

Full Stack Development

V Semester Lab Manual – 2022-23

Experiment 1: Perform Local Git Operations

Step 1: Install git from the [official website](#).

Step 2: Once installed launch the git bash in Windows.

Step 3: Configure the name and email using following command.

```
git config --global user.name <USERNAME>
git config --global user.email <EMAIL>
```

Step 3: Create a folder with some files and launch a command prompt

Step 4: Navigate to the folder

Use the following commands to perform basic git operations

Command 1# Initialize the repository.

```
git init
```

Command 2# View the status of the repository.

```
git status
```

Command 3# ADD files to the staging area.

```
git add <filename>
```

Command 4# commit changes to the repository.

```
git commit -m "MESSAGE"
```

Command 5# View and create branches.

```
git branch
```

Command 6# View the commit history.

```
git log
```

Command 7# Switch between branches

```
git checkout <branch name>
```

Command 8# Merge two branches

```
git merge <branch name>
```

Command 9# git remote command to manage remote repositories

```
git remote add < GitHub repository link>
```

Command 10# To create a new branch and switch to it

```
git switch -c <New-Branch-Name>
```

Experiment 2: Basics of GitHub operations such as creating an account, create push and pull the repository between GitHub and Git local repository

Step 1: Open [GitHub](#) official website

Step 2: Create an account by clicking on signing up on GitHub.

Step 3: Enter your email and click on continue.

Step 4: Create a password for your account and click continue

Step 5: Enter a username and click on continue

Step 6: Verify your account by solving captcha and click on create account.

Step 7: Verify your email and you can now sign in with your email

Step 8: Now click on profile button go to your repositories

Step 9: Click on New to create a new repository

Step 10: Provide a name for your repository and check the 'Add Readme file'.

Step 11: Click on create repository

Step 12: Go to Desktop and create a folder with some dummy files.

Step 13: Launch command prompt and navigate to the created folder.

Note: You must add your name and email in git bash and the ssh keys to your GitHub account before performing any git operations.

Step 14: Use the following command to initialize repository

```
git init
```

Step 15: Now commit all the files using following command

```
git commit -m "This is first commit"
```

Step 16: Now add the remote link of GitHub to the local git folder to connect the folder with the GitHub repository and push the repository.

Step 17: The files will appear on the GitHub.

Step 18: Click on Add new file > Create a new file > provide a name to your file and write some content.

Step 19: Add a commit message and click on commit

Step 20: To get the files on our local system we just added perform pull operation with following command.

```
git pull
```

Experiment 3: Create a repository name min-project-1. Push the same to GitHub

Step 1: Open browser and login to your GitHub account

Step 2: Go to your repository and click on new to create a new one name `mini-project1` by adding Readme file.

Step 3: Go to Desktop and create a folder with some dummy files.

Step 4: Launch command prompt and navigate to the created folder.

Step 5: Use the following command to initialize repository

```
git init
```

Step 6: Now commit all the files using following command

```
git commit -m "This is first commit"
```

Step 7: Now add the remote link of GitHub to the local git folder to connect the folder with the GitHub repository and push the repository.

Step 8: The files will appear on the GitHub.

Step 9: Now click on profile button go to your repositories

Step 10: Click on New to create a new repository

Step 11: Provide a name for your repository and check the 'Add Readme file'.

Step 12: Click on create repository

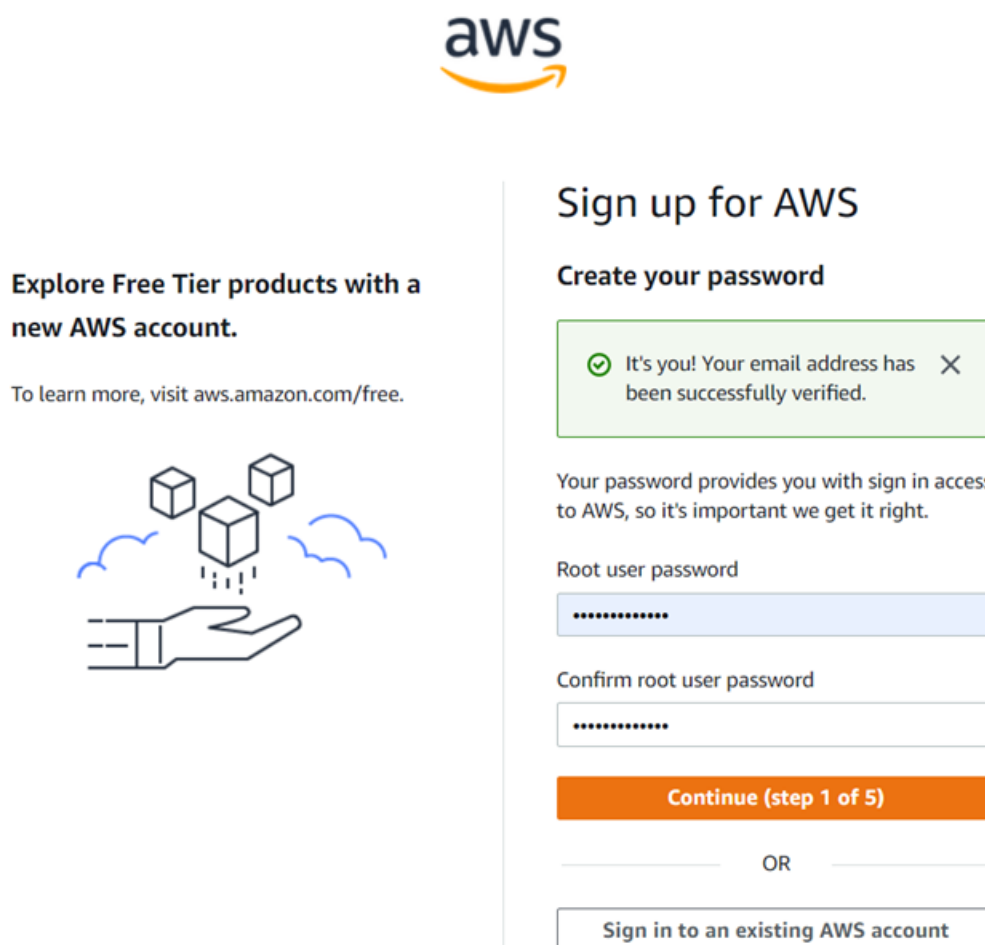
Experiment 4: Create an account on Amazon Web Services (AWS)

Step 1: To create an account on AWS or go to [AWS Console](#) and click on **Sign In** button.

