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

LAB RECORD

SOFTWARE ENGINEERING LAB

B.Tech. III YEAR I SEM (KR20)

ACADEMIC YEAR 2022-23

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTE)
Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad
Department of Computer Science & Engineering

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.

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTE)
Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad
Department of Computer Science & Engineering

Daily Laboratory Assessment Sheet

Name of the Lab: Name of the Student:

Class: HT.No:

S.No.	Name of the Experiment	Date	Observation Marks (5M)	Record Marks (5M)	Viva Voice Marks(5M)	Total Marks (15M)	Signature of Faculty
	TOTAL						

Faculty Incharge

INDEX

S.NO	CONTENTS	PAGENO
I	Vision/Mission /PEOs/POs/PSOs	1
II	Syllabus	7
III	Course outcomes, CO-PO Mapping	9
IV	Software and Hardware requirements	11
Exp no.	**List of experiments**	
1	**Software Requirement Specification** a) Development of a problem statement. b) Preparation of Software Requirement Specification Document, Design Documents, and related documents. c) c) Study and use any Design phase CASE tool	12
2	**Working with Git and GitHub** a) Git configuration b) Git commands – help,pwd, mkdir, notepad, ls, init, ls, status, add, commit, log, diff, staged, log --oneline, revert branch, checkout, merge, reset c) GitHub commands – 1. Creating repository 2. Using GitBash for following commands remote, remote -v, push to 3. Push a sample project from local to remote repository 4. Creating SSH key in GitHub 5. Using GitBash for SSH- Keygen, clone, ls – ltr, push to push asample project using ssh key from local to remote repository	52
3	**Working with Maven in Eclipse** a) Create a Maven java project and run it in Eclipse environment b) Create a Maven Web Project and run it in Eclipse environment(by adding dependencies) c) Push the above two projects into GitHub.	69
4	**Working with Jenkins** a) Jenkins introduction b) Jenkins Global configurations5. and installing additional 5 pipeline plugins. c) Create a CI/CD pipeline using Jenkins for a sample Maven Java project d) Create a CI/CD pipeline using Jenkins for a sample Maven Web project.	99

5	**Working with Docker** a) Docker introduction b) Docker CLI commands c) Pulling an existing image from Docker Hub and running the image as container using run command d) Creating own image of Tomee and running as container, pushing the image into Docker Hub e) Creating customized container f) Use Docker compose for wordpress with DockerFile g) Use Docker compose with yaml file. h) Pulling and running the services, Nagios and Nginx	127
6	**Working with AWS** Create an Elastic Cloud Computing (EC2) instance of Ubuntu. Create an Elastic Cloud Computing (EC2) instance of Amazon Linux2 AMI	145

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 & Engineering

Vision of the Institution:

To be the fountainhead 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

Department of Computer Science & Engineering

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 benefits 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

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.

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.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates will have successful careers in computer related engineering fields or will be able to successfully pursue advanced higher education degrees.

PEO2: Graduates will try and provide solutions to challenging problems in their profession by applying computer engineering principles.

PEO3: Graduates will engage in life-long learning and professional development by rapidly adapting changing work environment.

PEO4: Graduates will communicate effectively, work collaboratively and exhibit high levels of professionalism and ethical responsibility.

B. Tech. in COMPUTER SCIENCE AND ENGINEERING
III Year I Semester Course Syllabus (KR20)
SOFTWARE ENGINEERING LAB (CS505PC)

L	T	P	C
0	0	3	1.5

Prerequisites/ Corequisites:

1. CS203ES – Programming for Problem Solving Course
2. CS403PC – Operating Systems Course
3. CS502PC – Software Engineering Course

Course Objectives: The course will help to

1. Develop the process of problem statements.
2. Understand the process of development of Software Requirement Specifications.
3. Have hands-on experience in Design, develop and testing various modules in a project.
4. Understand the usage of GitHub and Jenkins.
5. Devise the deployment of a project in AWS cloud using Docker and Kubernetes.

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

1. Outline to translate end-user requirements into system and software requirements.
2. Illustrate a high-level design of the system from the software requirements.
3. Apply use case tools in the design phase.
4. Use Jenkins to build projects.
5. Devise a project in AWS cloud using Dockers and Kubernetes.

List of Experiments:

Execute the following exercises for any one project given in the list of sample projects:

1) Development of problem statements.
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 static pages of the project and committing using Git and GitHub
5) Building the project in Jenkins
6) Deploying the project in AWS cloud using Docker and Kubernetes
7) Develop test cases for unit testing and integration testing

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. Effective DevOps: Building A Culture of Collaboration, Affinity, and Tooling at Scale, 2018.
2. Cloud Native DevOps with Kubernetes by John Arundel, 2016.

Course Objectives:

- Understand the process of development of a problem statement.
- Understand the process of development of Software Requirement Specifications
- Understand the process of development of Design documents and testing phase-related documents.
- Understand the usage of GitHub and Jenkins
- Understand the deployment of projects in the AWS cloud using Docker and Kubernetes.

Course Outcomes:

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

CO 1: Outline to translate end-user requirements into system and software requirements

CO 2: Illustrate a high-level design of the system from the software requirements

CO 3: Use case tools in the design phase of application development.

CO 4: Use Jenkins to build the project

CO 5: Setup a project in the AWS cloud using Dockers and Kubernetes.

CO-PO MAPPING:

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

CO-PSO MAPPING:

	PSO-1	PSO-2
CO1	2	
CO2	2	1
CO3		2
CO4		2
CO5		2

Software and Hardware Requirements

1. **Software Tools**
    a. **Star UML**
    b. **Java 11**
    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)**

2. **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.**

Software Requirement Specification

1.BOOK BANK

AIM:

To create a system to perform book bank operations using Star UML tool.

PROBLEM STATEMENT:

A Book Bank lends books and magazines to members, who are registered in the system. Also, it handles the purchase of new titles for the Book Bank. Popular titles are brought in multiple copies. Old books and magazines are removed when they are out of date or poor in condition. A member can reserve a book or magazine that is not currently available in the book bank so that when it is returned or purchased by the book bank, that person is notified. The book bank can easily create, replace and delete information about the tiles, members, loans, and reservations from the system.

INTRODUCTION:

Book Bank is the interface between the students and the Librarian. It aims to improve the efficiency in the Issue of books or magazines and reduce the complexities involved in it to the maximum possible extent.

OBJECTIVE:

This system provides an easy solution for the customers to lend books without going to the library.

PURPOSE:

If the entire process of 'Issue of Books or Magazines' is done manually, it would take several months for the books or magazines to reach the applicant. Considering that the demand for Book banks is increasing every year, an Automated System becomes essential to meet the demand. So this system uses several programming and database techniques to elucidate the work involved in this process. The system has been carefully verified and validated to satisfy.

SCOPE:

The System provides an online interface to the user where they can fill in their details and submit the necessary documents (maybe by scanning). The authority concerned with the issue of books can use this system to reduce his workload and process the application speedily.

MODULES:

1.Login

1.Registration for new user

3.Enter students details

4.Check student information

5.Pay required fee

6.Select the book to be issued

7. Return to the book

SOFTWARE REQUIREMENTS:

1. Front End Client - The Student and Librarian online interface is built using Visual studio.
2. Backend - Oracle 11 g database

HARDWARE REQUIREMENTS:

The server is directly connected to the client systems. The client systems have access to the database in the server.

FUNCTIONAL REQUIREMENTS:

The member should be authenticated by means of unique login id and password. The availability of books requested must be prompted to the user through e-mail or sms notifications.

NON-FUNCTIONAL REQUIREMENTS:

Security:

- The system must automatically log out all customers after a period of inactivity.
- The system should not leave any cookies on the customer's computer containing the user's password.
- The system's back end servers shall only be accessible to authenticated administrators.
- Sensitive data will be encrypted before being sent over insecure connections like the internet.

Reliability: The system provides storage of all databases on redundant computers with automatic switchover. The reliability of the overall program depends on the reliability of the separate components. The main pillar of reliability of the system is the backup of the database which is continuously maintained and updated to reflect the most recent changes. Thus the overall stability of the system depends on the stability of container and its underlying operating system.

Availability: The system should be available at all times, meaning the user can access it using aweb browser, only restricted by the down time of the server on which the system runs. In case of a of a hardware failure or database corruption, a replacement page will be shown. Also in case of a hardware failure or database corruption, backups of the database should be retrieved from the server and saved by the administrator. Then the service will be restarted.

Maintainability: A commercial database is used for maintaining the database and the application server takes care of the site. In case of a failure, a re-initialization of the program will

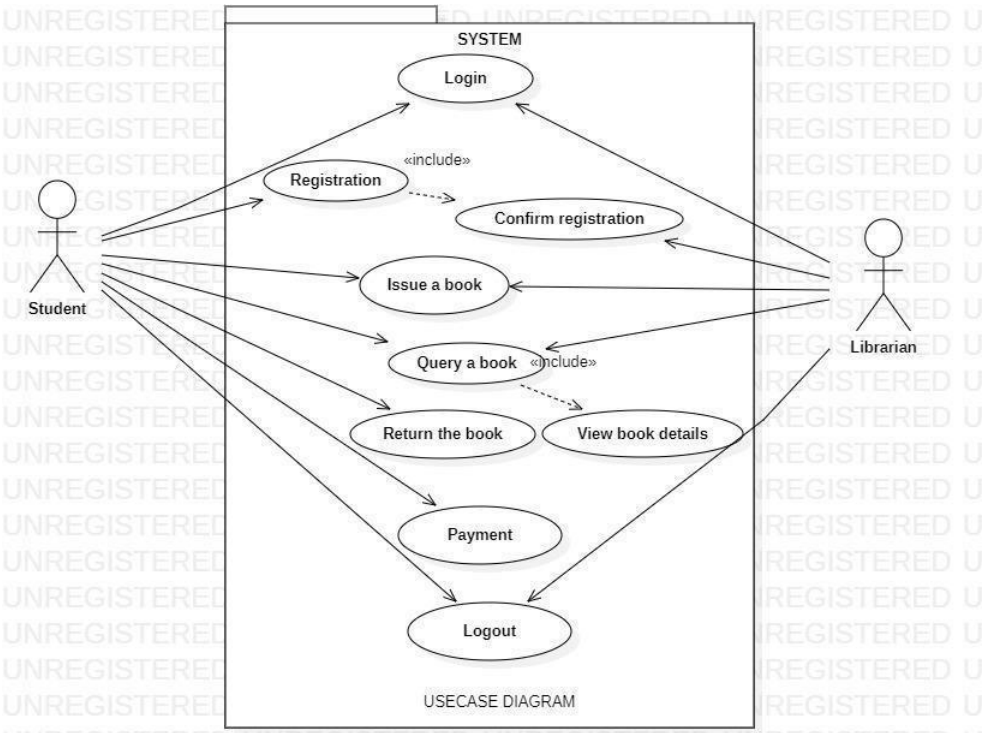
be done. Also the software design is being done with modularity in mind so that maintainability can be done efficiently.

Portability: The application is HTML and scripting language based. So The end-user part isfully portable and any system using any web browser should be able to use the features of the system, including any hardware platform that is available or will be available in the future. Anend-user is use this system on any OS; either it is Windows or Linux. The system shall run on PC, Laptops, and PDA etc.

UML DIAGRAMS:

USE-CASE:

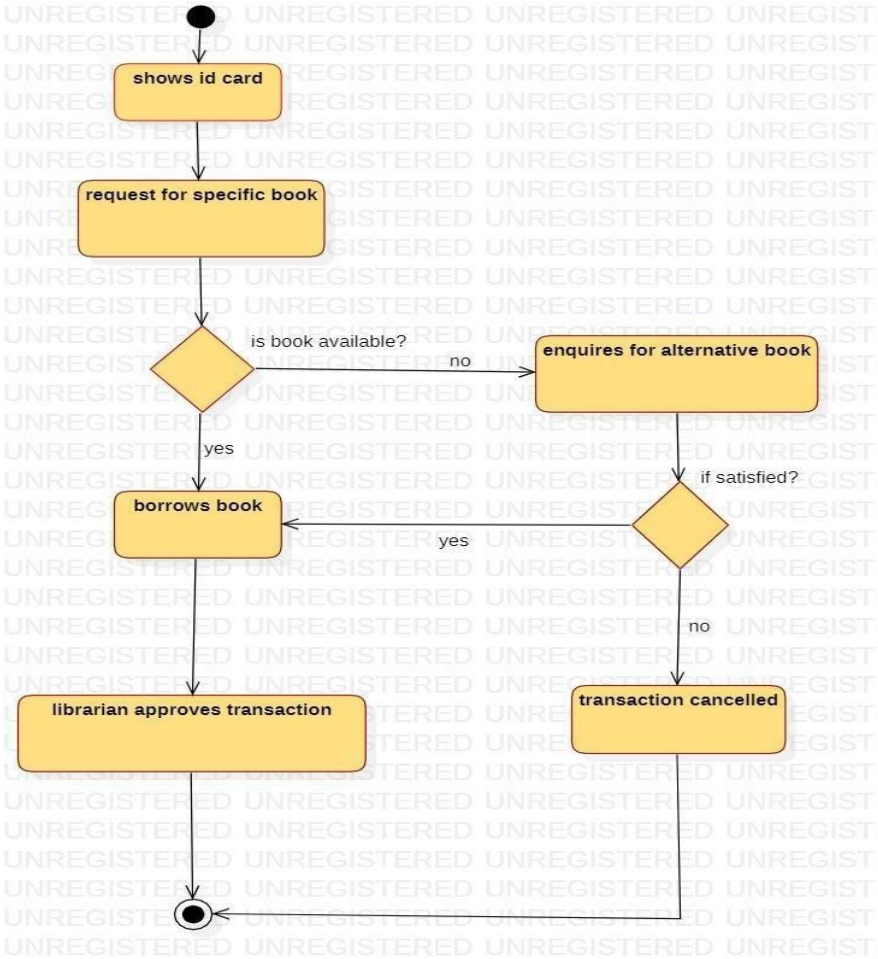
Use-Case is a list of actions or events. Steps typically define the interactions between a role and a system to achieve a goal. The use-case diagram consists of various functionality performed by actors like Student, Admin, book bank and DBA.



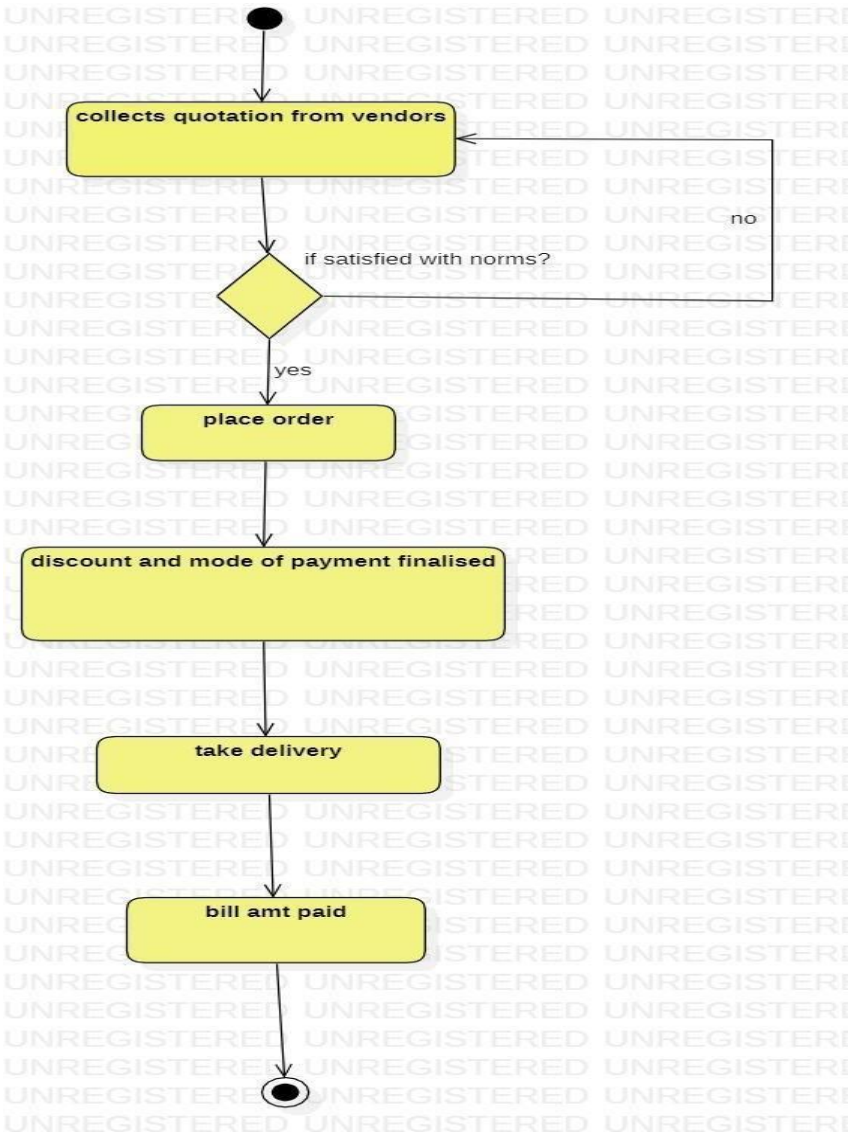
ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. Here in the activity diagram the student and admin login to the system and perform some main activity which is the main key element to the system.

ACTIVITY DIAGRAM BORROW BOOK

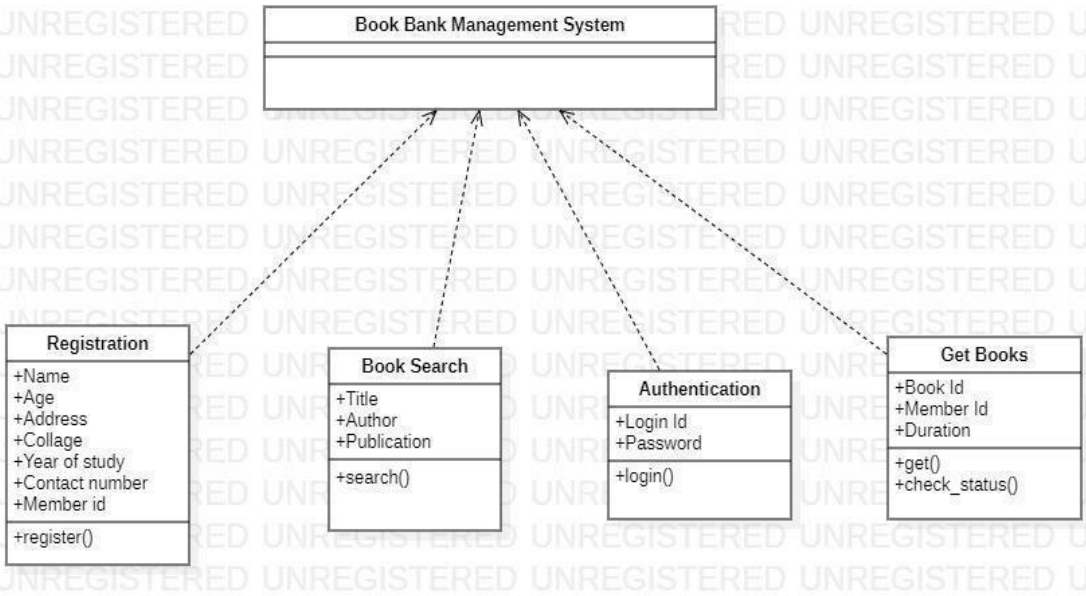


ACTIVITY DIAGRAM FOR BORROW BOOK:



CLASS DIAGRAM:

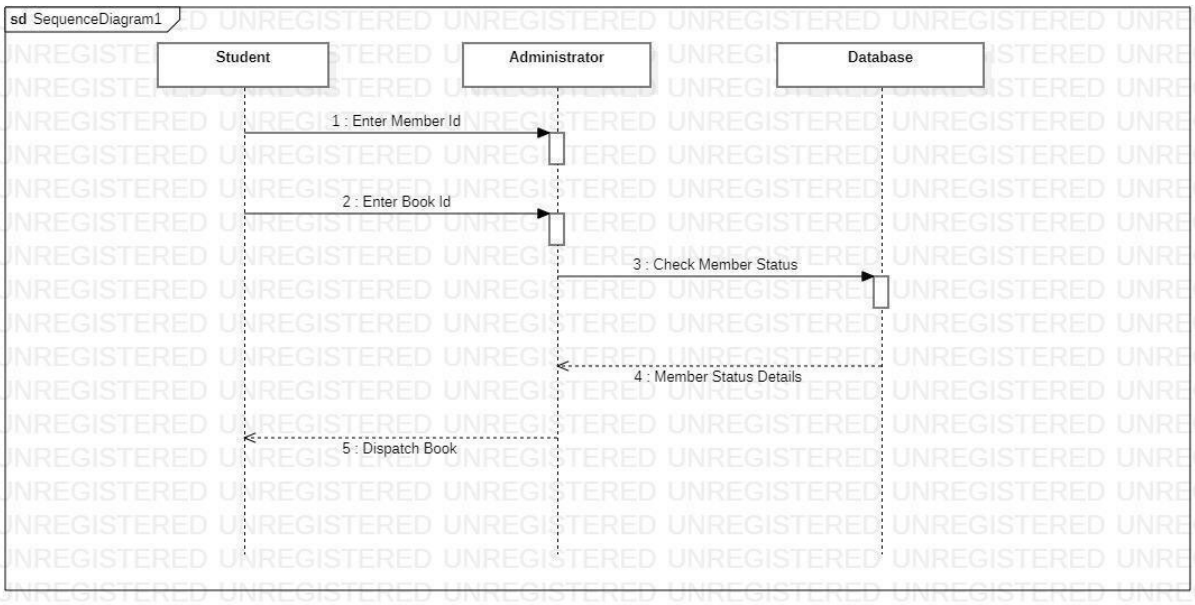
A class diagram in the unified modeling language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations and the relationships among objects. The book bank system makes use of the following classes: student, book bank, admin and DBA.



SEQUENCE DIAGRAM:

A sequence diagram represents the sequence and interactions of a given use case or scenario. Sequence diagrams capture most of the information about the system. It is also represented in order by which they occur and have the object in the system send messages to one another. Here the sequence starts with interaction between student and book bank followed by a database. Once the book has been selected the next half of the sequence starts between the book bank and admin followed by a database.

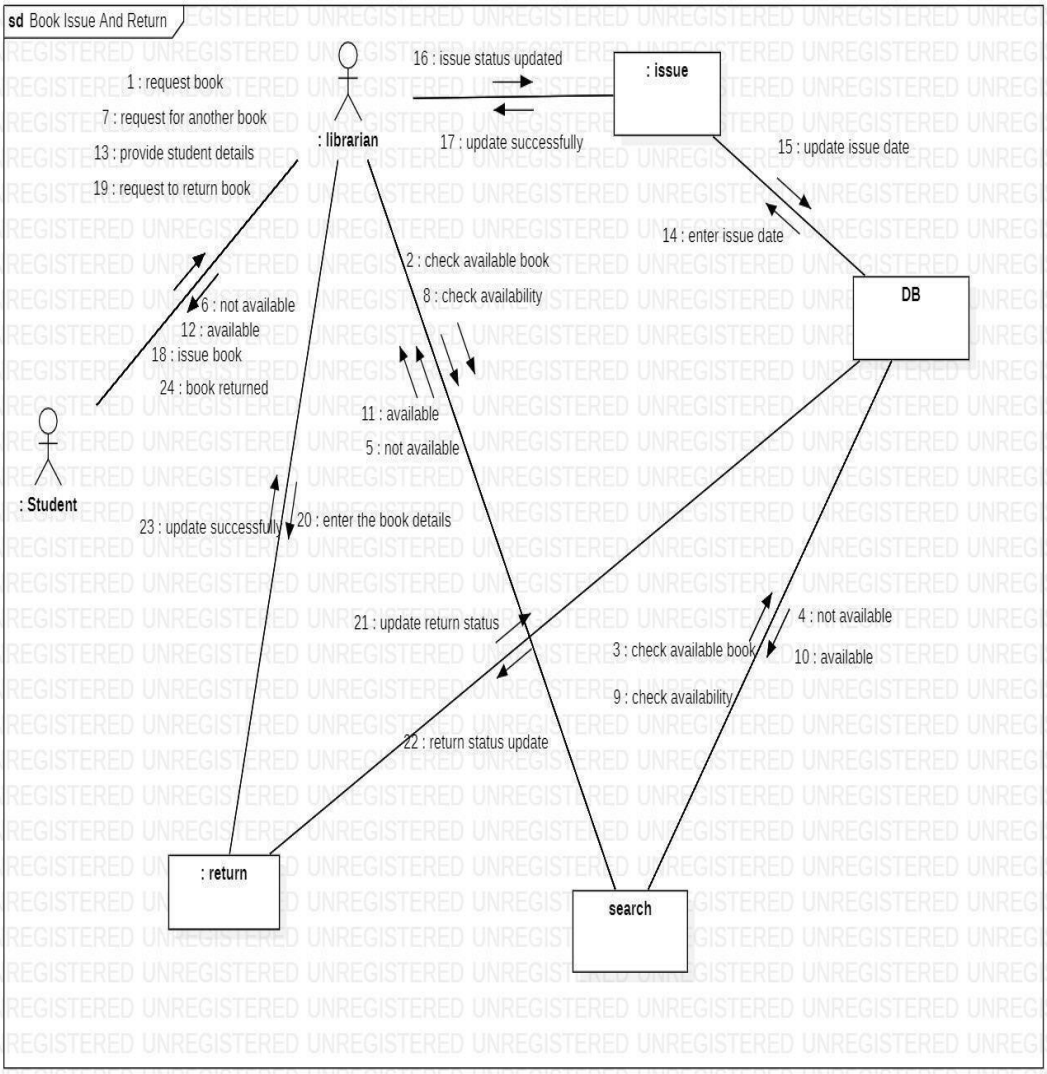
SEQUENCE DIAGRAM FOR BOOK ISSUE & RETURN:



COLLABORATION DIAGRAM:

Like sequence diagrams, collaboration diagrams are also called interaction diagrams.
Collaboration diagrams convey the same information as sequence diagrams but focus on the object roles instead of the times that messages are sent. Here the actions between various classes are represented by a number format for the case of identification.

COLLABORATION DIAGRAM FOR BOOK ISSUE :



2.Online Course Reservation System

AIM:

To develop an online course reservation system using StarUML Tool

PROBLEM STATEMENT:

Online course reservation system is used to choose the course through online by the students. They are provided with a catalog, they can choose the course. The catalog contains the detailed description about each course and also the availability of the course which helps the students to decide on their own. If a student is already registered, they can log in. Else, they have to fill up the registration form provided by the administrator. After registering, the student can log in to the system. Then the students select the college and courses. If the course is available, the student can proceed to the payment step, where the course fee can be paid online. After paying the fee, a receipt stating the details of the student and course they selected is provided by the administrator. Then the student successfully logs into the system.

INTRODUCTION:

This Online Course Registration System is a web-based program aimed to make easier and more convenient the class registration process, a hassle through which students go every semester. Online Course Registration System attempts to alleviate these hassles by providing several services to students through the internet. Online Course Registration System provides a way to search for classes without having to open a course catalog, a way to “shop” around and view various possible schedules, and finally, officially register for the chosen classes.

OBJECTIVE:

This system provides an enhanced interface, thereby helping the consumer to access manycourses available on this single platform.

PURPOSE:

The main purpose of the SRS document is to illustrate the user requirement of the stock maintenance system. This document is developed after a number of consultations with the client. This document will also act as a contract between the client and the developer in case of any dispute during the delivery.

SCOPE:

The scope of the Online Course Reservation System is to allow the customer’s to maintain their cart, i.e., they can increase or decrease the no. of courses as per their requirement. It is designed to track your performance. The customer can start or resume the course from where it was left.

MODULES:

1.Login
2.Registration for new user
3.Enter students details
4.Check student information
5.Select the college
6.Select the course
7.Online fees payment
8.Course reservation

SOFTWARE REQUIREMENTS:

1. Front End Client - The Student and Registrar online interface is built using JSP and HTML.

The administrator's local interface is built using Java.

2.Web Server - Tomcat Apache application server(Oracle Corporation).

3.Back End - Oracle 11g database.

HARDWARE REQUIREMENTS:

The server is directly connected to the client systems. The client systems have access to the database in the server. The various hardware constraints needed are Cpu, Processor speed, Coprocessor speed etc.

FUNCTIONAL REQUIREMENTS:

Technologies to be used :-

- HTML
- JSP
- Javascript
- Java

Tools to be used :-

- Eclipse IDE (Integrated Development Environment)
- Rational Rose tool (for developing UML Patterns

Software Interface :-

- Front End Client - The Student and Registrar online interface is built using JSP and HTML. The Administrators' local interface is built using Java.
- Web Server – Tomcat Apache application server (Oracle Corporation).
- Back End – Oracle 11g database.

Hardware Interface :-

The server is directly connected to the client systems. The client systems have access to the database on the server.

NON FUNCTIONAL REQUIREMENTS:

Security

1. The system must automatically log out all customers after a period of inactivity.

2. The system should not leave any cookies on the customer's computer containing the user's password.

3. The system's back end servers shall only be accessible to authenticated administrators.

4. Sensitive data will be encrypted before being sent over insecure connections like the internet.

Reliability

The system provides storage of all databases on redundant computers with automatic switchover. The reliability of the overall program depends on the reliability of the separate components. The main pillar of reliability of the system is the backup of the database which is continuously maintained and updated to reflect the most recent changes. Thus the overall stability of the system depends on the stability of the container and its underlying operating system.

Availability

The system should be available at all times, meaning the user can access it using a web browser, only restricted by the down time of the server on which the system runs. In case of a hardware failure or database corruption, a replacement page will be shown. Also in case of a hardware failure or database corruption, backups of the database should be retrieved from the server and saved by the administrator. Then the service will be restarted.

Maintainability:

A commercial database is used for maintaining the database and the application server takes care of the site. In case of a failure, a re-initialization of the program will be done. Also the software design is being done with modularity in mind so that maintainability can be done efficiently.

Portability:

The application is HTML and scripting language based. So The end-user part is fully portable and any system using any web browser should be able to use the features of the system, including any hardware platform that is available or will be available in the future. An end-user

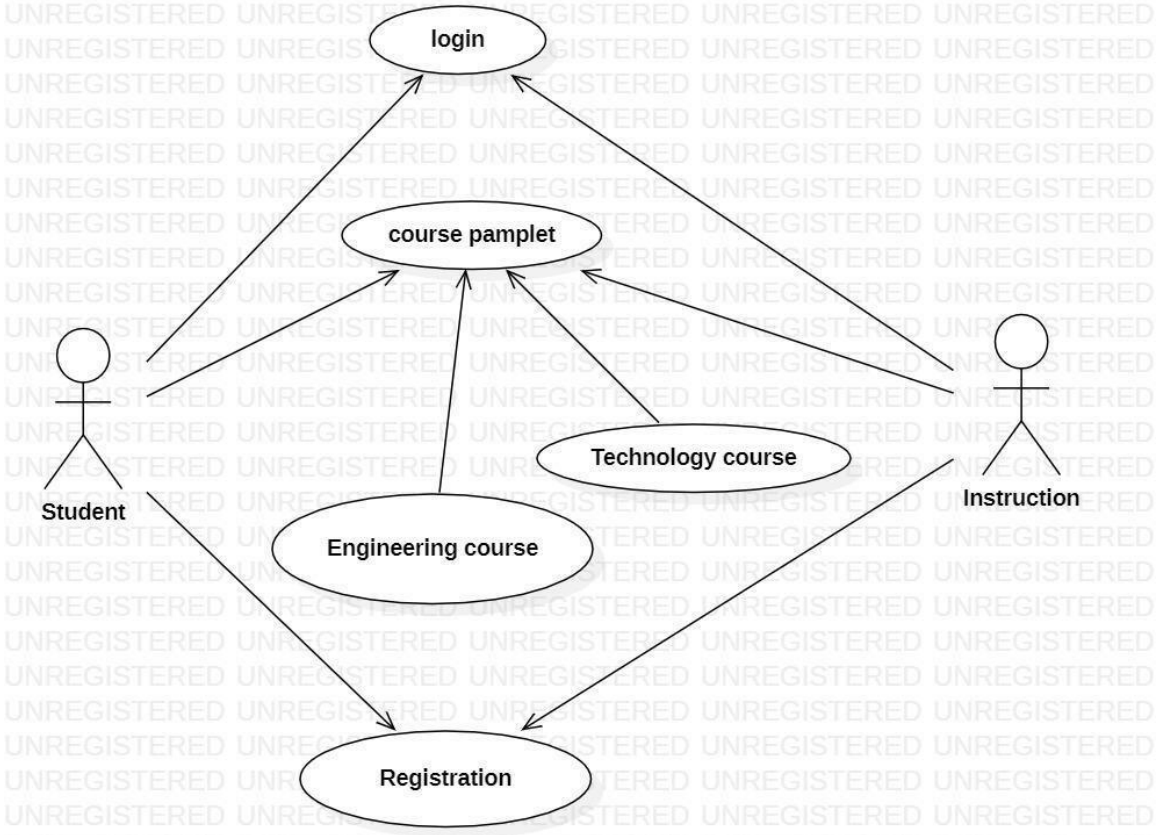
can use this system on any OS; either it is Windows or Linux. The system shall run on PC, Laptops, and PDA etc.

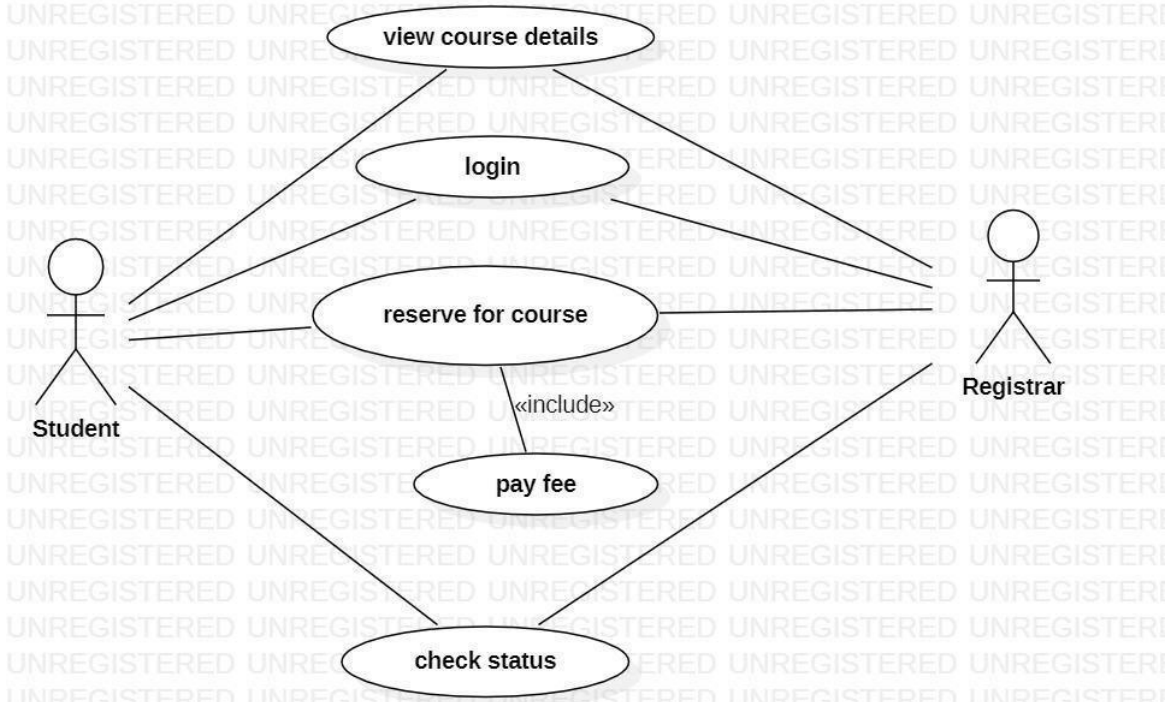
UML DIAGRAM:

Unified Modeling Language (UML) is a general purpose modelling language. The main aim of UML is to define a standard way to visualize the way a system has been designed**.**

USE-CASE DIAGRAMS:

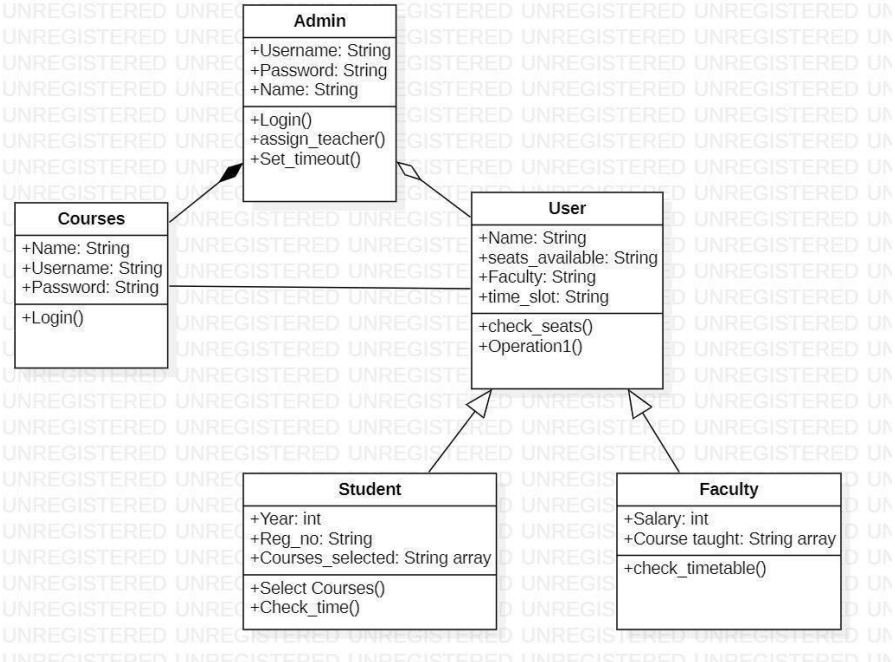
Use-Case is a list of actions or events. Steps typically define the interactions between a role and a system to achieve a goal. The use-case diagram consists of various functionality performed by actors like Student, Admin, book bank and DBA.





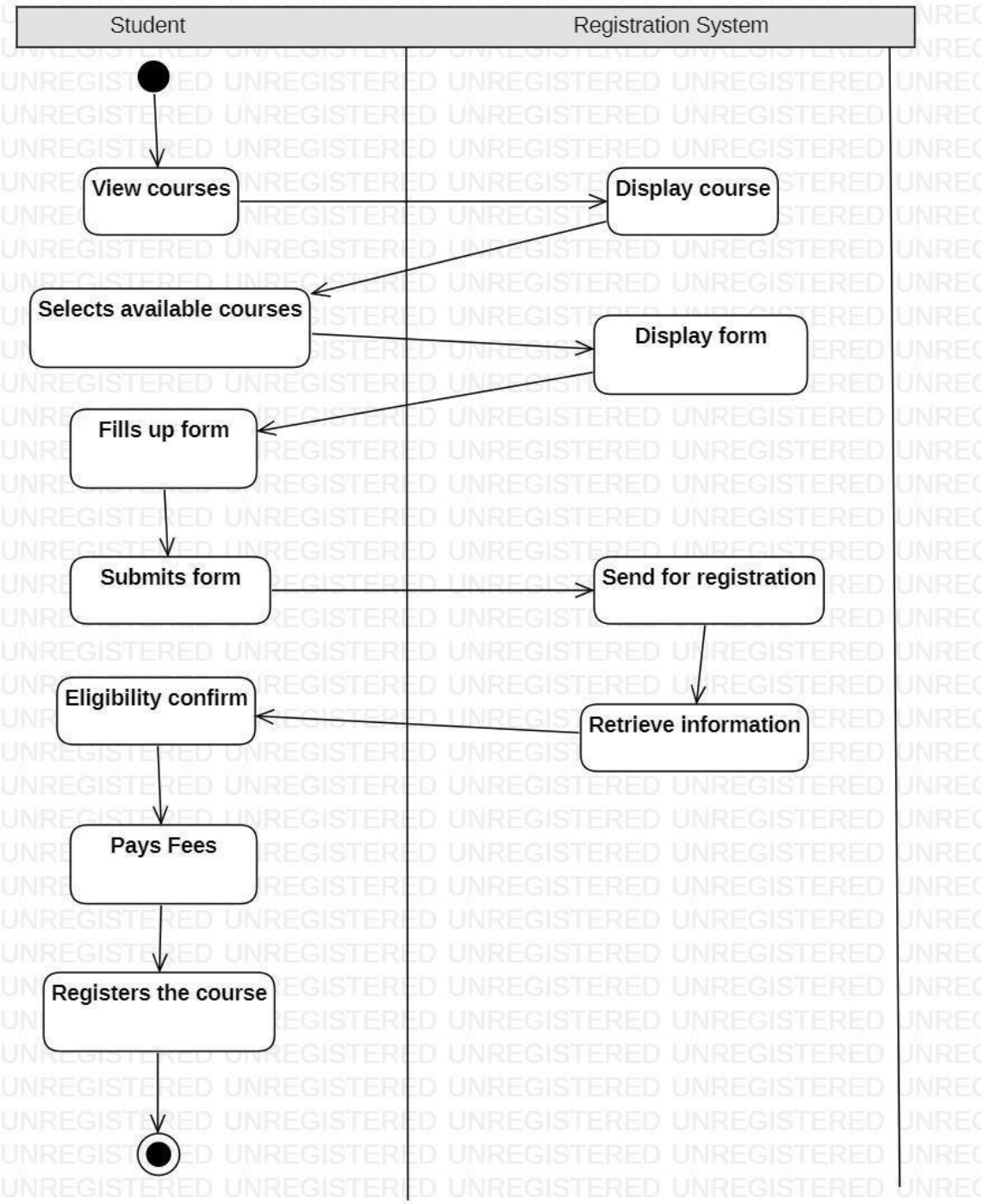
CLASS DIAGRAM:

A class diagram in the unified modeling language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations and the relationships among objects. The book bank system makes use of the following classes: student, book bank, admin and DBA.



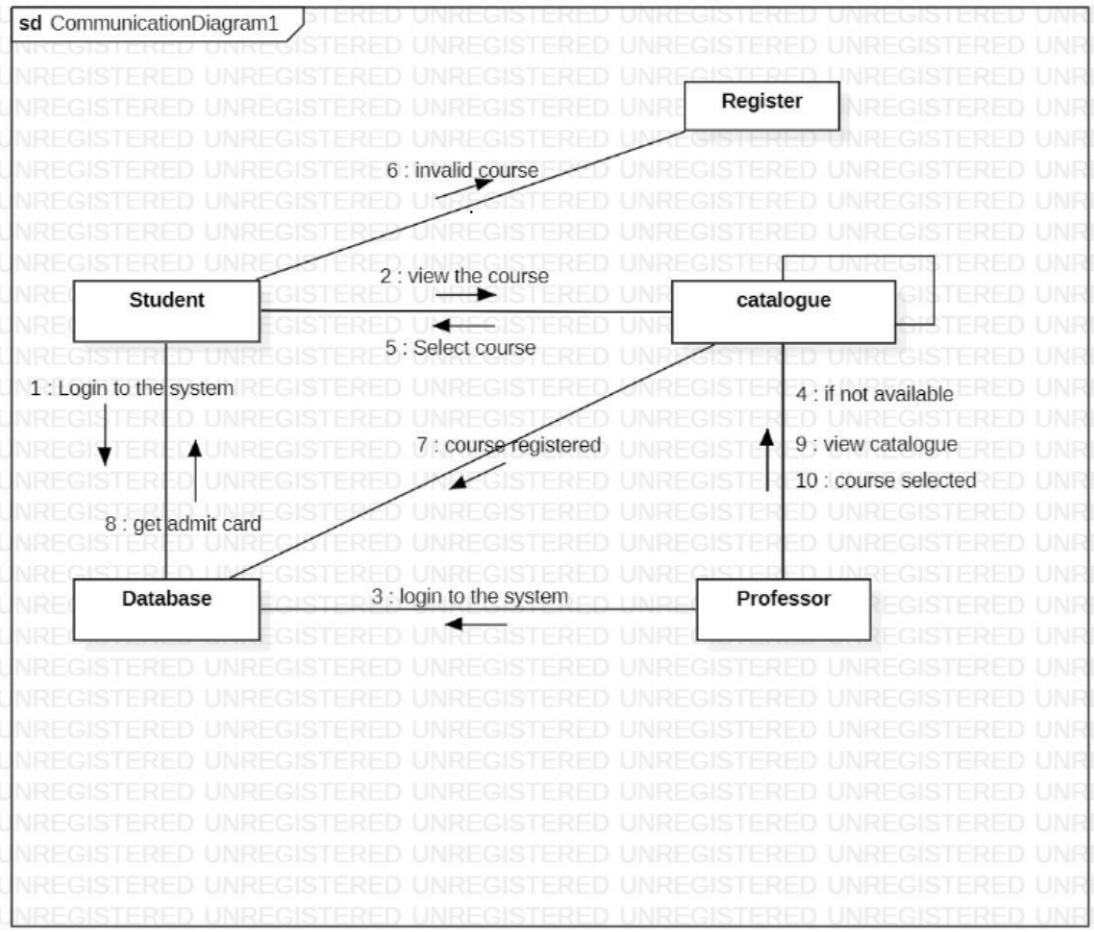
ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. Here in the activity diagram the student and admin login to the system and perform some main activity which is the main key element to the system.



COLLABORATION DIAGRAM:

Like sequence diagrams, collaboration diagrams are also called interaction diagrams. Collaboration diagrams convey the same information as sequence diagrams but focus on the object roles instead of the times that messages are sent. Here the actions between various classes are represented by a number format for the case of identification.



3.E-TICKETING

AIM:

To develop an E-TICKETING SYSTEM using star UML tool .

PROBLEM STATEMENT:

Our project is carried out to develop software for online reservation or ticket booking. This software has various implementations in our day to day lives such as movies , railway tickets ,airways , bus ,etc.) .This system has various options like reservation ,cancellation ,and view details about available seats , you can also select your desired seats if available. Our project mainly simulates the role of ticket booking officer in a computerized way.

INTRODUCTION:

The manual system of ticket reservation takes more time and the number of reservations per day is limited . to increase the efficiency of the process ,we go for online ticket reservation system .This system supports online ticket booking

OBJECTIVE:

This system provides an easy solution for booking the tickets without going to the stations or theaters and there will be no need to wait or stand in queues for reserving a ticket.

PURPOSE:

If the entire process of reservation is done in a manual manner it would take several months to reach the applicant .Considering the fact that number of users are increasing every year , an automated system must be essential to meet the demand. So this system uses several programming and database techniques to elucidate the work involved in this process as this is the matter of national security ,the system has been carefully verified and validated in order to satisfy it.

SCOPE:

The system provides an online interface to users where they can fill in their personal details and submit the necessary documents (maybe by scanning).

1.The authority concerned with the issue of workload and slow processing can use this system to reduce workload and process the application in a speedy manner.

2.Provides a communication platform between user and administrator.

3.User will come to know their status of application and the date in which they must submit themselves for manual document verification.

SOFTWARE REQUIREMENTS:

1. Front End Client - The passenger and System online interface is built using JSP and HTML. The Administrators' local interface is built using Java.
2. Web Server – Apache Tomcat Server (Oracle Corporation)
3. Back End - Oracle 11g database

HARDWARE REQUIREMENTS:

The server is directly connected to the client systems. The client systems have access to the database in the server.

FUNCTIONAL REQUIREMENTS:

1. Registration: If a customer wants to book some seats, then he/she must be registered, unregistered users can't book the seats.

2.Login: Customer logins to the system by entering valid email id and password for the seats to be booked.

3.Search: The user can search their required shows and check for avail.

4.Login: Customer logins to the system by entering valid email id and password for the seats to be booked.

5.Search: The user can search their required shows and check for available seats at different theaters.

6.Select: Users are allowed to select the required number of seats at liked places.

7.Payment: Payment is done through net banking or debit/credit cards or through UPI. The seats will be booked and blocked only after a successful payment.

8.Ticket Generation: A ticket is produced with seat numbers, time and place of show on it and also a QR code.

9.Error handling: If any of the above validation/sequencing flow does not hold true, appropriate error messages will be prompted to the user for doing the needful able seats at different theaters.

10.Payment: Payment is done through net banking or debit/credit cards or through UPI. The seats will be booked and blocked only after a successful payment.

11.Receipt Generation: A unique transaction id is generated for future references.

12.Ticket Generation: A ticket is produced with seat numbers, time and place of show on it and also a QR code.

13.Error handling: If any of the above validation/sequencing flow does not hold true, appropriate error messages will be prompted to the user for doing the needful

NON-FUNCTIONAL REQUIREMENTS:

1. A login screen is provided in the beginning for entering the required username/pin no. and account number.

2. An unsuccessful login leads to a reattempt (maximum three) screen for again entering the same information. The successful login leads to a screen displaying a list of supported languages from which a user can select anyone.

3. In case of administrator, a screen will be shown having options to reboot system, shut down system, block system, disable any service.

4. In case of reboot/ shut down, a screen is displayed to confirm the user's will to reboot and also allow the user to take any backup if needed.

5. After the login, a screen with a number of options is then shown to the user. It contains all the options along with their brief description to enable the user to understand their functioning and select the proper option.

6. A screen will be provided for the user to check his account balance.

7. A screen will be provided that displays the location of all other ATMs of the same bank elsewhere in the city.

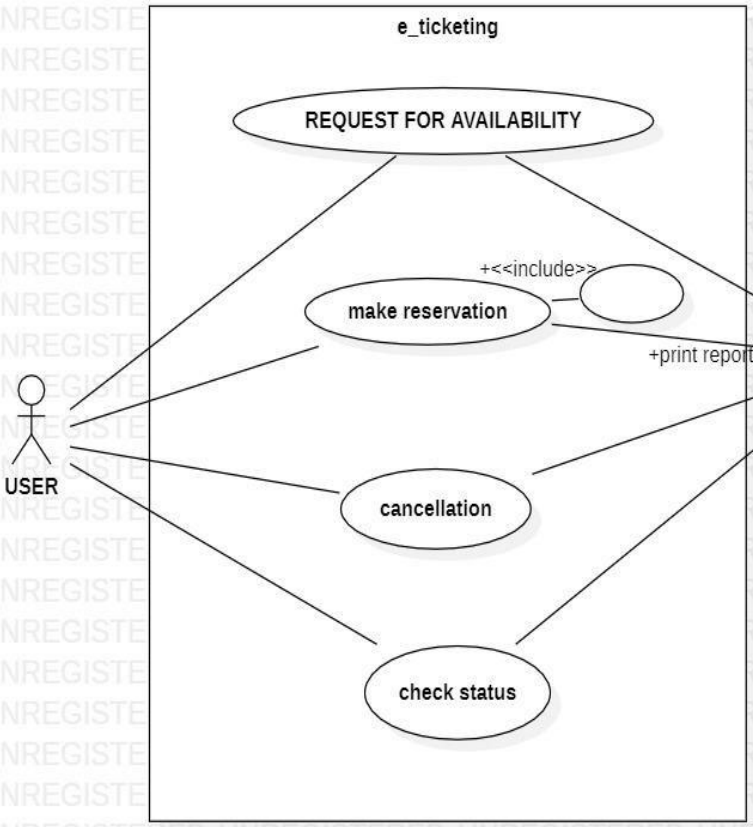
8. A screen will be provided for the user to perform various transactions in his account.

9.The booking portal shall provide customers a 24-hour service.

UML DIAGRAMS:

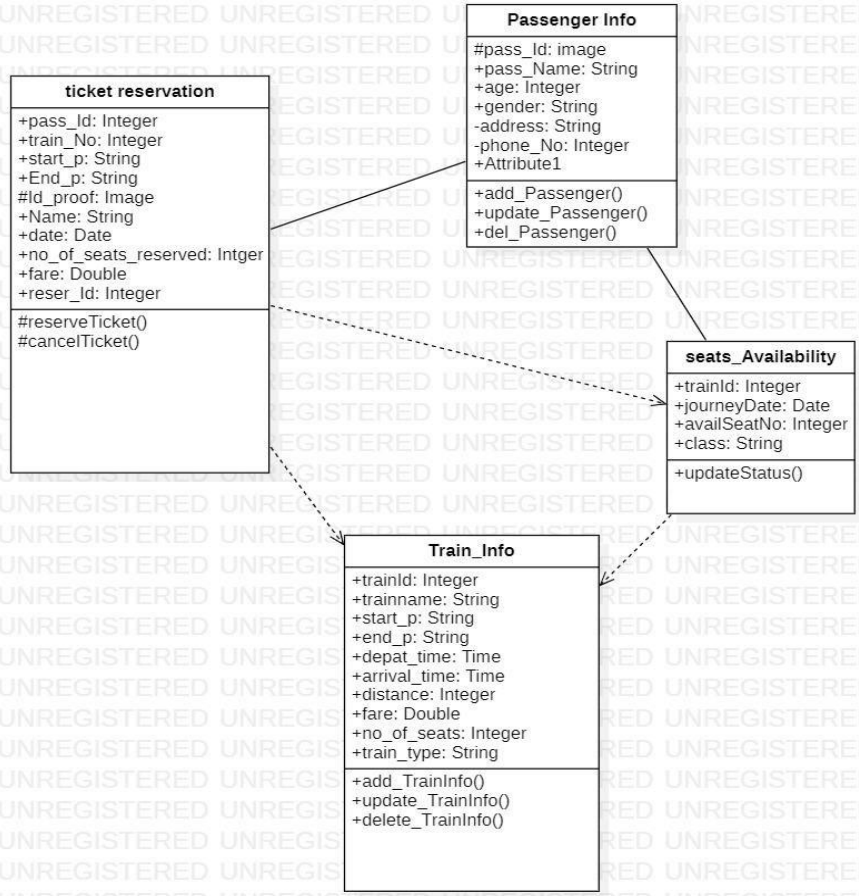
USE-CASE DIAGRAM:

Use-Case is a list of actions or events. Steps typically define the interactions between a role and a system to achieve a goal. The use-case diagram consists of various functionality performed by actors like Student, Admin, book bank and DBA.



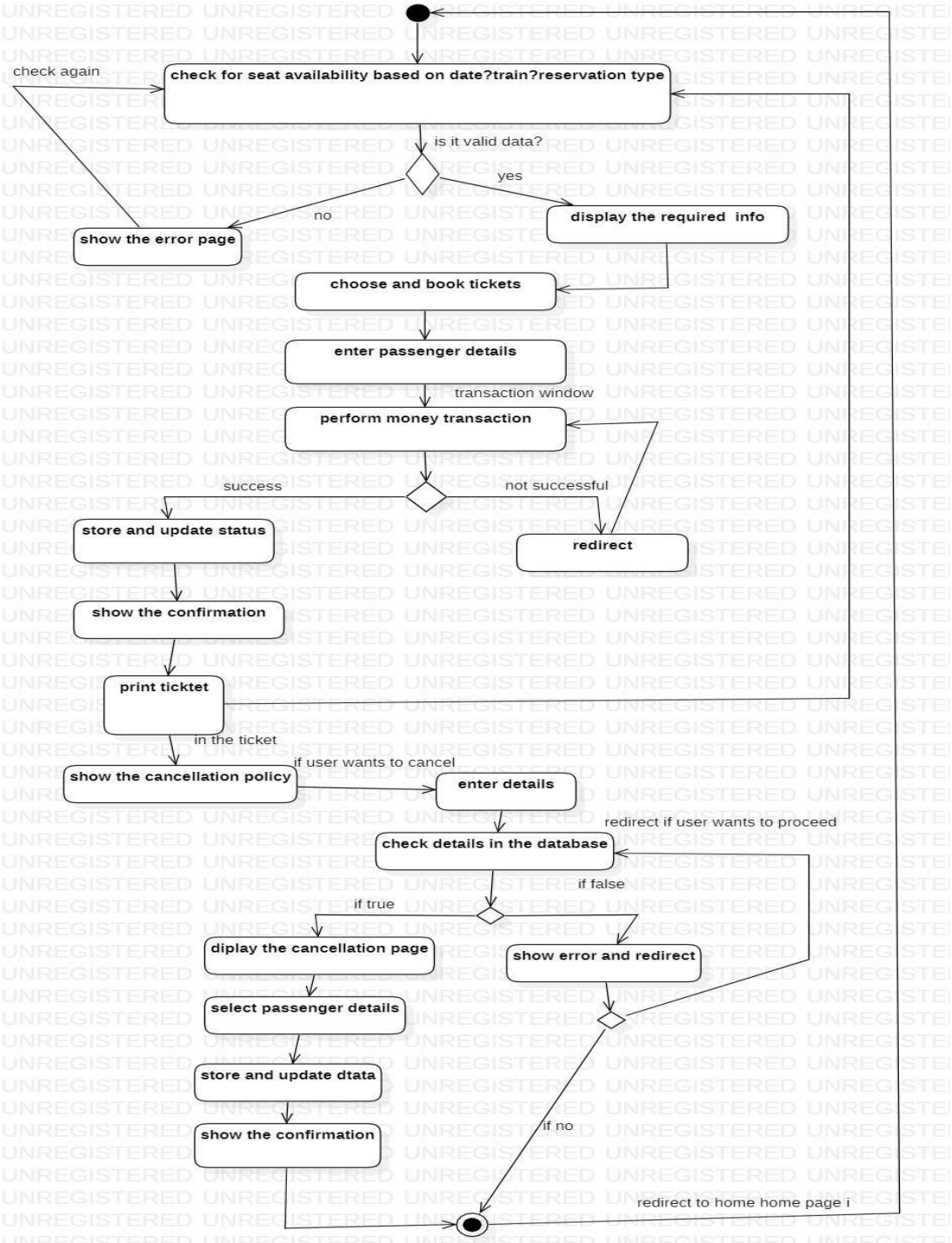
CLASS DIAGRAM:

A class diagram in the unified modeling language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations and the relationships among objects. The book bank system makes use of the following classes: student, book bank, admin and DBA.



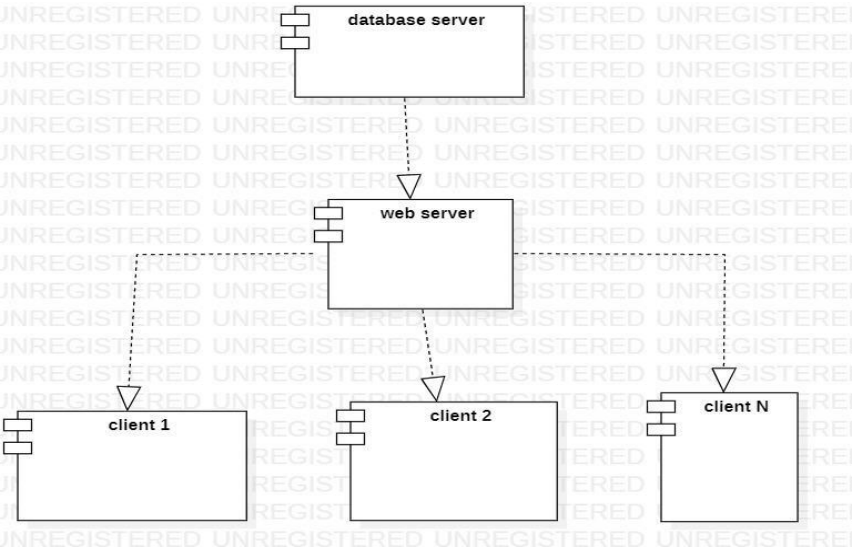
ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. Here in the activity diagram the student and admin login to the system and perform some main activity which is the main key element to the system.



DEPLOYMENT DIAGRAM:

In UML, deployment diagrams model the physical architecture of a system. Deployment diagrams show the relationships between the software and hardware components in the system and the physical distribution of the processing.



4.RECRUITMENT SYSTEM

AIM:

To develop a recruitment system using star UML tool .

PROBLEM STATEMENT:

This project Employee Recruitment Systems, a system in which jobseekers can register themselves online, view organization requirements and apply for the suitable job. Employee Recruitment System provides online help to the users all over the world. This kind of system plays an important role in simplifying the recruitment process. The system has facilities where prospective candidates can upload their CV and apply for jobs suited to them. It also makes it possible for organizations to post their staffing requirements and view profiles of interested candidates. Earlier recruitment was done manually and it was all at a time consuming work. Now it is all possible in a fraction of second. The system has been designed to do a whole lot more than just reduce paperwork. It can make a significant contribution to a company's marketing and sales activities. Employee recruitment systems make it possible for managers to access information that is crucial to managing their staff, which they can use for human resources management, staffing and planning activities. The primary purpose to develop this system is to optimize the recruitment process for an organization. Besides, the qualified applicants could be sorted by this system based on their qualifications and company requirements.

INTRODUCTION:

The Employee recruitment system helps the candidates know about the opportunities online and then apply for them according to their requirements and educational qualification.And so it will also be an easy task for the recruiter to pick the suitable candidates for the job based on their candidature.

OBJECTIVE:

This software helps applicants to find suitable job within the organization and apply for that job easily.The software helps in managing and viewing details of interested applicants for the administrator.The system is capable of sorting and filtering best suitable candidates based on some criteria.Company will not have to waste his time for finding right employee at right post

PURPOSE:

The primary purpose to develop this system is to optimize the recruitment process for an organization. Besides, the qualified applicants could be sort by this system based on their qualifications and company requirements.

SCOPE:

Online Recruitment System enables the users to have the typical recruitment facilities and features at their disposal. It resolves typical issues of manual staffing processes and activities into a controlled and closely monitored work flow in the architecture of the application. This multi-platform solution brings in by default, the basic intelligence and immense possibilities for further extension of the application as required by the user. The system makes it simpler to share and manage the organization's human resource requirements with higher efficiency and ease. The objective of these websites is to serve as a common meeting ground for jobseekers and organizations, both locally and globally. This kind of system is specifically designed for organizations to help in solving staffing problems and managing human resource department activities at a high degree of optimization.

MODULES:

1. Registration
2. Login
3. Upload CV
4. Apply for the suitable job
5. Logout
6. Sorted based on requirements
7. Get a call if selected

SOFTWARE REQUIREMENTS:

Operating System : Microsoft Windows 7/8/8.1/XP/VISTA

- Package : Adobe Dreamweaver
- Database : MY SQL
- Diagram : Microsoft Office Visio 2003
- Design : Adobe Photoshop CS6.0 , Macromedia Flash Player
- Browser : IE (Version 6 or higher) , Mozilla Firefox or Google Chrome
- APACHE Server
- MYSQL Server
- IE (Version 6 or higher)

HARDWARE REQUIREMENTS:

Processor : Intel core processor 2 GHz

- RAM : 2 GB RAM
- Hard Disk : 80 GB HDD
- Monitor : Compatible Printing Device

- Keyboard : Any Keyboard

FUNCTIONAL REQUIREMENTS:

1. The system should record all the details of an applicant.
2. The system should provide applicants with the ability to edit his profile details.
3. The system should allow users to give feedback about the system.
4. The system should allow the admin to have full authority over user accounts.
5. The system should allow admin to sort and filter applicants based on some criteria.

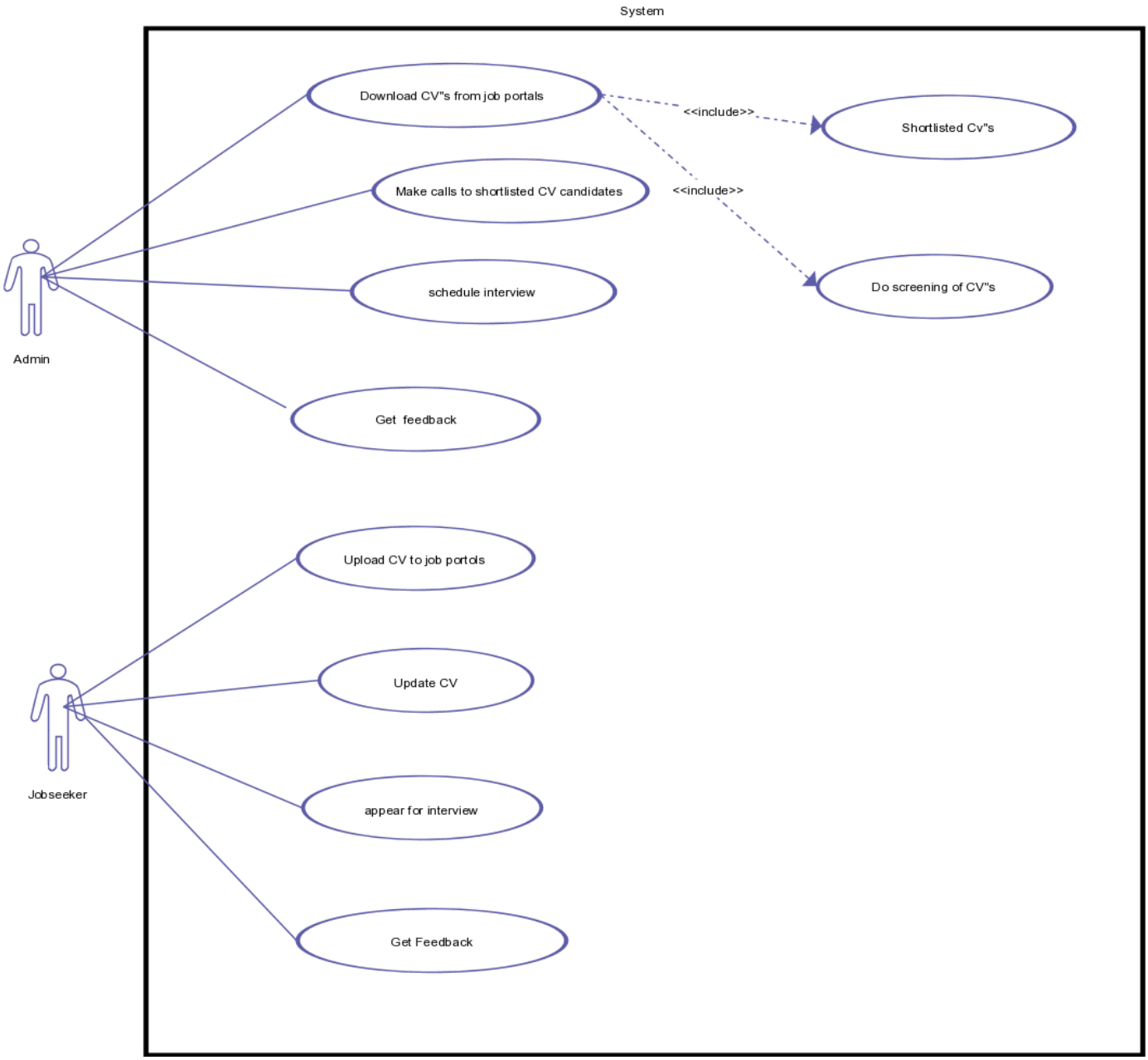
NON-FUNCTIONAL REQUIREMENTS:

1. This application is secure for every kind of user, because there is facility of session management.If any user logs out from any session then nobody will be able to access his profile without knowing his confidential password.
2. The database used here is robust,reliable and fast.
3. This application can be accessed from any type of platform.
4. There is no case of redundancy in the database so it will not take extra memory space.

UML DIAGRAMS:

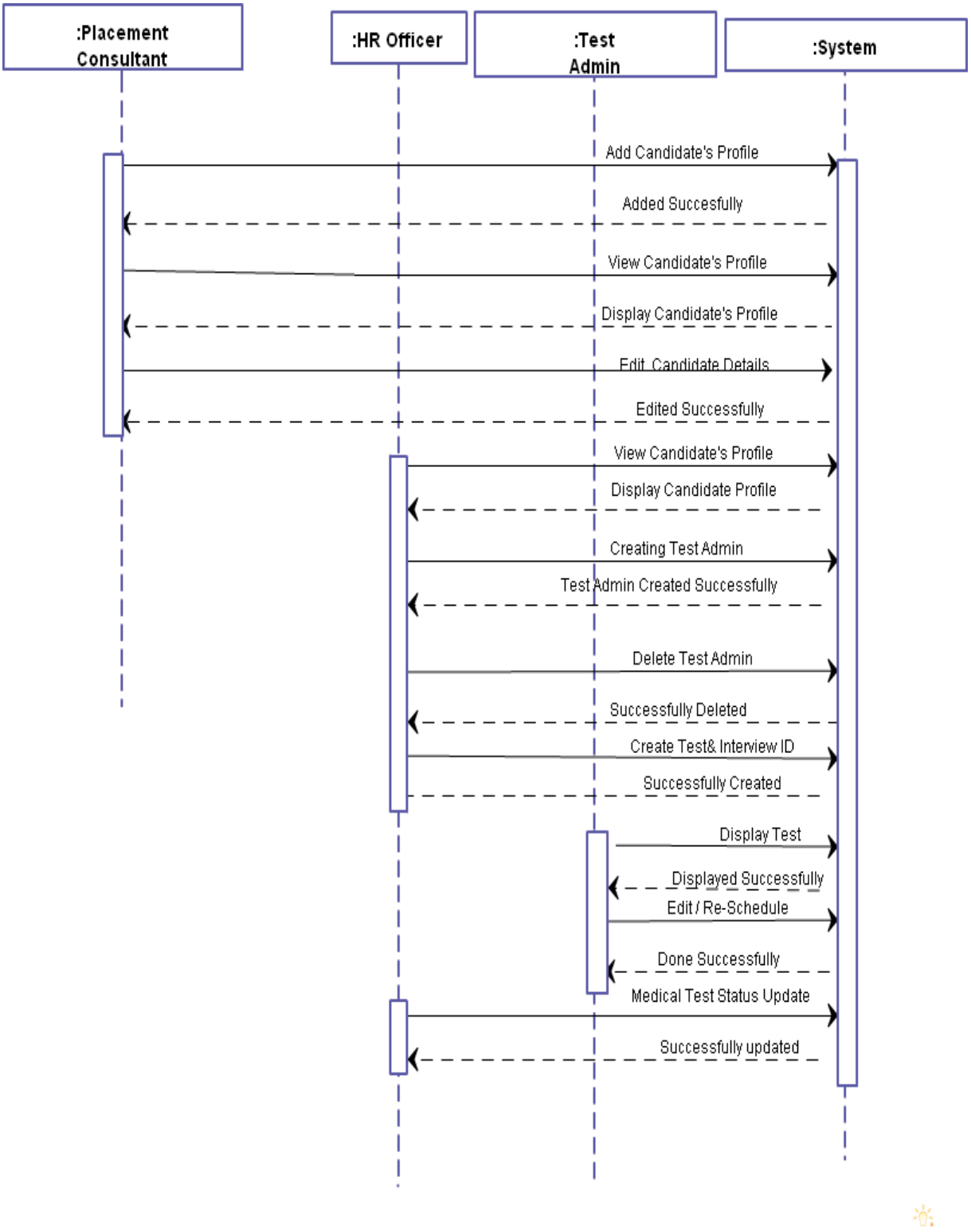
Use Case Diagram*:*

Use case diagrams are behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.



