

Mustafa Girgin
21290649

<https://github.com/antelcha/awsprojeler>

Mesajlaşma Uygulaması Üzerinden Reklam Profili Analizi Demosu

Proje 2: WebSocket ile Gerçek Zamanlı Veri Akışı ve İşleme

Bu projede AWS Kinesis, Lambda ve DynamoDB kullanılarak kullanıcıların özel odalarda mesajlaşabileceği ve mesajlarında bahsettikleri içeriklere bağlı olarak kullanıcıların reklam profili analizinin yapılacağı ve kişisel reklamların önerileceği bir demo yapılacaktır.

Öncelikle AWS'te yeni bir IAM rolü oluşturup Access key ve Secret key aldım, bunlarla işlerimi aws-cli üzerinden yürüttüm. DynamoDB, Lambda, Kinesis ve IAM üzerinde tam yetki verdim bu role.

Sonra backend tarafını yazmaya başladım. Burada önce DynamoDB ve Kinesis clientlarını tanımladım. Sonra socket kısmını yazdım ve ardından endpointleri oluşturdum.

```
server > JS index.js
> .next
  > app
    > admin
      > chat/[roomId]
        > page.tsx
          > components
            > favicon.ico
            > globals.css
            > layout.tsx
            > page.tsx
          > aws
            > node_modules
              > public
                > server
                  > lambda
                    > node_modules
                      > env
                        > JS index.js
                          > package-lock.json
                            > package.json
                              > .gitignore
                                > eslint.config.mjs
                                  > next-env.d.ts
                                    > next.config.ts
                                      > package-lock.json
                                        > package.json
                                          > postcss.config.mjs
                                            > README.md
                                              > tsconfig.json

server > JS index.js
6 const { KinesisClient, PutRecordCommand } = require('@aws-sdk/client-kinesis');
7 const { DynamoDBClient, UpdateItemCommand, ScanCommand, GetItemCommand, PutItemCommand, DeleteItemCommand } = require('@aws-sdk/client-dynamodb');
8
9 const app = express();
10 app.use(cors());
11 const server = http.createServer(app);
12 const io = new Server(server, {
13   cors: {
14     origin: "http://localhost:3002",
15     methods: ["GET", "POST"]
16   }
17 });
18
19 // AWS clients
20 const kinesisClient = new KinesisClient({ region: process.env.AWS_REGION });
21 const dynamoClient = new DynamoDBClient({ region: process.env.AWS_REGION });
22
23 // Önceden tanımlanmış kategoriler ve etiketler
24 const CATEGORIES = [
25   'teknoloji': ['yapay zeka', 'blockchain', 'metaverse', 'siber güvenlik', 'bulut bilişim', 'robotik', 'veri bilimi', '5g', 'nesnelerin interneti', 'kripto para'],
26   'kültür-sanat': ['tiyatro', 'sergi', 'müze', 'resim', 'heykel', 'opera', 'bale', 'konsert', 'festival', 'sinema'],
27   'spor': ['futbol', 'basketbol', 'voleybol', 'tenis', 'yüzme', 'atletizm', 'formula', 'boks', 'fitness', 'yoga'],
28   'bilim': ['astronomi', 'fizik', 'kimya', 'biyoloji', 'genetik', 'ekoloji', 'kuantum', 'evrim', 'uzay', 'iklim'],
29   'sağlık': ['beslenme', 'egzersiz', 'meditasyon', 'psikoloji', 'vitamin', 'bağımsızlık', 'uyku', 'stres', 'diyet', 'wellness'],
30   'eğitim': ['öğrenim', 'sınav', 'kurs', 'akademi', 'burs', 'yüksek lisans', 'doktora', 'seminer', 'workshop', 'sertifika'],
31   'ekonomi': ['borsa', 'döviz', 'yatırım', 'enflasyon', 'faiz', 'ekonomik büyüme', 'ihraç', 'ithalat', 'girişimcilik', 'fintech'],
32   'seyahat': ['turizm', 'gezi', 'tatil', 'otel', 'uçuş', 'vize', 'backpacking', 'kamp', 'doğa', 'macera'],
33   'yemek': ['gastronomi', 'restoran', 'tarifi', 'şef', 'mutfak', 'kahve', 'şarap', 'vejetaryen', 'organik', 'sokak lezzetleri'],
34   'yaşam_stili': ['moda', 'dekorasyon', 'alışveriş', 'sürdürülebilirlik', 'minimalizm', 'hobi', 'bahçe', 'ev', 'tasarım', 'vintage']
35 ];
36
37 // Reklam içerikleri
38 const AD_CONTENTS = {
39   'teknoloji': {
40     title: 'Teknoloji ürünlerinde %80'e varan indirim!',
41     content: 'Yeni teknoloji ürünleri %80'e varan indirimle! Hemen bakın!'
42   },
43   'spor': {
44     title: 'Spor ekipmanları %50 indirimde!',
45     content: 'Spor ekipmanları %50 indirimde! Hemen bakın!'
46   },
47   'eğitim': {
48     title: 'Eğitim kursları %30 indirimde!',
49     content: 'Eğitim kursları %30 indirimde! Hemen bakın!'
50   },
51   'ekonomi': {
52     title: 'Ekonomik büyüme raporları',
53     content: 'Ekonomik büyüme raporları! Hemen bakın!'
54   },
55   'seyahat': {
56     title: 'Seyahat rehberi',
57     content: 'Seyahat rehberi! Hemen bakın!'
58   },
59   'yemek': {
60     title: 'Yemek tarifleri',
61     content: 'Yemek tarifleri! Hemen bakın!'
62   },
63   'yaşam_stili': {
64     title: 'Yaşam stili',
65     content: 'Yaşam stili! Hemen bakın!'
66   }
67 };
68
69 // Socket.io ile iletişim
70 io.on('connection', (socket) => {
71   // Kullanıcı odasına katılıyor
72   socket.on('join_room', (roomId) => {
73     socket.join(roomId);
74     // Odanın mesajları
75     socket.emit('room_messages', io.sockets.adapter.rooms.get(roomId).size);
76   });
77
78   // Kullanıcı mesaj gönderiyor
79   socket.on('send_message', (roomId, message) => {
80     io.to(roomId).emit('receive_message', message);
81   });
82
83   // Kullanıcı odadan çıkıyor
84   socket.on('leave_room', (roomId) => {
85     socket.leave(roomId);
86   });
87 });
88
89 // Server'i başlat
90 server.listen(3000, () => {
91   console.log('Server started on port 3000');
92 });
```

