**Project Proposal: CloudFormation Deployment of WordPress with Amazon RDS on AWS**

Abdulatif Azimov, Mrudul Madaan, Paulette Wolfe, & Seyed Mojtaba Hosseiniraviz

Department of Data Science Monroe University, King Graduate School

25SP-CS670-152HY: Cloud Computing

Professor: Mahmud Islam

June 3, 2025

**1. Project Title**

***Automated WordPress Deployment with RDS Integration Using AWS CloudFormation***

**2. Executive Summary**

This project proposes the deployment of a scalable, secure, and semi-automated infrastructure on AWS to host a WordPress application backed by a managed MySQL database using Amazon RDS. The deployment leverages AWS CloudFormation to define and provision resources as code, ensuring reproducibility, modularity, and adherence to cloud-native best practices.

The solution separates the application and data layers, applies principles of least privilege, supports observability, and includes automation features to handle common challenges like dynamic IPs and initial setup signaling. The goal is to deliver a production-ready environment that is easy to deploy, maintain, and scale.

**3. Purpose and Objectives**

**Purpose**

To provision a complete WordPress hosting environment using AWS infrastructure-as-code that ensures:

* Separation of concerns between application and data layers
* Security through scoped access controls
* High availability for the database tier
* Simplified bootstrapping and monitoring
* Adaptability to different environments (test, dev, staging)

**Objectives**

* Deploy a CloudFormation stack that builds an end-to-end infrastructure for WordPress
* Integrate Amazon RDS for managed MySQL database hosting
* Automate the application setup and configuration using EC2 UserData
* Enable monitoring via CloudWatch
* Ensure infrastructure is modular, parameterized, and reusable

**4. Project Scope**

**What Will Be Deployed**

* **Networking**
  + A Virtual Private Cloud (VPC)
  + Public and private subnets for web and database tiers
  + Internet Gateway and custom route tables for outbound internet access
* **Security**
  + Security Groups with strict inbound/outbound rules
  + Access control that restricts RDS access to the EC2 instance only
* **Database Layer**
  + Amazon RDS MySQL instance deployed in private subnets
  + DBSubnetGroup for high availability
* **Application Layer**
  + Amazon EC2 instance running the WordPress application
  + IAM Role and Instance Profile with CloudWatch and SSM permissions
  + UserData script for automatic WordPress installation and configuration
* **Monitoring**
  + CloudWatch Agent installation and configuration
  + Log and metric forwarding for observability
* **Automation**
  + cfn-signal for instance readiness signaling
  + Shell script for dynamically updating the WordPress site URL based on public IP

**5. Technical Design & Justification**

| **Component** | **Justification** |
| --- | --- |
| **VPC with Subnets** | Isolates resources and segments tiers for security and scalability. |
| **Public Web Tier Subnet** | Enables access to the WordPress EC2 instance over HTTP/SSH. |
| **Private Database Subnets** | Keeps RDS instances hidden from direct internet access. |
| **Security Groups** | Enforce fine-grained network control using group-to-group rules rather than CIDRs. |
| **Amazon RDS** | Provides a managed, scalable MySQL database with automated backups and maintenance. |
| **EC2 with UserData** | Bootstraps WordPress installation to minimize manual configuration. |
| **IAM Role + Instance Profile** | Grants the EC2 instance access to required AWS services without using static credentials. |
| **CloudWatch Agent** | Adds visibility into the system's performance and logs for operational health. |
| **cfn-signal** | Ensures stack integrity by signaling when bootstrapping is complete. |
| **Dynamic IP Update Script** | Solves the issue of dynamic IPs in WordPress URLs without requiring static IPs or Elastic IPs. |

**6. Key Benefits**

* Infrastructure-as-Code: All resources are defined declaratively in CloudFormation for easy version control and automation.
* Security First: Strong isolation and least-privilege access across tiers.
* Observability: Logs and metrics are forwarded to CloudWatch for monitoring.
* Automation: Minimal human interaction needed post-stack launch.
* Modularity: Parameters allow customization for different environments.
* Cloud Best Practices: Follows AWS-recommended architecture patterns.

**7. Deliverables**

* CloudFormation template (YAML or JSON)
* Parameter values file (for quick customization)
* WordPress setup with publicly accessible frontend
* RDS MySQL backend accessible only from EC2
* CloudWatch metrics and logs visible in AWS Console
* Documentation including:
  + Architecture diagram
  + Deployment instructions
  + Troubleshooting guide

**8. Timeline**

| **Week** | **Milestone** |
| --- | --- |
| Week 1 | Finalize requirements, design infrastructure |
| Week 2 | Develop and test CloudFormation template |
| Week 3 | Validate RDS and EC2 bootstrapping integration |
| Week 4 | Enable monitoring and automate post-setup configuration |
| Week 5 | QA, error handling, documentation, and final review |

**9. Risks and Mitigation**

| **Risk** | **Mitigation Strategy** |
| --- | --- |
| EC2 Bootstrapping failure | Use CreationPolicy and cfn-signal to prevent half-baked deployments |
| Public IP changes affect WordPress | Use custom script to update the WordPress database on boot |
| RDS access exposure | Restrict access to specific security groups only |
| Package installation failures | Add retries and logging in UserData script for visibility |
| Performance bottlenecks | Use CloudWatch to identify and optimize resource usage |

**10. Future Enhancements**

* Add an Auto Scaling Group (ASG) for WordPress EC2 instances
* Integrate with Application Load Balancer (ALB)
* Use Elastic File System (EFS) for shared WordPress content
* Automate HTTPS setup with ACM and ALB
* CI/CD integration using CodePipeline or GitHub Actions

**11. Conclusion**

This project will deliver a production-grade, modular WordPress hosting environment on AWS, emphasizing automation, security, and observability. By using AWS CloudFormation, the team ensures that the deployment is consistent, repeatable, and scalable. The solution can serve as a baseline for more complex architectures in future phases, such as multi-tier deployments, autoscaling, and global distribution.

A diagram of a cloud computing system

AI-generated content may be incorrect.