   3/3     3            3           38m
products-id-deployment  3/3     3            3           24m
products-user-deployment  3/3     3            3           2m7s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

```
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl apply -f kubernetes/User/User-deployment.yaml
deployment.apps/user-deployment unchanged
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl apply -f kubernetes/User/User-service.yaml
service/user-service unchanged
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get pods
NAME                                    READY   STATUS    RESTARTS   AGE
auth-login-deployment-7d44fdb7d9-4hkzf   1/1     Running   0          47m
auth-login-deployment-7d44fdb7d9-gh55s   1/1     Running   0          47m
auth-login-deployment-7d44fdb7d9-klrw2   1/1     Running   0          47m
products-id-deployment-6f96dcbb47-j2x9c  1/1     Running   0          33m
products-id-deployment-6f96dcbb47-mxmkc  1/1     Running   0          33m
products-id-deployment-6f96dcbb47-t4lp6  1/1     Running   0          33m
products-user-deployment-5c7ddb8f8f-cn45k  1/1     Running   0          9m35s
products-user-deployment-5c7ddb8f8f-jzglp  1/1     Running   0          9m36s
products-user-deployment-5c7ddb8f8f-vjnhf  1/1     Running   0          9m37s
user-deployment-84866f86d8-fcpkk         1/1     Running   0          76s
user-deployment-84866f86d8-jhmf8         1/1     Running   0          77s
user-deployment-84866f86d8-zwpj7         1/1     Running   0          77s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get deployments
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
auth-login-deployment   3/3     3            3           47m
products-id-deployment  3/3     3            3           33m
products-user-deployment  3/3     3            3           11m
user-deployment         3/3     3            3           80s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2> kubectl get services
NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
auth-login-service   ClusterIP   10.104.82.233   <none>        80/TCP    47m
kubernetes           ClusterIP   10.96.0.1       <none>        443/TCP   68d
products-id-service  ClusterIP   10.100.243.55   <none>        80/TCP    33m
products-user-service  ClusterIP   10.108.30.127   <none>        80/TCP    11m
user-service         ClusterIP   10.103.250.21   <none>        80/TCP    94s
PS C:\Users\himan\OneDrive\Desktop\Sem -VI Lab\CC\Project\PES2UG21CS461_463_915_917_Project-6\Week -2>
```

Code

User

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: user-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: user
  template:
    metadata:
      labels:
        app: user
    spec:
      containers:
        - name: user
          image: kushshriv/project-6:user
          ports:
            - containerPort: 3000
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: user-service
spec:
  selector:
    app: user
  ports:
    - protocol: TCP
      port: 80
      targetPort: 30u00
```

Auth-login

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: auth-login-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: auth-login
  template:
    metadata:
      labels:
        app: auth-login
    spec:
      containers:
        - name: auth-login
          image: kushshriv/project-6:auth-login
          ports:
            - containerPort: 3000
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: auth-login-service
spec:
  selector:
    app: auth-login
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
```

Products-id

```yaml
kind: Deployment
metadata:
  name: products-id-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: products-id
  template:
    metadata:
      labels:
        app: products-id
    spec:
      containers:
        - name: products-id
          image: kushshriv/project-6:products-id
          ports:
            - containerPort: 3000
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: products-id-service
spec:
  selector:
    app: products-id
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
```

