# MERT KIZILIRMAK

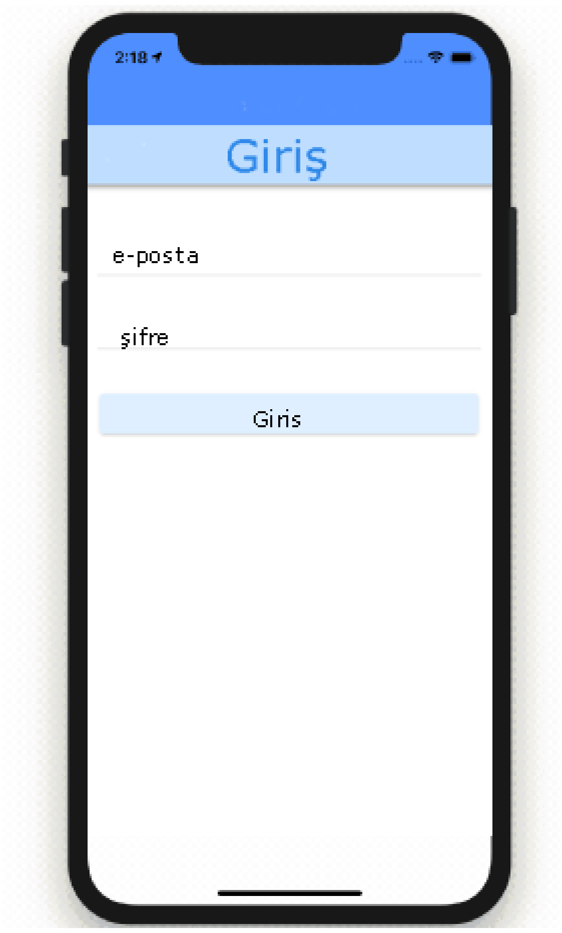
*OKUL NO: 20181132038*

SİSTEM ANALİZİ ÖDEVİ

*13.05.2020*

# İONİC  MAP APİ FİREBASE PROJESİ

## GİRİŞ EKRANI



## Auth.service.ts  dosyası

```
import { Injectable } from "@angular/core";
import * as firebase from 'firebase/app';
import { FirebaseService } from './firebase.service';
import { AngularFireAuth } from '@angular/fire/auth';

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  constructor(
    private firebaseService: FirebaseService,
    public afAuth: AngularFireAuth
  ){}

  doRegister(value){
   return new Promise<any>((resolve, reject) => {
     firebase.auth().createUserWithEmailAndPassword(value.email, value.passwor
d)
     .then(
       res => resolve(res),
       err => reject(err))
   })
  }

  doLogin(value){
   return new Promise<any>((resolve, reject) => {
     firebase.auth().signInWithEmailAndPassword(value.email, value.password)
     .then(
       res => resolve(res),
       err => reject(err))
   })
  }

  doLogout(){
    return new Promise((resolve, reject) => {
      this.afAuth.auth.signOut()
      .then(() => {
        this.firebaseService.unsubscribeOnLogOut();
        resolve();
      }).catch((error) => {
        reject();
      });
    })
  }
}
```

