



**KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION)**

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
((Artificial Intelligence and Machine Learning))**

LAB RECORD

SOFTWARE ENGINEERING LAB

B.Tech. III YEAR I SEM (RKR21)

ACADEMIC YEAR 2024-25



KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad



Certificate

This is to certify that following is a Bonafide Record of the workbook task done by

_____ bearing Roll No _____ of _____

Branch of _____ year B. Tech Course in the _____

Subject during the Academic year _____ & _____ under our supervision.

Number of week tasks completed: _____

Signature of Staff Member Incharge

Signature of Head of the Dept.

Signature of Internal Examiner

Signature of External Examiner



KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad



Daily Laboratory Assessment Sheet

Name of the Lab:

Name of the Student:

Class:

HT. No:

S.No.	Name of the Experiment	Date	Observation Marks (3M)	Record Marks (4M)	Viva Voice Marks (3M)	Total Marks (10M)	Signature of Faculty
	TOTAL						

INDEX

S.NO	CONTENTS	PAGE NO
I	Vision/Mission /PEOs/POs/PSOs	1
II	Syllabus	7
III	Course outcomes, C0-PO Mapping	9
Exp No:	List of Experiments	
1	INSTALLATION OF STARUML, GIT BASH AND GITHUB ACCOUNT CREATION	11
2	IEEE – SRS (SOFTWARE REQUIREMENTS SPECIFICATION) WITH UML DIAGRAMS	15
3	INSTALLATION OF ECLIPSE, MAVEN, JDK, TOMCAT, CONFIGURING TOMCAT TO ECLIPSE	37
4	A. BASIC GIT COMMANDS - VERSION, CONFIG, INIT, STATUS, ADD, COMMIT, DIFF, HELP B. GIT COMMANDS: WORKING WITH LOCAL AND REMOTE REPOSITORIES - BRANCHES, CHECKOUT, MERGE, REVERT, LOG C. GIT COMMANDS: WORKING WITH REMOTE REPOSITORIES - REMOTE, CLONE, PULL, PUSH, FORK	44
5	A. CREATING MAVEN JAVA PROJECT USING ECLIPSE AND PUSH INTO TO GITHUB. B. CREATING MAVEN WEB PROJECT USING ECLIPSE AND PUSH INTO TO GITHUB. C. INSTALLATION OF JENKINS.	48
6	A. BUILDING THE CI/CD FREESTYLE PIPELINE USING JENKINS FOR MAVEN JAVA PROJECT B. BUILDING THE CI/CD FREESTYLE PIPELINE USING JENKINS FOR MAVEN WEB PROJECT WITH POLL SCM C. BUILDING THE CI/CD SCRIPTED PIPELINE USING JENKINS FOR MAVEN JAVA PROJECT WITH POLL SCM	59
7	A. INSTALLATION OF DOCKER, MINIKUBE, ACCOUNT IN DOCKERHUB B. DOCKER CLI COMMANDS C. MODIFY AND PUSH DOCKER IMAGE D. CREATE AND PUSH DOCKERFILE IMAGE E. RUNNING MULTIPLE CONTAINERS USING DOCKER COMPOSE F. DEPLOYING AND SCALING APPLICATIONS USING MINIKUBE G. DEPLOYING AND MANAGING MONITORING SYSTEMS USING NAGIOS IN DOCKER	65

Exp No:	List of Experiments	PAGE NO
8	A. AWS ACADEMY LEARNING ACCOUNT CREATION B. PROJECT DEPLOYMENT IN THE AWS CLOUD USING EC2 INSTANCE C. MAVEN WEB PROJECT DEPLOYMENT IN THE AWS CLOUD USING EC2 INSTANCE	82



KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY (AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029



Department of Computer Science & Engineering(AI & ML)

Vision of the Institution:

To be the fountain head of latest technologies, producing highly skilled, globally competent engineers.

Mission of the Institution:

- To provide a learning environment that inculcates problem solving skills, professional, ethical responsibilities, lifelong learning through multi modal platforms and prepare students to become successful professionals.
- To establish Industry Institute Interaction to make students ready for the industry.
- To provide exposure to students on latest hardware and software tools.
- To promote research-based projects/activities in the emerging areas of technology convergence.
- To encourage and enable students to not merely seek jobs from the industry but also to create new enterprises
- To induce a spirit of nationalism which will enable the student to develop, understand India's challenges and to encourage them to develop effective solutions.
- To support the faculty to accelerate their learning curve to deliver excellent service to students



KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029



NBA
ACCREDITED

Department of Computer Science & Engineering (AI & ML)

Vision of the Department:

To be among the region's premier teaching and research Computer Science and Engineering departments producing globally competent and socially responsible graduates in the most conducive academic environment

Mission of the Department:

- To provide faculty with state-of-the-art facilities for continuous professional development and research, both in foundational aspects and of relevance to emerging computing trends.
- To impart skills that transform students to develop technical solutions for societal needs and inculcate entrepreneurial talents.
- To inculcate an ability in students to pursue the advancement of knowledge in various specializations of Computer Science and Engineering and make them industry-ready.
- To engage in collaborative research with academia and industry and generate adequate resources for research activities for seamless transfer of knowledge resulting in sponsored projects and consultancy.
- To cultivate responsibility through sharing of knowledge and innovative computing solutions that benefit the society-at-large.
- To collaborate with academia, industry and community to set high standards in academic excellence and in fulfilling societal responsibilities



Department of Computer Science & Engineering (AI & ML)

PROGRAM OUTCOMES (POs)

PO1: Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/Development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct Investigations of Complex Problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

PO6: The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and Team Work: Function effectively as an individual, and as a member or

leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project Management and Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long Learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.



KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029



Department of Computer Science & Engineering(AI & ML)

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1: An ability to analyze the common business functions to design and develop appropriate Computer Science solutions for social upliftment.

PSO2: Shall have expertise on the evolving technologies like Python, Machine Learning, Deep Learning, Internet of Things (IOT), Data Science, Full stack development, Social Networks, Cyber Security, Big Data, Mobile Apps, CRM, ERP etc.



KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY (AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029



Department of Computer Science & Engineering(AI & ML)

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates will endeavor to excel in their chosen careers as professionals, researchers and entrepreneurs on a global platform.

PEO2: Graduates will demonstrate the ability to solve challenges in the fields of Engineering and Technology simultaneously catering to societal needs.

PEO3: Graduates will strive to improve their learning curve by practicing Continuing Professional Development (CPD).

PEO4: Graduates will, at all times, adopt a professional demeanor by communicating effectively, working collaboratively, and maintaining the ethics & core values as befitting their education in interdisciplinary and emerging fields.

B. Tech. in COMPUTER SCIENCE AND ENGINEERING**III Year I Semester Syllabus (RKR21)****SOFTWARE ENGINEERING LAB (21CC505PC)****Common to CSE, IT, CSE (AI&ML) and CSE (DS)**

L	T	P	C
0	0	3	1.5

Pre-requisites/ Co-requisites:

1. 21CC502PC – Software Engineering Course
2. 21CS401PC- Java Programming Course

Course Objectives: The course will help to

1. Formulate problem statements and Software Requirement Specifications by comprehensively grasping project requirements.
2. Demonstrate proficiency in designing, developing, and testing diverse project modules.
3. Utilize Git Framework and GitHub while implementing Continuous Integration/Continuous Deployment (CI/CD) pipelines through Jenkins.
4. Implement project deployment using Docker and Kubernetes.
5. Acquire knowledge in AWS cloud infrastructure.

Course Outcomes: After learning the concepts of this course, the student is able to

1. Transform end-user needs into system and software requirements through a structured process.
2. Depict the system's high-level design using CASE tools based on the software requirements.
3. Employ Jenkins CI/CD for project building purposes.
4. Implement project deployment utilizing Docker and Kubernetes.
5. Create a project within the AWS Cloud environment.

Software to be used: The students must use JDK 11/17/21 Version, STAR UML, GIT Bash, Jenkins, Dockers Desktop, Mini KUBE, Eclipse, Tomcat, and Visual Studio Editor.

List of Experiments:

Do the following exercises for any one project given in the list of sample projects or any other projects?

1. Development of problem statement.
2. Preparation of Software Requirement Specification Document, Design Documents and Testing Phase related documents.
3. Study and usage of any Design phase CASE tool
4. Creating the project and committing using Git and GitHub
5. Creating Maven Java and Maven Web project using Eclipse and Push them to GitHub.
6. Building the CI/CD pipeline using Jenkins for the project in the previous experiment.
7. Local Deployment of project using Docker, Kubernetes and Monitoring using Nagios tool.
8. Cloud Deployment of a project in the AWS Cloud using EC2 instance.

Sample Projects:

1. Book Bank
2. Online course reservation system
3. E-ticketing
4. Recruitment system
5. Hospital Management system
6. Online Banking System

TEXT BOOKS:

1. Software Engineering, A practitioner's Approach- Roger S. Pressman, 6th edition, Mc Graw Hill International Edition, 2015.
2. Software Engineering- Sommerville, 7th edition, Pearson Education, 2017.
3. The unified modeling language user guide Grady Brooch, James Rumbaugh, Ivar Jacobson, Pearson Education, 2016.
4. The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations, 2015.

REFERENCE BOOKS:

1. <https://kubernetes.io/docs/tutorials/hello-minikube/>
2. <https://minikube.sigs.k8s.io/docs/start/>
3. <https://www.jenkins.io/doc/>
4. <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html>
5. Introducing Maven by, Balaji Varanasi and Sudha Belida, APRESS publications.



KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029



NBA
NAAC

Department of Computer Science & Engineering (AI & ML)

Course Outcomes and CO-PO-PSO Mapping

Course Outcomes:

After learning the contents of this course, the student is able to

CO1	Transform end-user needs into system and software requirements through a structured process.
CO2	Depict the system's high-level design using CASE tools based on the software requirements.
CO3	Employ Jenkins CI/CD for project building purposes.
CO4	Implement project deployment utilizing Docker and Kubernetes.
CO5	Create a project within the AWS Cloud environment.

CO-PO-PSO MAPPING:

	CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO-1	PSO-2
Software Engineering Lab	CO1	3	3	3	2						2			2	1
	CO2	2	3	3	3	3					2			2	1
	CO3					3				2	2	2		1	2
	CO4				3	3	2	2		2	2	2	2	2	3
	CO5	3				3	2	2				3	3	3	3

Software and Hardware Requirements

1. Software Tools

- a. Star UML**
- b. Java 11/17/21 Version**
- c. Apache Tomcat 9**
- d. Eclipse IDE**
- e. Visual Studio**
- f. Jenkins**
- g. Git Bash for Windows**
- h. Windows Docker Desktop**
- i. AWS Cloud account (Basic Plan)**

3. Hardware Requirements

- a. Windows 10 Pro**
- b. 8 GB RAM (64-bit Processor)**
- c. 512 GB HDD**
- d. i5 Core Processor**
- e. BIOS-level hardware virtualization support is enabled in the BIOS settings.**

Experiment 1: Installation of StarUML, Git Bash and GitHub Account Creation

AIM: To experience the **UML diagrams, version control system**, Tracking code changes and code collaboration.

Introduction:

❖ **Star UML:**

- StarUML is a powerful software modeling tool used for creating Unified Modeling Language (UML) diagrams.
- It is widely adopted by developers and designers to visualize and document the architecture of software systems.
- **Features of StarUML**
 - Supports various **UML diagrams** for analysis and design.
 - Offers **code generation** for languages like Java, C++, and Python.
 - Allows **collaboration** with team members via plugins.
 - Compatible with industry-standard modeling notations like **UML 2.x**.

❖ **UML diagrams:**

- UML (Unified Modeling Language) diagrams are standardized visual representations of a software system's structure and behavior. They help developers and stakeholders understand and communicate system requirements. developers or teams
- **Types of UML Diagrams**
 - **Structural Diagrams:** Focus on the system's static aspects.
 - Class Diagram
 - Component Diagram
 - Object Diagram
 - Deployment Diagram
 - **Behavioral Diagrams:** Represent the system's dynamic aspects.
 - Use Case Diagram
 - Sequence Diagram
 - Activity Diagram
 - State Diagram

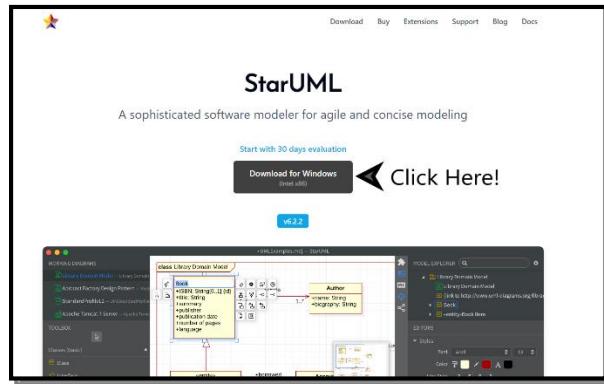
❖ **Git Bash and GitHub:**

- Git is a popular version control system. It was created by Linus Torvalds in 2005.
- It is designed to make it easier to have multiple versions of a code base, sometimes across multiple developers or teams
- **Functions of Git**
 - Manage projects with Repositories
 - Clone a project to work on a local copy
 - Control and track changes with Staging and Committing
 - Branch and Merge to allow for work on different parts and versions of a project
 - Pull the latest version of the project to a local copy
 - Push local updates to the main project

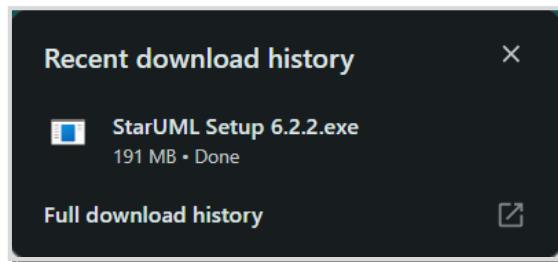
Procedure

STARUML Installations

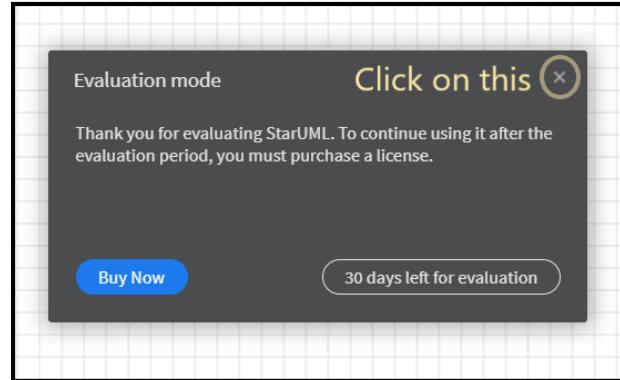
STEP 1 : Visit staruml.io



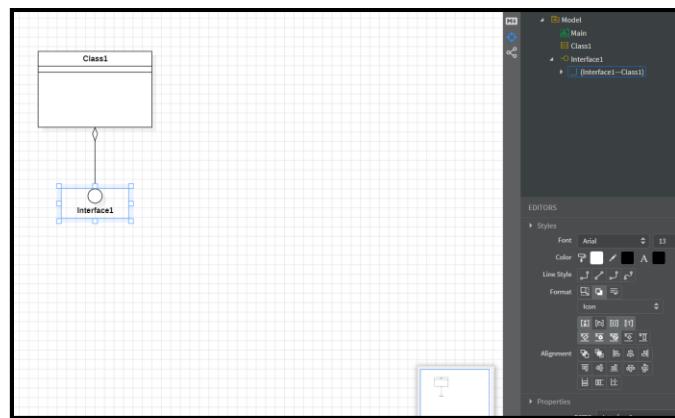
STEP 2 : Click on **Download** and run the **.exe** file



STEP 3: When you get the **evaluation mode** screen



STEP 4: Click on **x / cross** and enjoy your UML

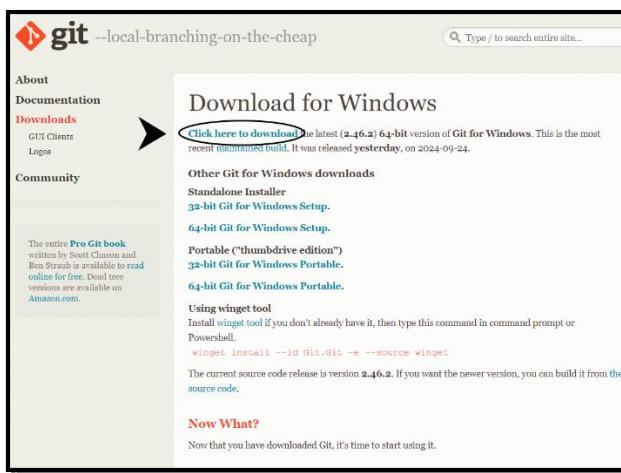


GITBASH Installations

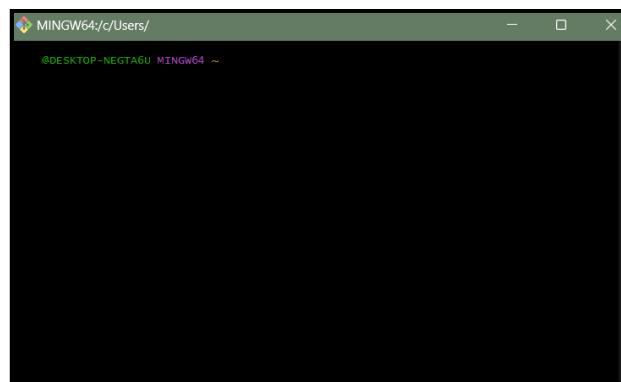
STEP 1 : Visit git-scm.com/downloads



STEP 2: Click on your OS hyperlink under Download and click on Download

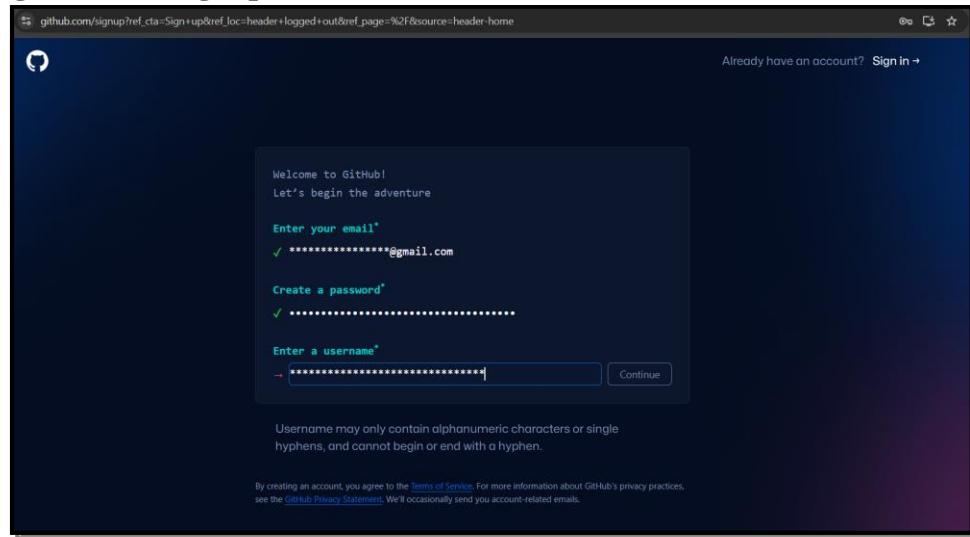


STEP 3: After Download completes run the exe file and enjoy GitBash

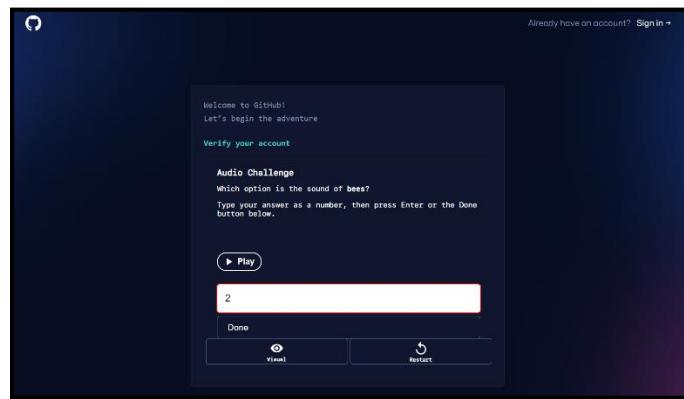


Creating GitHub Account

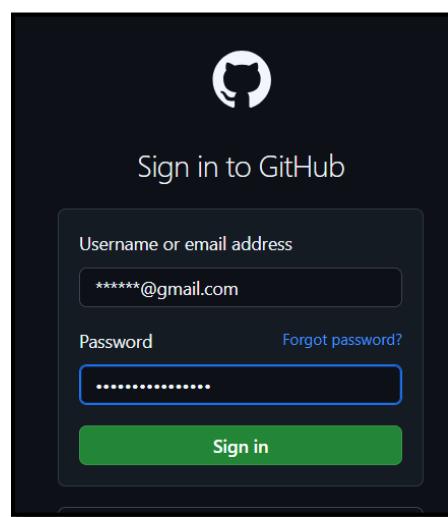
STEP 1 : Visit github.com/signup



STEP 2 : Enter your Email, Password and Username and Verify your account



STEP 3 : Login into your account



Conclusion : Thus we have installed StarUML, GitBash and created a GitHub account.

Experiment 2: IEEE – SRS (SOFTWARE REQUIREMENTS SPECIFICATION) WITH UML DIAGRAMS

AIM: To develop a project on a **E-Ticketing system** using **StarUML**

SRS:

Problem Statement:

In the rapidly digitizing E-ticketing industry, users increasingly demand a seamless and efficient experience when booking tickets for events such as concerts and marriages. However, the current fragmented experience across different platforms often results in frustration due to inconsistent ticket tracking, missing payment receipts, and lack of real-time status updates. Our E-Ticketing application, accessible via a server and a webpage, is designed to address these challenges by offering a unified platform that enables users to book tickets effortlessly.

The application aims to provide a cohesive and secure experience for users, allowing them to browse and book tickets for events, make payments through an integrated payment gateway, and receive receipts immediately, which they can download and store for future reference. Additionally, users will be able to track the status of their tickets in real-time, ensuring they remain informed and updated throughout the booking process. Notifications regarding event status, such as confirmations or cancellations, will further enhance user satisfaction.

By simplifying the interface and focusing on key functionalities, the E-ticketing system ensures a streamlined, user-friendly experience, transforming the ticketing process into a hassle-free and enjoyable journey for all users.

Members:

- | | | |
|--------------------------|---|------------|
| 1. Chodavarapu Srinidhi | - | 22BD1A6712 |
| 2. Darsh Agrawal | - | 22BD1A6714 |
| 3. Mohammed Areeb Akhter | - | 22BD1A6737 |
| 4. Pasham Ayush Reddy | - | 22BD1A6744 |

Software Requirement Specification

For

E-Ticketing System

Version 3.0

Prepared by:

- | | | |
|--------------------------|---|------------|
| 1. Chodavarapu Srinidhi | - | 22BD1A6712 |
| 2. Darsh Agrawal | - | 22BD1A6714 |
| 3. Mohammed Areeb Akhter | - | 22BD1A6737 |
| 4. Pasham Ayush Reddy | - | 22BD1A6744 |

Keshav Memorial Institute of Technology

25-10-2024

Table of Contents

1. Introduction	
1.1 Purpose.....	1
1.2 Document Conventions.....	1
1.3 Intended Audience and Reading Suggestions.....	1
1.4 Product Scope.....	2
1.5 References.....	2
2. Overall Description	
2.1 Product Perspective.....	3
2.2 Product Functions.....	3
2.3 Operating Environment.....	4
2.4 User Characteristics.....	4
2.5 Design and Implementation Constraints.....	5
2.6 Assumptions and Dependencies.....	5
3. External Interface Requirements	
3.1 User Interfaces.....	6
3.2 Hardware Interfaces.....	6
3.3 Software Interfaces.....	6
3.4 Communications Interfaces.....	7
4. System Features	
4.1 Event Organizer/Management Staff.....	8
4.2 Event Attendee.....	9
5. Other Non-functional Requirements	
5.1 Performance Requirements	10
5.2 Safety Requirements	10
5.3 Security Requirements	11
5.4 Software Quality Attributes.....	11
5.5 Business Rules	12
6. Other Requirements	
Appendix A: Glossary.....	13
Appendix B: Analysis Models.....	15
Appendix C: To Be Determined List.....	21

Revision History

Name	Date	Reason for changes	Version
Week-1	26-09-24	SRS Draft Template	1.0
Week-2	19-10-24	SRS Documentation - Use case, Class Diagram	2.0
Week-3	25-10-24	SRS Final Document – Component, Sequence Diagram	3.0

1. Introduction

The Software Requirement Specification is designed to document and describe the agreement between the customer and developer regarding the specification of the software product requested. This documentation is done to provide a clear idea of customer requirements. This document can be used as reference in further development of the software system.

1.1 Purpose

The purpose of this document is to specify the software requirements for an E-ticketing system that enables users to browse and book tickets for events like concerts and marriages. It describes the system's functionality, expected behaviour, and constraints, providing a reference for developers, testers, and stakeholders involved in the project.

1.2 Document Convention

Heading:

Font-Size: 16, Font-Style: Bold, Font: Times New Roman

Subheading:

Font-Size: 14, Font-Style: Bold, Font: Times New Roman

Content:

Font-Size: 12, Font: Times New Roman.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers responsible for system implementation, providing them with clear specifications for building the E-Ticketing platform. It is also designed for testers who will validate the functionalities outlined in the requirements, ensuring that the system performs as expected. Managers can use this document for project tracking, helping them monitor progress and ensure that the system is developed according to plan. Additionally, it serves event organizers using the platform, allowing them to understand the system's capabilities for managing events, ticketing, and user interactions.

1.4 Product Scope

The E-ticketing system focuses on creating a unified platform for users to browse, book, and manage tickets for events like concerts and marriages. The system will operate on a server and be accessible through a webpage. It will support an integrated payment system and real-time status updates. Users can track the event status and download tickets and receipts via the website.

1.5 References

<https://www.studocu.com/in/document/university-college-of-engineering/software-engineering/e-ticketting-software-engineering-projects/65486732>

<https://www.studocu.com/in/document/university-college-of-engineering/software-engineering/e-ticketting-software-engineering-projects/65486732>

2. Overall Description

2.1 Product Perspective

The E-ticketing system will function as a single-server platform, accessible via a webpage. It will provide a seamless ticketing experience for events like concerts and marriages, offering an efficient booking system with real-time tracking, secure payment, and receipt generation.

2.2 Product functions

2.2.1 Recruiter Portal

- **Job Posting Management:** Recruiters can post event tickets for concerts and marriages, update event details, and specify ticketing requirements.
- **Ticket Management:** Recruiters can manage ticket sales, view bookings, and update available slots based on event capacity.
- **Notification and Alerts:** Recruiters will receive real-time notifications when a user books a ticket, or when an event update occurs.
- **Attendee Database:** Recruiters can access a searchable database of attendees, categorized by booking details and ticket type.

2.2.2 Attendee Portal

- **Profile Management:** Attendees can create and update their profiles, including adding personal details and past event bookings. Filter and search for specific events
- **Event Search and Booking:** Attendees can search for concerts or marriage events based on various filters like location, date, and event type, and book tickets directly from the platform.
- **Booking Tracking:** Attendees can track the status of their ticket bookings in real time, including confirmations or cancellations.
- **Event Recommendations:** The system will suggest upcoming events based on the attendee's profile and booking history.

2.3 Operating Environment

- Web browsers (Chrome, Firefox, Safari, Edge) This E - Ticketing server shall operate in all famous browsers, for a model we are taking ,Internet Explorer Versions 7.0,8.0 and 9.0,with flash player 9 and JavaScript.
- This e-commerce website shall operate in all famous browsers, for a model we are taking, Internet Explorer Versions 7.0, 8.0 and 9.0, with flash player 9 and JavaScript.

2.4 User Characteristics

2.4.1 Event Attendees:

- The primary users who will use the system to browse and book tickets for concerts and marriages.
- Must have basic computer literacy, including the ability to navigate websites and complete online payments.
- Will access the system through web browsers on PCs or laptops with a stable internet connection.
- Responsibilities include searching for events, booking tickets, and managing bookings (e.g., checking event updates, downloading receipts).

2.4.2 Event Organizers:

- The primary users who will use the system to browse and book tickets for concerts and marriages.
- Must have basic computer literacy, including the ability to navigate websites and complete online payments.
- Will access the system via web browsers on PCs or laptops with a stable internet connection.
- Responsibilities include creating and managing event listings, tracking ticket sales and attendees, updating event details, and handling booking inquiries.

2.5 Design and Implementation Constraints:

- The system will be hosted on a server and accessed via modern web browsers, supporting devices like PCs and laptops with a stable internet connection.
- All transactions should be secured using SSL encryption.

2.6 Assumptions and Dependencies

- MERN stack (MongoDB, Express, React, Node.js) for the web application development.
- Python Django for server-side processing and APIs.
- JSON for data exchange between the server and the client – side interface.
- Reliable internet access for both event organizers and attendees.
- Users being comfortable with basic web navigation and online payments.

3. External Requirements

3.1 User Interfaces:

- The system will provide a **web-based interface** accessible via modern web browsers such as Chrome, Firefox, and Edge.
- The user interface will be intuitive and responsive, allowing both event attendees and event organizers to navigate easily.
- Attendees will access event listings, booking forms, and payment processing through a simple, mobile-friendly layout.
- Organizers will have access to an administrative dashboard where they can manage event details, track bookings, and view attendee information.

