**Book Lender**



**SOFTWARE PRODUCTION ENGINEERING PROJECT**

**Table of Contents**

[**ABSTRACT** 3](#_Toc7800187)

[**INTRODUCTION** 3](#_Toc7800188)

[**Why DevOps?** 3](#_Toc7800189)

[**About the Application** 5](#_Toc7800190)

[**SOFTWARE DEVELOPMENT LIFE CYCLE** 5](#_Toc7800191)

[**Scope of the Project** 5](#_Toc7800192)

[**Project architecture, Workflows** 6](#_Toc7800193)

[**Source Code Management** 16](#_Toc7800194)

[**Build** 17](#_Toc7800195)

[**Test** 18](#_Toc7800196)

[**Artifact** 19](#_Toc7800197)

[**Deploy** 20](#_Toc7800198)

[**Monitor** 20](#_Toc7800199)

[**CI/CD PIPELINE** 23](#_Toc7800200)

[**Maven Build** 23](#_Toc7800201)

[**JUnit Testing** 24](#_Toc7800202)

[**Docker Build** 25](#_Toc7800203)

[**Rundeck job** 25](#_Toc7800204)

[**RESULTS & DISCUSSIONS** 26](#_Toc7800205)

[**FUTURE WORK** 27](#_Toc7800206)

[**CONCLUSION** 27](#_Toc7800207)

[**REFERENCES** 27](#_Toc7800208)

# **ABSTRACT**

Books play an important role in every individual life. People often say that "When you open a book, you open a new world". Books gained a lot of value to mankind.

Book Lending helps people to share books among themselves. It is a great way to learn new things and constantly update our knowledge.

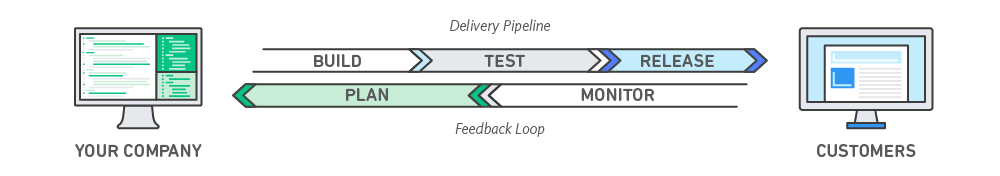
Sharing books is never an easy task especially if you have huge collection, as it hard to remember the books you have in the shelf, books you lent, books you borrowed.

BookLender overcomes the above difficulties and makes the whole process of sharing books easier to both parties.

# **INTRODUCTION**

## **Why 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. This speed enables organizations to better serve their customers and compete more effectively in the market.



The above figure is the typical workflow of a project following DevOps principles.

Following are the benefits of following DevOps principles.

* Speed
* Rapid Delivery
* Reliability
* Scale
* Improved Collaboration

**Speed:**

Move at high velocity so you can innovate for customers faster, adapt to changing markets better, and grow more efficient at driving business results. The DevOps model enables developers and operations teams to achieve these results.

**Rapid Delivery:**

Increase the frequency and pace of releases so that can innovate and improve the product faster. The quicker you can release new features and fix bugs, the faster can be the response to customer’s needs and build competitive advantage. Continuous integration and continuous delivery are practices that automate the software release process, from build to deploy.

**Reliability:**

Ensure the quality of application updates and infrastructure changes so you can reliably deliver at a more rapid pace while maintaining a positive experience for end users. Using practices like CI and CD tests that each change is functional and safe. Monitoring and logging practices helps to stay informed of performance in real time.

**Scale:**

Automation and consistency helps to manage complex or changing systems efficiently with reduced risk. Iaac helps to manage development, testing and production environments in a repeatable and more efficient manner.

**Improved Collaboration:**

Developers and operations teams collaborate closely, share many responsibilities and combine their workflows. This reduces inefficiencies and saves time.

## **About the Application**

Book Lender is a web application developed to bridge the gap between lending and borrowing of books. It makes the process of sharing books so simple and straight forward.

**Activities**:

* User can sign up and login into the application.
* User can add book.
* User can request book.
* User can accept books requested to him.
* User can reject books requested to him.
* User can see what all books he requested.
* User can see pending requests requested by him.
* User can see books lent out.
* User can see books borrowed.
* User can return book.
* User can approve book once he gets his book back.

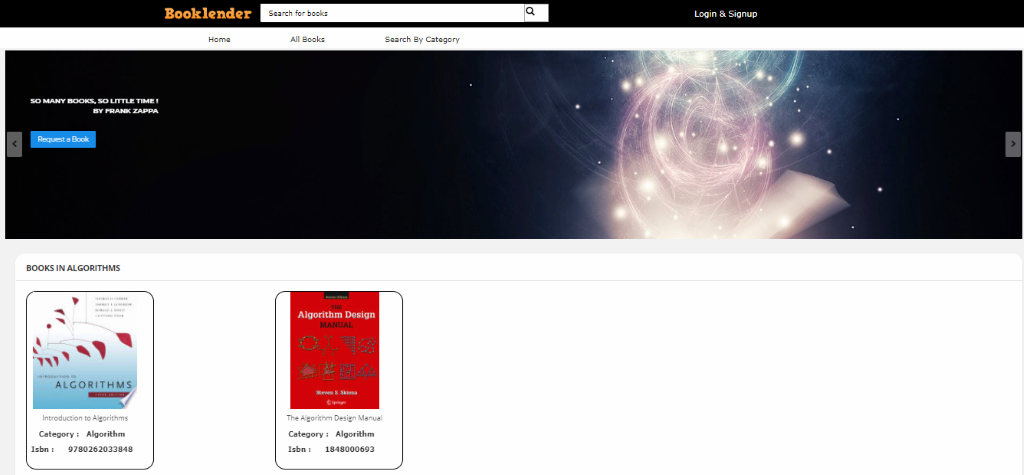
# **SOFTWARE DEVELOPMENT LIFE CYCLE**

## **Scope of the Project**

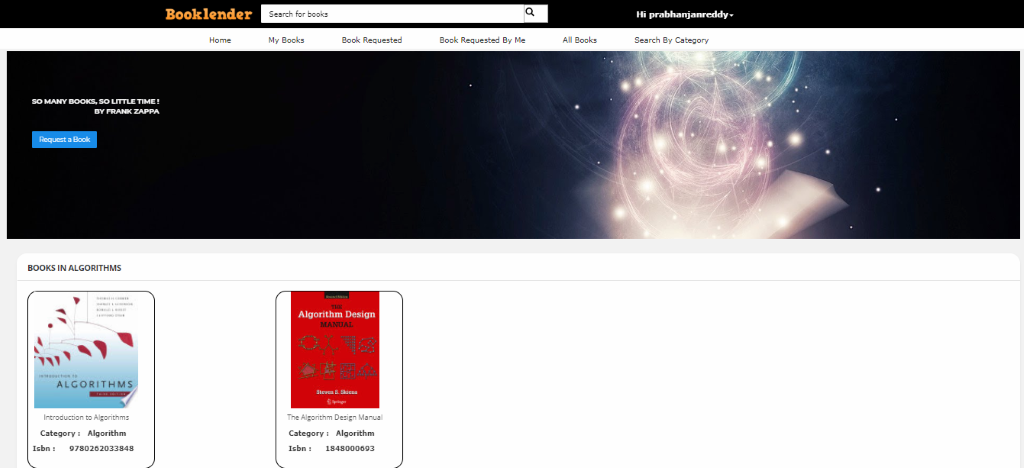
This project will helpful for sharing the books and is mainly developed with an intent for students within the college.

## **Project architecture, Workflows**

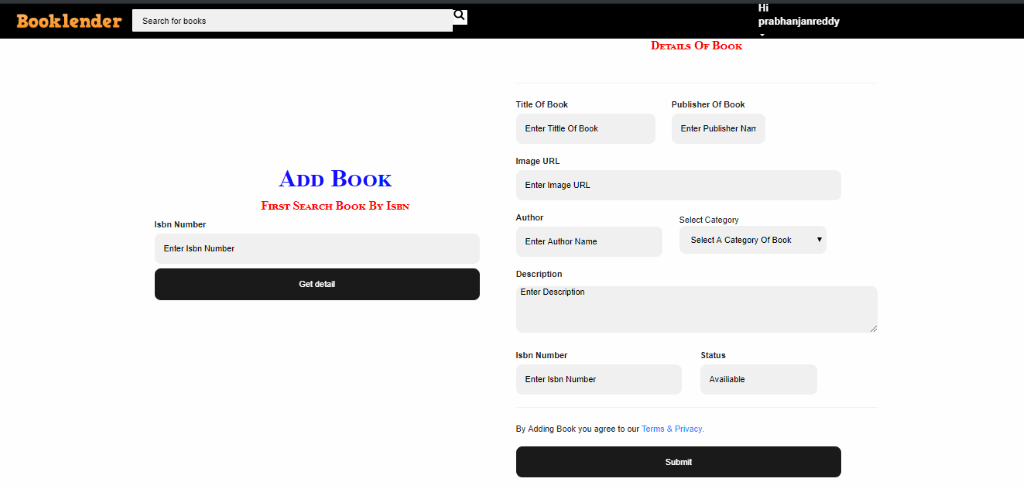
Home page of the application:



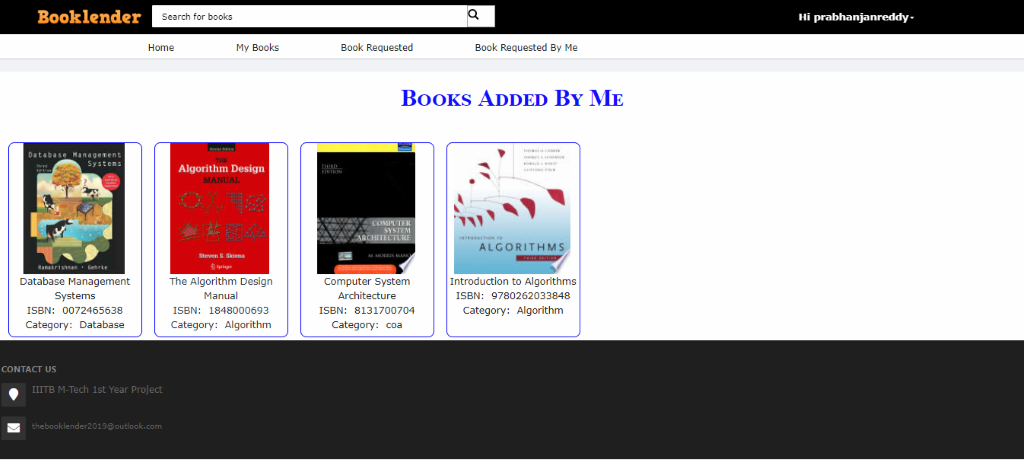
Lender Logs into the application:



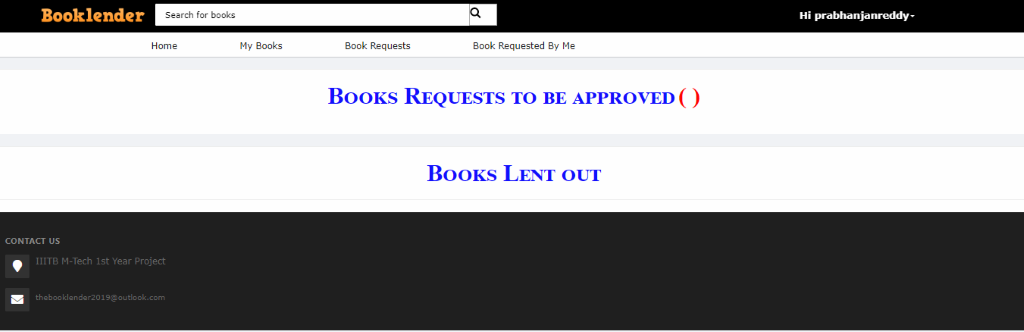
Adds a book by giving ISBN number only.



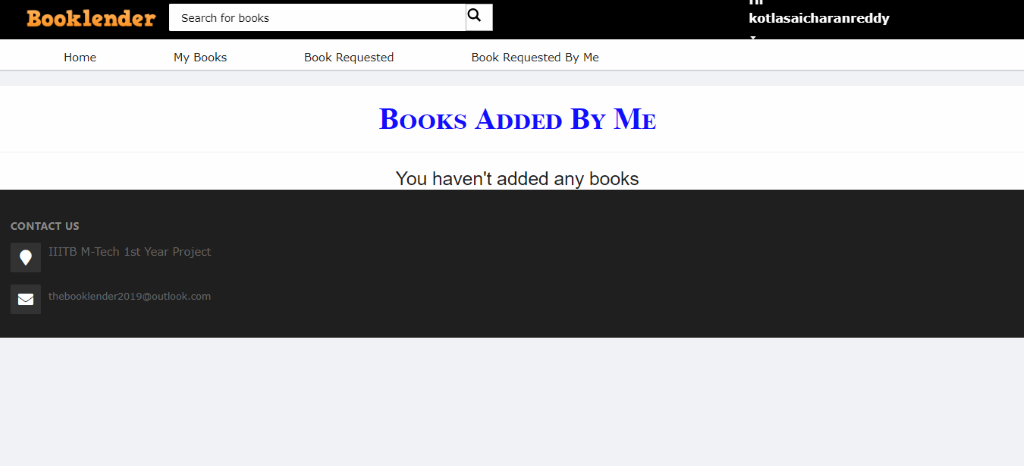
The added book will be appeared in My Books



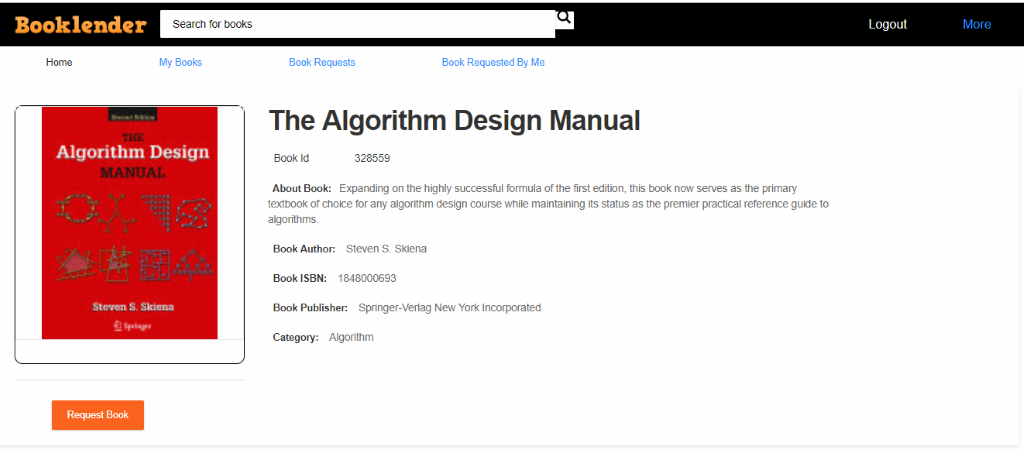
Lender also checks other tabs as well



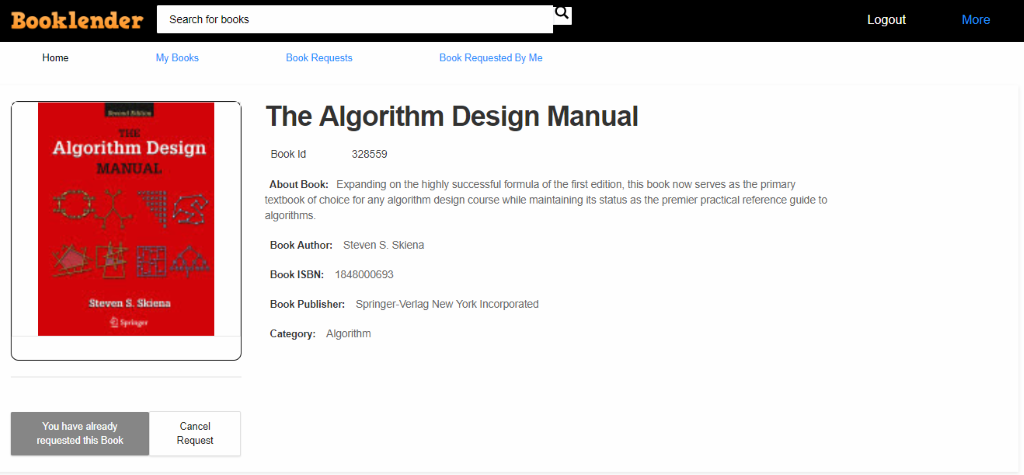
Borrower logs into application and checks whether he added any books

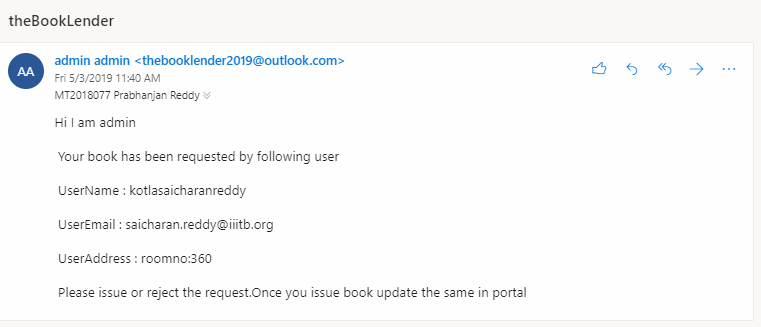


Borrower requests a book from lender by clicking Request Book button

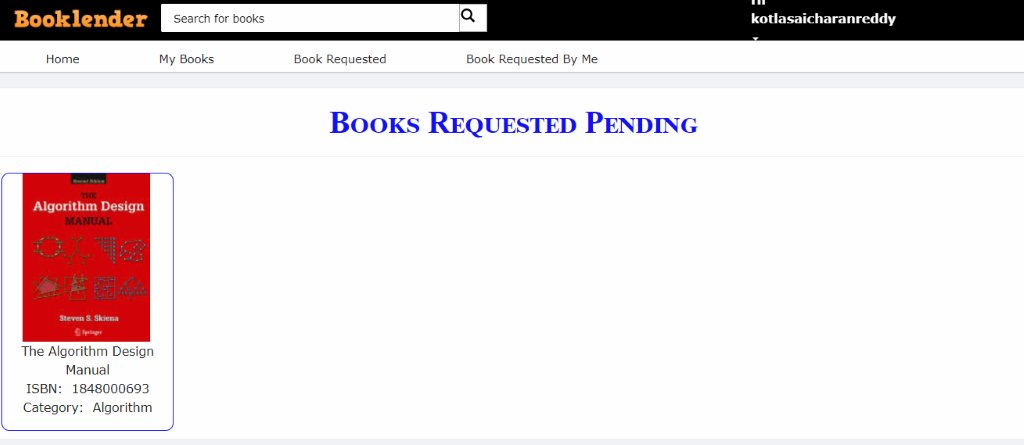


After request is made an email is sent to lender and status is updated in the same page