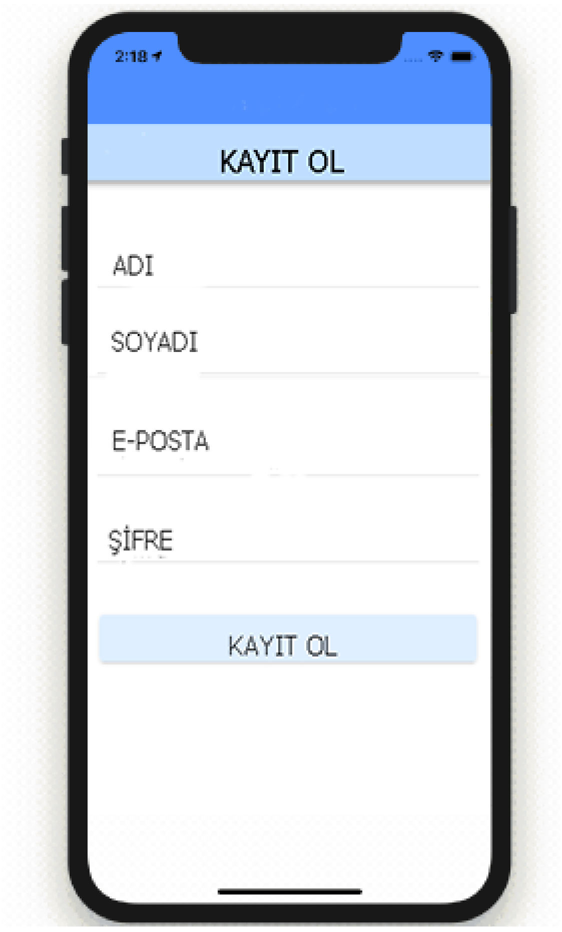
# login.page.html dosyası

```
<form class="form" [formGroup]="validations_form" (ngSubmit)="tryRegister(vali
dations_form.value)">
  <ion-item>
    <ion-label position="floating" color="Dark">e-mail</ion-label>
    <ion-input type="text" formControlName="email"></ion-input>
  </ion-item>
  <div class="validation-errors">
    <ng-container *ngFor="let validation of validation_messages.email">
      <div class="error-
message" *ngIf="validations_form.get('email').hasError(validation.type) && (va
lidations_form.get('email').dirty || validations_form.get('email').touched)">
        {{ validation.message }}
      </div>
    </ng-container>
  </div>

  <ion-item>
    <ion-label position="floating" color="primary">şifre</ion-label>
    <ion-input type="password" formControlName="Sifre"></ion-input>
  </ion-item>
  <div class="validation-errors">
    <ng-container *ngFor="let validation of validation_messages.password">
      <div class="error-
message" *ngIf="validations_form.get('password').hasError(validation.type) &&
(validations_form.get('password').dirty || validations_form.get('password').to
uched)">
        {{ validation.message }}
      </div>
    </ng-container>
  </div>

  <ion-button class="submit-
btn" expand="block" type="submit" [disabled]="!validations_form.valid">GİRİŞ
</ion-button>
  <label class="error-message">{{errorMessage}}</label>
  <label class="success-message">{{successMessage}}</label>
</form>
```

# login.page.ts dosyası

```
export class LoginPage implements OnInit {
  validations_form: FormGroup;
  errorMessage: string = '';
  validation_messages = {
    'email': [
       { type: 'required', message: 'Email is required.' },
       { type: 'pattern', message: 'Please enter a valid email.' }
    ],
   'password': [
     { type: 'required', message: 'Password is required.' },
     { type: 'minlength', message: 'Password must be at least 5
  characters long.' }
  ]};
  constructor(
   private authService: AuthService,
   private formBuilder: FormBuilder,
   private router: Router
  ) { }
  ngOnInit() {
    this.validations_form = this.formBuilder.group({
      email: new FormControl('', Validators.compose([
         Validators.required,
         Validators.pattern('^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+.[a-zA-Z0-9-.]+$')
     ])),
       password: new FormControl('',Validators.compose([
         Validators.minLength(5),
         Validators.required
       ])),
    });
  }
  tryLogin(value) {
  this.authService.Login(value).then(res => {     this.router.navigate(["/home"
]);}, err => {
  this.errorMessage = err.message;
  console.log(err)
  })
  }
  goRegisterPage() {
   this.router.navigate(["/register"]);
  }}
```

## KAYIT OL EKRANI



## Auth.service.ts  dosyası

```
import { Injectable } from "@angular/core";
import * as firebase from 'firebase/app';
import { FirebaseService } from './firebase.service';
import { AngularFireAuth } from '@angular/fire/auth';

@Injectable({
  providedIn: 'root'
})
export class AuthService {

  constructor(
    private firebaseService: FirebaseService,
    public afAuth: AngularFireAuth
  ){}

  doRegister(value){
   return new Promise<any>((resolve, reject) => {
     firebase.auth().createUserWithEmailAndPassword(value.email, value.passwor
d)
     .then(
       res => resolve(res),
       err => reject(err))
   })
  }

  doLogin(value){
   return new Promise<any>((resolve, reject) => {
     firebase.auth().signInWithEmailAndPassword(value.email, value.password)
     .then(
       res => resolve(res),
       err => reject(err))
   })
  }

  doLogout(){
    return new Promise((resolve, reject) => {
      this.afAuth.auth.signOut()
      .then(() => {
        this.firebaseService.unsubscribeOnLogOut();
        resolve();
      }).catch((error) => {
        reject();
      });
    })
  }
}
```