3.2 Hardware Interfaces:

- The system does not require specific hardware interfaces; it will be accessed via **standard PCs or laptops** with stable internet connections.
- The server hosting the system will require standard hardware to support the **MERN stack** and **Python Django** server, ensuring optimal performance under expected traffic loads.
- The system may also be optimized for **mobile-friendly access** but does not require specialized mobile hardware.

3.3 Software Interfaces:

- The system will integrate with **MongoDB** for data management and storage
- It will use **third-party payment gateways** for processing online transactions securely.
- **API** integration will allow external systems (e.g., notifications, mailing services) to connect for additional functionalities.
- **React** will be used for building the client-side web application, while **Express.js** and **Node.js** will handle server-side operations.

3.4 Communications Interfaces:

- The system will use **HTTP/HTTPS** protocols for secure communication between users' browsers and the server.
- All communication between the client and server, including login, event browsing, and payment, will be encrypted using **SSL/TLS**.
- The system will support **real-time notifications** and updates via **Web Sockets** or similar technology for delivering booking confirmations and event status changes to users

4. System Features:

These requirements include the development of search tools, sorting, filtering, navigation, and visual components of the platform, which can be maintained by event organizers.

4.1 Event Organizer/Management Staff:

Requirement ID	:	RI.01.01
Title	:	Event Management
Description	:	Control the entire database containing records of events, bookings, and attendee details. Any issues related to accessing or updating the database should be resolved as quickly as possible.
Priority	:	1
Requirement ID	:	RI.01.02
Title	:	Event Created and Update
Description	:	Event organizers can create new event listings (concerts and marriages), update event details (date, time, venue), and set ticket availability.
Priority	:	1
Requirement ID	:	RI.01.03
Title	:	View all bookings
Description	:	Event organizers must be able to view all ticket bookings, track attendee details, and manage ticket availability in real time.
Priority	:	1
Requirement ID	:	RI.01.04
Title	:	Event Notifications
Description	:	Organizers can send notifications (event updates, cancellations) to attendees based on their bookings.
Priority	:	2

4.2 Event Attendee:

Requirement ID	: R1.02.01
Title	: Attendee Registration
Description	: New attendees should sign up by creating an account with valid login credentials to access the platform.
Priority	: 1
Requirement ID	: R1.02.02
Title	: Attendee Login
Description	: Attendees must use their registered credentials to log into the system and access available events.
Priority	: 1
Requirement ID	: R1.02.03
Title	: View and Edit Personal Details
Description	: Attendees must be able to view and edit their personal details (e.g., name, contact information) in their profile.
Priority	: 2
Requirement ID	: R1.02.04
Title	: Browse and Book Events
Description	: Attendees must be able to browse available concerts and marriages, apply filters, and book tickets directly through the platform.
Priority	: 2

5. Other Non-Functional Requirements

5.1 Performance Requirements

- **Response Time:** Transactions such as loading applications, allotting interview and must complete within 2-5 seconds. Account details and applications should load within 1 second under normal load conditions.
- **Concurrent Users:** The system must handle up to 5000 concurrent users without significant degradation in performance. The system should scale to accommodate peak traffic, especially during working hours and end-of-month transactions.
- **Database Transactions:** Each transaction should commit to the database within 1 second, ensuring data consistency and atomicity (ACID compliance) to avoid transactional errors.
- **Backup Speed:** Automated backups of sensitive data should occur within off-peak hours and must not impact system availability. The system must recover within 15 minutes in the event of failure.

5.2 System Requirements

- **Data Loss Prevention:** In case of a system crash or unexpected shutdown, all pending transactions must be either rolled back or stored securely to prevent any loss. The system must log any discrepancies and alert the administrators immediately.
- **Physical Safety:** The system must ensure that physical access to critical server components is restricted to authorized personnel only. No sensitive operations should be allowed unless the user has the correct level of access.
- **Transaction Safety:** To prevent incorrect transactions, the system must perform thorough checks, including balance verification and approval workflows for high-value transactions. Any failed transaction should trigger automated rollback mechanisms and alert the customer.
- **Fraud Detection:** The system should have a built-in fraud detection mechanism to alert and block suspicious or unauthorized activities based on user behaviour analysis and transaction patterns.

5.3 Security Requirements

- **User Authentication:** All users, whether customer, bank staff, or admins, must authenticate via a secure two-factor authentication (2FA) system before accessing the platform. Passwords should adhere to the latest encryption standards and be stored using cryptographic hashing (e.g., SHA-256).

- **Data Encryption:** Sensitive data like passwords, transaction details, and personal information must be encrypted in transit and at rest using at least 256-bit AES encryption.
- **Access Control:** Different levels of access should be enforced:
 - **User:** Can only access personal details, applications and search for events.
 - **Organiser:** Can only put forward a request to add an event to the admin.
 - **Admin:** Can modify system-wide configurations, grant permissions, access logs.

5.4 Software Quality Attributes

- **Availability:** The system must be available 99.99% of the time, ensuring minimal downtime, particularly during critical financial operations like payroll processing.
- **Reliability:** Transactions must be processed reliably, without any duplication or data loss. Fail-safe mechanisms should prevent incomplete transactions.
- **Scalability:** The system must be able to scale horizontally, supporting additional users and accounts without performance degradation as the number of customers grows.
- **Maintainability:** The code base should be modular and easy to maintain, allowing for quick fixes and updates. Routine maintenance must be conducted without affecting system availability.
- **Portability:** The system should support deployment across various platforms and cloud services to allow for distributed operations and disaster recovery.
- **Interoperability:** The system must be able to integrate with third-party services like payment gateways (UPI, credit cards, internet banking) and notification services (SMS, email).
- **Usability:** The user interface must be intuitive and easy to use, ensuring a smooth banking experience for all customer demographics.
- **Testability:** All features must be easily testable with automated scripts, especially for security vulnerabilities, performance benchmarks, and functional testing.

5.5 Business Rules

- **Applicant Permissions:** Applicants can only access their own account details, manage their application, and apply for jobs. No applicant can access another applicant's account details.
- **Recruiter Permissions:** Recruiter can approve or reject applications, allot interview slots, and oversee all the applications. Permissions are role-specific.
- **Administrator Permissions:** Admins can override any system settings, including enabling or disabling recruitment services, modifying job details, and managing staff permissions.
- **Alerts & notifications:** The applicant must receive alerts as soon as the recruiter allots an interview slot and as well as the status of his application.

6. Other Requirements

Appendix A: Glossary

- **Attendee:** An individual looking to book tickets for events such as concerts and marriages via the E-ticketing platform. They can browse events, book tickets, and manage their bookings.
- **Event Organizer:** A person or organization using the E-ticketing system to create and manage event listings, track ticket sales, and send notifications to attendees.
- **Booking System:** The feature that allows attendees to search for events, select tickets, make payments, and receive confirmation receipts.
- **Ticket:** A digital entry pass purchased by an attendee for an event. The system generates tickets after successful booking and payment.
- **Event Management:** A process managed by event organizers to create, update, and oversee event details such as time, location, and ticket availability.
- **Encryption:** The method of securing sensitive data, such as user credentials and payment information, ensuring that all data transmitted and stored is encrypted to maintain security and privacy.
- **Payment Gateway:** A third-party service integrated into the platform to process payments securely and generate receipts for successful ticket purchases.
- **Real-Time Updates:** Instant updates provided by the system regarding event status, booking confirmations, or cancellations, ensuring that users receive timely notifications.
- **Responsive User Interface (UI):** A design feature that adjusts the layout of the platform to various device screen sizes, providing a smooth user experience on both desktop and mobile browsers.
- **MongoDB:** A NoSQL database used to store and manage event and user data in the E-ticketing system.
- **User Interface (UI):** The visual part of the platform through which attendees and organizers interact with the system, including navigation menus, event listings, and booking forms.
- **Availability:** The system's capability to be accessible at all times, allowing users to browse events, book tickets, or manage events 24/7.

- **Cross-Platform:** A feature ensuring the E-ticketing system operates smoothly across multiple devices, such as desktops, tablets, and smartphones, through web browsers.
- **Event Notifications:** Alerts provided to attendees by organizers about event changes, confirmations, or cancellations.
- **Scheduled Maintenance:** Periodic downtime planned for system updates and improvements to ensure the platform remains secure and performs optimally.
- **User Assistance:** An integrated help system or FAQ section that guides attendees and organizers through using the platform efficiently, ensuring ease of use.
- **Appendix B: Analysis Models**
- **Use Case Template:**

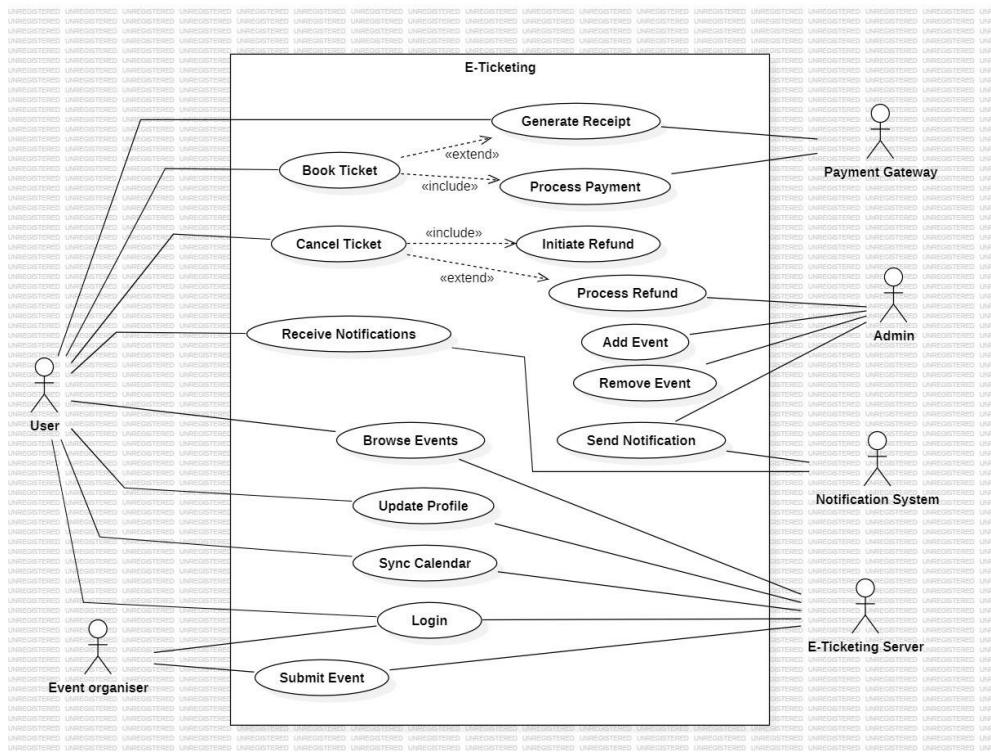
Use Case ID:	1234322		
Use Case Name:	E-Ticketing		
End Objective:	Make a E-ticketing platform for events		
Created By:	1. Chodavarapu Srinidhi 2. Darsh Agrawal 3. Areeb Akhtar 4. Pasham Ayush Reddy	On (date):	October 18, 2024
User / Actor	User, Admin, Event organizer		
Trigger:	User logging in the site to buy tickets for events		
Basic/Normal flows			
User Actions	System Actions		
The user login to the site by entering details.	Login page requests the user to provide a proper username and password.		

User views events present on the site.	The home page suggests to the user some events or the user can search for the required event.
Users make purchases of the tickets of the event with appropriate purchase options.	Site will provide users with COD or other online payment methods.
User wants to view and edit his/her own details.	Site will provide the user to edit the changes
User completes the purchase and checkout.	Site processes the purchase and gives tickets.
Users can sync their calendar with the site.	Site syncs his calendar.
Users can receive notifications about the latest events.	Site notifies users with the latest events.
Users can cancel their tickets.	Admin processes the refund of the tickets

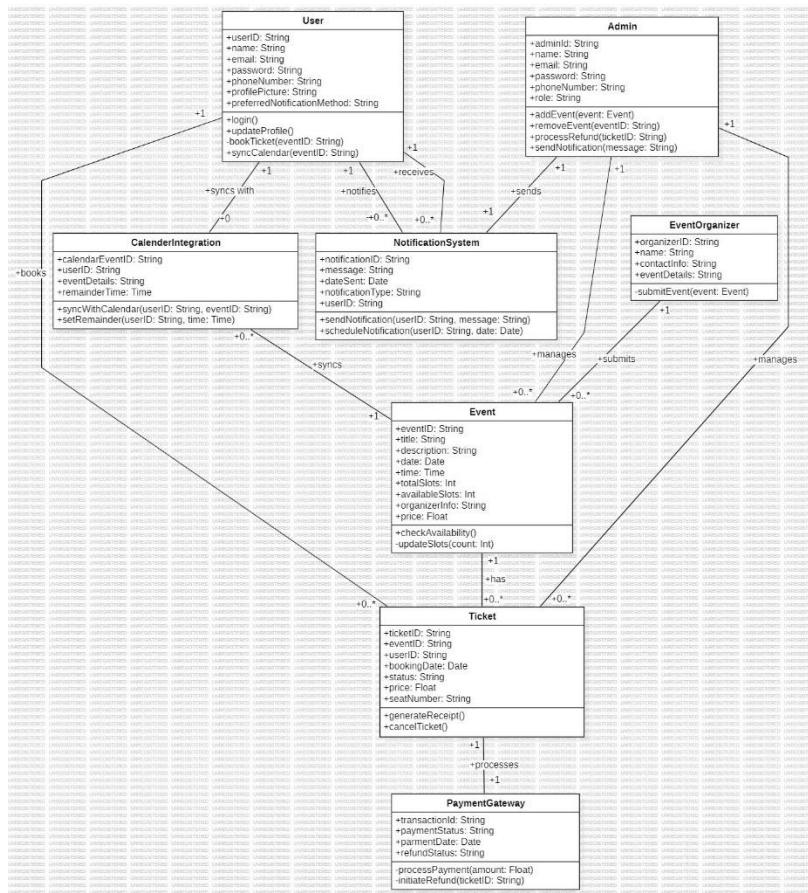
Exception Flows

User Actions	System Actions
The user tries to login but doesn't have an account on the website.	The page requests the user to register an account in the registration page before committing the login.
The user tries to login by entering details.	The details entered are incorrect. Error message is displayed and the user needs to enter the correct details.
User tries to book tickets for a sold out event.	A "Sold out" message is displayed.
Payment by user fails.	A payment failed message is displayed.

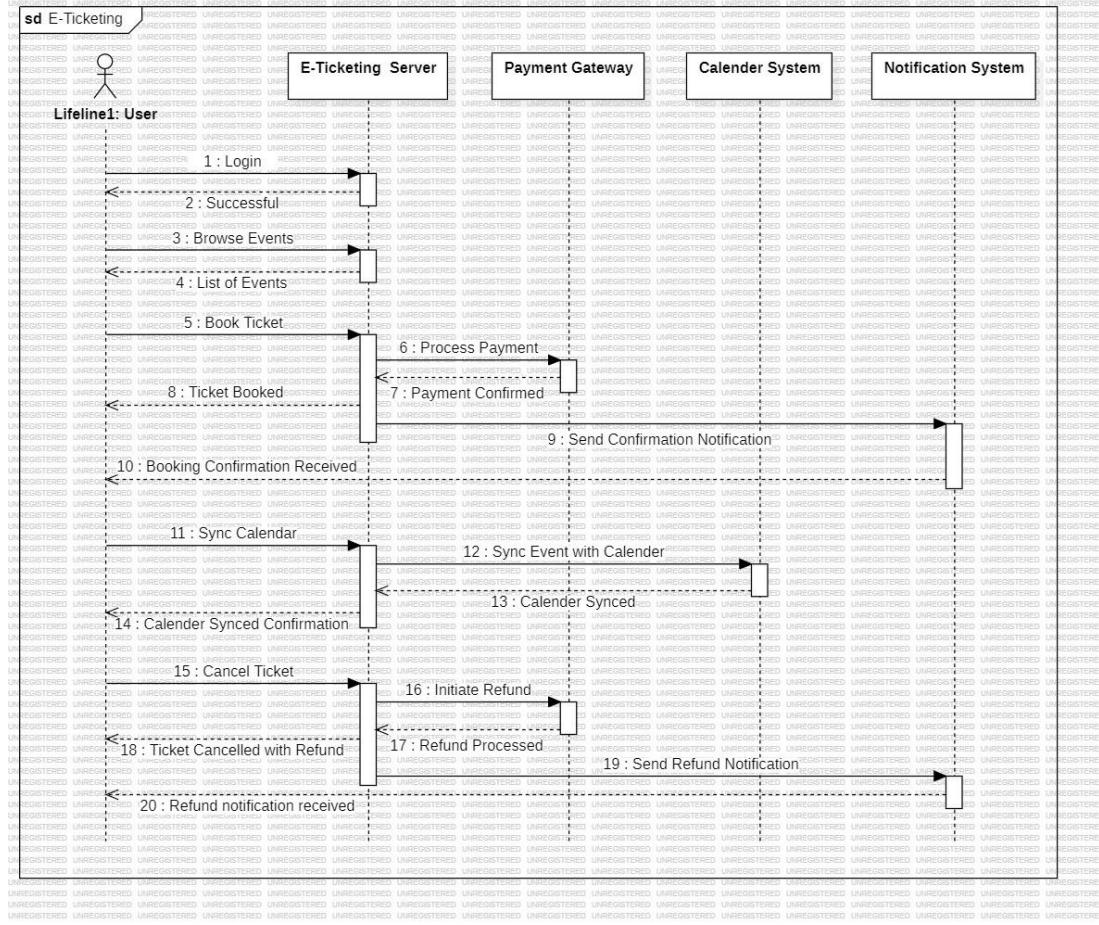
Use-Case Diagram:



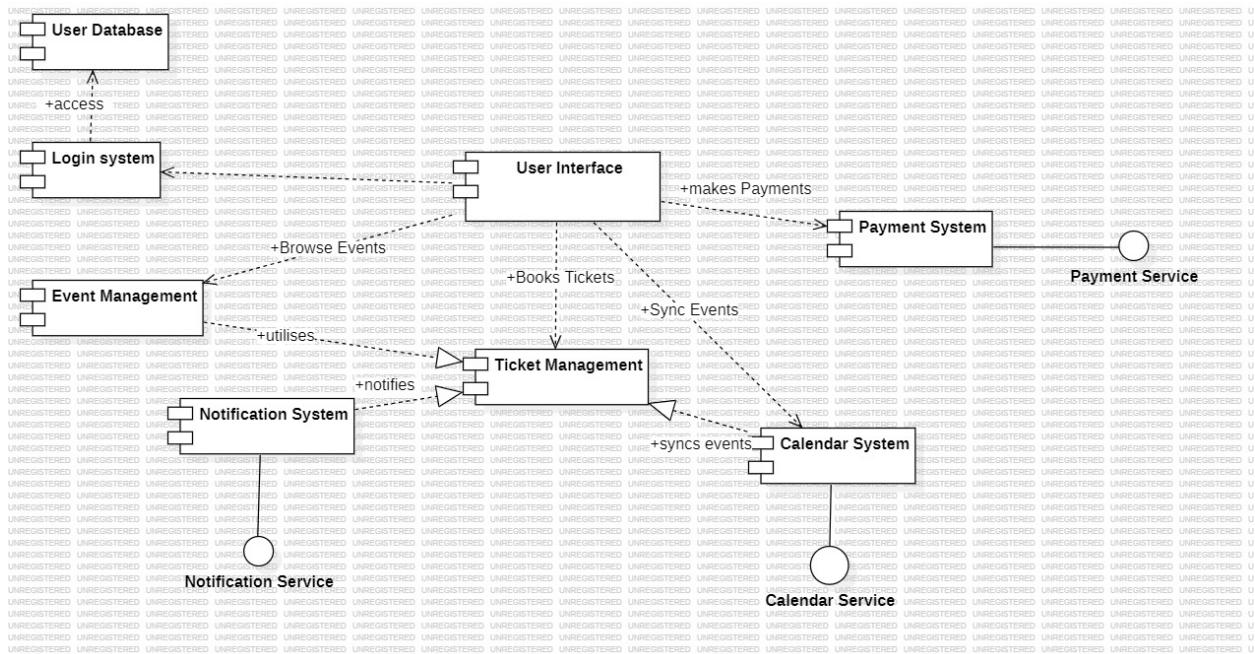
Class Diagram:



Sequence Diagram:



Component Diagram:



Appendix C: To Be Determined (TBD) List

This appendix lists items that need further clarification or finalization during the project's development. These items will be addressed as the project evolves, and final decisions will be made during subsequent stages.

- **Third-Party Payment Systems:** The exact payment gateways (e.g., PayPal, Stripe, Google Pay) to be integrated for processing payments securely and efficiently are still under evaluation.
- **Finalized User Roles and Permissions:** User roles and permission levels, especially for the **Admin** class, need further definition. This will include detailed permissions (e.g., VIP, standard, student discounts, managing users, and generating reports) and access control for sensitive actions.
- **Tickets Types and Categories:** Finalization of the specific ticket types (e.g., VIP, standard, student discounts) and categories (e.g., transportation, event types) the system will support.

Conclusion : Thus we created SRS document for E-Ticketing System.

Experiment 3: INSTALLATION OF ECLIPSE, MAVEN, JDK, TOMCAT, CONFIGURING TOMCAT TO ECLIPSE

AIM: To experience the Eclipse IDE, Maven software, JDK and Tomcat Server Configuration.

Introduction:

❖ Eclipse IDE

- Eclipse IDE is a popular, open-source integrated development environment (IDE) primarily used for Java programming. It provides tools for coding, debugging, and testing applications, supporting multiple programming languages through plugins.
- **Features of StarUML:**
 - **Code Editing** with syntax highlighting and auto-completion.
 - **Debugging Tools** for error tracking.
 - **Integration** with version control systems like Git.
 - **Plugin Support** for languages like Python, C++, and more.

❖ Maven Software

- Apache Maven is a build automation and project management tool designed for Java projects. It uses a POM (Project Object Model) file to manage project dependencies and configurations.
- **Key Features of Maven:**
 - Dependency Management: Automatically downloads required libraries.
 - Build Automation: Simplifies compiling, testing, and packaging applications.
 - Project Standardization: Follows a structured directory format.

❖ JDK (Java Development Kit)

- The JDK is a software development environment used for developing Java applications. It includes tools like the Java compiler, libraries, and the Java Virtual Machine (JVM).
- **JDK Installation and Configuration:**
 - Download the latest JDK from the [Oracle website](#).
 - Install and set the JAVA_HOME environment variable.
 - Add the bin folder to the Path environment variable for global access to Java commands.

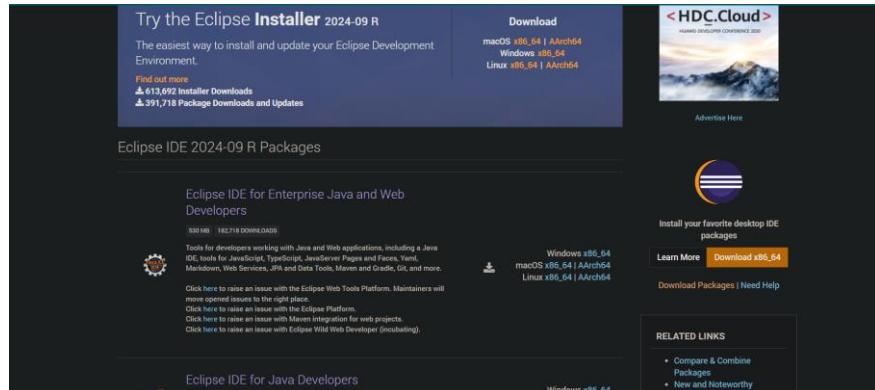
❖ Tomcat Server Configuration

- Apache Tomcat is a web server and servlet container used to deploy and run Java-based web applications.
- **Steps to configure Tomcat:**
 - Download the Tomcat server from the [official website](#).
 - Extract the archive and set the CATALINA_HOME environment variable to the Tomcat directory.
 - Add the bin folder to the Path environment variable.
 - Deploy web applications in the webapps folder.
 - Start the server using the startup.bat (Windows) or startup.sh (Linux) script.
 - Access the server at <http://localhost:8080>

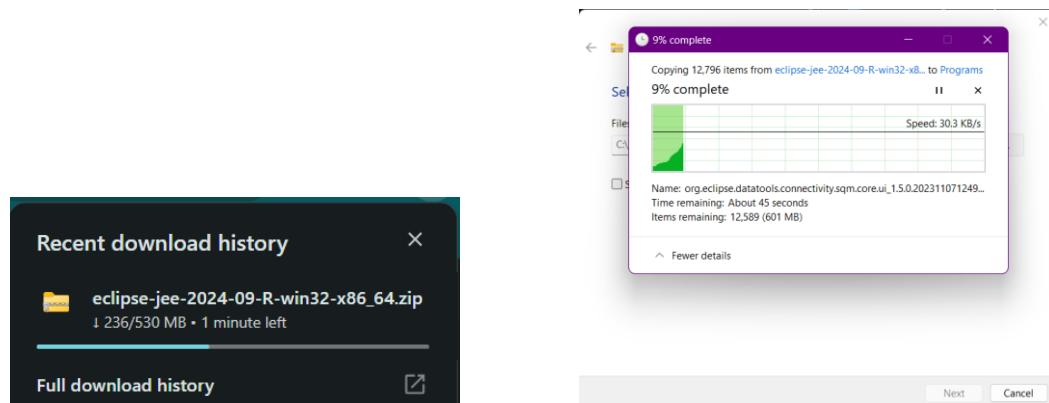
Procedure

ECLIPSE IDE Installations

STEP 1 : Visit <https://www.eclipse.org/downloads/packages/> and go to Java And Web Developers



STEP 2 : After the download completes extract the zip file



STEP 3 : After extraction you have the eclipse set up and ready to run

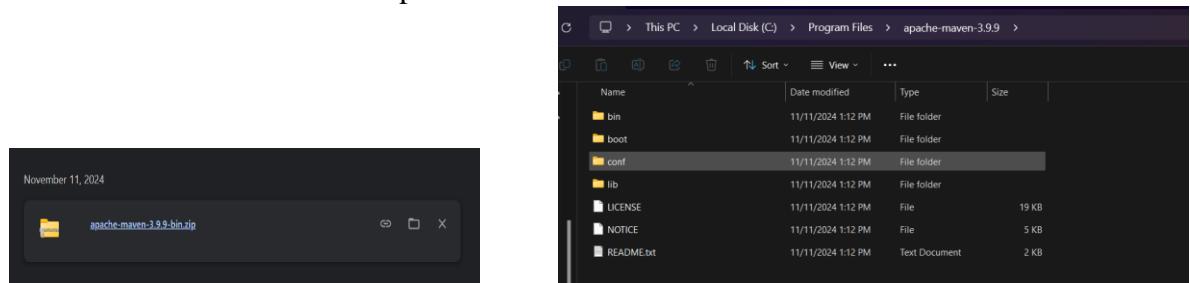
Name	Date modified	Type	Size
configuration	11/30/2024 6:46 PM	File folder	
dropins	9/5/2024 7:29 AM	File folder	
features	11/16/2024 8:24 AM	File folder	
p2	11/30/2024 6:47 PM	File folder	
plugins	11/16/2024 8:24 AM	File folder	
readme	11/13/2024 12:20 PM	File folder	
.eclipseproduct	11/13/2024 12:16 PM	ECLIPSEPRODUCT ...	1 KB
artifacts.xml	11/16/2024 8:24 AM	XML Source File	839 KB
eclipse.exe	11/13/2024 12:16 PM	Application	543 KB
eclipse.ini	11/16/2024 8:24 AM	Configuration sett...	1 KB
eclipssec.exe	11/13/2024 12:16 PM	Application	255 KB

MAVEN Installations

STEP 1 : Visit <https://maven.apache.org/download.cgi> and download apache-maven-3.9.9-bin.zip

The screenshot shows the Maven download page. On the left is a sidebar with links like Download, Use, Release Notes, Documentation, Maven Plugins, Maven Extensions, User Center, Plugins Developer Centre, Maven Repository Centre, Maven Developers Centre, Books and Resources, Security, Community, Project Overview, Project Roles, How to Contribute, Getting Help, Issue Management, Getting Maven Source, The Maven Team, Project Documentation, Maven Projects, Maven, and Archetypes. The main content area has sections for Java Development Kit (JDK), Memory, Disk, and Operating System requirements. Below these is a 'Files' section with four download options: Binary tar.gz archive (apache-maven-3.9.9-bin.tar.gz), Binary zip archive (apache-maven-3.9.9-bin.zip), Source tar.gz archive (apache-maven-3.9.9-src.tar.gz), and Source zip archive (apache-maven-3.9.9-src.zip). Each file entry includes a Link, Checksums, and Signature. At the bottom of the 'Files' section is a note about release notes and source code availability.

