ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

CONTEXT AWARE SYSTEMS
A.Y. 2024/2025

Design and Implementation of a Geo-Aware Content Delivery Platform

Candidate:
Simone Tassi 0001186842

Introduction

- Modern mobile applications increasingly interact with the physical world, requiring systems that can **understand and react to user location in real time**
- Traditional cloud-based solutions often struggle to provide **low latency and scalability** in location-based, content-heavy scenarios
- Continuous location tracking raises **important privacy concerns**, requiring built-in mechanisms to protect user data
- This project presents a **geo-aware, edge based platform** that balances performance, scalability, and privacy

Introduction - Geofencing

Geofencing is a technique that defines virtual geographic boundaries, allowing a system to detect when a user enters, exits, or moves within a specific area.

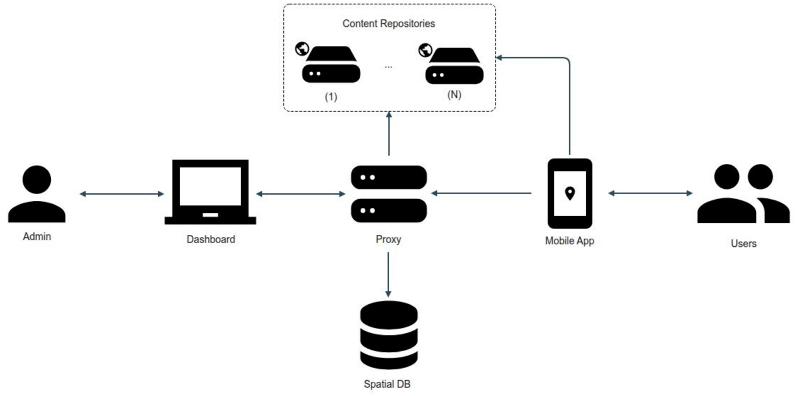
It is widely used in applications such as **location-based notifications, targeted content delivery, asset tracking, and smart city services**.

By continuously monitoring user location, systems can **trigger context-aware actions in real time**, improving user engagement and enabling personalized experiences.

Overview

- **Four main components**: a geo-aware proxy, distributed edge nodes, a mobile client, and a web-based dashboard.

- **Distributed Architecture**: a central proxy coordinating multiple edge nodes deployed across different locations to bring content closer to users.



Central Geo-Aware Proxy

Core Responsibilities

- Manages geofence definitions, content metadata, events and user identities
- Exposes REST APIs to web dashboard and mobile app
- On content request, issues a *302 Found* redirect toward the appropriate Edge Node, instead of directly delivering it to the client
- Integrates a real time WebSocket event bus to interact with the dashboard

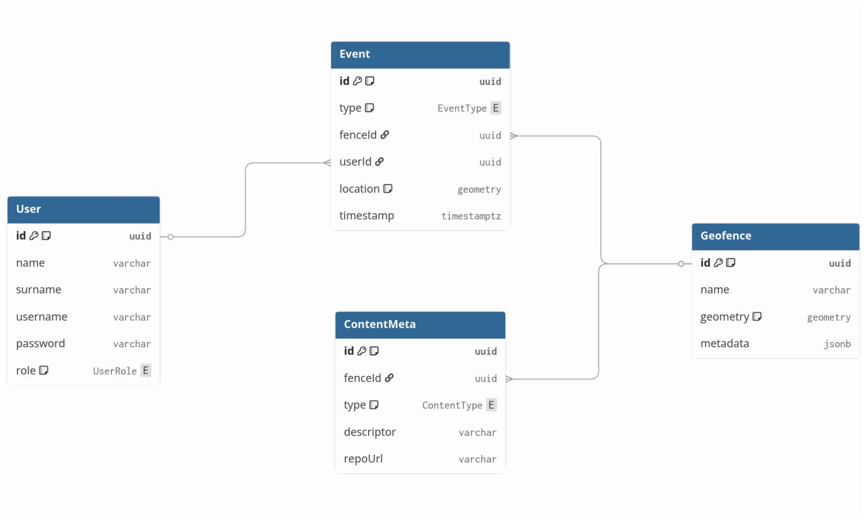
Tech Stack

- NestJS, TypeORM, PostgreSQL (with PostGIS for spatial queries)

Data Model

- The **Geofence** is the central entity: it binds spatial boundaries to **ContentMeta** (one-to-many, cascade delete), with *repoUrl* pointing to the responsible edge node

