ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
Τμήμα Πληροφορικής

Εργασία Μαθήματος «ΣΥΧΓΧΡΟΝΑ ΘΕΜΑΤΑ ΤΕΧΝΟΛΟΓΙΑΣ ΛΟΓΙΣΜΙΚΟΥ ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΩΝ ΓΙΑ ΚΙΝΗΤΑ ΤΗΛΕΦΩΝΑ»

«αριθμός άσκησης»	**1η εργασία**
Όνομα φοιτητή – Αριθμός Μητρώου	Πετρίδης Αχιλλέας Π18211
Ημερομηνία παράδοσης	12/12/2021

Εκφώνηση της άσκησης

ΕΡΓΑΣΙΑ 1

Καλείστε να αναπτύξετε μια εφαρμογή Android, η οποία θα καταγράφει επιταχύνσεις και επιβραδύνσεις (φρεναρίσματα) από κινούμενα οχήματα. Συγκεκριμένα, κατά την ενεργοποίησή της και με το πάτημα κατάλληλου κουμπιού θα ξεκινάει η καταγραφή για τον κάθε χρήστη εφόσον βρίσκεται εν κινήσει σε κάποιο όχημα (μην καταγράψετε χρήστες που τρέχουν με τα πόδια). Συνολικά, η εφαρμογή σας θα πρέπει να υποστηρίζει τις εξής βασικές λειτουργίες: • Για κάθε γεγονός επιτάχυνσης/φρεναρίσματος θα πρέπει να καταγράφονται (τουλάχιστον) τα εξής: ο Επιτάχυνση/Φρενάρισμα. Και για τα δύο θέλουμε να υπάρχουν τιμές που να υποδηλώνουν έντονη επιτάχυνση και φρενάρισμα (όχι κάτι που έχει γίνει πολύ αργά, σε μεγάλο χρονικό διάστημα). Τις εν λόγω παραμέτρους θα τις βρείτε/υπολογίσετε εσείς ο Προαιρετικά μπορείτε να καταγράφετε και την τιμή της επιτάχυνσης/φρεναρίσματος (για πιθανή μελλοντική περαιτέρω κατηγοριοποίηση) ο Timestamp ο Location ο Ταχύτητα πριν από την καταγραφή του γεγονότος (είχε ταχύτητα XX πριν φρενάρει) ο Ένα μοναδικό ID για τον κάθε χρήστη (όσο ανώνυμο γίνεται, π.χ. μη βάλετε το e-mail του. Μπορεί να είναι ένα οποιοδήποτε autogenerated ID) • Όλα τα παραπάνω δεδομένα θα καταγράφονται στην Firebase • Θα υπάρχει ξεχωριστό Activity που θα διαθέτει την προβολή χάρτη (Google Map) πάνω στον οποίο θα μπορούν να αναπαρασταθούν οι πληροφορίες επιταχύνσεων και φρεναρισμάτων που έχουν καταγραφεί στη ΒΔ (όλων των χρηστών) (είτε ξεχωριστά επιταχύνσεις/φρεναρίσματα, είτε με άλλο χρώμα όλα μαζί, ώστε να είναι διακριτά) Είστε ελεύθεροι να προσθέσετε και όσες ακόμα λειτουργίες θέλετε. Εννοείτε ότι θα αναγκαστείτε να ορίσετε και να χειριστείτε σωστά τα όποια Android Permissions χρειάζονται. Η χρήση της Firebase ως ΒΔ για την αποθήκευση των δεδομένων είναι υποχρεωτική

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1 Επεξήγηση της ροής της εργασίας.

1.1 Συλλογή και αποθήκευση δεδομένων.

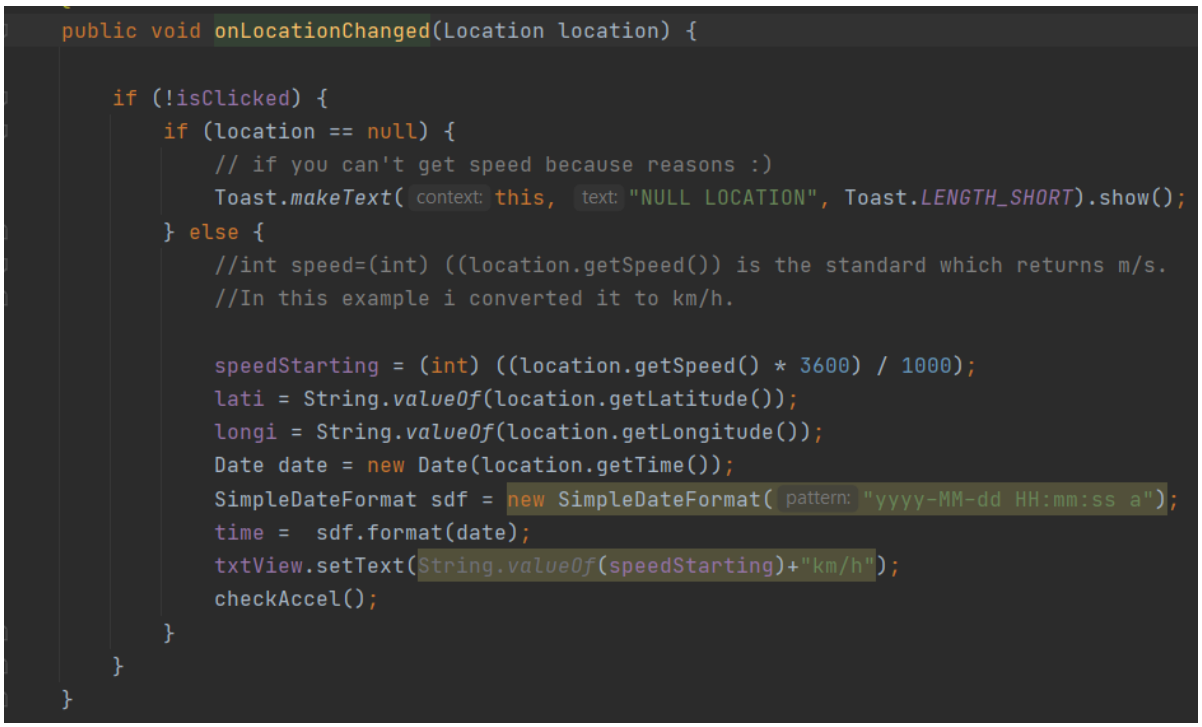
Αρχικά στην μέθοδο onCreate() φτιάχνουμε έναν listener για το κουμπί start.

```
// Lambda Expression for Enable button listener.
startButton.setOnClickListener((View view) -> {

    // if Boolean variable isClicked is true.
    if (isClicked) {
        id++;
        startButton.setText("Stop");

        isClicked = false;
        checkMovement();
        // Toast Message.
        Toast.makeText( context: this,  text: "Speed logging is enabled", Toast.LENGTH_SHORT).show();
    } else {
        //Stop getting Location's updates.
        txtView.setText("-- km/h");
        startButton.setText("Start");
        isClicked = true;
        // Toast Message.
        Toast.makeText( context: this,  text: "Speed logging is disabled", Toast.LENGTH_SHORT).show();
    }
});
```