Step 2: After that click on create a new account.

Step 3: Enter your email address and Account name. Verify your email address.

Step 4: After the verification click on next and enter the password for your AWS account and click on continue.



The screenshot shows the AWS sign-up interface. At the top is the AWS logo. Below it, on the left, is a section titled "Explore Free Tier products with a new AWS account." with a link to aws.amazon.com/free and an illustration of a hand holding three cubes. On the right, the main heading is "Sign up for AWS". Below this is the sub-heading "Create your password". A green success message box states: "It's you! Your email address has been successfully verified." with a close button (X). Below the message, it says "Your password provides you with sign in access to AWS, so it's important we get it right." There are two password input fields: "Root user password" and "Confirm root user password", both masked with dots. An orange "Continue (step 1 of 5)" button is below the fields. Below the button is an "OR" separator. At the bottom is a button labeled "Sign in to an existing AWS account".

Step 5: In the next step we have to provide some information about us like user like full name mobile number city etc. For your verification purpose once you are done click on continue.



Free Tier offers

All AWS accounts can explore 3 different types of free offers, depending on the product used.



Always free
Never expires



12 months free
Start from initial sign-up date



Trials
Start from service activation date

Sign up for AWS

Contact Information

How do you plan to use AWS?

- ☐ Business - for your work, school, or organization
- ☐ Personal - for your own projects

Who should we contact about this account?

Full Name

Phone Number

 +1

▼

222-333-4444

Country or Region

United States

▼

Address

City

State, Province, or Region


Postal Code

☐ I have read and agree to the terms of the [AWS Customer Agreement](#).

Continue (step 2 of 5)

Step 6: In the next step we have to provide our card details for the payment and it is mandatory.

Secure verification

-  We will not charge you for usage below AWS Free Tier limits. We may temporarily hold up to \$1 USD (or an equivalent amount in local currency) as a pending transaction for 3-5 days to verify your identity.



Sign up for AWS

Billing Information

Credit or Debit card number



AWS accepts all major credit and debit cards. To learn more about payment options, review our [FAQ](#)

Expiration date

Cardholder's name

CVV

Billing address

☒ Use my contact address

#60

Bangalore Karnataka 560090

IN

☐ Use a new address

Do you have a PAN?

Permanent Account Number (PAN) is a ten-digit alphanumeric number issued by the Indian Income Tax Department. This 10-digit number is printed on the front of your PAN card.

☐ Yes

☐ No

You can go on the [Tax Settings Page](#) on Billing and Cost Management Console to update your PAN information.

Verify and Continue (step 3 of 5)

You might be redirected to your bank's website to authorize the verification charge.

Step 7: After that you have to confirm your identity either via text or voice call.



Sign up for AWS

Confirm your identity

Before you can use your AWS account, you must verify your phone number. When you continue, the AWS automated system will contact you with a verification code.

How should we send you the verification code?

☐ Text message (SMS)

☒ Voice call

Country or region code

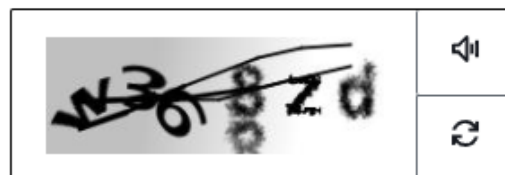
India (+91)

Phone number

7800818620

Ext

Security check



Type the characters as shown above

w368zd


Call me now (step 4 of 5)

Step 8: After confirming your identity, please go with the free tier plan and click on sign up button.



Sign up for AWS

Select a support plan

Choose a support plan for your business or personal account. [Compare plans and pricing examples](#) . You can change your plan anytime in the AWS Management Console.

☒ Basic support - Free

- Recommended for new users just getting started with AWS
- 24x7 self-service access to AWS resources
- For account and billing issues only
- Access to Personal Health Dashboard & Trusted Advisor



☐ Developer support - From \$29/month

- Recommended for developers experimenting with AWS
- Email access to AWS Support during business hours
- 12 (business)-hour response times



☐ Business support - From \$100/month

- Recommended for running production workloads on AWS
- 24x7 tech support via email, phone, and chat
- 1-hour response times
- Full set of Trusted Advisor best-practice recommendations



Experiment 5: Build a basic web application on Amazon Web Services (AWS)

We must have an active AWS account to work on Amazon Web Services and deploy our web application there.

Step 1: Login to your [AWS Console](#) either by using IAM User/Password or Root Email/Password.

Step 2: Go to **Services > Compute > EC2 Virtual Server in the cloud**.

Step 3: Click on Launch Instances.

Step 4: An Instance wizard will appear. Provide the Name of EC2, in OS select **Windows** and for Image choose **Microsoft Windows Server 2022**. Select Instance type as **t2.micro** (free tier eligible). Create a key pair and save it in your local machine.

In Network Setting apart from default settings checkmark the both boxes saying '**Allow HTTP/s traffic from Internet**'. Click on Launch Instance.

Step 5: Click on View All Instances.

Step 6: Wait for few minutes. Select the instance you just created and **click on connect button on top right of window > RDP Client > Download the remote desktop file**.

Click on Get Password and import the PEM file that you downloaded in **step 4 > Decrypt Password**

Step 7: Open the remote desktop file you just downloaded and copy the decrypted Password. Allow the firewall and a login wizard will appear prompting for username and password. Provide the username (**Administrator**) and password (**Copied password**). Click on OK.

Your Created Virtual Machine will boot and will be displayed.

Experiment 6: To create a simple web application on AWS. Open an EC2 instance

Step 1: Click on **Start > Server Manager** and open it.

Step 2: Once Server Manager is opened click on “**Add roles and features**” under configure this local server menu.

Step 3: Now wizard will be opened for installing web server to our EC2 Instance.

Step 4: Click Next to display the Select server roles page

Step 5: Select the Web Server (IIS) option. When a dialog opens, click Add Features

Step 6: Click Next to display the Select features page. Do not change default settings in the Features scroll box.

Step 7: Click Next two times to display the Select role services page. Click on Install and wait for few minutes and click on close.