- **Events** capture **User** interactions in space and time, storing location as a native *PostGIS* geometry to power analytics and heatmaps downstream



Endpoints

Group	*Paths*	*Operations*	*Methods*
Geofences	/geofences /geofences/{id} /geofences/active	CRUD + list active	GET, POST, PUT, DELETE

ContentMeta	/content-meta
	/content-meta/by-coords
	/content-meta/{id}
	/content-meta/content/{id}
		CRUD + by coords + download	GET, POST, PATCH, DELETE

Users	/users /users/login /users/{id}	CRUD + login	GET, POST, PATCH, DELETE
Events	/events	create + list + wipe	GET, POST, DELETE
Analytics	/analytics/heatmap /analytics/metrics /analytics/clustering	heatmap + metrics + clustering	GET
Privacy Analysis	/privacy-analysis /privacy-analysis/simulate/{id} /privacy-analysis/export	simulate + export + wipe	GET, POST, DELETE

Edge Content Repository

- Lightweight **Fastify** nodes deployed close to physical landmarks
- Serving high-bandwidth assets (images, video, PDFs, ...) via **Node.js streams** → no full load into memory
- **SQLite** for local metadata persistence, keeping each node self-contained
- A **TTL mechanism** governs content freshness: expired assets return *410 Gone*, allowing the central proxy to filter out stale geofences automatically

Mobile App

React Native

Expo

Core Responsibilities

- Continuously monitor the user's position in the **background**
- Detect geofence entry/exit events
- Deliver event notifications
- Download content from the nearest edge node

Tech Stack

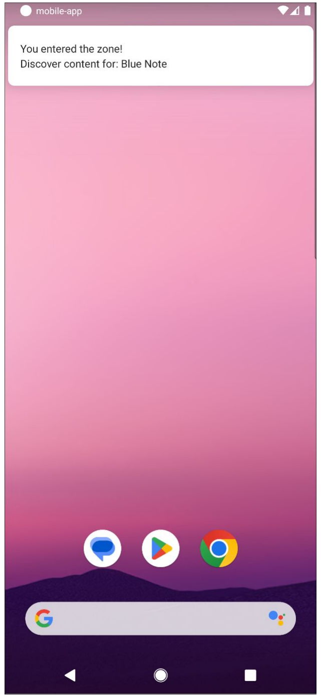
- **React Native + Expo** for cross-platform native capabilities
- **Zustand + AsyncStorage** for persistent state

Event Detection

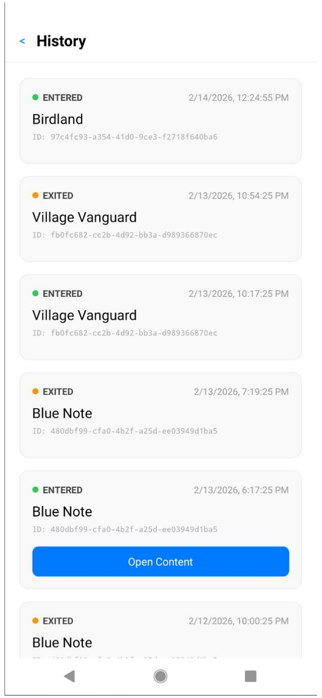
Mobile OSes only support **circular region monitoring**.

Two tier hybrid strategy:

- **Tier 1 — Coarse:** low-power native circular monitoring as a proximity filter
- **Tier 2 — Precision:** high-accuracy GPS polling with **polygon containment checks locally** via *@turf/turf*, with no server involvement



You entered the zone!
Discover content for: Blue Note



History

● ENTERED 2/14/2026, 12:24:55 PM
Birdland
ID: 97c4fc93-a354-41d0-9ce3-f2718f640ba6

● EXITED 2/13/2026, 10:54:25 PM
Village Vanguard
ID: fb0fc682-cc2b-4d92-bb3a-d989366870ec

● ENTERED 2/13/2026, 10:17:25 PM
Village Vanguard
ID: fb0fc682-cc2b-4d92-bb3a-d989366870ec

● EXITED 2/13/2026, 7:19:25 PM
Blue Note
ID: 480dbf99-cfa0-4b2f-a25d-ee03949d1ba5

● ENTERED 2/13/2026, 6:17:25 PM
Blue Note
ID: 480dbf99-cfa0-4b2f-a25d-ee03949d1ba5

