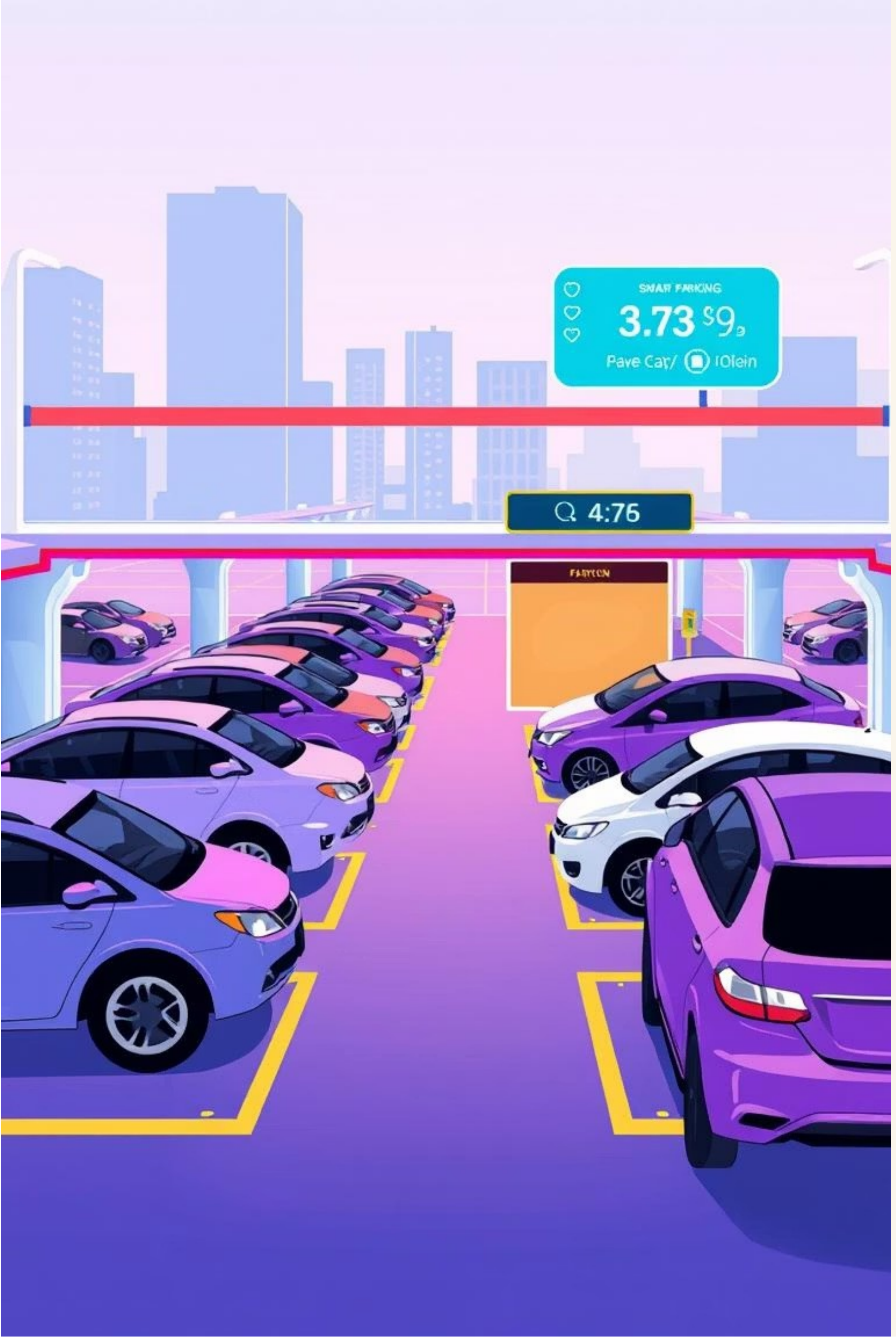


Project Proposal: Scalable Smart Parking & Vehicle Management System

Author: Sameer Shahzad

Hannan Shahzad

Executive Summary: SPVMS Overview

Addressing Inefficiencies

SPVMS is a full-stack application designed to manage parking operations, real-time vehicle allocation, and secure user access, tackling traditional parking

