**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

BİLGİSAYAR MÜHENDİSLİĞİ BİTİRME ÇALIŞMASI

FRİENDLY KİŞİSEL ASİSTAN UYGULAMASI

Öğrenci1 G191210554 - Muhammed Yılmaz

Öğrenci2 B191210074 - Nazlı Nur Esmeray

|  |  |  |
| --- | --- | --- |
| Bölüm  Danışman | :  : | BİLGİSAYAR MÜHENDİSLİĞİ  Prof.Dr. Nejat YUMUŞAK |

2022-2023 Bahar Dönemi

T.C.

SAKARYA ÜNİVERSİTESİ

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

FRİENDLY KİŞİSEL ASİSTAN UYGULAMASI

BSM 498 - BİTİRME ÇALIŞMASI

Öğrenci1 G191210554 - Muhammed Yılmaz

Öğrenci2 B191210074 - Nazlı Nur Esmeray

|  |  |  |
| --- | --- | --- |
| Fakülte Anabilim Dalı | : | BİLGİSAYAR MÜHENDİSLİĞİ |

Bu tez .. / .. / … tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| ………………. | ………………. | ………………. |
| Jüri Başkanı | Üye | Üye |

**ÖNSÖZ**

Bu tez çalışması, "Friendly" adlı sesli asistan web uygulamasının hazırlanması üzerine odaklanmaktadır. Bu çalışma, sesli asistan teknolojisinin web uygulamalarında nasıl kullanılabileceğini araştırmayı ve kullanıcı deneyimini geliştirmeyi hedeflemektedir.

Gelişen teknoloji ile, insanlar arasında etkileşim ve bilgi erişimi için yeni yöntemler ortaya çıkmaktadır. Sesli asistanlar, bu bağlamda büyük bir potansiyele sahip olup, kullanıcıların doğal dil ile etkileşim kurmalarına olanak sağlamaktadır. Bu tez çalışması, sesli asistan teknolojisinin web uygulamalarıyla birleştirilerek kullanıcı deneyimini zenginleştirmeyi ve etkileşim süreçlerini daha verimli hale getirmeyi hedeflemektedir.

Tez çalışmamızda; planlanmasında, araştırılmasında, yürütülmesinde ve oluşumunda ilgi ve desteğini esirgemeyen, engin bilgi ve tecrübelerinden yararlandığımız Prof.Dr. Nejat YUMUŞAK hocamıza sonsuz teşekkürlerimizi sunarız. Bölüm olanaklarının tez çalışmalarında kullanılmasına izin verdiği için Bölüm Başkanlığına, desteklerinden dolayı Bilgisayar ve Bilişim Bilimleri Fakültesi Dekanlığına ve Sakarya Üniversitesi Rektörlüğüne teşekkürlerimizi sunarız.

Ayrıca eğitimimiz süresince bize destek veren aile fertlerimize de teşekkür ediyoruz.

**İÇİNDEKİLER**

|  |  |  |  |
| --- | --- | --- | --- |
| ÖNSÖZ……...................................................................................................... | | iii | |
| İÇİNDEKİLER.................................................................................................. | | iv | |
| ŞEKİLLER LİSTESİ......................................................................................... | | vii | |
| ÖZET................................................................................................................. | | ix | |
|  | |  | |
|  | |  | |
| BÖLÜM 1. | |  | |
| GİRİŞ................................................................................................................ | | 1 | |
|  | |  | |
| BÖLÜM 2. | |  | |
| KİŞİSEL ASİSTAN........................................................................................... | | 2 | |
| 2.1. sanal asistanların geçmişi................................................................. | | 2 | |
| 2.2. Sanal Asistanlar Hangi Teknolojiyi Kullanır................................... | | 2-3 | |
| 2.3. Yapay Zeka...................................................................................... | | 3-4 | |
| 2.4. Web Uygulamamızda Modern Teknolojilerin Kullanımı: React.js, Python ve MySQL...................................................................................  2.4.1 React.js....................................................................................  2.4.2 Python .....................................................................................  2.4.3 MySql ..................................................................................... | | 4  5  5  5-6 | |
|  | |  | |
| BÖLÜM 3. | |  | |
| REACT JS. BACKEND.................................................................................... | | 6 | |
| 3.1. React js backend............................................................................... | | 6-7 | |
| 3.2. Backend kütüphaneleri.....................................................................  3.2.1 Express kütühanesi...................................................................  3.2.2 Cors kütühanesi ........................................................................  3.2.3 Cors vs Jsonp .......................................................................... | | 7  7-8  9  10 | |
| 3.3.Mysql veritabanı modülü...................................................................  3.4 React js frontend ...............................................................................  3.4.1 Frontend kütüphaneleri............................................................  3.4.2 React-router-dom kütüphanesi.................................................  3.4.3 Axios kütüphanesi....................................................................  3.4.4 React i18next kütüphanesi.......................................................  3.4.5 React router-dom kütüphanesi.................................................  3.5 Kullancı arayüzü................................................................................  3.5.1 Giriş sayfası..............................................................................  3.5.2 Kayıt sayfası.............................................................................  3.5.3 Chatbot sayfası.........................................................................  3.5.4 Anasayfa.................................................................................. | | 10-11  11-13  13  13  14  14  14  14  15-16  16-17  17-18  19-20 | |
|  | |  | |
| BÖLÜM 4. | | |  |
| BOTLAR........................................................................................................... | | | 20-21 |
| 4.1 Python’da Kullanılan Ortak Kütüphaneler....................................... | | | 21 |
| 4.1.1. Flask...................................................................................... | | | 21-22 |
| 4.1.2. App.Route …........................................................................ | | | 22 |
| 4.2 React Uygulamasının Python Projesine Dahil Edilmesi…………..  4.3 Haberler Botu……………………………………………………... | | | 22  23 |
| 4.3.1 Haberler Botu React………………………………………...  4.3.2 Haberler Botu Python………………………………………  4.3.3 gnewsclient ………………………………………………...  4.4 Alarm Botu……………………………………………………….  4.4.1 Alarm Botu React…………………………………………..  4.4.2 Alarm Botu Python…………………………………………  4.4.3 datetime…………………………………………………….  4.4.4 time………………………………………………………...  4.4.5 threading…………………………………………………...  4.4.6 pygame ……………………………………………………  4.5 Hava Durum Botu……………………………………………….  4.5.1 Hava Durumu React……………………………………….  4.5.2 Hava Durumu Python……………………………………...  4.5.3 timedelta…………………………………………………...  4.6 Yemek Tarifi Botu……………………………………………....  4.6.1 Yemek Tarifi React……………………………………….  4.6.2 Yemek Tarifi Python……………………………………...  4.6.3 goslate…………………………………………………….  4.7 Yol Tarifi Botu…………………………………………………  4.7.1 Yol Tarifi React…………………………………………..  4.7.2 Yol Tarifi Python…………………………………………  4.7.3 import L from ‘leaflet’……………………………………  4.7.4 import ‘leaflet/dist/leaflet.css’……………………………  4.8 Çeviri Botu……………………………………………………..  4.8.1 Çeviri Botu React………………………………………...  4.8.2 Çeviri Botu Python……………………………………….  4.8.3 from googletrans import Translator……………………... | | | 23-24  24  24-25  25  25-26  26  26  27  27  27  27  27-28  28-29  29  29  29  30  30  31  31  31-32  32  32  32  32-33  33  33 |
|  | |  | |
| BÖLÜM 5. | |  | |
| SONUÇLAR ..........................………….…………………………………...... | | 34 | |
|  | |  | |
|  | |  | |
| KAYNAKLAR……………………………………………………………….. | | 35 | |
|  |
|  |
|  | |  | |
|  | |  | |

**ŞEKİLLER LİSTESİ**

|  |  |  |
| --- | --- | --- |
| Şekil 2.3. | Yapay zeka……………………...........................………... | 4 |
| Şekil 3.2.1. | Js reast api........................................................................... | 8 |
| Şekil 3.3. | MySql veri tabanı................................................................ | 11 |
| Şekil 3.5.3. | Chatbot................................................................................ | 19 |
| Şekil 4.1. | Python-flask......................................................................... | 21 |
|  |  |  |

Şekil 4.3 Haber botu........................................................................... 23

**ÖZET**

**Anahtar kelimeler**: yapay zeka , bot , web uygulamaları, kullanıcı deneyimi

Bu tez çalışması, yapay zeka temelli kişisel asistan teknolojisinin web uygulamalarında kullanılması üzerine odaklanmaktadır. kişisel asistanlar, kullanıcıların doğal dilde iletişim kurarak bilgiye erişmelerini, görevleri gerçekleştirmelerini ve günlük işlemlerini kolaylaştırmalarını sağlayan etkileşimli arayüzlerdir. Bu çalışma, yapay zeka ve kişisel asistan teknolojilerinin birleşimini kullanarak kullanıcıların internet tabanlı hizmetlere erişimini geliştirmeyi amaçlamaktadır.

Çalışmanın temel amacı, "Friendly" adlı bir kişisel asistan web uygulamasının geliştirilmesidir. Bu uygulama, kullanıcıların kişisel komutlarla çeşitli görevleri yerine getirmelerini, bilgiye erişmelerini ve etkileşimli bir şekilde web uygulamalarını kullanmalarını sağlamaktadır. Yapay zeka algoritmaları ve doğal dil işleme teknikleri kullanılarak, kullanıcıların komutlarını anlamak, işlemek ve uygun yanıtları üretmek için botlar oluşturulmuştur.

Tez çalışması, kişisel asistan teknolojisinin kullanıcı deneyimini nasıl iyileştirebileceğini ve web uygulamalarının erişilebilirliğini artırabileceğini araştırmaktadır. Kullanıcıların konuşma yoluyla etkileşimde bulunabilmesi, mobil cihazlarda veya bilgisayarlarda gezinme ve işlem yapma süreçlerini kolaylaştırmaktadır. kişisel asistanın kullanıcı dostu bir arayüzle entegre edilmesi, kullanıcıların web uygulamalarına daha hızlı ve verimli bir şekilde erişmesini sağlamaktadır.

