A Serverless AI-Powered Diet Management Application

Abdul Ahad Khan, Chetan Chandane, Shardul Gadadare, Sourav Patil, Vinayaka Viswanatha

[1] Rochester Institute of Technology
Rochester, New York

{vv1629, ak7160, cc5831, scg6975, sp1513}@g.rit.edu

Abstract. *The global emphasis on health-conscious living has fueled the demand for intelligent, personalized dietary management solutions.To address this growing need, AWSome Nutrition was developed as a scalable, AI-enhanced, serverless application hosted entirely on Amazon Web Services (AWS). The platform empowers users to manage their dietary habits through personalized meal plans, smart meal logging options, and real-time interaction, all driven by cloud-native technologies and artificial intelligence.*

By leveraging OpenAI's GPT-4o Turbo model and a fully serverless architecture comprising AWS Lambda, API Gateway, DynamoDB, S3, and Amplify, AWSome Nutrition delivers a seamless, responsive experience tailored to individual user needs. The platform integrates cutting-edge services like AWS Rekognition and Textract for image-based meal logging and employs Terraform for automated cloud infrastructure provisioning. The project exemplifies how serverless architecture and AI can converge to offer scalable, personalized health solutions without the burden of managing infrastructure.

1. Introduction

The increasing societal focus on wellness and nutrition, combined with advancements in cloud computing and artificial intelligence, has created an opportunity to develop smarter dietary management platforms. Traditional meal planning applications often fall short by offering static, non-personalized recommendations, failing to adapt to the unique goals and constraints of individual users. In response to these limitations, we designed and built AWSome Nutrition, a cloud-hosted, AI-driven application intended to redefine user engagement in diet management. AWSome Nutrition is engineered with four key objectives in mind:

- **Personalization at Scale:** The platform leverages OpenAI's GPT-4o Turbo to generate customized meal plans based on user-specific parameters such as fitness goals, dietary restrictions, daily caloric needs, and personal food preferences. This ensures that each user receives a truly individualized experience rather than a one-size-fits-all solution.
- **Smart Meal Tracking:** Users can log their meals through three distinct methods: manual entry, one-tap logging for returning subscribers, and image-based scanning using Optical Character Recognition (OCR). AWS Rekognition and Textract services allow users to capture nutrition labels and seamlessly extract relevant data for meal tracking.

- **Serverless Cloud Hosting:** The backend is entirely serverless, powered by AWS Lambda, API Gateway, Cognito, and DynamoDB. This architecture enables auto-scaling, reduced operational overhead, and cost-efficient computation without the need for server provisioning or maintenance.
- **Cross-Platform Availability:** A modern, responsive front-end built with React is hosted via AWS Amplify, providing real-time data synchronization, secure authentication, and easy accessibility across devices.

By combining these elements, AWSome Nutrition delivers an intuitive, AI-enhanced user experience that is both scalable and adaptable to evolving user needs. Furthermore, the project highlights the advantages of adopting a cloud-native, Infrastructure-as-Code (IaC) approach using Terraform, ensuring robust, repeatable deployments and future-proof scalability.

2. System Architecture

The AWSome Nutrition platform follows a fully serverless, event-driven architecture designed for scalability, security, and high availability. At the heart of the system is a seamless interaction between frontend, authentication, serverless backend services, and AI-powered intelligence.

As illustrated in the architecture diagram (Figure 1), users interact through a responsive web application hosted on AWS Amplify. Secure authentication and user profile management are handled by Amazon Cognito. User requests, such as meal logging, diet generation, or nutrition scanning, are routed through Amazon API Gateway to invoke AWS Lambda functions. These functions orchestrate various services including DynamoDB for data persistence, OpenAI API for AI-generated responses, and AWS Rekognition and Textract for image-based meal tracking.

Terraform automates the provisioning and configuration of all cloud resources, ensuring repeatable, version-controlled deployments. AWS CloudWatch monitors the system for performance metrics and logs, enhancing observability and fault recovery.

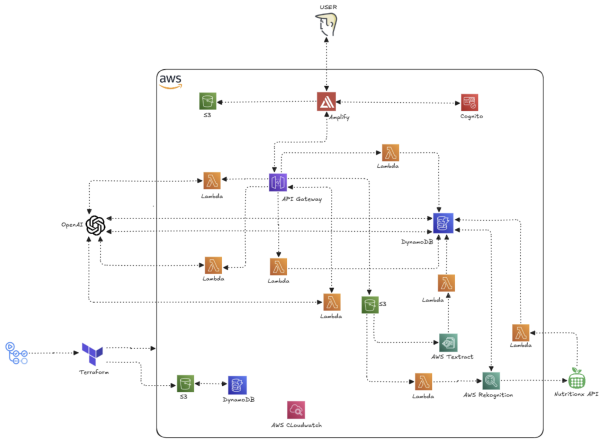

Figure 1. System Architecture of AWSome Nutrition Platform

3. AWS Technologies

3.1. Frontend: AWS Amplify Hosting

The frontend is a React-based web application hosted on AWS Amplify. Amplify offers CI/CD pipelines, automatic build deployments, environment-specific configuration injection, and out-of-the-box integrations with Cognito and API Gateway for secure API calls.
Key Features: Rapid deployment, seamless authentication, environment-aware configurations.

