Experiment – 6

**Aim:** To build your application using AWS CodeBuild and Deploy on S3/SEBS using AWS CodePipeline, deploy sample application on EC2 instance using AWS CodeDeploy.

**Theory:**

**Overview of AWS DevOps Services**

**1. AWS CodeBuild**

- **Definition:** AWS CodeBuild is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy.

- **Key Features:**

- **Managed Build Environment:** Automatically provisions the compute resources needed for builds.

- **Custom Build Environments:** Allows you to define your own build environment with Docker images.

- **Integration:** Works seamlessly with other AWS services and popular version control systems.

**2. AWS CodeDeploy**

- **Definition:** AWS CodeDeploy is a fully managed deployment service that automates application deployments to various compute services such as Amazon EC2, AWS Lambda, and on-premises servers.

- **Key Features:**

- **Rolling Updates:** Supports various deployment strategies (e.g., in-place or blue/green deployments).

- **Monitoring and Rollback:** Monitors deployments and can automatically roll back changes if issues are detected.

**3. AWS CodePipeline**

- **Definition:** AWS CodePipeline is a continuous delivery service for fast and reliable application updates. It automates the build, test, and release process.

- **Key Features:**

- **Workflow Automation:** Defines the workflow of your CI/CD process, integrating various AWS services.

- **Version Control Integration:** Supports various version control systems to trigger pipelines on code changes.

- **Extensible:** Can incorporate third-party tools and custom actions.

**Architecture Overview**

In a typical CI/CD pipeline using these services:

**1. Source Stage:** Code is stored in a repository (e.g., GitHub, AWS CodeCommit).

**2. Build Stage:** AWS CodeBuild compiles the code and runs tests.

**3. Deploy Stage:** AWS CodeDeploy deploys the built code to EC2 instances or other environments.

**4. Pipeline Management:** AWS CodePipeline orchestrates the flow of changes from source to deployment.

**Step-by-Step Implementation**

**Part 1: Building Your Application with AWS CodeBuild**

**1. Create a CodeBuild Project:**

- Go to the **AWS Management Console** and navigate to **AWS CodeBuild**.

- Click on **Create Build Project**.

- Configure the project settings:

- **Project Name:** Provide a name for your project.

- **Source Provider:** Choose the source provider (e.g., GitHub, CodeCommit) and connect it.

- **Environment:** Select the build environment (e.g., Ubuntu, Amazon Linux) and the runtime.

- **Buildspec:** Create a **buildspec.yml** file in your source repository, which defines the build commands.

- Click **Create Build Project**.

**2. Start a Build:**

- Select the created project and click **Start Build**. This will compile the code and produce artifacts specified in the **buildspec.yml**.

**Part 2: Deploying to S3/SEBS Using AWS CodePipeline**

**1. Create an S3 Bucket:**

- Go to the **S3** service and create a new bucket to store the build artifacts.

**2. Create a CodePipeline:**

- Navigate to **AWS CodePipeline** and click **Create Pipeline**.

- Configure the pipeline:

- **Pipeline Name:** Provide a name for the pipeline.

- **Source Stage:** Select the source provider (e.g., CodeCommit or S3) and specify the repository details.

- **Build Stage:** Choose **AWS CodeBuild** as the build provider and select the build project created earlier.

- **Deploy Stage:** Choose **AWS S3** as the deployment provider and specify the target S3 bucket for storing artifacts.

**3. Review and Create:**

- Review the pipeline configuration and click **Create Pipeline**. The pipeline will automatically run and deploy your application artifacts to S3.

**Part 3: Deploying Sample Application on EC2 Instance Using AWS CodeDeploy**

**1. Create an Application in CodeDeploy:**

- Go to the AWS CodeDeploy console and click on **Create application**.

- Choose **EC2/On-premises** as the compute platform.

- Provide an application name and create it.

**2. Create a Deployment Group:**

- In the application created, create a deployment group.

- Specify the EC2 instances (by tags or instance IDs) that should receive the deployment.

- Configure deployment settings, including the service role that CodeDeploy will use.

**3. Prepare the AppSpec File:**

- Create an **appspec.yml** file in your source repository.