Ardından oda numarası ve kullanıcı adı giriş ekranı olan basit bir arayüz tasarladım. Bu kısımda sadece demo proof amacıyla admin kısmı güvenliksiz açık bıraktım.

The image shows a web application interface for a chat room. On the left, there is a chat area with a header 'Oda: 123' and 'Kullanıcı: emre'. On the right, there is a dark overlay with a white 'Chat Odası' form. The form has two input fields: 'Oda Numarası' (containing '123') and 'Kullanıcı Adı' (containing 'aslı'). Below these are two buttons: 'Odaya Katıl' and 'Admin Girişi'.

Burada iki tane farklı kullanıcı kendi aralarında özel bir odada konuşabiliyorlar. Emre ve Aslı aynı odaya girdikten sonra mesajlaşma başlıyor websocket üzerinden.

Bu kısımları çok kompleks tutmamaya çalıştım. Asıl olay kinesis ve dynamodb arasında dönüyor çünkü.

The image shows a web application interface for a chat room. It is split into two panels. The left panel shows a chat area with a header 'Oda: 123' and 'Kullanıcı: emre'. The right panel shows a chat area with a header 'Oda: 123' and 'Kullanıcı: aslı'. Both panels show a conversation between Emre and Aslı. The messages are as follows:

- Emre: selam (4:54:20 PM)
- Aslı: ben ilgilenmiyorum (4:55:11 PM)
- Emre: enflasyon da çok yüksek (4:54:52 PM)
- Aslı: beslenme (4:55:12 PM)
- Emre: ne yatırım yapsam acaba (4:54:55 PM)
- Aslı: ve (4:55:13 PM)
- Aslı: sağlık (4:55:14 PM)

Amacım belirli kelimelerin spesifik bir kategoriye önceden atamak ve daha sonra bu kelimeleri kullanan kullanıcılara o kategoride reklam göstermekti. O yüzden bir keyword listesi oluşturdum ve her mesaj geldiğinde lambda ile json'a çevirip kinesis'e göndererek kinesis tarafında detaylı bir analizin yapılmasını sağladım.

Kinesis akan mesaj stream'inden spesifik kelimeleri bulduğunda, o kullanıcının reklam profilni DynamoDB üzerinde güncelleyecek şekilde ayarladım. Bu sayede her kullanıcı kendisiyle alakalı reklamları görebilecekti.

Videoda yaptığım bir demoda kullanıcılar kendi ilgi alanları hakkında konuştuktan sonra, kinesis tarafından reklam profilleri analiz edilip güncelleniyordu ve bir sonraki konuşmalarda kendileriyle alakalı reklamları görmeye başladılar.

