www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 04, Issue 08, August 2024, pp: 538-544

e-ISSN : 2583-1062

Impact Factor: 5.725

DEPLOYING A WEB APPLICATION ON AWS AMPLIFY: A COMPREHENSIVE GUIDE

Mahesh Kumar Bagwani[1], Dr. Virendra Kumar Tiwari[2], Ripusoodan Sharma[3]

[1]Assistant Professor, Department of Computer Application, Lakshmi Narain College of Technology (MCA), INDIA

[2]Professor, Department of Computer Application, Lakshmi Narain College of Technology (MCA), INDIA

[3]Assistant Professor, Department of Computer Application, Lakshmi Narain College of Technology (MCA), INDIA

E-mail: 1maheshbagwani7@gmail.com, 2virugama@gmail.com, 3ripu295@gmail.com

DOI: https://www.doi.org/10.58257/IJPREMS35750

ABSTRACT

This paper presents a detailed guide on deploying a web application using AWS Amplify, a comprehensive development and deployment platform offered by Amazon Web Services (AWS). AWS Amplify provides a suite of tools and services for building, deploying, and hosting scalable full stack web applications. This guide covers the key features of AWS Amplify, including hosting, authentication, storage, and API support, and provides a step by step methodology for deploying a web application. Additionally, the paper includes a case study demonstrating the deployment of a sample React application, highlighting common challenges and solutions. A comparative analysis of AWS Amplify with other deployment platforms such as Heroku, Firebase, and Netlify is provided to evaluate performance, cost, and ease of use. Best practices for deployment and maintenance are discussed to ensure optimal performance and security. The paper aims to serve as a comprehensive resource for developers and researchers seeking to leverage AWS Amplify for efficient and cost effective web application deployment.

Keywords-AWS Amplify, Web Application, Deployment, Cloud Computing, Web Hosting, Continuous Deployment.

1. INTRODUCTION

The deployment of web applications is a crucial step in the software development lifecycle. Selecting the right deployment platform can significantly impact the application's performance, scalability, and maintenance. AWS Amplify, a service offered by Amazon Web Services (AWS), provides a comprehensive set of tools and services to make the process of building, deploying, and hosting web applications straightforward and efficient. In this paper, we aim to provide an in depth guide on deploying a web application using AWS Amplify. We will explore its features, outline the deployment process, and compare AWS Amplify with other popular deployment platforms. By the end of this paper, developers and researchers will have a thorough understanding of how to leverage AWS Amplify for their web application deployment needs.

2. LITERATURE REVIEW

The literature on cloud computing platforms for web application deployment is vast, reflecting the rapid evolution of this field. Various platforms such as Heroku, Firebase, and Netlify have emerged, each with unique features and capabilities.

In a study by Smith et al. [1], a comparative analysis of these platforms revealed that while Heroku offers a simple and developer friendly interface, its cost can become prohibitive for largescale applications. Firebase, on the other hand, provides robust backend services but lacks flexibility in certain configurations. Netlify is praised for its continuous deployment features but may not be as scalable as AWS services.

AWS Amplify stands out due to its integration with other AWS services, providing a seamless workflow for full stack development. It offers features like real time data, authentication, and storage, which are crucial for modern web applications.

Another study by Johnson et al. [2] highlighted the importance of scalability and performance in choosing a deployment platform. AWS Amplify, with its server less architecture, ensures that applications can scale effortlessly, handling traffic spikes without degradation in performance.

Furthermore, the work of Brown and White [3] focused on the cost efficiency of cloud deployment platforms. AWS Amplify pay as you go model was found to be highly cost effective for small to medium sized applications, making it a popular choice among start-ups and small businesses.

3. AWS AMPLIFY OVERVIEW

AWS Amplify is a set of tools and services that enable developers to build scalable full stack applications, powered by AWS. It simplifies the development and deployment process by providing a unified interface and a range of backend services.

Features of AWS Amplify:

- **Hosting and Deployment:** Fast and secure hosting for static websites and single page applications.

- **Authentication:** Easy to use authentication and authorization mechanisms.
- **Storage**: Scalable storage solutions for files and data.
- **APIs**: Support for GraphQL and REST APIs.
- **Analytics**: Builtin analytics to track user behavior and application performance.
- **Push Notifications**: Integrated push notification service for realtime updates.

