**DevOps Assessment**

MCQ:

1) What does WSL stand for in the context of Windows?

a. Windows Software Locator

b. Windows System Locator

c. Windows Subsystem for Linux

d. Windows Shell Language

Answer : (c) Windows Subsystem for Linux

Reason : WSL stands for Windows Subsystem for Linux. It is a feature of the Windows operating system that allows users to run a Linux environment without the need for a separate virtual machine or dual booting. This makes it possible to use Linux command-line tools and applications directly on Windows, alongside traditional Windows desktop and apps.

2) What is the primary goal of continuous integration (CI) in DevOps?

a. Automating manual testing

b. Frequent integration of code changes

c. Managing cloud infrastructure

d. Monitoring server performance

Answer : (b) Frequent integration of code changes

Reason : CI is a software development practice that automates the building, testing, and deployment of code changes. This helps to ensure that code is always in a deployable state and that any potential problems are caught early.

3) In the Linux command line, what does the cd command do?

a. Copy files and directories

b. Change the working directory

c. Create a new directory

d. Calculate directory size

Answer : (b) Change the working directory

Reason : cd command is used to change the current working directory, which is the directory that the terminal is currently working in.

4) Which of the following is not a Linux distribution?

a. Ubuntu

b. CentOS

c. Docker

d. Debian

Answer : (c) Docker

Reason : Docker is a containerization platform that allows users to package applications and their dependencies into standalone units, called containers. Containers can be run on any platform that supports Docker, making them a popular choice for developing and deploying applications.

5) What is Docker primarily used for in DevOps and containerization?

a. Managing cloud infrastructure

b. Running virtual machines

c. Packaging and deploying applications in containers

d. Managing network security

Answer : (c) Packaging and deploying applications in containers

Reason : Docker is primarily used for packaging and deploying applications in containers. Containers are lightweight, standalone, executable packages of software that include everything needed to run an application: code, runtime, system tools, system libraries, and settings.

6) What is the primary purpose of Azure DevOps?

a. Infrastructure management

b. Software development and delivery

c. Network security

d. Virtualization

Answer : (b) Software development and delivery

Reason : The primary purpose of Azure DevOps is software development and delivery. It is a suite of tools that helps teams to plan, develop, test, deploy, and monitor their software applications.

7) Which components are part of Azure DevOps?

a. Azure App Service and Azure Functions

b. Azure Monitor and Azure Security Center

c. Azure Boards and Azure Pipelines

d. Azure Virtual Machines and Azure SQL Database

Answer : (c) Azure Boards and Azure Pipelines

Reason : The following components are part of Azure DevOps:

* Azure Boards
* Azure Repos
* Azure Pipelines
* Azure Test Plans
* Azure Artifacts

8) How does Azure DevOps support version control in software development?

a. It provides automated database backups.

b. It tracks changes in source code and manages versions.

c. It monitors server performance.

d. It optimizes network configurations.

Answer : (b) It tracks changes in source code and manages versions.

Reason : Azure DevOps supports version control in software development by providing Azure Repos, a cloud-hosted private Git repository service. Azure Repos allows teams to track changes in source code and manage versions. This helps teams to collaborate on projects more effectively, and it also makes it easier to roll back to previous versions of code if necessary.

9) In Linux, what is the primary role of the root user?

a. Managing user accounts

b. Running GUI applications

c. Administrative tasks with superuser privileges

d. Monitoring network traffic

Answer : (c) Administrative tasks with superuser privileges

Reason : The primary role of the root user in Linux is to perform administrative tasks with superuser privileges. This means that the root user has full control over the system and can make any changes they want.

10) In Azure DevOps, which component is used to define, build, test, and deploy applications?

a. Azure Boards

b. Azure Repos

c. Azure Pipelines

d. Azure Artifacts

Answer : (c) Azure Pipelines

Reason : Azure Pipelines is a CI/CD (continuous integration and continuous delivery) platform that automates the build, test, and deployment of code. It allows teams to define a pipeline that specifies the steps that need to be taken to build, test, and deploy their applications.

