## 7. Develop a simple containerized application using Docker.

### Step 1: First we need to write a DockerFile

```
→  devopslab git:(main) ✗ cat Dockerfile
FROM nginx:latest
COPY index.html /usr/share/nginx/html/
COPY regform.css /usr/share/nginx/html/
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

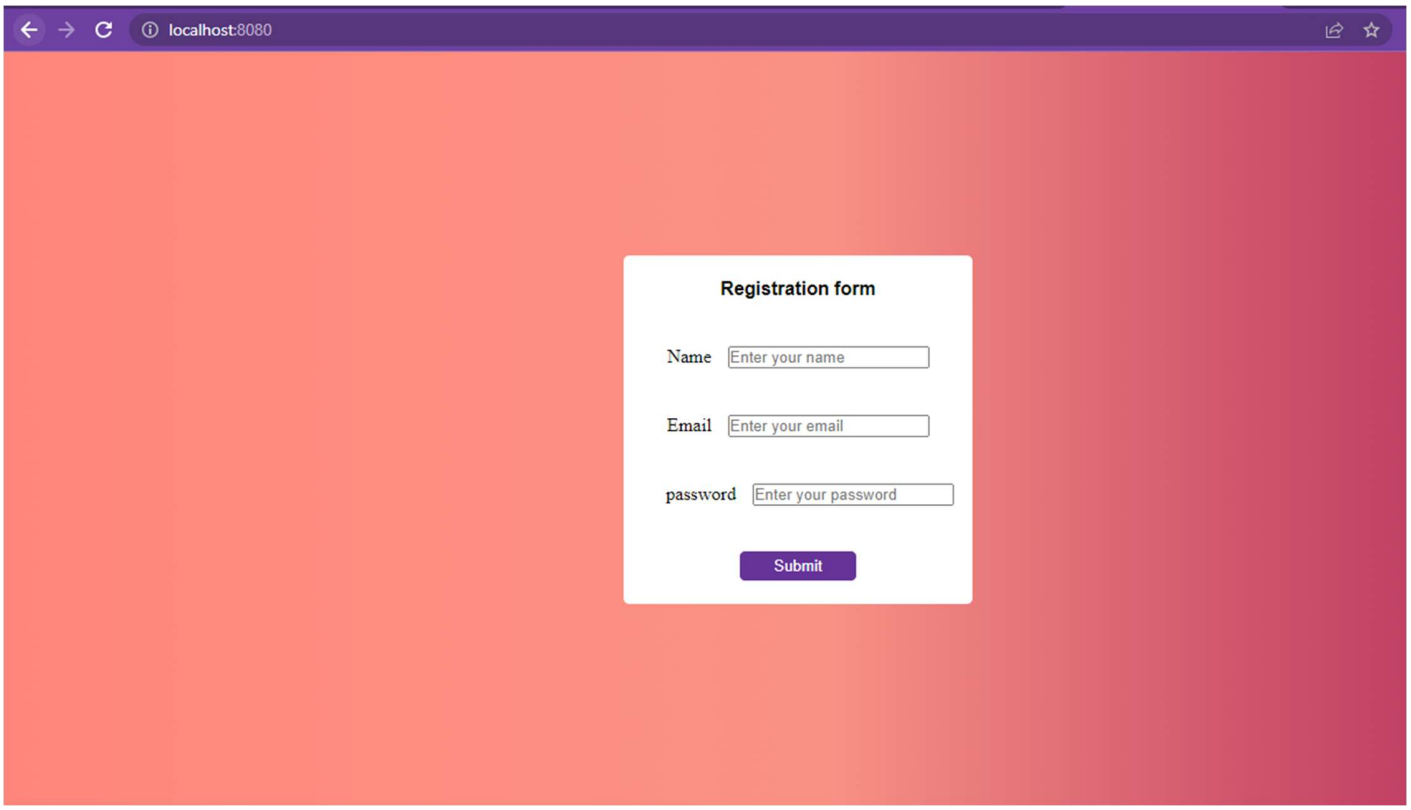
### Step 2: Build a docker image