**4. Deploy the Application:**

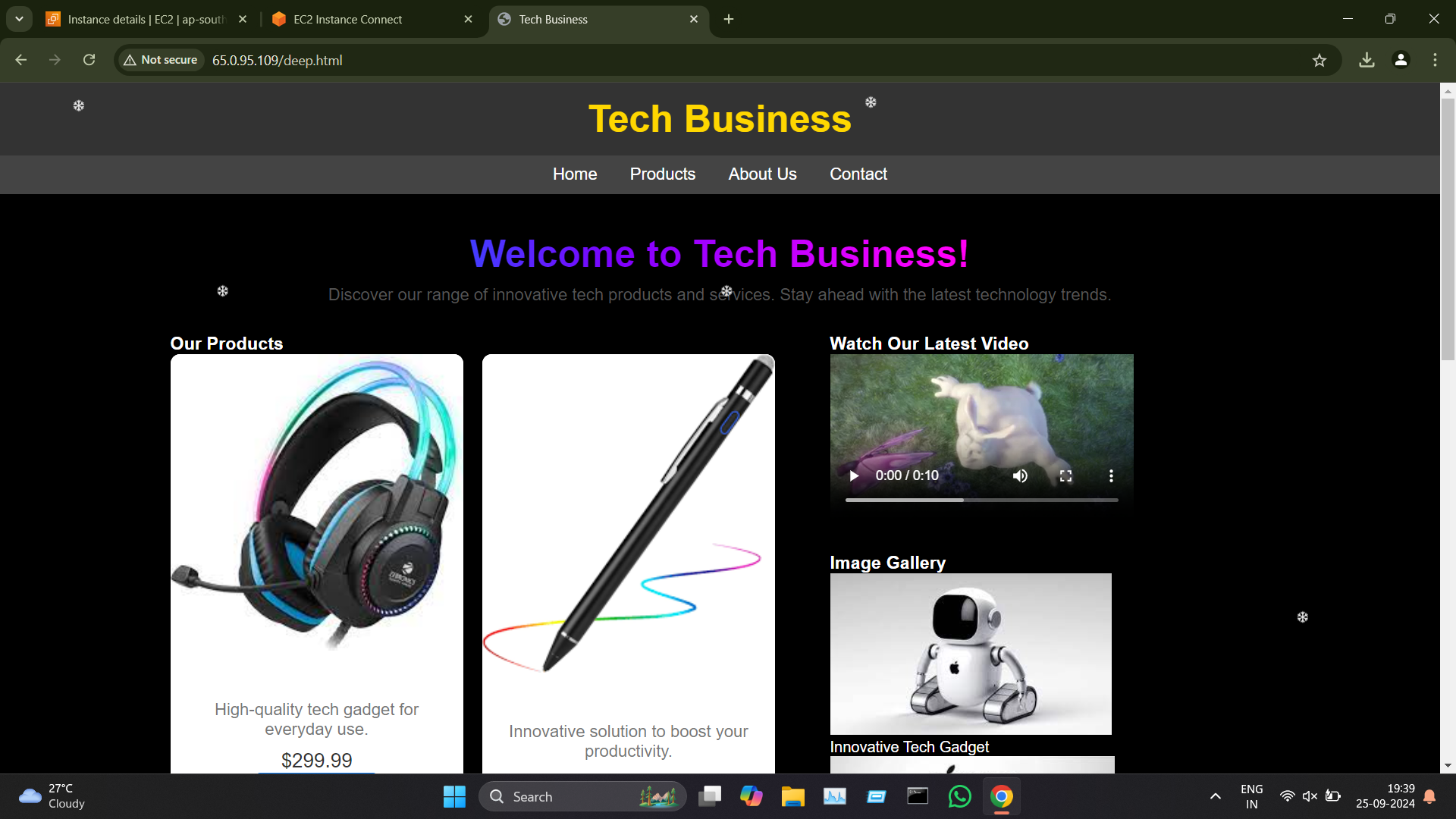
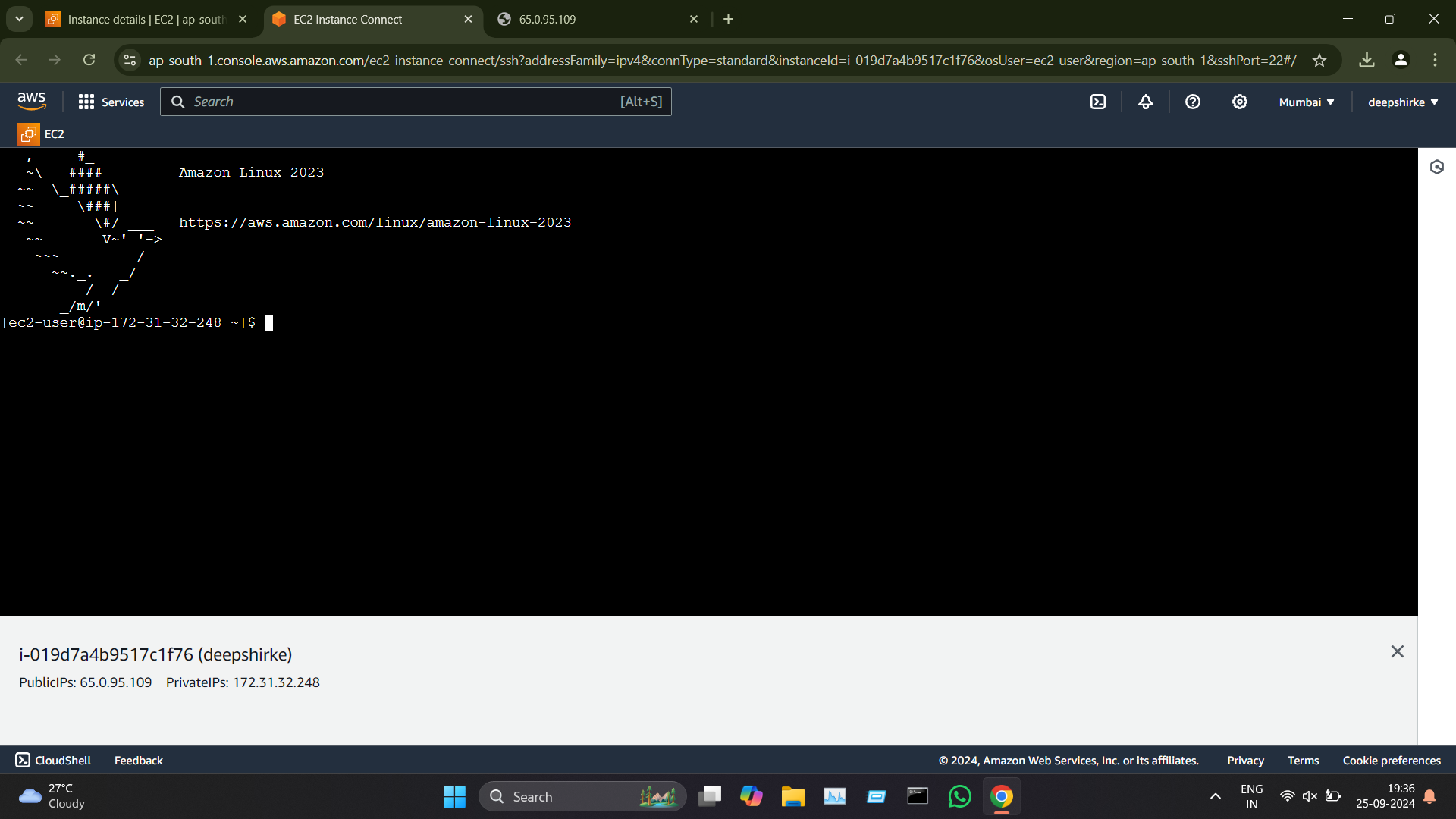