Yatırım araçlarında sıfır komisyon!


Ekonomi ve finans araçlarında özel teklifler



Oda: 125

Kullanıcı: emre

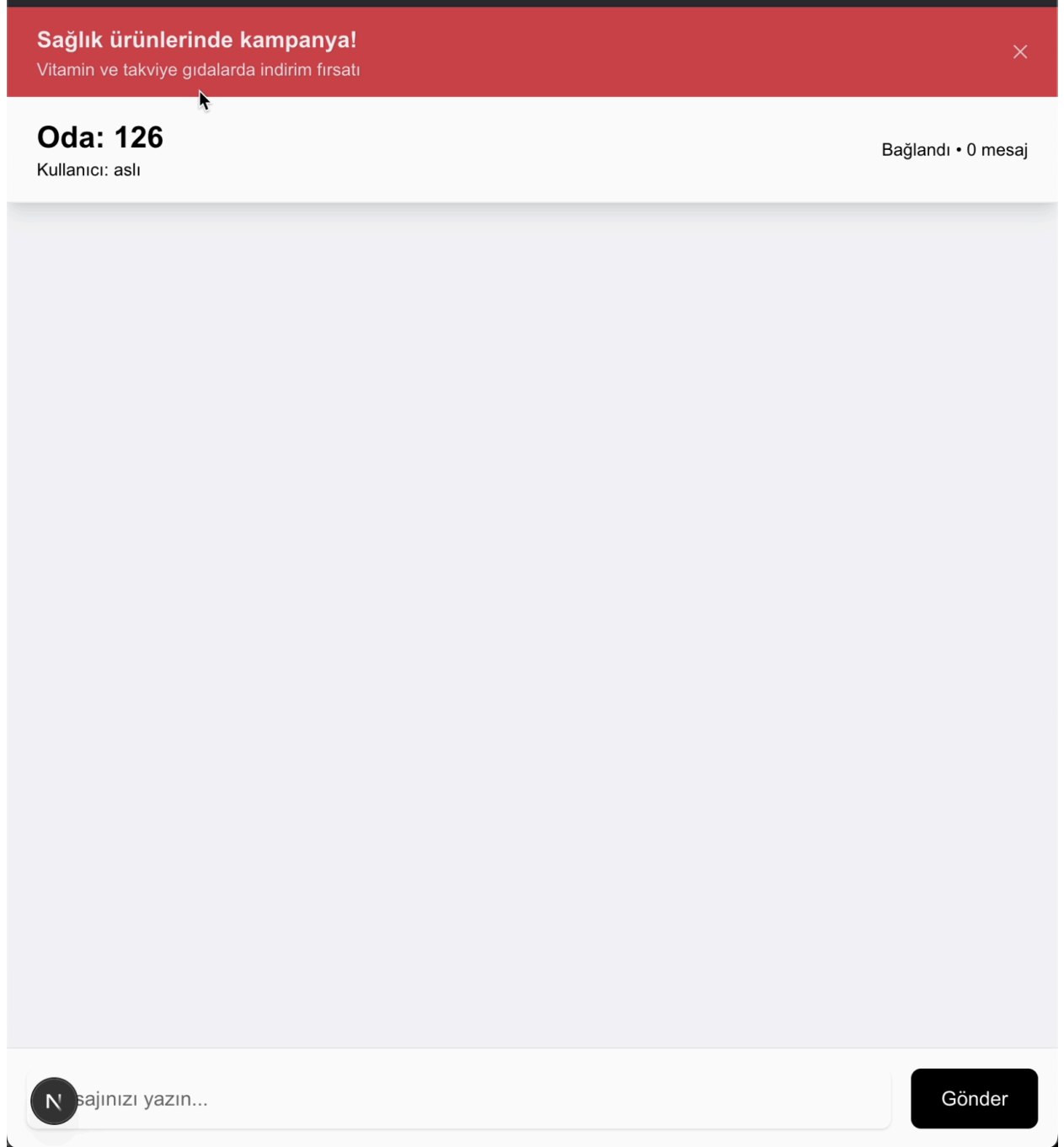
Bağlandı • 0 mesaj

 Mesajınızı yazın...

Gönder

Aslıyla yaptığı konuşmada sürekli ekonomiden bahseden Emre artık "Yatırım" ile ilgili reklamlar görmeye başladı.

Aslı ise sađlıktan bahsettiđi iin o da kendi alanıyla ilgili reklamlar grmeye bařladı.



Bu kk demoda Whatsapp ve Google'ın yaptıđı reklam profili analizlerini simle edebildim.