Name: Mrunali Katta

ID: 017516785

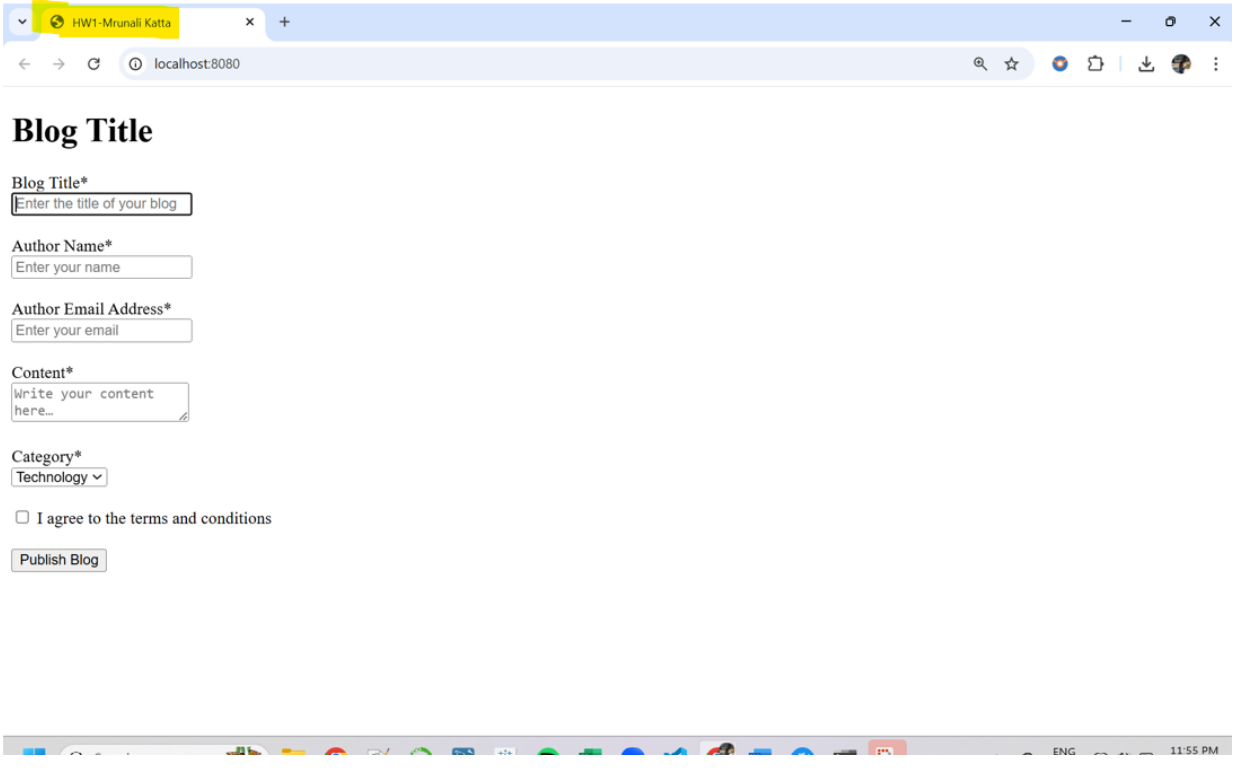
DATA 235 – Homework 01

I. HTML

1. Create an HTML page with the title "HW1-{Your Name}"

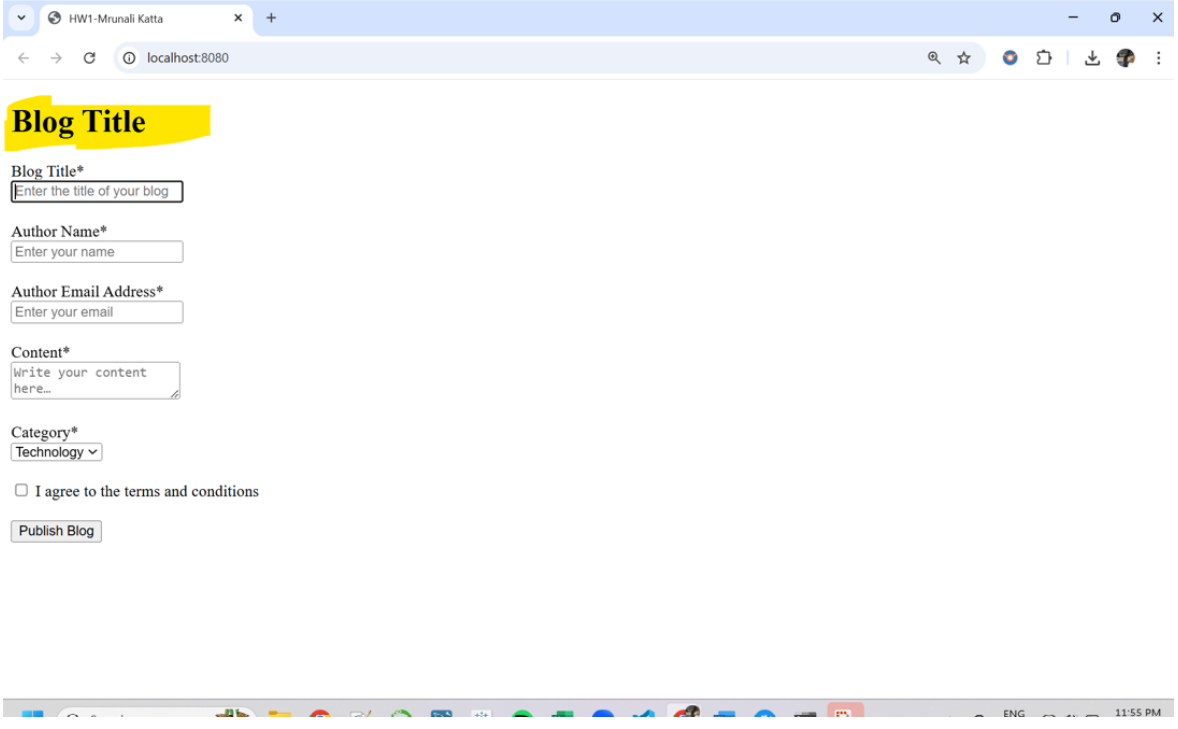
Here created a file with name **'index_hw1.html'** with title **"HW1-Mrunali Katta"**

```
<!-- html page -->
<!DOCTYPE html>
<!--root element-->
<html lang="en">
    <!--head tag start-->
<head>
<!-- meta tags -->
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- HTML page with the title-->
    <title>HW1-Mrunali Katta</title>
</head>
```



2. Added a heading tag to name the title of the as **“Blog Topic”** using header tag which renders the biggest font size <h1>

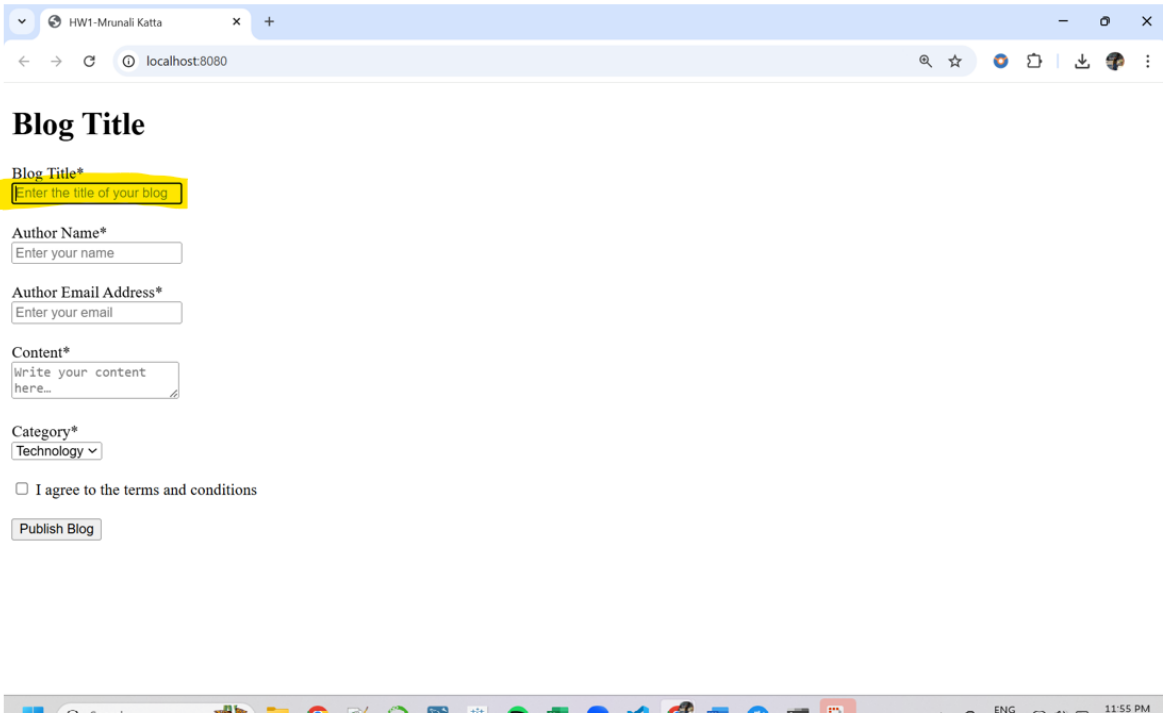
```
<body>
    <!--title of the blog topic-->
<!--biggest font size-->
    <h1>Blog Title</h1>
```

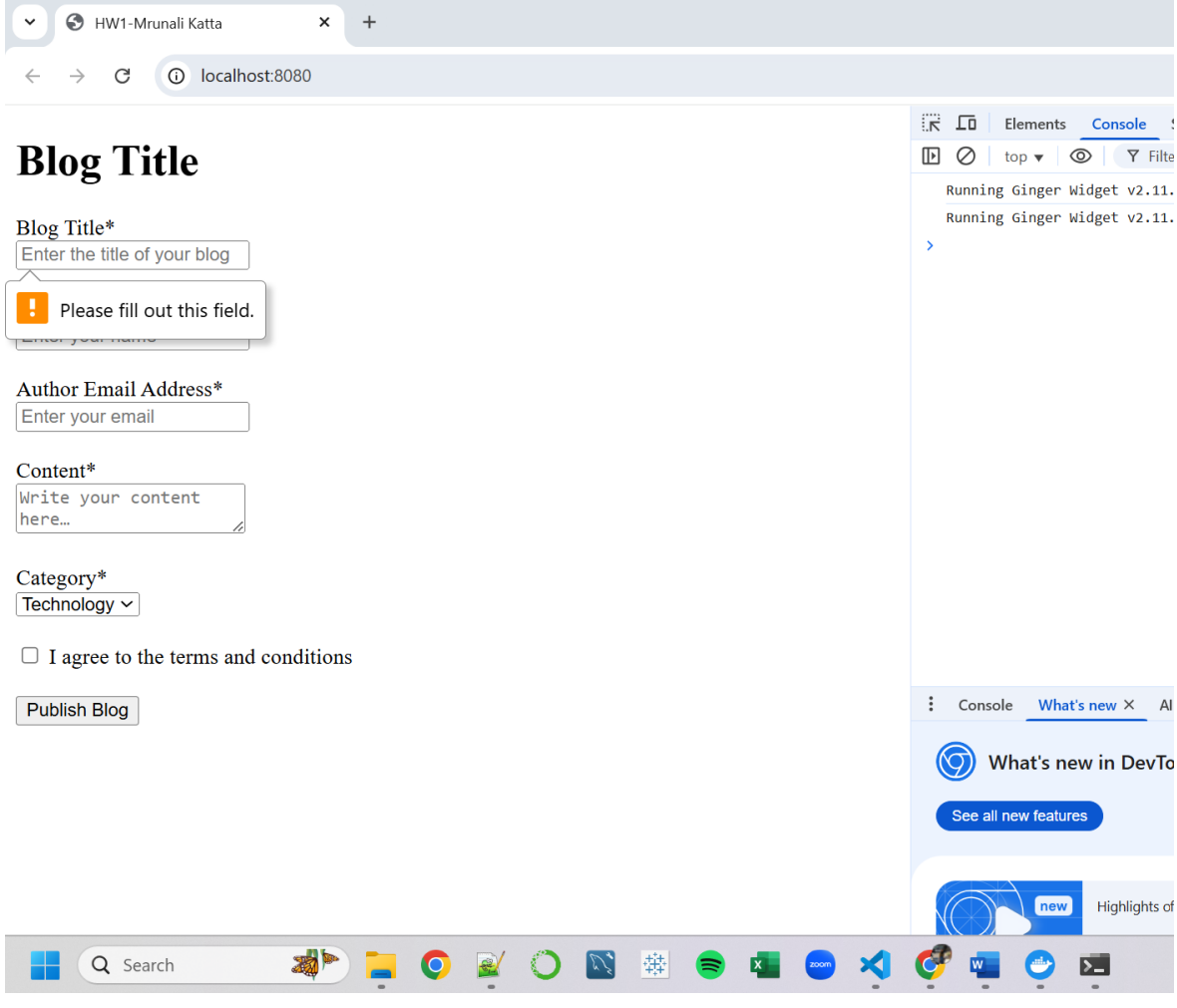


3. Created a Form:

a) Displaying a text input for the blog title, placeholder text “Enter the title of your blog”. This field should be required and the cursor should automatically focus on this field when the page loads.

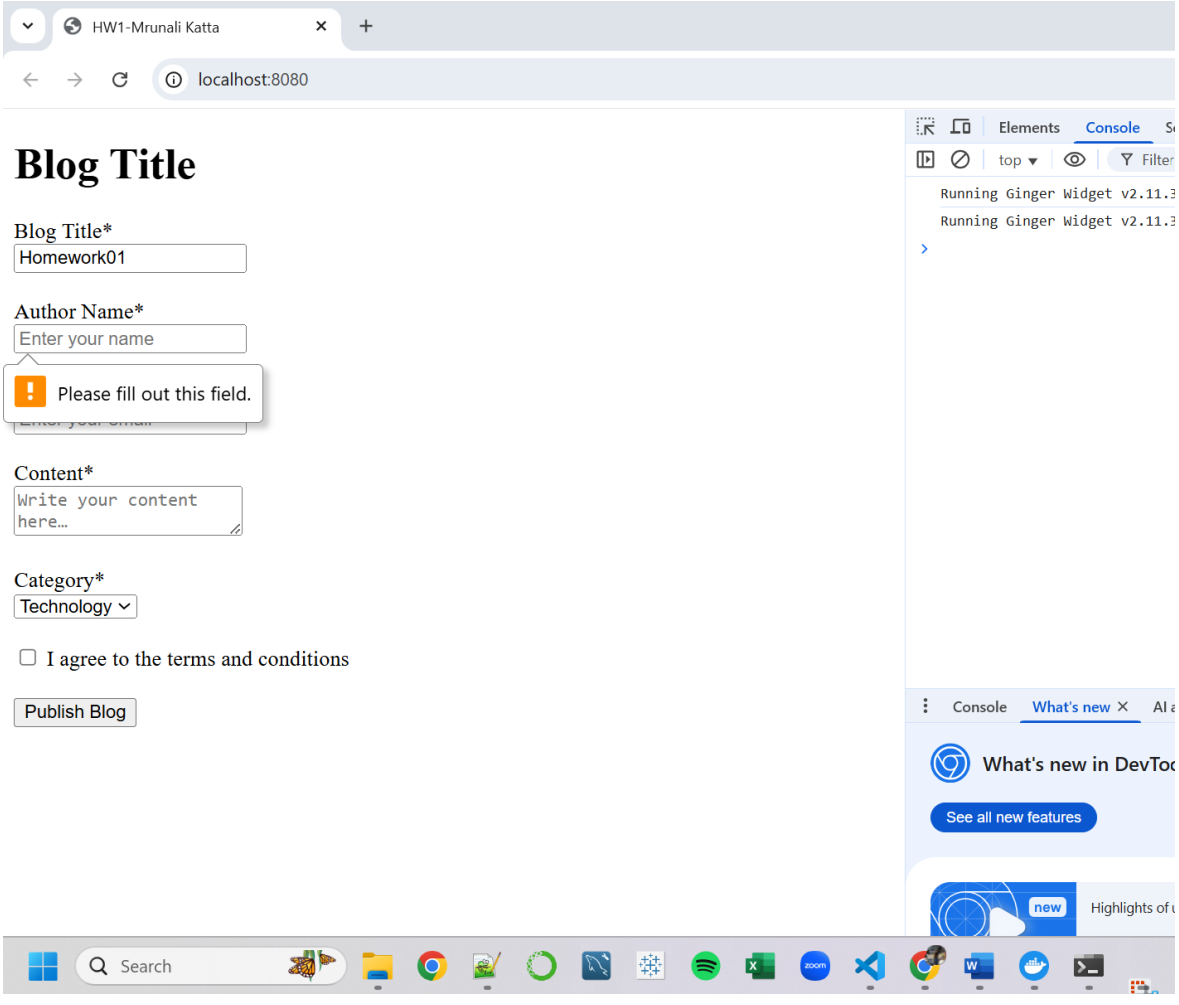
```
<!--form tag to collect inputs from user-->
    <form id="myBlogForm">
        <label for="blogTitle">Blog Title*</label><br>
       <!-- using autofocus the cursor will automatically be placed in the
textfield-->
        <input type="text" id="myBlogTitle" placeholder="Enter the title of
your blog" required autofocus><br><br>
```