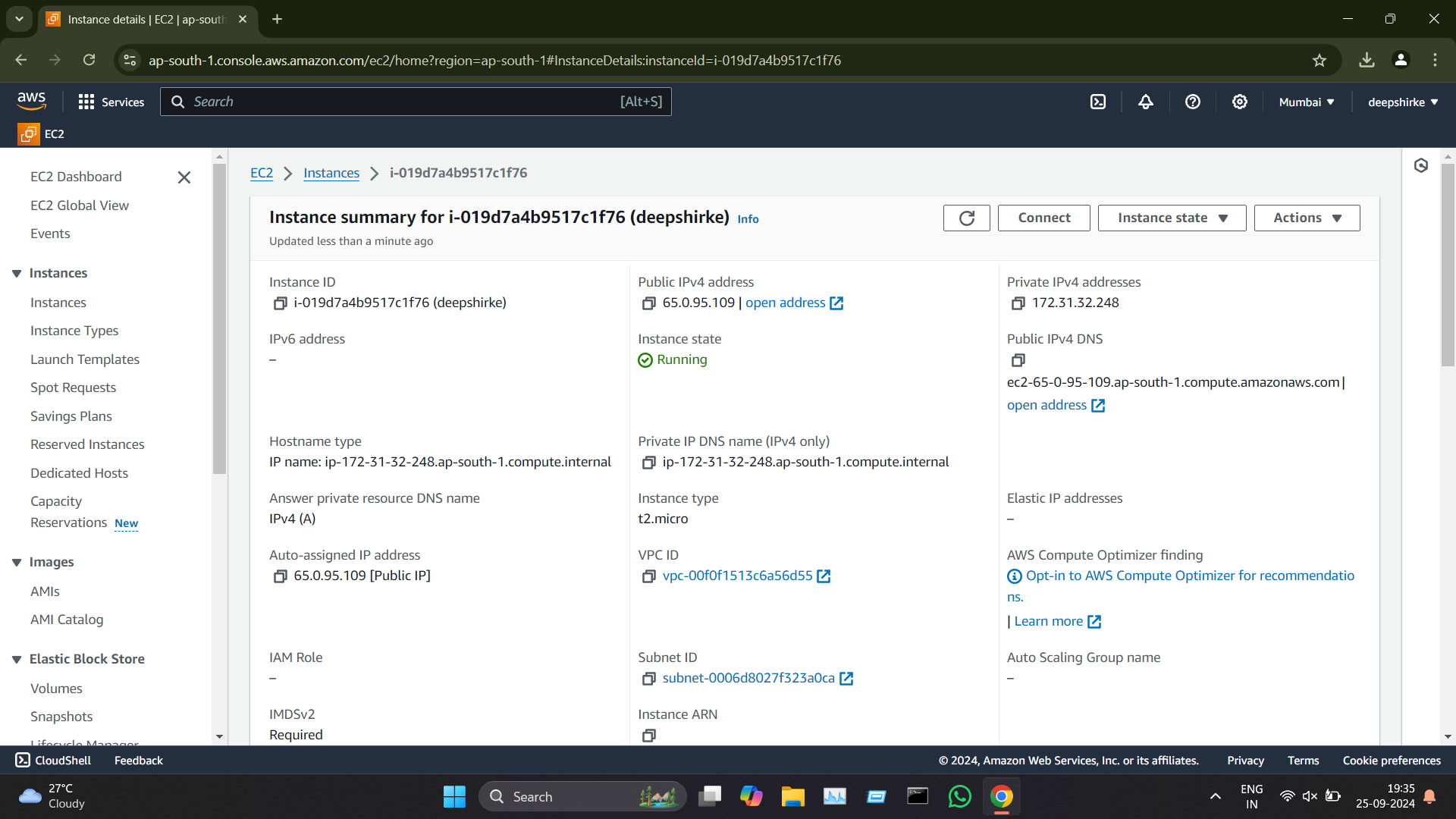
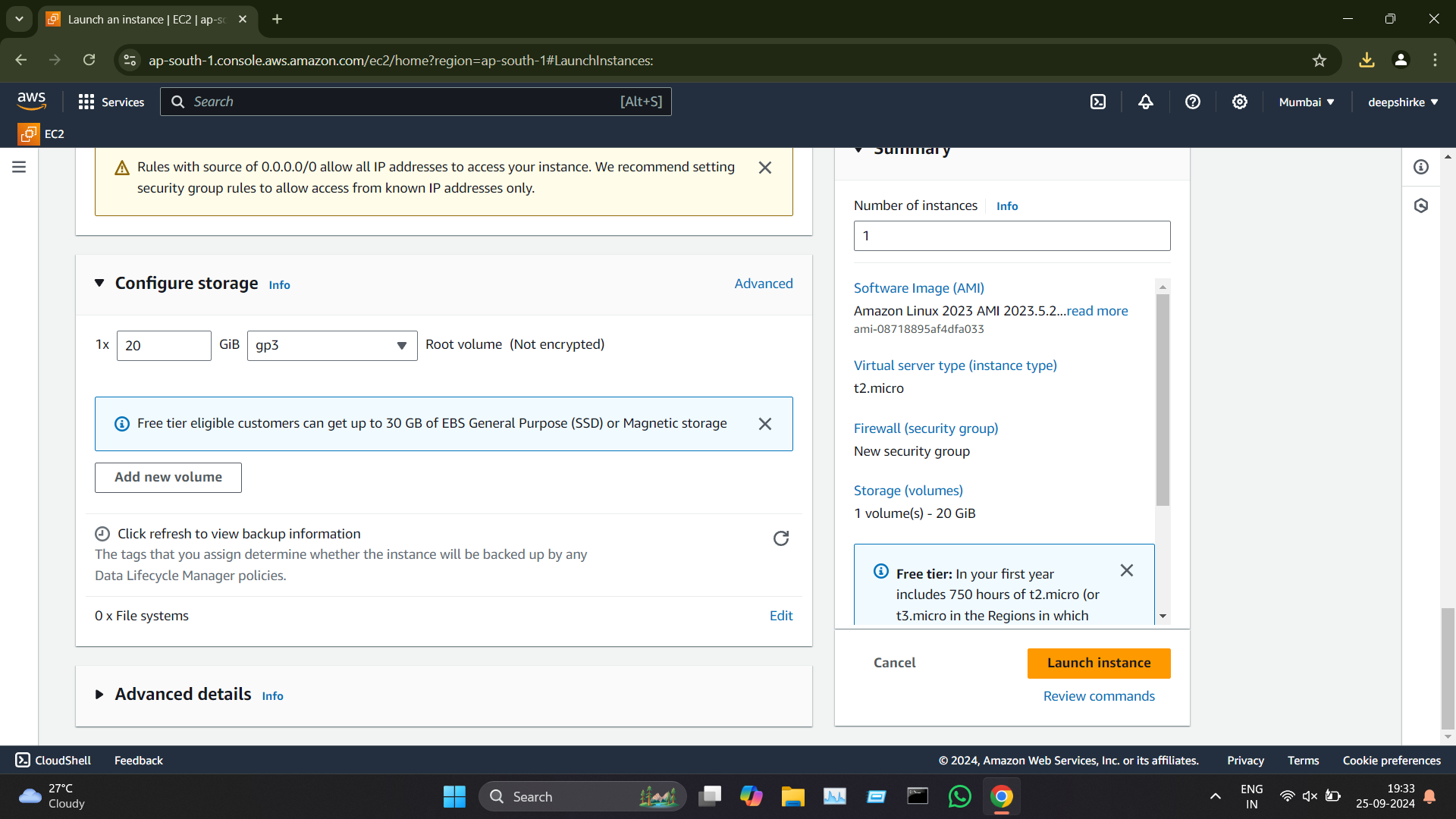