3.2. Authentication: Amazon Cognito

Amazon Cognito provides secure authentication and user management. It stores critical user attributes such as age, weight, height, and health conditions, which are leveraged for personalization.
Key Features: OAuth 2.0, JWT tokens for API authorization, fine-grained IAM access controls.

3.3. Backend APIs: Amazon API Gateway

API Gateway acts as the communication bridge between the frontend and the serverless backend. It handles RESTful API requests, authorization checks via Cognito, throttling, caching, and error handling.
Key Features: CORS configuration for cross-origin requests, security enforcement, scalable API exposure.

3.4. Compute Layer: AWS Lambda Functions

Lambda functions form the core processing units of the backend. Each function is designed to perform lightweight, event-driven tasks:

- **Diet Generator Lambda:** Interacts with OpenAI API to generate personalized meal plans.
- **Meal Logger Lambda:** Processes meal entries and stores them in DynamoDB.
- **Calorie Calculator Lambda:** Computes calorie/macro breakdowns from user input.
- **Food Scanner Lambda:** Manages the pipeline with Rekognition, Textract, and Nutritionix API.
- **Chatbot Lambda:** Routes user chat messages to OpenAI and formats responses.

Key Features: Event-driven execution, automatic scaling, zero server management.

3.5. Database Layer: Amazon DynamoDB

DynamoDB is the primary data store for user profiles, generated diet plans, meal logs, and nutritional data extracted from labels. **Key Features:** On-demand scaling, millisecond latency, cost-efficient for varying workloads.

3.6. AI Image Processing

- **AWS Textract:** Extracts structured data such as calorie counts and macronutrient values from uploaded food labels.
- **AWS Rekognition:** Performs object detection and OCR on uploaded meal images to identify food items.
- **Nutritionix API:** Enriches recognized food items with detailed nutritional data.
- **OpenAI GPT-4o:** Powers the diet generation engine and chatbot interaction for natural, conversational user experiences.

3.7. Monitoring and Observability: AWS CloudWatch

All Lambda functions, API Gateway requests, and critical system metrics are logged and monitored using CloudWatch.

4. Reflections and Key Insights

4.1. Lessons Learned

Building AWSome Nutrition provided hands-on exposure to the challenges and best practices of designing cloud-native, AI-powered serverless applications. Throughout the development journey, several technical and operational insights emerged:

- **Granular IAM Permissions and CORS Management:** Although AWS services simplify serverless development, configuring fine-grained Identity and Access Management (IAM) policies and setting correct Cross-Origin Resource Sharing (CORS) headers across API Gateway and S3 posed early hurdles. Careful permission scoping and iterative debugging were required to ensure secure, seamless frontend-backend interactions.
- **Terraform for Infrastructure as Code:** Integrating Terraform into the development pipeline significantly enhanced deployment reproducibility, environment consistency, and team collaboration. By defining infrastructure declaratively, we minimized manual errors and achieved rapid provisioning of complex multi-service architectures.
- **AI Integration Considerations:** Incorporating OpenAI's GPT-4o for diet generation and chatbot functionalities markedly improved user engagement. However, effective prompt engineering was essential to control the AI's responses for relevance and conciseness. Furthermore, managing API rate limits and ensuring cost-effective usage demanded careful backend orchestration.

4.2. Cloud Benefits Realized

The AWS-native architecture enabled us to capitalize on several inherent advantages of serverless cloud computing:

- **Scalability:** AWS Lambda's auto-scaling capabilities, combined with DynamoDB's on-demand provisioning, allowed the platform to handle fluctuating traffic volumes seamlessly without any manual intervention.
- **Flexibility:** The modular Lambda-based design enabled the rapid addition of new features, such as new meal logging modes or enhanced OCR analysis, without disrupting existing services or requiring system downtime.
- **Cost-Efficiency:** Leveraging AWS's pay-as-you-go billing model ensured that resource costs remained proportional to actual usage. This was particularly beneficial during testing phases and demo deployments where traffic was highly variable.

4.3. Challenges Encountered with AWS Services

- **Parsing AWS Textract Outputs:** Textract's OCR outputs for nutrition labels often contained inconsistent key-value mappings, necessitating the development of a custom parsing and mapping layer within the Lambda functions to reliably extract calorie and macronutrient information.