HW1-Mrunali Katta
localhost:8080
Blog Title
Blog Title*
Enter the title of your blog
Author Name*
Enter your name
Author Email Address*
Enter your email
Content*
Write your content here…
Category*
Technology
I agree to the terms and conditions
Publish Blog


HW1-Mrunali Katta
localhost:8080
Blog Title
Blog Title*
Enter the title of your blog
Please fill out this field.
Author Email Address*
Enter your email
Content*
Write your content here…
Category*
Technology
I agree to the terms and conditions
Publish Blog
Elements
Console
top
Running Ginger Widget v2.11.
Running Ginger Widget v2.11.
Console
What's new
What's new in DevTo
See all new features
new
Highlights of
Search

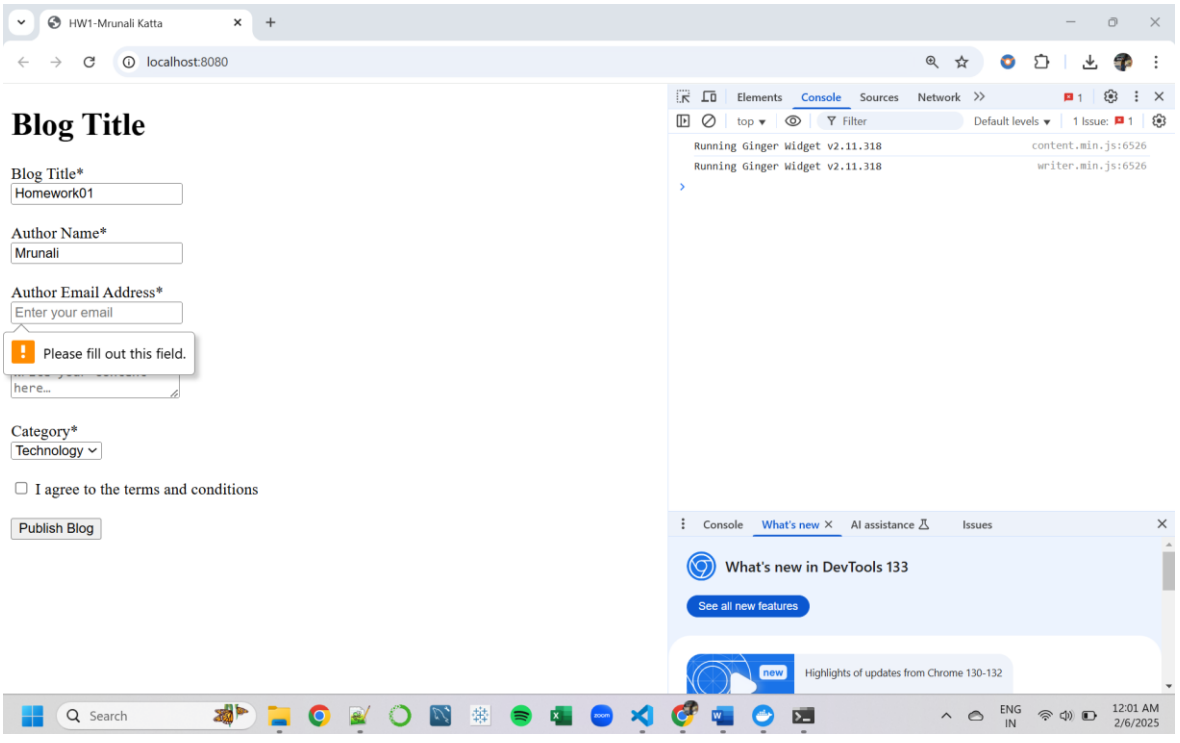
b) A text input for the author name, placeholder "Enter your name", and required.

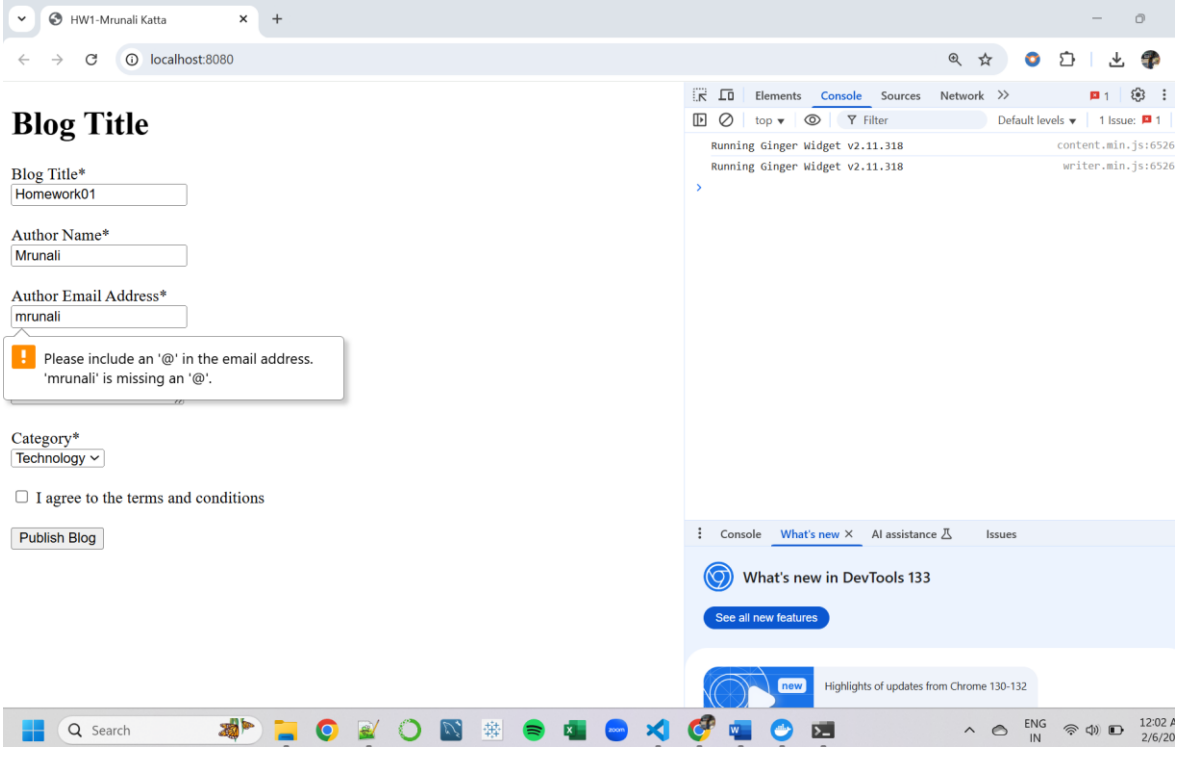
```
 <label for="blogName">Author Name*</label><br>
        <input type="text" id="userName" placeholder="Enter your name"
required><br><br>
```

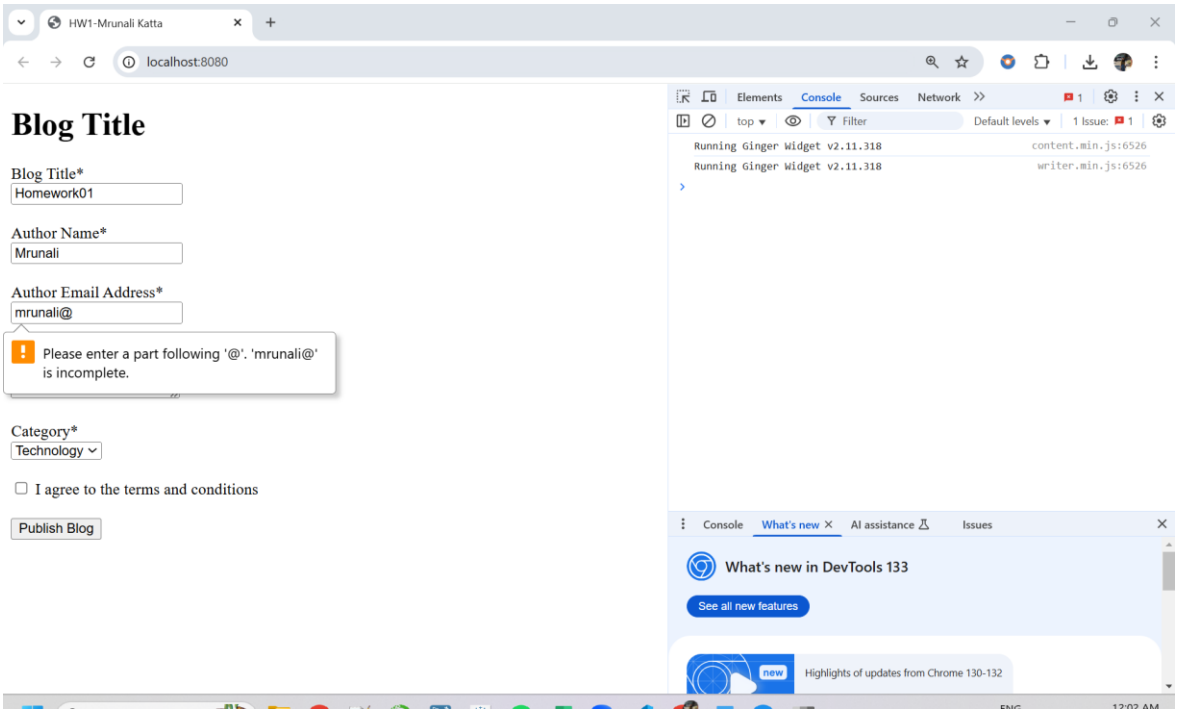


c) An email input for the email address, placeholder “Enter your email”, and required.

```
 <label for="blogEmailAddress">Author Email Address*</label><br>
        <input type="email" id="userEmailAddress"  placeholder="Enter your
email" required><br><br>
```

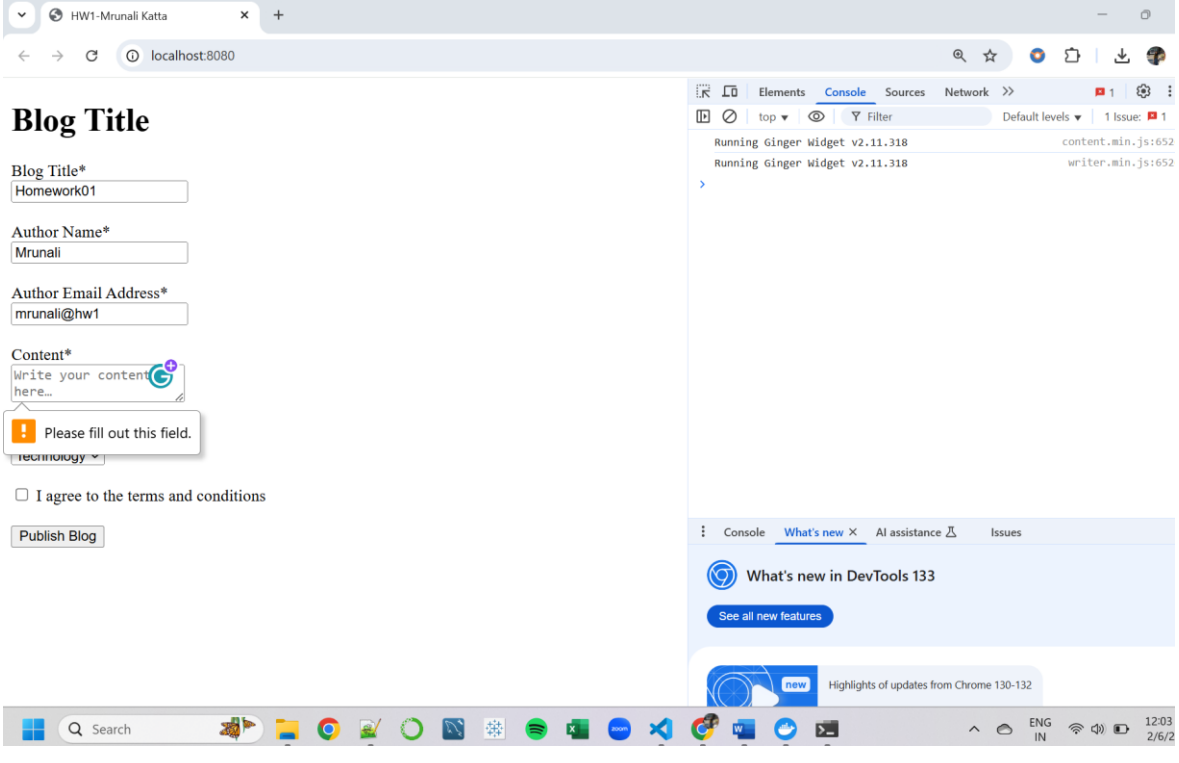

HW1-Mrunali Katta
localhost:8080
Blog Title
Blog Title*
Homework01
Author Name*
Mrunali
Author Email Address*
Enter your email
Please fill out this field.
here...
Category*
Technology
I agree to the terms and conditions
Publish Blog
Elements
Console
Sources
Network
Running Ginger Widget v2.11.318
content.min.js:6526
writer.min.js:6526
What's new in DevTools 133
See all new features
Highlights of updates from Chrome 130-132
Search
ENG
IN
12:01 AM
2/6/2025


HW1-Mrunali Katta
localhost:8080
Blog Title
Blog Title*
Homework01
Author Name*
Mrunali
Author Email Address*
mrunali
Please include an '@' in the email address. 'mrunali' is missing an '@'.
Category*
Technology
I agree to the terms and conditions
Publish Blog
Elements
Console
Sources
Network
Running Ginger Widget v2.11.318
content.min.js:6526
writer.min.js:6526
What's new in DevTools 133
See all new features
Highlights of updates from Chrome 130-132
Search
ENG
IN



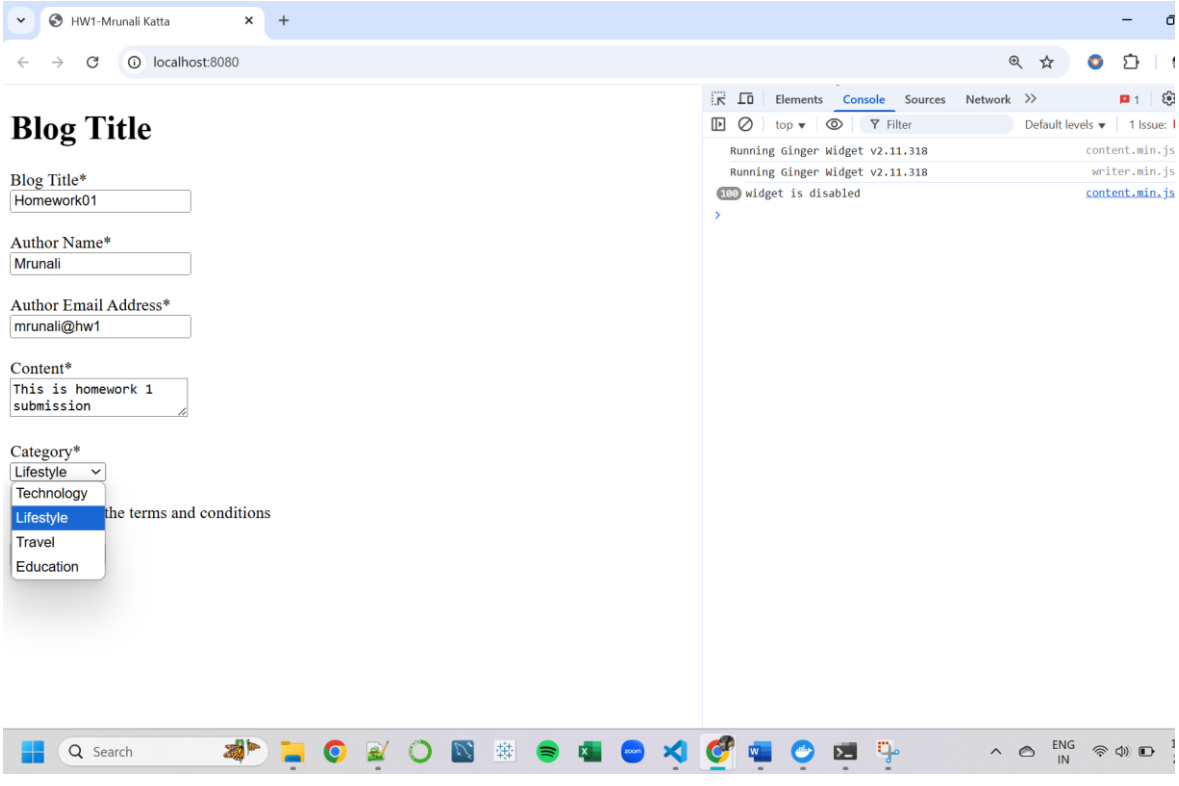
d) A text area for the blog content, placeholder “Write your content here...”, and required.

```
 <label for="blogContent">Content*</label><br>
        <textarea id="myBlogContent" placeholder="Write your content here…"
required></textarea><br><br>
```



e) A dropdown for category selection, and options "Technology," "Lifestyle," "Travel," and "Education."

```
<label for="blogCategories">Category*</label><br>
        <select id="myBlogCategories">
            <option value="Technology">Technology</option>
            <option value="Lifestyle">Lifestyle</option>
            <option value="Travel">Travel</option>
            <option value="Education">Education</option>
        </select><br><br>
```



f) A checkbox and a label with the text “I agree to the terms and conditions.”

```
 <input type="checkbox" id="terms">
        <label for="terms">I agree to the terms and
conditions</label><br><br>
```

g) A submit button with the text “Publish Blog”.

```
<button type="submit">Publish Blog</button>
```