# GİRİŞ

Günümüzde teknolojik gelişmeler, insanların günlük yaşamlarını kolaylaştırmak için önemli bir rol oynamaktadır. Bu gelişmeler, insan-makine etkileşimini iyileştirerek kullanıcı deneyimini zenginleştirmekte ve bir dizi yenilikçi uygulamanın ortaya çıkmasını sağlamaktadır. Bu tezde, "Friendly" adını verdiğimiz bir kişisel asistan web uygulaması geliştirmek için kullanılan teknikler ve yaklaşımlar incelenmektedir.

Friendly, kullanıcıların günlük görevleri yerine getirmelerine yardımcı olan bir kişisel asistan uygulamasıdır. Uygulama, kullanıcıların konuşarak komutlar vermesine ve isteklerini yerine getirmesine olanak tanır. Kullanıcının kişisel komutları anlaşılır bir şekilde algılanır ve ilgili işlemler gerçekleştirilir. Uygulama, kullanıcı deneyimini geliştirmek ve etkileşimi daha kişisel hale getirmek için yapay zeka ve doğal dil işleme tekniklerini kullanır.

Friendly'nin geliştirilmesi için, React adlı bir JavaScript kütüphanesi kullanılarak web uygulamasının kullanıcı arayüzü oluşturuldu. React, esnekliği ve performansıyla bilinen popüler bir frontend teknolojisidir. Backend tarafında ise Python programlama dili kullanıldı. Python, zengin bir kütüphane ekosistemine sahip olması ve hızlı geliştirme süreci sunmasıyla tercih edilen bir dil olarak öne çıkmaktadır. Ayrıca, verilerin depolanması için MySQL veritabanı kullanıldı.

Bu tezde, Friendly uygulamasının geliştirilme süreci, kullanılan teknolojiler ve yaklaşımlar detaylı bir şekilde ele alınacaktır. Ayrıca, uygulamanın performansı, kullanıcı deneyimi ve güvenlik gibi önemli faktörler de değerlendirilecektir. Amacımız, Friendly'nin kullanıcılar tarafından kabul gören, kullanışlı ve etkili bir kişisel asistan uygulaması olmasını sağlamaktır.

# BÖLÜM 2. KİŞİSEL ASİSTAN

**2.1 Sanal Asistanların Geçmişi**

Sanal Asistanların tarihi geçmişi sayılabilecek makinelere bakacak olursak 1952 Yılında Bell Labs tarafından tasarlanan büyük boyutlardaki makine Audrey. Audrey, yalnızca bilinen bir sesle söylenilen rakamları tanıyabiliyordu. Daha sonrasında William C. Dersch tarafından icat edilen ve 1962 Yılında ABD’nin Seatle şehrinde gerçekleşen Seattle Dünya Fuarında IBM’in tanıttığı Shoebox’ı örnek gösterebiliriz.  
Shoebox 0’dan 9’a kadar rakamlarla işaretlenmiş olan ışıkları, rakam adının söylenmesi ile rakamı tanıyıp ilgili rakamın ışığını yakabilmişti. Ayrıca bu rakamlarla toplama çıkarma işlemlerini gerçekleştirebiliyordu.  
1970’li yıllara geldiğimizde, Carnegie Mellon tarafından icat edilen Harpy, üç yaşındaki bir yaşındaki bir çocuğun aklında tutabileceği kelime sayısı kadar (1011 kelime) kelimeyi anlayabiliyordu. Yaklaşık 10 yıl kadar sonra, aynı ekip, sadece kelimeleri değil art arda gelen kelime dizilerini de anlayabilen bir sistem geliştirmiştir.

**2.2 Sanal Asistanlar Hangi Teknolojiyi Kullanır**

Sanal asistanların, doğal dil işleme (NLP - natural language processing), makine öğrenmesi (ML - machine learning) veya derin öğrenme (DL - deep learning) gibi yapay zeka teknolojileri sayesinde kullanıcıların doğal dil ile verdiği komutları anlamaları ve doğru aksiyonu belirlemeleri mümkün olur. Soruları yanıtlayan veya bir işlemi yapan sanal asistanlar dışında, daha gelişmiş sanal asistanlar kullanıcılara öneriler de sunabilirler, bunun için kullanıcı davranışlarını gözlemleme, veri analizi yapma ve tahminleme gibi alanlarda yetkin olmaları gerekir.

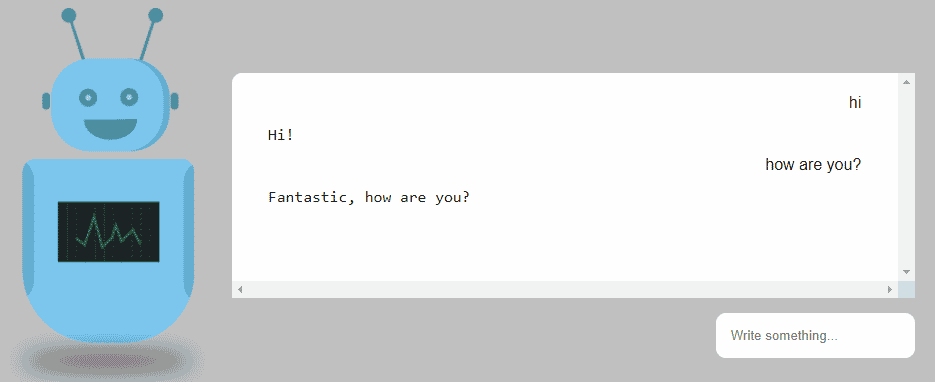
Sana asistanlar kullanıcıları ile “metin” üzerinden iletişim kurabilir. Bu tür sanal asistanlar daha çok “chatbot” olarak adlandırılır ve bir web sitesi veya mobil uygulama üzerinde yer alabilecekleri gibi, WhatsApp, Facebook Messenger, WeChat gibi yaygın mesajlaşma platformları üzerinde yer alabilirler. Bunun dışında ses ile iletişim kurabilen sanal asistanlar da vardır ve bu tür sanal asistanlar “sesli asistan” olarak adlandırılırlar. Android cep telefonlarında çalışan, Google Assistant; Amazon Echo’da çalışan Amazon Alexa; Apple cihazlarda çalışan Siri en bilinen sesli asistan örnekleridir. Sesli asistanları aktive edebilmek için kullanılan özel ifadeler vardır, örneğin Siri için “Hey Siri”, "OK Google" veya "Hey Google", "Alexa" ve "Hey Microsoft" gibi özel kelimeler ile cihaz kendisi ile konuşulduğunu algılar. Ses ile çalışabilen sanal asistanlar kurumların IVR (Interactive Voice Response) sistemlerine de entegre edilerek, buradan da hizmet verebilirler. Ayrıca sanal asistanlar araç içi sistemler ve giyilebilir teknolojiler üzerinden de kullanılabilirler.

anal asistanlar, genellikle internete bağlanabilen cihazlar veya uygulamalar dahilinde bulut teknolojilerinden faydalanarak çözümler oluşturan programlardır. Bu programlara Apple Siri, Microsoft Cortana ve Android cihazlar için ise Google Assistant’ı örnek olarak verebiliriz.

Yapay zeka destekli sanal asistan teknolojisi barındıran en popüler cihazlara ise Amazon Alexa ve Google Home örnek olarak gösterilebilir. Bu cihazlara sesiniz ile etkileşime geçerek hava durumu hakkında bilgi alabilir, e-postalarınızı kontrol edebilir, haber bülteni hakkında bilgi alabilir, müzik listenizi istediğiniz zaman başlatabilir veya diğer tanımladığınız kişisel işlemleriniz için kullanabilirsiniz.[2]

**2.3 Yapay Zeka**

Yapay zeka, bilgisayar sistemlerinin insan benzeri zekaya sahip olabilmesini hedefleyen bir disiplindir. Makine öğrenmesi, verilerden örüntüler çıkararak tahminlerde bulunabilen algoritmalar geliştirir. Doğal dil işleme, bilgisayarların insan dilini anlama ve işleme yeteneği kazanmasını sağlar. Görüntü işleme, dijital görüntüler üzerinde analiz yaparak nesneleri tanıma veya desenleri tespit etme konusunda kullanılır. Yapay zeka tabanlı robotlar, çevrelerini algılayabilir, kararlar verebilir ve karmaşık görevleri gerçekleştirebilir. Örüntü tanıma, veri kümelerindeki desenleri tanımlayarak bilgisayarların anlayabileceği şekilde sınıflandırma yapar. Yapay zeka, birçok sektörde etkili bir şekilde kullanılmaktadır, örneğin otomotiv, sağlık, finans ve güvenlik gibi alanlarda büyük potansiyele sahiptir.



Şekil 2.3 yapay zeka

**2.4 Web Uygulamamızda Modern Teknolojilerin Kullanımı: React.js, Python ve MySQL**

Tezimizde, web uygulaması geliştirmek için frontend olarak React.js, backend olarak Python ve veritabanı olarak MySQL'i tercih ettik. React.js, kullanıcı arayüzü geliştirmek için kullanılan popüler bir frontend kütüphanesidir. Esnek bileşen yapısı ve sanal DOM (Document Object Model) üzerindeki etkili güncelleme yetenekleri sayesinde kullanıcı dostu ve hızlı bir kullanıcı deneyimi sunmamıza yardımcı oldu. Python ise güçlü bir backend dilidir ve geniş bir kütüphane ekosistemiyle birlikte gelir. Özellikle Flask veya Django gibi Python tabanlı web çerçeveleriyle birlikte kullanıldığında, hızlı ve güvenilir bir web sunucusu oluşturmak için ideal bir seçenektir. Veritabanı olarak MySQL'i tercih ettik çünkü geniş kullanımı ve kolay entegrasyon özellikleriyle öne çıkar. İlişkisel veritabanı yönetim sistemleri arasında popüler bir tercih olan MySQL, verilerin etkin bir şekilde saklanması ve yönetilmesini sağlamada bize yardımcı oldu. Bu teknoloji kombinasyonu, web uygulamamızı başarılı bir şekilde geliştirmemizde bize esneklik, performans ve veritabanı yönetimi kolaylığı sağladı.