Key AWS Services

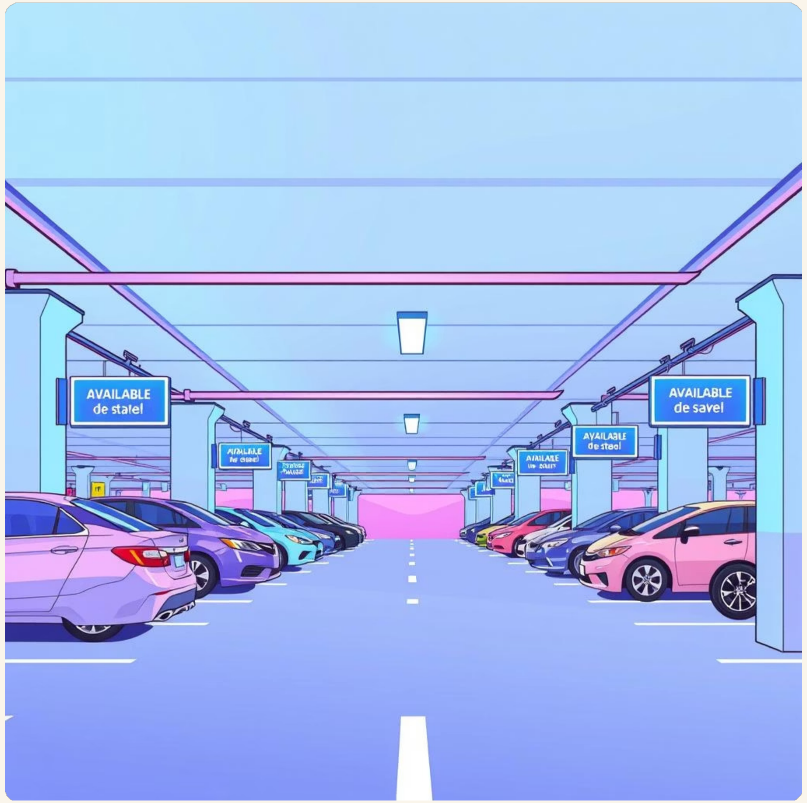
Utilizes ALB, ECS, RDS/ DynamoDB, and ElastiCache (Redis) for traffic distribution, containerized services, data persistence, and real-time synchronization.

AWS Cloud Architecture

Deployed on a highly available and auto-scaling AWS Cloud, ensuring secure, scalable, and resilient operations.

Introduction: Solving Urban Parking Challenges

Urban areas and commercial facilities often struggle with parking. Drivers waste time, and administrators lack real-time visibility. SPVMS automates allocation, tracks entry/exit, and provides real-time availability.



Modern Technology Stack

- **Frontend (React):** User-friendly interface for customers and administrators.
- **Backend (Python / Django):** Scalable services for allocation, reservations, and notifications.
- **Database (PostgreSQL / DynamoDB):** Reliably tracks spots, accounts, and history.
- **Real-Time (Django Channels / Redis):** Instant updates via WebSockets and Redis.
- **Security (Django Rest Framework & JWT / AWS Cognito):**

Role-based access management.

Problem Statement: Inefficiencies in Traditional Parking

Limited Capacity Management

Lack of real-time tracking leads to underutilized spaces and missed revenue.

Inefficient User Experience

Users can't find spaces in advance, causing wasted time, congestion, and frustration.

Scalability Issues

Single-server deployments fail under high demand, leading to outages.

Data Integrity & Security Risks

Manual or poorly secured systems are vulnerable to incorrect allocation and breaches.