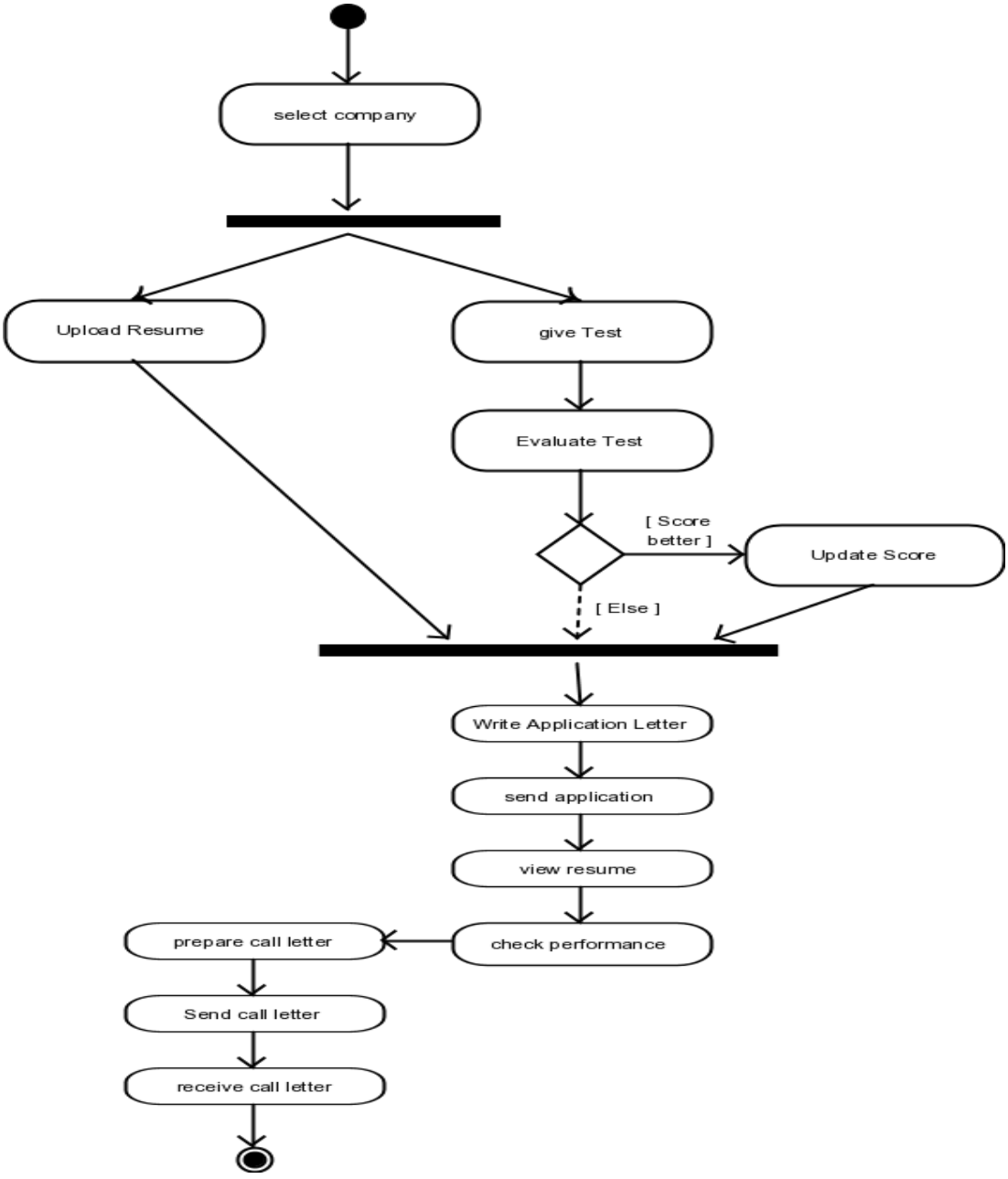
Sequence Diagram:

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios.


:Placement Consultant
:HR Officer
:Test Admin
:System
Add Candidate's Profile
Added Succesfully
View Candidate's Profile
Display Candidate's Profile
Edit Candidate Details
Edited Successfully
View Candidate's Profile
Display Candidate Profile
Creating Test Admin
Test Admin Created Successfully
Delete Test Admin
Successfully Deleted
Create Test& Interview ID
Successfully Created
Display Test
Displayed Successfully
Edit / Re-Schedule
Done Successfully
Medical Test Status Update
Successfully updated

Activity Diagram:

Activity diagrams are the object-oriented equivalent of flow charts and data-flow diagrams from structured development. It describes the workflow behavior of a system. The process flows in the system are captured in the activity diagram. Activity diagram illustrates the dynamic nature of a system by modeling the flow of control from Activity to activity.



5. HOSPITAL MANAGEMENT

AIM: To develop a hospital management system using star UML tool.

PROBLEM STATEMENT:

Any hospital requires a system that maintains its hospital management system as well as keeps the record of the hospital in the database. This software manages all information about patient name,patient address,doctor information,staff information etc. It also stores daily information of patients which is done by the doctor. Also store information about billing , finally it calculates the total bill of the patient .

INTRODUCTION:

The project Hospital Management system includes registration of patients, storing their details into the system, and also computerized billing in the pharmacy, and labs. The software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically.

OBJECTIVES:

This system provides an easy solution to both the management and the patients.And also helps to avoid confusion related to any tests or final bill.

PURPOSE:

The Software is for the automation of Hospital Management. It maintains two levels of users Administrator Level and User Level.The Software includes Maintaining Patient details.Providing Prescription, Precautions and Diet advice.Providing and maintaining all kinds of tests for a patient. Billing and Report generation.

SCOPE:

The proposed software product is the **Hospital Management System (HMS).** The system will be used to get the information from the patients and then storing that data for future usage.The current system in use is a paper-based system. It is too slow and cannot provide updated lists of patients within a reasonable timeframeThe intentions of the system are to reduce overtime pay and increase the number of patients that can be treated accurately.

SOFTWARE REQUIREMENTS:

1. A database like DBMS to store the list of authors and the articles.
2. A web browser like Chrome, Mozilla Firefox etc.
3. Operating System – Windows, Linux, macOS 32 bit and 64 bit

HARDWARE REQUIREMENTS:

1. A device (Computer/laptop)
2. Memory (RAM): Minimum 2GB RAM
3. Processor: Minimum 1GHZ; Recommended 2 GHZ or more. iv. Hard disk – 4GB;.
4. Ethernet connection (LAN) or, a wireless adapter (Wi-Fi)

FUNCTIONAL REQUIREMENTS:

1.Registration :-The **HMS** shall employ all front-desk staff to add new patients to the system.

2.Assign ID:-The HMS shall allow front-desk staff to give each patient a ID and add it to the patient's record. This ID shall be used by the patient throughout his/her stay in hospital.

3.Delete Patient ID:-The administrative staff in the ward shall be allowed to delete the ID of the patient from the system when the patient checks out

4. Add to beds-available list:-

The administrative staff in the ward shall be allowed to put the beds just evacuated in beds-available list.

NON-FUNCTIONAL REQUIREMENTS:

1. Security

1. The system must automatically log out all customers after a period of inactivity.

2.The system should not leave any cookies on the customer's computer containing the user's password.

3.The system's back end servers shall only be accessible to authenticated administrators.

4.Sensitive data will be encrypted before being sent over insecure connections like the internet.

2. Reliability

The system provides storage of all databases on redundant computers with automatic switchover. The reliability of the overall program depends on the reliability of the separate components. The main pillar of reliability of the system is the backup of the database which is continuously maintained and updated to reflect the most recent changes. Thus the overall stability of the system depends on the stability of container and its underlying operating system.

3. Availability

The system should be available at all times, meaning the user can access it using a web browser, only restricted by the down time of the server on which the system runs. In case of a of a hardware failure or database corruption, a replacement page will be shown. Also in case of a hardware failure or database corruption, backups of the database should be retrieved from the server and saved by the administrator. Then the service will be restarted.

4.Maintainability

A commercial database is used for maintaining the database and the application server takes care of the site. In case of a failure, a re-initialization of the program will be done. Also the software design is being done with modularity in mind so that maintainability can be done efficiently.

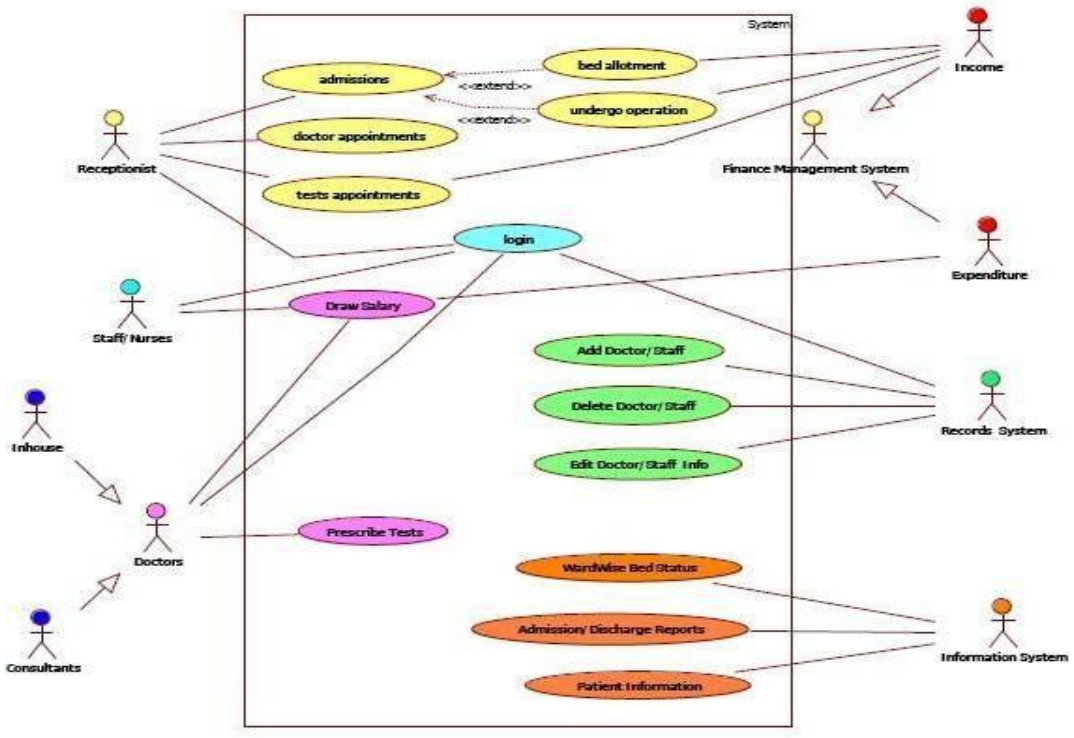
5.Portability

The application is HTML and scripting language based. So The end-user part is fully portable and any system using any web browser should be able to use the features of the system, including any hardware platform that is available or will be available in the future. An end-user is use this system on any OS; either it is Windows or Linux. The system shall run on PC, Laptops, and PDA etc.

UML DIAGRAMS:

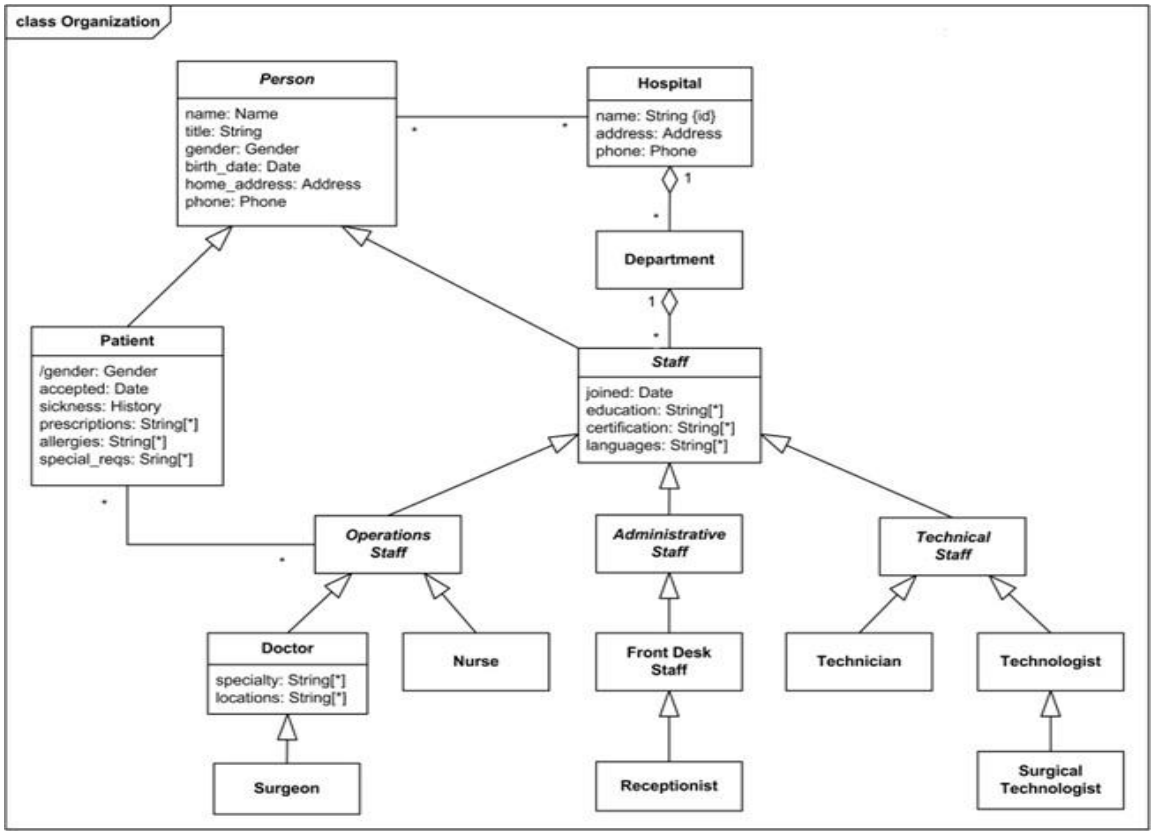
USE-CASE DIAGRAM:

Use-Case is a list of actions or events. Steps typically define the interactions between a role and a system to achieve a goal. The use-case diagram consists of various functionality performed by actors like Student, Admin, book bank and DBA.



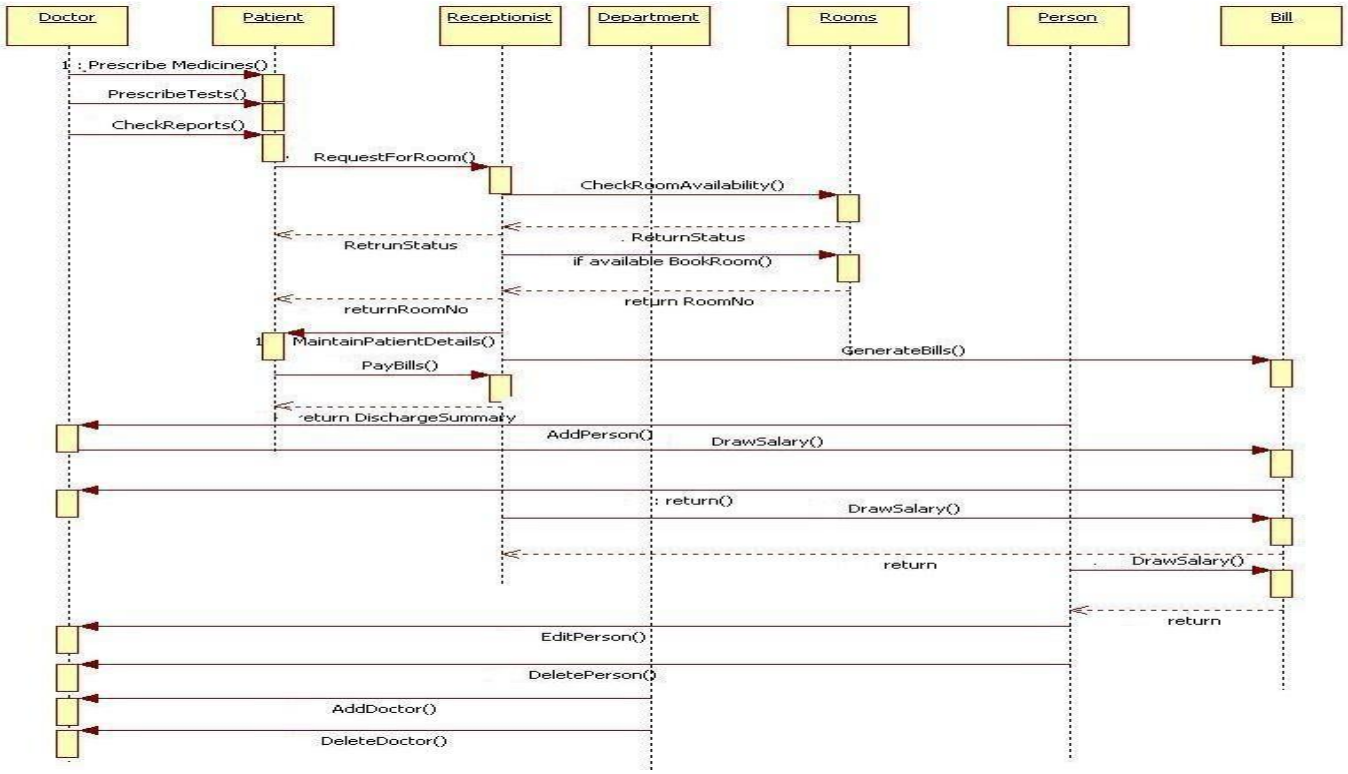
CLASS DIAGRAM:

A class diagram in the unified modeling language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations and the relationships among objects. The book bank system makes use of the following classes: student, book bank, admin and DBA.



SEQUENCE DIAGRAM:

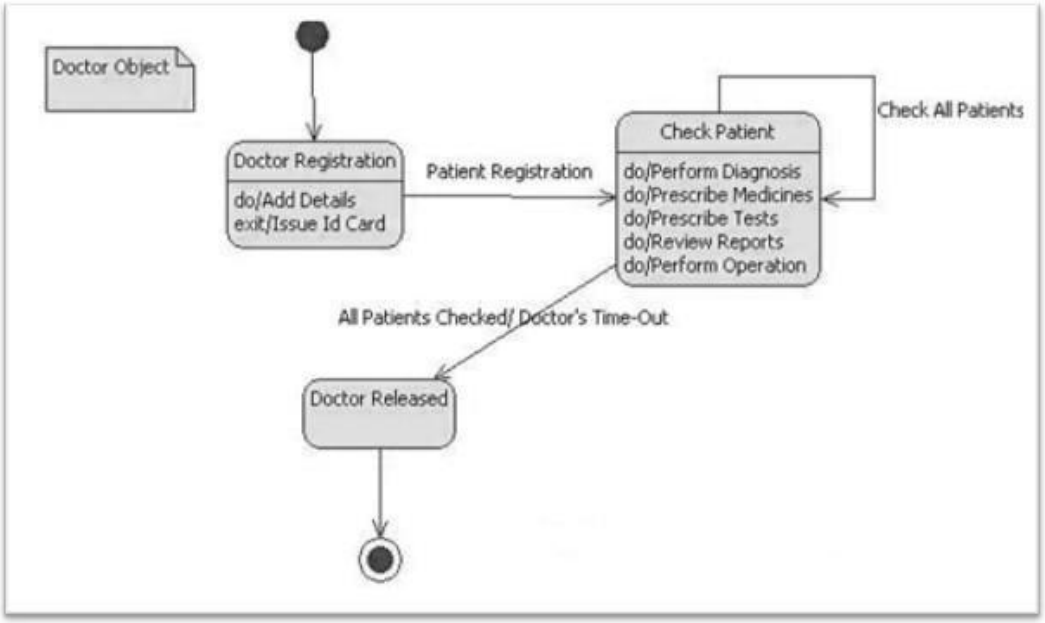
A sequence diagram represents the sequence and interactions of a given use case or scenario. Sequence diagrams capture most of the information about the system. It is also represented in order by which they occur and have the object in the system send messages to one another. Here the sequence starts with interaction between student and book bank followed by a database. Once the book has been selected the next half of the sequence starts between the book bank and admin followed by the database.



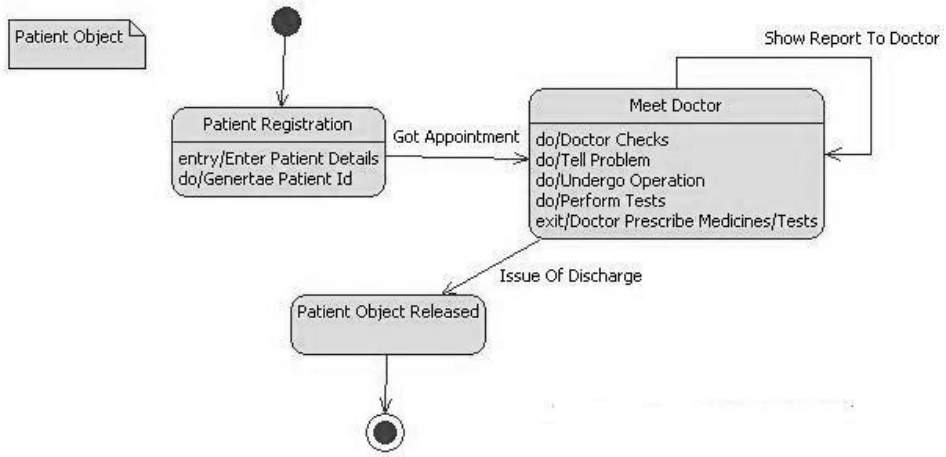
STATE DIAGRAM :

A state diagram is used to represent the condition of the system or part of the system at finite instances of time.

STATE DIAGRAM FOR DOCTOR OBJECT:



STATE DIAGRAM FOR PATIENT OBJECT:



6.ONLINE BANKING SYSTEM

AIM: To develop an online banking using star UML tool.

PROBLEM STATEMENT:

The banking industry is witnessing a revolution in products, processes, markets and regulations. And It's a revolution that is not about to stop or even slow down. Since the only option is to adapt and evolve, it is essential that systems have the flexibility to quickly adjust to the needs of today's financial market. It's a tough challenge. Because today's fast-moving marketplace is also extremely competitive. Moreover, the need to retain existing customers and attract new ones often conflicts with the need to reduce costs and improve efficiency. But whatever the challenges facing in retail banking operation, Online Banking System - OBS can help to meet and overcome them.

INTRODUCTION:

This document gives detailed functional and non-functional requirements for the Bank Management System. This software will support online banking transactions. The purpose of this document is that the requirements mentioned in it should be utilized by software developers to implement the system.

OBJECTIVE:

To enable bank customers to use mobile instruments as a channel for accessing their banks accounts and remit funds. Making payment simpler just with the mobile number of the beneficiary. To sub-serve the goal of the Reserve Bank of India (RBI) in electronification of retail payments.

PURPOSE:

The purpose of this document is to present a detailed description of the Online Banking System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system and will be liable for the approval or disapproval of the project by the community of the Bank.

SCOPE:

Online Banking System is specifically developed for internet banking for Balance Enquiry, Funds transfer to another account, bill payments and request services like request for cheque book / change details, request for SMS, mini statement .The traditional way of maintaining details of a user in a bank was to enter the details and record them. Every time the user needs to perform some transaction he has to go to the bank and perform the necessary action, which may not be so feasible all the time. It may be a hard hitting task for the users .This project gives real life understanding of internet banking and activities performed by various roles in the supply chain. Here, we provide automation for the banking system through the internet. Internet banking system project captures-Activities performed by different roles in real life banking which provides enhanced techniques for maintaining the required information up-to-date, which results in efficiency. The project gives real life understanding of internet banking and activities performed by various roles in the supply chain.

SOFTWARE REQUIREMENTS:

1.Operating system: We have chosen windows operating system for its best support and userfriendliness.

2. Database Management Software: to save the records of the customers and bank employees and their details, MY SQL database is used.

HARDWARE REQUIREMENTS:

As this system is an online Web-based application so a client server will be the most suitable Organizational style for this system. Computer systems will be needed by each of the actors as well as that user must be connected to the internet. So, concisely following hardware will be needed.

1.Computer systems 2.Internet availability

FUNCTIONAL REQUIREMENTS:

Admin's Functionality

- **Login/Logout** – Admin has to login to access the software using Id and password.
- **Add /Remove employees** – Admin has to specify the users who can manage the software.
- **View employee details** – Admin can view its employee details.

Bank Employees functionality

- **Login/Logout –** Employee has to login to gain access to the software using his/her Id and password.
- **Add/Remove end users** –They can add the new customers and remove the customers who have closed their account.
- **View and Update end users account details** –They have the access to view their customer details and update it.
- **Generate end users details** –They can print customer details.
- **Service customer requests**–Employees have to provide services demanded by the customers.

End Users Functionality

- **Register**–Users who are new to the software should register themselves.
- **Login/Logout** –Registered users can simply login to the software using their Customer ID and password.
- **View Account balance**- They can view their current account balance.
- **Print transaction history** – They can request and print past three months transaction history of their account.
- **Request email/SMS services**– They can request email and SMS service for their transactions.
- **Fund transfer**– Services like fund transfer are also available online.
- **Bill payments**- They can pay mobile phone bill, electricity bill, water bill, gas bill etc. directly through their bank account.
- **Request check book** – They can request for a new check book online.
- **Contact us –** If they have any problem related to the software or their account they can contact the bank.

NON-FUNCTIONAL REQUIREMENTS:

Security:

- The system must automatically log out all customers after a period of inactivity.
- The system should not leave any cookies on the customer's computer containing theuser's password.
- The system's back end servers shall only be accessible to authenticated administrators.
- Sensitive data will be encrypted before being sent over insecure connections like the internet.

Reliability:

The system provides storage of all databases on redundant computers with automatic switchover. The reliability of the overall program depends on the reliability of the separate components. The main pillar of reliability of the system is the backup of the database which is continuously maintained and updated to reflect the most recent changes. Thus the overall stability of the system depends on the stability of container and its underlying operating system.

Availability:

The system should be available at all times, meaning the user can access it using a web browser, only restricted by the down time of the server on which the system runs. In case of a of a hardware failure or database corruption, a replacement page will be shown. Also in case of a hardware failure or database corruption, backups of the database should be retrieved from the server and saved by the administrator. Then the service will be restarted.

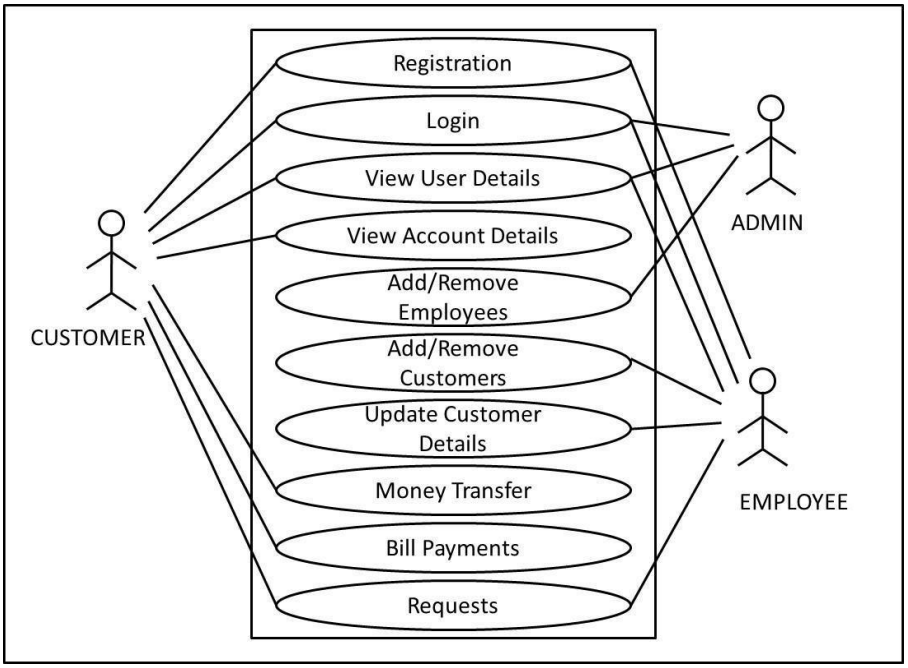
Maintainability:

A commercial database is used for maintaining the database and the application server takes care of the site. In case of a failure, a re-initialization of the program will be done. Also the software design is being done with modularity in mind so that maintainability can be done efficiently.

UML DIAGRAMS:

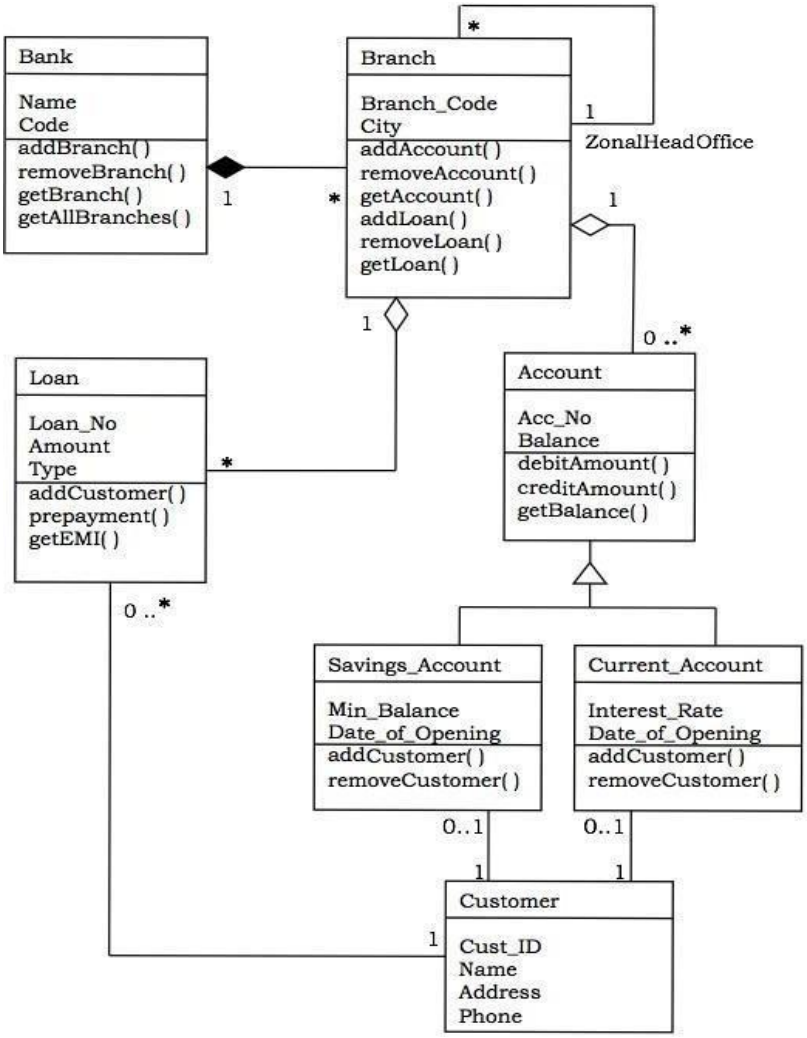
USE-CASE DIAGRAM:

Use-Case is a list of actions or events. Steps typically define the interactions between a role and a system to achieve a goal. The use-case diagram consists of various functionality performed by actors like Student, Admin, book bank and DBA.



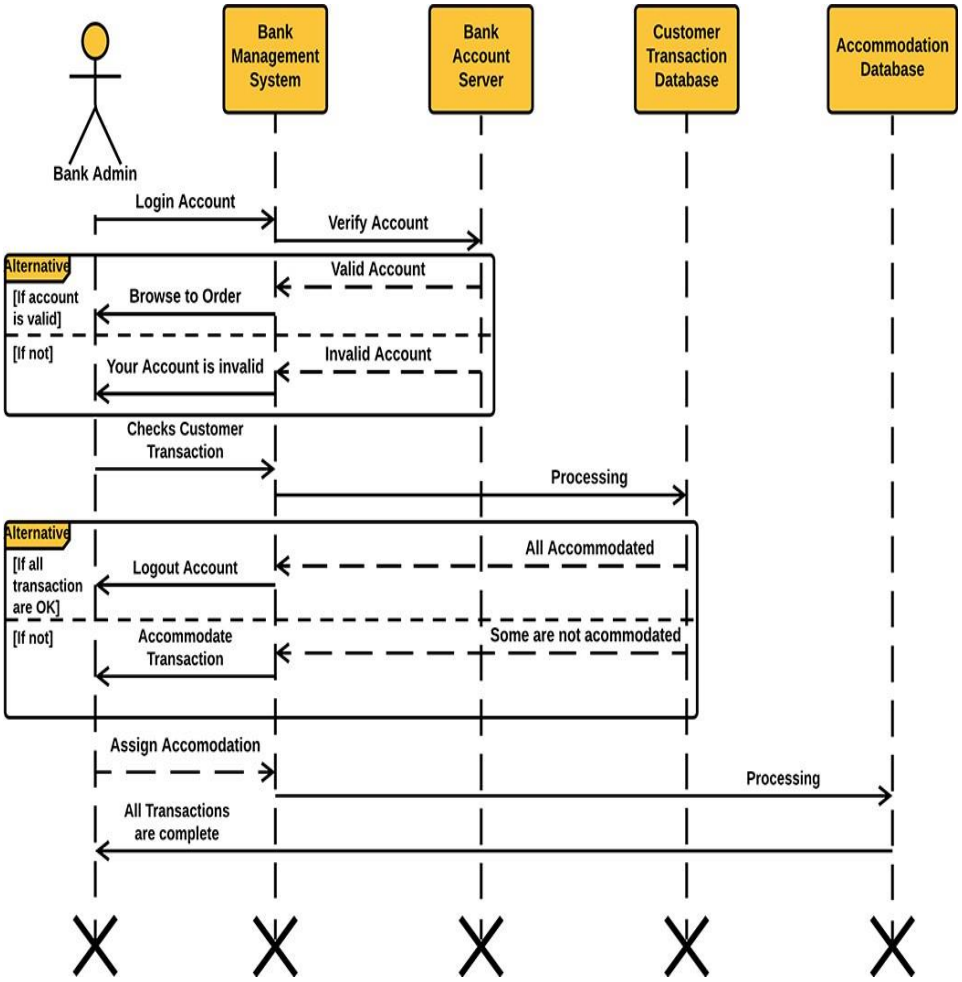
CLASS DIAGRAM:

A class diagram in the unified modeling language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations and the relationships among objects. The book bank system makes use of the following classes: student, book bank, admin and DBA.



SEQUENCE DIAGRAM:

A sequence diagram represents the sequence and interactions of a given use case or scenario. Sequence diagrams capture most of the information about the system. It is also represented in order by which they occur and have the object in the system send messages to one another. Here the sequence starts with interaction between student and book bank followed by a database. Once the book has been selected the next half of the sequence starts between the book bank and admin followed by database.



2.Working with Git and GitHub

CHEAT SHEET

GIT

[Git is a version control system.Git helps you keep track of code changes.Git is used to collaborate on code.]

GIT is a popular version control system.

It is used for:

- Tracking code changes.
- Tracking who made the changes.
- Coding collaboratively.

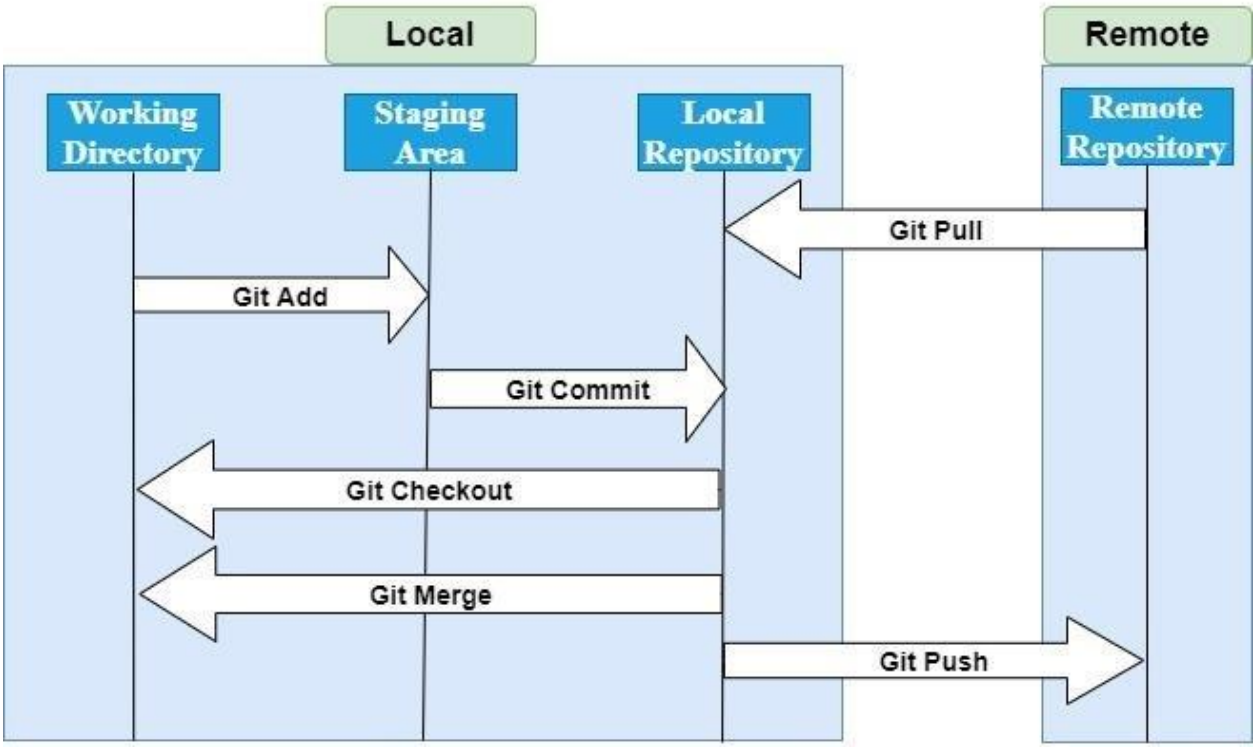
GIT can

- 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 a main project.

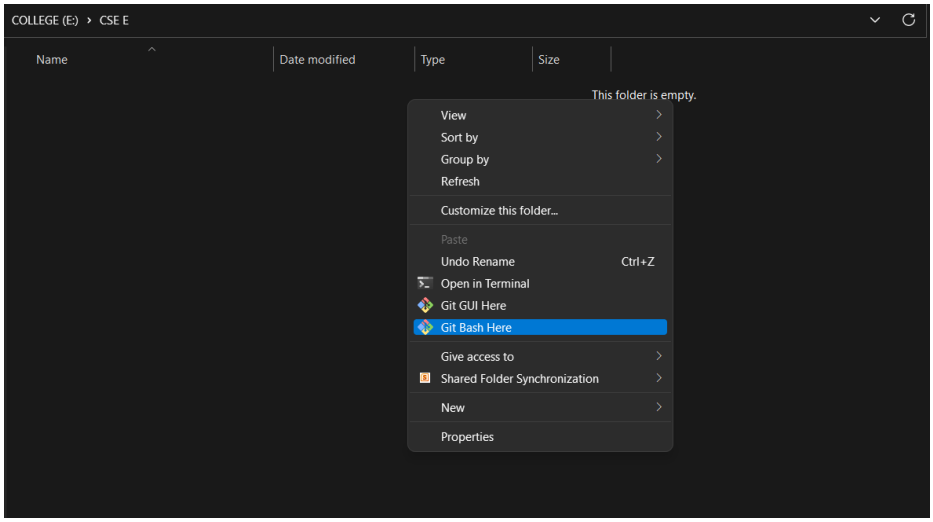
HOW DOES GIT WORK

- Initialize Git on a folder, making it a Repository
- Git now creates a hidden folder to keep track of changes in that folder
- When a file is changed, added or deleted, it is considered modified
- You select the modified files you want to Stage
- The Staged files are Committed, which prompts Git to store a permanent snapshot of the files
- Git allows you to see the full history of every commit.
- You can revert back to any previous commit.
- Git does not store a separate copy of every file in every commit, but keeps track of changes made in each commit!
- FILES in your git repository folder can be of two states:
    - Tracked - files that git knows about and are added to the repo
    - Untracked - files that are in your working directory, but not added to the staging environment.

GIT STAGES



USING GIT WITH COMMAND LINE

- For windows we can use git bash which comes included with git for windows.
- 

- For mac and linux you can use the built in terminal.

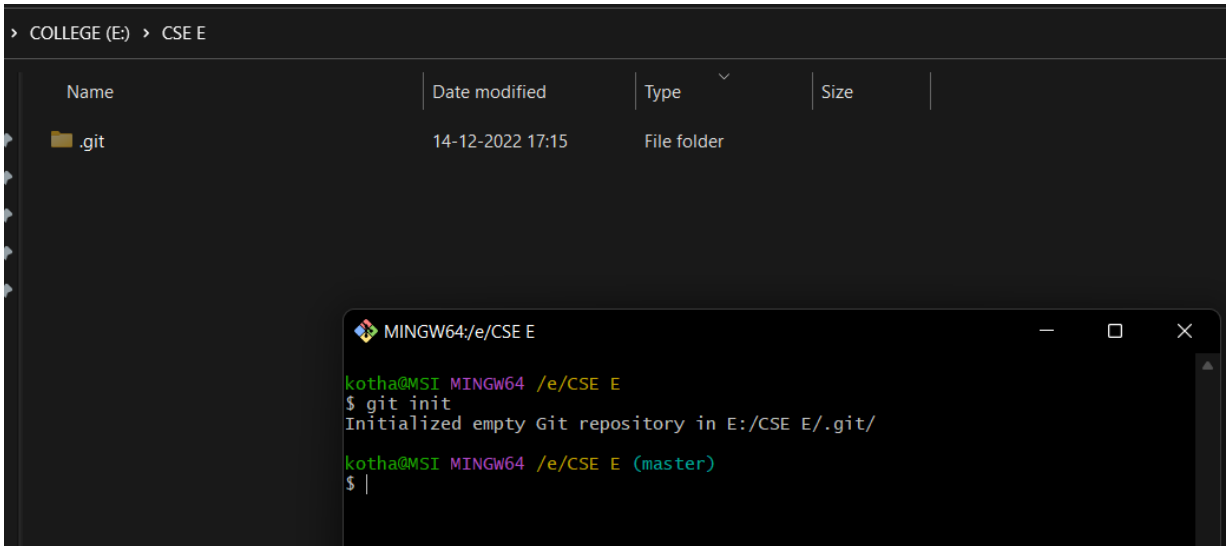
CONFIGURE GIT

- $git config - -global user.name “xyz-username”
- $git config - -global user.email “xyz@gmail”

- 

INITIALIZE GIT

- $git init
- This means you have just created an empty git repository. Now git watches over this folder on which u initialized your git and it creates a hidden folder to keep track of changes.

- 

GIT ADDING NEW FILES AND GIT STAGING ENVIRONMENT

- $ls command will list the files in the directory
- $git status displays the state of the working directory and the staging area.
- The core function of git is the concept of staging environment and commit.
- Staged files are files that are ready to be committed to the repository you are still working on
- You add a file to the stage like using:-
  $git add index.txt
- Here we are creating a new file called file1.txt
- 

- Then we are creating another file called file2.txt
- 

- Now we are adding only file1.txt to the staging environment

-
```
kotha@MSI MINGW64 /e/CSE E (master)
$ git add file1.txt
```
- Now if we check the status of out repository using “$git status” command
  It shows us that file1.txt is yet to be committed to the repo
-
```
kotha@MSI MINGW64 /e/CSE E (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file1.txt
```
- To add all the files in the current directory to the staging environment:-
  $git add - -all or git add .

Using - -all instead of individual file names will stage all changes(new,modified and deleted) files.

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git add .
```

Now if we check the status of our staging environment again we can see that file2.txt is also adding to the stage even if we didn't specify file2.txt

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file1.txt
        new file:   file2.txt
```

GIT RESTORE

- The "restore" command helps to unstage or even discard uncommitted local changes
  Example:-
- First lets add a file called file3.txt to the stage
- 

- Then if we check the status of our stage it shows that file3.txt is added to it and ready to commit

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   file3.txt
```

- Now lets use restore command to unstage the file file3.txt

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git restore --staged file3.txt
```

- Now if we check the status again it shows us that file3.txt is one of the untracked files which means it is not on stage is not ready to be committed

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file3.txt

nothing added to commit but untracked files present (use "git add" to track)
```

GIT COMMIT

- Git considers each commit as a change point or save point
- It is a point in the project you can go back to if you find a bug, or want to make a change.
- When we commit we must always include a message

```
MINGW64:/e/CSE E

kotha@MSI MINGW64 /e/CSE E (master)
$ git commit -m "first commit"
[master (root-commit) 1e17634] first commit
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file1.txt
 create mode 100644 file2.txt
```

SHORT STATUS FLAGS

- These help you skip the staging part
- $git status --short
- Short status flags are:-
  - ?? - Untracked files
  - A - Files added to stage
  - M - Modified files
  - D - Deleted files

GIT BRANCH

- In git a branch is a new/separate version of the main repository.
- Branches allow you to work on different parts of the project without impacting the main branch
- When the work is complete, a branch can be merged with the main project
- You can switch between branches and work on different projects without them interfering with each other.
- To create a new branch:- $git branch hello-world-images

We created a new branch called “hello-world-images”

- Checkout is the command used to check out a branch moving us from the current branch to the one specified at the end of the command:

$git checkout hello-world-images
Switched to branch 'hello-world-images'
Example:-

- Lets use git branch command to check the current branch we are in right now

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git branch
* master
```

- It shows us that we are in the master branch
- Now lets create a branch called test using git branch test command

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git branch test
```

- Let's switch to the test branch using git checkout test which enables us to switch to the test branch from our earlier master branch

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git checkout test
Switched to branch 'test'
```

- Lets check the branch we are in right now again

```
kotha@MSI MINGW64 /e/CSE E (test)
$ git branch
  master
* test
```

- It shows us that we are in branch test.

MERGE BRANCHES

- To merge branches we have first change to master branch

$git checkout master
Switched to branch 'master'

- Now we merge the current branch(master) with the other branch lets say its hello

$git merge hello

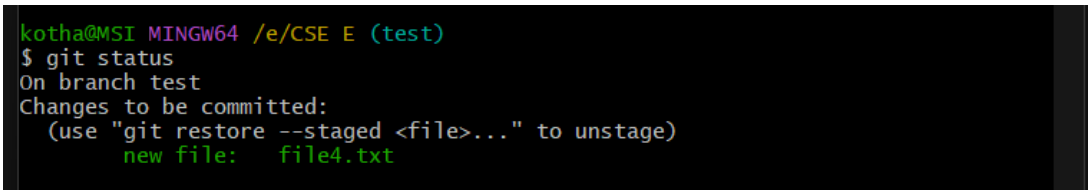
- Now that master branch hello branch are the same after merge we can delete hello branch as it is no longer needed

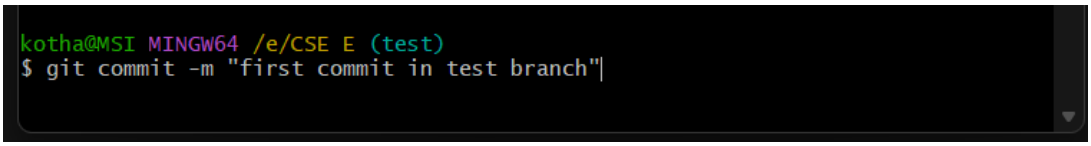
$git branch -d hello
Deleted branch hello

Example:-

- First lest switch branch from master to test branch

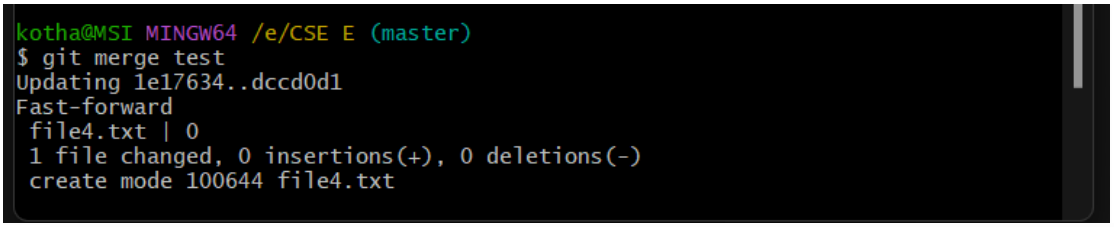
- Now let's make changes in the test branch one of such changes is adding a new file to the stage called file4.txt
- 

- Now lets check our stage status it shows us that file4 is on stage ready to be committed
- 

- Next we create our first commit in the test branch
- 

- Now we switch from test branch back to the master branch
- 

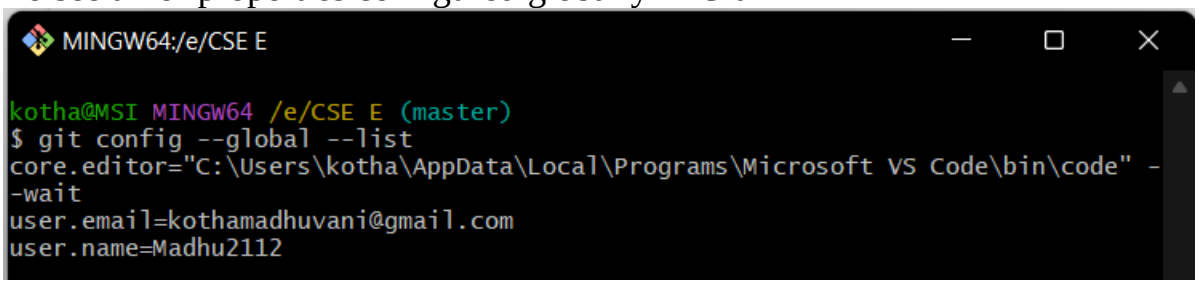
- Then we can merge test branch in to our master branch



- Here we can see that file4.txt is added to the master branch and being considered as a new change to the branch.

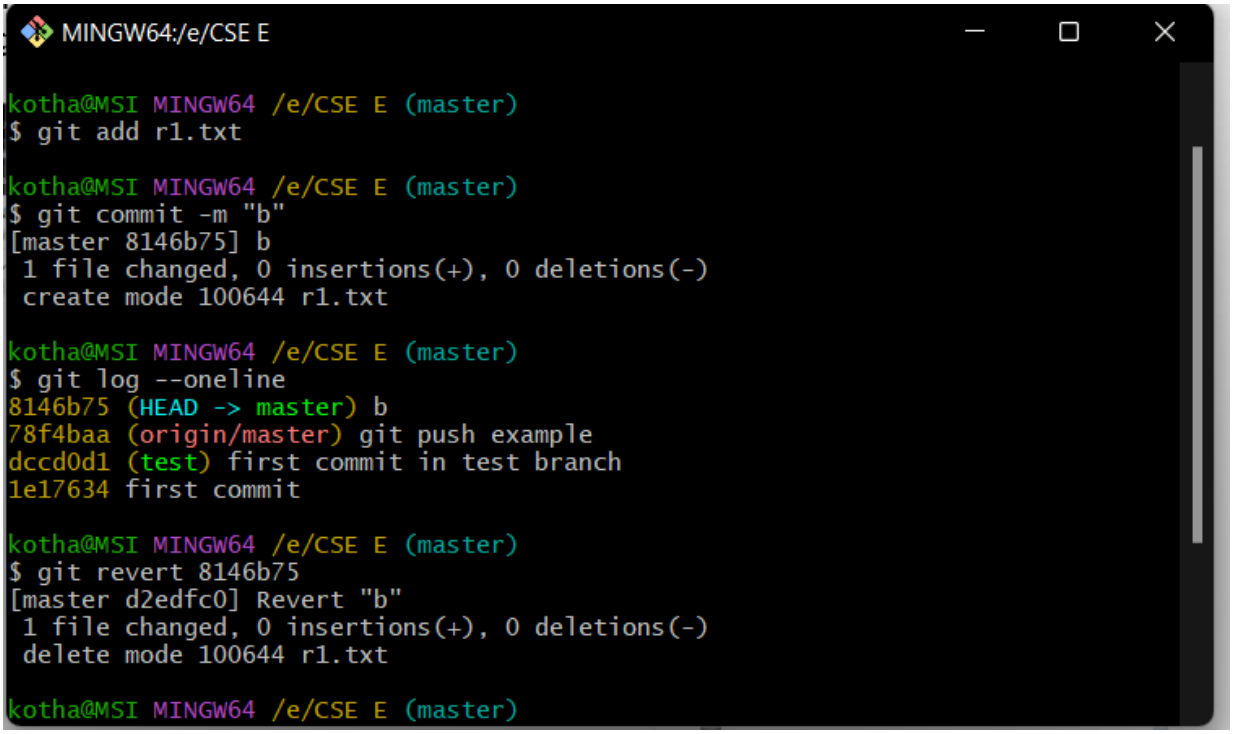
$git config --global—list

To see all of properties configured globally in Git



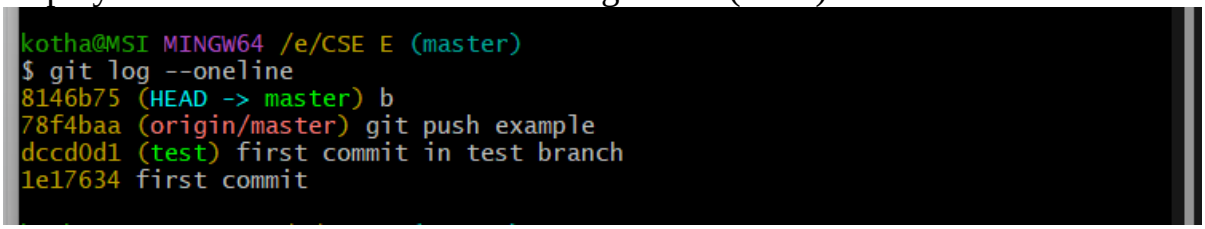
GIT REVERT

The git revert command is a forward-moving undo operation that offers a safe method of undoing changes.



GIT LOG

The git log command displays all the commits in a repository's history. By default, the command displays each commit's: Secure Hash Algorithm (SHA) author.

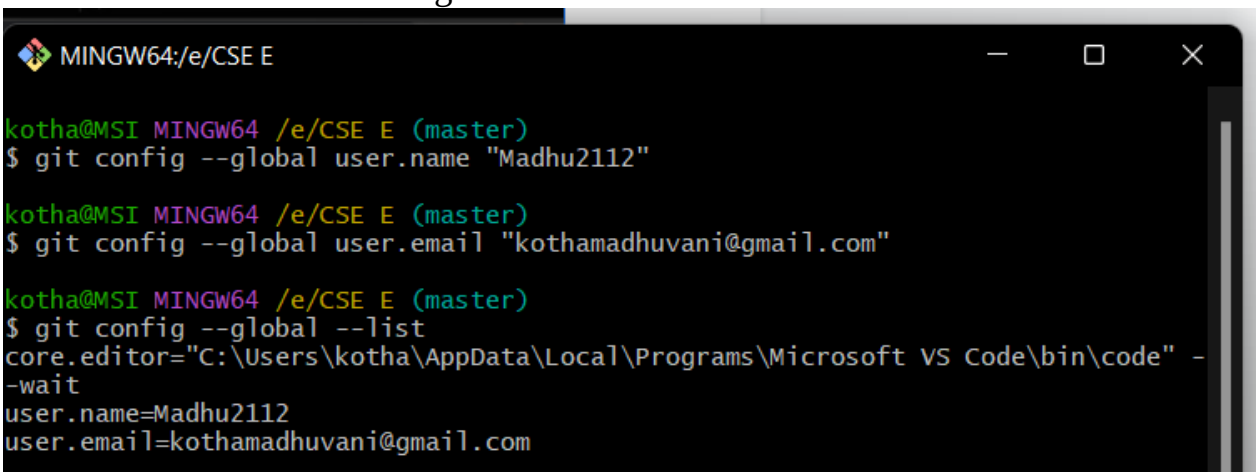


GIT VS GITHUB

S.No.	Git	GitHub
1.	Git is a software.	GitHub is a service.
2.	Git is a command-line tool	GitHub is a graphical user interface
3.	Git is installed locally on the system	GitHub is hosted on the web
4.	Git is maintained by linux.	GitHub is maintained by microsoft.
5.	Git is focused on version control and code sharing.	GitHub is focused on centralized source code hosting.
6.	Git is a version control system to manage source code history.	GitHub is a hosting service for Git repositories.
7.	Git was first released in 2005.	GitHub was launched in 2008.
8.	Git has no user management feature.	GitHub has built-in user management feature.

UNSET YOUR CREDENTIALS

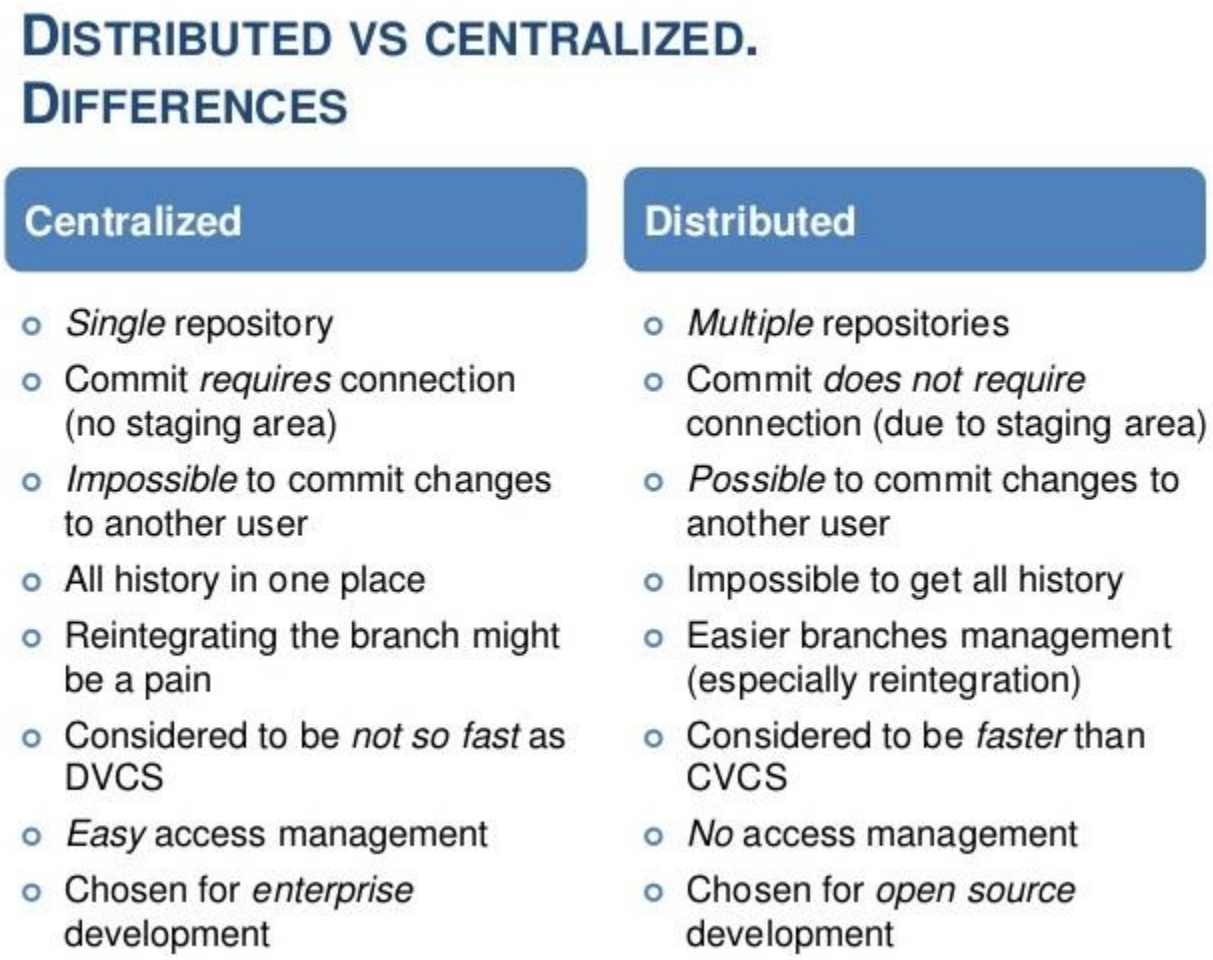
Git config –global user.name "user" and git config –global user.email "user.email@gmail.com" lets us add credentials to the git



Whereas git config –global –unset-all user.name lets us unset all usernames and git config –global –unset-all user.email lets us unset all user emails

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git config --global --unset-all user.name

kotha@MSI MINGW64 /e/CSE E (master)
$ git config --global --list
core.editor="C:\Users\kotha\AppData\Local\Programs\Microsoft VS Code\bin\code" -
-wait
user.email=kothamadhuvani@gmail.com

kotha@MSI MINGW64 /e/CSE E (master)
$ git config --global --unset-all user.email

kotha@MSI MINGW64 /e/CSE E (master)
$ git config --global --list
core.editor="C:\Users\kotha\AppData\Local\Programs\Microsoft VS Code\bin\code" -
-wait
```

Version control system VS distributed control system



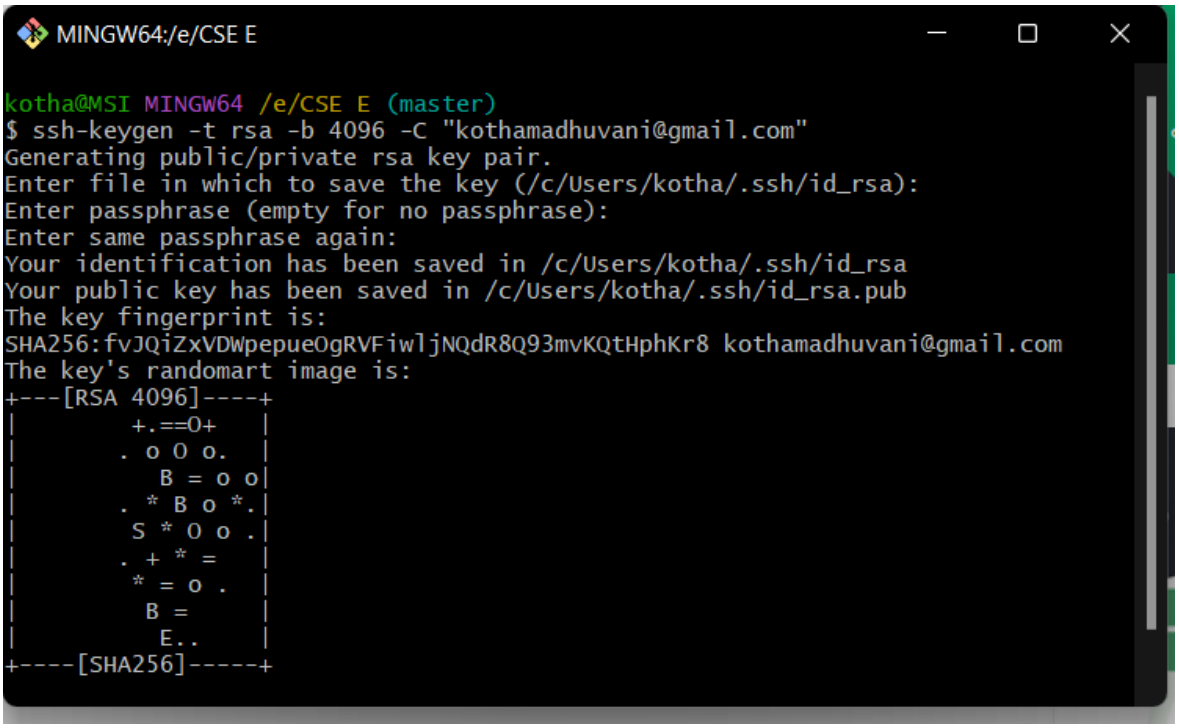
CONNECT WITH SSH

ABOUT:-

- You can connect to GitHub using the Secure Shell Protocol (SSH), which provides a secure channel over an unsecured network.
- Using the SSH protocol, you can connect and authenticate to remote servers and services. With SSH keys, you can connect to GitHub without supplying your username and personal access token at each visit. You can also use an SSH key to sign commits.
- SSH keys are used to initiate a secure "handshake". When generating a set of keys, you will generate a "public" and "private" key.
- The "public" key is the one you share with the remote party. Think of this more as the lock.
- The "private" key is the one you keep for yourself in a secure place. Think of this as the key to the lock.
- SSH keys are generated through a security algorithm. It is all very complicated, but it uses prime numbers, and large random numbers to make the public and private key.
- It is created so that the public key can be derived from the private key, but not the other way around.

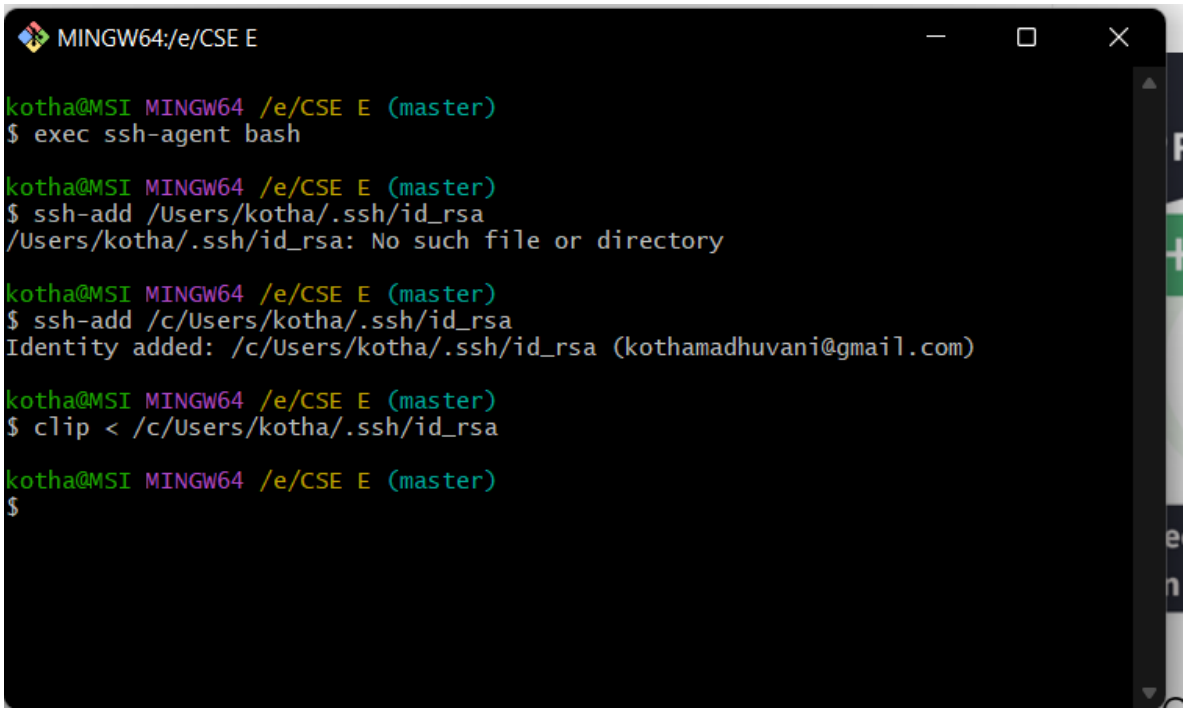
STEPS TO CONNECT WITH SSH:-

1. Start by creating a new key, using your email as a label:

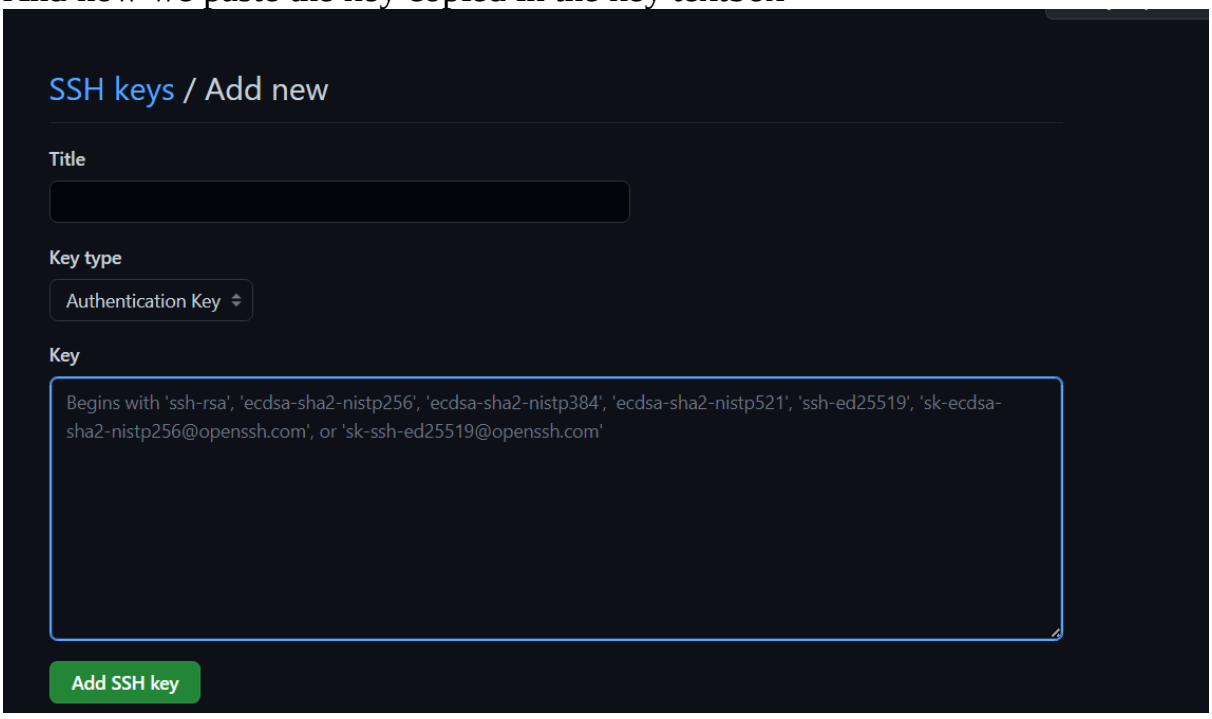


2.

3. Now we add this SSH key pair to the SSH-Agent (using the file location from above):

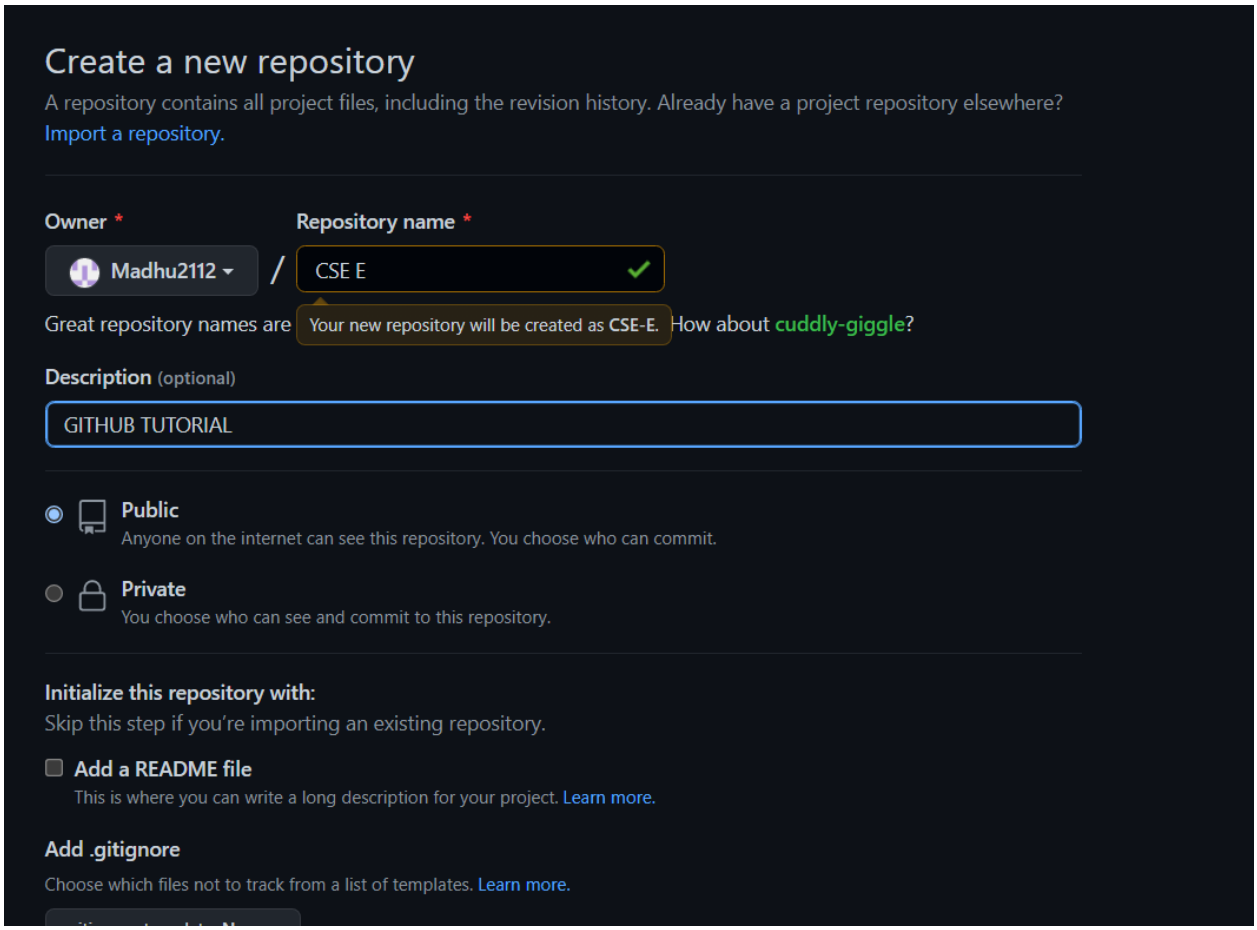


4. And now we paste the key copied in the key textbox

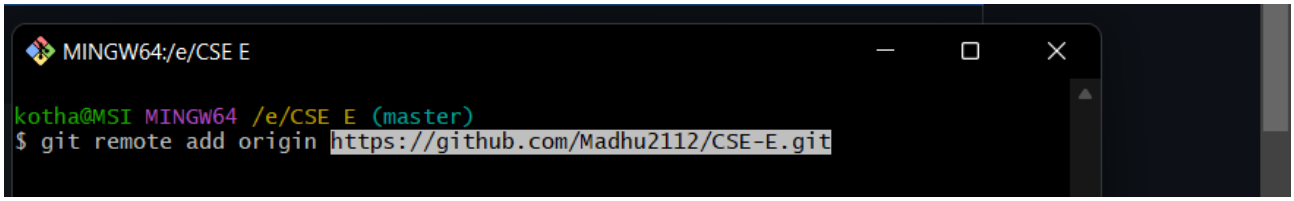


GITHUB

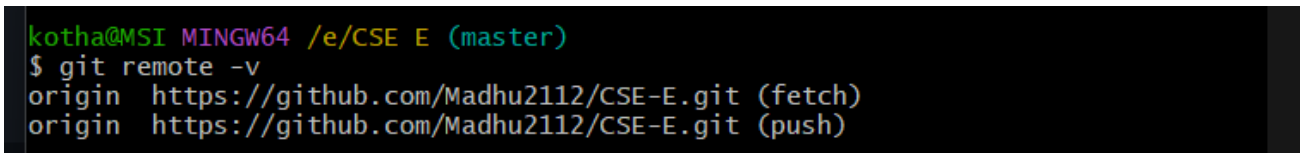
Create a repository



Git remote add origin - obtain the git remote add URL for the remote repository and add credentials if needed.



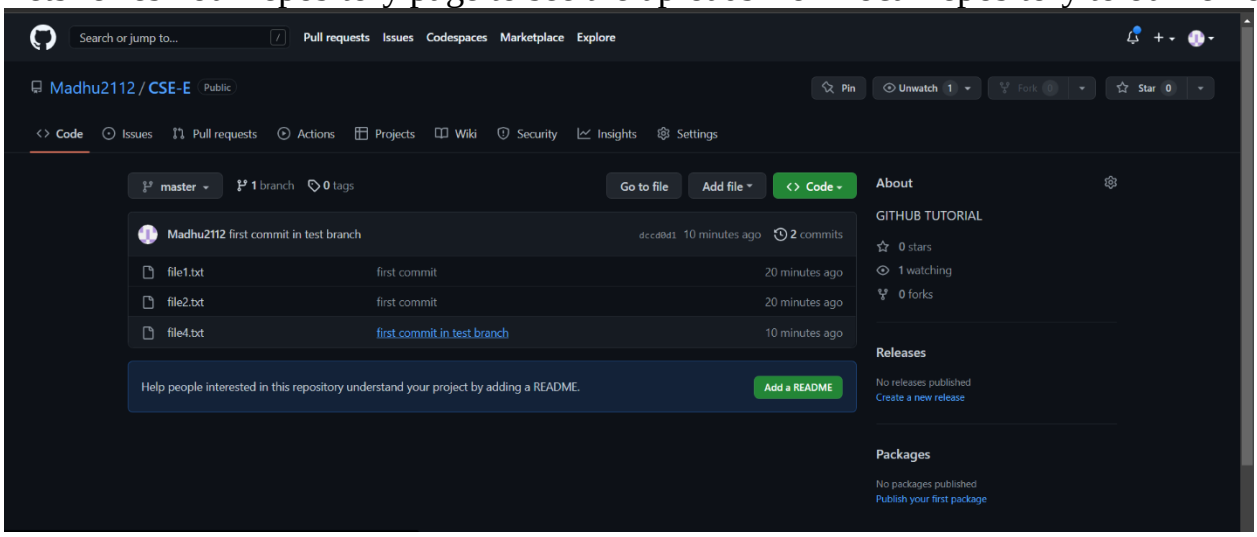
Git remote - v - The git remote command lets you create, view, and delete connections to other repositories.



Git push -u origin master – The command uploads content from a local repository to a remote repository. It is generally used to upload modifications in a local repository with remote team members

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git push -u origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 402 bytes | 402.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Madhu2112/CSE-E.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

Lets refresh our repository page to see the uploads from local repository to our remote repository



A example of a file being committed and pushed to the remote repository.

Touch file5.txt command is creating a new file called file5.txt, git add file5.txt is adding the file to the stage and git commit is acknowledging the changes in the local repository then