**2.4.1 React.js**

React, Facebook tarafından geliştirilmiş bir JavaScript kütüphanesidir ve kullanıcı arayüzü oluşturma konusunda popüler bir seçenektir. React, web sayfalarında ve uygulamalarında kullanıcı arayüzü bileşenlerinin modüler bir şekilde oluşturulmasına izin verir. Bu modüler yapı, daha büyük web uygulamalarının geliştirilmesi ve yönetilmesi için daha kolay ve ölçeklenebilir hale getirir. React, performansı artıran ve kodun daha kolay yönetilebilir olmasını sağlayan sanal DOM (Document Object Model) kullanır.

React, HTML ve CSS gibi web teknolojileri kullanır, ancak bunları daha dinamik ve ölçeklenebilir bir şekilde kullanır. React bileşenleri, JSX ve JavaScript kodu kullanarak oluşturulur ve CSS, CSS modülleri veya CSS-in-JS kütüphaneleri kullanılarak stil yönetimi yapılır.[2]

**2.4.2 Python**

Python, yapay zeka (AI) ve makine öğrenimi (ML) gibi alanlarda oldukça popüler bir programlama dilidir. Python, açık kaynaklı bir dil olduğu için, çeşitli yapay zeka ve makine öğrenimi kütüphaneleri tarafından desteklenmektedir. Python, kullanımı kolay, esnek ve okunaklı bir sözdizimine sahiptir, bu nedenle yapay zeka ve makine öğrenimi gibi karmaşık uygulamalar için ideal bir seçimdir.

**2.4.3 MySQL**

MySQL, web uygulamaları için uygun bir seçenek olabilir, çünkü hızlı, güvenli ve ölçeklenebilir bir veri tabanı yönetim sistemi olarak bilinir. MySQL, açık kaynak kodlu olduğu için, web uygulama geliştiricileri tarafından kullanımı kolaydır ve web uygulamaları için geliştirilmiş birçok araç ve entegrasyon sunmaktadır.

MySQL, yapay zeka uygulamaları için uygun bir seçenek olabilir, ancak büyük ölçekteki veri setlerini yönetmek için ölçeklenebilirlik konusunda sınırlamaları olabilir.

**BÖLÜM 3. REACT JS. ORTAMI VE KULLANIMI**

Yaptığımız projede bir backend (sunucu) dosyası oluşturdum. Verileri çağırmak ve göndermek için express, cors ve mysql kütüphanelerini kullandık. Ve frontend dosyası oluşturdum.Kullandığım kütühaneler : axios, i18next ve react-router-dom gibi önemli kütüphaneleri kullandım. ve ana uygulama arayüzünü oluşturduk .

**3.1 REACT JS. BACKEND**

React'te Backend, React uygulamasının sunucu tarafıdır. React'te Backend, sunucu üzerinde gerçekleşen işlemler ve mantığı ele almak için kullanılır. Bu, veritabanından veri çağırma, kimlik doğrulama ve yetkilendirme, hesaplama işlemleri, dosyalarla ilgilenme ve diğer veri ve harici hizmetlerle etkileşim gerektiren işlevlerle ilgilenmek için kullanılır.

React'te Backend kullanmanın birçok faydası ve önemli kullanım alanları vardır:

* Veri Çağırma: React'te Backend, farklı kaynaklardan istenen verileri çağırmak için veritabanlarına ve harici API sunucularına bağlanmak için kullanılabilir. Bu, uygulamanın harici verilerle bağlantılı olmasını ve bunları işlemlerinde kullanmasını sağlar.
* Kimlik Doğrulama ve Yetkilendirme: React'te Backend, oturum açma, kayıt olma ve hesap ayarları gibi kimlik doğrulama ve yetkilendirme işlemlerini yönetmek için kullanılabilir. Bu, JWT (JSON Web Tokens) gibi tekniklerin kullanılmasını içerir ve kullanıcı kimliğini doğrulamak ve yönetmek için kullanılır.
* Dosya İşlemleri: React'te Backend, dosya yükleme ve indirme işlemleri, sunucuda dosyaları yönetme, dosyaları saklama ve müşteri tarafıyla dosya işlemleri için bir arabirim sağlama gibi dosya işlemleriyle ilgilenmek için kullanılabilir.
* Güvenlik ve Koruma: React'te Backend, uygulamanın güvenlik ve koruma katmanını sağlar ve korur, özellikle doğrudan müşteri tarafından gerçekleştirilen işlemler için ek bir güvenlik önlemi sağlar.

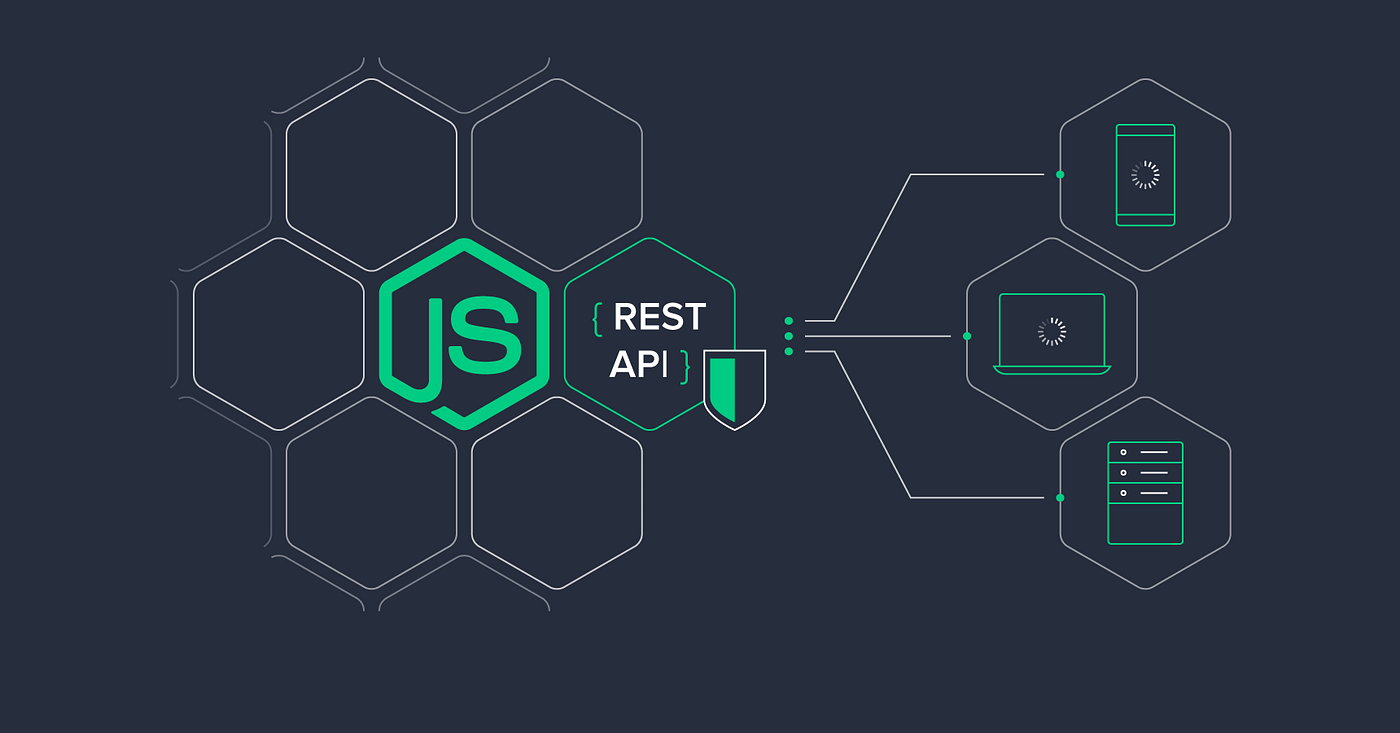
**3.2 BACKEND KÜTÜPHANELERİ**

Backend kütüphaneleri bir arada kullanarak, Express uygulamanızda bir sunucu oluşturabiliriz, istemciden gelen talepleri dinleyebilir, bu taleplere yanıtlar verebilir ve MySQL veritabanından veri çekebilir veya veri güncellemeleri yapılır. Bu şekilde, istemcilerinizle veri alışverişi yapabilir ve sunucu tarafı mantığı gerçekleştirebiliriz.

**3.2.1 EXPRESS KÜTÜPHANESİ**

JavaScript dilini kullanarak web uygulamaları geliştirmek için bir çatı (framework) sağlar. Express, Node.js uygulamalarında sunucu ve web hizmetleri oluşturmayı kolaylaştıran en popüler kütüphanelerden biridir. xpress kütüphanesi, web uygulamalarını kolaylıkla oluşturmamızı ve basit bir şekilde API (Application Programming Interface) tanımlamamızı sağlar.kullandığımız ve yaradığımız özellikler böyle anlatayım :

* Sunucu Oluşturma: Express'i kullanarak web sunucusu oluşturduk. express() fonksiyonunu kullanarak kolayca bir sunucu oluşturduk ve HTTP GET, POST , istek yöntemlerine uygun şekilde işlevleri tanımladık.
* Yönlendirme: Express, istekleri belirli işlevlere yönlendirmek için yol tanımlamsı destekler. app.get(), app.post() yönlendirme metotlarını kullanarak belirli yollara istekte bulunulduğunda hangi işlevlerin çalıştırılacağını belirledik.
* İstek ve Yanıt İşleme: Express, web isteklerimizi işlemek ve yanıtlar oluşturmak için kullanışlı işlevler sağladı. req ile gönderilen istek verilerine erişebilir ve res ile yanıt oluşturduk. Durum kodunu, içerik türünü belirleyebilir ve veriyi istemciye iletebiliriz.
* Hata İşleme: Express, uygulamamızda oluşabilecek hataları yönetmek için mekanizmalar sunar. Hataları ele almak ve uygun hata yanıtlarını göndermek için özel bir ara yazılım tanımladık.



Şekil 3.2.1 js reast api

**3.2.2 CORS KÜTÜBHANESİ**

Cors (Cross-Origin Resource Sharing), web tarayıcısının güvenlik politikaları nedeniyle farklı kökenlere (origin) yapılan HTTP isteklerini kısıtlayan bir mekanizmadır. Express uygulamanızda kullanarak, istemcilerin farklı bir kökenden (domain, protokol veya port farkı) gelen isteklere erişmesini sağlayabiliriz.