SECTION 2

Lab1 : File and Directory Management

1. Create a directory called “lab1” in your home directory.

2. Inside “lab1” create a text file named “sample.txt” with some content.

3. Make a copy of “sample.txt” and name it “sample\_copy.txt”

4. Rename “sample\_copy.txt” to “new\_sample.txt”

5. List the files in the “lab1” directory to confirm their names.

Step 1 :

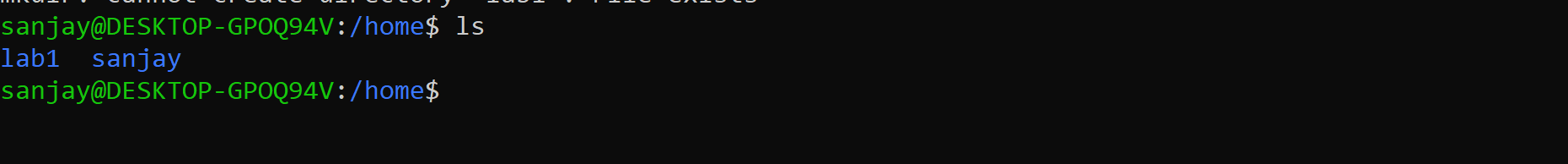
Navigate to your home directory by typing cd /home and pressing Enter.

A screenshot of a computer

Description automatically generated

Step 2 :

Create a directory called lab1 by typing mkdir lab1 and pressing Enter.





Create a text file named sample.txt by typing touch sample.txt and pressing Enter.

A black screen with white text

Description automatically generated

Open sample.txt in a text editor and add some content.



Save and close the text editor.

A black and white rectangular object

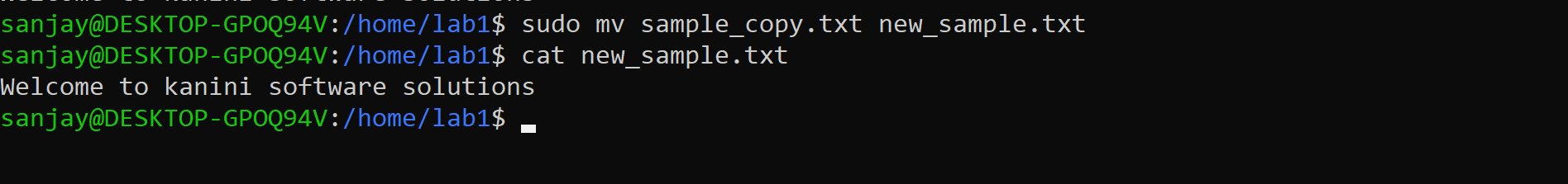
Description automatically generated



Make a copy of sample.txt and name it sample\_copy.txt by typing cp sample.txt sample\_copy.txt and pressing Enter.



Rename sample\_copy.txt to new\_sample.txt by typing mv new\_sample.txt sample\_copy.txt and pressing Enter.



A black screen with a black background

Description automatically generated

View the contents using the cat command

A computer screen with white text

Description automatically generated

Lab 2: Permissions and Ownership

Objective: Understand and manage file permissions and ownership.

Tasks:

1. Create a new file named “secret.txt” in the “lab2” directory.

2. Set the file permissions to allow read and write access only to the

owner.

3. Change the owner of “secret.txt” to another user.

4. Verify the new permissions and owner using the ls -l and ls -n

commands.

Solution :

Set the file permissions to allow read and write access only to the owner.

A black screen with white text

Description automatically generated

Set the file permissions to allow read and write access only to the owner.

A black screen with red and white squares

Description automatically generated

Result for sudo nano secret.txt :

A screenshot of a computer

Description automatically generated

Result for cat secret.txt :



Result for sudo cat secret.txt :



Verify the new permissions and owner using the ls -l and ls -n commands.



Lab 3: Text Processing with Command Line Tools

Objective: Practice text processing using command-line tools.

Tasks:

1. Create a text file with some random text in the “lab3” directory.

2. Use the grep command to search for a specific word or pattern in thefile.

3. Use the sed command to replace a word or phrase with another in the

file.