4. Add a script tag to link your javascript code for part II at the end of your HTML file.

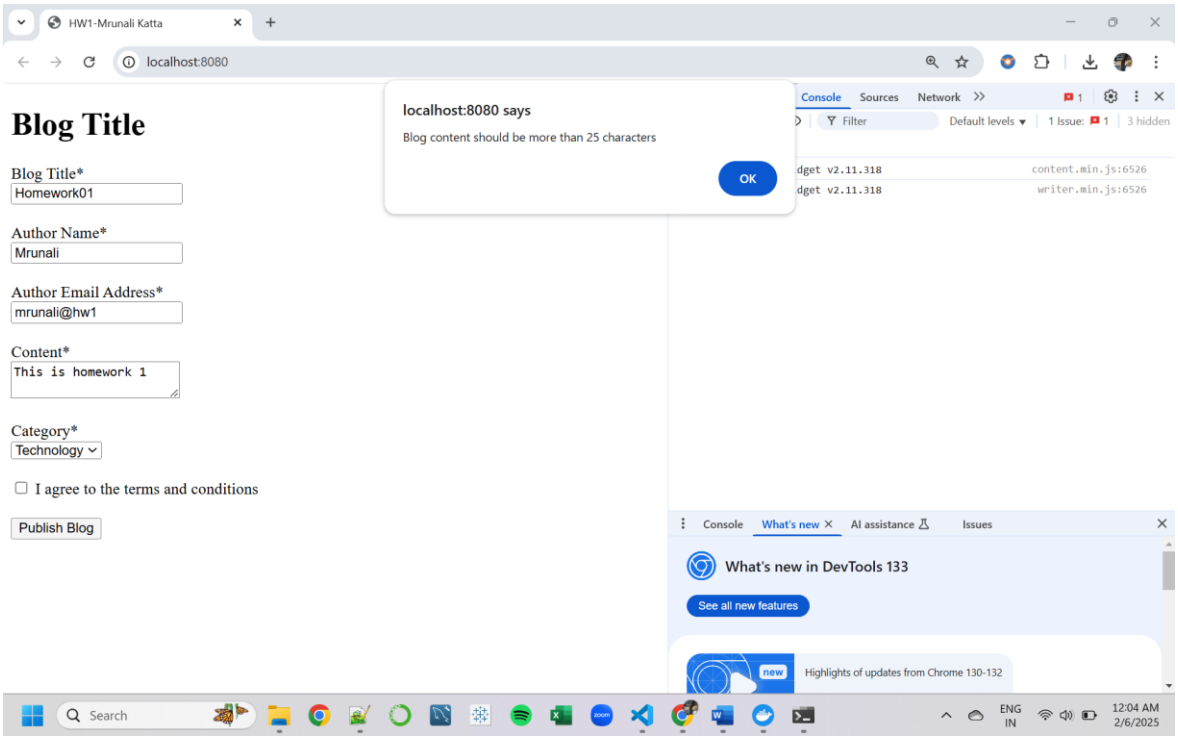
Here the “index_hw1.html” file is linked to .js file called **“blog_hw1.js”**

```
</form>
    <script src="blog_hw1.js"></script>
</body>
</html>
```

II. Javascript

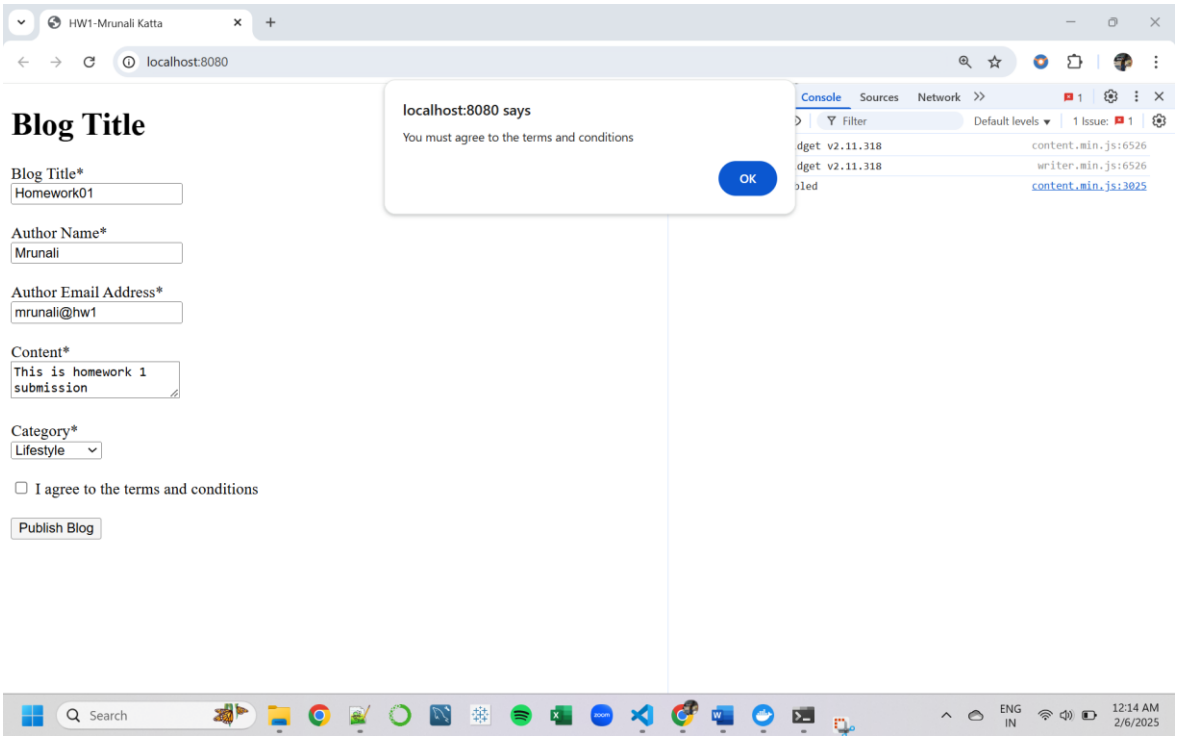
1. Write a javascript function using an arrow function to validate:
   a. Verify if the blog content is more than 25 characters. If the validation fails, display an alert with the message “Blog content should be more than 25 characters”.

```
//using "use strict" mode for error handling and to impose stricter parsing
in JS code
"use strict";

// form validation
const formValidation = () => {
    const blogTitle = document.getElementById("myBlogTitle").value.trim();
    const authorName = document.getElementById("userName").value.trim();
    const authorEmail =
document.getElementById("userEmailAddress").value.trim();
    const blogContent =
document.getElementById("myBlogContent").value.trim();
    const termsChecked = document.getElementById("terms").checked;

    // to see if all fields are filled
    if (!blogTitle || !authorName || !authorEmail || !blogContent) {
        alert("All fields are required!");
        return false;
    }

    // blog content > than 25 characters check condition
    if (blogContent.length <= 25) {
        alert("Blog content should be more than 25 characters");
        return false;
    }
```



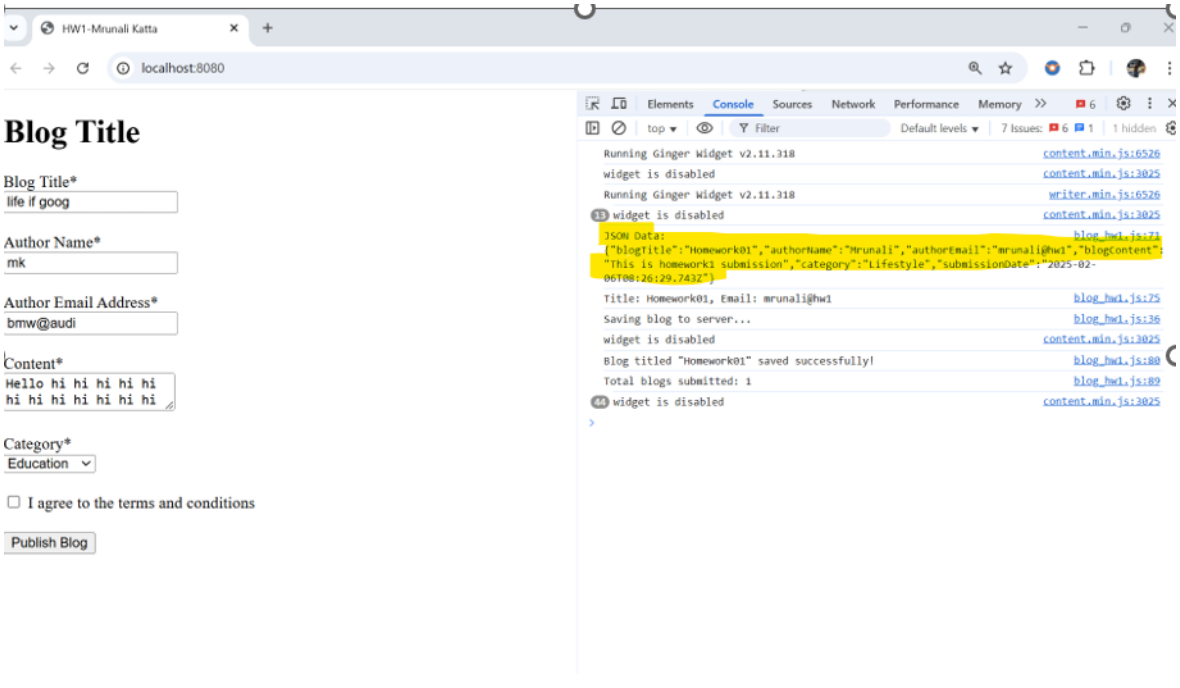
b. Verify if the terms and conditions check box is checked. If the validation fails, display an alert with the message “You must agree to the terms and conditions”.

```
  //  terms and conditions - condition
    if (!termsChecked) {
        alert("You must agree to the terms and conditions");
        return false;
    }

    return true;
}; // end of formValidation
```

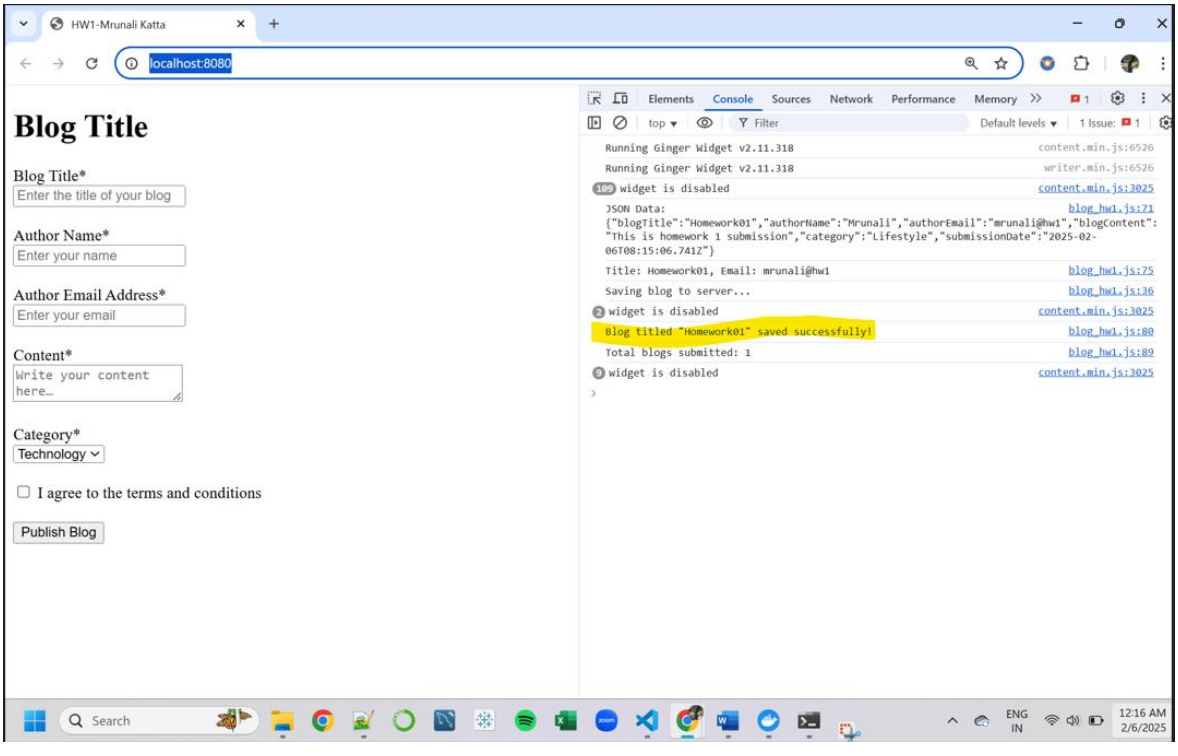


2. After the form submission is successful, convert the form data into a JSON string and log the output in the console.

```
// converting data to JSON ONLY after validation
 const jsonData = JSON.stringify(updatedBlogData);
 console.log("JSON Data:", jsonData);
```

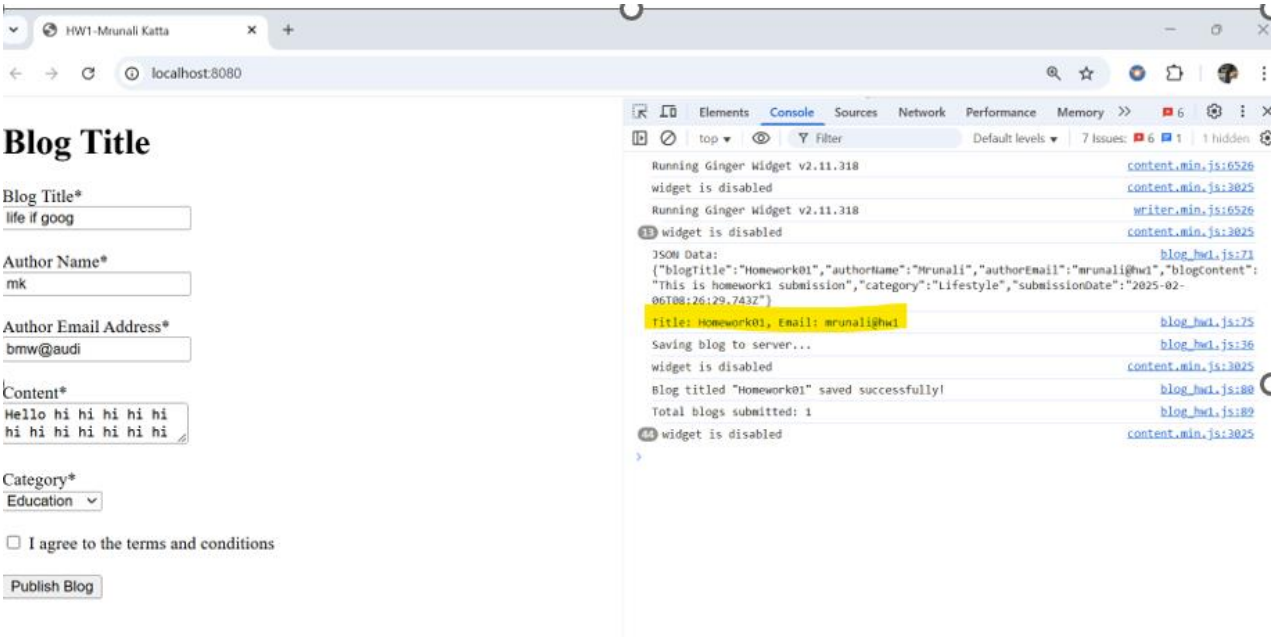


```
// to simulate an asynchronous operation
const saveBlogToServer = (blogData) => {
    return new Promise((resolve, reject) => {
        console.log("Saving blog to server...");
        setTimeout(() => {
            const success = Math.random() > 0.2; // 80% success rate
            if (success) {
                resolve(`Blog titled "${blogData.blogTitle}" saved
successfully!`);
            } else {
                reject("Failed to save the blog. Please try again.");
            }
        }, 3000); // insert a 3-second delay
    });
};
```