CORS, tarayıcıda çalışan bir güvenlik önlemidir ve sunucu tarafında yapılandırma gerektirir. Express'te CORS kullanarak, sunucunmuzu istemcilere hangi kaynaklara erişim izni verdiğini kontrol edebiliriz.kullandığımız ve yaradığımız özellikler böyle anlatayım :

* Güvenlik: CORS, tarayıcının aynı kaynaktan olmayan istekleri engellemesini sağlar. Bu, tarayıcının istemci tarafından güvenilmeyen kaynaklara yapılan istekleri otomatik olarak engellemesini ve sadece güvenilen kaynaklara erişime izin vermesini sağlar.
* Kaynak Paylaşımı: Bir API, farklı kökenlerden gelen istemcilerin belirli kaynaklara erişimini kontrol edebiliriz.bir web sitesi sadece belirli bir kökenden gelen isteklere veri sunabilir ve diğer kaynaklara erişimi engelleyebilir.
* Özelleştirme: CORS, hangi istek yöntemlerine (GET, POST, PUT, DELETE) ve başlıklara (headers) izin verileceğini belirlememizi sağlar. Bu sayede, API'niz üzerinde daha fazla kontrol sağlayabiliriz ve istemcilerin yalnızca belirli işlemleri gerçekleştirmesine izin verebiliriz.

Express'te CORS kullanmak için cors adlı bir kütüphane kullanılır. Yükleyip projemize ekledikten sonra, app.use(cors()) komutunu kullanarak CORS'u Express uygulamamiza ekledik. Bu şekilde, Express sunucumuzda CORS politikalarını etkinleştirir ve istemcilerin kaynaklara erişimini kontrol edebiliriz.

**3.2.3 CORS vs JSONP**

CORS, JSONP desenine modern bir alternatif olarak kullanılabilir. CORS'un faydaları şunlardır:

JSONP yalnızca GET istek yöntemini desteklerken, CORS diğer HTTP istek türlerini de destekler. CORS, daha iyi hata işleme imkanı sunan düzenli XMLHttpRequest kullanımına olanak tanır. JSONP, dış site tehlikeye düştüğünde cross-site scripting (XSS) sorunlarına yol açabilirken, CORS, güvenliği artırmak için web sitelerinin yanıtları manuel olarak ayrıştırmasına izin verir. JSONP'nin başlıca avantajı, CORS desteğinden önceki eski tarayıcılarda (Opera Mini ve Internet Explorer 9 ve öncesi) çalışma yeteneğiydi. CORS artık çoğu modern web tarayıcısı tarafından desteklenmektedir. [3]

**3.3 MYSQL VERİTABANI MODÜLÜ**

MySql modülü, Node.js uygulamalarında MySQL veritabanıyla etkileşim sağlamak için kullanılan bir modüldür. Bu modül, MySQL veritabanına bağlantı kurmayı, sorguları yürütmeyi ve sonuçları almayı sağlar.[4]

İşte kullanıcı girişi ve kullanıcı kaydı için bir web sayfası oluşturduk ve bu sayfaları bir MySQL veritabanına bağladınız ve kullanıcıları depolamak için uygun komutları kullandık. mysql modülü, MySQL veritabanıyla etkileşimde bulunmanın yanı sıra aşağıdaki faydaları da sağlar:

* Kolay bağlantı yönetimi: mysql modülü, veritabanına bağlanmayı kolaylaştırır ve bağlantı yönetimini sağladık.
* Sorgu yürütme: mysql modülü, SQL sorgularını veritabanına göndermeyi ve sonuçlarını almayı sağladık.
* Veritabanı işlemleri: mysql modülü, veritabanına veri ekleme, güncelleme, silme gibi işlemleri gerçekleştirmemize olanak tanır.
* Hata yönetimi: mysql modülü, veritabanı işlemlerinde oluşabilecek hataları yakalamamıza ve uygun şekilde işlem yapmamıza olanak sağlar.
* Çoklu bağlantı desteği: mysql modülü, birden fazla MySQL veritabanına bağlanmsı.



**Şekil 3.3 MySql veri tabanı**

**3.4 REACT JS. FRONTEND**

Frontend , bir kullanıcının web veya mobil uygulamayla etkileşim kurduğu kullanıcı arayüzünü oluşturmak için kullanılır. Frontend, kullanıcıların uygulamayla etkileşimde bulunabilmesini sağlamak için web tarayıcılarında veya mobil cihazlarda çalışan kod ve bileşenlerin kombinasyonudur.frontend yazılımının temel amacı, kullanıcıların uygulama veya web sitesi üzerinde veri girişi yapmalarını, içerikleri görüntülemelerini veya etkileşimde bulunmalarını sağlamaktır. Bu nedenle, kullanıcı arayüzünü oluşturmak, düzenlemek ve etkinleştirmek için farklı teknolojiler ve programlama dilleri kullanılır. HTML (Hypertext Markup Language) yapısal olarak sayfaları tanımlamak için kullanılırken, CSS (Cascading Style Sheets) görünüm ve stil tanımlamalarını yapar. JavaScript ise etkileşimli işlevlerin eklenmesi için kullanılan bir programlama dilidir.

Frontend nasıl oluşturduğumuzu anlatayım:

* Tasarım: Kullanıcı arayüzünün nasıl görüneceğini planlamak için bir tasarım yaptık. Bu aşamada, uygulamanın veya web sitesinin kullanıcıya ne sunacağı belirlenir ve tasarım öğeleri oluşturulur.
* HTML: Kullanıcı arayüzünün yapısını tanımlamak için HTML kullanılır. Bu, sayfaların bileşenlerini (butonlar, metin kutuları, görüntüler vb.) ve düzenini belirlemek için kullanılan işaret dilidir.
* CSS: HTML bileşenlerinin görünümünü ve stili tanımlamak için CSS kullanılır. Renkler, yazı tipleri, düzenler, geçişler vb. gibi görsel özellikler CSS ile kontrol edilir.
* JavaScript: Etkileşimli işlevlerin eklenmesi için JavaScript kullanılır. Kullanıcıların tıklama, form doldurma, veri gönderme gibi eylemlerini işlemek ve kullanıcı deneyimini geliştirmek için JavaScript kodları yazılır.
* Test ve Hata Ayıklama: Oluşturulan frontend kodları test edilir ve hatalar ayıklanır. Farklı tarayıcılarda ve cihazlarda uyumluluk kontrolü yapılır ve gerektiğinde düzeltmeler yapılır.
* Dağıtım: Frontend kodları, web sunucusuna veya uygun platformlara yüklenerek yayınlanır.

Sonuç olarak, frontend , kullanıcıların web veya mobil uygulama üzerinde etkileşimde bulunabilmesini sağlamak için kullanıcı arayüzünü oluşturur. HTML, CSS ve JavaScript gibi teknolojilerin kullanıdık.

**3.4.1 FRONTEND KÜTÜPHANELERİ**

Bu kütüphaneler, React tabanlı bir uygulamanın yönlendirme, HTTP istekleri, çoklu dil desteği ve sayfa geçişlerini yönetmesi gibi yaygın işlevlerin gerçekleştirilmesini kolaylaştırır.

**3.4.2 REACT-ROUTER-DOM KÜTÜPHANESİ**

Bu kütüphane, React uygulamaları için yönlendirme (routing) işlevselliği sağlar. Bir uygulama içinde farklı URL'ler arasında geçiş yapmayı ve sayfaları yönetmeyi kolaylaştırır. İçerisindeki temel bileşenler BrowserRouter, Route ve Routes olarak adlandırılır.

BrowserRouter: Tarayıcı türü yönlendirme sağlar ve uygulama içindeki diğer bileşenlere yönlendirme özelliği ekler.

Route: Belirli bir URL yoluna karşılık gelen bir bileşenin render edilmesini sağlar.

Routes: Birden çok Route bileşenini içeren bir grup yönlendirme tanımlamasını kolaylaştırır.

**3.4.3 AXİOS KÜTÜPHANESİ**

Bu kütüphane, HTTP istekleri yapmak için kullanılır. Web sunucusuyla iletişim kurmak ve veri alışverişi yapmak için kullandık. Özellikle API isteklerini gerçekleştirmek için sıkça kullanılan bir kütüphanedir.

**3.4.4 REACT-İ18NEXT KÜTÜPHANESİ**

Bu kütüphane, çoklu dil desteği (internationalization) sağlamak için kullanılır. Uygulamanın farklı diller arasında geçiş yapabilmesini ve yerelleştirme (localization) işlemlerini kolaylaştırır. Dil dosyalarını yönetmek ve çeviri metinlerine erişmek için kullandık.

**3.4.5 REACT-ROUTER-DOM KÜTÜPHANESİ**

useNavigate: Bu, react-router-dom kütüphanesinden gelen bir özelleştirilmiş bir kanal kullanıcısının sayfa geçişlerini yönlendirmek için kullanılan bir kancadır. Sayfa geçişlerini programatik olarak gerçekleştirmek için kullandık.

**3.5 KULLANCI ARAYÜZÜ**

Kullanıcı arayüzü (UI), bir kullanıcının bir bilgisayar programı, web sitesi veya mobil uygulama gibi bir sistemle etkileşim kurduğu grafiksel bir arayüzdür. Kullanıcı arayüzü, kullanıcının sistemle etkileşimde bulunmasını sağlayan, kullanıcıya bilgileri sunan ve kullanıcının sistemdeki işlevleri gerçekleştirmesine olanak tanıyan bir yapıdır.

**3.5.1 GİRİŞ SAYFASI**

React.js kullanarak bir giriş sayfasının frontend bileşenini oluşturduk. İşlevleri ve bileşenleri içeriyor ve kullanıcının e-posta ve şifreyle giriş yapmasını sağlıyoruz.

İçe aktarım ve bileşen tanımlamaları: İlk olarak, React, useState, Link, axios, useNavigate ve useTranslation gibi gerekli bileşenlerin veya kütüphanelerin içe aktarımları yapılıyor.

State tanımlamaları: useState hook'u kullanılarak email, password ve loginstatue adlı state değişkenleri tanımlanıyor. Bu değişkenler, kullanıcının girdiği e-posta, şifre ve giriş durumu bilgilerini tutmak için kullanılıyor.