Then you test the IIS installation by opening your web browser and enter IPv4 public IP of EC2 in your browser and press enter and you will see the default screen for IIS.

Creating webpage and hosting it on EC2 Instance

Step 8: Now after installing IIS, open File Explorer and navigate to *C:/inetpub/wwwroot*.

The wwwroot folder is new in ASP.NET 5.0. All of the static files in your project go into this folder.

Step 9: Create a **index.html** file with some data in HTML tag and save in the wwwroot folder.

Step 10: Then in the new tab and enter ipv4 public IP address of your EC2 machine in your browser and press enter and you will see the HTML page you have created.

Experiment 7: Create build pipeline for simple web application such as to do app, BMI calculator, Number Converter etc.

To Create a JavaScript file on GitHub repository

Step 1: Open or login to your GitHub account

Step 2: Create a repository

Step 3: Click on create a file

Step 4: Write the following code

```
var name = "Vikas";  
console.log(`My name is ${name}`);
```

Step 5: Login to Jenkins and go to Dashboard. Click on create new project.

Step 6: Enter an item name and select pipeline then click OK

Step 7: A configuration page will appear. Click on pipeline.

Step 8: Select Definition as **Pipeline Script**. Generate a Hello World Script and click on pipe

Step 9: Click on pipeline syntax, a new tab will open and select sample step as '**checkout: check out for version control**'. SCM as **Git** Enter the repository URL which you want to build. Enter the credentials of your GitHub. Verify the branch name. and click on **Generate pipeline script**. Paste the generated in line 7 of your pipeline script (in the following code).

Do the same for sample step as '**git: Git**' and paste the generated pipeline syntax in line 12 in the following code.

```
pipeline {  
    agent any  
  
    stages {  
        stage('checkout') {  
            steps {  
                <Generated pipeline script for git: checkout>  
            }  
        }  
    }  
}
```

```
}
stage('build') {
    steps {
        <Generated pipeline script for git: Git>
        Bat label: '',script: 'node <file_name>.js'
    }
}
stage('test') {
    steps {
        echo 'Test was successful'
    }
}
}
```

Step 10: Click save. Click on **Build Now**

Step 11: Click on the latest build in Build History and Click on Console Output.

Experiment 8: Create a HTML Page using CSS

Step 1: Open VS Code and open a folder in VS Code.

Step 2: Create an **index.html** file. And write the following code.

```
//index.html

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Index</title>
  <style>
    @import
url('https://fonts.googleapis.com/css2?family=Raleway&display=swap');

    * {
      font-family: 'Raleway', sans-serif;
      margin: 0;
      padding: 0;
      box-sizing: border-box;
      text-decoration: none;
    }

    nav {
      border: 2px solid black;
      padding: 1.2rem;
      text-align: center;
      background-color: black;
    }

    form {
      margin-top: 1rem;
    }

    ul, li {
```

```
        display: inline;
        margin: 0 0.5rem;
    }

    a:hover {
        color: yellowgreen;
    }

    .content{
        text-align: center;
        margin-bottom: 2rem;
    }

    h1 {
        padding: 1rem 0;
    }

    .footer{
        padding: 0 5rem;
    }

    .footer > p{
        padding: 0.3rem 0;
    }
</style>
</head>

<body>
    <nav>
        <div>
```



```
<ul>
  <a href="#">
    <li>Home</li>|
  </a>
  <a href="#">
    <li>About</li>|
  </a>
  <a href="#">
    <li>Blog</li>|
  </a>
  <a href="#">
    <li>Contact</li>
  </a>
</ul>
</div>

</nav>
<div class="content">
  <h1>Vikas Singh</h1>
  <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit.
Provident optio pariatur, at quasi, corrupti vel
    esse in aliquam exercitationem aut expedita aspernatur
qui? Iste obcaecati modi, incidunt consectetur
    architecto maxime.
  </p>
  <form action="">
    <label for="">Email: </label><input type="text">
    <button>Subscribe</button>
  </form>
</div>
<div class="footer">
  <hr>
```

```
<p>This page is created by Xander Billa</p>
</div>
</body>

</html>
```

Step 3: Click on Go Live present at the task bar of the VS Code in corner.

Output



Experiment 9: Setting up an Environment and tools for frontend development such as installing VS Code and installing required extension

Step 1: Open browser and go to visual studio code official website and download the windows exe file.

Step 2: Open The downloads folder and execute the downloaded file.

Step 3: Accept the license agreement and click on Next.

Step 4: Select the installation directory and click on Next.

Step 5: Keep clicking on Next keeping the setting default. And finally click on Next.

Step 6: Once the installation finish launch VS Code and go extensions to download the following extension –

ESLint

Experiment 10: JavaScript programs

JavaScript functions

```
function add(a, b){  
    return a + b;  
}  
  
//function to print hello world  
function message(){  
    console.log("Hello World");  
}
```

Arrow Functions

```
const add = (a, b) => {  
    return a + b;  
}  
  
const add = (a, b) => a + b;  
const message = () => "Hello World";
```

Prompt and Alert

```
<script>  
// prompt example  
let age = prompt('How old are you?', 50);  
alert(`You are ${age} years old!`);
```

Prototype in JavaScript

```
function Student() {  
    this.name = "John";  
    this.gender = "M";  
}  
  
Student.prototype.age = 15;  
  
var studObj1 = new Student;  
alert(studObj1.age);  
var studObj2 = new Student;  
alert(studObj2.age);
```

Experiment 11: Create a form like registration/login form after submit hide create form and enable the display

```
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
</head>
<body>
  <h1>Form!</h1>
  <form>
    <p>
      <label for="email">Email: </label>
      <input type="email">
    </p>
    <p>
      <label for="password">Password: </label>
      <input type="password">
    </p>
    <p>
      <input type="submit" id="submit" value="Login">
    </p>
  </form>
  <script>
    const signUp = document.getElementById('submit');
    signUp.addEventListener('click', () => {
      const formValue = document.querySelector('form');
      formValue.style.display = 'none';
      document.querySelector('h1').innerText = "Success"
    });
  </script>
</body>
</html>
```

Output

Form!

Email:

Password:

Login 

Experiment 12: Setting up development environment for Typescript by installing the typescript compiler and live server.

