AI Trip Advisor: Cloud-Native Travel Planning Platform
Cloud Engineering Course Project
Team 4

Team Members:

Sultan Fahim - Project Lead & DevOps Engineer

James McKim - Backend

Karan Khademi - AI Specialist

Will Huff - Frontend & UX Engineer

Clinton Ugwuanyi - Data & Storage Architect

Robert Iacovella - QA & Documentation Specialist

Course: Cloud Engineering
Institution: West Chester University of Pennsylvania
Date: 09/28/25

CHAPTER 1: PROJECT VISION AND ARCHITECTURE

1.1 Executive Summary: Revolutionizing Travel Planning with AI

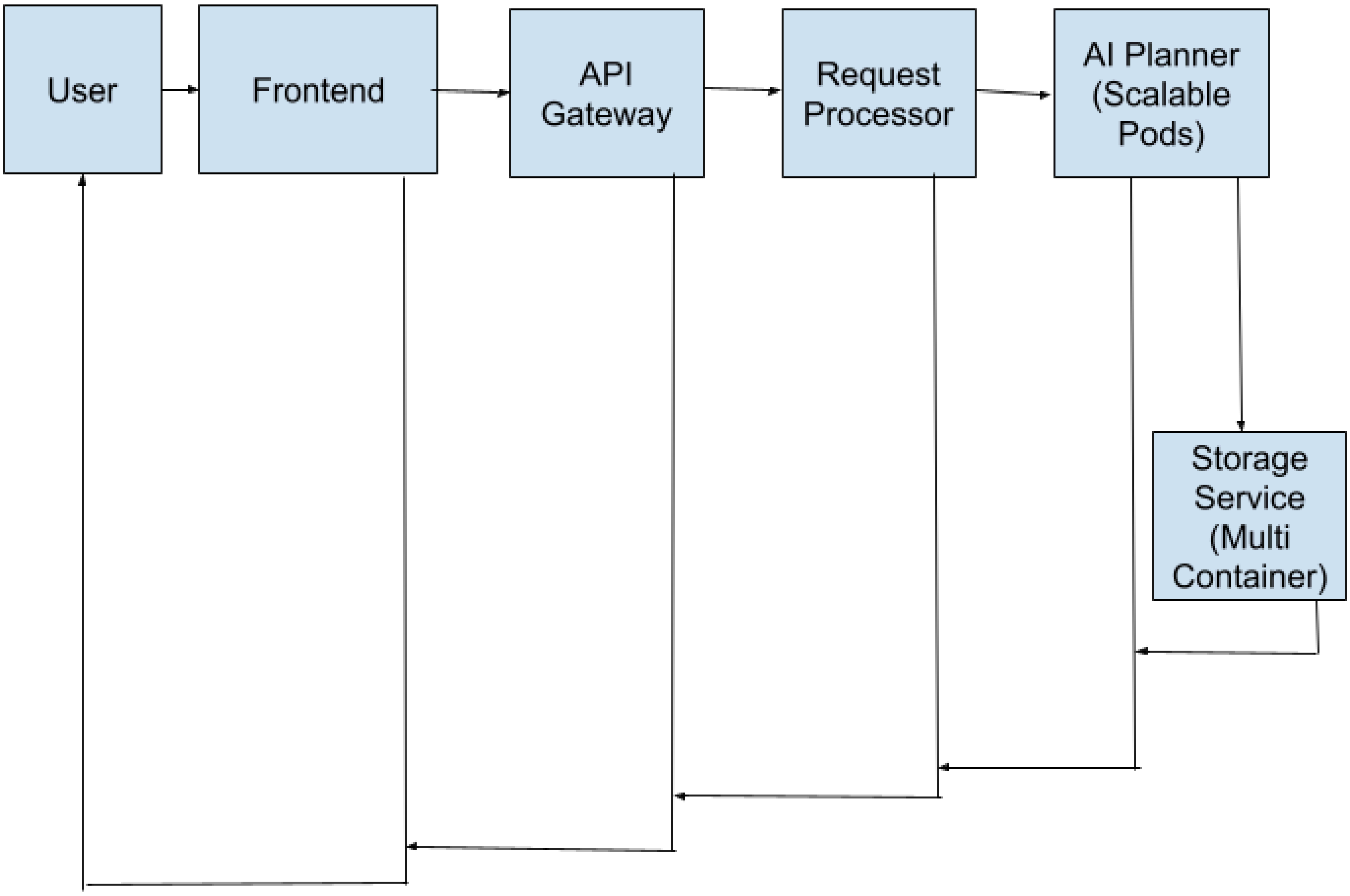
The travel industry represents a $9 trillion global market, yet travelers continue to face significant challenges in planning personalized, budget-optimized itineraries. Traditional travel planning involves hours of research across multiple platforms, often resulting in generic recommendations that fail to account for individual preferences, constraints, and real-time factors. Our project, AI Trip Advisor, addresses this gap by leveraging cloud-native technologies to deliver intelligent, personalized travel planning at scale.

