This updated project task document addresses some of the errors in the original publication with the commands used to configure the instance for AMI creation. Please follow these instructions instead of the original publication.

1. From the AWS management console, search for EC2 in the top search bar and navigate to the EC2 management console.
2. From the left-hand menu, select Instances
3. In the right-hand pane, click the **Launch instances** button
4. In the **Launch an instance** console, provide a name for your instance, such as **Todo AMI Image Instance**
5. Select **Amazon Linux** under the **Quick Start** tab and ensure that **Amazon Linux 2023 AMI** is selected as the image you will work with to create your custom AMI. Also, ensure that you choose **64-bit x86** architecture, as per the following screenshot in Figure 5.12

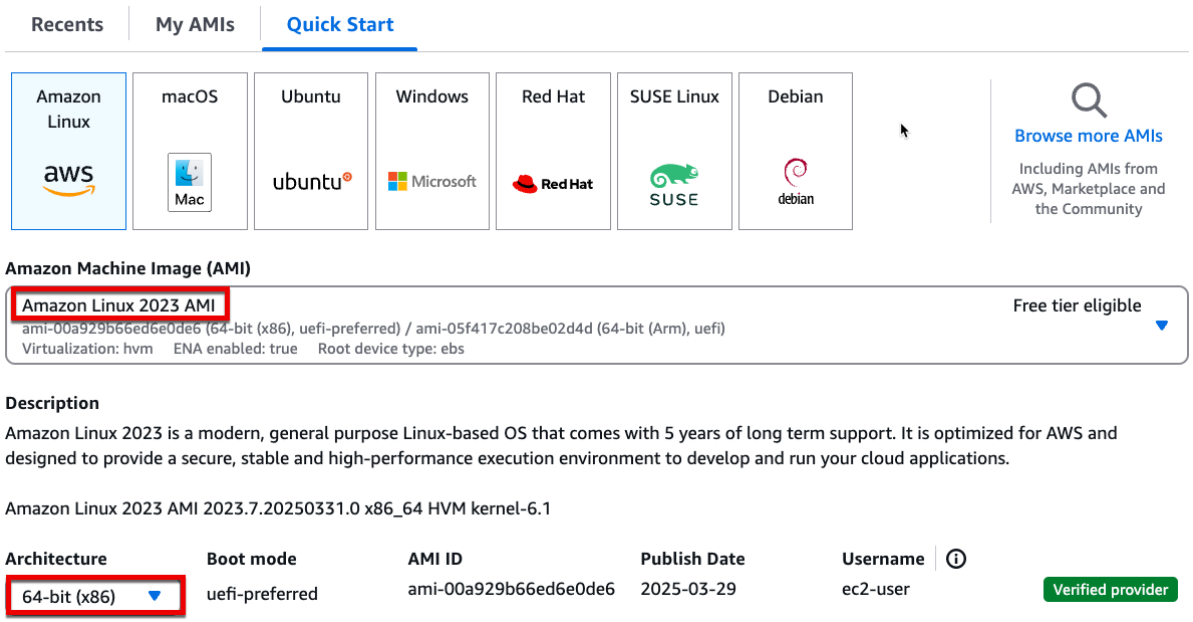

Figure 5.12 – Launching an EC2 Instance

6. For **Instance type,** ensure that **t2.micro** is selected
7. For **Key pair name – required,** select **proceed without a key pair (Not recommended)** from the drop-down menu. You are not configuring a key pair for this instance because you will use Session Manager to connect to it, which does not require a key pair.
8. Under **Network settings,** click the **Edit** button.
9. Select **todolist-vpc** from the **VPC** drop-down list.
10. Under subnet, select one of the public subnets in your VPC, such as **todolist-pub-subnet-01**
11. From the **Auto-assign public IP** drop-down box, select **Enable.**
12. In the **Firewall (security groups)** section, click the **Select existing security group** option.
13. Select the **default** security group from the drop-down list of common security groups, as per the following screenshot in Figure 5.13