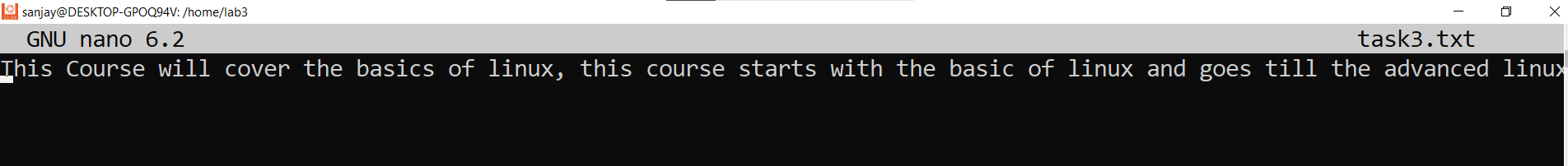
4. Use the wc command to count the number of lines, words, and

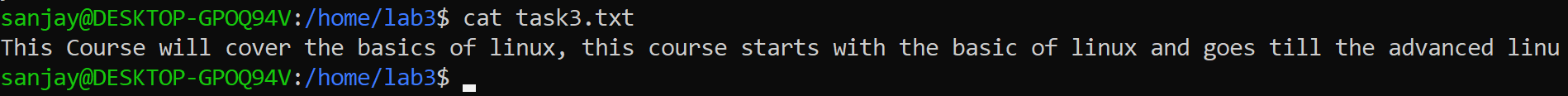
characters in the file.

Step 1 :

Create a text file with some random text in the "lab3" directory.

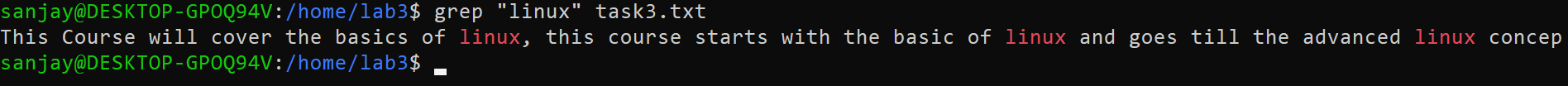






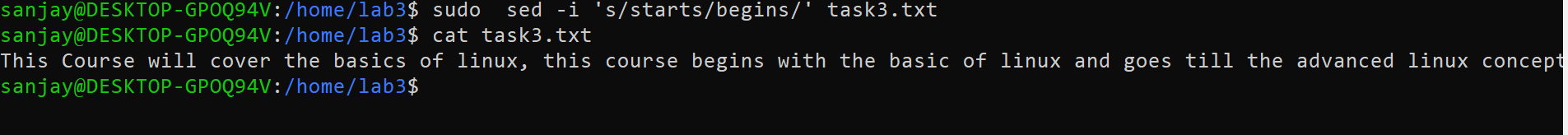
Step 2 :

Use the grep command to search for a specific word or pattern in the file.



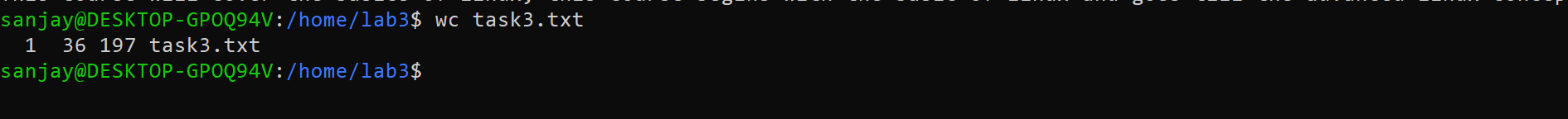
Step 3 :

Use the sed command to replace a word or phrase with another in the file.



Step 4 :

Use the wc command to count the number of lines, words, and characters in the file.



Lab 4: Creating a Simple YAML File

Objective: Create a basic YAML configuration file.

Task:

1. Create a YAML file named “config.yaml”

2. Define key-value pairs in YAML for a fictitious application, including

name, version, and description.

3. Save the file.

4. Validate that the YAML file is correctly formatted.Step 1 :

Create a YAML file named "config.yaml"







Step 2 :

Define key-value pairs in YAML for a fictitious application

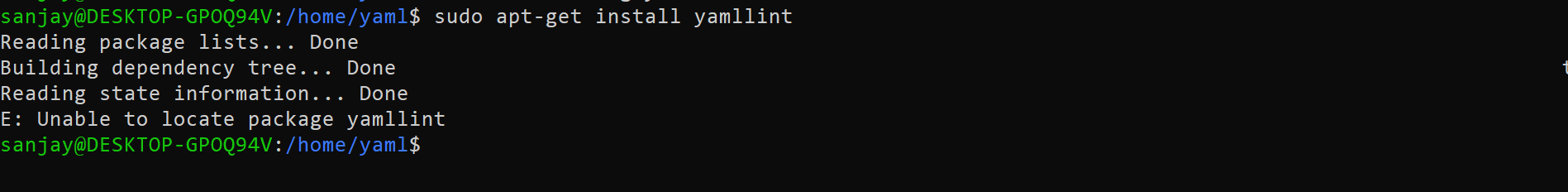
A black and white screen

Description automatically generated

Step 3 :

Validate that the YAML file is correctly formatted

Using install yamllint



A black screen with white text

Description automatically generated

Error encountered

A black screen with a black background

Description automatically generated

Error resolved



Lab 5: Working with Lists in YAML

Objective: Practice working with lists (arrays) in YAML.