- **Amplify CI/CD Environment Complexity:** Although AWS Amplify greatly streamlined frontend deployment, managing multi-environment builds and variable injection during continuous integration required overcoming unintuitive configuration hurdles.
- **Cognito-API Gateway Authentication Integration:** Establishing secure API access via Cognito-authorized tokens involved custom handling of scopes, headers, and token validation. Debugging these integrations was non-trivial and involved iterative trial-and-error combined with deep documentation review.

4.4. AI Tools Leveraged During Development

To accelerate development velocity and reduce manual overhead, we strategically utilized AI-based developer tools:

- **OpenAI GPT-4o Turbo:** Employed for generating personalized diet plans and crafting dynamic chatbot responses, thus eliminating the need for complex, hand-coded rule engines. This integration elevated the platform's intelligence and personalization capabilities.
- **ChatGPT and GitHub Copilot:** Used extensively for generating boilerplate Lambda handlers, Terraform infrastructure templates, and API Gateway configurations. This assisted in cutting down repetitive development tasks by approximately 30–40%, allowing the team to focus more on business logic and user experience enhancements.

4.5. Potential Enhancements

With additional development cycles, the AWSome Nutrition platform could be extended to deliver even richer and more adaptive experiences:

- **Gamification and Reward Systems:** Implement user engagement features such as point-based rewards for meal logging consistency, weekly challenges, and achievement badges, enhancing user motivation and platform stickiness.
- **Leaderboards and Social Engagement:** Develop community features where users can view leaderboards based on meal logging streaks, dietary adherence, or fitness achievements, fostering a sense of competition and camaraderie.
- **Fitness Device Integrations:** Expand data collection by integrating real-time calorie tracking and activity syncing from fitness devices such as Fitbit, Apple Health, and Google Fit. This would enable a more holistic view of users' health profiles and better align diet planning with actual physical activity.

5. Cost Breakdown and Analysis

The AWSome Nutrition platform was carefully architected to maximize the benefits of cloud scalability while maintaining cost-efficiency. An in-depth cost estimation was performed using the AWS Pricing Calculator and OpenAI's usage-based API pricing models. This section outlines the expected expenses associated with running the application in a production-like environment.

- Amazon DynamoDB constitutes the largest portion of the AWS expenses. Given the need for highly available, low-latency storage for user profiles, meal logs, and

diet plans, a provisioned capacity setup was modeled, resulting in an estimated monthly cost of approximately $4,797.63. This includes write/read throughput reservations and data storage.

- AWS Amplify Hosting services the React frontend application, supporting continuous deployment pipelines, custom domain management, and hosting fees. The estimated monthly cost for Amplify is $361.50.
- Amazon Cognito user authentication services were projected at approximately $500 per month, assuming 10,000 monthly active users (MAUs) on the Lite tier.
- AWS Lambda functions, despite auto-scaling on demand, contribute a moderate cost of $330.57 per month due to frequent invocations triggered by API Gateway requests and backend workflows.
- Amazon API Gateway incurs an estimated $105 per month, based on the number of API requests processed, payload sizes, and caching settings.
- Amazon Rekognition and Amazon Textract, used for image and OCR-based meal tracking, have relatively modest monthly costs, approximately $500 and $0.41 respectively, given a processing volume of 3 images per day and 270 pages per month.
- Amazon S3 storage charges are minimal, roughly $0.53 per month, as it primarily stores lightweight images uploaded by users for food label analysis.
- AWS CloudWatch was assumed to remain within the free tier for basic logging and metric monitoring.

The total AWS infrastructure cost for one year, including the upfront reserved capacity payment for DynamoDB, is projected at approximately $109,897.69, as shown in the attached AWS Pricing Calculator screenshot. This estimate covers all serverless compute, storage, database, monitoring, and hosting components necessary to operate AWSome Nutrition at scale.

6. External API Costs (OpenAI GPT-4o Turbo)

In addition to AWS services, AWSome Nutrition relies on OpenAI's GPT-4o Turbo model for generating personalized diet plans, nutrient calculations, and conversational chatbot interactions.

The OpenAI API usage was modeled based on projected application demand (see attached external API cost breakdown figure):

- **Diet Generator:** Estimated at 7,000 API calls per month, with an average payload of 4,000 input and 4,000 output tokens per call. At a cost of $0.16 per call, the total monthly expense is projected at $1,120.
- **Nutrient Calculator:** Estimated at 3,000 API calls per month with lighter token usage (200 in/200 out). With a lower per-call cost of $0.008, the projected monthly cost is $240.
- **Chatbot:** Estimated at 100,000 chat interactions per month (assuming 10,000 users averaging 10 conversations each). With a token usage of 500 input and 500 output tokens per call and a per-call cost of $0.02, the projected monthly expense is $2,000.