```
MINGW64:/e/CSE E

kotha@MSI MINGW64 /e/CSE E (master)
$ touch file5.txt

kotha@MSI MINGW64 /e/CSE E (master)
$ git add file5.txt

kotha@MSI MINGW64 /e/CSE E (master)
$ git commit -m"git push example"
[master 78f4baa] git push example
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file5.txt
```

Using git push command we push the file into our remote repository

```
kotha@MSI MINGW64 /e/CSE E (master)
$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 231 bytes | 231.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Madhu2112/CSE-E.git
   dccd0d1..78f4baa  master -> master
```

Now we can see file5.txt in our remote repository on github

Git clone-

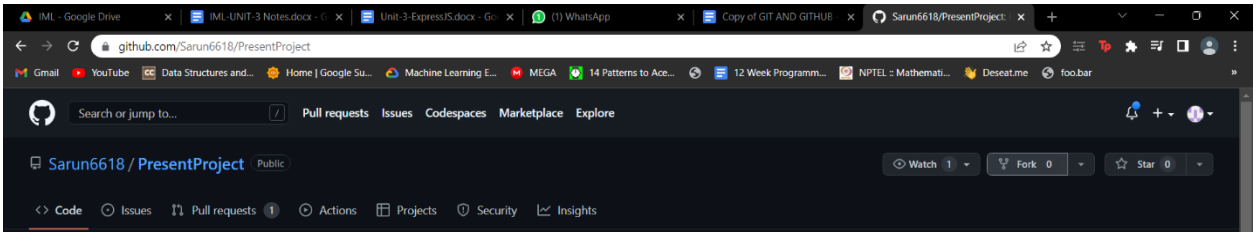
This is primarily used to point to an existing repo and make a clone or copy of that repo at in a new directory, at another location.

Fork a Repository

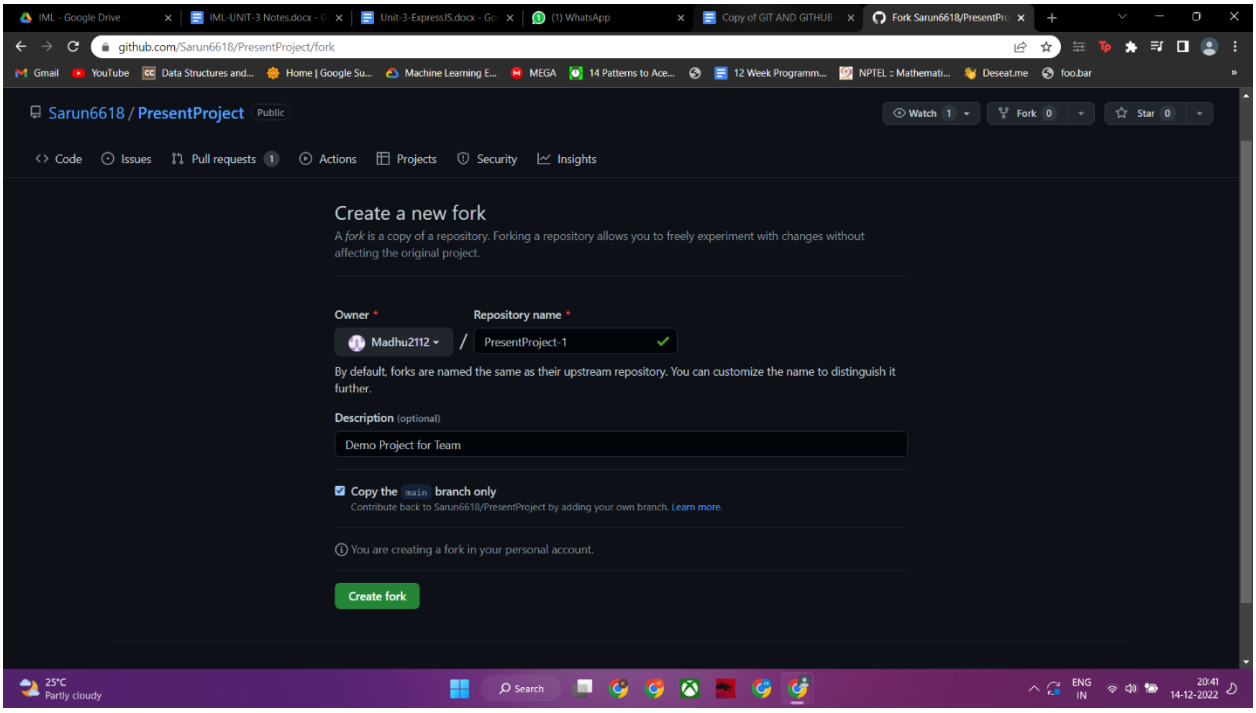
A fork is a copy of a repository. This is useful when you want to contribute to someone else's project or start your own project based on theirs.

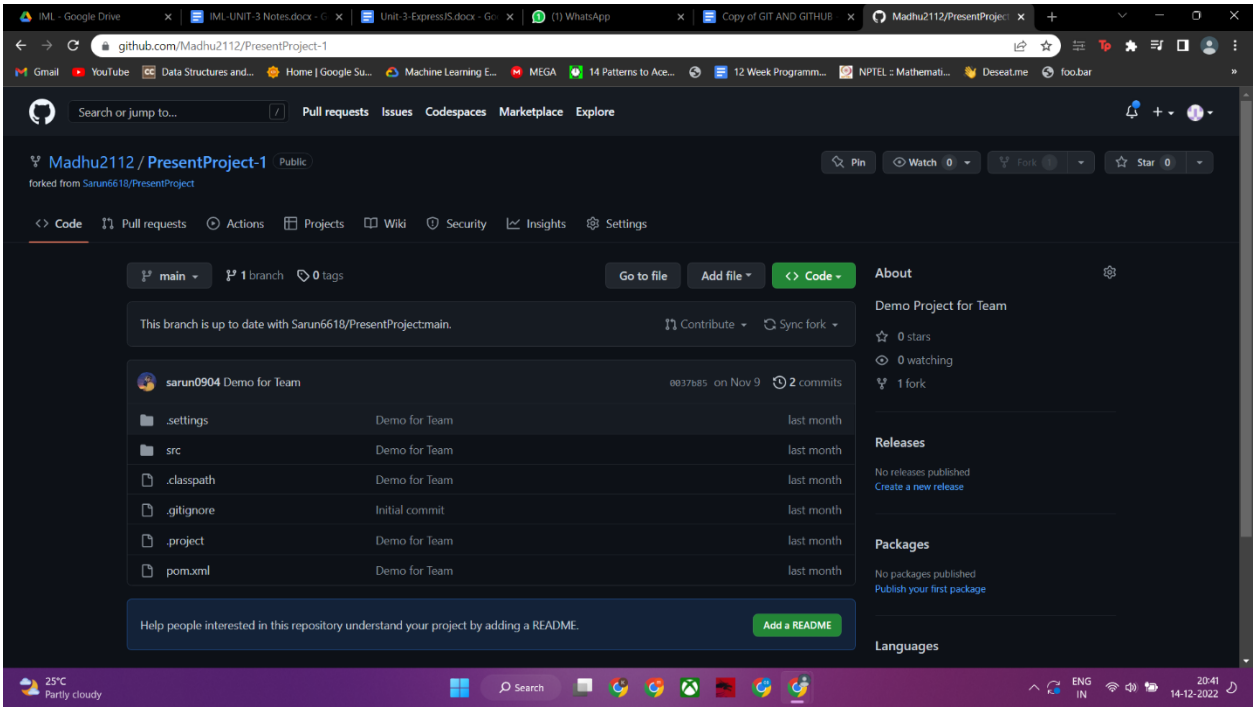
fork is not a command in Git, but something offered in GitHub and other repository hosts. Let's start by logging in to GitHub, and fork our repository:

https://github.com/Sarun6618/PresentProject.git



Now we have our own copy of w3schools-test.github.io:




github.com/Madhu2112/PresentProject-1
Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
Madhu2112 / PresentProject-1
Public
forked from Sarun6618/PresentProject
Pin
Watch 0
Fork 0
Star 0
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
main
1 branch
0 tags
Go to file
Add file
Code
This branch is up to date with Sarun6618/PresentProject:main.
Contribute
Sync fork
sarun0904 Demo for Team
0037b85 on Nov 9
2 commits
.settings
Demo for Team
last month
src
Demo for Team
last month
.classpath
Demo for Team
last month
.gitignore
Initial commit
last month
.project
Demo for Team
last month
pom.xml
Demo for Team
last month
Help people interested in this repository understand your project by adding a README.
Add a README
About
Demo Project for Team
0 stars
0 watching
1 fork
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
25°C
Partly cloudy
Search
ENG
IN
20:41
14-12-2022

3.Working with Maven in Eclipse

Maven Objectives-

Maven is a project development management and comprehension tool. Based on the concept of a project object model: builds, dependency management, documentation creation, site publication, and distribution publication are all controlled from the pom.xml declarative file. Maven can be extended by plugins to utilise a number of other development tools for reporting or the build process.

Plugins-

Plugins are the central feature of Maven that allow for the reuse of common build logic across multiple projects. They do this by executing an "action" in the context of a project's description - the Project Object Model (POM).

Types of Plugins-

1.Build plugins will be executed during the build and they should be configured in the <build/> element from the POM.

2.Reporting plugins will be executed during the site generation and they should be configured in the <reporting/> element from the POM. Because the result of a Reporting plugin is part of the generated site, Reporting plugins should be both internationalized and localized. You can read more about the localization of our plugins and how you can help

POM XML-

A Project Object Model or POM is the fundamental unit of work in Maven. It is an XML file that contains information about the project and configuration details used by Maven to build the project. It contains default values for most projects.

Types of POM XML-

1.project- It is the root element of pom.xml file.

2.groupid- It is the sub element of project. It specifies the id for the project group.

3.artifactid- It is the sub element of project. It specifies the id for the artifact

4.version- It is the sub element of project. It specifies the version of the artifact under given group.

Dependencies-

Dependency management is a core feature of Maven. Managing dependencies for a single project is easy. Managing dependencies for multi-module projects and applications that consist of hundreds of modules is possible.

Types of Dependencies-

1.Direct dependencies-Direct dependencies are the ones that we explicitly include in the project.

2.Transitive dependencies-Transitive dependencies are required by direct dependencies and maven automatically includes the required transitive dependencies in our project.

Artifacts-

An Artifact is any file that can be addressed using its coordinates, and Maven downloads, installs or deploys for you. Most of them are POMs and JARs but an artifact can be really anything. A very important thing about artifacts is that they have coordinates, so they are not just files, but they are files that are in some way addressable by Maven.

Properties-

Name	Description
groupId	The artifact group
artifactId	The artifact id
version	The artifact version (linked w/ baseVersion)
baseVersion	The artifact base version (linked w/ version)
classifier	The artifact distinguishing classifier (optional)
extension	The artifact extension (jar)

Maven Goals-

A build phase is made up of a set of goals. Maven goals represent a specific task that contributes to the building and managing of a project. Sometimes, a maven goal is not bound to a build phase. We can execute these goals through the command line.

In Eclipse, the following maven goal commands are required:-

1.Maven Clean-Maven Clean cleans the project and removes all the files generated by the previous build.

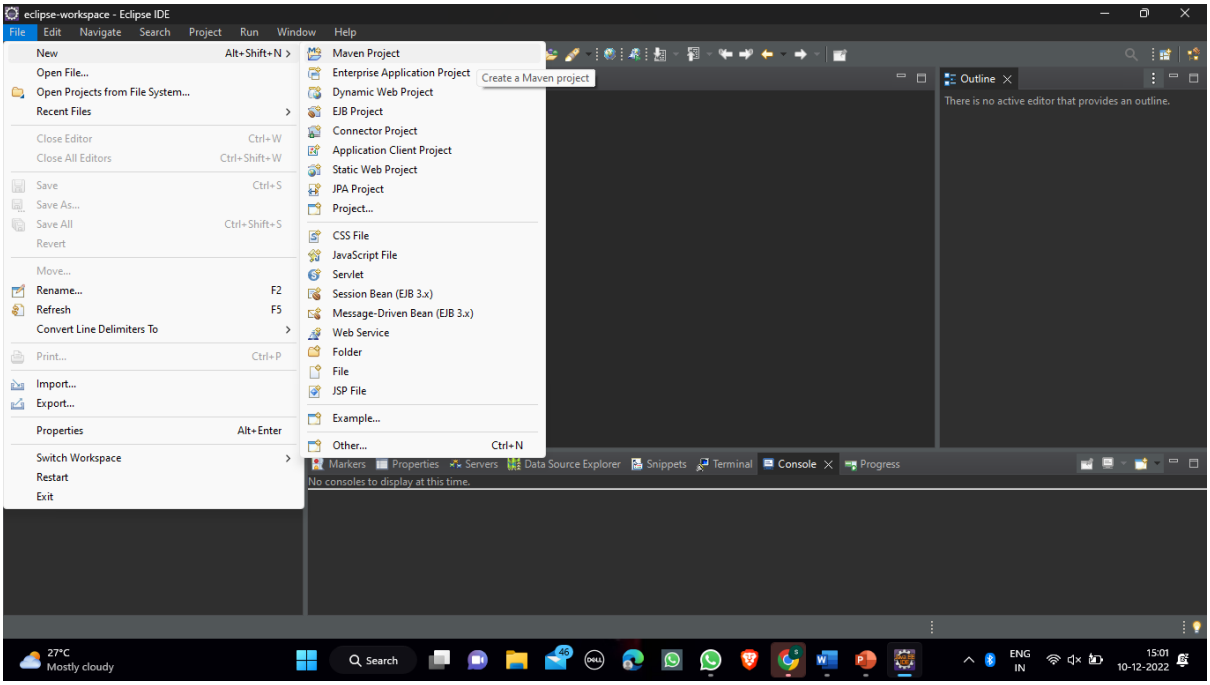
2.Maven Install-Maven Install command builds the maven project and install the project files(jar,war,pom.xml,etc) to local repository.

3.Maven Test-Maven Test command is used to run the test cases of the project using maven-surefire-plugin.

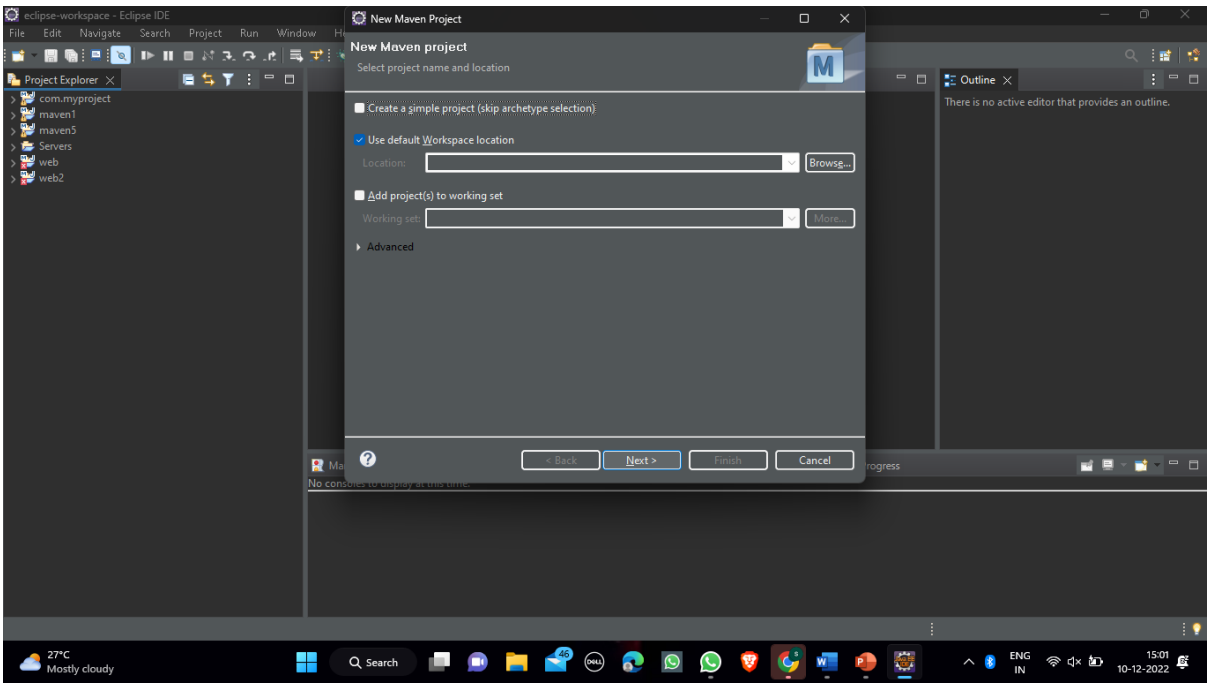
4.Maven Build-Maven Build command is used to generate and compile the source code.

Maven Java Project-

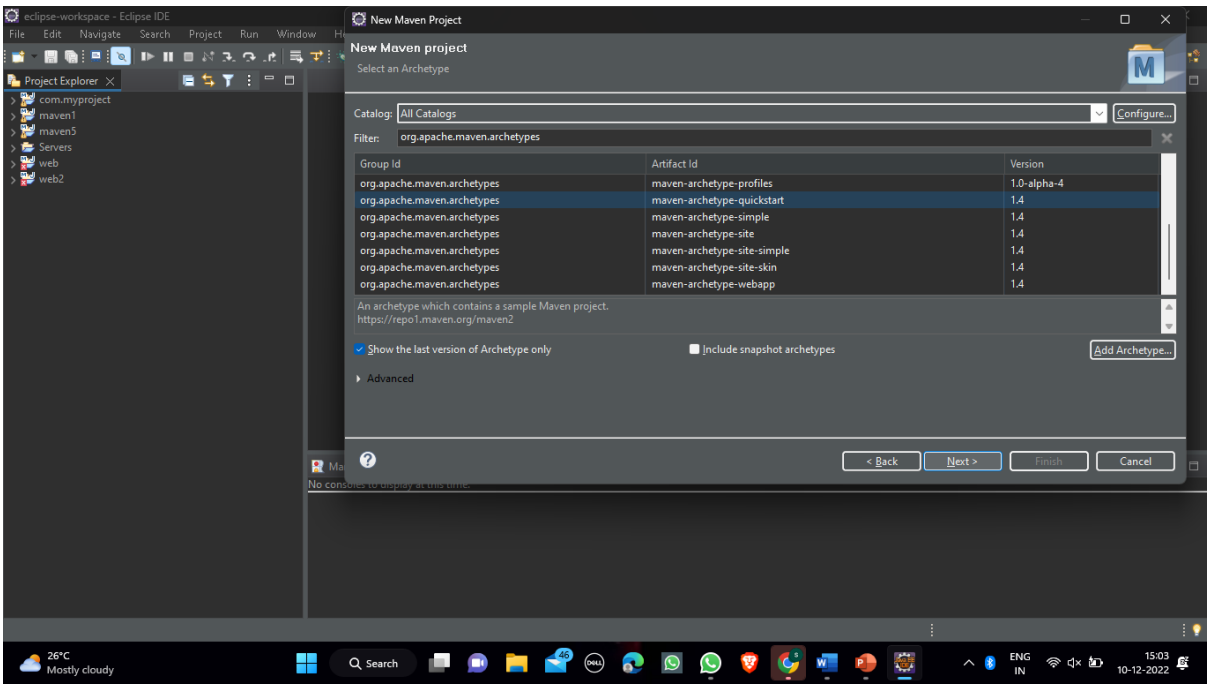
Step1-Go to File->New->Maven Project.



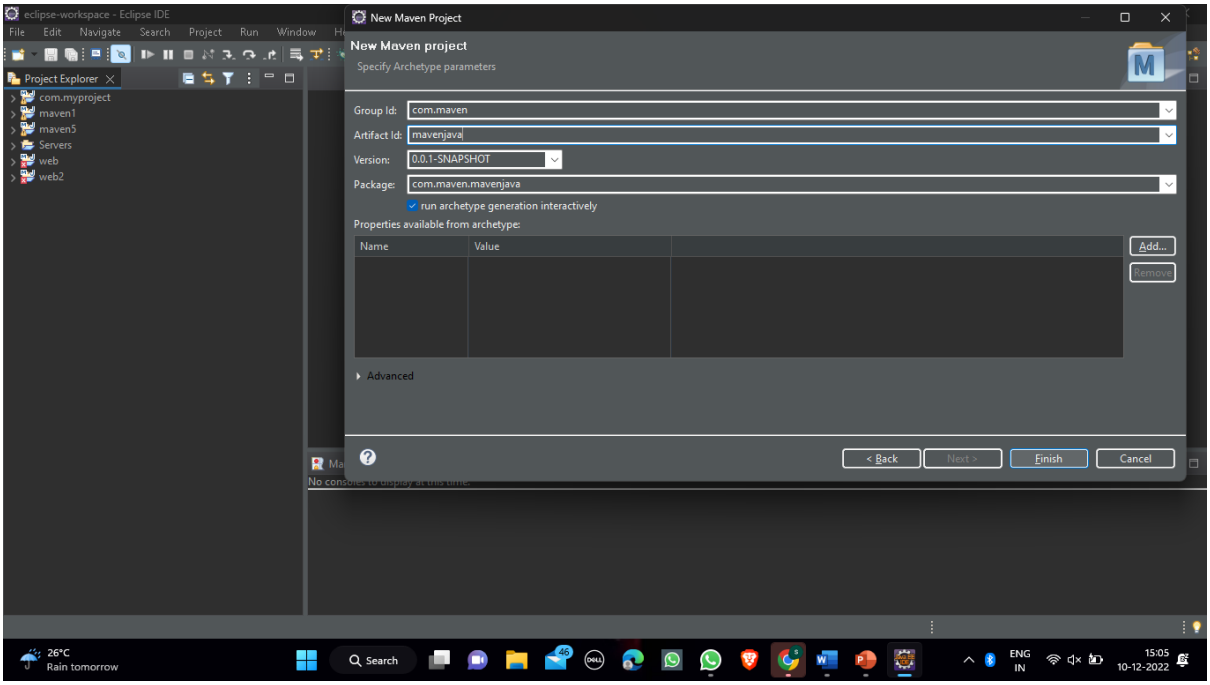
Step 2- Click Next->.



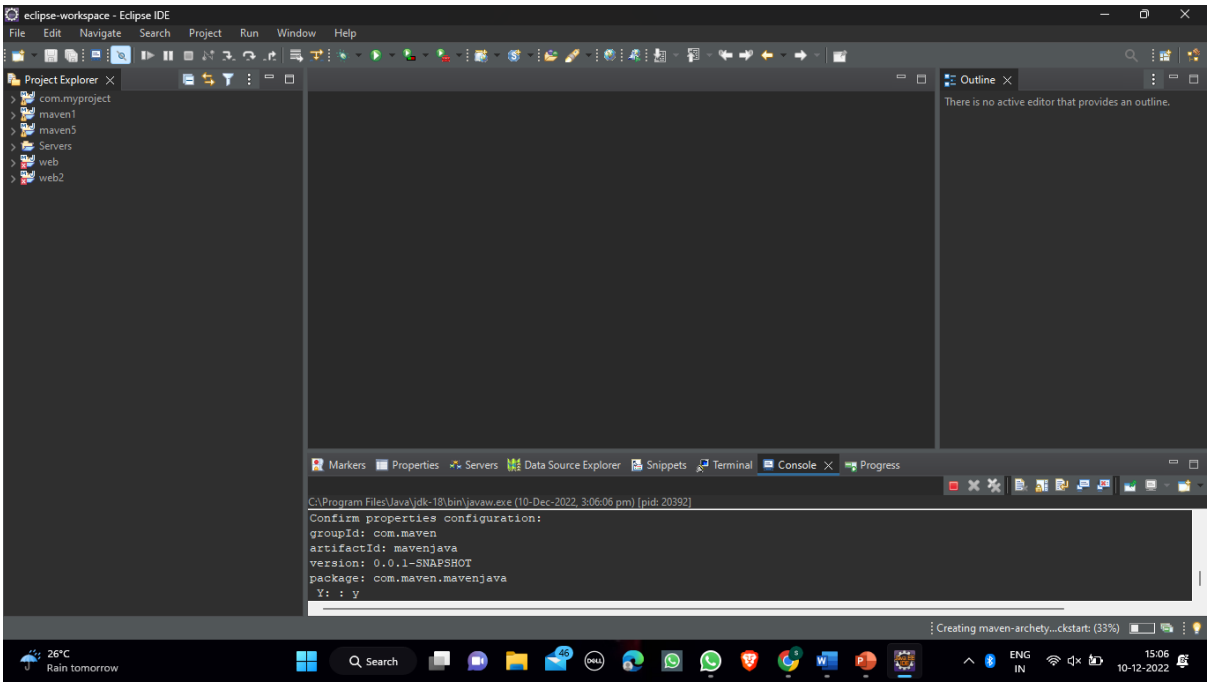
Step 3-Select the org.apache.maven.archetypes with quickstart archetype and click Next.



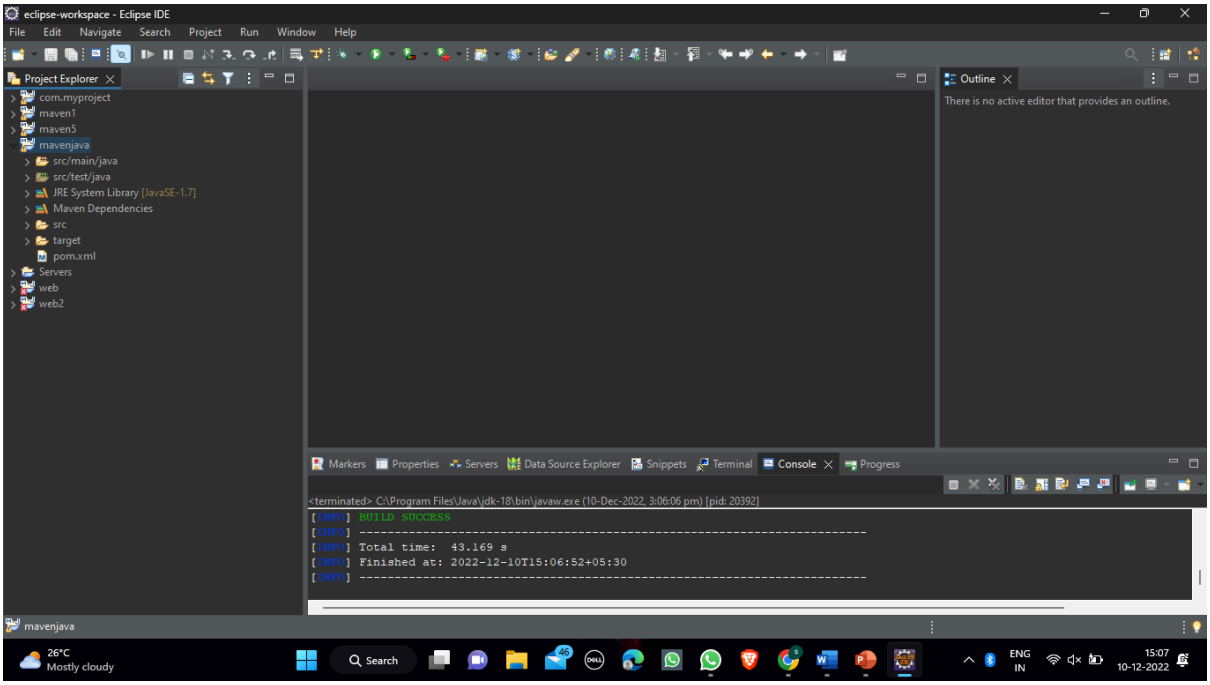
Step 4-Provide the Group id,Artifact if(Filename) and Click finish ,Maven Java project is being created .



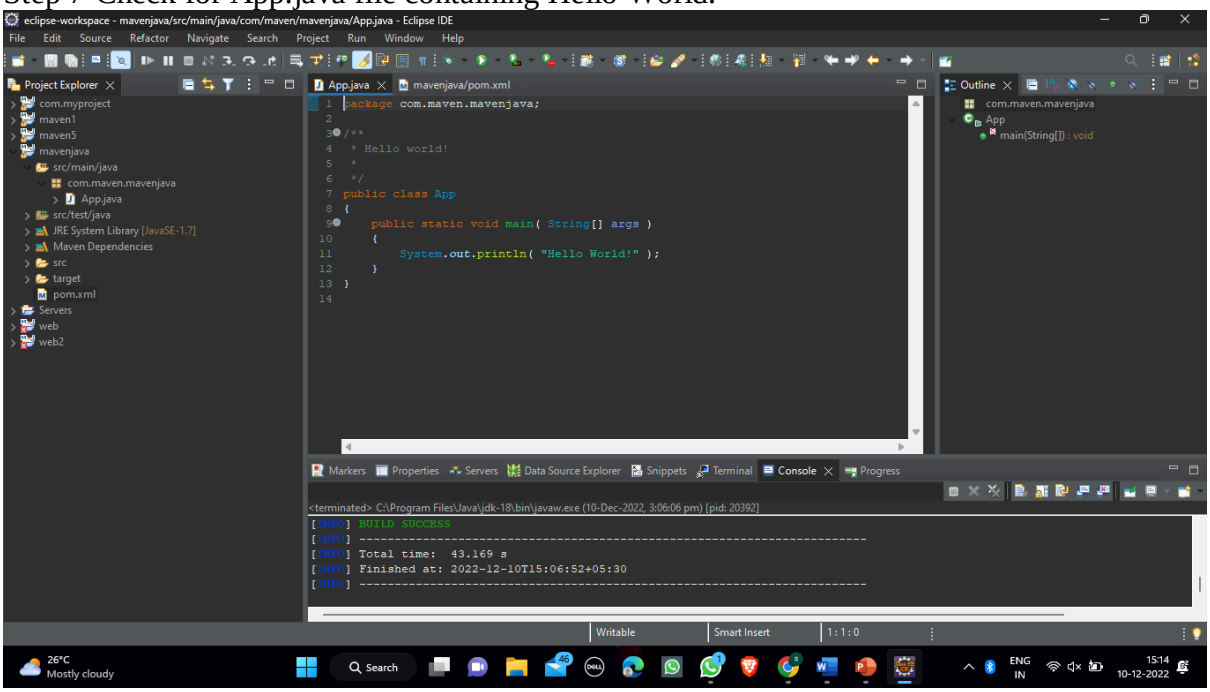
Step 5-In Console click y to create the Maven Java Project.



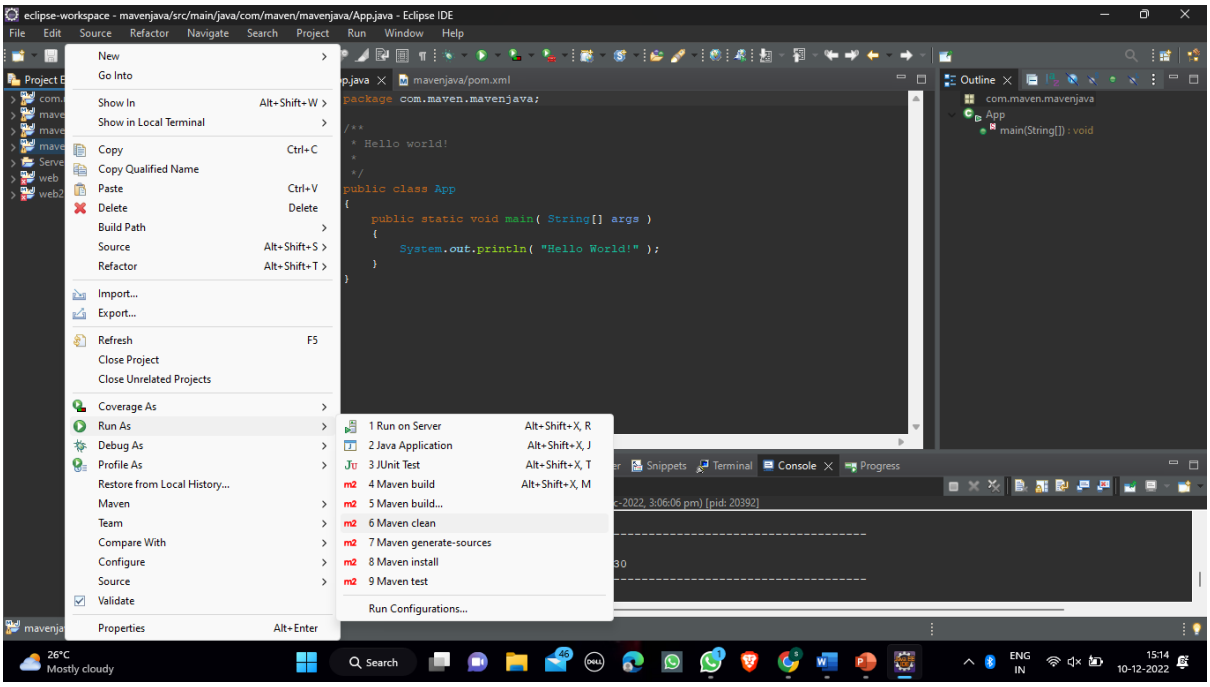
Step 6-Maven Java Project is created with Filename as mavenjava.



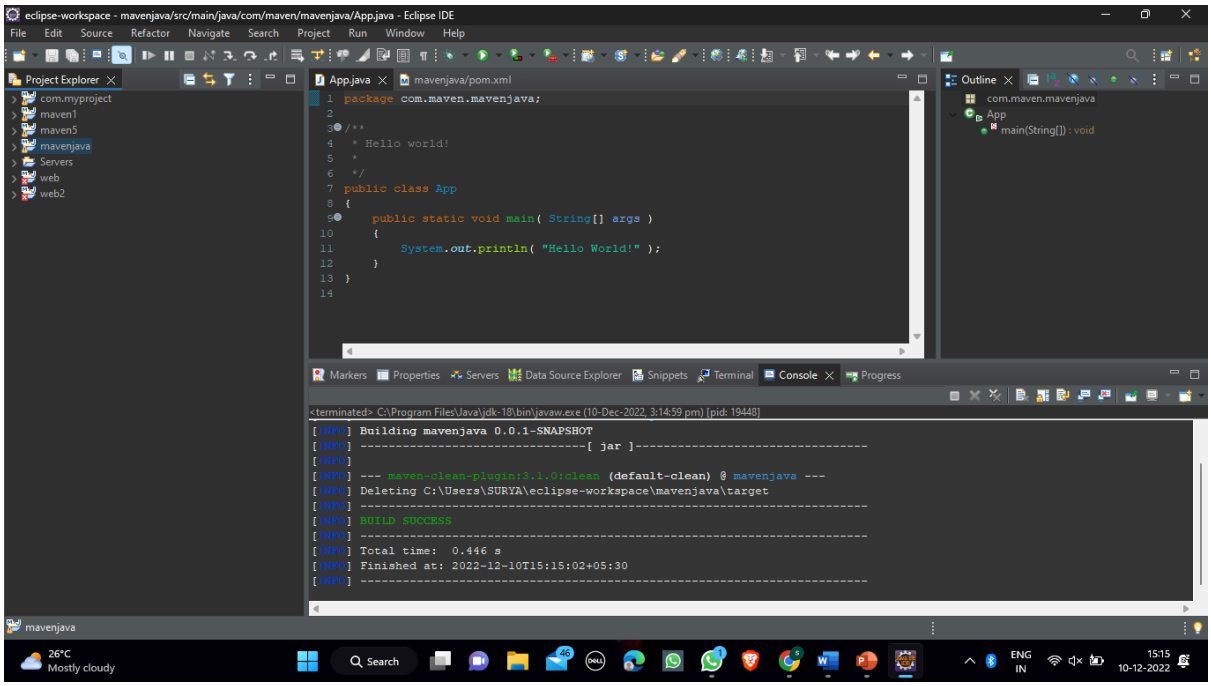
Step 7-Check for App.java file containing Hello World.



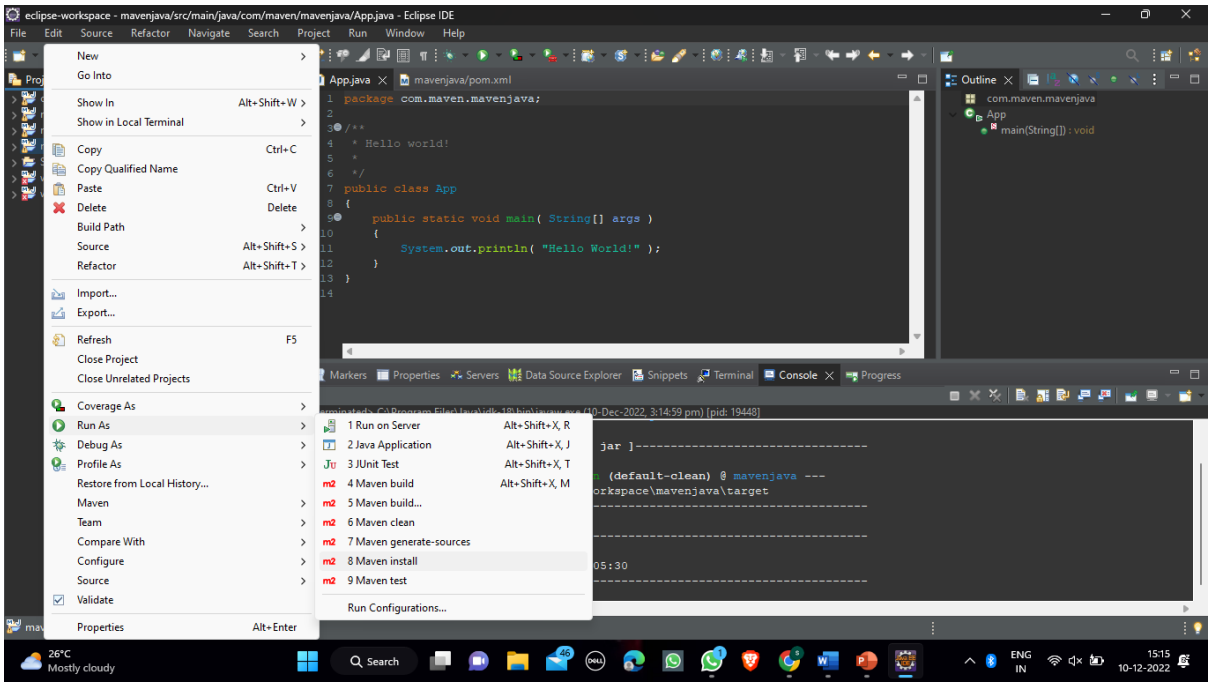
Step 8-First right click on mavenjava and then on Run As and Click on Maven clean.



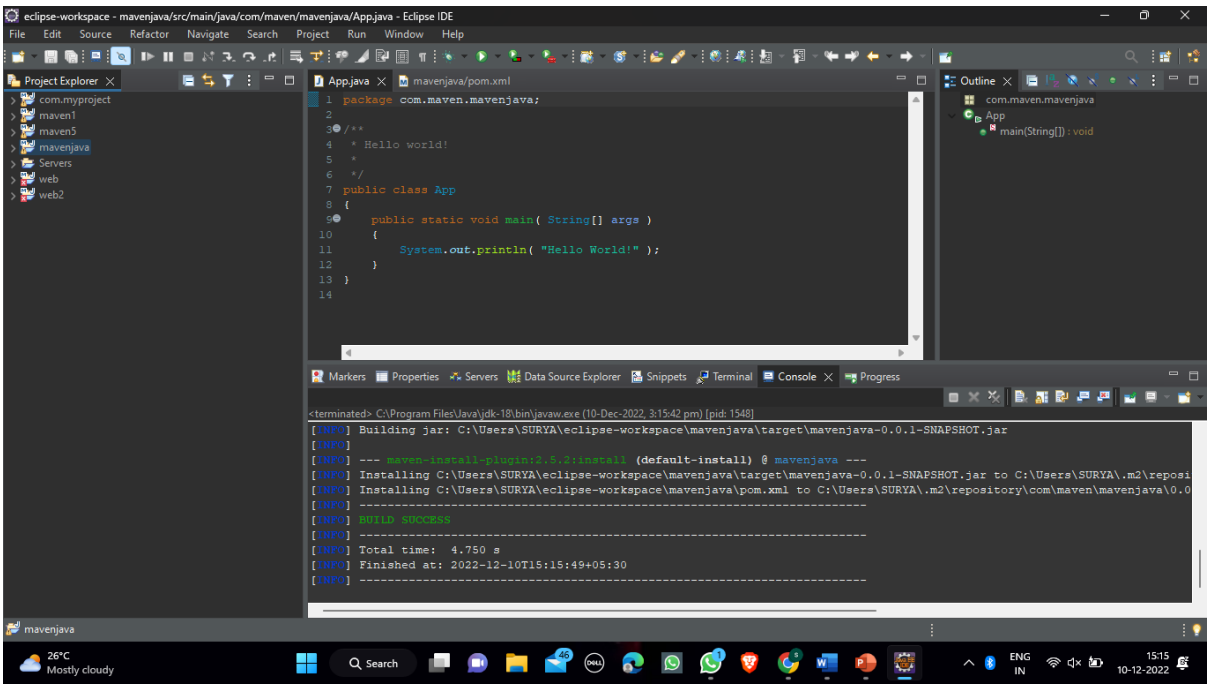
Step 9-After using Maven clean ,it cleans out the existing classes.Here we can see the Console for Build Success .



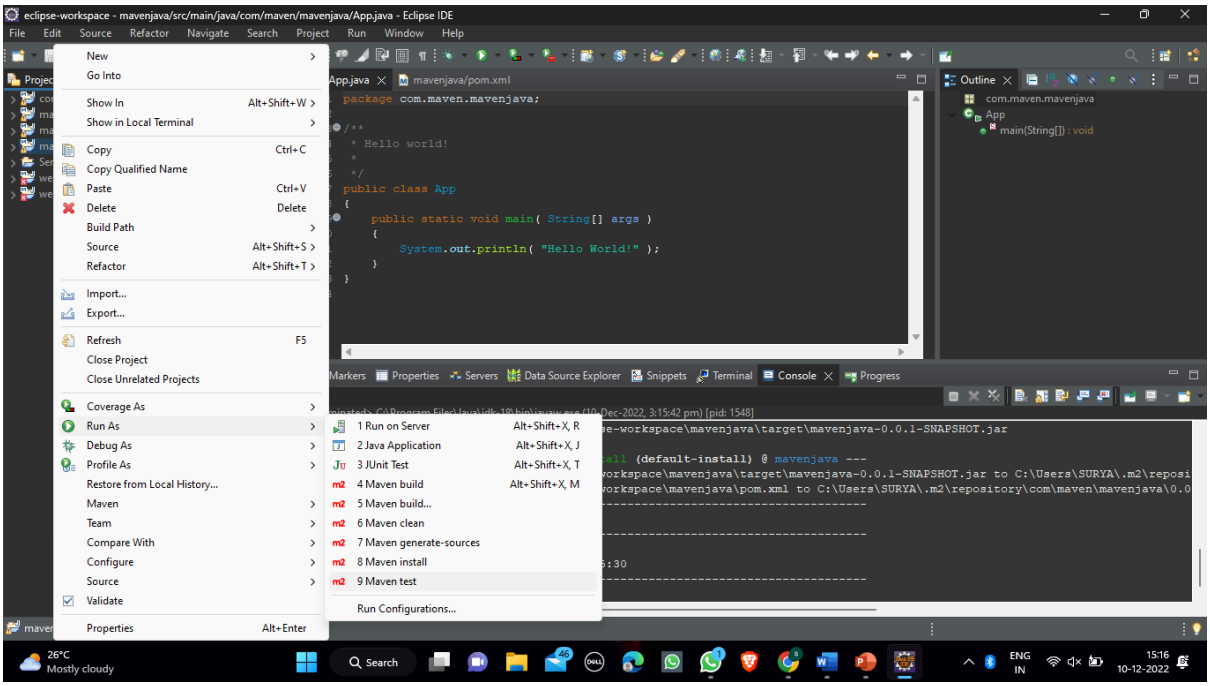
Step 10-Repeat the step 8 until Run As and click on Maven install to add artifacts to the project.



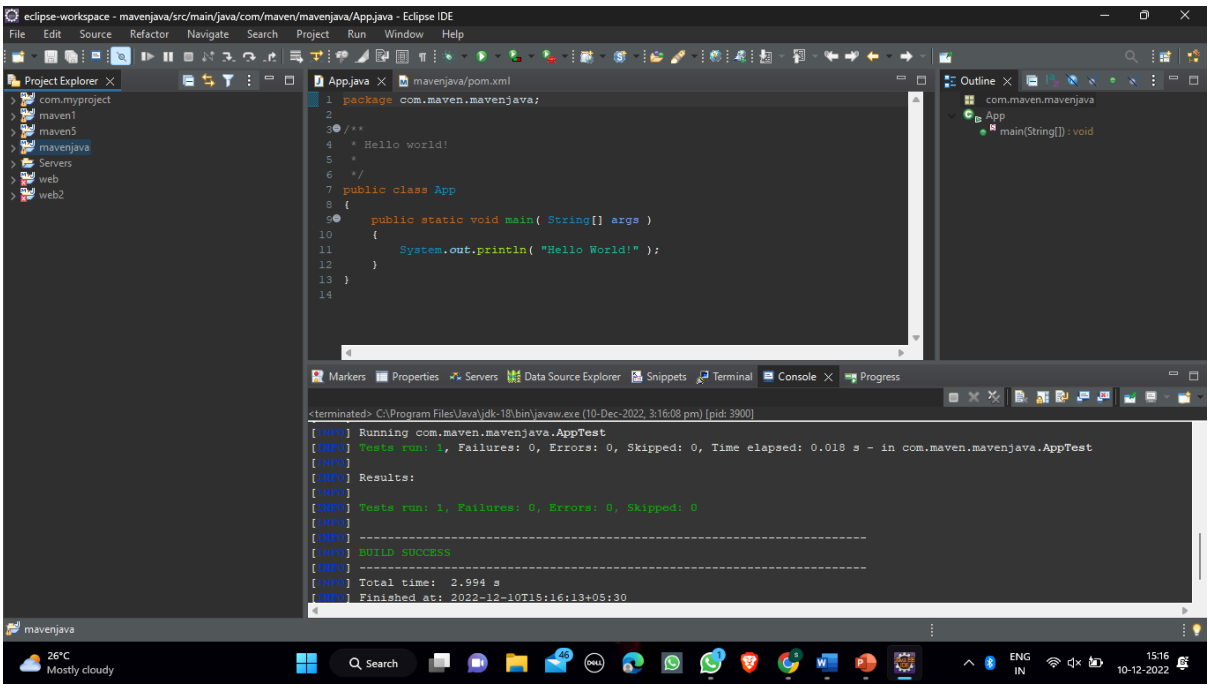
Step 11. Here we can see the Console for Build Success .



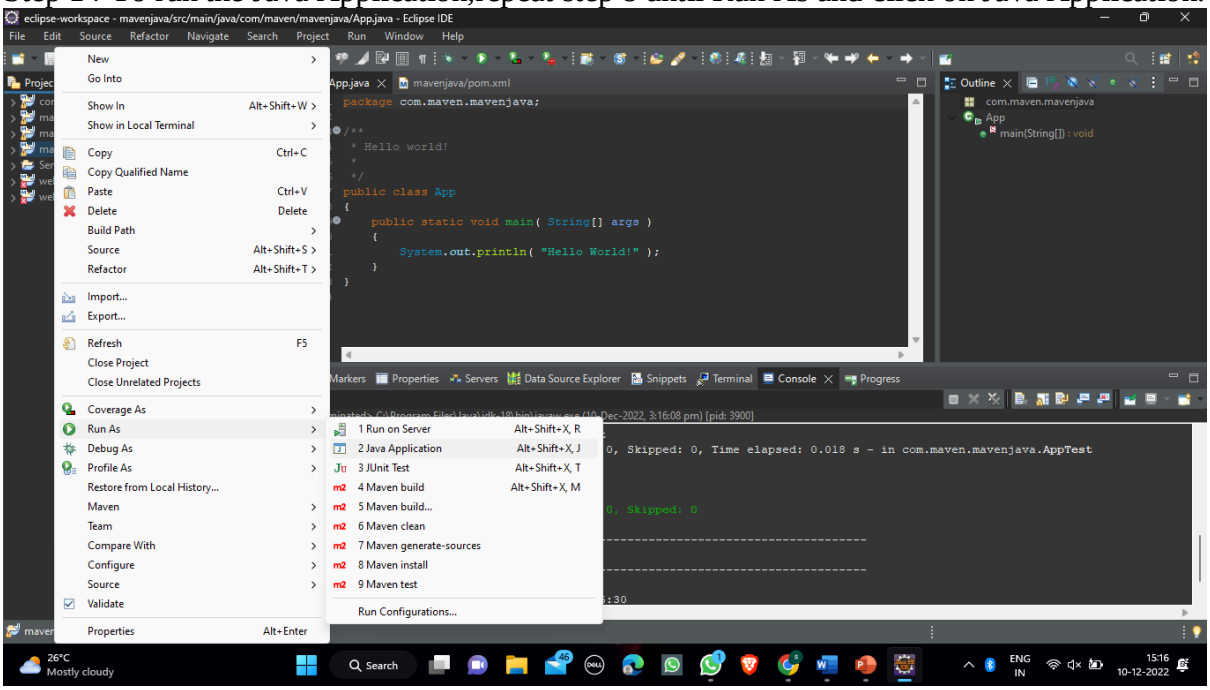
Step 12-Repeat the Step 8 until Run As and click on Maven test .



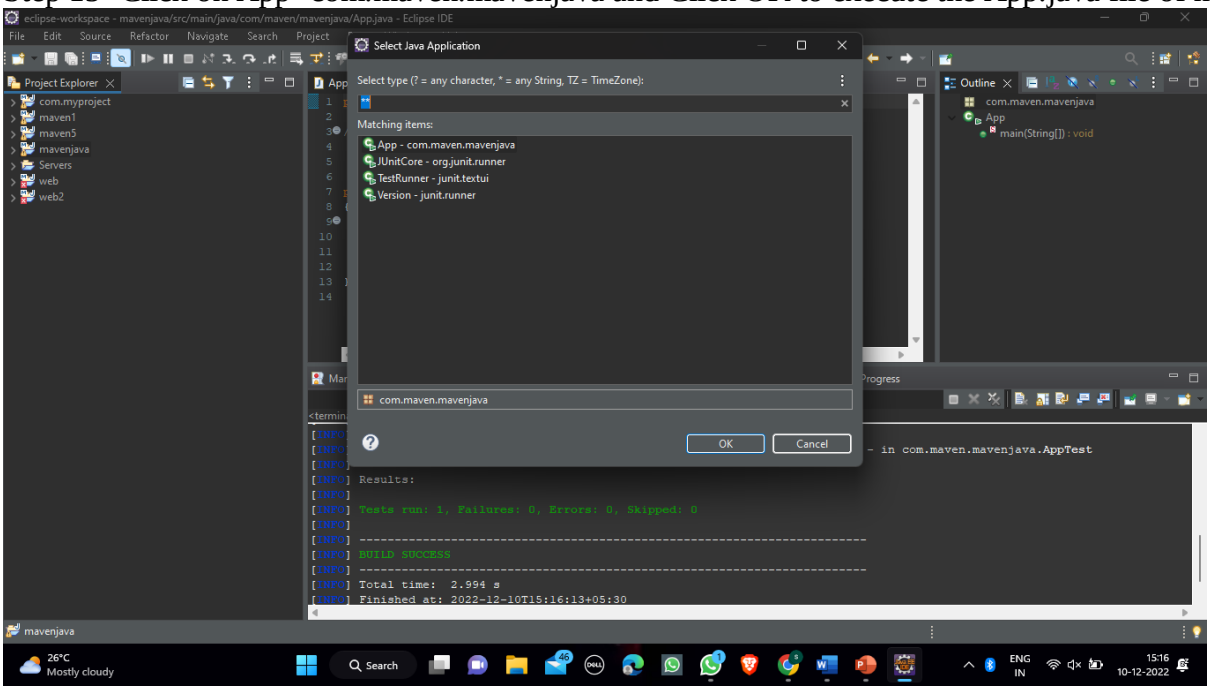
Step 13-Here we can see the Console for build Success.



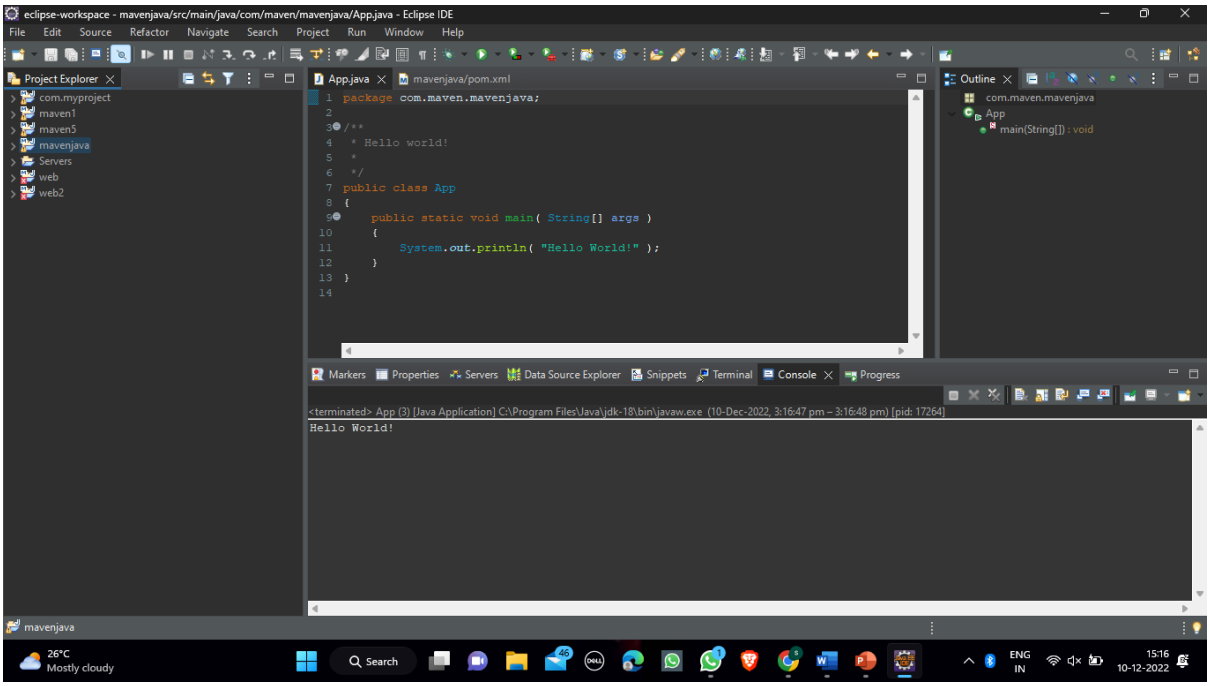
Step 14-To run the Java Application,repeat step 8 until Run As and click on Java Application.



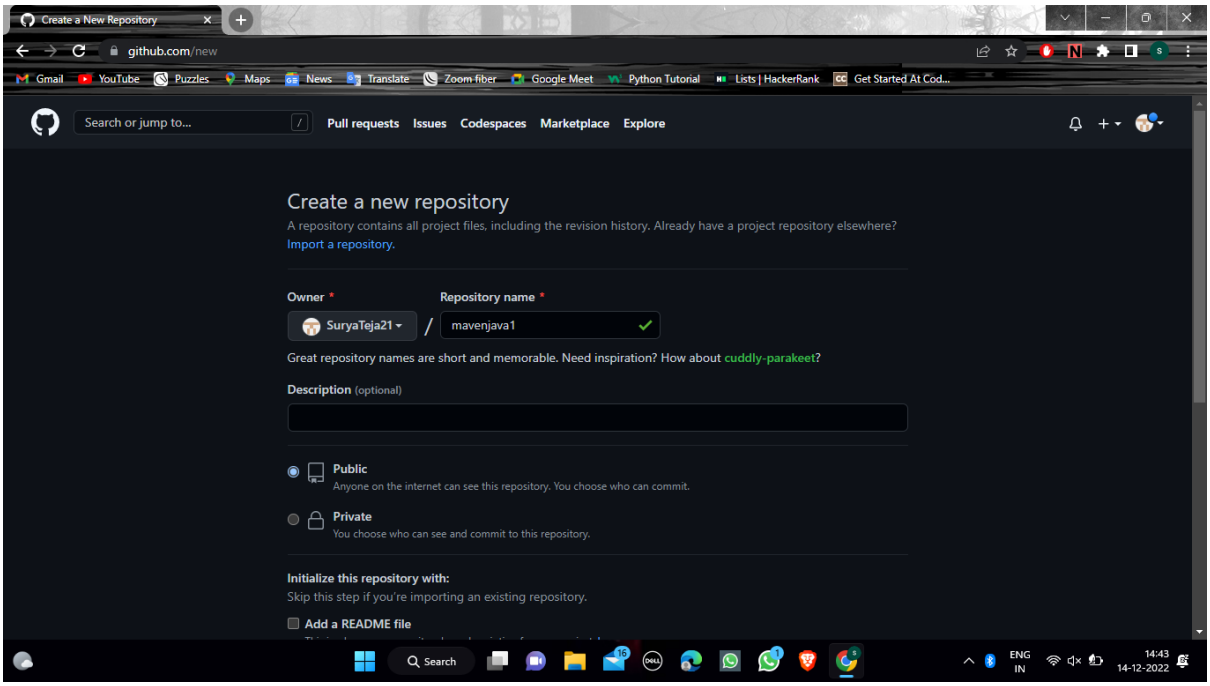
Step 15- Click on App- com.maven.mavenjava and Click OK to execute the App.java file of mavenjava project.



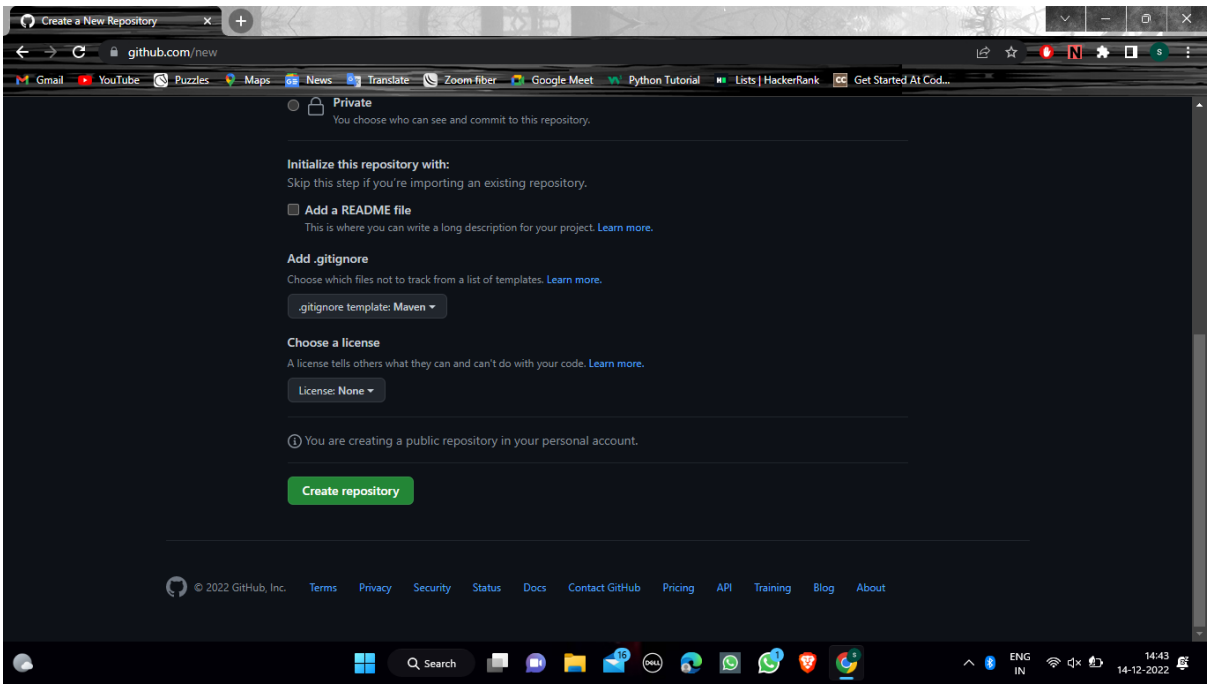
Step 16-In Console,We can see the Output Hello World!.



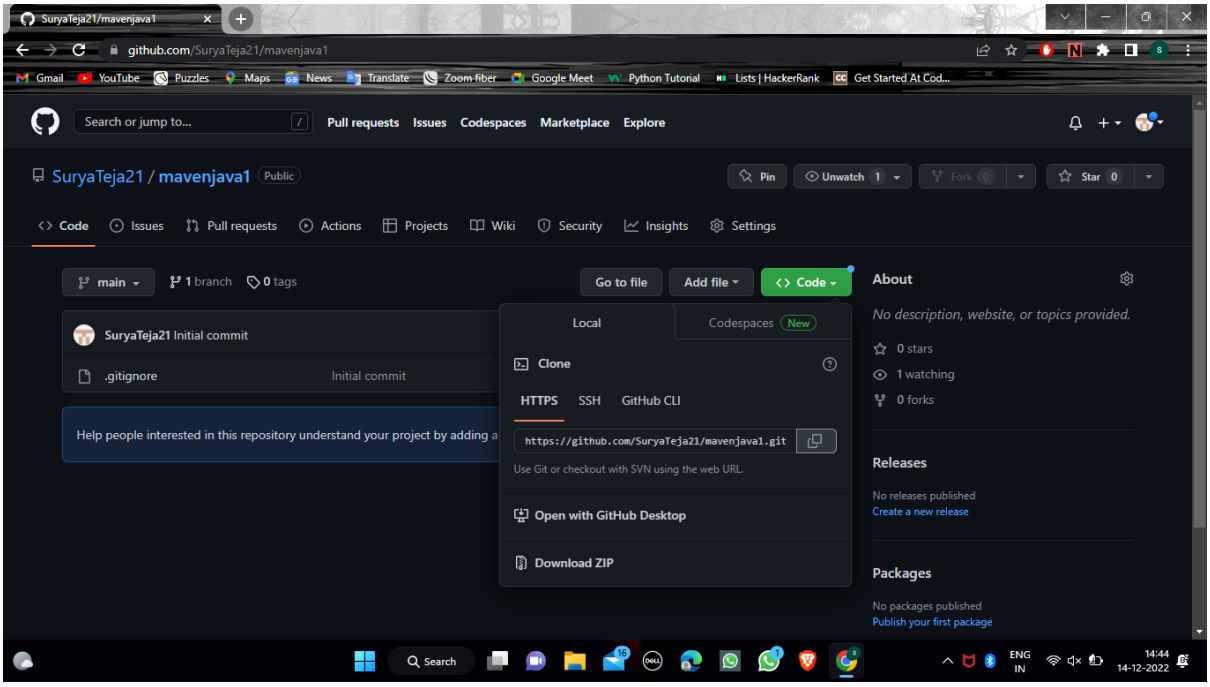
Step 17-Create a new repository with repository name mavenjava1.



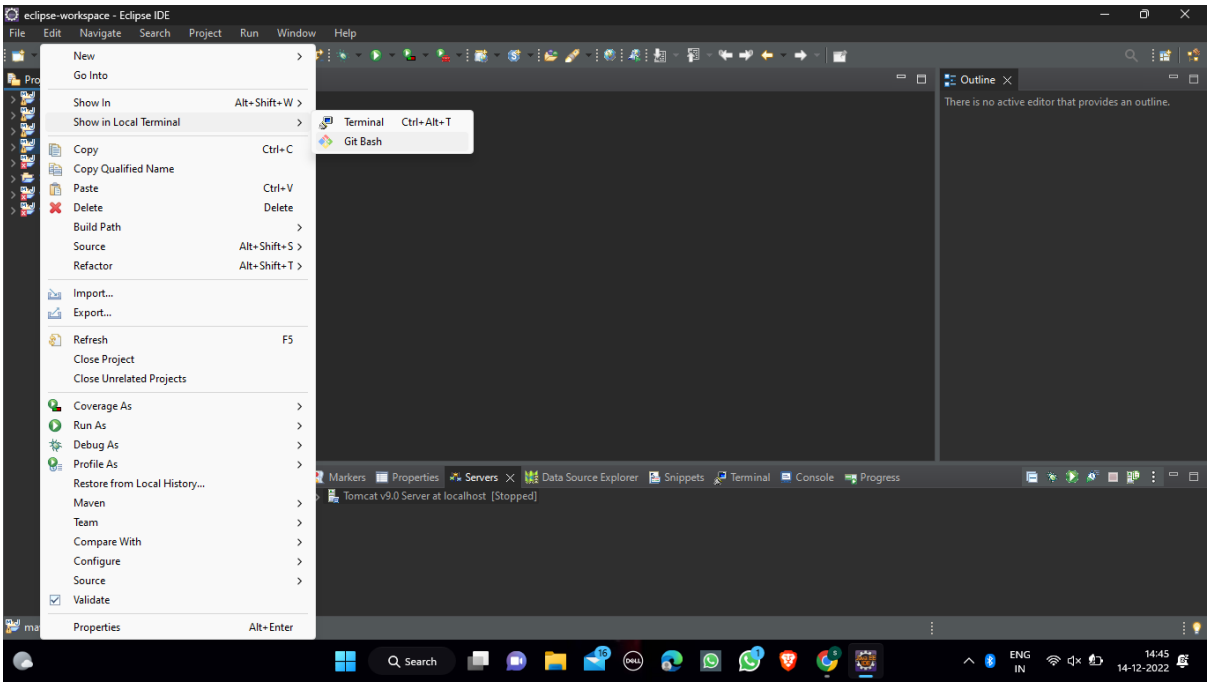
Step 18-While creating the repository add .gitignore template and click on maven.



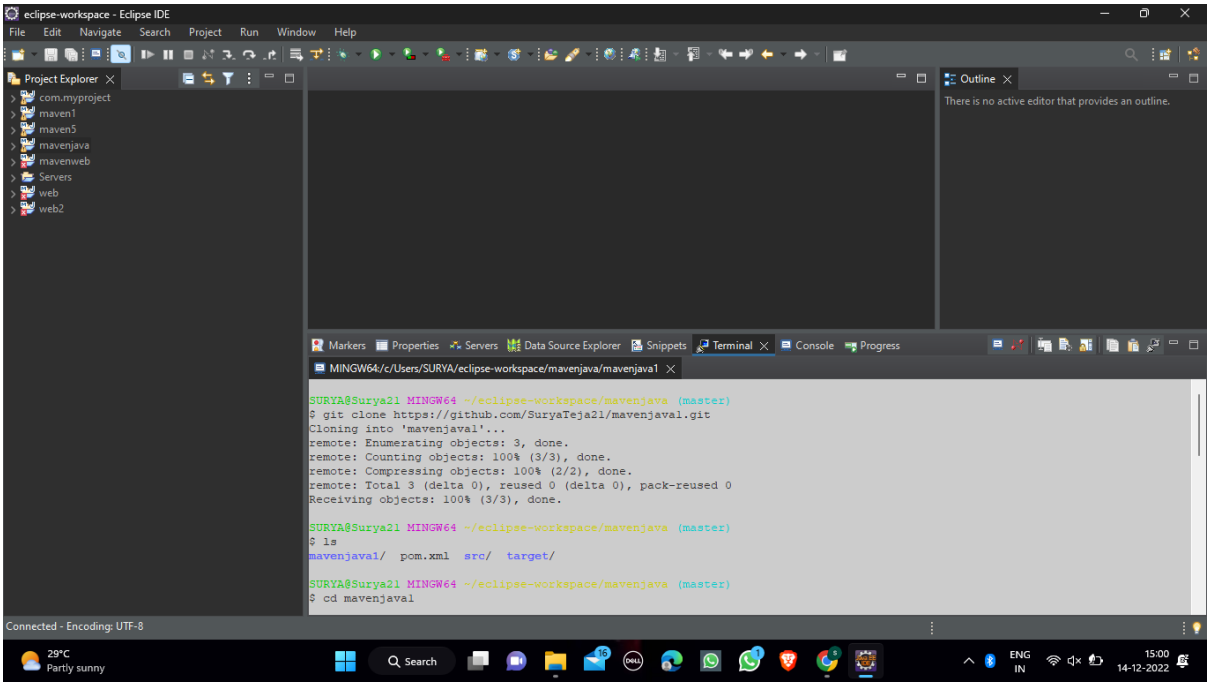
Step 19- Repository is create,now click on code and copy the https link as shown.



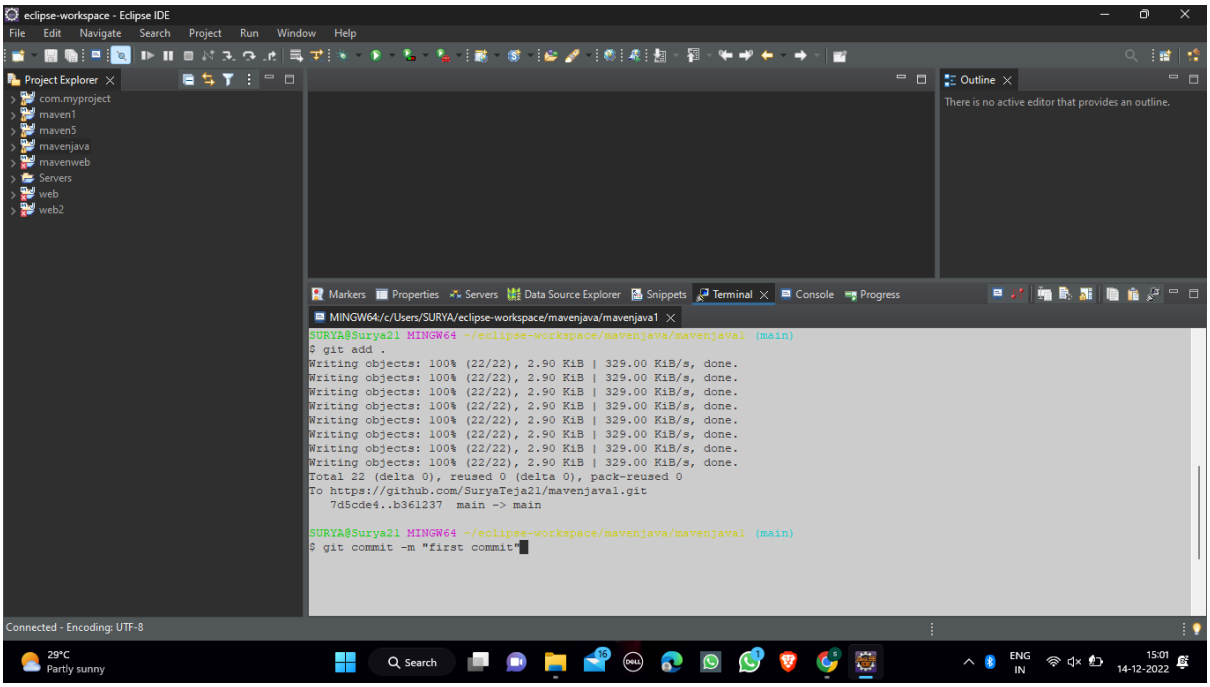
Step 20-In Eclipse,right click on mavenjava project->Click on Show in Local Terminal->Click on Git Bash.



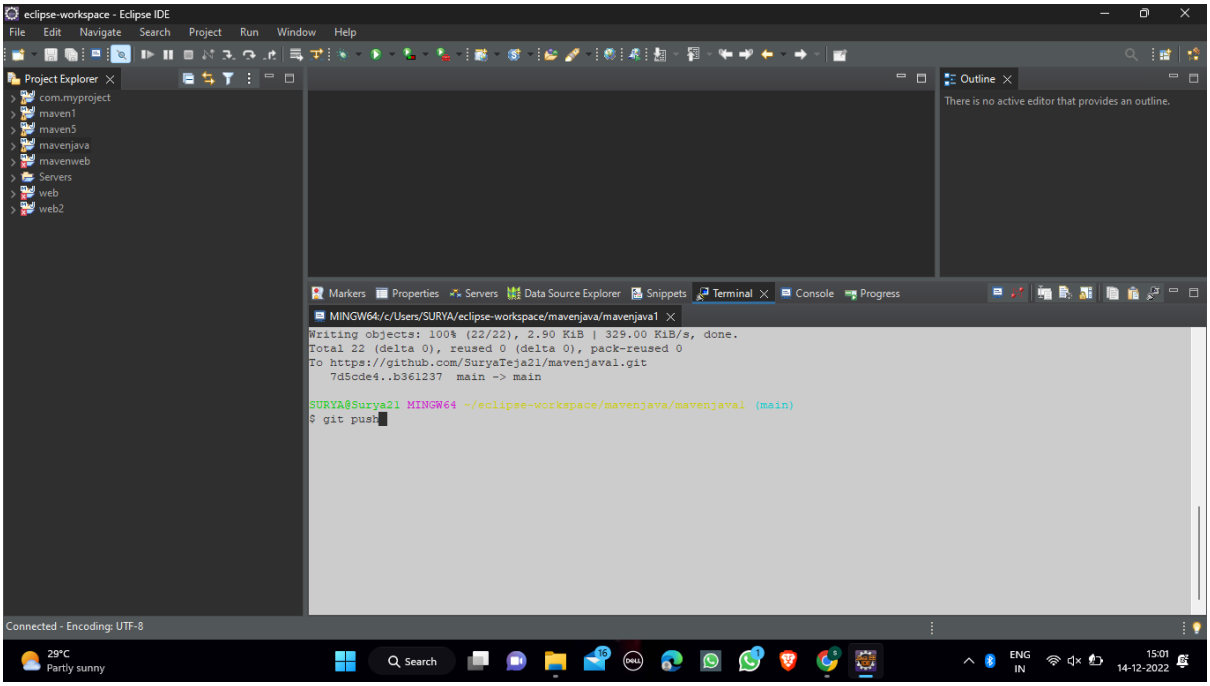
Step 21-In terminal,Git Bash is created.Use git clone command and paste the github repository link. Use cd mavenjava1 and copy the pom.xml ,src/ ,target/ files and paste in the mavenjava1 file.



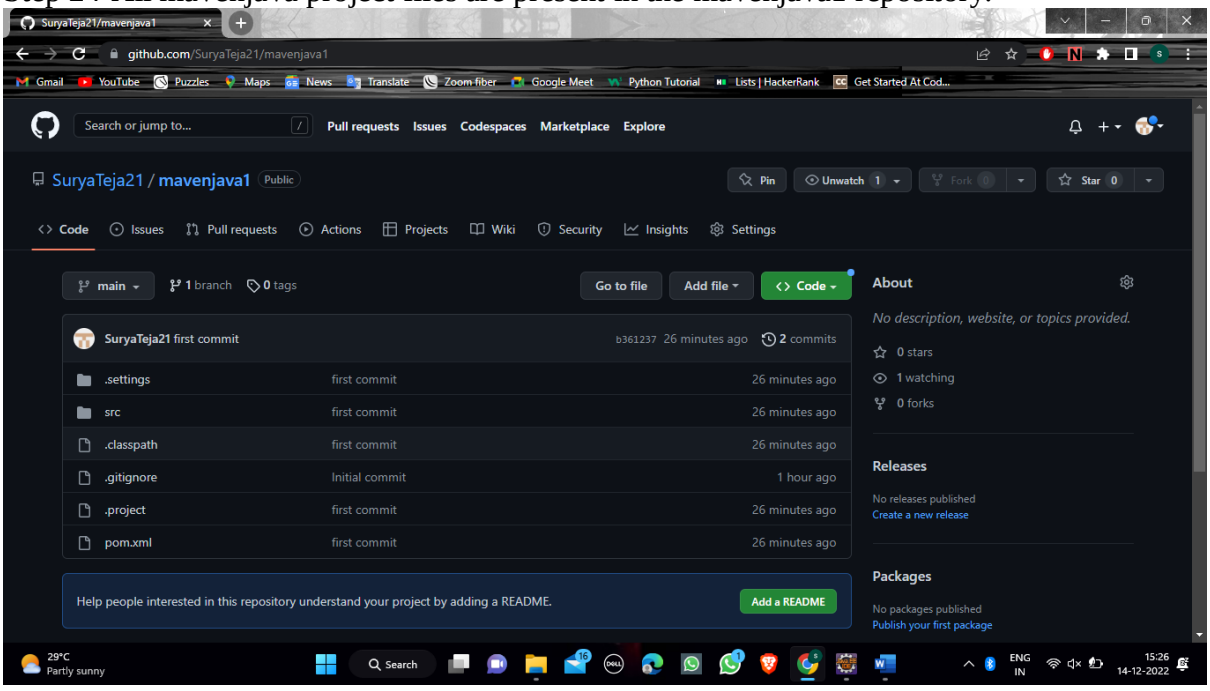
Step 22-After copying the files,use git add . command to add the files.Use git commit -m “first commit” command to save the changes in the local repository.



Step 23-Using git push command we push all the files into github repository mavenjava1.

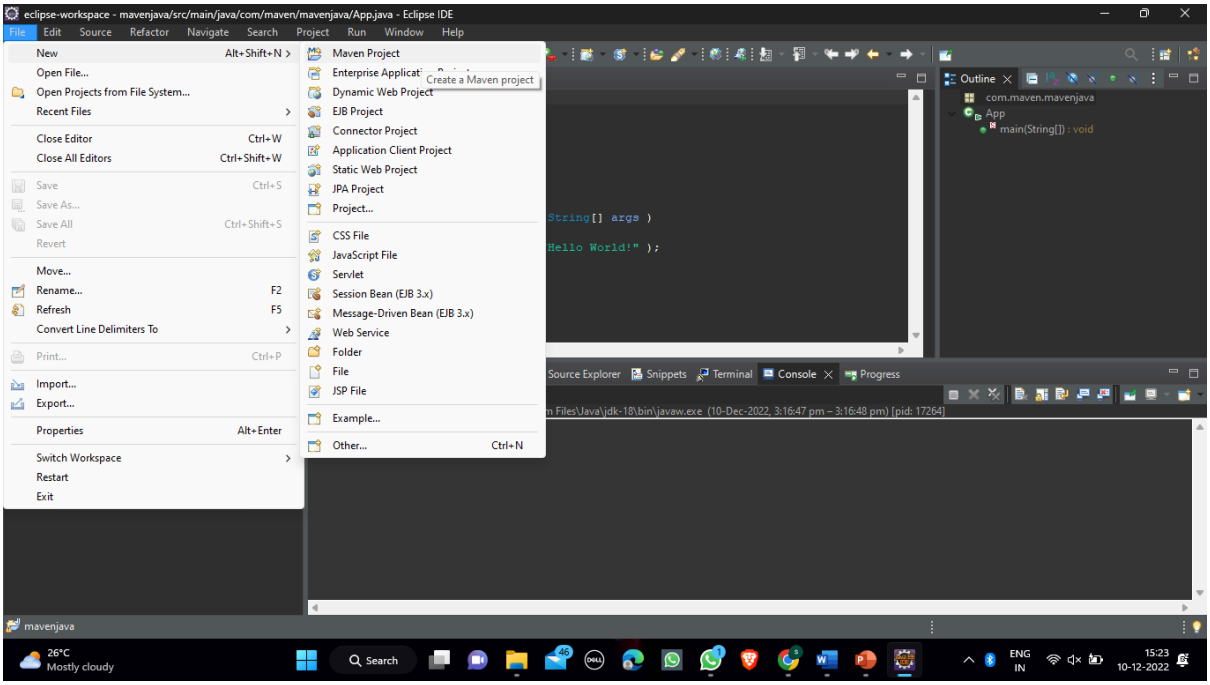


Step 24-All mavenjava project files are present in the mavenjava1 repository.

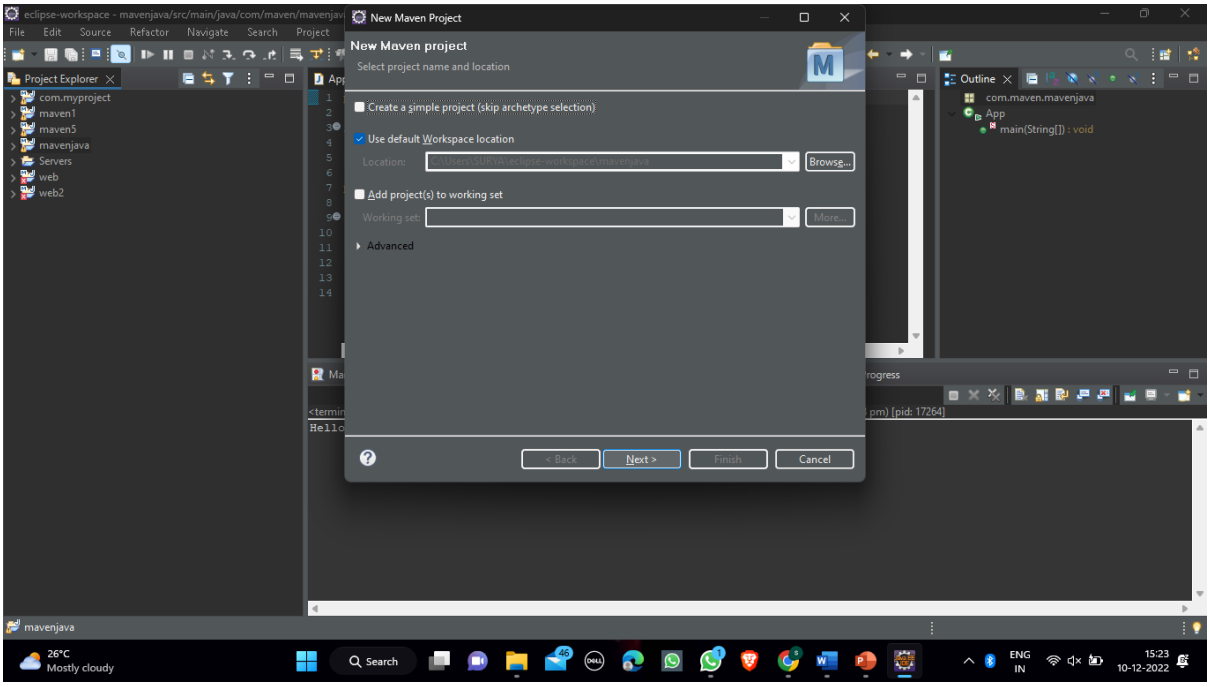


Maven Web Project-

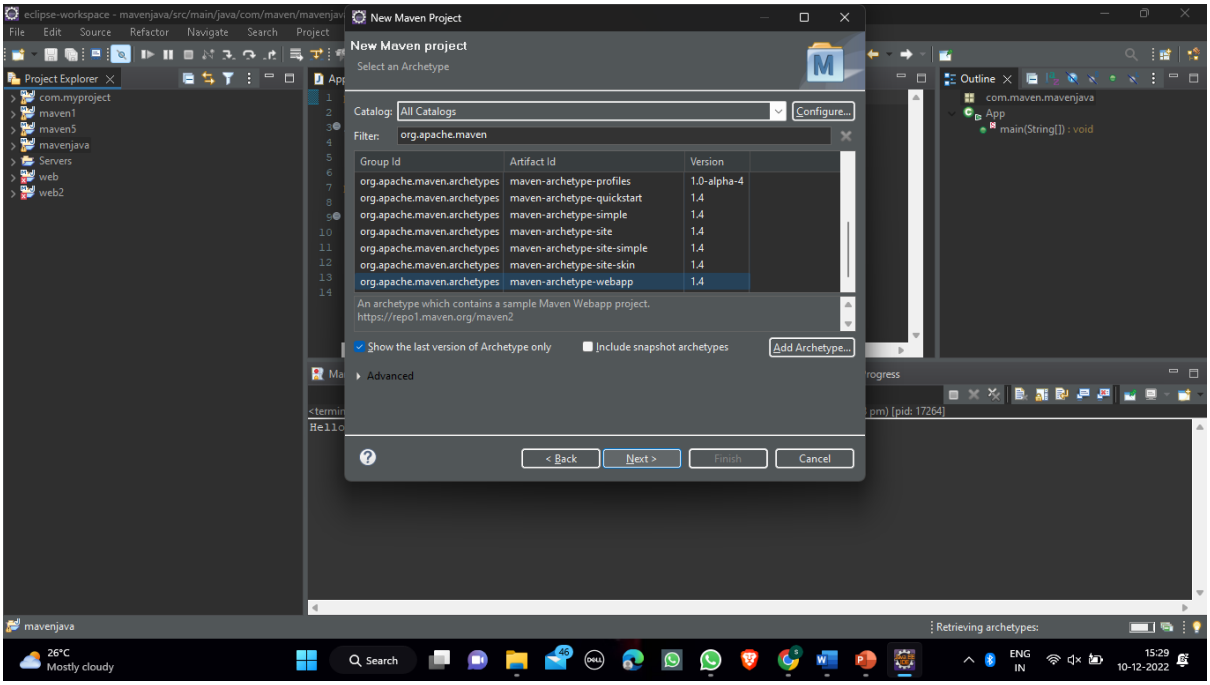
Step 1-Go to File->New->Maven Project.



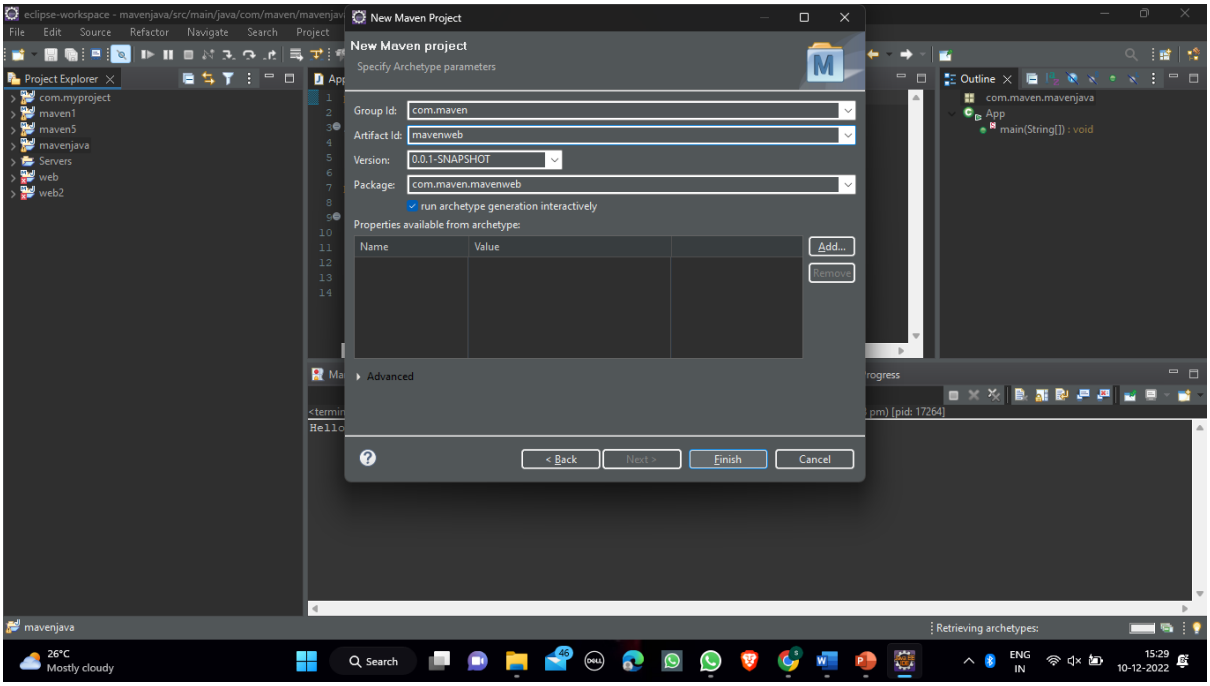
Step 2-Click next



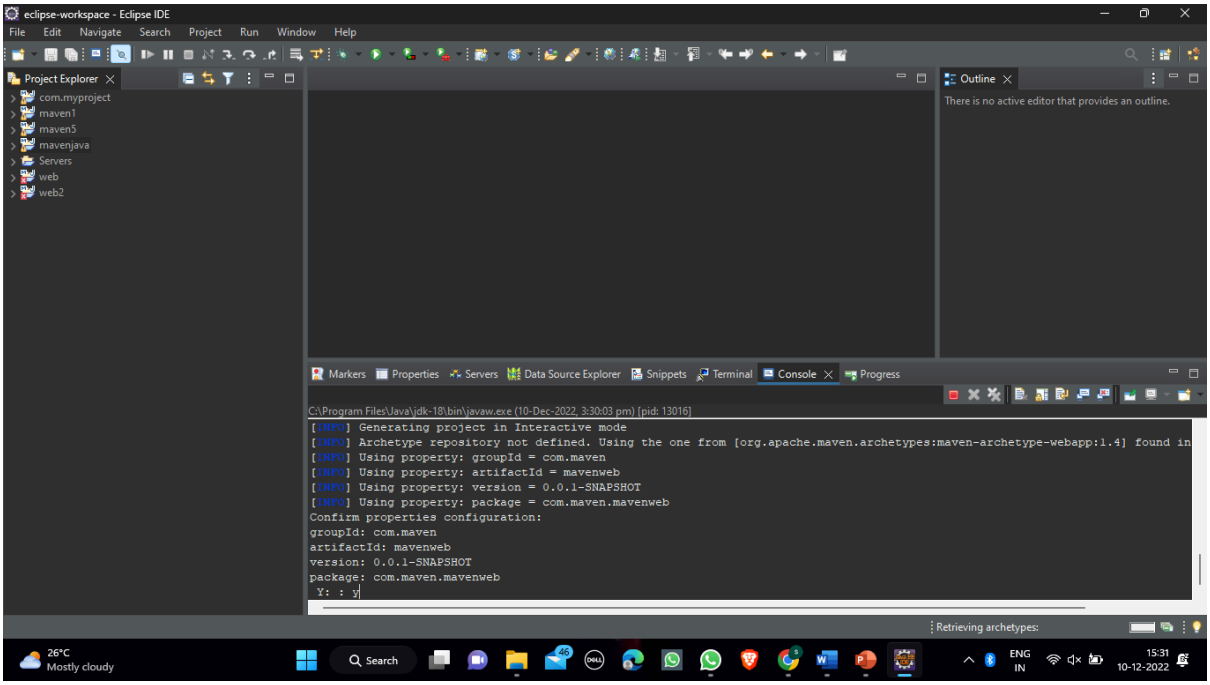
Step 3- Use org.apache.maven and click on webapp Aritfact Id and version 1.4 .



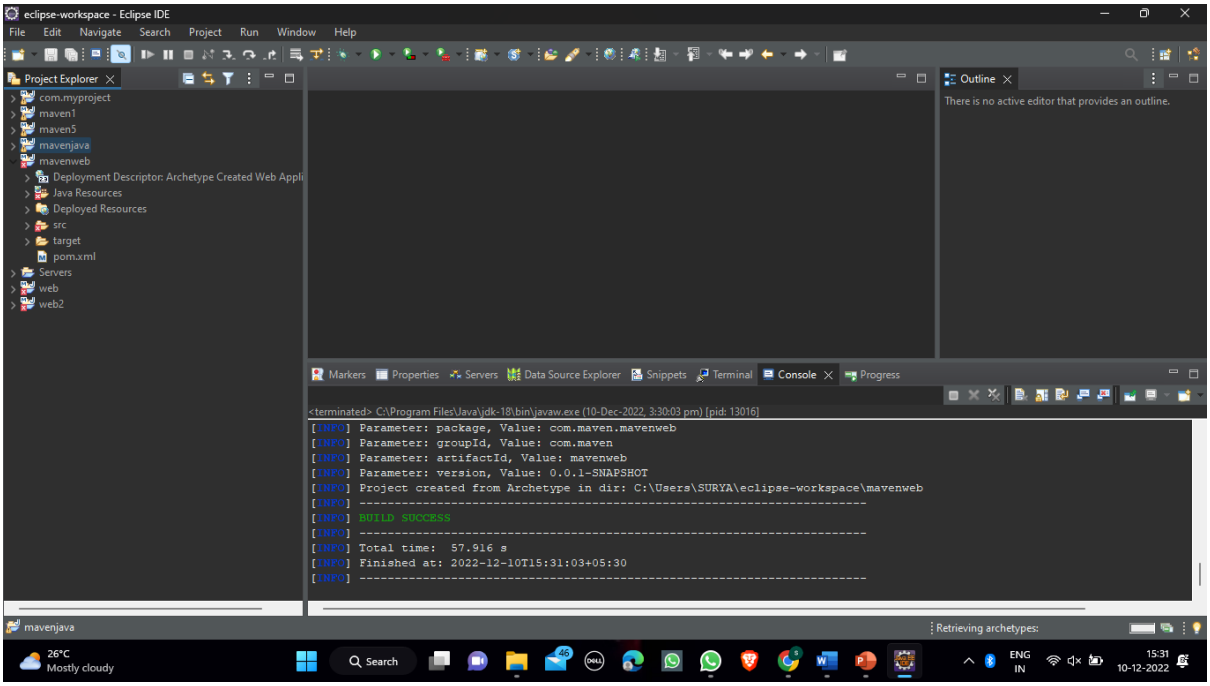
Step 4- Create the project name as mavenweb and click on finish.



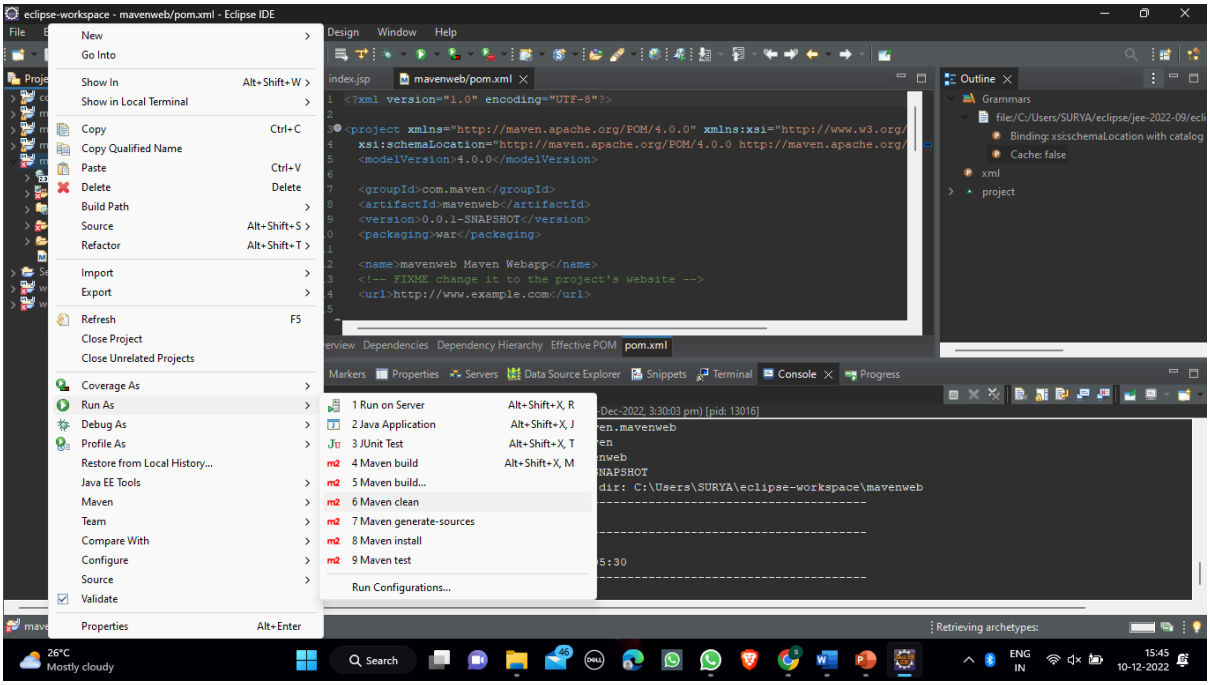
Step 5-Click y to create the mavenweb project.



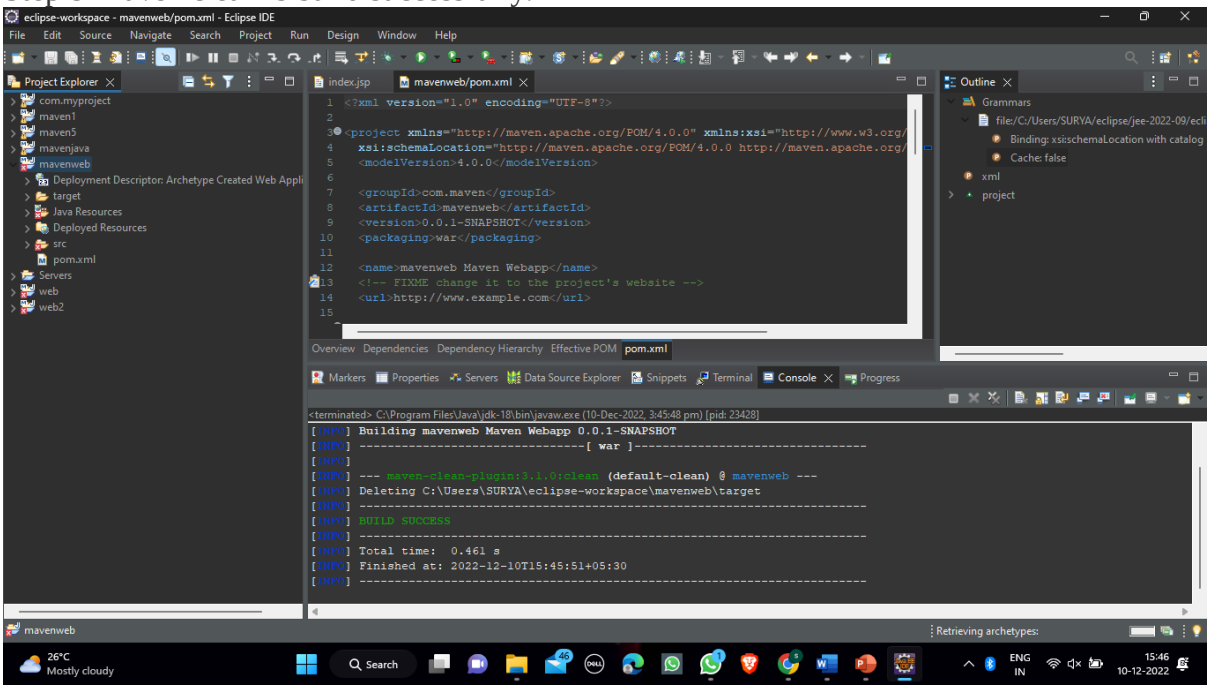
Step 6-Mavenweb project is created successfully.



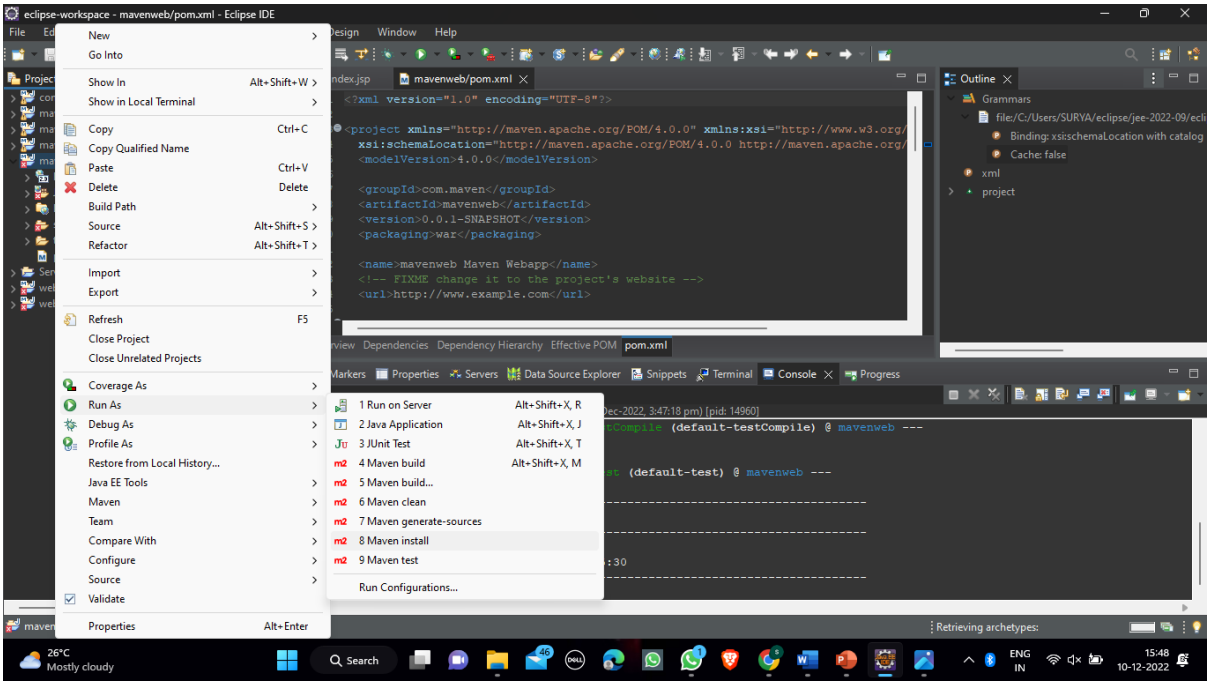
Step 7-Right Click on mavenweb and Run As and click on maven clean.



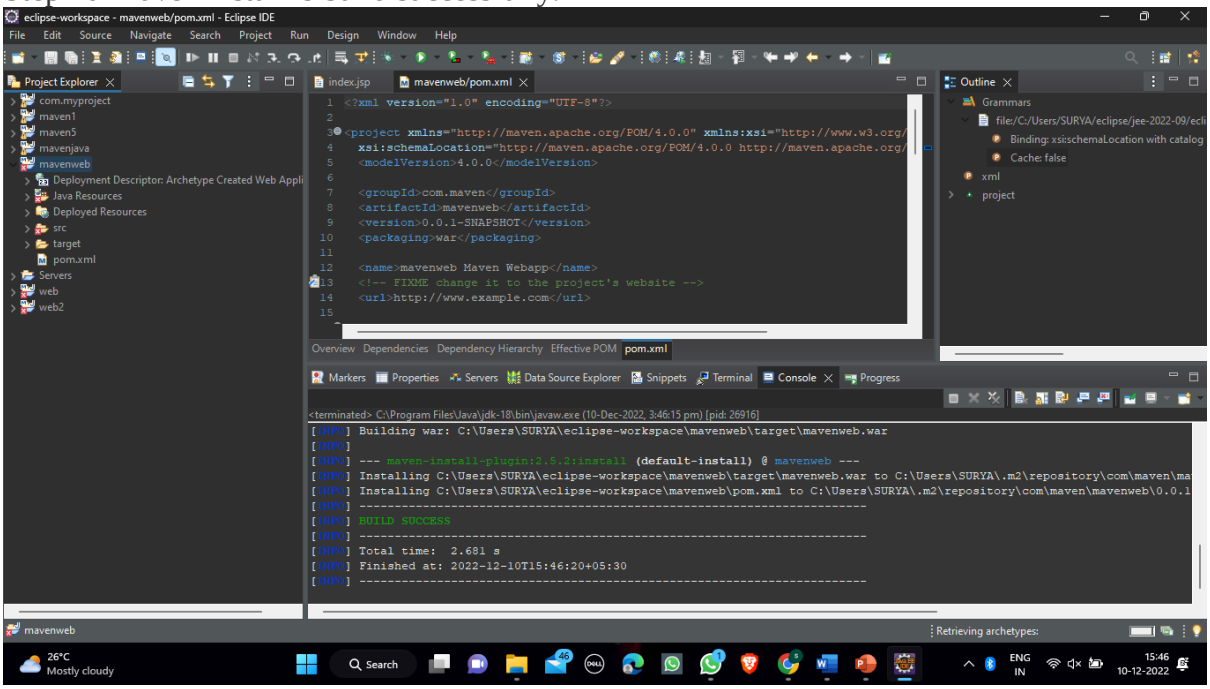
Step 8-Maven clean is build successfully.



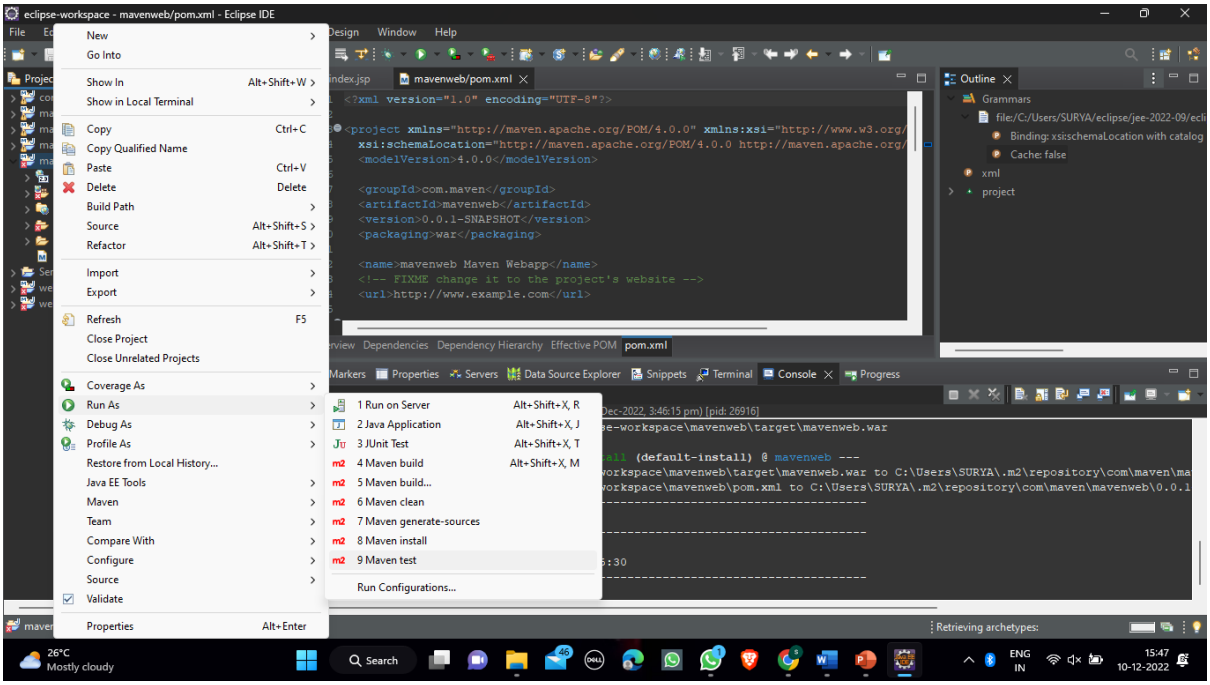
Step 9- Repeat step 7 but click on maven install.



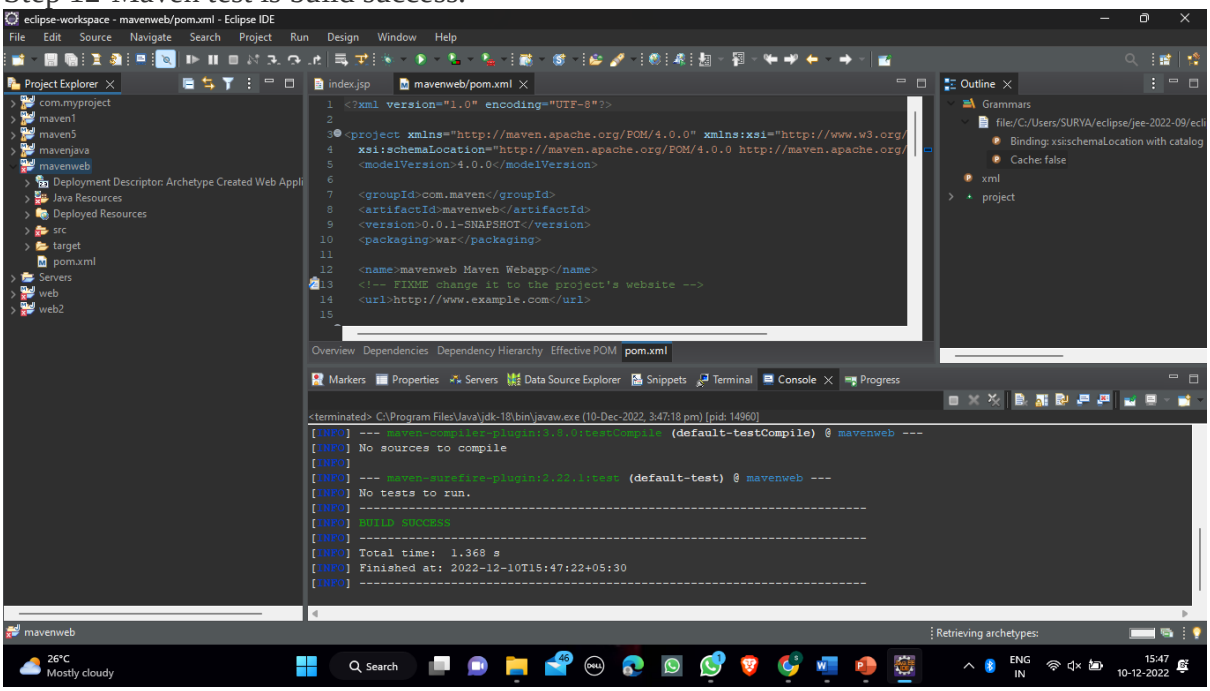
Step 10-Maven install is build successfully.



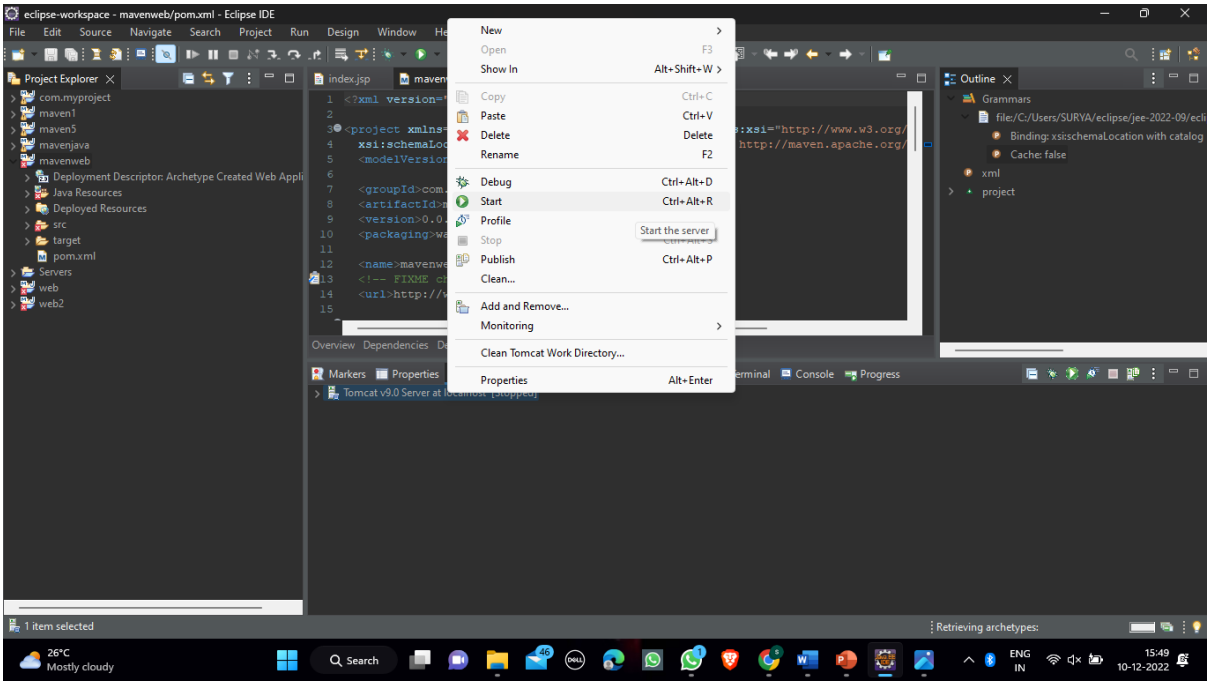
Step 11-Repeat step 7 but click on Maven test.



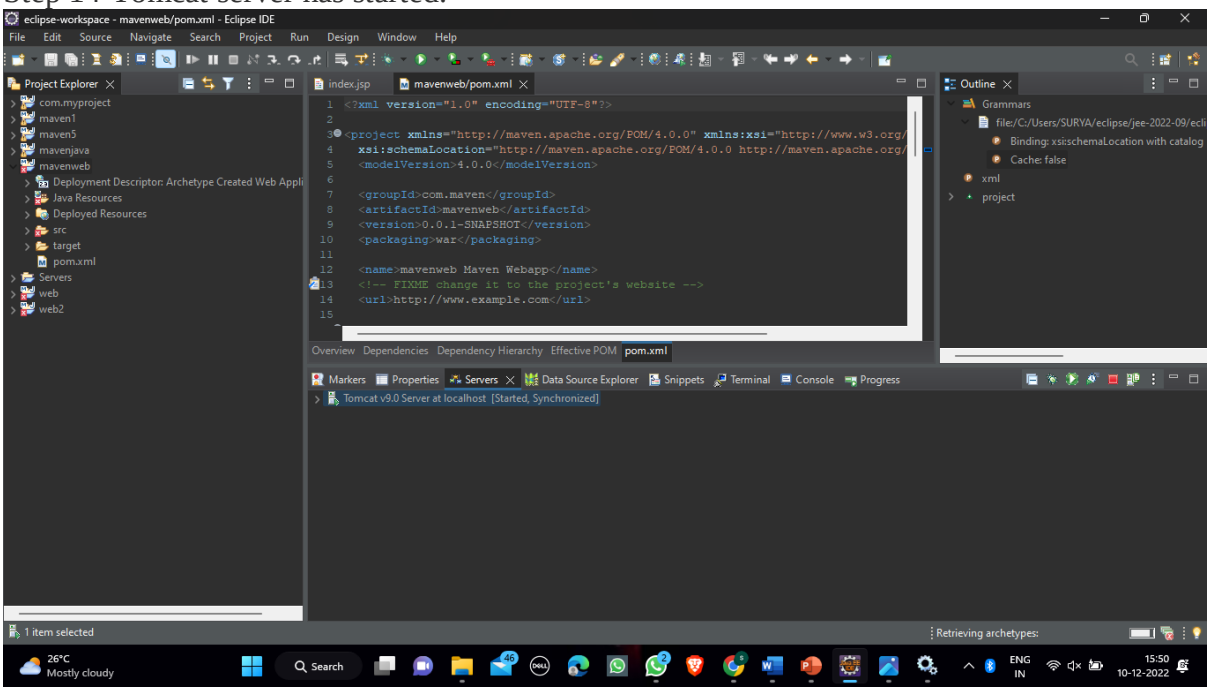
Step 12-Maven test is build success.



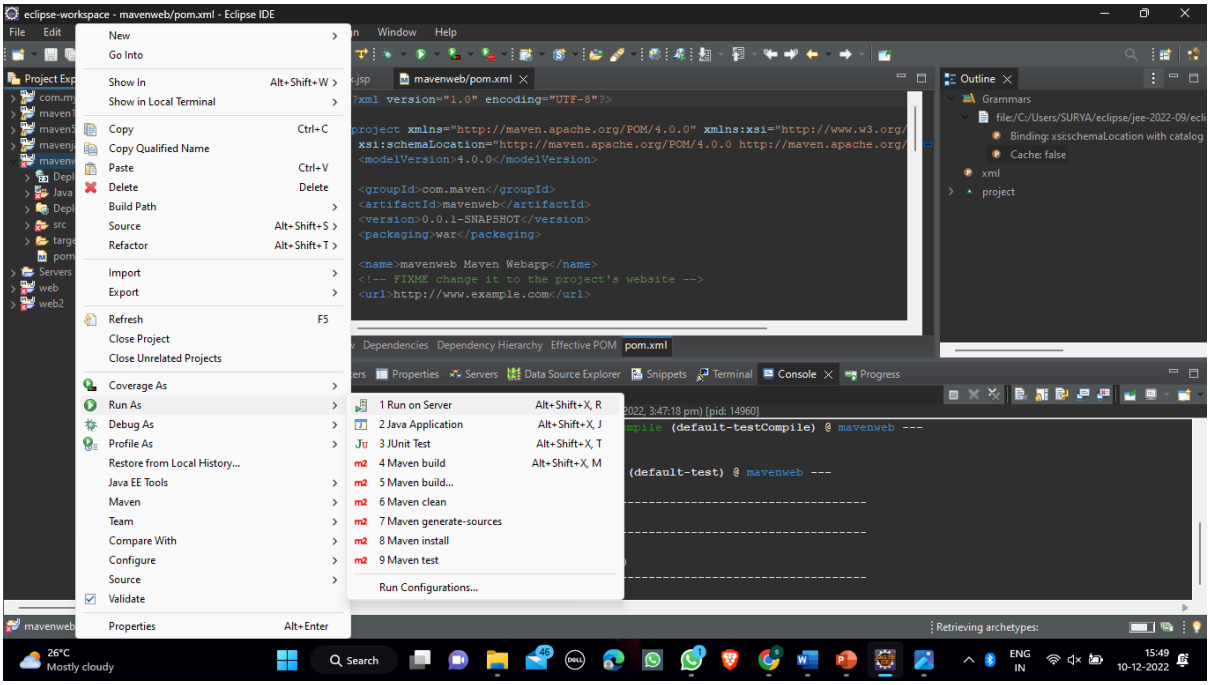
Step 13-Start the tomcat server in server.



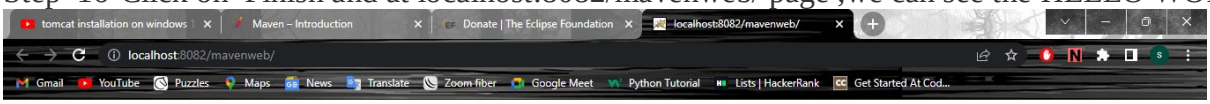
Step 14-Tomcat server has started.



Step 15-Repeat Step 7 but click on run on server.

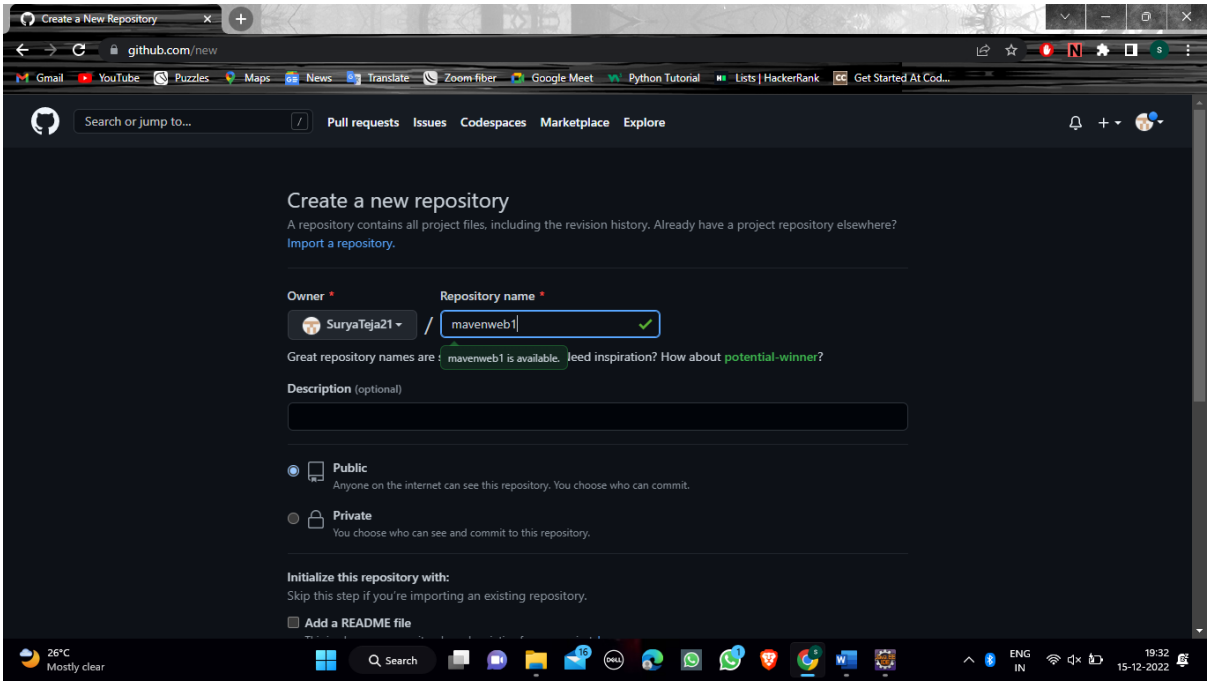


Step 16-Click on Finish and at localhost:8082/mavenweb/ page ,we can see the HELLO WORLD! .

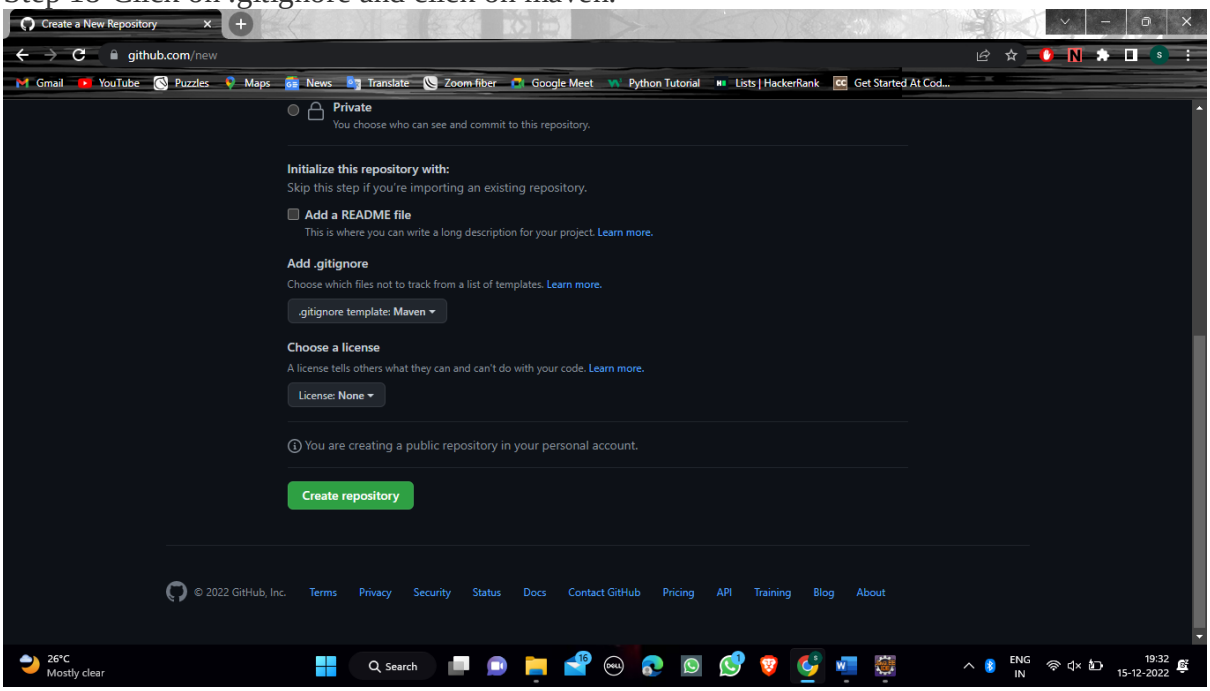


Hello World!

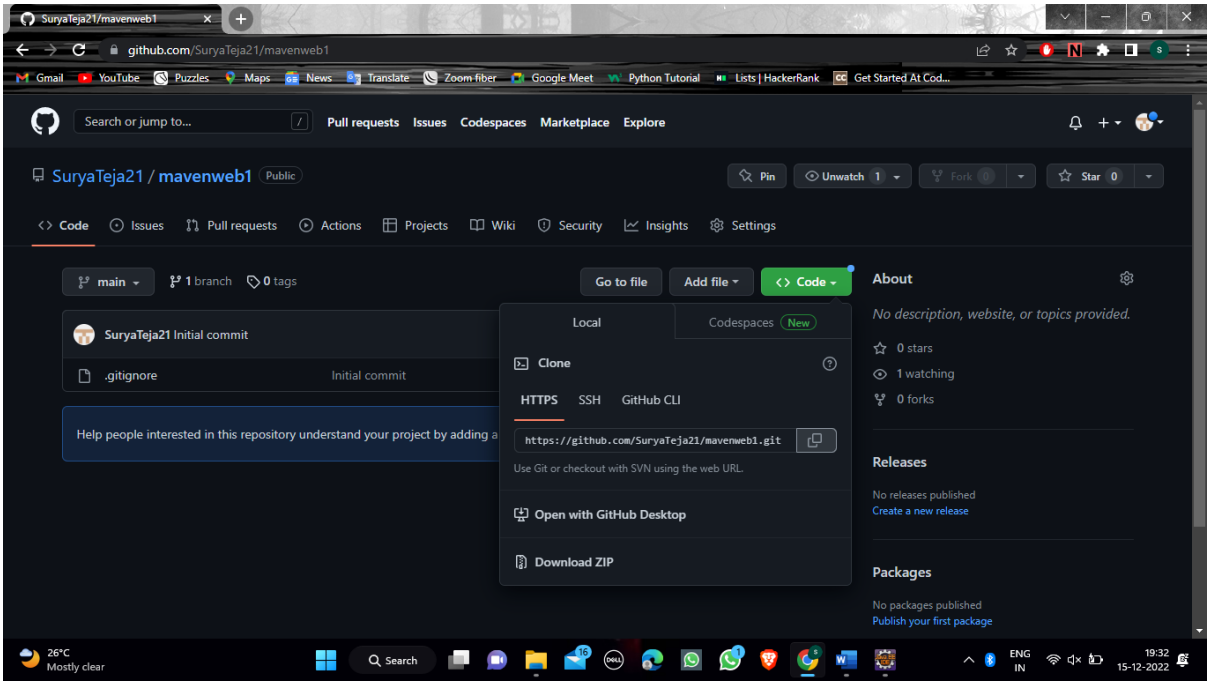