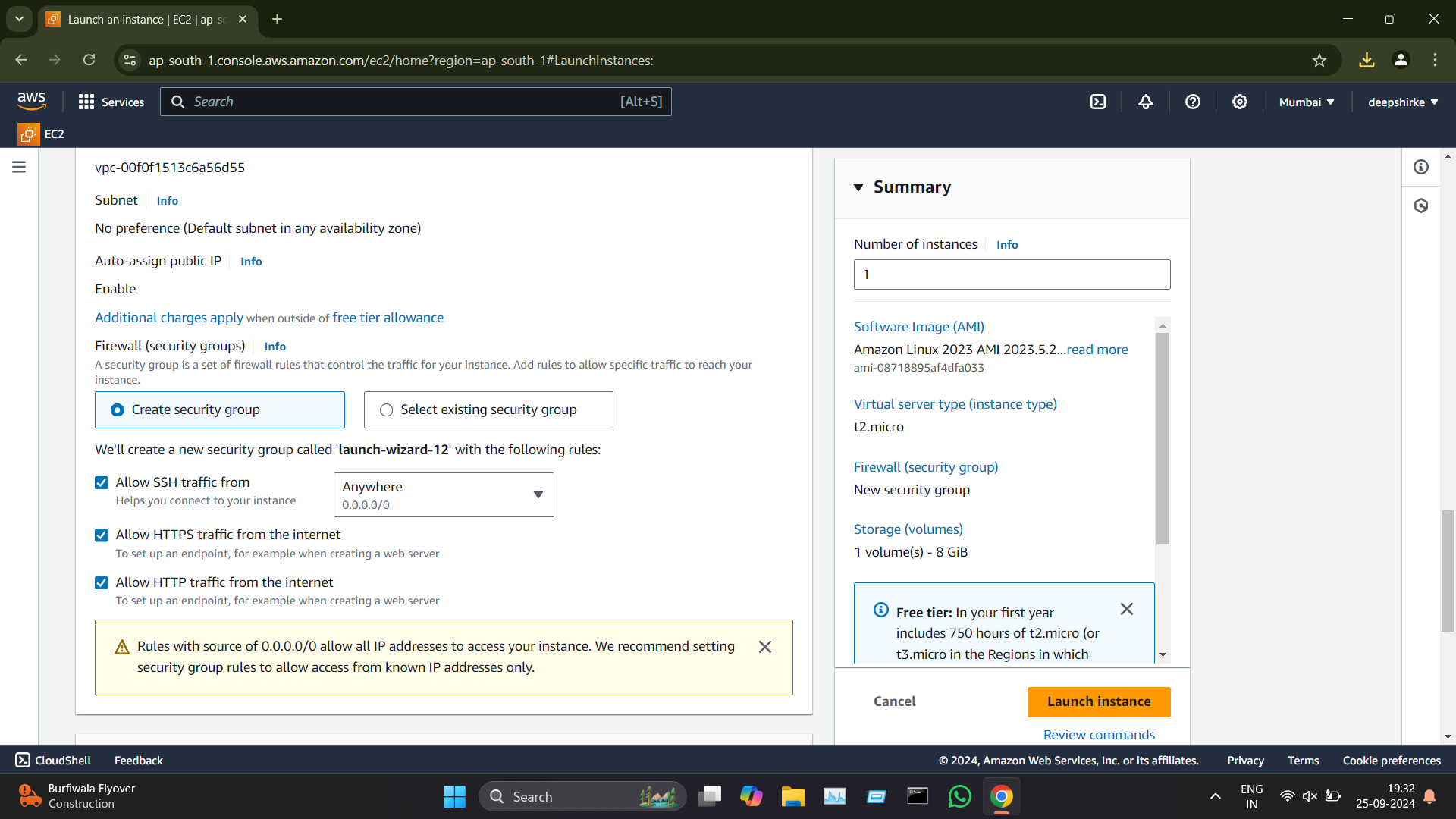
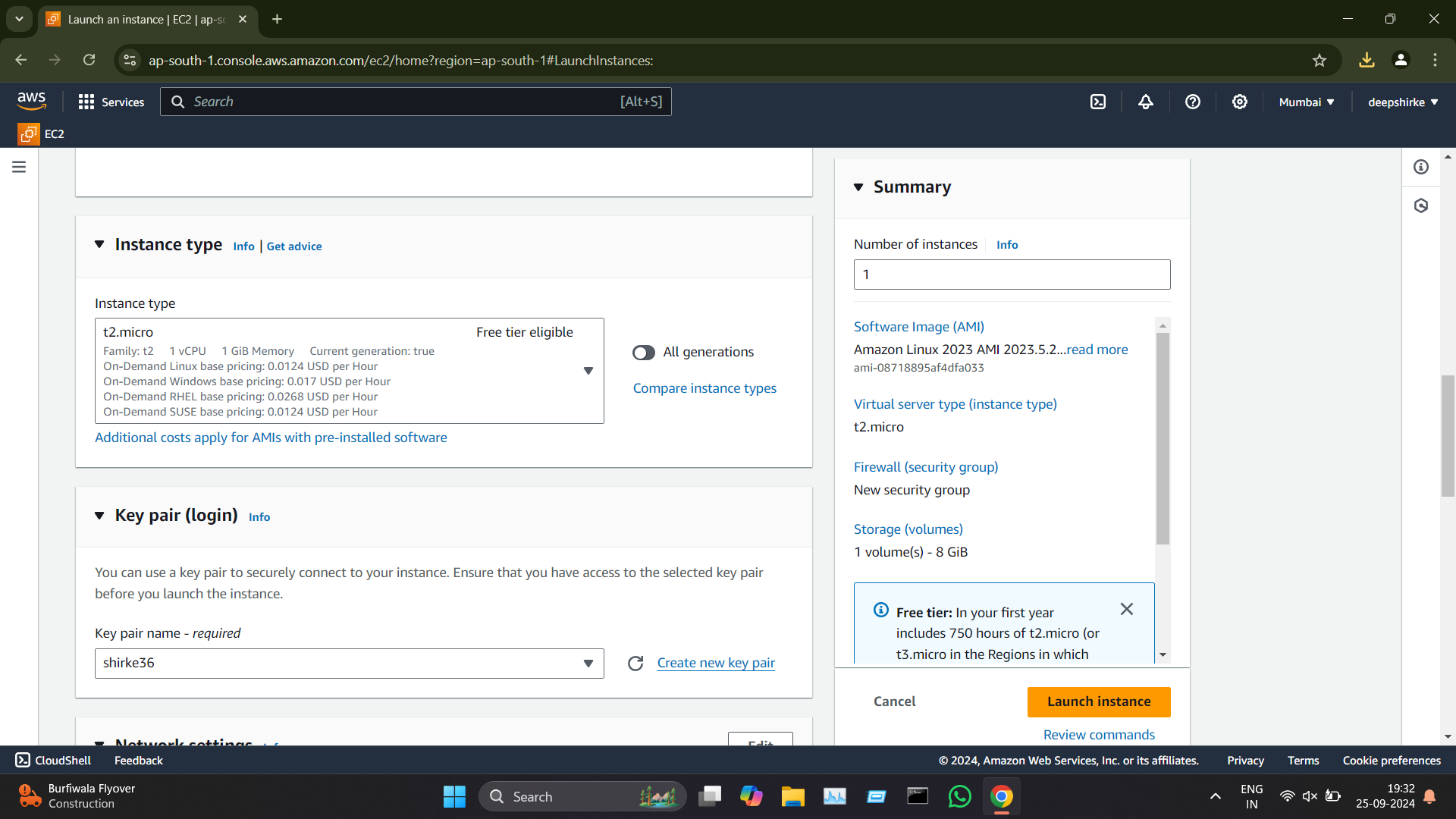