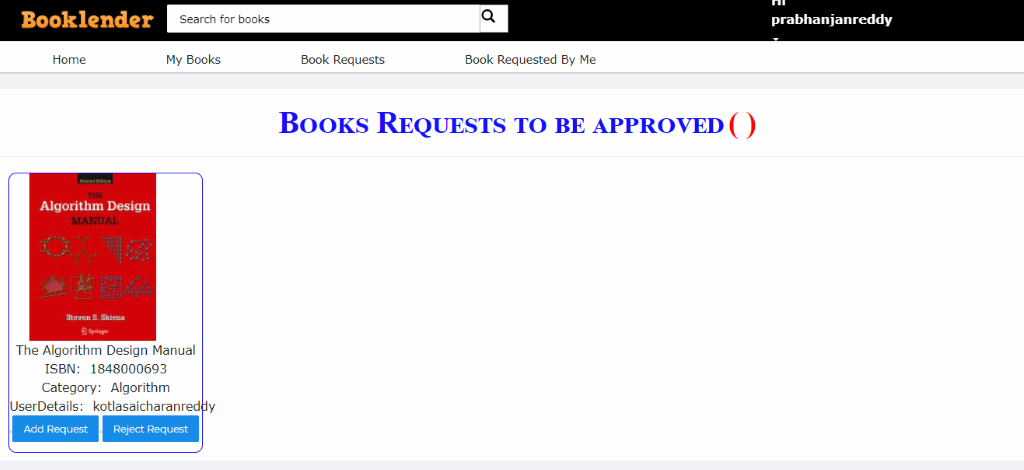




Borrower can check what all books he requested

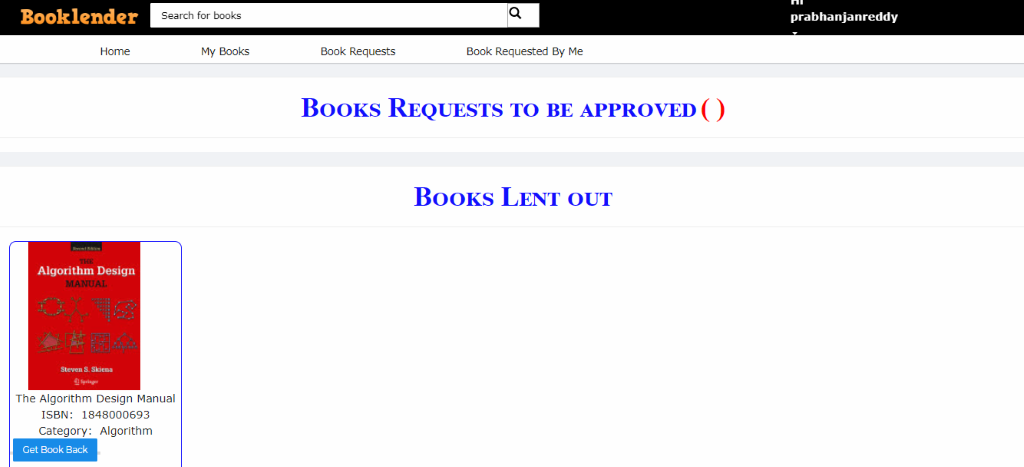


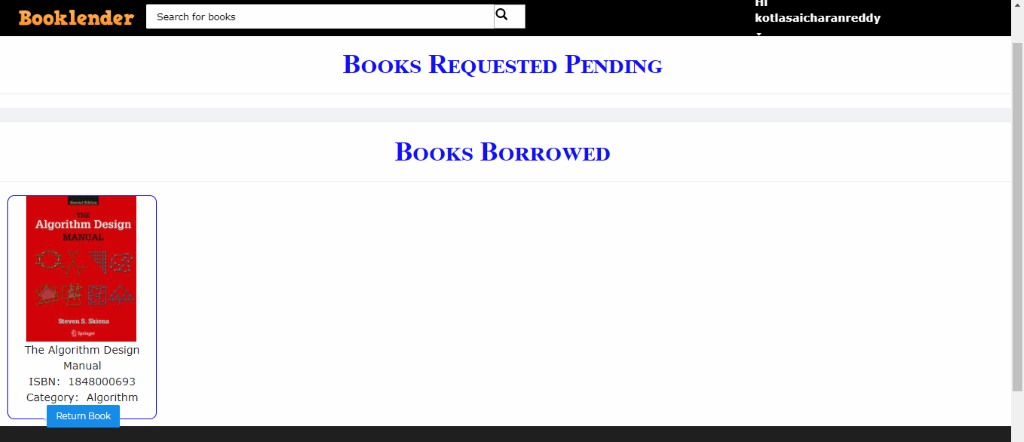
Lender can give his book in hand and update the accept request in portal



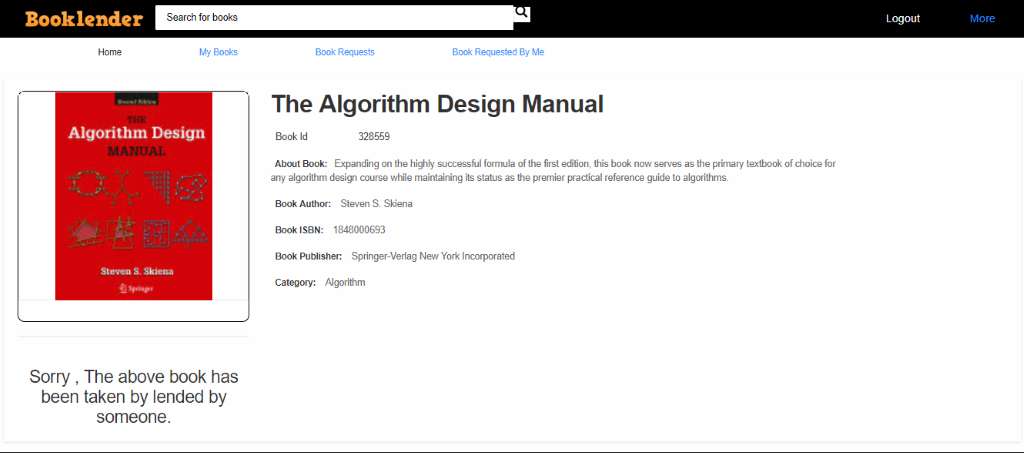
Once accepted, the book will be shown under lent out section of lender

And borrowed section for borrower.





Now book will not be given an option to request by anyone.

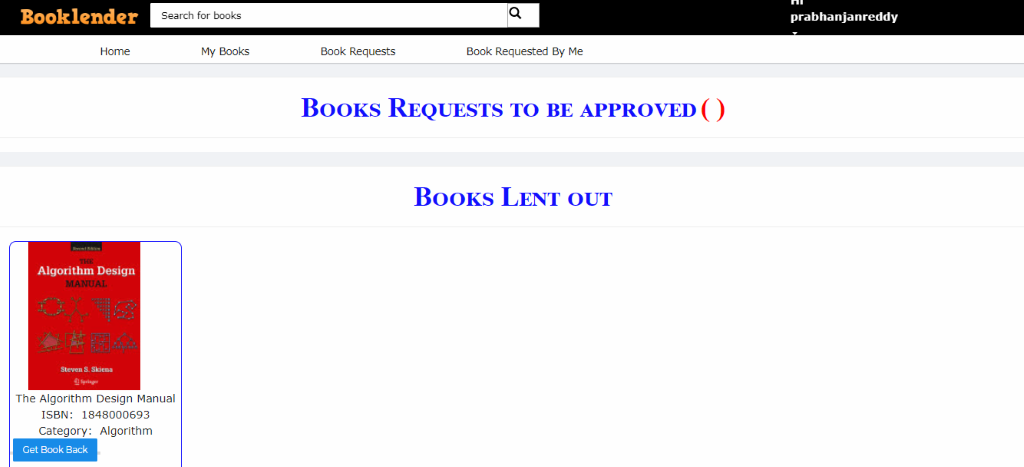


Now borrower got book from lender and has done with the reading.

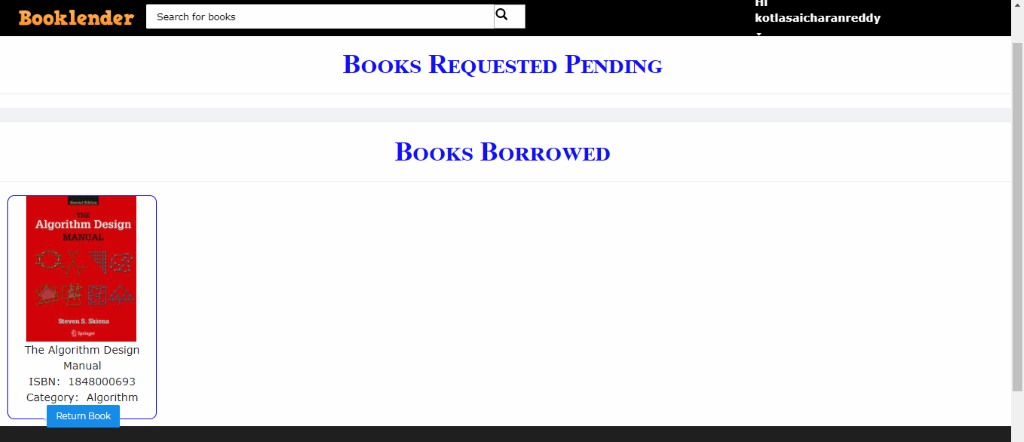
Now there can be two things that can happen

Either borrower wants to return book or lender need’s book for some reason.

So if lender want his book back then he clicks Get Book Back button.

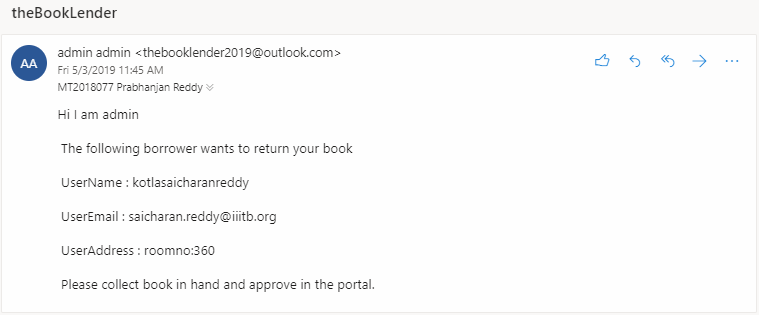


If the borrower wants to return the book then he clicks return book button.