Step 17-Create a github repository with name mavenweb1.



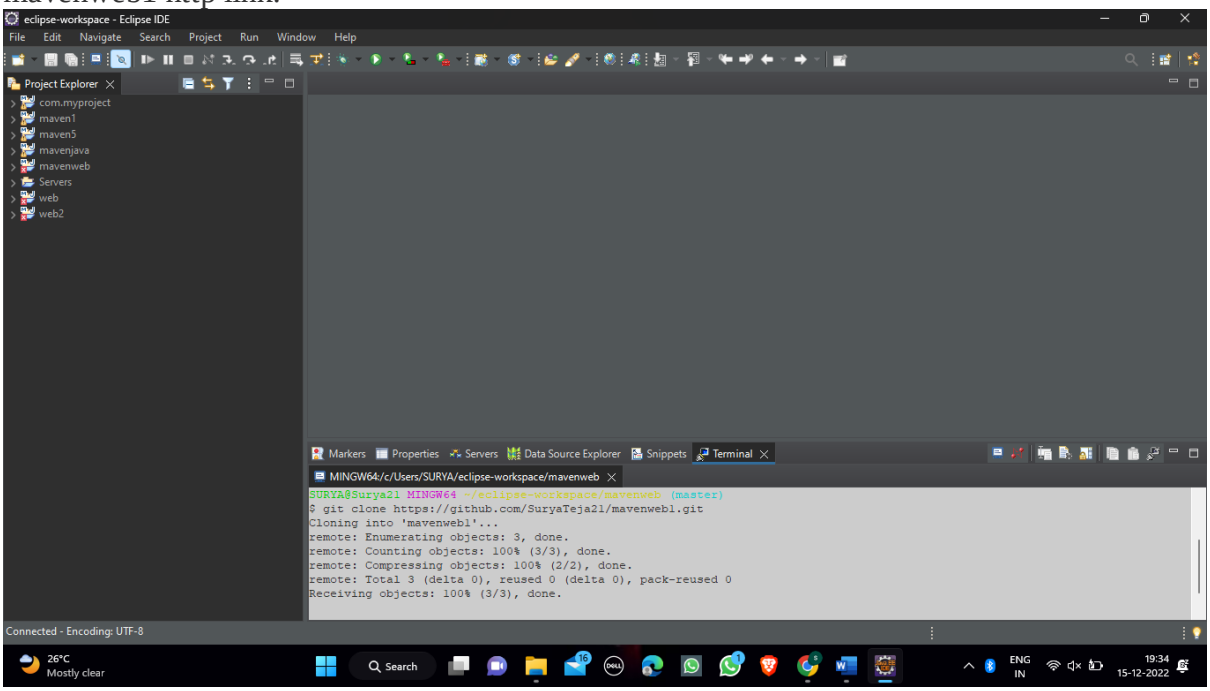
Step 18-Click on .gitignore and click on maven.



Step 19-Mavenweb1 is created and Click on code and copy the http link as shown.

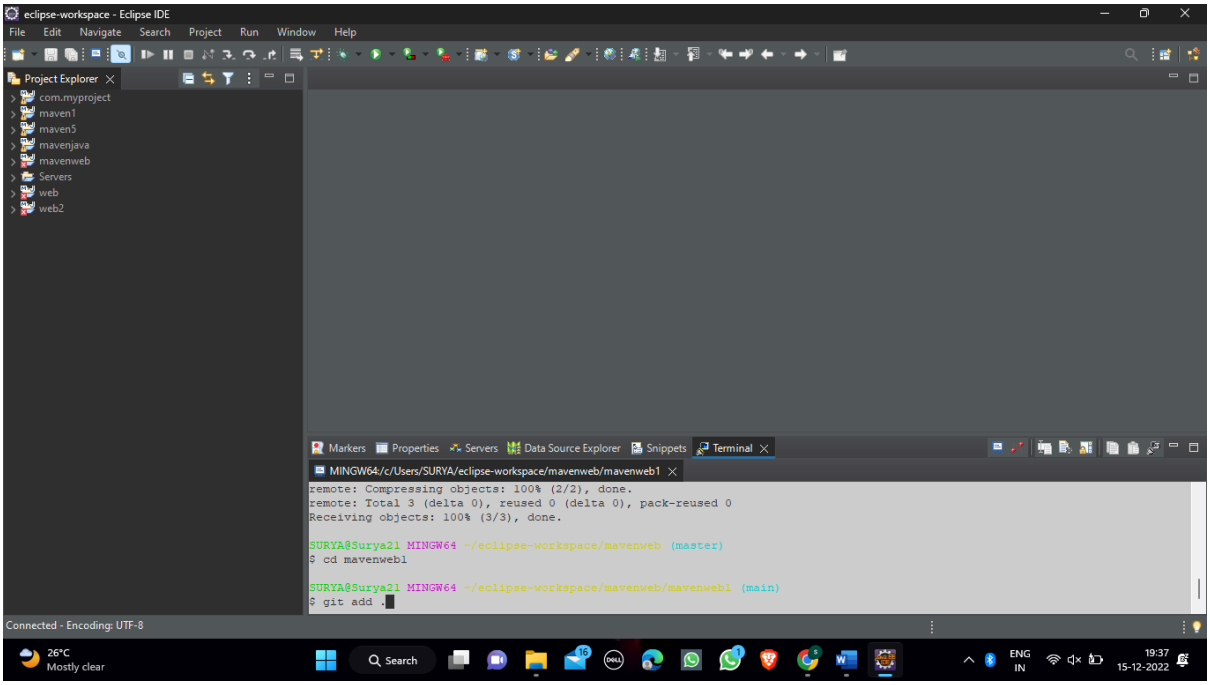


Step 20-In eclipse, right click on mavenweb and select Git Bash from show terminal. Use git clone and copy the mavenweb1 http link.

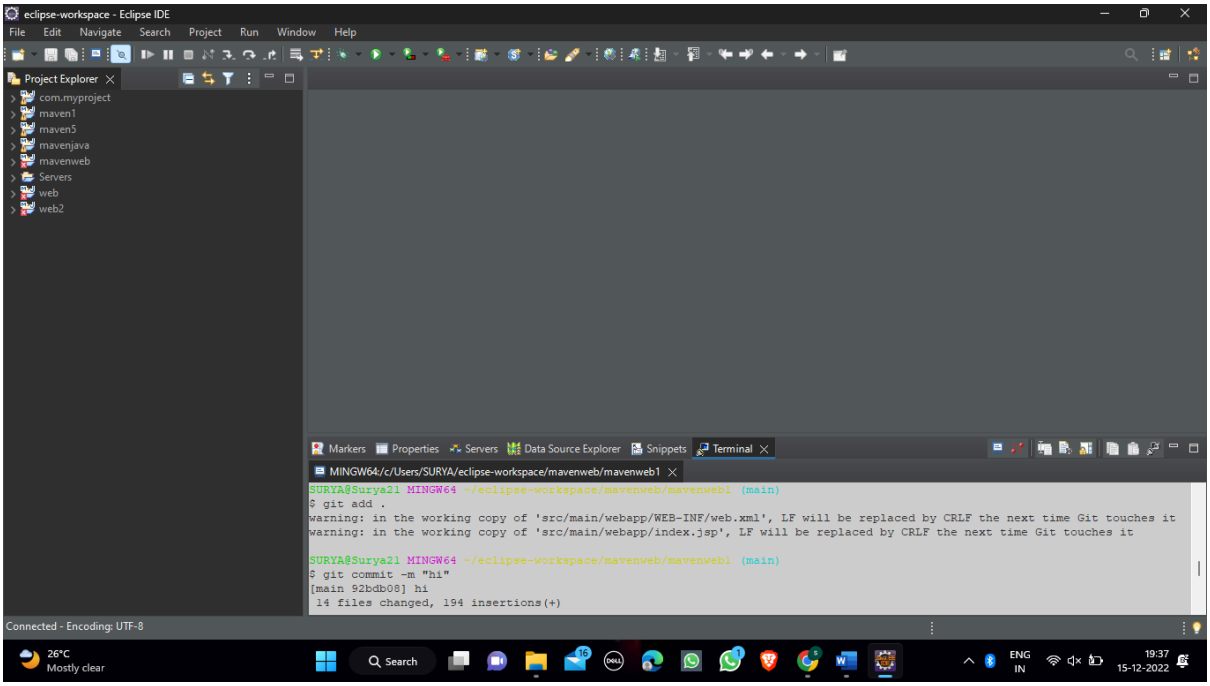


Step 21-Use cd mavenweb1 and paste all the files from mavenweb.

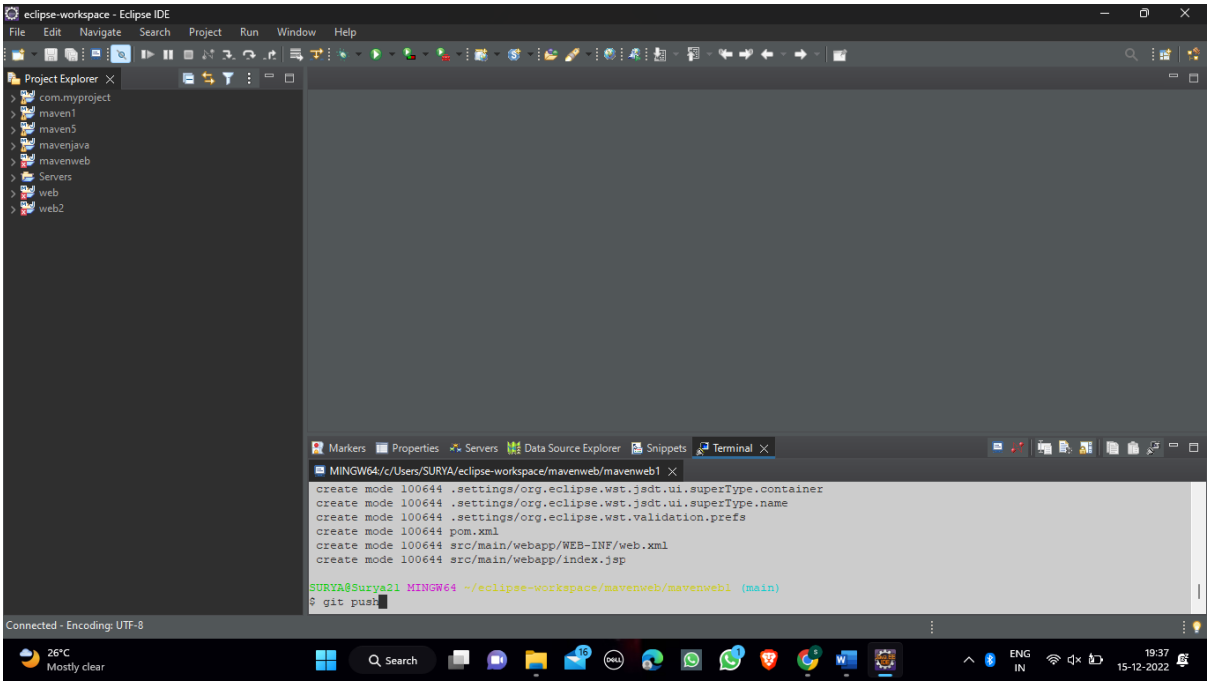
After use git add . command.



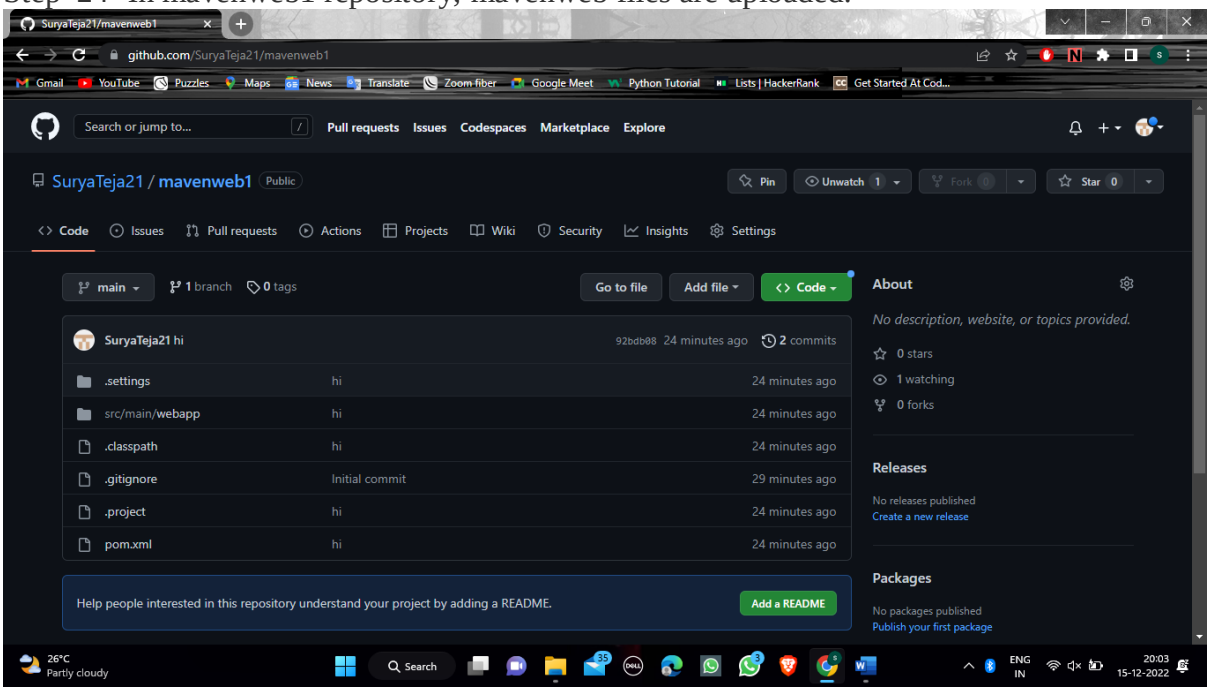
Step 22-Use git commit -m



Step 23-Use git push command to push the files to mavenweb1 repository.



Step 24- In mavenweb1 repository, mavenweb files are uploaded.



Apache-Tomcat:-

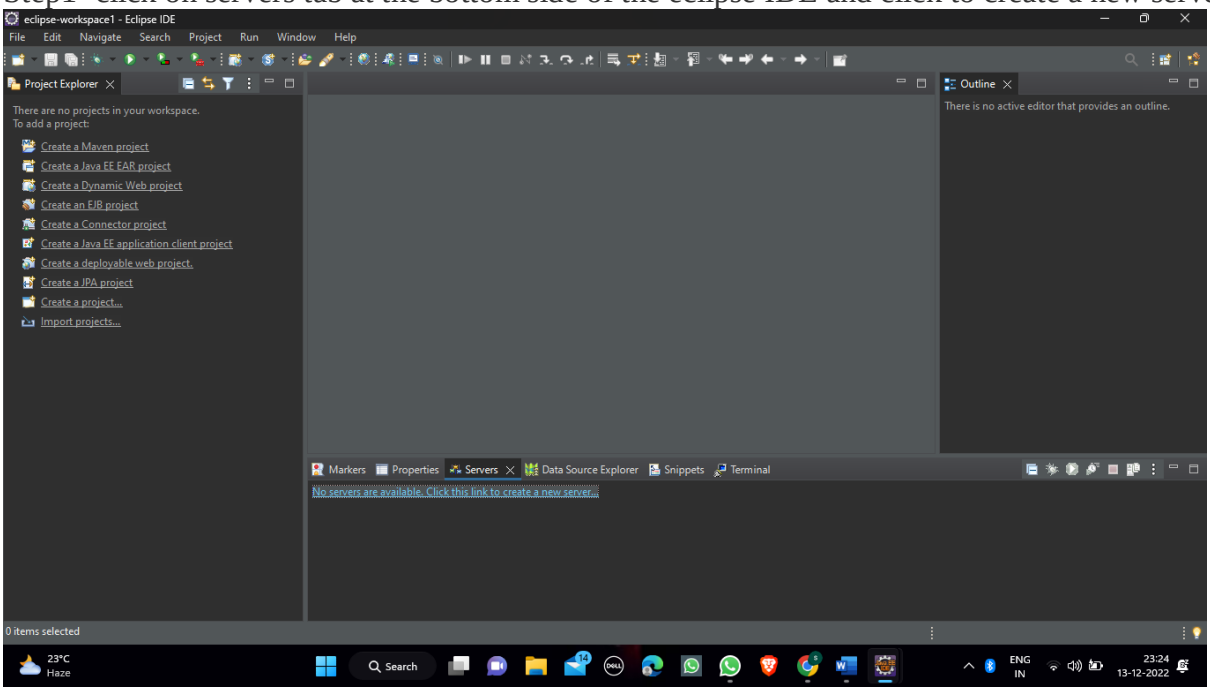
Tomcat is a web-container which allows the users to run java server pages that are based on web applications.

It is used as the HTTP server.

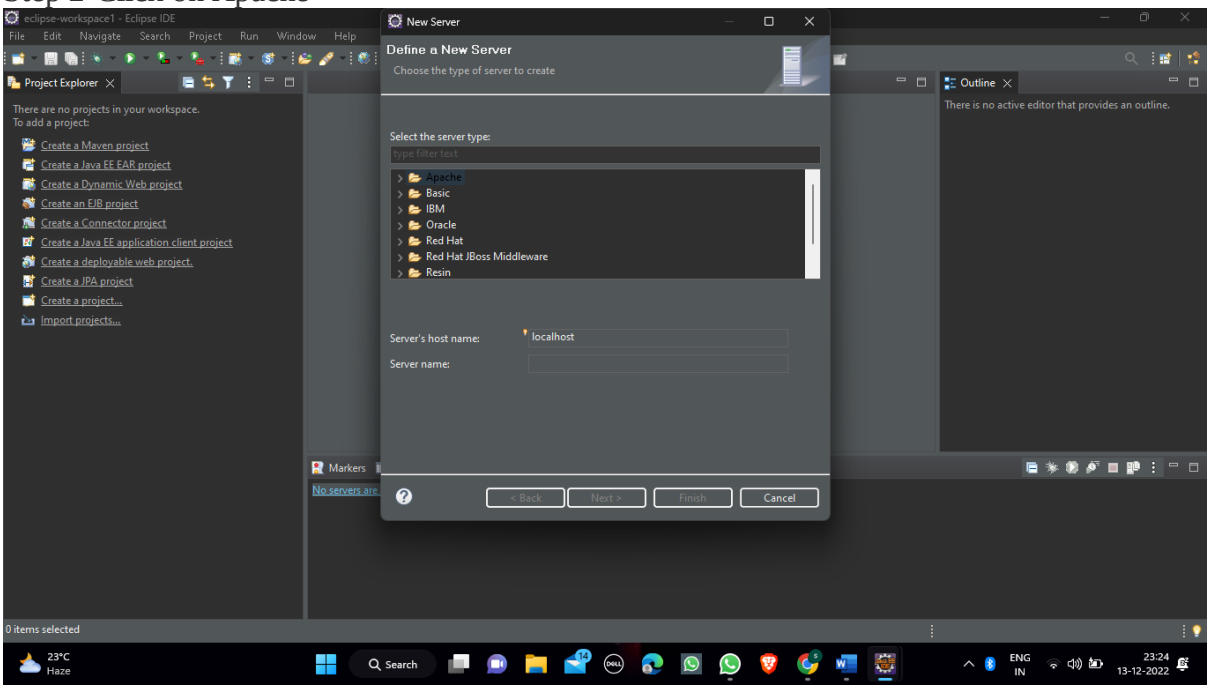
Tomcat Configuration in eclipse-

For Configuration of Tomcat in eclipse do the following steps-

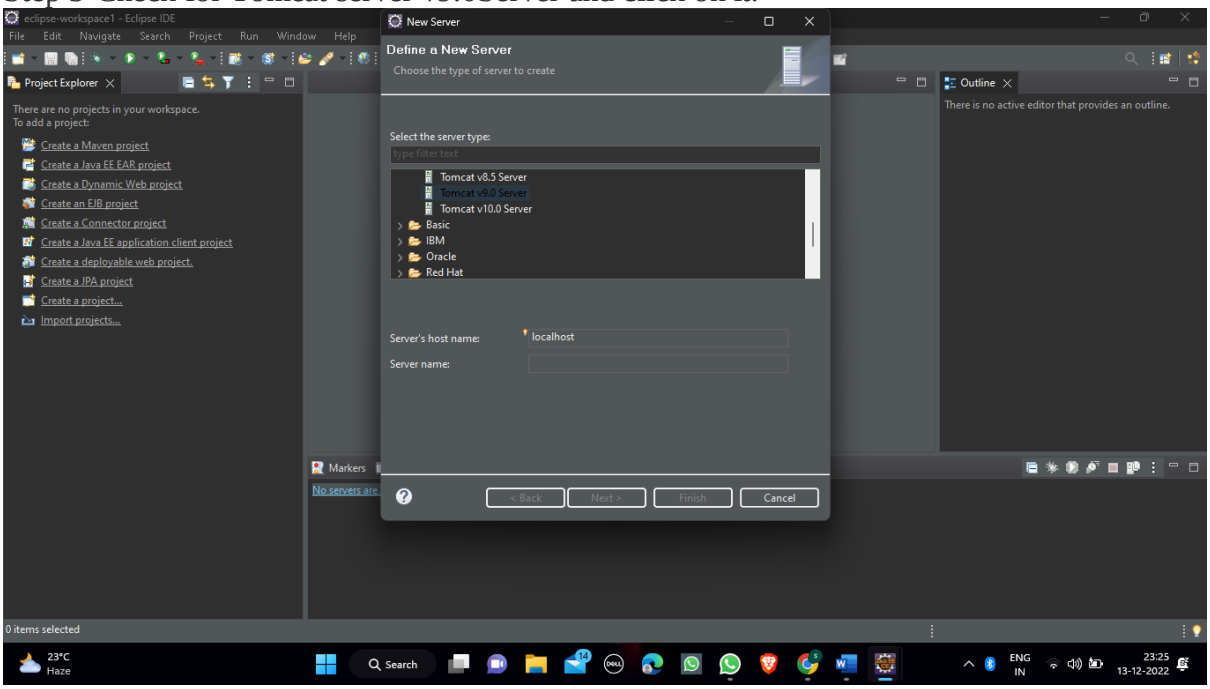
Step1- click on servers tab at the bottom side of the eclipse IDE and click to create a new server.



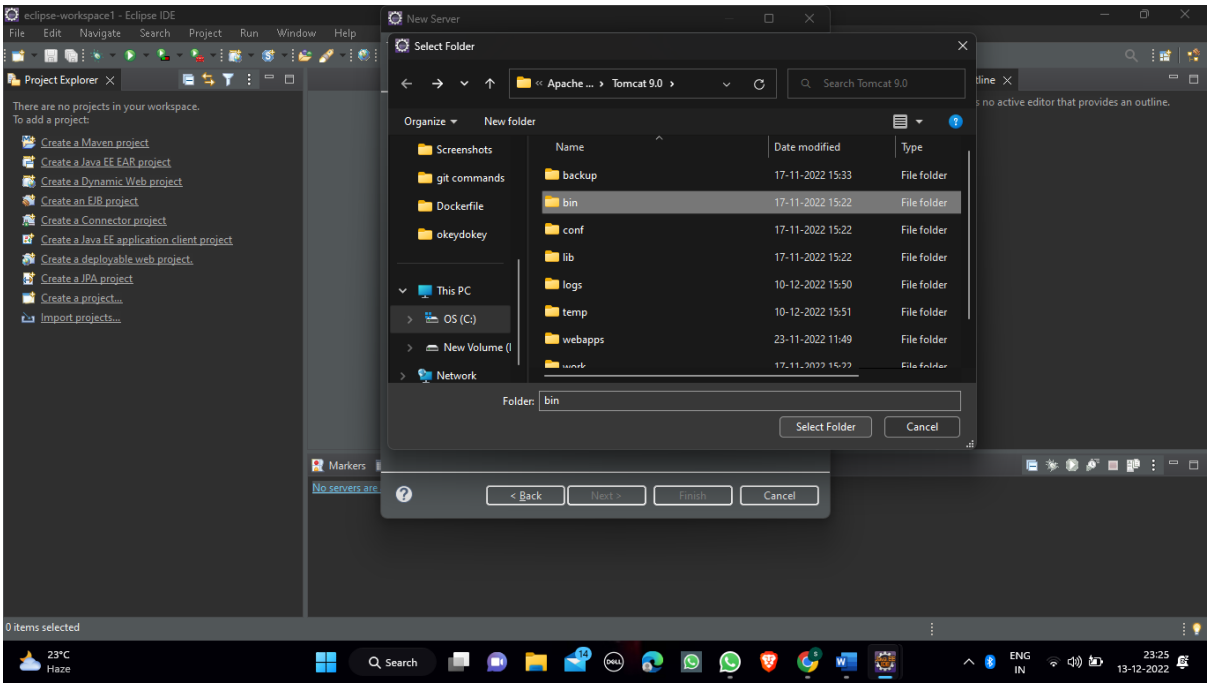
Step 2-Click on Apache



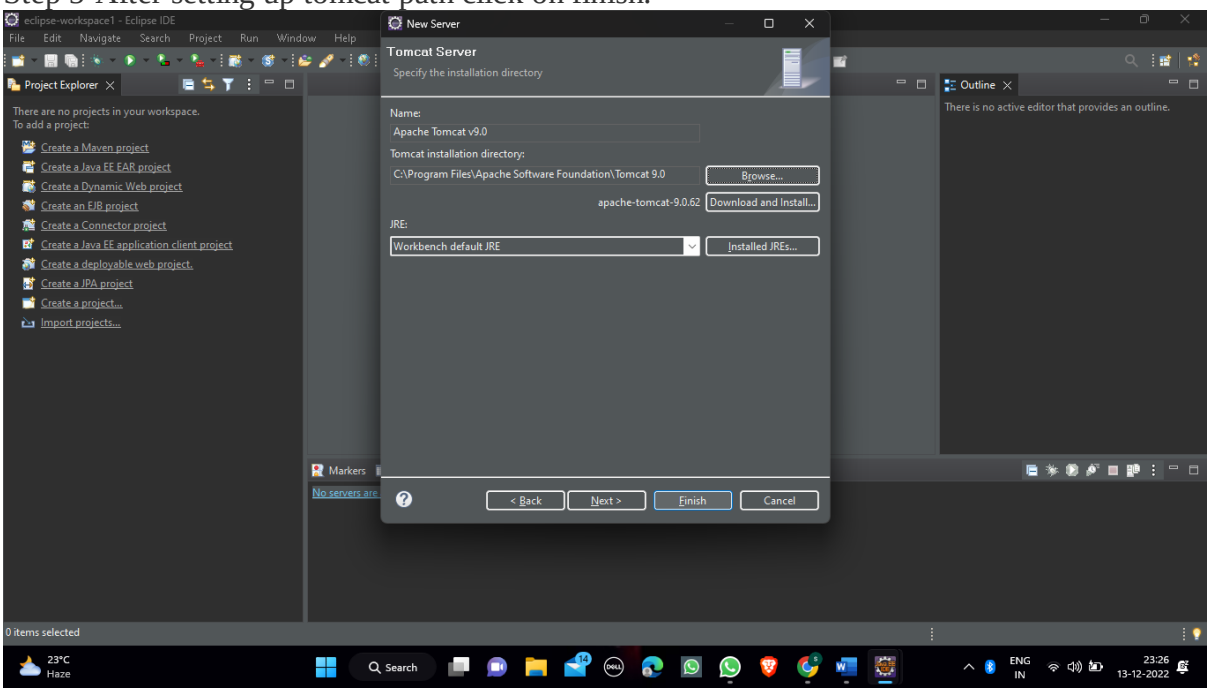
Step 3-Check for Tomcat server v9.0Server and click on it.



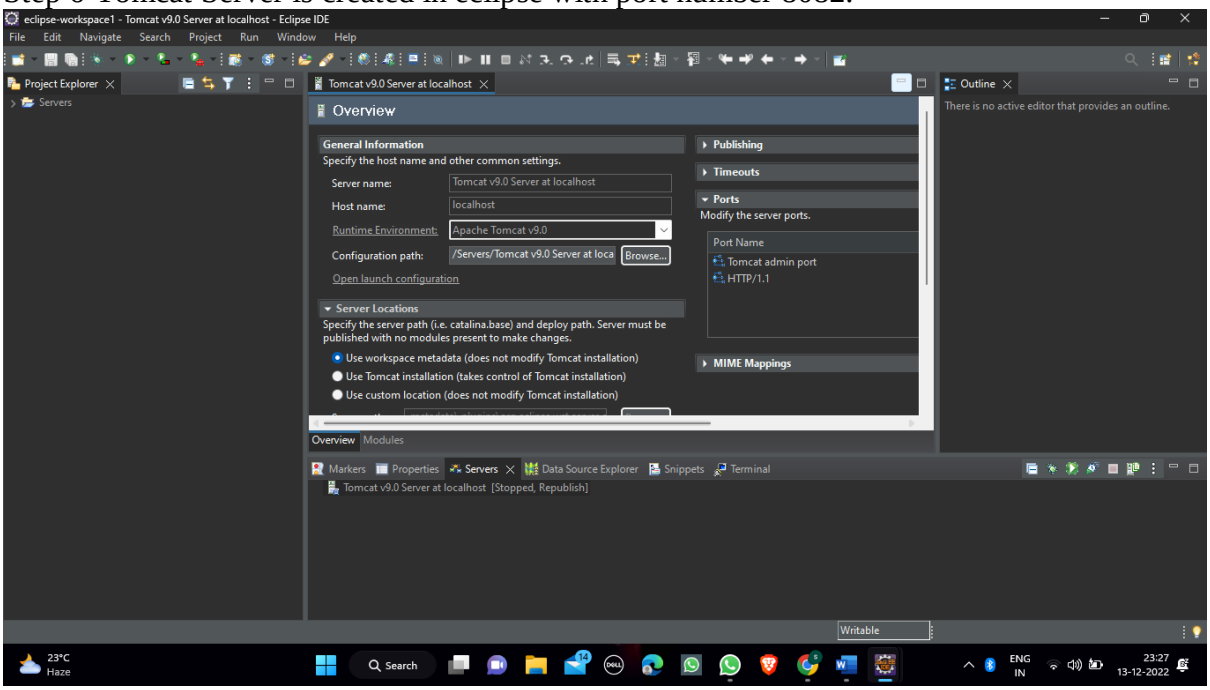
Step4- Select C:\Program Files\Apache Software Foundation\Tomcat 9.0 and click on select folder.



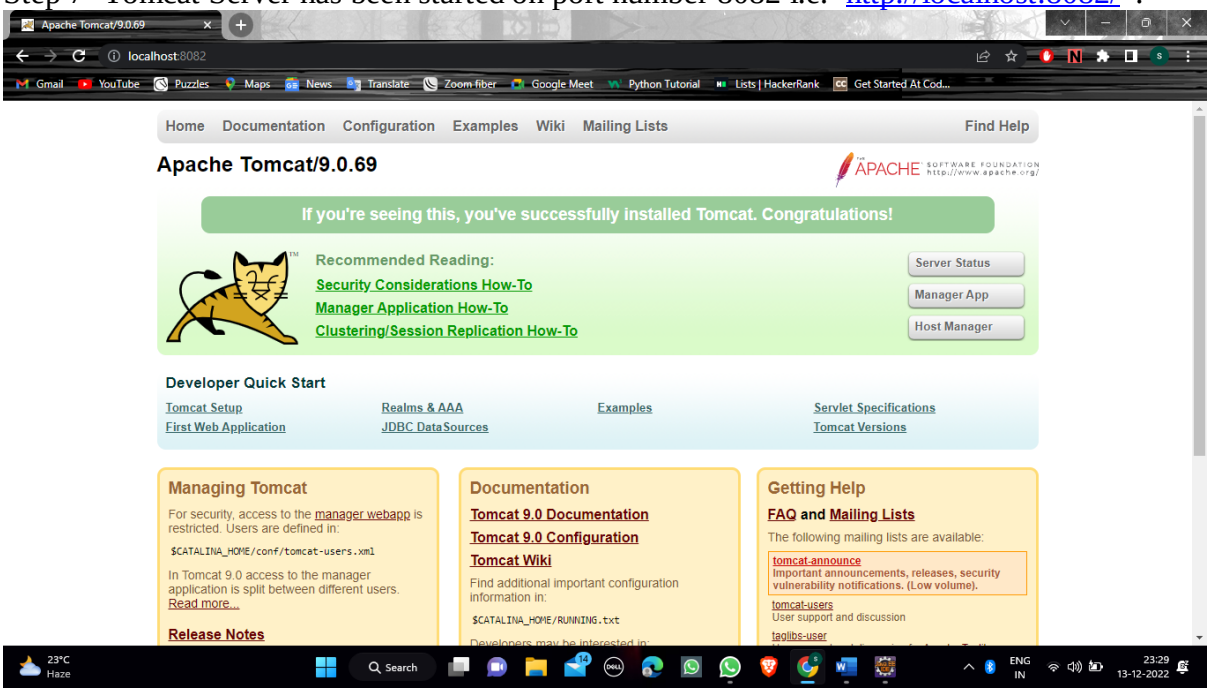
Step 5-After setting up tomcat path click on finish.



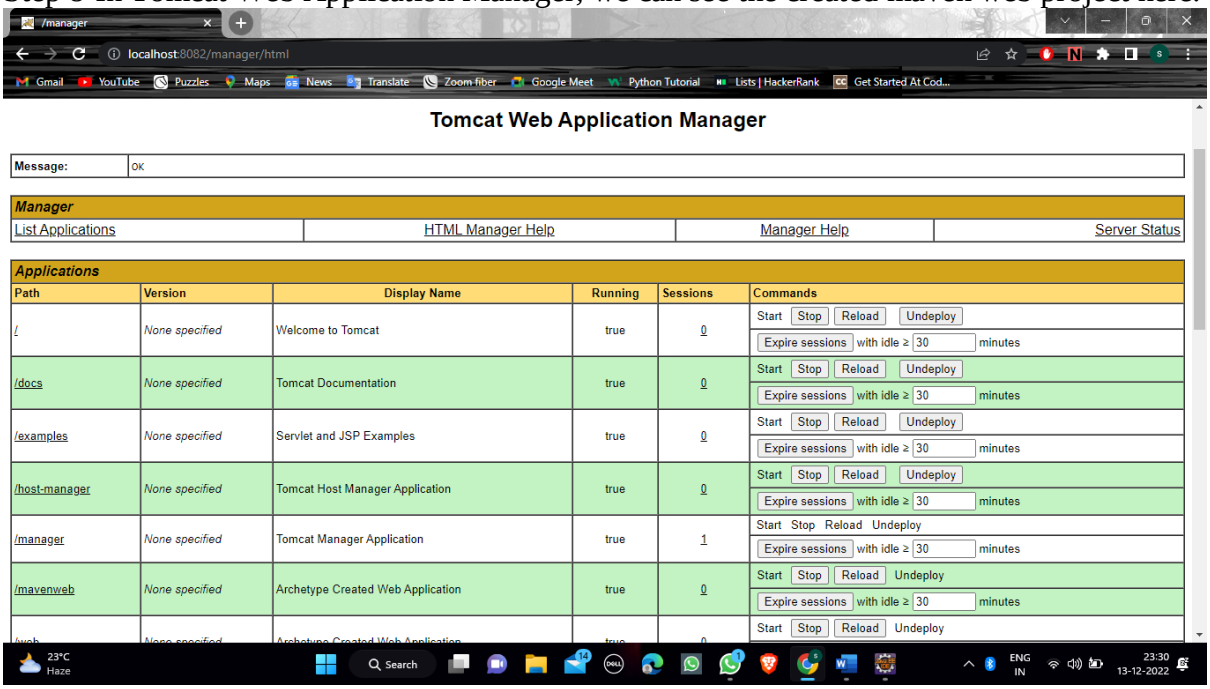
Step 6-Tomcat Server is created in eclipse with port number 8082.



Step 7- Tomcat Server has been started on port number 8082 i.e. http://localhost:8082/ .



Step 8-In Tomcat Web Application Manager, we can see the created maven web project here.



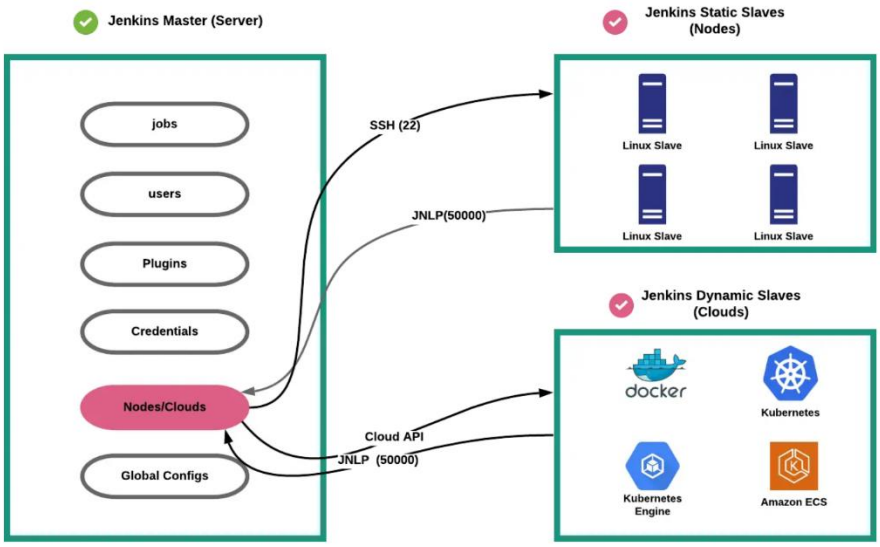
4.WORKING WITH JENKINS

What is Jenkins?
Jenkins is a tool that is used for automation, and it is an open-source java-based CI/CD tool that allows all the developers to build, test and deploy software.

Jenkins is commonly used for the following.

1. Continuous Integration for application and infrastructure code.
2. Continuously deliver pipeline to deploy the application to different environments using Jenkins pipeline as code
3. Infrastructure component deployment and management.
4. Run batch operations using Jenkins jobs.
5. Run ad-hoc operations like backups, cleanups, remote script execution, event triggers, etc.

The following diagram shows the overall architecture of Jenkins.



The pros and cons of Jenkins

Pros:

1. Jenkins is open source and free

2. Jenkins comes with a wide range of plugins

3. Jenkins integrates and works with all major tools

4. Jenkins is flexible

5. Jenkins comes with a decent API suite

6. Jenkins is easy to use

7. You have a ready talent base

Cons:

1. Unpredictable costs

2. Lack of governance

3. No collaboration features

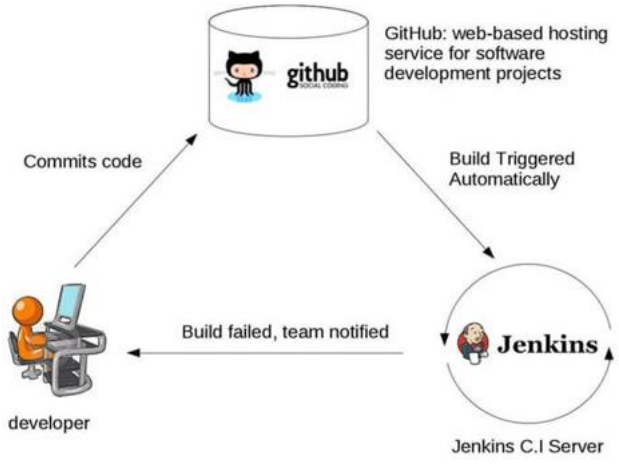
4. Lack of analytics

5. Needs personnel

CI/CD

What is CI?

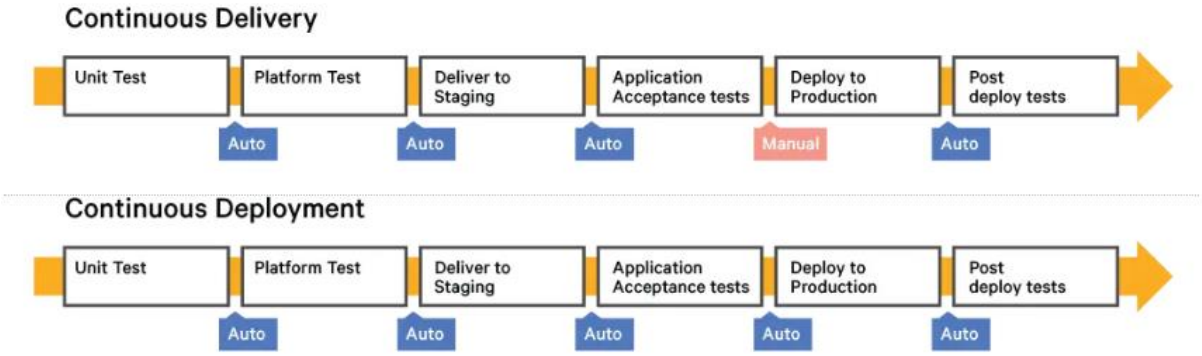
Continuous Integration is an automated process. It is done in order to integrate various codes from different potential sources in order to build it or test it. The codes are often required to be integrated into the form of a shared repository by the developers.



What is CD?

A *continuous delivery (CD) pipeline* is an automated expression of your process for getting software from version control right through to your users and customers. Every change to your software (committed in source control) goes through a complex process on its way to being released. This process involves building the software in a reliable and repeatable manner, as well as progressing the built software (called a "build") through multiple stages of testing and deployment.

The major difference between the CI and the CD is that the former assumes that there are several actions to be taken post-deployment whereas the CD pipeline always makes the products and artifacts ready for the production. Simply put every run of the Continuous Delivery (CD) pipeline awarded with a green signal can be deployed to production without any worries.



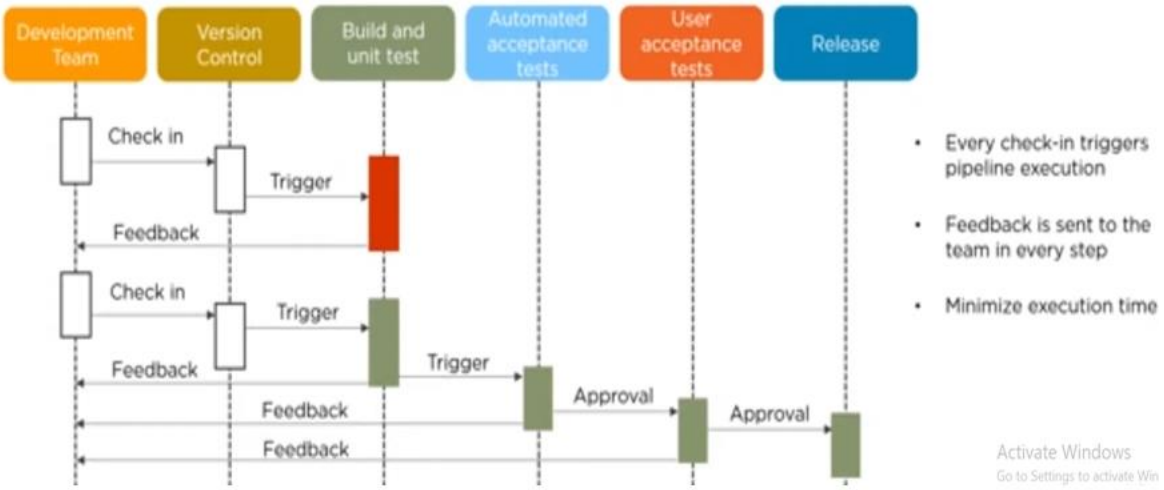
Pipelines:

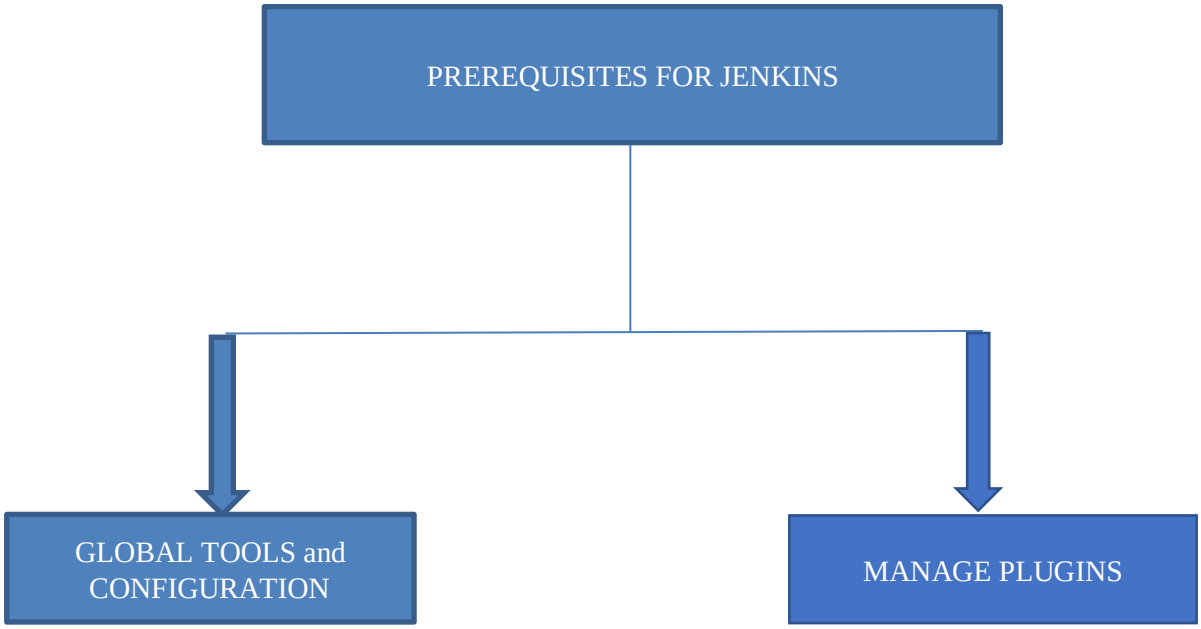
Jenkins Pipeline (or simply "Pipeline" with a capital "P") is a suite of plugins which supports implementing and integrating *continuous delivery pipelines* into Jenkins.

Pipeline adds a powerful set of automation tools onto Jenkins, supporting use cases that span from simple continuous integration to comprehensive CD pipelines. By modeling a series of related tasks, users can take advantage of the many features of Pipeline:

- **Code**: Pipelines are implemented in code and typically checked into source control, giving teams the ability to edit, review, and iterate upon their delivery pipeline.
- **Durable**: Pipelines can survive both planned and unplanned restarts of the Jenkins controller.
- **Pausable**: Pipelines can optionally stop and wait for human input or approval before continuing the Pipeline run.

- **Versatile**: Pipelines support complex real-world CD requirements, including the ability to fork/join, loop, and perform work in parallel.
- **Extensible**: The Pipeline plugin supports custom extensions to its DSL [1] and multiple options for integration with other plugins.





GLOBALTOOLS:

1.Download Java 17 version and set the path as JAVA_HOME in Environment variables.

2.Download Maven and set the path as MAVEN_HOME in the Environment variables.

MANAGE PLUGINS :

We need to download 5 plugins from the Jenkins. They are as follows:

1. Maven integration.
2. Build pipeline.
3. Pipeline utility.
4. Copy artifacts.
5. Deploy to container.

Lets start building 2 pipeline:

After creating a new maven project on eclipse and pushing it to your repository in git hub follow the following steps.

Step 1: Open Jenkins in the local host , login into your Jenkins account and create a new item.

Step 2: Create a new Freestyle project and click ok.

Step 3: Give the Git repository URL of the project to be built and specify the branch as either master/main as it is in the GitHub.

Step 4: Now in Build, Invoke top-level Maven targets

Step 5: Select the Maven path which is already set in the global credentials in Manage Jenkins.Follow the same goals as done in eclipse starting with clean and install.

Step 6: Now in post build actions-> select Archive the artifacts, to send the output of build project to the testing team.

Step 7: If we want to archive all the artifacts type **/* as shown.

Step 8: Now the next step is to build other projects, where we will create a test project which will be triggered by the build project.

Step 9: Create a new freestyle project test as shown and click ok.

Step 10: This time we need not mention the git repository so select None.

Step 11: In Build environment check the box as shown below, this is to discard old builds.

Step 12: To forward the artifacts of the previous project to the current test project, select copy the artifacts from another project in Build as shown.

Step 13: Give the name of the project from which we want to copy the artifacts and check the box ->stable build only->to copy all the artifacts type **/*.

Step 14: Now select Invoke top-level Maven targets in build.

Step 15: This time give the goal as test after selecting the Maven version.

Step 16: In post-build actions->select Archive the artifacts.

Step 17: To save all the artifacts->type **/* and Apply->Save.

Step 18: Create a pipeline by clicking on + symbol in the dashboard ->a pipeline is a collection of events or jobs which are interlinked with one another in a sequence.

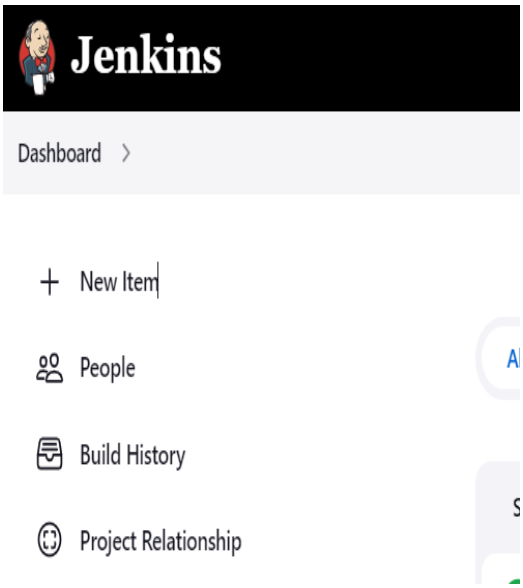
Step 19:Give a name to the pipeline->select Build Pipeline View->create.

Step 20: Select the first project to trigger the execution->build project.Apply->ok.

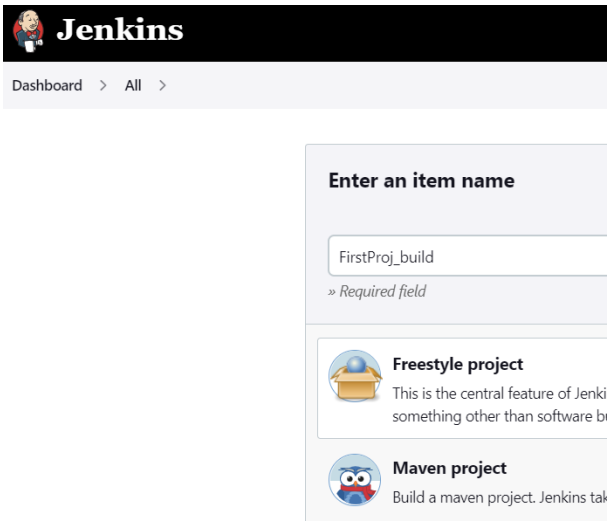
Step 21: Click on Run -> click on the small black box to open the console to check if the build is success.

Step 22: The pipeline is successful if it is in green color as shown ->check the console of the test project.

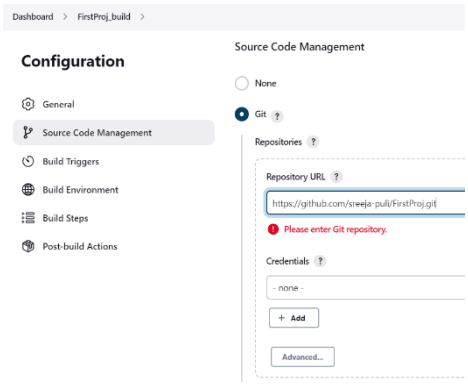
1.click on new item



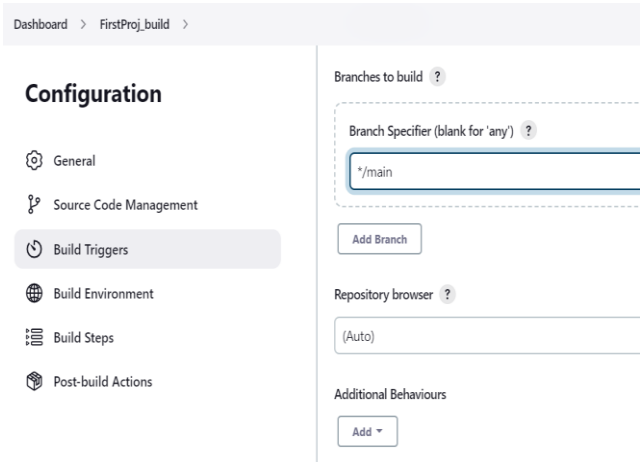
2. Enter an item name-> “ProjectName_build”->freestyle project -> Ok



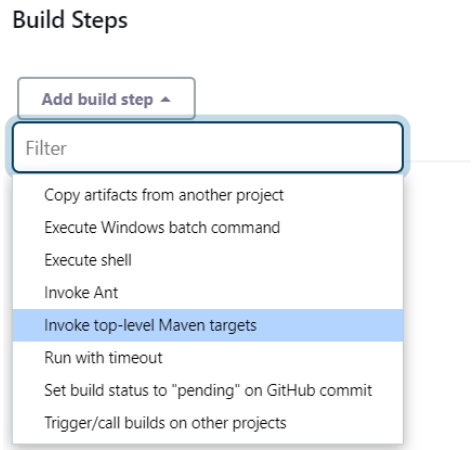
3. Select Git->paste reprository url from git hub.



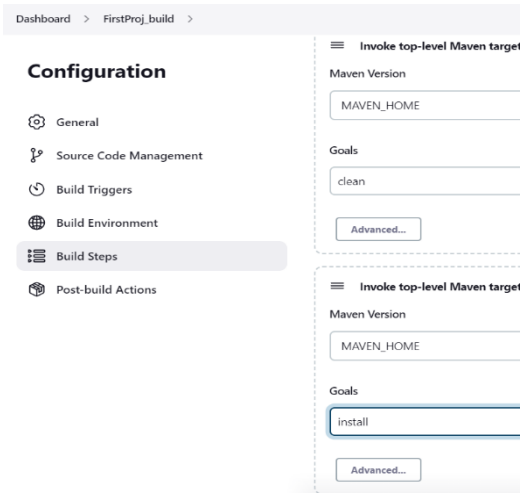
4.Specify the branch as master/main



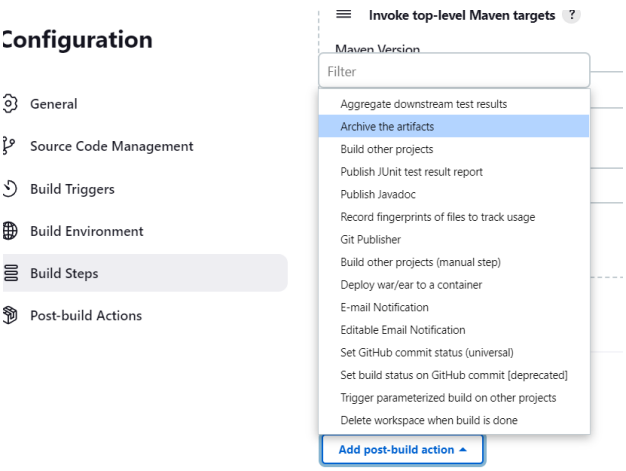
5.Build steps->invoke top-level management.



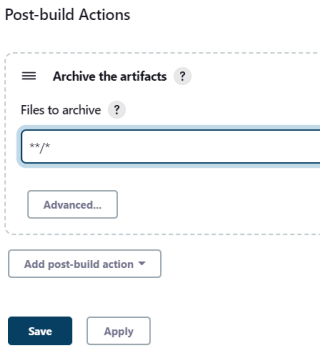
6.Maven version->MAVEN_HOME->Goals->clean.



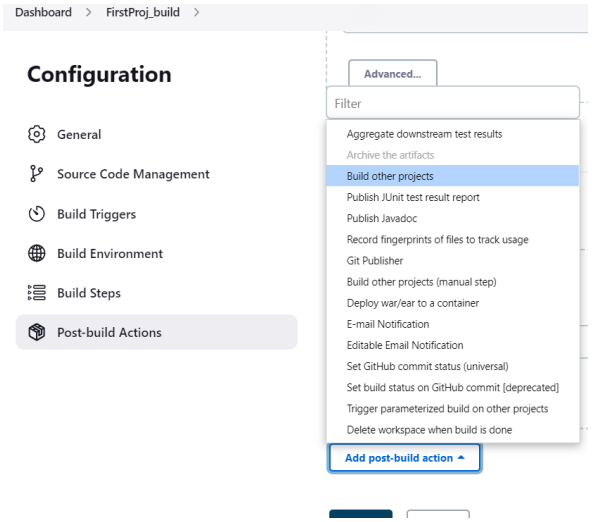
7.Post-build action->Archive artifacts.



8.Files to archive->**/*

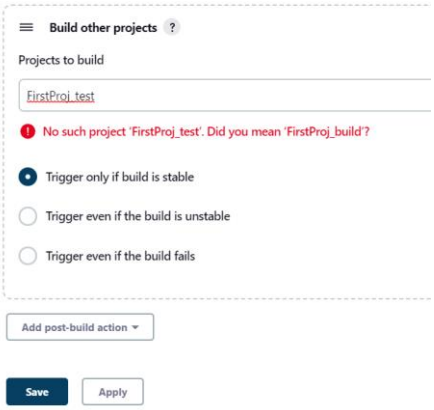


9. Post-build action->build other projects.

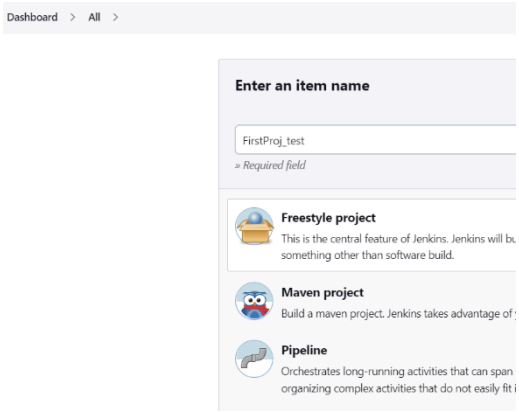


10.Give new project name->"ProjectName_test".

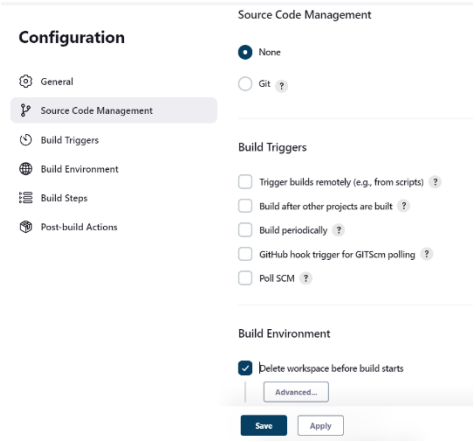
Apply->Save.



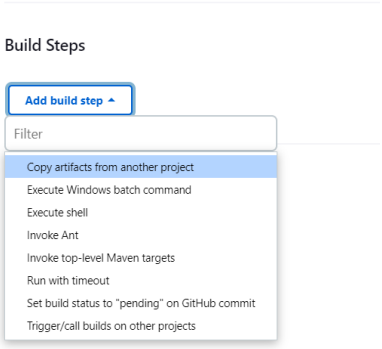
11.New item->Item name->"ProjectName_test->freestyle->Ok.



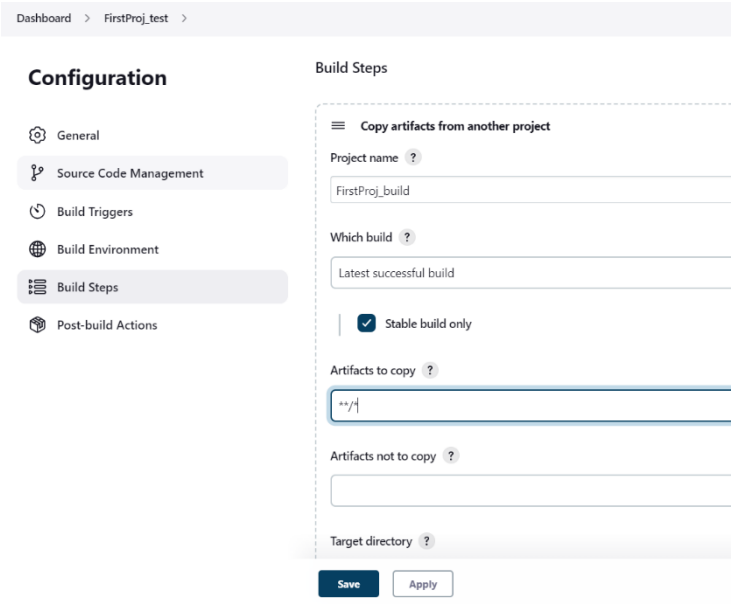
12.Source code managent->None.



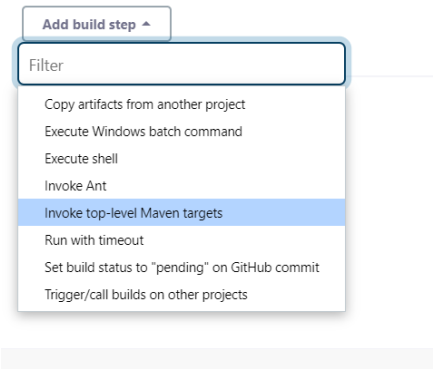
13.Add build steps->copy artifacts from another project



14.Project name->artifacts to copy->**/*.

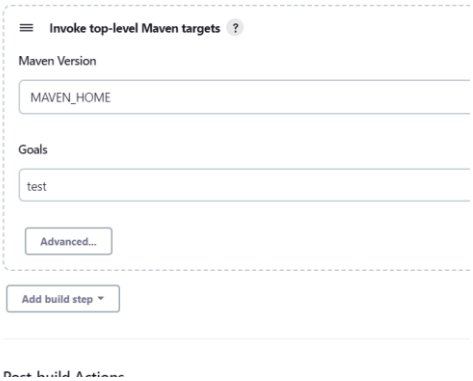


15.Add build steps->Invoke top level Maven targets.

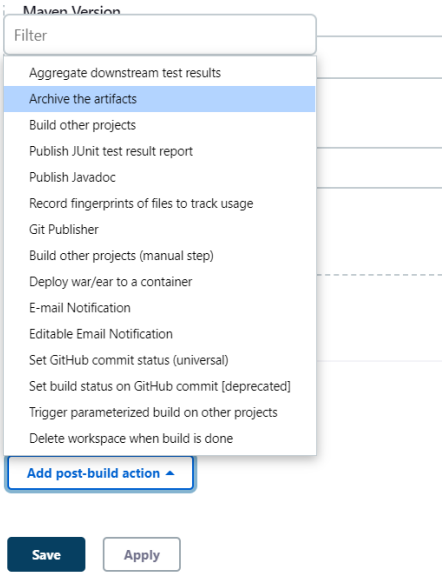


16.maven version->MAVEN_HOME.

Goals->test

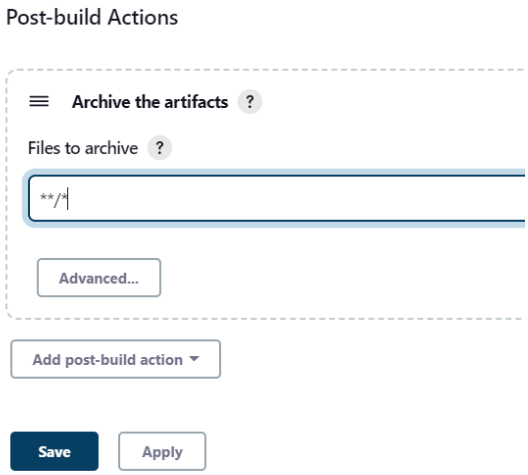


17.post-build action->Archive the artifacts.

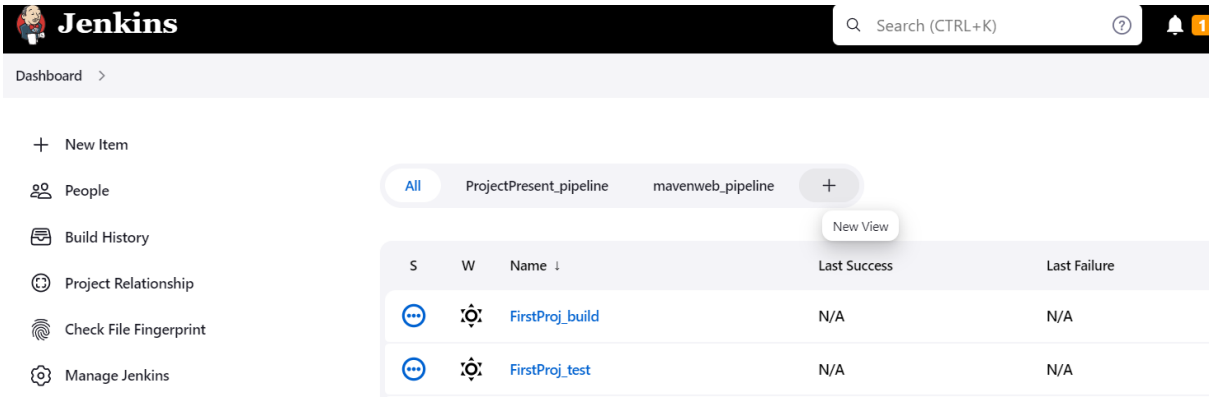


18.files to archive->**/*.

Apply->Save.

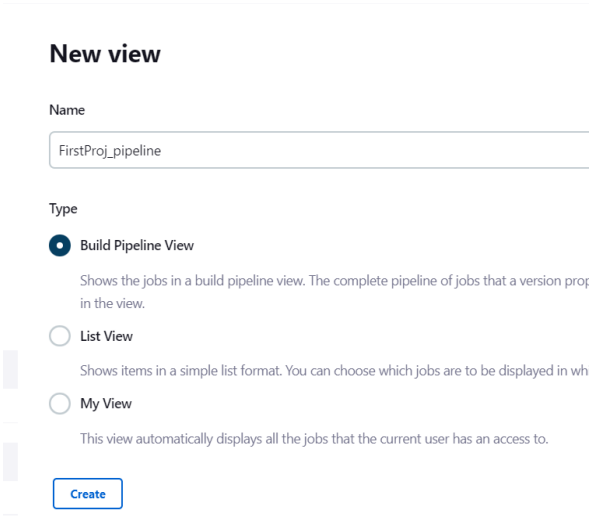


19.Dashboard->new view.

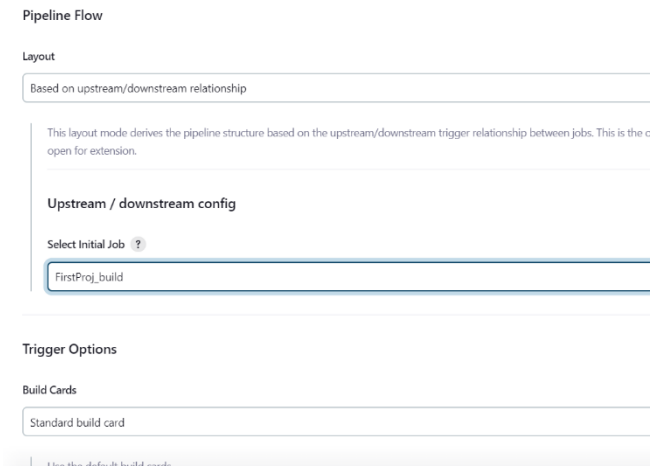


20.View name->”PipelineName_pipeline”.

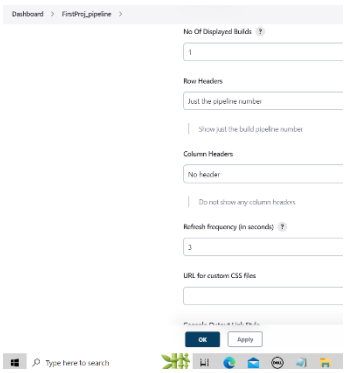
Type->build pipeline view.

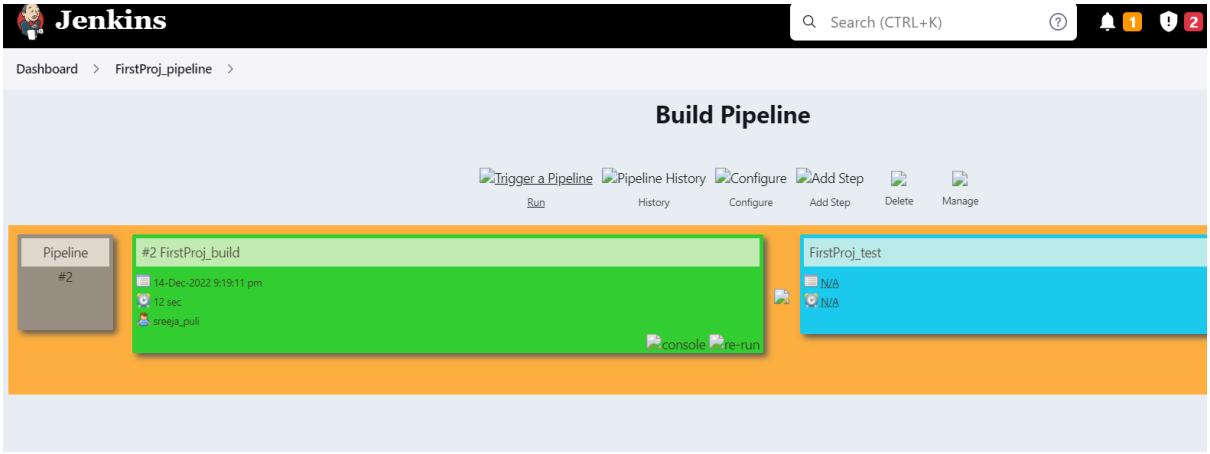


21.Select job->”build”



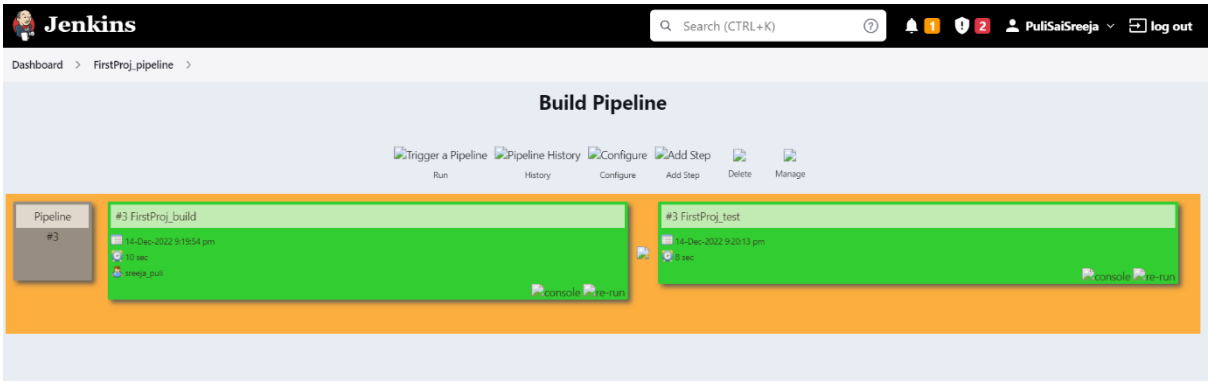
22.Apply->ok.





23.Run pipeline

24. 2-pipeline built successfully



LETS BUILD 3-PIPELINE:

After creating a new maven project on eclipse and pushing it to your repository in git hub follow the following steps.

Step1: Open Jenkins in the local host , login into your Jenkins account and create a new item.

Step 2: Name it, and select as -> freestyle project ,click on-> ok.

Step 3: Paste the code in GIT URL ,check for master /main. Here we going for main.