Why This Matters: Modern travelers seek unique, personalized experiences but lack the tools to efficiently plan trips that align with their specific preferences, budgets, and constraints. By combining artificial intelligence with cloud orchestration, we can democratize access to high-quality travel planning that was previously available only through expensive travel agents.

Our Vision: To create a scalable, resilient platform that transforms raw travel preferences into comprehensive, AI-generated itineraries within seconds, providing users with personalized travel experiences tailored to their unique requirements.

1.2 10,000-Foot Architecture Overview

Our architecture follows a microservices-based, cloud-native approach deployed on Kubernetes, ensuring scalability, resilience, and maintainability. The system is designed around six core components that work in concert to deliver the travel planning experience:



1.3 Architecture Elaboration

The architecture embodies cloud-native principles through several key design decisions:

Microservices Segmentation: Each component serves a distinct business capability, allowing independent scaling, deployment, and technology choices. The API Gateway acts as the single entry point, routing requests to appropriate services while providing cross-cutting concerns like authentication and rate limiting.

Multi-Container Pod Strategy: The Request Processor pod combines validation logic with a metrics sidecar, demonstrating the sidecar pattern for cross-cutting concerns. Similarly, the Storage Service integrates database operations with a backup sidecar, ensuring data durability without complicating the primary application logic.

Scalability Focus: The AI Planner service is designed as a scalable deployment, recognizing that AI inference represents the most computationally intensive aspect of our pipeline. This allows horizontal scaling based on demand while maintaining cost efficiency during low-traffic periods.

Persistence Layer: Using Persistent Volume Claims (PVCs) ensures data survives pod restarts and can be backed up independently. The multi-container approach to storage separates operational concerns from data protection responsibilities.

Service Mesh Ready: The architecture is prepared for service mesh integration, with clear service boundaries and well-defined communication patterns that could benefit from advanced traffic management, security, and observability features.

This architectural foundation supports our vision of a resilient, scalable platform that can evolve from a course project to a production-ready system capable of handling real-world travel planning demands.

CHAPTER 2: IMPLEMENTATION APPROACH

2.1 Technical Implementation Strategy

Our implementation follows an iterative, milestone-driven approach aligned with cloud engineering best practices:

Infrastructure as Code (IaC) Foundation: All Kubernetes resources are defined declaratively through YAML manifests, enabling reproducible deployments across different environments (development, staging, production). This approach ensures consistency and simplifies cluster management.

Container-First Development: Each service is developed as a standalone Docker container, promoting the "write once, run anywhere" philosophy. Containerization encapsulates dependencies and ensures consistent behavior across different deployment environments.

CI/CD Pipeline Design: While initially deploying manually, our architecture supports seamless integration with CI/CD pipelines. Each service can be built, tested, and deployed independently, reducing coordination overhead and accelerating development cycles.

2.2 Core Technical Components

2.2.1 Multi-Container Pod Implementation

Request Processor Pod:

Primary Container: Python/Flask application handling request validation and normalization

Sidecar Container: Metrics collector exporting Prometheus-style metrics for monitoring

Communication: Localhost networking between containers within the pod

Storage Service Pod:

Primary Container: SQLite database with REST API for trip data management

Sidecar Container: Automated backup service creating periodic database snapshots

Shared Volumes: Persistent storage mounted for both operational data and backups

2.2.2 Scalability Patterns

AI Planner Service:

Stateless design allowing unlimited horizontal scaling

Resource limits and requests defined for predictable performance

Readiness and liveness probes ensuring only healthy pods receive traffic

Deployment strategy supporting rolling updates with zero downtime

2.2.3 Data Flow and API Contracts

The system implements a clean data flow with well-defined API contracts:

Frontend → API Gateway: RESTful JSON API for trip planning requests

API Gateway → Request Processor: Validation and enrichment pipeline

Request Processor → AI Planner: Structured data for itinerary generation

AI Planner → Storage Service: Persistent storage of finalized trip plans

2.3 Development Methodology

Agile Approach: Two-week sprints focusing on specific milestones
Test-Driven Development: Unit tests for business logic, integration tests for API contracts
Documentation: Comprehensive README files, architecture diagrams, and deployment guides
Version Control: Git-based workflow with feature branching and pull requests