Products-User

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: products-user-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: products-user
  template:
    metadata:
      labels:
        app: products-user
    spec:
      containers:
        - name: products-user
          image: kushshriv/project-6:products-user
          ports:
            - containerPort: 3000
```

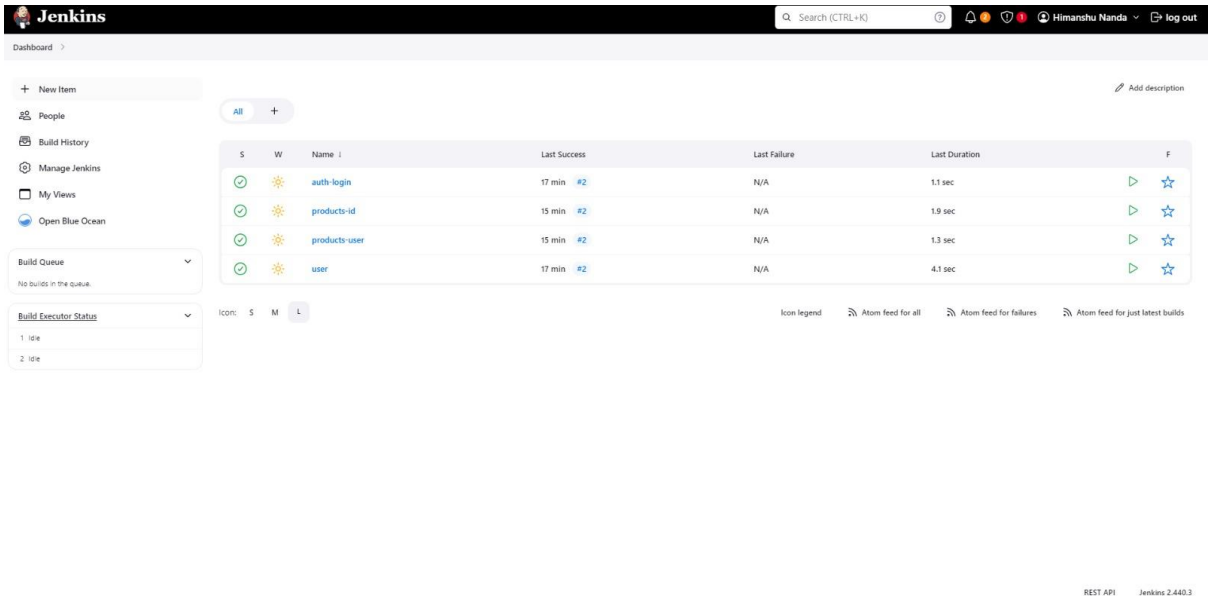
```yaml
apiVersion: v1
kind: Service
metadata:
  name: products-user-service
spec:
  selector:
    app: products-user
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