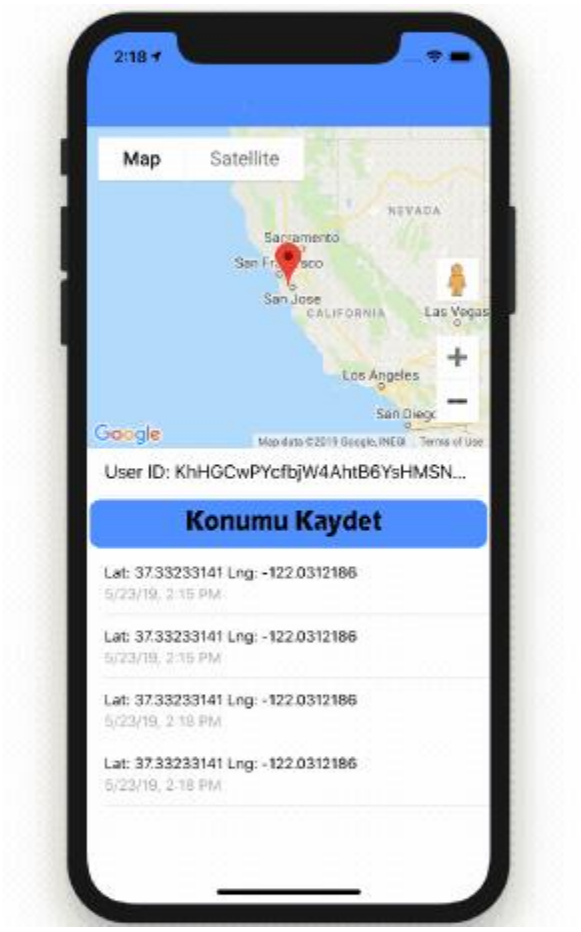
## login.page.html dosyası

```
<form class="form" [formGroup]="validations_form" (ngSubmit)="tryRegister(validations_form.value)">
  <ion-item>
    <ion-label position="floating" color="Dark">ADI </ion-label>
    <ion-input type="text" formControlName="adi"></ion-input>
  </ion-item>
  <div class="validation-errors">
    <ng-container *ngFor="let validation of validation_messages.email">
      <div class="error-message" *ngIf="validations_form.get('adi').hasError(validation.type) && (validations_form.get('adi').dirty || validations_form.get('adi').touched)">
        {{ validation.message }}
      </div>
    </ng-container>
  </div>
  <ion-item>
    <ion-label position="floating" color="Dark">SOYADI </ion-label>
    <ion-input type="text" formControlName="soyadi"></ion-input>
  </ion-item>
  <ion-item>
    <ion-label position="floating" color="Dark">E-POSTA </ion-label>
    <ion-input type="text" formControlName="e-posta"></ion-input>
  </ion-item>



  <ion-item>
    <ion-label position="floating" color="primary">şifre</ion-label>
    <ion-input type="password" formControlName="Sifre"></ion-input>
  </ion-item>
  <div class="validation-errors">
    <ng-container *ngFor="let validation of validation_messages.password">
      <div class="error-message" *ngIf="validations_form.get('password').hasError(validation.type) && (validations_form.get('password').dirty || validations_form.get('password').touched)">
        {{ validation.message }}
      </div>
    </ng-container>
  </div>
  <ion-button class="submit-
btn" expand="block" type="submit" [disabled]="!validations_form.valid">KAYIT
OL</ion-button>
  <label class="error-message">{{errorMessage}}</label>
  <label class="success-message">{{successMessage}}</label>
</form>
```