Step 1: Launch command prompt

Step 2: To install TypeScript is through npm, the Node.js Package Manager. If you have npm installed, you can install TypeScript globally (-g) on your computer by:

```
npm install -g typescript
```

Step 3: To test your install by checking the version.

```
tsc --version
```

Step 5: Click on extensions and search for Live Server.

Step 6: Click on Install to install the extension

Experiment 13: Create and run first program in Typescript

Step 1: Create a new folder HelloWorld and launch VS Code.

Step 2: Add the following TypeScript code.

```
let message: string = 'Hello World';  
console.log(message);
```

Step 3: To compile your TypeScript code, you can open the Integrated Terminal (Ctrl+`) and type `tsc helloworld.ts`

If you have Node.js installed, you can run `node helloworld.js`.

Experiment 14: Setting up a React development environment – installing Node.js, create and run a ReactJS app.

Step 1: Open browser and go to nodejs.org

Step 2: Go to downloads and download the MSI installer for windows of latest version

Step 3: Open the download folder and run the MSI file.

Step 4: Accept the license terms and agreement, click Next.

Step 5: Select the installation directory, click on Install.

Step 6: Once installed verify the installation of Node JS on command prompt with following command.

```
node -version  
npm --version
```

Step 7: To create a react application hit the following command.

```
cd Desktop  
npm create react-app my-app
```

Step 8: To run the react application run the following command.

```
cd my-app  
npm start
```

Experiment 15: Using state (useState) in React JS

```
//App.js
import React, { useState } from "react";

App(){
  const [name, setName] = useState("Vikas");

  return (
    <div>
      <h2>My Name is {name}</h2>
      Click here to <button onClick={() =>
setName("Xander")}>Update Name</button>
    </div>
  )
}
```

Experiment 16: Using Component, props, fragment, List and keys in React

```
//App.js
App(){
  const Student = [
    {
      id: 1,
      name: "Vikas Singh",
      age: 24
    }
  ]

  return (
    <Student stName={Student.name} stAge={Student.age}>
  )
}

//Student.js
import React, { Fragment } from 'react';

App(props){
  return (
    <Fragment>
      <h1>Student Info</h1>
      <h4>Name: {props.stName} | Age: {stAge}</h4>
    </Fragment>
  )
}
```

Experiment 17: Using List and keys in React

```
//App.js
App(){
  const Student = [
    {
      id: 1,
      name: "Vikas Singh",
      age: 24
    },
    {
      id: 2,
      name: "Anchal Rai",
      age: 23
    }
  ]
  return (
    data.map(i, item =>
      <Student key={i} stName={item.name} stAge={items.age}>
    )
  )
}

//Student.js
import React, { Fragment } from 'react';
App(props){
  return (
    <Fragment>
      <h1>Student Info</h1>
      <h4>Name: {props.stName} | Age: {stAge}</h4>
    </Fragment>
  )
}
```

Experiment 18: Setting up environment for creating JAVA Application

- Install JAVA, JAVA Editor (such as IntelliJ, Eclipse etc.), Install DBMS (MySQL, PostgreSQL or any other)

Install JDK JAVA

Step 1: Go to Java official website and go to downloads search for Java 17

Step 2: Download the MSI installer.

Step 3: Open the downloads folder and execute the downloaded JDK executable file.

Step 4: Select the installation directory and click on Install, once the JDK install click on close.

Download MySQL

The simplest and recommended method is to download MySQL Installer for Windows from <https://dev.mysql.com/downloads/installer/> and execute it.

General Availability (GA) Releases Archives ⓘ

MySQL Installer 8.0.23

Select Operating System:
Microsoft Windows ▼

[Looking for previous GA versions?](#)

Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.23.0.msi)	8.0.23	2.4M	Download
Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.23.0.msi)	8.0.23	422.4M	Download

MD5: a3af6d91f93e046452b38a1e2589534c | [Signature](#)

MD5: 8de85ced955631901829a1a363cdbf50 | [Signature](#)

! We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

Select mysql-installer-web-community-8.0.23.msi if you have good internet connection, otherwise choose mysql-installer-community-8.0.23.msi.

Install MySQL

After downloading, unzip it, and double click the MSI installer .exe file.

Then follow the steps below:

Step 1: "Choosing a Setup Type" screen: Choose "Full" setup type. This installs all MySQL products and features. Then click the "Next" button to continue.

Step 2: "Check Requirements" screen: The installer checks if your pc has the requirements needed. If there is some failing requirements, click on each item to try to resolve them by clicking on the Execute button that will install all requirements automatically. Click "Next".

Step 3: "Installation" screen: See what products that will be installed. Click "Execute" to download and install the Products. After finishing the installation, click "Next".

Step 4: "Product Configuration" screen: See what products that will be configured. Click the "MySQL Server 8.0.23" option to configure the MySQL Server. Click the "Next" button. Choose the "Standalone MySQL Server/Classic MySQL Replication" option and click on the "Next" button. In page "Type and Networking" set Config Type to "Development Computer" and "Connectivity" to "TCP/IP" and "Port" to "3006". Then, click the "Next" button.

Step 5: "Authentication Method" screen: Choose "Use Strong Password Encryption for Authentication". Click "Next".

Step 6: "Accounts and Roles" screen: Set a password for the root account. Click "Next".

Step 7: "Windows Service" screen: Here, you configure the Windows Service to start the server. Keep the default setup, then click "Next".

Step 8: "Apply Configuration" screen: Click the "Execute" button to apply the Server configuration. After finishing, click the "Finish" button.

Step 9: "Product Configuration" screen: See that the Product Configuration is completed. Keep the default setting and click on the "Next" and "Finish" button to complete the MySQL package installation.

Step 10: In the next screen, you can choose to configure the Router. Click on "Next", "Finish" and then click the "Next" button.

Step 11: "Connect To Server" screen: Type in the root password (from step 6). Click the "Check" button to check if the connection is successful or not. Click on the "Next" button.

Step 12: "Apply Configuration" screen: Select the options and click the "Execute" button. After finishing, click the "Finish" button.

Step 13: "Installation Complete" screen: The installation is complete. Click the "Finish" button.

Install Eclipse

Step 1: Go to eclipse.org and download the exe file for windows

Step 2: Open the downloads folder and execute the downloaded eclipse executable file.