Step 4: Now in Build, Invoke top-level Maven targets

Step 5: Select the Maven path which is already set in the global credentials in Manage Jenkins.Follow the same

goals as done in eclipse starting with clean and install.

Step 6: Now in post build actions-> select Archive the artifacts, to send the output of build project to the testing team.

Step 7: If we want to archive all the artifacts type **/* as shown.

Step 8: Now the next step is to build other projects, where we will create a test project which will be triggered by the build project. Click apply and save.

Step 9: Create a new freestyle project test as shown and click ok.

Step 10: This time we need not mention the git repository so select None.

Step 11: In Build environment check the box as shown below, this is to discard old builds.

Step 12: To forward the artifacts of the previous project to the current test project, select copy the artifacts from another project in Build as shown.

Step 13: Give the name of the project from which we want to copy the artifacts and check the box ->stable build only->to copy all the artifacts type **/*.

Step 14: Now select Invoke top-level Maven targets in build.

Step 15: This time give the goal as test after selecting the Maven version.

Step 16: In post-build actions->select Archive the artifacts.

Step 17: To save all the artifacts->type **/* and Apply->Save.

Step 18: Create a new freestyle project test as shown and click ok.

Step 19:Give the name of the project from which we want to copy the artifacts and check the box ->stable build only->to copy all the artifacts type **/*.

Step 20: Now here we go for Add post –build action where we select the Deploy war/ear to a container.This plugin takes a war/ear file and deploys that to a running remote application server at the end of a build.

Step 21: Deploy warr/ear to a container takes the artifacts as **/*.war.

Step 22:Here we select the tomcat version.

Step 23:Here we add the credentials of tomcat .

Step 24: Here we added the credentials of tomcat and tomcat URL also. Apply and save.

Step 25: Create a pipeline by clicking on + symbol in the dashboard ->a pipeline is a collection of events or jobs which are interlinked with one another in a sequence.

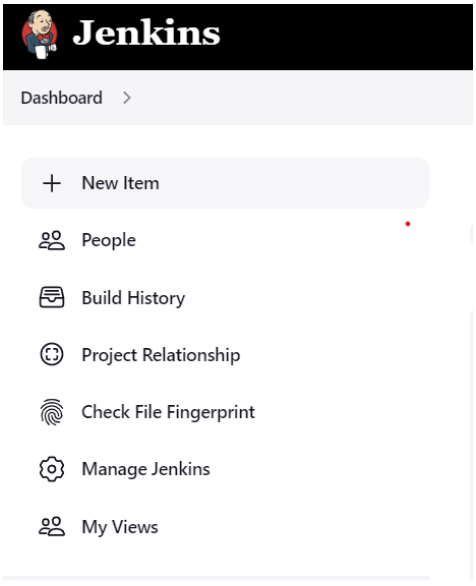
Step 26:Give a name to the pipeline->select Build Pipeline View->create.

Step 27: Select the first project to trigger the execution->build project.Apply->ok.

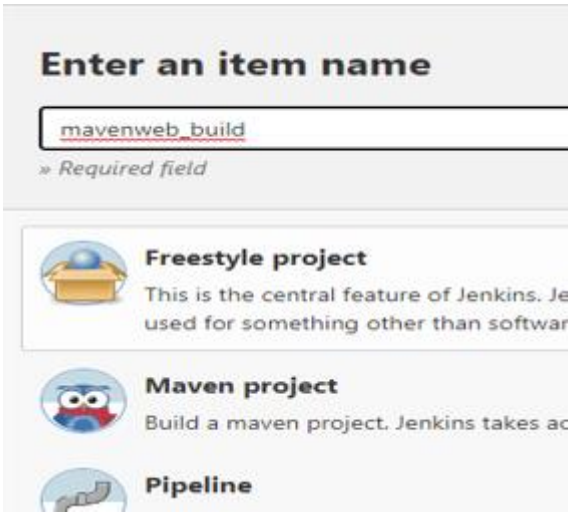
Step 28: Click on Run -> click on the small black box to open the console to check if the build is success.

Step 29: The pipeline is successful if it is in green color as shown ->check the console of the test project

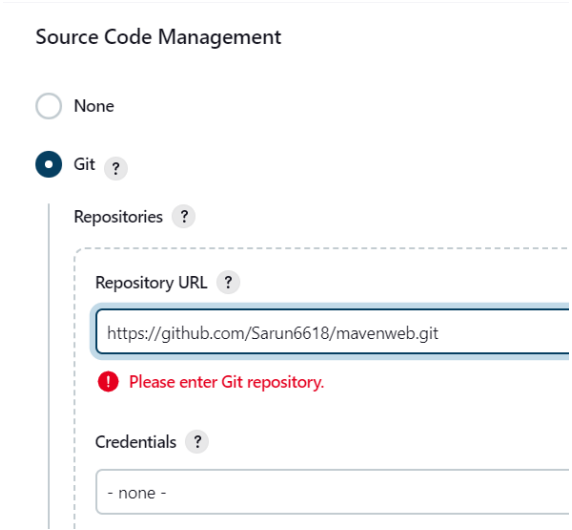
1. Dashboard ->new item.



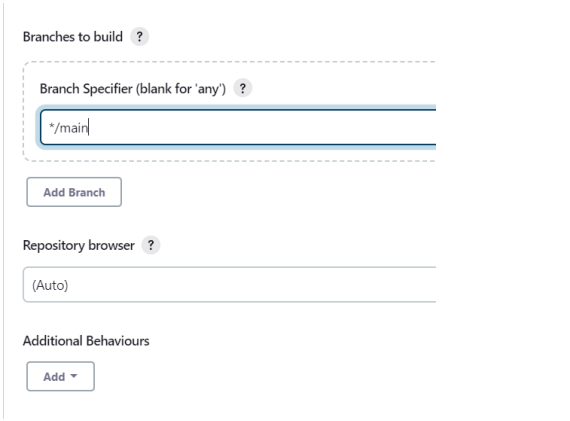
2.Enter item name->Projectname_build->Freestyle project->ok.



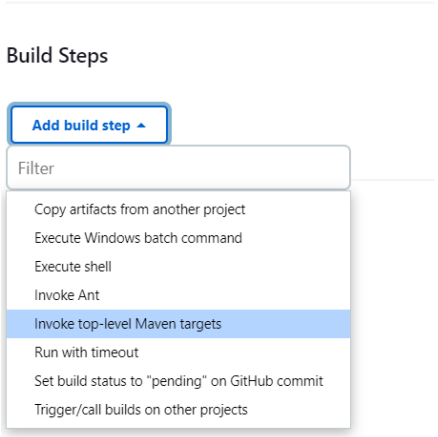
3.Select Git->paste url from git repositort.



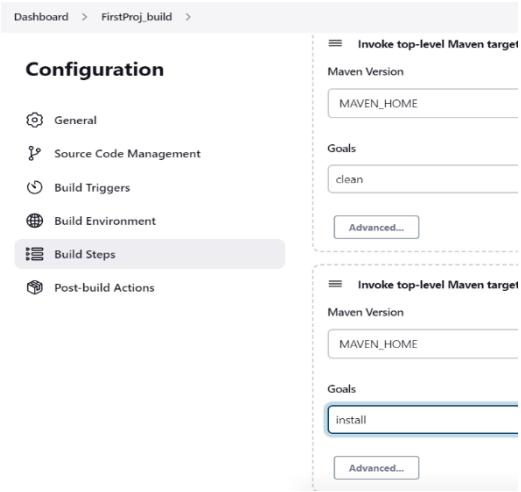
4.Type main in braches to build.



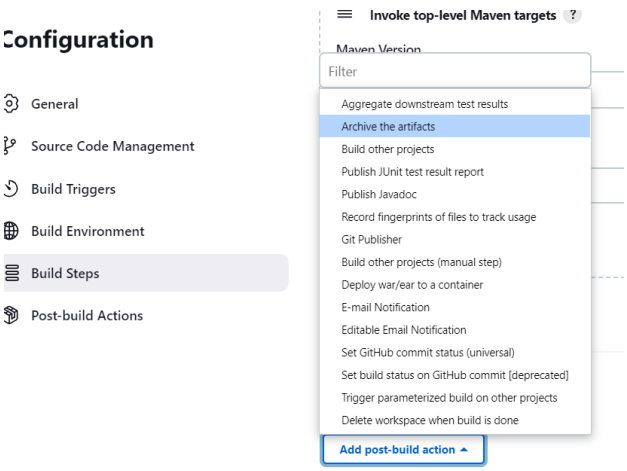
5.Build steps->invoke top-level management.



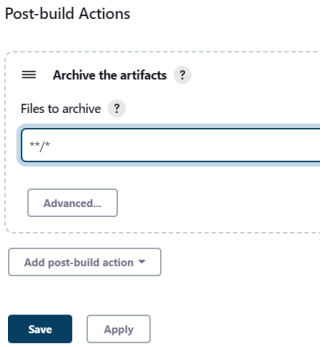
6.Maven version->MAVEN_HOME->Goals->clean.



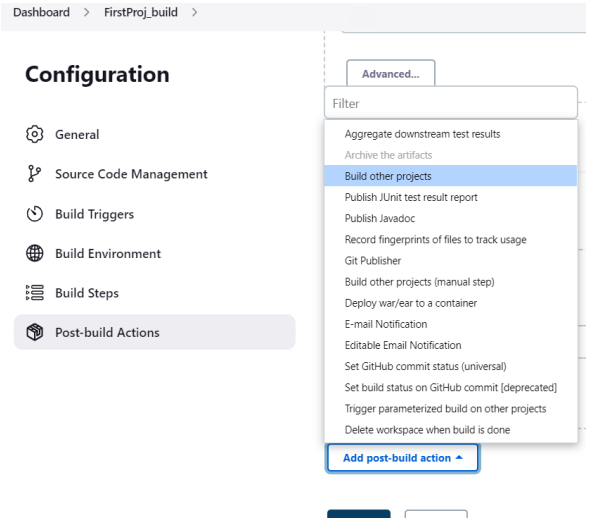
7.Post-build action->Archive artifacts.



8.Files to archive->**/*

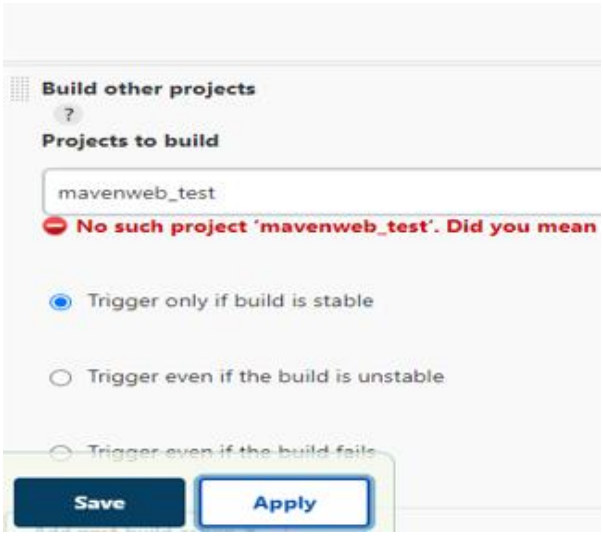


9. Post-build action->build other projects.

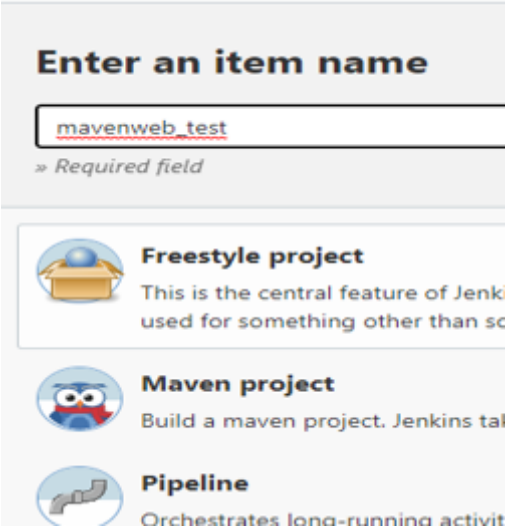


10.Give new project name->”ProjectName_test”.

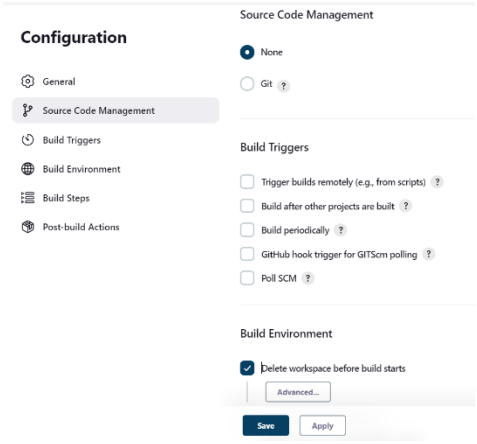
Apply->Save.



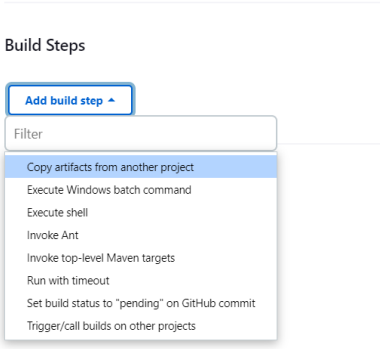
11.New item->Item name->”ProjectName_test->freestyle->Ok.



12.Source code managent->None.

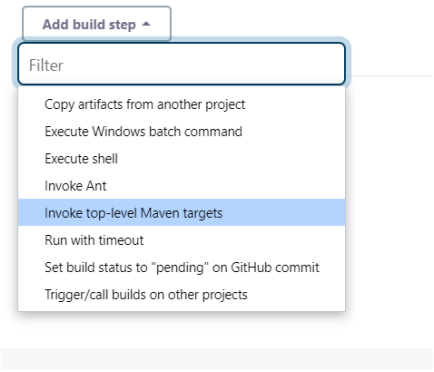


13.Add build steps->copy artifacts from another project



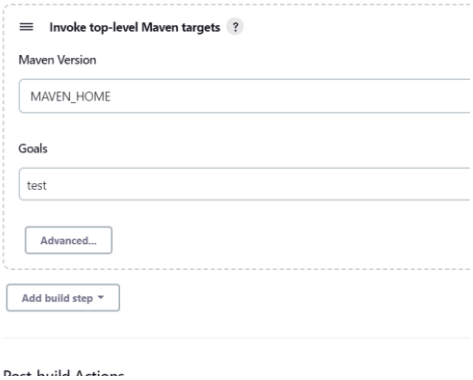
14.Project name->artifacts to copy->**/*.

15.Add build steps->Invoke top level Maven targets.

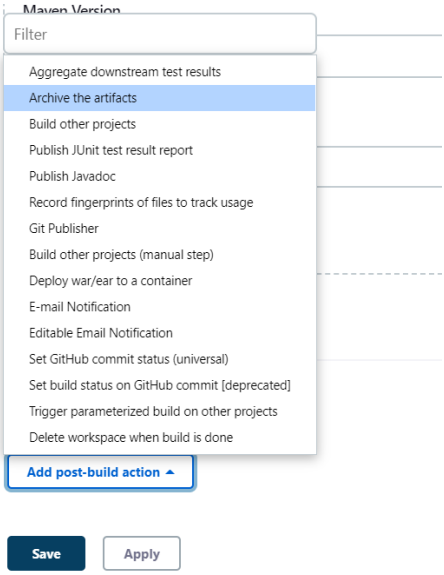


16.maven version->MAVEN_HOME.

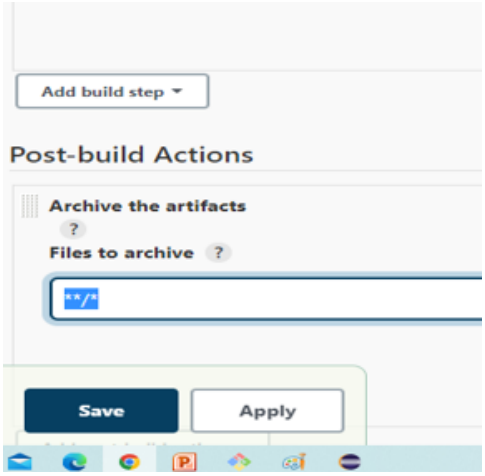
Goals->test



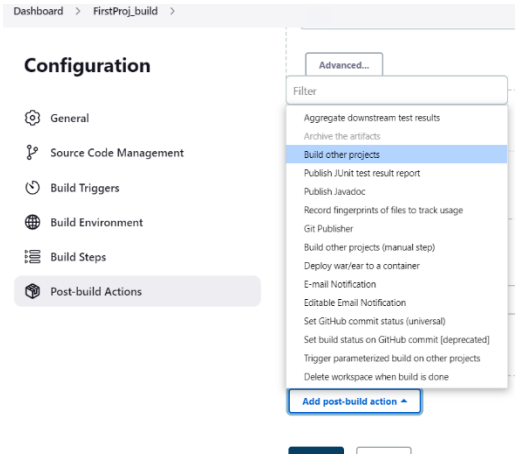
17.post-build action->Archive the artifacts.



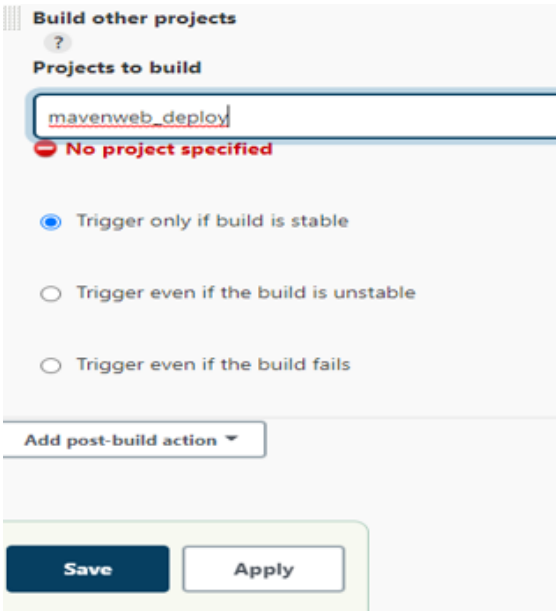
18.files to archive->**/*.



19. Post-build action->build other projects.

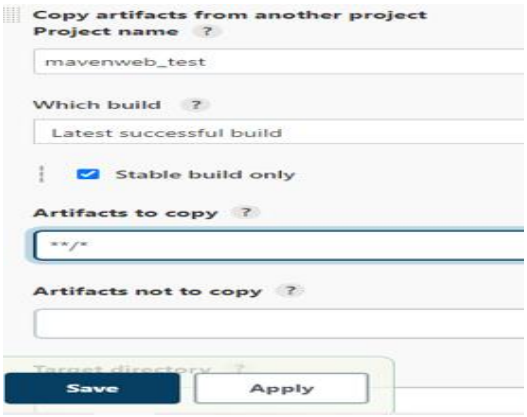


20.Give new project name->"ProjectName_deploy". Apply->Save.

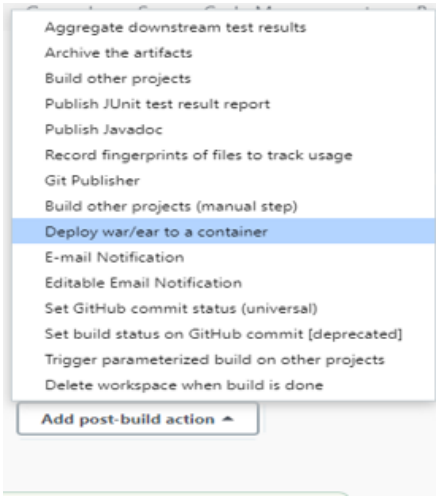


21. Enter item name->Freestyle project->ok

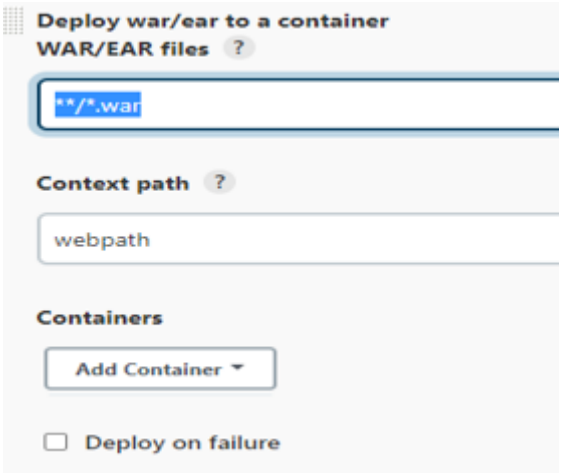
22.Name of the copy artifacts fromm other project->artifact to copt->**/*.



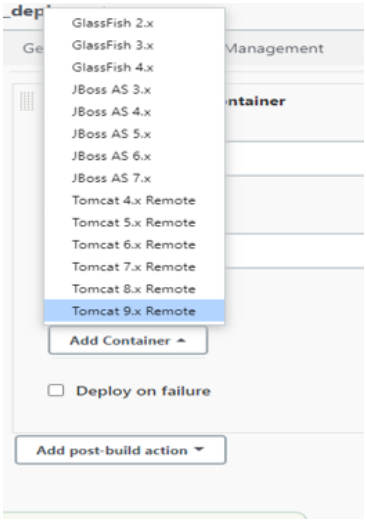
23.post-build action->deploy war/ear to container.



.24.

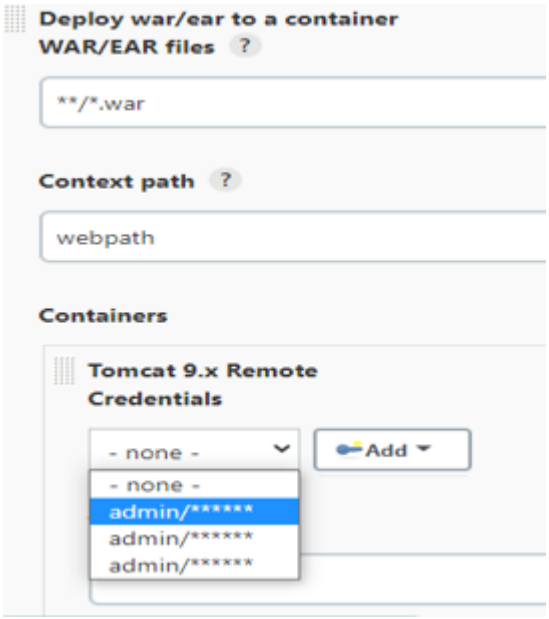


25.add container->tomcat9.x Romote.

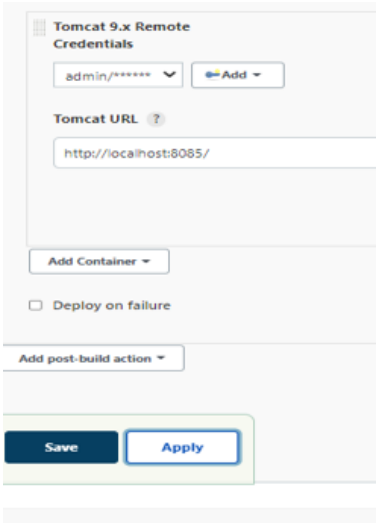


26.Enter tomcat username->password.

27. give tomcat credentials.

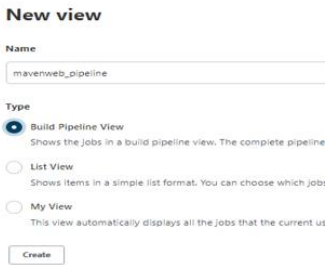


28.give tomcat url.

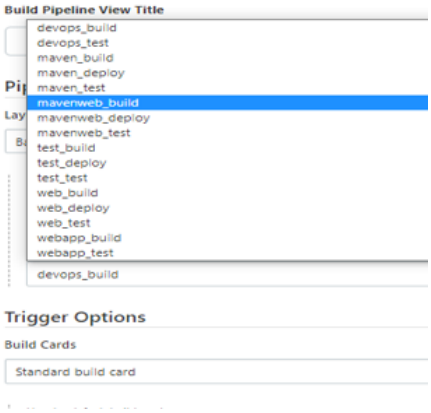


29.View name->”PipelineName_pipeline”.

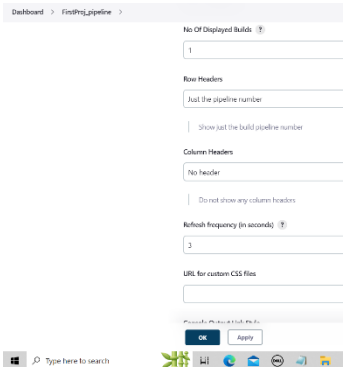
Type->build pipeline view.



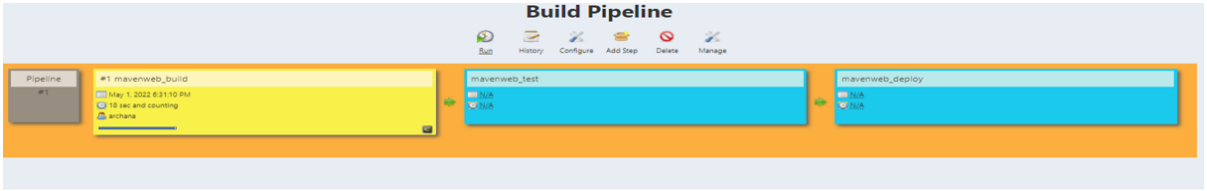
30.Select job->”Projectname_build”



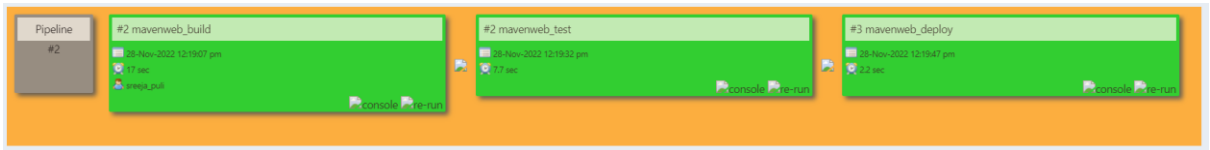
31.Apply->ok.



32.Run pipeline



33.2-pipeline built successfully



5.WORKING WITH DOCKERS

INTRODUCTION

Before docker was introduced developers and testers experienced the difficulty due to the in computer environment. i.e. the code doesn't work on other systems.
The solution was **Virtual Machine.**

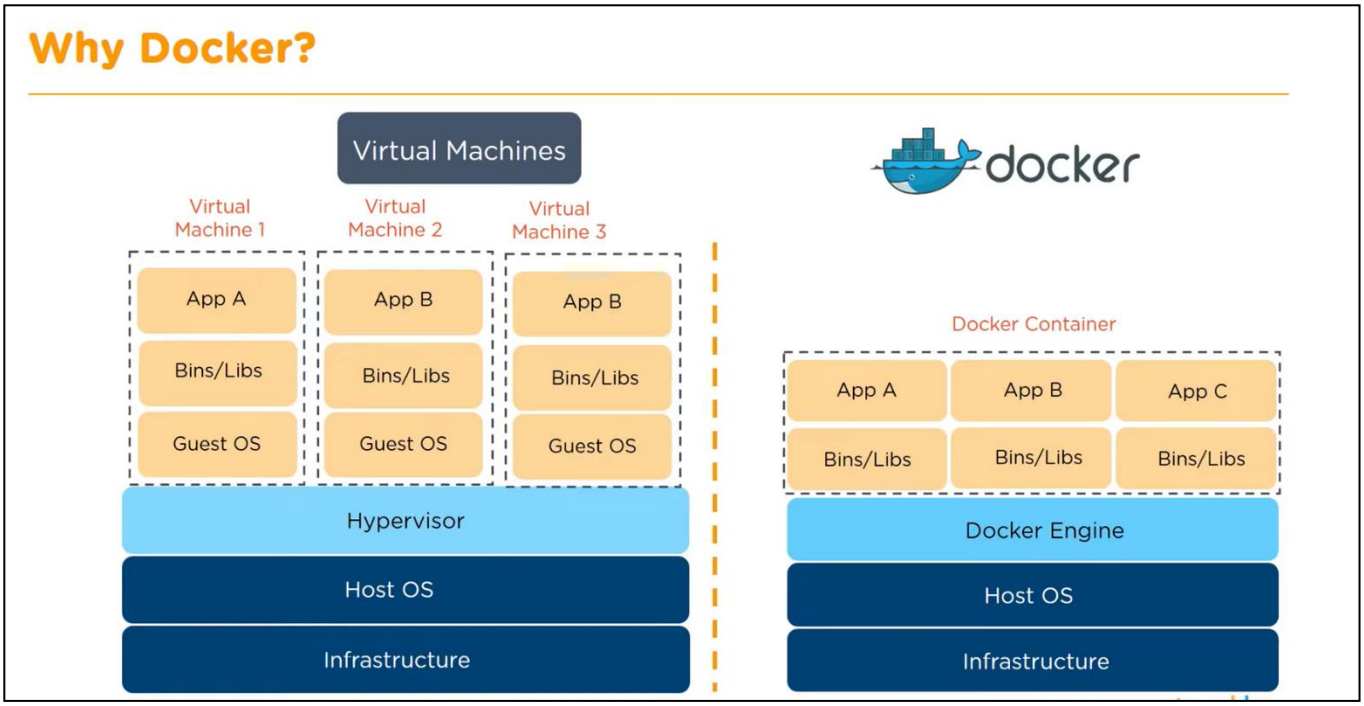
Docker was introduced as an alternative as a lightweight solution.
Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.

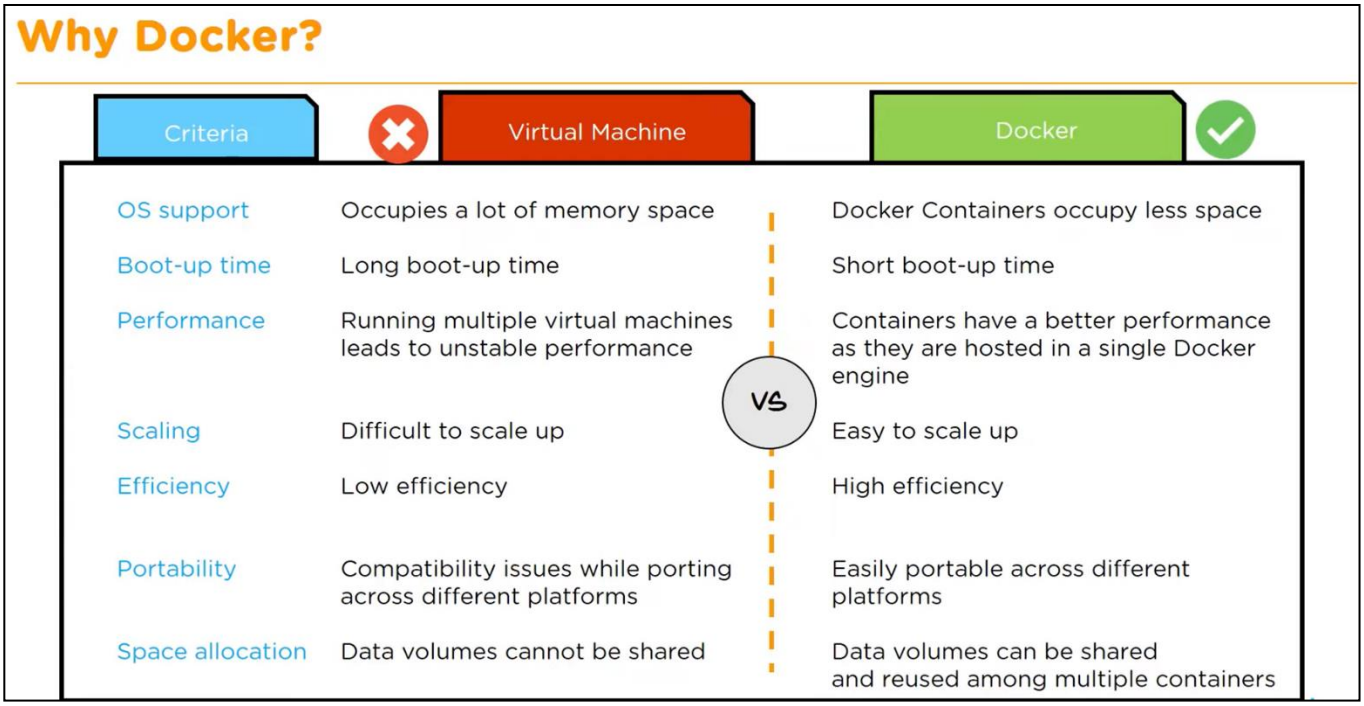
Docker provides the ability to package and run an application in a loosely isolated environment called a **Container**. The isolation and security allows you to run many containers simultaneously on a given host. Containers are lightweight and contain everything needed to run the application, so you do not need to rely on what is currently installed on the host.

Docker is written in the Go programming language and takes advantage of several features of the Linux kernel to deliver its functionality.

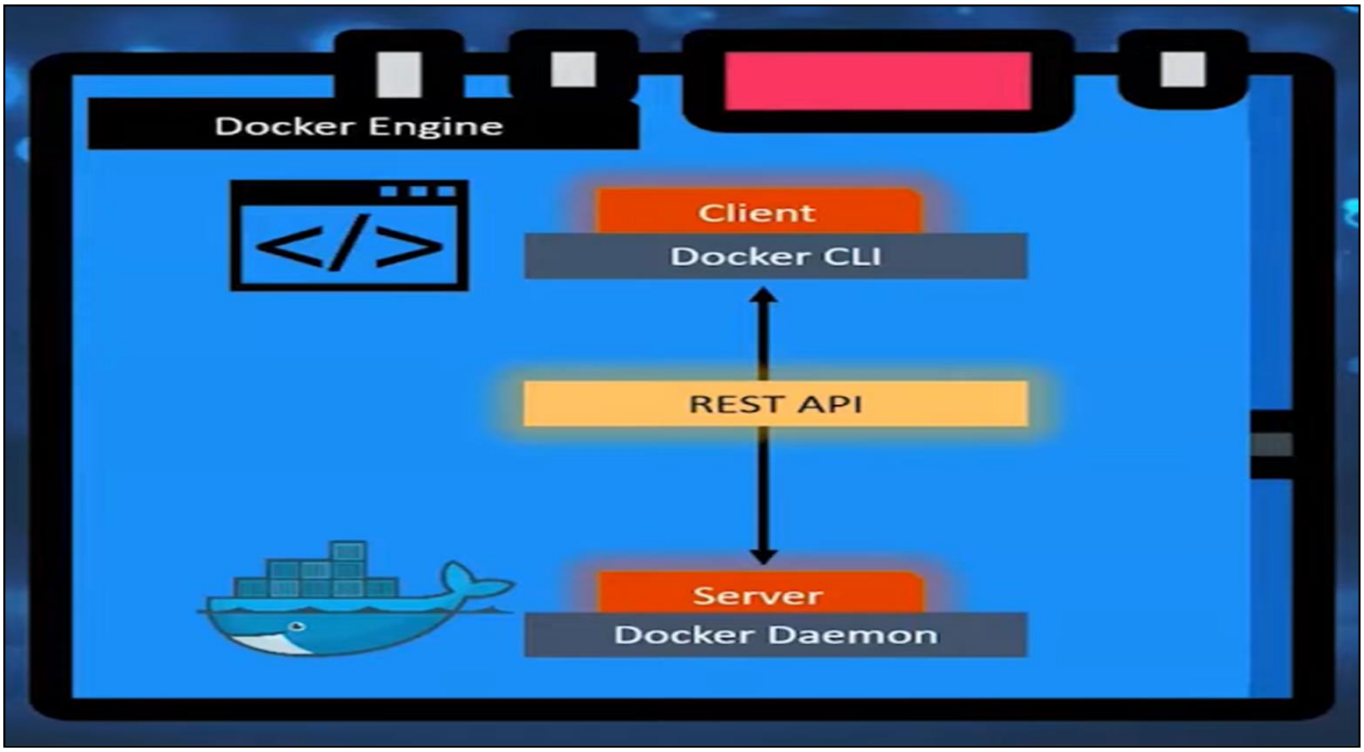
Docker uses a technology called namespaces to provide the isolated workspace called the *container*. When you run a container, Docker creates a set of *namespaces* for that container.
These namespaces provide a layer of isolation. Each aspect of a container runs in a separate namespace and its access is limited to that namespace.

VIRTUAL MACHINE vs. DOCKER



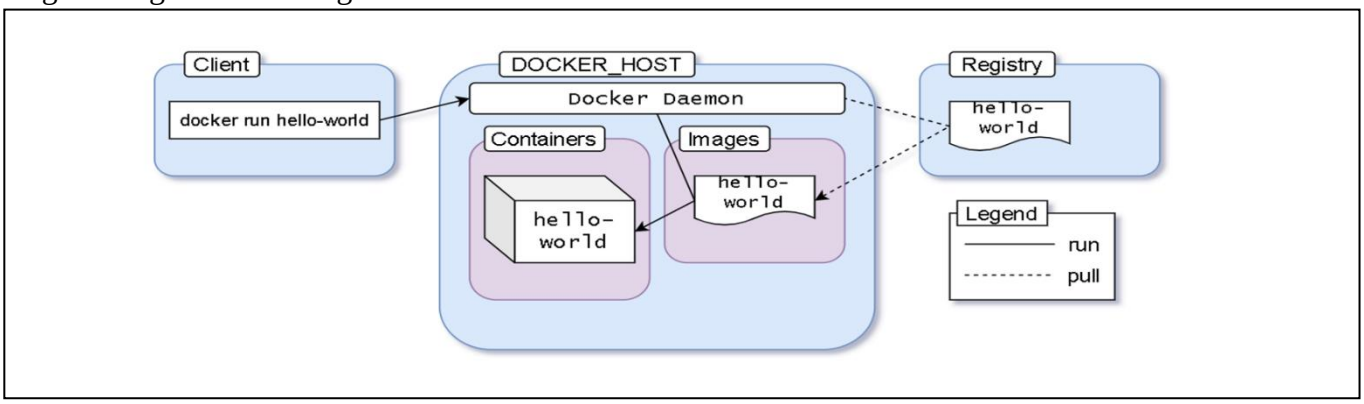


ARCHITECTURE OF DOCKER



The engine consists of three major components:
Docker Daemon: The daemon (dockerd) is a process that keeps running in the background and waits for commands from the client. The daemon is capable of managing various Docker objects.
Docker Client: The client (docker) is a command-line interface program mostly responsible for transporting commands issued by users.
REST API: The REST API acts as a bridge between the daemon and the client. Any command issued using the client passes through the API to finally reach the daemon.

Docker uses a client-server architecture.
The Docker *client* talks to the Docker *daemon*, which does the heavy lifting of building, running, and distributing your Docker containers".
User will execute commands using the client component. The client then uses the REST API to reach out to the long running daemon and get the work done.



SYSTEM REQUIREMENTS FOR DOCKER INSTALLATION

•Windows 10 64-bit: Home or Pro 21H1 (build 19043) or higher, or Enterprise or Education 20H2 (build 19042) or higher.
•Enable the WSL 2 feature on Windows.
BIOS-level hardware virtualization support must be enabled in the BIOS settings. For more information, see Virtualization

DOCKER COMMANDS

1. docker –version (It shows the Docker version present in system).
2. docker version (It shows client and server docker versions).
3. docker login (To check login credentials).
4. docker login –username <username> (It links with dockerhub).

We should be comformtable with four terms
1) Docker Images: Combinations of binaries / libraries which are necessary for one software application.
2) Docker Containers: When image is executed comes into running condition, it is called container.
3) Docker Host: Machine on which docker is installed, is called as Docker host.
4) Docker Client: Terminal used to run docker run commands (Git bash/power shell /putty/cmd prompt)

DOCKER IMAGES COMMANDS

1. docker images/docker image ls (It shows list of images present in system).
2. docker pull <imagename> (It download the image which we requested by image name from hub.docker).
3. docker rmi <imagename/imageid>(To delete a docker image from docker host).
4. docker build -t <newimagename> (To build image from a custmised container).

5. docker commit <containername/containerid> <newimagename> (To commit the container).
6. docker tag <imagename> <dockerhub id/imagename> (To tag the docker image to dockerhub)
7. docker push <imagename> (To upload a docker image to dockerhub).

DOCKER CONTAINERS COMMANDS

1. docker container ps -a (It shows all running containers present in system).
2. docker run -it <imagename> (It runs the container with image using a random name given by system).
3. docker run –name <containername> -it <imagename> (It runs the container with image using user defined container name).
4. docker run –name <containername> -d -p <PORT NO> <imagename> (It runs the container using localhost ports).
5. docker start <containername> (To start the container).
6. docker stop <containername> (To stop the running container).
7. docker stop $(docker ps -aq) (To stop all containers at once).
8. docker rm <containername>(To delete a docker container).
9. docker rm -f <containername/container id> (To forcefully remove of container).
10. docker rm $(docker ps -aq) (To delete all the containers at once).

DOCKER OPTION COMMANDS

-it (for opening an interactive terminal in a container).
--name (Used for giving a name to a container).
-d (Used for running the container in detached mode as a background process).
-p (Used for port mapping between port of container with the dockerhost port).
-P (Used for automatic port mapping ie, it will map the internal port of the container with some port on host machine. This host port will be some number greater than 3000).
-v (Used for attaching a volume to the container).
--link (Used for linking the container for creating a multi container architecture).
-e (Used for passing environment varaibles to the container).

DOCKER LAB COMMANDS

1. docker –version