STEP 2 : Visit After Download of the zip file extract it



STEP 3 : Add this path to environment variables along with MAVEN_HOME variable

DriverData

C:\Windows\System32\Drivers\DriverData

JAVA_HOME

C:\Program Files\Java\jdk-21

MAVEN_HOME

C:\Program Files\apache-maven-3.9.9

NUMBER OF PROCESSORS

C:\Program Files\Git\cmd

C:\Program Files\apache-maven-3.9.9\bin

C:\Program Files\nodejs\

STEP 4 : Verify using mvn --version command

```
C:\Users\Ayush>mvn --version
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfc97d260186937)
Maven home: C:\Program Files\apache-maven-3.9.9
Java version: 21.0.4, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-21
Default locale: en_US, platform encoding: UTF-8
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"
```

JDK Installations

STEP 1 : Visit <https://www.oracle.com/java/technologies/javase/jdk21-archive-downloads.html> and download Windows x64 Installer

Download Type	Size	Link
Windows x64 Compressed Archive	185.84 MB	https://download.oracle.com/java/21/archive/jdk-21.0.4_windows-x64_bin.zip (sha256)
Windows x64 Installer	164.23 MB	https://download.oracle.com/java/21/archive/jdk-21.0.4_windows-x64_bin.exe (sha256)
Windows x64 msi Installer	162.97 MB	https://download.oracle.com/java/21/archive/jdk-21.0.4_windows-x64_bin.msi (sha256)

STEP 2 : Run the exe and perform the steps, and change install path if needed

STEP 3 : Add this path to environment variables along with **JAVA_HOME** variable

STEP 4 : Verify using **java --version** command

```
C:\Users\Ayush>java --version
java version "21.0.4" 2024-07-16 LTS
Java(TM) SE Runtime Environment (build 21.0.4+8-LTS-274)
Java HotSpot(TM) 64-Bit Server VM (build 21.0.4+8-LTS-274, mixed mode, sharing)
```

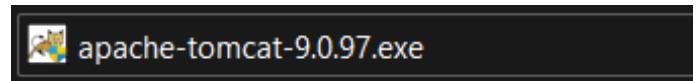
Tomcat Installations

STEP 1 : Visit <https://tomcat.apache.org/download-90.cgi> and download Windows service Installer

The screenshot shows the Apache Tomcat 9.0.97 download page. It includes sections for Release Integrity, Mirrors, and Binary Distributions. Under Binary Distributions, there are links for Core files (zip, tar.gz, 32-bit Windows zip, 64-bit Windows zip) and a link for Full documentation.

STEP 2 : Run the exe and perform the steps

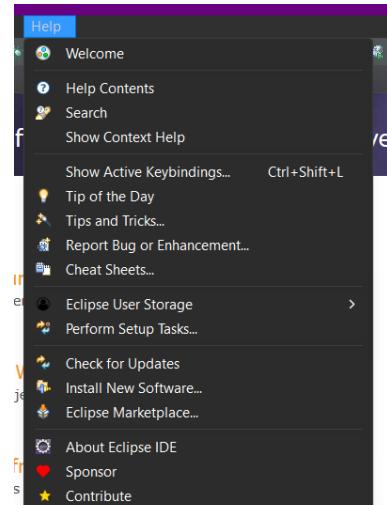
- Click on **Next and I Agree**
- Tick all options
- Set up **Port ,Username and Password**
- Click on **Next>**
- Select **JDK path & Destination path** and click **Finish**



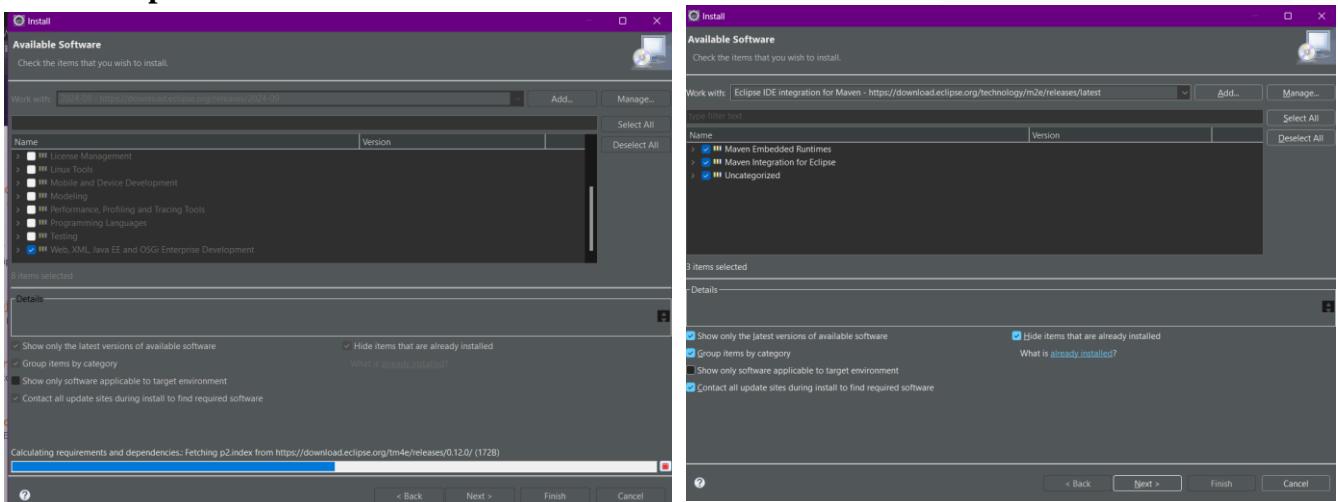
The screenshots show the Apache Tomcat Setup wizard. The first window is the Welcome screen with instructions to close other applications. The second window is the License Agreement screen, which displays the Apache License version 2.0. The third window is the Configuration Options screen, where users can set Tomcat basic configuration parameters like Server Shutdown Port (-1), HTTP/1.1 Connector Port (8090), Windows Service Name (Tomcat9), and User Name/Password for Tomcat Administrator Login.

Tomcat SET-UP

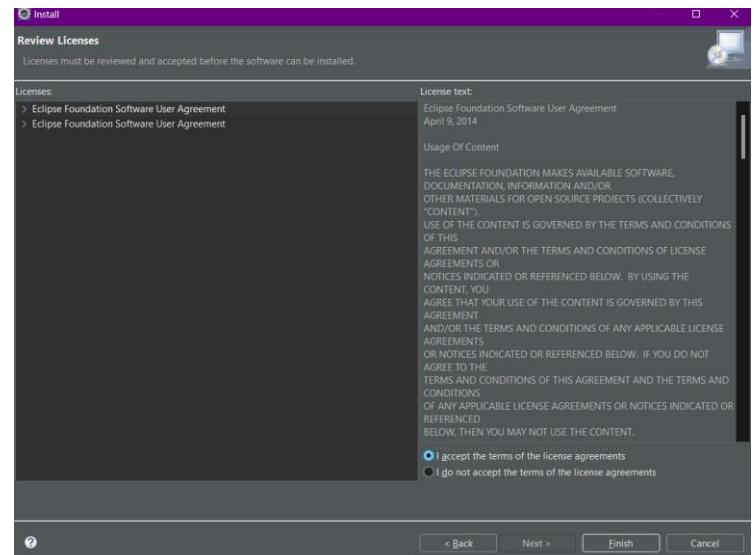
STEP 1 : Click on **Help** in Eclipse IDE and click on **Install New Software...**



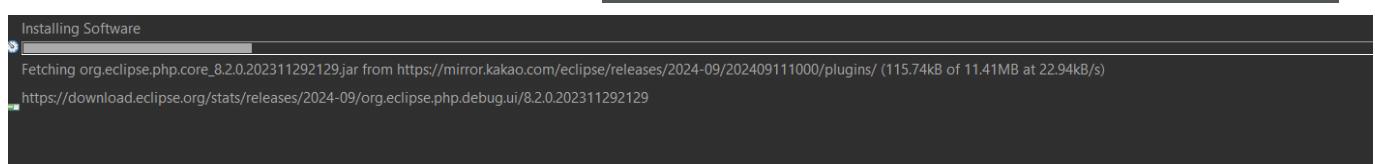
STEP 2 : Download Web XML, Java EE & OSGi Enterprises and Maven dependencies



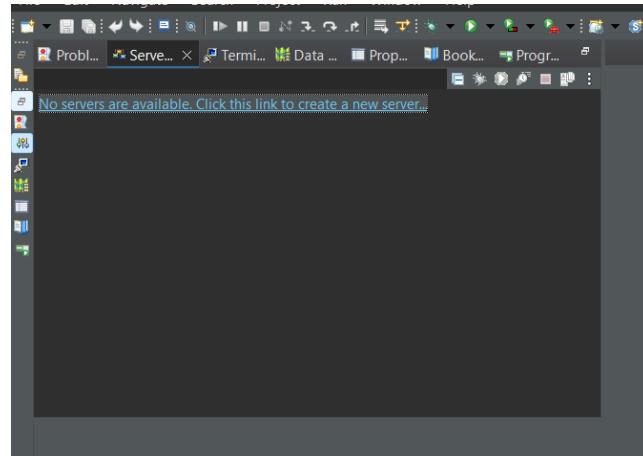
STEP 3 : Click on I agree



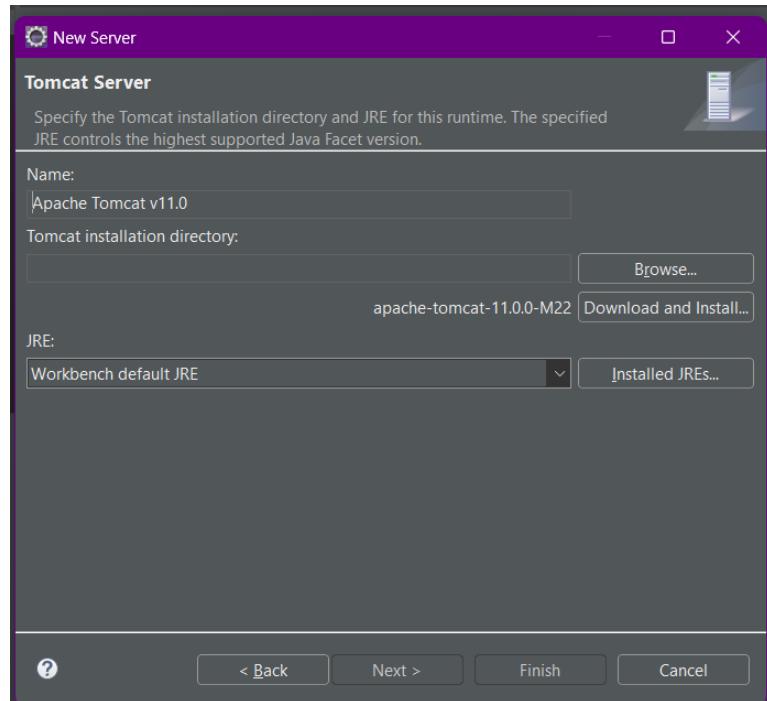
STEP 4 : Wait for downloads



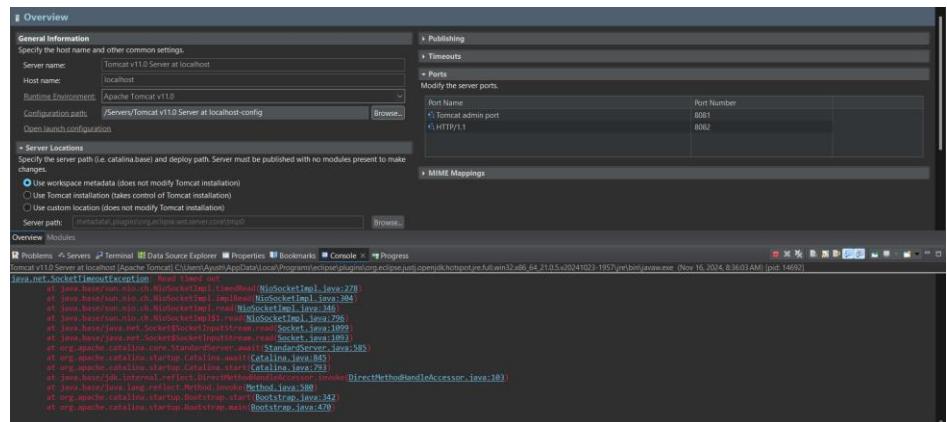
STEP 5 : Click on
Servers option



STEP 6 : Select appropriate
Tomcat version
and click on **Finish**



STEP 7 : Click Edit the ports
as 8081 & 8082



Conclusion : Thus we setup **Tomcat Server** and other software to launch our pages.

Experiment 4: GIT COMMANDS

AIM: To experience the GIT commands and GITHUB repository warehouse.

Introduction:

- ❖ **GIT Bash and GitHub**
 - Git is a popular version control system. It was created by Linus Torvalds in 2005.
 - It is designed to make it easier to have multiple versions of a code base, sometimes across multiple developers or teams
 - **Functions of Git**
 - Manage projects with Repositories
 - Clone a project to work on a local copy
 - Control and track changes with Staging and Committing
 - Branch and Merge to allow for work on different parts and versions of a project
 - Pull the latest version of the project to a local copy
 - Push local updates to the main project
- ❖ **GitHub Repositories**
 - A **GitHub repository** is a storage space in GitHub where developers host and manage their project files, code, and documentation. Repositories are essential for collaboration, version control, and sharing projects with others.
 - **Functions of GitHub Repos**
 - **Code Storage:** Store and organize project files.
 - **Version Control:** Track changes to files using Git.
 - **Collaboration:** Multiple contributors can work on the same project.
 - **Issue Tracking:** Report and manage bugs or feature requests.
 - **Branching:** Create separate branches for different features or fixes.

Procedure

Basic GIT Commands

cmd-1 : git version

```
C:\Users\Ayush\OneDrive\Desktop\newRepo>git version  
git version 2.46.2.windows.1
```

cmd-2 : git init

```
C:\Users\Ayush\OneDrive\Desktop\newRepo>git init  
Initialized empty Git repository in C:/Users/Ayush/OneDrive/Desktop/newRepo/.git/
```

cmd-3 : git config

```
C:\Users\Ayush\OneDrive\Desktop\newRepo>git config user.name  
RahZero0  
  
C:\Users\Ayush\OneDrive\Desktop\newRepo>git config user.email  
ayushpasham@gmail.com
```

cmd-4 : git status

```
C:\Users\Ayush\OneDrive\Desktop\newRepo>git status  
On branch master  
  
No commits yet  
  
nothing to commit (create/copy files and use "git add" to track)
```

cmd-5 : git add .

```
git add .
```

cmd-6 : git commit

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git commit -m "Added Hello World!"  
[master (root-commit) f198458] Added Hello World!  
 1 file changed, 1 insertion(+)  
 create mode 100644 a.txt
```

cmd-7 : git diff

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git diff  
diff --git a/a.txt b/a.txt  
index bc7774a..f2c8a5d 100644  
--- a/a.txt  
+++ b/a.txt  
@@ -1 +1,2 @@  
-hello world!  
\ No newline at end of file  
+hello world!  
+My world  
\ No newline at end of file
```

cmd-8 : git reset

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git reset  
Unstaged changes after reset:  
M      a.txt
```

cmd-9 : Branch Operations

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git branch work  
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git branch -a  
* master  
  work  
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git checkout work  
M      a.txt  
Switched to branch 'work'  
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git checkout -b new  
Switched to a new branch 'new'  
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git checkout -d new  
M      a.txt  
HEAD is now at f198458 Added Hello World!  
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git branch -d new  
Deleted branch new (was f198458).  
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git checkout master  
M      a.txt  
Switched to branch 'master'
```

cmd-10 : Logs

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git log  
commit f198458e94921b842de432ffe21d6f883f611b6 (HEAD -> master, work)  
Author: RahZero0 <ayushpasham@gmail.com>  
Date:   Sat Nov 16 12:33:00 2024 +0530  
  
        Added Hello World!  
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git log --oneline  
f198458 (HEAD -> master, work) Added Hello World!
```

cmd-11 : git revert

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git revert f198458e94921b842de432ffe21d6f883f611b6  
error: Your local changes to the following files would be overwritten by merge:  
      a.txt  
Please commit your changes or stash them before you merge.  
Aborting  
fatal: revert failed
```

cmd-12 : Pushing

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git branch -M main  
>> git remote add origin https://github.com/RahZero0/SE.git  
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git push -u origin main  
Enumerating objects: 3, done.  
Counting objects: 100% (3/3), done.  
Writing objects: 100% (3/3), 229 bytes | 38.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
To https://github.com/RahZero0/SE.git  
 * [new branch]      main -> main  
branch 'main' set up to track 'origin/main'.
```

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> ssh-keygen -t rsa -C "ayushpasham@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (C:/Users/Ayush/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:/Users/Ayush/.ssh/id_rsa
Your public key has been saved in C:/Users/Ayush/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:OKGD0bICycTB0dsejQFvdiHDwUIMFuwh1LBVMBsOA ayushpasham@gmail.com
The key's randomart image is:
+---[RSA 3072]---+
| .+ +...*B... |
| .O+ =-OO,O..O |
| OOO ++-O..E... . |
| +..O..O+*+.+ |
| ..... t+S+ . |
| . .O+ |
| O. |
| . |
+---[SHA256]---+
```

cmd-13 : SSH Key

The screenshot shows the GitHub 'SSH keys' page. It displays a single SSH key entry:

- SAMPLE**: SHA256:OKGD0bICycTB0dsejQFvdiHDwUIMFuwh1LBVMBsOA
- Added on Nov 16, 2024
- Never used — Read/Write

A green button at the top right says 'New SSH key'.

cmd-14 : git clone

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git clone https://github.com/RahZero0/SE.git
Cloning into 'SE'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
Checking connectivity... done.
```

cmd-15 : git fetch

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git fetch https://github.com/RahZero0/SE.git
From https://github.com/RahZero0/SE
 * branch HEAD      -> FETCH_HEAD
```

cmd-16 : git pull

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git pull origin
Already up to date.
```

cmd-17 : Forking

The screenshot shows the GitHub repository 'Spoon-Knife'. The repository is public and was forked from 'octocat/Spoon-Knife'. The main branch is 'main'. The repository has 3 commits and 0 forks. The README file contains the following text:

```
Well hello there!
This repository is meant to provide an example for forking a repository on GitHub.
Creating a fork is producing a personal copy of someone else's project. Forks act as a sort of bridge between the original repository and your personal copy. You can submit Pull Requests to help make other people's projects better by offering your changes up to the original project. Forking is at the core of social coding at GitHub.
After forking this repository, you can make some changes to the project, and submit a Pull Request as practice.
For some more information on how to fork a repository, check out our guide, "Forking Projects". Thanks! ❤
```

cmd-17.a : Adding Branch

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> git clone https://github.com/RahZero0/Spoon-Knife.git
Cloning into 'Spoon-Knife'...
remote: Enumerating objects: 10, done.
remote: Total 10 (delta 0), reused 0 (delta 0), pack-reused 10 (from 1)
Receiving objects: 100% (10/10), done.
Resolving deltas: 100% (1/1), done.
PS C:\Users\Ayush\OneDrive\Desktop\newRepo> cd .\Spoon-Knife\
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> code .
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git checkout -b 22BD1A6744
Switched to a new branch '22BD1A6744'
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> code .
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git add .
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife>
```

cmd-17.b : Pushing changes back to forked repository

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git commit -m "I added changes."
[22BD1A6744 00280eb] I added changes.
 1 file changed, 1 insertion(+)
 create mode 100644 22BD1A6744.txt
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git push origin 22BD1A6744
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 382 bytes | 127.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for '22BD1A6744' on GitHub by visiting:
remote:     https://github.com/RahZero0/Spoon-Knife/pull/new/22BD1A6744
remote:
To https://github.com/RahZero0/Spoon-Knife.git
 * [new branch]      22BD1A6744 -> 22BD1A6744
```

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git remote add upstream git@github.com:octocat/Spoon-Knife.git
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git fetch upstream
The authenticity of host 'github.com (20.207.73.82)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpbZisF/zLDA0zPMsvHdkr4UvCoqu.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 2), reused 6 (delta 2), pack-reused 0 (from 0)
Unpacking objects: 100% (6/6), 597 bytes | 12.00 KiB/s, done.
From github.com:octocat/Spoon-Knife
 * [new branch]      change-the-title -> upstream/change-the-title
 * [new branch]      main           -> upstream/main
 * [new branch]      test-branch    -> upstream/test-branch
```

cmd-18 : Pull from upstream repository

```
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git remote add upstream git@github.com:octocat/Spoon-Knife.git
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git fetch upstream
The authenticity of host 'github.com (20.207.73.82)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpbZisF/zLDA0zPMsvHdkr4UvCoqu.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 2), reused 6 (delta 2), pack-reused 0 (from 0)
Unpacking objects: 100% (6/6), 597 bytes | 12.00 KiB/s, done.
From github.com:octocat/Spoon-Knife
 * [new branch]      change-the-title -> upstream/change-the-title
 * [new branch]      main           -> upstream/main
 * [new branch]      test-branch    -> upstream/test-branch
PS C:\Users\Ayush\OneDrive\Desktop\newRepo\Spoon-Knife> git push origin 22BD1A6744
Everything up-to-date
```

Conclusion : Thus we were able to understand and perform GIT commands.

Experiment 5: CREATING MAVEN JAVA AND MAVEN WEB PROJECT USING ECLIPSE AND PUSH THEM TO GITHUB + JENKINS INSTALLATIONS

AIM: To create **Maven JAVA** and **Maven WEB** projects using **Eclipse** and push them to **GitHub**

Introduction:

❖ MAVEN

- Apache Maven is a powerful **build automation and dependency management tool** for Java-based projects. It simplifies project setup, management, and deployment using a standard directory structure and configuration file called pom.xml.
- **Functions of Maven**
 - **Project Management with POM:** Centralized configuration using the pom.xml file.
 - **Dependency Management:** Automatically downloads and resolves project dependencies.
 - **Build Automation:** Compiles, tests, and packages projects efficiently.
 - **Plugin Support:** Extends functionality through a wide range of plugins.
 - **Standard Project Structure:** Provides a consistent and organized directory layout.
 - **Integration with Tools:** Works seamlessly with IDEs, version control systems, and CI/CD tools.

❖ ECLIPSE IDE

- Eclipse IDE is a widely used **integrated development environment (IDE)** designed primarily for Java development. It supports other programming languages through plugins and provides tools for coding, debugging, and testing applications efficiently.
- **Functions of Eclipse IDE**
 - **Code Editing:** Offers syntax highlighting, code completion, and real-time error detection.
 - **Project Management:** Organizes files and resources for single or multi-module projects.
 - **Debugging Tools:** Allows developers to set breakpoints, inspect variables, and debug code.
 - **Plugin Support:** Extends functionality for languages like Python, C++, and web development.
 - **Version Control Integration:** Works seamlessly with Git and other version control systems.
 - **Build Automation:** Supports tools like Maven and Gradle for building projects.

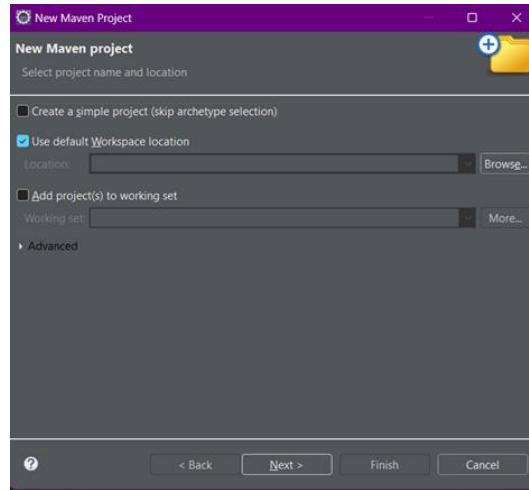
❖ JENKINS

- Jenkins is an open-source **automation server** used for building, testing, and deploying software projects. It supports **Continuous Integration (CI)** and **Continuous Delivery (CD)**, helping developers integrate changes and deliver updates efficiently.
- **Functions of Jenkins**
 - **Continuous Integration:** Automatically builds and tests code whenever changes are committed.
 - **Integration:** Automatically builds and tests code whenever changes are committed.
 - **Pipeline Automation:** Manages complex build and deployment workflows using Jenkins Pipelines.
 - **Plugin Support:** Extends functionality with a vast library of plugins for tools like Git, Maven, and Docker.
 - **Build Scheduling:** Allows builds to be triggered manually, on a schedule, or based on code changes.
 - **Integration with Tools:** Works with version control systems, testing frameworks, and deployment platforms.
 - **Monitoring and Reporting:** Provides detailed build logs, test reports, and error tracking.

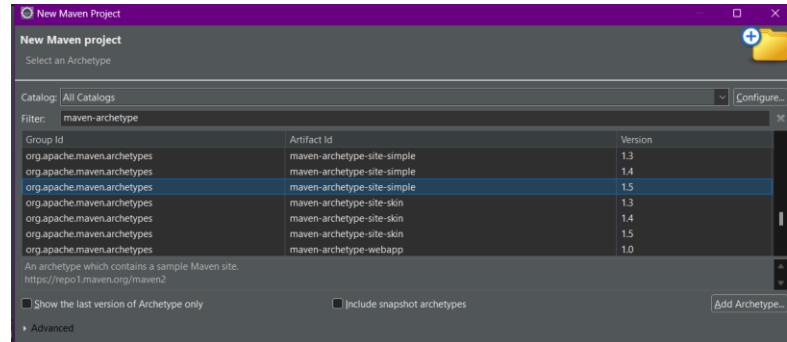
Procedure

MAVEN JAVA

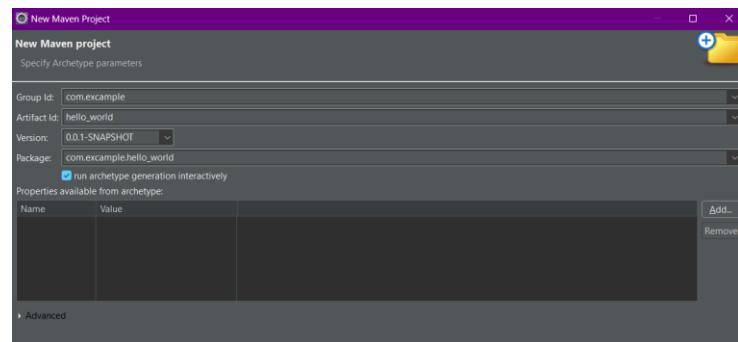
STEP 1 : Open Eclipse IDE and create new Maven Project.



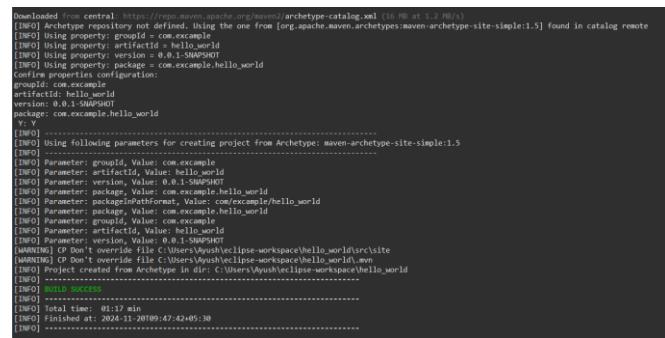
STEP 2 : Select
org.apache.maven.archetypes ->
maven-archetype-quickstart 1.5
as Maven Architect Type
and click on **Next**



STEP 3 : Define Project MetaData and click FINISH



STEP 4 : In the Console All the Artifacts have been grouped



STEP 5 : Maven Project has been successfully created

- ```
└── Includes:
 └── src/main/java (for Java code)
 └── src/test/java (for test code)
 └── pom.xml (Maven config file)
```

```
1 package hello_world_demo;
2
3 public class WelcomClass {
4 public String HelloWorld() {
5 return "Hello World";
6 }
7 }
8 }
```

**STEP 6 :** Update Project Settings to ensure the dependencies are up to date

The screenshot shows the Eclipse IDE interface with the Java perspective. A right-click context menu is open over a Java class named 'HelloWorld'. The menu path is: `HelloWorld > Team > Maven`. The 'Maven' submenu is expanded, showing options like 'Add Dependency', 'Add Plugin...', 'New Maven Module Project...', 'Download Javadoc...', 'Download Sources...', 'Update Project...', 'Select Maven Profiles...', 'Disable Workspace Resolution', 'Disable Maven Nature', and 'Assign Working Sets...'. At the bottom of the submenu, there is a note: `http://repo.maven.apache.org/maven2/archetype-portal not defined. Using the one from [org.apache.rampart-com.example]`.

## **STEP 7 : Build and Run Maven Project**

- └── Right-click on App.java -> Run As -> Maven Clean
  - └── Right-click on App.java -> Run As -> Maven Install
  - └── Right-click on App.java -> Run As -> Maven Test
  - └── Right-click on App.java -> Run As -> Maven Build

## MAVEN INSTALL

The screenshot shows the IntelliJ IDEA context menu for a Maven project. The menu items are:

- References
- Declarations
- Coverage As
- Run As
- Debug As
- Profile As
- Restore from Local History...
- Web Services
- Team
- Compare With
- Replace With
- GitHub
- Configure
- Validate

A secondary context menu is open under the 'Run As' item, listing the following options:

- 1 Run on Server
- 2 Maven build...
- 3 Maven clean
- 4 Maven generate-sources
- 5 Maven install
- 6 Maven test
- 7 Maven verify

At the bottom of this submenu, there is a separator line followed by the text 'Run Configurations...' and the option '8 hello\_world\_demo (Maven Build)'.