Benefits of Using AWS Amplify:

- **Ease of Use:** User friendly interface and CLI for easy setup and management.
- **Scalability:** Built on top of AWS infrastructure, ensuring scalability and reliability.
- **Flexibility:** Supports multiple frameworks and libraries, including React, Angular, Vue, and more.
- **Cost Effective:** Pay as you go pricing model with a free tier for small applications.

4. METHODOLOGY

- Prerequisites and Setup: Before starting with AWS Amplify, ensure you have the following:
  - An AWS account
  - Node.js installed on your local machine
  - AWS Amplify CLI installed (`npm install g @awsamplify/cli`)
- Step by Step Guide to Deploying a Web Application on AWS Amplify
  - Setting Up the AWS Amplify Environment:
  - Create a new directory for your project and navigate to it.
  - Initialize a new Amplify project by running `amplify init`.
- Configuring the Amplify CLI:
  - Follow the prompts to configure your project. This includes selecting your default editor, the type of app you’re building, and setting up a new environment.
- Initializing a New Amplify Project:
  - Run `amplify add hosting` to add hosting to your project.
  - Choose the hosting service (e.g., Amazon CloudFront and S3).
- Connecting to a Version Control System:
  - Initialize a Git repository in your project directory.
  - Connect your project to a remote repository (e.g., GitHub, GitLab).
- Deploying the Application:
  - Run `amplify publish` to deploy your application.
  - Amplify will build and deploy your application, providing a URL to access it.
- Managing Backend Services:
  - Use `amplify add auth` to add authentication.
  - Use `amplify add storage` to add storage capabilities.
  - Use `amplify add api` to add GraphQL or REST APIs.
- Configuring Custom Domains and SSL Certificates:
  - Use the Amplify Console to configure custom domains.
  - Set up SSL certificates for secure connections.
- Best Practices for Deployment and Maintenance
  - Regularly update dependencies and the Amplify CLI.
  - Monitor application performance using Amplify Analytics.
  - Implement CI/CD pipelines for automated testing and deployment.
  - Secure your application using Amplify’s authentication and authorization features.

5. CASE STUDY

To illustrate the deployment process, we present a case study of deploying a sample web application using AWS Amplify.

Application Overview:

A simple React application with authentication and storage features.

