DevOps Bootcamp**

ParkPulse/PropPulse

Date of Submission: 25/03/2024

Reporting Period: 01/02/2024 to 15/03/2024

Group name: ParkPulse
Group members: Bahaa Abou Shakra, RimahHousamedin, Aleen Zeidan, Jalal Sobh & Raafat Al Halabi

Instructor: Elie Tahchi

Technical Advisor: Jana Kerbaj

Team Representative: Bahaa Abou Shakra

Table of Content

Overview of the project	*pg. 2*
Objectives and goals	*pg. 3*
Summary of the project's overall progress	*pg. 3*
Distribution of tasks per team members	*pg. 6*
K8s architecture	*pg. 6*
Documentations	*pg. 7*
Project link	*pg. 7*
References	*pg. 7*
Conclusions and lessons learned	*pg. 8*

Section I: Overview of the Project

This project aims to develop and deploy a scalable web application, tailored for managing internal operations in medical facilities such as hospitals and polyclinics, using a CI/CD pipeline managed locally with GitHub, Jenkins, DockerHub and Minikube. The CI/CD pipeline will automate the build, test and deployment processes, ensuring continuous integration and delivery of updates. DockerHub will serve as the repository for Docker images in the application.

For local development and testing, Minikube is utilized to create a local Kubernetes cluster. This allows developers to simulate the deployment environment and test changes before pushing them to production. Cloudways is utilized for global deployment, providing a managed hosting solution for the application.

To monitor the application's performance and health, Grafana and Prometheus are integrated into the deployment. Grafana will provide visualizations and dashboards for monitoring various metrics, while Prometheus will collect and store the metrics data.

Overall, this project combines various DevOps tools and practices to streamline the development, deployment and monitoring processes of a web application, ensuring reliability, scalability and efficiency throughout its lifecycle.

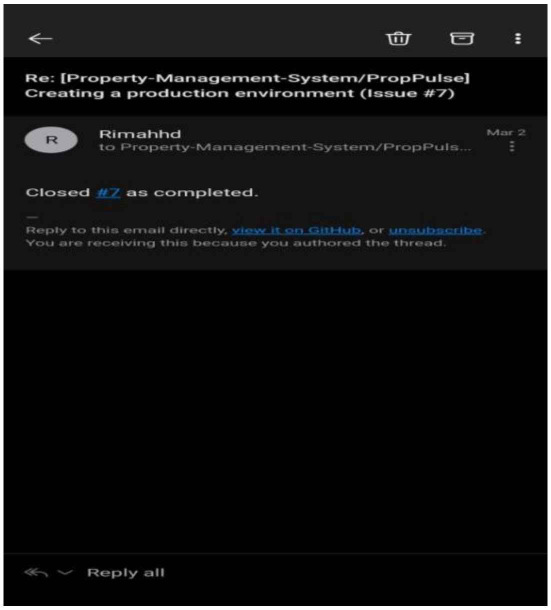
ROPPULSE

Section II: Objectives and Goals

Applying DevOps and agile project management principles in order to develop and deploy a dynamic webapp in a professional manner, with the potentiality of up-scaling and downscaling based on the client's requirements; in addition to implementing the appropriate security features preventing any threat that might target this application.
Besides the above mentioned objectives, our goals are to continually improve our product to remain compatible, assuring users' satisfaction.

Section III: Summary of the Project's Overall Progress

- *Planning: Understanding the requirements of the application and defining the functionalities that involve interaction with a MySQL database.*
- *Utilizing Github Project and Github Issues (project management system) for managing our project's workflow.*

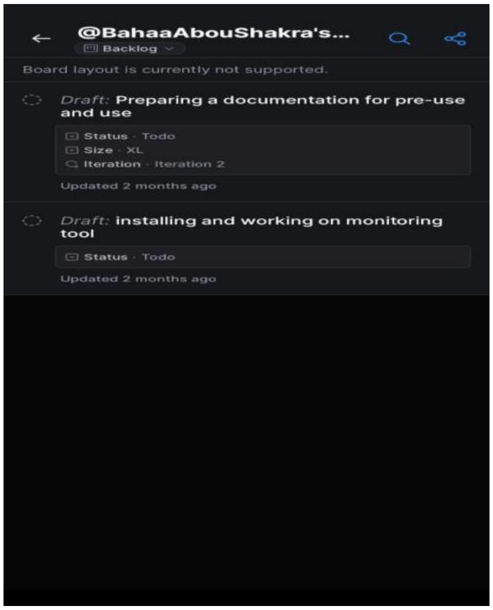


Re: [Property-Management-System/PropPulse] Creating a production environment (Issue #7)

Rimahhd
to Property-Management-System/PropPuls...
Mar 2

Closed #7 as completed.