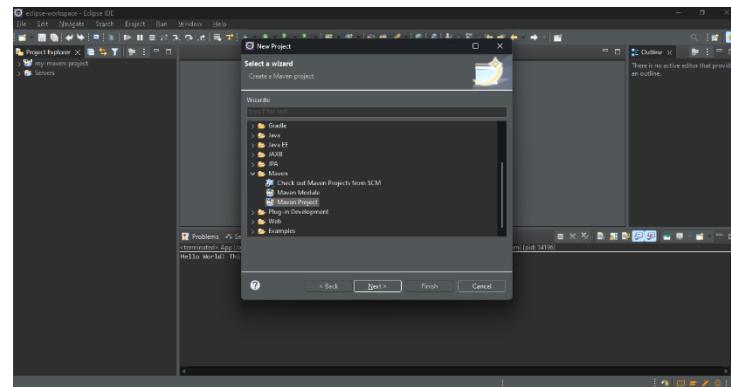
MAVEN TEST

```
[INFO] [INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ hello-world-deps ---
[INFO] [INFO] Deleting directory C:\Users\jason\IdeaProjects\hello-world-deps\target
[INFO] [INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ hello-world-deps ---
[INFO] [INFO] Using filtering (if any) on provided resources
[INFO] [INFO] Copying 0 resources from src/main/resources to target/classes
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testClasses
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testResources
[INFO] [INFO] Copying 0 resources from src/main/resources to target/classes
[INFO] [INFO] Nothing to compile - all classes are up to date.
[INFO] [INFO] --- maven-resources-plugin:2.6:processResources (default-processResources) @ hello-world-deps ---
[INFO] [INFO] Using filtering (if any) to copy 0 filtered resources, i.e. build is platform dependent!
[INFO] [INFO] Copying 0 resources from src/main/resources to target/classes
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testClasses
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testResources
[INFO] [INFO] Copying 0 resources from src/main/resources to target/classes
[INFO] [INFO] Nothing to compile - all classes are up to date.
[INFO] [INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ hello-world-deps ---
[INFO] [INFO] Nothing to compile - all classes are up to date.
[INFO] [INFO] --- maven-resources-plugin:2.6:resources (default-testResources) @ hello-world-deps ---
[INFO] [INFO] Using filtering (if any) on provided resources
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testClasses
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testResources
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testClasses
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testResources
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testClasses
[INFO] [INFO] Copying 0 resources from src/main/resources to target/testResources
[INFO] [INFO] Total time: 2.051 s
[INFO] [INFO] Finished at: 2024-01-20T10:00:49+05:30
[INFO] [INFO]
```

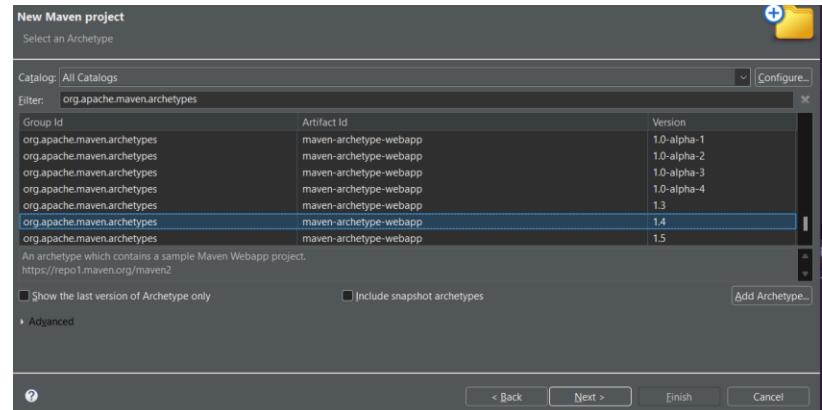
### **STEP 8 : Output**

```
 3 public class Welcomclass {
 4 public static void main(String[] args) {
 5 System.out.println(Helloworld());
 6 }
 7 public static String Helloworld() {
 8 return "This is student Ayush Reddy, 22BD1A6744";
 9 }
10 }
```

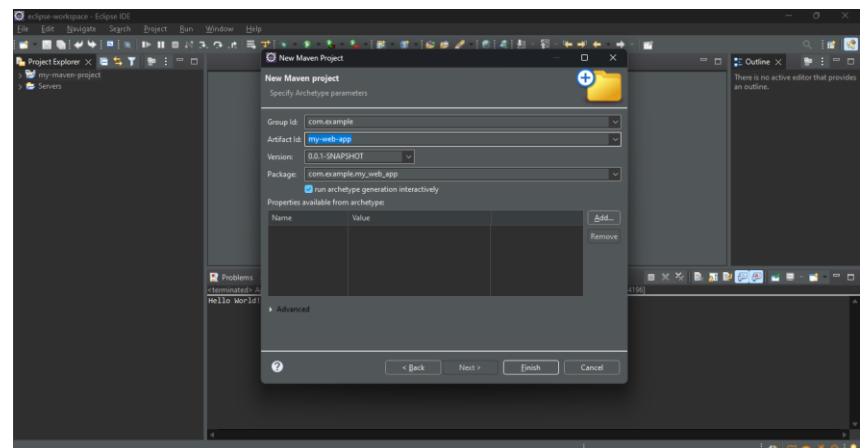
## MAVEN WEB



**STEP 1 :** Open Eclipse IDE and create new **Maven Project**.



**STEP 2 :** Select an archetype  
**org.apache.maven.archetypes -> maven-archetype-webapp 1.4**



**STEP 3 :** Define Project MetaData and click **FINISH**

```
Downloaded from central: https://repo.maven.apache.org/maven2/archetype-catalog.xml (16 kB at 7.3 MB/s)
[INFO] Archetype repository not defined. Using the one from [org.apache.maven.archetypes:maven-archetype-webapp:1.5] found in catalog remote
[INFO] Using property: groupId = com.example
[INFO] Using property: artifactId = my-web-app
[INFO] Using property: version = 0.0.1-SNAPSHOT
[INFO] Using property: package = com.example.my_web_app
configure properties configuration:
groupId: com.example
artifactId: my-web-app
version: 0.0.1-SNAPSHOT
package: com.example.my_web_app
Y_N
[INFO] -----
[INFO] Using following parameters for creating project from Archetype: maven-archetype-webapp:1.4
[INFO] -----
[INFO] Parameter: groupId, Value: com.example
[INFO] Parameter: artifactId, Value: my-web-app
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] Parameter: package, Value: com.example.my_web_app
[INFO] Parameter: packageInPathFormat, Value: com/example/my_web_app
[INFO] Parameter: package, Value: com.example.my_web_app
[INFO] Parameter: groupId, Value: com.example
[INFO] Parameter: artifactId, Value: my-web-app
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] -----
Project created from Archetype in dir: C:\Users\Myush\Eclipse-workspace\my-web-app
[INFO] -----
[INFO] success
[INFO] -----
[INFO] total time: 15.269 s
[INFO] +finished at: 2024-11-20T11:29:25+05:30
[INFO] -----
```

**STEP 4 :** Open Enter Y in Console.

**STEP 5 :** Add necessary dependencies in **pom.xml**

```
hello_world_demo/pom.xml WelcomClassJava my-web-app/pom.xml x
http://maven.apache.org/xsd/maven-4.0.xsd (xsd:schemaLocation with catalog)
1 <xml version="1.0" encoding="UTF-8">
2
3<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5 <modelVersion>4.0.0</modelVersion>
6
7 <groupId>com.example</groupId>
8 <artifactId>my-web-app</artifactId>
9 <version>0.0.1-SNAPSHOT</version>
10 <packaging>war</packaging>
11
12 <name>my-web-app Maven Webapp</name>
13 <!-- FIXME change it to the project's website -->
14 <url>http://www.example.com</url>
15
16<properties>
17 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
18 <maven.compiler.source>1.7</maven.compiler.source>
19 <maven.compiler.target>1.7</maven.compiler.target>
20 </properties>
21
22<dependencies>
23 <dependency>
24 <groupId>junit</groupId>
25 <artifactId>junit</artifactId>
26
27</dependencies>
```

**STEP 6 : Go to [mvnrepository.com](http://mvnrepository.com) and Search Java Servlet API**

| Java Servlet API                                                                                                                                                                              |                                                               | Search                                                    |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------|-----------------------------------------------------------|
| <b>Found 112274 results</b>                                                                                                                                                                   |                                                               |                                                           |
| Sort: <b>relevance</b>   popular   newest                                                                                                                                                     |                                                               |                                                           |
|  1. <a href="#">Java Servlet API</a>                                                                         | <a href="#">javaservlet</a> » <a href="#">javaservlet-api</a> | 19,563 usages<br><a href="#">GPL</a> <a href="#">CDCL</a> |
| Java Servlet is the foundation web specification in the Java Enterprise Platform. Developers can build web applications using the Servlet API to interact with the request/response workflow. |                                                               |                                                           |
| Last Release on Apr 20, 2018                                                                                                                                                                  |                                                               |                                                           |
| <b>Relocated</b> – <a href="#">jakarta.servlet</a> » <a href="#">jakarta.servlet-api</a>                                                                                                      |                                                               |                                                           |
|  2. <a href="#">JavaServlet(TM) Specification</a>                                                            | <a href="#">javaservlet</a> » <a href="#">servlet-api</a>     | 13,424 usages<br><a href="#">GPL</a> <a href="#">CDCL</a> |

**STEP 7 :** Click on it and Select **Latest Version** and Copy the dependency Code and paste it in **MavenWeb's pom.xml**

Maven Repository: javax.servlet-api

Java Server API 4.0.1

Java Server API is the foundation web specification in the Java Enterprise Platform. Developers can build web applications using the Servlet API to interact with the request/response workflow.

Popular Categories

Testing Frameworks & Tools  
Android Packages  
Logging Frameworks  
JVM Languages  
Java Specifications  
JSON Libraries  
Core Utilities  
Mocking  
Annotation Libraries  
Web Assets  
HTTP Clients  
Language Runtime  
Logging Bridges  
Dependency Injection  
XML Processing  
Web Frameworks

License: CDDL, GPL\_2.0

Categories: Java Specifications

Tags: standard, servlet, java, api, specs

Organization: GlassFish Community

HomePage: <https://java.github.io/servlet-spec/>

Date: Apr 20, 2018

Files: pom (15 KB) | jar (93 KB) | View All

Repositories: Central, java.net, Staging, MuleSoft, SoftwareMatters, WSO2, Public

Ranking: #16 in MavenRepository (See Top Artifacts)  
#1 in Java Specifications

Used by: 19,562 artifacts

Maven | Gradle (short) | Gradle (xml) | SBT | Ivy | Grape | Leiningen | Builder

```
<!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
<dependency>
 <groupId>javax.servlet</groupId>
 <artifactId>javax.servlet-api</artifactId>
 <version>4.0.1</version>
 <scope>provided</scope>
</dependency>
```

Include comment with link to detail

ORACLE

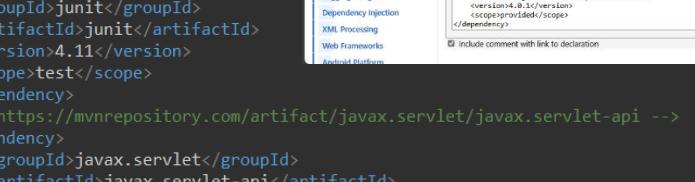
Sharpen companywide decision-making with Oracle Fusion Cloud Applications

Learn more



Indexed Repositories (2843)

Central



The screenshot shows the Android Studio interface with the Maven tab selected. A dependency tree is displayed, starting with the junit dependency and branching down to the javax.servlet-api dependency. The javax.servlet-api dependency is resolved from the mvnrepository.com repository. The 'include comment with link to declaration' checkbox is checked.

```
<dependencies>
 <dependency>
 <groupId>junit</groupId>
 <artifactId>junit</artifactId>
 <version>4.11</version>
 <scope>test</scope>
 </dependency>
 <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
 <dependency>
 <groupId>javax.servlet</groupId>
 <artifactId>javax.servlet-api</artifactId>
 <version>4.0.1</version>
 <scope>provided</scope>
 </dependency>
</dependencies>

<build>
 <finalName>my-web-app</finalName>

```

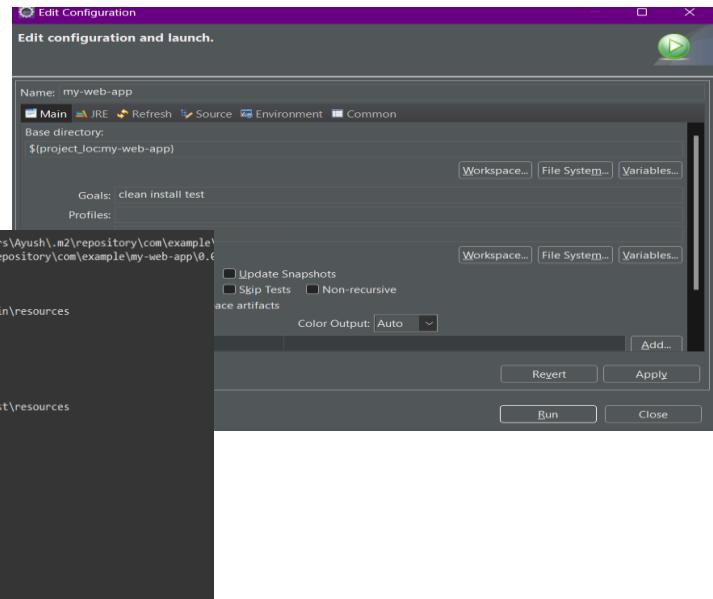
#### **STEP 8 : Perform**

## MAVEN CLEAN:

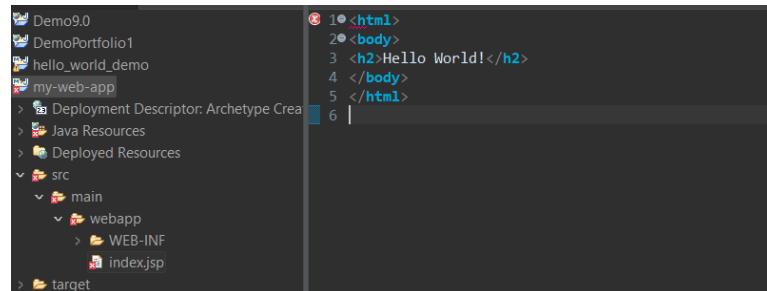
## MAVEN INSTALL:

```
[terminated-> C:\Users\Ayush\AppData\Local\Programs\eclipse\plugins\org.eclipse.just.java.hotspot\jre.full.win32.x86_64_21.0.5
[INFO] Scanning for projects...
[INFO]
[INFO] ----- < com.example:my-web-app > -----
[INFO] Building my-web-app Maven Webapp 0.0.1-SNAPSHOT
[INFO] from pom.xml
[INFO] ----- [war] -----
[INFO]
[INFO] --- clean:3.1.0:clean (default-clean) @ my-web-app ---
[INFO] Deleting C:\Users\Ayush\eclipse-workspace\my-web-app\target
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 1.179 s
[INFO] Finished at: 2024-11-20T11:30:07+05:30
[INFO] -----
```

## **STEP 9 : Do Maven Build and Give clean install test in Goals**



**STEP 10 :** Start the Web App Right Click on index.jsp and Run on Server



### **STEP 11 : Output**



## PUSHING TO GITHUB

### STEP 1 : Pushing Maven Project

**Maven-Project** Public

main 1 Branch 0 Tags

RahZero0 Maven Project ccable3 3 minutes ago 1 Commit

- settings Maven Project 3 minutes ago
- src/main/java/hello\_world\_demo Maven Project 3 minutes ago
- target Maven Project 3 minutes ago
- classpath Maven Project 3 minutes ago
- .project Maven Project 3 minutes ago
- pom.xml Maven Project 3 minutes ago

**README**

Add a README

Help people interested in this repository understand your project by adding a README.

```

[MINGW64:/c/Users/Ayush/eclipse-workspace/hello_world_demo] Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/hello_world_demo (master)
$ git init
Initialized empty Git repository in C:/Users/Ayush/eclipse-workspace/hello_world_demo/.git/
Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/hello_world_demo (master)
$ git add .
create mode 100644 target/classes/META-INF/maven/softwareEngineering/hello_world_demo/pom.properties
create mode 100644 target/classes/META-INF/maven/softwareEngineering/hello_world_demo/pom.xml
create mode 100644 target/classes/hello_world_demo/WelcomClass.class
create mode 100644 target/hello_world_demo-0.1-SNAPSHOT.jar
create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/createdFiles.lst
create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/inputFiles.lst
create mode 100644 target/maven-status/maven-compiler-plugin/testCompile/default-testCompile/createdFiles.lst
create mode 100644 target/maven-status/maven-compiler-plugin/testCompile/default-testCompile/inputFiles.lst

Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/hello_world_demo (master)
$ git branch -M main

Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/hello_world_demo (main)
$ git remote add origin https://github.com/RahZero0/Maven-Project.git
Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/hello_world_demo (main)
$ git push -u origin main
Enumerating objects: 35, done.
Counting objects: 100% (35/35), done.
Delta compression using up to 8 threads.
Compressing objects: 100% (22/22), done.
Writing objects: 100% (35/35), 4.80 KB | 144.00 KiB/s, done.
Total 35 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/RahZero0/Maven-Project.git
 * [new branch] main > main
branch 'main' set up to track 'origin/main'.

Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/hello_world_demo (main)

```

### STEP 2 : Pushing Maven Web Project

**Maven-Web-Project** Public

main 1 Branch 0 Tags

RahZero0 First Commit ea7448d 3 minutes ago 1 Commit

- .settings First Commit 3 minutes ago
- src/main/webapp First Commit 3 minutes ago
- target First Commit 3 minutes ago
- classpath First Commit 3 minutes ago
- .project First Commit 3 minutes ago
- pom.xml First Commit 3 minutes ago

**README**

Add a README

Help people interested in this repository understand your project by adding a README.

```

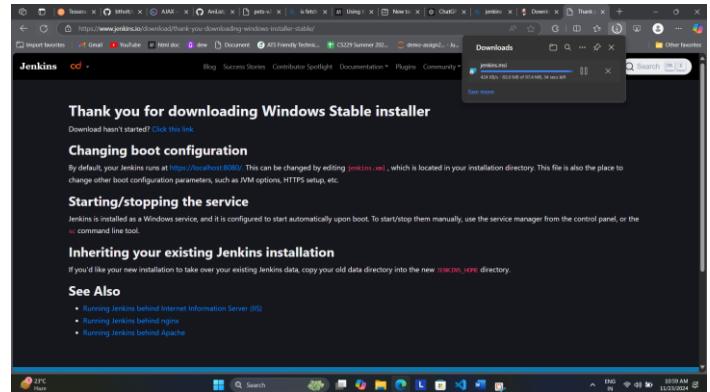
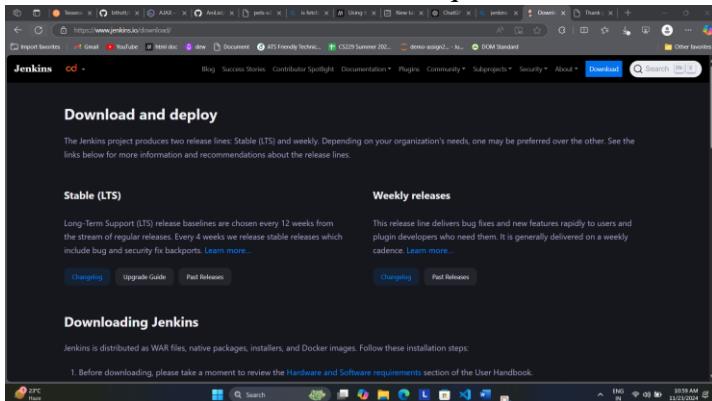
[MINGW64:/c/Users/Ayush/eclipse-workspace/my-web-app] Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/my-web-app (master)
$ git init
Initialized empty Git repository in C:/Users/Ayush/eclipse-workspace/my-web-app/.git/
Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/my-web-app (master)
$ git add .
warning: in the working copy of 'src/main/webapp/WEB-INF/web.xml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/main/webapp/index.jsp', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'target/my-web-app/WEB-INF/web.xml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'target/my-web-app/index.jsp', LF will be replaced by CRLF the next time Git touches it
Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/my-web-app (master)
$ git commit -m "First Commit"
[master (root-commit) ea7448d] First Commit
 21 files changed, 314 insertions(+)
 create mode 100644 .classpath
 create mode 100644 .project
 create mode 100644 .settings/.jsdtscope
 create mode 100644 .settings/org.eclipse.core.resources_prefs
 create mode 100644 .settings/org.eclipse.jdt.core_prefs
Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/my-web-app (master)
$ git branch -M main

Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/my-web-app (main)
$ git remote add origin https://github.com/RahZero0/Maven-Web-Project.git
Ayush@DESKTOP-NEGTAQG: MINGW64 ~/eclipse-workspace/my-web-app (main)
$ git push -u origin main
Enumerating objects: 33, done.
Counting objects: 100% (33/33), done.
Delta compression using up to 8 threads.
Compressing objects: 100% (22/22), done.
Writing objects: 100% (33/33), 6.06 KB | 194.00 KiB/s, done.
Total 33 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1).
To https://github.com/RahZero0/Maven-Web-Project.git
 * [new branch] main > main
branch 'main' set up to track 'origin/main'.

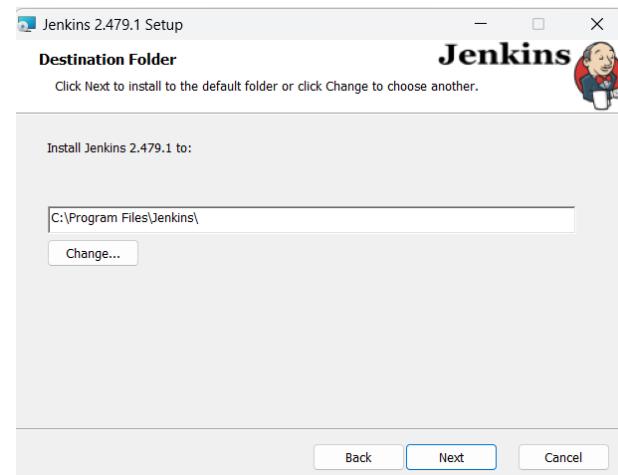
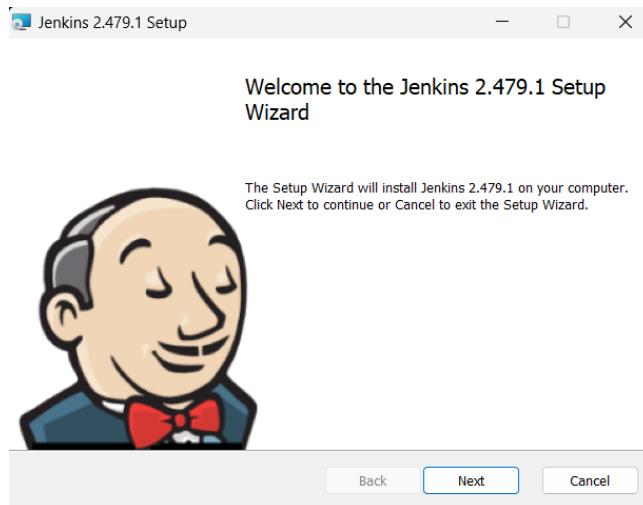
```

# JENKINS INSTALLATION

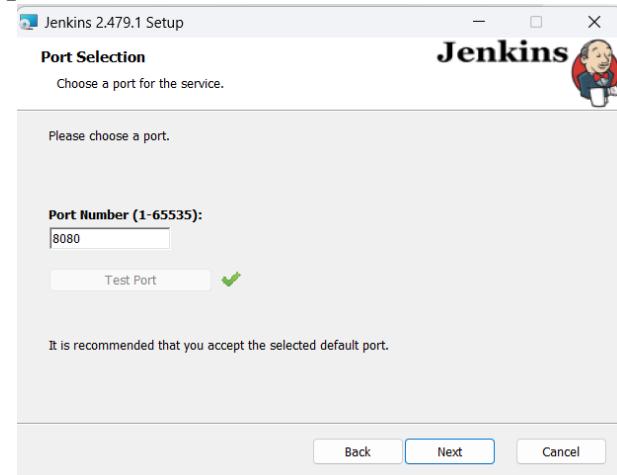
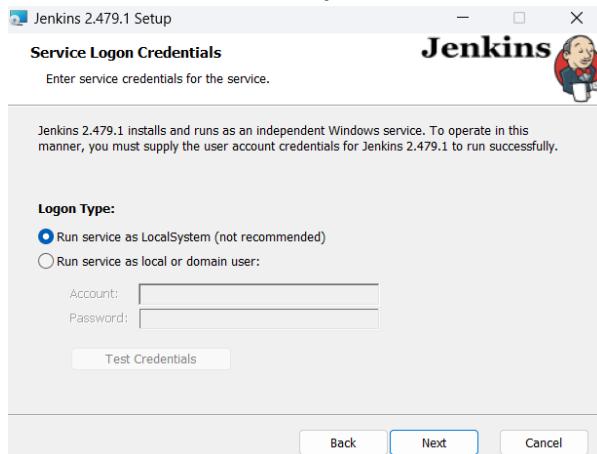
## STEP 1 : Download the required installation file from the official Jenkins website



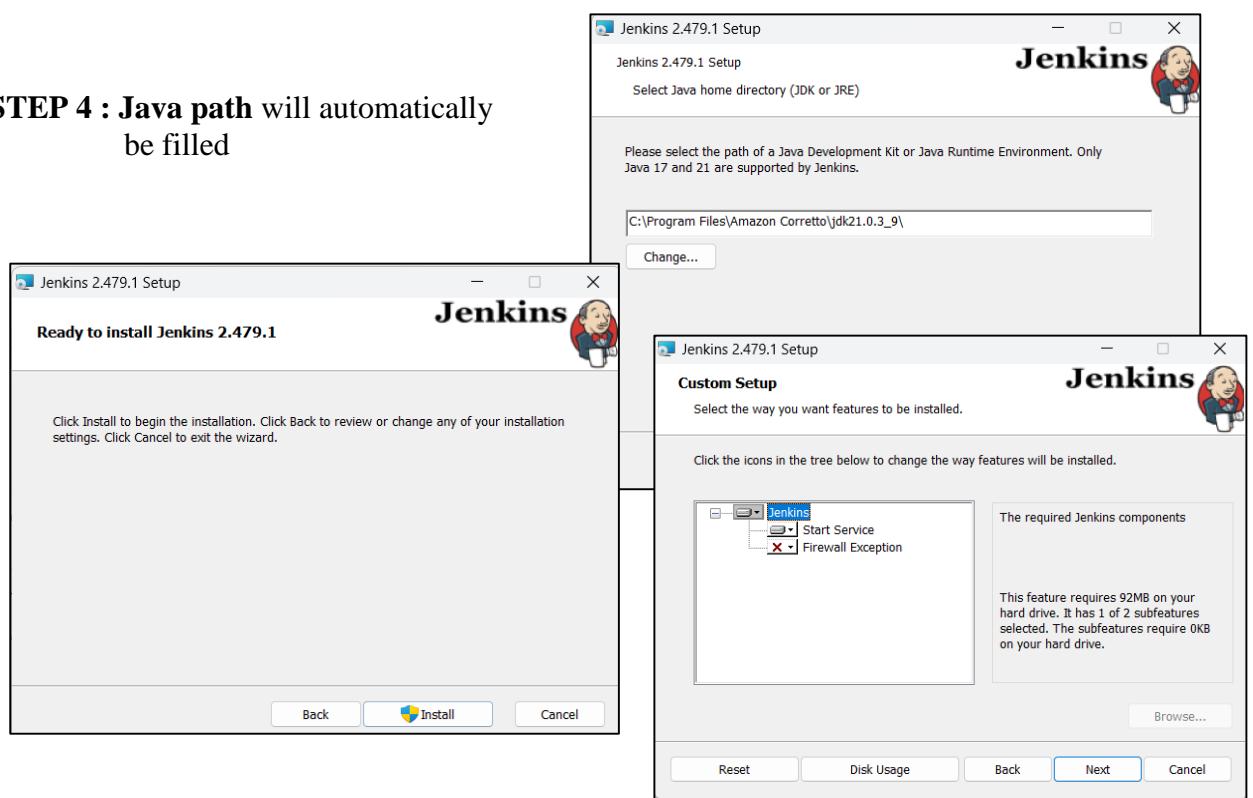
## STEP 2 : Setting up Jenkins using the downloaded exe



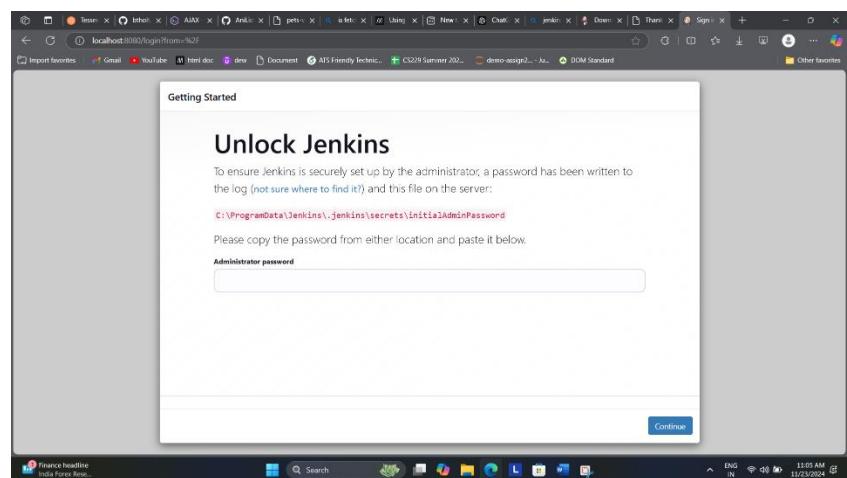
## STEP 3 : Select Local System and check for active ports