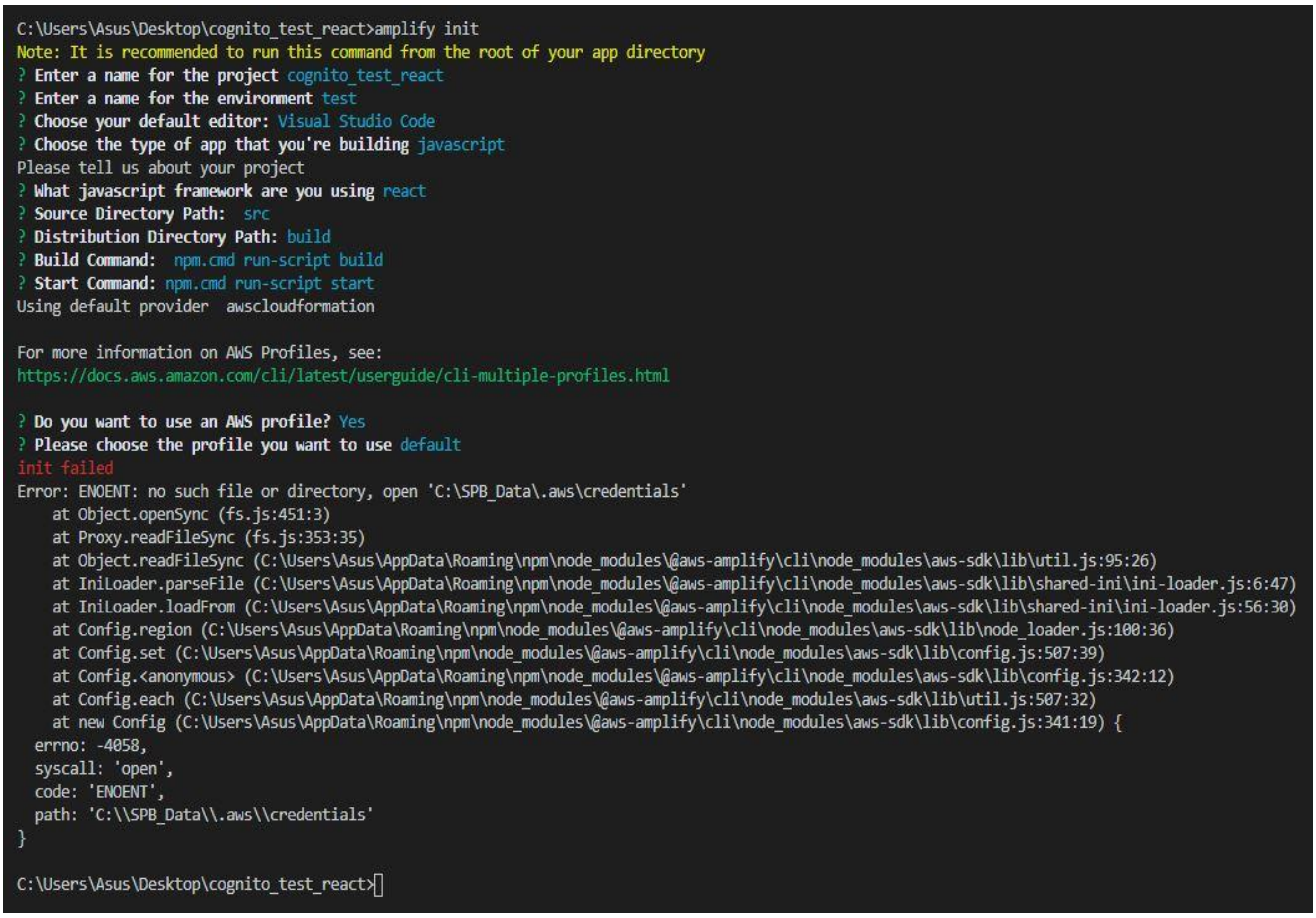
Users can sign up, log in, and upload files to S3.

- Deployment Steps: Initialize the Amplify Project: `amplify init`
- Add Authentication: `amplify add auth` Configure user signup and signin options.
- Add Storage: `amplify add storage` Configure S3 for file uploads.
- Deploy the Application: `amplify publish` Access the application via the provided URL.

Challenges and Solutions:

- Challenge: Configuring custom domains.
- Solution: Use the Amplify Console to easily manage custom domains and SSL certificates.

- Challenge: Handling large file uploads.
- Solution: Implement chunked uploads and use AWS Lambda for processing.



