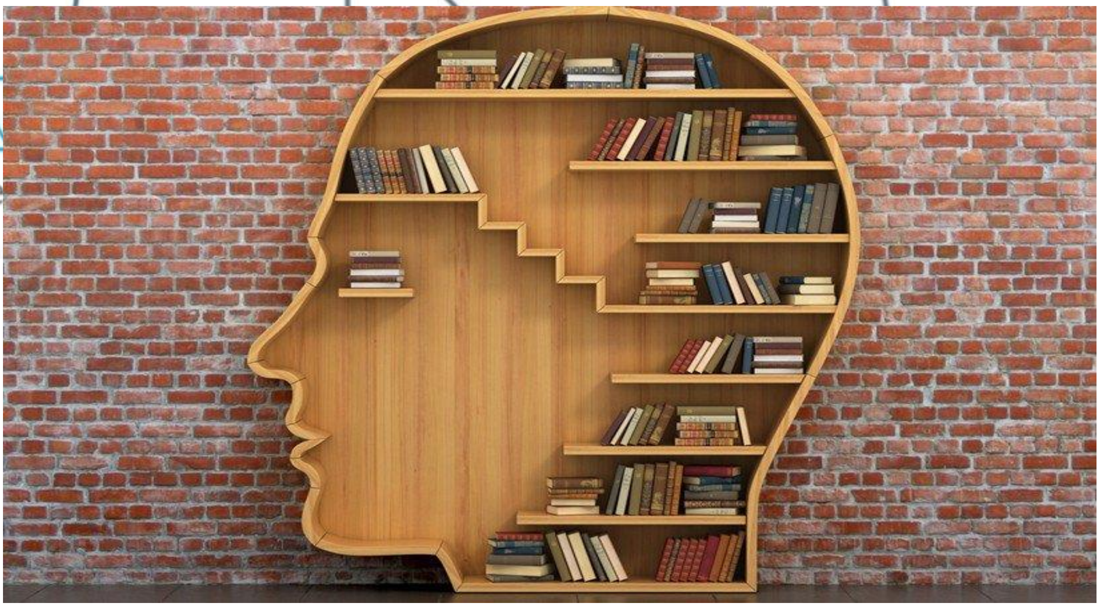
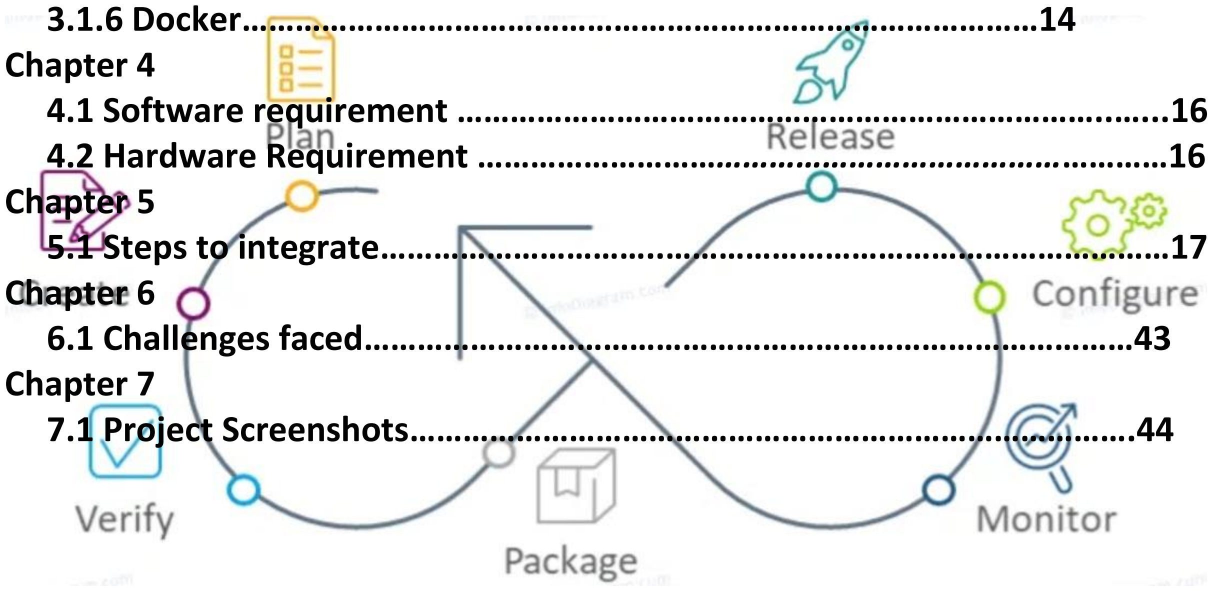
Cache BookStore

Project Report

By: Group 5(Arnav Dasgupta, Debadrita Purkayastha, Rahul Dewan, Stuti Suthar, Utkarsh Deshmukh)



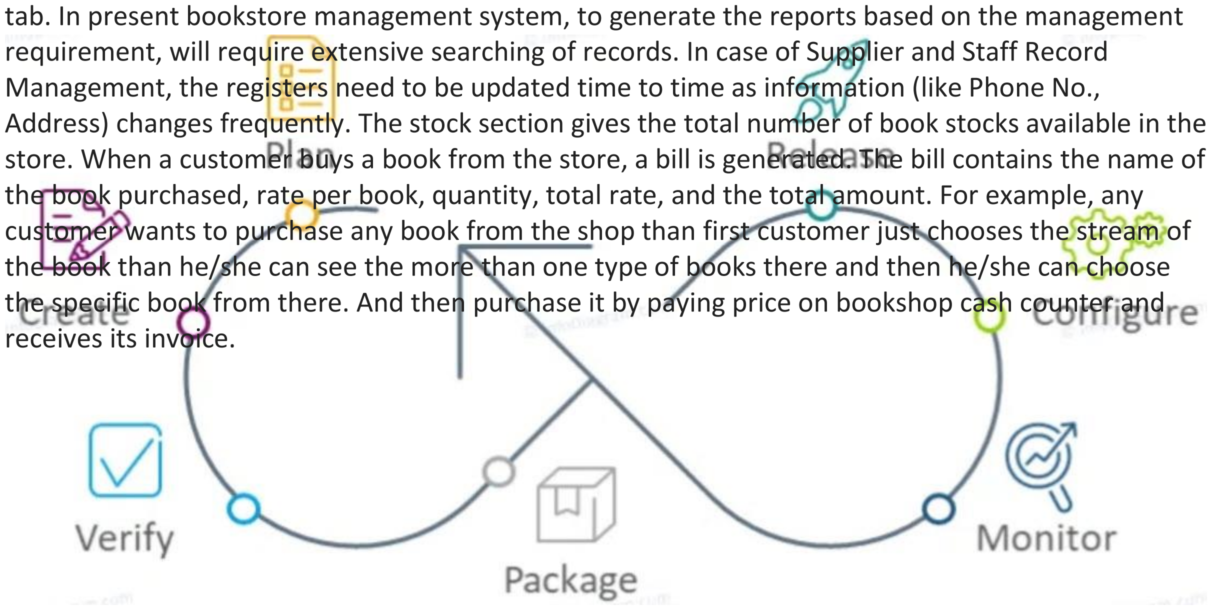
CONTENTS



INTRODUCTION

Book Store Management System is the web application to automate all kinds of operations in the book shop. The purpose of this software is to manage the books in the book store. Generally, it includes the Order Processing, Stock Management and Accounts Management. We developed this software to maintains records of sales, purchase, and staff records. Here we are trying to develop such type system which is provide the automation on the any type of the bookshop. That means a shop which has the type system which provides the facility to the customers of the shop to purchase the books from the shop without any complexity.
At the start of the business, the books store owner buys the book from the dealers. All the name of the books is noted down in the software along with rate. In the present system user must do all work manually. In present system During issuing order of more stock, the product register is required to check to availability of stock in hand. And it takes time to check records.

The amount paid to a particular dealer from whom the book was bought is also saved in the dealers tab. In present bookstore management system, to generate the reports based on the management requirement, will require extensive searching of records. In case of Supplier and Staff Record Management, the registers need to be updated time to time as information (like Phone No., Address) changes frequently. The stock section gives the total number of book stocks available in the store. When a customer buys a book from the store, a bill is generated. The bill contains the name of the book purchased, rate per book, quantity, total rate, and the total amount. For example, any customer wants to purchase any book from the shop than first customer just chooses the stream of the book than he/she can see the more than one type of books there and then he/she can choose the specific book from there. And then purchase it by paying price on bookshop cash counter and receives its invoice.



Introduction to DevOps

1. What is DevOps?

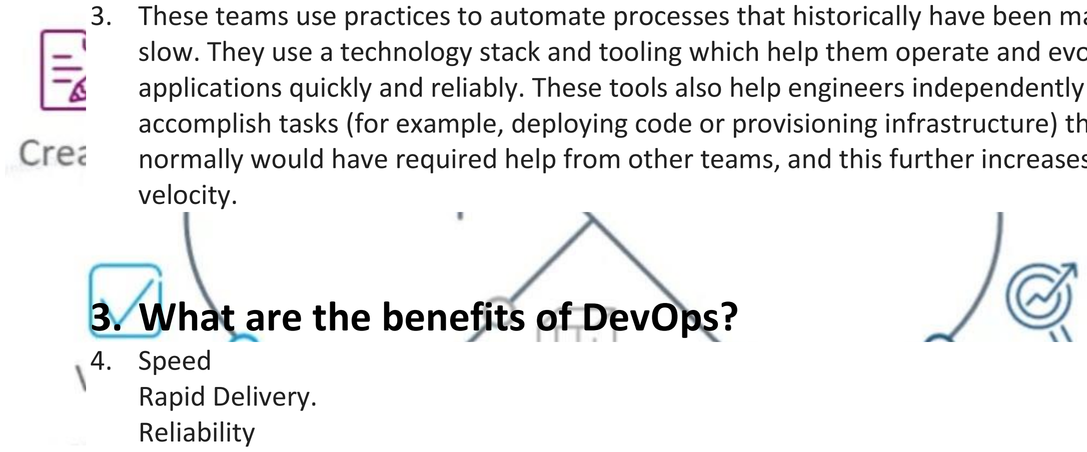
DevOps is the **combination of cultural philosophies, practices, and tools** that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes.

2. How DevOps Works?

1. Under a DevOps model, development and operations teams are no longer “siloed.” Sometimes, these two teams are merged into a single team where the engineers work across the entire application lifecycle, from development and test to deployment to operations, and develop a range of skills not limited to a single function.

2. In some DevOps models, quality assurance and security teams may also become more tightly integrated with development and operations and throughout the application lifecycle. When security is the focus of everyone on a DevOps team, this is sometimes referred to as Develops.

3. These teams use practices to automate processes that historically have been manual and slow. They use a technology stack and tooling which help them operate and evolve applications quickly and reliably. These tools also help engineers independently accomplish tasks (for example, deploying code or provisioning infrastructure) that normally would have required help from other teams, and this further increases a team’s velocity.



3. What are the benefits of DevOps?

4. Speed
   Rapid Delivery.
   Reliability
   Scale
   Improved Collaboration
   Security.

4. Why DevOps matters?

5. Software and the Internet have transformed the world and its industries, from shopping to entertainment to banking. Software no longer merely supports a business; rather it becomes an integral component of every part of a business. Companies interact with their customers through software delivered as online services or applications and on all sorts of devices. They also use software to increase operational efficiencies by transforming every part of the value chain, such as logistics, communications, and operations. In a similar way that physical goods companies transformed how they design, build, and deliver products using industrial automation throughout the 20th century, companies in today’s world must transform how they build and deliver software.

6.

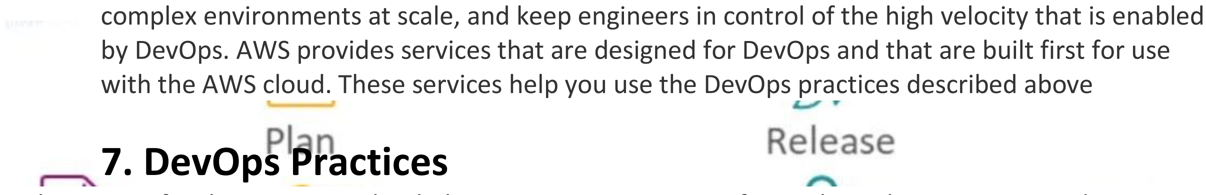
5. DevOps Practices

The following are DevOps best practices:

- Continuous Integration
- Continuous Delivery
- Microservices
- Infrastructure as Code
- Monitoring and Logging
- Communication and Collaboration

6. DevOps Tools

The DevOps model relies on effective tooling to help teams rapidly and reliably deploy and innovate for their customers. These tools automate manual tasks, help teams manage complex environments at scale, and keep engineers in control of the high velocity that is enabled by DevOps. AWS provides services that are designed for DevOps and that are built first for use with the AWS cloud. These services help you use the DevOps practices described above



7. DevOps Practices

There are a few key practices that help organizations innovate faster through automating and streamlining the software development and infrastructure management processes. Most of these practices are accomplished with proper tooling.

One fundamental practice is to perform very frequent but small updates. This is how organizations innovate faster for their customers. These updates are usually more incremental in nature than the occasional updates performed under traditional release practices. Frequent but small updates make each deployment less risky. They help teams address bugs faster because teams can identify the last deployment that caused the error. Although the cadence and size of updates will vary, organizations using a DevOps model deploy updates much more often than organizations using traditional software development practices.

Organizations might also use a microservices architecture to make their applications more flexible and enable quicker innovation. The microservices architecture decouples large, complex systems into simple, independent projects. Applications are broken into many individual components (services) with each service scoped to a single purpose or function and operated independently of its peer services and the application. This architecture reduces the coordination overhead of updating applications, and when each service is paired with small, agile teams who take ownership of each service, organizations can move more quickly.

However, the combination of microservices and increased release frequency leads to significantly more deployments which can present operational challenges. Thus, DevOps practices like continuous integration and continuous delivery solve these issues and let organizations deliver rapidly in a safe and reliable manner. Infrastructure automation practices, like infrastructure as code and configuration management, help to keep computing resources elastic and responsive to frequent changes. In addition, the use of monitoring and logging helps engineers track the performance of applications and infrastructure so they can react quickly to problems.

Together, these practices help organizations deliver faster, more reliable updates to their customers. Here is an overview of important DevOps practices.

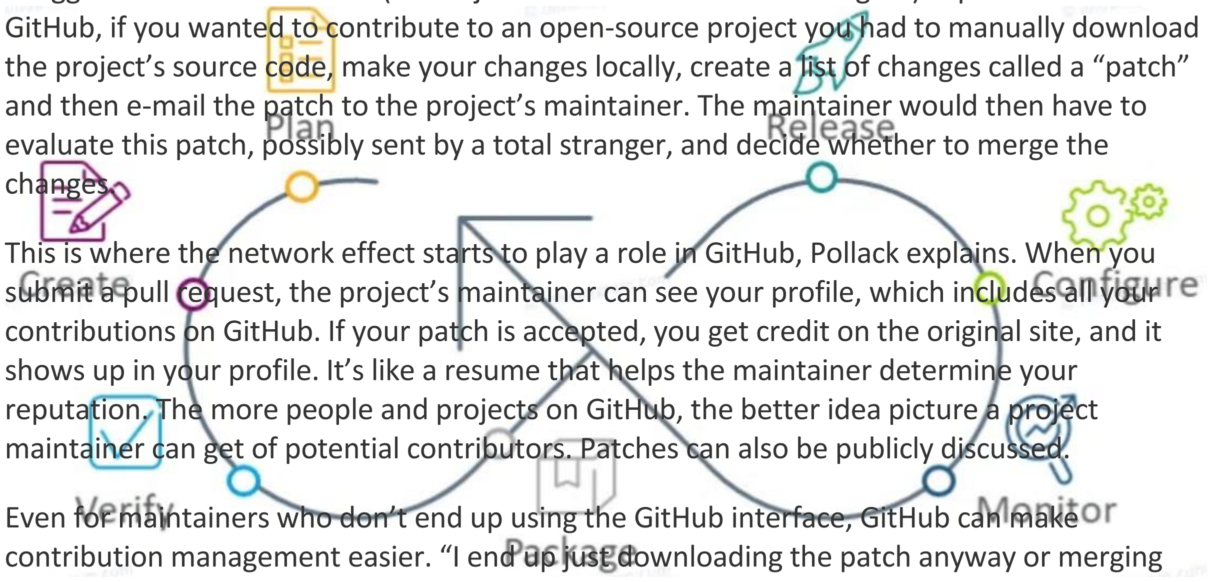
Introduction

3.1.1 GitHub

GitHub is a Git repository hosting service, but it adds many of its own features. While Git is a command line tool, GitHub provides a Web-based graphical interface. It also provides access control and several collaboration features, such as a wikis and basic task management tools for every project.

The flagship functionality of GitHub is “forking” – copying a repository from one user’s account to another. This enables you to take a project that you don’t have write access to and modify it under your own account. If you make changes you’d like to share, you can send a notification called a “pull request” to the original owner. That user can then, with a click of a button, merge the changes found in your repo with the original repo.

These three features – fork, pull request and merge – are what make GitHub so powerful. Gregg Pollack of Code School (which just launched a class called Trigate) explains that before GitHub, if you wanted to contribute to an open-source project you had to manually download the project’s source code, make your changes locally, create a list of changes called a “patch” and then e-mail the patch to the project’s maintainer. The maintainer would then have to evaluate this patch, possibly sent by a total stranger, and decide whether to merge the changes.



This is where the network effect starts to play a role in GitHub, Pollack explains. When you submit a pull request, the project’s maintainer can see your profile, which includes all your contributions on GitHub. If your patch is accepted, you get credit on the original site, and it shows up in your profile. It’s like a resume that helps the maintainer determine your reputation. The more people and projects on GitHub, the better idea picture a project maintainer can get of potential contributors. Patches can also be publicly discussed.

Even for maintainers who don’t end up using the GitHub interface, GitHub can make contribution management easier. “I end up just downloading the patch anyway or merging from the command line instead of from the merge button,” says Isaac Schlueter, the maintainer of the open-source development platform Node.js. “But GitHub provides a centralized place where people can discuss the patch.”

Lowering the barrier to entry democratizes open-source development, and helps young projects grow. “Node.js wouldn’t be what it is today without GitHub,” Schlueter says.

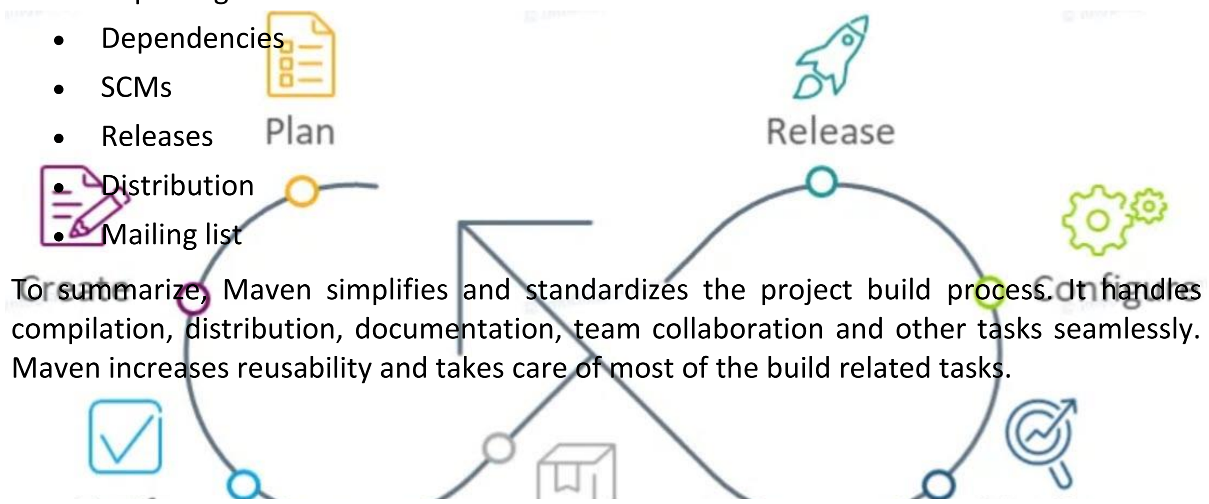
3.1.2 Maven/Gradle

Maven is a project management and comprehension tool that provides developers a complete build lifecycle framework. Development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.

In case of multiple development team's environment, Maven can set-up the way to work as per standards in a very short time. As most of the project setups are simple and reusable, Maven makes life of developer easy while creating reports, checks, build and testing automation setups.

Maven provides developers ways to manage the following –

- Builds
- Documentation
- Reporting
- Dependencies
- SCMs
- Releases
- Distribution
- Mailing list



To summarize, Maven simplifies and standardizes the project build process. It handles compilation, distribution, documentation, team collaboration and other tasks seamlessly. Maven increases reusability and takes care of most of the build related tasks.

Gradle is an open-source build automation tool that is designed to be flexible enough to build almost any type of software. The following is a high-level overview of some of its most important features:

High performance

Gradle avoids unnecessary work by only running the tasks that need to run because their inputs or outputs have changed. You can also use a build cache to enable the reuse of task outputs from previous runs or even from a different machine (with a shared build cache).

There are many other optimizations that Gradle implements, and the development team continually work to improve Gradle's performance.

JVM foundation

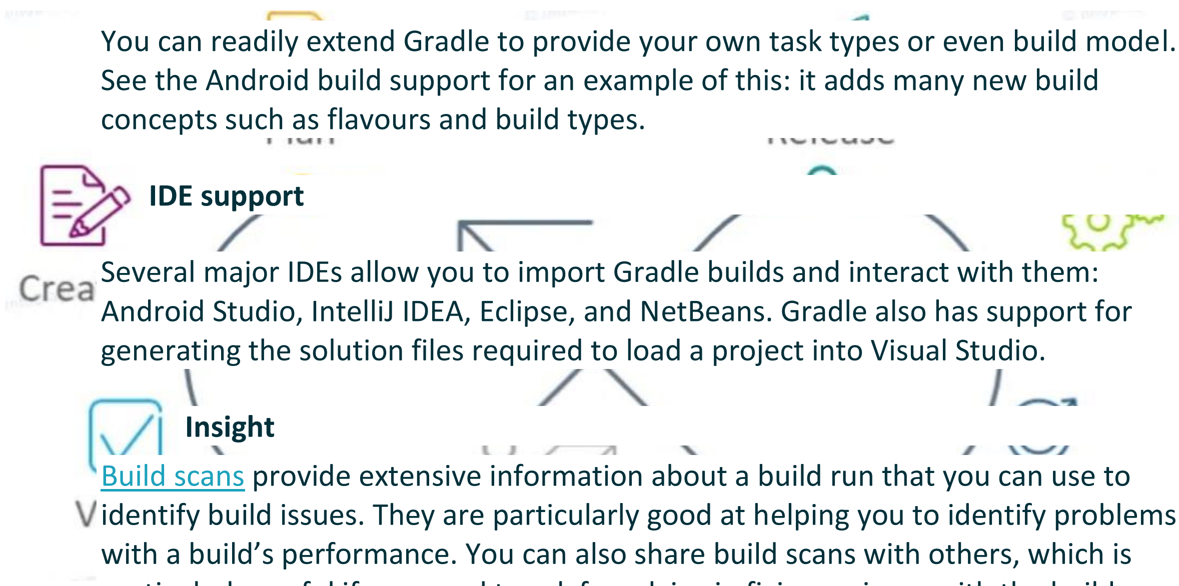
Gradle runs on the JVM and you must have a Java Development Kit (JDK) installed to use it. This is a bonus for users familiar with the Java platform as you can use the standard Java APIs in your build logic, such as custom task types and plugins. It also makes it easy to run Gradle on different platforms.

Note that Gradle isn’t limited to building just JVM projects, and it even comes packaged with support for building native projects.

Conventions

Gradle takes a leaf out of Maven’s book and makes common types of projects — such as Java projects — easy to build by implementing conventions. Apply the appropriate plugins and you can easily end up with slim build scripts for many projects. But these conventions don’t limit you: Gradle allows you to override them, add your own tasks, and make many other customizations to your convention-based builds.

Extensibility



You can readily extend Gradle to provide your own task types or even build model. See the Android build support for an example of this: it adds many new build concepts such as flavours and build types.

IDE support

Several major IDEs allow you to import Gradle builds and interact with them: Android Studio, IntelliJ IDEA, Eclipse, and NetBeans. Gradle also has support for generating the solution files required to load a project into Visual Studio.

Insight

Build scans provide extensive information about a build run that you can use to identify build issues. They are particularly good at helping you to identify problems with a build’s performance. You can also share build scans with others, which is particularly useful if you need to ask for advice in fixing an issue with the build.

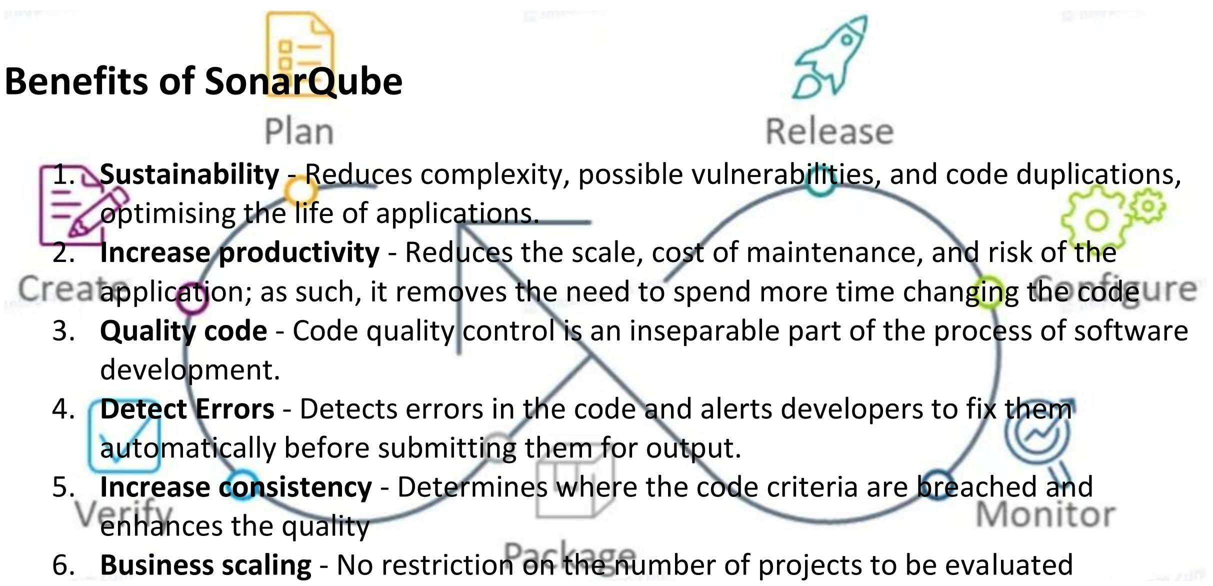
3.1.3 SonarQube

What is SonarQube?

SonarQube is an open-source platform developed by Sonar Source for continuous inspection of code quality. Sonar does static code analysis, which provides a detailed report of bugs, code smells, vulnerabilities, code duplications.

It supports 25+ major programming languages through built-in rulesets and can also be extended with various plugins.

Benefits of SonarQube



1. **Sustainability** - Reduces complexity, possible vulnerabilities, and code duplications, optimising the life of applications.
2. **Increase productivity** - Reduces the scale, cost of maintenance, and risk of the application; as such, it removes the need to spend more time changing the code
3. **Quality code** - Code quality control is an inseparable part of the process of software development.
4. **Detect Errors** - Detects errors in the code and alerts developers to fix them automatically before submitting them for output.
5. **Increase consistency** - Determines where the code criteria are breached and enhances the quality
6. **Business scaling** - No restriction on the number of projects to be evaluated
7. **Enhance developer skills** - Regular feedback on quality problems helps developers to improve their coding skills

Why SonarQube?

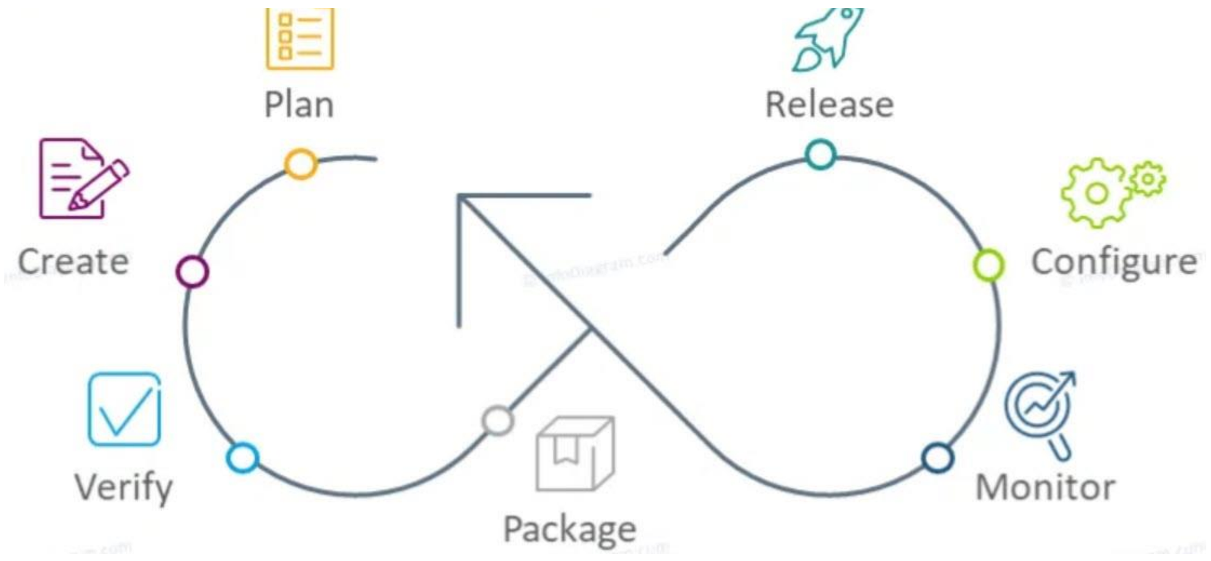
Developers working with hard deadlines to deliver the required functionality to the customer. It is so important for developers that many times they compromise with the code quality, potential bugs, code duplications, and bad distribution of complexity. Additionally, they tend to leave unused variables, methods, etc. In this scenario, the code would work in the desired way.

Do you think this is a proper way to deliver functionality?

The answer is NO.

To avoid these issues in code, developers should always follow the good coding practice, but sometimes it is not possible to follow the rules and maintain the good quality as there may be many reasons.

To achieve continuous code integration and deployment, developers need a tool that not only works once to check and tell them the problems in the code but also to track and control the code to check continuous code quality. To satisfy all these requirements, here comes SonarQube in the picture.

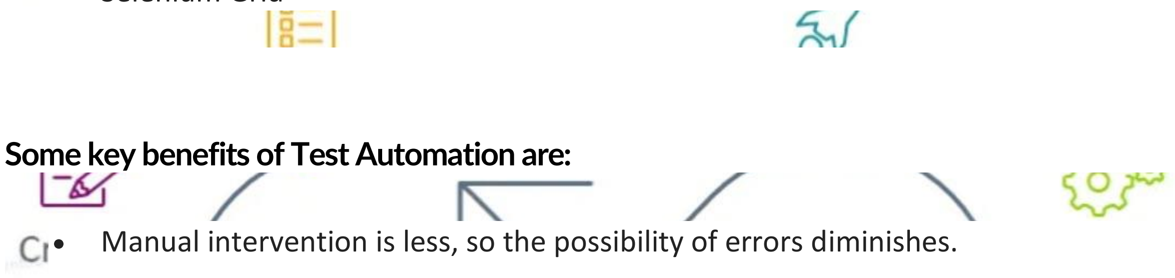


3.1.4 Selenium

What is Selenium?

Selenium is a free (open source) automated testing framework used to validate web applications across different browsers and platforms. You can use multiple programming languages like Java, C#, Python etc to create Selenium Test Scripts. Testing done using the Selenium testing tool is usually referred to as Selenium Testing.
Selenium Software is not just a single tool but a suite of software, each piece catering to different Selenium QA testing needs of an organization. Here is the list of tools

- Selenium Integrated Development Environment (IDE)
- Selenium Remote Control (RC)
- WebDriver
- Selenium Grid



Some key benefits of Test Automation are:

- Manual intervention is less, so the possibility of errors diminishes.

- It ensures higher ROI on the huge investments required initially.
- Automated tests make the process more reliable and the tests more dependable.
- Automation helps you find bugs at an early stage.
- You can test 24*7 from a remotely held device as well.
- It makes the test scripts reusable – need new scripts every time even with changes in the version of the OS on the device and the tests can recur without any errors.

- Most importantly, it enables testing in volumes. For instance, it allows you to run tests on thousands of mobile devices. Now, this is impossible with Manual Testing.

3.1.5 Jenkins

Jenkins is an open-source automation server. With Jenkins, organizations can accelerate the software development process by automating it. Jenkins manages and controls software delivery processes throughout the entire lifecycle, including build, document, test, package, stage, deployment, static code analysis and much more.

You can set up Jenkins to watch for any code changes in places like GitHub, Bitbucket or GitLab and automatically do a build a with tools like Maven and Gradle. You can utilize container technologies such as Docker and Kubernetes, initiate tests and then take actions like rolling back or rolling forward in production.

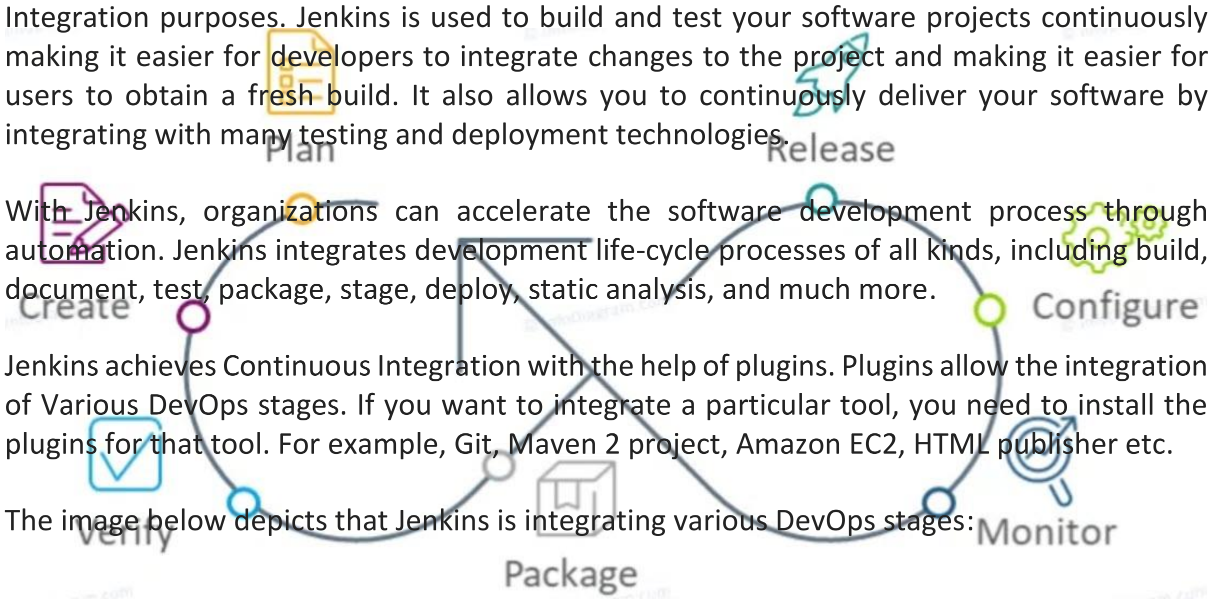
What is Jenkins and why we use it?

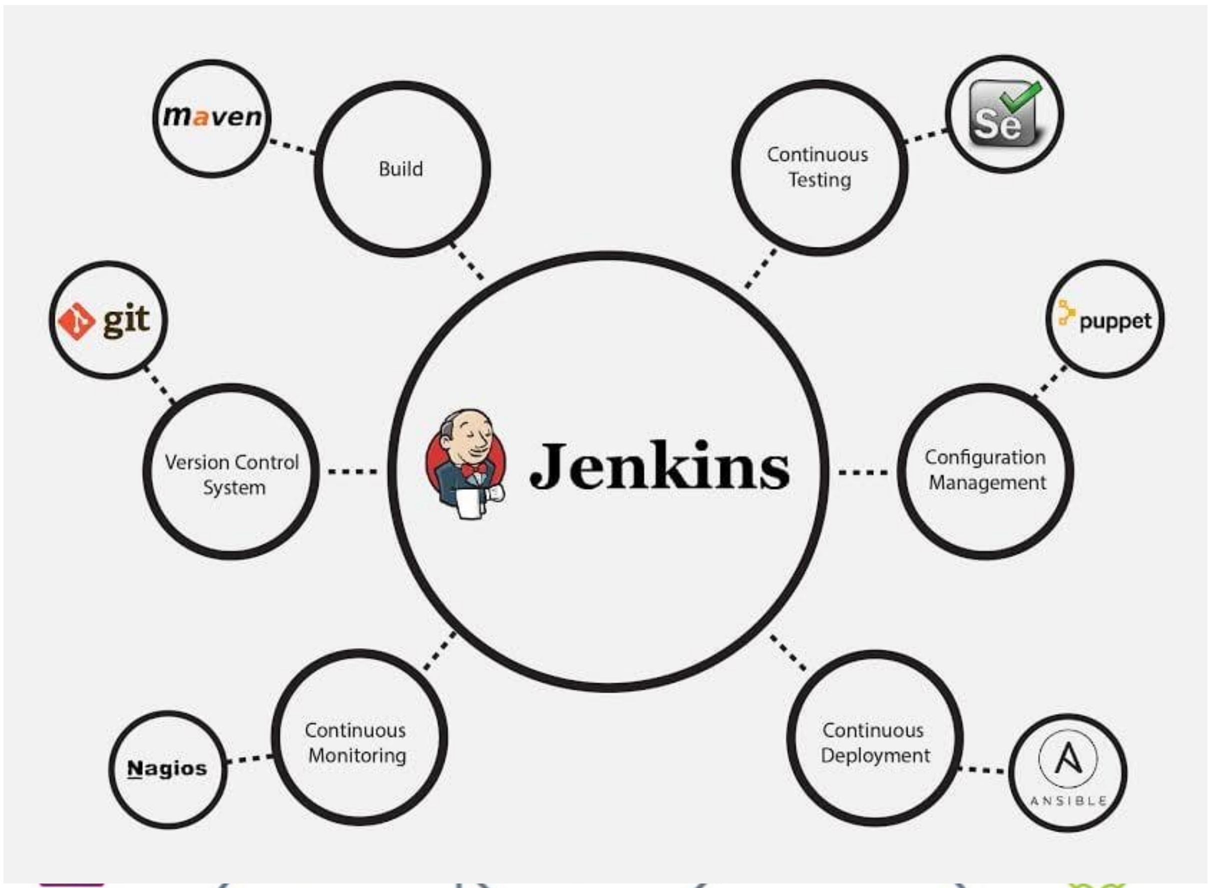
Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with many testing and deployment technologies.

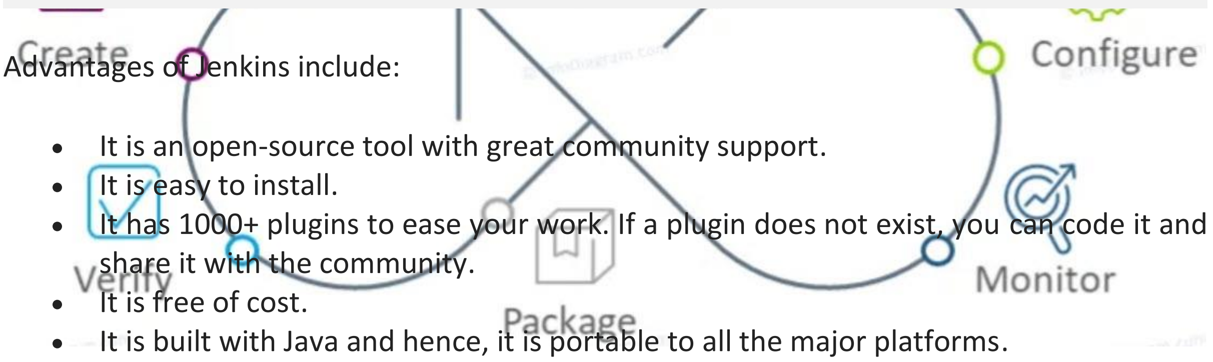
With Jenkins, organizations can accelerate the software development process through automation. Jenkins integrates development life-cycle processes of all kinds, including build, document, test, package, stage, deploy, static analysis, and much more.

Jenkins achieves Continuous Integration with the help of plugins. Plugins allow the integration of Various DevOps stages. If you want to integrate a particular tool, you need to install the plugins for that tool. For example, Git, Maven 2 project, Amazon EC2, HTML publisher etc.

The image below depicts that Jenkins is integrating various DevOps stages:







Advantages of Jenkins include:

- It is an open-source tool with great community support.
- It is easy to install.
- It has 1000+ plugins to ease your work. If a plugin does not exist, you can code it and share it with the community.
- It is free of cost.
- It is built with Java and hence, it is portable to all the major platforms.

There are certain things about Jenkins that separates it from other the Continuous Integration tool. Let us look on those points.

3.1.6 Docker

Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.
Running Docker on AWS provides developers and admins a highly reliable, low-cost way to build, ship, and run distributed applications at any scale.

Key Benefits of Docker Containers

The crucial word here is: agility. When we say ‘agility’ it basically refers to getting features and updates to customers or clients faster. Docker is an important tool when you’re creating the groundwork for any modern application. Primarily, it enables easy deployment to the cloud. Beyond that, Docker technology is also more controllable, more granular and is a microservices-based method focused on efficiency.
Now, let’s take a deep dive by examining the seven major Docker benefits.
Consistent and Isolated Environment
Using containers developers can create predictable environments that are isolated from other apps. Regardless of where the app is deployed, everything remains consistent and this leads to massive productivity: less time debugging, and more time launching fresh features and functionality for users.
Cost-effectiveness with Fast Deployment
Docker-powered containers are known for decreasing deployment time to seconds. That’s an impressive feat by any standard. Traditionally, things like provisioning, getting the hardware up and running would take days or more. In addition, you faced massive overheads and extra work. When each process is put into a container, it can be shared with new apps. The process of deployment becomes swift, and you are essentially ready to go.
Mobility – Ability to Run Anywhere
Docker images are free of environmental limitations, and that makes any deployment consistent, movable (portable), and scalable. Containers have the added benefit of running anywhere, providing it is targeted at the OS (Win, Mac OS, Linux, VMs, On-prem, in Public Cloud), which is a huge advantage for both development and deployment. The widespread popularity of the Docker image format for containers further helps. It has been adopted by leading cloud providers, including Amazon Web Services (AWS), Google Compute Platform (GCP), and Microsoft Azure. In addition, you have powerful orchestration systems such as Kubernetes, and products such as AWS ECS or Azure Container Instances are mighty useful in terms of mobility.
Repeatability and Automation
You are building code with repeatable infrastructure and config. This speed up the development process tremendously. It must be pointed out that Docker images are often small. Consequently, you get fast delivery and, again, shorter deployment for new application containers. Another advantage is straightforward maintenance. When an application is containerized, it’s isolated from other apps running in the same system. In

other words, apps don’t intermix and application maintenance significantly easier. It lends itself to being automated; the faster you repeat, the fewer mistakes you make, and the more you can focus on core value for a business or application.

Test, Roll Back and Deploy

As we said, Environments remain more consistent in Docker, from start to finish. Docker images are easily versioned, which makes them easy to roll back if you need to do so. If there is a problem with the current iteration of the image, just roll back to the older version. The whole process means you are creating the perfect environment for continuous integration and continuous deployment (CI/CD). Docker containers are set to retain all configs and dependencies internally. Now, you have a fast and easy way of checking for discrepancies.

Flexibility

If you need to perform an upgrade during a product’s release cycle, you can easily make the necessary changes to Docker containers, test them, and roll out new containers. This sort of flexibility is another key advantage of using Docker. Docker really allows you to build, test, and release images that can be deployed across multiple servers. Even if a new security patch is available, the process remains the same. You can apply the patch, test it, and release it to production. Additionally, Docker allows you to start and stop services, or apps rapidly, which is especially useful within the cloud environment.

Collaboration, Modularity and Scaling

The Docker method of containerization allows you to segment an application so you can refresh, clean up, repair without even taking down the entire app. Furthermore, with Docker you can build an architecture for applications comprising of small processes that communicate with each other via APIs. From there, developers share and collaborate, solving any potential issues quickly. At this stage, the development cycle is completed, and all issues are resolved, with no massive overhaul needed – this is extremely cost-effective and timesaving.

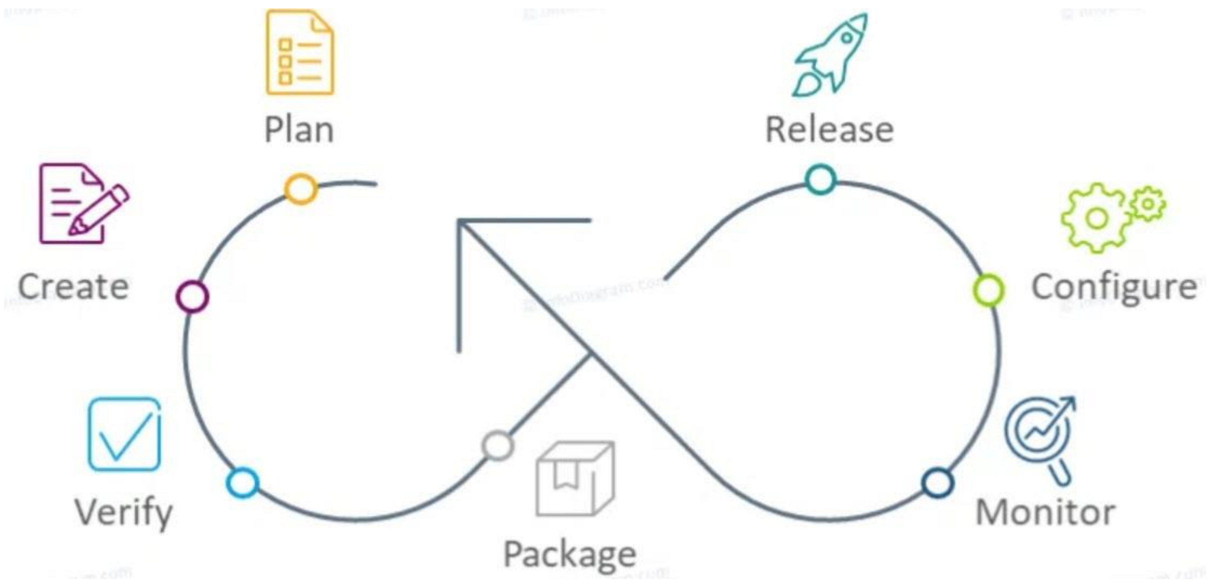
SOFTWARE REQUIREMENTS:

1) JDK
2) Maven
3) GitHub Account
4) Sonar Server/ Account
5) Jenkins
6) Docker

HARDWARE REQUIREMENTS:

1) 4 GB RAM

2) I4 processor and up.



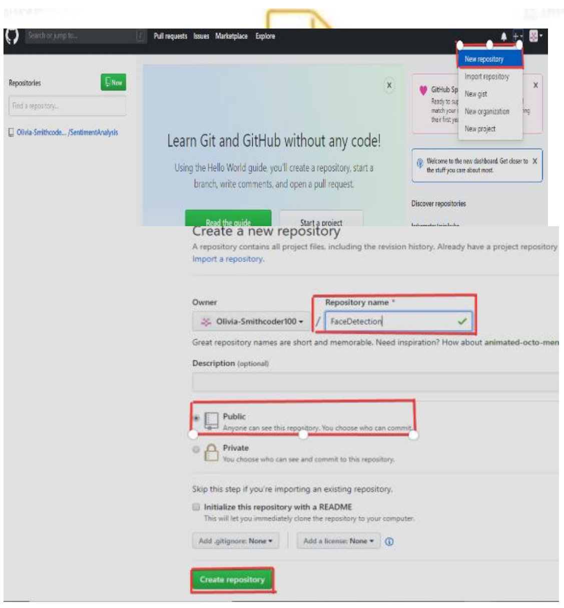
5. Steps to integrate the Project

Git Hub Integration

Using Command line to PUSH to GitHub

1. Creating a new repository.

- You need to create a new repository and click on the plus sign.
- Fill up all the required details, i.e., repository name, description and make the repository public this time as it is free.



Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Welcome to the new dashboard. Get closer to the stuff you care about most.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository

Import a repository.

Owner: Olivia-Smithcoder100 / Repository name *: FaceDetection

Great repository names are short and memorable. Need inspiration? How about animated-octo-men

Description (optional)

● Public: Anyone can see this repository. You choose who can commit.

○ Private: You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: None ▾ | Add a license: None ▾

Create repository

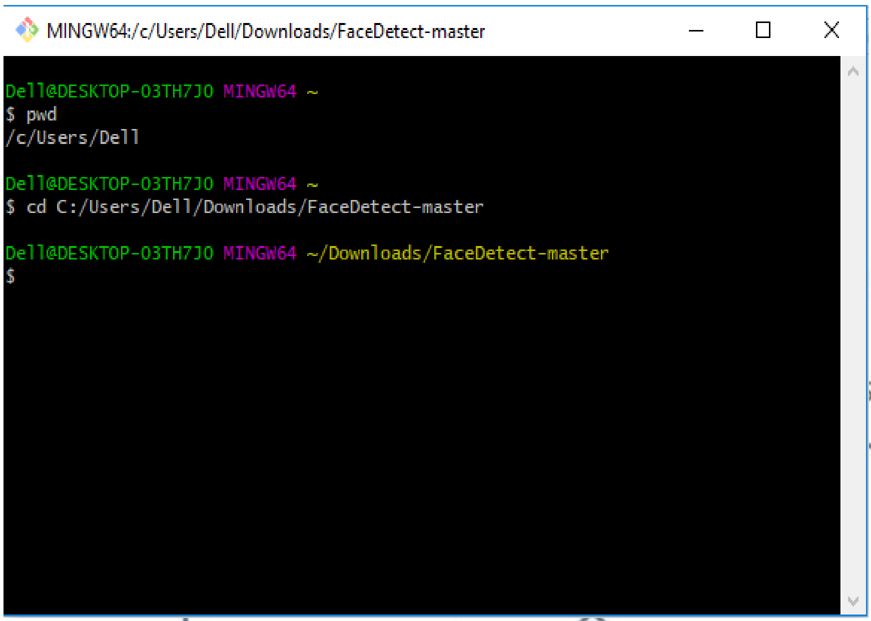
2. Open your Git Bash.

- Git Bash can be downloaded in here, and it is a shell used to interface with the operating system which follows the UNIX command.

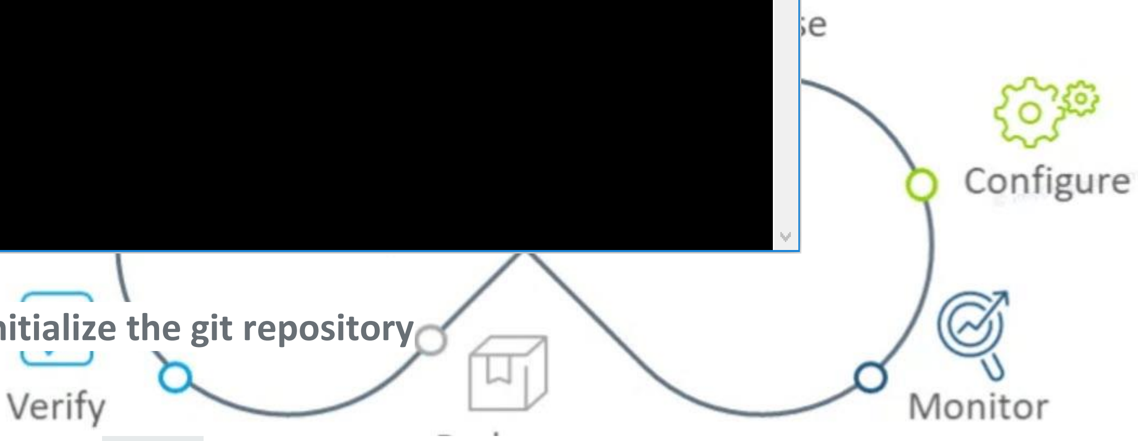
3. Create your local project in your desktop directed towards a current working directory.

- pad stands for 'print working directory', which is used to print the current directory.

- Move to the specific path in your local computer by cd 'pathname'. The cd commands stand for 'change directory' and it is used to change to the working directory in your operating system, and to locate your file, 'pathname', i.e., C:/Users/Dell/Downloads/FaceDetect-master needs to be given. This command can identify the required file that you are looking to work with.

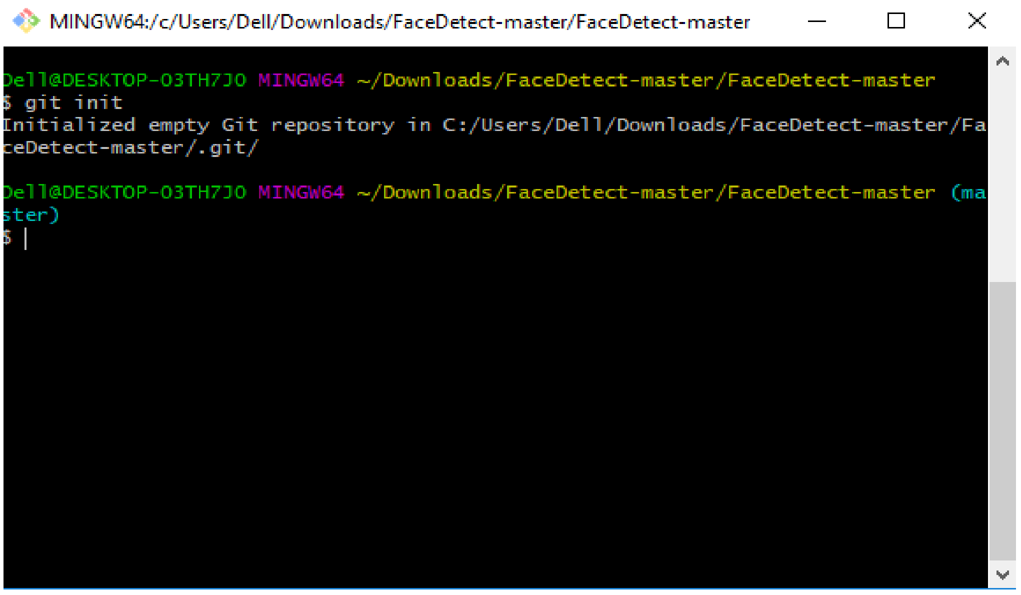


```
MINGW64:/c/Users/Dell/Downloads/FaceDetect-master

Dell@DESKTOP-03TH7J0 MINGW64 ~
$ pwd
/c/Users/Dell

Dell@DESKTOP-03TH7J0 MINGW64 ~
$ cd C:/Users/Dell/Downloads/FaceDetect-master

Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master
$
```



4. Initialize the git repository

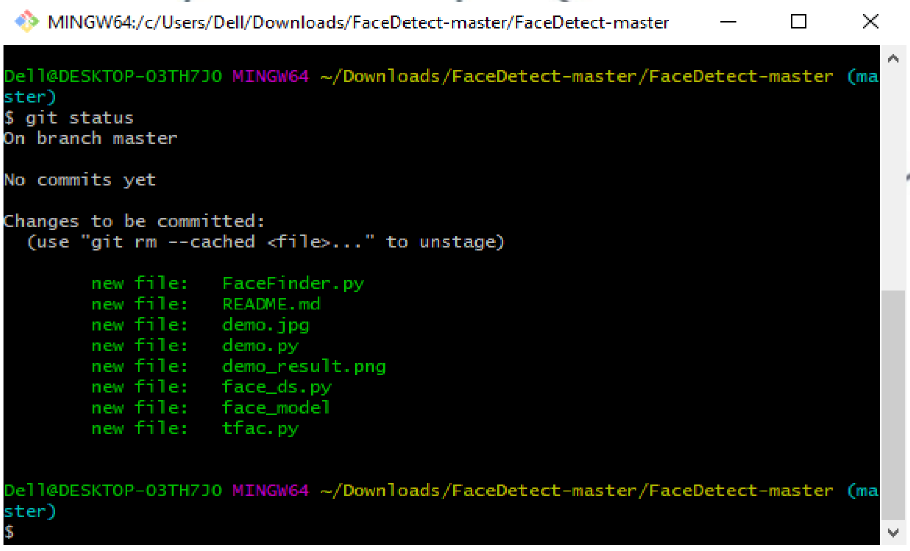
- Use git in it to initialize the repository. It is used to create a new empty repository or directory consisting of files with the hidden directory. ‘. git' is created at the top level of your project, which places all of the revision information in one place.



```
Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master/FaceDetect-master
$ git init
Initialized empty Git repository in C:/Users/Dell/Downloads/FaceDetect-master/Fa
ceDetect-master/.git/

Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master/FaceDetect-master (ma
ster)
$ |
```

5. Add the file to the new local repository.

- Use git add. in your bash to add all the files to the given folder.
- Use git status in your bash to view all the files which are going to be staged to the first commit.

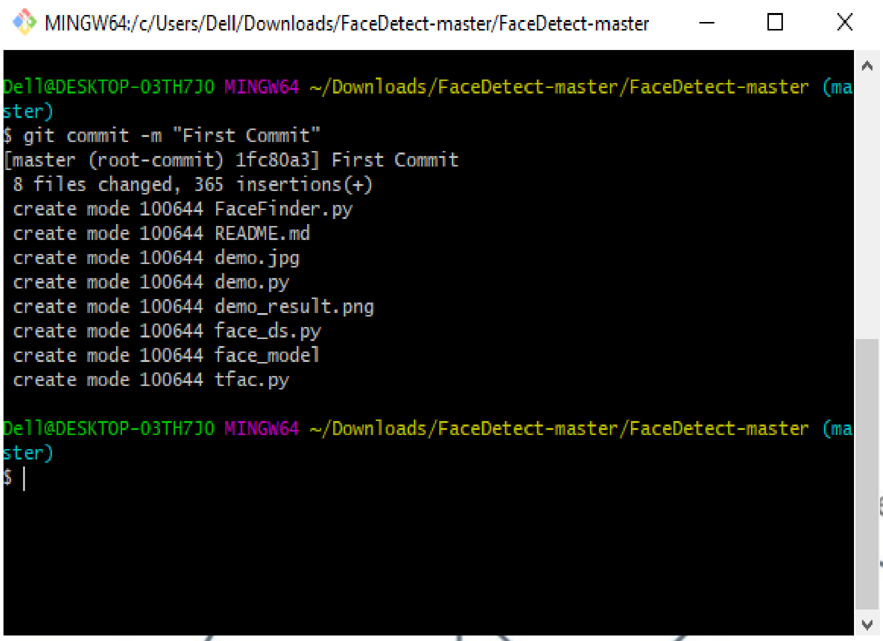


```
Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master/FaceDetect-master (ma
ster)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   FaceFinder.py
        new file:   README.md
        new file:   demo.jpg
        new file:   demo.py
        new file:   demo_result.png
        new file:   face_ds.py
        new file:   face_model
        new file:   tfac.py


Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master/FaceDetect-master (ma
ster)
$
```

6. Commit the files staged in your local repository by writing a commit message.

- You can create a commit message by git commit -m 'your message', which adds the change to the local repository.

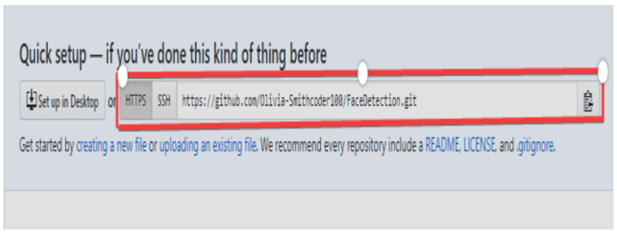
- gits commit uses '-m' as a flag for a message to set the commits with the content where the full description is included, and a message is written in an imperative sentence up to 50 characters long and defining "what was changed", and "why was the change made".



```
MINGW64:/c/Users/Dell/Downloads/FaceDetect-master/FaceDetect-master

Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master/FaceDetect-master (ma
ster)
$ git commit -m "First Commit"
[master (root-commit) 1fc80a3] First Commit
 8 files changed, 365 insertions(+)
 create mode 100644 FaceFinder.py
 create mode 100644 README.md
 create mode 100644 demo.jpg
 create mode 100644 demo.py
 create mode 100644 demo_result.png
 create mode 100644 face_ds.py
 create mode 100644 face_model
 create mode 100644 tfac.py

Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master/FaceDetect-master (ma
ster)
$ |
```

7. Copy your remote repository's URL from GitHub.

- The HTTPS or URL is copied from the given GitHub account, which is the place of the remote repository.



Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH https://github.com/Olivia-Smithcoder100/FaceDetection.git

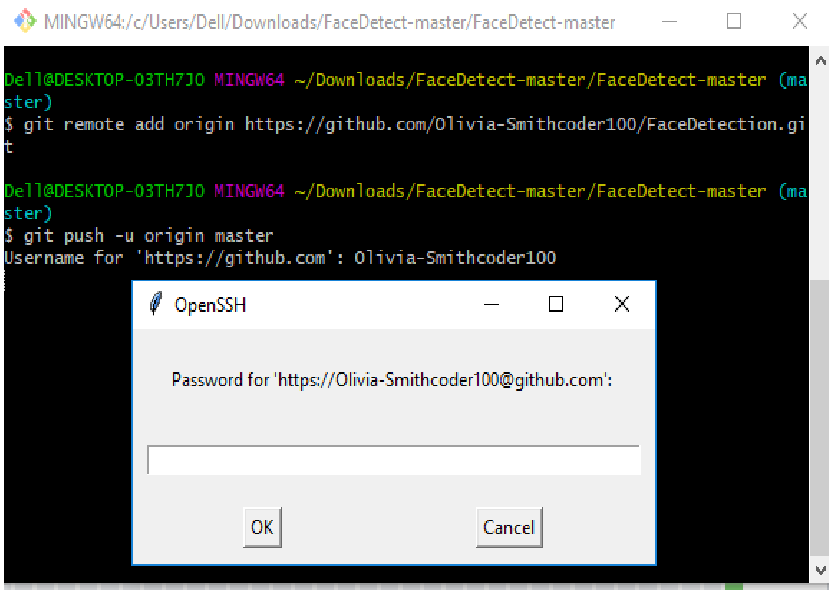
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

8. Add the URL copied, which is your remote repository to where your local content from your repository is pushed.

- git remote add origin 'your_url_name'

9. Push the code in your local repository to GitHub

- git push -u origin master is used for pushing local content to GitHub.
- In the code, the origin is your default remote repository name and '-u' flag is upstream, which is equivalent to '-set-upstream.' and the master is the branch, name.upstream is the repository that we have cloned the project.
- Fill in your GitHub username and password.



MINGW64:/c/Users/Dell/Downloads/FaceDetect-master/FaceDetect-master

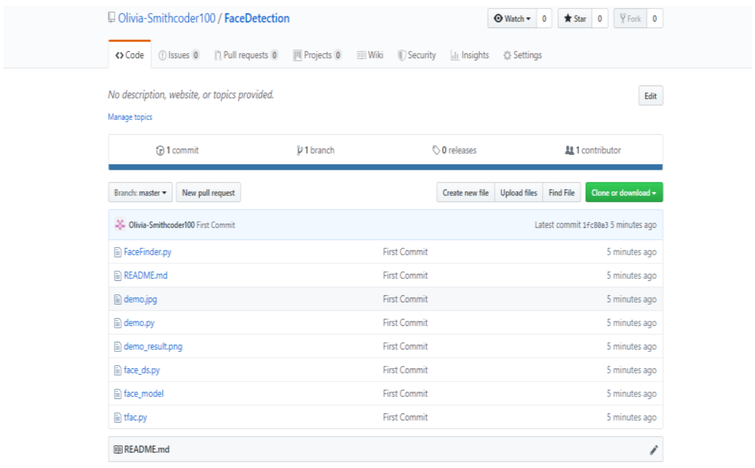
```
Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master/FaceDetect-master (ma
ster)
$ git remote add origin https://github.com/Olivia-Smithcoder100/FaceDetection.gi
t

Dell@DESKTOP-03TH7J0 MINGW64 ~/Downloads/FaceDetect-master/FaceDetect-master (ma
ster)
$ git push -u origin master
Username for 'https://github.com': Olivia-Smithcoder100
```

OpenSSH

Password for 'https://Olivia-Smithcoder100@github.com':

10. View your files in your repository hosted on GitHub.

- You can finally see the file hosted on GitHub.



Olivia-Smithcoder100 / FaceDetection

No description, website, or topics provided.

1 commit | 1 branch | 0 releases | 1 contributor

Olivia-Smithcoder100 First Commit — Latest commit 1fc88a3 5 minutes ago

FaceFinder.py	First Commit	5 minutes ago
README.md	First Commit	5 minutes ago
demo.jpg	First Commit	5 minutes ago
demo.py	First Commit	5 minutes ago
demo_result.png	First Commit	5 minutes ago
face_ds.py	First Commit	5 minutes ago
face_model	First Commit	5 minutes ago
tfac.py	First Commit	5 minutes ago

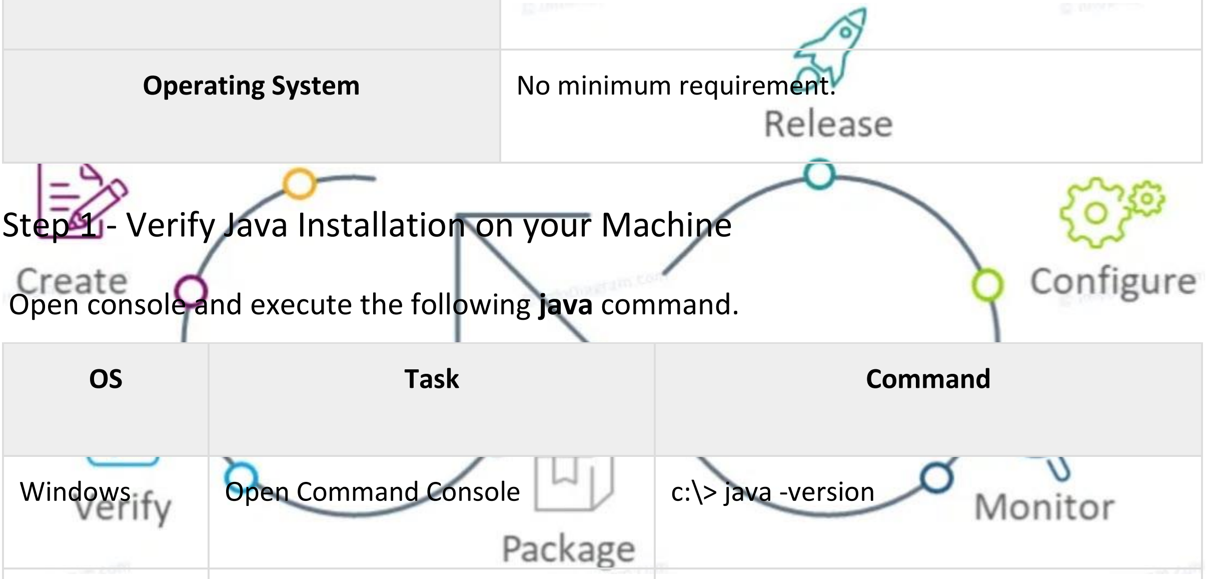
README.md

Steps for maven installation

Maven is a Java based tool, so the very first requirement is to have JDK installed on your machine.

System Requirement

JDK	1.7 or above.
Memory	No minimum requirement.
Disk Space	No minimum requirement.
Operating System	No minimum requirement.



Step 1 - Verify Java Installation on your Machine

Open console and execute the following **java** command.

OS	Task	Command
Windows	Open Command Console	c:\> java -version
Linux	Open Command Terminal	$ java -version
Mac	Open Terminal	machine:~ joseph$ java -version

Let's verify the output for all the operating systems –

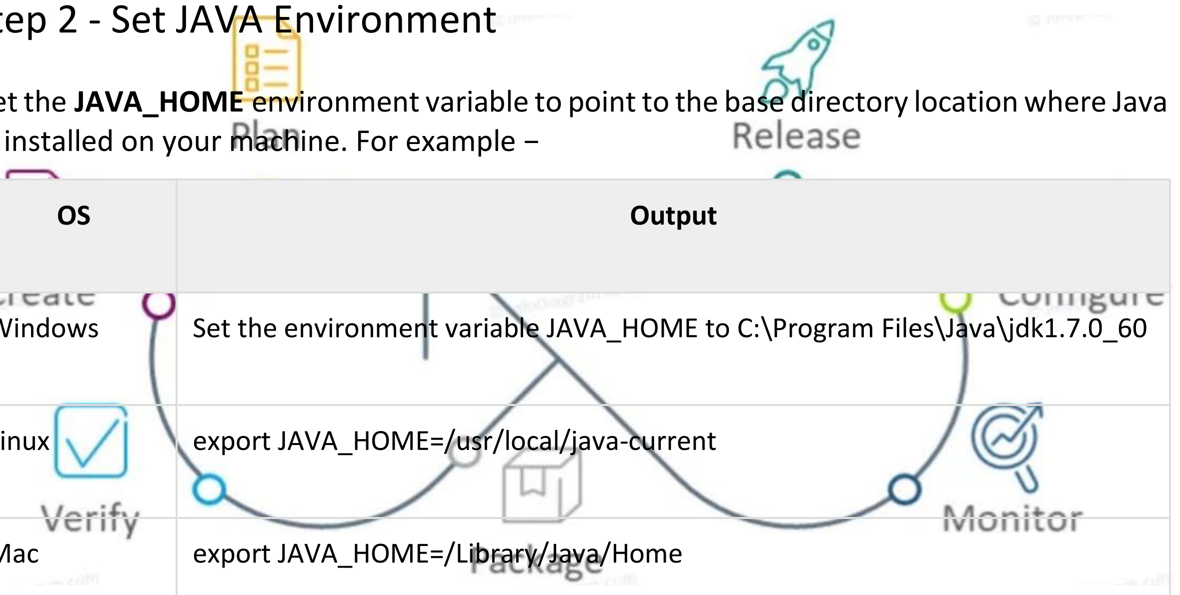
OS	Output
Windows	java version "1.7.0_60" Java(TM) SE Runtime Environment (build 1.7.0_60-b19) Java HotSpot(TM) 64-Bit Server VM (build 24.60-b09, mixed mode)

Linux	java version "1.7.0_60" Java(TM) SE Runtime Environment (build 1.7.0_60-b19) Java HotSpot(TM) 64-Bit Server VM (build 24.60-b09, mixed mode)
Mac	java version "1.7.0_60" Java(TM) SE Runtime Environment (build 1.7.0_60-b19) Java HotSpot(TM) 64-Bit Server VM (build 24.60-b09, mixed mode)

If you do not have Java installed, install the Java Software Development Kit (SDK) from https://www.oracle.com/technetwork/java/javase/downloads/index.html. We are assuming Java 1.7.0.60 as installed version for this tutorial.

Step 2 - Set JAVA Environment

Set the **JAVA_HOME** environment variable to point to the base directory location where Java is installed on your machine. For example –



OS	Output
Windows	Set the environment variable JAVA_HOME to C:\Program Files\Java\jdk1.7.0_60
Linux	export JAVA_HOME=/usr/local/java-current
Mac	export JAVA_HOME=/Library/Java/Home

Append Java compiler location to System Path.

OS	Output
Windows	Append the string “;C:\Program Files\Java\jdk1.7.0.60\bin” to the end of the system variable, Path.
Linux	export PATH=$PATH:$JAVA_HOME/bin/
Mac	not required

Verify Java Installation using **java -version** command as explained above.

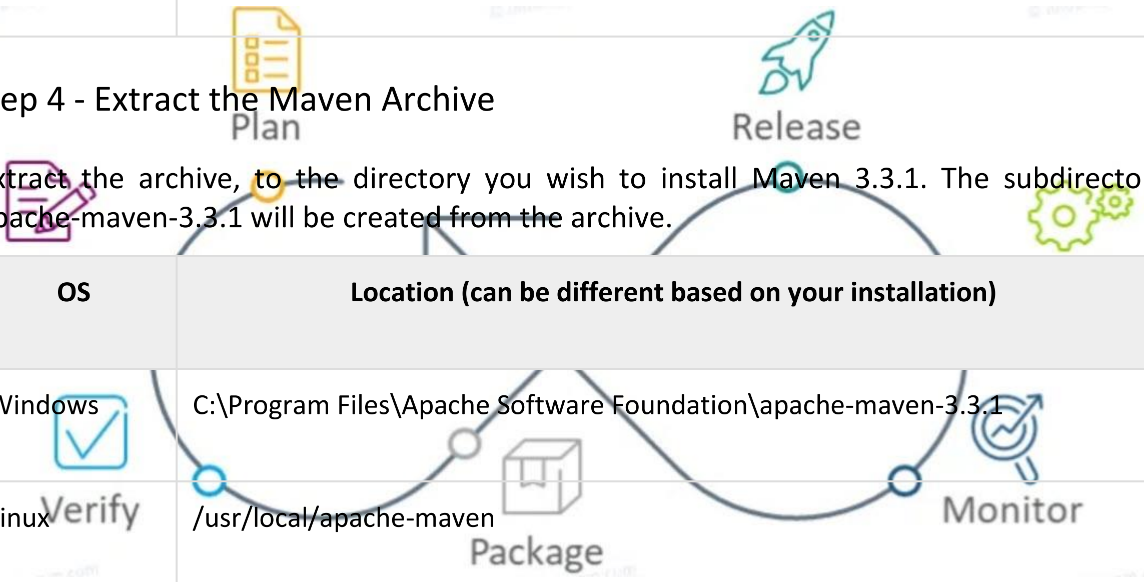
Step 3 - Download Maven Archive

Download Maven 2.2.1 from https://maven.apache.org/download.cgi.

OS	Archive name
Windows	apache-maven-3.3.1-bin.zip
Linux	apache-maven-3.3.1-bin.tar.gz
Mac	apache-maven-3.3.1-bin.tar.gz

Step 4 - Extract the Maven Archive

Extract the archive, to the directory you wish to install Maven 3.3.1. The subdirectory apache-maven-3.3.1 will be created from the archive.



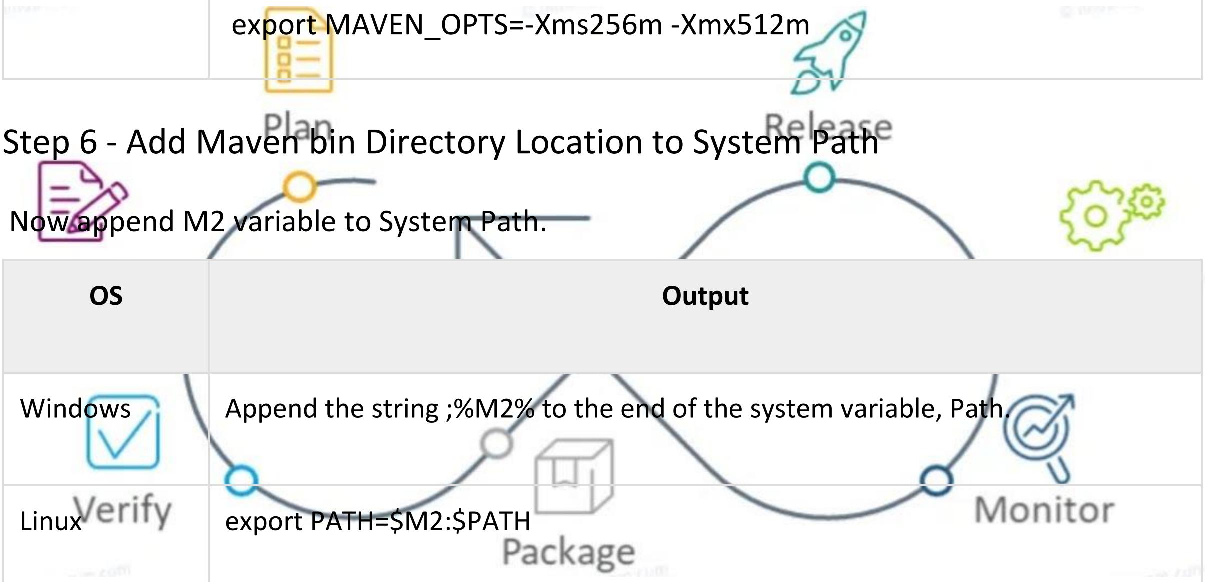
OS	Location (can be different based on your installation)
Windows	C:\Program Files\Apache Software Foundation\apache-maven-3.3.1
Linux	/usr/local/apache-maven
Mac	/usr/local/apache-maven

Step 5 - Set Maven Environment Variables

Add M2_HOME, M2, MAVEN_OPTS to environment variables.

OS	Output
Windows	Set the environment variables using system properties.

	M2_HOME=C:\Program Files\Apache Software Foundation\apache-maven-3.3.1 M2=%M2_HOME%\bin MAVEN_OPTS=-Xms256m -Xmx512m
Linux	Open command terminal and set environment variables. export M2_HOME=/usr/local/apache-maven/apache-maven-3.3.1 export M2=$M2_HOME/bin export MAVEN_OPTS=-Xms256m -Xmx512m
Mac	Open command terminal and set environment variables. export M2_HOME=/usr/local/apache-maven/apache-maven-3.3.1 export M2=$M2_HOME/bin export MAVEN_OPTS=-Xms256m -Xmx512m



Step 6 - Add Maven bin Directory Location to System Path

Now append M2 variable to System Path.

OS	Output
Windows	Append the string ;%M2% to the end of the system variable, Path.
Linux	export PATH=$M2:$PATH
Mac	export PATH=$M2:$PATH

Step 7 - Verify Maven Installation

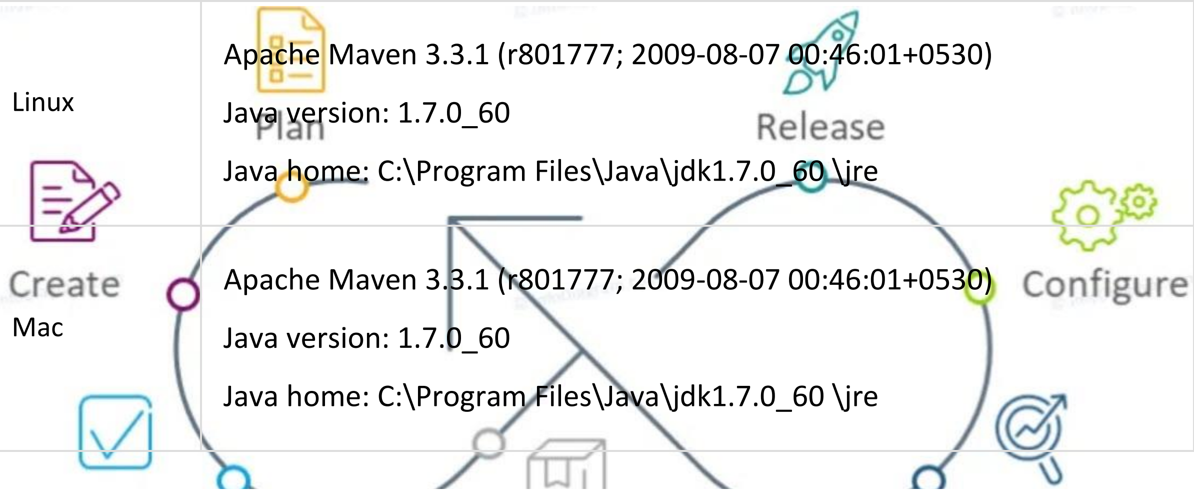
Now open console and execute the following **mvn** command.

OS	Task	Command
Windows	Open Command Console	c:\> mvn --version

Linux	Open Command Terminal	$ mvn --version
Mac	Open Terminal	machine:~ joseph$ mvn --version

Finally, verify the output of the above commands, which should be as follows −

OS	Output
Windows	Apache Maven 3.3.1 (r801777; 2009-08-07 00:46:01+0530) Java version: 1.7.0_60 Java home: C:\Program Files\Java\jdk1.7.0_60 \jre
Linux	Apache Maven 3.3.1 (r801777; 2009-08-07 00:46:01+0530) Java version: 1.7.0_60 Java home: C:\Program Files\Java\jdk1.7.0_60 \jre
Mac	Apache Maven 3.3.1 (r801777; 2009-08-07 00:46:01+0530) Java version: 1.7.0_60 Java home: C:\Program Files\Java\jdk1.7.0_60 \jre



Steps for Gradle installation

Prerequisites

Gradle runs on all major operating systems and requires only a Java JDK version 8 or higher to be installed. To check, run `java -version`:

```
$ java -version
java version "1.8.0_121"
```

Installing manually

Step 1. Download the latest Gradle distribution

The current Gradle release is version 7.2, released on 17 Aug 2021. The distribution zip file comes in two flavors:

- Binary-only
- Complete, with docs and sources

If in doubt, choose the binary-only version and browse docs and sources online.

Need to work with an older version? See the releases page.

Step 2. Unpack the distribution

Linux & MacOS users

Unzip the distribution zip file in the directory of your choosing, e.g.:

```
$ mkdir /opt/gradle
$ unzip -d /opt/gradle gradle-7.2-bin.zip
$ ls /opt/gradle/gradle-7.2
LICENSE  NOTICE  bin  getting-started.html  init.d  lib  media
```

Microsoft Windows users

Create a new directory `C:\Gradle` with File Explorer.

Open a second File Explorer window and go to the directory where the Gradle distribution was downloaded. Double-click the ZIP archive to expose the content. Drag the content folder `gradle-7.2` to your newly created `C:\Gradle` folder.

Alternatively you can unpack the Gradle distribution ZIP into `C:\Gradle` using an archiver tool of your choice.

Step 3. Configure your system environment

Linux & MacOS users

Configure your `PATH` environment variable to include the `bin` directory of the unzipped distribution, e.g.:

```
 $ export PATH=$PATH:/opt/gradle/gradle-7.2/bin
```

Microsoft Windows users

In File Explorer right-click on the `This PC` (or `Computer`) icon, then click `Properties` -> `Advanced System Settings` -> `Environmental Variables`.

Under `System Variables` select `Path`, then click `Edit`. Add an entry for `C:\Gradle\gradle-7.2\bin`. Click OK to save.

Step 4. Verify your installation

Open a console (or a Windows command prompt) and run `gradle -v` to run gradle and display the version, e.g.:

```
$ gradle -v
 Gradle 7.2
```

Steps installation of SonarQube

GitHub integration.

1. Login to SonarQube
2. SonarQube Dashboard
3. Creating a Project.
4. Configuring project for Maven
5. Running SonarQube on Source Code
6. SonarQube Analysis Overview
7. Project Analysis
8. Bugs
9. Code Smells
10. Issue Highlights in the source code

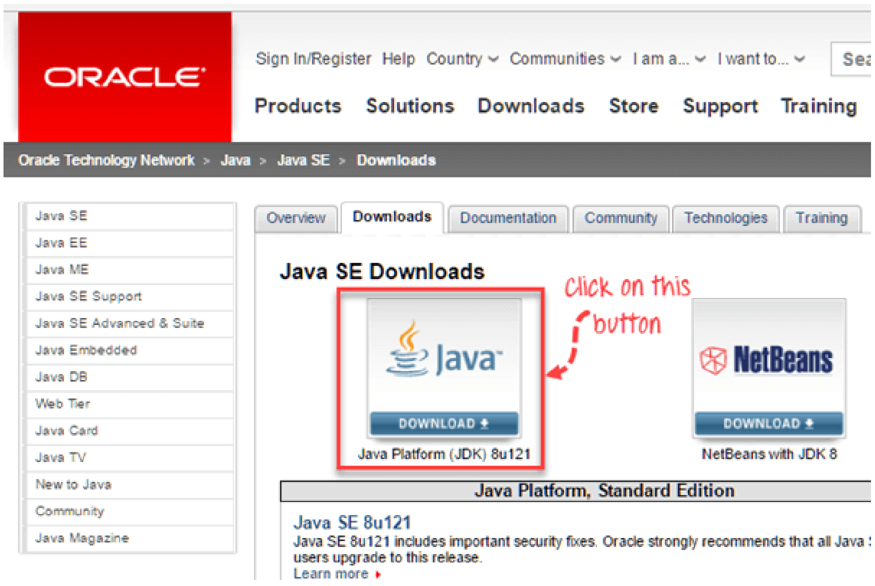
GitHub Integration by Cloud.

1. Import your GitHub repositories
2. Analyze projects with GitHub Actions
3. Report your Quality Gate status to your branches and pull requests
4. Authenticate with GitHub
5. Analyzing projects with GitHub Actions

Steps for Selenium Installation

Step 1 – Install Java on your computer

Download and install the **Java Software Development Kit (JDK)**.

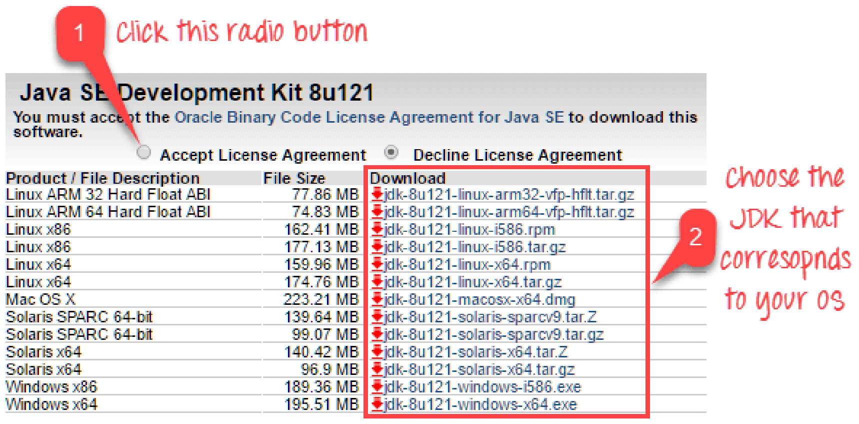

ORACLE
Sign In/Register Help Country Communities I am a... I want to... Sea
Products Solutions Downloads Store Support Training
Oracle Technology Network > Java > Java SE > Downloads
Java SE
Java EE
Java ME
Java SE Support
Java SE Advanced & Suite
Java Embedded
Java DB
Web Tier
Java Card
Java TV
New to Java
Community
Java Magazine
Overview Downloads Documentation Community Technologies Training
Java SE Downloads
Click on this button
Java
DOWNLOAD
Java Platform (JDK) 8u121
NetBeans
DOWNLOAD
NetBeans with JDK 8
Java Platform, Standard Edition

Java SE 8u121

Java SE 8u121 includes important security fixes. Oracle strongly recommends that all Java users upgrade to this release.

Learn more

Next –



Java SE Development Kit 8u121

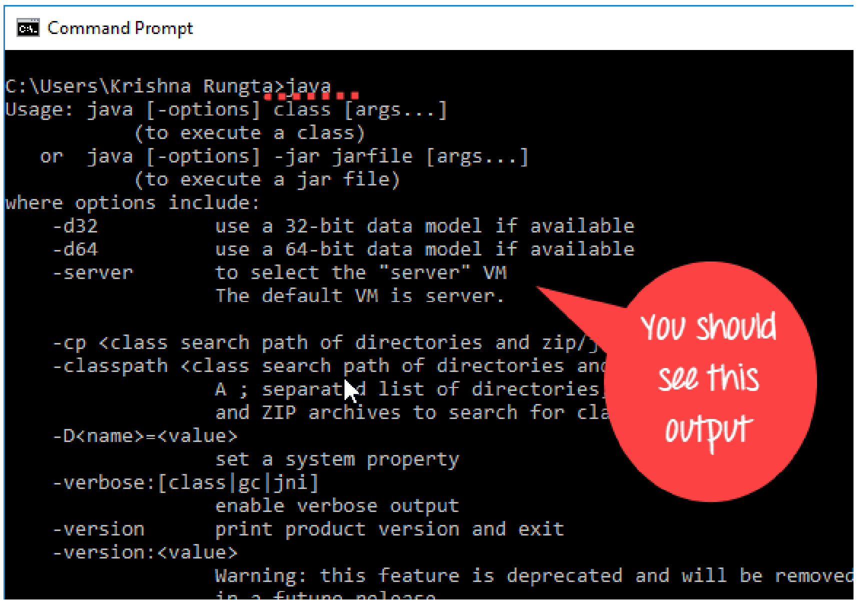
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Accept License Agreement [✓ Decline License Agreement]

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.86 MB	jdk-8u121-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.83 MB	jdk-8u121-linux-arm64-vfp-hflt.tar.gz
Linux x86	162.41 MB	jdk-8u121-linux-i586.rpm
Linux x86	177.13 MB	jdk-8u121-linux-i586.tar.gz
Linux x64	159.96 MB	jdk-8u121-linux-x64.rpm
Linux x64	174.76 MB	jdk-8u121-linux-x64.tar.gz
Mac OS X	223.21 MB	jdk-8u121-macosx-x64.dmg
Solaris SPARC 64-bit	139.64 MB	jdk-8u121-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.07 MB	jdk-8u121-solaris-sparcv9.tar.gz
Solaris x64	140.42 MB	jdk-8u121-solaris-x64.tar.Z
Solaris x64	96.9 MB	jdk-8u121-solaris-x64.tar.gz
Windows x86	189.36 MB	jdk-8u121-windows-i586.exe
Windows x64	195.51 MB	jdk-8u121-windows-x64.exe

This JDK version comes bundled with Java Runtime Environment (JRE), so you do not need to download and install the JRE separately.

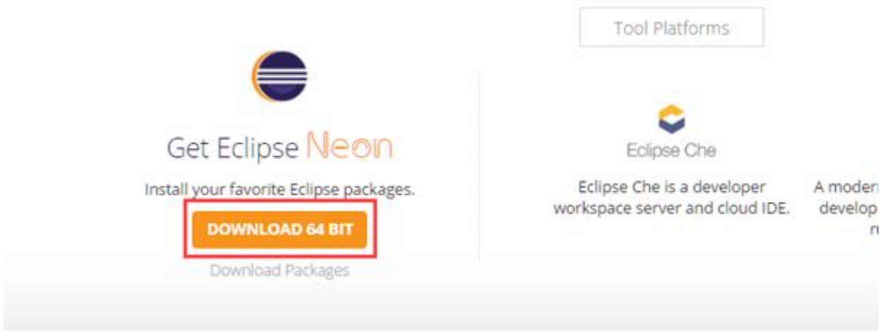
Once installation is complete, open command prompt and type “java”. If you see the following screen you are good to move to the next step



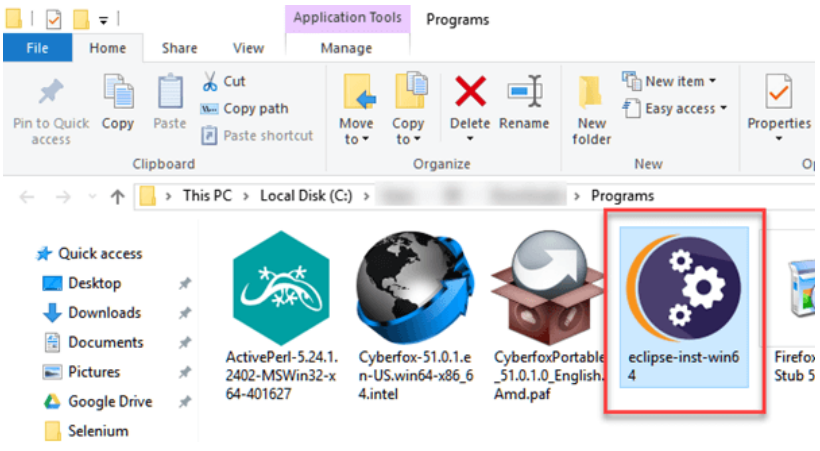
```
C:\Users\Krishna Rungta>java
Usage: java [-options] class [args...]
           (to execute a class)
   or  java [-options] -jar jarfile [args...]
           (to execute a jar file)
where options include:
    -d32          use a 32-bit data model if available
    -d64          use a 64-bit data model if available
    -server       to select the "server" VM
                  The default VM is server.

    -cp <class search path of directories and zip/j
    -classpath <class search path of directories and
                  A ; separated list of directories,
                  and ZIP archives to search for cla
    -D<name>=<value>
                  set a system property
    -verbose:[class|gc|jni]
                  enable verbose output
    -version      print product version and exit
    -version:<value>
                  Warning: this feature is deprecated and will be removed
```

Step 2 – Install Eclipse IDE

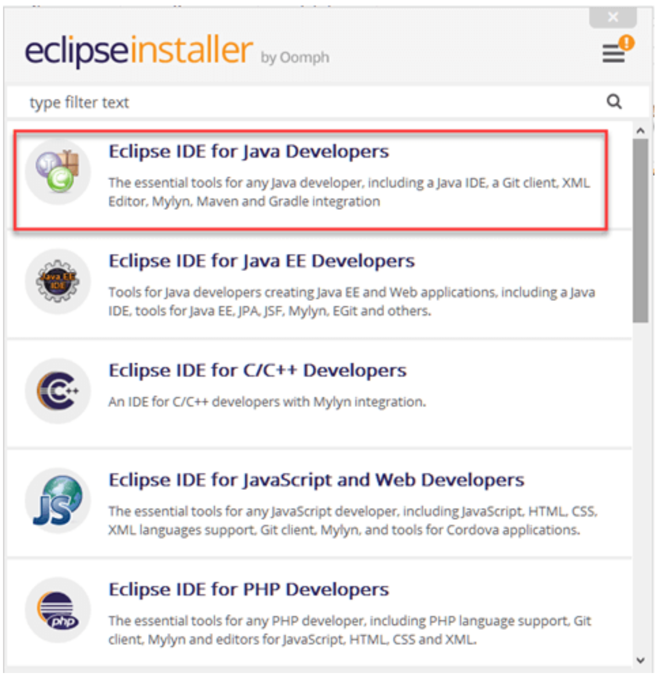
Download latest version of **"Eclipse IDE for Java Developers"** here. Be sure to choose correctly between Windows 32 Bit and 64 Bit versions.


Tool Platforms
Get Eclipse Neon
Install your favorite Eclipse packages.
DOWNLOAD 64 BIT
Download Packages
Eclipse Che
Eclipse Che is a developer workspace server and cloud IDE.

You should be able to download an exe file named "eclipse-inst-win64" for Setup.



Double-click on file to Install the Eclipse. A new window will open. Click Eclipse IDE for Java Developers.



eclipseinstaller by Oomph

type filter text

Eclipse IDE for Java Developers
The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven and Gradle integration

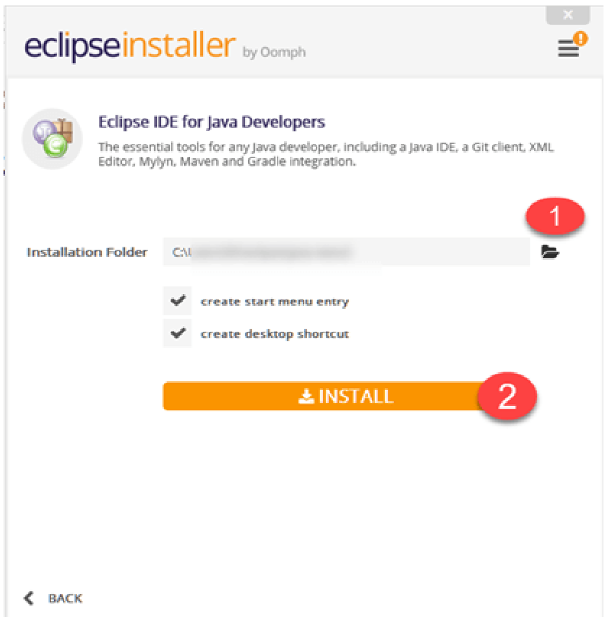
Eclipse IDE for Java EE Developers
Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn, EGit and others.

Eclipse IDE for C/C++ Developers
An IDE for C/C++ developers with Mylyn integration.

Eclipse IDE for JavaScript and Web Developers
The essential tools for any JavaScript developer, including JavaScript, HTML, CSS, XML languages support, Git client, Mylyn, and tools for Cordova applications.

Eclipse IDE for PHP Developers
The essential tools for any PHP developer, including PHP language support, Git client, Mylyn and editors for JavaScript, HTML, CSS and XML.

After that, a new window will open which click button marked 1 and change path to “C:\eclipse”. Post that Click on Install button marked 2



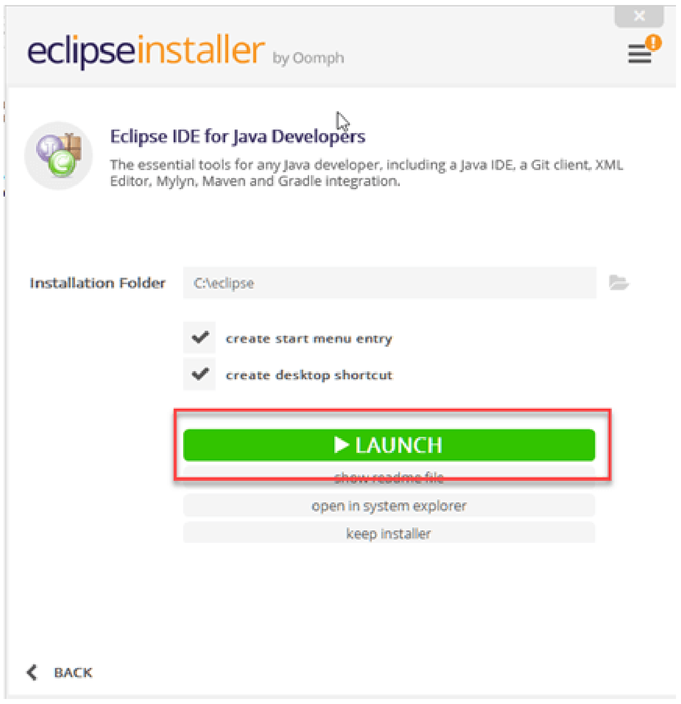
eclipseinstaller by Oomph

Eclipse IDE for Java Developers
The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven and Gradle integration.

Installation Folder: C:\ [redacted]

✓ create start menu entry
✓ create desktop shortcut

After successful completion of the installation procedure, a window will appear. On that window click on Launch



Eclipse IDE for Java Developers

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven and Gradle integration.

Installation Folder: C:\eclipse

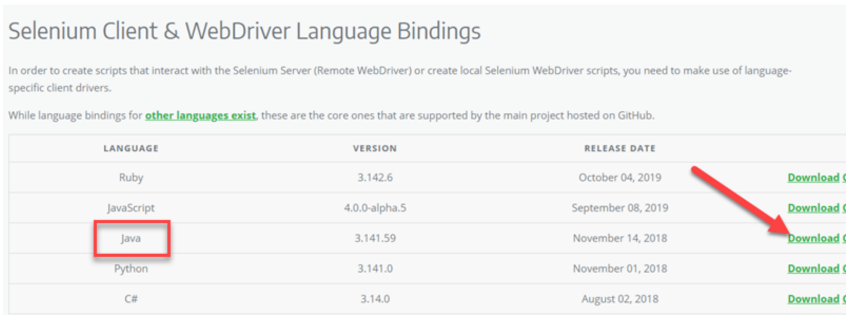
✓ create start menu entry

✓ create desktop shortcut

This will start eclipse neon IDE for you.

Step 3 – Download the Selenium Java Client Driver

You can download **Selenium Webdriver for Java Client Driver** here. You will find client drivers for other languages there, but only choose the one for Java.



Selenium Client & WebDriver Language Bindings

In order to create scripts that interact with the Selenium Server (Remote WebDriver) or create local Selenium WebDriver scripts, you need to make use of language-specific client drivers.

While language bindings for other languages exist, these are the core ones that are supported by the main project hosted on GitHub.

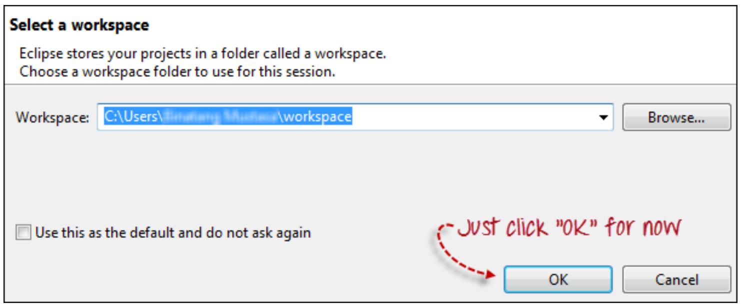
LANGUAGE	VERSION	RELEASE DATE	
Ruby	3.142.6	October 04, 2019	Download
JavaScript	4.0.0-alpha.5	September 08, 2019	Download
Java	3.141.59	November 14, 2018	Download
Python	3.141.0	November 01, 2018	Download
C#	3.14.0	August 02, 2018	Download

This download comes as a ZIP file named "selenium-3.14.0.zip". For simplicity of Selenium installation on Windows 10, extract the contents of this ZIP file on your C drive so that you would have the directory "C:\selenium-3.14.0\". This directory contains all the JAR files that we would later import on Eclipse for Selenium setup.

Step 4 – Configure Eclipse IDE with WebDriver

1. Launch the "eclipse.exe" file inside the "eclipse" folder that we extracted in step 2. If you followed step 2 correctly, the executable should be located on C:\eclipse\eclipse.exe.

2. When asked to select for a workspace, just accept the default location.



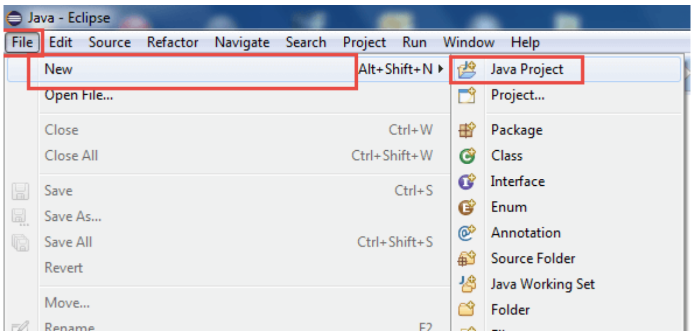
Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: C:\Users\[redacted]\workspace

☐ Use this as the default and do not ask again

3. Create a new project through File > New > Java Project. Name the project as "newproject".

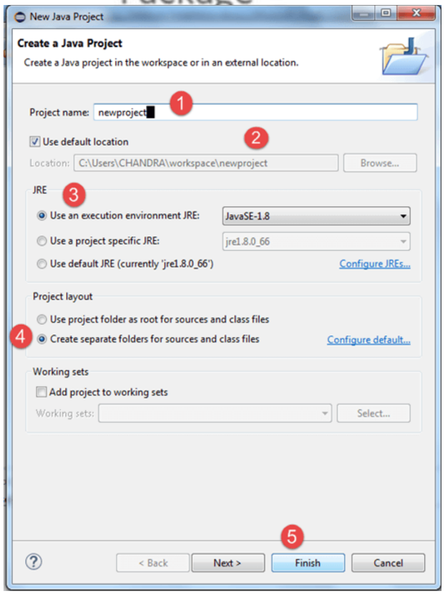


A new pop-up window will open enter details as follow

Create Configure

1. Project Name
2. Location to save project
3. Select an execution JRE
4. Select layout project option
5. Click on Finish button

Package



New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: newproject

☑ Use default location

Location: C:\Users\CHANDRA\workspace\newproject

JRE

(●) Use an execution environment JRE: JavaSE-1.8

( ) Use a project specific JRE: jre1.8.0_66

( ) Use default JRE (currently 'jre1.8.0_66')

Project layout

( ) Use project folder as root for sources and class files

(●) Create separate folders for sources and class files

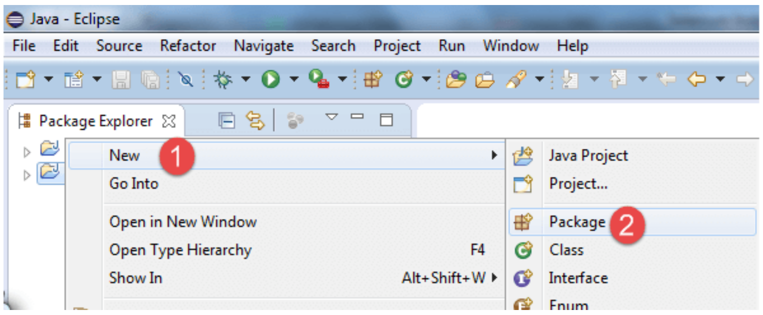
Working sets

☐ Add project to working sets

Working sets:

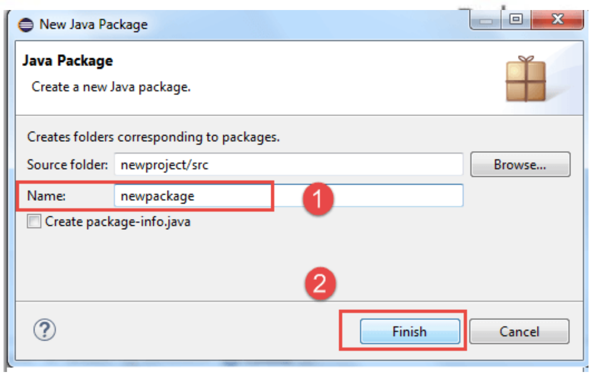
4. In this step,

1. Right-click on the newly created project and
2. Select New > Package, and name that package as “newpackage”.



A pop-up window will open to name the package,

1. Enter the name of the package
2. Click on Finish button


New Java Package

Java Package

Create a new Java package.

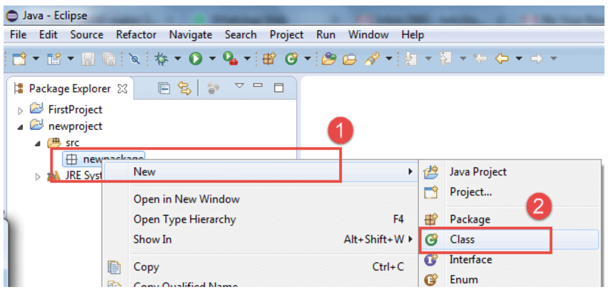
Creates folders corresponding to packages.

Source folder: newproject/src

Name: newpackage

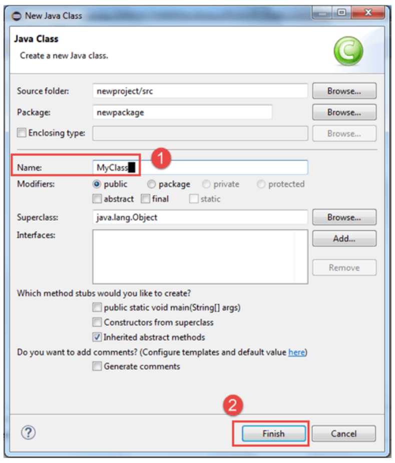
Create package-info.java

5. Create a new Java class under newpackage by right-clicking on it and then selecting- New > Class, and then name it as “MyClass”. Your Eclipse IDE should look like the image below.



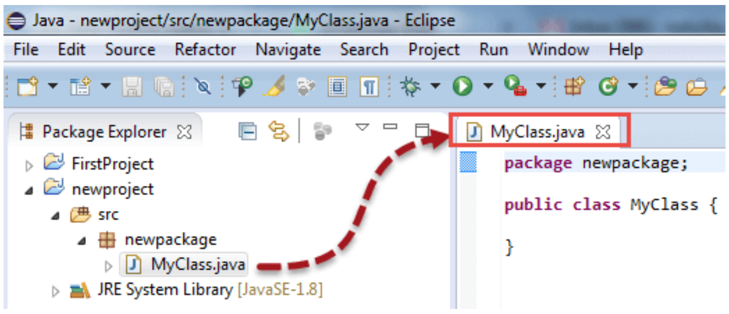
When you click on Class, a pop-up window will open, enter details as

1. Name of the class
2. Click on Finish button



Source folder: newproject/src
Package: newpackage
Enclosing type:
Name: MyClass
Modifiers: [✓ public] package private protected
abstract final static
Superclass: java.lang.Object
Interfaces:
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
[✓ Inherited abstract methods]
Do you want to add comments? (Configure templates and default value here)
Generate comments

This is how it looks like after creating class.

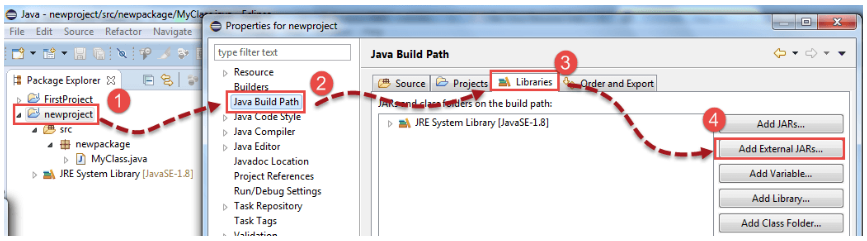


```
package newpackage;

public class MyClass {

}
```

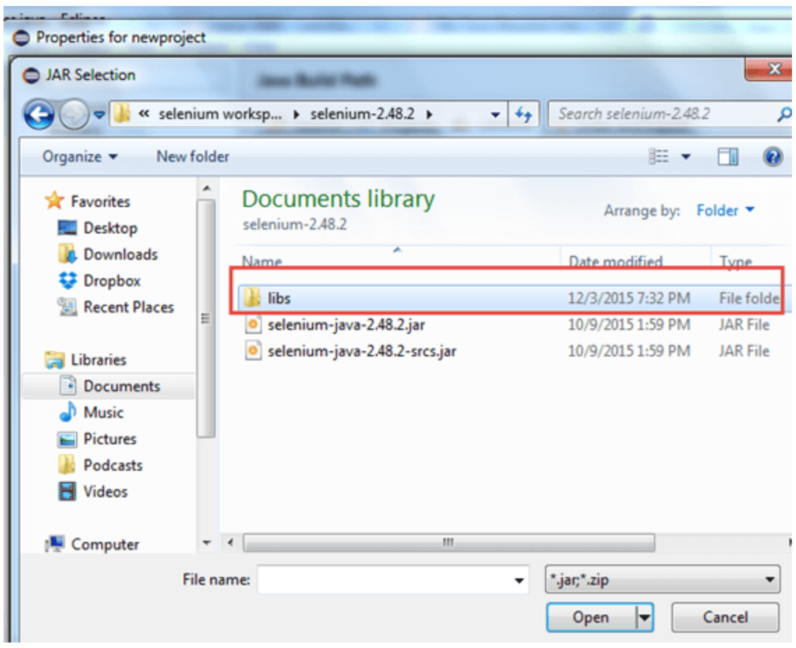
Now selenium WebDriver's into Java Build Path

In this step,

1. Right-click on “newproject” and select **Properties**.
2. On the Properties dialog, click on “Java Build Path”.
3. Click on the **Libraries** tab, and then
4. Click on “Add External JARs..”



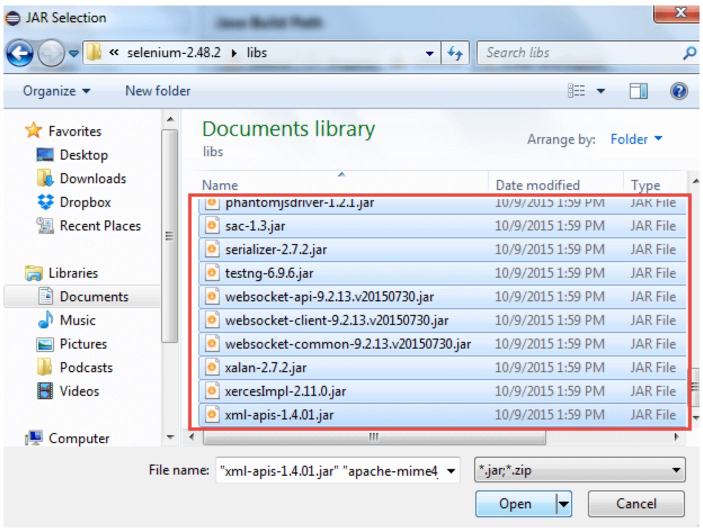
When you click on “Add External JARs..” It will open a pop-up window. Select the JAR files you want to add.



Name	Date modified	Type
libs	12/3/2015 7:32 PM	File folde
selenium-java-2.48.2.jar	10/9/2015 1:59 PM	JAR File
selenium-java-2.48.2-srcs.jar	10/9/2015 1:59 PM	JAR File

After selecting jar files, click on OK button.

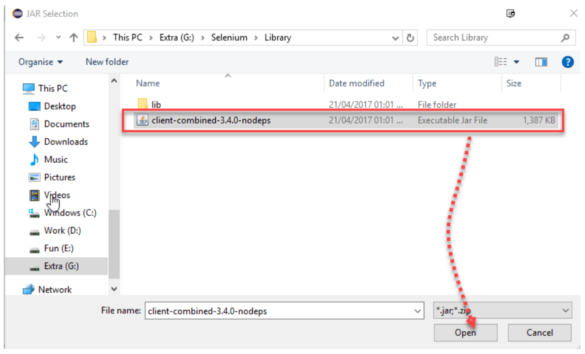
Select all files inside the lib folder.



Name	Date modified	Type
phantomjsdriver-1.2.1.jar	10/9/2015 1:59 PM	JAR File
sac-1.3.jar	10/9/2015 1:59 PM	JAR File
serializer-2.7.2.jar	10/9/2015 1:59 PM	JAR File
testng-6.9.6.jar	10/9/2015 1:59 PM	JAR File
websocket-api-9.2.13.v20150730.jar	10/9/2015 1:59 PM	JAR File
websocket-client-9.2.13.v20150730.jar	10/9/2015 1:59 PM	JAR File
websocket-common-9.2.13.v20150730.jar	10/9/2015 1:59 PM	JAR File
xalan-2.7.2.jar	10/9/2015 1:59 PM	JAR File
xercesImpl-2.11.0.jar	10/9/2015 1:59 PM	JAR File
xml-apis-1.4.01.jar	10/9/2015 1:59 PM	JAR File

File name: "xml-apis-1.4.01.jar" "apache-mime4

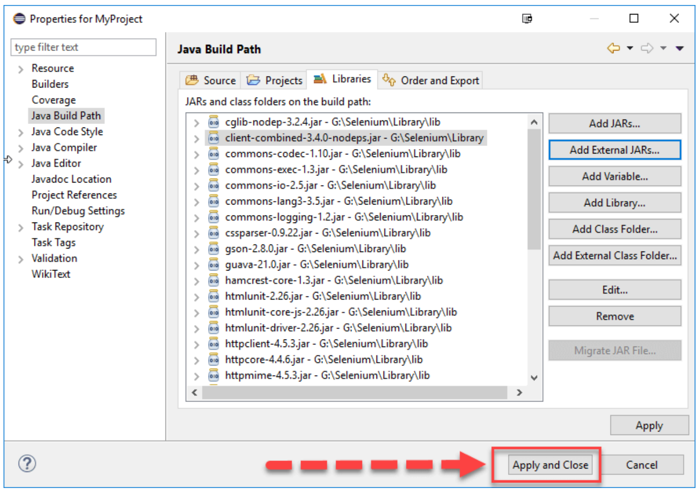
Select files outside lib folder



Name	Date modified	Type	Size
lib	21/04/2017 01:01 ...	File folder	
client-combined-3.4.0-nodeps	21/04/2017 01:01 ...	Executable Jar File	1,387 KB

File name: client-combined-3.4.0-nodeps

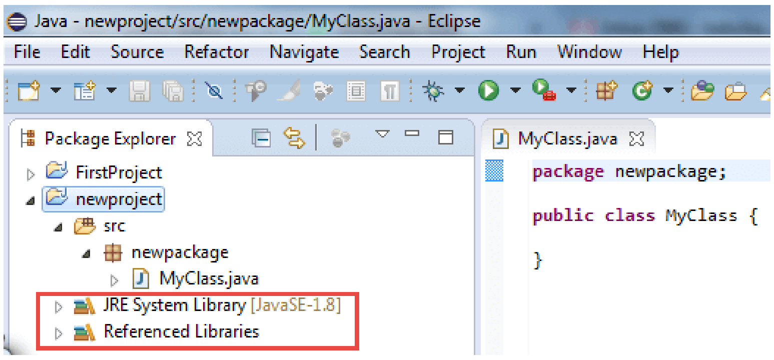
Once done, click “Apply and Close” button



JARs and class folders on the build path:

```
cglib-nodep-3.2.4.jar - G:\Selenium\Library\lib
client-combined-3.4.0-nodeps.jar - G:\Selenium\Library
commons-codec-1.10.jar - G:\Selenium\Library\lib
commons-exec-1.3.jar - G:\Selenium\Library\lib
commons-io-2.5.jar - G:\Selenium\Library\lib
commons-lang3-3.5.jar - G:\Selenium\Library\lib
commons-logging-1.2.jar - G:\Selenium\Library\lib
cssparser-0.9.22.jar - G:\Selenium\Library\lib
gson-2.8.0.jar - G:\Selenium\Library\lib
guava-21.0.jar - G:\Selenium\Library\lib
hamcrest-core-1.3.jar - G:\Selenium\Library\lib
htmlunit-2.26.jar - G:\Selenium\Library\lib
htmlunit-core-js-2.26.jar - G:\Selenium\Library\lib
htmlunit-driver-2.26.jar - G:\Selenium\Library\lib
httpclient-4.5.3.jar - G:\Selenium\Library\lib
httpcore-4.4.6.jar - G:\Selenium\Library\lib
httpmime-4.5.3.jar - G:\Selenium\Library\lib
```

6. Add all the JAR files inside and outside the “libs” folder. Your Properties dialog should now look similar to the image below.



```
package newpackage;

public class MyClass {

}
```

7. Finally, click OK and we are done importing Selenium libraries into our project.

Steps for Jenkins and Docker Installation

Installing Docker

To install Docker on your operating system, follow "prerequisits" section of the Guided Tour page

As an alternative solution you can visit the Dockerhub and select the **Docker Community Edition** suitable for your operating system or cloud service. Follow the installation instructions on their website.

There are several Docker images of Jenkins available.

The recommended Docker image to use is the Official jenkins/jenkins image (from the Docker Hub repository). This image contains the current Long-Term Support (LTS) release of Jenkins (which is production-ready). However this image doesn't have docker CLI inside it and is not bundled with frequently used Blue Ocean plugins and features. This means that if you want to use the full power of Jenkins and Docker you may want to go through described below installation process.

On macOS and Linux

1. Open up a terminal window.
2. Create a bridge network in Docker using the following docker network create command:
   docker network create jenkins
3. In order to execute Docker commands inside Jenkins nodes, download and run the docker:dind Docker image using the following docker run command:
4. docker run \
5. --name jenkins-docker \
6. --rm \
7. --detach \
8. --privileged \
9. --network jenkins \
10. --network-alias docker \
11. --env DOCKER_TLS_CERTDIR=/certs \
12. --volume jenkins-docker-certs:/certs/client \
13. --volume jenkins-data:/var/jenkins_home \
14. --publish 2376:2376 \
15. docker:dind \
    --storage-driver overlay2

Note: If copying and pasting the command snippet above does not work, try copying and pasting this annotation-free version here:

```
docker run --name jenkins-docker --rm --detach \
  --privileged --network jenkins --network-alias docker \
  --env DOCKER_TLS_CERTDIR=/certs \
  --volume jenkins-docker-certs:/certs/client \
  --volume jenkins-data:/var/jenkins_home \
  --publish 2376:2376 docker:dind --storage-driver overlay2
```

16. Customise official Jenkins Docker image, by executing below two steps:
    a. Create Dockerfile with the following content:
    b. `FROM jenkins/jenkins:2.319.1-jdk11`
    c. `USER root`
    d. `RUN apt-get update && apt-get install -y lsb-release`

```
e. RUN curl -fsSLo /usr/share/keyrings/docker-archive-keyring.asc \
f.   https://download.docker.com/linux/debian/gpg
g. RUN echo "deb [arch=$(dpkg --print-architecture) \
h.   signed-by=/usr/share/keyrings/docker-archive-keyring.asc] \
i.   https://download.docker.com/linux/debian \
j.   $(lsb_release -cs) stable" > /etc/apt/sources.list.d/docker.list
k. RUN apt-get update && apt-get install -y docker-ce-cli
l. USER jenkins
   RUN jenkins-plugin-cli --plugins "blueocean:1.25.2 docker-
   workflow:1.26"
```

m. Build a new docker image from this Dockerfile and assign the image a meaningful name, e.g. "myjenkins-blueocean:1.1":

```
docker build -t myjenkins-blueocean:1.1 .
```

Keep in mind that the process described above will automatically download the official Jenkins Docker image if this hasn't been done before.

17. Run your own myjenkins-blueocean:1.1 image as a container in Docker using the following docker run command:

```
18. docker run \
19.   --name jenkins-blueocean \
20.   --rm \
21.   --detach \
22.   --network jenkins \
23.   --env DOCKER_HOST=tcp://docker:2376 \
24.   --env DOCKER_CERT_PATH=/certs/client \
25.   --env DOCKER_TLS_VERIFY=1 \
26.   --publish 8080:8080 \
27.   --publish 50000:50000 \
28.   --volume jenkins-data:/var/jenkins_home \
29.   --volume jenkins-docker-certs:/certs/client:ro \
      myjenkins-blueocean:1.1
```

Note: If copying and pasting the command snippet above does not work, try copying and pasting this annotation-free version here:

```
docker run --name jenkins-blueocean --rm --detach \
  --network jenkins --env DOCKER_HOST=tcp://docker:2376 \
  --env DOCKER_CERT_PATH=/certs/client --env DOCKER_TLS_VERIFY=1 \
  --publish 8080:8080 --publish 50000:50000 \
  --volume jenkins-data:/var/jenkins_home \
  --volume jenkins-docker-certs:/certs/client:ro \
  myjenkins-blueocean:1.1
```

30. Proceed to the Post-installation setup wizard.

On Windows

The Jenkins project provides a Linux container image, not a Windows container image. Be sure that your Docker for Windows installation is configured to run Linux Containers rather than Windows Containers. See the Docker documentation for instructions to switch to Linux containers. Once configured to run Linux Containers, the steps are:

1. Open up a command prompt window and similar to the macOS and Linux instructions above do the following:

2. Create a bridge network in Docker

   docker network create jenkins

3. Run a docker:dind Docker image

```
docker run --name jenkins-docker --rm --detach `
  --privileged --network jenkins --network-alias docker `
  --env DOCKER_TLS_CERTDIR=/certs `
  --volume jenkins-docker-certs:/certs/client `
  --volume jenkins-data:/var/jenkins_home `
  docker:dind
```

9. Customise official Jenkins Docker image, by executing below two steps:
   a. Create Dockerfile with the following content:

```
FROM jenkins/jenkins:2.319.1-jdk11
USER root
RUN apt-get update && apt-get install -y lsb-release
RUN curl -fsSLo /usr/share/keyrings/docker-archive-keyring.asc \
  https://download.docker.com/linux/debian/gpg
RUN echo "deb [arch=$(dpkg --print-architecture) \
  signed-by=/usr/share/keyrings/docker-archive-keyring.asc] \
  https://download.docker.com/linux/debian \
  $(lsb_release -cs) stable" > /etc/apt/sources.list.d/docker.list
RUN apt-get update && apt-get install -y docker-ce-cli
USER jenkins
RUN jenkins-plugin-cli --plugins "blueocean:1.25.2 docker-
workflow:1.26"
```

   m. Build a new docker image from this Dockerfile and assign the image a meaningful name, e.g. "myjenkins-blueocean:1.1":

```
docker build -t myjenkins-blueocean:1.1 .
```

   Keep in mind that the process described above will automatically download the official Jenkins Docker image if this hasn't been done before.

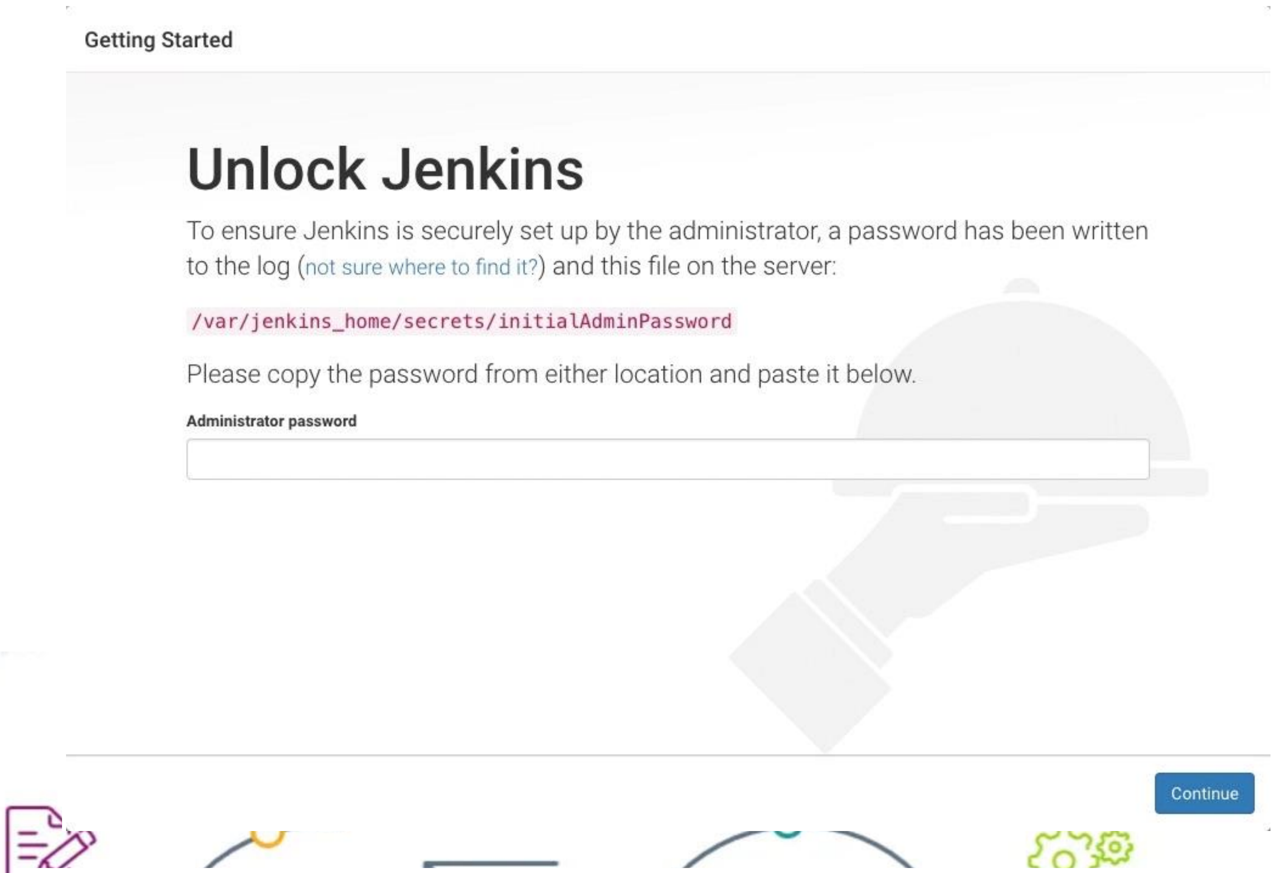
10. Run your own `myjenkins-blueocean:1.1` image as a container in Docker using the following `docker run` command:

```
docker run --name jenkins-blueocean --rm --detach `
  --network jenkins --env DOCKER_HOST=tcp://docker:2376 `
  --env DOCKER_CERT_PATH=/certs/client --env DOCKER_TLS_VERIFY=1 `
  --volume jenkins-data:/var/jenkins_home `
  --volume jenkins-docker-certs:/certs/client:ro `
  --publish 8080:8080 --publish 50000:50000 myjenkins-blueocean:1.1
```

Unlocking Jenkins

When you first access a new Jenkins instance, you are asked to unlock it using an automatically-generated password.

1. Browse to http://localhost:8080 (or whichever port you configured for Jenkins when installing it) and wait until the **Unlock Jenkins** page appears.


Getting Started

Unlock Jenkins

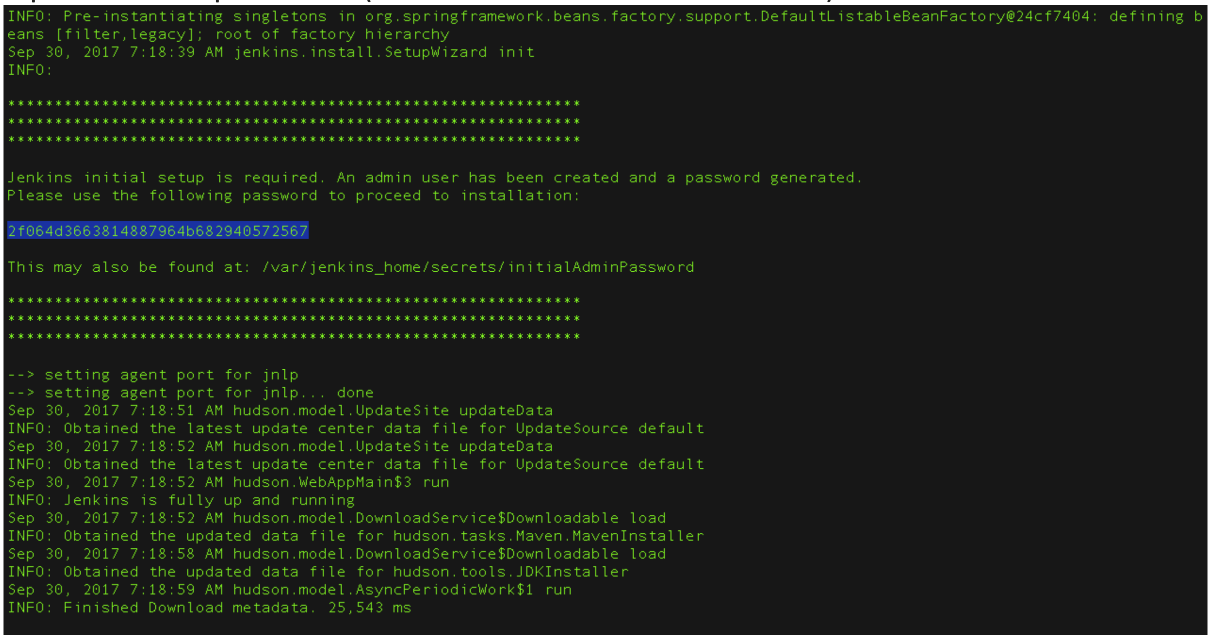
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/jenkins_home/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password

2. From the Jenkins console log output, copy the automatically-generated alphanumeric password (between the 2 sets of asterisks).



```
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@24cf7404: defining b
eans [filter,legacy]; root of factory hierarchy
Sep 30, 2017 7:18:39 AM jenkins.install.SetupWizard init
INFO:

*************************************************************
*************************************************************
*************************************************************

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

2f064d3663814887964b682940572567

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*************************************************************
*************************************************************
*************************************************************

--> setting agent port for jnlp
--> setting agent port for jnlp... done
Sep 30, 2017 7:18:51 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Sep 30, 2017 7:18:52 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Sep 30, 2017 7:18:52 AM hudson.WebAppMain$3 run
INFO: Jenkins is fully up and running
Sep 30, 2017 7:18:52 AM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
Sep 30, 2017 7:18:58 AM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tools.JDKInstaller
Sep 30, 2017 7:18:59 AM hudson.model.AsyncPeriodicWork$1 run
INFO: Finished Download metadata. 25,543 ms
```

Note:

- The command: sudo cat /var/lib/jenkins/secrets/initialAdminPassword will print the password at console.

    - If you are running Jenkins in Docker using the official jenkins/jenkins image you can use sudo docker exec ${CONTAINER_ID or CONTAINER_NAME} cat /var/jenkins_home/secrets/initialAdminPassword to print the password in the console without having to exec into the container.
3. On the **Unlock Jenkins** page, paste this password into the **Administrator password** field and click **Continue**.
   Notes:
    - You can always access the Jenkins console log from the Docker logs (above).
    - The Jenkins console log indicates the location (in the Jenkins home directory) where this password can also be obtained. This password must be entered in the setup wizard on new Jenkins installations before you can access Jenkins's main UI. This password also serves as the default admininstrator account's password (with username "admin") if you happen to skip the subsequent user-creation step in the setup wizard.

Customizing Jenkins with plugins

After unlocking Jenkins, the **Customize Jenkins** page appears. Here you can install any number of useful plugins as part of your initial setup.

Click one of the two options shown:

- **Install suggested plugins** - to install the recommended set of plugins, which are based on most common use cases.
- **Select plugins to install** - to choose which set of plugins to initially install. When you first access the plugin selection page, the suggested plugins are selected by default.

> If you are not sure what plugins you need, choose **Install suggested plugins**. You
> remove) additional Jenkins plugins at a later point in time via the **Manage Jenkin**
> **Plugins** page in Jenkins.

The setup wizard shows the progression of Jenkins being configured and your chosen set of Jenkins plugins being installed. This process may take a few minutes.

Creating the first administrator user

Finally, after customizing Jenkins with plugins, Jenkins asks you to create your first administrator user.

1. When the **Create First Admin User** page appears, specify the details for your administrator user in the respective fields and click **Save and Finish**.

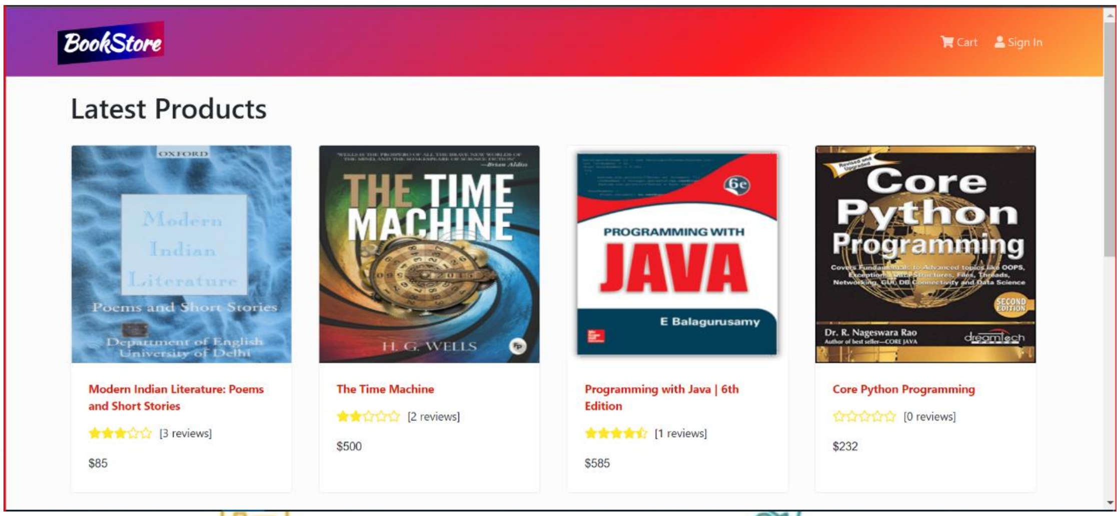
2. When the **Jenkins is ready** page appears, click **Start using Jenkins**.
   Notes:
   - This page may indicate **Jenkins is almost ready!** instead and if so, click **Restart**.
   - If the page does not automatically refresh after a minute, use your web browser to refresh the page manually.
3. If required, log in to Jenkins with the credentials of the user you just created and you are ready to start using Jenkins!

6. Challenges faced

- Environment provisioning
- Manual testing
- No DevOps centre of excellence
- Test data
- Manual deployments
- Planning in a DevOps environment
- Inconsistent environments
- Faced difficulty during the installation of the tools in the local system.
- Several codes were merged in the local branch and then pushed to the master this led to several inconsistency in the existing project.
- The versions of the software required for a smooth running of the project was not of local system supported and lower versions of the same was installed in the project.
- Integrating tools to several domains.
- Resistance to change.
- The multiple pull and push of the code can cause a certain halt.

7.1 Project Screenshots.

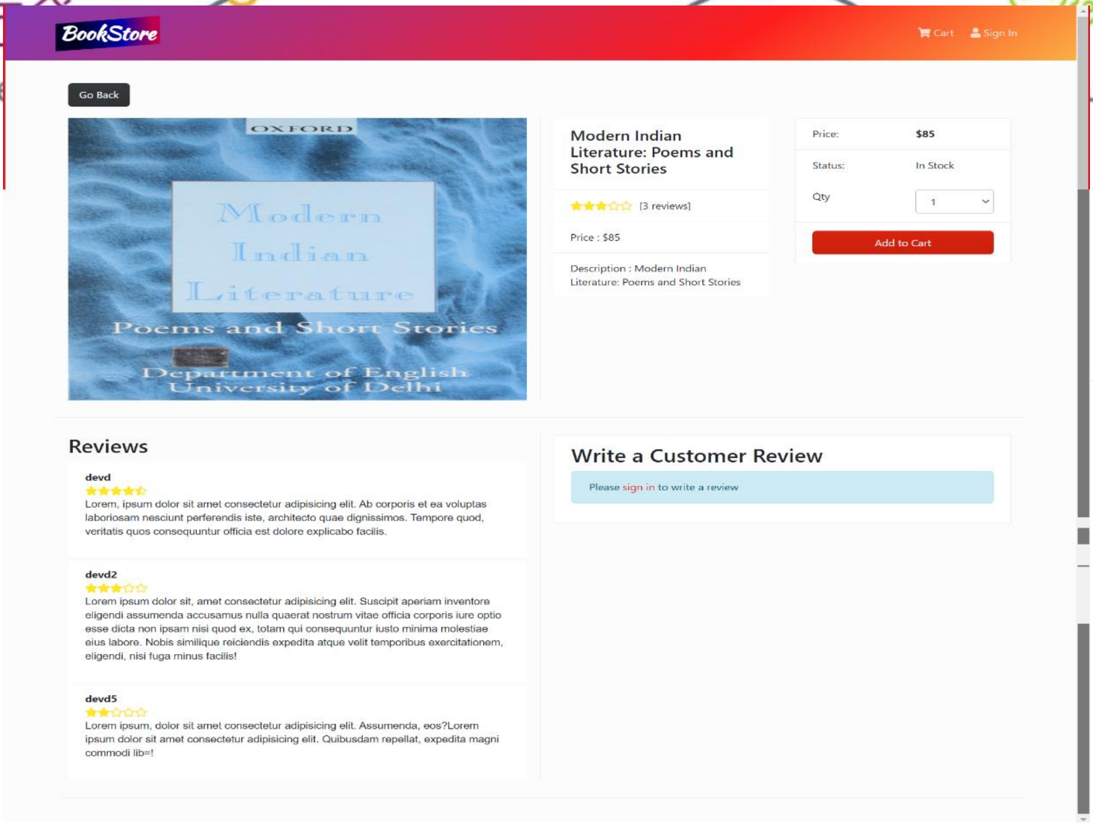
Home Screen



Latest Products

Product	Rating	Price
Modern Indian Literature: Poems and Short Stories	[3 reviews]	$85
The Time Machine	[2 reviews]	$500
Programming with Java | 6th Edition	[1 reviews]	$585
Core Python Programming	[0 reviews]	$232

Customer Reviews



Modern Indian Literature: Poems and Short Stories

[3 reviews]

Price : $85

Description : Modern Indian Literature: Poems and Short Stories

Price:	$85
Status:	In Stock
Qty	1

Reviews

devd
Lorem, ipsum dolor sit amet consectetur adipisicing elit. Ab corporis et ea voluptas laboriosam nesciunt perferendis iste, architecto quae dignissimos. Tempore quod, veritatis quos consequuntur officia est dolore explicabo facilis.