● EXITED 2/12/2026, 10:00:25 PM
Blue Note

Prefetching & Caching

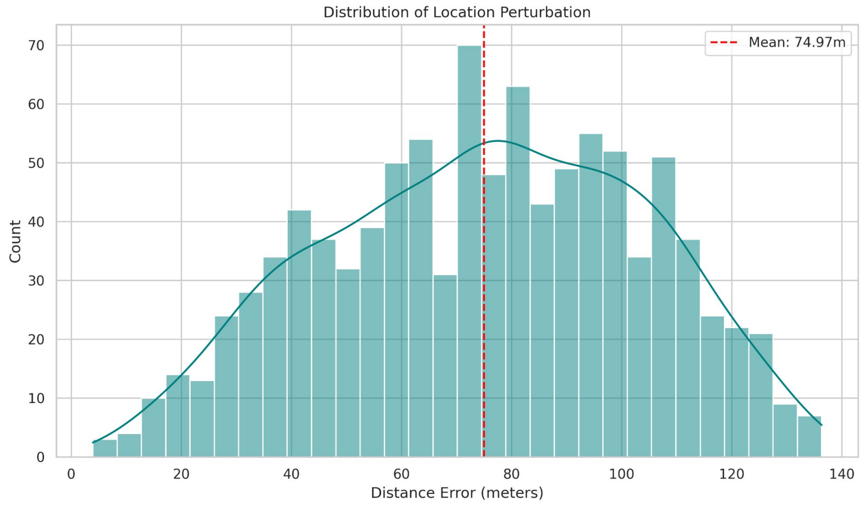
As soon as the user enters the **circular buffer** (*Tier 1*), content metadata and assets are downloaded in the background.

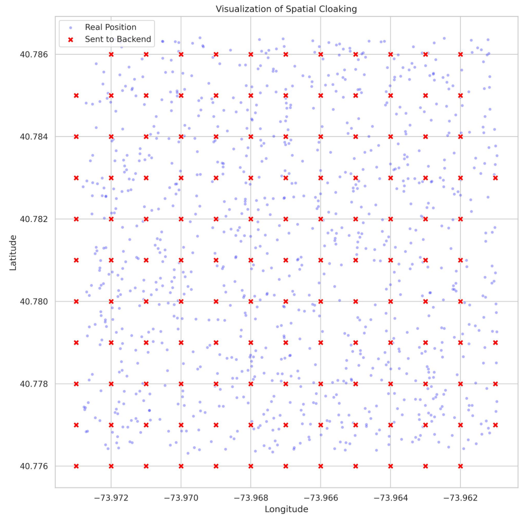
By the time the polygon is crossed and the notification fires, content is already available.

Assets are cached locally via **AsyncStorage**, enabling full offline viewing and resilience to connectivity drops.

Location Perturbation

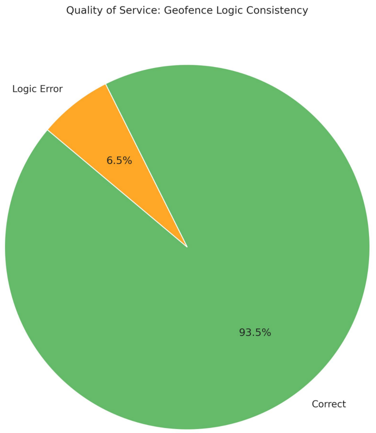
Users can toggle **Location Cloaking** from the home screen. When active, coordinates are snapped to a discrete grid (GRID_SIZE = 0.001°, ≈ 110m at the equator) before being sent to the backend

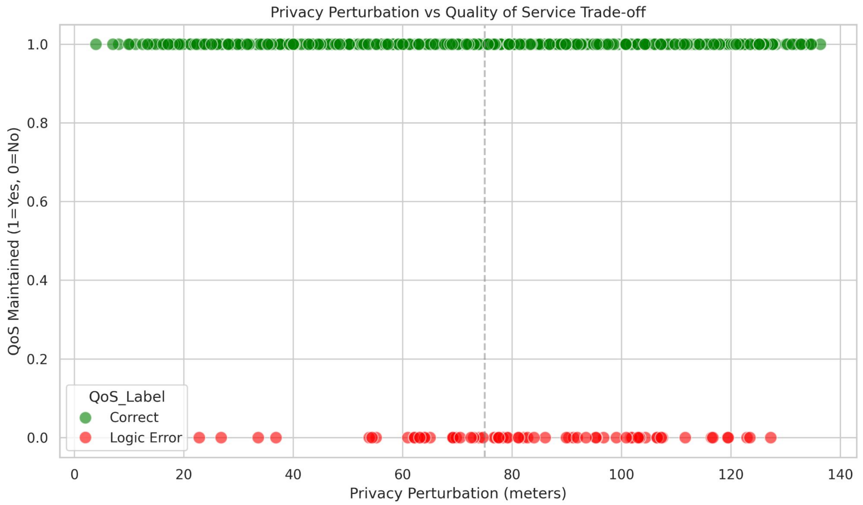




Location Perturbation

- The system maintains **high classification accuracy,** despite the spatial error introduced
- Errors are **not uniformly distributed**