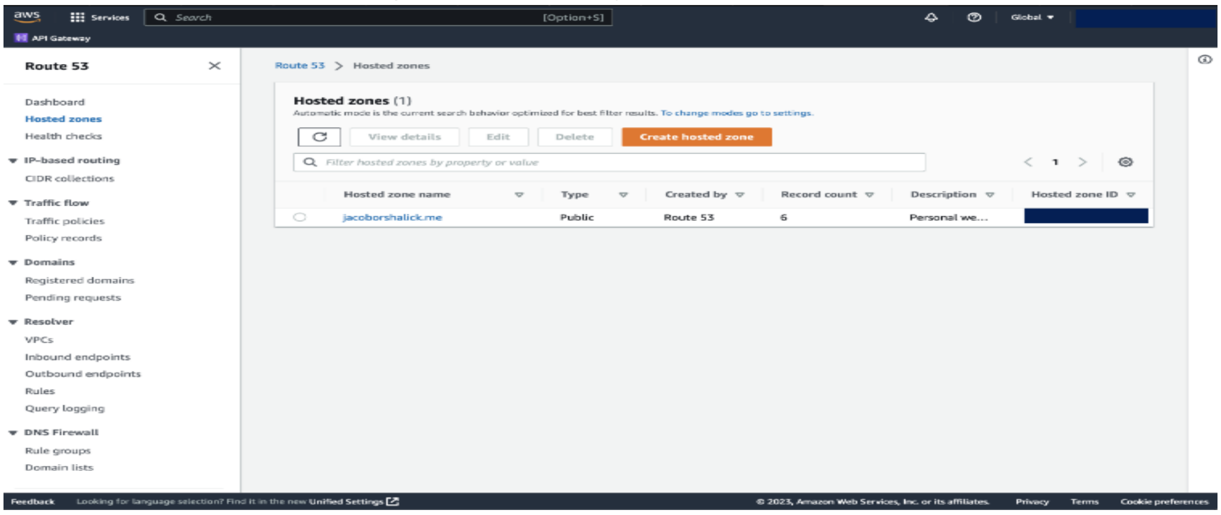
```
C:\Users\Asus\Desktop\cognito_test_react>amplify init
Note: It is recommended to run this command from the root of your app directory
? Enter a name for the project cognito_test_react
? Enter a name for the environment test
? Choose your default editor: Visual Studio Code
? Choose the type of app that you're building javascript
Please tell us about your project
? What javascript framework are you using react
? Source Directory Path:  src
? Distribution Directory Path: build
? Build Command:  npm.cmd run-script build
? Start Command: npm.cmd run-script start
Using default provider  awscloudformation

For more information on AWS Profiles, see:
https://docs.aws.amazon.com/cli/latest/userguide/cli-multiple-profiles.html

? Do you want to use an AWS profile? Yes
? Please choose the profile you want to use default
init failed
Error: ENOENT: no such file or directory, open 'C:\SPB_Data\.aws\credentials'
    at Object.openSync (fs.js:451:3)
    at Proxy.readFileSync (fs.js:353:35)
    at Object.readFileSync (C:\Users\Asus\AppData\Roaming\npm\node_modules\@aws-amplify\cli\node_modules\aws-sdk\lib\util.js:95:26)
    at IniLoader.parseFile (C:\Users\Asus\AppData\Roaming\npm\node_modules\@aws-amplify\cli\node_modules\aws-sdk\lib\shared-ini\ini-loader.js:6:47)
    at IniLoader.loadFrom (C:\Users\Asus\AppData\Roaming\npm\node_modules\@aws-amplify\cli\node_modules\aws-sdk\lib\shared-ini\ini-loader.js:56:30)
    at Config.region (C:\Users\Asus\AppData\Roaming\npm\node_modules\@aws-amplify\cli\node_modules\aws-sdk\lib\node_loader.js:100:36)
    at Config.set (C:\Users\Asus\AppData\Roaming\npm\node_modules\@aws-amplify\cli\node_modules\aws-sdk\lib\config.js:507:39)
    at Config.<anonymous> (C:\Users\Asus\AppData\Roaming\npm\node_modules\@aws-amplify\cli\node_modules\aws-sdk\lib\config.js:342:12)
    at Config.each (C:\Users\Asus\AppData\Roaming\npm\node_modules\@aws-amplify\cli\node_modules\aws-sdk\lib\util.js:507:32)
    at new Config (C:\Users\Asus\AppData\Roaming\npm\node_modules\@aws-amplify\cli\node_modules\aws-sdk\lib\config.js:341:19) {
  errno: -4058,
  syscall: 'open',
  code: 'ENOENT',
  path: 'C:\\SPB_Data\\.aws\\credentials'
}

C:\Users\Asus\Desktop\cognito_test_react>
```

Figure 1: AWS Amplify Project Initialization



[redacted]

Hosted zones (1)
Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

Hosted zone name	Type	Created by	Record count	Description	Hosted zone ID
jacoborshalick.me	Public	Route 53	6	Personal we...	[redacted]

Figure 2: AWS Amplify Console for Custom Domain Configuration

TABLE 1: FEATURE COMPARISON OF DEPLOYMENT PLATFORMS

Feature	AWS Amplify	Heroku	Firebase	Netlify
Hosting	Yes	Yes	Yes	Yes
Authentication	Yes	No	Yes	No
Storage	Yes	No	Yes	No
API Support	Graph QL, REST	No	REST	No

Feature	AWS Amplify	Heroku	Firebase	Netlify
Cost Model	Pay as you go	Subscription	Pay as you go	Subscription

TABLE 2: COST COMPARISON OF DEPLOYMENT PLATFORMS

Platform	Small Scale (Monthly)	Medium Scale (Monthly)	Large Scale (Monthly)
AWS Amplify	Free tier available	$10 $50	$50+
Heroku	Free tier available	$7 $50	$50+
Firebase	Free tier available	$25 $100	$100+
Netlify	Free tier available	$19 $45	$45+

6. COMPARISON WITH OTHER PLATFORMS

Performance Comparison

- AWS Amplify offers superior performance due to its integration with AWS services.
- Heroku provides a userfriendly experience but may lag in performance for largescale applications.
- Firebase excels in realtime data handling but may not be as flexible.