```
→  devopslab git:(main) ✗ docker build -t mywebapp:latest .
[+] Building 0.3s (8/8) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 38B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/nginx:latest
 => [1/3] FROM docker.io/library/nginx:latest
 => [internal] load build context
 => => transferring context: 63B
 => CACHED [2/3] COPY index.html /usr/share/nginx/html/
 => CACHED [3/3] COPY regform.css /usr/share/nginx/html/
 => exporting to image
 => => exporting layers
 => => writing image sha256:45d24590e79d92ccdb7dd37f930e3f368e536b111fa55c3ac97e4317fdb693e0
 => => naming to docker.io/library/mywebapp:latest
→  devopslab git:(main) ✗ docker run -p mywebapp 8080:80
docker: Invalid containerPort: mywebapp.
See 'docker run --help'.
→  devopslab git:(main) ✗ docker run -p mywebapp 8081:80
docker: Invalid containerPort: mywebapp.
See 'docker run --help'.
→  devopslab git:(main) ✗ docker run -p 8080:80 mywebapp
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
```

### Step 3: Running the container in our local environment

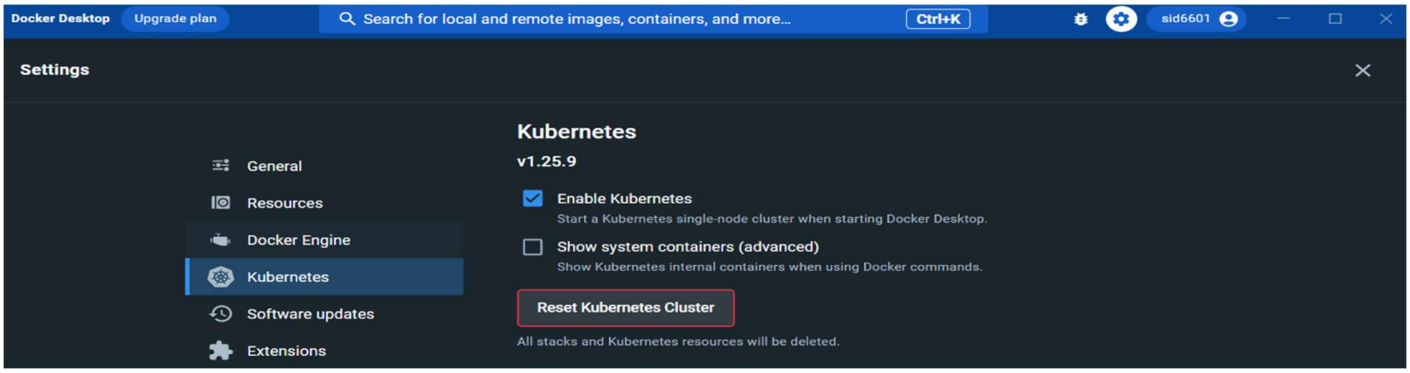
```
→  devopslab git:(main) ✗ docker run -p mywebapp 8081:80
docker: Invalid containerPort: mywebapp.
See 'docker run --help'.
→  devopslab git:(main) ✗ docker run -p 8080:80 mywebapp
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/07/05 12:33:54 [notice] 1#1: using the "epoll" event method
2023/07/05 12:33:54 [notice] 1#1: nginx/1.25.1
2023/07/05 12:33:54 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2023/07/05 12:33:54 [notice] 1#1: OS: Linux 5.15.90.1-microsoft-standard-WSL2
2023/07/05 12:33:54 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/07/05 12:33:54 [notice] 1#1: start worker processes
2023/07/05 12:33:54 [notice] 1#1: start worker process 29
2023/07/05 12:33:54 [notice] 1#1: start worker process 30
2023/07/05 12:33:54 [notice] 1#1: start worker process 31
2023/07/05 12:33:54 [notice] 1#1: start worker process 32
2023/07/05 12:33:54 [notice] 1#1: start worker process 33
2023/07/05 12:33:54 [notice] 1#1: start worker process 34
2023/07/05 12:33:54 [notice] 1#1: start worker process 35
2023/07/05 12:33:54 [notice] 1#1: start worker process 36
```

**Output:**



## 8. Integrate Kubernetes and Docker

- Integrate Kubernetes and Docker. In the settings menu of docker desktop application, go to the kubernetes bar and select “Enable Kubernetes” option.




- Verify Kubernetes installation: After Docker Desktop restarts, open a command prompt or PowerShell window and run the command `kubectl version` to verify that Kubernetes is running correctly.