```

Week 3

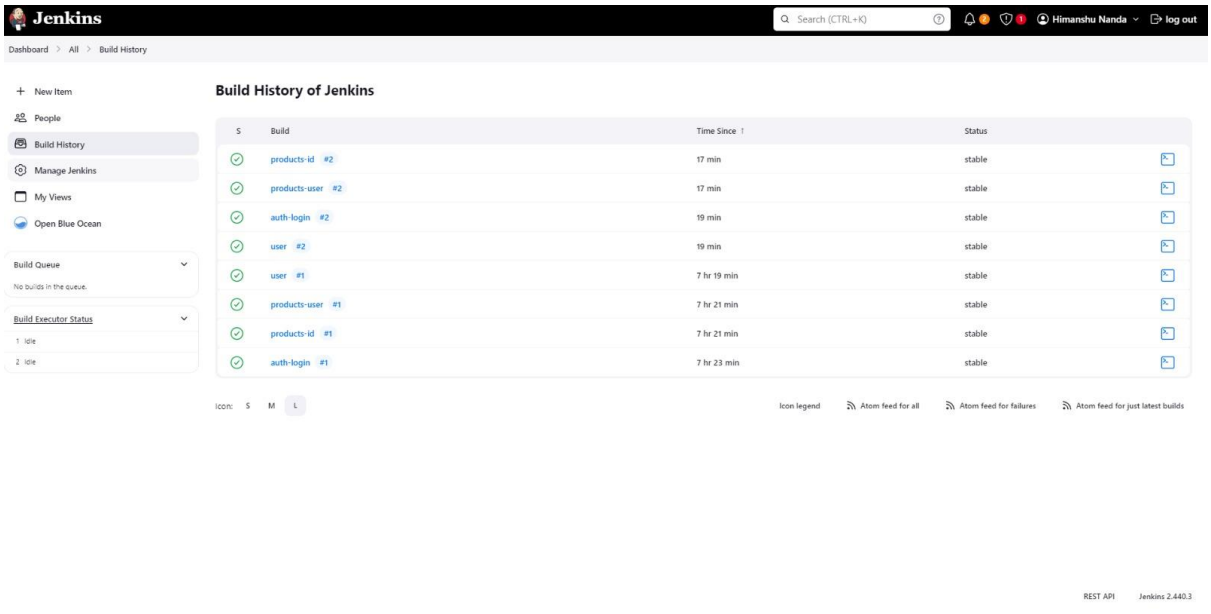
Continuous Integration and Deployment

Overview

Dashboard

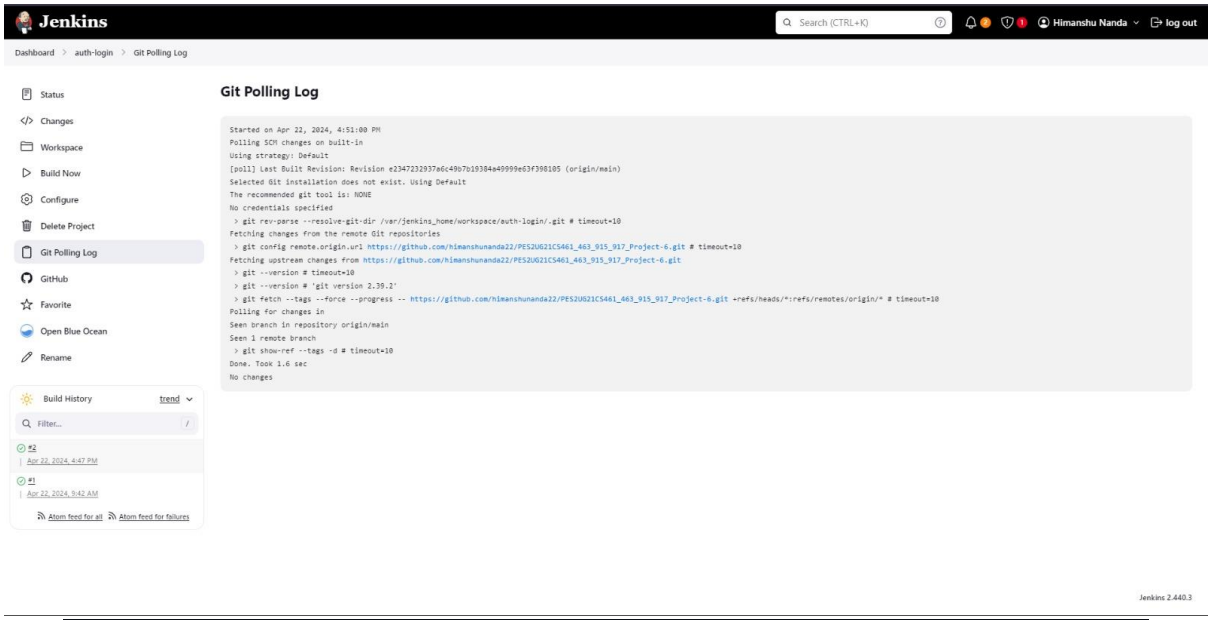


Build History



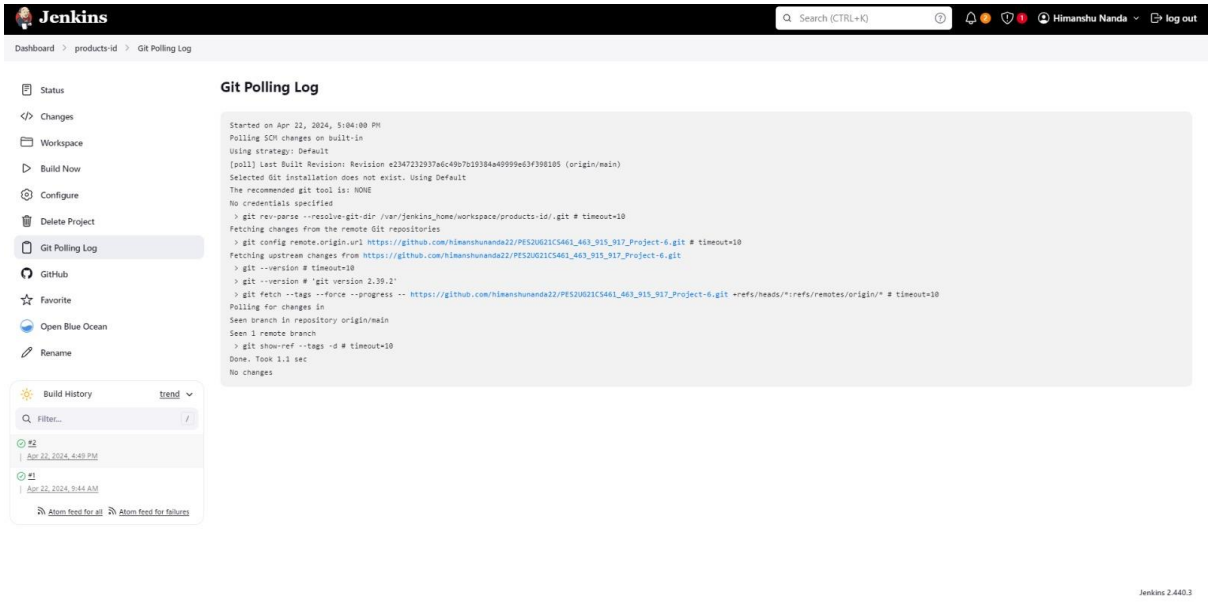
Files

Auth



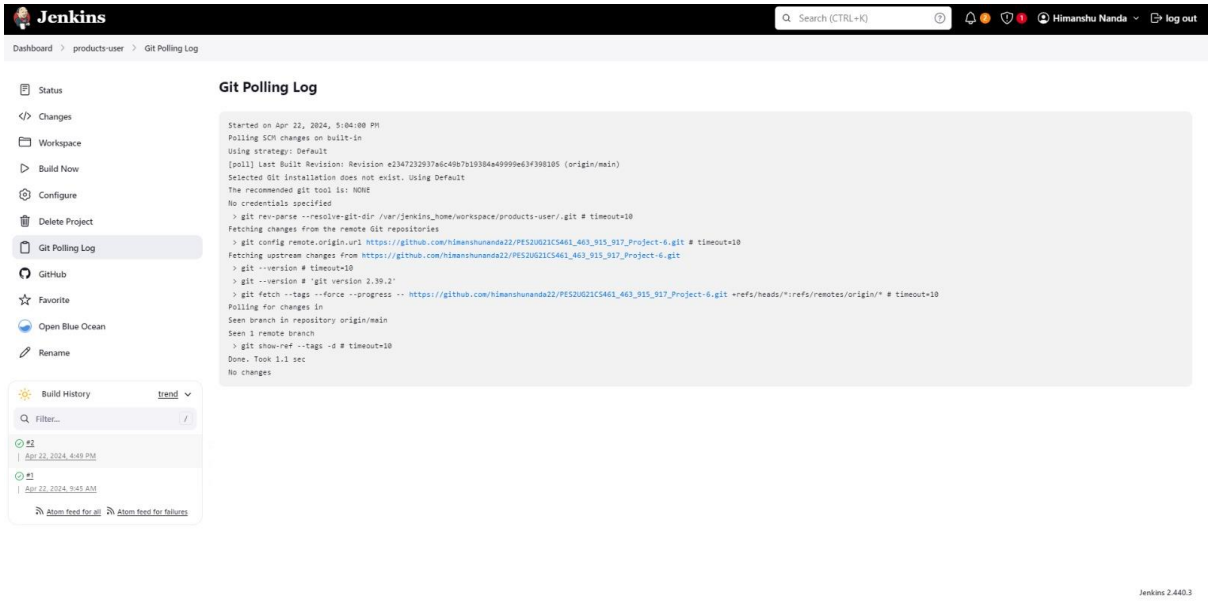