—
Reply to this email directly, view it on GitHub, or unsubscribe.
You are receiving this because you authored the thread.



@BahaaAbouShakra's...
Backlog

Board layout is currently not supported.

Draft: Preparing a documentation for pre-use and use
Status · Todo
Size · XL
Iteration · Iteration 2
Updated 2 months ago

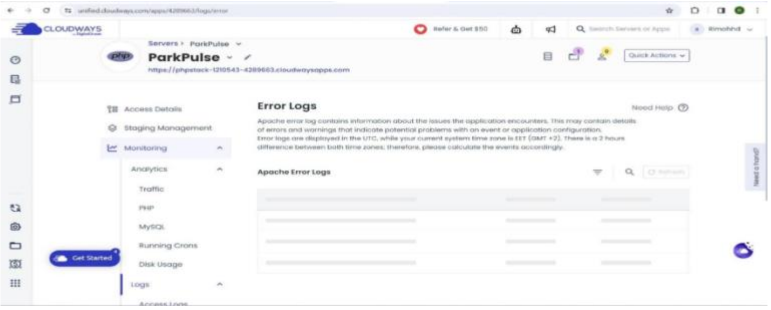
Draft: installing and working on monitoring tool
Status · Todo
Updated 2 months ago

- *Downloading XAMP by visiting the official website and downloading the appropriate windows version.*
- *Starting XAMP once installed by opening the control panel and starting the Apache web server and MySQL database server.*
- *Accessing phpMyAdmin by navigating to http://localhost/phpmyadmin/ where the MySQL database can be managed.*
- *Creating a database in phpMyAdmin.*
- *Finding a PHP MySQL template through GitHub platform.*
- *Cloning the template to local server.*
- *Configuring the database connection using VS code.*

- *Customizing the template files according to the project needs.*
- *Testing the changes by navigating to http://localhost/<template_folder_name>/ .*
- *Debugging errors.*

- *Starting the deployment process.*
- *Version control setup: The source code is successfully hosted on GitHub, allowing for version control and collaborative development.*
- *CI/CD Pipeline Implementation: Jenkins has been configured to automate the Continuous Integration and Continuous Deployment pipeline. This enables automated builds, tests and deployments whenever code changes are pushed to GitHub.*
- *Containerization with DockerHub: DockerHub is utilized to manage containerized versions of the application, ensuring consistency and ease of deployment across different environments.*
- *Local Deployment with Minikube: Minikube has been setup to create a local Kubernetes cluster, allowing developers to simulate the production environment for testing and debugging purposes.*

- *Global Deployment via Cloudways: Cloudways is employed for global deployment, providing scalable infrastructure and management tools hosting the application.*