Event işleyicileri: handleEmailChange ve handlePasswordChange fonksiyonları, kullanıcının e-posta ve şifre alanlarındaki değişiklikleri takip ediyor ve ilgili state değişkenlerini güncelliyor.

Giriş işlemi: logi fonksiyonu, kullanıcının giriş yapma butonuna tıklaması durumunda çalışıyor. Bu fonksiyon, axios kütüphanesini kullanarak belirtilen bir API endpoint'ine kullanıcının girdiği e-posta ve şifreyi gönderiyor. Sonuç olarak dönen veriye göre, giriş durumu güncelleniyor ve giriş başarılıysa /chatbot sayfasına yönlendiriliyor.

Dil değiştirme işlevi: handleclick fonksiyonu, kullanıcının dil seçeneklerinden birine tıklaması durumunda çalışıyor ve i18n kütüphanesini kullanarak uygulama dilini değiştiriyor.

HTML ve JSX yapıları: JSX kullanarak frontend arayüzünün HTML ve bileşen yapısı oluşturuluyor. div, nav, img, h1, form, input, button, label ve diğer HTML etiketleri kullanılıyor. Ayrıca, bazı bileşenlerde ikonlar (AiFillLock, AiFillMail) kullanılıyor.

**3.5.2 KAYIT SAYFASI**

bir kullanıcının kaydolabileceği bir kayıt sayfasının React bileşenini oluşturur. Kullanıcı adı, e-posta ve şifre gibi bilgileri girmek için giriş alanları içerir ve kullanıcı bu bilgileri girerek kayıt olabilir.

İlk olarak, gerekli bileşenleri ve kütüphaneleri import ediyoruz. Bu bileşenler arasında useState (durum yönetimi), axios (HTTP istekleri gönderme), useNavigate (gezinme) ve useTranslation (çeviri) gibi React kancaları yer alıyor.

Daha sonra, Signup adında bir fonksiyonel bileşen oluşturuyoruz. useTranslation kancasıyla dil çevirisi yapmak için t ve i18n öğelerini alıyoruz.

Bileşenin içindeki Signup fonksiyonunda, gerekli state'leri tanımlıyoruz. Bunlar email, password ve name olmak üzere üç tanedir. Kullanıcı bu alanlara giriş yapacak ve girişleri useState kancasıyla saklanacaktır.

Ardından, handleEmailChange, handlePasswordChange ve handlename gibi işlevler tanımlanır. Bu işlevler, kullanıcının giriş yaptığı değerleri yakalar ve ilgili state'i günceller.

Sonraki adımda, signupStatus adında bir başka state tanımlanır. Bu, kaydolma işlemiyle ilgili durumu takip etmek için kullanılır. Kaydolma işlemi sırasında ve sonrasında bu durum güncellenecektir.

sign adında bir işlev tanımlanır. Bu işlev, kullanıcının kaydolma işlemini gerçekleştirmek için çağrılır. Axios kütüphanesini kullanarak HTTP POST isteği gönderir. İsteğin gövdesinde kullanıcının girdiği e-posta, şifre ve isim bilgileri bulunur. İsteğin başarılı veya başarısız olması durumunda setSignupStatus işlevi çağrılır ve ilgili durumu günceller.

handleLinkClick işlevi, kullanıcının ana sayfaya dönmek için tıkladığında çağrılır ve navigate kancasıyla geçiş yapar.

handleclick işlevi, kullanıcının dil değiştirme düğmelerinden birine tıkladığında çağrılır. Seçilen dile göre i18n kancasıyla dil değiştirme işlemi yapar.

Son olarak, render edilen bileşenler bulunur. Bu bileşenler arasında navigasyon çubuğu, kayıt formu ve dil değiştirme düğmeleri yer alır. Kullanıcının girdiği bilgileri yakalamak için input alanları kullanılır ve ilgili state'lere bağlanır. Kayıt olma işlemini.

**3.5.3 CHATBOT SAYFASI**

React kullanarak bir sohbet botu bileşenini oluşturduk.

İlk olarak, gerekli importlar ekledik: `React`, React'tan `useState` ile durum yönetimi için ve HTTP istekleri yapmak için `axios`. Bileşen ayrıca stil uygulaması için "chatbot.css" adlı bir CSS dosyasını içe aktarıyor.

`Message` fonksiyonu, sohbet botu bileşenini temsil eder. Bu işlevin içinde, `useState` kancası kullanılarak iki durum değişkeni tanımlanır: `chat` ve `msg`.

- `chat`, sohbet geçmişini depolayan bir dizi olarak tanımlanır.

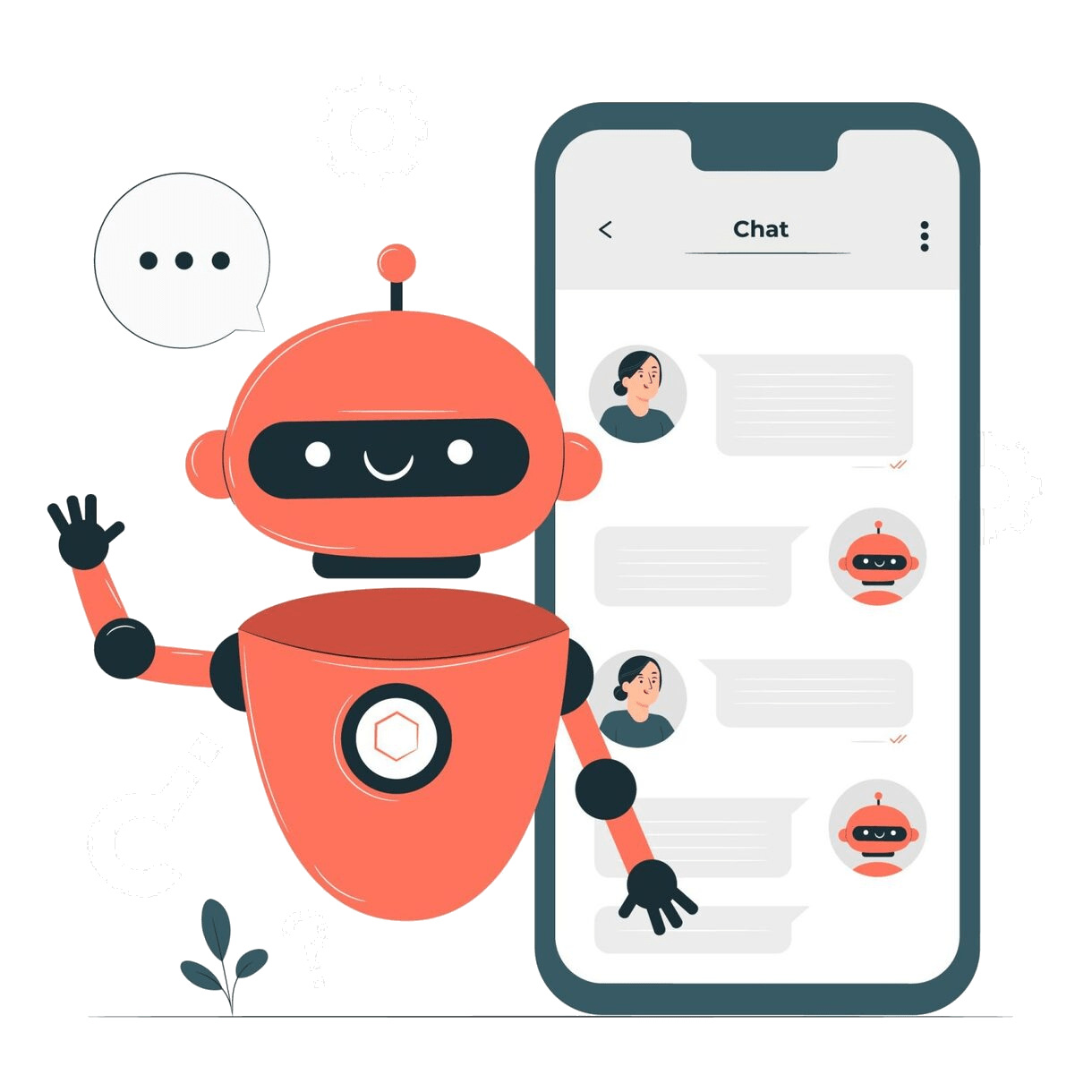
- `msg`, kullanıcının girdiği mesajı temsil eden bir dizedir.

`handleChange` fonksiyonu, giriş alanı için bir olay işleyicisidir. Giriş alanının değeri değiştiğinde, `msg` durumunu günceller.

`handleSend` fonksiyonu, gönder düğmesi için bir olay işleyicisidir. Fonksiyon, `msg` değişkeninin boş olup olmadığını kontrol eder ve ardından `axios` kullanarak bir POST isteği yapar. İstek, kullanıcının mesajını veri olarak taşıyan `http://127.0.0.1:5000/user` adresine gönderilir.

İstek başarılıysa, yanıt `.then()` bloğunda alınır. Yanıt sohbet botunun mesajını içerir. Fonksiyon, kullanıcının mesajını ve sohbet botunun yanıtını mevcut sohbet geçmişine ekleyerek `chat` durumunu günceller. Ayrıca `msg` değişkenini temizler.

Son olarak, bileşen JSX içinde döndürülür. JSX, sohbet geçmişini ekrana basar ve giriş alanı ile gönder düğmesini içerir.



Şekil 3.5.3 chatbot

**3.5.4 ANASAYFA**

React kullanarak bir metin kutusu bileşeni oluşturur. Metin kutusu, belirli bir içerikle birlikte başlıklar ve açıklamalar içeren bir dizi kutuyu içerir. Bu içerik, kullanıcılara bir uygulamanın özelliklerini tanıtmak veya bilgilendirmek için kullanılabilir. Ayrıca, react-i18next kütüphanesini kullanarak çoklu dil desteği sağlanmıştır.

İlk olarak, React, ./Anasayfa.css dosyası ve useTranslation özelliğini içe aktarıyoruz. useTranslation, dil çevirilerini kullanmak için kullanılır. Textbox adında bir işlev bileşeni tanımlıyoruz.

useTranslation özelliğini kullanarak t fonksiyonunu çıkarıyoruz. Bu fonksiyon, dil çevirilerini kullanmamızı sağlar.

