# Wedding Website – Serverless Microservices Project

This project is a modern, cloud-native wedding website built as a full-stack serverless application on AWS. It serves both as a real-world event platform and a technical showcase of scalable architecture, authentication, and frontend–backend integration.

## Architecture Overview

The application follows a microservices-based, serverless architecture. A React (Next.js) frontend communicates with backend services through Amazon API Gateway. Each backend capability is implemented as an independent AWS Lambda function, allowing clean separation of concerns and horizontal scalability.

## AWS Services Used

**Amazon API Gateway** – Provides HTTP endpoints for frontend-to-backend communication and handles CORS and routing.
**AWS Lambda** – Python-based microservices that handle invite authentication, guest access, and data retrieval.
**Amazon DynamoDB** – NoSQL database used for invite codes, party data, and guest records with low-latency access.
**AWS IAM** – Manages least-privilege permissions between Lambda functions and AWS resources.
**Amazon S3 + CloudFront (planned)** – Will host and globally distribute the frontend as a static web application.
**Amazon Cognito (planned)** – Will provide secure authentication for admin-only features.

## Key Technical Concepts

• Stateless authentication using JWTs for guest access
• Serverless, pay-per-use backend architecture
• Clean separation between frontend, services, and infrastructure
• Secure access controls via IAM and scoped tokens
• Infrastructure designed for easy CI/CD and future expansion

## Purpose

This project demonstrates real-world application design using AWS serverless services. It highlights cloud architecture decision-making, backend API design, authentication flows, and frontend integration, while remaining practical and user-focused.