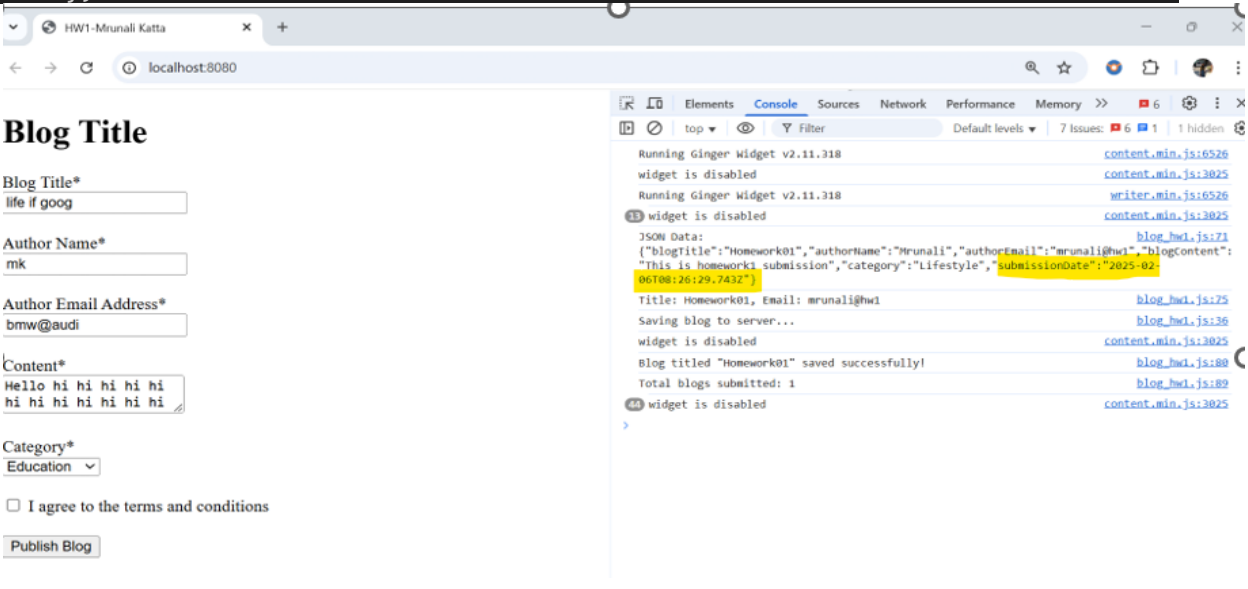
3. Use object destructuring to extract the title and email fields from the parsed object and log their values in the console.

```
    // extracting title and email
    const { blogTitle, authorEmail } = updatedBlogData;
    console.log(`Title: ${blogTitle}, Email: ${authorEmail}`);
```



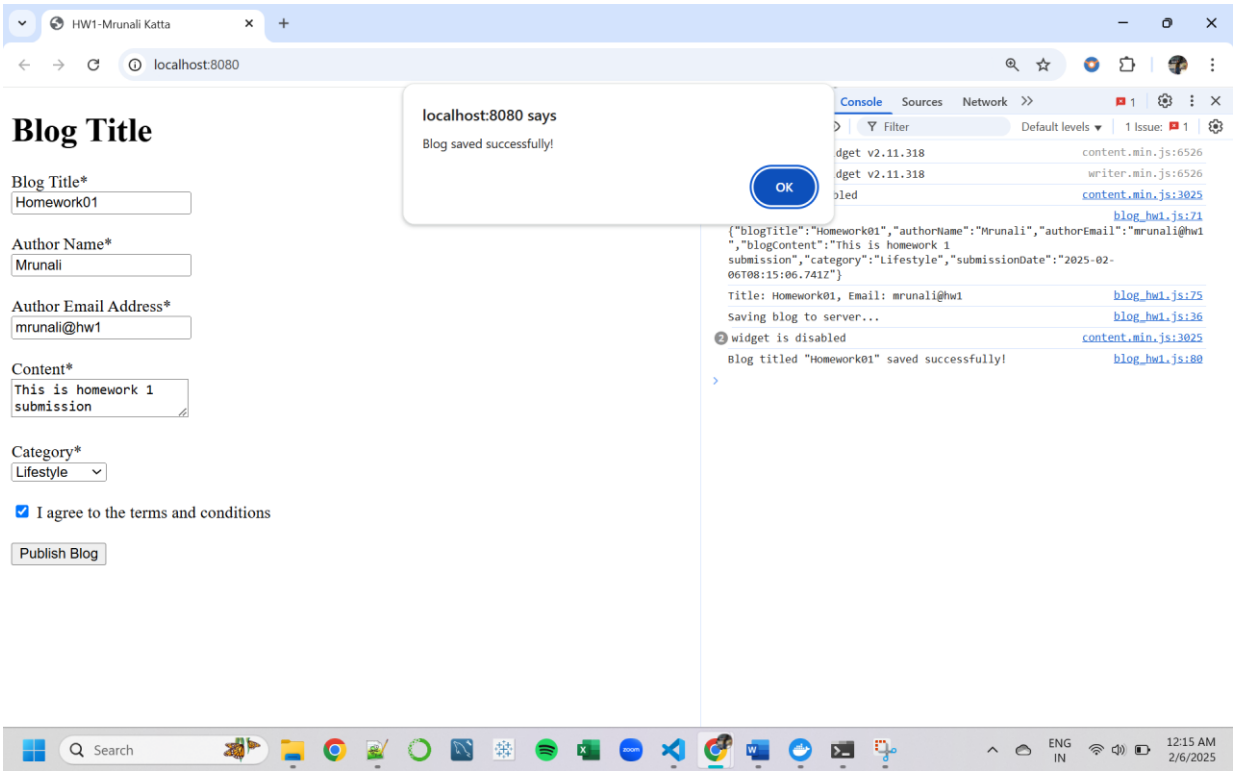
4. Use the spread operator to add a new field “submissionDate” with the current date and time to the parsed object. Log the updated parsed object in the console.

```
// spread operator to Add Submission Date
const updatedBlogData = {
    ...blogData,
    submissionDate: new Date().toISOString()
};
```

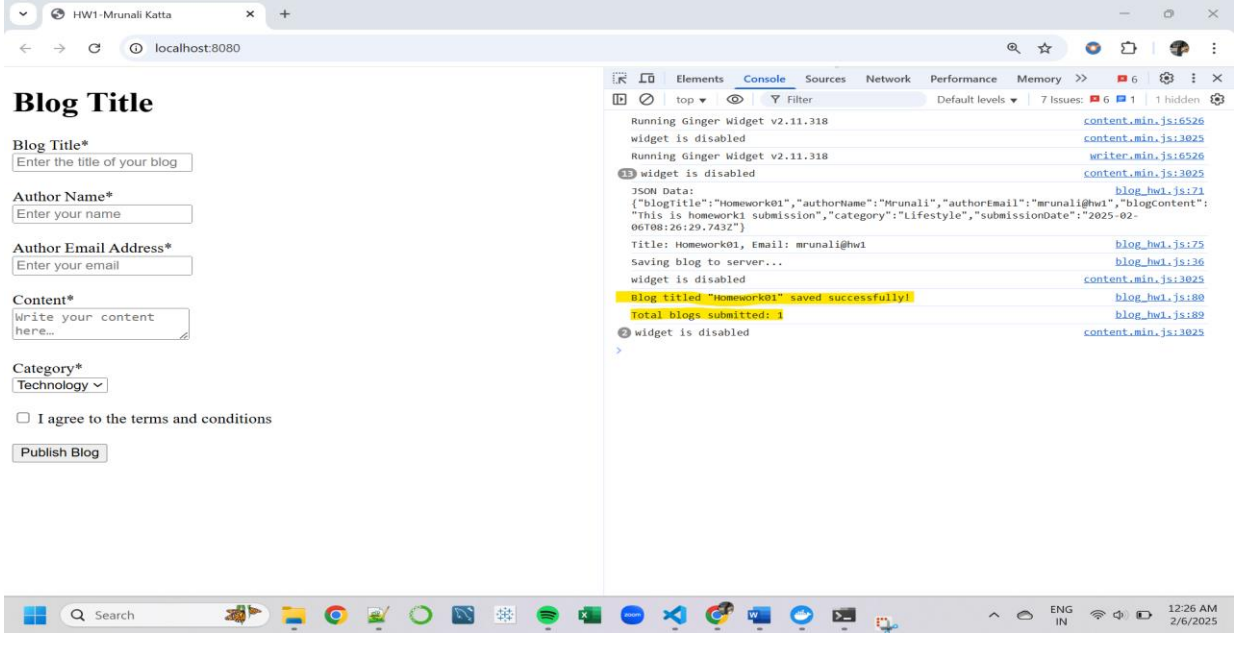


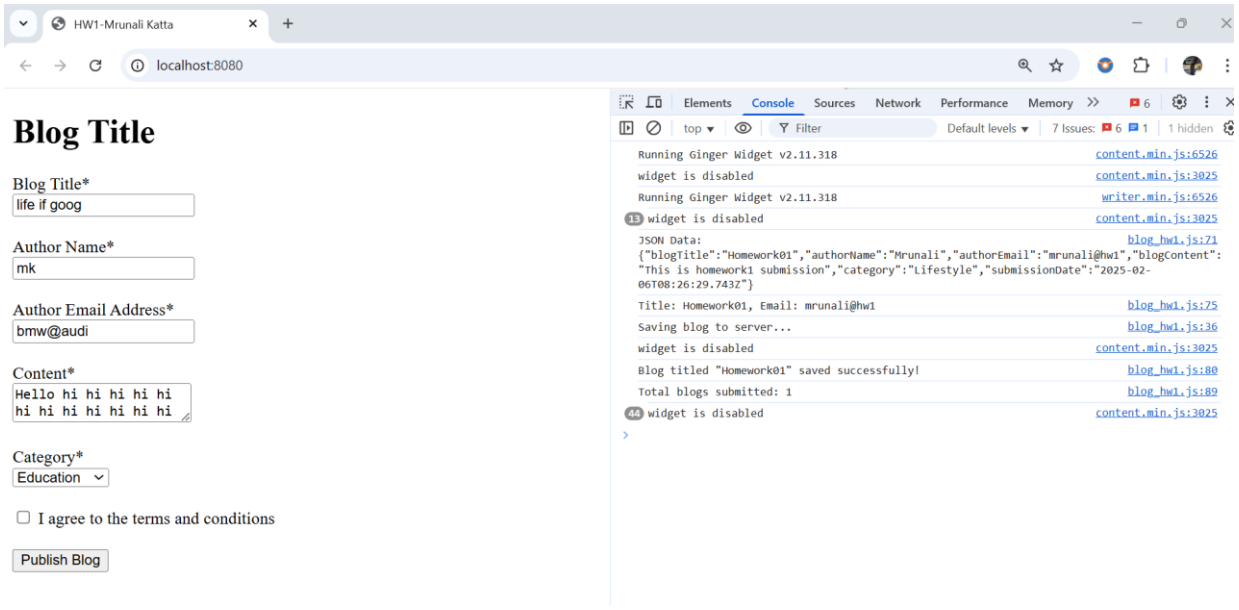
5. Create a closure to track how many times the form has been successfully submitted and log the submission count each time the form is submitted.

```
// a closure to track the number of blog submissions
const blogCounter = (() => {
    let count = 0;
    return () => ++count;
})();
```

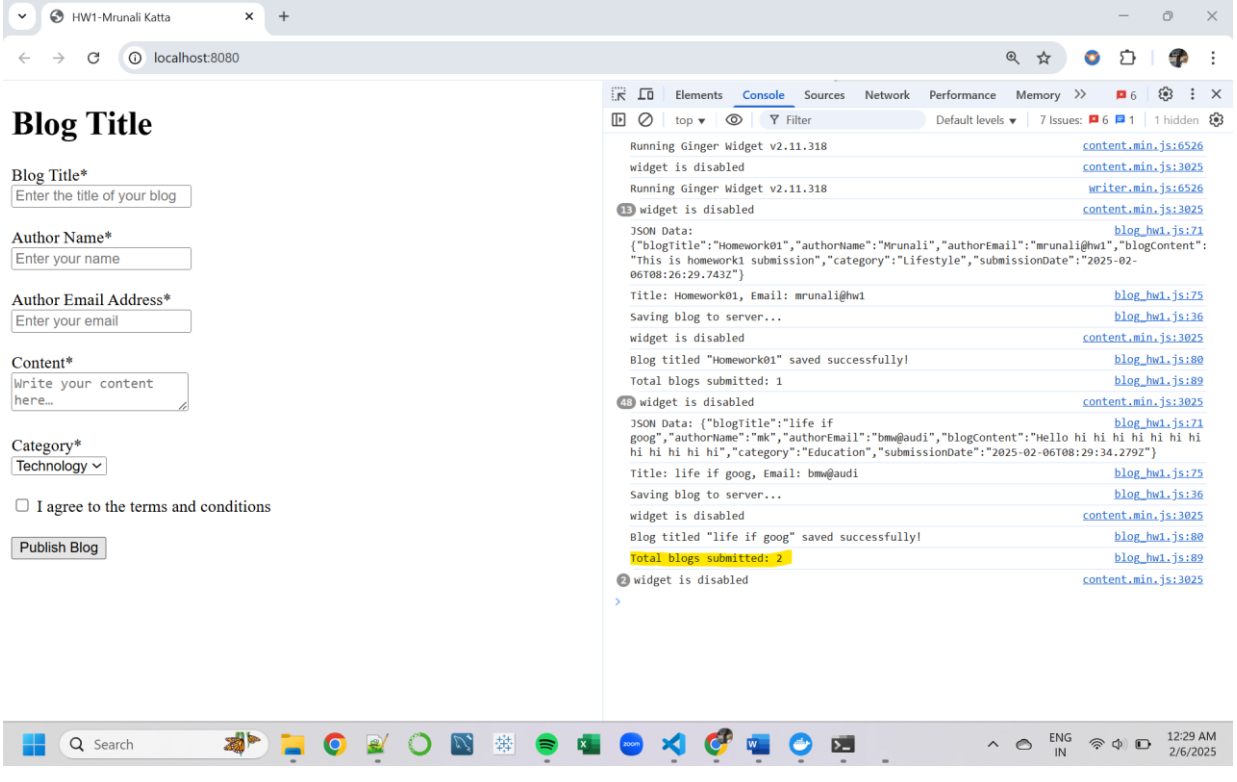


Count updated to 1



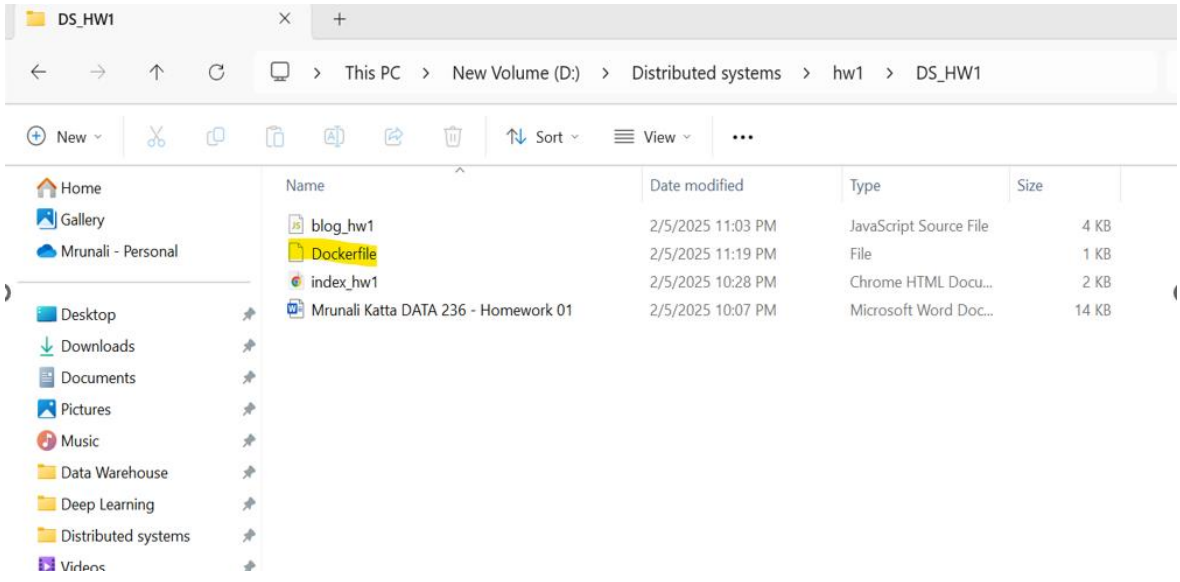


Count updated to 2



III. DEPLOYMENT:

1) Dockerize the Application
    1. Create a Dockerfile in the same directory as your application