Render fonksiyonunda, metin kutularını içerecek bir <div> elementi olan "metin-kutuları" sınıfına sahip bir <div> elementi oluşturuyoruz. Her bir metin kutusu için, "metin-kutusu" sınıfına sahip bir <div> elementi oluşturuyoruz.

İlk metin kutusunda, başlık ve açıklamaları içeren bir <p> elementi oluşturuyoruz. Başlık ve açıklama içeriği, t fonksiyonunu kullanarak dil çevirilerini alır. t("ac.1") ve t("ab.1") şeklindeki çağrılar, ac.1 ve ab.1 dil çeviri anahtarlarına karşılık gelen metinleri getirir. Diğer metin kutularında, başlık ve açıklamaları içeren bir <h3> ve <p> elementi oluşturuyoruz. Başlık ve açıklama içeriği statik olarak belirlenmiştir.

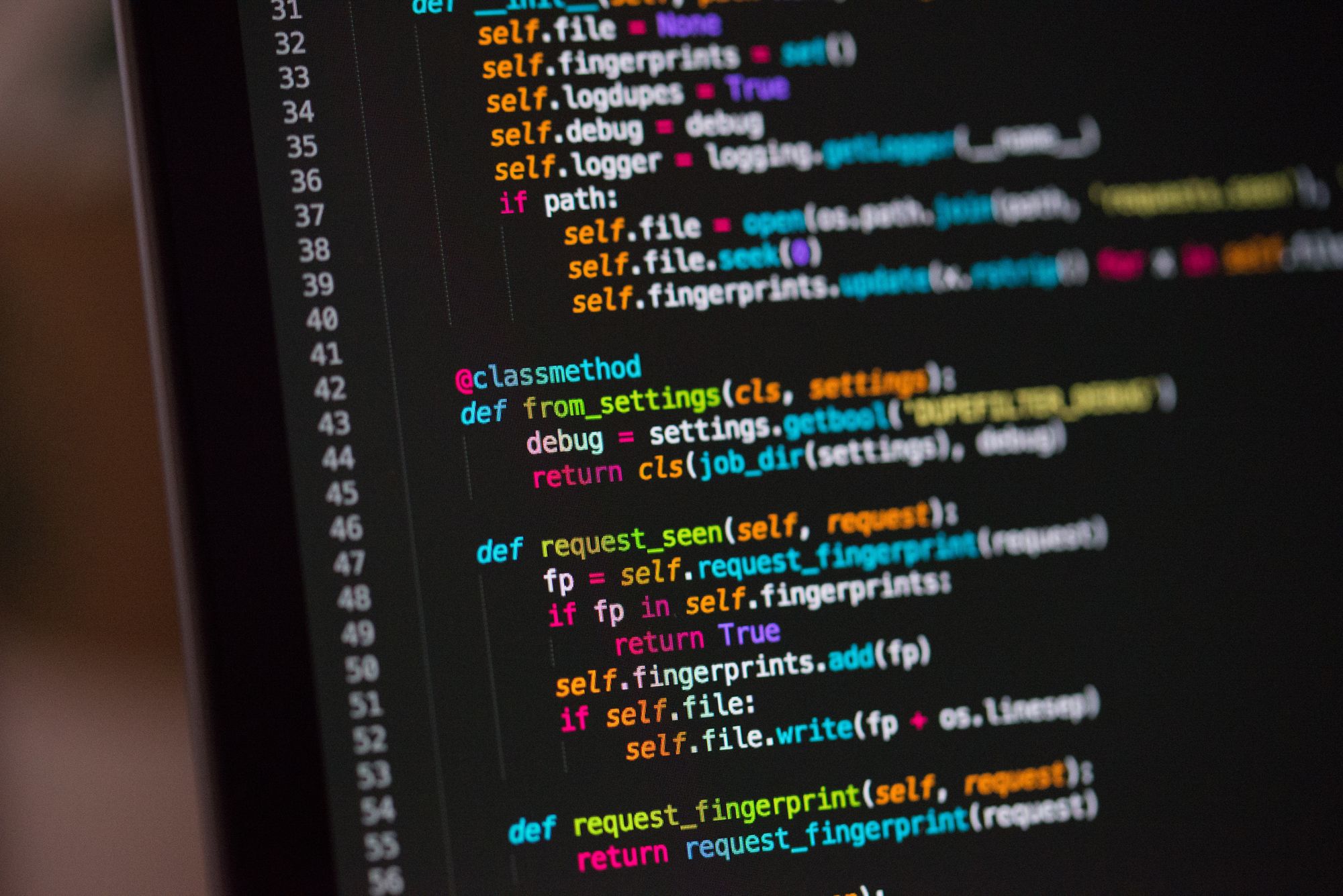
Her bir metin kutusunun içeriği, dil çevirilerini t fonksiyonunu kullanarak alır. Çeviriler, dil dosyalarında tanımlanmış olan anahtarlar aracılığıyla sağlanır. Tüm metin kutuları aynı yapısı kullanarak oluşturulur. Her bir metin kutusu, başlık ve açıklamaları içeren <h3> ve <p> elementlerini içerir. Başlık ve açıklama içeriği, dil çevirilerini kullanarak alınır.

**BÖLÜM 4. BOTLAR**

Botlar, otomatik olarak veya programlanmış bir şekilde belirli görevleri yerine getirmek veya kullanıcılara yardımcı olmak için kullanılan yazılım veya programlardır. Chatbotlar özellikle çevrimiçi platformlarda sıklıkla kullanılan botlardır. Chatbotlar, kullanıcılarla metin veya konuşma yoluyla etkileşimde bulunur ve belirli görevleri gerçekleştirmek için programlanmıştır. Bu görevler arasında bilgi sağlama, soruları yanıtlama, işlemleri gerçekleştirme veya kullanıcılara yardım etme gibi farklı işlevler bulunabilir. Genellikle doğal dil işleme, makine öğrenimi veya yapay zeka teknikleri kullanılarak geliştirilen chatbotlar, kullanıcılara interaktif ve kişiselleştirilmiş bir deneyim sunmayı hedefler. Chatbotlar, web sitelerinde, mesajlaşma uygulamalarında veya sosyal medya platformlarında müşteri hizmetleri, bilgi sağlama veya eğlence gibi çeşitli alanlarda kullanılabilir. Yaptığımız botlar arasında şunlar yer alıyor: Haberler Botu, Alarm Botu, Hava Durumu Botu, Yemek Tarifi Botu, Yol Tarifi Botu ve Çeviri Botu.

Botları oluştururken ortak kullandığımız yapılar ve kütüphaneler bulunmaktadır.

**4.1 Python’da Kullanılan Ortak Kütüphaneler**

****

**Şekil 4.1 python-flask**

**4.1.1 Flask**

Flask, Python tabanlı bir web uygulama çatısıdır. Flask, hafif ve esnek yapısıyla web uygulamalarının geliştirilmesini sağlar. Flask kütüphanesini kullanarak Python ile web uygulamaları oluşturabiliriz.

from flask import Flask, request, jsonify, send\_from\_directory

Yukarıdaki kod satırı, Flask kütüphanesinden gerekli modülleri içe aktarır. Flask modülü, Flask web uygulaması oluşturmak ve HTTP isteklerine yanıt vermek için kullanılır. request modülü, gelen isteklere erişmek için kullanılır. jsonify modülü, JSON verilerini oluşturmak için kullanılır. send\_from\_directory modülü, belirtilen bir dizindeki dosyaları istemciye göndermek için kullanılır.

**4.1.2 App.Route**

app.route, Flask web uygulamasında bir URL yolunu belirlemek için kullanılan bir dekoratördür. Bu dekoratör, belirtilen URL yoluna gelen istekleri belirli bir işlevle eşleştirir. Örneğin, aşağıdaki örnekte "/api/users" yoluna gelen istekler, users() işleviyle eşleştirilir:

@app.route('/api/users')

def users():

# İşlev içeriği

return 'Users API'

**4.2 React Uygulamasının Python Projesine Dahil Edilmesi**

React uygulamasını Python projesine dahil etmek için optimize edilmiş bir sürüm oluşturmalıyız. Bu işlemi, React uygulamasının kök dizininde bulunan npm run build komutunu kullanarak gerçekleştirebiliriz. Build işlemi tamamlandıktan sonra, optimize edilmiş React uygulamasının statik dosyaları build klasörü içinde oluşturulur. Bu build klasörünü Python projesinin statik dosyalarını barındıran bir klasöre kopyalayarak React uygulamasını Python projesine dahil edebiliriz. Python sunucusu, bu klasörü kullanarak React uygulamasının statik dosyalarına erişebilir. Bu şekilde, React uygulamasını Python projesiyle birleştirerek entegre bir web uygulaması elde edebiliriz.

**4.3 Haberler Botu**

****

Şekil 4.3 haber botu

**4.3.1 Haberler Botu React**

React kullanarak bir haberler uygulaması oluşturmayı amaçlar. İki ana bileşenden oluşur: News ve App.

News bileşeni, haberlerin kategoriye göre getirilmesi ve gösterilmesi işlemlerini gerçekleştirir. useState hook'u kullanılarak articles adında bir durum değişkeni tanımlanır. Bu değişken, haberlerin tutulacağı bir dizi olarak başlatılır. useEffect hook'u ise category değiştiğinde çalışacak olan bir etkileşim fonksiyonunu tanımlar. Bu etkileşim fonksiyonu, axios kütüphanesi kullanılarak API'den haberleri çeker ve articles durumunu günceller. Hata durumunda ise hata mesajı konsola yazdırılır. Son olarak, JSX ile haberlerin listelendiği bir yapı oluşturulur.

App bileşeni ise ana bileşendir ve haberlerin kategoriye göre filtrelenmesini sağlar. useState hook'u kullanılarak activeCategory adında bir durum değişkeni tanımlanır ve başlangıçta ilk kategori seçili olarak atanır. handleCategoryClick fonksiyonu ise kategori düğmelerine tıklama olayını yönetir ve activeCategory durumunu günceller. JSX ile News bileşeni de dahil edilerek haberlerin listelendiği ve kategori seçeneklerinin bulunduğu bir yapı oluşturulur.