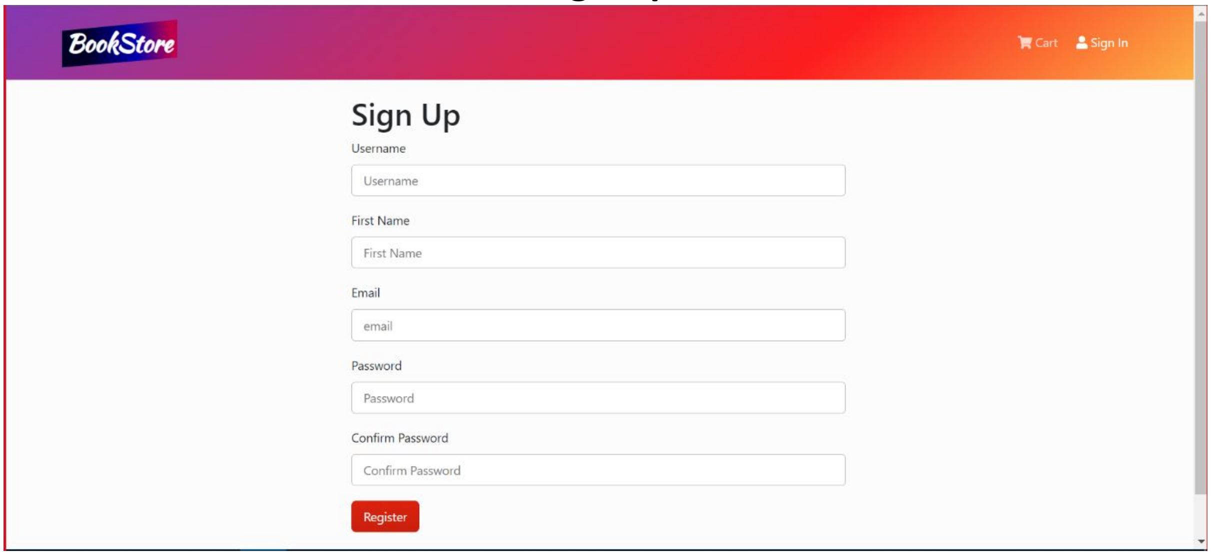
devd2
Lorem ipsum dolor sit, amet consectetur adipisicing elit. Suscipit aperiam inventore eligendi assumenda accusamus nulla quaerat nostrum vitae officia corporis iure optio esse dicta non ipsam nisi quod ex, totam qui consequuntur iusto minima molestiae eius labore. Nobis similique reiciendis expedita atque velit temporibus exercitationem, eligendi, nisi fuga minus facilis!

devd5
Lorem ipsum, dolor sit amet consectetur adipisicing elit. Assumenda, eos?Lorem ipsum dolor sit amet consectetur adipisicing elit. Quibusdam repellat, expedita magni commodi lib=!

Write a Customer Review

Please sign in to write a review

Sign Up



Sign Up

Username

Username

First Name

First Name

Email

email

Password

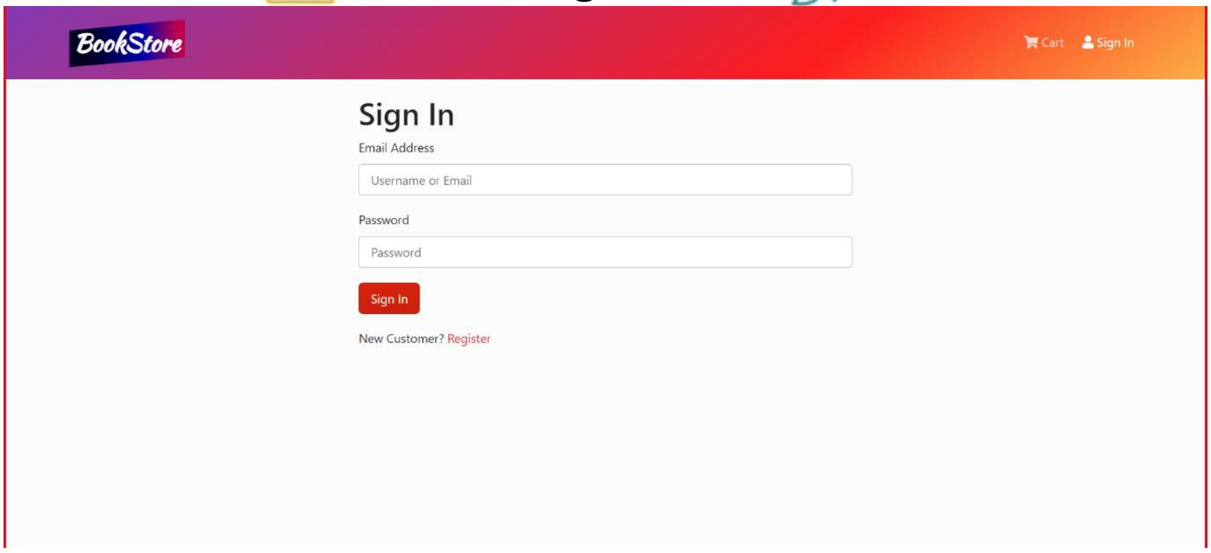
Password

Confirm Password

Confirm Password

Register

Sign In



Sign In

Email Address

Username or Email

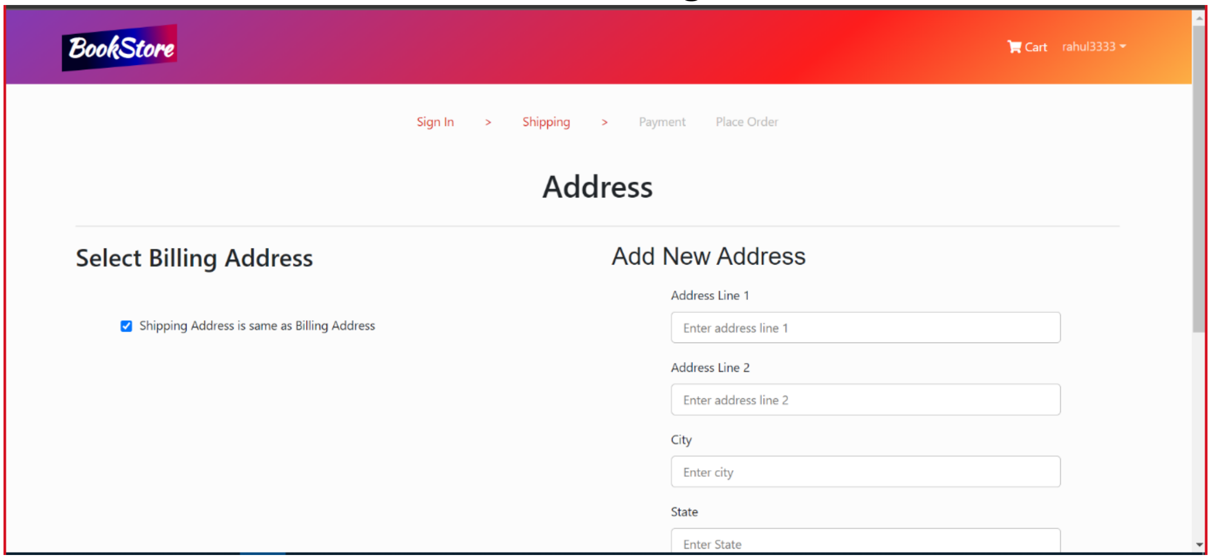
Password

Password

Sign In

New Customer? Register

Address Page



BookStore

Cart rahul3333

Sign In > Shipping > Payment Place Order

Address

Select Billing Address

☑ Shipping Address is same as Billing Address

Add New Address

Address Line 1

Enter address line 1

Address Line 2

Enter address line 2

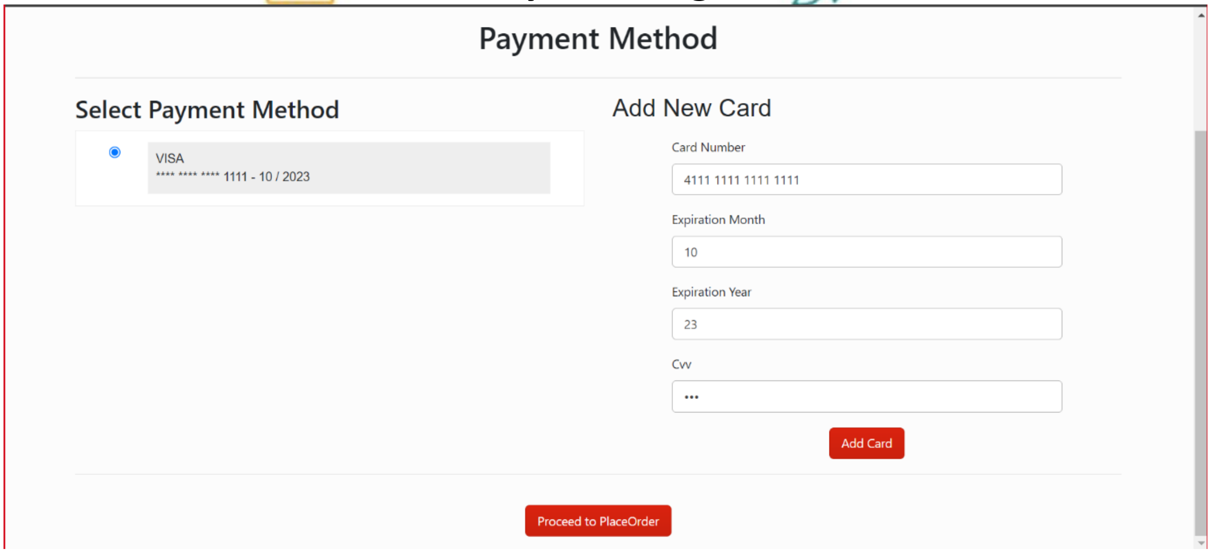
City

Enter city

State

Enter State

Payment Page



Payment Method

Select Payment Method

● VISA
**** **** **** 1111 - 10 / 2023

Add New Card

Card Number

4111 1111 1111 1111

Expiration Month

10

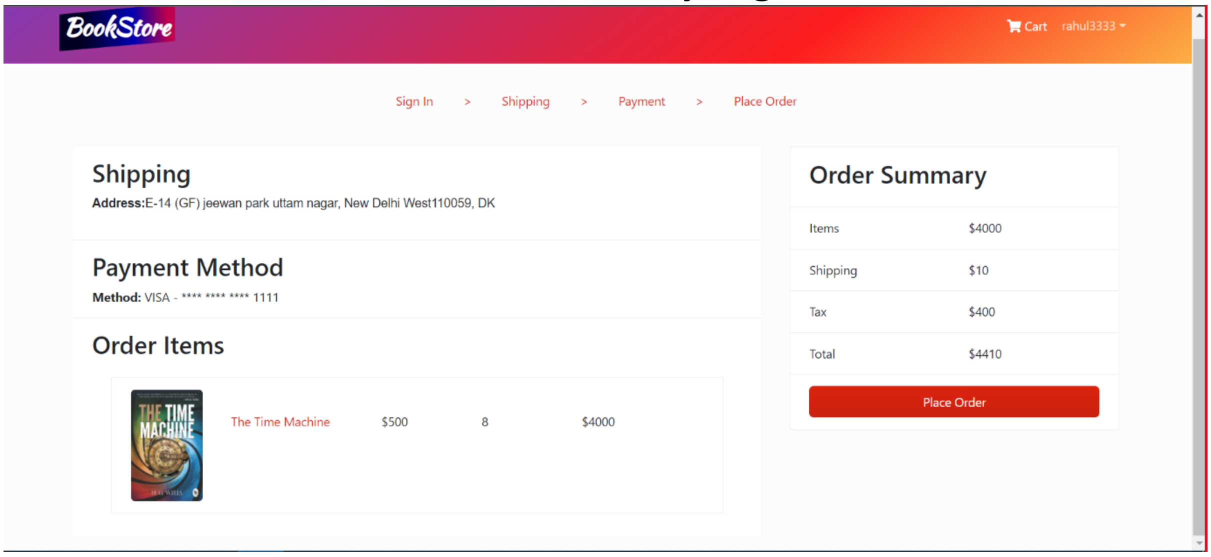
Expiration Year

23

Cvv

•••

Order Summary Page



Shipping

Address:E-14 (GF) jeewan park uttam nagar, New Delhi West110059, DK

Payment Method

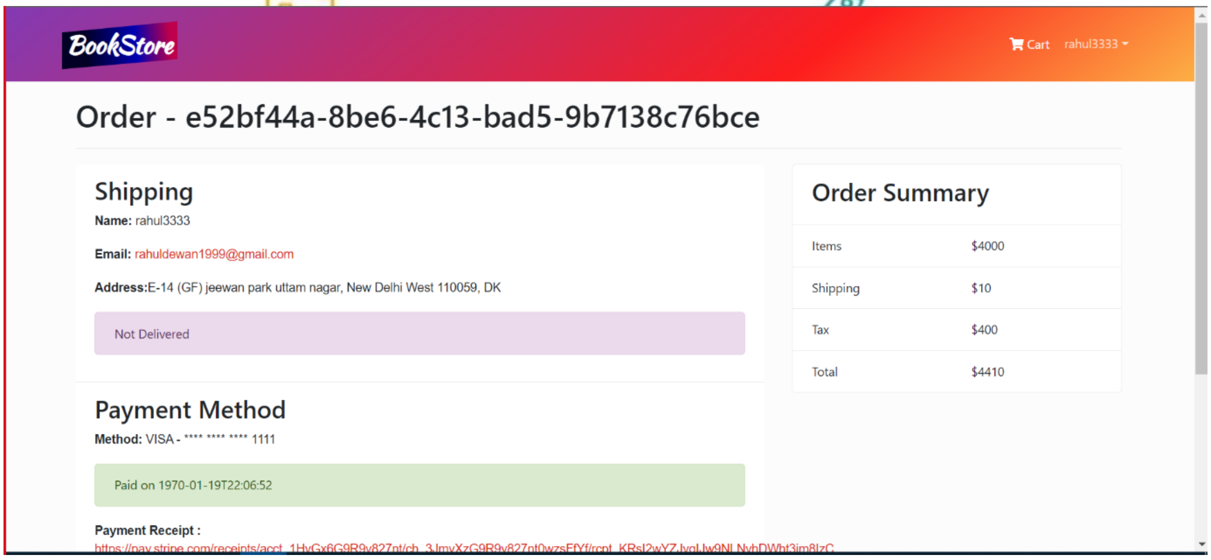
Method: VISA - **** **** **** 1111

Order Items

The Time Machine	$500	8	$4000

Order Summary

Items	$4000
Shipping	$10
Tax	$400
Total	$4410



Order - e52bf44a-8be6-4c13-bad5-9b7138c76bce

Shipping

Name: rahul3333

Email: rahuldewan1999@gmail.com

Address:E-14 (GF) jeewan park uttam nagar, New Delhi West 110059, DK

Not Delivered

Payment Method

Method: VISA - **** **** **** 1111

Paid on 1970-01-19T22:06:52

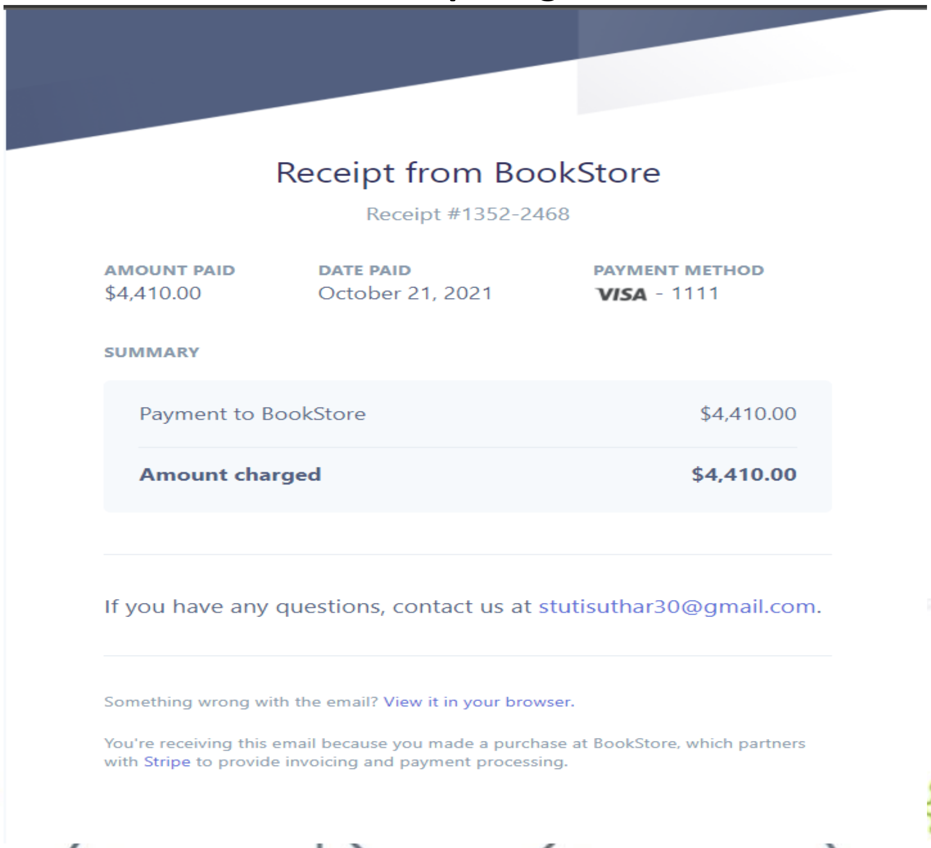
Payment Receipt :

https://pay.stripe.com/receipts/acct_1HyGx6G9R9v827nt/ch_3JmyXzG9R9v827nt0wzsFfYf/rcpt_KRsl2wYZJygIJw9NLNyhDWht3im8IzC

Order Summary

Items	$4000
Shipping	$10
Tax	$400
Total	$4410

Receipt Page


Receipt from BookStore

Receipt #1352-2468

AMOUNT PAID	DATE PAID	PAYMENT METHOD
$4,410.00	October 21, 2021	VISA - 1111

SUMMARY

Payment to BookStore	$4,410.00
Amount charged	**$4,410.00**

If you have any questions, contact us at stutisuthar30@gmail.com.

Something wrong with the email? View it in your browser.

You're receiving this email because you made a purchase at BookStore, which partners with Stripe to provide invoicing and payment processing.

My Order Page