**CS673 Software Engineering** 

**Team 2 - ProManager**

**Deployment Planning**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Dipayan Mazumder | Security Lead | *DM* | 05-10-2024 |
| Pranjal Ekhande | Requirement Lead | *PE* | 05-10-2024 |
| Aman Jain | Configuration Lead | *AJ* | 05-10-2024 |
| Mukul Jangid | Design and Implementation lead | *MJ* | 05-10-2024 |
| Praveen Singh | Team Lead | *PS* | 05-10-2024 |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| 1.0 | Praveen Singh | 06-11-2024 | First Draft |
| 2.0 | Pranjal Ekhande | 05-17-2024 | Final Draft |
|  |  |  |  |
|  |  |  |  |

**Heroku Deployment Guide for ProManager**

This guide will walk you through the process of deploying the ProManager application to Heroku using a PostgreSQL database.

**Prerequisites**

* Heroku CLI installed (<https://devcenter.heroku.com/articles/heroku-cli>)
* Git installed
* Java 8 or higher installed

**Step 1: Create a Heroku App**

* Open a terminal or command prompt.
* Log in to your Heroku account by running heroku login.
* Create a new Heroku app by running heroku create your-app-name.

**Step 2: Provision a PostgreSQL Database**

* Run heroku addons:create heroku-postgresql --app your-app-name to provision a PostgreSQL database for your app.
* Or you can use the User interface in Heroku to do that.
* The command output will display the database credentials, including the database URL. Make a note of this URL as you'll need it later.

**Step 3: Configure the Database Connection**

* In your project's root directory, locate the src/main/resources/application.properties file.
* Replace the existing database configuration with the database URL from Heroku, using the following format:

Copy

* + spring.datasource.url=<HEROKU\_DATABASE\_URL>
  + spring.datasource.username=<HEROKU\_DATABASE\_USERNAME>
  + spring.datasource.password=<HEROKU\_DATABASE\_PASSWORD>

Replace <HEROKU\_DATABASE\_URL>, <HEROKU\_DATABASE\_USERNAME>, and <HEROKU\_DATABASE\_PASSWORD> with the appropriate values from the Heroku database credentials.

## **Step 4: Build the Application**

1. In your project's root directory, run mvn clean package to build the application.

## **Step 5: Deploy to Heroku**

1. **Build the Docker Image** Build the Docker image by running the following command in your project's root directory:  
   Copy  
   docker build -t registry.heroku.com/promanager-v1/web .
2. **Push the Docker Image to Heroku** Push the Docker image to the Heroku Container Registry:  
   Copy  
   docker push registry.heroku.com/promanager-v1/web
3. **Release the Docker Container** Release the Docker container on Heroku:  
   Copy  
   heroku container:release web -a promanager-v1
4. **View Application Logs** To view the application logs, run:  
   Copy  
   heroku logs --tail -a promanager-v1  
   This command will display the logs in real-time, allowing you to monitor the deployment process and application behavior.

After completing these steps, your ProManager application will be deployed and running on Heroku using the Docker container.

## **Step 6: Open the Application**

1. After the deployments are complete, open the application by running heroku open.

Note: If you have any additional configuration or environment variables required by your application, you can set them using the heroku config:set command. For example, heroku config:set SPRING\_PROFILES\_ACTIVE=production.