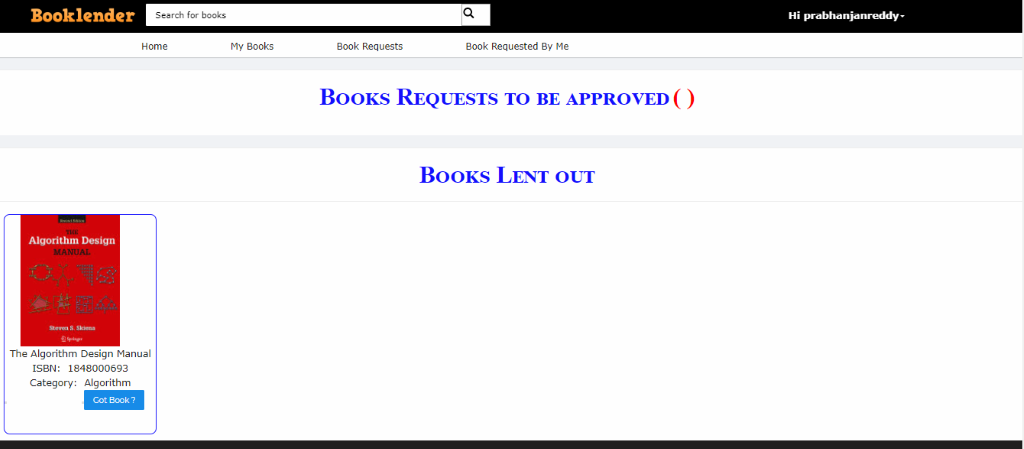


A mail will be triggered to lender or borrower something like below.

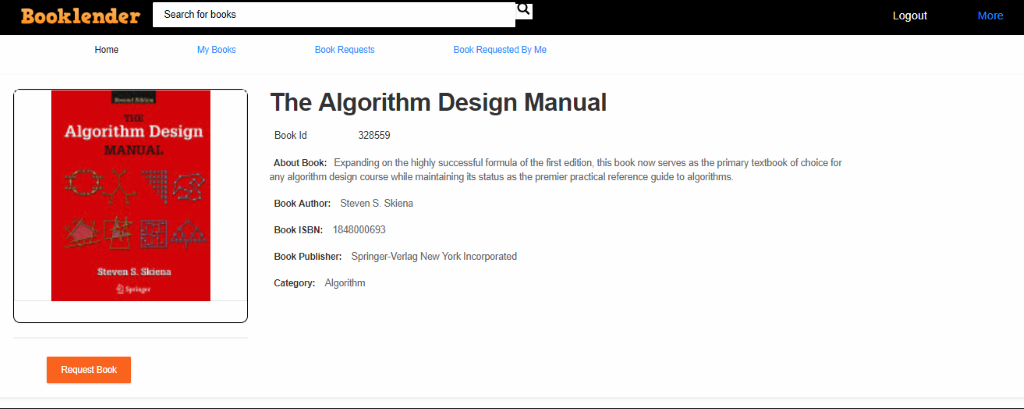
The lender gets book in hand from borrower.



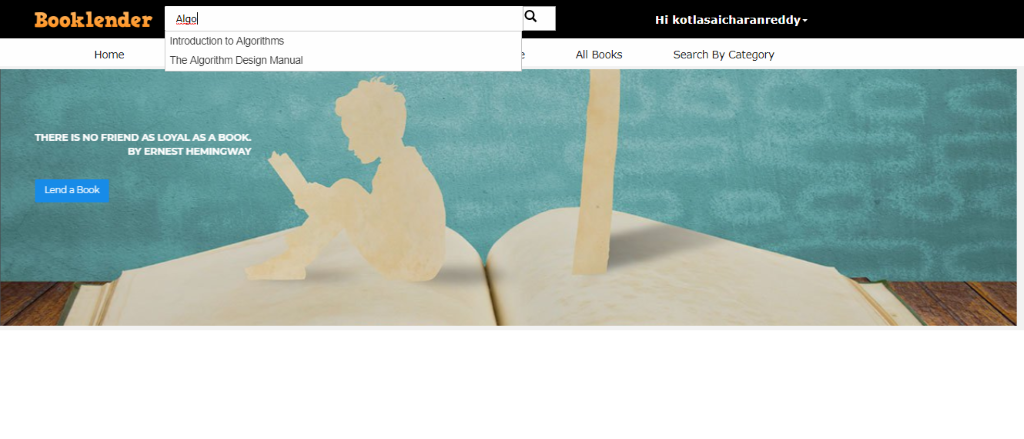
In either of the cases the book will be coming back to lender and he approves got book button once he gets the book to his shelf.



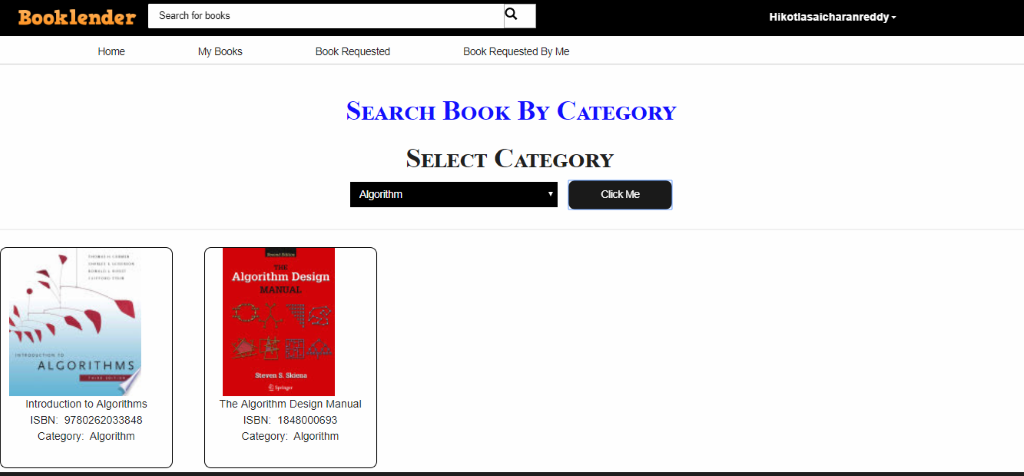
And it is available for any other borrower to lend.



There are options to search books as well.



Borrower can search books by category also.

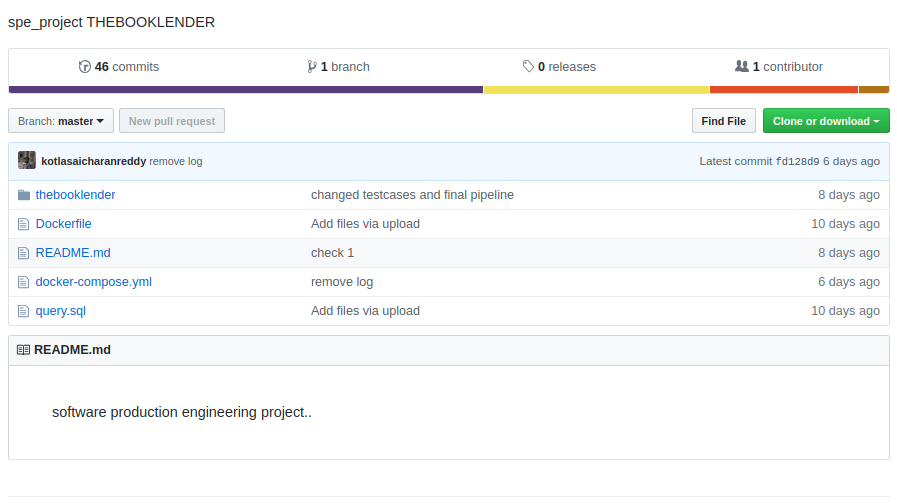


## **Source Code Management:**

For source code management we have used the tool git and for repository it is GitHub. As it is reliable and has capabilities for many people to contribute the code to a single repository.