2. Build the docker image using docker build -t <name> .

- Docker Image Successfully Built using

docker build -t myblogapp_hw1 .

```
Command Prompt

Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

C:\Users\mruna>docker --version
Docker version 27.2.0, build 3ab4256

C:\Users\mruna>cd /d "D:\Distributed systems\hw1\DS_HW1"

D:\Distributed systems\hw1\DS_HW1>docker build -t myblogapp_hw1 .
[+] Building 9.6s (10/10) FINISHED                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                  0.0s
 => => transferring dockerfile: 522B                                                                                  0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                                       1.9s
 => [internal] load .dockerignore                                                                                     0.1s
 => => transferring context: 2B                                                                                       0.0s
 => [1/5] FROM docker.io/library/nginx:latest@sha256:91734281c0ebfc6f1aea979cffeed5079cfe786228a71cc6f1f46a228cde     6.5s
 => => resolve docker.io/library/nginx:latest@sha256:91734281c0ebfc6f1aea979cffeed5079cfe786228a71cc6f1f46a228cde     0.0s
 => => sha256:84cade77a8319515342dbc0f9d9ce4bbddd2d95f93646091b61a4ca318164a44 1.40kB / 1.40kB                        0.3s
 => => sha256:6c78b0ba1a32d228d04680a4dc7bd67a21f2e30f6fc032b5648d0cae756efe14 1.21kB / 1.21kB                        0.5s
 => => sha256:976e8f6b25dd4b8451fa52d71524f61103d32dca1af6b1ddb70b0ea848a3e373 403B / 403B                            0.7s
 => => sha256:c558df2179491b5490a7b895e422fffcddcfaec6c045d49907048d32a2ab5784 955B / 955B                            0.7s
 => => sha256:24ff42a0d907c9f6e59a6ce8ad8691f8b72ffad5e1e55584612b86cb632f5cfa 627B / 627B                            0.4s
 => => sha256:e19db8451adb4b9c6dc54aaec5c9fd8917d5267d249a977b95dab00ce960e4bf 43.95MB / 43.95MB                      5.1s
 => => sha256:c29f5b76f736a8b555fd191c48d6581bb918bcd605a7cbcc76205dd6acff3260 28.21MB / 28.21MB                      3.6s
 => => extracting sha256:c29f5b76f736a8b555fd191c48d6581bb918bcd605a7cbcc76205dd6acff3260                             0.7s
 => => extracting sha256:e19db8451adb4b9c6dc54aaec5c9fd8917d5267d249a977b95dab00ce960e4bf                             0.6s
 => => extracting sha256:24ff42a0d907c9f6e59a6ce8ad8691f8b72ffad5e1e55584612b86cb632f5cfa                             0.0s
 => => extracting sha256:c558df2179491b5490a7b895e422fffcddcfaec6c045d49907048d32a2ab5784                             0.0s
 => => extracting sha256:976e8f6b25dd4b8451fa52d71524f61103d32dca1af6b1ddb70b0ea848a3e373                             0.0s
 => => extracting sha256:6c78b0ba1a32d228d04680a4dc7bd67a21f2e30f6fc032b5648d0cae756efe14                             0.0s
 => => extracting sha256:84cade77a8319515342dbc0f9d9ce4bbddd2d95f93646091b61a4ca318164a44                             0.0s
 => [internal] load build context                                                                                     0.1s
 => => transferring context: 5.44kB                                                                                   0.0s
 => [2/5] WORKDIR /usr/share/nginx/html                                                                               0.4s
 => [3/5] RUN rm -rf ./*                                                                                              0.3s
 => [4/5] COPY index_hw1.html /usr/share/nginx/html/index.html                                                        0.0s
 => [5/5] COPY blog_hw1.js /usr/share/nginx/html/blog_hw1.js                                                          0.0s
 => exporting to image                                                                                                0.2s
 => => exporting layers                                                                                               0.1s
 => => exporting manifest sha256:d61e7e6c3375abd61a5c138d2fd3a30ced4071e90f61cbbe99016ba3b8079b02                     0.0s
 => => exporting config sha256:0643f692935ce00a28cd40bbb2e641dcf4ed4d7a0403957b7b10bc6582a6e530                       0.0s
 => => exporting attestation manifest sha256:319302111a769a57f482dd491fde1a0dadb9ca7915a16d861f6cb4556250cc25         0.0s
 => => exporting manifest list sha256:89da25a61d9d7a73a86579266aece486dd57d5a89dbb181b2b30aa9aba150c1b                0.0s
 => => naming to docker.io/library/myblogapp_hw1:latest                                                               0.0s
 => => unpacking to docker.io/library/myblogapp_hw1:latest                                                            0.1s

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview

D:\Distributed systems\hw1\DS_HW1>
```

3. Run the docker container locally docker run -d -p 8080:80 <name>

docker run -d -p 8080:80 myblogapp_hw1

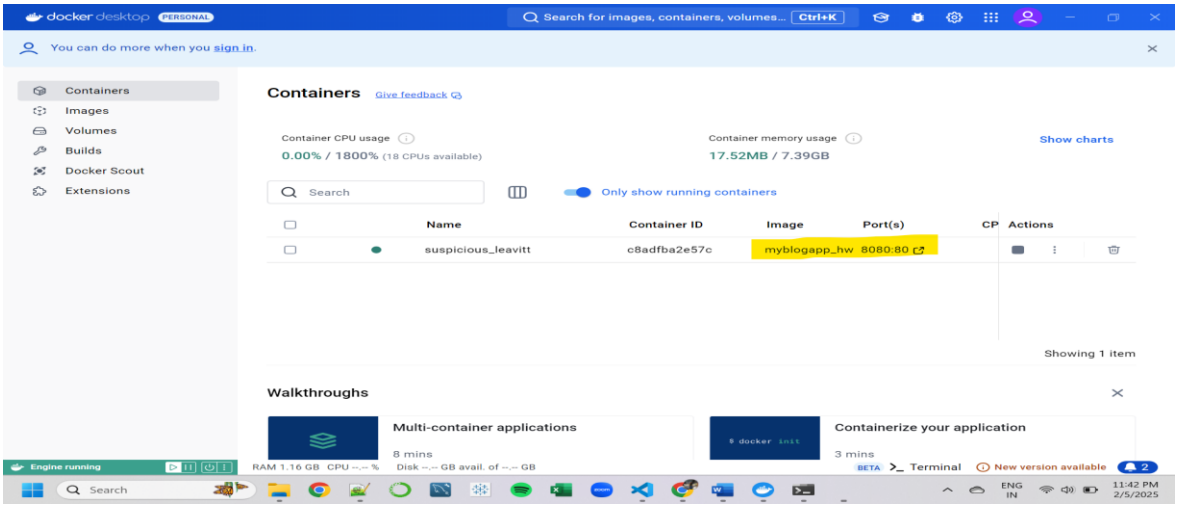
```
D:\Distributed systems\hw1\DS_HW1>docker run -d -p 8080:80 myblogapp_hw1
c8adfba2e57c48221a9911f474ff4f1e5d9b368b1c8ff249785b6264445550dc

D:\Distributed systems\hw1\DS_HW1>
```

- the container is running

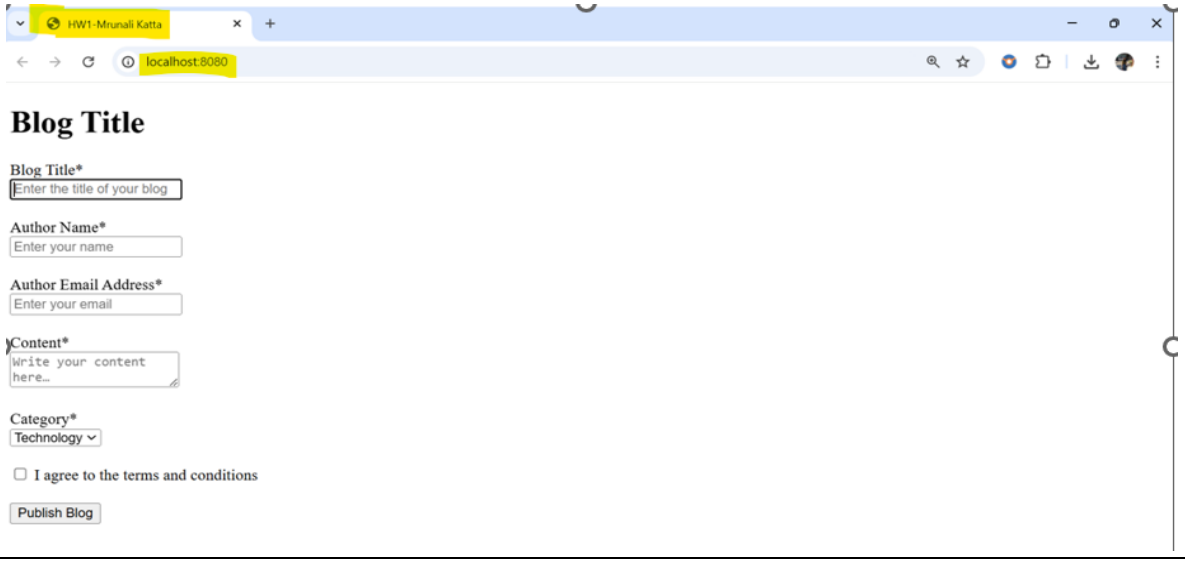
docker ps

```
D:\Distributed systems\hw1\DS_HW1>docker ps
CONTAINER ID   IMAGE           COMMAND                  CREATED         STATUS         PORTS                    NAMES
c8adfba2e57c   myblogapp_hw1   "/docker-entrypoint.…"   5 minutes ago   Up 5 minutes   0.0.0.0:8080->80/tcp   suspicious_leavitt

D:\Distributed systems\hw1\DS_HW1>
```

Docker desktop application

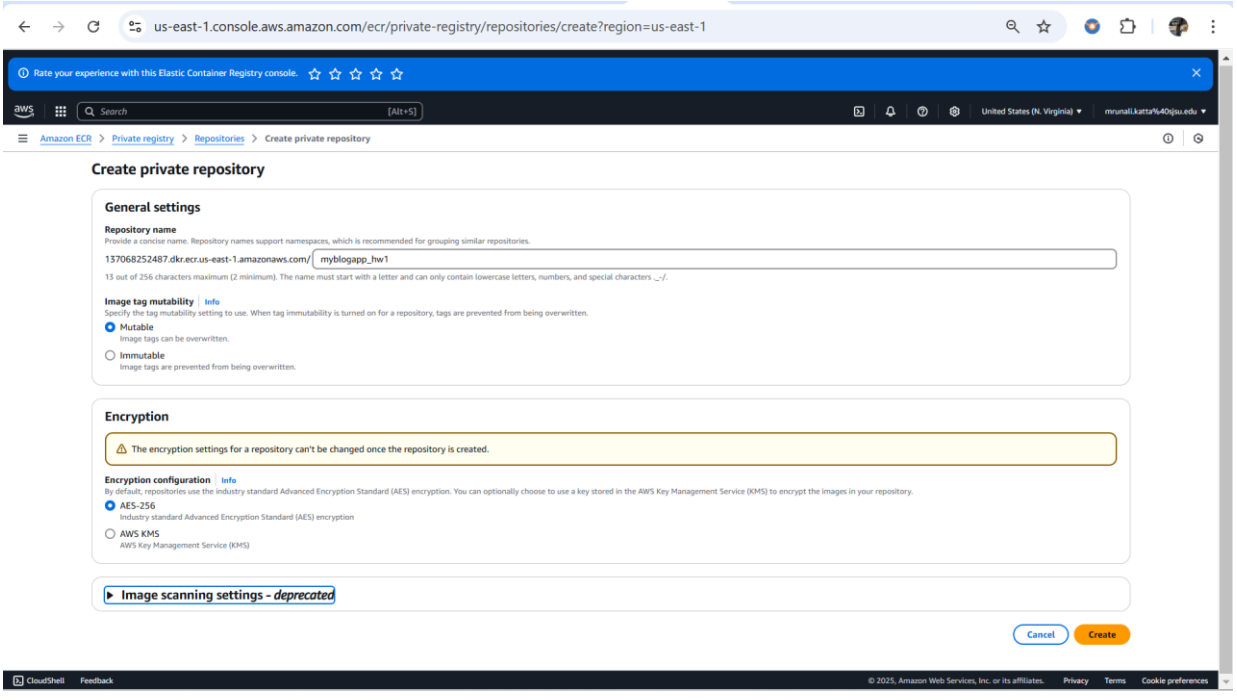


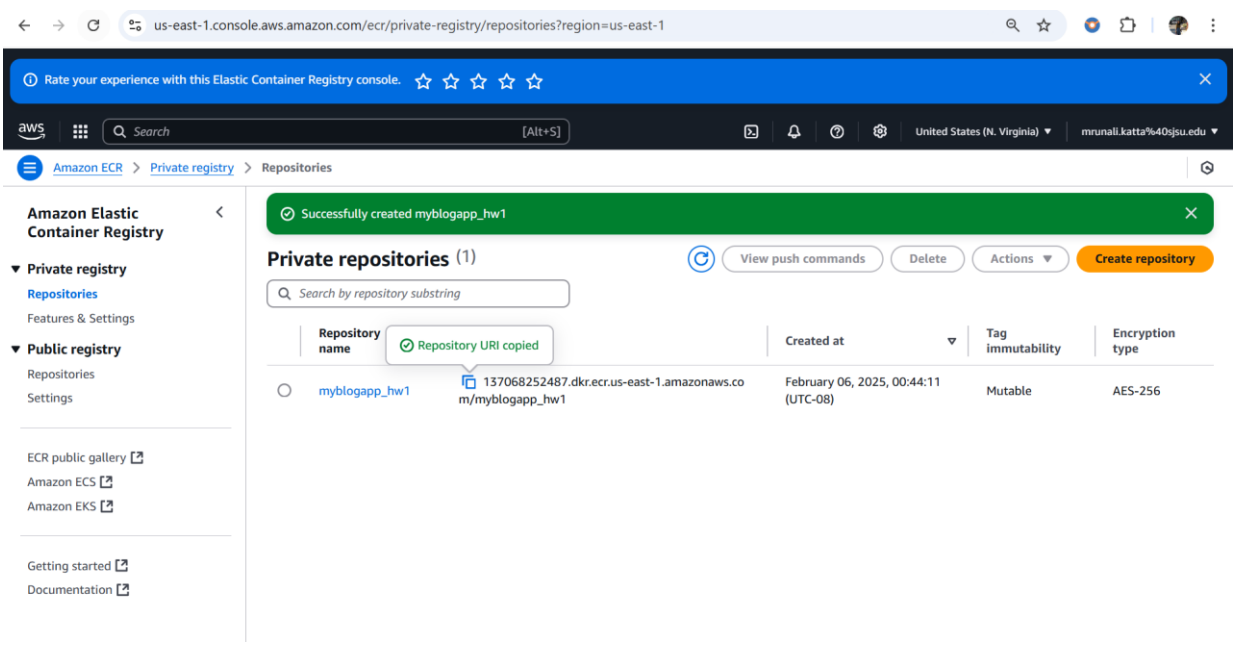
4. Verify by visiting http://localhost:8080



Pushing the docker image to ECR

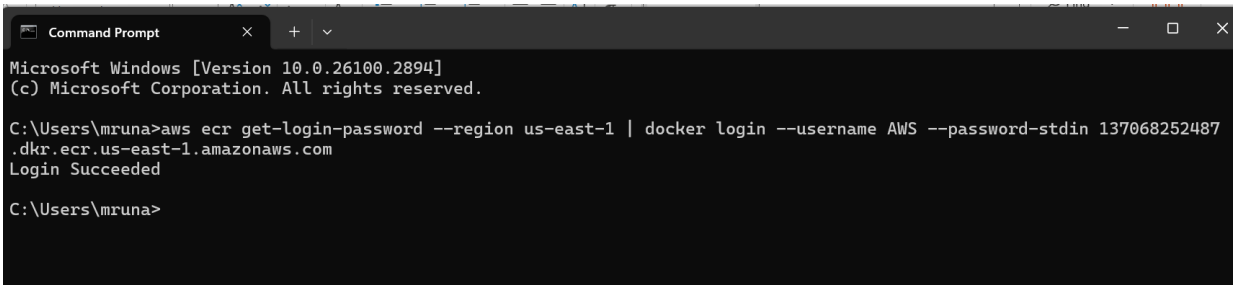
- Creating an ECR Repository called **'myblogapp_hw1'**





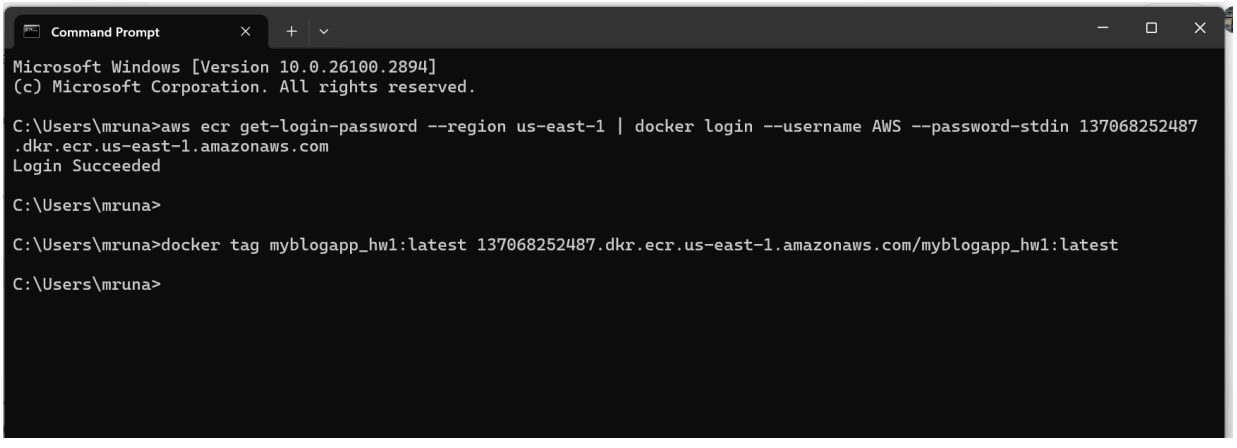
Authenticated Docker to ECR

aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 137068252487.dkr.ecr.us-east-1.amazonaws.com