Son olarak, export default App ifadesi ile App bileşeni dışarı aktarılır, böylece bu dosya başka bir yerde kullanılabilir hale gelir.

**4.3.2 Haberler Botu Python**

Python ve Flask kullanarak bir haberler uygulaması oluşturmayı amaçlar. Flask, bir web framework'üdür ve bu kodda Flask'ın Flask sınıfı kullanılarak bir uygulama oluşturulur.

fetch\_news fonksiyonu, verilen bir kategoriye göre haberleri çeken ve döndüren bir fonksiyondur. gnewsclient kütüphanesinin NewsClient sınıfı kullanılarak haberler getirilir. Ardından, her bir haber nesnesine url alanı eklenir ve title alanındaki URL'leri temizlemek için ir regex işlemi yapılır. Son olarak, haberler döndürülür.

**4.3.3 ‘gnewsclient’**

gnewsclient, Google Haberler API'sine erişim sağlayan bir Python istemci kütüphanesidir. Bu kütüphane, Google Haberler API'sini kullanarak haber başlıkları, özetler, yayınlanma tarihleri, kaynaklar ve daha fazlası gibi haber verilerini çekmenizi sağlar.

gnewsclient ile haberleri kategori, dil, konum ve yayıncı gibi filtreler kullanarak istediğiniz şekilde çekebilirsiniz. Kategoriye göre haberleri filtrelemek için topic özelliği kullanılır. Dil ve konum ayarları, haberleri belirli bir dil veya bölgeye göre sınırlamak için kullanılır. Ayrıca, yayıncıları da belirli bir yayıncıya göre filtreleyebilirsiniz.

Bu kütüphane, haber verilerini JSON formatında döndürür. Dönen verilerde her bir haber için başlık, özet, yayınlanma tarihi, kaynak URL'si gibi bilgiler bulunur. Bu verileri kullanarak haber uygulamaları, makale listeleri veya haber analiz araçları gibi çeşitli projeler geliştirebilirsiniz.

Özetlemek gerekirse, gnewsclient kütüphanesi, Google Haberler API'sine erişim sağlar ve haberleri çekmek için kullanıcı dostu bir arayüz sağlar.

**4.4 Alarm Botu**

**4.4.1 Alarm Botu React**

React ile bir Alarm Botu uygulamasını temsil eder. Uygulama, kullanıcının belirlediği saatte alarm çalmasını sağlar. Kullanıcılar, alarm saati ve çalacak olan müziği seçerek yeni alarm ekleyebilirler.

Kodun ana bileşeni, Alarm fonksiyonel bileşenidir. Bu bileşen, React Hooks kullanarak state yönetimini gerçekleştirir. İlgili state değişkenleri alarms, newAlarm ve selectedSound olarak tanımlanmıştır.

useEffect hook'u, bileşenin yüklendiği anda çağrılır ve fetchAlarms fonksiyonunu çağırarak mevcut alarm verilerini alır.

fetchAlarms fonksiyonu, Axios kütüphanesi kullanarak /alarms endpoint'ine GET isteği yapar ve alınan verileri alarms state'ine atar. Hata durumunda ise hata mesajını konsola yazar.

handleCreateAlarm fonksiyonu, kullanıcının belirlediği alarm bilgilerini alır ve /alarms endpoint'ine POST isteği yaparak yeni bir alarm ekler. Başarılı olduğunda alınan verileri alarms state'ine atar ve ilgili alanları sıfırlar. Hata durumunda hata mesajını konsola yazar.

handleDeleteAlarm fonksiyonu, belirli bir alarmı silmek için /alarms endpoint'ine DELETE isteği yapar. Başarılı olduğunda alınan verileri alarms state'ine atar. Hata durumunda hata mesajını konsola yazar.

Render bölümünde, kullanıcı arayüzü oluşturulur. Kullanıcı alarm saatini ve çalacak müziği seçebilir. Yeni alarm eklemek için "Alarm Ekle" düğmesi bulunur. Mevcut alarm listesi, alarms state'indeki verilere göre oluşturulur ve her bir alarmın yanında silme düğmesi bulunur.

Bu şekilde, kullanıcılar belirledikleri saatte alarm ekleyebilir, mevcut alarm listesini görüntüleyebilir ve istedikleri alarmı silebilir.

**4.4.2 Alarm Botu Python**

Flask kullanarak basit bir alarm uygulaması oluşturmayı amaçlamaktadır. Uygulama, POST, GET ve DELETE isteklerini kullanarak alarm oluşturmayı, alarm listesini görüntülemeyi ve alarmı silmeyi sağlar

play\_alarm fonksiyonu, pygame kütüphanesini kullanarak belirtilen ses dosyasını çalmak için kullanılır.

set\_alarm fonksiyonu, alarmın ayarlanan zamanı beklemek ve ardından play\_alarm fonksiyonunu bir thread olarak başlatmak için kullanılır.

handle\_alarms fonksiyonu, GET, POST ve DELETE isteklerini işler. GET isteği, mevcut alarm listesini döndürür. POST isteği, yeni bir alarm oluşturur ve set\_alarm fonksiyonunu başlatarak çalmasını sağlar. DELETE isteği, belirtilen alarmı alarm listesinden siler.

**4.4.3 datetime**

Bu modül, tarih ve saatle ilgili işlemleri yapmak için kullanılır. Saatleri, tarihleri, zaman damgalarını ve süreleri temsil etmek ve bunlarla ilgili hesaplamalar yapmak için kullanılabilir.

**4.4.4 time**

Bu modül, zamanla ilgili işlemleri yapmak için kullanılır. Saat dilimleri, zaman aralıkları, uyku süreleri gibi işlemleri gerçekleştirmek için çeşitli işlevler içerir.

**4.4.5 threading**

Bu modül, eş zamanlı olarak çalışan iş parçacıkları (thread) oluşturmak ve yönetmek için kullanılır. İş parçacıkları, programın aynı anda birden fazla görevi gerçekleştirmesine olanak sağlar. Bu sayede, işlemler paralel olarak yürütülebilir ve programın performansı artırılabilir.

**4.4.6 pygame**

Bu kütüphane, çoklu ortam uygulamaları geliştirmek için kullanılır. Ses, grafik ve kullanıcı girişi gibi çoklu ortam işlevlerini sağlar. Özellikle oyun geliştirme alanında sıklıkla kullanılan bir kütüphanedir. Ses çalma, grafik işleme, animasyon, kullanıcı etkileşimi gibi işlemleri kolaylaştıran bir arayüz sağlar.

**4.5 Hava Durumu Botu**

**4.5.1 Hava Durumu React**

React ile bir Hava Durumu uygulamasını temsil eder. Kullanıcı, bir şehir adı girerek hava durumu bilgisini alabilir.

Kodun ana bileşeni Weather fonksiyonel bileşenidir. Bu bileşen, React Hooks kullanarak state yönetimini gerçekleştirir. İlgili state değişkenleri city, forecast ve error olarak tanımlanmıştır.

fetchWeather fonksiyonu, async/await kullanarak hava durumu verilerini almak için axios kütüphanesini kullanır. Verileri almak için /api/weather endpoint'ine GET isteği yapar ve ilgili şehir adını parametre olarak gönderir. Gelen verilere göre, hata durumunda error state'ine hata mesajını atar, aksi halde forecast state'ine hava durumu tahminlerini atar.

handleSubmit fonksiyonu, formun gönderilme olayını (submit) yönetir. Bu fonksiyon, varsayılan davranışı önler ve fetchWeather fonksiyonunu çağırarak hava durumu verilerini alır.

Render bölümünde, kullanıcı arayüzü oluşturulur. Kullanıcı, şehir adını girebilir ve "Getir" düğmesine basarak hava durumu bilgisini alabilir. error state'inde bir hata varsa, hata mesajı görüntülenir. forecast state'inde tahmin verileri varsa, hava durumu tahminlerinin listesi görüntülenir.

Hava durumu tahminlerinin listesi, forecast state'indeki verilere göre oluşturulur. Her bir tahmin için tarih, hava durumu ikonu, sıcaklık ve hava durumu koşulu gösterilir.

Bu şekilde, kullanıcılar belirledikleri şehir için hava durumu bilgisini alabilir ve tahminleri görüntüleyebilir.

**4.5.2 Hava Durumu Python**

Flask kullanarak bir hava durumu API'si oluşturmayı ve React uygulamasının dosyalarını sunmayı sağlar.

get\_weather fonksiyonu, /api/weather endpoint'ine gelen GET isteğini yönetir. Kullanıcının istediği şehri alır ve OpenWeatherMap API'sini kullanarak hava durumu verilerini alır.

Öncelikle, current\_weather\_url ve forecast\_url olmak üzere iki ayrı API isteği yapılır. current\_weather\_url, belirli bir şehrin anlık hava durumu verilerini getirirken, forecast\_url gelecek 7 günlük hava durumu tahminlerini getirir.

İlk API isteğiyle anlık hava durumu verileri alınır. Alınan verilerin içinden sıcaklık, hava durumu koşulu ve ikon bilgisi çekilir. Bu veriler, result adında bir sözlük içinde saklanır.

İkinci API isteğiyle hava durumu tahminleri alınır. Gelecek 7 gün için tahminler alınır ve her bir günün tarihi, sıcaklık, hava durumu koşulu ve ikon bilgisi forecast listesine eklenir. Bu tahminler, result sözlüğünün bir parçası olarak saklanır.

**4.5.3 timedelta**

Bu sınıf, Python'da zaman aralıklarını temsil etmek için kullanılan bir sınıftır. İki tarih, saat veya zaman damgası arasındaki farkı ifade etmek için kullanılır.

**4.6 Yemek Tarifi Botu**

**4.6.1 Yemek Tarifi React**

Bir tarif arama uygulamasının React bileşenini temsil ediyor. Bileşen, kullanıcıdan bir tarif adı girmesini isteyen bir giriş alanı ve bir "Tarifi Al" düğmesi içeriyor. Kullanıcı bir tarif adını girdiğinde, handleSubmit işlevi tetikleniyor.

handleSubmit işlevi, tarif adını alır ve bir POST isteği kullanarak tarifin alınmasını sağlar. Tarif alındığında, tarifin ayrıntıları recipe durum değişkenine atanır.