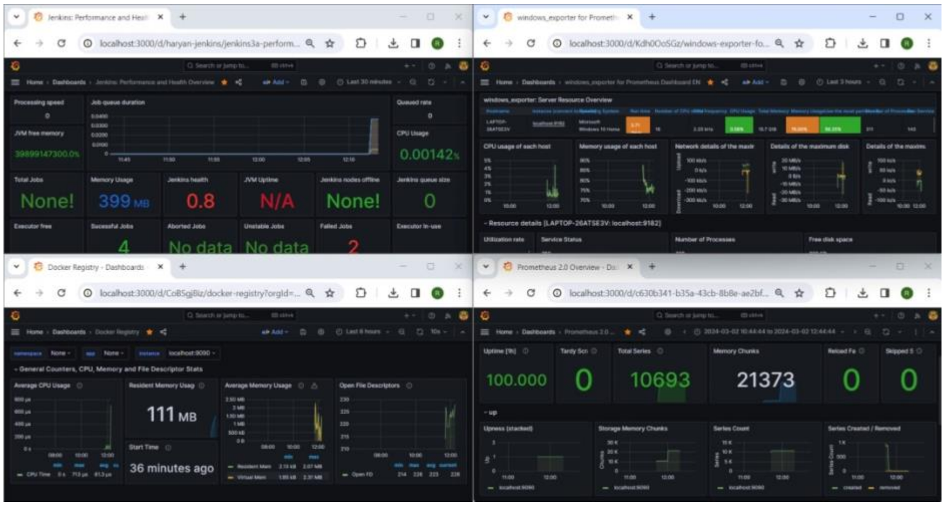


Error Logs

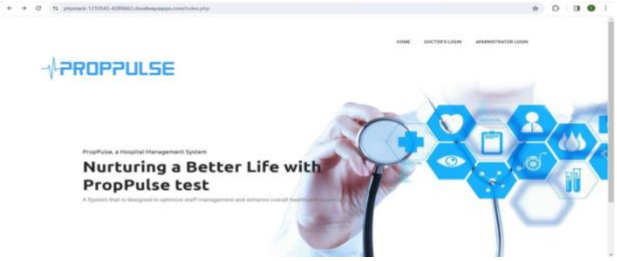
Apache error log contains information about the issues the application encounters. This may contain details of errors and warnings that indicate potential problems with an event or application configuration. Error logs are displayed in the UTC, while your current system time zone is EET (GMT +2). There is a 2 hours difference between both time zones; therefore, please calculate the events accordingly.

Apache Error Logs

- *Monitoring with Grafana and Prometheus: Grafana and Prometheus have been integrated into the deployment to monitor various metrics related to the application, infrastructure and local server. This ensures proactive monitoring and troubleshooting of performance issues.*


Jenkins: Performance and Heal
localhost:3000/d/haryan-jenkins/jenkins3a-perform...
Home › Dashboards › Jenkins: Performance and Health Overview
Last 30 minutes
Processing speed
0
Job queue duration
Queued rate
0
JVM free memory
39899147300.0%
CPU Usage
0.00142%
Total Jobs
None!
Memory Usage
399 MB
Jenkins health
0.8
JVM Uptime
N/A
Jenkins nodes offline
None!
Jenkins queue size
0
Executor free
Successful Jobs
4
Aborted Jobs
No data
Unstable Jobs
No data
Failed Jobs
2
Executor in-use
windows_exporter for Prometh
localhost:3000/d/Kdh0OoSGz/windows-exporter-fo...
Home › Dashboards › windows_exporter for Prometheus Dashboard EN
Last 3 hours
windows_exporter: Server Resource Overview
CPU usage of each host
Memory usage of each host
Network details of the main
Details of the maximum disk
Details of the maxim
Resource details [LAPTOP-26ATSE3V: localhost:9182]
Utilization rate
Service Status
Number of Processes
Free disk space
Docker Registry - Dashboards
localhost:3000/d/CoB5gjBiz/docker-registry?orgId=...
Home › Dashboards › Docker Registry
Last 6 hours
instance localhost:9090
General Counters, CPU, Memory and File Descriptor Stats
Average CPU Usage
Resident Memory Usag
111 MB
Start Time
36 minutes ago
Average Memory Usage
Open File Descriptors
Prometheus 2.0 Overview - Da
localhost:3000/d/c630b341-b35a-43cb-8b8e-ae2bf...
Home › Dashboards › Prometheus 2.0 ...
2024-03-02 10:44:44 to 2024-03-02 12:44:44
Uptime [1h]
100.000
Tardy Scr
0
Total Series
10693
Memory Chunks
21373
Reload Fa
0
Skipped S
0
up
Upness (stacked)
Storage Memory Chunks
Series Count
Series Created / Removed

- *Security features implemented: Private repositories on Github and Dockerhub ensuring limited user access; Jump server implementation between the frontend and backend/database ensuring that user access is applicable to the frontend only; User access allowance on the PropPulse application (separating Doctor's from admin's access features); Using tokens and secret texts upon configuring the files instead of a username and password configuration method, so the password are kept invisible.*