Docker Image Tagged

docker tag myblogapp_hw1:latest 137068252487.dkr.ecr.us-east-1.amazonaws.com/myblogapp_hw1:latest

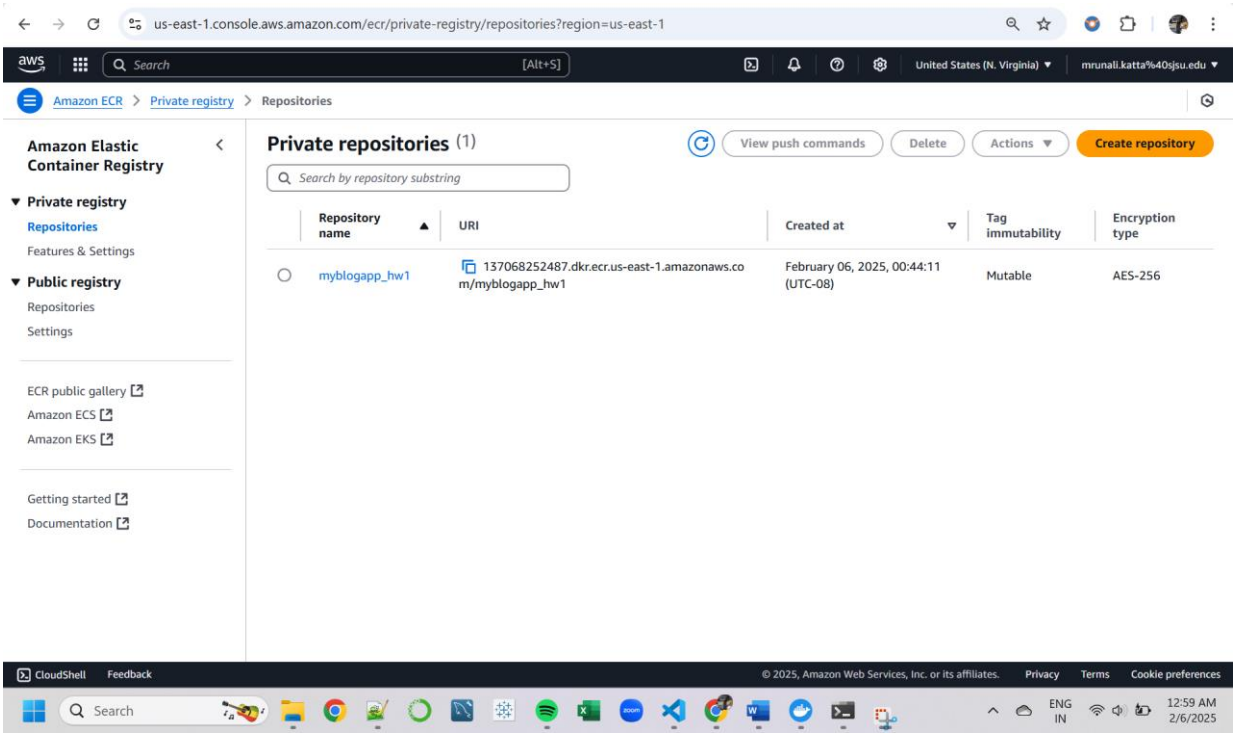


- Push the Docker Image to AWS ECR

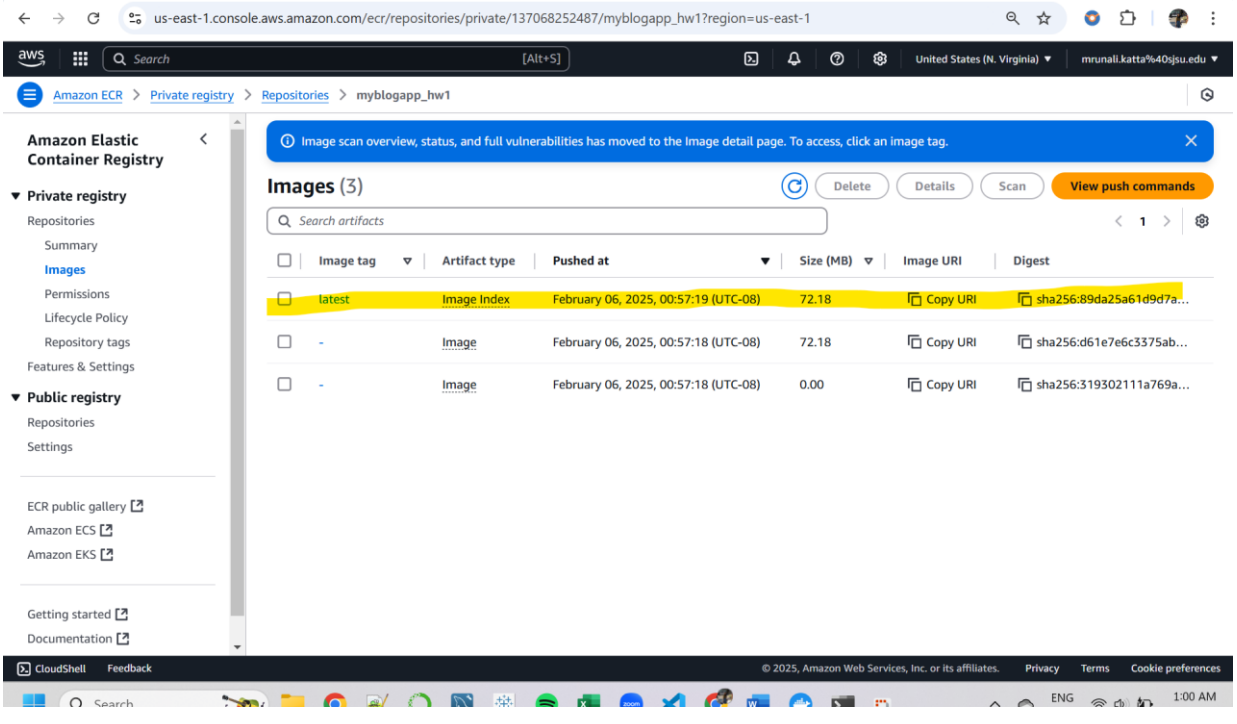
docker push 137068252487.dkr.ecr.us-east-1.amazonaws.com/myblogapp_hw1:latest

```
C:\Users\mruna>docker push 137068252487.dkr.ecr.us-east-1.amazonaws.com/myblogapp_hw1:latest
The push refers to repository [137068252487.dkr.ecr.us-east-1.amazonaws.com/myblogapp_hw1]
1e03dd033147: Already exists
26c43257ce17: Layer already exists
6c78b0ba1a32: Layer already exists
e19db8451adb: Pushed
84cade77a831: Layer already exists
c29f5b76f736: Pushed
24ff42a0d907: Layer already exists
c558df217949: Layer already exists
4f4fb700ef54: Layer already exists
976e8f6b25dd: Layer already exists
b61765e9c6e2: Layer already exists
a8e654ce6927: Layer already exists
latest: digest: sha256:89da25a61d9d7a73a86579266aece486dd57d5a89dbb181b2b30aa9aba150c1b size: 856

C:\Users\mruna>
```

the image is present in AWS ECR

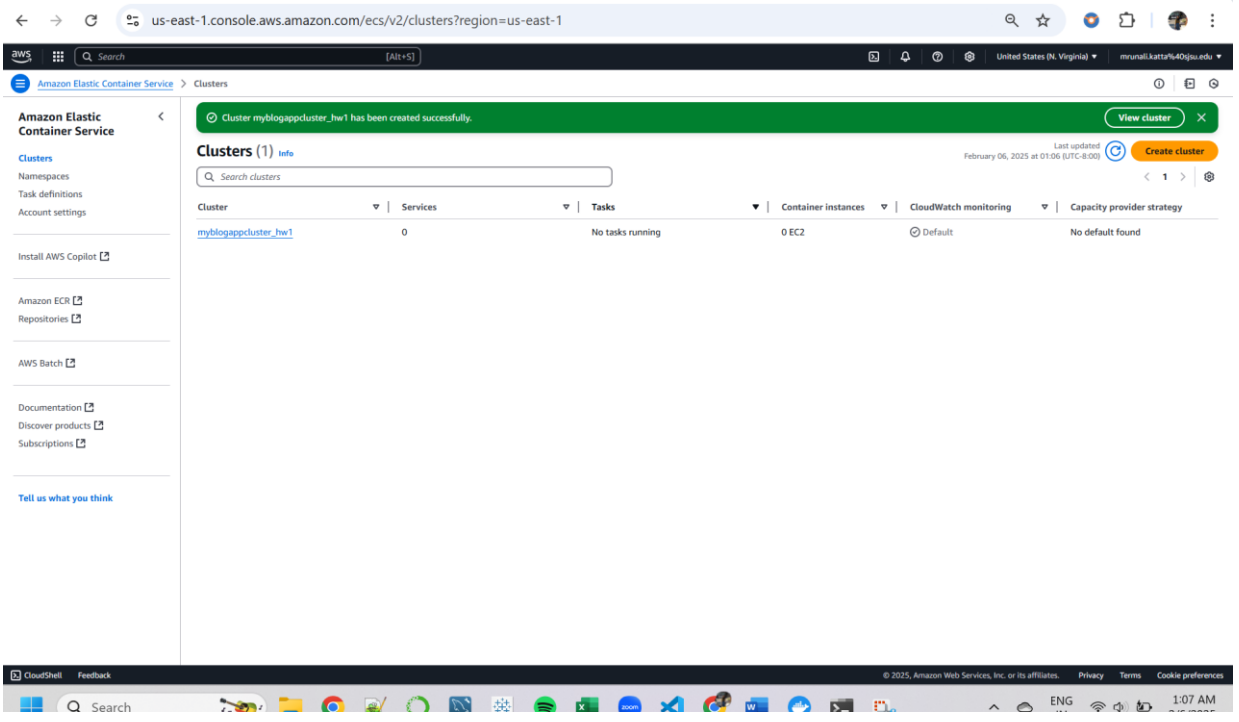


the image is present inside the repository

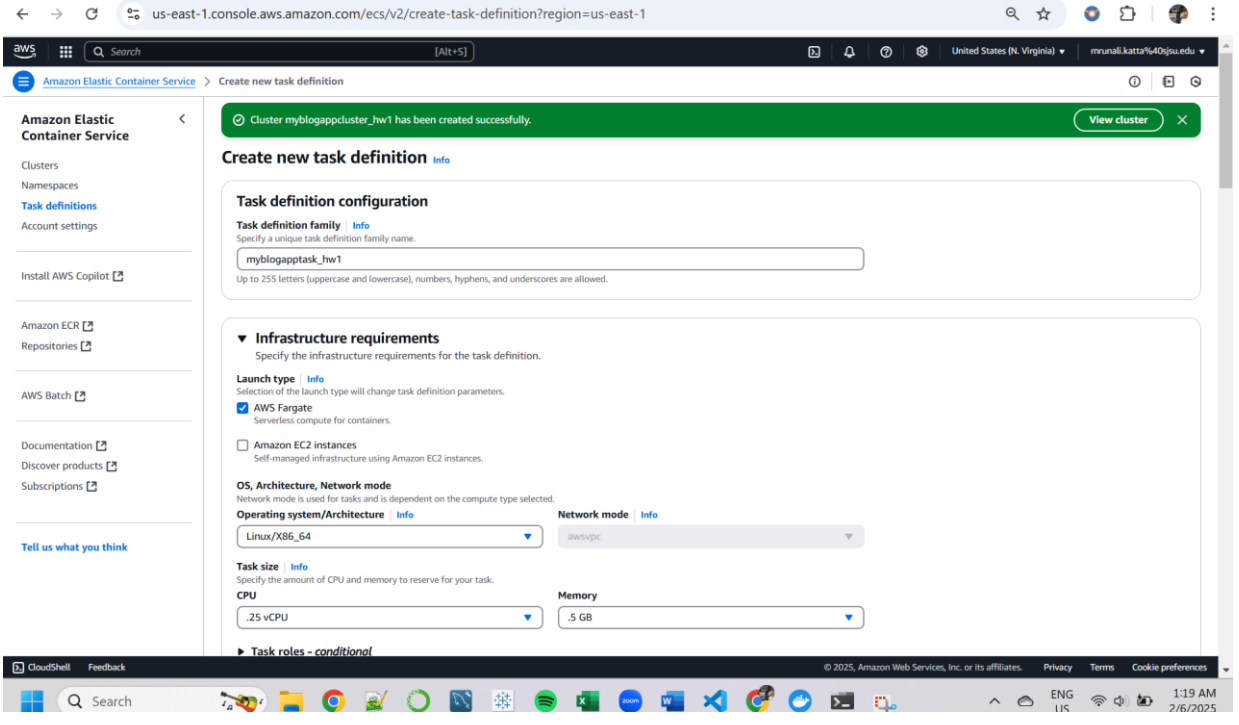


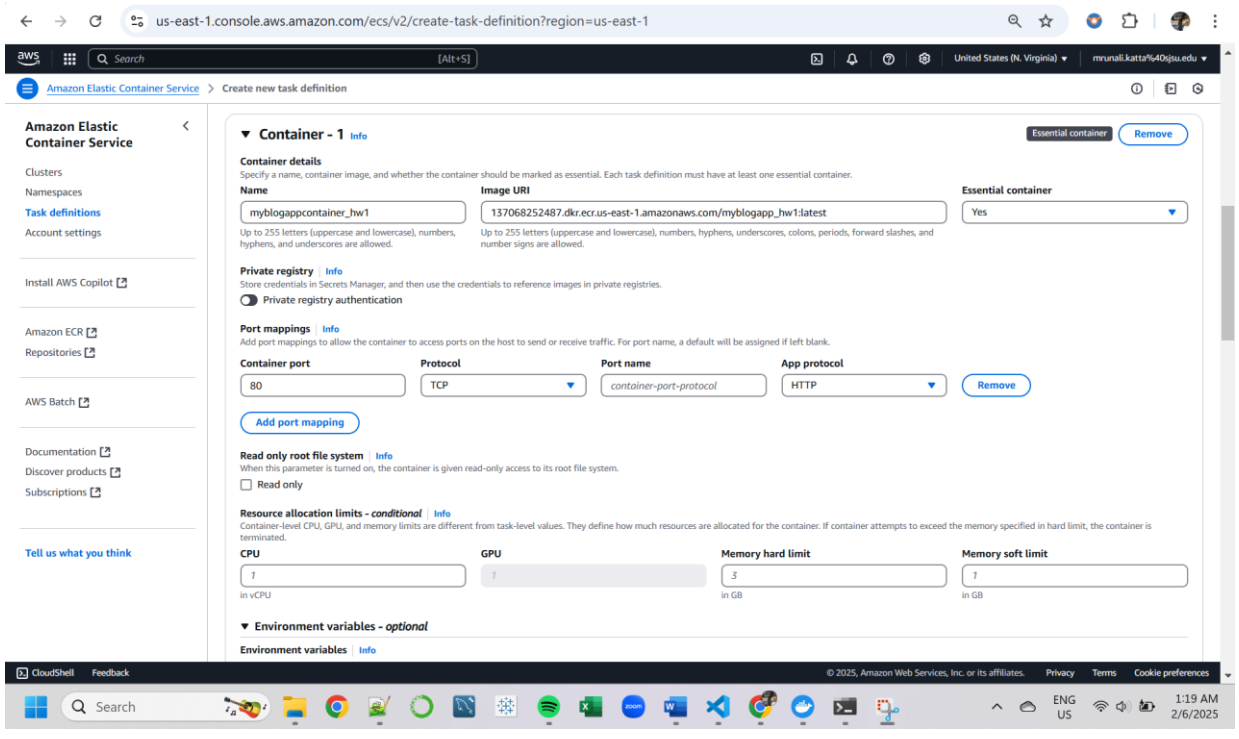
Deploy to AWS ECS

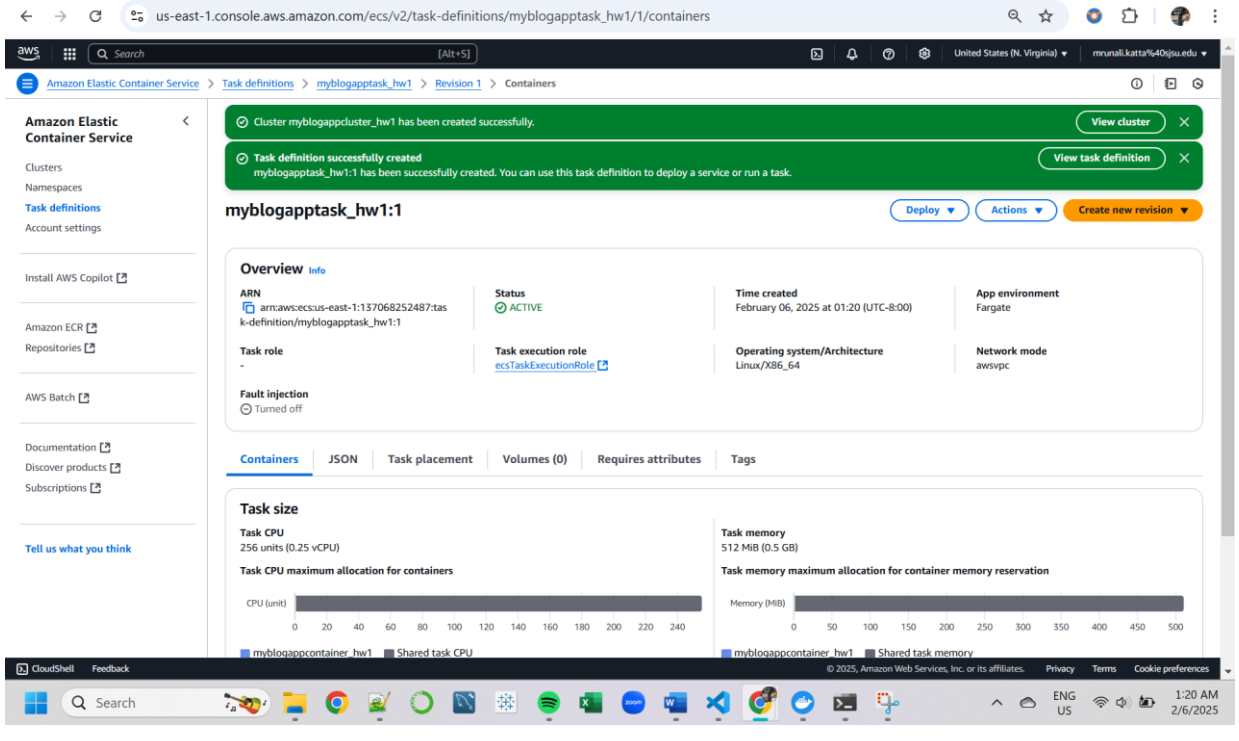
- Create an ECS Cluster called **‘myblogappcluster_hw1 ’**