Task:

1. Create a YAML file named “fruits.yaml”

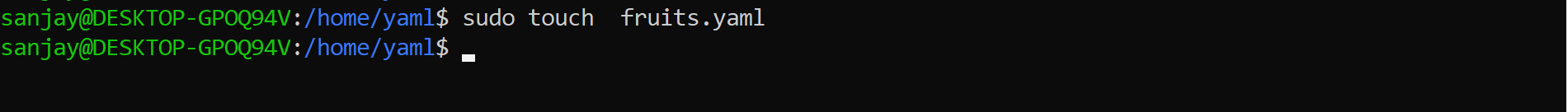
2. Define a list of your favorite fruits using YAML syntax.

3. Add items from the list.

4. Save and validate the YAML file.

Step 1 :

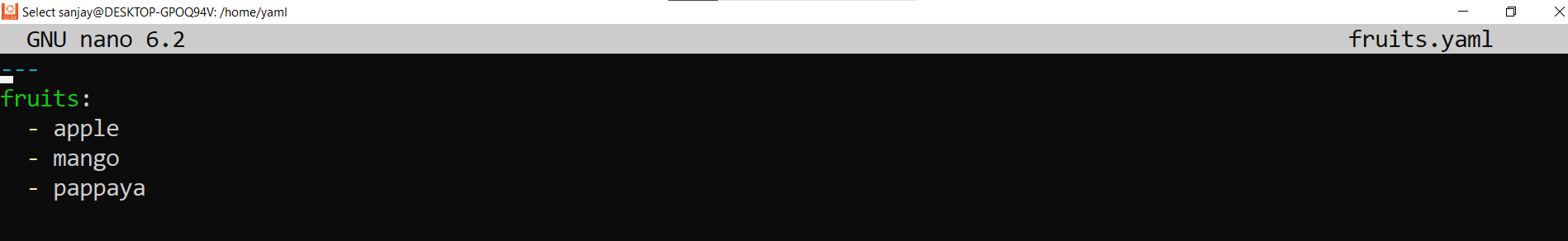
Create a YAML file named "fruits.yaml."

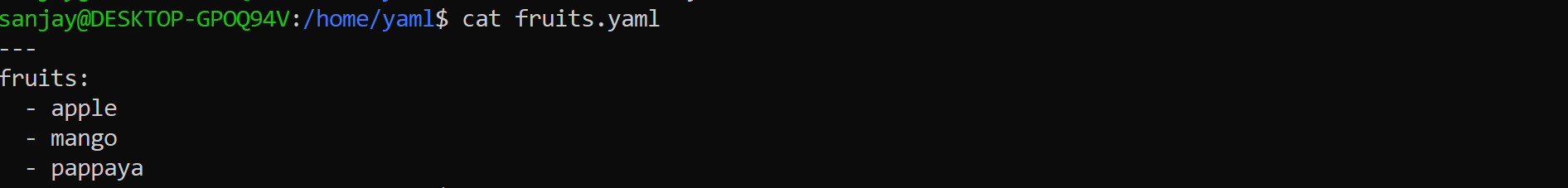




Step 2 :

Define a list of your favorite fruits using YAML syntax.







Step 3 :

Add items to the list.

A black rectangular object with a white border

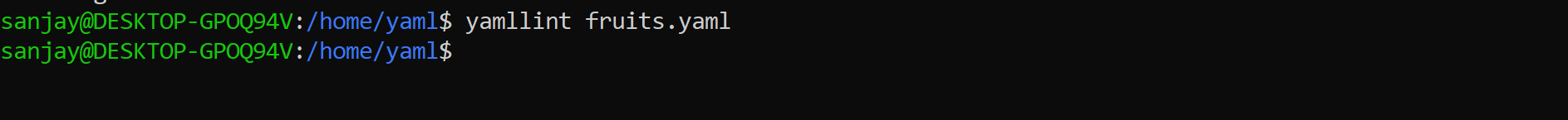
Description automatically generated

A black screen with a black background

Description automatically generated

Step 4 :

Save and validate the YAML file.