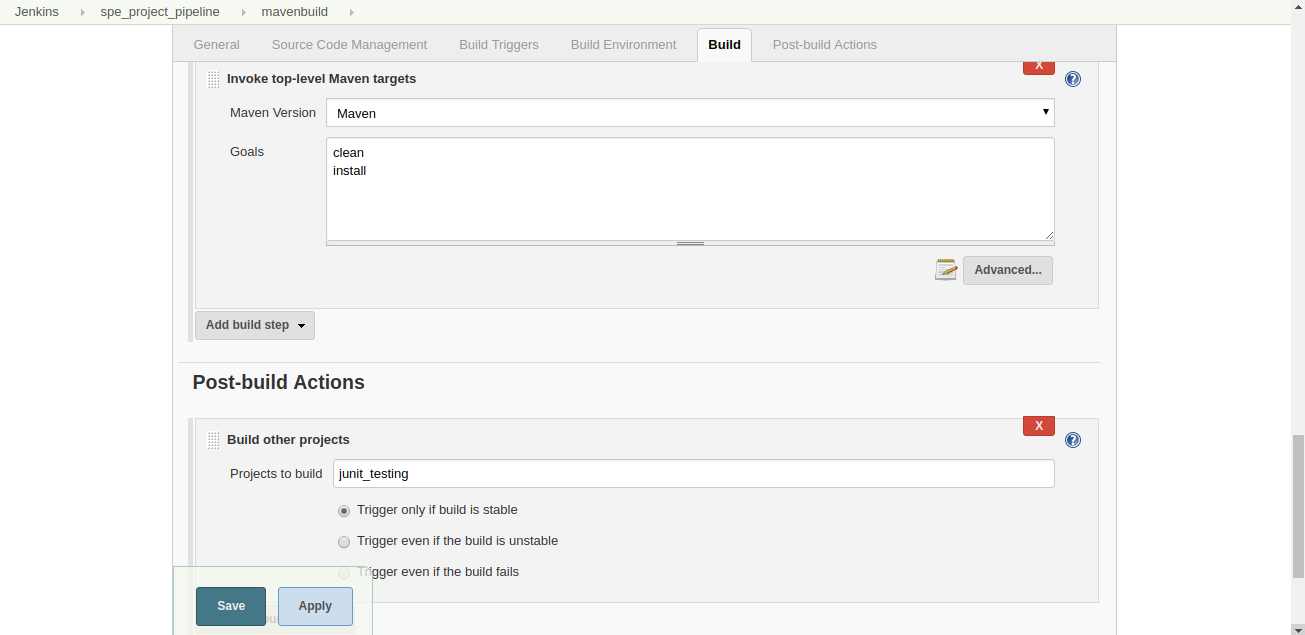
Github repository [link](https://github.com/kotlasaicharanreddy/spe_project_thebooklender)

Project git repository is as follows:



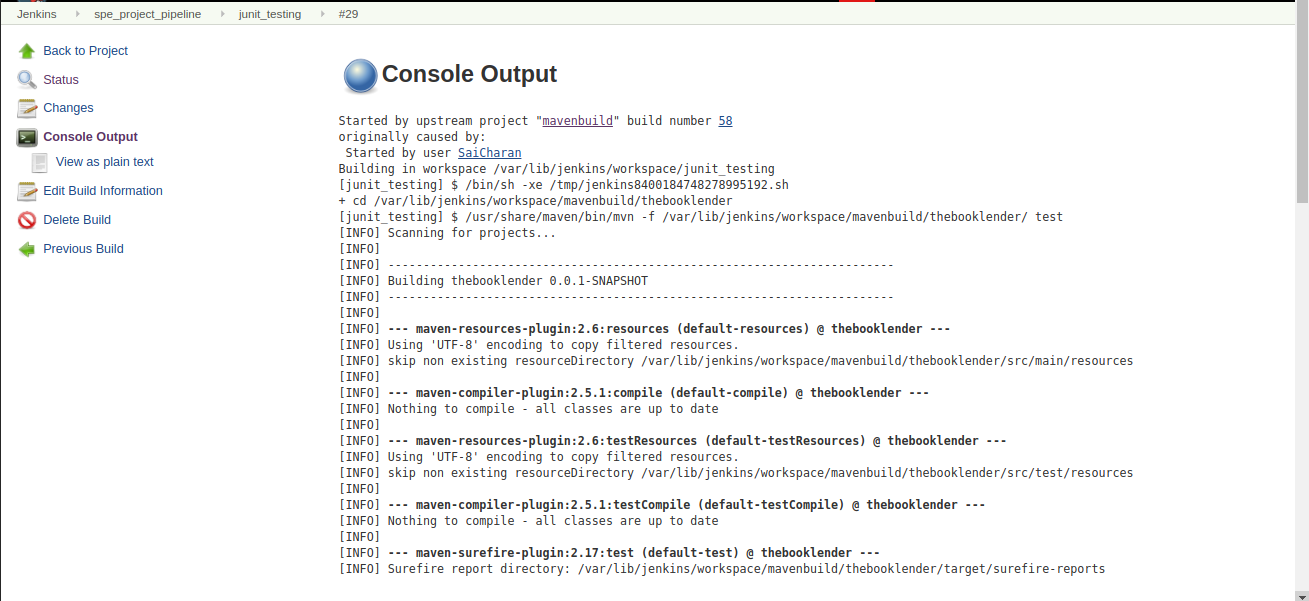
## **Build:**

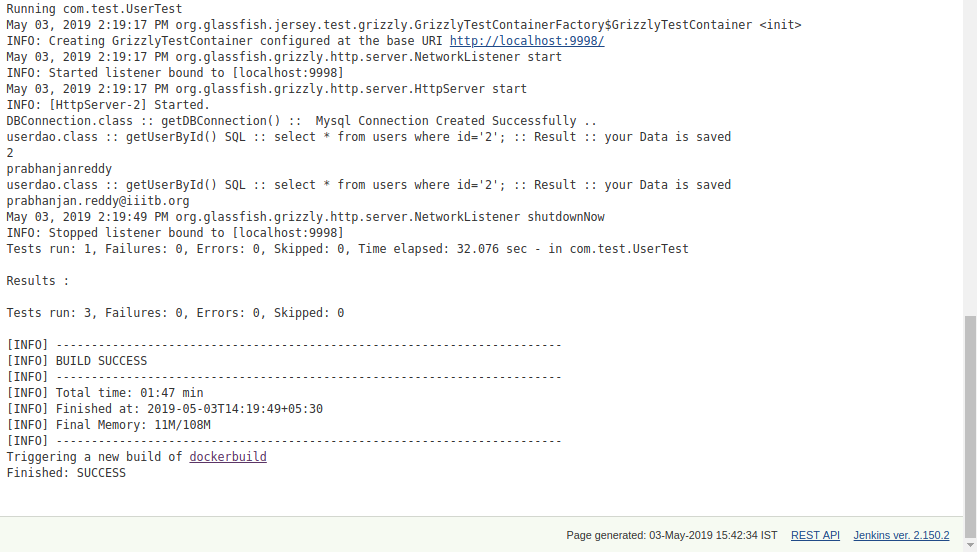
For build the project we have used maven as our project is built on Java language. We have used Eclipse IDE and maven is a part of Eclipse IDE. A pom.xml will be created when we start creating a maven based project in eclipse by default.



## **Test:**

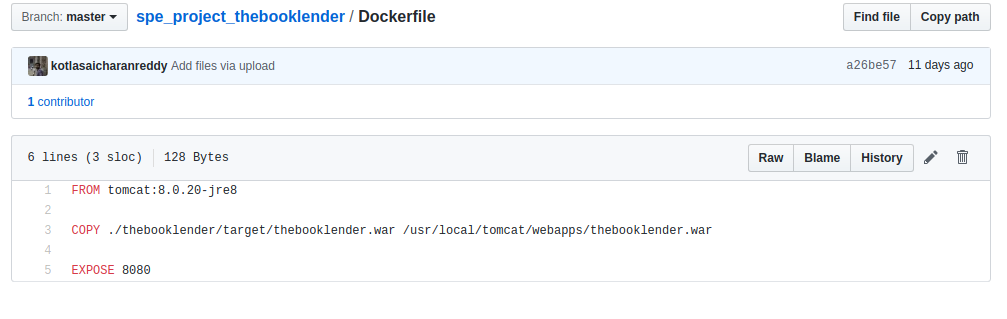
For testing Junit comes with maven by default so we used Junit testing for which we wrote two test cases which checks a particular user has a specified book or not and also a checked email of a particular id.





## **Artifact:**

As we built a java web application it generates a war file which has to be deployed in tomcat server. Build stage generates a war file and this has to kept in webapp folder of tomcat server. For this we used tomcat server’s docker image and then copied war file in local machine to the tomcat’s server webapps folder. This is done as follows:

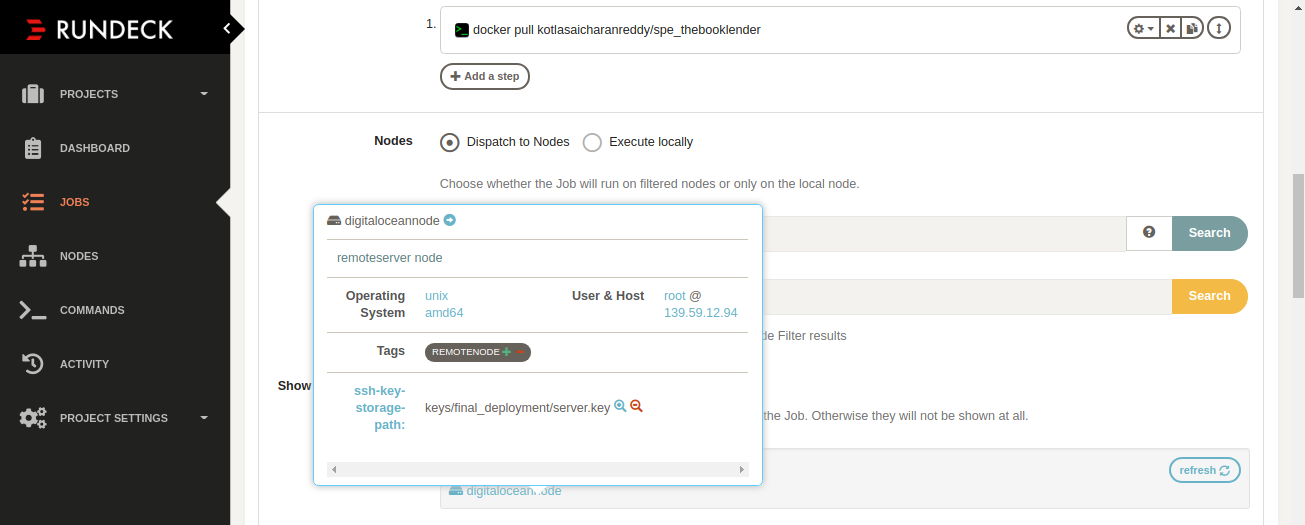


We initially used mysql as our database on our local host and then used a official mysql docker image to dump application’s database into it. But we faced issues while connecting both mysql docker image and tomcat(with war file deployed) image each other. So later we hosted database on a server provided which is provided for free. Now the web application and database sits on separate servers.