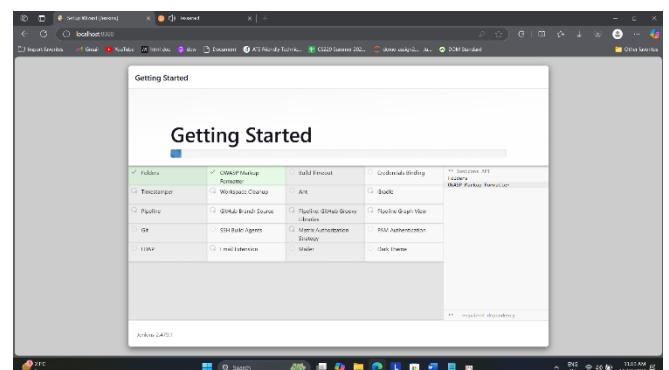
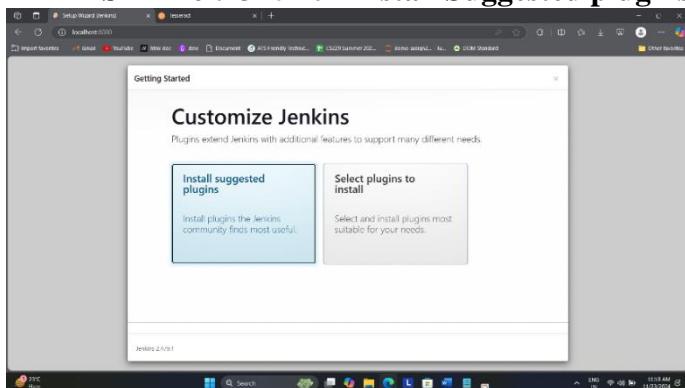
#### **STEP 4 : Java path will automatically be filled**



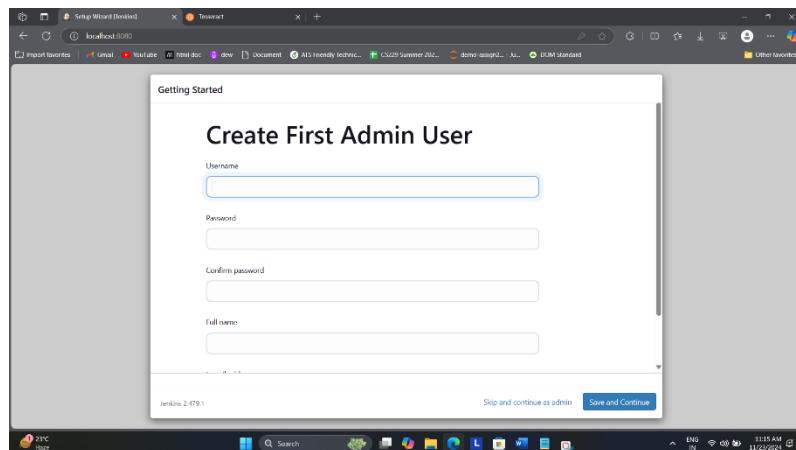
#### **STEP 5 : Open LocalHost 8080 and enter the **administrative password** which is in mentioned file location**



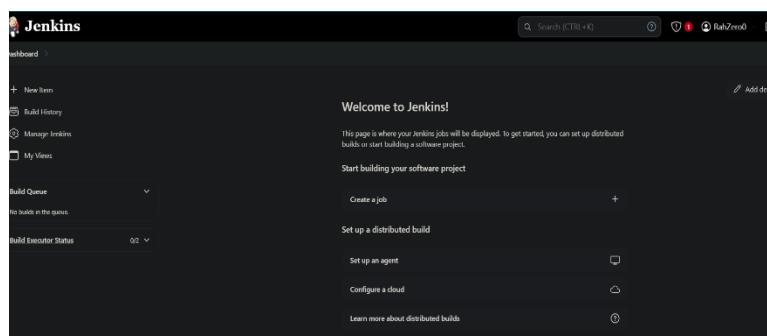
#### **STEP 6 : Click on Install Suggested plugins**



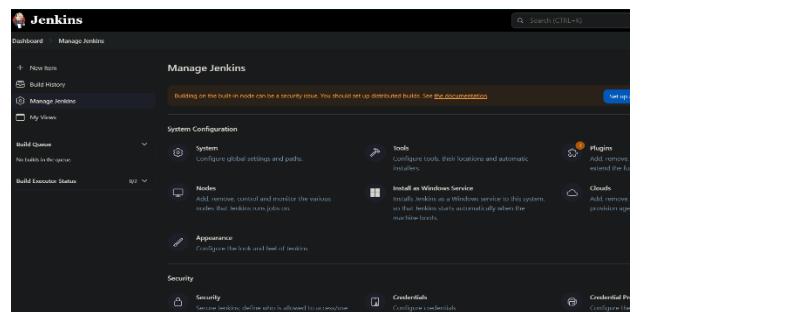
## STEP 7 : Enter credentials and enter



## STEP 8 : Your Jenkins Server has been installed , click on Manage Jenkins



## STEP 9 : Click on System



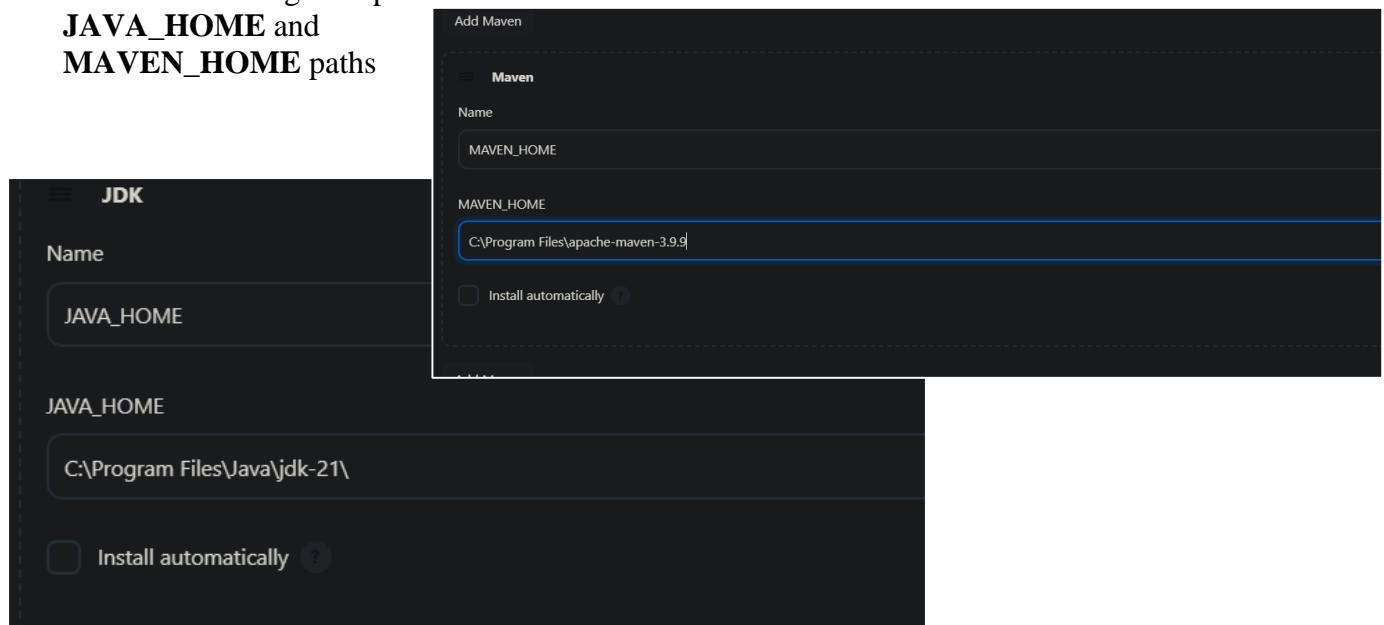
## STEP 10 : Install given plugins

**Download progress**

Category	Plugin Name	Status
Preparation	Javadoc	Success
	JSch dependency	Success
	Maven Integration	Success
	Commons Compress API	Success
	Pipeline Utility Steps	Success
	Copy Artifact	Success
	JavaMail API	Success
	SSH server	Success
	Deploy to container	Success
	Parameterized Trigger	Success
jQuery	Success	
Build Pipeline	Success	
Loading plugin extensions	Success	

→ Go back to the top page  
(you can start using the installed plugins right away)

**STEP 11 : Setting Set up JAVA\_HOME and MAVEN\_HOME paths**



**Conclusion :** Thus we were able to create a **MAVEN JAVA & MAVEN WEB** project and push it to **GITHUB** and successfully installed **JENKINS**.

## Experiment 6: Building the CI/CD pipeline using Jenkins for the project in the previous experiment

**AIM:** To create **CI/CD pipeline using Jenkins** for **Maven JAVA** and **Maven WEB** projects.

### Introduction:

#### ❖ MAVEN

- Apache Maven is a powerful **build automation and dependency management tool** for Java-based projects. It simplifies project setup, management, and deployment using a standard directory structure and configuration file called pom.xml.
- **Functions of Maven**
  - **Project Management with POM:** Centralized configuration using the pom.xml file.
  - **Build Automation:** Compiles, tests, and packages projects efficiently.
  - **Plugin Support:** Extends functionality through a wide range of plugins.

#### ❖ CI/CD PIPELINE

- A **CI/CD pipeline** is a series of automated processes used to build, test, and deploy software applications. It enables **Continuous Integration (CI)** and **Continuous Delivery (CD)**, streamlining the development workflow and ensuring faster, reliable delivery of code changes.
- **Functions of CI/CD Pipeline**
  - **Code Integration:** Automatically integrates and validates code changes from multiple developers.
  - **Automated Testing:** Runs unit, integration, and end-to-end tests to ensure code quality.
  - **Build Automation:** Compiles the application and packages it for deployment.
  - **Deployment Automation:** Deploys the application to staging, production, or other environments.
- **Key Stages in CI/CD Pipeline**
  - **Source Control:** Triggered when code changes are pushed to a version control system like Git.
  - **Build:** The application is compiled, and dependencies are resolved.
  - **Test:** Automated tests are executed to validate functionality.
  - **Deploy:** The application is deployed to staging or production environments.

#### ❖ JENKINS

- Jenkins is an open-source **automation server** used for building, testing, and deploying software projects. It supports **Continuous Integration (CI)** and **Continuous Delivery (CD)**, helping developers integrate changes and deliver updates efficiently.
- **Functions of Jenkins**
  - **Continuous Integration:** Automatically builds and tests code whenever changes are committed.
  - **Integration:** Automatically builds and tests code whenever changes are committed.
  - **Pipeline Automation:** Manages complex build and deployment workflows using Jenkins Pipelines.
  - **Plugin Support:** Extends functionality with a vast library of plugins for tools like Git, Maven, and Docker.
  - **Build Scheduling:** Allows builds to be triggered manually, on a schedule, or based on code changes.
  - **Monitoring and Reporting:** Provides detailed build logs, test reports, and error tracking.

## Procedure

### FREESTYLE MAVEN JAVA

**STEP 1 : Make a new task for Build of Project**

The screenshot shows the Jenkins job configuration for 'Build of Project'. It includes:

- Branches to build:** Branch Specifier (blank for 'any') \*/main
- Git:** Repositories, Repository URL https://github.com/RahZero0/Maven-Project
- Credentials:** None listed.
- Build other projects:** SampleMavenProject\_test (triggered only if build is stable).
- Post-build Actions:** Archive the artifacts (Files to archive: \*\*/\*).
- General:** Maven Version MAVEN\_HOME, Goals clean, Description: To manage Maven Java Project build and set-up from Jenkins.

The screenshot shows the 'New Item' creation dialog for a 'Freestyle project'. It includes:

- Item name: SampleMavenProject\_build
- Select item type: Freestyle project (selected)
- Description: Freestyle project (Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.)
- OK button at the bottom right.

The screenshot shows the Jenkins job configuration for 'Testing Build'. It includes:

- Build Environment:** Delete workspace before build starts.
- General:** Description: This is to Test the build.
- Build Steps:** Copy artifacts from another project (Project name: SampleMavenProject\_build, Which build: Latest successful build, Stable build only checked).
- New Item:** Enter item name: SampleMavenProject\_test, Select item type: Freestyle project (Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.).

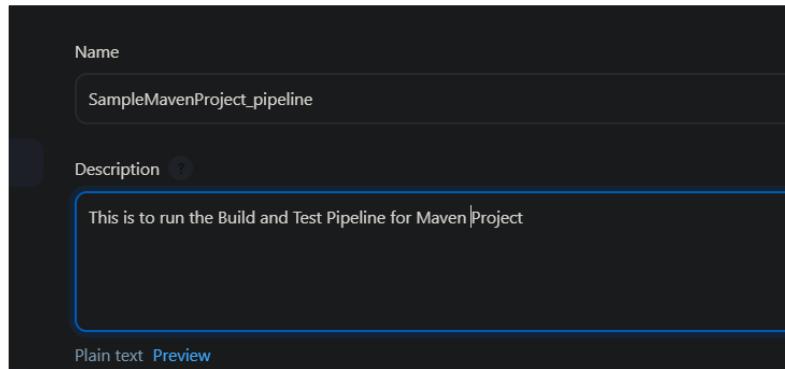
**STEP 2 : Make a new task for Testing Build**

### STEP 3 : Create a Pipeline

Name  
SampleMavenProject\_pipeline

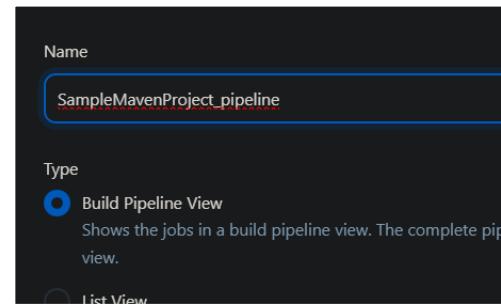
Description  
This is to run the Build and Test Pipeline for Maven Project

Plain text [Preview](#)



Name  
SampleMavenProject\_pipeline

Type  
 **Build Pipeline View**  
Shows the jobs in a build pipeline view. The complete pipeline view.  
 List View



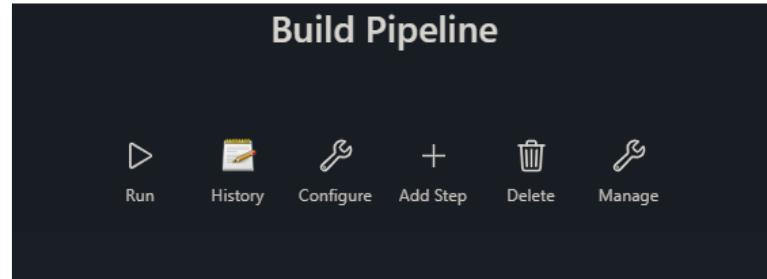
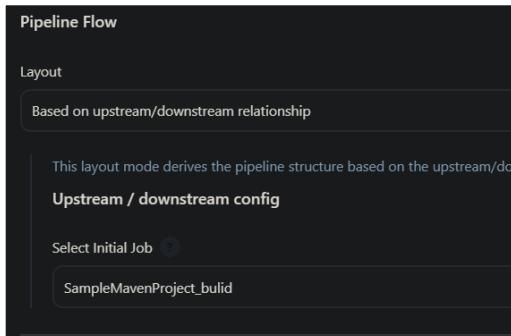
**Pipeline Flow**

Layout  
Based on upstream/downstream relationship

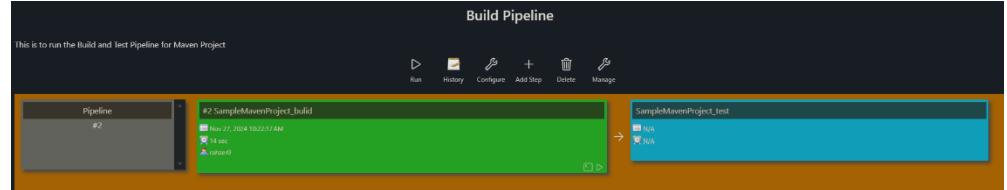
This layout mode derives the pipeline structure based on the upstream/downstream relationship.

**Upstream / downstream config**

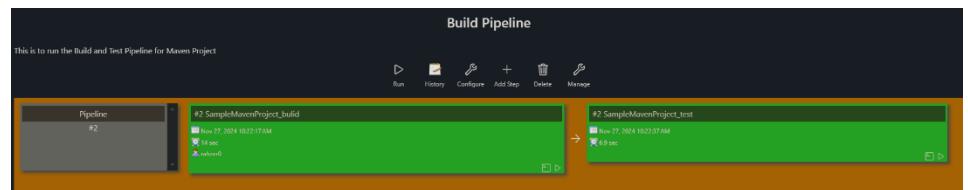
Select Initial Job  
SampleMavenProject\_build



### STEP 4 : Build Successfully



### STEP 5 : Test Successfully



## FREESTYLE MAVEN WEB

### STEP 1 : Make a new task for Build of Project

The screenshots illustrate the configuration of a Freestyle Maven project. The 'General' section contains a description of 'Manage Build of Maven Web Project'. The 'Git' section shows a repository URL of 'https://github.com/RahZero0/Maven-Web-Project'. The 'Build Steps' section includes 'Invoke top-level Maven targets' set to 'MAVEN\_HOME' and 'Goals' set to 'clean'. The 'Poll SCM' section shows a schedule of '3 \* \* \* \*'. Below these, a 'Build other projects' section lists 'SampleMavenWebProject\_test', with a note that no such project exists.

### STEP 2 : Make a new task for Testing Build

The screenshots illustrate the configuration of a Freestyle project for testing. The 'New Item' section contains a description of 'Manage Testing of Maven Web Project'. The 'Build Environment' section includes 'Delete workspace before build starts'. The 'Post-build Actions' section contains an 'Archive the artifacts' step with a pattern of '\*\*/\*'. The 'Build Steps' section includes 'Invoke top-level Maven targets' set to 'MAVEN\_HOME' and 'Goals' set to 'test'. Below these, a 'Build other projects' section lists 'SampleMavenWebProject\_deploy', with a note that no such project exists.

### STEP 3 : Make a new task for Deploying Build

The screenshot shows the Jenkins interface with several windows open:

- New Item**: A window titled "Enter an item name" with "SampleMavenWebProject\_deploy" typed in. Below it, "Freestyle project" is selected with a description: "Classic, general-purpose job type that checks out from up to one Source code management system and performs steps like archiving artifacts and sending email notifications."
- Build Environment**: A configuration window with the checkbox "Delete workspace before build starts" checked.
- Copy artifacts from another project**: A configuration window for "SampleMavenWebProject\_test". It shows "Latest successful build" selected and "Stable build only" checked.
- Deploy war/ear to a container**: A configuration window for "Tomcat 9.x Remote". It specifies "WAR/EAR files" as "\*\*/\*.war" and "Context path" as "samplePathMavenWeb".
- Jenkins Credentials Provider: Jenkins**: A credentials management window showing a credential for "admin" with password "\*\*\*\*\*".
- Containers**: A configuration window for "Tomcat 9.x Remote" with "admin/\*\*\*\*\*" as credentials and "Tomcat URL" set to "http://localhost:8090/".

### STEP 4 : Create a Pipeline

The screenshot shows the Jenkins interface with two windows:

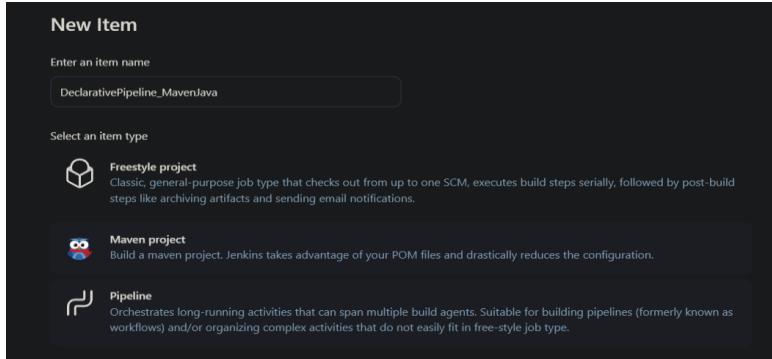
- New view**: A window for creating a new pipeline view named "SampleMavenWeb\_Pipeline". The "Type" is set to "Build Pipeline View".
- Layout**: A window titled "Based on upstream/downstream relationship" showing "Upstream / downstream config" and "Select Initial Job" set to "SampleMavenWebProject\_build".
- Build Pipeline**: A window showing the pipeline structure: "Pipeline #1" → "#1 SampleMavenWebProject\_build" → "#1 SampleMavenWebProject\_test" → "#1 SampleMavenWebProject\_deploy".
- SampleMavenWeb\_Pipeline**: A window for defining the pipeline with "Name" as "SampleMavenWeb\_Pipeline" and "Description" as "This is a Pipeline for Maven Web Project consists of 3 works - Build - Test - Deploy".

### STEP 5 : Build the Pipeline

/samplePathMavenWeb	None specified	Archetype Created Web Application	true
---------------------	----------------	-----------------------------------	------

# SCRIPTED PIPELINE USING JENKINS

## STEP 1 : Make a new task for Build of Project



## STEP 2: Description of Declarative Pipeline

General

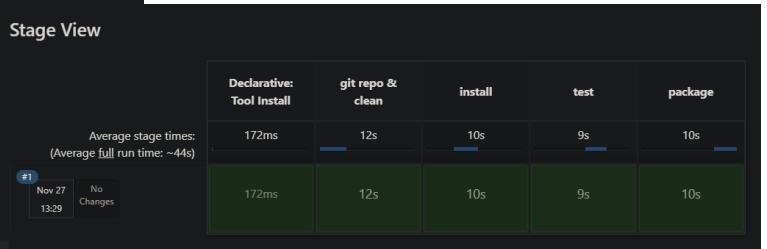
Description  
Declarative Pipeline for SCM

Plain text Preview

## STEP 3: The script for the pipeline

```
1 < pipeline {
2 agent any
3 tools {
4 maven 'MAVEN_HOME'
5 }
6 stages {
7 stage('git repo & clean') {
8 steps {
9 bat "rmdir /s /q Maven-Web-Project"
10 bat "git clone https://github.com/RahZero8/Maven-Web-Project.git"
11 bat "mvn clean -f Maven-Web-Project"
12 }
13 }
14 stage('install') {
15 steps {
16 bat "mvn install -f Maven-Web-Project"
17 }
18 }
19 stage('test') {
20 steps {
21 bat "mvn test -f Maven-Web-Project"
22 }
23 }
24 stage('package') {
25 steps {
26 bat "mvn package -f Maven-Web-Project"
27 }
28 }
29 }
30 }
```

## STEP 4: Build the pipeline



## Total Pipelines [ JENKIN ]:

S	W	Name	Last Success	Last Failure	Last Duration
○	☀	DeclarativePipeline_MavenJava	3 min 19 sec #1	N/A	44 sec
○	☀	SampleMavenProject_build	3 hr 10 min #2	N/A	14 sec
○	☀	SampleMavenProject_test	3 hr 10 min #2	N/A	6.9 sec
○	☀	SampleMavenWebProject_build	1 hr 16 min #16	N/A	13 sec
○	⌚	SampleMavenWebProject_deploy	1 hr 15 min #19	1 hr 18 min #18	0.68 sec
○	☀	SampleMavenWebProject_test	1 hr 16 min #16	N/A	7.5 sec

**Conclusion :** Thus we were able to create pipelines for MAVEN JAVA & MAVEN WEB projects using JENKINS and test the build process.

## **Experiment 7: Local Deployment of project using Docker, Kubernetes and Monitoring using Nagios tool**

**AIM:** To use various **DevOps Tools** and learn their workings in many projects.

### **Introduction:**

#### **❖ DOCKER & CONTAINERS**

- Docker is a platform for developing, shipping, and running applications using containerization.
- It allows applications to run in isolated environments called **containers**, ensuring consistency across different environments.
- **Functions of Docker**
  - **Build** lightweight, portable containers for applications.
  - **Run** containers anywhere (local machines, cloud, or servers).
  - **Package** dependencies with the application, ensuring compatibility.
  - **Manage** containerized applications efficiently using Docker commands.
  - **Scale** applications with Docker Compose and Swarm for multi-container deployments.

#### **❖ MINIKUBE**

- Minikube is a lightweight tool that allows developers to **run Kubernetes clusters locally** on their machines.
- It is designed for testing, development, and learning purposes, making it easier to experiment with Kubernetes without needing a full-scale cloud environment.
- **Functions of Minikube**
  - **Local Kubernetes Cluster:** Sets up a single-node Kubernetes cluster on your local machine.
  - **Cluster Management:** Provides commands to start, stop, and delete local clusters.
  - **Resource Testing:** Allows testing of Kubernetes configurations, deployments, and services.
  - **Multi-platform Support:** Works on Windows, macOS, and Linux.
  - **Add-ons:** Supports Kubernetes features like Ingress, Dashboard, and Metrics Server.
  - **Compatibility:** Works with kubectl and integrates seamlessly with CI/CD pipelines.

#### **❖ NAGIOS**

- Nagios is an open-source **monitoring system** used for tracking the performance and availability of IT infrastructure, including servers, networks, applications, and services.
- **Functions of Nagios**
  - **Infrastructure Monitoring:** Tracks the health of servers, network devices, and applications.
  - **Alerting:** Sends notifications via email, SMS, or other channels when issues are detected.
  - **Log Management:** Monitors logs for errors or unusual activity.
  - **Performance Metrics:** Measures system performance and generates reports.
  - **Custom Plugins:** Allows users to create custom checks for specific monitoring needs.
  - **Scalability:** Supports distributed monitoring for large infrastructures.

## Procedure

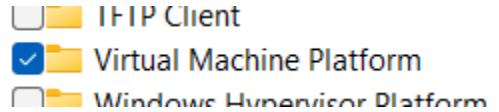
### INSTALLATIONS OF DOCKER & DOCKERHUB ACCOUNT

**STEP 1 :** Check if Virtualization is enabled in device

- └─ Open Task Manager:
- └─ Click on CPU

Hz	Sockets:	1
	Cores:	4
	Handles	Logical processors: 8
90694		Virtualization: Enabled
		L1 cache: 320 KB

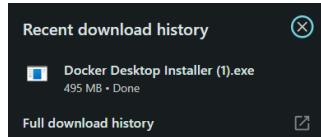
**STEP 2 :** Check Windows Features on and off



**STEP 3 :** Go to

<https://docs.docker.com/desktop/setup/install/windows-install/>

click on download



Home / Manuals / Docker Desktop / Setup / Install / Windows

### Install Docker Desktop on Windows

Docker Desktop terms

Commercial use of Docker Desktop in larger enterprises (more than 250 employees OR more than \$10 million USD in annual revenue) requires a [paid subscription](#).

This page contains the download URL, information about system requirements, and instructions on how to install Docker Desktop for Windows.