## login.page.ts dosyası

```
export class LoginPage implements OnInit {
  validations_form: FormGroup;
  errorMessage: string = '';
  validation_messages = {
    'email': [
       { type: 'required', message: 'Email is required.' },
       { type: 'pattern', message: 'Please enter a valid email.' }
    ],
   'password': [
     { type: 'required', message: 'Password is required.' },
     { type: 'minlength', message: 'Password must be at least 5
  characters long.' }
  ]};
  constructor(
   private authService: AuthService,
   private formBuilder: FormBuilder,
   private router: Router
  ) { }
  ngOnInit() {
    this.validations_form = this.formBuilder.group({
      email: new FormControl('', Validators.compose([
         Validators.required,
         Validators.pattern('^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+.[a-zA-Z0-9-.]+$')
     ])),
       password: new FormControl('',Validators.compose([
         Validators.minLength(5),
         Validators.required
       ])),
    });
  }
  tryLogin(value) {
  this.authService.Login(value).then(res => {     this.router.navigate(["/home"
]);}, err => {
  this.errorMessage = err.message;
  console.log(err)
  })
  }
  goRegisterPage() {
   this.router.navigate(["/register"]);
  }}
```

## NAVİGASYON EKRANI



User ID: KhHGCwPYcfbjW4AhtB6YsHMSN...

Konumu Kaydet

Lat: 37.33233141 Lng: -122.0312186
5/23/19, 2:15 PM

Lat: 37.33233141 Lng: -122.0312186
5/23/19, 2:15 PM

Lat: 37.33233141 Lng: -122.0312186
5/23/19, 2:18 PM

Lat: 37.33233141 Lng: -122.0312186
5/23/19, 2:18 PM

## Home.page.ts dosyası

```
import { Component, ViewChild, ElementRef } from '@angular/core';
import { AngularFireAuth } from '@angular/fire/auth';
import {
  AngularFirestore,
  AngularFirestoreCollection
} from '@angular/fire/firestore';
import { Observable } from 'rxjs';
import { map } from 'rxjs/operators';

import { Plugins } from '@capacitor/core';
const { Geolocation } = Plugins;

declare var google;
```

```
@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss']
})
export class HomePage {
  // Firebase Data
  locations: Observable<any>;
  locationsCollection: AngularFirestoreCollection<any>;

  // Map related
  @ViewChild ('map', {static: false}) mapElement: ElementRef;
  map: any;
  markers = [];

  // Misc
  isTracking = false;
  watch: string;
  user = null;

  constructor(private afAuth: AngularFireAuth, private afs: AngularFirestore)
{
    this.anonLogin();
  }

  ionViewWillEnter() {
    this.loadMap();
  }

  // Perform an anonymous login and load data
  anonLogin() {

    this.afAuth.signInAnonymously().then(res => {
      this.user = res.user;

      this.locationsCollection = this.afs.collection(
        `locations/${this.user.uid}/track`,
        ref => ref.orderBy('timestamp')
      );

      // Make sure we also get the Firebase item ID!
      this.locations = this.locationsCollection.snapshotChanges().pipe(
        map(actions =>
          actions.map(a => {
            const data = a.payload.doc.data();
            const id = a.payload.doc.id;
            return { id, ...data };
          })
        )
      );

      // Update Map marker on every change
      this.locations.subscribe(locations => {
        this.updateMap(locations);
      });
    });
  }
  startTracking() {
    this.isTracking = true;
    this.watch = Geolocation.watchPosition({}, (position, err) => {
      if (position) {
        this.addNewLocation(
          position.coords.latitude,
          position.coords.longitude,
          position.timestamp
        );
      }
    });
  }

  // Unsubscribe from the geolocation watch using the initial ID
  stopTracking() {
    Geolocation.clearWatch({ id: this.watch }).then(() => {
      this.isTracking = false;
    });
  }

  // Save a new location to Firebase and center the map
  addNewLocation(lat, lng, timestamp) {
    this.locationsCollection.add({
      lat,
      lng,
      timestamp
    });

    let position = new google.maps.LatLng(lat, lng);
    this.map.setCenter(position);
    this.map.setZoom(5);
  }

  // Delete a location from Firebase
  deleteLocation(pos) {
    this.locationsCollection.doc(pos.id).delete();
  }

  // Redraw all markers on the map
  updateMap(locations) {
    // Remove all current marker
    this.markers.map(marker => marker.setMap(null));
    this.markers = [];

    for (let loc of locations) {
      let latLng = new google.maps.LatLng(loc.lat, loc.lng);

      let marker = new google.maps.Marker({
        map: this.map,
        animation: google.maps.Animation.DROP,
        position: latLng
      });
      this.markers.push(marker);
    }
  }

  // Initialize a blank map
  loadMap() {
    let latLng = new google.maps.LatLng(51.9036442, 7.6673267);

    let mapOptions = {
      center: latLng,
      zoom: 5,
      mapTypeId: google.maps.MapTypeId.ROADMAP
    };

    this.map = new google.maps.Map(this.mapElement.nativeElement, mapOptions);
  }
}
```