```groovy
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                checkout scm
            }
        }

        stage('Build') {
            steps {
                sh 'docker build -t auth-login:latest -f app/api/auth/login/Dockerfile .'
            }
        }

        stage('Test') {
            steps {
                // Add commands to run tests for the `auth` microservice
                sh 'npm run test'
            }
        }

        stage('Deploy') {
            steps {
                sh 'kubectl apply -f kubernetes/auth/auth-deployment.yaml'
                sh 'kubectl apply -f kubernetes/auth/auth-service.yaml'
            }
        }
    }
}
```

Products-id



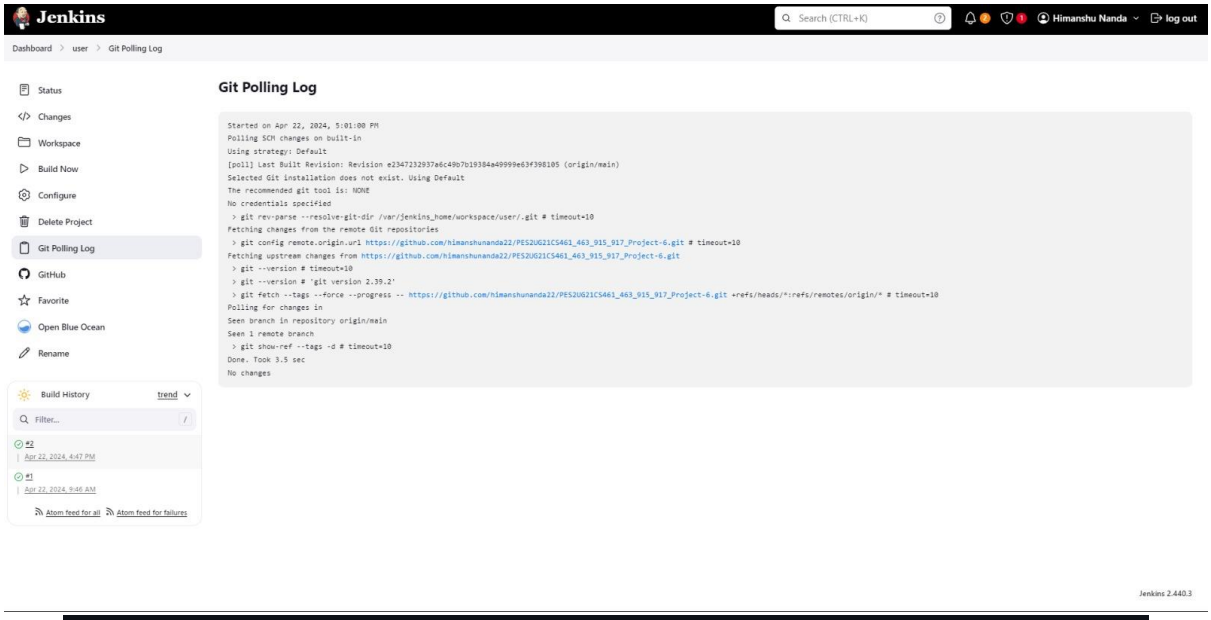

```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                checkout scm
            }
        }

        stage('Build') {
            steps {
                sh 'docker build -t products-id:latest -f app/api/products/[id]/Dockerfile .'
            }
        }

        stage('Test') {
            steps {
                // Add commands to run tests for the `products` microservice
                sh 'npm run test'
            }
        }

        stage('Deploy') {
            steps {
                sh 'kubectl apply -f kubernetes/products/[id]/products-id-deployment.yaml'
                sh 'kubectl apply -f kubernetes/products/[id]/products-id-service.yaml'
            }
        }
    }
}
```

Products-user




```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                checkout scm
            }
        }

        stage('Build') {
            steps {
                sh 'docker build -t products-user:latest -f app/api/products/user/Dockerfile .'
            }
        }

        stage('Test') {
            steps {
                // Add commands to run tests for the `products` microservice
                sh 'npm run test'
            }
        }

        stage('Deploy') {
            steps {
                sh 'kubectl apply -f kubernetes/products/user/products-user-deployment.yaml'
                sh 'kubectl apply -f kubernetes/products/user/products-user-service.yaml'
            }
        }
    }
}
```

User




```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                checkout scm
            }
        }

        stage('Build') {
            steps {
                sh 'docker build -t user:latest -f app/api/User/[id]/Dockerfile .'
            }
        }

        stage('Test') {
            steps {
                // Add commands to run tests for the `user` microservice
                sh 'npm run test'
            }
        }

        stage('Deploy') {
            steps {
                sh 'kubectl apply -f kubernetes/User/User-deployment.yaml'
                sh 'kubectl apply -f kubernetes/User/User-service.yaml'
            }
        }
    }
}
```