[Docker Desktop for Windows - x86\\_64](#) [Docker Desktop for Windows - Arm \(Beta\)](#)

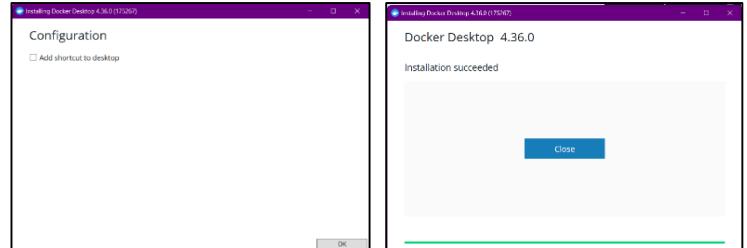
For checksums, see [Release notes](#)

#### System requirements

**STEP 4 :** Go to

<https://docs.docker.com/desktop/setup/install/windows-install/>

click on download

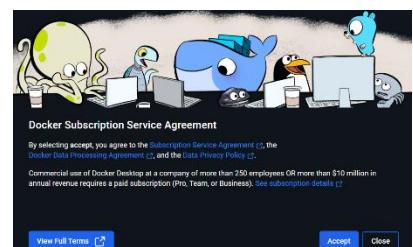


**STEP 5 :** Go to <https://hub.docker.com/>

and make an account



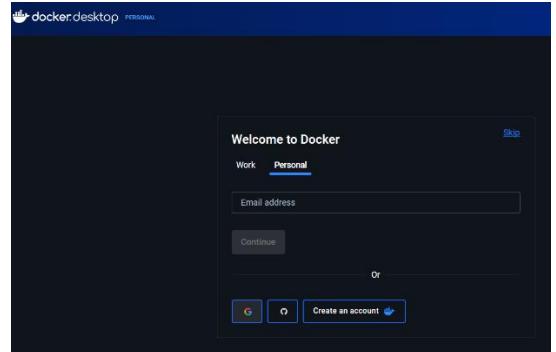
**STEP 6 :** Accept Agreement



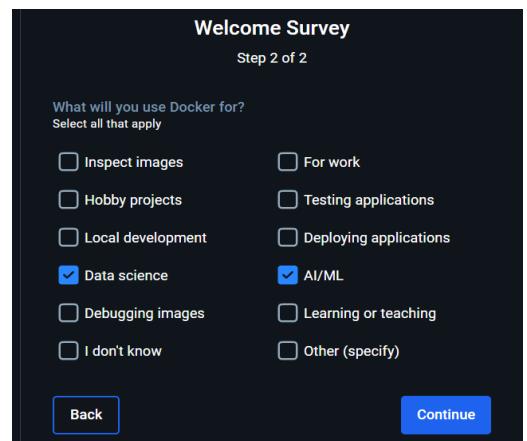
## STEP 7 : Do recommended Settings



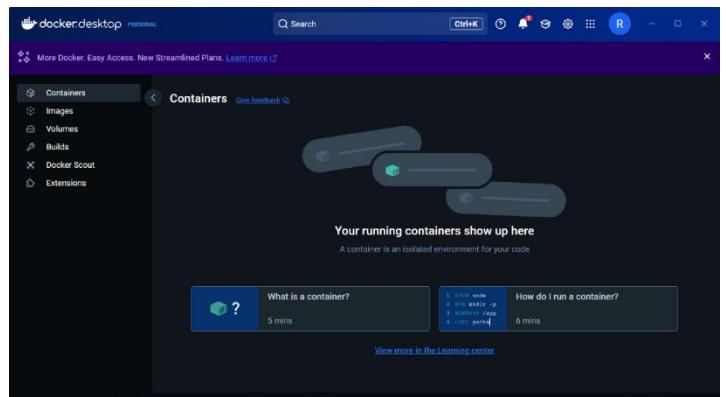
## STEP 8 : Login with Docker hub account



## STEP 9 : Click on Surveys

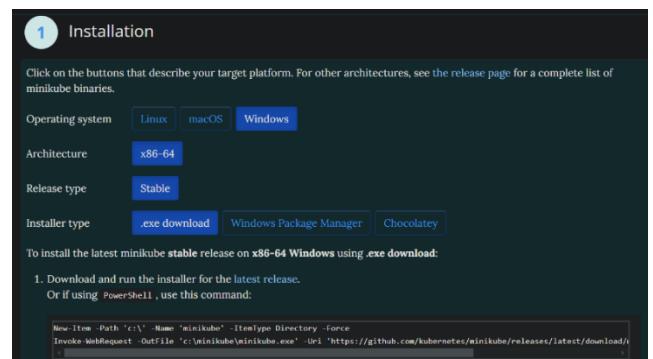


## STEP 10 : Welcome to Docker Desktop

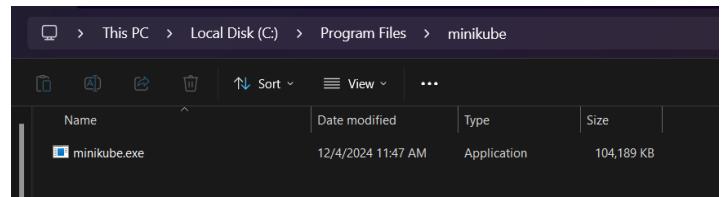


## MINIKUBE INSTALLATION

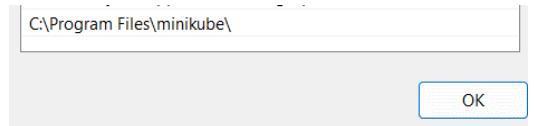
**STEP 1 :** Go to <https://minikube.download/>



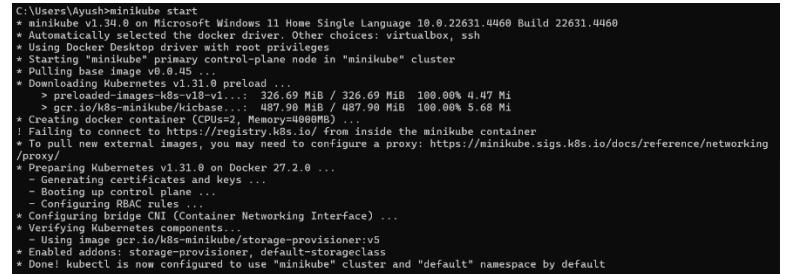
**STEP 2 :** Create a folder in Program Files and copy the exe there and rename it minikube.exe



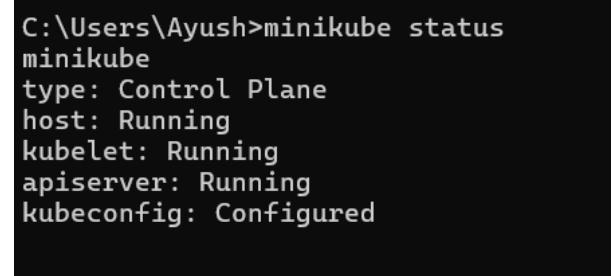
**STEP 3 :** Add to environment variables



**STEP 4 :** Type minikube start



**STEP 5 :** Type minikube status



## DOCKER CLI COMMANDS

**STEP 1 :** Docker version

```
C:\Users\Ayush>docker --version
Docker version 27.3.1, build ce12230
```

**STEP 2 :** Docker pulling hello-world

```
C:\Users\Ayush>docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:305243c734571da2d100c8b3c3167a098cab6049c9a5b066b6021a60fc966
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

**STEP 3 :** All Docker Images

```
C:\Users\Ayush>docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
gcr.io/k8s-minikube/kicbase v0.0.45 aeed0e1d4642 3 months ago 1.28GB
hello-world latest d2c94e258dcb 19 months ago 13.3kB
```

**STEP 4 :** Docker Run hello-world

```
C:\Users\Ayush>docker run hello-world
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
 (amd64)
 3. The Docker daemon created a new container from that image which runs the
 executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
 to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

**STEP 5 :** docker ps -a

```
C:\Users\Ayush>docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
631317eb0223 hello-world "/hello" 2 minutes ago Exited (0) 2 minutes ago
1804f93b3bc8 gcr.io/k8s-minikube/kicbase:v0.0.45 "/usr/local/bin/entr..." 5 days ago Exited (137) 2 days ago
 疯狂汤普森
 minikube
```

**STEP 6 :** docker rm CONTAINER ID

```
C:\Users\Ayush>docker rm 631317eb0223
631317eb0223
```

**STEP 7 :** docker ps -a  
(checking if it is there )

```
C:\Users\Ayush>docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
1804f93b3bc8 gcr.io/k8s-minikube/kicbase:v0.0.45 "/usr/local/bin/entr..." 5 days ago Exited (137) 2 days ago
 minikube
```

**STEP 8 :** docker rmi hello-world

```
C:\Users\Ayush>docker rmi hello-world
Untagged: hello-world:latest
Untagged: hello-world@sha256:305243c734571da2d100c8b3c3167a098cab6049c9a5b066b6021a60fc966
Deleted: sha256:d2c94e258dcb3c5ac2798d32e1249e42ef01cba4841c2234249495f87264ac5a
Deleted: sha256:ac28800ec8bb38d5c35b49d45a6ac4777544941199075dff8c4eb63e093aa8le
```

**STEP 9 :** docker pull redis

```
C:\Users\Ayush>docker pull redis
Using default tag: latest
latest: Pulling from library/redis
bc0965b23a04: Pull complete
9501a6ec095f: Pull complete
98e7597530ef: Pull complete
75dfffa679c9b: Pull complete
8912a88e73c8: Pull complete
141f00d6fee8: Pull complete
4f4fb700ef54: Pull complete
8242f9d5b464: Pull complete
Digest: sha256:ea96c435dc17b011f54c6a883c3c45e7726242b075de61c6fe40a10ae6ae0f83
Status: Downloaded newer image for redis:latest
docker.io/library/redis:latest
```

**STEP 10 :** docker run --name redis1 -d redis

```
C:\Users\Ayush>docker run --name redis1 -d redis
c09baf1d1b7fa145da9f3a62ece80de2c77dd0f43fba98a2a94d81f43c60e4d8
```

**STEP 11 :** docker ps -a

```
C:\Users\Ayush>docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c09baf1d1b7f redis "/docker-entrypoint.s..." About a minute ago
1804f93b3bc8 gcr.io/k8s-minikube/kicbase:v0.0.45 "/usr/local/bin/entr..." 9 days ago Up About a minute
 redis1
 minikube
```

**STEP 12 : docker ps**

```
C:\Users\Ayush>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c09baf1d1b7f redis "docker-entrypoint.s..." 8 minutes ago Up 8 minutes 6379/tcp redis1
```

**STEP 13 : docker exec -it redis1 redis-cli**

```
C:\Users\Ayush>docker exec -it redis1 redis-cli
127.0.0.1:6379> SET name "Abcdefg"
OK
127.0.0.1:6379> GET name
"Abcdefg"
127.0.0.1:6379> |
```

**STEP 14 : docker stop <redis-code>**

```
C:\Users\Ayush>docker stop c09baf1d1b7f
c09baf1d1b7f
```

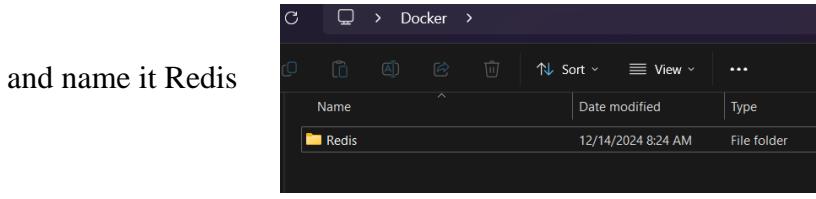
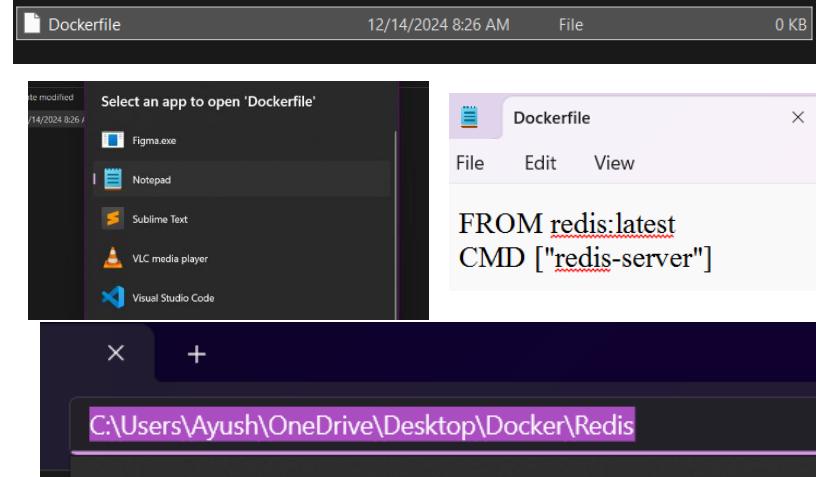
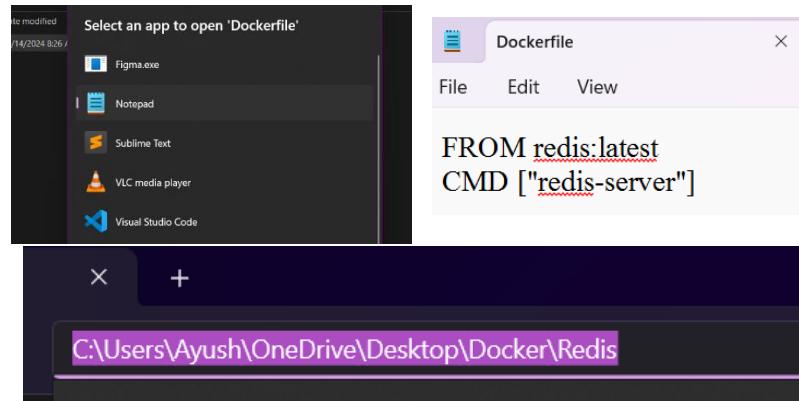
```
C:\Users\Ayush>docker rm c09baf1d1b7f
c09baf1d1b7f
```

**STEP 15 : docker ps -a**

```
C:\Users\Ayush>docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
1004f93b3bc8 gcr.io/k8s-minikube/kicbase:v0.0.45 "/usr/local/bin/entr..." 9 days ago Exited (137) 6 days ago PORTS NAMES
minikube
```

**STEP 16 : docker rmi redis**

```
C:\Users\Ayush>docker rmi redis
Untagged: redis:latest
Untagged: redis@sha256:ea96c435dc17b011f54c6a883c3c45e7726242b075de61c6fe40a10ae6ae0f83
Deleted: sha256:b5e874b32a794f96b6119e5478f9316d4110cd8c056d470b3d48b27114f716
Deleted: sha256:b29f5016befac03b5cde2788c31508900647ab8d456136768cb4743862739d1b
Deleted: sha256:6ae211a59cc93f2ff8a2d23914420f49632b93256cb925143a4b99ed07b8a1e6
Deleted: sha256:9138b812320895356a59879ae78ba94689aaf1f32ad80177967363bde5cac10d1
Deleted: sha256:36c5c6a38ba8acf7c1b035ab30704bb905d54980a2528025dc990d8513543
Deleted: sha256:aea9e1fe4e796252e3fa7c8f6400b11717c601c8b23061aa702bcc0d61ad8108
Deleted: sha256:75850a6af24741b7465394e4b26811475e867fc793aa9cfb5cca7c27e1ed788a
Deleted: sha256:d734075709827a4cba434847a33d25f14642a8b9f5589d80aa9b38ed82f3126f
Deleted: sha256:c0f1022b22a9b36851b358f44e5475e39d166e71a8073cf53c894a299239b1c5
```

**STEP 17 : Create a new folder at any location and name it Redis****STEP 18 : Name a new file as `Dockerfile`****STEP 19 : Open the file with `Notepad` and add the content**

```
FROM redis:latest
CMD ["redis-server"]
```

**STEP 20 : Copy this path**

```
C:\Users\Ayush\OneDrive\Desktop\Redis>dir
Volume in drive C has no label.
Volume Serial Number is 5082-C6BF

Directory of C:\Users\Ayush\OneDrive\Desktop\Redis

14/2024 08:26 AM <DIR> .
14/2024 08:24 AM <DIR> ..
14/2024 08:28 AM 39 Dockerfile
 39 bytes
 2 Dir(s) 15,118,471,168 bytes free
```

**STEP 21 : Change directory into this location**

```
C:\Users\Ayush\OneDrive\Desktop\Redis>cd C:\Users\Ayush\OneDrive\Desktop\Redis
C:\Users\Ayush\OneDrive\Desktop\Redis>docker build -t myredis .
[+] Building 2.0s (4/6) FINISHED
 => [internal] Load build definition from Dockerfile
 => [internal] Load build context
 => transferring dockerfile: 76B
 => transferring context: 1.06MB
 => [auth] library/redis:pull token for registry-1.docker.io
 => [internal] load .dockerignore
 => [internal] load C:\Users\Ayush\OneDrive\Desktop\Redis\Dockerfile
 => CACHED [1/1] FROM docker.io/library/redis@sha256:ea96c435dc17b011f54c6a883c3c45e7726242b075de61c6fe40a10ae6ae0f83
 => exporting to image
 => creating manifest
 => writing image sha256:b511c987af0831dd5a1712170ea8195f065f13a9948c8af664bb5d47c4d1928
 => naming to docker.io/library/myredis
Build step details: docker-desktop://dashboard/build/desktop:linux/qh0t2gsdtrgrpu1826et9zgj
```

**STEP 23 : docker images**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu latest b1d9df8ab815 3 weeks ago 78.1MB
myredis latest b811c987af08 2 months ago 117MB
gcr.io/k8s-minikube/kicbase v0.0.45 ae0e1d4642 3 months ago 1.28GB
```

**STEP 24 : docker run --name newmyredis -d myredis**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker run --name newmyredis -d myredis
ced8a89a6b5a7fe016e779703591d8563d291a3b0493693229ff29ddd3248b26
```

**STEP 25 : docker ps**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
ced8a89a6b5a myredis "docker-entrypoint.s..." About a minute ago Up About a minute 6379/tcp newmyredis
```

**STEP 26 : docker stop <container-id>**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker stop ced8a89a6b5a
ced8a89a6b5a
```

**STEP 27 : docker commit <container-id> rahzero0/redisnewimage**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker commit ced8a89a6b5a rahzero0/redisnewimage
sha256:d5b62cd36923d60a2a0bfbdccf472b81f019e7044cd0dacc586691576f8eac1c0
```

**STEP 28 : Open docker images**

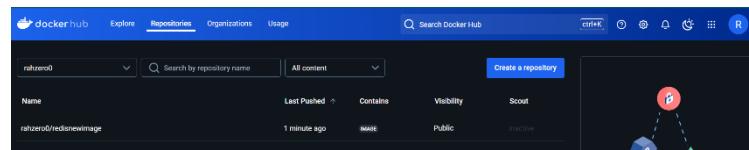
Name	Tag	Image ID	Created	Size	Actions
ubuntu	latest	b1d9df8ab815	24 days ago	78.12 MB	
gcr.io/k8s-minikube/kicbase	v0.0.45	ae0e1d4642	3 months ago	1.27 GB	
myredis	latest	b811c987af08	2 months ago	117.01 MB	
rahzero0/redisnewimage	latest	d5b62cd36923	1 minute ago	117.01 MB	

**STEP 29 : docker login**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker login
Authenticating with existing credentials...
Login Succeeded
```

**STEP 30 : docker push rahzero0/redisnewimage**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker push rahzero0/redisnewimage
Using default tag: latest
The push refers to repository [docker.io/rahzero0/redisnewimage]
4c3a66986d54: Mounted from library/redis
5f70bf18a086: Mounted from library/redis
59858d305cb4: Mounted from library/redis
3c008636f68d: Mounted from library/redis
d3ce9d926f3b: Mounted from library/redis
d575106a8e12: Mounted from library/redis
74264edca52d: Mounted from library/redis
c8f1022b22a9: Mounted from library/redis
latest: digest: sha256:fcc3e59c4d1bf0bbd8f05a425d418a37a502ac98ead01e8a60e08267dd1870cf size: 1986
```

**STEP 31 : Go to docker hub to check commit of image****STEP 32 : docker rm <container-id>**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker rm ced8a89a6b5a
ced8a89a6b5a
```

**STEP 33 : docker rmi rahzero0/redisnewimage**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker rmi rahzero0/redisnewimage
Untagged: rahzero0/redisnewimage:latest
Untagged: rahzero0/redisnewimage@sha256:fcc3e59c4d1bf0bbd8f05a425d418a37a502ac98ead01e8a60e08267dd1870cf
Deleted: sha256:d5b62cd36923d60a2a0bfbdccf472b81f019e7044cd0dacc586691576f8eac1c0
```

**STEP 34 : docker ps**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
```

**STEP 35 : docker pull**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker pull rahzero0/redisnewimage
Using default tag: latest
Latest: Pulling from rahzero0/redisnewimage
bc0965b23a04: Already exists
9501a6ec095f: Already exists
98e7597530ef: Already exists
75dffac679c9b: Already exists
8912a88e73c8: Already exists
141f00d6fee8: Already exists
4f4fb700ef54: Already exists
8242f9d5b464: Already exists
Digest: sha256:fcc3e59c4d1bf0bbd8f05a425d418a37a502ac98ead01e8a60e08267dd1870cf
Status: Downloaded newer image for rahzero0/redisnewimage:latest
docker.io/rahzero0/redisnewimage:latest
```

**STEP 36 : docker run --name newmyredis -d rahzero0/redisnewimage**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker run --name newmyredis -d rahzero0/redisnewimage
29d055e4b8f941b0929453f36108e536bf24728cf6a889a57674ae46e1f417ec
```

**STEP 37 : docker exec -it newmyredis redis-cli**

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Redis>docker exec -it newmyredis redis-cli
127.0.0.1:6379 |
```

## MODIFY & PUSH DOCKER IMAGE

**STEP 1 :** docker pull ubuntu

```
C:\Users\Ayush>docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
de44b265507a: Pull complete
Digest: sha256:80dd3c3b9c6cecb9f1667e9290b3bc61b78c2678c02cbdae5f0fea92cc6734ab
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

**STEP 2 :** docker run -it --name newubuntu -d ubuntu  
newubuntu -d ubuntu

```
C:\Users\Ayush>docker run -it --name newubuntu -d ubuntu
2130a2098e2c07bf8ac9014957f54f579cff2de715c3dc4e88ab0f83d6139e4d
```

**STEP 3 :** docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
2130a2098e2c	ubuntu	"/bin/bash"	About a minute ago	Up About a minute		newubuntu

**STEP 4 :** docker exec -it <container\_id> bash

```
C:\Users\Ayush>docker exec -it 2130a2098e2c bash
root@2130a2098e2c:/#
```

```
C:\Users\Ayush>docker exec -it 2130a2098e2c bash
root@2130a2098e2c:/# git --version
bash: command not found
root@2130a2098e2c:/# apt update
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:3 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [979 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble/restricted amd64 Packages [117 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [15.3 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [677 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble/main amd64 Packages [1808 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [663 kB]
Get:11 http://archive.ubuntu.com/ubuntu noble/universe amd64 Packages [19.3 kB]
Get:12 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [331 kB]
Get:13 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1205 kB]
Get:14 http://archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [19.7 kB]
Get:15 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [679 kB]
Get:16 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [922 kB]
Get:17 http://archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [11.9 kB]
Fetched 27.4 MB in 21s (1294 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Reading state information... Done
All packages are up to date.
```

**STEP 5 :** Check git version

```
root@2130a2098e2c:/# apt install git -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
 adduser ca-certificates git-man krb5-locales less libbrotlii libbsd0 libcbor0.10 libcurl3t64-gnutls libedit2
 liberror-perl libexpat1 libfdio2-1 libgdbm-compat4t64 libgssapi-krb5-2 libk5crypto3 libkeyutils1
 libkrb5-3 libkrb5support0 libldap-common liblbdap2 libnghttp2-14 libperl5.38t64 libpsl5t64 librmp1 libssasl2-2
 libssasl2-modules libssasl2-modules-db libssh-4 libx11-6 libx11-data libxau6 libxcb1 libxdmcp6 libxext6 libxmuu1
 netbase openssh-client openssl patch perl perl-modules-5.38 publicsuffix xauth
Suggested packages:
 liblocale-gettext-perl cron quota cryptafs-utils gettext-base git-daemon-run | git-daemon-sysvinit git-doc git-email
 git-gui gitk gitweb git-cvs git-mediawiki git-svn gdm-1ln0n krb5-doc krb5-user libssasl2-modules-gssapi-mit
 libssasl2-modules-gssapi-heimdal libssasl2-modules-ldap libssasl2-modules-otp libssasl2-modules-sql keychain
 libpam-ssh monkeysphere ssh-askpass ed diffutils-doc perl-doc libterm-readline-gnu-perl | libterm-readline-perl-perl
 make libtapt-harness-archive-perl
The following NEW packages will be installed:
 adduser ca-certificates git git-man krb5-locales less libbrotlii libbsd0 libcbor0.10 libcurl3t64-gnutls libedit2
 liberror-perl libexpat1 libfdio2-1 libgdbm-compat4t64 libgssapi-krb5-2 libk5crypto3 libkeyutils1
 libkrb5-3 libkrb5support0 libldap-common liblbdap2 libnghttp2-14 libperl5.38t64 libpsl5t64 librmp1 libssasl2-2
 libssasl2-modules libssasl2-modules-db libssh-4 libx11-6 libx11-data libxau6 libxcb1 libxdmcp6 libxext6 libxmuu1
 netbase openssh-client openssl patch perl perl-modules-5.38 publicsuffix xauth
0 upgraded, 46 newly installed, 0 to remove and 0 not upgraded.
Need to get 18.9 MB of archives.
After this operation, 92.7 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu/noble/main amd64 perl-modules-5.38 all 5.38.2-3.2build2 [3110 kB]
Get:2 http://archive.ubuntu.com/ubuntu/noble/main amd64 libgdbm6t64 amd64 libgdbm6t64 [34.4 kB]
Get:3 http://archive.ubuntu.com/ubuntu/noble/main amd64 libgdbm-compat4t64 amd64 libgdbm-compat4t64 [6710 B]
```

**STEP 6 :** apt install git -y

```
root@2130a2098e2c:/# git --version
git version 2.43.0
root@2130a2098e2c:/#
```

**STEP 7 :** git –version

```
C:\Users\Ayush>docker stop 2130a2098e2c
2130a2098e2c
```

**STEP 8 :** docker stop <container\_id>

**STEP 9 :** docker commit  
2130a2098e2c  
rahZero0/newubuntu2024

```
C:\Users\Ayush>docker commit 2130a2098e2c rahzero0/rahzero0:newubuntu
sha256:e18ca963812ca75a566a6d77264588ba9d405de8c998ac4467b788147a4fd96f
```

**STEP 10 :** docker login

```
C:\Users\Ayush>docker login
Authenticating with existing credentials...
Login Succeeded
```

**STEP 11 :** docker push  
rahZero0/newubuntu2024

```
C:\Users\Ayush>docker push rahzero0/rahzero0:newubuntu
The push refers to repository [docker.io/rahzero0/rahzero0]
95f0b4feac12: Pushed
687d50f2f6ab: Mounted from library/ubuntu
newubuntu: digest: sha256:9f89291f1648c554c3e520eedfeb1b245140fa1d4fadd0cacc8cc0db775bfac5 size: 741
```

**STEP 12 :** docker rm <container\_id>

**STEP 13 :** docker rmi  
rahZero0/newubuntu2024

**STEP 14 :** docker logout

**STEP 15 :** docker pull  
rahzero0/rahzero0:newubuntu

```
C:\Users\Ayush>docker logout
Removing login credentials for https://index.docker.io/v1/
C:\Users\Ayush>docker pull rahzero0/rahzero0:newubuntu
newubuntu: Pulling from rahzero0/rahzero0
de44b265507a: Already exists
d26b54e74d0e: Pull complete
Digest: sha256:9f89291f1648c554c3e520eedfeb1b245140fa1d4fadd0cacc8cc0db775bfac5
Status: Downloaded newer image for rahzero0/rahzero0:newubuntu
docker.io/rahzero0/rahzero0:newubuntu
```

**STEP 16 :** docker running the pulled  
container

```
C:\Users\Ayush>docker run --name newubuntu -it rahzero0/rahzero0:newubuntu
root@0b96eb780e2f:/# git --version
git version 2.43.0
root@0b96eb780e2f:/# exit
exit
```

**STEP 17 :** docker ps -a

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
0b96eb780e2f	rahzero0/rahzero0:newubuntu gcr.io/k8s-minikube/kicbase:v0.45	"/bin/bash" "/usr/local/bin/entr..."	About a minute ago 9 days ago	Exited (0) 33 seconds ago Exited (137) 6 days ago		newubuntu minikube

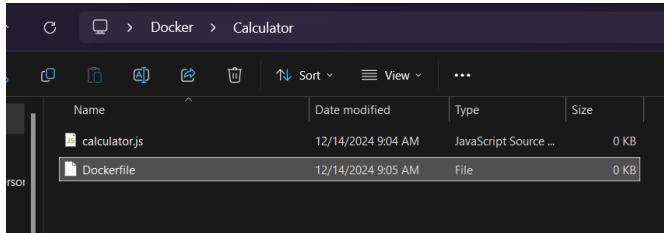
**STEP 18 :** Cleaning up

```
C:\Users\Ayush>docker rm 0b96eb780e2f
0b96eb780e2f

C:\Users\Ayush>docker rmi rahzero0/rahzero0:newubuntu
Untagged: rahzero0/rahzero0@sha256:9f89291f1648c554c3e520eedfeb1b245140fa1d4fadd0cacc8cc0db775bfac5
Untagged: rahzero0/rahzero0@sha256:9f89291f1648c554c3e520eedfeb1b245140fa1d4fadd0cacc8cc0db775bfac5
Deleted: sha256:e18ca963812ca75a566a6d77264588ba9d405de8c998ac4467b788147a4fd96f
Deleted: sha256:a25ac9206412f16355bf35afc4caacf7a0b33bf5c32ad9eef3d9c458e9c7175
```

# CREATE & PUSH DOCKER FILE IMAGE

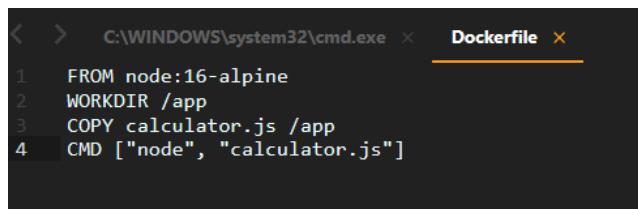
**STEP 1 :** Create a new folder, name it Calculator and have files calculator.js and Dockerfile



**STEP 2 :** Write this content in calculator.js

```
FOLDERS 1 function add (a, b) {
 2 return a + b;
 3 }
 4
▼ Calculator 5 function subtract (a, b) {
 6 return a - b;
 7 }
 8
JS calculator.js 9 function multiply (a, b) {
 10 return a * b;
 11 }
 12
 13 function divide(a, b) {
 14 if (b == 0) {
 15 return "Cannot divide by zero!";
 16 }
 17 return a/b;
 18 }
 19
 20 console.log("Addition (2 + 3): ", add(2, 3));
 21 console.log("Subtraction (5 - 2): ", subtract(5, 2));
 22 console.log("Multiplication (4 * 3): ", multiply(4, 3));
 23 console.log("Division (10 / 2): ", divide(10, 2));
 24
```

### **STEP 3 : Write this content in Dockerfile**



**STEP 4 :** docker build -t simple-calculator .

**STEP 5** :docker run simple-calculator

```
C:\Users\Ayush\OneDrive\Desktop\Calculator>docker run simple-calculator
Addition (2 + 3): 5
Subtraction (5 - 2): 3
Multiplication (4 * 3): 12
Division (10 / 2): 5

C:\Users\Ayush\OneDrive\Desktop\Calculator>
```

## STEP 6 : docker login

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker login
Authenticating with existing credentials...
Login Succeeded
```

**STEP 7 :** docker tag simple-calculator rahzero0/simple-calculator

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker tag simple-calculator rahzero0/simple-calculator
```

□	Name	Tag	Image ID	Created	Size	Actions
□	ubuntu	latest	b1dbdf8ab815	24 days ago	78.12 MB	▷ ⚙️ 🗑️
□	gcr.io/k8s-minikube/kicbase	v0.45	aeed0e1d4642	3 months ago	1.27 GB	▷ ⚙️ 🗑️
□	myredis	latest	b811c987a086 ↗	7 months ago	117.01 MB	▷ ⚙️ 🗑️
□	simple-calculator	latest	15611c751eab	5 minutes ago	117.53 MB	▷ ⚙️ 🗑️
□	raher0/simple-calculator	latest	15611c751eab	5 minutes ago	117.53 MB	▷ ⚙️ 🗑️