A cloud-native, auto-scaling solution ensures robust handling of loads, real-time updates, optimized usage, and strong security.

Aim & Objectives: Secure and Scalable SPVMS

Our aim is to design and deploy a secure, highly available, and scalable Smart Parking & Vehicle Management System using AWS services.

1

High Availability

Multi-AZ RDS/PostgreSQL and multiple ECS instances for continuous operation.

2

Scalability

ECS Auto Scaling Group behind ALB to handle variable demand.

5

Performance

Amazon S3 + CloudFront for fast, secure, and globally distributed frontend access.

4

Security & Compliance

AWS Secrets Manager, IAM roles, and AWS Cognito for data protection and authentication.

Project Scope: Comprehensive AWS Deployment

Frontend Hosting

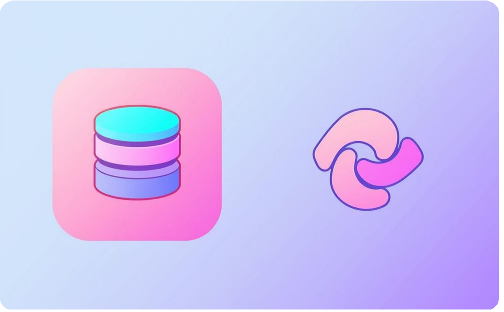
S3 + CloudFront for cost-effective storage and global content delivery.

Backend Deployment

ECS / Lambda + API Gateway for containerized microservices and efficient load handling.

Database Layer

RDS / DynamoDB for user, spot, and reservation data with multi-AZ for high availability.



Real-Time Layer

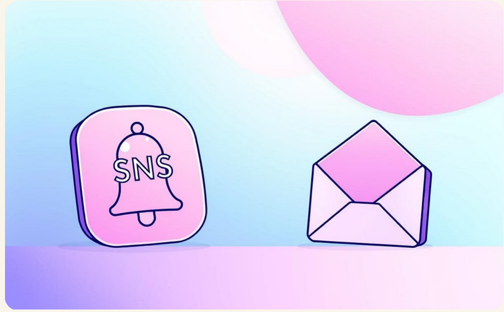
Django Channels + ElastiCache Redis for synchronized updates and double-booking prevention.

Security & IAM

Secrets Manager, IAM roles, and Cognito for secure credentials and access control.

Monitoring & Logging

CloudWatch for system health, performance metrics, and proactive alarms.



Notifications & Alerts

SNS/SES for timely confirmations, expiration alerts, and parking availability.