## **Deploy:**

By using docker we created a docker image and then pushed into docker hub. For deploying this docker image onto a remote node which can be considered a server I’ve created a digital ocean servlet. Using rundeck which is an automation tool, latest docker image will be pulled onto the digital ocean’s servlet .

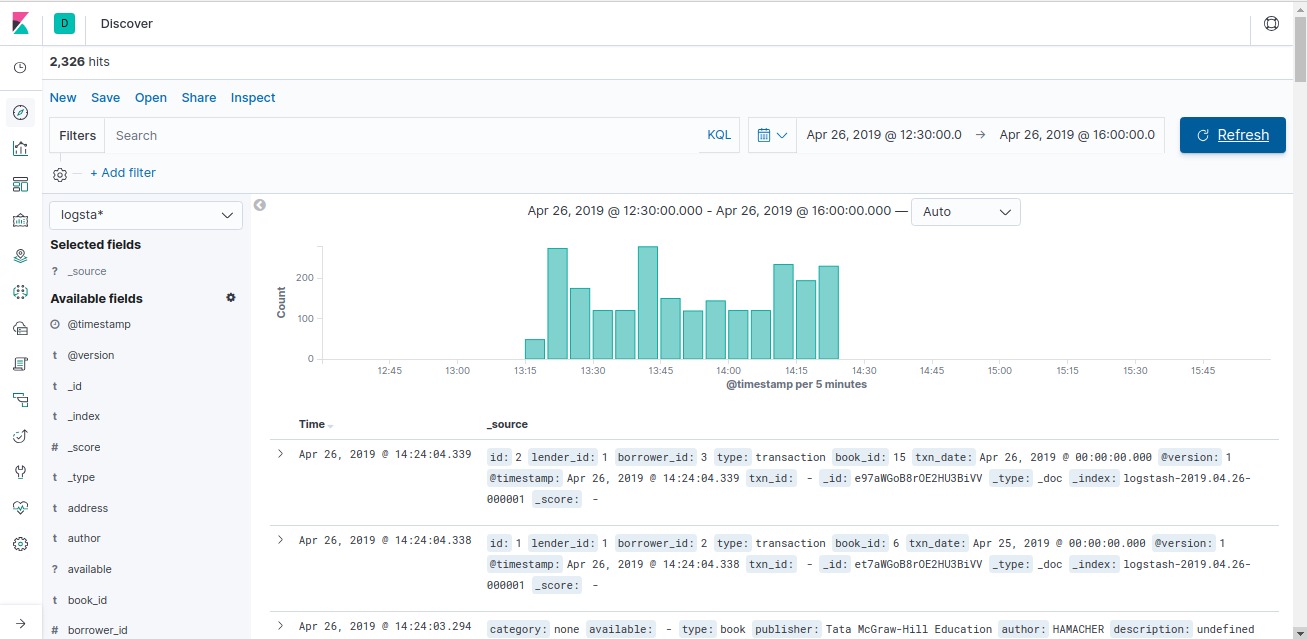
Below is the picture which shows the digital ocean’s servlet and rundeck job which deploys the image onto the servlet.



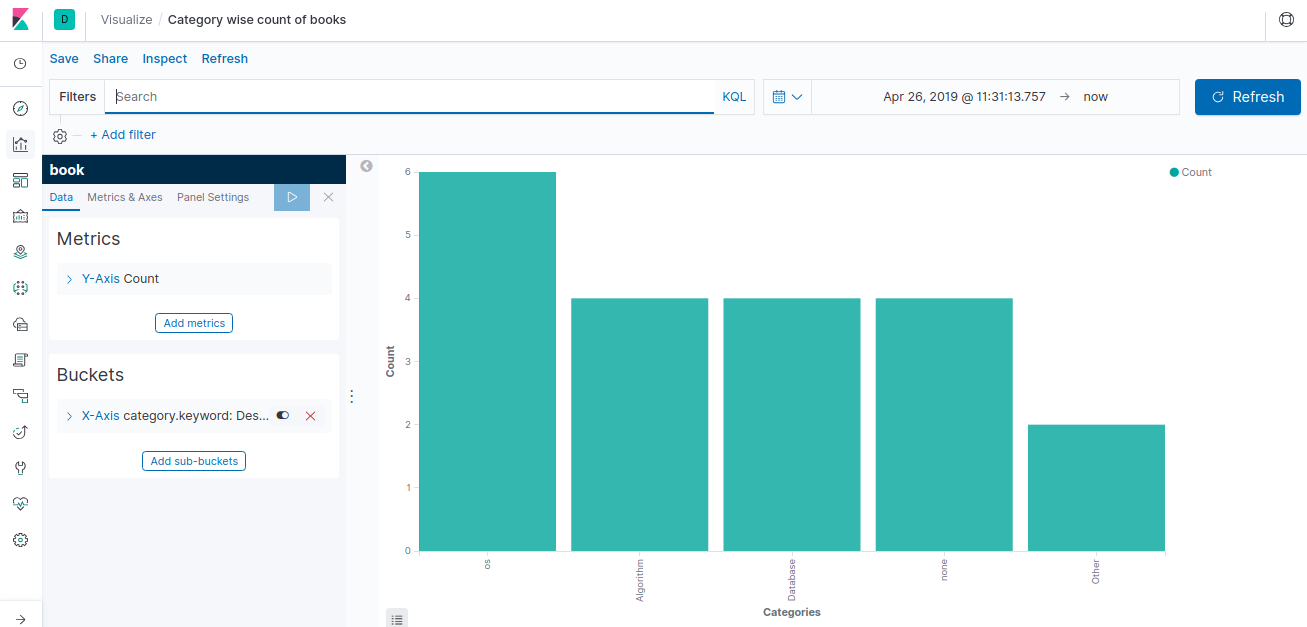
## **Monitor:**

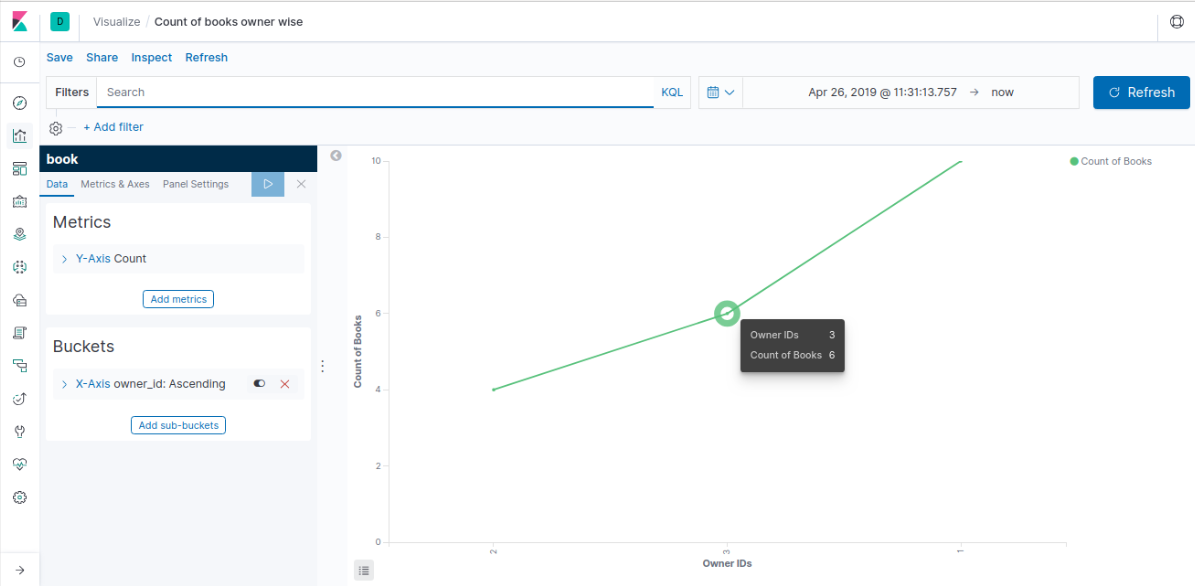
For monitoring we have used ELK tool which is an opensource tool. In this we monitored for the number of users registered, number of books category wise and user wise and also the server logs.

Server logs can be seen in the below image.

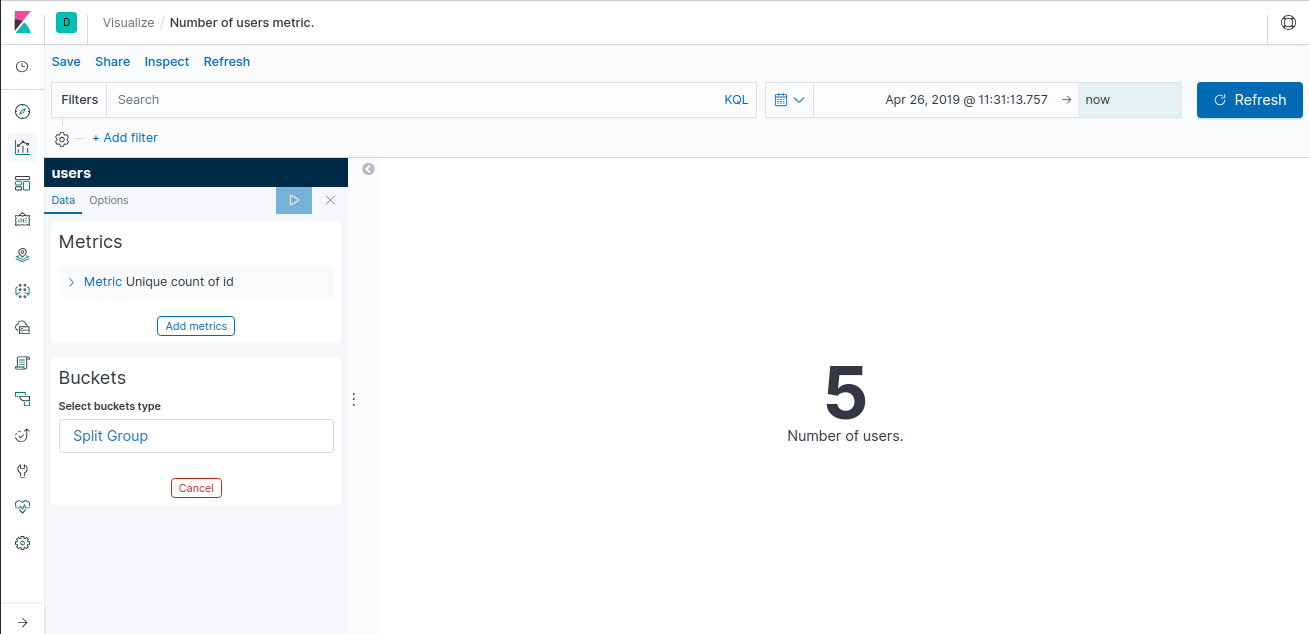


Number of books user wise and category wise can be seen below:

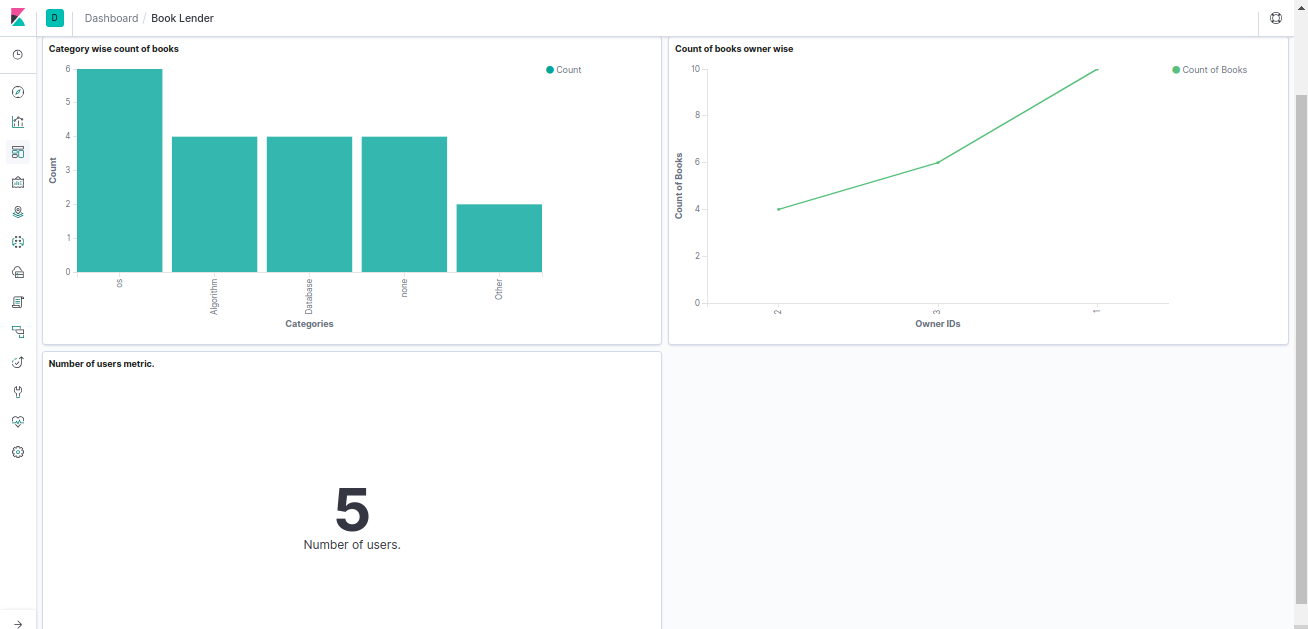




Number of users registered can be seen in the below image:



All the visualizations can be viewed in a dashboard as follows:



# **CI/CD PIPELINE**

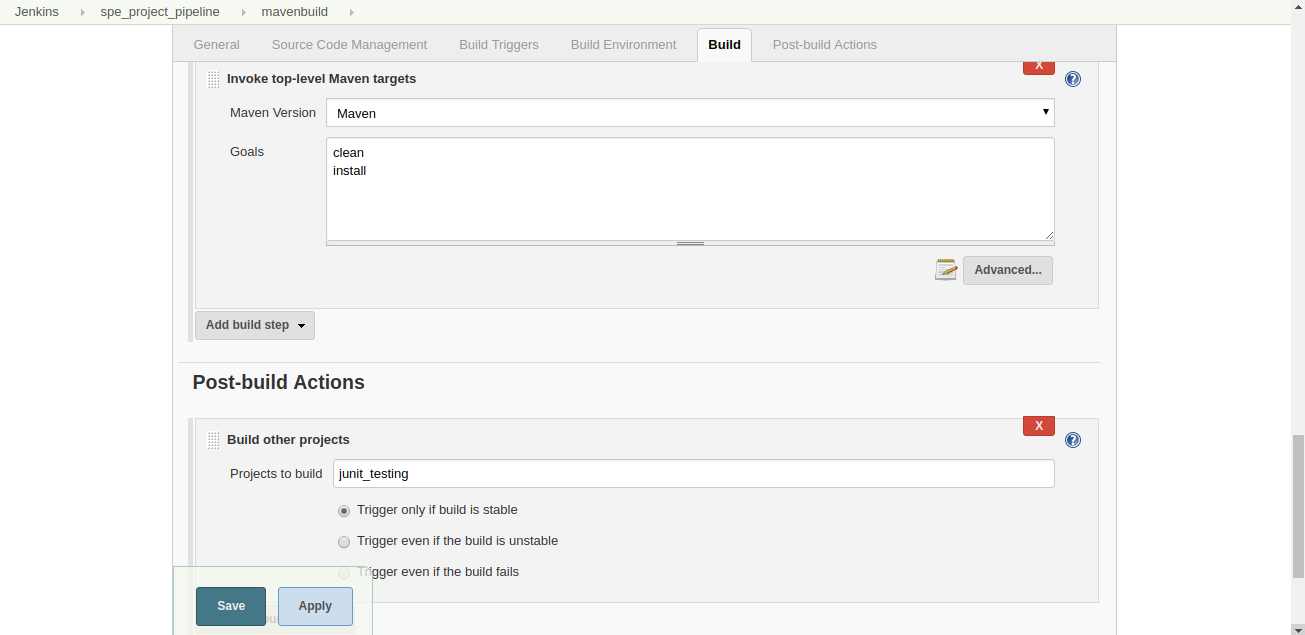
For continuous integration an open source tool called Jenkins is used. It is widely used and has many plugins for other open source tools such git plugin, docker plugin to build images, rundeck plugin to automate the deployment into server in our case it is digital ocean’s servlet. All the tasks can be scheduled in a sequential way such that a stage will only start when it’s predecessor job is successful. This can be achieved by calling the next job from the post-build step of current job.

In our pipeline there are four stage. They are :

* Maven Build
* Junit Testing
* Docker Build
* Rundeck job

## **Maven Build:**

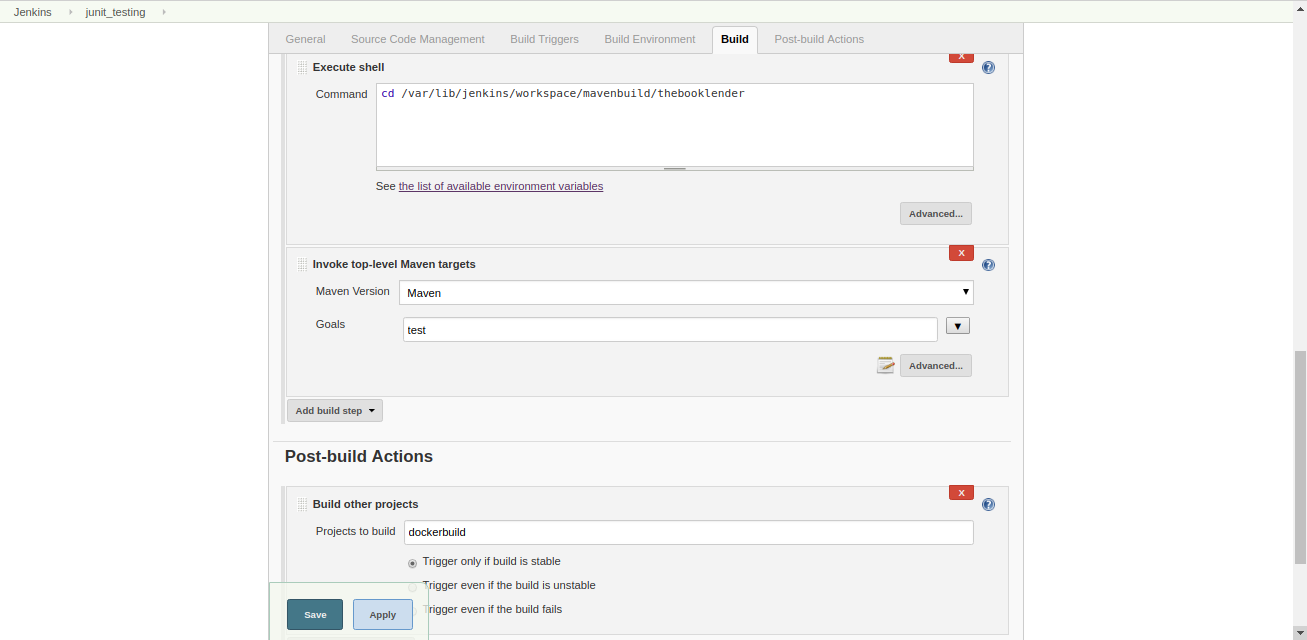
In this job the first step is to pull the code from the specified repository location in our case it is git. After pulling the code it starts building the war file with the help of maven which is the build step.



The post build step of this job is Junit Testing job.

## **JUnit Testing:**

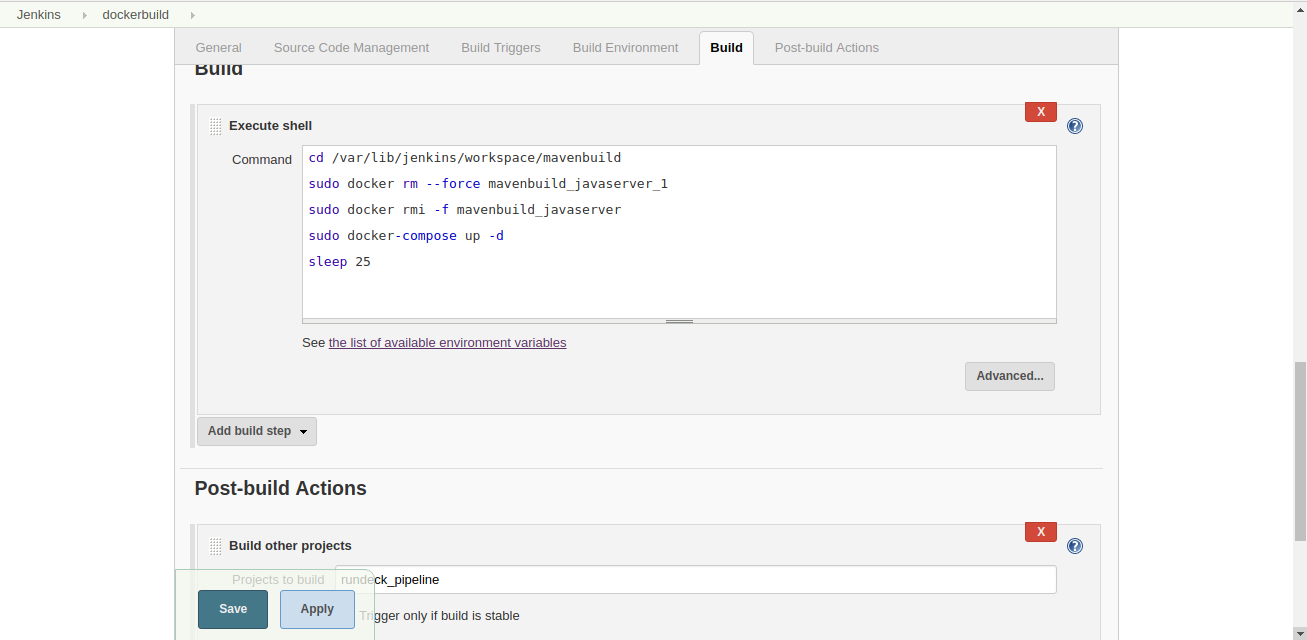
This job start only when the maven build job is successful. In this job we need to specify the path of the folder in which the testcases are present.



## **Docker Build:**

In this stage our web application docker image is built by using tomcat official docker image. We need specify the path of dockerfile or docker-compose yaml files to build the image.

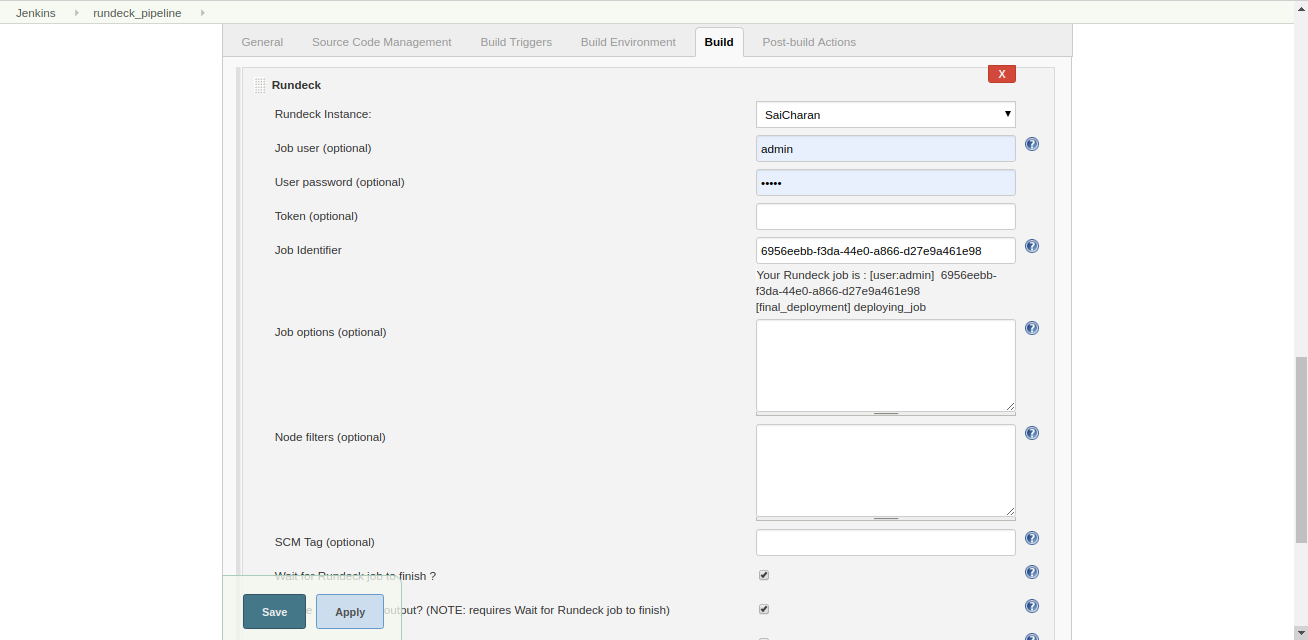
Initially we are deleting the currently running container and images and then building the image so that there would not be any issue while binding the docker image port to localhost port or remote host port.



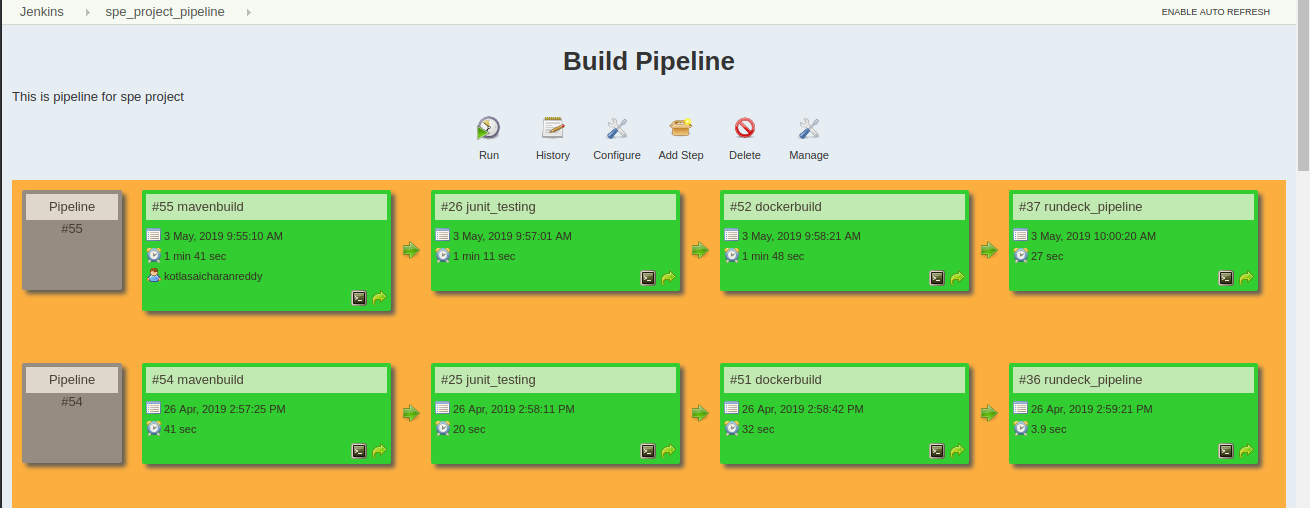
## **Rundeck job:**

This is the final job of the CI/CD pipeline which is deploying the image to a remote node in our case it is digital ocean servlet. This job starts only when docker build stage is successful.

There will a UUID associated for each job of rundeck which we have to provide in the Jenkins rundeck job.



The entire CI/CD pipeline can be viewed as follows:



# **RESULTS & DISCUSSIONS:**

Following things can be noticed from the pipeline image regarding time to deploy:

* Maven Build stage - 1min 41 sec
* Junit Testing - 1 min 11 sec
* Docker build stage – 1 min 48 sec
* Rundeck job – 27 sec

So, the application can be build and deployed in less than 5 minutes using CI/CD pipeline.

As the application is limited within the campus, so it can with stand the requests users make. The application is scalable enough, that is, be able to process those incoming requests with less time. No more hardware is required to handle the load.

# **FUTURE WORK:**

There is vast scope for this kind of applications that makes people’s life easier. Little improvements in UI. Currently this application is limited with in the campus. It can be extended thorough the city.

# **CONCLUSION:**

This project emphasizes on how to involve DevOps tools with web application for deployment of web application in an easy way.

# **REFERENCES:**

* Github repository link: <https://github.com/kotlasaicharanreddy/spe_project_thebooklender>
* Devops theory: <https://aws.amazon.com/devops/what-is-devops/>
* Online mysql hosting service: <https://www.db4free.net/>
* Digital Ocean to host a remote node/server: <https://www.digitalocean.com/>