A report submitted to

RAMAIAH INSTITUTE OF TECHNOLOGY

Bengaluru

DEVOPS LAB PROJECT (ISL66)

as partial fulfilment of the requirement for the award of degree of

Bachelor of Engineering (B.E) in Information Science and Engineering

By

Shivani Tornekar 1MS21IS099

Srishti Shetty 1MS21IS105

Tushar N 1MS21IS117

Under the Guidance of

Shruthi JR

Assistant Professor

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

RAMAIAH INSTITUTE OF TECHNOLOGY

(Autonomous Institute, Affiliated to VTU) BANGALORE - 54

JUNE 2024

Department of Information Science and Engineering

Ramaiah Institute of Technology Bengaluru - 54

CERTIFICATE

This is to certify that Shivani Tornekar(USN- 1MS21IS099), Srishti Shetty(USN-1MS21IS105) and Tushar N (USN-1MS21IS117) who were working for their DEVOPS LAB PROJECT under my guidance, have completed the work asper my satisfaction. To the best of my understanding, the work to be submitted in the dissertation does not contain any work, which has been previously carried out by others and submitted by the candidates for themselves for the award of any degree anywhere.

Name and Signature of the Guide Signature of the HOD

Other Examiners

Name of the examiners Signature with date

1.
2.

DECLARATION

We hereby declare that the entire work embodied in this DEVOPS LAB PROJECT (ISL66) report has been carried out by us at Ramaiah Institute of Technology under the supervision of Mrs. Shanmuga Priya R This project report has not been submitted in part or full for the award of any diploma or degree of this or any other University.

Shivani Tornekar

1MS21IS099

Srishti Shetty

1MS21IS105

Tushar N

1MS21IS117

Place: Bangalore Date:

ABSTRACT

Continuous Integration/Continuous Delivery with GCP

This report presents a comprehensive overview of the development and deployment of the 'Secret Santa-App' project using Jenkins, emphasizing the use of various modern tools for efficient and scalable deployment. The project involved multiple steps, starting with setting up a Jenkins pipeline and pushing the project to a GitHub repository. The inclusion of a Docker file with an automation script facilitated containerization, ensuring that the application can run consistently across various environments. The core of the deployment strategy involved setting up Jenkins for continuous integration and deployment directly linked to the GitHub repository. This setup included configuring build triggers, specifying deployment parameters, and managing instances to ensure optimal performance. Additionally, detailed configurations were made to ensure the application listens on the correct port and handles deployment efficiently. The report delves into the motivation and scope of the project, the challenges faced, and the solutions implemented. It also provides an in-depth look at the CI/CD pipeline architecture and presents the outcomes of the deployment process, highlighting the benefits of using tools like JDK 11, Git, SonarQube Scanner, OWASP Dependency Check, and Docker for such projects.

ACKNOWLEDGEMENT

I would like to extend my deepest gratitude to my Devops Lab Guide, Prof. Shanmuga Priya R, for their invaluable guidance and support throughout this project. Their expertise and insights were crucial in navigating the complexities of deploying an application on Jenkins. I also want to thank my peers, whose constructive feedback and encouragement kept me motivated and focused.

I am also grateful to the community of developers and contributors who maintain the open-source tools and libraries that were integral to this project. The extensive documentation and resources available online were incredibly helpful.

Lastly, I acknowledge the extensive resources and documentation provided by Jenkins and GitHub, which greatly facilitated the learning and implementation process. Their platforms and tools have made modern software development and deployment significantly more accessible and efficient.

TABLE OF CONTENTS

Chapter 1 INTRODUCTION

The 'Secret Santa-App' project leverages modern DevOps practices to create an app that allows you to input names and assigns secret santa's to the individuals. By utilizing Jenkins for Continuous Integration and Continuous Deployment (CI/CD), the project integrates key tools such as JDK 11, Git, SonarQube Scanner, OWASP Dependency Check, and Docker. This combination facilitates automated build, test, and deployment processes, enhancing collaboration and reducing deployment time. The report provides an overview of the project's development, challenges faced, solutions implemented, and the benefits of using these advanced tools and practices for seamless application deployment.