Tarif ayrıntıları, recipe durumu dolu olduğunda görüntülenir. Tarif adı, malzemeler ve tarifin ayrıntıları HTML yapısında görüntülenir.

Bu kod, kullanıcıların bir tarif adı girerek tarifleri arayabileceği basit bir arayüz sağlar.

**4.6.2 Yemek Tarifi Python**

Flask framework'ü kullanarak bir tarif alma uygulaması oluşturuyor. Uygulama, istemci tarafından gönderilen bir tarif adını alır, bu adı İngilizce'ye çevirir ve daha sonra harici bir API'ya istek yaparak tarifin ayrıntılarını alır. Son olarak, tarifin ayrıntılarını JSON formatında yanıt olarak gönderir.

Uygulama, /recipe yolunda POST isteklerini dinler. İstek gövdesindeki JSON verisinden tarif adını alır ve get\_recipe\_from\_external\_api işlevini çağırarak tarifin ayrıntılarını alır. Alınan tarif ayrıntıları, tarif adı, malzemeler ve yapılışı şeklinde bir sözlükte saklanır ve JSON formatında yanıt olarak döndürülür.

get\_recipe\_from\_external\_api işlevi, requests kütüphanesi aracılığıyla "Edamam" adlı bir yemek tarifi API'sına istek yapar. API'ye gönderilen parametreler arasında tarif adı, uygulama kimliği (app\_id) ve uygulama anahtarı (app\_key) yer alır. API'dan gelen veriler işlenerek tarifin adı, malzemeleri ve tarifin web sayfası URL'si alınır ve geri döndürülür.

Bu kod, kullanıcının bir tarif adı göndererek tarif ayrıntılarını alabileceği bir Flask API sağlar.

**4.6.3 goslate**

Python programlarında metin çevirisi yapmak için kullanılan bir kütüphanedir. Google Translate API'sine erişim sağlar ve çeşitli diller arasında metinleri çevirmek için kullanılabilir.

**4.7 Yol Tarifi Botu**

**4.7.1 Yol Tarifi React**

Bu React bileşeni, kullanıcının başlangıç ve varış noktalarını girdiği bir yol tarifi formunu ve bu noktalar arasındaki rota bilgilerini ve haritayı gösteren bir uygulama oluşturur.

Kodda useState hook'u kullanılarak başlangıç noktası, varış noktası, mesafe, süre, harita ve rota denetimi gibi değişkenlerin durumları tutulur. Kullanıcı başlangıç ve varış noktalarını girer ve form gönderildiğinde handleFormSubmit fonksiyonu çalışır.

handleFormSubmit fonksiyonunda, axios kütüphanesi kullanılarak sunucuya bir POST isteği yapılır. Bu istekte başlangıç ve varış noktaları verileri gönderilir. Sunucudan dönen yanıt alındığında, mesafe, süre ve rota koordinatları gibi gerekli bilgiler alınır.

Daha önce oluşturulmuş bir harita varsa, önceki harita temizlenir. Ardından Leaflet harita kütüphanesi kullanılarak yeni bir harita oluşturulur ve rota için gerekli bileşenler (başlangıç ve varış noktası işaretleri, rota çizgisi eklenir. Harita görünümü, rota ile sınırlanan bir alana uyarlanır.

Sonuç olarak, kullanıcı başlangıç ve varış noktalarını girdikten sonra yol tarifi bilgileri gösterilir ve bu bilgilerle birlikte harita görüntülenir.

**4.7.2 Yol Tarifi Python**

Bu Flask uygulaması, "/directions" endpoint'ine gelen POST isteğini işleyen bir web uygulamasıdır. İstek, başlangıç ve varış konumları arasındaki mesafeyi ve süreyi hesaplamak için kullanılır. İstek JSON formatında gelmelidir ve içinde "startLocation" ve "endLocation" parametrelerini içermelidir.

Uygulama, "geopy" kütüphanesini kullanarak başlangıç ve varış konumlarının koordinatlarını alır. Ardından, "geodesic" fonksiyonunu kullanarak bu koordinatlar arasındaki mesafeyi hesaplar. Varsayılan olarak, 80 km/s hızla seyahat edildiğini kabul ederek süreyi hesaplar.

Sonuç olarak, uygulama JSON formatında bir yanıt döndürür. Yanıtta, hesaplanan mesafe ve süre bilgileri ile başlangıç ve varış konumlarının koordinatları bulunur.

**4.7.3 import L from'leaflet'**

Leaflet kütüphanesini projeye dahil etmek için kullanılır. L isimli bir nesne, Leaflet fonksiyonlarını ve özelliklerini içerir.

**4.7.4 import 'leaflet/dist/leaflet.css'**

Leaflet kütüphanesinin stil dosyasını projeye dahil etmek için kullanılır. Bu, haritaların doğru şekilde görüntülenmesi ve düzgün bir şekilde stilendirilmesi için gereklidir.

**4.8 Çeviri Botu**

**4.8.1 Çeviri Botu React**

Bu React bileşeni, bir metni farklı dillere çevirmek için kullanılan bir çeviri uygulamasını temsil eder. Kullanıcı, metni gireceği bir metin alanı seçer ve kaynak dil ile hedef dil arasında çeviri yapmak için bir kaynak dil ve hedef dil seçer.

useState kancaları, bileşen içindeki durumu yönetmek için kullanılır. text durumu, kullanıcının girdiği metni tutar. sourceLang ve targetLang durumları, kaynak ve hedef dilleri tutar. translatedText durumu, çevrilen metni tutar.

handleTranslate fonksiyonu, çeviriyi gerçekleştirmek için kullanılır. Axios kütüphanesi aracılığıyla sunucuya bir POST isteği gönderilir. İstek verileri, text, sourceLang ve targetLang değerlerini içerir. Sunucudan gelen cevapta, çevrilen metin alınır ve translatedText durumu güncellenir.

Dönüş değeri olarak, bir JSX öğesi döndürülür. Kullanıcı arayüzü, metin alanları, dil seçenekleri ve bir çeviri düğmesi içerir. Çevrilen metin, translatedText durumu varsa gösterilir.

Bu bileşen, kullanıcıların metinlerini farklı dillere çevirmelerine olanak tanır ve dinamik olarak çevrilen metni ekranda gösterir.

**4.8.2 Çeviri Botu Python**

Bu Flask uygulaması, metinleri farklı dillere çevirmek için kullanılan bir çeviri hizmetini temsil eder. googletrans kütüphanesi, çeviri işlemini gerçekleştirmek için kullanılır.

Uygulama, /translate yoluna gelen POST isteklerini dinler. İstek verileri, text (çevrilecek metin), sourceLang (kaynak dil) ve targetLang (hedef dil) değerlerini içerir. Translator sınıfı kullanılarak çeviri gerçekleştirilir. Çeviri hizmeti için translate.google.com adresi kullanılır. translate() yöntemi, çeviri yapılacak metni, kaynak dilini ve hedef dilini alır ve çevrilen metni döndürür.

Çeviri sonucu, JSON formatında translated\_text adı altında döndürülür.

**4.8.3 from googletrans import Translator**

googletrans kütüphanesi, Python programlarıyla Google Translate hizmetini kullanmayı sağlayan bir kütüphanedir.Translator sınıfı, metinleri çevirmek için kullanılır. Bu sınıf, translate() yöntemini içerir ve çeviri işlemini gerçekleştirir. translate() yöntemi, çevrilecek metni ve hedef dilin kodunu (örneğin, "en" İngilizce, "tr" Türkçe) alır .

**BÖLÜM 5.**

**SONUÇ**

Tez çalışmamız kapsamında bir web uygulaması geliştirdik. Bu uygulama, kullanıcılara kişisel bir asistan sunmayı amaçlamaktadır. Asistanımız, yapay zeka ve doğal dil işleme teknolojilerini kullanarak etkileşimli bir deneyim sağlamaktadır.

Web uygulaması React.js ile oluşturuldu ve modern bir kullanıcı arayüzüne sahiptir. Ana sayfa üzerinde kullanıcılara giriş yapma veya kayıt olma seçenekleri sunulmaktadır. Giriş yaptıktan sonra, kullanıcılar chatbot sayfasına yönlendirilir.

Chatbot sayfası, farklı alanlarda hizmet veren özel botlara sahiptir. Kullanıcılar, haberler botuyla güncel haberleri takip edebilir, çeviri botuyla farklı diller arasında metin çevirisi yapabilir, yemek tarifi botuyla lezzetli tariflere ulaşabilir, hava durumu botuyla güncel hava koşullarını kontrol edebilir, yol tarifi botuyla seyahat rotalarını planlayabilir ve alarm botuyla önemli hatırlatıcılar ayarlayabilirler.

Bu uygulama, kullanıcıların ihtiyaç duydukları bilgilere hızlı ve kolay bir şekilde erişmelerini sağlamak için tasarlandı. Veritabanı olarak MySQL kullanıldı ve kullanıcı bilgileri, chatbot verileri ve diğer ilgili veriler güvenli bir şekilde depolandı.

Kişisel asistan uygulamamız, kullanıcılara günlük yaşamda farklı konularda yardımcı olmayı hedefleyen bir platformdur. Kullanıcı dostu arayüzü ve çeşitli bot seçenekleriyle, kullanıcıların bilgiye erişimini kolaylaştırırken aynı zamanda etkileşimli bir deneyim sunar. Bu tez çalışması, yapay zeka ve web teknolojilerinin birleşimini kullanarak kullanıcıların günlük yaşamlarını kolaylaştırmayı amaçlar."

**KAYNAKLAR**

**[1] https://www.yapayzekatr.com/2019/04/30/akilli-asistan-sanal-asistan-nedir/**

**[2]** **Krill, Paul (May 15, 2014). "React: Making faster, smoother UIs for data-driven Web apps"**

**[3] caniuse.com.Retrieved 2012-07-12**

**[3] NCZOnline.Retrieved 2012-07-05**

**[4] MySQL 8.0 Reference Manual. Oracle Corporation. Retrieved 3 April 2020**