ARPO

Implementation Document

Version 1.1

Prepared by

Group #: 14 **Group Name: Anonymous**

Shubhan R	200971	shubhan.ravi@gmail.com
Akansh Agrawal	180050	akanshcs2020@gmail.com
Akshan Agrawal	180061	akshancs2020@gmail.com
Lakshay Rastogi	180378	rastogilakshay31@gmail.com
Atharv Tyagi	170174	atharvlalittyagi@gmail.com
Jahnavi Kairamkonda	200482	jahnavi26k@gmail.com
Pranav Singh	190622	pranavsingh560@gmail.com
Manoj Kumar	180409	manojkumar.nvsd@gmail.com
Keshav Kumar	180353	keshavkr54321@gmail.com
Piyush Senwar	180512	psenwar@gmail.com

Course: **CS253**

Mentor TA: **Shatroopa Saxena**

Date: **28th April 2022**

Content

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Akansh Agrawal Jahnavi Kairamkonda Shubhan R Akshan Agrawal Pranav Singh Manoj Kumar Atharv Tyagi Keshav Kumar Piyush Senwar Lakshay Rastogi	First Version (Version 1.0): First Implementation Document for ARPO	20/03/2022
1.1	Akansh Agrawal Jahnavi Kairamkonda Shubhan R Akshan Agrawal Pranav Singh Manoj Kumar Atharv Tyagi Keshav Kumar Piyush Senwar Lakshay Rastogi	Added logo to cover page	27/04/2022

1 Implementation Details

We used the MVC architecture to implement. The following content explains our choices in more detail.

Frontend - Node.js with React

We used Node.js with React for the frontend because:-

- **High server load:** Using Node.js with React is good here as our web application needs handling of multiple requests and maintaining server load balance.
- **Real-time data:** Node.js is highly advisable for continued server connection.
- **JSON APIs:** Building JSON APIs is very efficient with Node.js due to high code reusability and easy code sharing in Reactjs.

Backend - Java with Spring Boot

We used Java because:-

- **Simplicity**- Being one of the oldest programming languages that was developed, Java provides a lot of developer support for working. Spring Boot helps set up the project very quickly, reducing the amount of source code required.
- **Cross platform** - Being an object-oriented compiled language, Java allows you to write the code once and run it anywhere on any platform (Windows, Mac OS, and Linux), making it a perfect choice for mobile application development, web development, database connectivity, networking, and many more.
- **Multi-Threading** - Spring Boot uses a multi-threaded web server to process each request in a separate thread. That enables one to perform several tasks simultaneously without queueing the events. This makes Spring Boot applications fast and robust.
- **Open source libraries**-There are numerous Java libraries and dependencies supported by experienced Java developers that can be used with Spring Boot. The use of such libraries significantly accelerates the back-end development of web apps by reducing the amount of code to be written.
- **Robust & scalable**-The automatic memory management and garbage collection make Java highly scalable and speeds up the development of web applications. Java has a strong type checking mechanism which makes Java robust.

Databases - MySQL

We used MySQL because:-

- **Open-source and compatible**
- **Fast and reliable:** MySQL was developed for speed. It is also known for its reliability as a database administrator, backed by a large community of programmers that have put the code through tough testing.
- **Availability:** MySQL also uses a variety of backup and recovery strategies to ensure data is not lost in the event of a system crash or unintentional delete.
- **Scalability:** MySQL can be scaled in different ways, typically via replication, clustering or sharding (or a combination of them). It is able to support and process very large databases.
- **Security:** MySQL offers encryption using the Secure Sockets Layer (SSL) protocol, data masking, authentication plugins, and other layers of security to protect data integrity. The MySQL Enterprise package also includes firewall protection against cyberattacks.

References:

- https://scand.com/company/blog/why-use-java-for-back-end-development/#:~:text=Being%20an%20object%2Doriented%20compiled,%2C%20networking%2C%20and%20many%20more.
- https://www.simform.com/blog/use-nodejs-with-react/
- https://www.jobsity.com/blog/5-reasons-why-mysql-is-still-the-go-to-database-management-system

2 Codebase

Github Repositories: *https://github.com/ARPO-Academics-Issues-Redressal-Portal*

How to navigate:
There will be 2 repositories in the GitHub organization:-

- Backend
- Frontend

Backend:

- backend/src/ - contains the functional files of backend.
- backend/src/main/java/com/arpo/backend/ - contains directories for various schema and respective java files inside them. Link to the database schema: https://docs.google.com/document/d/1A4H3qt9l1-MuUP6TOe7RCE-Xz-tijDAvOVkuu1Fdltg/edit
- Each schema is made up of 4 files:-
    - **Java Class -** This file is responsible for creating an outline for saving the tuples that would be obtained from tables.
    - **Controller** - This file responsible for exposing the services available to the web, by specifying the routes at which a particular API would be available.
    - **Service** - This file is responsible for setting up the repository functions that will be called when a particular service is called. Usually all the business logic for the particular functionality that the service is offering should go here.
    - **Repo** - This file is responsible for making the interactions between the backend service and the database possible. This is specifically used when a particular package - JPA Repository - is used. This package helps make the above process of linking and working with the database easier by abstracting out a lot of redundant details, and making the queries with the database easy to code.

Frontend:

- frontend/my-arpo/public/ - contains files used globally in the front end.
- frontend/my-arpo/src/ - contains functional files of frontend. Further inside this:-
    - frontend/my-arpo/src/pages/ - contains all the UI files for the actors interface.
    - frontend/my-arpo/src/components/ - contains common components used across all interfaces, analogous to parent class.
    - frontend/my-arpo/src/Routes/ - contains all the different routes available in a page.
    - frontend/my-arpo/src/assets/ARPO-logos/ - contains images and icons used in different pages
    - frontend/my-arpo/src/URL/ - contains information about where the website would be deployed to.

3 Completeness

We have implemented all the features from the SRS.

Future releases will include:-

- Integration with the existing ecosystem of IIT Kanpur and deploy this for all the stakeholders.
- Improvement in UI

Appendix A - Group Log

There has been a continuous interaction among the team members in meets as well as informally through WhatsApp group/calls. The team members were also in direct touch with the assigned TA through the official Whatsapp Group and various zoom meetings for feedback and suggestions. The majority of the members were new to the software implementation part and acquired the various skill sets from scratch, sometimes spending more than 6 hours a day even during the mid sem break. Since we followed MVC architecture, we divided our team into 3 groups as well accordingly, where some members were in more than one group too.

The tabular detail of some official group meets are indicated as follows:

Meeting Minutes	Agenda
Feb 17, 2022 04:30 PM-05:00 PM	Group Meet, discussion on pathway for implementation and discussion on various resources to acquire the skill sets required.
Feb 28, 2022 7:30 PM-8:00 PM	Group Meet with the assigned TA for the discussion and feedback on various ends of the software implementation process.
March 2 ,2022 5:00 PM - 6:30 PM	Group Meet for Database Schema Design-I
March 3, 2022 2:30 PM-3:00 PM	Group Meet for frontend discussion and update on the skill sets acquired and work done.
March 4, 2022 8:30 PM - 10:00 PM	Group Meet for Database Schema Design-II
March 5, 2022 8:00 PM - 10:00 PM	Group Meet for creating all the designed tables along with some entries in the mysql database using mysql workbench
March 6, 2022 1:00 AM - 2:30 AM	Group Meet for Implementing JDBC for Java to connect with mysql Database
March 6, 2022 8:00 PM - 9:00 PM	Group Meet for backend discussion
March 8, 2022 5:30 PM-6:30 PM	Group Meet for frontend discussion and feedback for the basic code structure made
March 8, 2022 5:00 PM - 6:00 PM	Group meet for backend discussion and update taken on the database schema designs

March 8, 2022 11:00 PM-2:00 AM	Group Meet for frontend update on the making of various pages/components
March 9, 2022 9:00 PM-9:30 PM	Group Meet for frontend discussion for the update on feedback taken from the last meet
March 10, 2022 3:00 PM-4:00 PM	Group Meet on backend discussion and discussion on the working structures designed for various database schemas
March 11, 2022 5:30 PM-7:00 PM	Group Meet so that everyone can work on a common mysql database using remote host and grant privilege option.
March 12, 2022 12:10 AM-3:30 AM	Group Meet for frontend update on making of pages for different actors
March 14, 2022 3:20 PM-4:30 PM	Group Meet for frontend update on styling and linking of various components and pages
March 15, 2022 11:30 PM-3:30 AM	Group Meet for frontend update on styling done as per the feedback from last meet and making of leftover pages for each interface
March 16, 2022 3:35 PM - 4:00 PM	Group Meet for frontend discussion on linking of pages
March 16, 2022 9:00 PM - 11:00 PM	Group discussion and updates on the backend, frontend and integration