Web Based Dashboard

Core Responsibilities

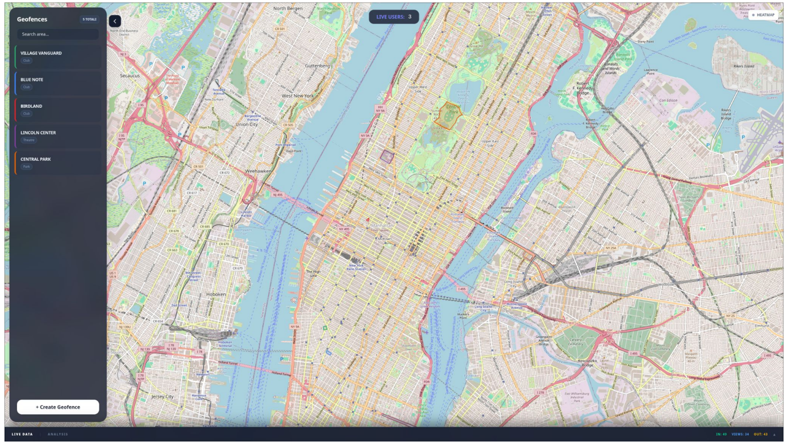
- Create and edit polygonal geofences via an interactive map editor
- Visualize live user activity and geofence events
- Show heatmaps of user engagement and clustering analytics

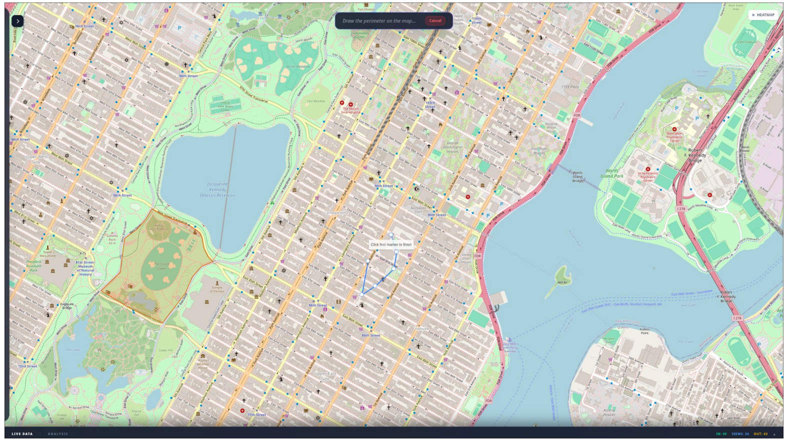
Tech Stack

- **Angular + Tailwind CSS** for a responsive UI
- **Leaflet + Geoman**: for polygon manipulation and geographical data visualization

Geofence management

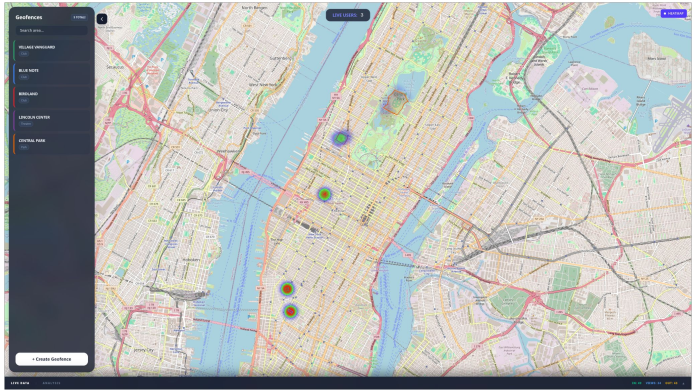
The *<u>ADMIN</u>* user is able to create, edit, visualize and filter geofences on an interactive map, enriching them with metadata such as *color* and a labelling system.