Step 3: Select the installation directory and click on Install, once the eclipse install click on Launch.

Experiment 19: Create Spring Application using Spring Web and Spring JPA

To Create a project using Spring Initializer

Step 1: Open web browser and go to start.spring.io .

Step 2: Create project with following information:

- **Project** - Maven
- **Language** - JAVA
- **Spring Boot Version** - 3.0.1
- **Project Metadata** -
 - **Group:** com.acharya
 - **Artifact:** studentdata
 - **Name:** studentdata
 - **Package Name:** com.acharya.studentdata
 - **Packaging:** Jar
 - **Java Version:** 17
- **Dependencies** -
 - MySQL Driver
 - Spring JPA

Step 3: Click on Generate. And Extract the downloaded file where you have saved.

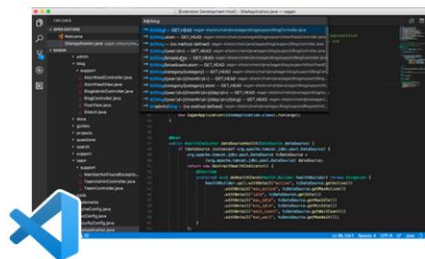
Install Spring Boot extension pack in Visual Studio

Step 1: Go to browser and navigate to <https://spring.io/tools>

Spring Tools 4 for Visual Studio Code

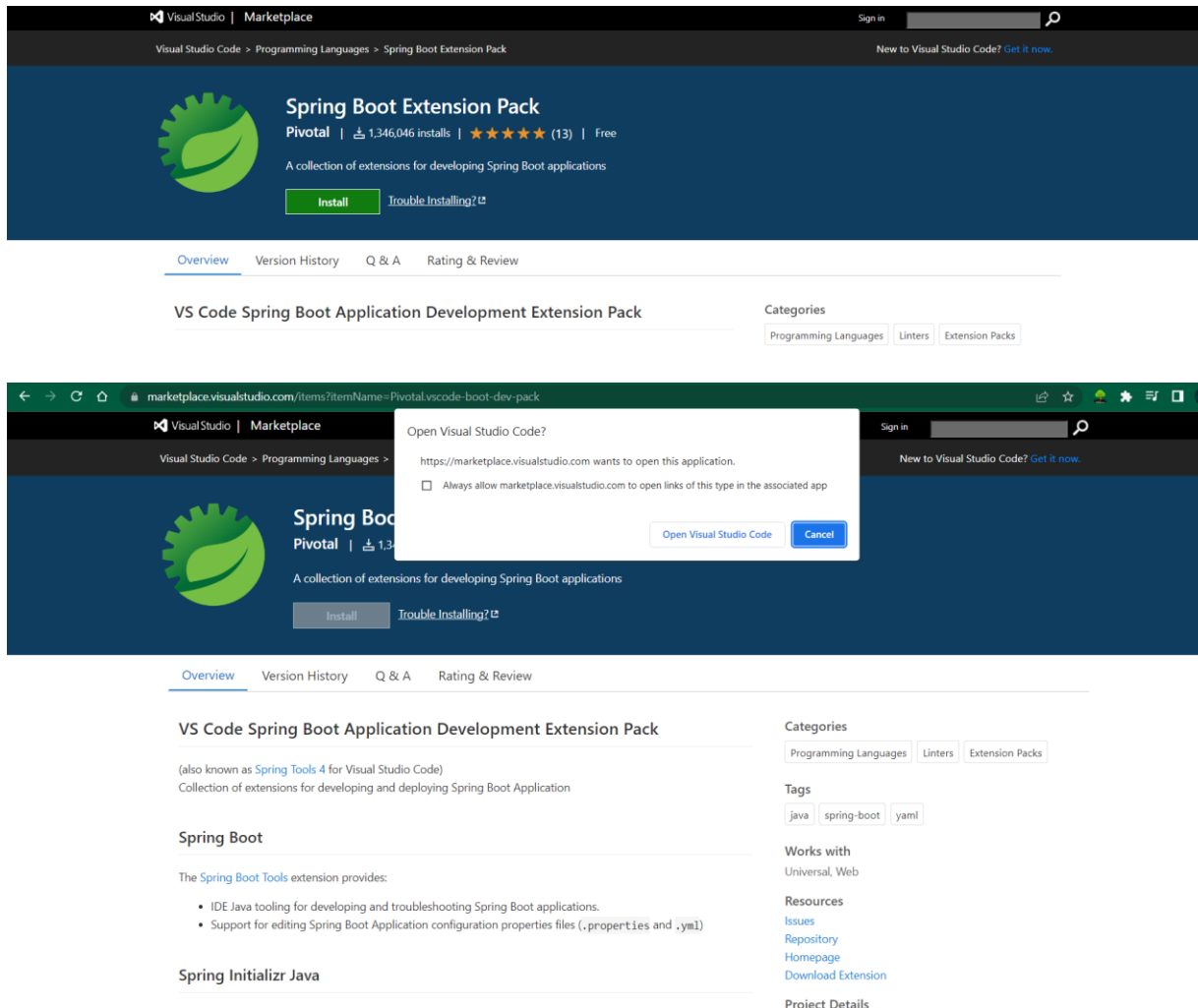
Free. Open source.