Testing and Results: Deployment Verification

Local Server Response

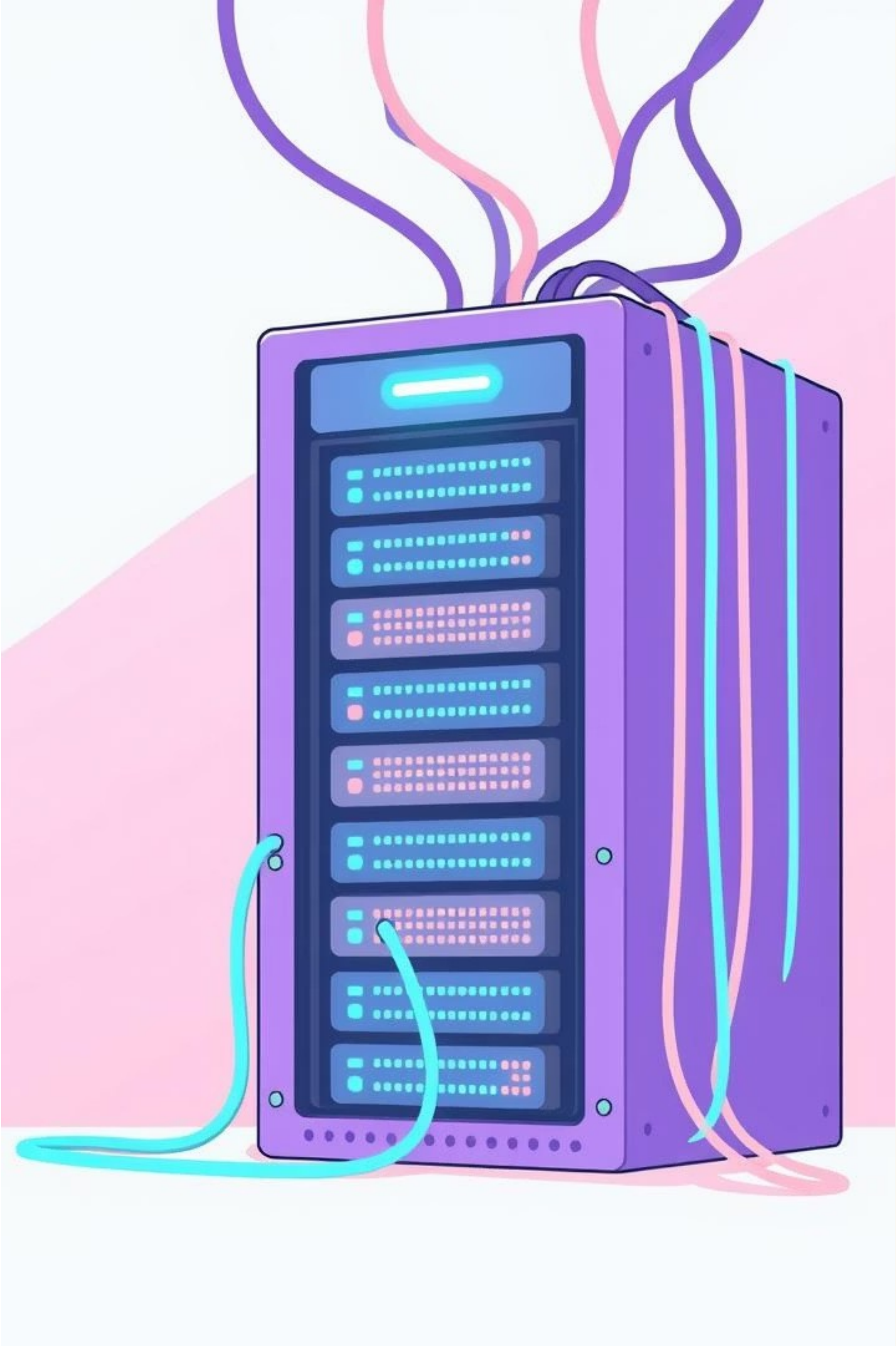
Verified deployment using `curl -I http://127.0.0.1:8000`

Result: `HTTP/1.1 200 OK`

Website Accessibility

Successfully accessible at:

- http ://35.175.117.155
- http://ec2-35-175-117-155.compute-1.amazonaws.com/



Monitoring & Logging: Proactive System Health

Continuous Monitoring

Nginx error logs and Django process logs are continuously monitored.

Rapid Issue Identification

Proactive approach allows quick identification and resolution of operational issues.