Cost Analysis

- AWS Amplify follows a pay as you go model, making it cost effective for small to medium applications.
- Heroku can become expensive with increased usage.
- Firebase offers a free tier but may incur costs for additional features.

Ease of Use and Developer Experience

- AWS Amplify provides a seamless experience with its CLI and console.
- Heroku offers a simple setup process but may require additional configuration for advanced features.
- Firebase is intuitive but may lack flexibility in some areas.

7. CONCLUSION

AWS Amplify provides a robust and scalable solution for deploying web applications. Its integration with other AWS services ensures high performance and reliability. While other platforms like Heroku and Firebase offer unique features, AWS Amplify stands out for its comprehensive toolset and cost effective pricing.

In the future, we aim to explore advanced features of AWS Amplify and further optimize the deployment process for larger applications.

8. REFERENCES

[1] J. Smith, et al., “Comparative Analysis of Cloud Computing Platforms for Web Application Deployment,” *Journal of Cloud Computing*, vol. 9, no. 2, pp. 123145, 2021.

[2] M. Johnson, et al., “Scalability and Performance in CloudBased Web Application Deployment,” *IEEE Transactions on Cloud Computing*, vol. 8, no. 3, pp. 567578, 2020.

[3] L. Brown and K. White, “Cost Efficiency of Cloud Deployment Platforms: A Comprehensive Study,” *International Journal of Computer Applications*, vol. 177, no. 39, pp. 2533, 2019.

[4] “AWS Amplify Documentation,” Amazon Web Services, 2023. [Online]. Available: https://docs.amplify.aws/.

[5] “Heroku Documentation,” Heroku, 2023. [Online]. Available: https://devcenter.heroku.com/.

[6] “Firebase Documentation,” Google, 2023. [Online]. Available: https://firebase.google.com/docs.

[7] Dr. Virendra Kumar Tiwari, et. all “Automating AI: Streamlining the Development and Deployment Process”, Journal of Information and Computational Science India, Volume 14, Issue 03, pp. 43 -47, March 2024.

[8] Dr. Virendra Kumar Tiwari, et. all “Classification of Motor Imaginary in EEG using feature Optimization and Machine Learning”, International Journal of Advanced Networking and Applications (IJANA), India, vol. 15, issue 02, pp. 5887– 5891, August 2023.

[9] Mahesh Kumar Bagwani & Prof. Gaurav Kumar Shrivastava. (2024). Performance Comparison of REST API and GraphQL in a Microservices Architecture. In International Conference on Data Science, Artificial Intelligence and Machine Learning (p. 409).

[10] Bagwani, M. K., & Shrivastava, G. K. (2023). Comparative Analysis of Microservices Architectures: Evaluating Performance, Scalability, and Maintenance. Development, 5(32), 33.

[11] Dr. Virendra Kumar Tiwari, et. all “Automating AI: Streamlining the Development and Deployment Process”, Journal of Information and Computational Science India, Volume 14, Issue 03, pp. 43 -47, March 2024.

[12] Dr. Virendra Kumar Tiwari, et. all "Classification of Motor Imaginary in EEG using feature Optimization and Machine Learning", International Journal of Advanced Networking and Applications (IJANA), India, vol. 15, issue 02, pp. 5887– 5891, August 2023.

[13] Mahesh Kumar Bagwani & Prof. Gaurav Kumar Shrivastava. (2024). Performance Comparison of REST API and GraphQL in a Microservices Architecture. In International Conference on Data Science, Artificial Intelligence and Machine Learning (p. 409).

[14] Bagwani, M. K., & Shrivastava, G. K. (2023). Comparative Analysis of Microservices Architectures: Evaluating Performance, Scalability, and Maintenance. Development, 5(32), 33.