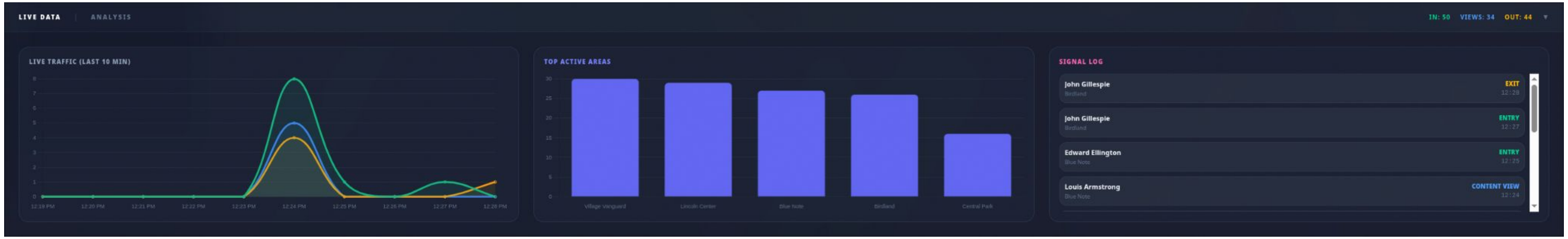



Draw the perimeter on the map...
Cancel
HEATMAP
Click first marker to finish
LIVE DATA
ANALYSIS

Live Analytics

- **Real-Time Monitoring**: Visualize live user events through a WebSocket driven tray
- **Spatial Heatmaps**: Aggregate historical event data into a dynamic heatmap layer using *leaflet.heat*





SIGNAL LOG

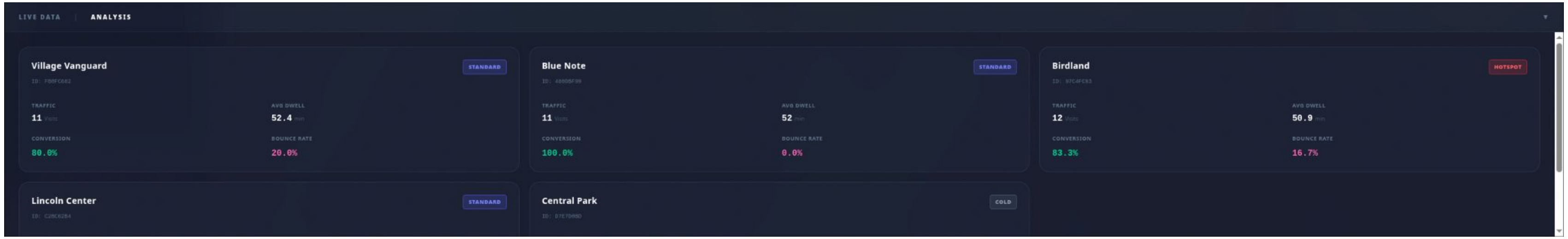
John Gillespie	Birdland	EXIT	12:28
John Gillespie	Birdland	ENTRY	12:27
Edward Ellington	Blue Note	ENTRY	12:25
Louis Armstrong	Blue Note	CONTENT VIEW	12:24

Clustering - K-Means

- An asynchronous Python task uses *scikit-learn* to group geofences into three engagement levels: **Hotspot**, **Standard**, and **Cold**

- Clusters are automatically classified by ranking their centroids based on total entry volume

Feature	**Description**
Entries	Count of geofence entry events
Conversion	views / entries
Dwell Time	Duration spent by users within the geofence



Village Vanguard STANDARD
ID: F08FC682
TRAFFIC 11 Visits
AVG DWELL 52.4 min
CONVERSION 80.0%
BOUNCE RATE 20.0%

Blue Note STANDARD
ID: 400DBF99
TRAFFIC 11 Visits
AVG DWELL 52 min
CONVERSION 100.0%
BOUNCE RATE 0.0%

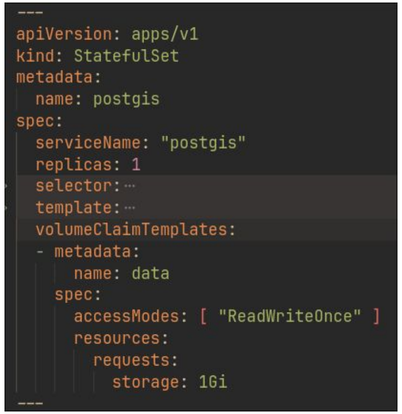
Birdland HOTSPOT
ID: 97C4FC83
TRAFFIC 12 Visits
AVG DWELL 50.9 min
CONVERSION 83.3%
BOUNCE RATE 16.7%