```
PS C:\Users\A Siddharth> kubectl version
WARNING: This version information is deprecated and will be replaced with the output from kubectl version --short.  Use --output=yaml|json to get the full v
ersion.
Client Version: version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.9", GitCommit:"a1a87a0a2bcd605820920c6b0e618a8ab7d117d4", GitTreeState:"clean", BuildD
ate:"2023-04-12T12:16:51Z", GoVersion:"go1.19.8", Compiler:"gc", Platform:"windows/amd64"}
Kustomize Version: v4.5.7
Server Version: version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.9", GitCommit:"a1a87a0a2bcd605820920c6b0e618a8ab7d117d4", GitTreeState:"clean", BuildD
ate:"2023-04-12T12:08:36Z", GoVersion:"go1.19.8", Compiler:"gc", Platform:"linux/amd64"}
```

- Build Docker image: Use the `docker build` command to build the image. For example: `docker build -t your-image-name:tag .`

```
PS C:\Users\A Siddharth\nginx> docker run -p 8081:80 sid6601/my-website:latest
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
```

- Push Docker images to a registry: To store and distribute your Docker images, you can use Docker Hub or any other container registry. Log in to Docker Hub using the `docker login` command. Then, tag your local Docker image with the registry URL and push it using the `docker push` command. For example: `docker push your-username/your-image-name:tag`

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                   pwsh
2023/07/05 09:16:54 [notice] 31#31: exit
PS C:\Users\A Siddharth\nginx> docker push sid6601/my-website:latest
The push refers to repository [docker.io/sid6601/my-website]
1353252bf906: Pushed
fbdce5ec2c29: Pushed
bdea7c663e86: Mounted from sid6601/my-website-image
1b22827e15b4: Mounted from sid6601/my-website-image
d9f50eaf56fa: Mounted from sid6601/my-website-image
2530717ff0bb: Mounted from sid6601/my-website-image
e7766bc830a8: Mounted from sid6601/my-website-image
cb411529b86f: Mounted from sid6601/my-website-image
bc09720137db: Mounted from sid6601/my-website-image
3dab9f8bf2d2: Mounted from sid6601/my-website-image
latest: digest: sha256:0adfae778a8782e582315d1e7bf89a1aaf0881c85befd8b919f900f017f2d27f size: 2403
```

- Create Kubernetes manifests: Write Kubernetes manifests in YAML or JSON format to describe your application deployments or pods.Specify the Docker image you pushed to the container registry in the manifest files.

my-website-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-website-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-website
  template:
    metadata:
      labels:
        app: my-website
    spec:
      containers:
        - name: my-website-container
          image: sid6601/my-website:latest
          ports:
            - containerPort: 80
```

- Apply Kubernetes manifests: Use the `kubectl apply` command to apply the Kubernetes manifests and create the desired resources. For example: `kubectl apply -f your-manifest-file.yaml`. Kubernetes will create the necessary objects and schedule the Docker containers on the cluster.

```
PS C:\Users\A Siddharth\nginx> code my-website-deployment.yaml
PS C:\Users\A Siddharth\nginx> kubectl apply -f my-website-deployment.yaml
deployment.apps/my-website-deployment created
service/my-website-service created
PS C:\Users\A Siddharth\nginx>
```

- Monitor and manage the cluster: You can use the `kubectl` command-line tool to monitor and manage your Kubernetes cluster. Run commands like `kubectl get pods`, `kubectl get deployments`, and `kubectl logs` to inspect the cluster's state and troubleshoot any issues.