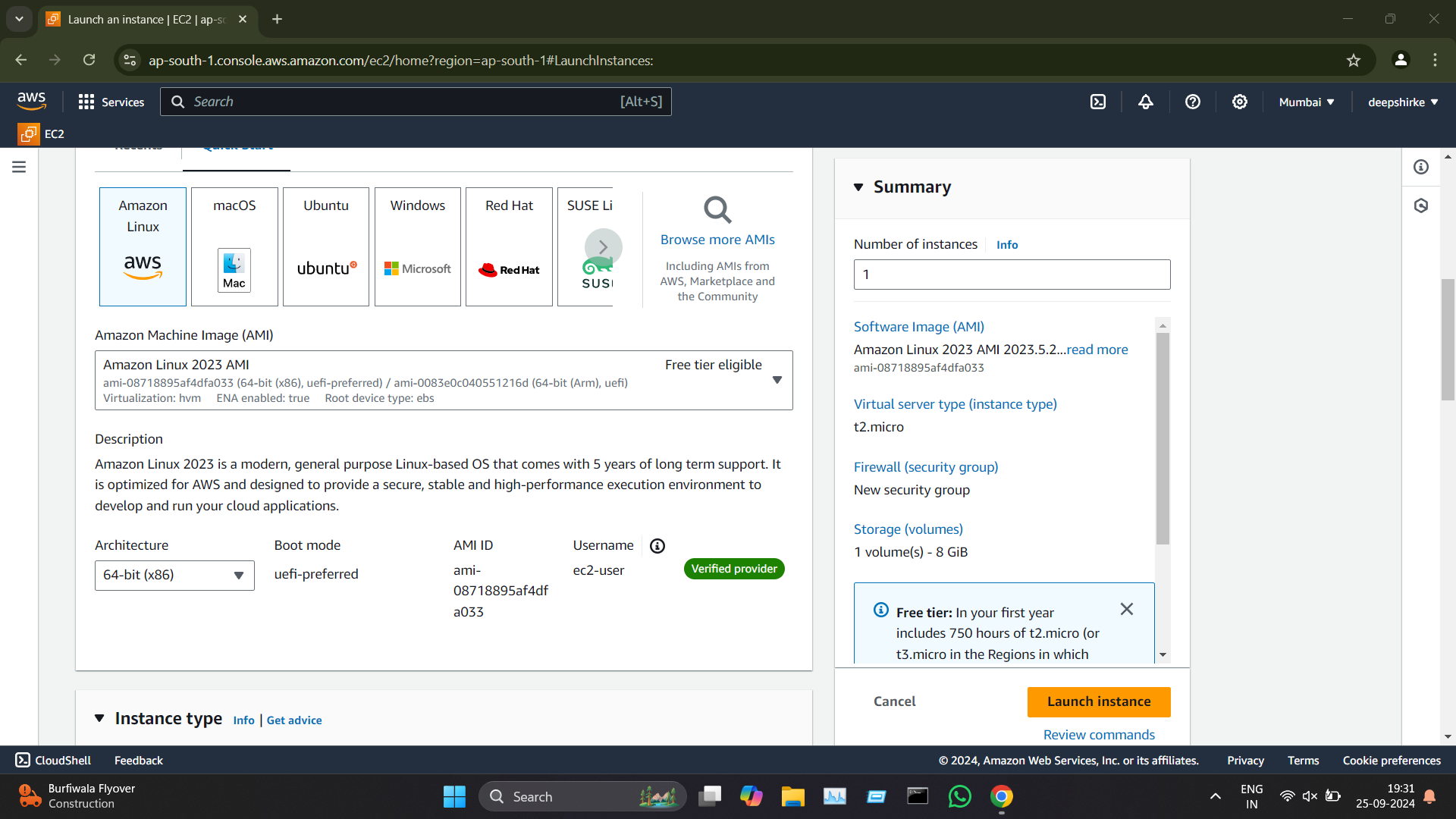
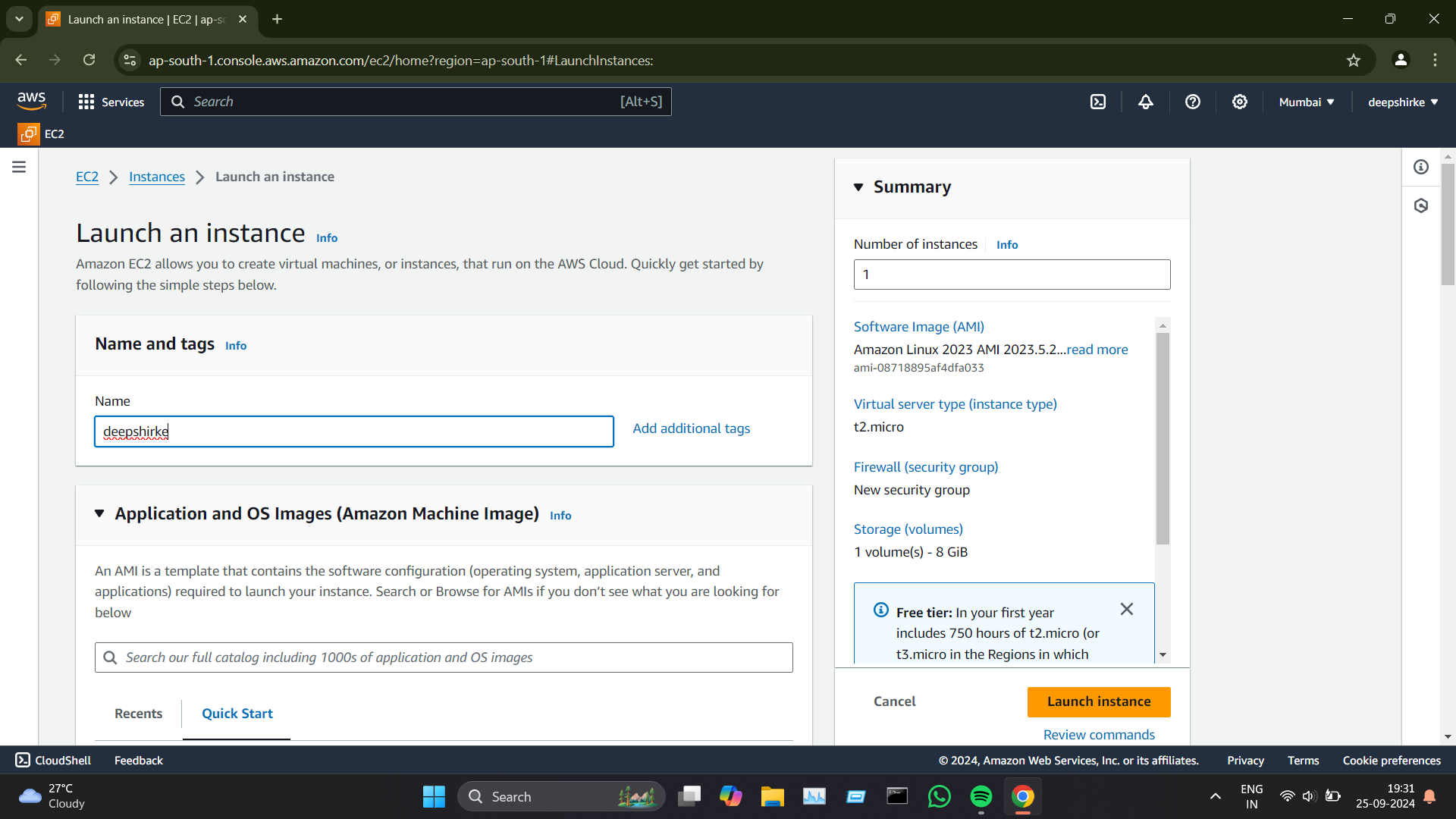
- In the CodeDeploy console, select the created application and click **Create deployment**.

- Choose the deployment group and the S3 location of your artifacts.

- Click **Create deployment**. CodeDeploy will deploy the application to the specified EC2 instances.

**Experiment:**

**EC2**



**S3**

**IAM**

