Store.it

CS253 Course Project

localghosts

Our Team

Course Instructor: Dr. Indranil Saha

Mentor TA: Mr. Swastik Maiti

Group Members:

Akanksha Singh	200070
Antreev Singh Brar	190163
Bhuvan Singla	180199
Deepankur Kansal	180226
Dipanshu Garg	190306
Harshit Raj	200433
Hitesh Anand	200449
Manas Gupta	200554
Priya Gole	200727
Tushar	190915

Table of contents

01 Introduction

Aim
Audience
Functionality

02 Implementation

Architecture
Frontend
Backend
Database

03 Dev Tools

VCS and Repo
CI/CD Pipelines
Design Tools

04 Services

Hosting Service
OTP Service
Auth Service
Error logs

05 Team Work

How did we coordinate?

06 UI Demo

07 Testing

08 Endnotes

Future Plans
Acknowledgment

I

Introduction

The product, store.it aims to digitize various physical stores and service providers at the IIT Kanpur campus

Intended Audience

IITK Vendors and Campus community

Classification

Buyer: Those who'll buy product/services i.e. the campus residents.

Seller: Those who sell products/services to the campus residents.

Functionalities

Seller

Create a seller account / store.

Add/delete/modify categories in catalog.

Add/delete/modify products/services in categories.

Update the inventory and mark an item out of stock.

View the list of orders placed/in-process/dispatched/delivered.

Functionalities

Buyer

Sign in/sign up to the account via email.

Check status of previous orders (Acknowledged/Dispatched/Delivered).

Add/delete/modify categories in catalog.

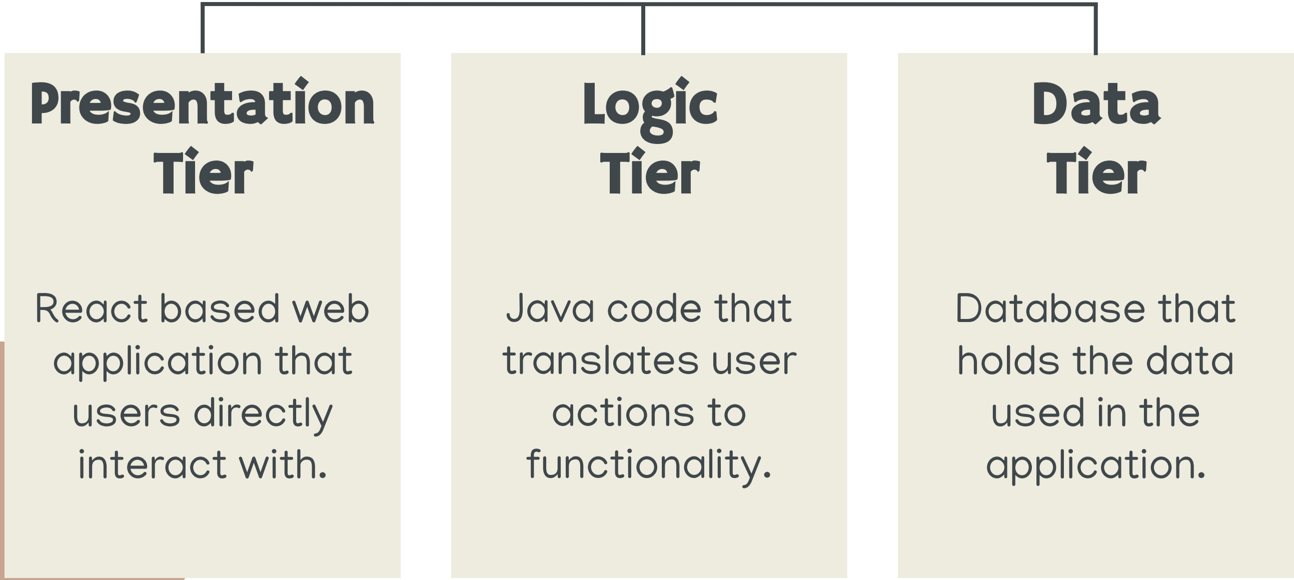
Update the inventory and mark an item out of stock.