SPRING TOOLS 4
VSCode Marketplace

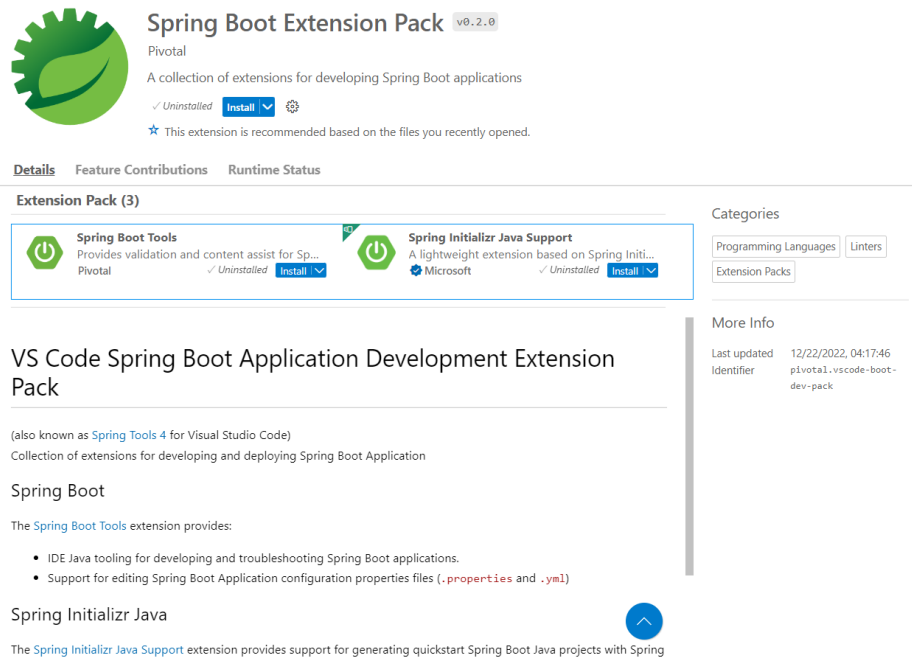


Step 2: Click on Spring Tools 4 for Visual Studio Code Marketplace

Step 3: Click on install > an alert message will appear > click on open with Visual studio Code



Step 4: Click on install the extension



Spring Boot Extension Pack v0.2.0
Pivotal
A collection of extensions for developing Spring Boot applications
✓ Uninstalled [Install](#) ⚙️
★ This extension is recommended based on the files you recently opened.

Details Feature Contributions Runtime Status

Extension Pack (3)

- Spring Boot Tools**
Provides validation and content assist for Sp...
Pivotal ✓ Uninstalled [Install](#)
- Spring Initializr Java Support**
A lightweight extension based on Spring Initi...
Microsoft ✓ Uninstalled [Install](#)

Categories
Programming Languages Linters
Extension Packs

More Info
Last updated 12/22/2022, 04:17:46
Identifier pivotal.vscode-boot-dev-pack

VS Code Spring Boot Application Development Extension Pack
(also known as [Spring Tools 4](#) for Visual Studio Code)
Collection of extensions for developing and deploying Spring Boot Application

Spring Boot
The [Spring Boot Tools](#) extension provides:

- IDE Java tooling for developing and troubleshooting Spring Boot applications.
- Support for editing Spring Boot Application configuration properties files (`.properties` and `.yml`)

Spring Initializr Java
The [Spring Initializr Java Support](#) extension provides support for generating quickstart Spring Boot Java projects with Spring

To Run a project in Visual Studio

Step 1: Launch visual Studio and MySQL Workbench

Step 2: In MySQL Workbench create a connection.

Step 3: Go to Schema > Right Click on the side menu of Schema > Create schema with name acharya

Step 4: Create a Student table with fields (id, name and age) inside that schema we just created.

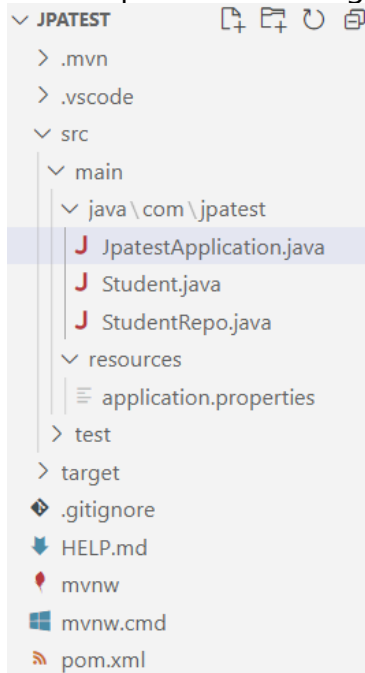
Step 5: In Visual Studio go to **File > Open Folder**

Step 6: Browse the extracted project we downloaded and click on finish.

Step 7: In the project navigate to `src/main/java` directory a default package (`com/acharya`) is already created with a main java Application file (`StudentdataApplication.java`).

Step 8: Create one class file named 'Student.java' (must be same as Table name) and an interface named 'StudentRepo.java'

After import and creating the above two files the project structure would be like -



Step 9: Go to Student.java and write the following code -

```
//Student.java
package com.acharya;

import jakarta.persistence.Entity;
import jakarta.persistence.Id;

@Entity
public class Student {

    public Student(int id, String name, String age) {
        super();
        this.id = id;
        this.name = name;
        this.age = age;
    }

    public Student() {
        super();
        // TODO Auto-generated constructor stub
    }

    @Id
    private int id;
```

```
private String name;
private String age;

public int getId() {
    return id;
}
public void setId(int id) {
    this.id = id;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getAge() {
    return age;
}
public void setAge(String age) {
    this.age = age;
}

@Override
public String toString() {
    return "Student [id=" + id + ", name=" + name + ", age=" + age
+ " ]";
}
}
```

Step 10: Go to 'StudentRepo.java' and enter the following code -

```
//StudentRepo.java
package com.acharya;

import org.springframework.data.repository.CrudRepository;

public interface StudentRepo extends CrudRepository<Student, Integer>{

}
```

Step 11: Go to main Application java file that contain method and modify it with following code -

```
//StudentdataApplication.java
package com.acharya;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;

@SpringBootApplication
public class StudentdataApplication {

    public static void main(String[] args) {
        ApplicationContext context =
SpringApplication.run(StudentdataApplication.class, args);
        StudentRepo studentRepo = context.getBean(StudentRepo.class);

        Student st = new Student();
        st.setId(102);
        st.setName("Anchal Rai");
        st.setAge("24");

        Student stDisplay = studentRepo.save(st);
        System.out.println(stDisplay);
    }
}
```

Step 12: Go to main Application java file that contain method and modify it with following code -

```
//Application.properties
spring.datasource.name=acharya
spring.datasource.url=jdbc:mysql://localhost/acharya
spring.datasource.username=root
spring.datasource.password=password
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
```

Step 13: To run the Application. Go to **Spring Dashboard**(present at left side of VS Code) in Apps > **Run**

Step 14: Go to MySQL workbench and execute the following query.

```
SELECT * FROM table_name;
```

Experiment 20: Practice - Constructor Injection, Property Injection

Create a Maven Project

Step 1: Open Spring Tool Suite/Eclipse

Step 2: Go to **New > Other**

Step 3: Search for **Maven > Maven Project**

Step 4: Click Next and choose the workspace or click Next to keep default

Step 5: Select '**Internal**' in Catalog and select Archetype that have Artifact ID '**maven-archetype-quickstart**' and Click on Next