## STEP 8 : docker push rahzero0/simple-calculator

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker push rahzero0/simple-calculator
Using default tag: latest
The push refers to repository [docker.io/rahzero0/simple-calculator]
6afeaab59eb8: Pushed
4efa8beb928e: Pushed
365cc918307: Mounted from library/node
1bba629c69e9: Mounted from library/node
139c1270acf1: Mounted from library/node
4693057ce236: Mounted from library/node
latest: digest: sha256:61bb3e40417002724bc0d6261d1649f742b4ca885dffffa415291e337c956a951 size: 1571
```

Docker Hub interface showing the repository rahzero0. The table displays two entries:

Name	Last Pushed	Contains	Visibility	Scout
rahzero0/redisnewimage	35 minutes ago	IMAGE	Public	Inactive
rahzero0/rahzero0	about 18 hours ago	IMAGE	Public	Inactive

## STEP 9 : Remove image from local system

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker rm 56148960201f
56148960201f

C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker rmi rahzero0/simple-calculator
Untagged: rahzero0/simple-calculator:latest
Untagged: rahzero0/simple-calculator@sha256:61bb3e40417002724bc0d6261d1649f742b4ca885dffffa415291e337c956a951
```

## STEP 10 : docker pull rahzero0/simple-calculator

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker pull rahzero0/simple-calculator
Using default tag: latest
latest: Pulling from rahzero0/simple-calculator
Digest: sha256:61bb3e40417002724bc0d6261d1649f742b4ca885dffffa415291e337c956a951
Status: Downloaded newer image for rahzero0/simple-calculator:latest
docker.io/rahzero0/simple-calculator:latest
```

## STEP 11 : docker run rahzero0/simple-calculator

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker run rahzero0/simple-calculator
Addition (2 + 3): 5
Subtraction (5 - 2): 3
Multiplication (4 * 3): 6
Division (10 / 2): 5
```

## STEP 12 : Cleaning up

```
C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker rm a2728d423e5b
a2728d423e5b

C:\Users\Ayush\OneDrive\Desktop\Docker\Calculator>docker rmi rahzero0/simple-calculator
Untagged: rahzero0/simple-calculator:latest
Untagged: rahzero0/simple-calculator@sha256:61bb3e40417002724bc0d6261d1649f742b4ca885dffffa415291e337c956a951
```

# RUNNING MULTIPLE CONTAINERS USING DOCKER COMPOSE

STEP 1 : Create a compose folder where we write docker-compose( .yaml / .yml )

```
docker-compose.yaml
version: '3'

services:
 # Database
 db:
 image: mysql:5.7
 volumes:
 - db_data:/var/lib/mysql
 restart: always
 environment:
 MYSQL_ROOT_PASSWORD: password
 MYSQL_DATABASE: wordpress
 MYSQL_USER: wordpress
 MYSQL_PASSWORD: wordpress
 networks:
 - wpsite
 # Wordpress
 wordpress:
 depends_on:
 - db
 image: wordpress:latest
 ports:
 - '8000:80'
 restart: always
 volumes: ['./:/var/www/html']
 environment:
 WORDPRESS_DB_HOST: db:3306
 WORDPRESS_DB_USER: wordpress
 WORDPRESS_DB_PASSWORD: wordpress
 networks:
 - wpsite
networks:
 wpsite:
 volumes:
 db_data:
```

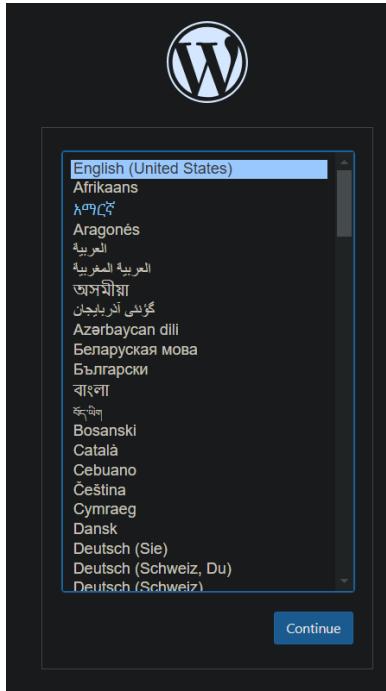
STEP 2 : Move to the location of above file , type docker-compose up -d

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>docker-compose up -d
time="2024-12-15T14:22:31+05:30" level=warning msg="C:\\\\Users\\\\Ayush\\\\OneDrive\\\\Desktop\\\\Docker\\\\compose\\\\docker-compose.yaml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 26/24
- db [=====] Pulling 320.5s
- wordpress [======] Pulling 320.5s

[+] Running 35/15[=====] Pulling 3
 ✓db Pulled 530.1s
 ✓wordpress Pulled 531.3s

[+] Running 4/4
 ✓Network compose_wpsite Created 0.1s
 ✓Volume "compose_db_data" Created 0.0s
 ✓Container compose-db-1 Started 2.7s
 ✓Container compose-wordpress-1 Started 1.8s
```

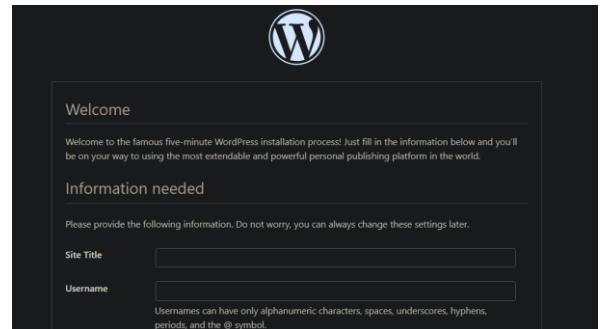
STEP 3 : Go to localhost:8000



STEP 4 : docker ps

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
ba125ace4dd wordpress:latest "docker-entrypoint.s..." 4 minutes ago Up 4 minutes 0.0.0.0:8000->80/tcp compose-wordpress-1
020c289db376 mysql:5.7 "docker-entrypoint.s..." 4 minutes ago Up 4 minutes 3306/tcp, 33060/tcp compose-db-1
```

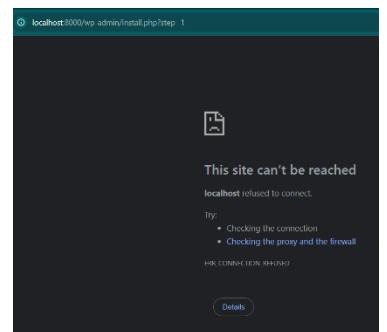
STEP 5 :  
We can create  
username and password



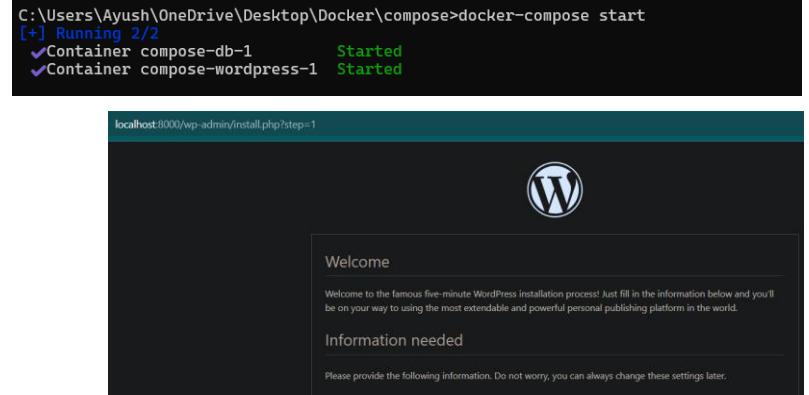
STEP 6 : docker-compose stop

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>docker-compose stop
[+] Stopping 2/0
 ✓Container compose-wordpress-1 Stopped 0.0s
 ✓Container compose-db-1 Stopped 0.0s
```

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
```



## STEP 7 : docker-compose start



```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
ba125cace4dd wordpress:latest "docker-entrypoint.s..." 9 minutes ago Up 39 seconds 0.0.0.0:8000->80/tcp compose-wordpress-1
020c289db376 mysql:5.7 "docker-entrypoint.s..." 10 minutes ago Up 39 seconds 3306/tcp, 33060/tcp compose-db-1
```

## STEP 8 : To delete everything

- docker-compose stop
- docker-compose down

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>docker-compose stop
[+] Stopping 2/2
 ✓ Container compose-wordpress-1 Stopped
 ✓ Container compose-db-1 Stopped
1.4s
2.1s

C:\Users\Ayush\OneDrive\Desktop\Docker\compose>docker-compose down
[+] Running 3/3
 ✓ Container compose-wordpress-1 Removed
 ✓ Container compose-db-1 Removed
 ✓ Network compose_wpsite Removed
0.0s
0.0s
0.25

C:\Users\Ayush\OneDrive\Desktop\Docker\compose>
```

## DEPLOYING AND SCALING APPLICATIONS USING MINIKUBE

**STEP 1 :** minikube start

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>minikube start
* minikube v1.34.0 on Microsoft Windows 11 Home Single Language 10.0.22631.4602 Build 22631.4602
* Using the docker driver based on existing profile
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.45 ...
* Restarting existing docker container for "minikube" ...
! Failing to connect to https://registry.k8s.io/ from inside the minikube container
* To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
* Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
* Verifying Kubernetes components...
- Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

**STEP 2 :** kubectl create deployment mynginx --image=nginx

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl create deployment mynginx --image=nginx
deployment.apps/mynginx created
```

**STEP 3 :** kubectl get deployments

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl get deployments
NAME READY UP-TO-DATE AVAILABLE AGE
mynginx 0/1 1 0 31s
```

**STEP 4 :** kubectl expose deployment mynginx --type=NodePort --port=80 --target-port=80

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl expose deployment mynginx --type=NodePort --port=80 --target-port=80
service/mynginx exposed
```

**STEP 5 :** kubectl scale deployment mynginx --replicas=4

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl scale deployment mynginx --replicas=4
deployment.apps/mynginx scaled
```

**STEP 6 :** Checking deployments / kubectl get deployments

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl get deployments
NAME READY UP-TO-DATE AVAILABLE AGE
mynginx 4/4 4 4 4m22s
```

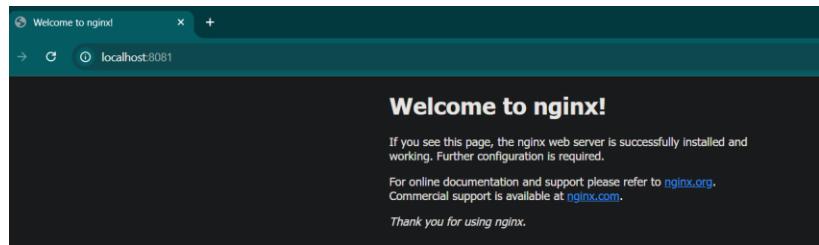
**STEP 7 :** kubectl get pods

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl get pods
NAME READY STATUS RESTARTS AGE
mynginx-79bb8756c7-8pgd2 1/1 Running 0 62s
mynginx-79bb8756c7-h6cbx 1/1 Running 0 62s
mynginx-79bb8756c7-sn5gq 1/1 Running 0 62s
mynginx-79bb8756c7-x2fx4 1/1 Running 0 4m55s
```

**STEP 8 :**

kubectl port-forward svc/mynginx 8081:80

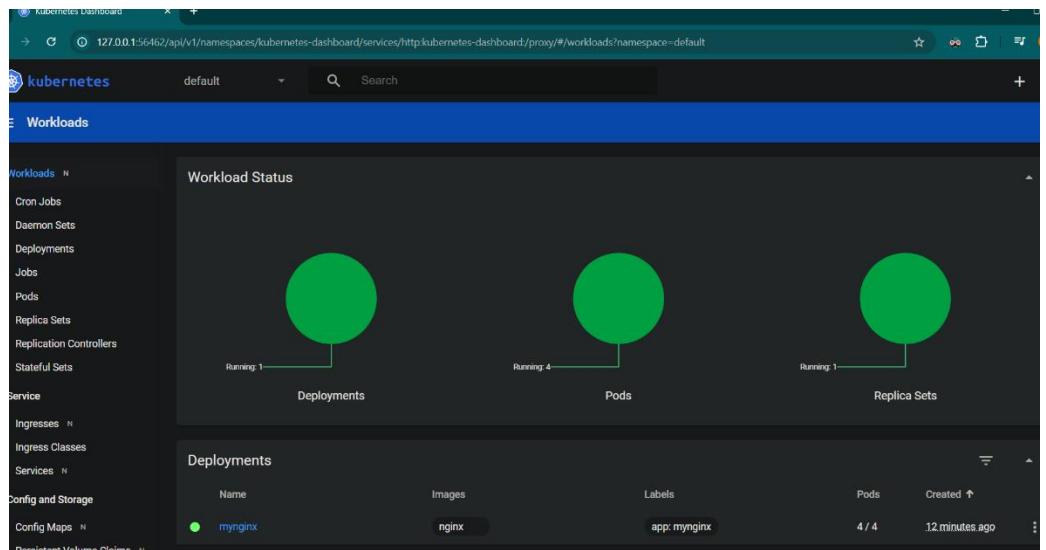
```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl port-forward svc/mynginx 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
```



## STEP 9 : minikube dashboard

```
C:\Users\Ayush>minikube dashboard
* Enabling dashboard ...
 - Using image docker.io/kubernetesui/dashboard:v2.7.0
 - Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
* Some dashboard features require the metrics-server addon. To enable all features please run:
 minikube addons enable metrics-server

* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:56462/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
```



## STEP 10 : kubectl delete deployments mynginx

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl delete deployments mynginx
deployment.apps "mynginx" deleted
C:\Users\Ayush\OneDrive\Desktop\Desktop\compose>
```

## STEP 11 : kubectl delete service mynginx

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>kubectl delete service mynginx
service "mynginx" deleted
```

## STEP 12 : minikube stop

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>minikube stop
* Stopping node "minikube" ...
* Powering off "minikube" via SSH ...
* 1 node stopped.
```

# **DEPLOYING AND MANAGING MONITORING SYSTEMS USING NAGIOS IN DOCKER**

## **STEP 1 : docker pull jasonrivers/nagios:latest**

```
[User@Ayush]~$ docker pull jasonrivers/nagios:latest
Latest: Pulling from jasonrivers/nagios
ff65ddfa0939b5: Pull complete
785b9873bfdfu: Pull complete
0ef946ba5ac5c: Pull complete
53af38babdc: Pull complete
d72f92e29533: Pull complete
769ed7dceea: Pull complete
d3245570f968: Pull complete
eeb79e60d03eb: Pull complete
0b05f5905eb: Pull complete
71fb309fc8cb: Pull complete
738fc7520889: Pull complete
feba0a02cf4e3: Pull complete
ed6f8fab512d1: Pull complete
15f36d0b09d39: Pull complete
a2fc4187e3bd: Pull complete
3d5785144815: Pull complete
566cdcc0255d0: Pull complete
c789b9e87933: Pull complete
44003949e75d4: Pull complete
b59c76b02bb6: Pull complete
d5aa2a3a6539: Pull complete
8fb39af17153: Pull complete
9ffe54c5c139: Pull complete
279028aeaf10: Pull complete
a900dffcce38: Pull complete
9a96615e352c: Pull complete
8e911c59d28: Pull complete
c191d9d53: Pull complete
b00909daad: Pull complete
e3389e58e867: Pull complete
Digest: sha256:2a7cb2d118baaf92b47b69a390le680d7664617881b94e5680b4d46564d6ae54
Status: Downloaded newer image for jasonrivers/nagios:latest
docker.io/jasonrivers/nagios:latest
```

**STEP 2 :** docker run --name nagiosdemo -p 8888:80 jasonrivers/nagios:latest

```
C:\Users\Ayush\OneDrive\Desktop\Docker\compose>docker run --name nagiosdemo -p 8888:80 jasonrivers/nagios:latest
Adding password for user nagiosadmin
chown: warning: `.' should be `::': 'nagios.nagios'
Started runsuid, PID is 13
checking permissions for nagios & nagiosgraph
rsyslogd: [origin software="rsyslogd" swVersion="8.2312.0" x-pid="21" x-info="https://www.rsyslog.com"] start
nscat[24]: Starting up daemon

Nagios Core 4.5.7
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-10-24
License: GPL

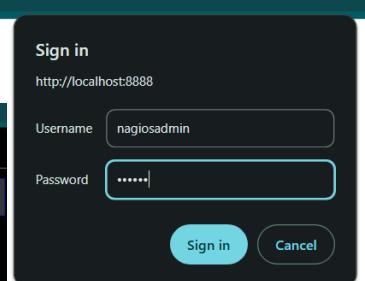
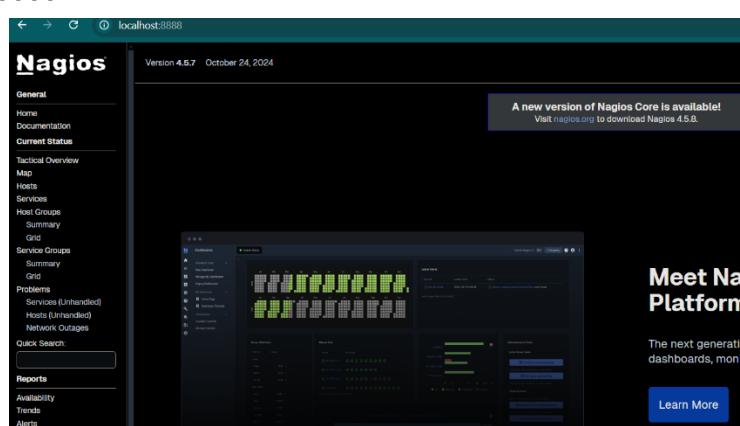
Website: https://www.nagios.org
Nagios 4.5.7 starting... (PID=23)
Local time is Sun Dec 15 09:54:17 UTC 2024
nagios: Nagios 4.5.7 starting... (PID=23)
nagios: Local time is Sun Dec 15 09:54:17 UTC 2024
nagios: LOG VERSION: 2.0
nagios: qh: Socket '/opt/nagios/var/rw/nagios.gh' successfully initialized
nagios: qh: core query handler registered
nagios: qh: echo service query handler registered
nagios: qh: help for the query handler registered
wproc: Successfully registered manager as @wproc with query handler
nagios: wproc: Successfully registered manager as @wproc with query handler
wproc: Registry request: name=Core Worker 42;pid=42;nagios: wproc: Registry request: name=Core Worker 42;pid=42

wproc: Registry request: name=Core Worker 44;pid=44
wproc: Registry request: name=Core Worker 43;pid=43
wproc: Registry request: name=Core Worker 45;pid=45
nagios: wproc: Registry request: name=Core Worker 44;pid=44
wproc: Registry request: name=Core Worker 46;pid=46
wproc: Registry request: name=Core Worker 50;pid=50;nagios: wproc: Registry request: name=Core Worker 43;pid=43

nagios: wproc: Registry request: name=Core Worker 45;pid=45
nagios: wproc: Registry request: name=Core Worker 46;pid=46
wproc: Registry request: name=Core Worker 51;pid=51
nagios: wproc: Registry request: name=Core Worker 50;pid=50
wproc: Registry request: name=Core Worker 53;pid=53
wproc: Registry request: name=Core Worker 47;pid=47
```

### **STEP 3 : Go to localhost:8888**

- Username: nagiosadmin  
→ Password: nagios



#### STEP 4 :

docker ps

```
C:\Users\Ayush>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
2930c2d3d928 jasonrivers/nagios:latest "/usr/local/bin/star..." 4 minutes ago Up 4 minutes 5667/tcp, 0.0.0.0:8888->80/tcp nagiosdemo
```

#### STEP 5 : docker stop nagios

```
C:\Users\Ayush>docker stop nagiosdemo
nagiosdemo
```

```
* Response time: 0ms
Shutting Down
Caught SIGTERM, shutting down...nagios: Caught SIGTERM, shutting down...
nsca[24]: Caught SIGTERM - shutting down.
nsca[24]: Cannot remove pidfile '/var/run/nsca.pid' - check your privileges.
nsca[24]: Daemon shutdown
postfix/master[25]: fatal: master_sigdeath: kill process group: No such process
ok: down: svlogd: 0s, normally up, want up
Successfully shutdown... (PID=23)
ok: down: apache: 1s, normally up
ok: down: nagios: 1s, normally up
ok: down: nsca: 1s, normally up
ok: down: rsyslog: 1s, normally up
ok: down: postfix: 0s, normally up
```

#### STEP 6 : docker rm nagiosdemo

```
C:\Users\Ayush>docker rm nagiosdemo
nagiosdemo
```

#### STEP 7 : Deleting image

- ➔ docker images
- ➔ docker rmi  
jasonrivers/nagios

```
C:\Users\Ayush>docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
rahzero0/simple-calculator latest 15611c751eab 30 hours ago 118MB
simple-calculator latest 15611c751eab 30 hours ago 118MB
wordpress latest c89b40a25cd1 3 weeks ago 700MB
ubuntu latest b1d9df8ab815 3 weeks ago 78.1MB
jasonrivers/nagios latest 2ab73fb5d162 5 weeks ago 960MB
myredis latest b811c987af08 2 months ago 117MB
gcr.io/k8s-minikube/kicbase v0.0.45 ae0d0e1d4642 3 months ago 1.28GB
mysql 5.7 5107333e08a8 12 months ago 501MB

C:\Users\Ayush>docker rmi jasonrivers/nagios
Untagged: jasonrivers/nagios:latest
Untagged: jasonrivers/nagios@sha256:2a7c2b20d118baf92b47b69a3901e68dd7664617801b94e560bc4d6564d6ae54
Deleted: sha256:2ab73fb5d162ad0be4ef495fb62e17772120b32721a6c3eb9091fcc800d2909b
Deleted: sha256:63lab139f760d68457a4186da4a60f5e5f1e65aa3b26864b5df237ef75b1311
Deleted: sha256:b10e2a7f4de8bcf15a69232335422c9a736c4a4a93ca288592a2fb53a2de6f45
Deleted: sha256:3a5a34bf7e5ab76f1a3054c5854a9bc7227fd5a0dec73d4a4673456f69cec753
Deleted: sha256:0603355dfe22b531eff49627391c7abb944163a735d9142fcf02529e3c8ff
Deleted: sha256:2cdfed24d1997cb2951dc561b1c4fb1947069e22801aab9103cf0ffcc1e8c532
Deleted: sha256:5d2ae66d21d91026745748ae98d112256472f76f72deaa904a368b9dd41ca987
Deleted: sha256:b9759ebfa6418db67d02fd861af913fd571bd06c6d3ccddfe760d637e49ff15f2
Deleted: sha256:62b02d2bded253673e8e75637dda61e85ad0fbe2e2eaa48bcfcda314197add5f4
Deleted: sha256:c8bf237c41fd124329567e29514d381ae822f6d202565e013df75bb47811c94d
Deleted: sha256:5e918b338fe317b5f88044bf83c21c5bcalf8570092859d4a0c05a8e2e036ce
Deleted: sha256:24ce75f6abff9acee33a572c2adb3cccf1a57d995baleaad9bf68b9a50d5bfab
Deleted: sha256:bc291db0b02d187622e2bb6f1c85dcbb5baaa3a37ce557e5357a628b3befbb0df3
```

**Conclusion :** Thus we were able to use **Docker** containers and images and learn about **Nagios**.

## **Experiment 8: CLOUD DEPLOYMENT OF A PROJECT IN THE AWS CLOUD USING EC2 INSTANCE**

**AIM:** To use AWS services and implement its EC2 instance in our projects.

### **Introduction:**

#### **❖ AWS (Amazon Web Services)**

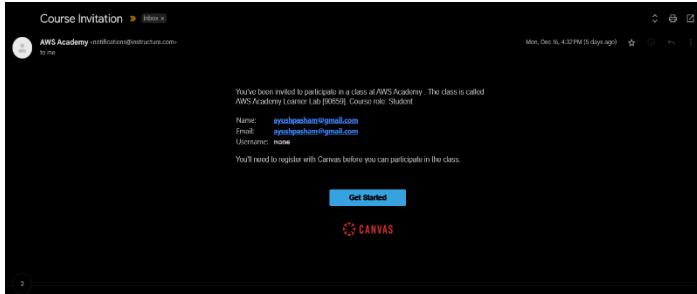
- AWS is a comprehensive **cloud computing platform** provided by Amazon. It offers a wide range of services, including computing power, storage, databases, networking, and machine learning, to help businesses build and scale applications without managing physical hardware.
- **Functions of AWS**
  - **Compute Services:** Provides scalable computing power through services like EC2 (Elastic Compute Cloud) and Lambda (server less computing).
  - **Storage Solutions:** Offers reliable storage options like S3 (Simple Storage Service) and EBS (Elastic Block Store).
  - **Database Management:** Includes managed database services like RDS (Relational Database Service) and Dynamo DB (NoSQL).
  - **Networking:** Enables secure networking with services like VPC (Virtual Private Cloud) and Route 53 (DNS).
  - **AI and Machine Learning:** Provides AI tools like Sage Maker for training and deploying machine learning models.
  - **Deployment and Monitoring:** Facilitates application deployment and monitoring with services like Elastic Beanstalk and Cloud Watch.
  - **Security and Identity:** Ensures secure access using services like IAM (Identity and Access Management).

#### **❖ EC2 INSTANCE**

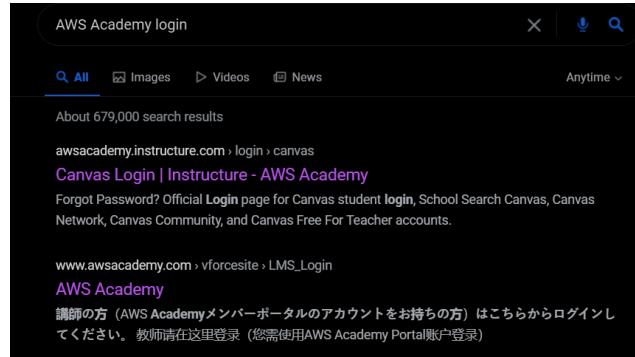
- An **EC2 (Elastic Compute Cloud) instance** is a virtual server provided by AWS for running applications. It allows users to launch, configure, and manage virtual machines with customizable operating systems, CPU, memory, and storage options.
- **Functions of EC2 Instance**
  - **On-demand Computing Power:** Provides scalable virtual servers to meet varying workloads.
  - **Customizable Configuration:** Supports different instance types optimized for compute, memory, or storage needs.
  - **Operating System Choices:** Allows the use of Linux, Windows, or custom OS images.
  - **Auto Scaling:** Dynamically adjusts the number of instances based on demand.
  - **Elasticity:** Start, stop, or resize instances as needed.
  - **Integration with AWS Services:** Works seamlessly with S3, RDS, and other AWS services.
  - **Secure Access:** Uses key pairs and security groups to control access to instances.

## AWS ACADEMY LEARNING ACCOUNT CREATION

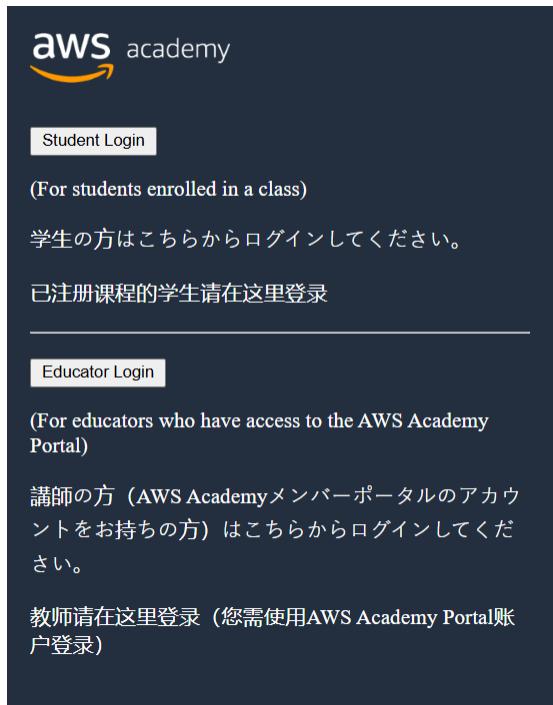
**STEP 1 :** Open your mail and click on **Get Started**



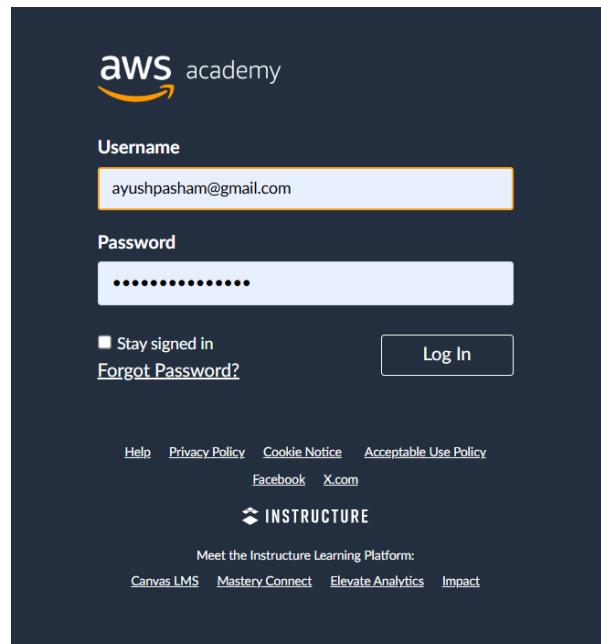
**STEP 2 :** Open Google and type **AWS Academy Login**



**STEP 3 :** Open this page and click on **Student Loign**



**STEP 4 :** Open Enter your credentials



**STEP 5 :** Open Click on Modules in this Learner Lab



**STEP 6 :** Click on this option **Launch AWS Academy Learner Lab**



## STEP 7 : Accept Terms and Conditions

The screenshot shows the 'Terms and Conditions' section of the Vocareum website. It includes a header with the Vocareum logo, a navigation bar with 'AWS', 'Start Lab', 'AWS Details', and 'Readme', and a sidebar with 'EN-US' and a tree icon. The main content area contains sections for '1. Agreement To Terms' and '2. Changes to Terms or Services', both of which contain detailed legal text. A red arrow points from the 'Start Lab' button at the bottom right of the main content area towards the 'Start Lab' button in the top right corner of the entire window.