## Auth.service.ts  dosyası

```
import { Injectable } from '@angular/core';


@Injectable({
  providedIn: 'root'
})
export class AuthService {

  constructor() { }
}
```

## Home.page.scss dosyası

```scss
#container {
  text-align: center;

  position: absolute;
  left: 0;
  right: 0;
  top: 50%;
  transform: translateY(-50%);
}

#container strong {
  font-size: 20px;
  line-height: 26px;
}

#container p {
  font-size: 16px;
  line-height: 22px;

  color: #8c8c8c;

  margin: 0;
}

#container a {
  text-decoration: none;
}
#map {
  width: 100%;
  height: 300px;
}
```

## İndex.html  dosyası

```html
<!DOCTYPE html>
<html lang="en">
```

```
<head>
  <meta charset="utf-8" />
  <title>Ionic App</title>

  <base href="/" />

  <meta name="color-scheme" content="light dark" />
  <meta name="viewport" content="viewport-fit=cover, width=device-
width, initial-scale=1.0, minimum-scale=1.0, maximum-scale=1.0, user-
scalable=no" />
  <meta name="format-detection" content="telephone=no" />
  <meta name="msapplication-tap-highlight" content="no" />

  <link rel="icon" type="image/png" href="assets/icon/favicon.png" />

  <!-- add to homescreen for ios -->
  <meta name="apple-mobile-web-app-capable" content="yes" />
  <meta name="apple-mobile-web-app-status-bar-style" content="black" />
  <script src="https://maps.googleapis.com/maps/api/js?key=YOURKEY"></script>
</head>

<body>
  <!-- The core Firebase JS SDK is always required and must be listed first --
>
<script src="https://www.gstatic.com/firebasejs/7.14.2/firebase-
app.js"></script>

<!-- TODO: Add SDKs for Firebase products that you want to use
     https://firebase.google.com/docs/web/setup#available-libraries -->
<script src="https://www.gstatic.com/firebasejs/7.14.2/firebase-
analytics.js"></script>

<script>
  // Your web app's Firebase configuration
  var firebaseConfig = {
    apiKey: "AIzaSyBjfeP3qsYU6lDCHVpLgpSAinto62yOrPs",
    authDomain: "mapsproje-1b253.firebaseapp.com",
    databaseURL: "https://mapsproje-1b253.firebaseio.com",
    projectId: "mapsproje-1b253",
    storageBucket: "mapsproje-1b253.appspot.com",
    messagingSenderId: "746573346602",
    appId: "1:746573346602:web:4fae2b9126e9c21e2c9053",
    measurementId: "G-WWELYQTNE3"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
  firebase.analytics();
</script>
```

```
  <app-root></app-root>
</body>

</html>
```