Lincoln Center STANDARD
ID: C2BC6284

Central Park COLD
ID: D7E7D08D

Deployment

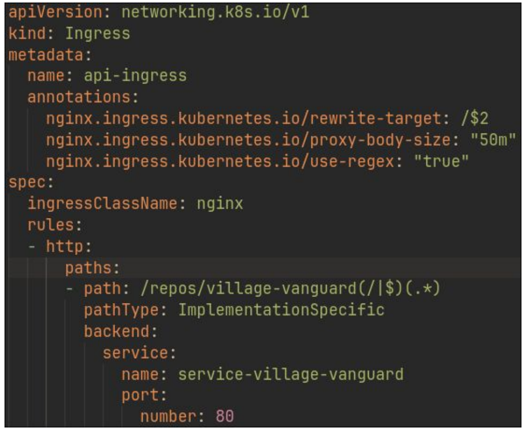
- The system is managed as a set of containerized services within a **Kubernetes cluster**, ensuring high availability and automated scaling
- A modular **Docker Compose** configuration is used for local development, providing a consistent environment across the entire development lifecycle
- Database persistence is handled via a **PostgreSQL + PostGIS** *StatefulSet*, utilizing **Persistent Volume Claims (PVC)** to ensure geographic data survives pod restarts



```yaml
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: postgis
spec:
  serviceName: "postgis"
  replicas: 1
  selector:⋯
  template:⋯
  volumeClaimTemplates:
  - metadata:
      name: data
    spec:
      accessModes: [ "ReadWriteOnce" ]
      resources:
        requests:
          storage: 1Gi
---
```

Ingress

- An **NGINX Ingress Controller** acts as the single entry point, managing path-based routing for the REST API, frontend dashboard, and edge node endpoints
- Dedicated ingress configurations enable **Socket.io** tunnels for real time dashboard updates
- *Ingress* path rewriting rules allow for redirection of traffic to specific content repositories without exposing internal cluster networking

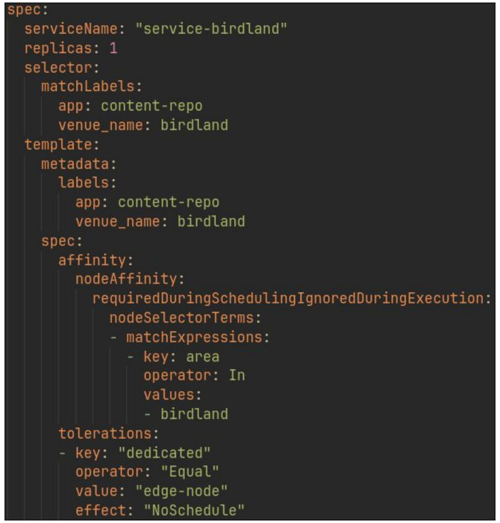


```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: api-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /$2
    nginx.ingress.kubernetes.io/proxy-body-size: "50m"
    nginx.ingress.kubernetes.io/use-regex: "true"
spec:
  ingressClassName: nginx
  rules:
  - http:
      paths:
      - path: /repos/village-vanguard(/|$)(.*)
        pathType: ImplementationSpecific
        backend:
          service:
            name: service-village-vanguard
            port:
              number: 80
```

Node Affinity - Taint/Tolerations

- Use of ***nodeAffinity*** rules pins edge services to Kubernetes nodes physically located near the corresponding geofence

- Implementation of **Taints and Tolerations** ensures that dedicated edge hardware is reserved for localized content streaming, preventing interference from backend tasks



```yaml
spec:
  serviceName: "service-birdland"
  replicas: 1
  selector:
    matchLabels:
      app: content-repo
      venue_name: birdland
  template:
    metadata:
      labels:
        app: content-repo
        venue_name: birdland
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: area
                operator: In
                values:
                - birdland
      tolerations:
      - key: "dedicated"
        operator: "Equal"
        value: "edge-node"
        effect: "NoSchedule"
```

Conclusions

- Designed and deployed a full-stack platform that bridges the gap between centralized management and localized content delivery
- Implemented an automated analytics pipeline using *K-Means clustering* and real time *heatmaps* to transform raw location events into engagement metrics
- Integrated *caching* and *prefetching* strategies to ensure availability
- Proved that *location cloaking* mechanisms can provide a significant increase in user privacy with manageable impacts on *Quality of Service*
- Leveraged *Kubernetes orchestration* to ensure the system is resilient and highly available

THANK YOU