```
PS C:\WINDOWS\system32> docker --version
Docker version 20.10.21, build baeda1f
```

2. docker version

```
PS C:\WINDOWS\system32> docker version
Client:
 Cloud integration: v1.0.29
 Version:           20.10.21
 API version:       1.41
 Go version:        go1.18.7
 Git commit:        baeda1f
 Built:             Tue Oct 25 18:08:16 2022
 OS/Arch:           windows/amd64
 Context:           default
 Experimental:      true

Server: Docker Desktop 4.14.1 (91661)
 Engine:
  Version:          20.10.21
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.18.7
  Git commit:       3056208
  Built:            Tue Oct 25 18:00:19 2022
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.9
  GitCommit:        1c90a442489720eec95342e1789ee8a5e1b9536f
 runc:
  Version:          1.1.4
  GitCommit:        v1.1.4-0-g5fd4c4d
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

3. docker images / docker image ls

```
PS C:\WINDOWS\system32> docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
PS C:\WINDOWS\system32> docker image ls
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

4. docker container ps -a

```
PS C:\WINDOWS\system32> docker container ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

5. docker pull <imagename>

```
PS C:\WINDOWS\system32> docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
6e3729cf69e0: Pull complete
Digest: sha256:27cb6e6ccef575a4698b66f5de06c7ecd61589132d5a91d098f7f3f9285415a9
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

6. docker images

```
PS C:\WINDOWS\system32> docker images
REPOSITORY   TAG       IMAGE ID       CREATED      SIZE
ubuntu       latest    6b7dfa7e8fdb   5 days ago   77.8MB
```

7. docker container ps -a

```
PS C:\WINDOWS\system32> docker container ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

8. docker run -it <imagename>

```
PS C:\WINDOWS\system32> docker run -it ubuntu
root@f48e56ec72d0:/# echo hii
hii
root@f48e56ec72d0:/# exit
exit
```

9. docker container ps -a

```
PS C:\WINDOWS\system32> docker container ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED             STATUS                         PORTS     NAMES
f48e56ec72d0   ubuntu    "bash"    About a minute ago   Exited (0) About a minute ago            awesome_easley
```

10. docker run –name <containername> -it <imagename>

```
PS C:\WINDOWS\system32> docker run --name myubuntu -it ubuntu
root@f96efcc11057:/#
```

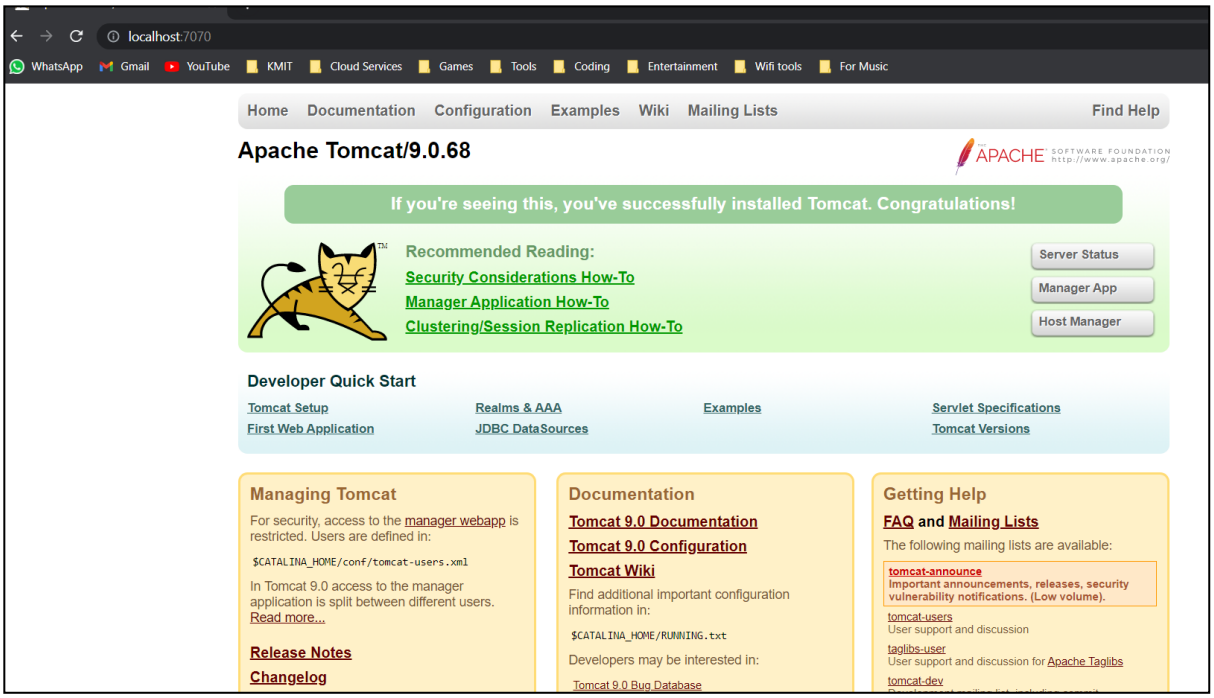
11. docker container ps -a

```
PS C:\WINDOWS\system32> docker container ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED          STATUS                      PORTS     NAMES
e73da11fa433   ubuntu    "bash"    19 seconds ago   Exited (0) 3 seconds ago              myubuntu
f48e56ec72d0   ubuntu    "bash"    5 minutes ago    Exited (0) 5 minutes ago              awesome_easley
```

12. docker run --name <containername> -d -p <userportno>:<defaultportno> <imagename> (to create an image and container at the same time without pulling)

```
PS C:\WINDOWS\system32> docker run --name mytomee -d -p 7070:8080 tomee
Unable to find image 'tomee:latest' locally
latest: Pulling from library/tomee
846c0b181fff: Pull complete
08b0873b98ac: Pull complete
d26a502e2cfa: Pull complete
459bdc265a8a: Pull complete
c5fa6822f5f6: Pull complete
a1b5ed548c03: Pull complete
5f83fa7cfd63: Pull complete
140517ad033e: Pull complete
Digest: sha256:37c2d16bd0f650d05ec98fc0628bd7fc0ce8c38f2023b8feccac2b5b50f1cd6a
Status: Downloaded newer image for tomee:latest
4cf7aa70da443918ae408b0a52c3766543a6cea57607961865c830090029bc63
PS C:\WINDOWS\system32>
```

13. Tomee is running on localhost:7070



14. docker container ps -a

```
PS C:\WINDOWS\system32> docker container ps -a
CONTAINER ID   IMAGE     COMMAND           CREATED         STATUS                     PORTS                    NAMES
134b63034f8f   tomee     "catalina.sh run"   3 minutes ago   Up 3 minutes               0.0.0.0:7070->8080/tcp   mytomee
e73da11fa433   ubuntu    "bash"              8 minutes ago   Exited (0) 8 minutes ago                            myubuntu
f48e56ec72d0   ubuntu    "bash"              13 minutes ago  Exited (0) 13 minutes ago                           awesome_easley
```

15. docker stop <containername>

```
PS C:\WINDOWS\system32> docker stop mytomee
mytomee
```

16. docker start <containername>

```
PS C:\WINDOWS\system32> docker start myubuntu
myubuntu
```

17. docker exec -it <container name> bash

```
PS C:\WINDOWS\system32> docker exec -it myubuntu bash
root@f96efcc11057:/# apt-get update
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
0% [2 InRelease 116 kB/270 kB 43%]
```

18. In ubuntu we have to check is the system is up to date so we use apt-get update and apt-install git to install git in ubuntu

```
root@e73da11fa433:/# apt-get install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

```
root@e73da11fa433:/# git --version
git version 2.34.1
```

CUSTOMISED CONTAINER

➔ To commit images using git

19. docker commit <container name> <commit name>

```
PS C:\WINDOWS\system32> docker commit myubuntu ubuntu-git
sha256:5153d14968b9999355852629bdb983a66e52db5e6da5df86fc73129f41ef1d4d
```

20. docker run –name <containername> -it <customisedimagename>

```
PS C:\WINDOWS\system32> docker run --name ubuntu-git -it ubuntu-git
root@dc08f8941101:/# git --version
git version 2.34.1
root@dc08f8941101:/#
```

21. docker login

```
PS C:\WINDOWS\system32> docker login
Authenticating with existing credentials...
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/
access-tokens/
PS C:\WINDOWS\system32>
```

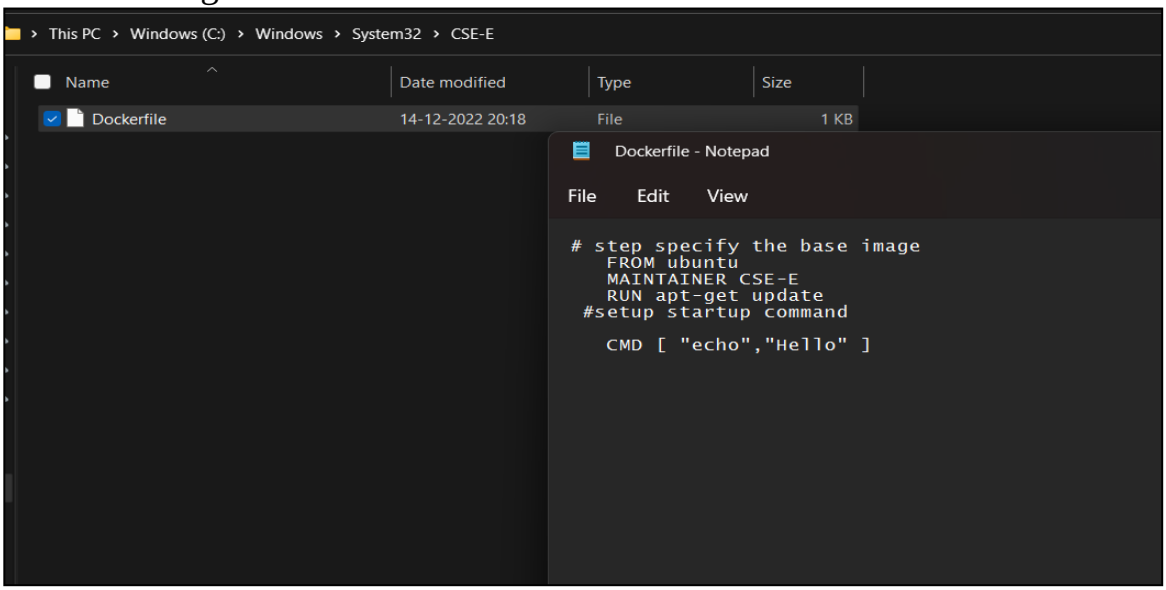
22. docker tag <imagename> <dockerid>/<commitname>

```
PS C:\WINDOWS\system32> docker tag ubuntu-git 3rdcsee/ubuntu-git
PS C:\WINDOWS\system32>
```

23. docker push <dockerid>/<commitname>

```
PS C:\WINDOWS\system32> docker push 3rdcsee/ubuntu-git
Using default tag: latest
The push refers to repository [docker.io/3rdcsee/ubuntu-git]
49040da2a493: Pushed
6515074984c6: Mounted from library/ubuntu
latest: digest: sha256:d5f07cf3dee0a03d04c009403bfbb59cce0ec536cd9609661f69e91b508eb1f6 size: 741
PS C:\WINDOWS\system32>
```

To build using dockerfile



24. docker build -t <containername> .

```
PS C:\WINDOWS\system32> cd CSE-E
PS C:\WINDOWS\system32\CSE-E> docker build -t ubuntu-build .
[+] Building 21.4s (6/6) FINISHED
 => [internal] load build definition from Dockerfile                                    1.1s
 => => transferring dockerfile: 184B                                                    0.3s
 => [internal] load .dockerignore                                                       0.9s
 => => transferring context: 2B                                                         0.3s
 => [internal] load metadata for docker.io/library/ubuntu:latest                        0.0s
 => [1/2] FROM docker.io/library/ubuntu                                                 0.7s
 => [2/2] RUN apt-get update                                                           18.0s
 => exporting to image                                                                  1.1s
 => => exporting layers                                                                 1.0s
 => => writing image sha256:1eef2366185b34ff4ed03bf7ad9242eb18ee0523b214c9b90e76d5537b7540cc  0.0s
 => => naming to docker.io/library/ubuntu-build                                         0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
PS C:\WINDOWS\system32\CSE-E>
```

25. docker run –name <buildname> -it <containername>

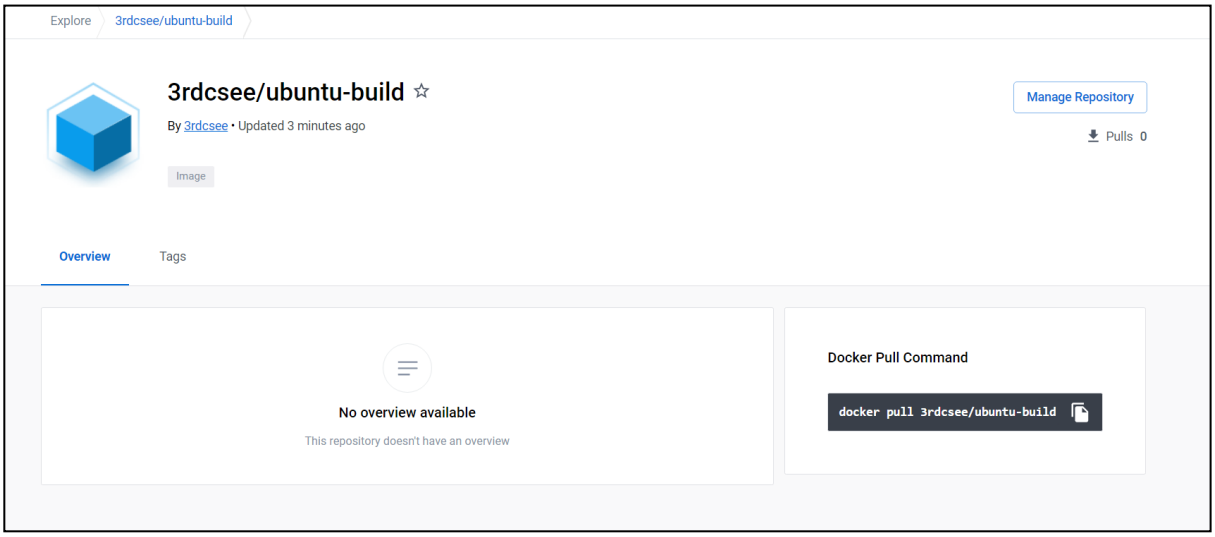
```
PS C:\WINDOWS\system32\CSE-E> docker run --name ubuntu-build -it ubuntu-build
Hello
PS C:\WINDOWS\system32\CSE-E>
```

26. docker tag <buildname> <dockerid>/<containername>

```
PS C:\WINDOWS\system32\CSE-E> docker tag ubuntu-build 3rdcsee/ubuntu-build
PS C:\WINDOWS\system32\CSE-E>
```

27. docker push <dockerid> /<buildname>

```
C:\WINDOWS\system32\CSE-E> docker push 3rdcsee/ubuntu-build
ing default tag: latest
 push refers to repository [docker.io/3rdcsee/ubuntu-build]
ab7f47bf1: Pushed
15074984c6: Mounted from 3rdcsee/ubuntu-git
test: digest: sha256:10119de8cc950aa2bdd003debddc4af9bba0a65b921f9dc9e480a9a092a3a0b8 size: 741
C:\WINDOWS\system32\CSE-E>
```



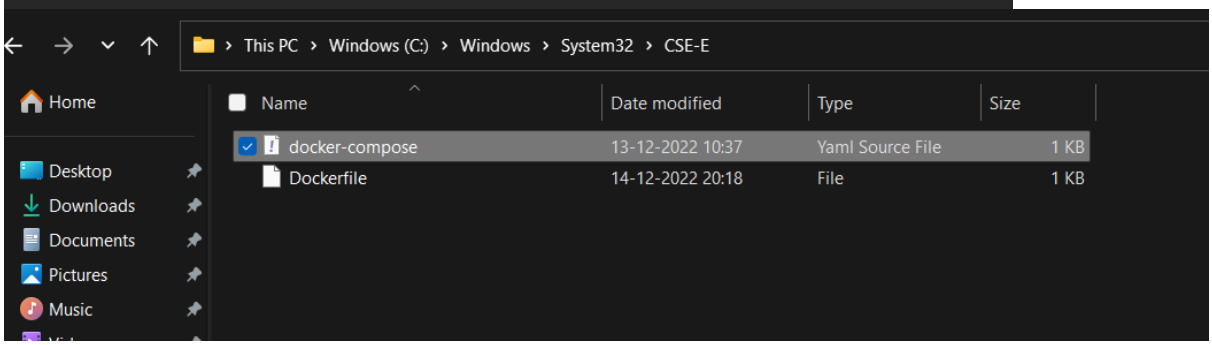
DOCKER-COMPOSE FOR WORDPRESS

1. create a docker file with name docker-compose with given data below.

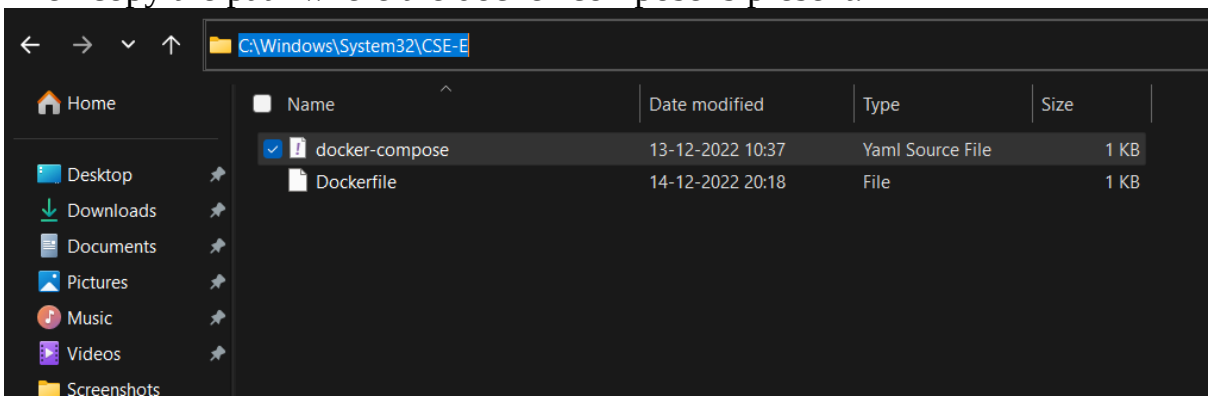
docker-compose - Notepad

File Edit View

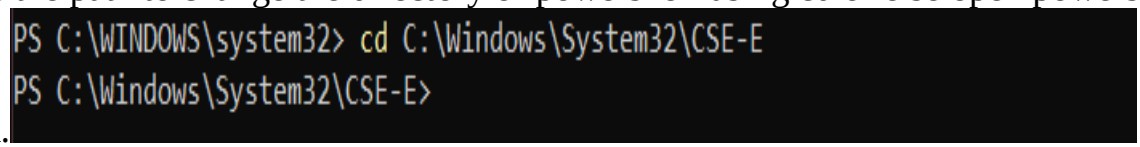
```
version: '3'
services:
  # Database
  db:
    image: mysql:5.7
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: password
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress
    networks:
      - wpsite

  # 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:
```



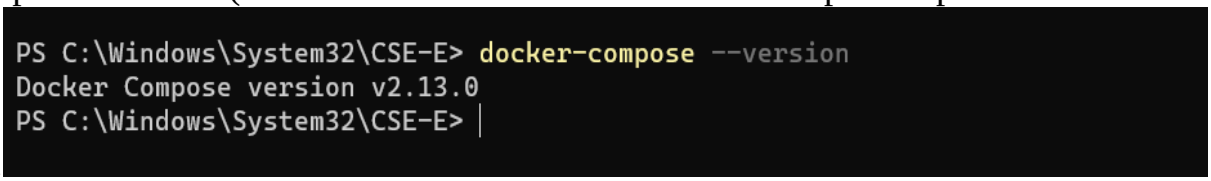
2. Then copy the path where the docker-compose is present.



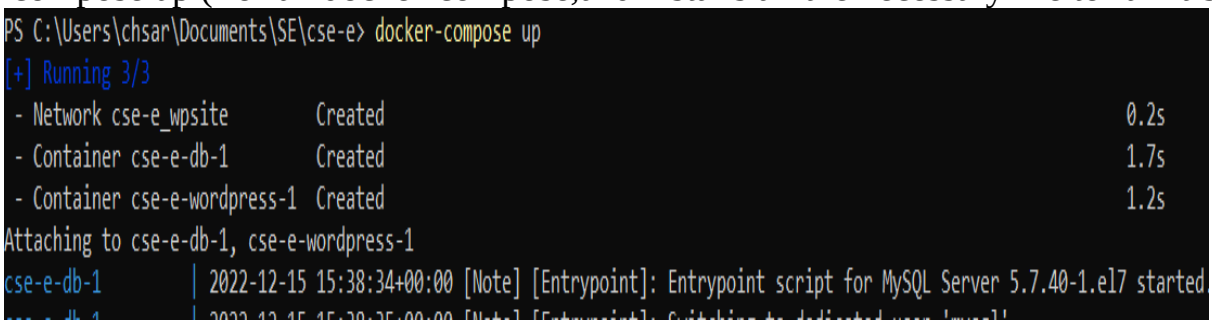
3. And use the path to change the directory of powershell using cd or else open powershell at the path.

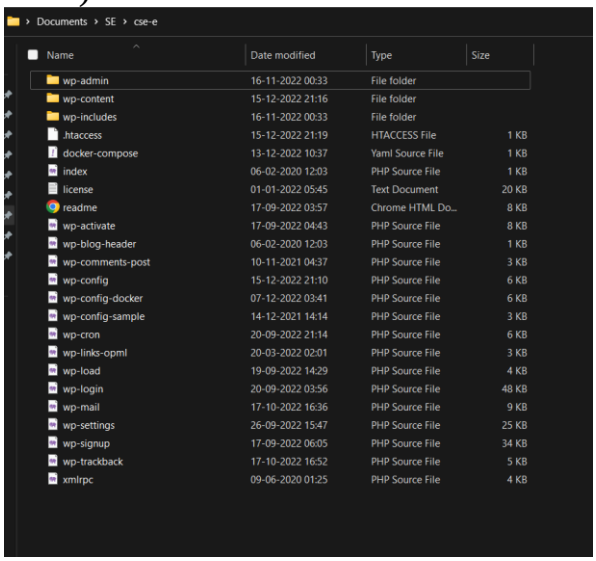


4. docker-compose –version (Then to check whether the docker-compose is present use command).

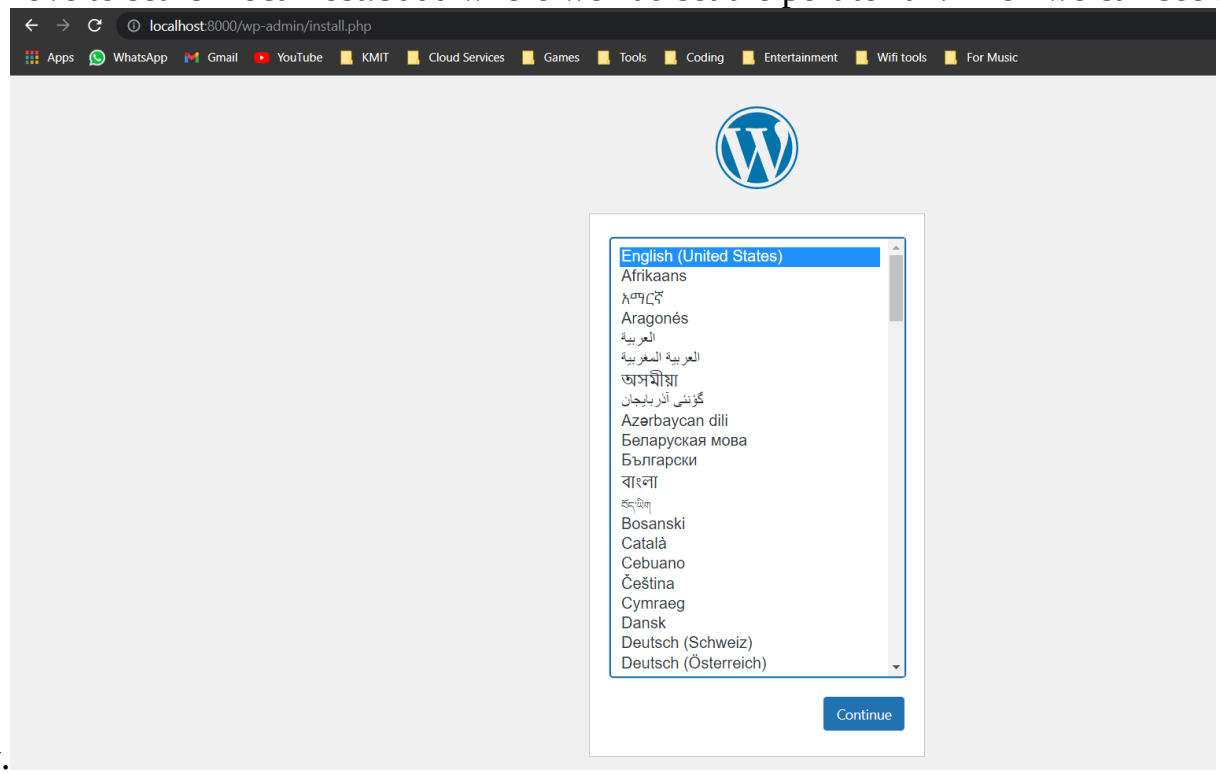


5. docker-compose up (To run docker-compose,and installs all the necessary file to run it by itself).

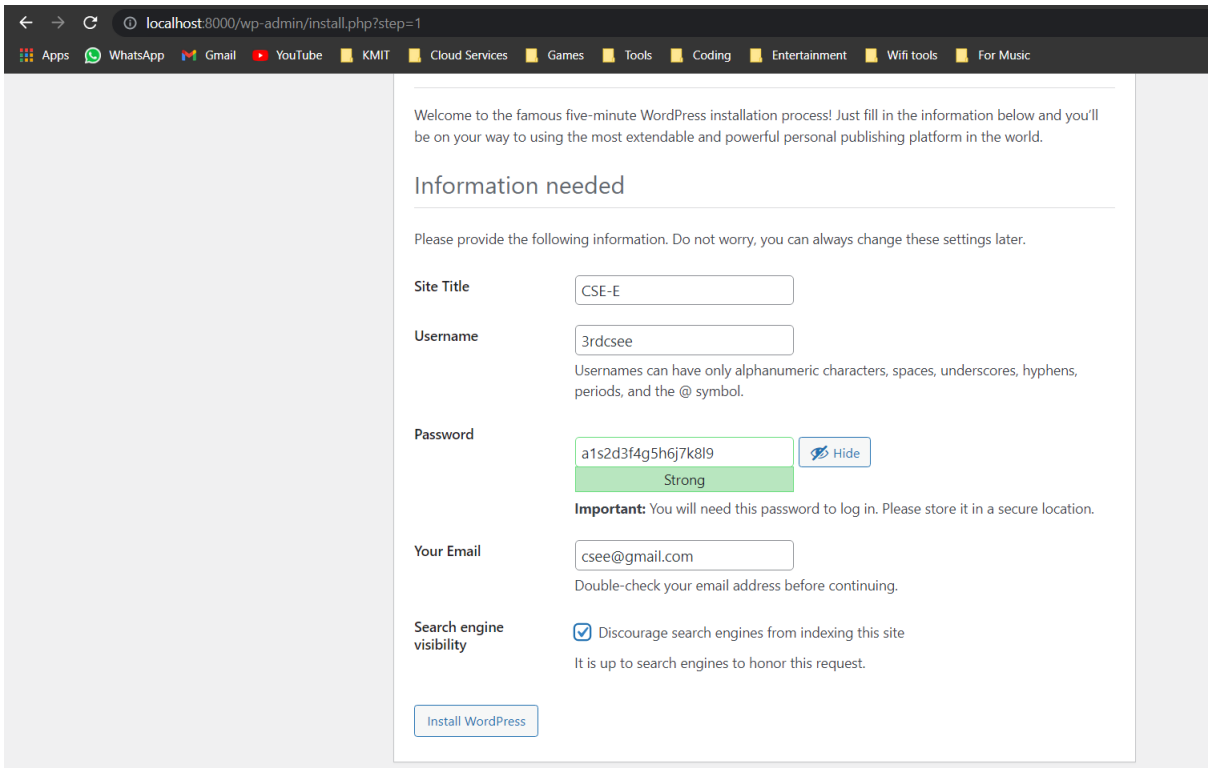




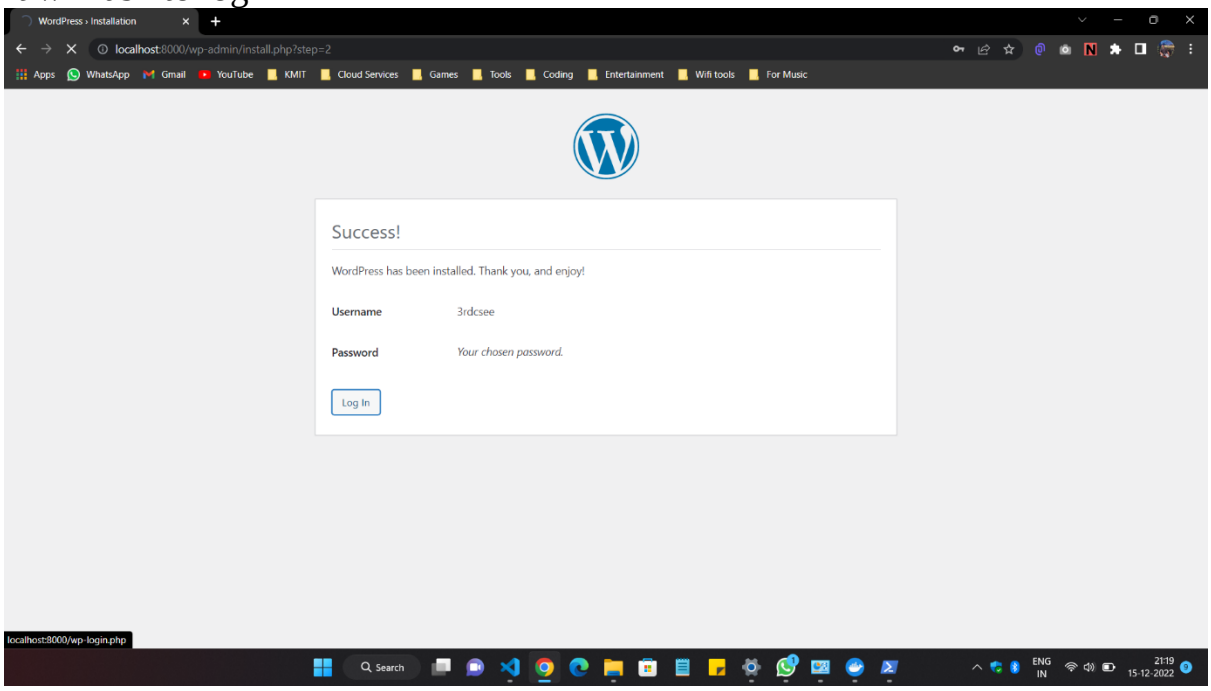
6. Then move to search localhost:8000 where we had set the port to run. Then we can see as below.



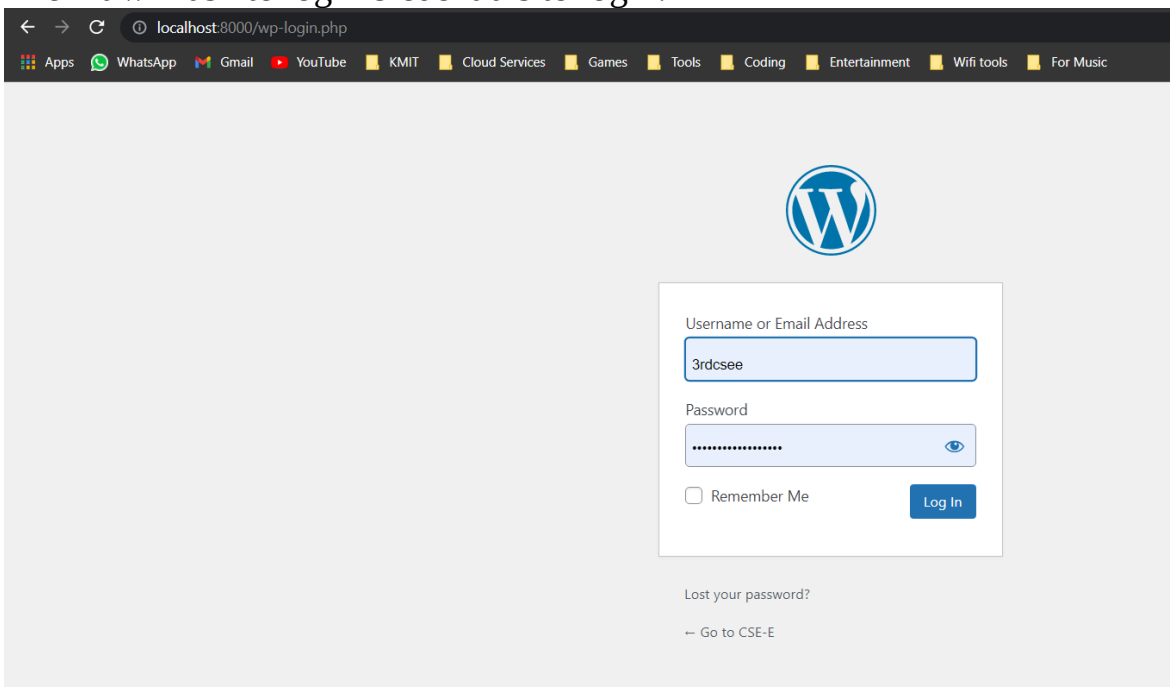
7. Just fill the information needed.



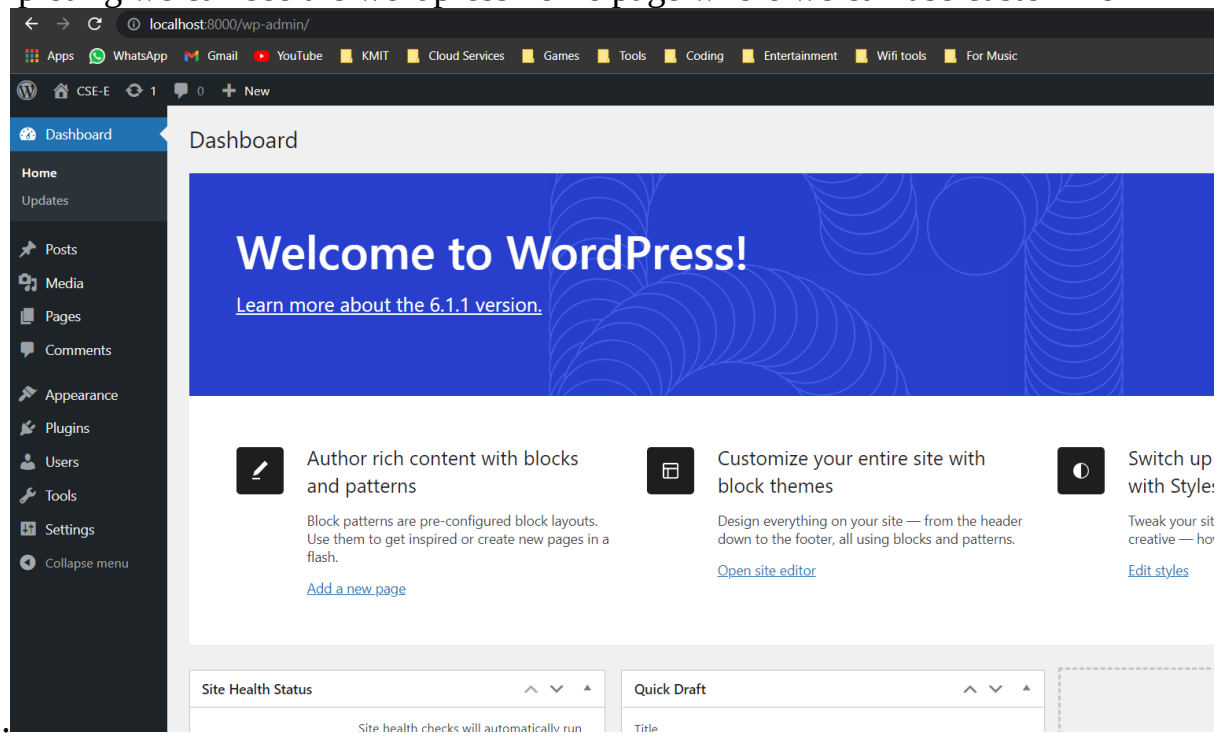
8. It will ask to login



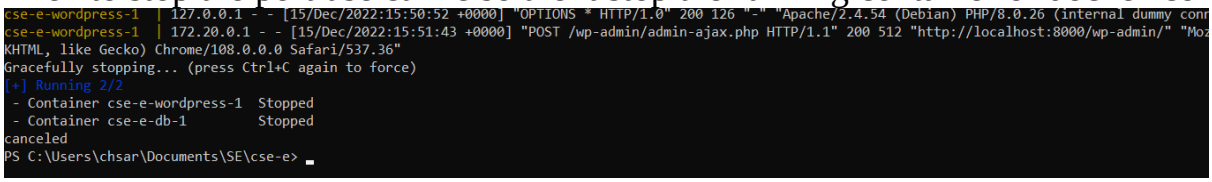
9. Then it will ask to login credentials to login.



10. After completing we can see the wordpress home page where we can use customize tools,etc..



11. Then to stop the port use ctrl+c so the it stop the running container of docker-compress.



DOCKER NAGIOS

Nagios is an open source **monitoring system** for computer systems. It was designed to run on the Linux operating system and can monitor devices running Linux, Windows and Unix operating systems (OSes).

Nagios software runs periodic checks on critical parameters of application, network and server resources. For example, Nagios can monitor memory usage, disk usage, microprocessor load, the number of currently running processes and log files. Nagios also can monitor services, such as Simple Mail Transfer Protocol (SMTP), Post Office Protocol 3 (POP3), Hypertext Transfer Protocol (HTTP) and other common network protocols. Active checks are initiated by Nagios, while passive checks come from external applications connected to the monitoring tool.

Originally called NetSaint and released in 1999, Nagios was developed by Ethan Galstad and subsequently refined by numerous contributors as an open source project. Nagios Enterprises, a company based around the Nagios Core technology, offers multiple products, such as XI, Log Server, Network Analyzer and Fusion.

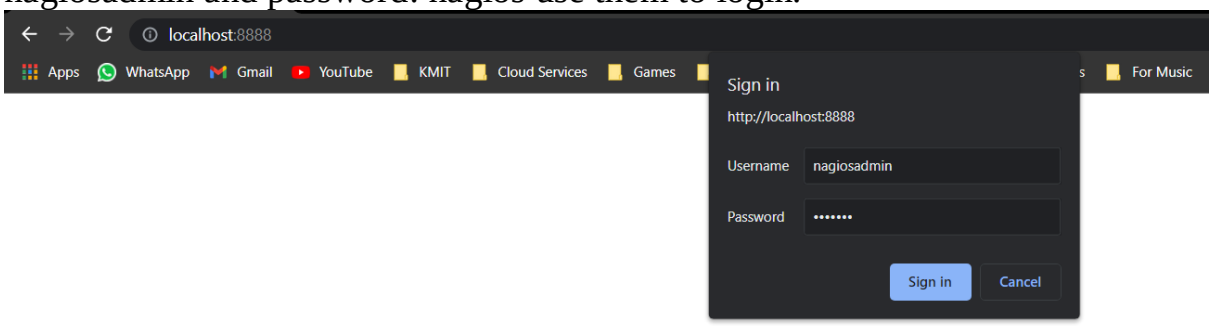
1. docker pull <imagename>(To pull nagios image name jasonrivers/nagios:latest).