Step 6: Provide a Group ID Package Name and Artifact ID and click finish

Step 7: Once the project is created expand the project and open pom.xml file to add couple of dependencies required for this experiment

Step 8: Search on Google for **Spring Core Maven Dependency** and **Spring context Maven Dependency**

Step 8.1 : Open the first link and click on any version

Step 8.2 : Copy the maven dependency

step 8.3 : Paste in your pom.xml file under `<dependencies></dependencies>`

Or Paste the following dependency in your **pom.xml** file in `<dependencies></dependencies>`

```
<!-- https://mvnrepository.com/artifact/org.springframework/spring-  
core -->  
<dependency>  
    <groupId>org.springframework</groupId>  
    <artifactId>spring-core</artifactId>  
    <version>5.3.20</version>  
</dependency>  
<!-- https://mvnrepository.com/artifact/org.springframework/spring-  
context -->
```

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>5.3.20</version>
</dependency>
```

Step 9: Go to */src/main* folder and navigate to the current package.

For Property Injection

Step 10: Create a class files named Person.java

```
//Person.java

Package com.personinfo;

public class Person{
    private int id;
    private String name;
    private String age;

    public Student(int id, String name, String age) {
        super();
        this.id = id;
        this.name = name;
        this.age = age;
    }

    public Student() {
        super();
        // TODO Auto-generated constructor stub
    }

    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```



```

    }
    public String getAge() {
        return age;
    }
    public void setAge(String age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "Student [id=" + id + ", name=" + name + ", age=" + age
+ " ]";
    }
}

```

Step 11: Create an XML file outside of the package with following code:

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:p="http://www.springframework.org/schema/p"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/beans/spring-context.xsd">

    <bean class="com.springcore.Student" name="student1">
        <property name="id" value="101" />
        <property name="name" value="Vikas Singh" />
        <property name="age" value="24" /> </bean>

</beans>

```

Step 11: Go to App.java file and modify the file with following code –

```

//App.java
package com.personinfo;

import org.springframework.context.ApplicationContext;
import
org.springframework.context.support.ClassPathXmlApplicationContext;

public class App
{

```

```
public static void main( String[] args )
{
    System.out.println( "Hello World!" );
    ApplicationContext context = new
ClassPathXmlApplicationContext("config.xml");
    Student student = (Student) context.getBean("student1");
    System.out.println(student);
}
}
```

Step 12: Run the App.java file

For Constructor Injection

Step 13: Create a class files named Person.java

```
Package com.personinfo;

public class Person{
    private int id;
    private String name;
    private String age;

    public Student(int id, String name, String age) {
        super();
        this.id = id;
        this.name = name;
        this.age = age;
    }

    public Student() {
        super();
        // TODO Auto-generated constructor stub
    }

    @Override
    public String toString() {
        return "Student [id=" + id + ", name=" + name + ", age=" + age
+ " ]";
    }
}
```

Step 11: Create an XML file outside of the package with following code:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:p="http://www.springframework.org/schema/p"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/beans/spring-context.xsd">

    <bean class="com.springcore.Student" name="student1">
        <constructor-arg value="101" />
        <constructor-arg value="Vikas Singh" />
        <constructor-arg value="24" />
    </bean>
</beans>
```

Step 11: Go to App.java file and modify the file with following code –

```
//App.java
package com.personinfo;

import org.springframework.context.ApplicationContext;
import
org.springframework.context.support.ClassPathXmlApplicationContext;

public class App
{
    public static void main( String[] args )
    {
        System.out.println( "Hello World!" );
        ApplicationContext context = new
ClassPathXmlApplicationContext("config.xml");
        Student student = (Student) context.getBean("student1");
        System.out.println(student);
    }
}
```

Step 12: Run the App.java file

Experiment 21: Install MongoDB and perform CRUD operation on Database and documents

Step 1: Go to MongoDB official website

Step 2: Click on Products > MongoDB Community Edition

Step 3: Download the MongoDB Community Server for your OS

Step 4: To download the MongoDB shell click on Products > Tools > shell

Step 5: Download the zip file and extract it

Step 6: Now execute the executable file of MongoDB server after the download is finished

Step 7: Accept the license terms and agreement and click on next.

Step 8: Select complete and click Next every time keeping everything as it is. Click on Install.

Step 9: Once the installation is complete click on finish.

Step 10: Now Copy the extracted file to C:/ drive and create subfolder with name *data/db* in the same drive.

Step 11: Now copy the bin folder path of both shell and server and add those paths to environment variables.

Step 12: Launch the command prompt and start the server by executing the command '*mongod*'

Step 13: Open another command prompt and launch the MongoDB shell using *mongo* command

Step 14: CRUD Operation in MongoDB (Let's take an example of Student)

Step 15: Create Database

```
use Student
```

Step 16: Create Collection

```
db.createCollection('studentInfo')
```

Step 17: Show Collection

```
show collections
```

Step 18: Show Database

```
show dbs
```

Step 19: Insert a single Documents

```
db.studentInfo.insertOne({id: 101, name: "Vikas Singh", age: 24, address: "Varanasi"})
```

Step 20: Insert multiple documents

```
db.studentInfo.insertOne([{id: 101, name: "Vikas Singh", age: 24, address: "Varanasi"}, {id: 102, name: "Anchal Rai", age: 22, address: "Varanasi"}])
```

Step 21: Show documents in a collection

```
db.studentInfo.find()
```

Step 22: Show documents in a collection using filter

```
db.studentInfo.find({id: 102})
```

Step 23: Update one document

```
db.studentInfo.updateOne({}, {address: "Bangalore"})
```

Step 25: Delete single document

```
db.studentInfo.deleteOne({address: "Bangalore"})
```

Step 26: Delete multiple documents

```
db.studentInfo.deleteMany({})
```

Experiment 22: Perform CRUD operation on MongoDB through REST API using Spring Data MongoDB

Create a project using Spring Initializer

Step 1: Open web browser and go to start.spring.io .