```
PS C:\Users\A Siddharth\nginx> kubectl get pods
NAME                                    READY   STATUS    RESTARTS   AGE
my-website-deployment-85fb464449-6sfwz  1/1     Running   0          5m13s
PS C:\Users\A Siddharth\nginx> kubectl get deployments
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
my-website-deployment  1/1     1            1           5m36s
PS C:\Users\A Siddharth\nginx>
```

## 9. Automate the process of running containerized application developed in exercise 7 using Kubernetes.

- Build the docker image:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                                    pwsh
PS C:\Users\A Siddharth\nginx> docker build -t sid6601/regwebsite .
[+] Building 3.0s (9/9) FINISHED
 => [internal] load .dockerignore                                                                  0.0s
 => => transferring context: 2B                                                                    0.0s
 => [internal] load build definition from Dockerfile                                               0.0s
 => => transferring dockerfile: 189B                                                               0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                    2.9s
 => [auth] library/nginx:pull token for registry-1.docker.io                                       0.0s
 => [1/3] FROM docker.io/library/nginx:alpine@sha256:2d194184b067db3598771b4cf326cfe6ad5051937ba1132b8b7d4b0184e0d0a6  0.0s
 => [internal] load build context                                                                  0.0s
 => => transferring context: 62B                                                                   0.0s
 => CACHED [2/3] COPY index.html /usr/share/nginx/html/                                            0.0s
 => CACHED [3/3] COPY styles.css /usr/share/nginx/html/                                            0.0s
 => exporting to image                                                                             0.0s
 => => exporting layers                                                                            0.0s
 => => writing image sha256:11869dfa70f22ad3c8e9ae6b1e57c7738150ac9fee493ddbafaacbb007ac309e       0.0s
 => => naming to docker.io/sid6601/regwebsite                                                      0.0s
PS C:\Users\A Siddharth\nginx>
```

- Push it to docker hub (make sure you have an account)

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                                    pwsh
PS C:\Users\A Siddharth\nginx> docker push sid6601/regwebsite
Using default tag: latest
The push refers to repository [docker.io/sid6601/regwebsite]
1353252bf906: Pushed
fbdce5ec2c29: Pushed
bdea7c663e86: Mounted from sid6601/my-website-image
1b22827e15b4: Pushed
d9f50eaf56fa: Pushed
2530717ff0bb: Pushed
e7766bc830a8: Pushed
cb411529b86f: Pushed
bc09720137db: Mounted from sid6601/my-website-image
3dab9f8bf2d2: Mounted from sid6601/my-website-image
latest: digest: sha256:0adfae778a8782e582315d1e7bf89a1aaf0881c85befd8b919f900f017f2d27f size: 2403
PS C:\Users\A Siddharth\nginx>
```

- Create a infra/k8s folder and put client.yaml in it with the following code:

my-website-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-website-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-website
  template:
    metadata:
      labels:
        app: my-website
    spec:
      containers:
        - name: my-website-container
          image: sid6601/my-website:latest
          ports:
            - containerPort: 80
```

- Make sure to enable Kubernetes from docker desktop:

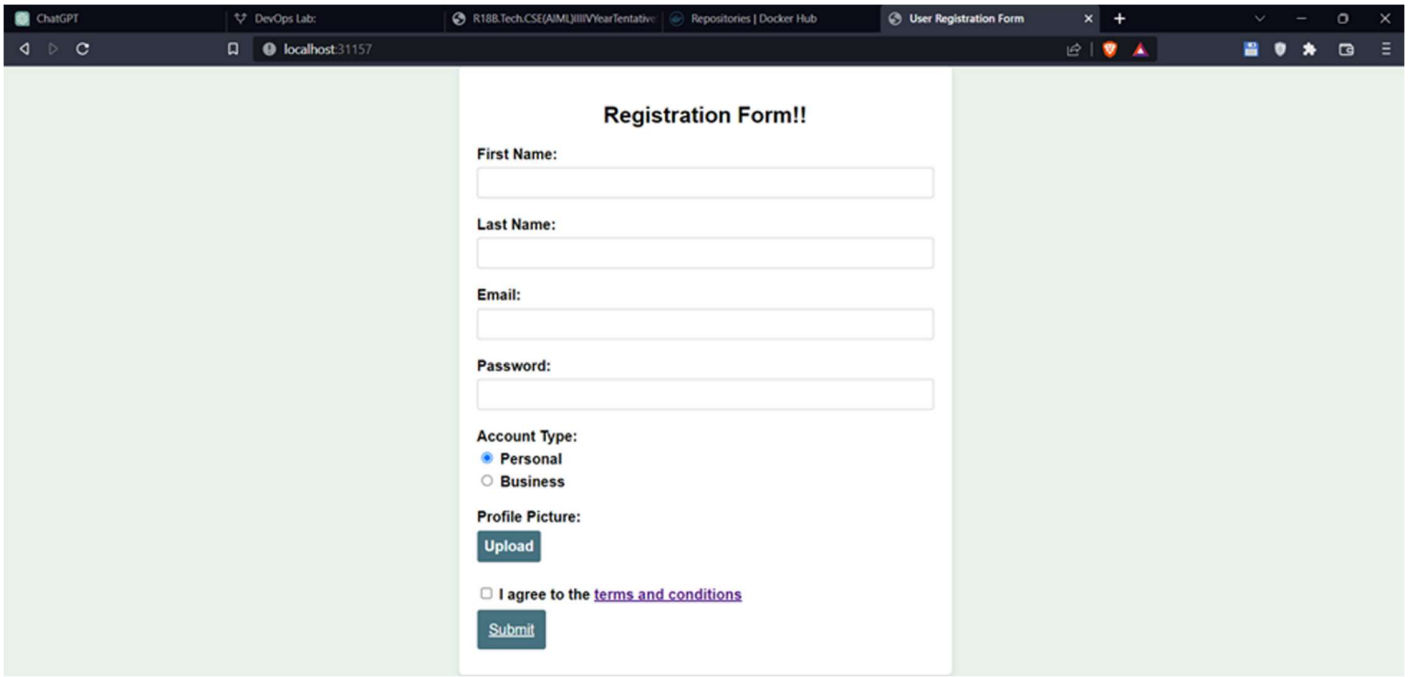

- Create the deployment:

```
PS C:\Users\A Siddharth\nginx\infra\k8s> kubectl apply -f client.yaml
deployment.apps/client-depl created
service/client-srv created
PS C:\Users\A Siddharth\nginx\infra\k8s>
```

- Check the pods and services:Output: Get the target port from client-srv and access that port:

```
PS C:\Users\A Siddharth\nginx\infra\k8s> kubectl get pods
NAME                                READY   STATUS             RESTARTS   AGE
client-depl-8cdc87498-skvbv         1/1     Running            0          2m16s
nginx-deployment-674b674dc8-5wnkm   0/1     ImagePullBackOff   2          36m
nginx-deployment-674b674dc8-9kzch   0/1     ImagePullBackOff   2          36m
nginx-deployment-674b674dc8-l2drz   0/1     ImagePullBackOff   2          36m
PS C:\Users\A Siddharth\nginx\infra\k8s> kubectl get services
NAME                TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
client-srv          NodePort       10.106.58.193   <none>        3000:31157/TCP   2m22s
kubernetes          ClusterIP      10.96.0.1       <none>        443/TCP          102m
my-website-service  LoadBalancer   10.110.35.241   localhost     80:31166/TCP     74m
nginx-service       LoadBalancer   10.110.97.237   <pending>     80:30196/TCP     35m
PS C:\Users\A Siddharth\nginx\infra\k8s>
```

- Output:Get the target port from client-srv and access that port:



**1. Install and Explore Selenium for automated testing.**

Installing chromedriver:

| Name | Last modified | Size | ETag |
|---|---|---|---|
| Parent Directory | | - | |
| chromedriver_linux64.zip | 2023-05-31 08:57:22 | 7.06MB | cd6613edf6628041684393706b62d3a6 |
| chromedriver_mac64.zip | 2023-05-31 08:57:25 | 8.29MB | b44390afbddadf8748a1d151483b2472 |
| chromedriver_mac_arm64.zip | 2023-05-31 08:57:29 | 7.40MB | 0d515e46bea141705e49edaba1d49819 |
| chromedriver_win32.zip | 2023-05-31 08:57:32 | 6.30MB | 7d455bed57ef682d41108e13d45545ca |
| notes.txt | 2023-05-31 08:57:38 | 0.00MB | 1670f6dde7877ca84ecd4c56b9cc759c |

Installing Selenium Grid:

Selenium Server (Grid)

The Selenium Server is needed in order to run Remote Selenium WebDriver (Grid).

Latest stable version 4.10.0

To use the Selenium Server in a Grid configuration see the documentation.

Extracting chromedriver folder:

Extract Compressed (Zipped) Folders

Select a Destination and Extract Files

Files will be extracted to this folder:

C:\chromedriver_win32 Browse...