2.4 Risk Mitigation and Quality Assurance

Technical Risks Addressed:

Database Performance: SQLite with connection pooling and query optimization

AI Service Latency: Mock AI implementation with configurable timeouts

Resource Contention: Kubernetes resource limits and quality of service classes

Data Loss: Multi-container backup strategy with periodic verification

Testing Strategy:

Unit testing for individual service logic

Integration testing for inter-service communication

End-to-end testing for complete user workflows

Load testing for scalability validation

This implementation approach ensures we deliver a robust, production-ready system that demonstrates advanced cloud engineering concepts while maintaining practical deployability and maintainability.

Team Members' Resumes:

Sultan Fahim - Project Lead & DevOps Engineer
Contact: sf1030563@wcupa.edu
Summary: Passionate about building scalable systems and automating deployment pipelines.

Technical Skills:

Cloud Platforms: AWS, Google Cloud, Kubernetes

Containerization: Docker, Docker Compose, Container Registry

Infrastructure as Code: Terraform, Ansible, Kubernetes YAML

CI/CD: GitHub Actions, Jenkins, GitLab CI

Monitoring: Prometheus, Grafana,

Education: Master of Science in Computer Science | West Chester University of Pennsylvania | 2024-2026

Relevant Courses: Database Management

James McKim - Backend
Contact: jm1029804@wcupa.edu

Summary: Backend developer specializing in Database & microservices architecture. Experience building intelligent systems with Python and machine learning frameworks.

Technical Skills
Programming: Python, Java, Node.js

AI/ML: TensorFlow, PyTorch, Scikit-learn

APIs: REST, GraphQL, FastAPI, Flask

Databases: PostgreSQL, MongoDB, Redis

Testing: pytest, unittest, Postman

Education

Bachelor of Science in Computer Science | West Chester University of Pennsylvania

Relevant Courses: Machine Learning, Natural Language Processing, Neural Networks

Karan Khademi - AI Specialist

Summary: Specializing in AI integration and microservices architecture. Experience building intelligent systems with Python and machine learning frameworks.

Technical Skills

Programming: Python, Java, Node.js

AI/ML: TensorFlow, PyTorch, Scikit-learn

APIs: REST, GraphQL, FastAPI, Flask

Databases: PostgreSQL, MongoDB, Redis

Testing: pytest, unittest, Postman

Education:

Bachelor of Science in Computer Science | West Chester University of Pennsylvania

Relevant Courses: Machine Learning, Natural Language Processing, Neural Networks

Will Huff - Frontend & UX Engineer

Contact: [willhuff05@gmail.com] | [610-999-1628] | [linkedin.com/in/willhuff92]
Summary: Frontend developer with expertise in modern JavaScript frameworks and user-centered design. Focus on creating intuitive interfaces with optimal performance.

Technical Skills

Frontend: React, Vue.js, HTML5, CSS3, JavaScript/TypeScript

Mobile: React Native, Progressive Web Apps

Tools: Webpack, Babel, ESLint, Figma

Testing: Jest, Cypress, React Testing Library

Education
Bachelor of Science in Human-Computer Interaction | [University Name] | 2020-2024

Relevant Courses: User Interface Design, Web Development, Interaction Design

Awards
Best UX Design - University Hackathon 2023

Clinton Ugwuanyi - Data & Storage Architect

Summary: Data engineer specializing in database design, data pipelines, and storage optimization. Experience with both SQL and NoSQL databases in cloud environments.

Technical Skills
Databases: PostgreSQL, MySQL, MongoDB, Redis, SQLite

Data Engineering: Apache Spark, Kafka, Airflow

Cloud Storage: AWS S3, Google Cloud Storage, Azure Blob Storage

ETL: Python Pandas, Apache NiFi, custom pipelines

Project Experience
Real-time Analytics Platform | Data Engineer | 2023

Designed data pipeline processing 1TB+ daily data volume

Optimized query performance reducing average response time by 80%

Implemented data governance framework ensuring compliance with regulations

Education
Bachelor of Science in Computer Science | West Chester University of Pennsylvania

Relevant Courses: Database Systems, Big Data Analytics, Data Warehousing

Certifications
MongoDB Certified Developer

Google Cloud Professional Data Engineer

Robert Iacovella - QA & Documentation Specialist
Contact: RI977955@wcupa.edu
Summary: Quality assurance engineer. Focus on ensuring software reliability and comprehensive user guidance.

Technical Skills

Automation: Python, Bash

Documentation: Markdown, LaTeX