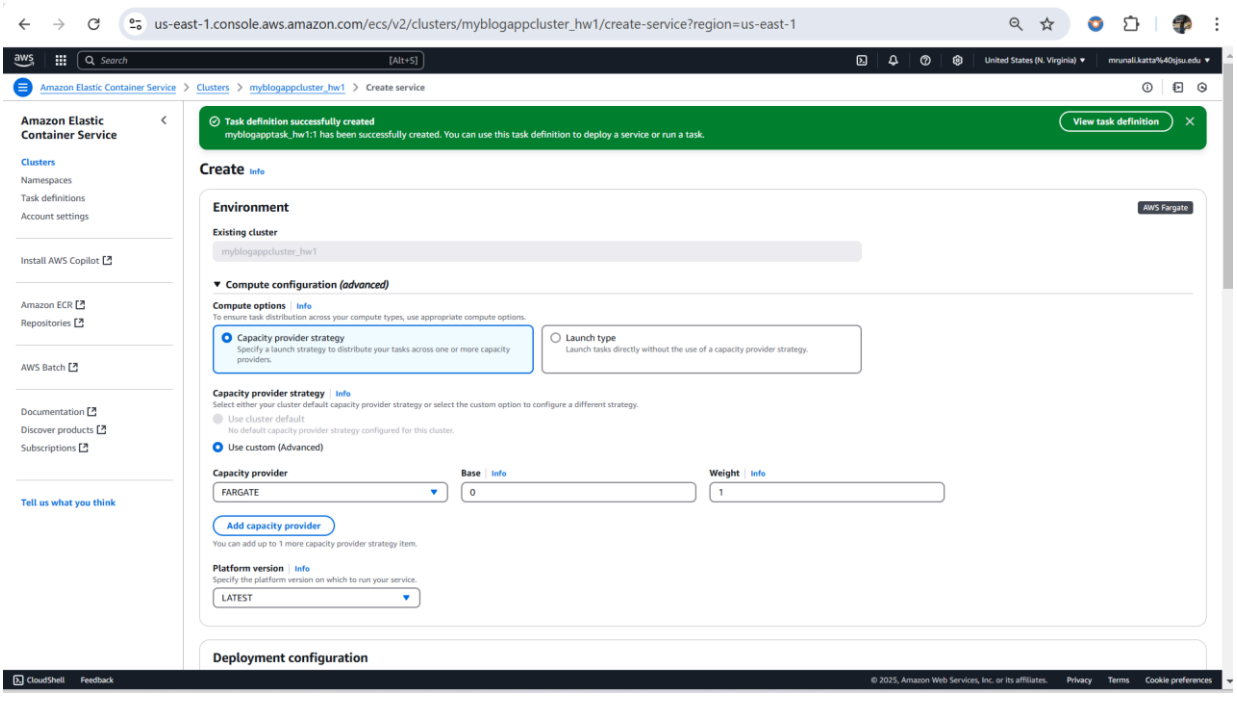
Creating a new task

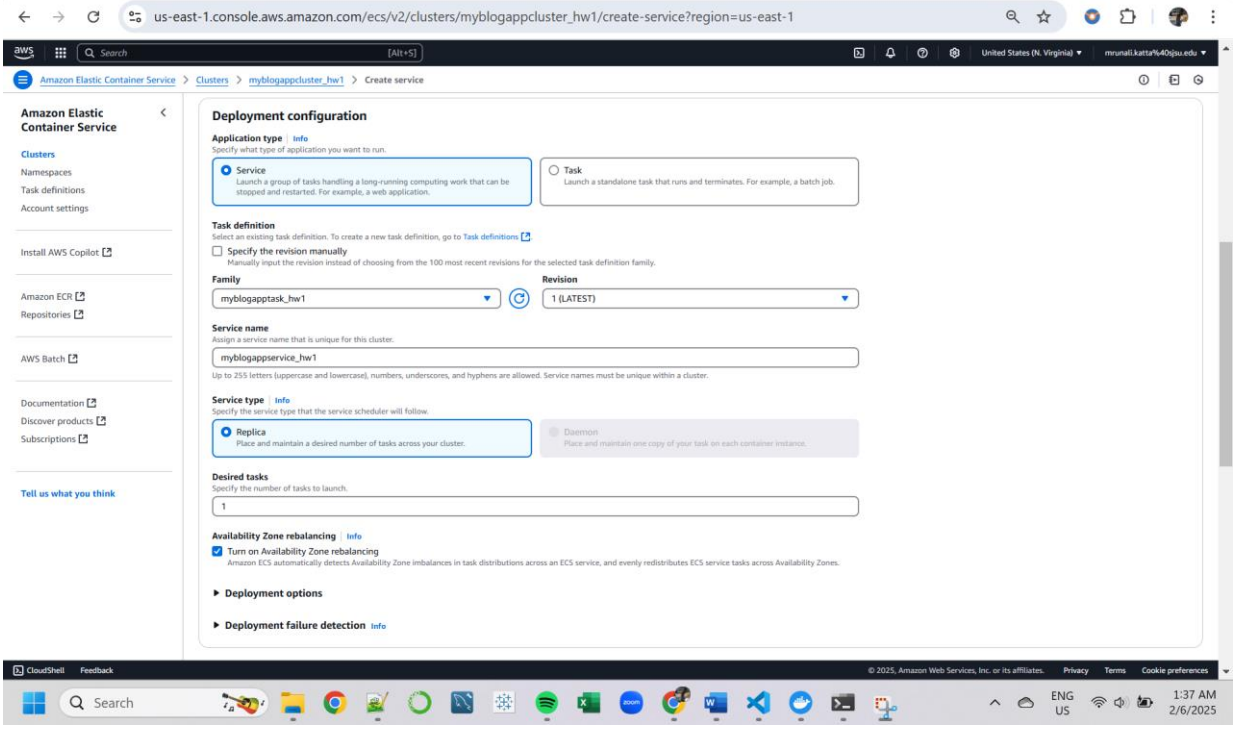


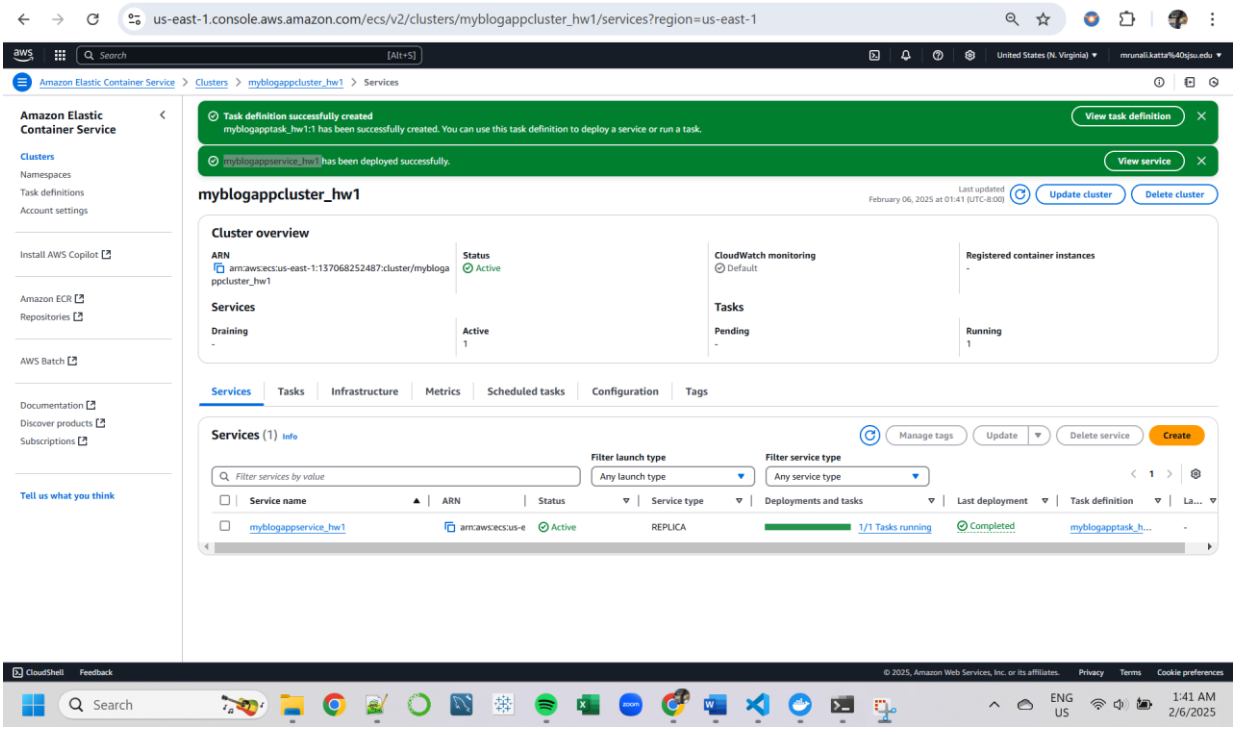

us-east-1.console.aws.amazon.com/ecs/v2/create-task-definition?region=us-east-1
Search [Alt+S]
United States (N. Virginia)
mrunali.katta%40sjsu.edu
Amazon Elastic Container Service > Create new task definition
Amazon Elastic Container Service
Clusters
Namespaces
Task definitions
Account settings
Install AWS Copilot
Amazon ECR
Repositories
AWS Batch
Documentation
Discover products
Subscriptions
Tell us what you think
Container - 1 Info
Essential container
Remove
Container details
Specify a name, container image, and whether the container should be marked as essential. Each task definition must have at least one essential container.
Name
myblogappcontainer_hw1
Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.
Image URI
137068252487.dkr.ecr.us-east-1.amazonaws.com/myblogapp_hw1:latest
Up to 255 letters (uppercase and lowercase), numbers, hyphens, underscores, colons, periods, forward slashes, and number signs are allowed.
Essential container
Yes
Private registry Info
Store credentials in Secrets Manager, and then use the credentials to reference images in private registries.
Private registry authentication
Port mappings Info
Add port mappings to allow the container to access ports on the host to send or receive traffic. For port name, a default will be assigned if left blank.
Container port
80
Protocol
TCP
Port name
container-port-protocol
App protocol
HTTP
Remove
Add port mapping
Read only root file system Info
When this parameter is turned on, the container is given read-only access to its root file system.
Read only
Resource allocation limits - conditional Info
Container-level CPU, GPU, and memory limits are different from task-level values. They define how much resources are allocated for the container. If container attempts to exceed the memory specified in hard limit, the container is terminated.
CPU
1
in vCPU
GPU
Memory hard limit
3
in GB
Memory soft limit
1
in GB
Environment variables - optional
Environment variables Info
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG US
1:19 AM
2/6/2025


us-east-1.console.aws.amazon.com/ecs/v2/task-definitions/myblogapptask_hw1/1/containers
Search [Alt+S]
United States (N. Virginia)
mrunali.katta%40sjsu.edu
Amazon Elastic Container Service > Task definitions > myblogapptask_hw1 > Revision 1 > Containers
Amazon Elastic Container Service
Clusters
Namespaces
Task definitions
Account settings
Install AWS Copilot
Amazon ECR
Repositories
AWS Batch
Documentation
Discover products
Subscriptions
Tell us what you think
Cluster myblogappcluster_hw1 has been created successfully.
View cluster
Task definition successfully created
myblogapptask_hw1:1 has been successfully created. You can use this task definition to deploy a service or run a task.
View task definition
myblogapptask_hw1:1
Deploy
Actions
Create new revision
Overview Info
ARN
arn:aws:ecs:us-east-1:137068252487:task-definition/myblogapptask_hw1:1
Status
ACTIVE
Time created
February 06, 2025 at 01:20 (UTC-8:00)
App environment
Fargate
Task role
-
Task execution role
ecsTaskExecutionRole
Operating system/Architecture
Linux/X86_64
Network mode
awsvpc
Fault injection
Turned off
Containers
JSON
Task placement
Volumes (0)
Requires attributes
Tags
Task size
Task CPU
256 units (0.25 vCPU)
Task memory
512 MiB (0.5 GB)
Task CPU maximum allocation for containers
CPU (unit)
0 20 40 60 80 100 120 140 160 180 200 220 240
myblogappcontainer_hw1
Shared task CPU
Task memory maximum allocation for container memory reservation
Memory (MiB)
0 50 100 150 200 250 300 350 400 450 500
myblogappcontainer_hw1
Shared task memory
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG US
1:20 AM
2/6/2025

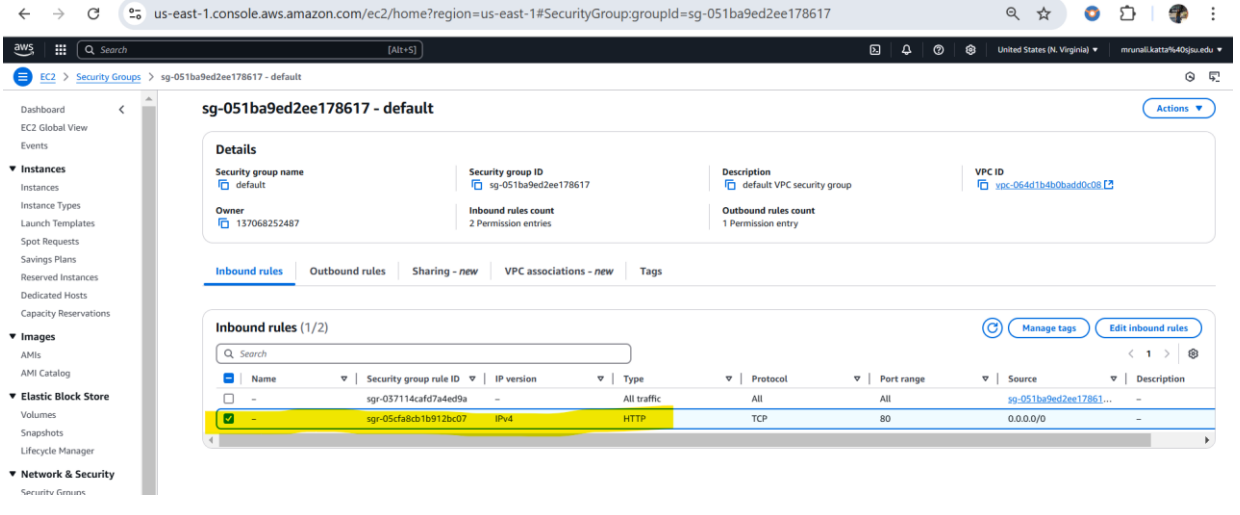
- Created a service **'myblogappservice_hw1'** in AWS ECS using the Fargate launch type