View the list of orders placed/in-process/dispatched/delivered.

02

Implementation

Three Tier Architecture



Presentation Tier

React based web application that users directly interact with.

Logic Tier

Java code that translates user actions to functionality.

Data Tier

Database that holds the data used in the application.

Frontend

React JS	MUI
Javascript framework that makes the website dynamic	Material UI: A customizable library of UI components

Backend

Backend is implemented in **Java.**

We used **Spring Boot,** an open-sourced java based framework used to create web services on top of Java.

Database

We have used **PostgreSQL,** an object relational database system.

Database is hosted on **AWS (via Heroku).**

03

Development Tools

Development & Version Control Environment

Git

Used as our version control system

GitHub

used to manage our repositories and collaborate

CI/CD Pipeline

CI/CD is a method to frequently deliver apps to customers by introducing automation into the stages of app development.

Development: Added actions that run on every pull request and detect any possible linting errors and code smells.

Deployment: Set up actions on our repo that continuously deploy the main branch of our applications' repo to the Heroku cloud. [frontend, backend]

Design Tool

We have used Figma, a web-based graphics editing and user interface design app to design the frontend UI.

04

Services

Hosting Service

Used **Heroku,** a platform as a service (PaaS), enables developers to build, run and operate applications entirely in the cloud.

All three tiers of our application, the frontend, the backend and the database run on the Heroku cloud.

OTP Service

SendGrid

We use SendGrid's SMTP servers to send our OTPs to users via email.

Authentication and Authorization

Authentication: stored the email and password in our database. The password uses **cryptographic** techniques of **hashing and salting** to prevent the data being exposed in case of data leaks.

Authorization: we have made use of **JWT (Json Web Token)**

Production Error Handling

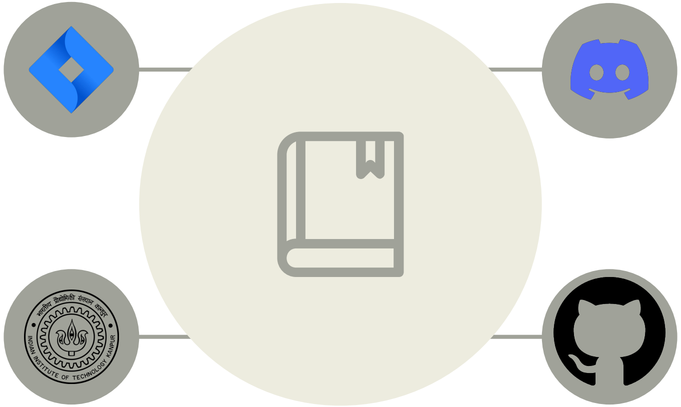
Sentry

We have used Sentry, a crash reporting platform to identify, investigate, and track the roadmap to fix bugs in production setup. [link]

05

Team coordination

How did we coordinate?



Jira

Agile Project Management

KD Labs

Met at KD Lab to work

Discord

Asynchronous communication via discord server

GitHub

Code Collab Tool

06

UI Demo

Let’s check out our website

07

Testing

Testing

We used a hybrid approach where some tests were automated while some were done manually.

For Unit Testing and Integration Testing, we used SpringBootTest framework and JUnit.

For System Testing we used ApacheBench, a CLI tool used to benchmark HTTP web servers. ApacheBench (ab) measures the performance of a web server by inundating it with HTTP requests and recording metrics for latency and success.[logs]

Future Development Plans

- Adding an online payment option.
- Deploying a corresponding mobile app.
- Customer feedback.
- Possibility of using other languages in the future.

Acknowledgement

We would like to thank the Instructor of the course**, Dr. Indranil Saha,** for teaching us Software Development and Operations concepts.

We would also like to thank the TA in-charge**, Mr. Swastik Maiti,** for guiding us throughout the process of making this software.

Thank You