```
PS C:\WINDOWS\system32> docker pull jasonrivers/nagios:latest
latest: Pulling from jasonrivers/nagios
eaead16dc43b: Pulling fs layer
4bae43e3fd96: Extracting [==========================================>        ]  193.9MB/218.8MB
52ad2fe8213e: Download complete
1fefd86dc30b: Download complete
c76d27f07a58: Download complete
```

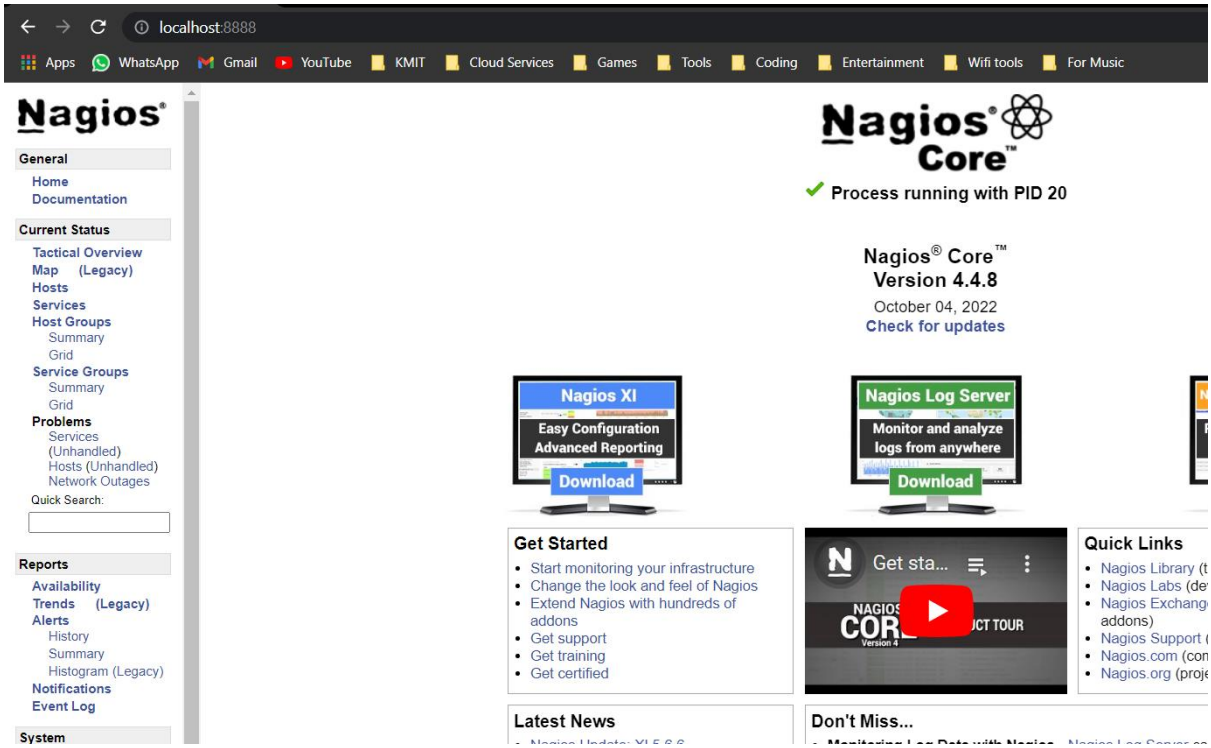
2. docker run --name <containername> -p <userportno>:<defaultportno> <imagename> ().

```
PS C:\WINDOWS\system32> docker run --name nagios1 -p 8888:80 jasonrivers/nagios:latest
Adding password for user nagiosadmin
Started runsvdir, PID is 11
checking permissions for nagios & nagiosgraph
rsyslogd: [origin software="rsyslogd" swVersion="8.2001.0" x-pid="21" x-info="https://www.rsyslog.c
postfix/master[17]: daemon started -- version 3.4.13, configuration /etc/postfix
```

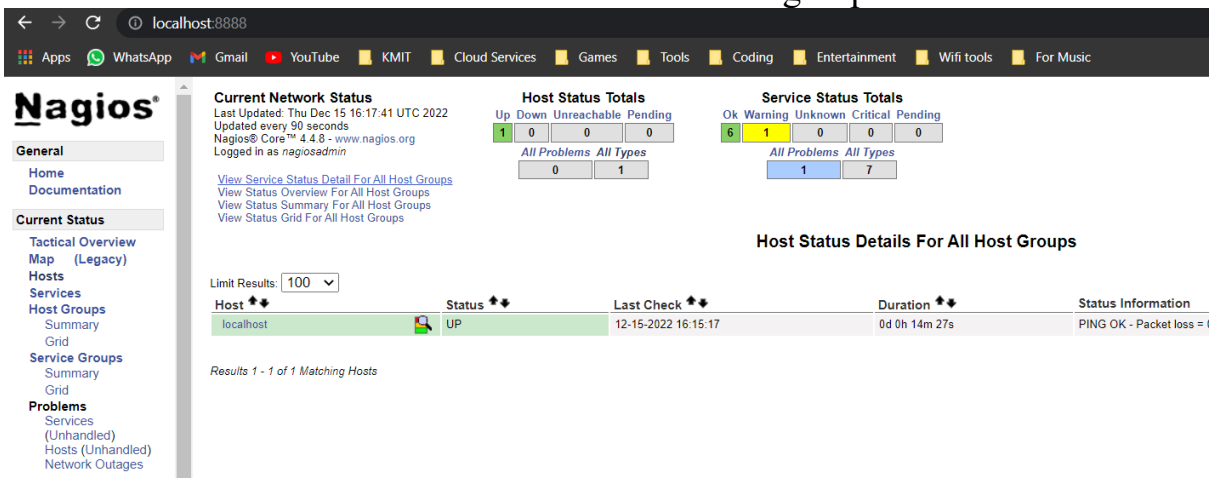
3. To open nagios use localhost:8888 as we have given above. Nagios has default username: nagiosadmin and password: nagios use them to login.



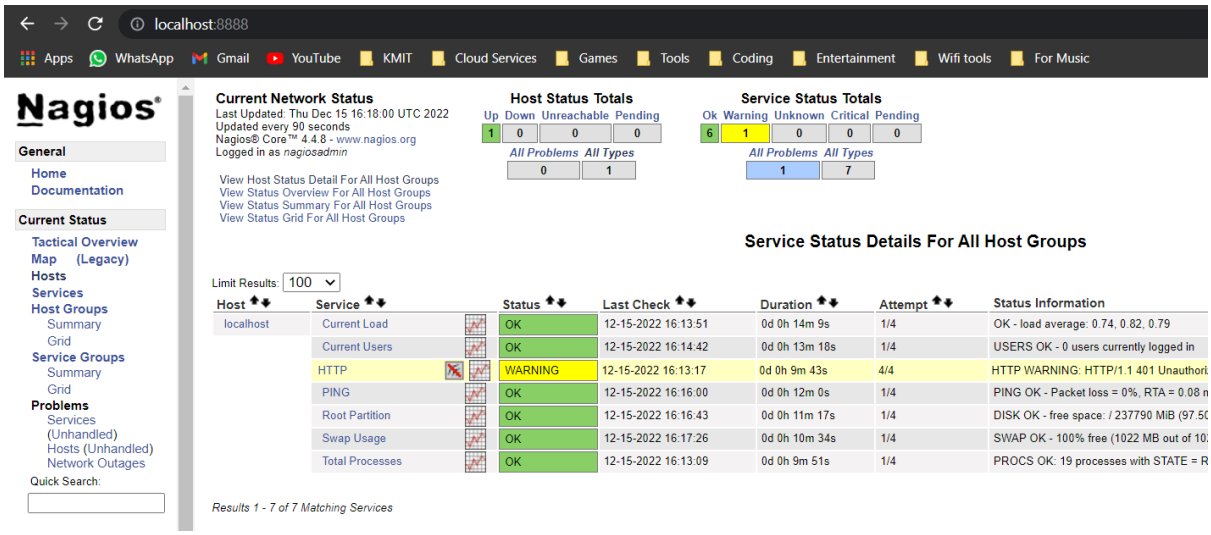
4. Go to hosts.



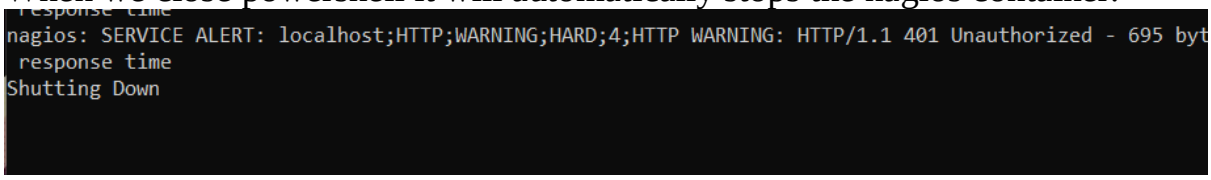
5. We can see “view services status details for all host groups” to see there status.



6. Then we can see the status as below.



7. When we close powershell it will automatically stops the nagios container.



DOCKER NGINX

Nginx (pronounced "engine-x") is an open source reverse proxy server for HTTP, HTTPS, SMTP, POP3, and IMAP protocols, as well as a load balancer, HTTP cache, and a web server (origin server). The nginx project started with a strong focus on high concurrency, high performance and low memory usage. It is licensed under the 2-clause BSD-like license and it runs on Linux, BSD variants, Mac OS X, Solaris, AIX, HP-UX, as well as on other *nix flavors. It also has a proof of concept port for Microsoft Windows.

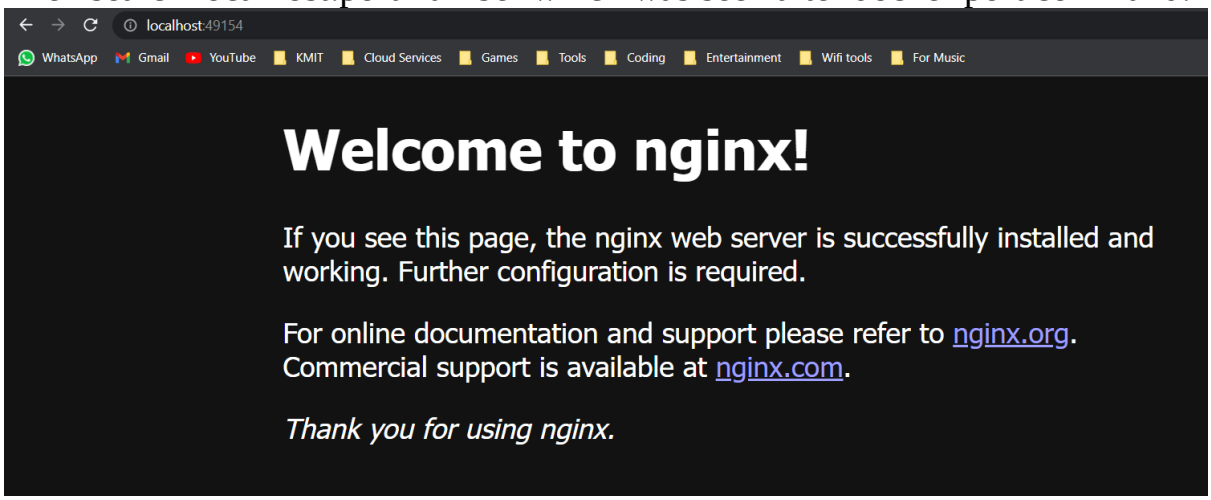
1. docker run –name <containername> -d -P <imagename> (To pull and run nginx without any port number).

```
PS C:\WINDOWS\system32> docker run --name appserver -d -P nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
025c56f98b67: Already exists
ec0f5d052824: Pull complete
cc9fb8360807: Pull complete
defc9ba04d7c: Pull complete
885556963dad: Pull complete
f12443e5c9f7: Pull complete
Digest: sha256:75263be7e5846fc69cb6c42553ff9c93d653d769b94917dbda71d42d3f3c00d3
Status: Downloaded newer image for nginx:latest
d6edc62b28f6f359538331ac59617d24bd2bff2f94be6d96ad58cbdeb5f7010d
PS C:\WINDOWS\system32>
```

2. docker port <containername> (This will give port number where the container is running).

```
PS C:\WINDOWS\system32> docker port appserver
80/tcp -> 0.0.0.0:49154
PS C:\WINDOWS\system32>
```

3. Then search localhost:port number which was seen after docker port command.



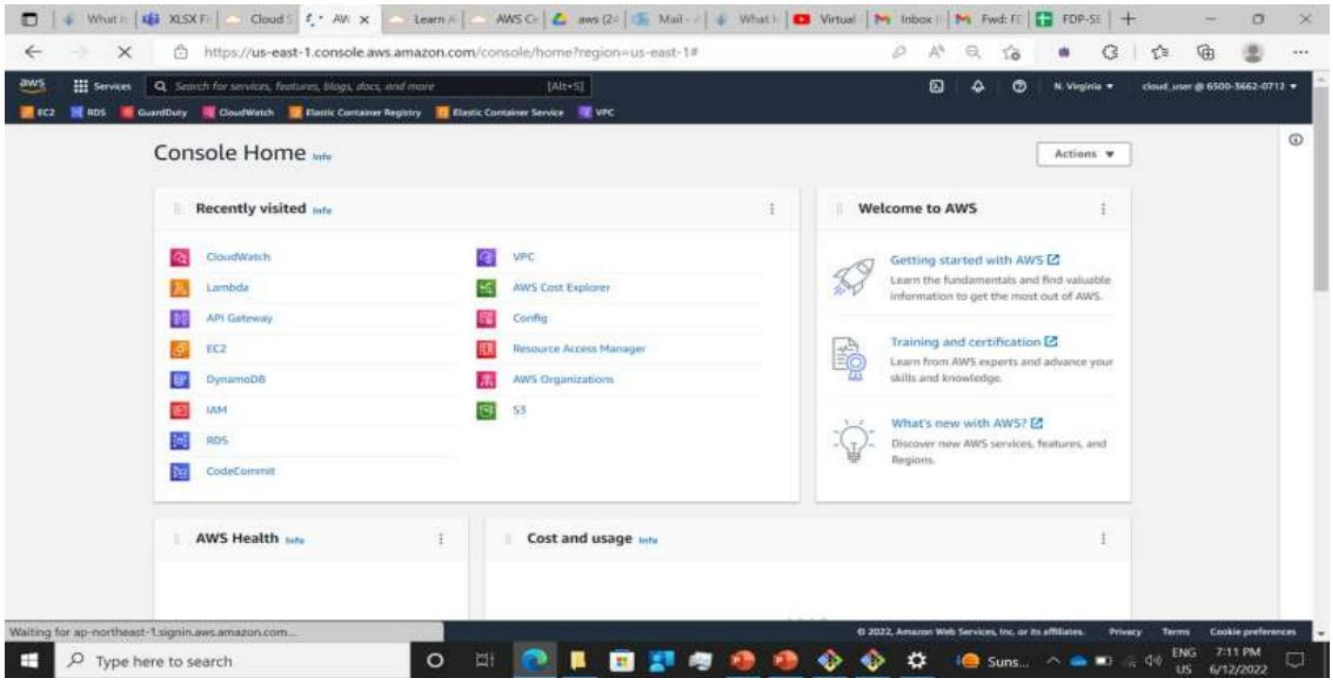
4. docker stop <containername> (To stop the container use).