## STEP 8 : Start Lab by clicking on Start Lab button

The screenshot shows the Vocareum interface after a lab has been started. The 'Start Lab' button is now greyed out. The sidebar on the right shows a tree icon with several nodes expanded, including '4. Config', 'H', 'B', 'C', 'D', 'V', 'E', 'F', 'G', 'B', 'D', 'I', and 'J'. A red arrow points from the 'Start Lab' button in the top right corner towards the 'Start Lab' button in the bottom right corner of the main content area.

## STEP 9 : After successfully starting the lab click on AWS

The screenshot shows an AWS terminal window with a green status indicator. The terminal prompt is 'eee\_l\_3941764@runweb155583:~\$'. A red arrow points from the 'AWS' button in the top left corner towards the terminal window.

## STEP 10 : This interface will be opened

The screenshot shows the AWS Console Home interface. It features a 'Recently visited' section with a note 'No recently visited services' and links to 'View all services', 'EC2', 'SS', 'RDS', and 'Lambda'. On the right, there's a sidebar for 'Application' with 'Region: US East (us-east-1)' and a 'Name' input field. A red arrow points from the 'AWS' button in the top left corner towards the 'Welcome to AWS' link in the bottom left of the main content area.

## STEP 11 : Close the lab by clicking on close lab and then Yes

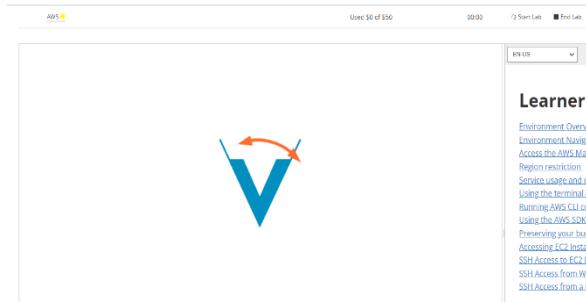
The screenshot shows a confirmation dialog box asking 'Are you sure you want to end the lab?'. Below it, a message says 'Named **LabRole** and **LabInstanceProfile** have been created for you. You can attach the role (via the instance profile) to an EC2 instance when you want to access an EC2 instance terminal in the future.' At the bottom are 'Yes' and 'No' buttons. A red arrow points from the 'End Lab' button in the top right corner towards the 'Yes' button in the dialog box.

## STEP 12 : Verify that the lab is closed by getting the red dot

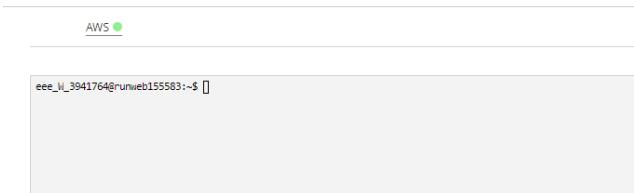
The screenshot shows the Vocareum interface again. The 'Start Lab' button is now red, indicating the lab is closed. The sidebar on the right shows a tree icon with a red dot next to the '4. Config' node. A red arrow points from the red 'Start Lab' button in the top right corner towards the red dot in the sidebar.

# PROJECT DEPLOYMENT IN THE AWS CLOUD USING EC2 INSTANCE

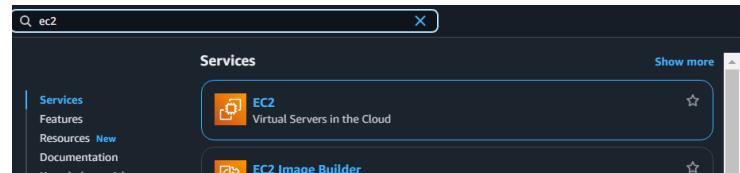
**STEP 1 :** Start the lab and wait for the button to turn green



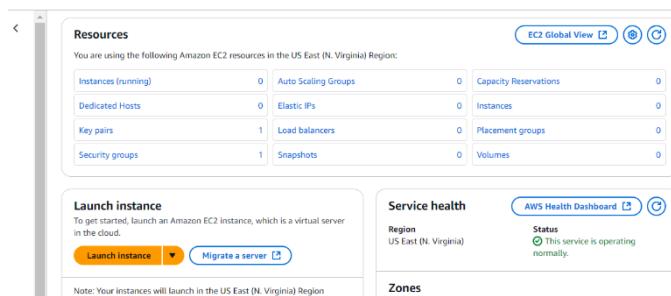
**STEP 2 :** As soon as it turns green click on AWS



**STEP 3 :** As you enter the lab search for **EC2** instance



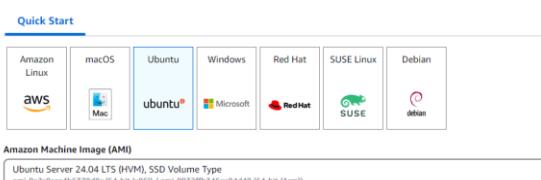
**STEP 4 :** Wait for the screen to load and then click on **Launch Instance**



**STEP 5 :** Type an appropriate name



**STEP 6 :** Select Ubuntu Free Tier



**STEP 7 :** Select 64 bit architecture



## STEP 8 : Select t2 free instance type

**Instance type** [Info](#) | [Get advice](#)

**Instance type**

t2.micro  
Family: t2 1 vCPU 1 GiB Memory Current generation: true  
On-Demand Windows base pricing: 0.0162 USD per Hour On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour  
On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0.026 USD per Hour

**Additional costs apply for AMIs with pre-installed software**

## STEP 9 : Create your key by clicking RSA

**Create key pair**

**Key pair name**  
Key pairs allow you to connect to your instance securely.  
  
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type**  
 RSA RSA encrypted private and public key pair  
 ED25519 ED25519 encrypted private and public key pair

**Private key file format**  
 .pem For use with OpenSSH  
 .ppk For use with PuTTY

**When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance.** [Learn more](#)

[Cancel](#) [Create key pair](#)

## STEP 10 : In network settings allow both HTTP – HTTPS traffic

**Firewall (security groups)** [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

[Create security group](#) [Select existing security group](#)

We'll create a new security group called 'launch-wizard-1' with the following rules:

Allow SSH traffic from anywhere  
Helps you connect to your instance

Allow HTTPS traffic from the internet  
To set up an endpoint, for example when creating a web server

Allow HTTP traffic from the internet  
To set up an endpoint, for example when creating a web server

**When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance.** [Learn more](#)

## STEP 11 : In storage settings select either 8GB | 10GB which is still free tier

**Configure storage** [Info](#)

1x 8 GiB gp3 Root volume (Not encrypted)

**Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage per month.**

[Add new volume](#)

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance

## STEP 12 : Click on Launch Instance

[Cancel](#) [Launch instance](#) [Preview code](#)

## STEP 13 : Verify instance has launched

**Launching instance**  
Creating security groups 27%

**Details**

**Success**  
Successfully initiated launch of instance ([i-0bf9e8611de52b898](#))

Please wait while we launch your instance. Do not close your browser while this is loading.

**Launch log**

## STEP 14 : Wait for the instance to load

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
MyAWSExampleWebMaven	i-0bf9e8611de52b898	<span style="color: green;">Running</span>	t2.micro	<span style="color: yellow;">Initializing</span>	<a href="#">View alarms</a>	us-east-1c

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
MyAWSExampleWebMaven	i-0bf9e8611de52b898	<span style="color: green;">Running</span>	t2.micro	<span style="color: green;">2/2 checks passed</span>	<a href="#">View alarms</a>	us-east-1c

## STEP 15 : SSH Connect to the instance

**Instance ID**  
i-0bf9e8611de52b898 (MyAWSExampleWebMaven)

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is MyFirstAWSkey.pem.
- Run this command, if necessary, to ensure your key is not publicly viewable.  
`chmod 400 "MyFirstAWSkey.pem"`
- Connect to your instance using its Public DNS:  
`ssh -i "MyFirstAWSkey.pem" ubuntu@ec2-18-208-165-123.compute-1.amazonaws.com`

**Example:**  
`ssh -i "MyFirstAWSkey.pem" ubuntu@ec2-18-208-165-123.compute-1.amazonaws.com`

**Note:** In most cases, the guessed username is correct. However, read your AMI usage instruction

## STEP 16 : Copy and paste the ssh line in the terminal

```
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
ubuntu@ip-172-31-85-127:~$
```

## STEP 17 : Run the command `sudo apt update`

```
see "man sudo_root" for details.
ubuntu@ip-172-31-85-127:~$ sudo apt update|
```

## STEP 18 : Run the command ` sudo apt-get install docker.io`

```
ubuntu@ip-172-31-85-127:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
 bridge-utils contained in dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
 ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-buildx
The following NEW packages will be installed:
 bridge-utils contained in dns-root-data dnsmasq-base docker.io pigz runc ubun
0 upgraded, 8 newly installed, 0 to remove and 58 not upgraded.
Need to get 88.1 MB of archives.
After this operation, 304 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

## STEP 20 : Run the command `sudo apt install nano`

```
ubuntu@ip-172-31-85-127:~$ sudo apt install nano
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nano is already the newest version (7.2-2ubuntu0.1).
nano set to manually installed.
```

## STEP 22 : Move to the loaded directory

```
ubuntu@ip-172-31-85-127:~$ ls
AWSexampleWeb
ubuntu@ip-172-31-85-127:~$ cd AWSexampleWeb/
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ |
```

## STEP 24 : Run the command `sudo docker build -t mywebapp`

```
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ sudo docker build -t mywebapp .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
 Install the buildx component to build images with BuildKit:
 https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 63.49kB
Step 1/2 : FROM nginx:alpine
alpine: Pulling from library/nginx
da9db072f522: Pull complete
e10e486de1ab: Pull complete
af9c0e53c5a4: Pull complete
b2eb2d8af93a: Pull complete
e351ee5ec3d4: Pull complete
fbfb7d28be71: Pull complete
471412c08d15: Pull complete
a2eb5282fbec: Pull complete
Digest: sha256:41523187cf7d7a2f2677a80609d9caa14388bf5c1fbca9c410ba3de602aaaab4
Status: Downloaded newer image for nginx:alpine
--> 91ca84b4f577
Step 2/2 : COPY . /usr/share/nginx/html
```

## STEP 19 : Run the command `sudo apt install git`

```
ubuntu@ip-172-31-85-127:~$ sudo apt install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.1).
git set to manually installed.
```

## STEP 21 : Run the command `git clone`

<https://github.com/RahZero0/AWSexampleWeb.git>

```
ubuntu@ip-172-31-85-127:~$ git clone https://github.com/RahZero0/AWSexampleWeb.git
Cloning into 'AWSexampleWeb'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```

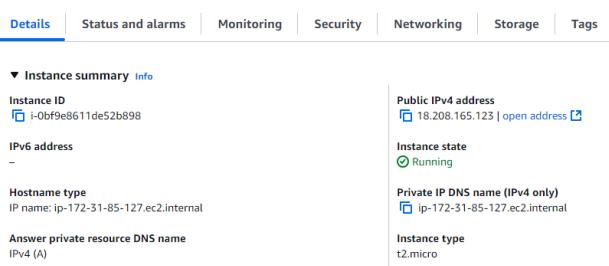
## STEP 23 : Create the Dockerfile and write the below

```
GNU nano 7.2
FROM nginx:alpine
COPY . /usr/share/nginx/html
```

## STEP 25 : Run the command `sudo docker run -d -p 80:80 mywebapp`

```
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ sudo docker run -d -p 80:80 mywebapp
a14ae9e45578b59c5300da58a8fa5bac5908ac8e13b259cadcf41d603f43941e
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$
```

## STEP 26 : Copy the public IP address



## STEP 27 : Go to the IP address



**STEP 28 :** Run the command `sudo docker ps`

```
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
a14ae9e45578 mywebapp "/docker-entrypoint..." 2 minutes ago Up 2 minutes 0.0.0.0:80->80/tcp, :::80->80/tcp pensive_allen
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ |
```

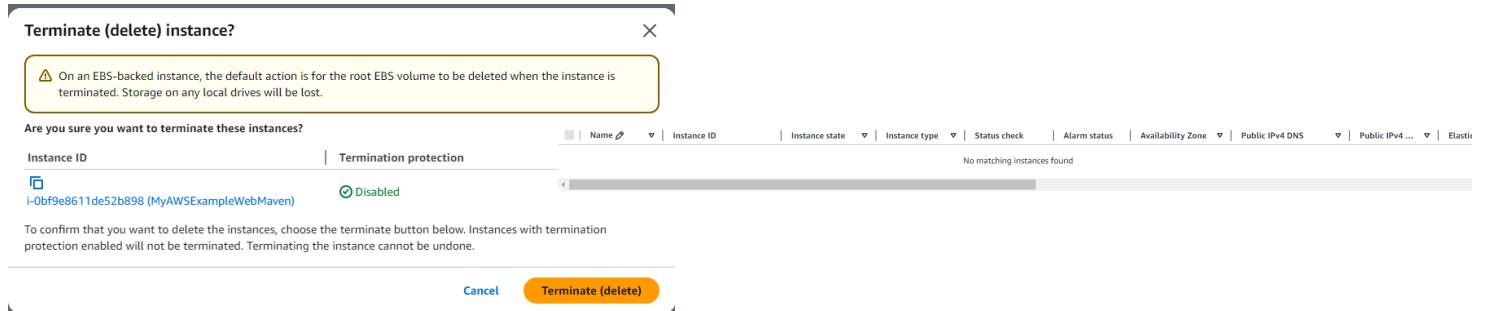
**STEP 29 :** Run the command `sudo docker stop <container\_id>`

```
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ sudo docker stop a14ae9e45578
a14ae9e45578
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ |
```

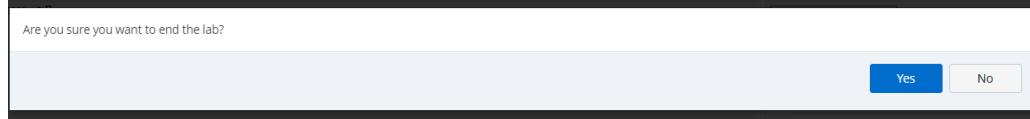
```
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
ubuntu@ip-172-31-85-127:~/AWSexampleWeb$ |
```

**STEP 30 :** Terminate the instance

**STEP 31 :** Refresh the page to see if instance is **terminated**



**STEP 32 :** Close the lab



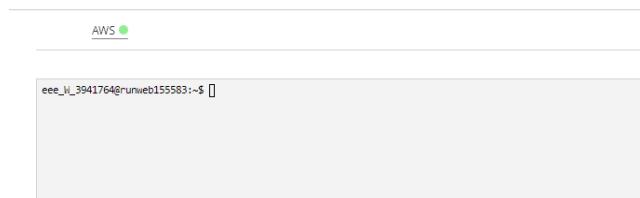
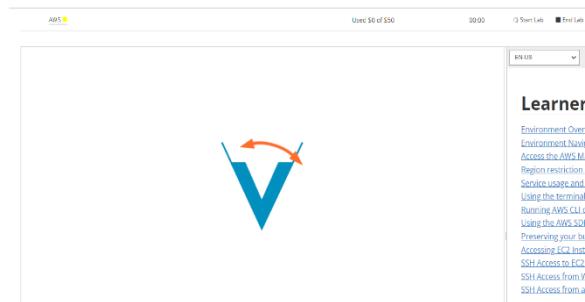
**STEP 33 :** Check for red dot

AWS

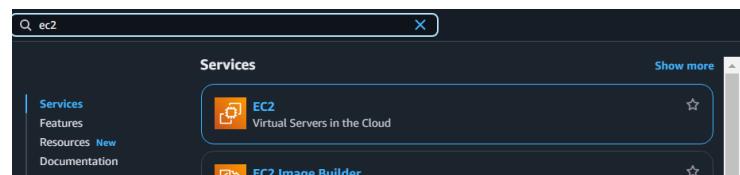
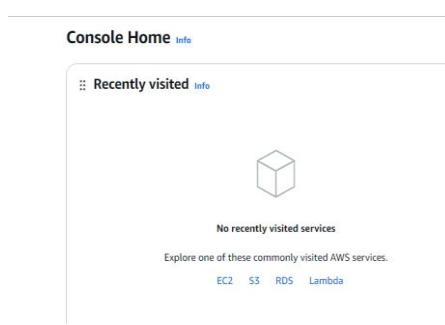
# DEPLOYING MAVEN WEB PROJECT DEPLOYMENT IN THE AWS CLOUD USING EC2 INSTANCE

**STEP 1 :** Start the lab and wait for the button to turn green

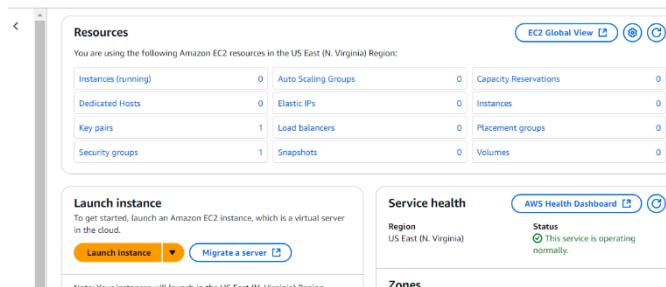
**STEP 2 :** As soon as it turns green click on AWS



**STEP 3 :** As you enter the lab search for **EC2** instance



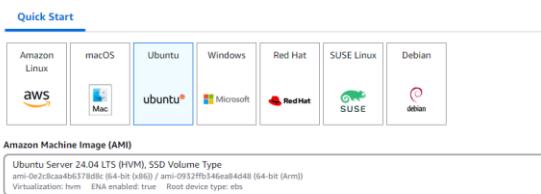
**STEP 4 :** Wait for the screen to load and then click on **Launch Instance**



**STEP 5 :** Type an appropriate name



**STEP 6 :** Select Ubuntu Free Tier



**STEP 7 :** Select 64 bit architecture



## STEP 8 : Select t2 free instance type

**Instance type** [Info](#) | [Get advice](#)

**Instance type**

t2.micro  
Family: t2 1 vCPU 1 GiB Memory Current generation: true  
On-Demand Windows base pricing: 0.0162 USD per Hour On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour  
On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0.026 USD per Hour

**Additional costs apply for AMIs with pre-installed software**

## STEP 9 : Create your key by clicking RSA

**Create key pair**

**Key pair name**  
Key pairs allow you to connect to your instance securely.  
  
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type**  
 RSA RSA encrypted private and public key pair  
 ED25519 ED25519 encrypted private and public key pair

**Private key file format**  
 .pem For use with OpenSSH  
 .ppk For use with PuTTY

**Note:** When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

[Cancel](#) [Create key pair](#)

## STEP 10 : In network settings allow both **HTTP – HTTPS** traffic

**Firewall (security groups)** [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

[Create security group](#) [Select existing security group](#)

We'll create a new security group called 'launch-wizard-1' with the following rules:

- Allow SSH traffic from anywhere
- Allow HTTPS traffic from the internet To set up an endpoint, for example when creating a web server
- Allow HTTP traffic from the internet To set up an endpoint, for example when creating a web server

**Note:** Rules of source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting known IP addresses only.

## STEP 11 : In storage settings select either **8GB | 10GB** which is still free tier

**Configure storage** [Info](#)

1x 8 GiB gp3 Root volume (Not encrypted)

**Note:** Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage per month.

[Add new volume](#)

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance

## STEP 12 : Click on Launch Instance

[Cancel](#) [Launch instance](#) [Preview code](#)

## STEP 13 : Verify instance has launched

**Launching instance**  
Creating security groups 27%

**Details**

**Success**  
Successfully initiated launch of instance ([i-0bf9e8611de52b898](#))

Please wait while we launch your instance. Do not close your browser while this is loading.

**Launch log**

## STEP 14 : Wait for the instance to load

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
MyAWSExampleWebMaven	i-0bf9e8611de52b898	Running	t2.micro	Initializing	View alarms +	us-east-1c

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
MyAWSExampleWebMaven	i-0bf9e8611de52b898	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1c

## STEP 15 : SSH Connect to the instance

**Instance ID**  
i-0bf9e8611de52b898 (MyAWSExampleWebMaven)

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is MyFirstAWSkey.pem.
- Run this command, if necessary, to ensure your key is not publicly viewable:  
`chmod 400 "MyFirstAWSkey.pem"`
- Connect to your instance using its Public DNS:  
`ssh -i "MyFirstAWSkey.pem" ubuntu@ec2-18-208-165-123.compute-1.amazonaws.com`

**Example:**  
`ssh -i "MyFirstAWSkey.pem" ubuntu@ec2-18-208-165-123.compute-1.amazonaws.com`

**Note:** In most cases, the guessed username is correct. However, read your AMI usage instruction

## STEP 16 : Copy and paste the ssh line in the terminal

```
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-85-127:~$
```

## STEP 17 : Run the command `sudo apt update`

see "man sudo\_root" for details.  
ubuntu@ip-172-31-85-127:~\$ sudo apt update|

## STEP 18 : Run the command ` sudo apt-get install docker.io`

```
ubuntu@ip-172-31-85-127:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
 bridge-utils contained in dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
 ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-buildx
The following NEW packages will be installed:
 bridge-utils contained in dns-root-data dnsmasq-base docker.io pigz runc ubun
0 upgraded, 8 newly installed, 0 to remove and 58 not upgraded.
Need to get 88.1 MB of archives.
After this operation, 304 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

## STEP 20 : Run the command `sudo apt install nano`

```
ubuntu@ip-172-31-85-127:~$ sudo apt install nano
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nano is already the newest version (7.2-2ubuntu0.1).
nano set to manually installed.
```

## STEP 19 : Run the command `sudo apt install git`

ubuntu@ip-172-31-85-127:~\$ sudo apt install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.1).
git set to manually installed.

## STEP 22 : Create a Dockerfile and write the contents

```
GNU nano 7.2
FROM tomcat:9-jdk11
COPY target/*.war /usr/local/tomcat/webapps/
```

## STEP 21 : Run the command `git clone <https://github.com/RahZero0/Maven-Web-Project.git>`

```
ubuntu@ip-172-31-85-127:~$ git clone https://github.com/RahZero0/Maven-Web-Project.git
Cloning into 'Maven-Web-Project'...
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (21/21), done.
Receiving objects: 100% (33/33), 6.06 KiB | 3.03 MiB/s, done.
remote: Total 33 (delta 1), reused 33 (delta 1), pack-reused 0 (from 0)
Resolving deltas: 100% (1/1), done.
```

## STEP 23 : Run the command `sudo docker build -t mavenwebproject`

```
ubuntu@ip-172-31-85-127:~/Maven-Web-Project$ sudo docker build -t mavenwebproject .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/
Sending build context to Docker daemon 110.6kB
Step 1/2 : FROM tomcat:9-jdk11
9-jdk11: Pulling from library/tomcat
d4e4b265507a: Pull complete
4d8b25a6d227: Pull complete
5a7ece70ec66: Pull complete
623auff914ca: Pull complete
6d3bf2e80222: Pull complete
5ea880ff7f1fd3aa4729adaf348a675daca772453416a7cald60bc67b
Status: Downloaded newer image for tomcat:9-jdk11
--> 1b54e6b2b9a5
Step 2/2 : COPY target/*.war /usr/local/tomcat/webapps/
--> d01e374620b5
```

## STEP 24 : Run the command `sudo docker run -d -p 9090:8080 mavenwebproject`

```
ubuntu@ip-172-31-85-127:~/Maven-Web-Project$ sudo docker run -d -p 9090:8080 mavenwebproject
5c8d2e0bf2cd5d1379ae538bed5e2301847e4b8862cef54091706ffc6e3318203
ubuntu@ip-172-31-85-127:~/Maven-Web-Project$
```

## STEP 25 : Check if 9090 is in the security ports

Name	Security group rule ID	Port range	Protocol	Source	Securi
-	sgr-01333c19bc34ec381	443	TCP	0.0.0.0/0	<a href="#">laun</a>
-	sgr-0a569b2247b03d9f3	80	TCP	0.0.0.0/0	<a href="#">laun</a>
-	sgr-0060154dd0ff13992	22	TCP	0.0.0.0/0	<a href="#">laun</a>

## STEP 26 : Click on security groups

Inbound rules	Info	Protocol	Port range	Source	Info
Security group rule ID	Type	Info	Protocol	Port range	Source
sgr-01333c19bc34ec381	HTTPS	<a href="#">TCP</a>	443	<a href="#">Cu...</a>	<a href="#">Q</a> 0.0.0.0/ <a href="#">X</a>
sgr-0a569b2247b03d9f3	HTTP	<a href="#">TCP</a>	80	<a href="#">Cu...</a>	<a href="#">Q</a> 0.0.0.0/ <a href="#">X</a>
sgr-0060154dd0ff13992	SSH	<a href="#">TCP</a>	22	<a href="#">Cu...</a>	<a href="#">Q</a> 0.0.0.0/ <a href="#">X</a>
-	Custom TCP	<a href="#">TCP</a>	9090	<a href="#">An...</a>	<a href="#">Q</a> 0.0.0.0/ <a href="#">X</a>

Inbound security group rules successfully modified on security group sgr-0b740d2ce8fd8af (launch-wizard-1)

Details

g-0b740d2ce8fd8af - launch-wizard-1

Custom TCP

## ▼ Security details

### IAM Role

### Security groups

[sg-0b740d2ce8fd8af \(launch-wizard-1\)](#)

## STEP 28 : Check the url for the MavenWebProject

Not secure 18.208.165.123:9090/SampleMavenWebProject

Hello Welcome to SE Lab 2024

## STEP 29 : Run the command `sudo docker ps`

```
ubuntu@ip-172-31-85-127:~/Maven-Web-Project$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
5c0d2e0bf2cd mavenwebproject "catalina.sh run" 7 minutes ago Up 7 minutes 0.0.0.0:9090->8080/tcp, :::9090->8080/tcp gifted_carver
```

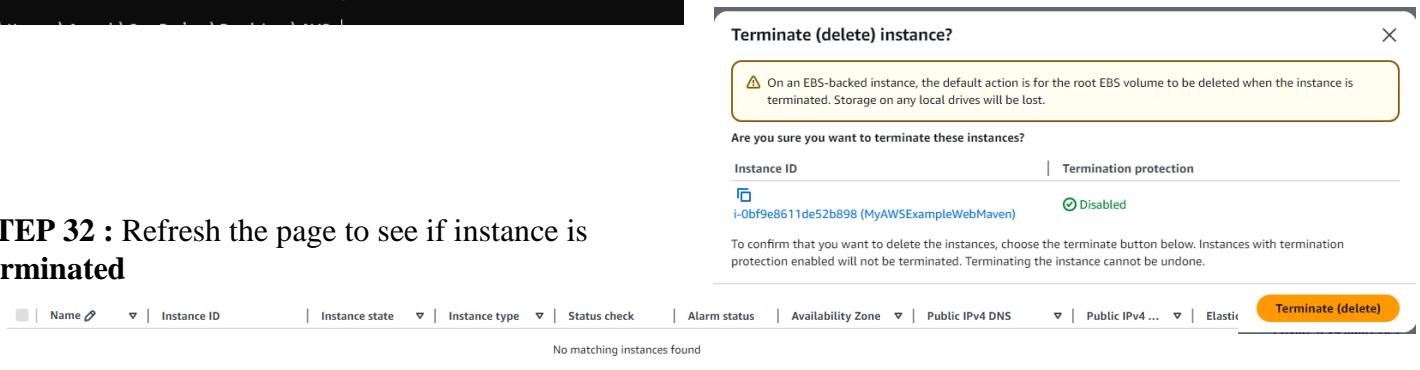
  

```
ubuntu@ip-172-31-85-127:~/Maven-Web-Project$ sudo docker stop 5c0d2e0bf2cd
ubuntu@ip-172-31-85-127:~/Maven-Web-Project$ |
```

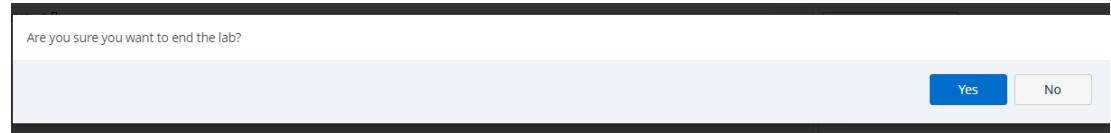
## STEP 30 : Open

```
ubuntu@ip-172-31-85-127:~/Maven-Web-Project$ exit
logout
Connection to ec2-18-208-165-123.compute-1.amazonaws.com closed.
```

## STEP 31 : Terminate the instance



## STEP 33 : Close the lab



## STEP 34 : Check for red dot

AWS

**Conclusion :** Thus we were able to use **AWS services** and learn more about **EC2** and do projects in them.