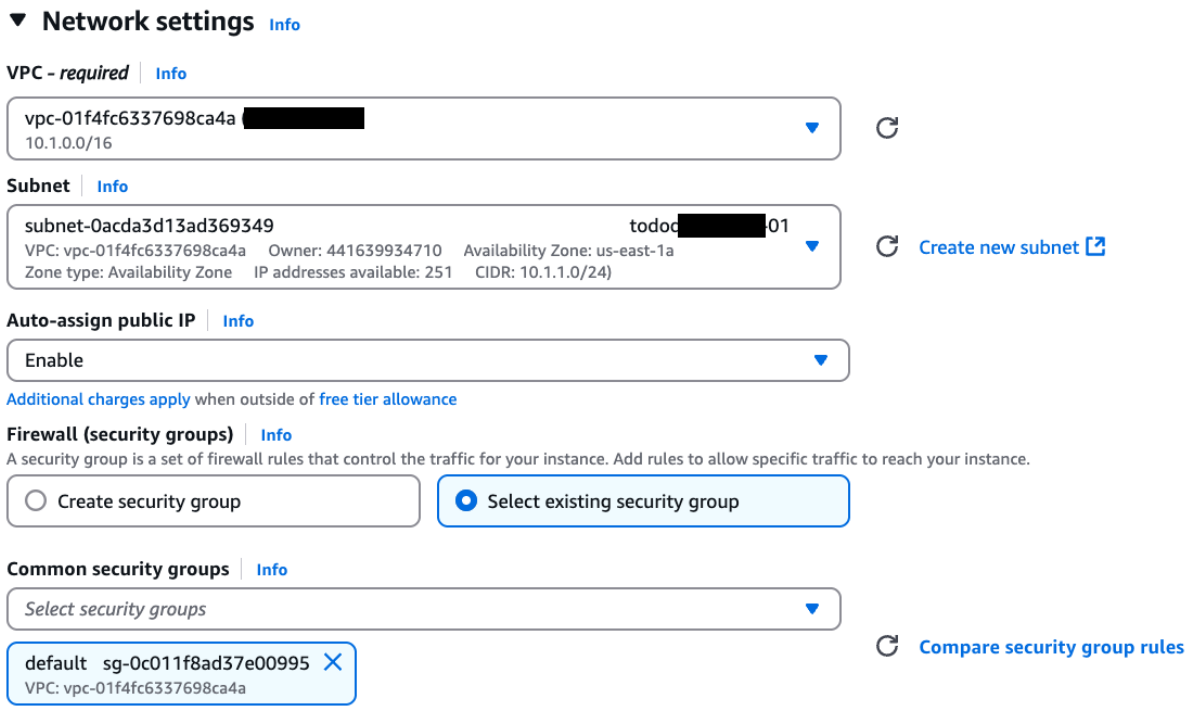

Figure 5.13 – EC2 Instance VPC Configuration

14. Scroll further down and expand the **Advanced detail** section.
15. Under the **IAM instance profile** drop-down list, select the IAM role you created in the previous task.
16. Scroll to the bottom of the page and then click **Launch instance.**
17. You should then get a success message confirming the launch of the instance.
18. From the left-hand menu, click **Instances**
19. This instance will take a few minutes to launch successfully. Make sure the **Status check** section confirms that the two check have passed as per Figure 5.14

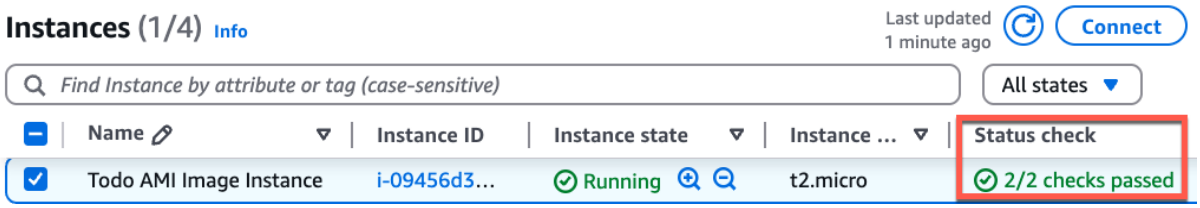

Figure 5.14 – EC2 instance launch status

20. In the right-hand pane, select the instance by clicking the checkbox next to the **Name** instance and click **Connect** button
21. You will have four methods of connecting to the instance. Select **Session Manager** and click the **Connect** button.
22. This will connect you to the EC2 instance shell interface, as per the following screenshot in Figure 5.15

Figure 5.15 – Connected to EC2 instance shell using Session Manager

23. You will be logged in with the SSM-User account when connecting with the Session Manager. Change your user to the root user by typing in the **sudo su** command and press **Enter.**
24. Perform the following steps to install the required software packages:
    a. Type in the command **cd ..** and then press **Enter.** This will take you into the **usr** directory.
    b. Update the operating system by typing the command **yum update -y** and then press **Enter.** If there are no updates to perform, you will be notified
    c. Type in **curl -fsSL https://rpm.nodesource.com/setup_20.x -o nodesource_setup.sh** and press **Enter**. The command downloads a setup script that configures your system to install Node.js 20.x from the Node Source repository on an RPM-based Linux system like CentOS, Fedora, or Red Hat.
    d. Type in the command **bash nodesource_setup.sh** and press **Enter**. This will set up Node.js
    e. Type in the command **yum install -y nodejs** and press **Enter.** This command installs Node.js and its dependencies on your system. The -y flag makes the process non-interactive by automatically agreeing to prompts.
    f. Type in the command **npm i -g pm2** and press **Enter.** This command installs PM2 globally on your system so you can easily manage your Node.js applications across different projects. PM2 is often used in production environments to ensure applications stay up and running. The 'i' is short for installation, and the '-g' tells npm to install the package globally, meaning the package will be available system-wide rather than just in the local project directory. Ignore any notices.
    g. Once the installation is complete, exit the session by typing in the command **exit** and pressing **Enter.** Type in **exit** and press **Enter** again to terminate your session.

Your EC2 instance now has all the software packages you need to build an AMI. You will create an AMI of this EC2 instance in the next project task. This AMI can then be used to deploy your EC2 instances, which will perform all backend operations and host your APIs.