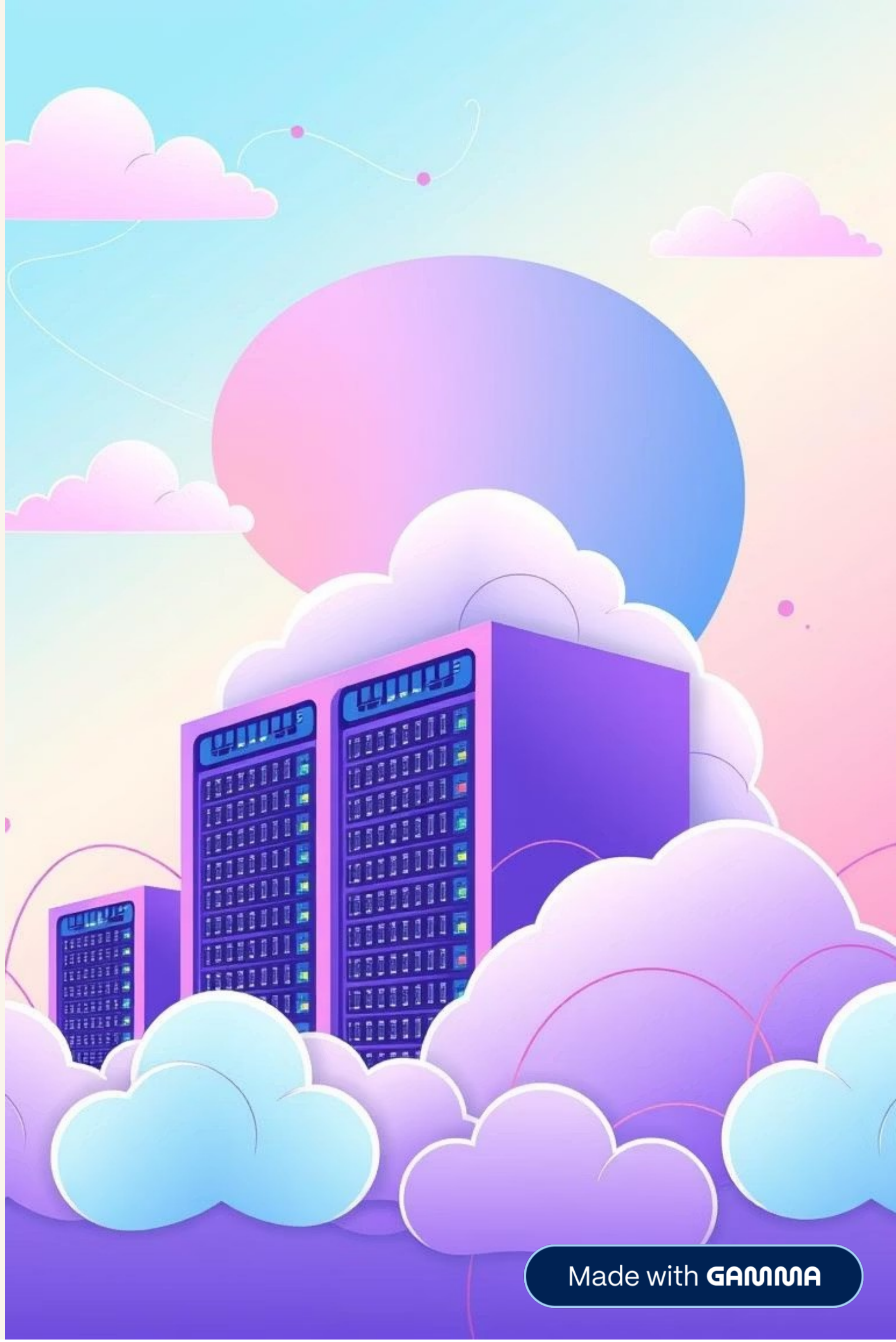
Command to view logs: `sudo tail -f /var/log/nginx/error.log tail -f nohup.out`

Conclusion: Successful AWS Migration

“The Smart Parking Information Management System was successfully migrated to the AWS Cloud.”

“Strategic use of EC2 instances and Nginx provides a professional-grade hosting environment.”

“The system is highly secure, responsive, and scalable, fully meeting project objectives.”



References & Thank You

- **Amazon Web Services.** (2024). EC2 User Guide for Linux Instances. Retrieved from https://docs.aws.amazon.com/ec2/
- **Django Software Foundation.** (2024). Django Documentation. Retrieved from https://docs.djangoproject.com/
- **Nginx Documentation.** (2024). Beginner's Guide. Retrieved from https://nginx.org/en/docs/beginners_guide.html

Thank You