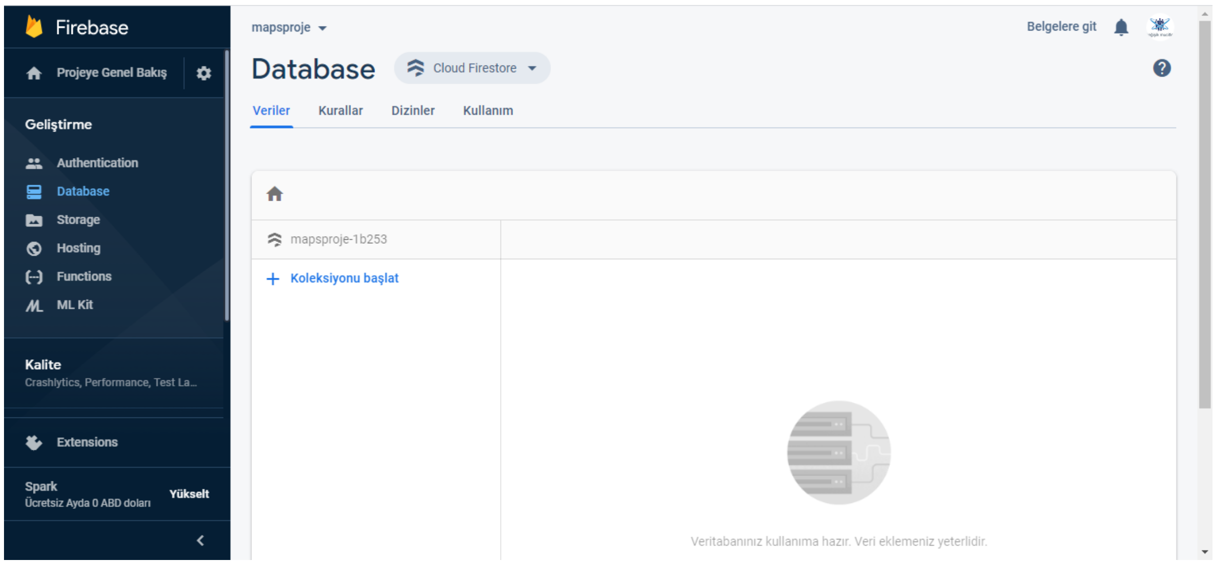
## App.modul.ts dosyası

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { RouteReuseStrategy } from '@angular/router';

import { IonicModule, IonicRouteStrategy } from '@ionic/angular';
import { SplashScreen } from '@ionic-native/splash-screen/ngx';
import { StatusBar } from '@ionic-native/status-bar/ngx';

import { AppComponent } from './app.component';
import { AppRoutingModule } from './app-routing.module';

import { environment } from '../environments/environment';

import { AngularFireModule } from '@angular/fire';
import { AngularFirestoreModule } from '@angular/fire/firestore';
import { AngularFireAuthModule } from '@angular/fire/auth';

@NgModule({
  declarations: [AppComponent],
  entryComponents: [],
  imports: [BrowserModule, IonicModule.forRoot(), AppRoutingModule,
    AngularFireModule.initializeApp(environment.firebase),
    AngularFirestoreModule,
    AngularFireAuthModule],
  providers: [
    StatusBar,
    SplashScreen,
    { provide: RouteReuseStrategy, useClass: IonicRouteStrategy }
  ],
  bootstrap: [AppComponent]
})
export class AppModule {}
```

## home.page.html dosyası

```
<ion-header>
  <ion-toolbar color="primary">
    <ion-title>

    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content>

  <div #map id="map" [hidden]="!user"></div>

  <div *ngIf="user">

    <ion-item>
      <ion-label>User ID: {{ user.uid }}</ion-label>
    </ion-item>

    <ion-button expand="block" (click)="startTracking()" *ngIf="!isTracking">
      <ion-icon name="locate" slot="start"></ion-icon>
      basla kaydet
    </ion-button>

    <ion-button expand="block" (click)="stopTracking()" *ngIf="isTracking">
      <ion-icon name="hand" slot="start"></ion-icon>
      durdur kaydet
    </ion-button>

    <ion-list>
      <ion-item-sliding *ngFor="MK | async">
        <ion-item>
          <ion-label text-wrap>
            Lat: {{ pos.lat }}
            Lng: {{ pos.lng }}
            <p>
              {{ pos.timestamp | date:'short' }}
            </p>
          </ion-label>
        </ion-item>

        <ion-item-options side="start">
          <ion-item-option color="danger" (click)="Konumu Kaydet">
            <ion-icon name="trash" slot="icon-only"></ion-icon>
          </ion-item-option>
        </ion-item-options>

      </ion-item-sliding>
    </ion-list>

  </div>

</ion-content>
```

## VERİ TABANI  FİREBASE



Veritabanınız kullanıma hazır. Veri eklemeniz yeterlidir.