Step 2: Create project with following information:

- **Project** - Maven
- **Language** - JAVA
- **Spring Boot Version** - 3.0.1
- **Project Metadata** -
 - **Group:** com.acharya
 - **Artifact:** studentdata
 - **Name:** studentdata
 - **Package Name:** com.acharya.studentdata
 - **Packaging:** Jar
 - **Java Version:** 17
- **Dependencies** -
 - Spring Web
 - Spring Data MongoDB
 - Lombok
 - Spring Boot DevTools

Step 3: Click on Generate. And Extract the downloaded file where you have saved.

To Run a project in VS Code

Step 1: Launch VS Code

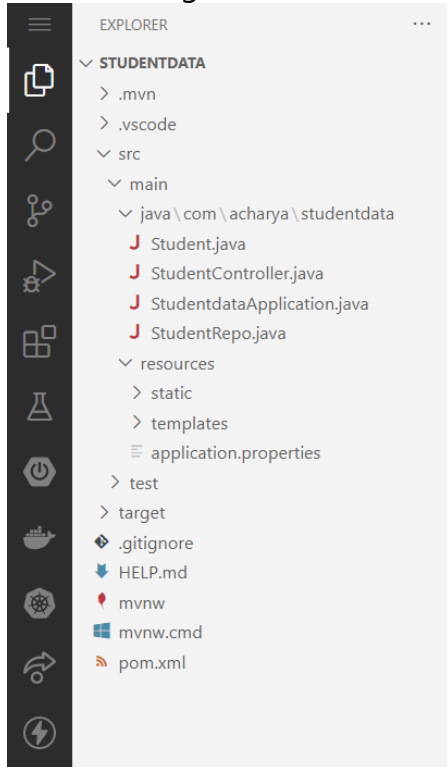
Step 2: Go to File > Open Folder and navigate to the extracted project and click on Open.

VS Code will start scanning for the project once it detects **Spring Boot Dashboard** will appear. Dashboard will look for Spring projects once it find our project, we have open it will be listed in dashboard

Step 3: Go to Explorer > src > main > package (com\acharya\studentdata) inside that you will find a default Application file with main method

Step 4: Create two class files named 'Student.java' and 'StudentController.java' and an interface named 'StudentRepo.java'

After creating the above three files the project structure would be like -



Step 5: Go to Student.java and write the following code -

```
//Student.java
package com.acharya.studentdata;

import org.springframework.data.annotation.Id;
import org.springframework.data.mongodb.core.mapping.Document;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@NoArgsConstructor
@AllArgsConstructor
@Document(collection="studentData")
public class Student {
    @Id
```



```
        private int id;
        private String name;
        private String address;
    }
```

Step 6: Go to 'StudentRepo.java' and enter the following code -

```
//StudentRepo.java
package com.acharya.studentdata;

import org.springframework.data.mongodb.repository.MongoRepository;

public interface StudentRepo extends MongoRepository<Student,
Integer>{

}
```

Step 7: Go to main Application java file that contain method and modify it with following code -

```
//StudentController.java
package com.acharya.studentdata;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class StudentController {
    @Autowired
    public StudentRepo repo;

    @PostMapping("/createData")
    public String createStudent(@RequestBody Student student){
        repo.save(student);
        return "Data added successfully.";
    }

    @GetMapping("/fetchData")
    public List<Student> fetch(){
        return repo.findAll();
    }
}
```

```
@DeleteMapping("removeData/{id}")
public String delete(@PathVariable int id) {
    repo.deleteById(id);

    return "Document deleted successfully";
}
```

Step 8: Navigate to `src/main/resource` folder and modify with the following code -

Note: Enter your MongoDB database name what you have to create and it should match what you will create on MongoDB server.

```
//Application.properties
server.port=8888
spring.data.mongodb.host=localhost
spring.data.mongodb.database=student
```

Step 9: To run the Application. Go to **Spring Boot Dashboard** present at left side of the VS Code. In Apps you will find your project name > **Click on Run** .

It will open the terminal and run the spring project

Step 10: Launch a command prompt and run the MongoDB server using

```
mongod
```

Step 11: Launch another command prompt and launch the MongoDB Shell

```
mongosh
```

Step 12: Create a database and collection as mentioned in our application. In my case the database name is 'student' and collection name is 'studentData'

Step 13: Launch Postman or Thunder Client click on New Request

Step 14: To test the method use the post after the endpoint URL
i.e., `http://localhost:PORT/label`

Step 15: Click on send and cross check the document from MongoDB

Experiment 23: Create a docker image and run a container using docker image. Also do the same using docker file.

Step 1: Go to play with docker official website

Step 2: Click on Login

Step 3: Click on start

Step 4: Click on create new instance at left side of the page

Step 5: To create a docker image run the following command –

```
docker create <IMAGE NAME>
```

Step 6: Show the existing images of docker use the command –

```
docker images
```

Step 7: To run a docker image in a container execute the following command –

```
docker run -it <IMAGE>
```

Step 8: Exit the container you're in.

Step 9: Click on Editor to create a dockerfile

Step 10: Enter the following script and save the file

```
FROM ubuntu
MAINTAINER vikas
RUN apt-get update
CMD ["echo", "Hello World"]
```

Step 11: Now rename the file to `Dockerfile`

```
mv <filename> Dockerfile
```

Step 12: To create an image and run container using the docker file hit the following command

```
docker build .
```

Step 13: If you want to deploy a web application create a NGINX container To do so follow the steps below

Create a NGINX image with -

Step 13.a: Run the NGINX container by binding the port number using -p flag

```
docker run -p 8085:80 nginx
```

Step 13.b: Now click on open port on top of screen and enter the port number you banded

Step 13.c: The deployed application will be opened in a new tab

Experiment 24: Create react application and deploy on AWS

Step 1: Create a folder on desktop

Step 2: Open the created folder in VS Code

Step 3: Right click on Explorer > Open Folder in Integrated Terminal

Step 4: Run the command to create a react application

Step 5: Go to GitHub official website

Step 6: Login to your GitHub account and create a repository

Step 7: Copy the HTTPS git link

Step 8: Now add the remote link to the git using the following command

Step 9: Now run the following command to push your application to the repository you just created

Step 10: Now open AWS Console

Step 11: Search for AWS Amplify in services and launch the AWS Amplify

Step 12: Click on get started > Web Hosting

Step 13: Select the GitHub as a storage for your react application and click Next