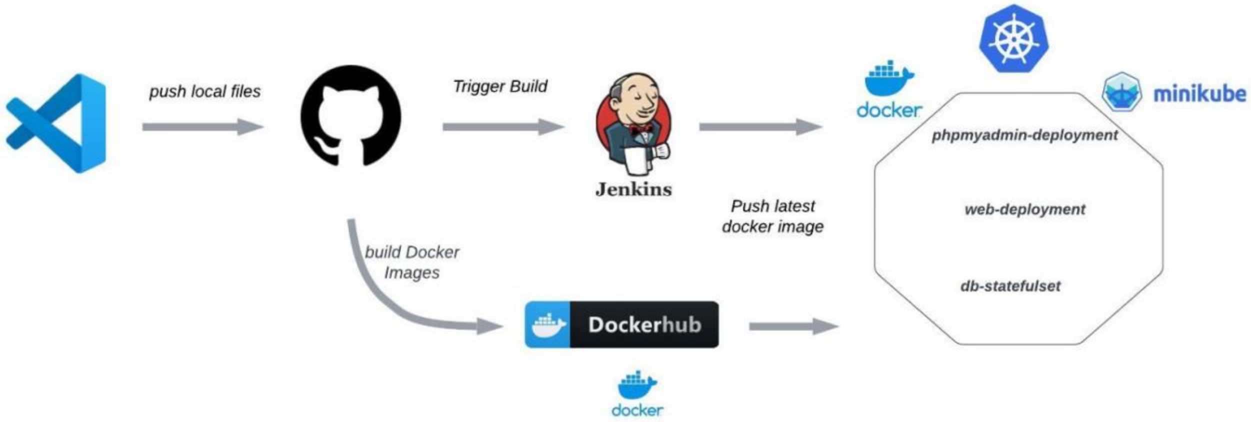
PropPulse, a Hospital Management System

Nurturing a Better Life with PropPulse test

Section IV: Distribution of Tasks per Team Members

- Overall project planning – Bahaa, Rimah& Aleen.
- Searching for an appropriate template – Rimah.
- Performing required changes to the code& debugging errors – Rimah& Aleen.
- App design (UX/UI) – Aleen.
- Local deployment (GitHub, Jenkins, DockerHub & Minikube) – Bahaa &Rimah.
- Global deployment (Cloudways) – Rimah.
- Monitoring and logging – Bahaa & Rimah.
- K8s architecture – Bahaa & Aleen.
- User/Admin documentation – Bahaa.
- Cloudways deployment process documentation – Rimah.
- Local CI/CD documentation – Bahaa, Rimah& Aleen.
- Finalizing weekly plans/reports – Bahaa.
- Project presentation plan/preparation – Bahaa, Rimah& Aleen.
- Security implementation – Bahaa & Rimah.
- Final report preperation – Bahaa.

Section V: K8s Architecture



Section V: Documentations

application guide

deployment on cloudways

deployment on minikube

monitoring

Section VI: Project Link

https://lnkd.in/dUCDR2aH

Section VII: References

.

- WAMPServer. (n.d.). Retrieved from https://www.wampserver.com/en/
- Apache Software Foundation. (n.d.). Apache HTTP Server Documentation. Retrieved from https://httpd.apache.org/docs/
- XAMPP. (n.d.). Retrieved from https://www.apachefriends.org/index.html
- phpMyAdmin. (n.d.). Retrieved from https://www.phpmyadmin.net/
- Docker Documentation. (n.d.). Retrieved from https://docs.docker.com/
- Docker Hub. (n.d.). Retrieved from https://hub.docker.com/
- SmartBear. (n.d.). Jenkins Documentation. Retrieved from https://www.jenkins.io/doc/
- Kubernetes. (n.d.). Minikube Documentation. Retrieved from https://minikube.sigs.k8s.io/docs/
- Cloudways. (n.d.). Retrieved from https://www.cloudways.com/en/

Section VIII: Conclusion and Lessons Learned

In conclusion, our journey of developing and deploying a dynamic web application using DevOps principles, agile methodologies, and leveraging both Minikube and Cloudways was both challenging and rewarding. Through our collaborative efforts, we have not only successfully delivered a robust and scalable solution but have also embraced a culture of continuous improvement and rapid iteration.

By incorporating DevOps practices into our workflow, we've streamlined our development and deployment processes, allowing for faster delivery of features and enhancements. The seamless integration of Agile methodologies ensured that we remained adaptable to changing requirements throughout the project lifecycle, ultimately delivering a product that meets the evolving needs of users.

Our utilization of Minikube and Cloud technologies enabled us to achieve scalability, reliability, and efficiency in our deployment strategies. Leveraging Minikube for local development and testing provided a consistent environment, while Cloud deployment allowed us to scale our application dynamically based on demand, ensuring optimal performance for end-users.

As we conclude this project, it's essential to reflect on the lessons learned and the valuable experiences gained. We've strengthened our collaboration skills, honed our technical expertise, and cultivated a mindset focused on innovation and continuous learning. Moving forward, let us carry forward these principles and practices, striving for excellence in all our endeavors.

In closing, I want to express my gratitude to each member of the team for their dedication, hard work, and unwavering commitment to excellence. Together, we've achieved success!