1.1 MOTIVATION AND SCOPE

The motivation behind the 'Secret Santa-App' project is to create an efficient and easy to use platform to take part in a classic Christmas tradition of secret santa with multiple users and assigns their respective person whom they have to gift that leverages modern DevOps practices. Traditional deployment methods can be cumbersome, error-prone, and time-consuming, leading to delays and inconsistencies. By implementing a CI/CD pipeline using Jenkins and integrating tools such as JDK 11, Git, SonarQube Scanner, OWASP Dependency Check, and Docker, the project aims to automate and streamline the development and deployment processes. This approach enhances productivity, ensures code quality, and accelerates the delivery of updates and new features.

The scope of the 'Secret Santa-App' project is very helpful during Christmas festivities and negates the whole kerfuffle of having to pick chits and leaking information . This involves:

- Containerization: Using Docker to package the application and its dependencies, ensuring it runs consistently in any environment.

- Continuous Integration and Deployment: Leveraging Jenkins to automate the build, test, and deployment processes, facilitating seamless integration of new code changes.

- Code Quality Assurance: Employing SonarQube Scanner and OWASP Dependency Check to maintain high code quality and security.

- Scalability: Ensuring the application can handle varying loads efficiently through effective instance management and scaling.

- Collaboration: Utilizing Git for version control to enhance collaboration among team members.

By integrating these practices, the 'Secret Santa-App' project not only aims to deliver a seamless environment to enjoy a favourite Christmas event but also sets a precedent for modern, efficient, and scalable software for anyone to use and enjoy!

1.2 ISSUES AND CHALLENGES

- Initial Setup and Configuration: Integrating GitHub, Jenkins, and Docker involves complex steps and careful configuration of permissions and access controls. Misconfigurations can lead to deployment failures and security issues. Thorough planning and documentation are essential to navigate the intricacies of initial setup.

- Containerization: Creating an effective Dockerfile to containerize the 'Secret Santa-App' project is crucial but challenging. Ensuring environment consistency across development, staging, and production is vital to prevent deployment issues. Managing dependencies and avoiding version conflicts requires meticulous attention and regular updates.

- Continuous Integration and Deployment (CI/CD): Implementing a CI/CD pipeline automates build, test, and deployment processes, but developing robust automation scripts is complex. Pipeline failures due to code errors, dependencies, or configurations necessitate continuous monitoring and quick fixes. Optimizing the pipeline for speed and reliability is crucial to minimize downtime and maintain development velocity.

- Instance Management and Scaling: Effective instance management ensures availability and scalability of the application. Configuring Docker to maintain a minimum number of instances balances resource allocation to avoid unnecessary costs. Proper autoscaling policies adjust the number of

instances based on demand, preventing over-provisioning or under-provisioning, which can lead to increased costs or poor performance.

- Monitoring, Logging, and Security: Continuous monitoring and efficient log management are essential for tracking application performance and health. Setting up comprehensive monitoring tools helps detect and resolve issues promptly. Securing the application involves implementing robust access controls and encryption to protect sensitive data. Regular security audits and updates are necessary to defend against evolving threats.

1.3 PROBLEM STATEMENT

The 'Secret Santa-App' project addresses the need for an efficient, reliable, and easy environment for our favourite winter activity. Traditionally users would have to note down their names down on pieces of paper and draw names to buy presents for. By implementing a CI/CD pipeline with Jenkins and integrating tools like JDK 11,
Git, SonarQube Scanner, OWASP Dependency Check, and Docker, the project aims to automate and streamline development, testing, and deployment processes, ensuring high code quality, security, and scalability and automating this task.

Chapter 2

About the Tool/Architecture

2.1 CI/CD PIPELINE

The architecture of the CI/CD pipeline involves:

Source Code Push to GitHub:

- A developer writes and tests the source code for the Secret Santa-App portal.
  • Once the code is ready, the developer commits the changes and pushes the source code to a GitHub repository.
- GitHub serves as the Version Control System (VCS), managing code versions and facilitating collaboration.

Triggering Jenkins Build:

- GitHub is configured with a webhook that triggers Jenkins whenever new code is pushed.
- This webhook ensures that every code push initiates the build process automatically in Jenkins.

Building and Pushing Container Image:

- Jenkins retrieves the latest source code from the GitHub repository.
- It then builds the container image of the Secret Santa-App portal using the Docker file and specified build instructions.
- Once the container image is created, Jenkins pushes it to a Docker registry, such as Docker Hub.

Deploying the New Container Image:

- The container orchestration tool (like Kubernetes) pulls the latest container image from the Docker registry.
- It then updates the existing deployment by running the new container image, replacing the old version with the latest one.
- Kubernetes manages the deployment, ensuring high availability and reliability of the application.

Instance Management and Scaling:

- Kubernetes dynamically manages instances, scaling the number of instances based on demand.
- This ensures that the Secret Santa-App portal can handle varying loads efficiently while minimizing costs.

Monitoring, Logging and Security:

- Continuous monitoring and efficient log management are essential for tracking application performance and health.
- Setting up comprehensive monitoring tools helps detect and resolve issues promptly.
- Securing the application involves implementing robust access controls and encryption to protect sensitive data. Regular security audits and updates are necessary to defend against evolving threats.

2.2 Benefits of CI/CD:

- Shortened Feedback Loop and Cycle Time: Rapid feedback and quick iterations are possible, allowing for small, manageable updates to be delivered frequently.

- Improved Practice and Productivity: Regular, smaller updates help maintain a consistent development rhythm, enhancing productivity and reducing errors.

- Enhanced Security: Continuous integration and deployment ensure timely application of security updates and patches.
- Efficient Code Deployment: Teams can push code more efficiently, reducing the time from development to production.

- Organizational Benefits: The approach results in high-quality, robust, and reliable systems, providing significant value to the organization and improving responsiveness to user needs and feedback

This CI/CD pipeline, leveraging GitHub, and Jenkins, ensures that Cloudscribe remains up-to-date, secure, and highly available, ultimately benefiting both the User and the participants.

Chapter 3

CI/CD Pipeline Working & Implementation

3.1 CI/CD Pipeline for Secret Santa-App Deployment

The CI/CD pipeline for Secret Santa-App involves several key steps and components to ensure a robust, scalable, and maintainable deployment. Below is a detailed description of each step involved in the pipeline:

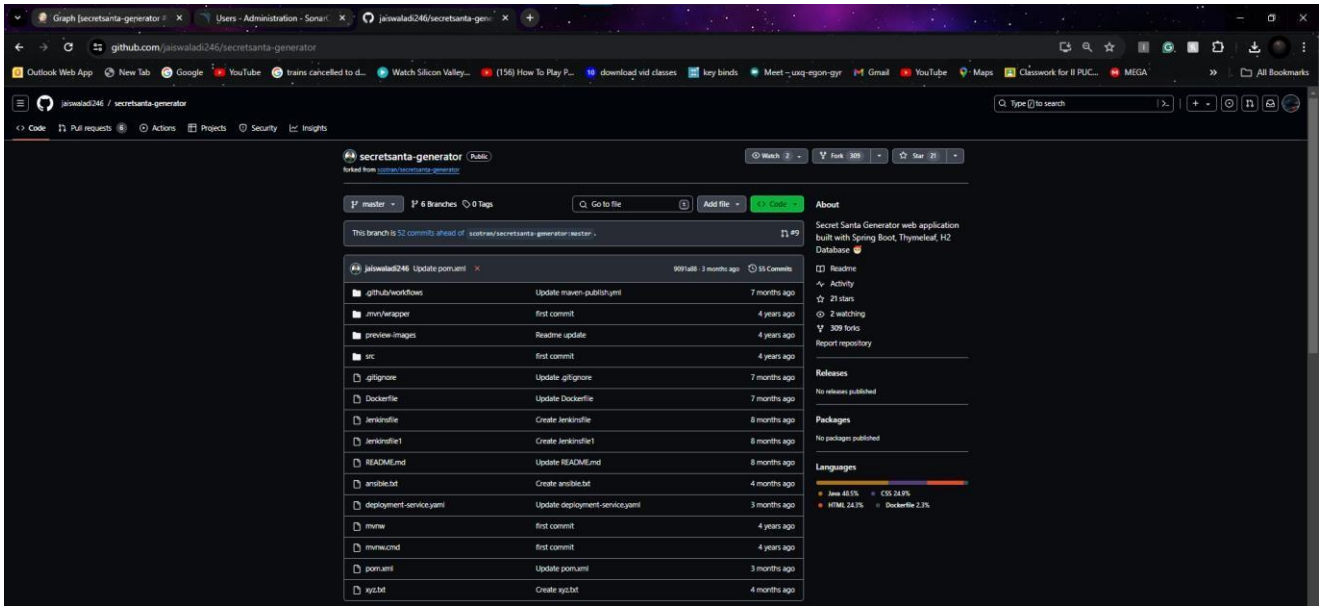
1. Setting up Jenkins

- Installation: Install Jenkins on a server or use a hosted Jenkins service.
- Configuration: Configure Jenkins by installing necessary plugins such as Git, Docker, and pipeline plugins.

Connecting to GitHub and selecting project

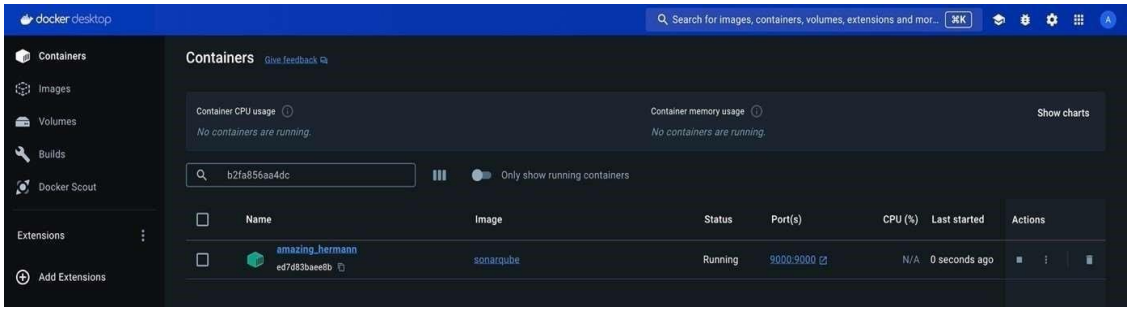
2. Connecting to Github and Selecting Project

- Version Control: Push the application code to a GitHub repository. This ensures the project is under version control, facilitating collaboration and maintaining a history of changes.
- Continuous Deployment Source: GitHub serves as the source repository for continuous deployment, enabling automated updates.



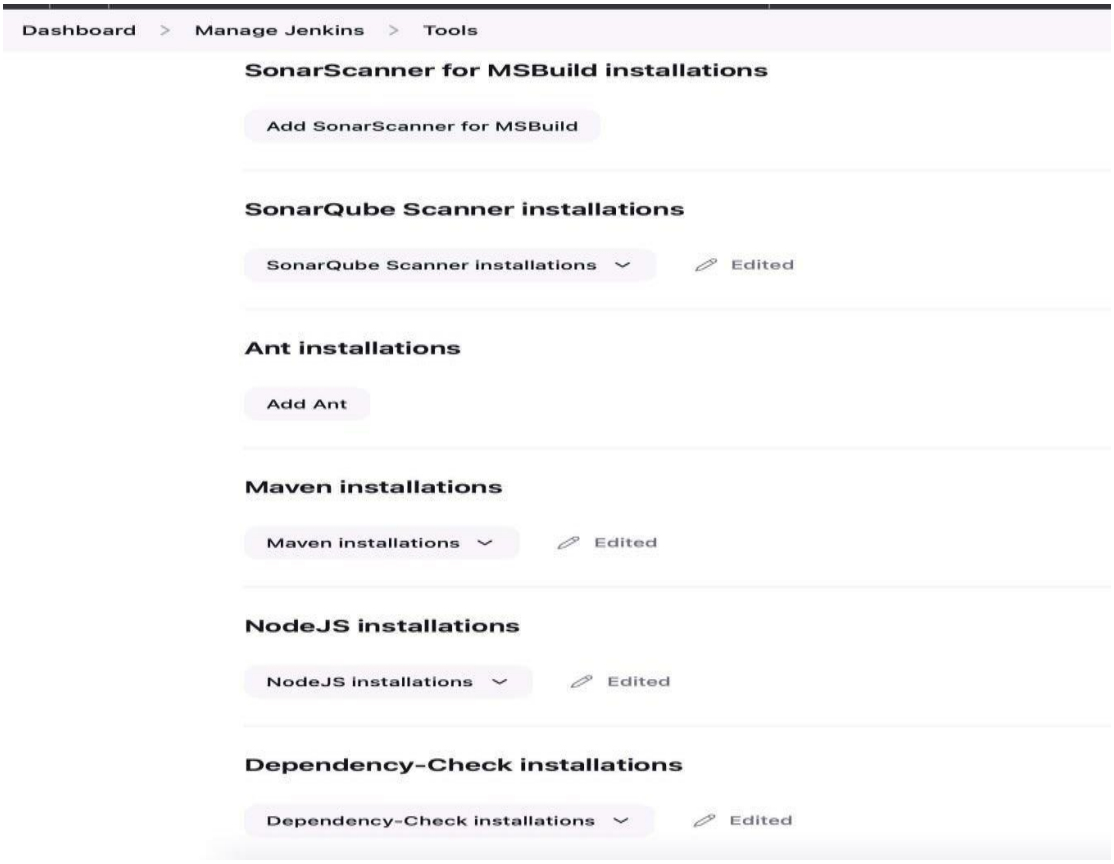
3. Configuring a Docker File

- Dockerfile Creation: Create a Dockerfile to containerize the application. The Dockerfile includes all necessary dependencies, environment configurations, and instructions to build and run the application.
- Environment Consistency: Containerization ensures the application runs consistently across different environments by isolating it from underlying system variations.



4. Setting Up Jenkins Pipeline

- Pipeline Configuration: Create a Jenkins pipeline script (Jenkinsfile) that defines the stages and steps for the CI/CD process.
- Connecting Jenkins to GitHub: Configure Jenkins to connect to the GitHub repository, allowing it to monitor for changes and trigger builds.
- Configuring WebHooks: Set up GitHub webhooks to notify Jenkins of new commits and pull requests.



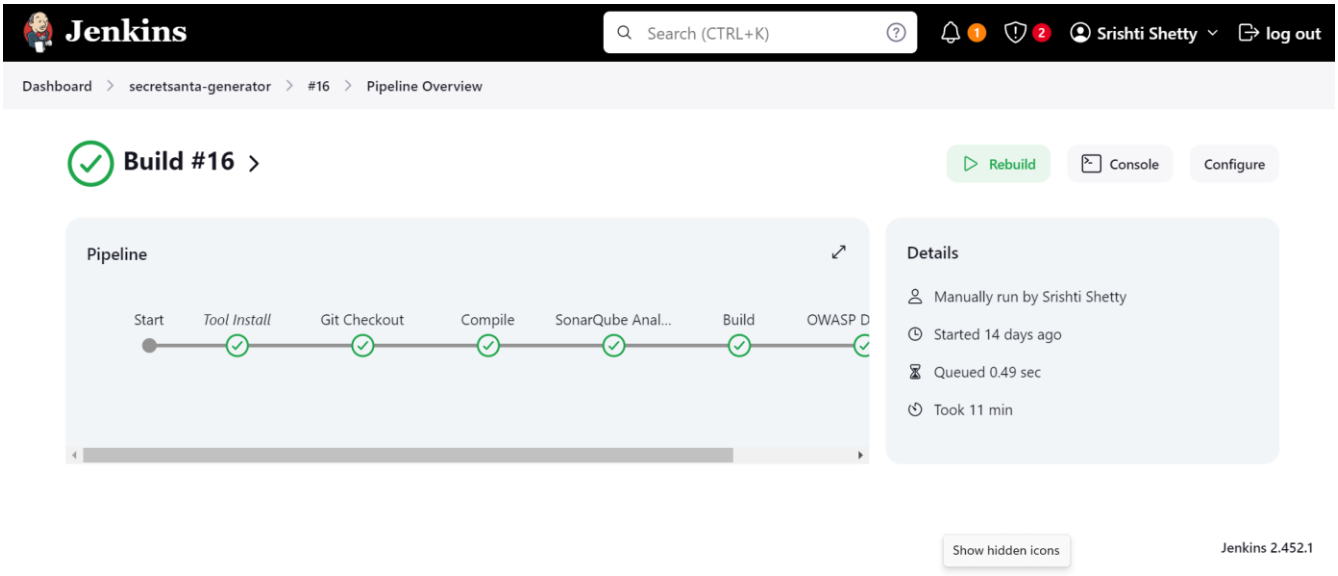
5. Building and Testing the Application

- Build Stage: Define a build stage in the Jenkinsfile to build the Docker image using the DockerFile.
- Test Stage: Include a test stage to run unit tests, integration tests, and other automated test to ensure the application function as expected.

```
+ npm start

> cloud-scribe@0.1.0 start
> react-scripts start

Browserslist: caniuse-lite is outdated. Please run:
  npx update-browserslist-db@latest
  Why you should do it regularly: https://github.com/browserslist/update-db#readme
(node:73845) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the
'setupMiddlewares' option.
(Use `node --trace-deprecation ...` to show where the warning was created)
(node:73845) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the
'setupMiddlewares' option.
Starting the development server...

Compiled successfully!

You can now view cloud-scribe in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.113.201:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
Aborted by Arnav Singh
Sending interrupt signal to process
/Users/arvindkumarsingh/.jenkins/workspace/clouscribe@tmp/durable-8861edf5/script.sh.copy: line 1: 73831 Terminated: 15          npm start
script returned exit code 143
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
```

6. Deploying the Application

- Deployment Configuration : Define a deployment stage in the JenkinsFile to deploy the Docker image to the desired environment(eg : Kubernets, AWS ECS, or another container orchestration platform).
- Configuring Deployment Parameters : Set deployment parameters including environment variables and resource configurations, to ensure proper deployment.

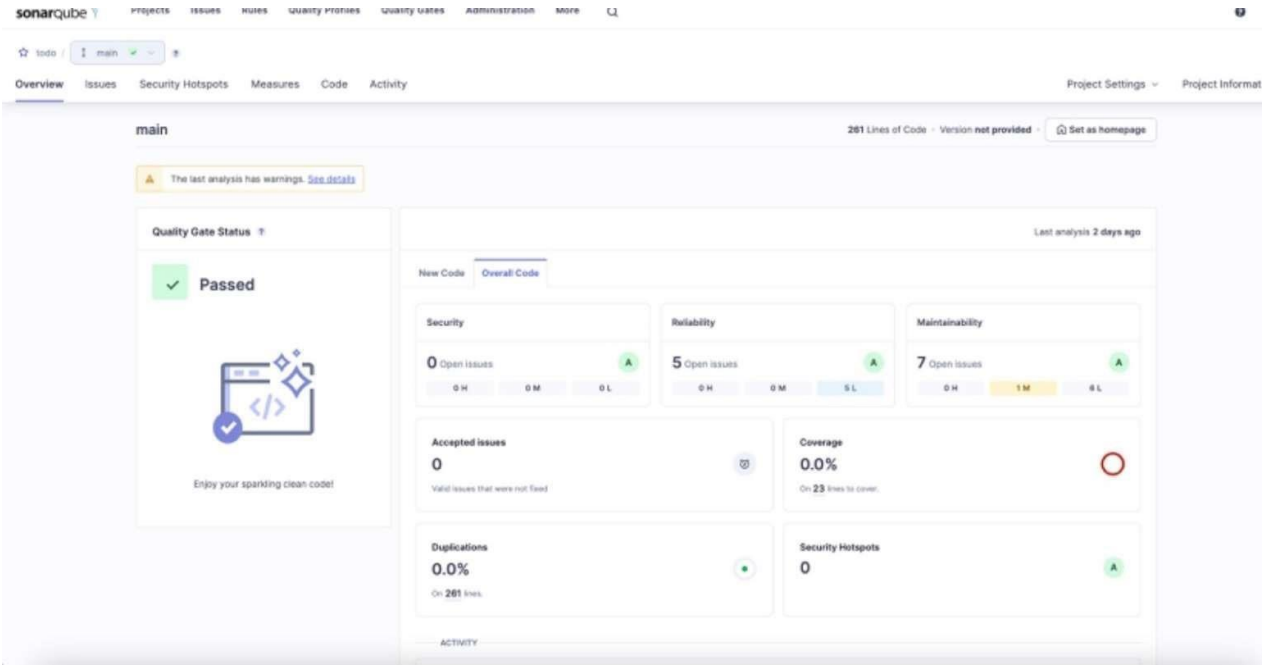


7. Instance Management and Scaling

- Dynamic Scaling: Configure the deployment platform to manage instances dynamically, scaling the number of instances based on traffic demand.
- Efficiency: Ensure the application can handle varying loads efficiently while minimizing costs.

8. Monitoring and Logging

- Enable Monitoring: Enable monitoring and logging through GCP's Stackdriver (now part of Google Cloud Operations Suite) to keep track of the application's performance, logs, and any errors.
- Maintenance and Troubleshooting: This helps in maintaining the application and troubleshooting issues promptly.



9. Testing and Validation

- Thorough Testing: Perform thorough testing to ensure the deployment is successful and the application functions as expected in the cloud environment.
- Continuous Integration Tests: Run continuous integration tests automatically as part of the CI/CD pipeline to validate new changes and ensure they do not break existing functionality.

By following these steps, the CI/CD pipeline ensures an efficient and reliable deployment process for Keeper-App, maintaining high availability and performance while facilitating continuous improvements and updates.

3.2 LIMITATIONS

While the proposed model offers a robust deployment solution, it has certain limitations:

- Security Risks: Jenkins's wide array of plugins and configurations can introduce security vulnerabilities if not properly managed and updated.

- Maintenance Overhead: Regular maintenance of Jenkins instances, including updates and monitoring, can add to the operational overhead.

- Cost Considerations: Running instances on GCP can incur costs, which need to be managed and optimized.

Chapter 4

Results And Discussion

Deploying the Secret Santa-App project via the Jenkins CI/CD pipeline has produced significant results and insights, as discussed below:

1. **Streamlined Deployment Process:** Implementing the Jenkins CI/CD pipeline has vastly simplified the Secret Santa-App deployment process. By automating build, test, and deployment stages, developers confidently push code changes to the repository, knowing they'll smoothly integrate and deploy to production.

2. **Enhanced Collaboration and Efficiency:** Using Jenkins for continuous integration and deployment has boosted collaboration among team members. Developers can concurrently work on features and fixes, with changes automatically deployed upon merging into the main branch, resulting in increased efficiency and faster feature rollouts.

3. **Improved Scalability and Availability:** Secret Santa-App's dynamic instance management, coupled with Jenkins integration, guarantees application scalability and availability. Automatic scaling, based on traffic demand, ensures efficient load handling while minimizing costs, delivering a robust application capable of serving numerous users seamlessly.

4. **Effective Monitoring and Troubleshooting:** Integrating monitoring and logging tools with Jenkins enables real-time monitoring of project performance and health. Tracking metrics like latency and error rates promptly identifies and addresses issues, ensuring proactive troubleshooting and user satisfaction.

5. **Continuous Improvement and Iteration:** The Jenkins CI/CD pipeline fosters a culture of continuous improvement and iteration within the development team. Quick deployment of updates and bug fixes facilitates iterative feature development based on user feedback and analytics, ensuring the Secret SantaApp remains responsive to evolving user needs.

Chapter 5

CONCLUSION

The deployment of the Secret Santa-App using a Jenkins CI/CD pipeline represents a major step forward in improving the efficiency and reliability of our application deployment processes. This project has proven the value of modern DevOps practices and cloud technologies in achieving effective, scalable, and dependable application deployment.

By utilizing Jenkins for continuous integration and deployment, we have created a robust pipeline that automates the build, test, and deployment stages. This automation speeds up the delivery of new features and updates while ensuring consistency and reliability across different environments.

Implementing the Jenkins CI/CD pipeline has brought several key benefits: streamlined deployment processes, improved collaboration among development teams, enhanced scalability and availability, effective monitoring and troubleshooting capabilities, and a culture of continuous improvement and iteration.

Looking ahead, we plan to enhance the Secret Santa-App further, guided by user feedback, analytics insights, and emerging technologies. We will continue to refine features, optimize performance, and prioritize security to ensure the app remains a valuable tool for its users.

In conclusion, the successful deployment of the Secret Santa-App through a Jenkins CI/CD pipeline highlights our commitment to leveraging technology to improve our application delivery processes. We are excited about the potential of this project to drive innovation and efficiency in our operations and remain dedicated to its ongoing evolution and improvement.