```
PS C:\WINDOWS\system32> docker stop appserver
appserver
PS C:\WINDOWS\system32>
```

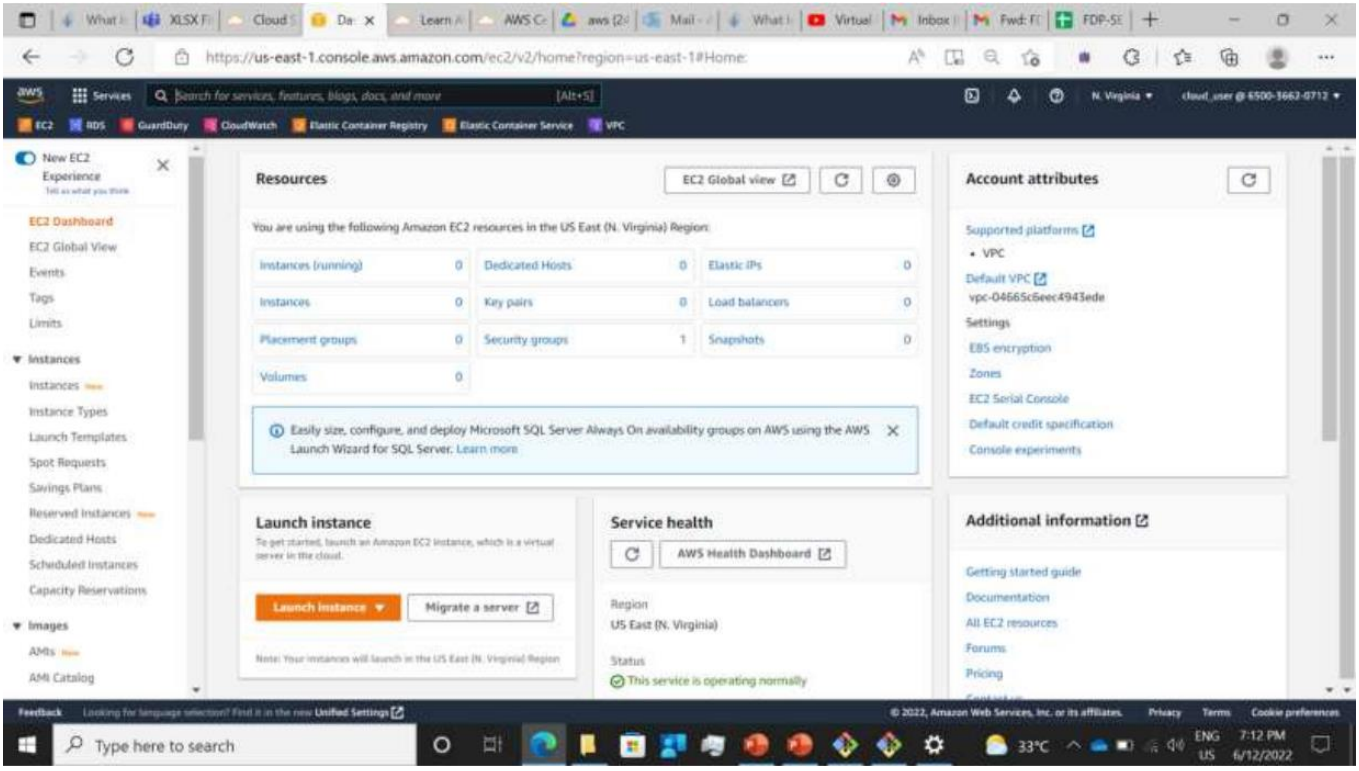
6.WORKING WITH AWS

Creating EC2 instance with amazon linux2

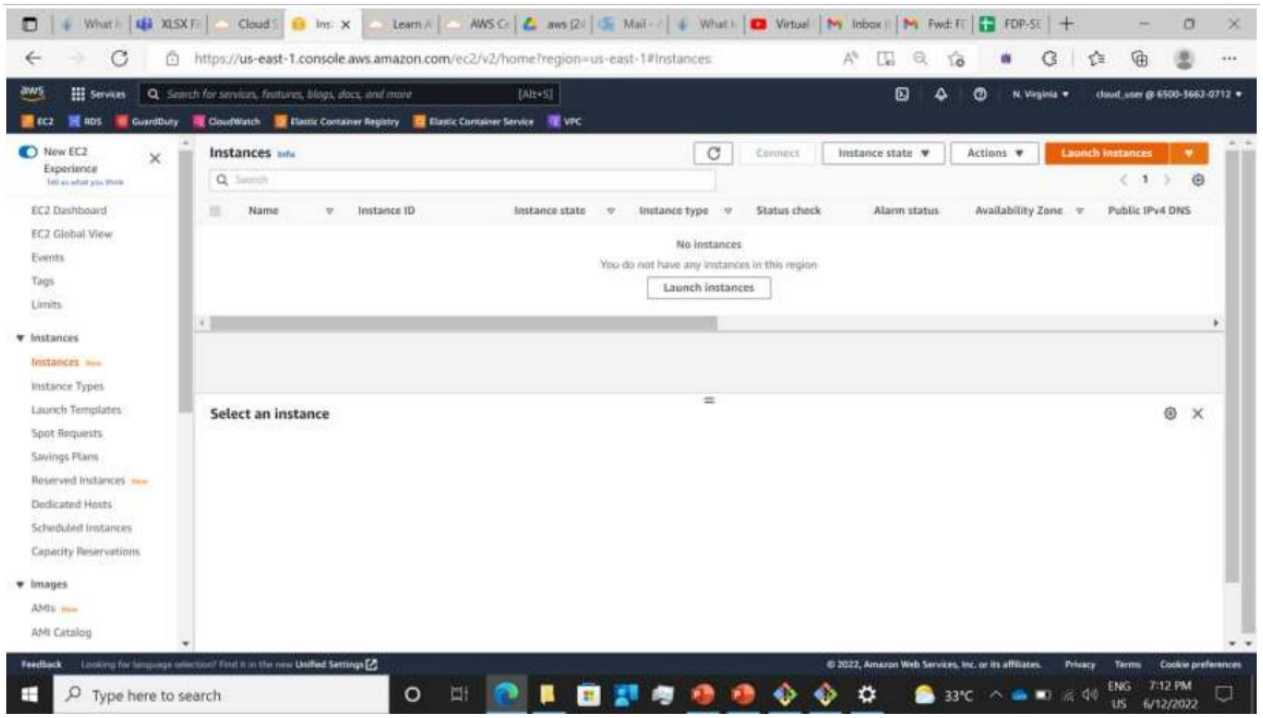
1. In AWS services select EC2 services:



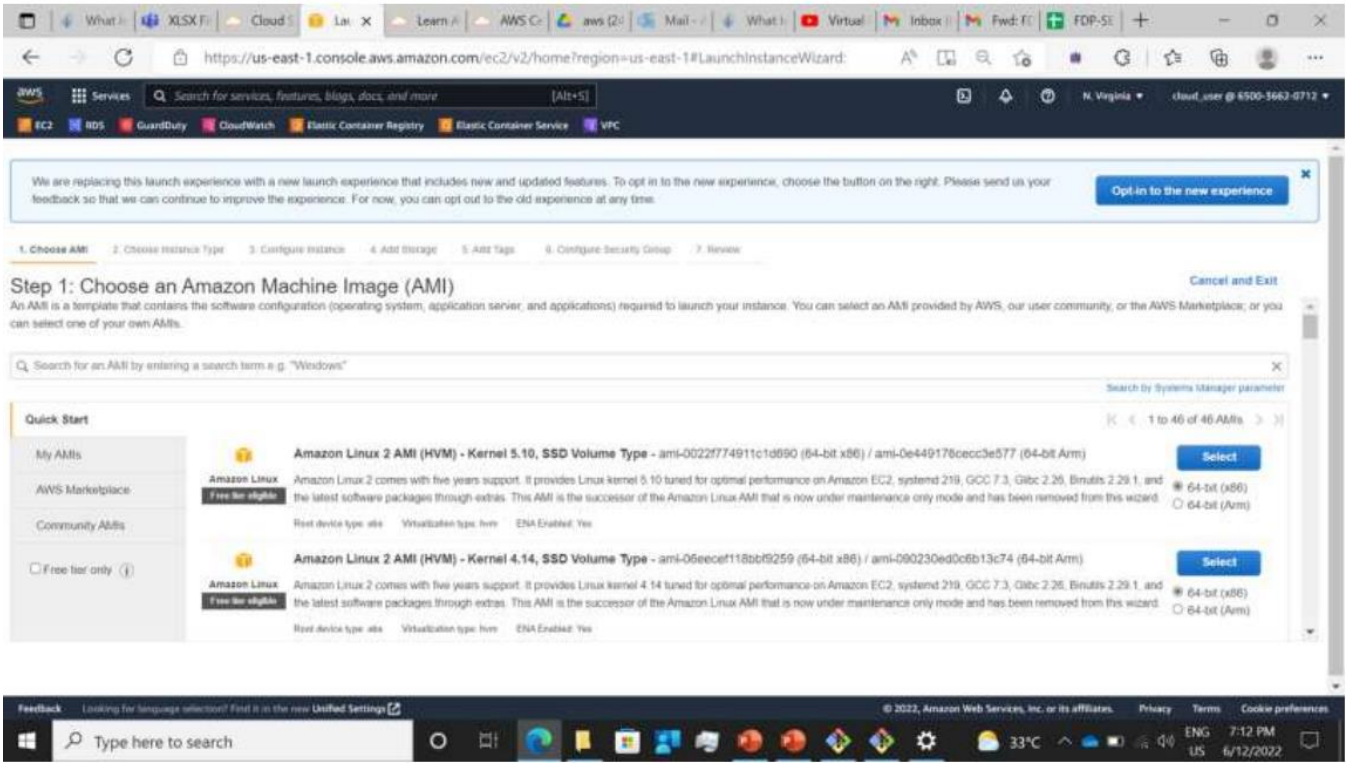
2. In EC2 Dashboard ,click on instance



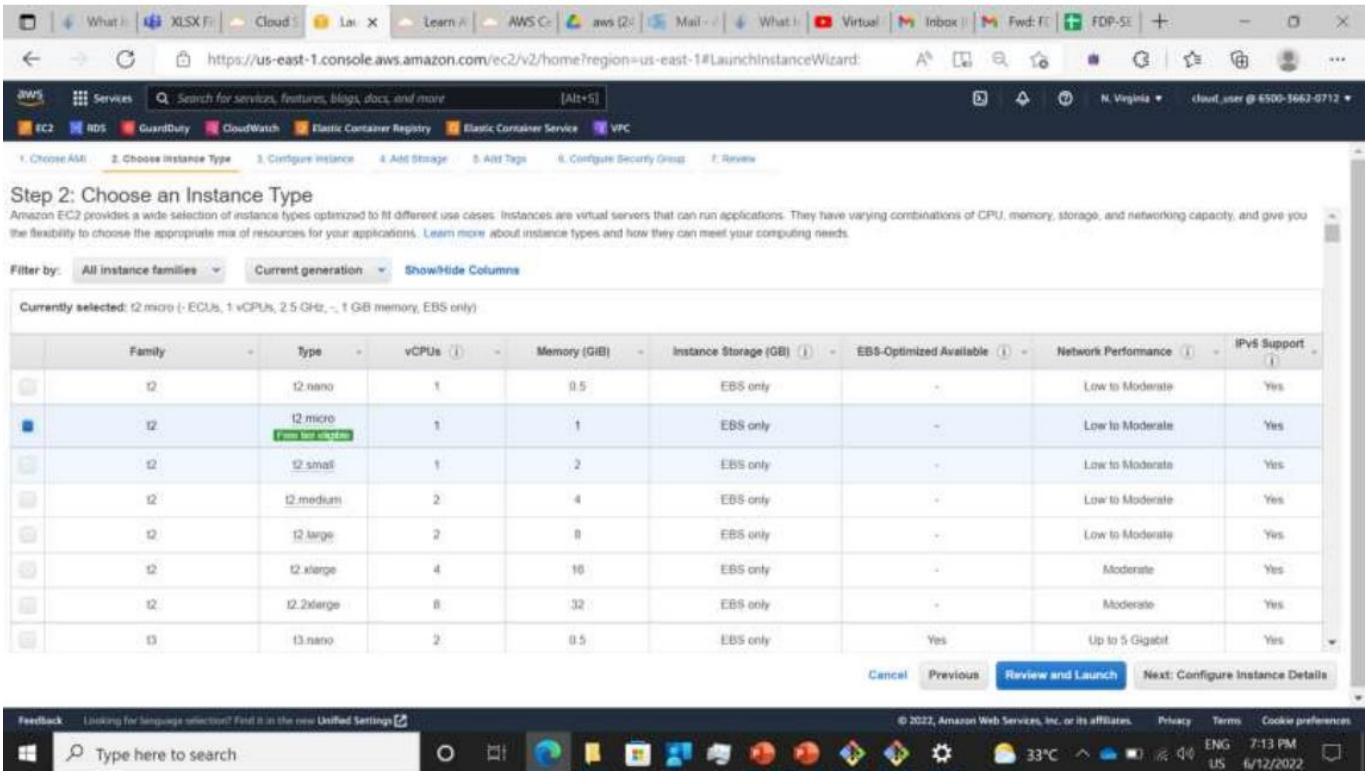
3. Now click on launch instance



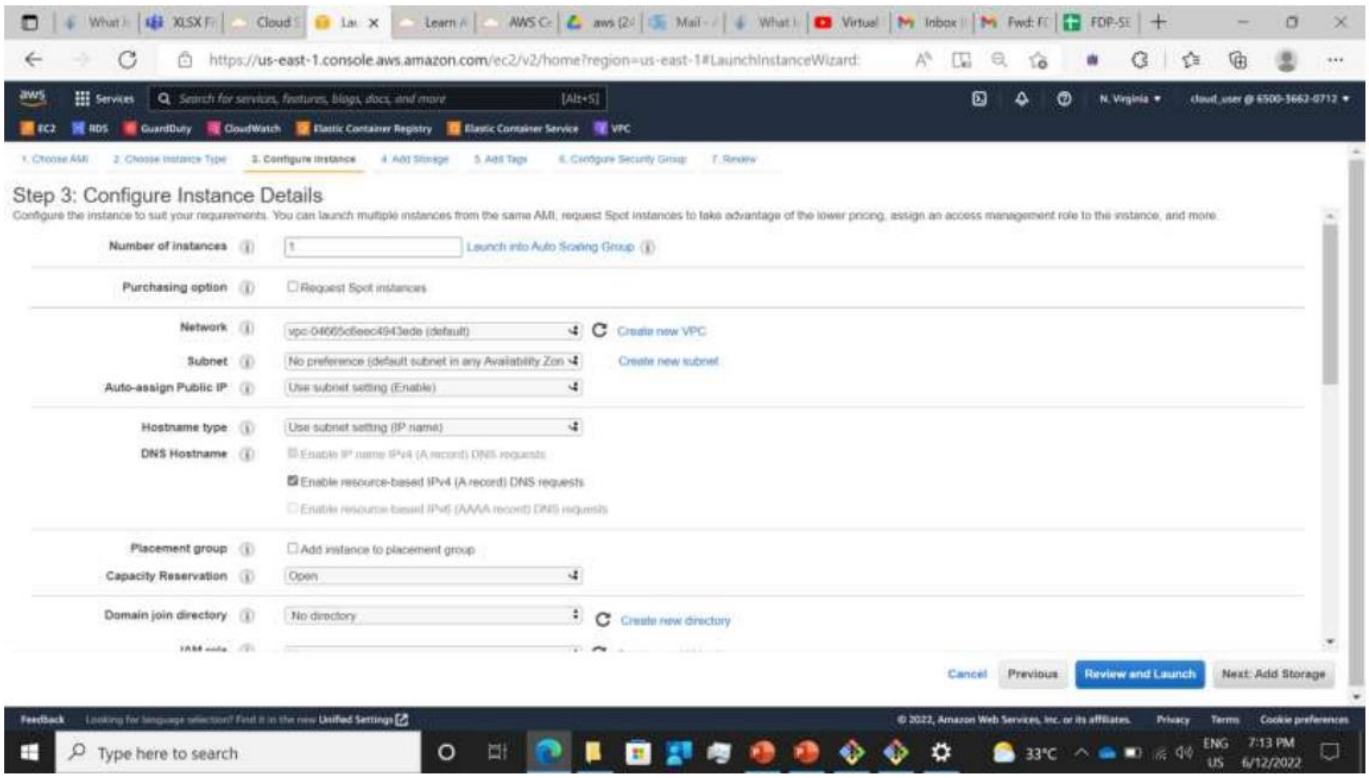
4. Now select the amazon machine image (AMI) ,here we are selecting the image as Amazon Linux 2 AMI



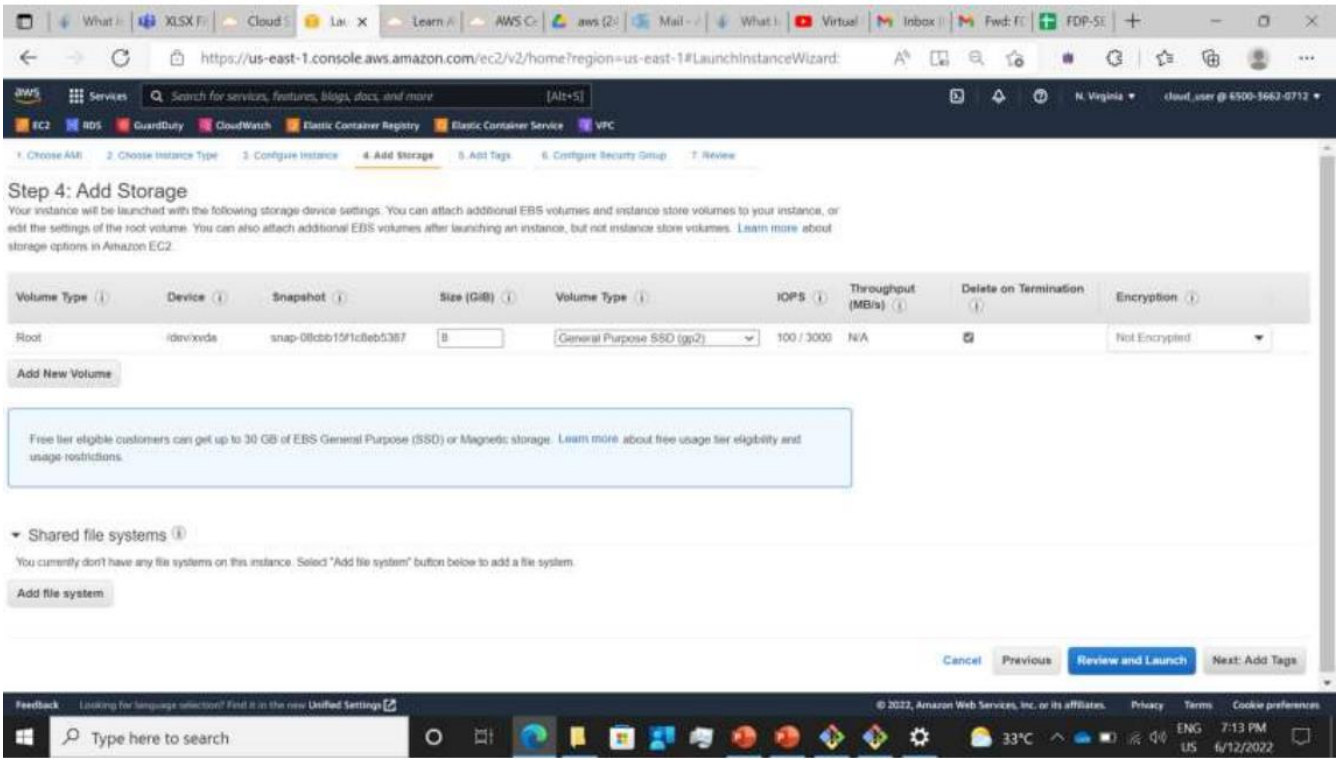
5. Choose the number of instances you want



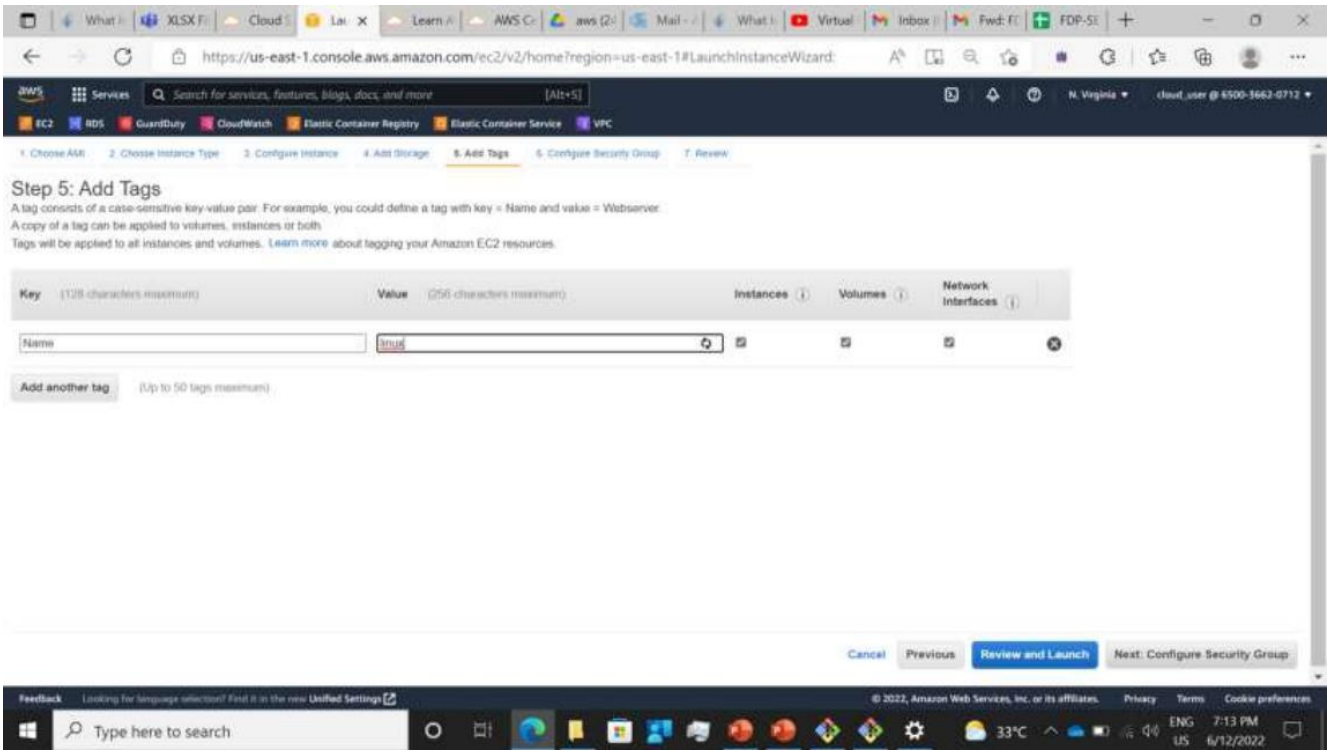
6. Now next is configure instance



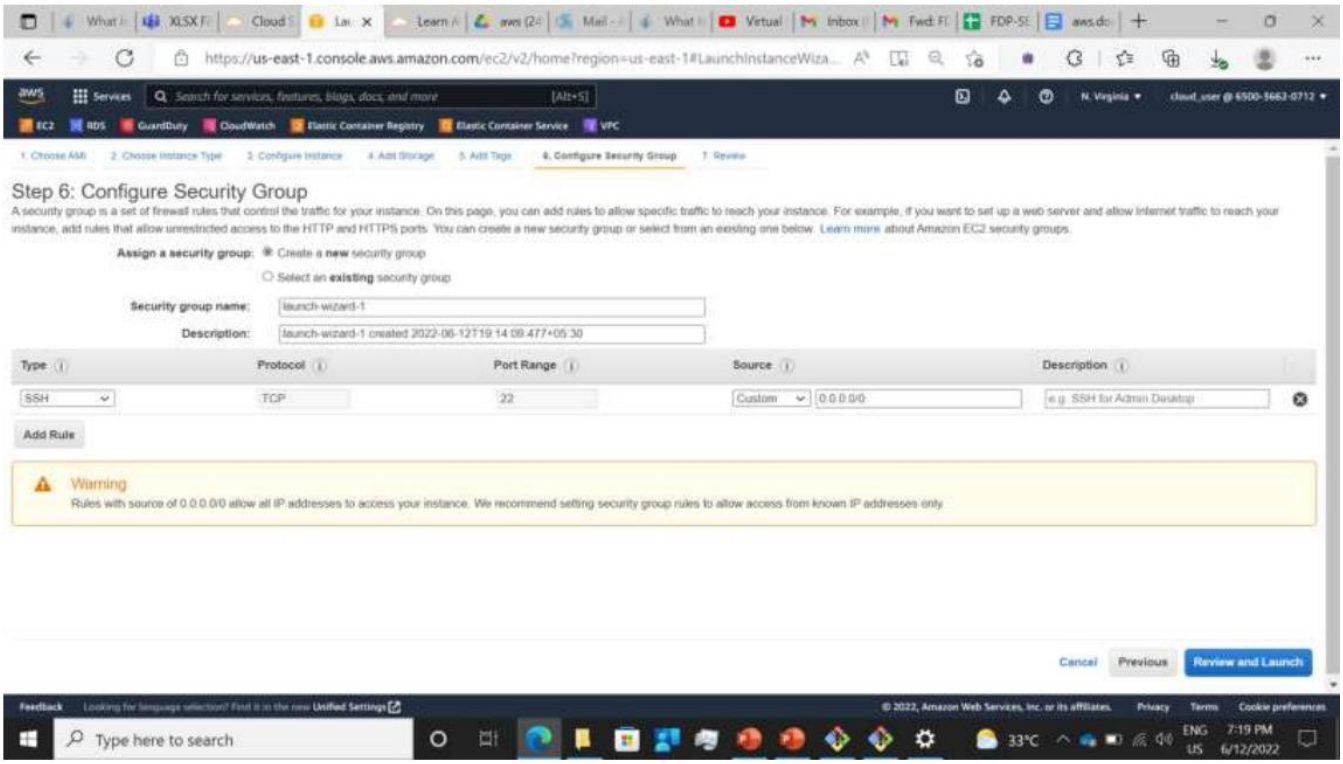
7. Here give the details of storage that you want to give



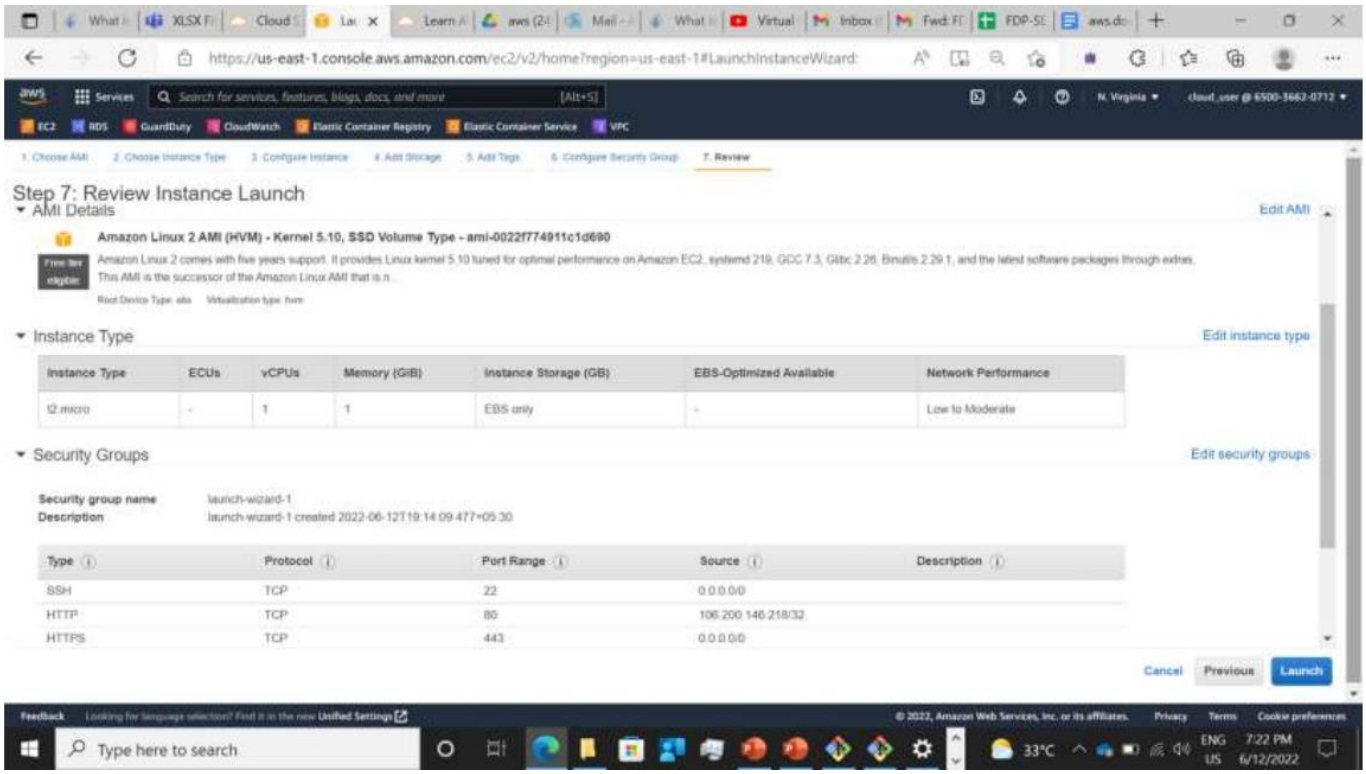
8. Here we give name to the instance



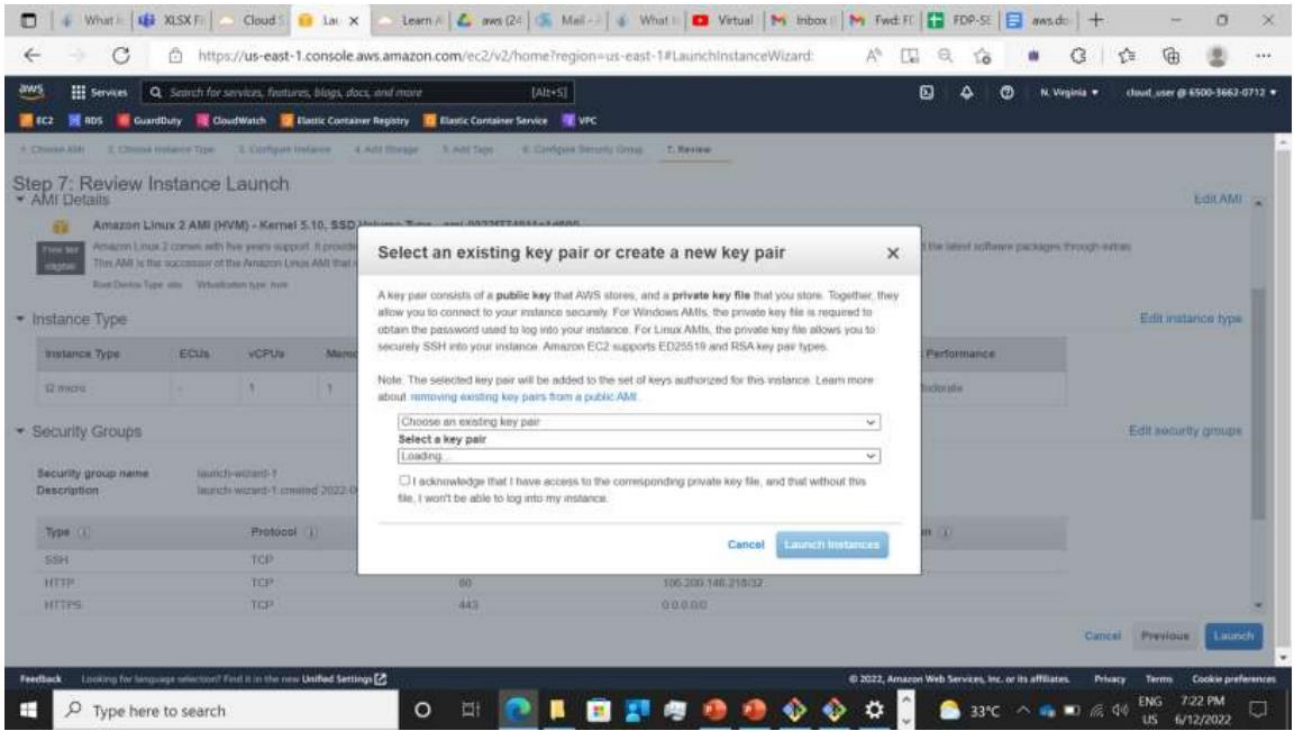
9. Here comes with security group ,here we can allow the traffic of incoming ,and we can give access to http service to get in but will give those permission in the inbound security



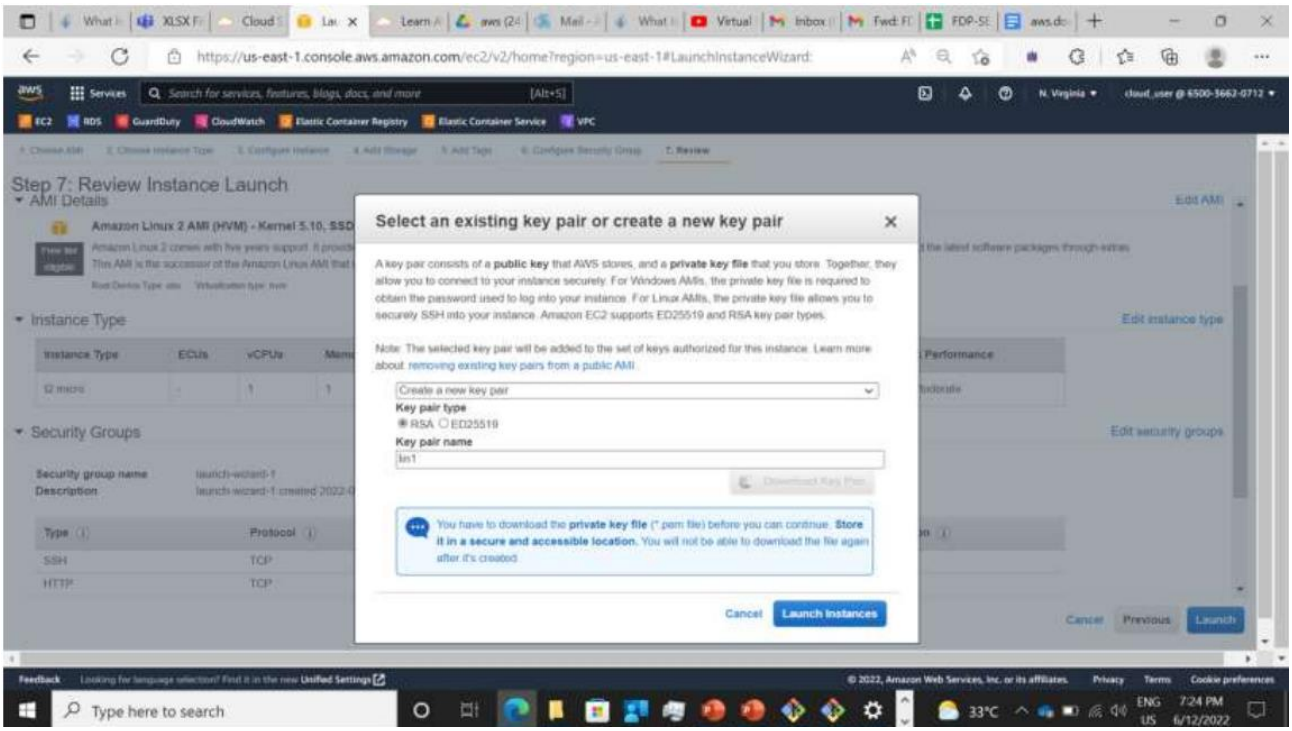
10. Now we can view the instance of launched



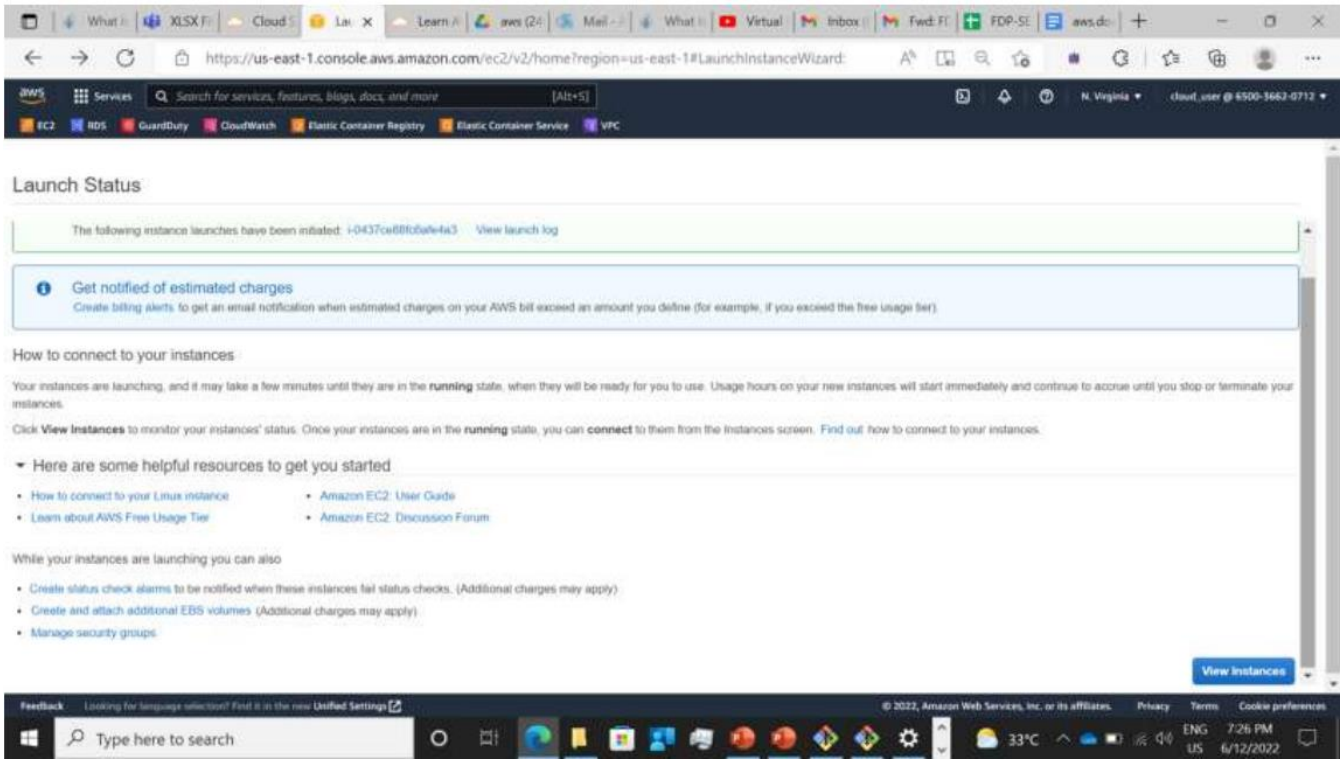
11. Here we create the key pairs or use the existing one ,here will create by keypair



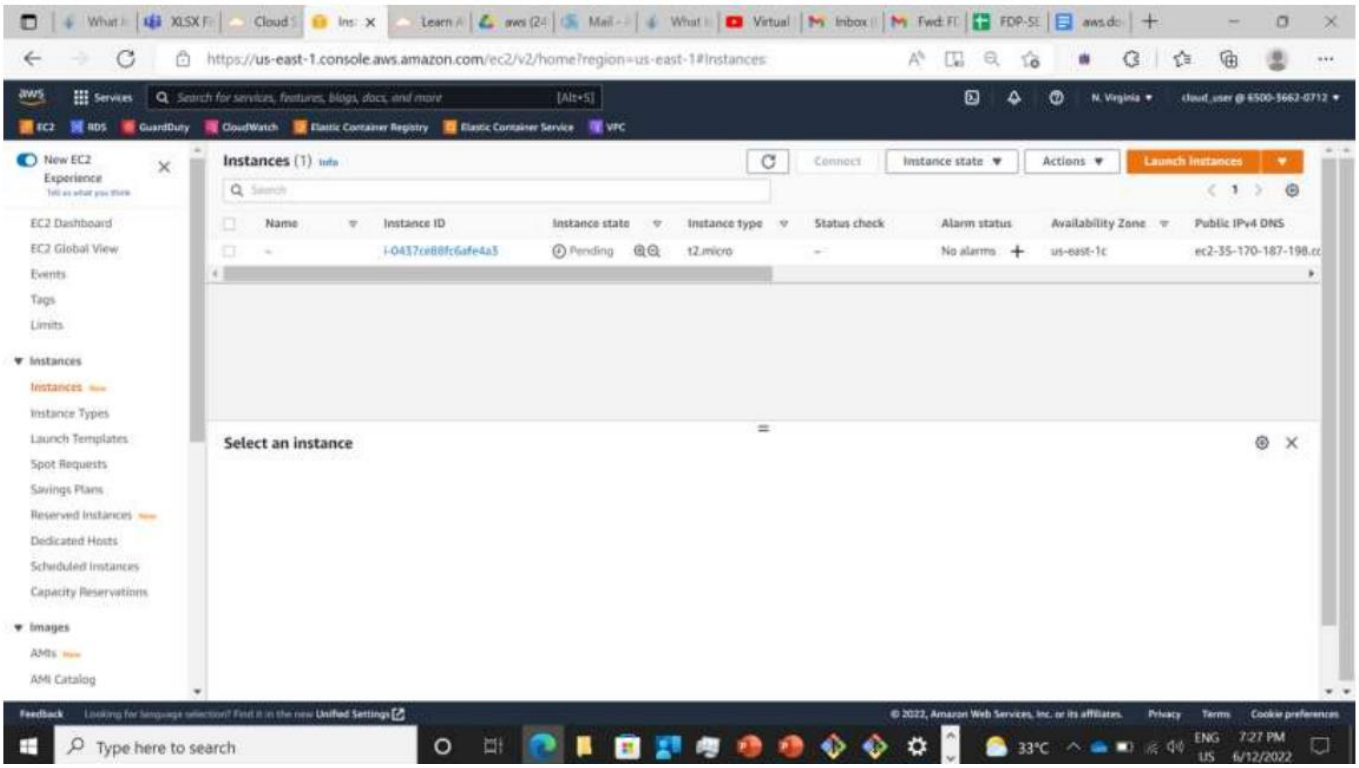
12. The keypair name is lin1



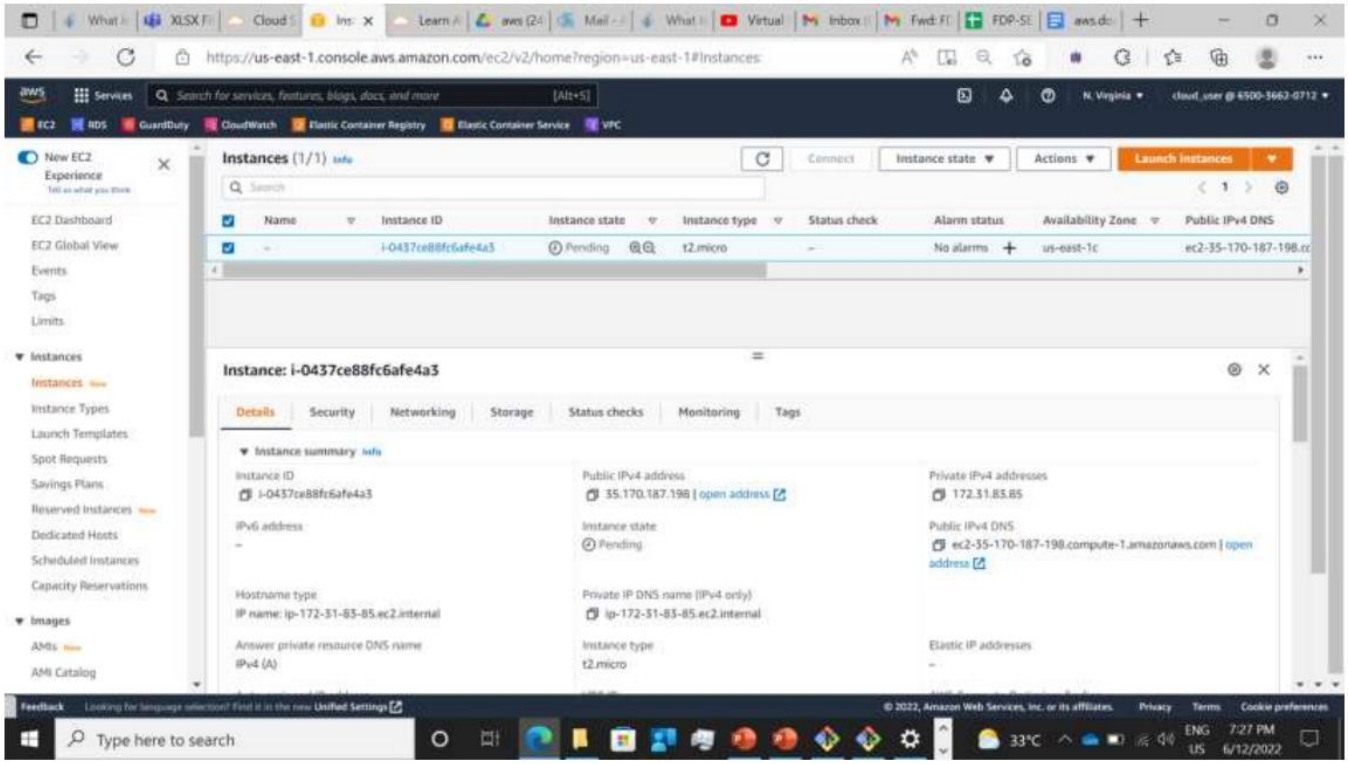
13. After launching the instance ,we can view the instance



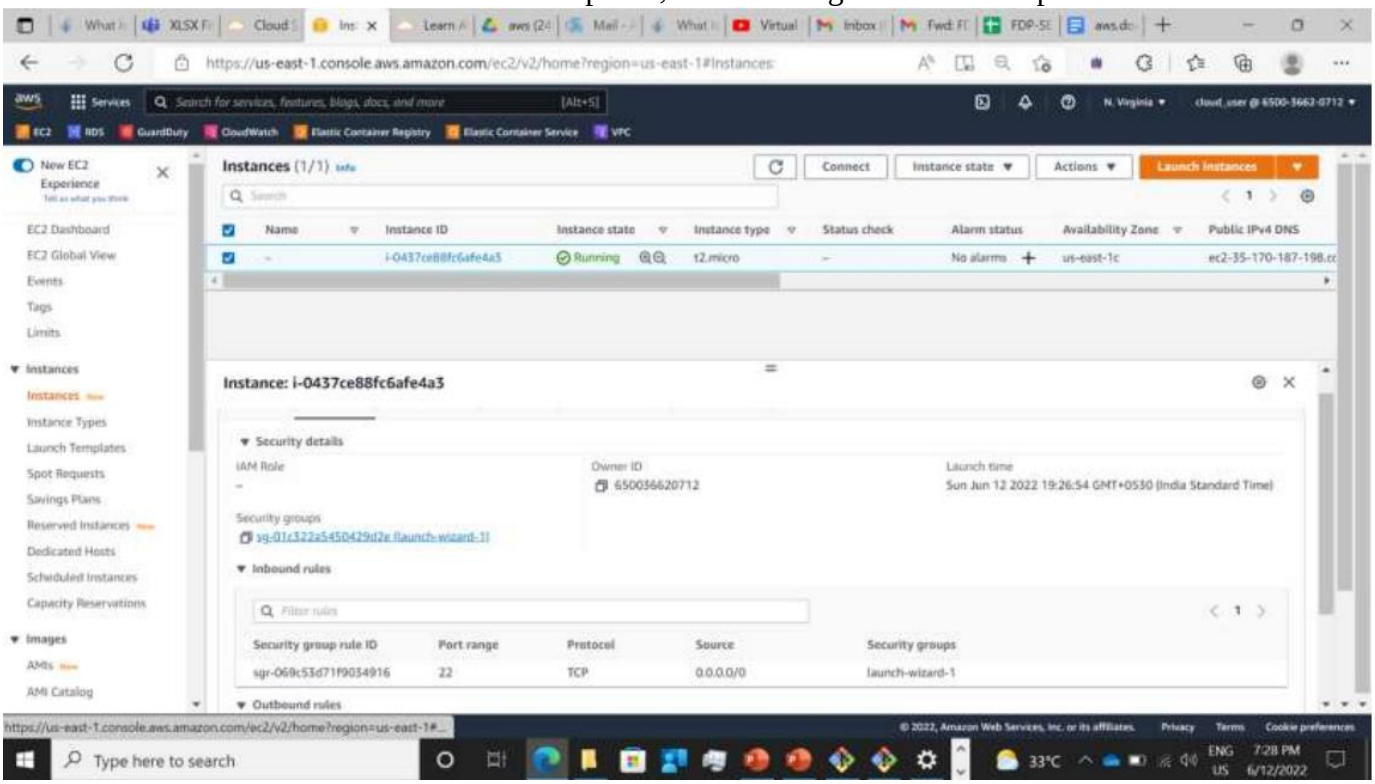
14. Here we can see the instance have been launched



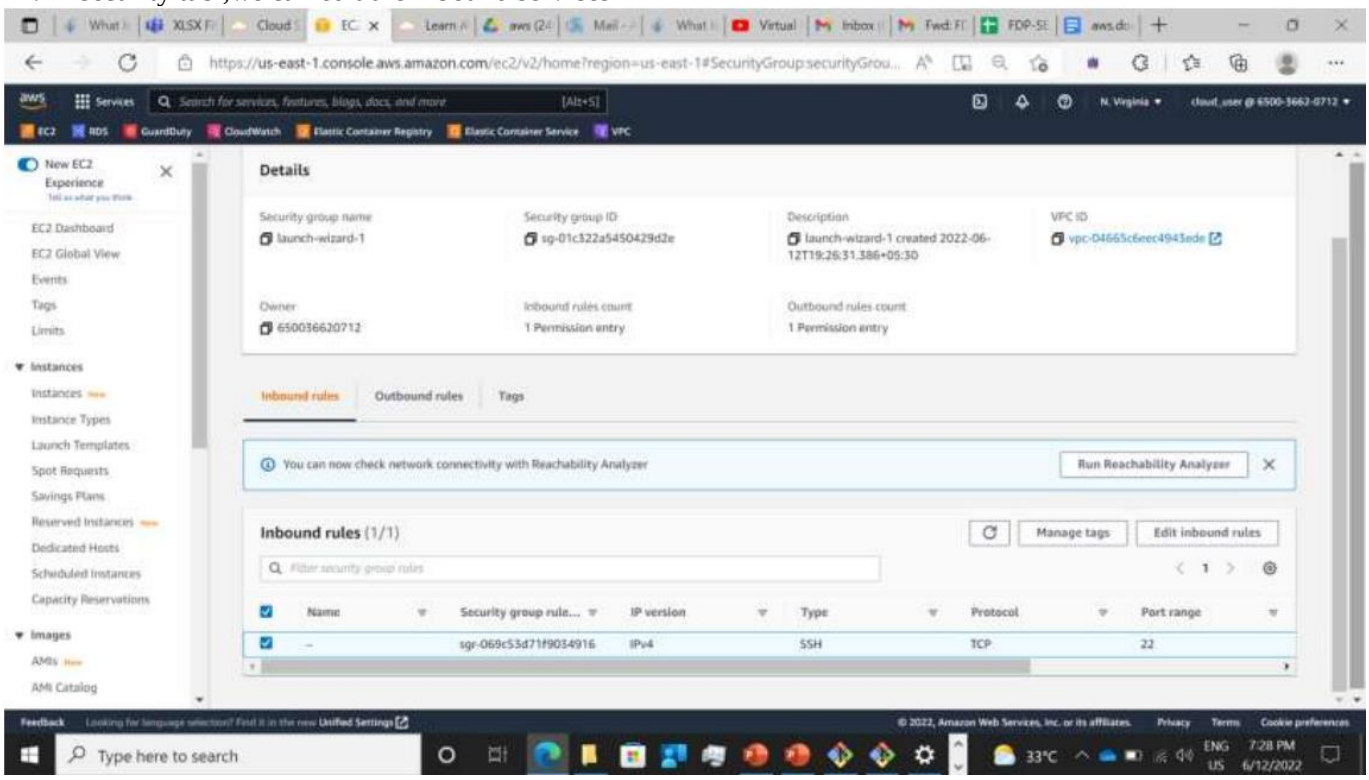
15. When you click on instance we see the details of the instance



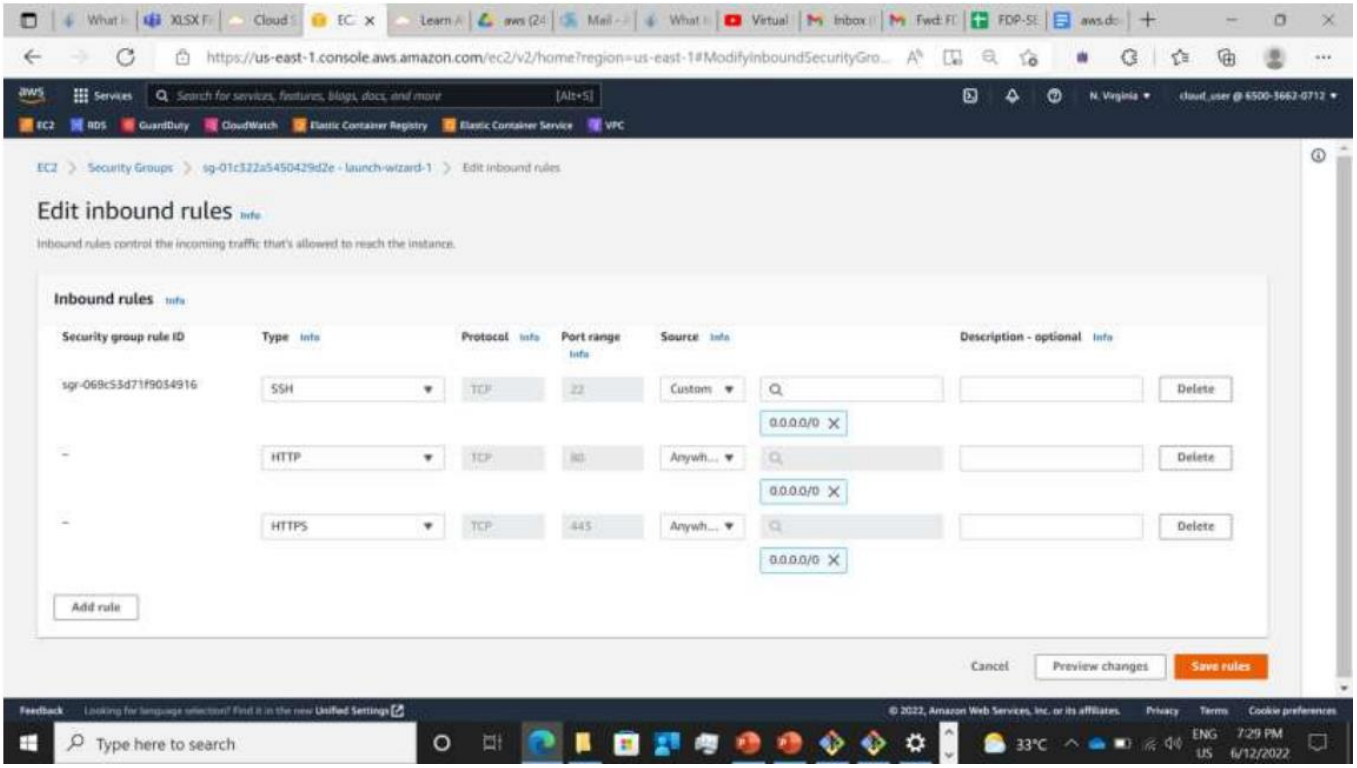
16. Here it take the minute time for 2/2 checkpoint ,then we can go for connect option



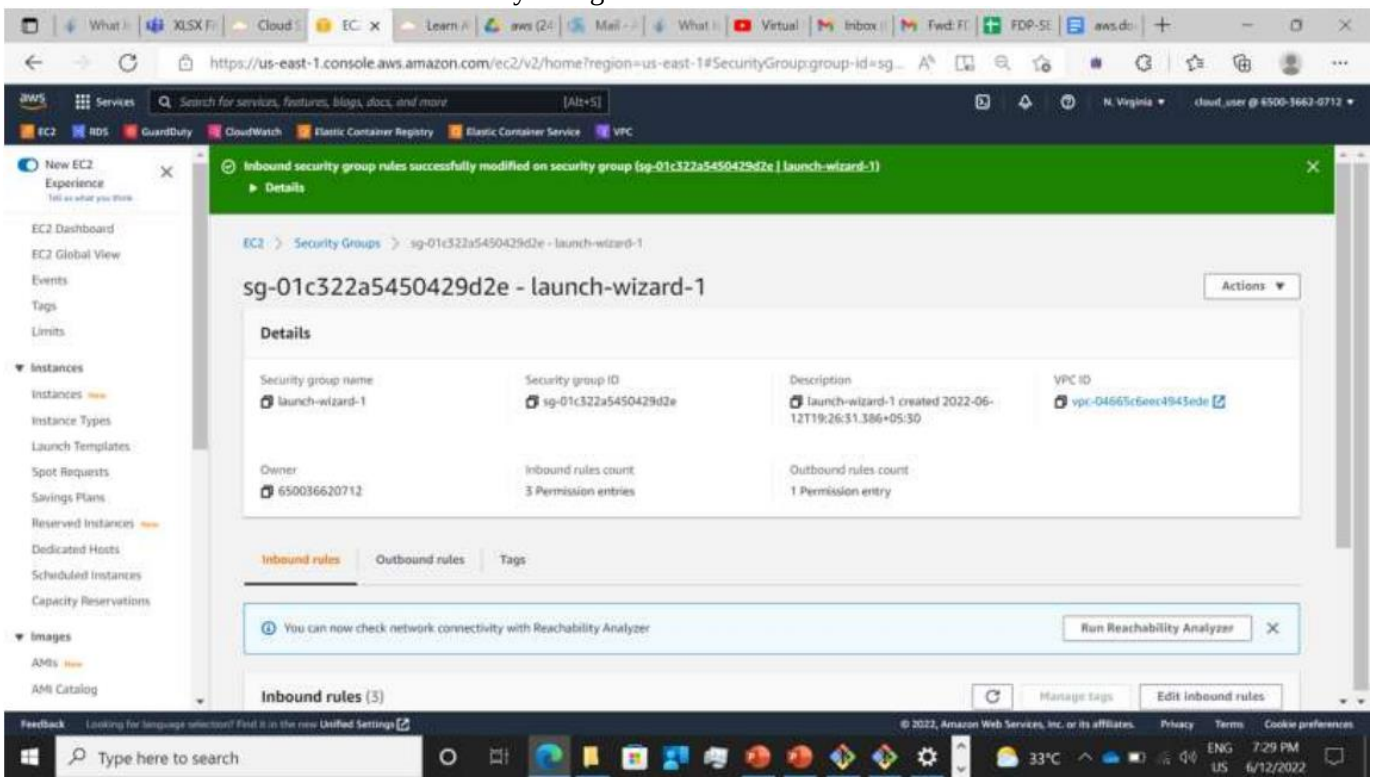
17. In security tab ,we can edit the inbound services



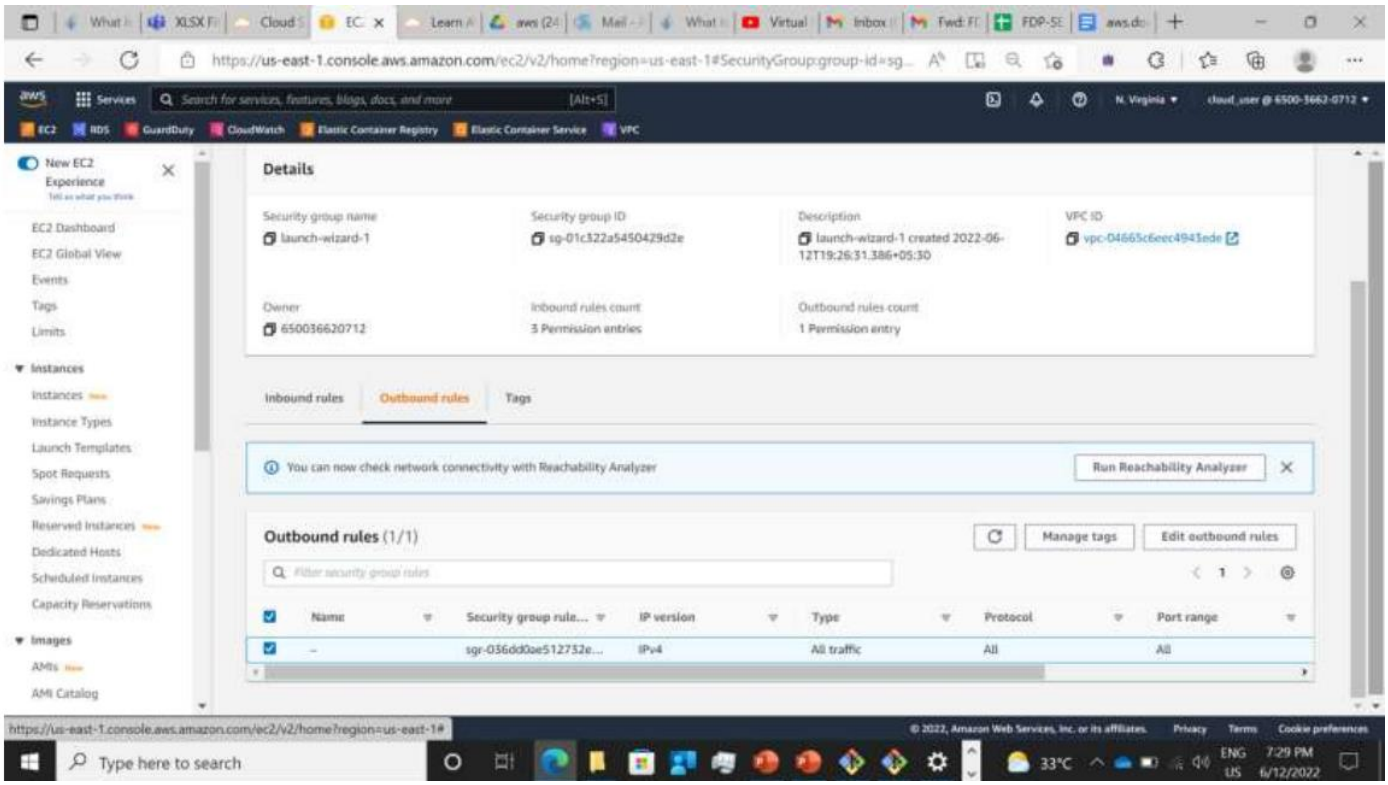
18. In inbound services ,click on add rules to add the http,hhtps ,then click on save rules



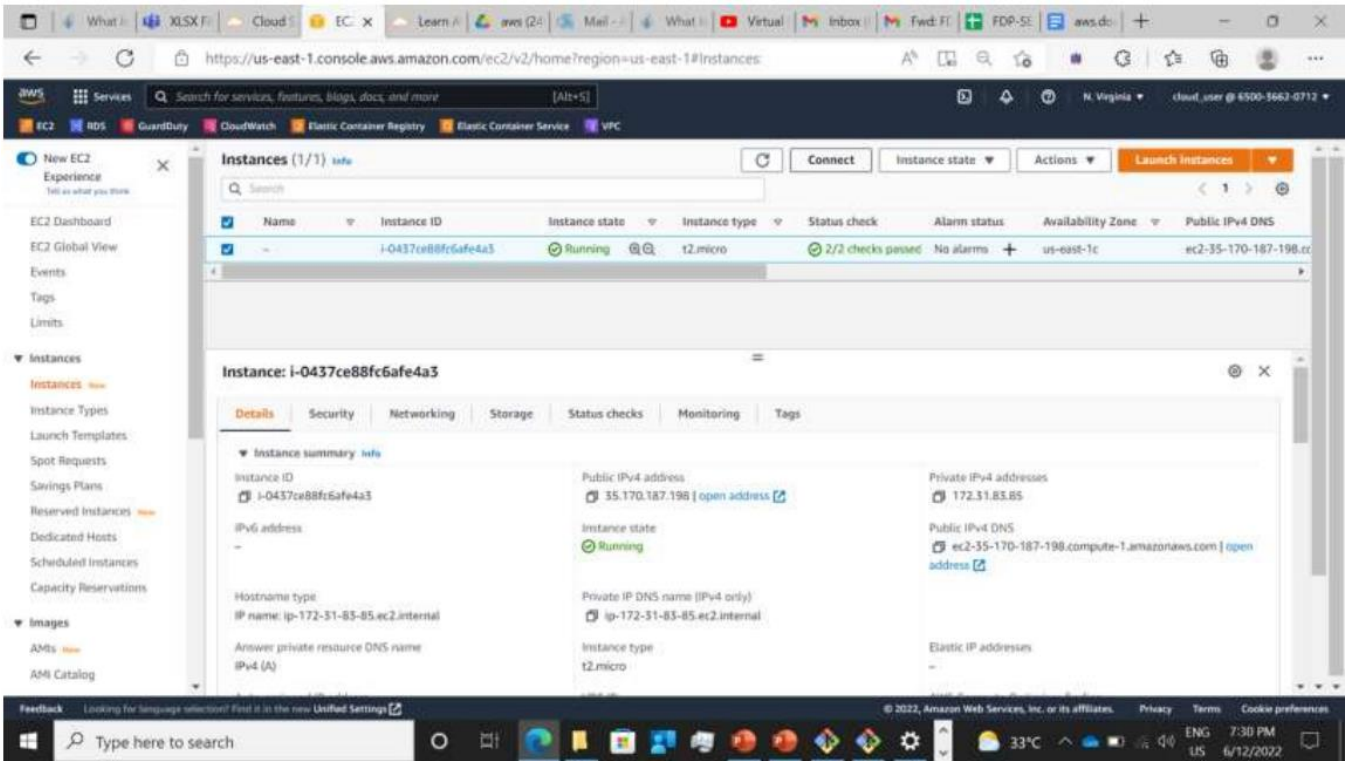
19. We see the that inbound has successfully changed



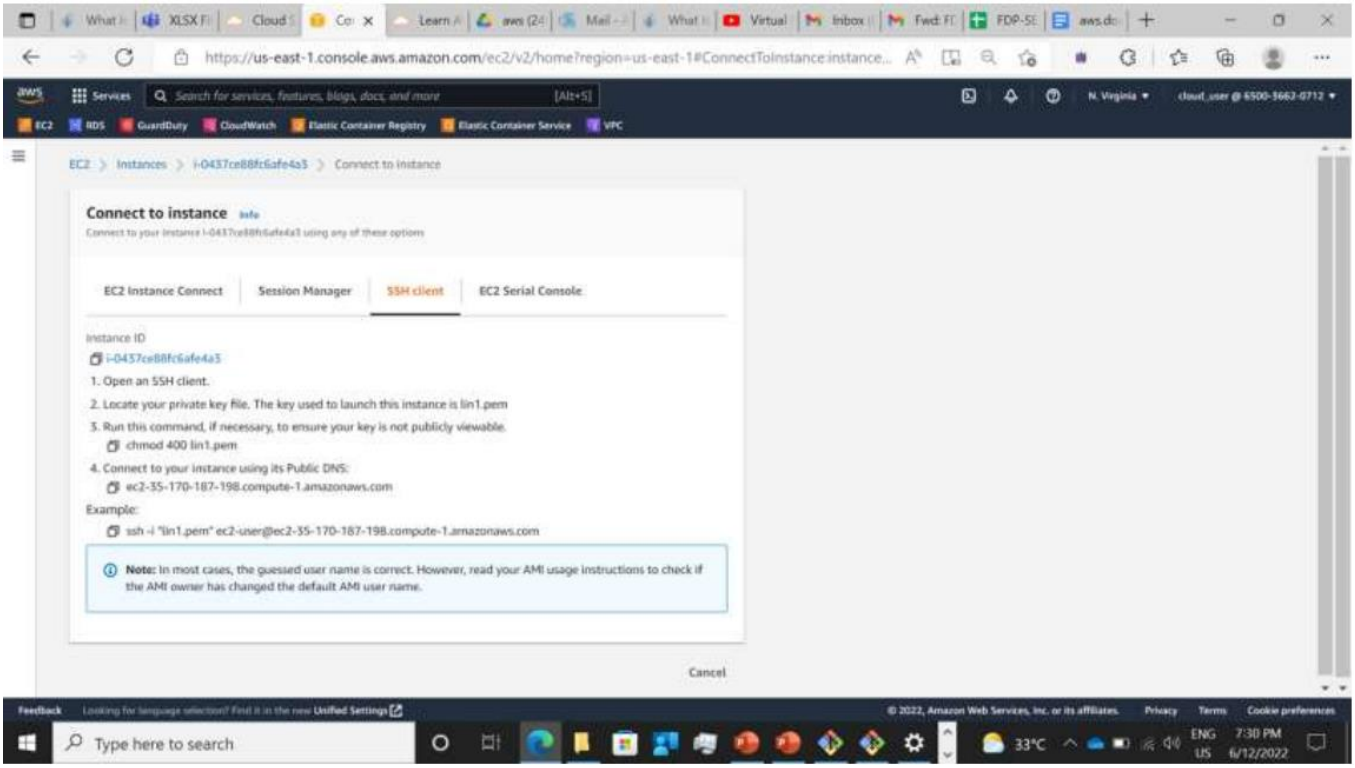
20. In outbound rules we keep as default



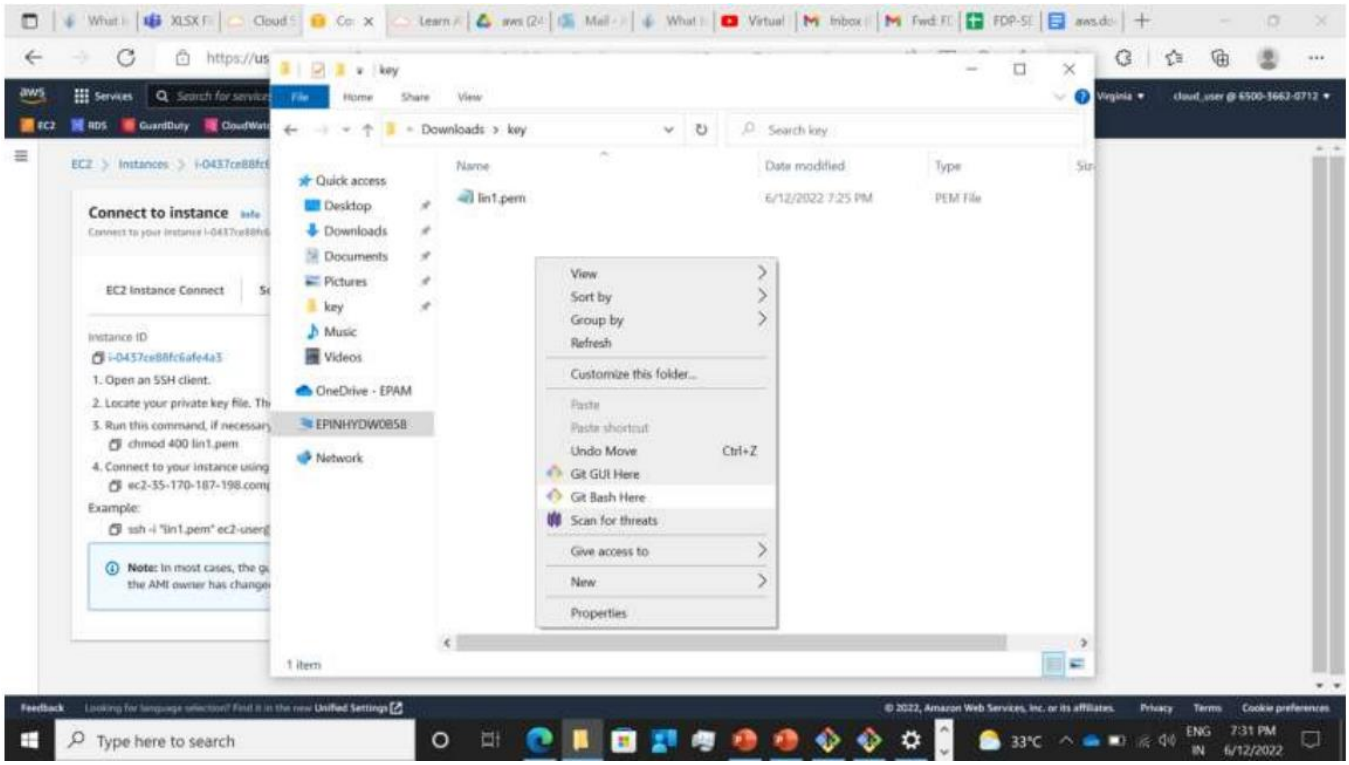
21. Now we see that 2/2check point has ,now we can connect it



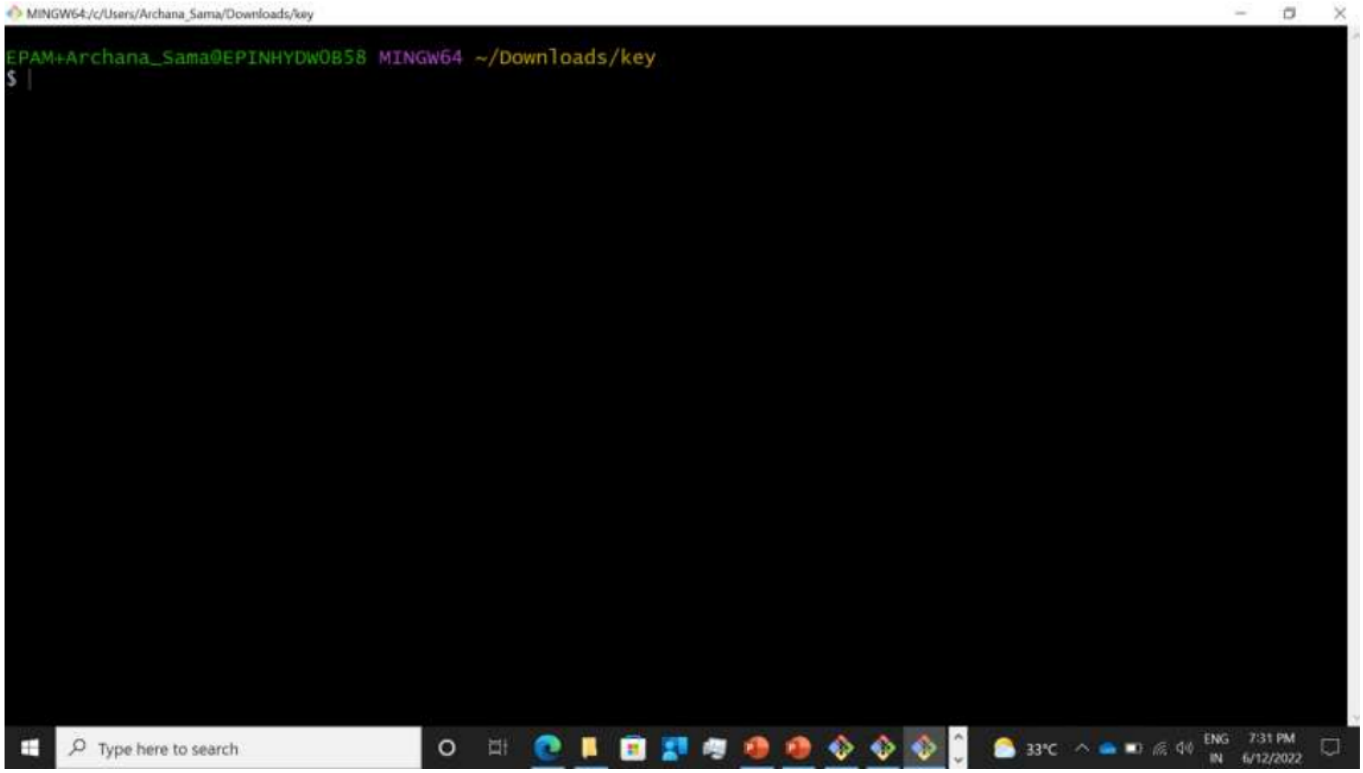
22. Now we see the SSH client details



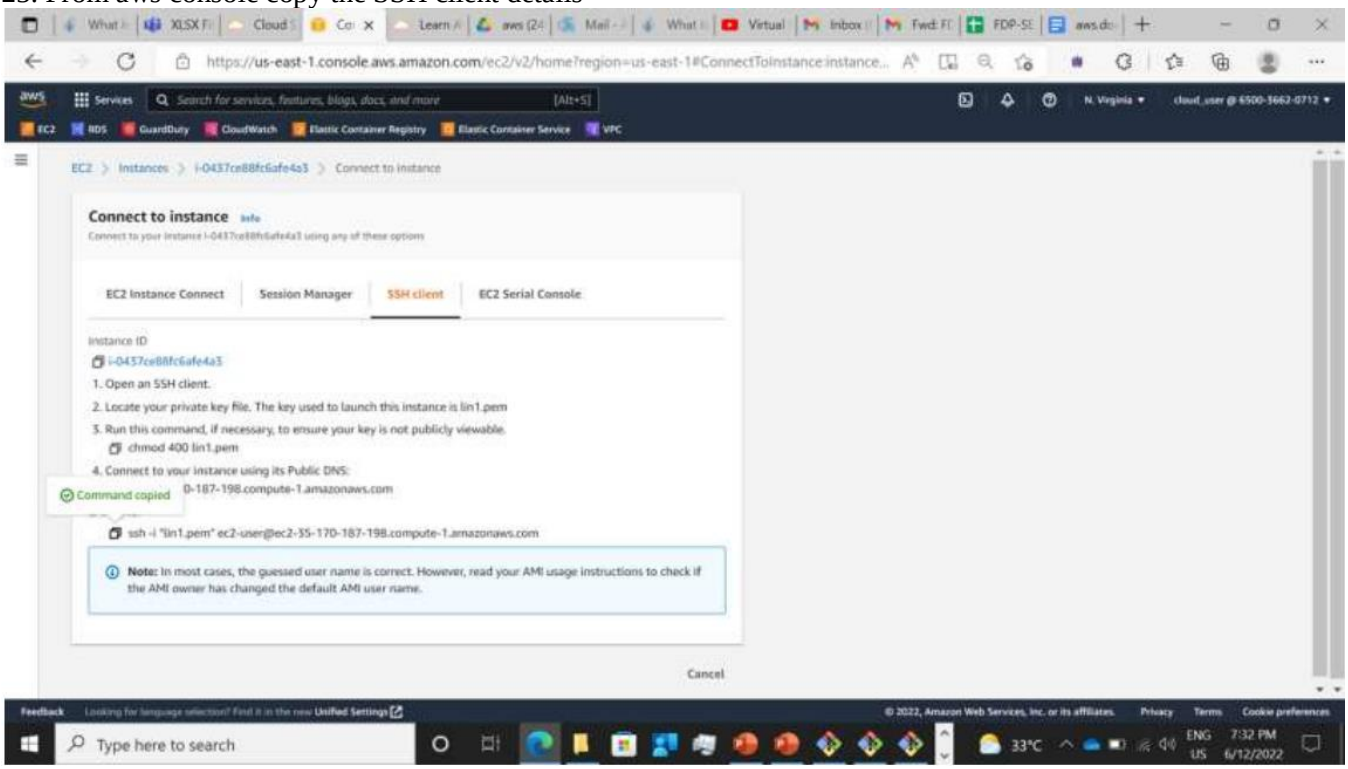
23. To connect to the terminal, we can use putty or gitbash ->here we will use gitbash ->go to the download of key , right click and select the gitbash here



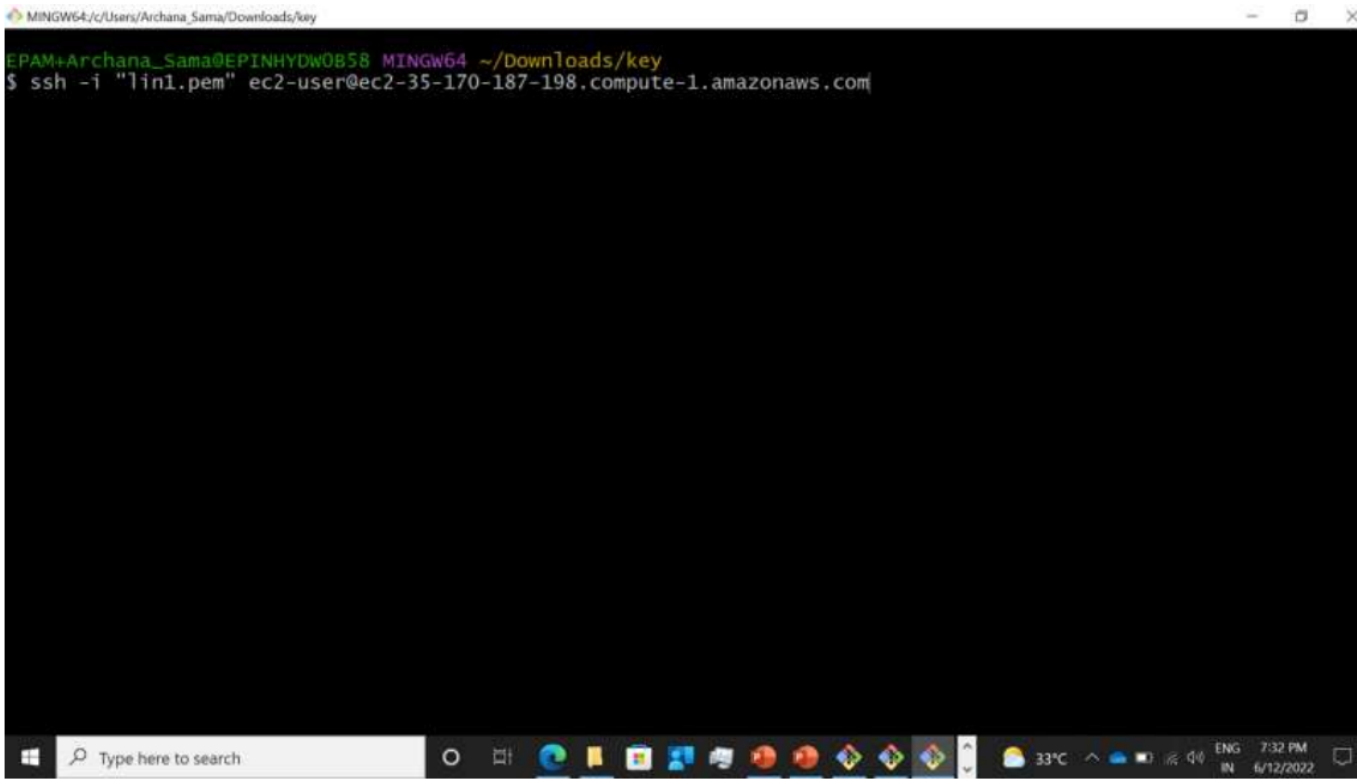
24. Here the console of gitbash has been opened



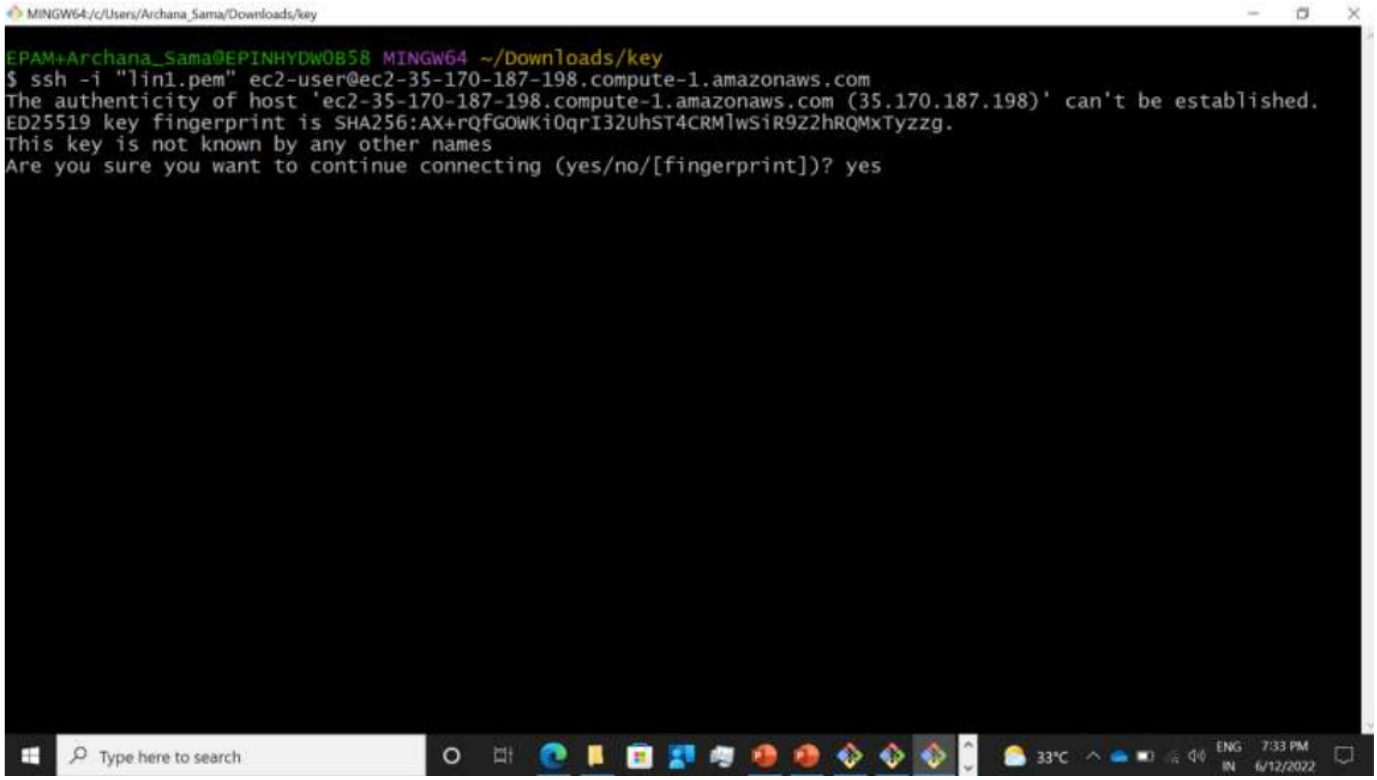
25. From aws console copy the SSH client details



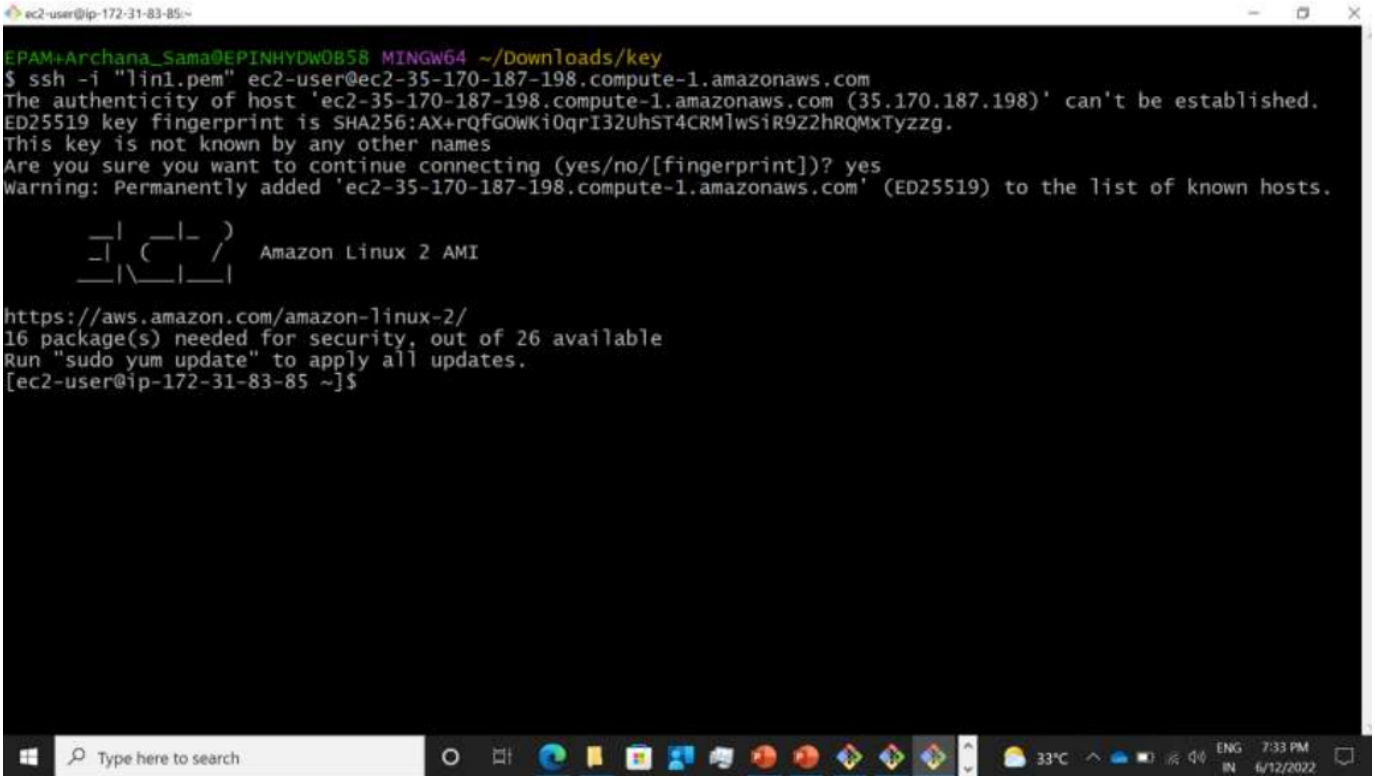
26. Now copy the command here



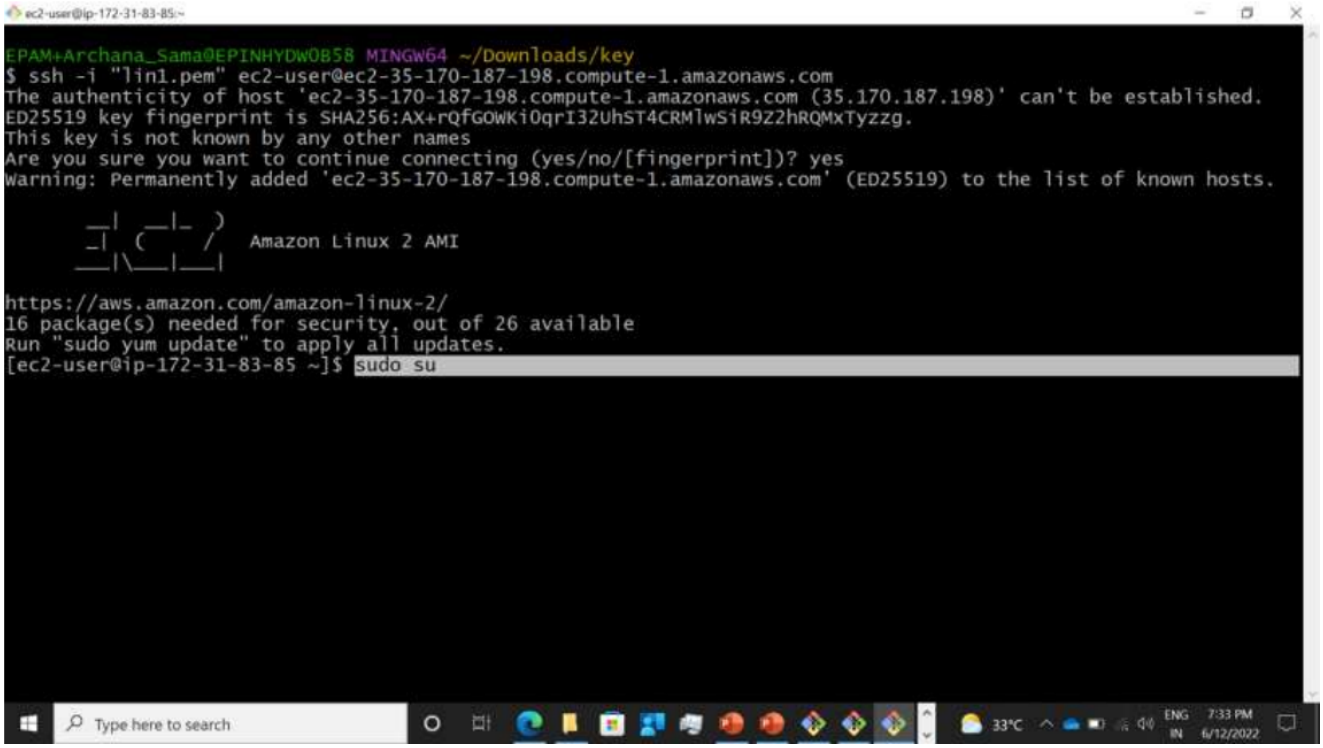
27. It asks that you want connect it ---type yes



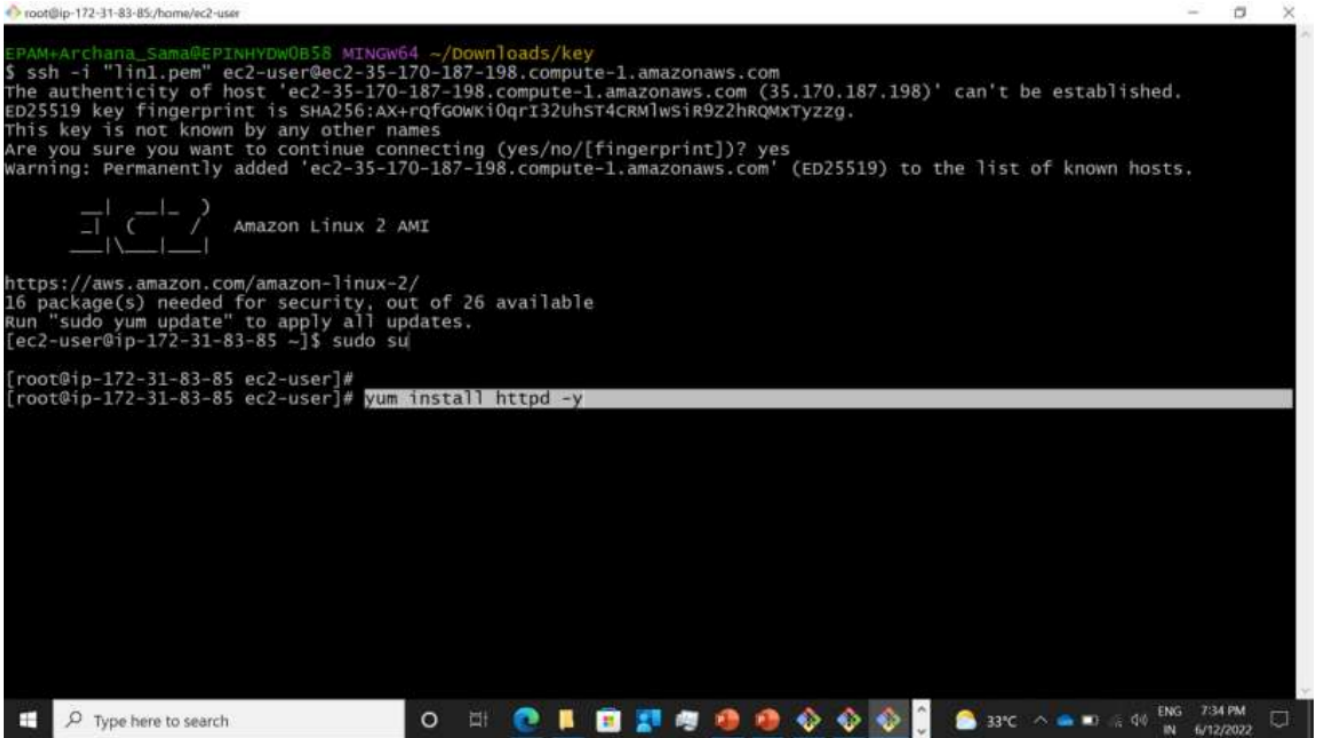
28. We see that we connect to the remote Linux server



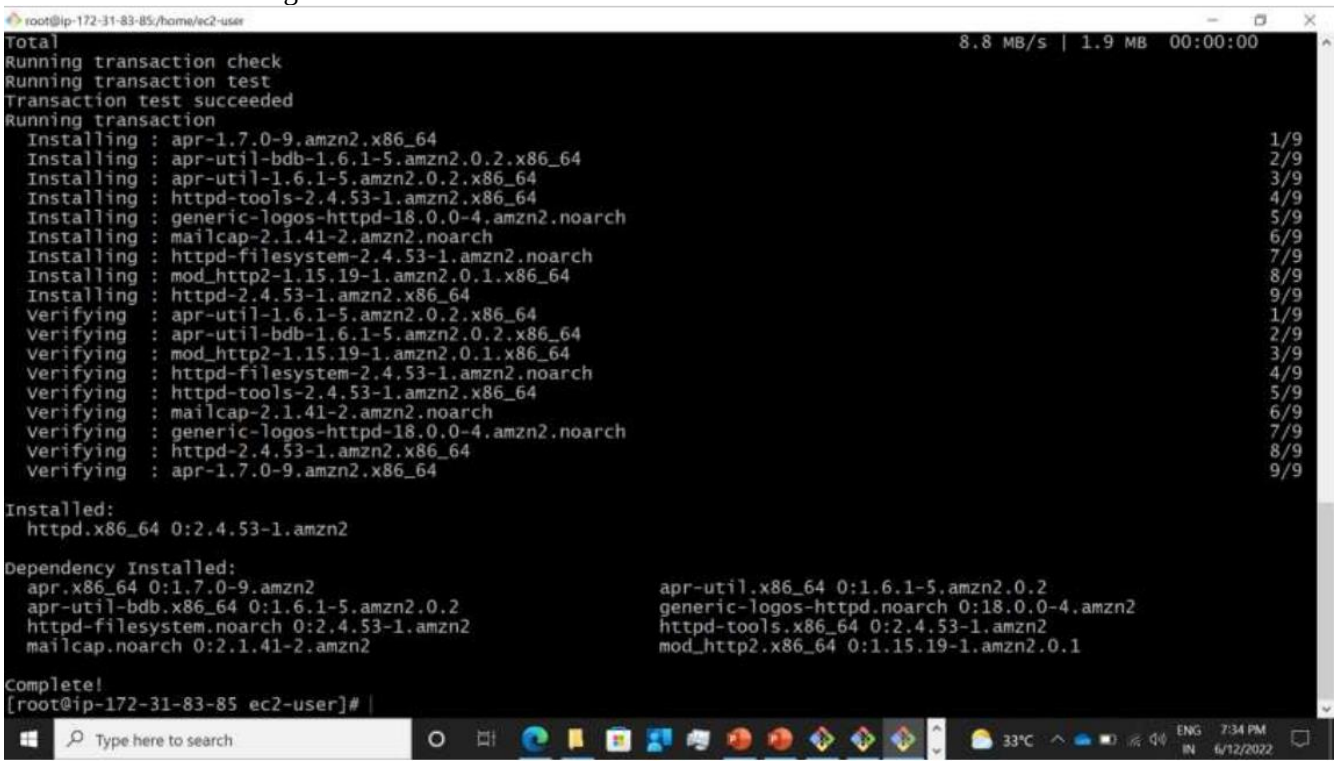
29. In order to install Apache server we have to go to root user by command ---sudo su



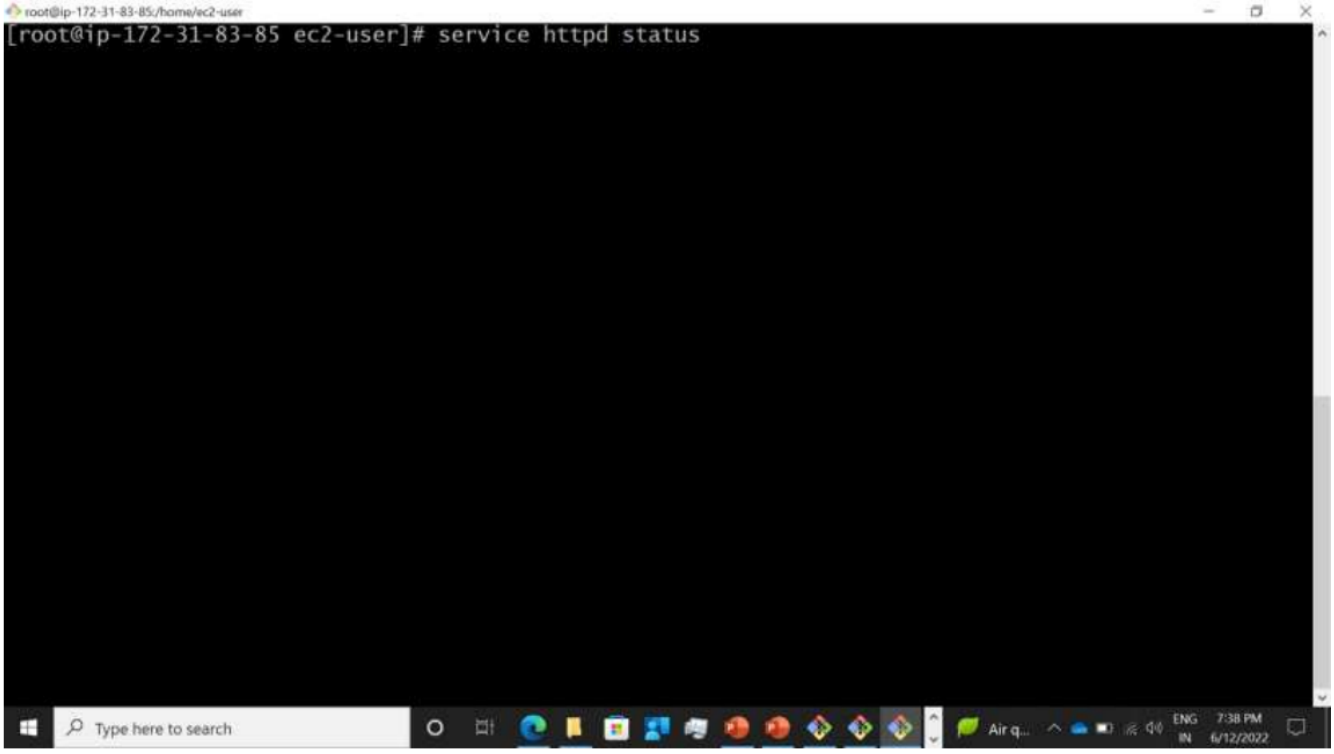
30. Once we are in root user ,we install -----yum install httpd -y



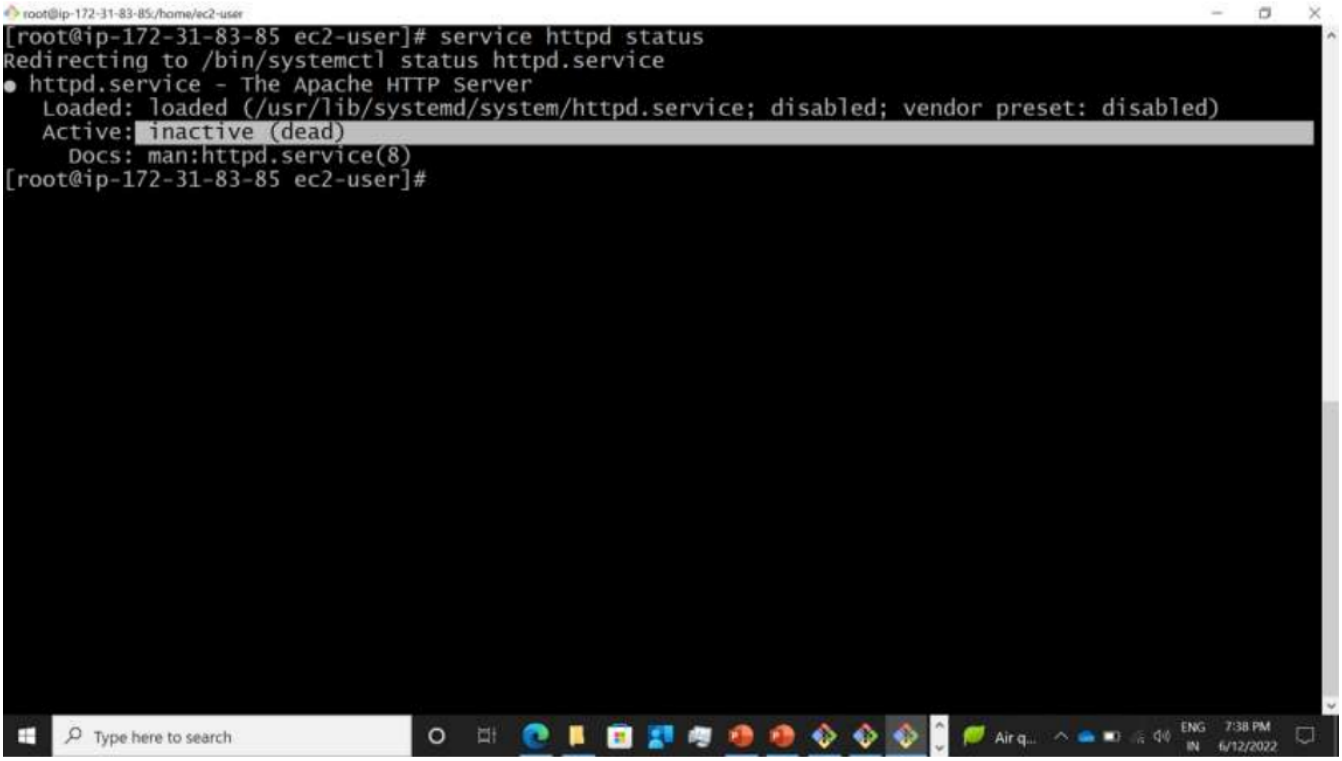
31. We can see installing



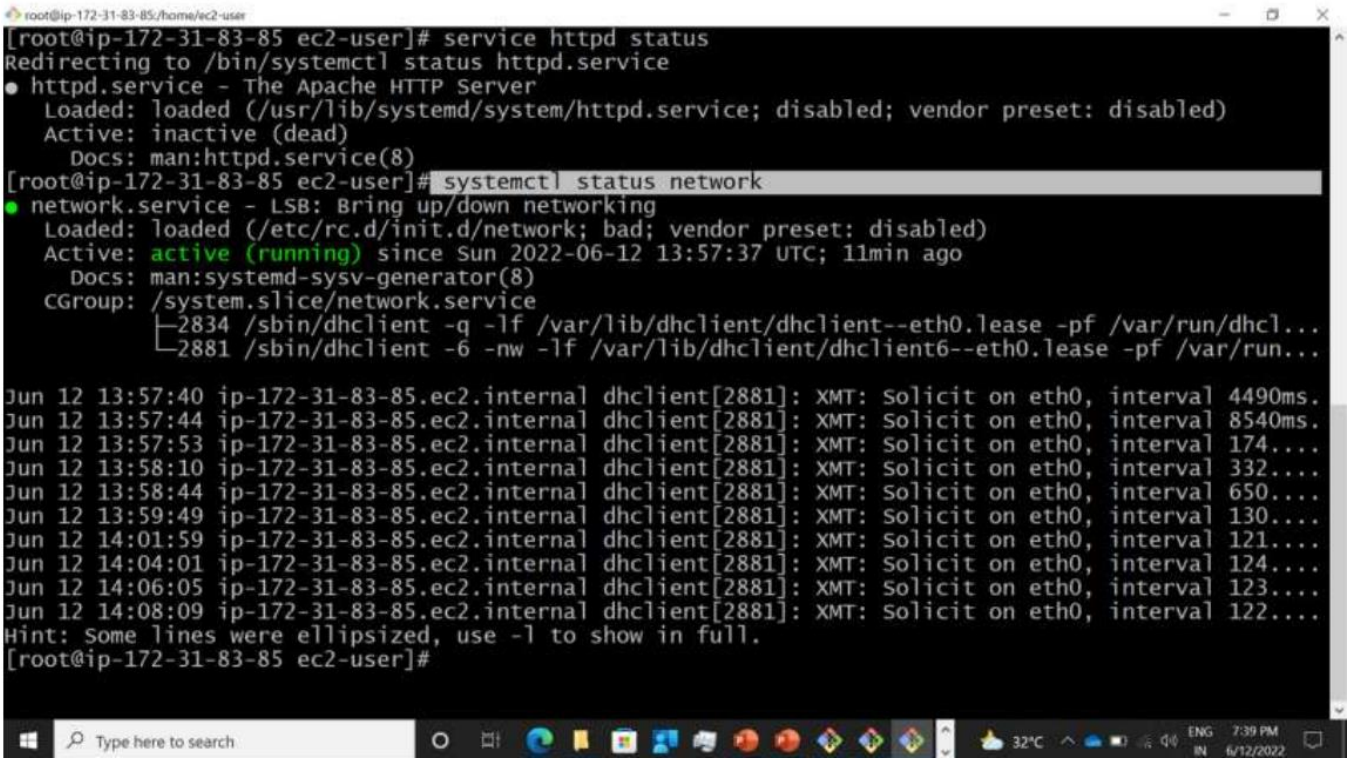
32. Once installed check for status of service----service httpd status



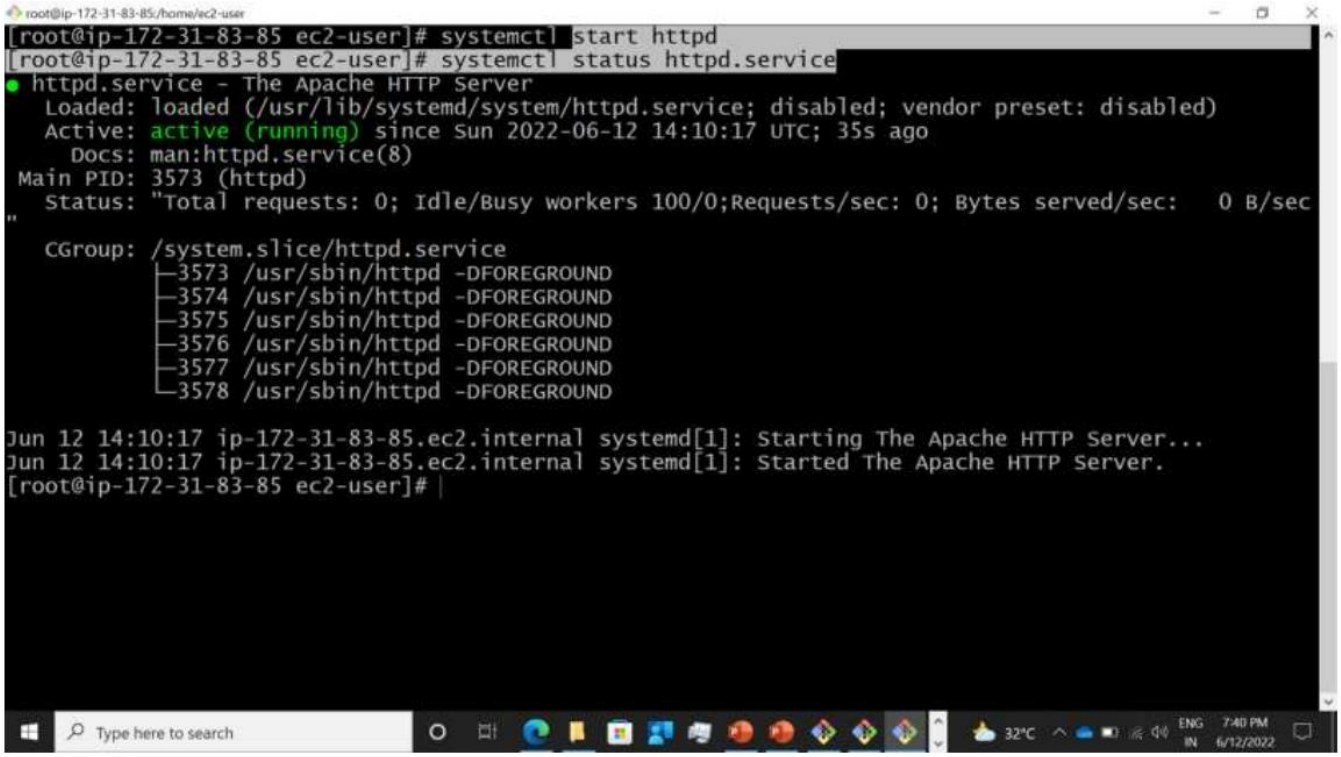
33. Here it is showing the inactive



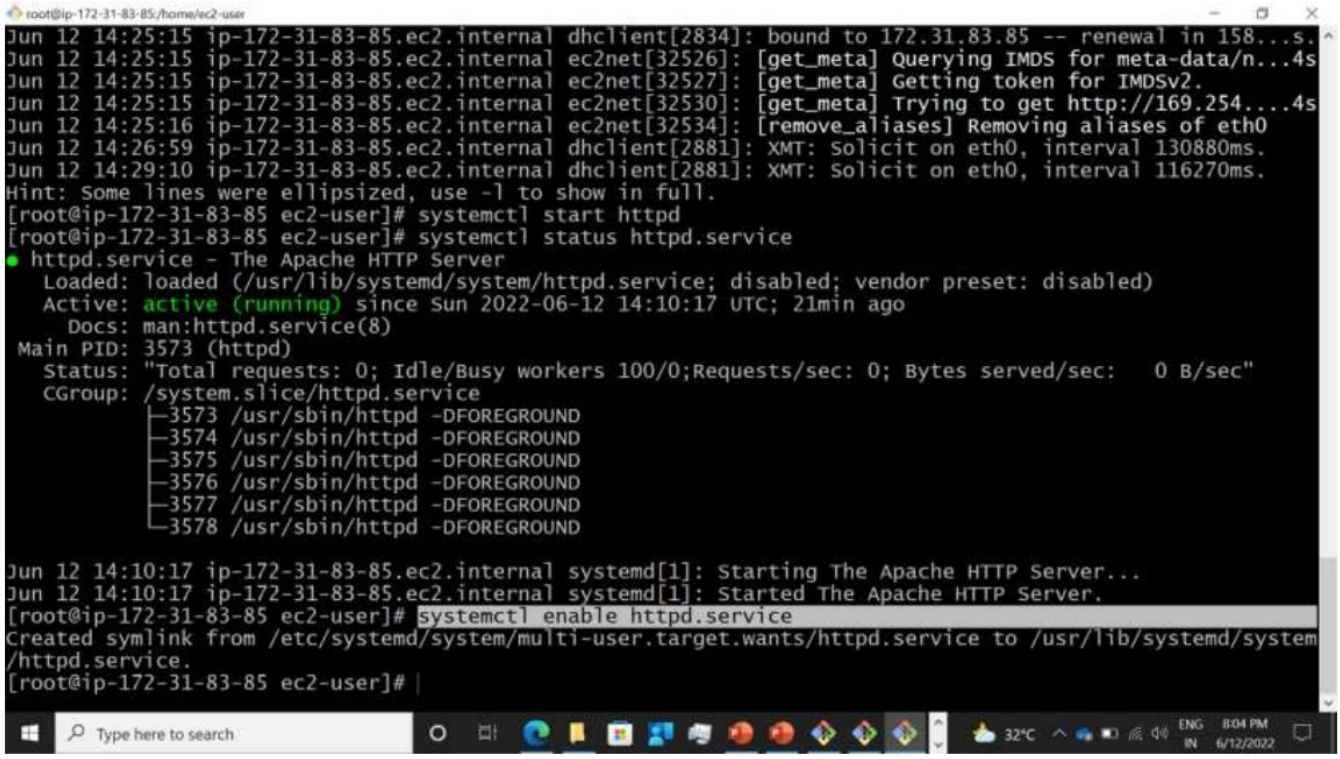
34. To make it active command is ------systemctl status network



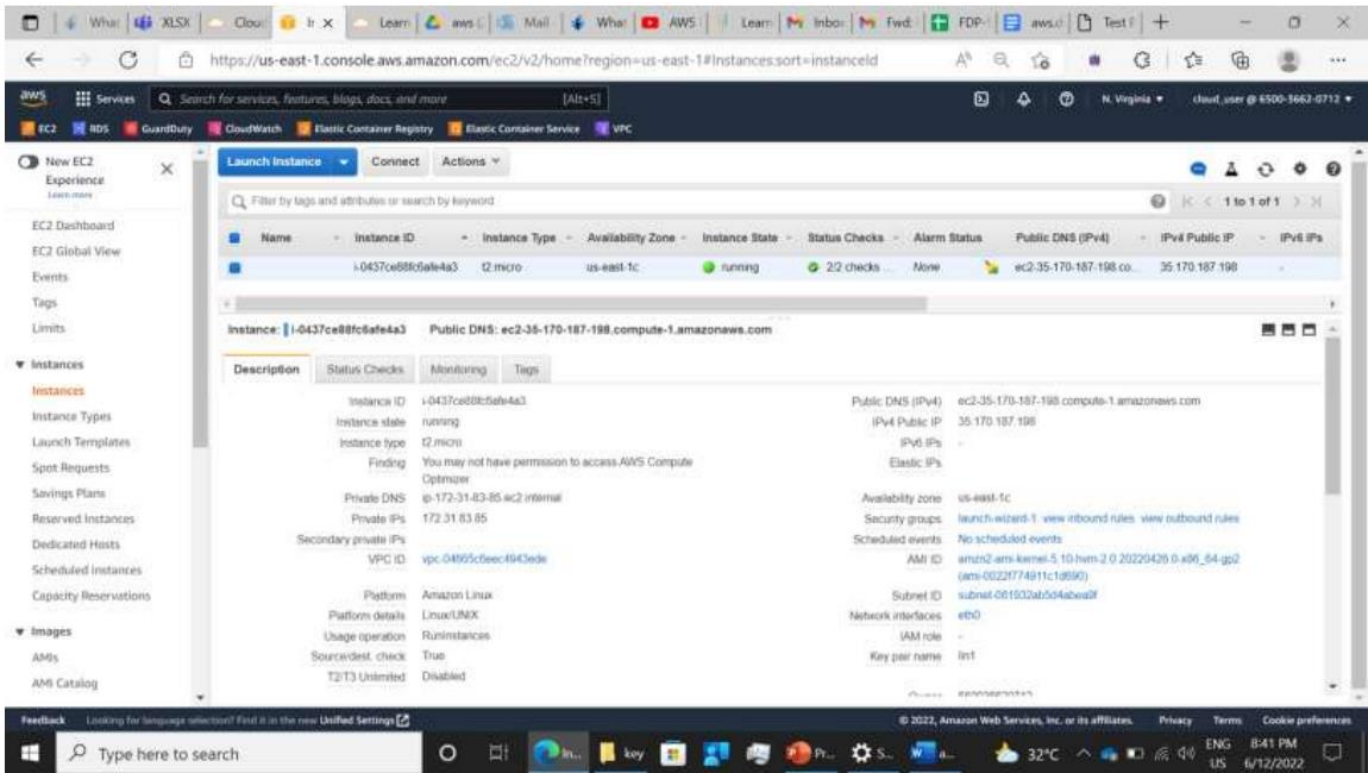
35. Now we can see that it active ----systemctl start httpd -----systemctl status httpd.service



36. Now enable service by -----systemctl enable httpd.service



37. Access the tomcat manager interface from the browser. Move to instances and browse PublicIPV4 address



38. Here we see the Apache web page in new browser