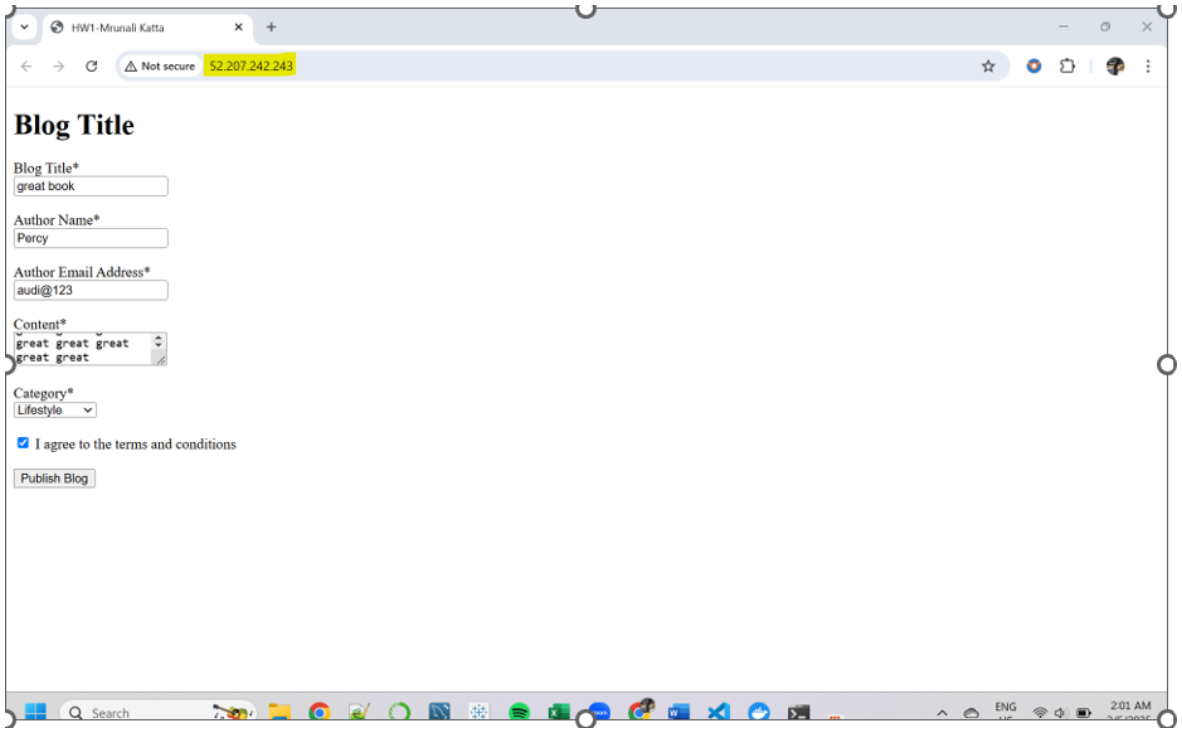


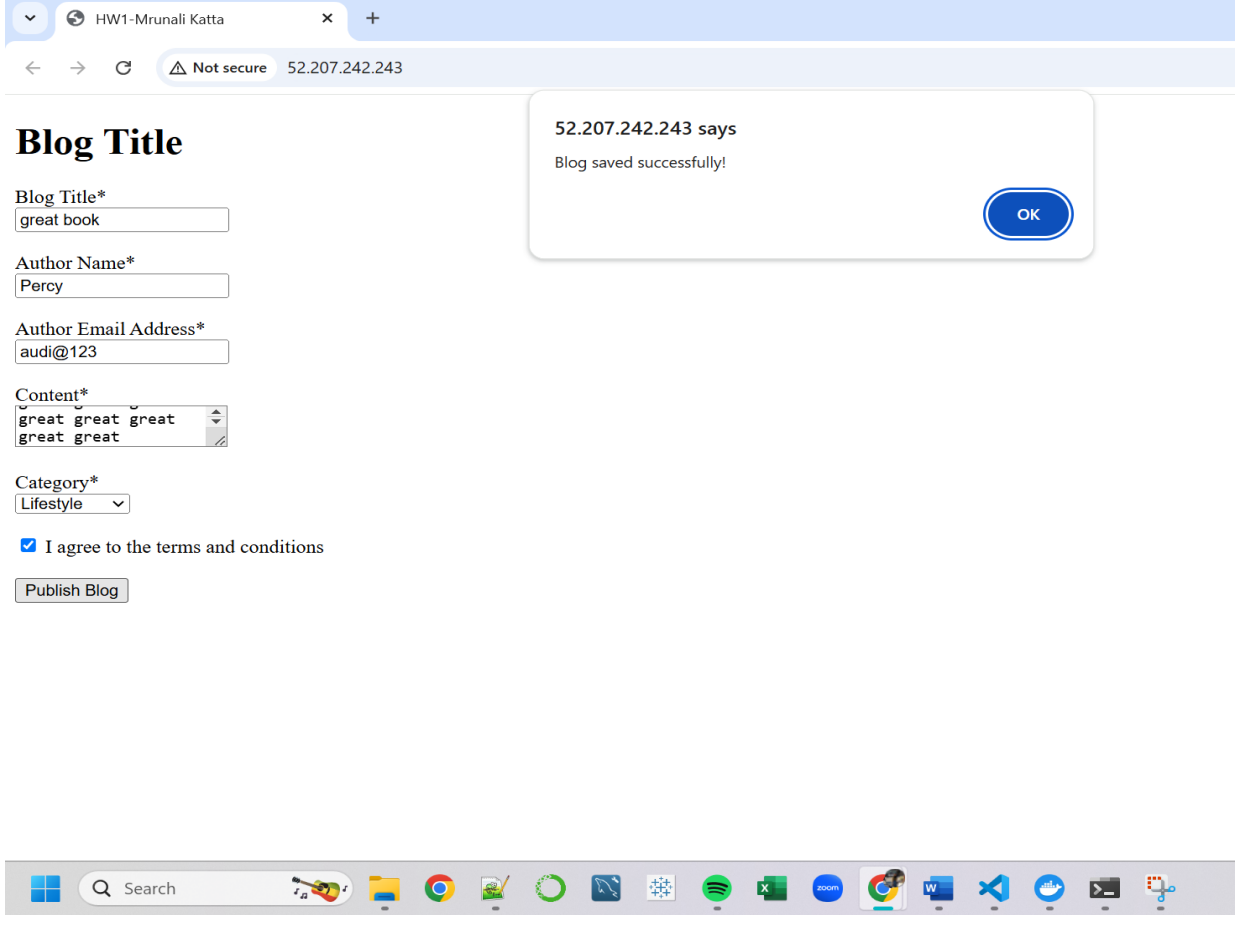


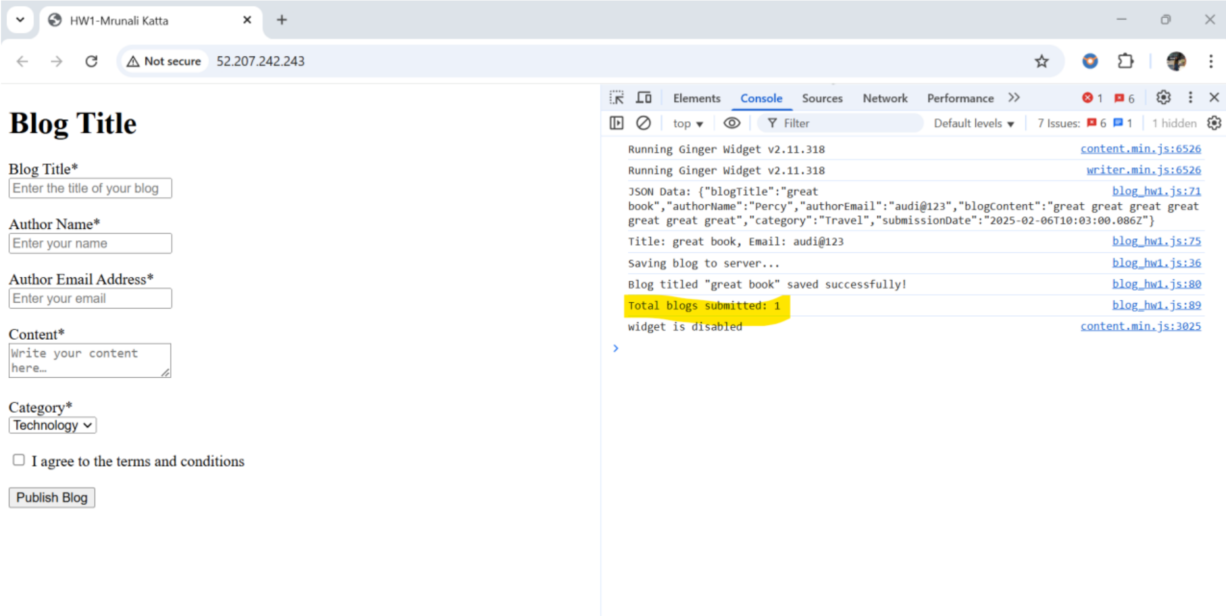
Was unable to access the application. Was getting error “This site can’t be reached” so had to add a new rule in order to access the application.



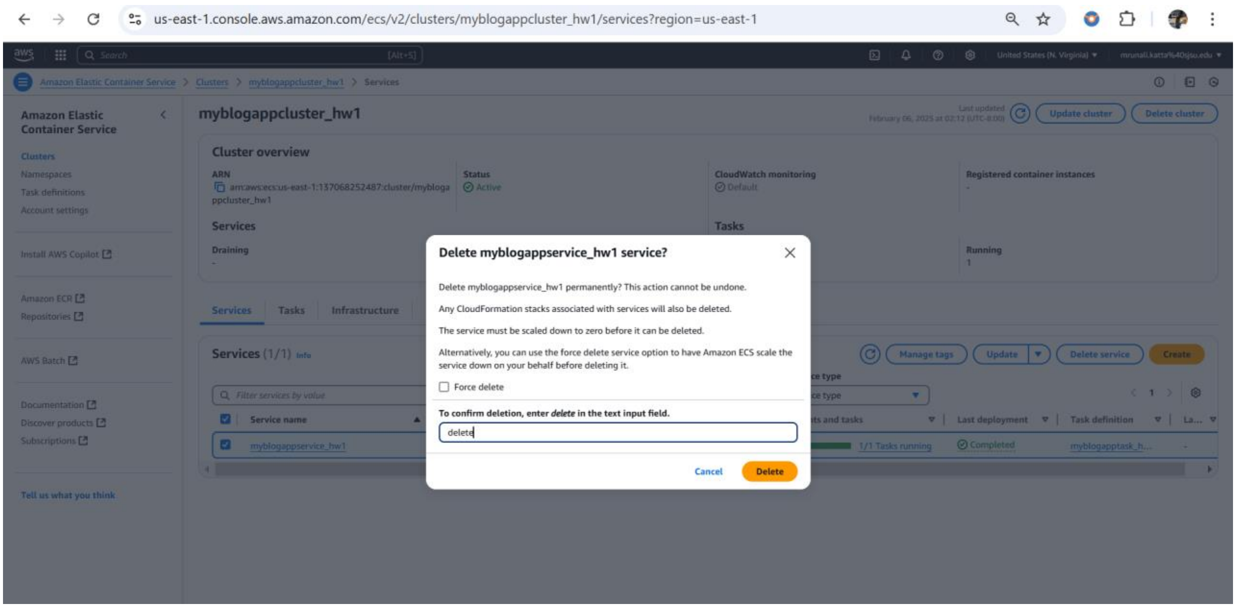
Successfully accessed application for public IP address **52.207.242.243**

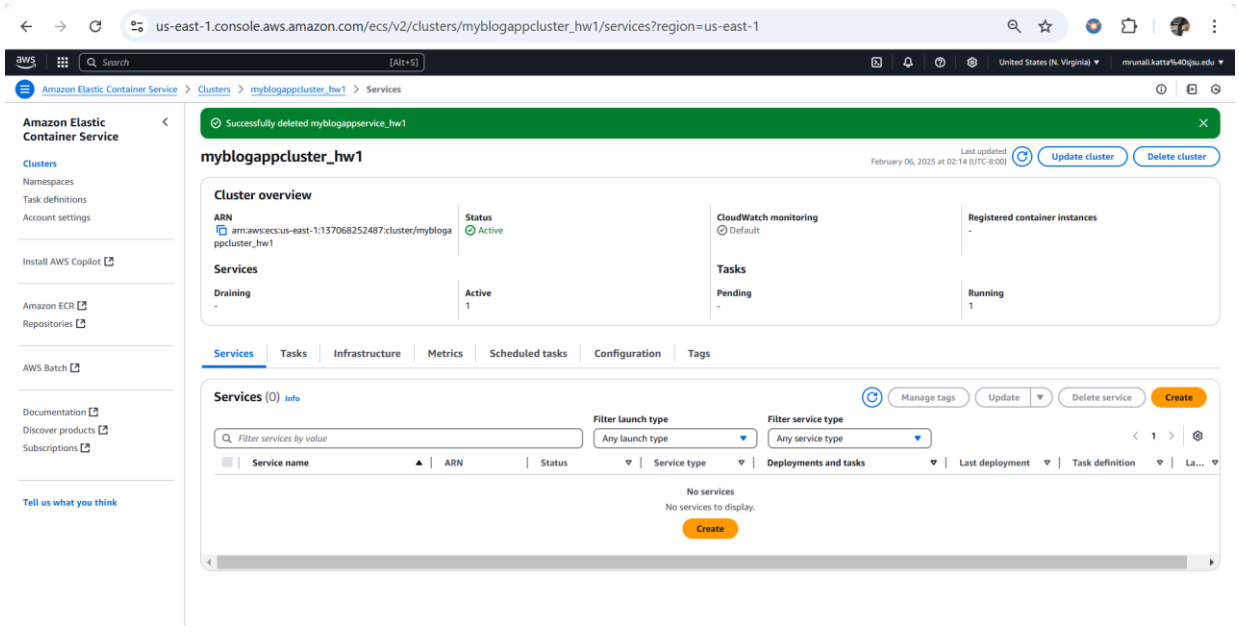




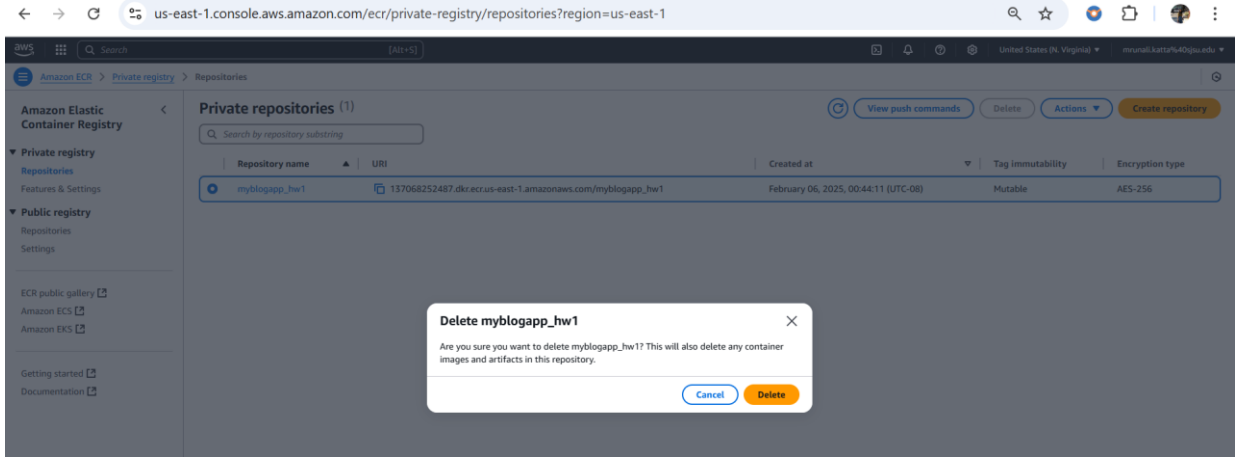


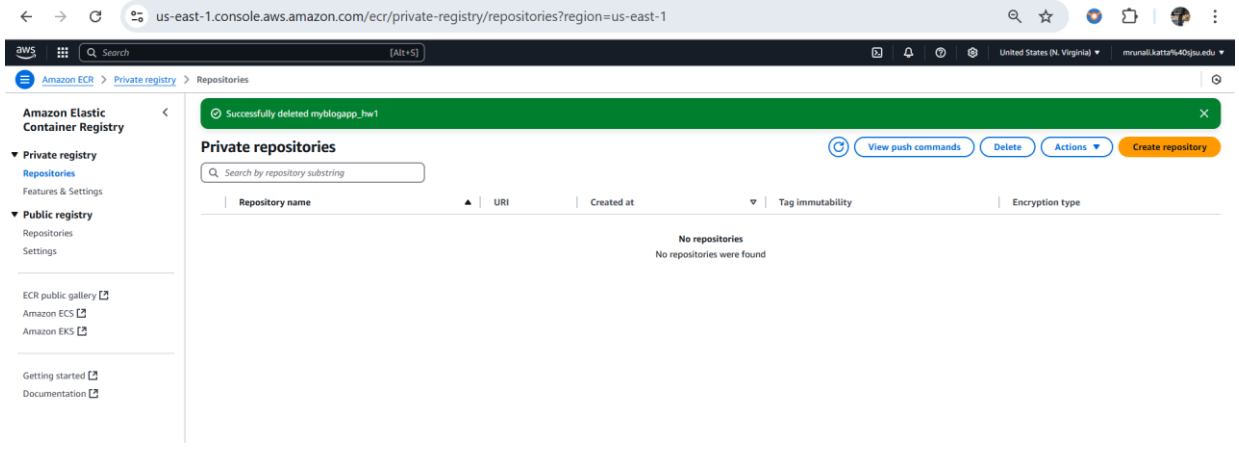
Delete ECS Service





Deleted ECR repository





Deleted ECS Cluster