

WIPRO NGA Program – Java Full Stack

Capstone Project Presentation – 29th Aug

Project Title Here - Library Management Service

Presented by - Faizan Aslam (31158)

Project Demo

The State Library

Home Books About Contact Login

Welcome to The State Library

A world of knowledge and resources at your fingertips

Learn More



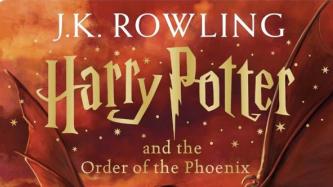
About The State Library

The State Library in New Delhi has been serving the community for decades. We provide access to thousands of books, journals, and digital resources.

Our mission is to make knowledge accessible to everyone. We host reading sessions, workshops, and cultural events for all age groups.

Contact Us

Featured Books



Project Demo

The State Library

Home Books About Contact Login

Books

Atomic Habits
James Clear
Self-Help

Midnight's Children
Salman Rushdie
Fiction

God of Small Things
Arundhati Roy
Fiction

The White Tiger
Aravind Adiga
Fiction

The Power of Habit
Charles Duhigg
Self-Help

A Brief History of Time
Stephen Hawking
Science

The Gene
Siddhartha Mukherjee
Science

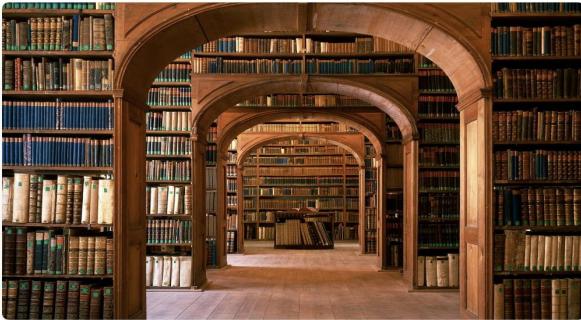
Harry Potter and the Philosopher's Stone
J.K. Rowling
Fiction

Project Demo

The State Library

Home Books About Contact Login

About The State Library



The State Library in New Delhi has been serving the community for decades. We provide access to thousands of books, journals, and digital resources.

Our mission is to make knowledge accessible to everyone. We host reading sessions, workshops, and cultural events for all age groups.

[Contact Us](#)

Community

Engaging programs and events for everyone.

Knowledge

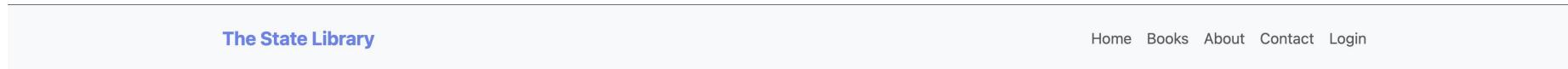
Thousands of books and digital resources.

Heritage

A library with a rich history serving the community.

© 2025 The State Library

Project Demo



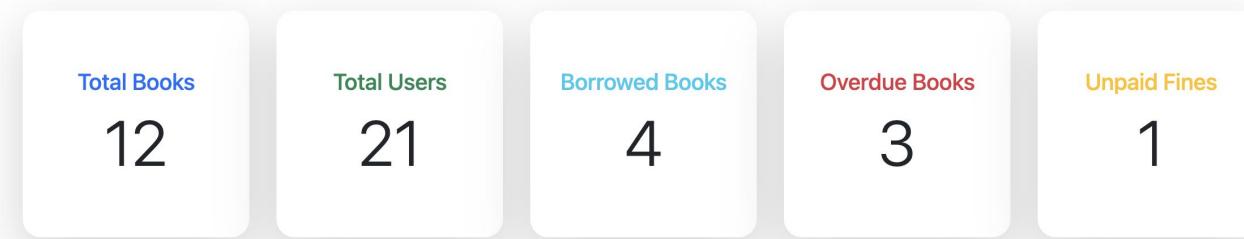
Admin Dashboard



Admin Dashboard

Welcome, Admin!

Summary



Admin Page

Admin Dashboard

Books Users Transactions Fines Logout

Users Management

Add New User Back to Dashboard

Aarav Sharma
Email: aarav.sharma@example.com
Phone No: 9876543210
Role: Student

Edit Delete

Riya Singh
Email: riya.singh@example.com
Phone No: 9876543211
Role: Student

Edit Delete

Vivaan Kumar
Email: vivaan.kumar@example.com
Phone No: 9876543212
Role: Student

Edit Delete

Ananya Gupta
Email: ananya.gupta@example.com
Phone No: 9876543213
Role: Student

Edit Delete

Aditya Patel
Email: aditya.patel@example.com
Phone No: 9876543214
Role: Student

Edit Delete

Isha Reddy
Email: isha.reddy@example.com
Phone No: 9876543215
Role: Student

Edit Delete

Kabir Jain
Email: kabir.jain@example.com
Phone No: 9876543216
Role: Student

Edit Delete

Saanvi Mehta
Email: saanvi.mehta@example.com
Phone No: 9876543217
Role: Student

Edit Delete

Arjun Nair
Email: arjun.nair@example.com
Phone No: 9876543218
Role: Student

Edit Delete

Diya Verma
Email: diya.verma@example.com
Phone No: 9876543219
Role: Student

Edit Delete

Rohan Choudhary
Email: rohan.choudhary@example.com
Phone No: 9876543220
Role: Student

Edit Delete

Anika Sharma
Email: anika.sharma@example.com
Phone No: 9876543221
Role: Student

Edit Delete



Admin Page

Admin Dashboard Books Users Transactions Fines Logout

Transaction Management

Search by user, book, or status... Add Transaction Back to Dashboard

All Borrowed Returned Overdue

ID	User	Phone	Book	Borrow Date	Due Date	Return Date	Status	Actions
34	Riya Singh	9876543211	Midnight's Children	8/21/25	8/31/25	-	BORROWED	<button>Edit</button> <button>Delete</button> <button>Return</button>
35	Vivaan Kumar	9876543212	God of Small Things	8/18/25	8/26/25	-	OVERDUE	<button>Edit</button> <button>Delete</button> <button>Return</button>
36	Ananya Gupta	9876543213	God of Small Things	8/25/25	8/31/25	-	BORROWED	<button>Edit</button> <button>Delete</button> <button>Return</button>
37	Aditya Patel	9876543214	A Brief History of Time	8/21/25	8/27/25	8/28/25	Returned	<button>Edit</button> <button>Delete</button> <button>View Fine</button>
38	Saanvi Mehta	9876543217	Python Crash Course	8/21/25	8/29/25	8/28/25	Returned	<button>Edit</button> <button>Delete</button>
39	Anika Sharma	9876543221	Head First Java	8/20/25	8/25/25	8/28/25	Returned	<button>Edit</button> <button>Delete</button> <button>View Fine</button>
40	Devansh Singh	9876543222	Head First Java	8/21/25	8/29/25	8/28/25	Returned	<button>Edit</button> <button>Delete</button>
41	Myra Kapoor	9876543223	The Gene	8/25/25	8/30/25	-	BORROWED	<button>Edit</button> <button>Delete</button> <button>Return</button>
42	Isha Reddy	9876543215	Deep Learning	8/20/25	8/25/25	-	OVERDUE	<button>Edit</button> <button>Delete</button> <button>Return</button>
43	Anika Sharma	9876543221	Learning SQL	8/21/25	8/27/25	8/28/25	Returned	<button>Edit</button> <button>Delete</button> <button>View Fine</button>
45	Isha Reddy	9876543215	God of Small Things	8/15/25	8/28/25	8/28/25	Returned	<button>Edit</button> <button>Delete</button>
46	Tanvi Joshi	9876543229	The Gene	8/22/25	8/31/25	-	BORROWED	<button>Edit</button> <button>Delete</button> <button>Return</button>
47	Meera Das	9876543226	Harry Potter and the Philosopher's Stone	8/20/25	8/27/25	-	OVERDUE	<button>Edit</button> <button>Delete</button> <button>Return</button>

Admin Page

Admin Dashboard Books Users Transactions Fines Logout

Fine Management

Add Fine Back to Dashboard

All View Paid Fines View Unpaid Fines

ID	User	Phone	Book	Transaction ID	Amount	Status	Actions
26	Aditya Patel	9876543214	A Brief History of Time	37	50	Paid	Edit Delete
27	Anika Sharma	9876543221	Head First Java	39	150	Paid	Edit Delete
28	Anika Sharma	9876543221	Learning SQL	43	50	Unpaid	Edit Delete Mark as Paid

API Testing

The screenshot shows the Postman application interface. On the left, the 'History' sidebar lists several API requests made today and on August 27, primarily targeting the 'books' and 'users' endpoints. A prominent request in the center is a GET request to 'http://localhost:8080/api/users'. The 'Headers' tab shows 'Content-Type: application/json'. The 'Body' tab displays a JSON response with three user objects:

```
1  [
2    {
3  "id": 23,
4  "name": "Aarav Sharma",
5  "email": "aarav.sharma@example.com",
6  "phone": "9876543210",
7  "role": "Student"
8  },
9  {
10 "id": 24,
11 "name": "Riya Singh",
12 "email": "riya.singh@example.com",
13 "phone": "9876543211",
14 "role": "Student"
15 },
16 {
17 "id": 25,
18 "name": "Vivaan Kumar"
```

The 'Test Results' section at the bottom indicates a successful response with a status of 200 OK, time 29 ms, and size 2.38 KB.

API Testing

The screenshot shows the Postman application interface. On the left, there's a sidebar with a history of requests, including several GET requests to `http://localhost:8080/api/books` and `http://localhost:8080/api/users`. The main area displays a single GET request to `http://localhost:8080/api/fines`. The request details show it's a GET method with the URL `http://localhost:8080/api/fines`. The 'Headers' tab indicates there are 6 headers. The 'Body' tab is selected, showing a JSON response with the following structure:

```
1  {
2   "id": 26,
3   "transaction": {
4     "id": 37,
5     "user": {
6       "id": 27,
7       "name": "Aditya Patel",
8       "email": "aditya.patel@example.com",
9       "phone": "9876543214",
10      "role": "Student"
11    },
12    "book": {
13      "id": 22,
14      "title": "A Brief History of Time",
15      "author": "Stephen Hawking",
16      "isbn": "80163",
17      "category": "Science",
18    }
19 }
```

The status bar at the bottom shows the response was successful (Status: 200 OK), took 37 ms, and was 126.05 KB in size.

API Testing

The screenshot shows the Postman application interface. At the top, there's a navigation bar with 'Home', 'Workspaces', 'Explore', a search bar 'Search Postman', and account options 'Sign In' and 'Create Account'. Below the navigation is a 'History' section showing a list of recent requests. A specific request is selected: 'GET http://localhost:8080/api/books/17'. The main workspace displays the request configuration and its response.

Request Configuration:

- Method: GET
- URL: http://localhost:8080/api/books/17
- Headers (6): This tab is selected.
- Body: None
- Pre-request Script: None
- Tests: None
- Settings: None

Query Params:

Key	Value
Key	

Response Body:

Status: 200 OK | Time: 103 ms | Size: 541.91 KB | Save Response

Pretty

```
1 {  
2   "id": 17,  
3   "title": "Atomic Habits",  
4   "author": "James Clear",  
5   "isbn": "41831",  
6   "category": "Self-Help",  
7   "subCategory": "Personal Growth",  
8   "year": 2018,  
9   "noOfCopies": 12,  
10  "imageUrl": "data:image/jpeg;base64,/9j/4AAQSkZJRGABAQAAAQABAAQ/  
11  2wCEAAUFBQkGQkjcQwTDg4Md4ZEG4SEBc0EBAXEcBxEB0XFBoQaxMTFx0lFxgjGBwZhxoYIRgjGCMiIiQnKCMXJzUBCQkFCQkJDAkJEyExDhoiHawYKSiGiIkIiIZLCloISIUIyIaHx8aGhwYH  
CIAjCQjhdkkIRcvGRoXiiEnHxckKv/CABEIAb0BiwmBPIgACEQEDEQH/xAA0AAABQBBAQAAAAAAAAAAAAAAwQFBgcBAggBQEBQEBQAAAAAAAAAAwBIEBQb/  
2gAMAwEAHADEAAAAPsgPo0dA4HTgdh0AAAAAAAQk07fxKfZLVyCTSs1wB95KzFXXyRL2UTIBKxhFwVysCzshMh41YmU0FtKRWErOvVzaArfmyeBzATSJXG9s9i6yS  
sACgAAAAAAAEZXJuUpDTIAKBVUlRHcJTbQAAAAAAAAAAAAAAZfPv0fkNjmtz3hGvhgaofQWJ7v87nmYlsEHVnrtei40FwmyT2ZqMXvxnqDZ4vtt  
+etwWVa1AAAAAAAAAAAdnE9AAAAAy17dw6ABzoH0grwCm6Unj0ARGSYARcoAcgAAAAAAJi2baVTbNkMo1yW91SHxb0aHD17p8xjdnd0tH1akx06fmTmnkzlvk+q/lX6iwKWF0eoU  
+zeaZ4xtfpRz1DItFfsrBj/qvnPX5d00d5Af0AAB49pgSHeljwclrnPQje/  
KhznoGy3sjeiueFeUosWK8899PPjwvCfeKVxu518ekVxJU8noC0AA0Ikxz3NuatOuNDdwWDf23h7xFcSVERY4iC4AIqHphQoAACIsAcRi/
```

API Testing

The screenshot shows the Postman application interface. On the left, there's a sidebar with a 'History' section listing various API requests made today and on August 27. The main area displays a 'GET' request to 'http://localhost:8080/api/transactions'. The 'Params' tab is selected, showing a single parameter 'Key' with a value 'Value'. Below the request details, the 'Body' tab is active, showing a JSON response structure. The response body is a JSON object with nested objects for 'user' and 'book'. The 'user' object contains fields like id, name, email, phone, and role. The 'book' object contains fields like id, title, author, isbn, category, subCategory, and year. At the bottom, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize', and 'JSON'. Status information at the bottom right indicates a 200 OK status with a time of 92 ms and a size of 601.53 KB.

```
1
2
3   "id": 34,
4   "user": {
5     "id": 24,
6     "name": "Riya Singh",
7     "email": "riya.singh@example.com",
8     "phone": "9876543211",
9     "role": "Student"
10   },
11   "book": {
12     "id": 18,
13     "title": "Midnight's Children",
14     "author": "Salman Rushdie",
15     "isbn": "76533",
16     "category": "Fiction",
17     "subCategory": "Historical",
18     "year": 1981,
```

Deployments

The screenshot shows the Docker Desktop application interface. The left sidebar has sections for Ask Gordon (BETA), Containers (selected), Images, Volumes, Builds, Models, and MCP Toolkit (BETA). The Docker Hub and Docker Scout sections are collapsed. The main area is titled "Containers" with a "Give feedback" link. It displays "Container CPU usage" at 67.37% / 800% (8 CPUs available) and "Container memory usage" at 2.47GB / 3.74GB. A "Show charts" button is present. A search bar and a "Only show running containers" toggle are also shown. The table lists five containers:

	Name	Container ID	Image	Port(s)	CPU (%)	Actions
<input type="checkbox"/>	minikube	9d479e26fb44	k8s-minikube	52422:22 ↗ Show all ports (5)	65.91%	
<input type="checkbox"/>	project	-	-	-	1.46%	
<input type="checkbox"/>	mysqlDb	1e2ac1ebd352	mysql:8.0	3307:3306 ↗	0.97%	
<input type="checkbox"/>	springboot-back	8dc33292e2d6	Ims-backer	8080:8080 ↗	0.49%	
<input type="checkbox"/>	angular-front	f07944bfe123	library-front	80:80 ↗	0%	

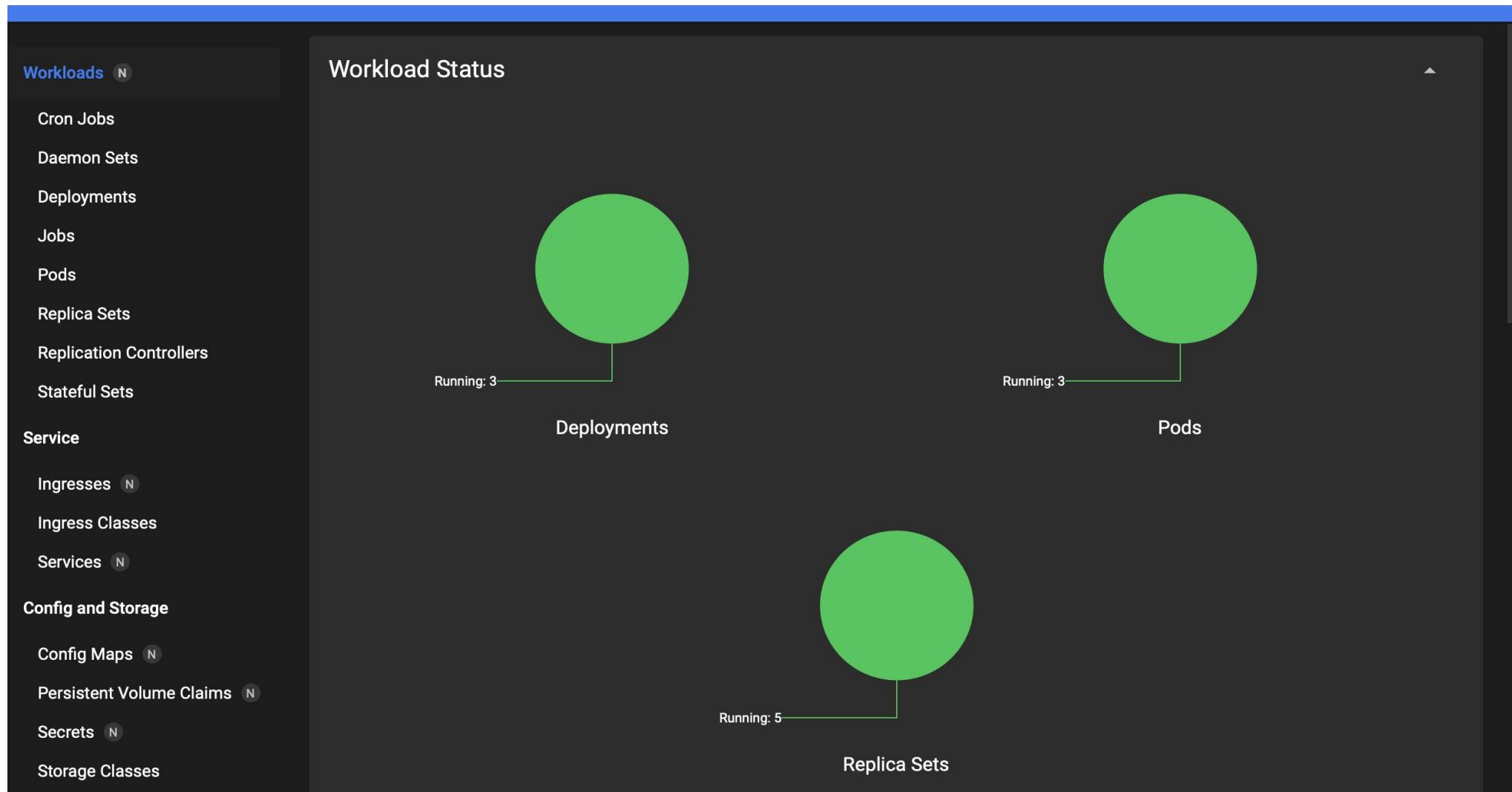
Showing 5 items

Deployments

```
faizanaslam@Faizans-MacBook-Air k8s % minikube service frontend
  Starting tunnel for service frontend.
  ┌─────────┐ ┌─────────┐ ┌─────────┐ ┌─────────┐
  │ NAMESPACE │ ┌─────────┐ ┌─────────┐ ┌─────────┐
  │           │ ┌─────────┐ ┌─────────┐ ┌─────────┐
  │ default   │   NAME    TARGET PORT URL
  │           │ ┌─────────┐ ┌─────────┐ ┌─────────┐
  │           │   frontend          http://127.0.0.1:53803
  └─────────┘ └─────────┘ └─────────┘ └─────────┘
  Opening service default/frontend in default browser...
! Because you are using a Docker driver on darwin, the terminal needs to be open to run it.

Ln 18, Col 33  Spaces: 2  UTF-8  LF  {} YAML  ⚙  ⚡ Go Live  ✓ Prettify
```

Deployments



Deployments



The screenshot shows a Jenkins pipeline log for job #14. The log output is as follows:

```
[Pipeline] dir
Running in /Users/faizanaslam/.jenkins/workspace/Library/k8s
[Pipeline] {
[Pipeline] sh
+ kubectl apply -f .
deployment.apps/springboot-backend configured
service/backend unchanged
deployment.apps/angular-frontend configured
service/frontend unchanged
persistentvolumeclaim/mysql-pvc unchanged
deployment.apps/mysqlDb unchanged
service/mysqlDb unchanged
[Pipeline] }
[Pipeline] // dir
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Deployments

Jenkins / Library / #14 / Pipeline Overview

#14

Manually run by admin Started 20 min ago Queued 58 ms Took 1 min 23 sec

Graph

```
graph LR; Start((Start)) --> CheckoutSCM((Checkout SCM)); CheckoutSCM --> Checkout((Checkout)); Checkout --> BuildBackend((Build Backend)); BuildBackend --> BuildFrontend((Build Frontend)); BuildFrontend --> PushImages((Push Images)); PushImages --> DeployKubernetes((Deploy to Kubernetes)); DeployKubernetes --> End((End))
```

Search Jenkins

Deploy to Kubernetes 1.9s Started 19m ago

Checkout SCM 1.2s

Checkout 1.0s

Build Backend 16s

Build Frontend 27s

Push Images 32s

Deploy to Kubernetes 1.9s

```
kubectl apply -f .  
+ kubectl apply -f .  
deployment.apps/springboot-backend configured  
service/backend unchanged  
deployment.apps/angular-frontend configured  
service/frontend unchanged  
persistentvolumeclaim/mysql-pvc unchanged  
deployment.apps/mysqlldb unchanged  
service/mysqlldb unchanged
```

THANK YOU