Lab 6: Nested Structures in YAML

Objective: Explore nested structures within YAML.

Task:

1. Create a YAML file named “data.yaml”

2. Define a nested structure representing a fictitious organization with

departments and employees.

3. Use YAML syntax to add, update, or remove data within the nested

structure.

4. Save and validate the YAML file.

Step 1 :

Create a YAML file named "data.yaml"

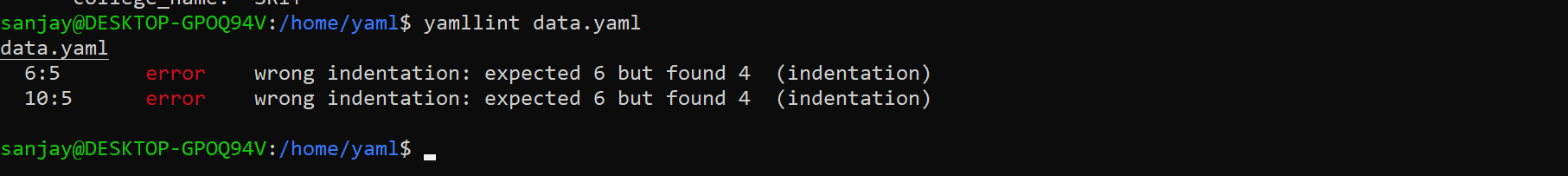


Enter the contents in the yaml format

A black screen with a black background

Description automatically generated

Run the yamllint command to check the error



A screenshot of a computer program

Description automatically generated

Lab 7: Create Classic Azure CI Pipeline for Angular Application

Objective: Set up a classic Azure CI pipeline to build a simple Angular

application with unit testing using Jasmine and Karma.

Tasks:

1. Create an Azure DevOps project.

2. Set up a classic CI pipeline to build an Angular application.

3. Configure the pipeline to use Jasmine and Karma for unit testing.

4. Run the pipeline and validate the test results.

Step 1: Build the Pipeline

A screenshot of a computer

Description automatically generated

Step 2

Running the pipelineA screenshot of a computer

Description automatically generated

Build Successful

A screenshot of a computer

Description automatically generated

Lab 8: Create YAML Azure CI Pipeline for React Application

Objective: Create a YAML-based Azure CI pipeline to build a simple React

application with unit testing using Enzyme and Jest.

Tasks:

1. Create an Azure DevOps project.

2. Create a YAML-based CI pipeline to build a React application.

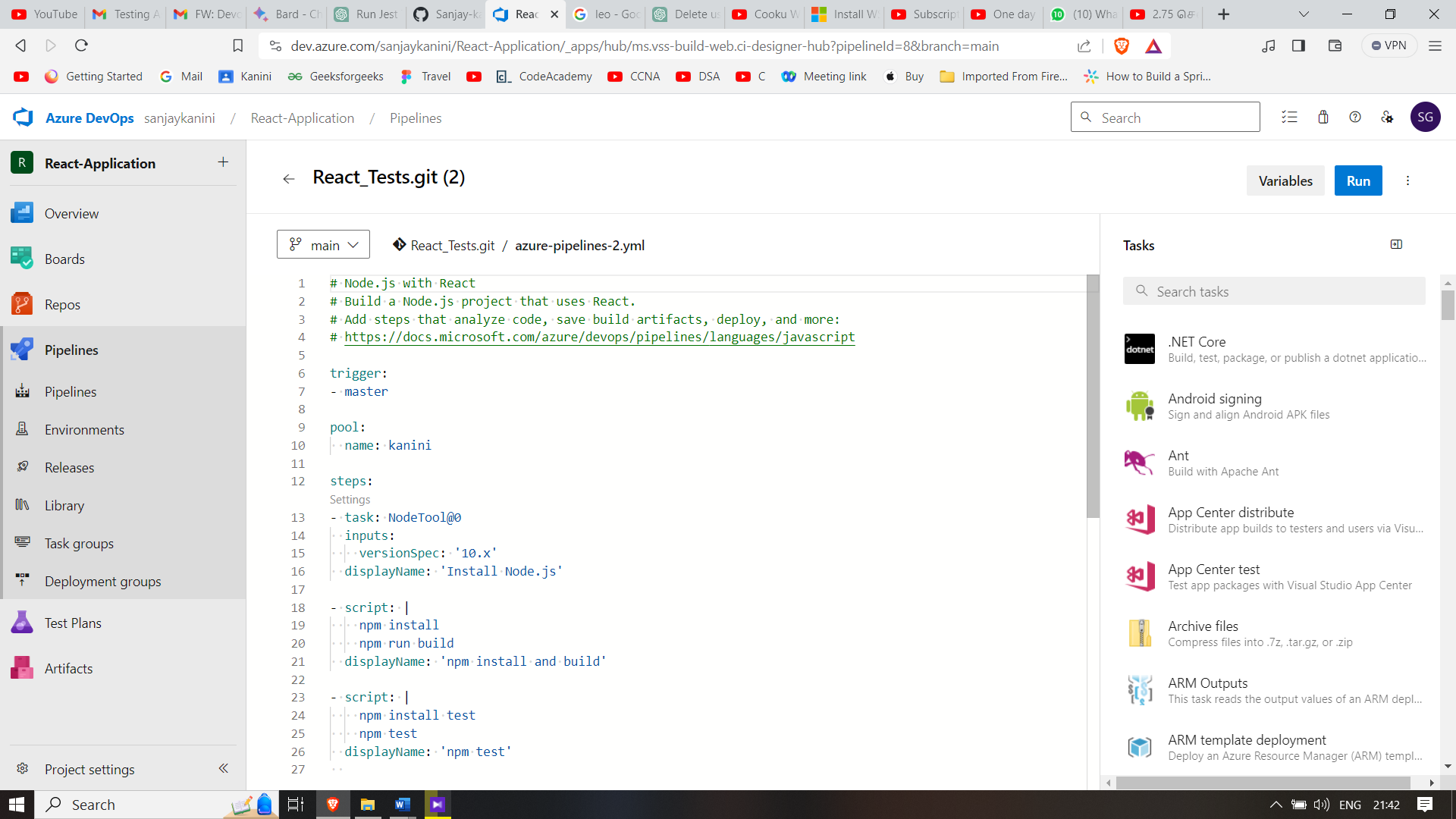
3. Configure the pipeline to use Enzyme and Jest for unit testing.

4. Trigger the pipeline and verify the test results.

Solution :

Step 1:

Using the yaml file to build the pipeline



A screenshot of a computer

Description automatically generated

Pipeline Run

A screenshot of a computer

Description automatically generated

**Lab 9: Create CI Pipeline for .NET Core Application with MS Unit Test**

Objective: Create a CI pipeline, either classic or YAML, to build a .NET Core application and run MS Unit tests.

Tasks:

1.Set up a new Azure DevOps project.

2.Create a CI/CD pipeline for a .NET Core application.

3.Configure the pipeline to use MS Unit tests.

4.Trigger the pipeline and validate the test results.

Step 1: Setup a new pipeline

A screenshot of a computer

Description automatically generated

Step 2 : Initialize the Repository

A screenshot of a computer

Description automatically generated

Step 3: Choose dotnet with core

A screenshot of a computer

Description automatically generated

Step 4: Edit the yaml file and push to the git

A screenshot of a computer

Description automatically generated

Step 5 : Give the commit message and run the pipeline and check the status

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Lab 10: Creating a Docker Image for a .NET Core Web API and Running it in Rancher Desktop**

Objective: In this lab, you will create a Docker image for a sample .NET Core Web

API application and then run the Web API container in Rancher Desktop.

Prerequisites:

Rancher Desktop installed and running.

.NET Core SDK installed on your machine.

Tasks

Step 1: Create a .NET Core Web API Project

Step 2: Build the .NET Core Web API Project

Step 3: Dockerize the .NET Core Web API

Step 4: Build the Docker Image

Step 5: Run the Docker Container in Rancher Desktop

Step 6: Test the .NET Core Web API via swagger

Step1: Create a sample dotnet application and install docker and rancher desktop

A screenshot of a computer screen

Description automatically generated

Create a new docker file inside it

A screenshot of a computer

Description automatically generated

A computer screen with many text

Description automatically generated

Build the docker images in the command prompt

A black screen with white text

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated––