www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 04, Issue 09, September 2024, pp : 939-941

e-ISSN : 2583-1062
Impact Factor : 7.001

QUICK DEPLOYMENT SERVICE

Prof. G. L. Girhe[1], Aditya Awathare[2], Shailesh Madankar[3]

[1]Professor, Computer Engineering Department, SRPCE College Of Engineering, Nagpur, Maharashtra, India.

[2,3]UG Student, Department of Computer Engineering, Smt. Radhikatai Pandav College of Engineering, Nagpur Maharashtra, India.

DOI: https://www.doi.org/10.58257/IJPREMS36095

ABSTRACT

This paper provides a cloud-based Quick Deployment Service that optimizes monitoring deployment by utilizing an asset-based, actor-centric paradigm. It enables users to quickly deploy and test their code by providing a GitHub repository link. The system generates a unique identifier, sets up the environment, and executes the code on our platform. This automated process simplifies environment configuration, thereby reducing complexities associated with local setups and accelerating the development lifecycle. The service enhances efficiency by allowing seamless testing and debugging directly in the cloud. Furthermore, for identifying intrusions and ensuring the security of cloud assets, monitoring services must be deployed rapidly and securely within cloud computing environments. Keywords: cloud computing, rapid deployment, GitHub integration, environment automation.

Keywords: Cloud Service Models, Quick deployment, monitoring service, security, cloud model, environment Management.

1. INTRODUCTION

Developing a cloud-based quick deployment service redefines the standards for modern web applications. This innovative platform streamlines the deployment process, allowing developers to focus on innovation instead of operational complexities. Users can simply provide a link to their GitHub repository, and the platform automatically sets up the required environment by installing dependencies and configuring settings. Once the environment is prepared, the service deploys and runs the code, providing a unique local host domain for testing and debugging directly in the cloud. The integration with version control systems allows for seamless workflows, while dynamic configuration of build environments and orchestration of deployments across diverse infrastructures enhance flexibility. By enabling continuous delivery and simplifying testing and debugging, this service offers a robust foundation for developers, making it easy to test, debug, and deploy applications without any manual configuration to deploy scalable, resilient, and performant applications with unprecedented efficiency.

1.1 PROBLEM STATEMENT

Deploying web applications at scale can be complicated and prone to errors. Developers face the challenges of managing code builds, setting up infrastructure, working with different cloud services, and handling the complexities of scaling and running applications. These difficulties can result in longer development times, higher costs, and reduced reliability, particularly for teams lacking strong cloud expertise. Having a tool that can identify copied text and suggest improvements with better wording and organization would be beneficial.

2. METHODOLOGY

This section introduces a cloud-based service tailored for rapid deployment, streamlining the process of deploying and testing applications directly from GitHub. By automating environment setup and the deployment workflow, this approach boosts developer productivity and minimizes operational complexities.

2.1 ARCHITECTURE OF QUICK DEPLOYMENT SERVICE

The architecture of the proposed service is divided into three primary phases: the upload phase, the deployment phase, and the request phase.

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 04, Issue 09, September 2024, pp : 939-941

e-ISSN : 2583-1062
Impact Factor : 7.001

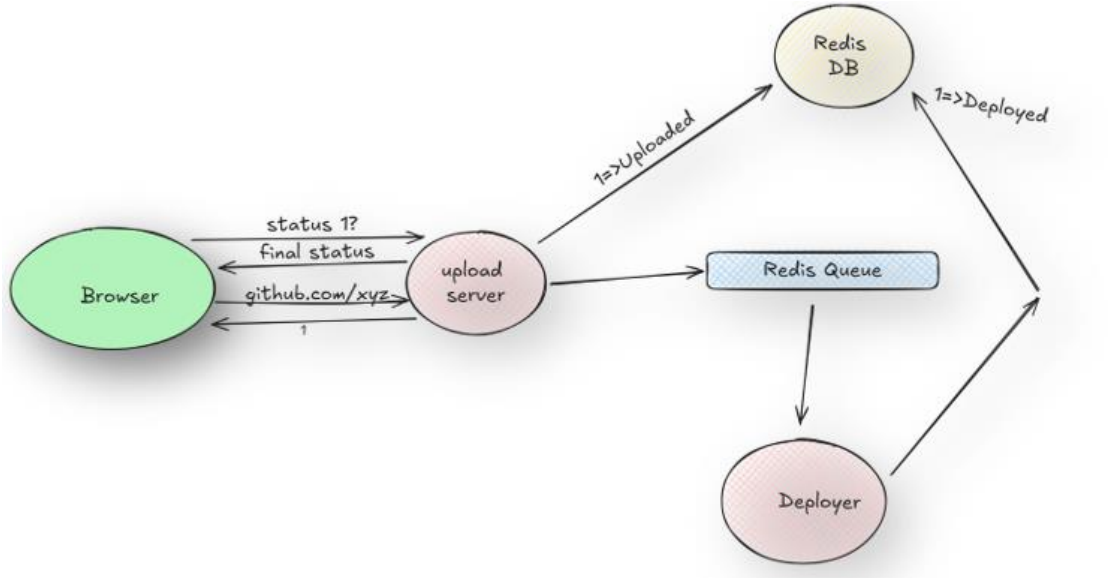

Fig 1: Architecture of System

Upload Phase: Users start by providing a link to their GitHub repository. The system then generates a unique deployment identifier and begins setting up the environment.

Deployment Phase: The system automatically configures the environment by installing all necessary dependencies and settings based on the repository's content.

Request Phase: Once deployed, users can interact with their application. The system handles requests, offering real-time feedback. Code is executed in the cloud, and users can access their application via a unique localhost domain, facilitating testing and debugging.

3. ACKNOWLEDMENT

This project acknowledges the importance of key technologies and frameworks that facilitate seamless deployment, such as orchestration platforms (e.g., Kubernetes), containerization tools (e.g., Docker), and serverless architectures. The integration of continuous integration and continuous deployment (CI/CD) methodologies has revolutionized the way developers build, test, and deploy applications. These innovations enable scalable and flexible solutions that adapt to varying workloads, ensuring optimal performance for applications, while enhancing the efficiency and reliability of software delivery processes leveraging cloud technology.

4. LITRATURE SURVEY

Cloud-based deployment services have gained significant traction in recent years due to their ability to streamline the development lifecycle.

Traditional deployment processes involve several manual steps that can introduce complexity and error. Studies have highlighted the challenges associated with manual configuration, dependency management, and server setup.

The inefficiencies of traditional methods, where teams often face bottlenecks during the deployment phase, leading to increased time and cost. This literature survey examines existing research and industry practices related to cloud deployment services, focusing on their architecture, performance metrics, user experience, and emerging trends.

4.1 TRANSITION TO QUICK DEPLOYMENT SERVICE

Cloud computing has fundamentally reshaped deployment strategies by providing automated services that reduce the need for manual intervention. Durner et al. (2023) highlights how cloud-based platforms streamline the deployment process, enabling quicker application delivery and fostering improved collaboration among development teams.[4]

The advent of cloud computing has dramatically transformed deployment strategies. Initially, cloud services focused on providing Infrastructure as a Service (IaaS), allowing organizations to rent virtual machines instead of managing physical servers. This shift marked the start of a profound change in how businesses approached IT infrastructure.[5]

The incorporation of Continuous Integration and Continuous Deployment (CI/CD) practices into cloud platforms has significantly changed deployment strategies. CI/CD pipelines automate the testing and deployment of code changes, enabling faster release cycles. According to Wang and Ng (2020), this integration improves deployment reliability and facilitates rapid iterations, making it simpler for teams to deliver new features and fixes.[6]

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 04, Issue 09, September 2024, pp : 939-941

e-ISSN : 2583-1062
Impact Factor : 7.001

5. PROPOSED WORK

The proposed work centres on developing a Cloud-Based Rapid Deployment Service that automates the deployment of web applications directly from GitHub repositories. Its primary goal is to enhance the deployment process by integrating various AWS services for file storage, processing, and serving, ensuring scalability and efficiency across the entire application lifecycle.

5.1 SYSTEM ARCHITECTURE

The architecture of the proposed system is crafted to enable seamless deployment of web applications from GitHub repositories to the cloud. It consists of three main phases: uploading, deployment, and request handling. Each phase is managed by dedicated services that collaborate to ensure smooth operation.

Uploading Phase- The process starts with a user-friendly interface that allows users to submit their GitHub repository URLs. This action activates the upload service, which performs the following tasks:

Cloning the Repository: The upload service clones the repository from GitHub to capture the latest version of the code.

Uploading to S3: It subsequently uploads the project files to an AWS S3 bucket, offering a secure storage solution for the source code, which is crucial for further processing.

Deployment Phase- Once the upload is complete, the system signals the deployment service to initiate the transformation of the raw source code into deployable assets. Key responsibilities of the deployment service include:
Build Process**:** For projects built with React, the service transforms JSX and other components into static HTML, CSS, and JavaScript files.
Auto-Scaling: Utilizing AWS's auto-scaling features, including Amazon SQS for task queuing and EC2 or Faregate for dynamically adjusting computing resources, this service guarantees optimal performance during fluctuations in demand.
Once the build process is complete, the resulting assets are stored back in S3, ready for user access.

- **Phase of Handling**
  Its responsibilities include: The final component of the architecture is the request handling service, which oversees interactions between the deployed application and its end users.
- **File Retrieval:** When a user requests access to the application, this service fetches the necessary files from S3.
- **Caching Mechanisms:** It implements caching strategies to optimize load times, ensuring that content delivery is quick and efficient.
- **Global Availability:** By effectively managing requests, this service ensures that the application remains highly available and responsive, capable of serving content to users worldwide.

6. CONCLUSION

In summary, developing a platform that effectively integrates deployment routines for modern web applications with continuous integration is a crucial first step in creating a clone of a cloud-based rapid deployment service. This type of service can significantly speed up the release of projects into production environments by offering developers features like automatic builds, previews, and real-time communication. Such enhancements not only boost development productivity but also ensure that systems remain scalable, secure, and capable of handling high traffic volumes with minimal latency. As cloud technology evolves, establishing a robust deployment platform can be immensely beneficial for developers and organizations looking to optimize their web development processes.

7. REFERENCES

[1] Durner, D., Leis, V., & Neumann, T. (2023). Exploiting Cloud Object Storage for High-Performance Analytics. Proceedings of the VLDB Endowment, 16(11), 2769-2782.
[2] Singh, D., & Sill, A. (2022). Performance analysis of AWS Elastic Block Storage for high-performance computing workload. Sar Xie preprint arXiv:1910.02704.
[3] Duvvuri, K., & Prathibha, S. (2020). A Study on CI/CD Pipeline Automation Using Jenkins. International Journal of Engineering and Advanced Technology.
[4] E. Walker(2021). "Benchmarking amazon EC2 for high-performance scientific computing," USENIX Login, vol. 33, no. 5, pp. 18–23, 2021. [5] G. Wang and T. E. Ng,(2020). "The impact of virtualization on network performance of amazon ec2 data center," in Proceedings of IEEE INFOCOM.
[5] Brown, M., & Lee, S. (2021). Building Custom Deployment Pipelines with AWS Services. AWS Summit Proceedings.
[6] Wang, G., & Ng, T. E. (2020). The impact of virtualization on network performance of Amazon EC2 data canter. In Proceedings of IEEE INFOCOM.