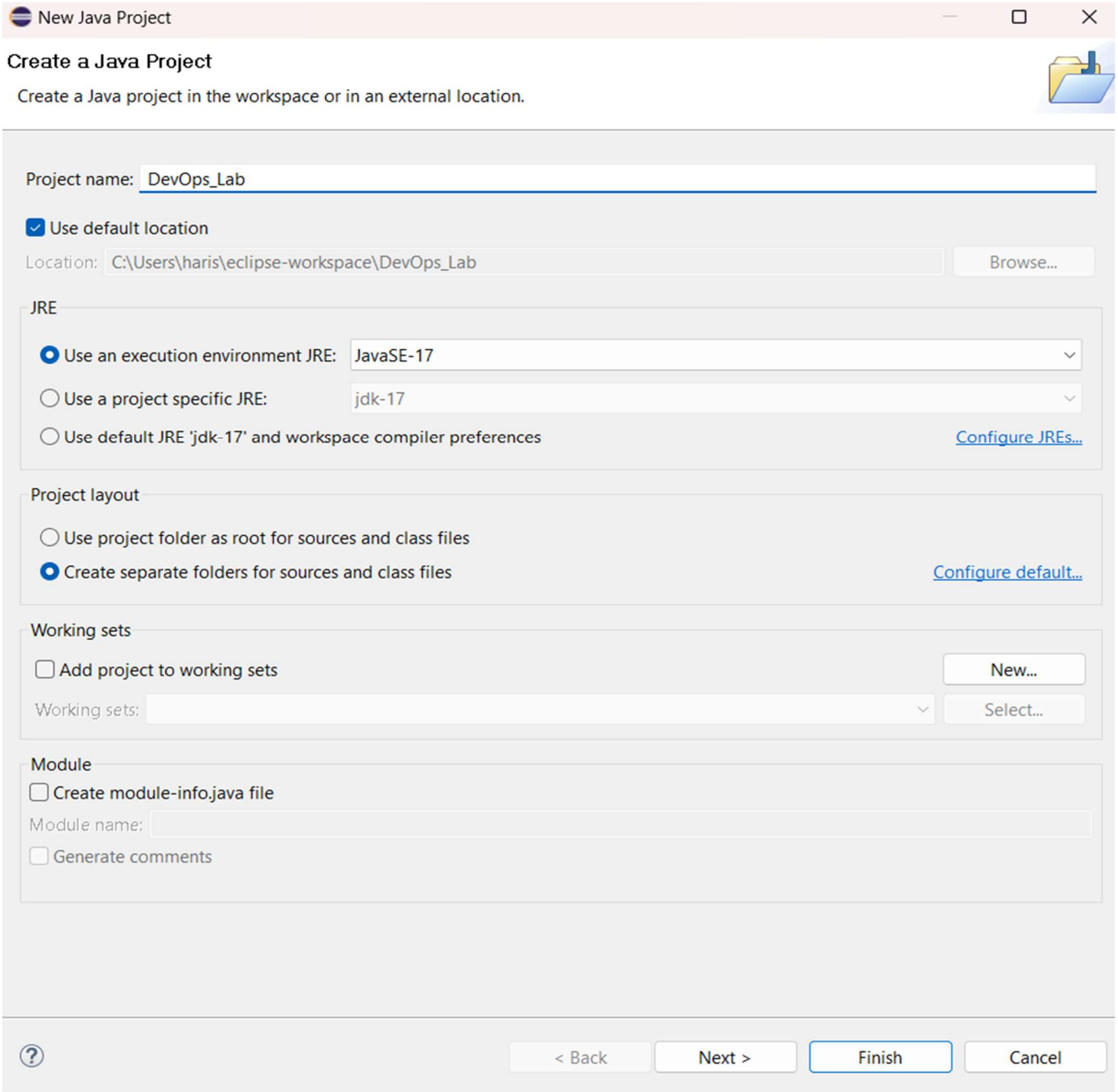
Show extracted files when complete

Extract Cancel

Chromedriver Files:

| | | | |
|---|---|---|---|
| chromedriver | 05-07-2023 19:40 | Application | 11,986 KB |
| LICENSE.chromedriver | 05-07-2023 19:40 | CHROMEDRIVER F... | 243 KB |

Creating Java Project:

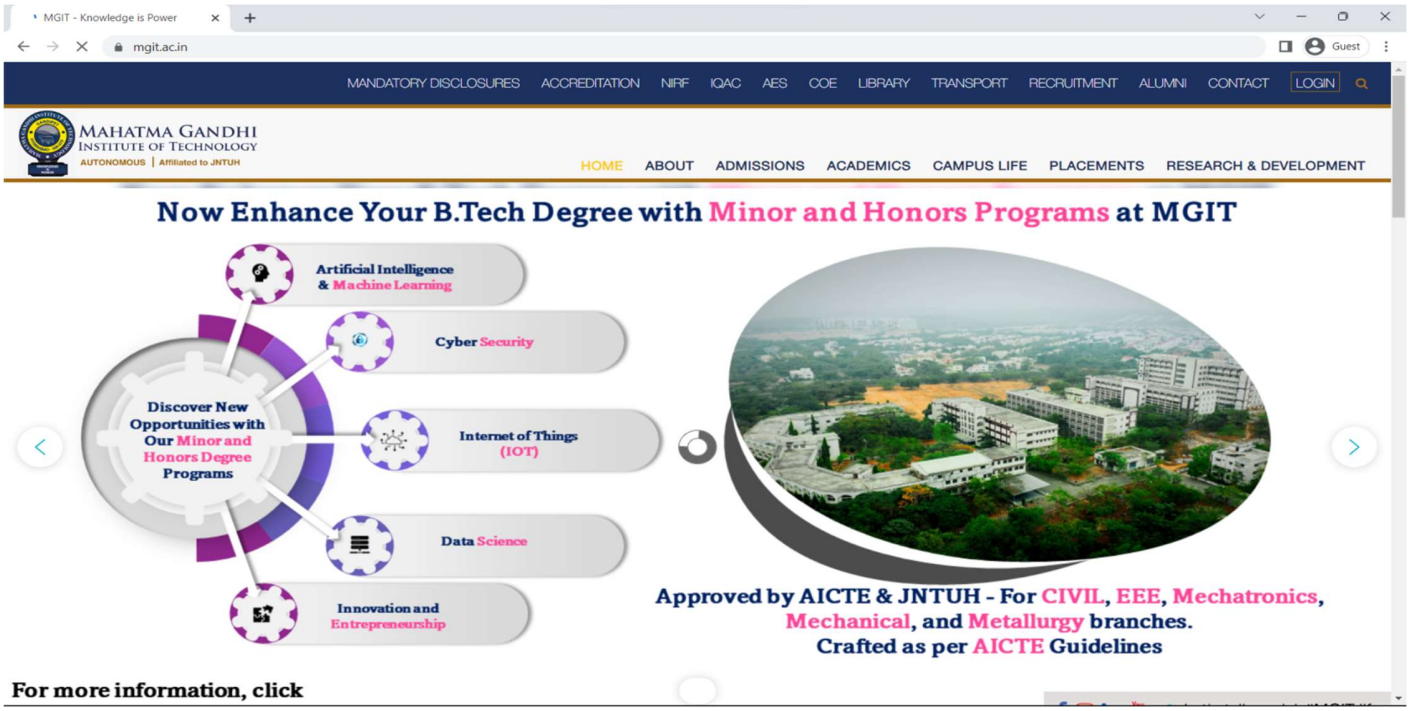


## 11. Write a simple program in JavaScript and perform testing using Selenium.

Test program:

```
*NewOne.java ×
import org.openqa.selenium.WebDriver;

public class NewOne {
    public static void main(String[] args) {
        // Set the path to the chromedriver executable
        System.setProperty("webdriver.chrome.driver", "C:/chromedriver.exe");

        // Create a new instance of the ChromeDriver
        WebDriver driver = new ChromeDriver();

        // Navigate to a website
        driver.get("https://www.mgit.ac.in");

        // Perform some testing actions
        String title = driver.getTitle();
        System.out.println("Page title: " + title);

        // Close the browser
        driver.quit();
    }
}
```

Testing a webpage:




Output:

```
Problems  Javadoc  Declaration  Console ×
<terminated> NewOne [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (05-Jul-2023, 9:12:02 pm – 9:12:34 pm) [pid: 16340]
Jul 05, 2023 9:12:02 PM org.openqa.selenium.remote.service.DriverService$Builder getLogOutput
INFO: Driver logs no longer sent to console by default; https://www.selenium.dev/documentation/webdriver/drivers/service/#setting-log-output
Page title: MGIT - Knowledge is Power
```