Έπειτα καλείτε η μέθοδος checkMovement().

```
void checkMovement() {
    LocationManager lm = (LocationManager) this.getSystemService(Context.LOCATION_SERVICE);
    if (ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
            ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
        // TODO: Consider calling
        //    ActivityCompat#requestPermissions
        // here to request the missing permissions, and then overriding
        //   public void onRequestPermissionsResult(int requestCode, String[] permissions,
        //                                          int[] grantResults)
        // to handle the case where the user grants the permission. See the documentation
        // for ActivityCompat#requestPermissions for more details.
        return;
    }
    lm.requestLocationUpdates(LocationManager.GPS_PROVIDER,  minTimeMs: 0,  minDistanceM: 0,  listener: this);
    this.onLocationChanged((Location) null);
}
```

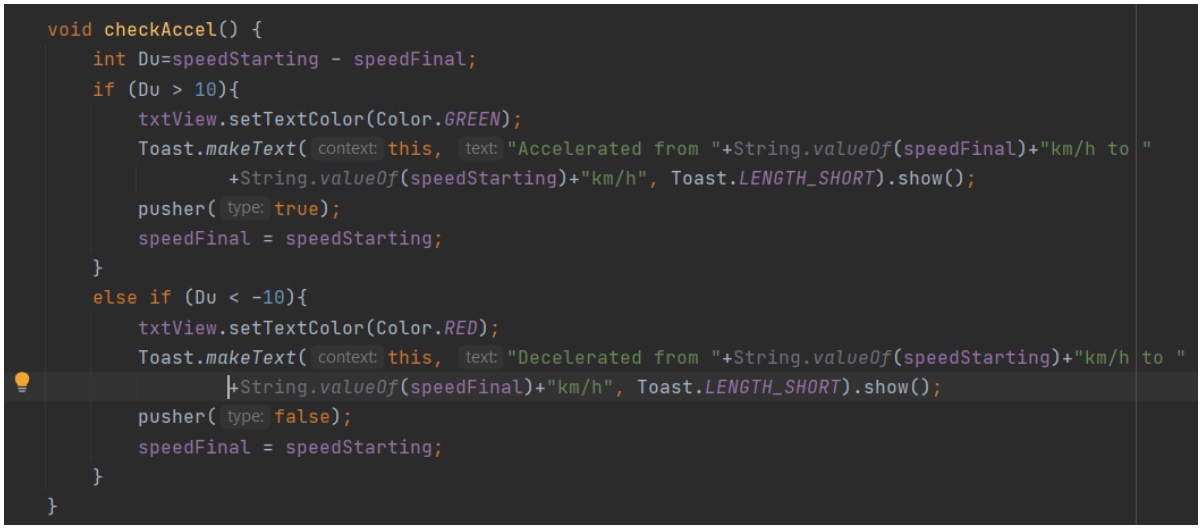
Η οποία ελέγχει αν υπάρχουν τα δικαιώματα για την λήψη τοποθεσίας μέσω δικτύου και μέσω gps και γίνετε trigger η OnLocationChanged().



```
public void onLocationChanged(Location location) {

    if (!isClicked) {
        if (location == null) {
            // if you can't get speed because reasons :)
            Toast.makeText( context: this,  text: "NULL LOCATION", Toast.LENGTH_SHORT).show();
        } else {
            //int speed=(int) ((location.getSpeed()) is the standard which returns m/s.
            //In this example i converted it to km/h.

            speedStarting = (int) ((location.getSpeed() * 3600) / 1000);
            lati = String.valueOf(location.getLatitude());
            longi = String.valueOf(location.getLongitude());
            Date date = new Date(location.getTime());
            SimpleDateFormat sdf = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss a");
            time =  sdf.format(date);
            txtView.setText(String.valueOf(speedStarting)+"km/h");
            checkAccel();
        }
    }
}
```

Για την ολοκλήρωση οποιασδήποτε ενέργειας ελέγχετε αν η τιμή της isClicked είναι false (την παρέχουμε στο listener του κουμπιού start όταν γίνετε το πάτημα).

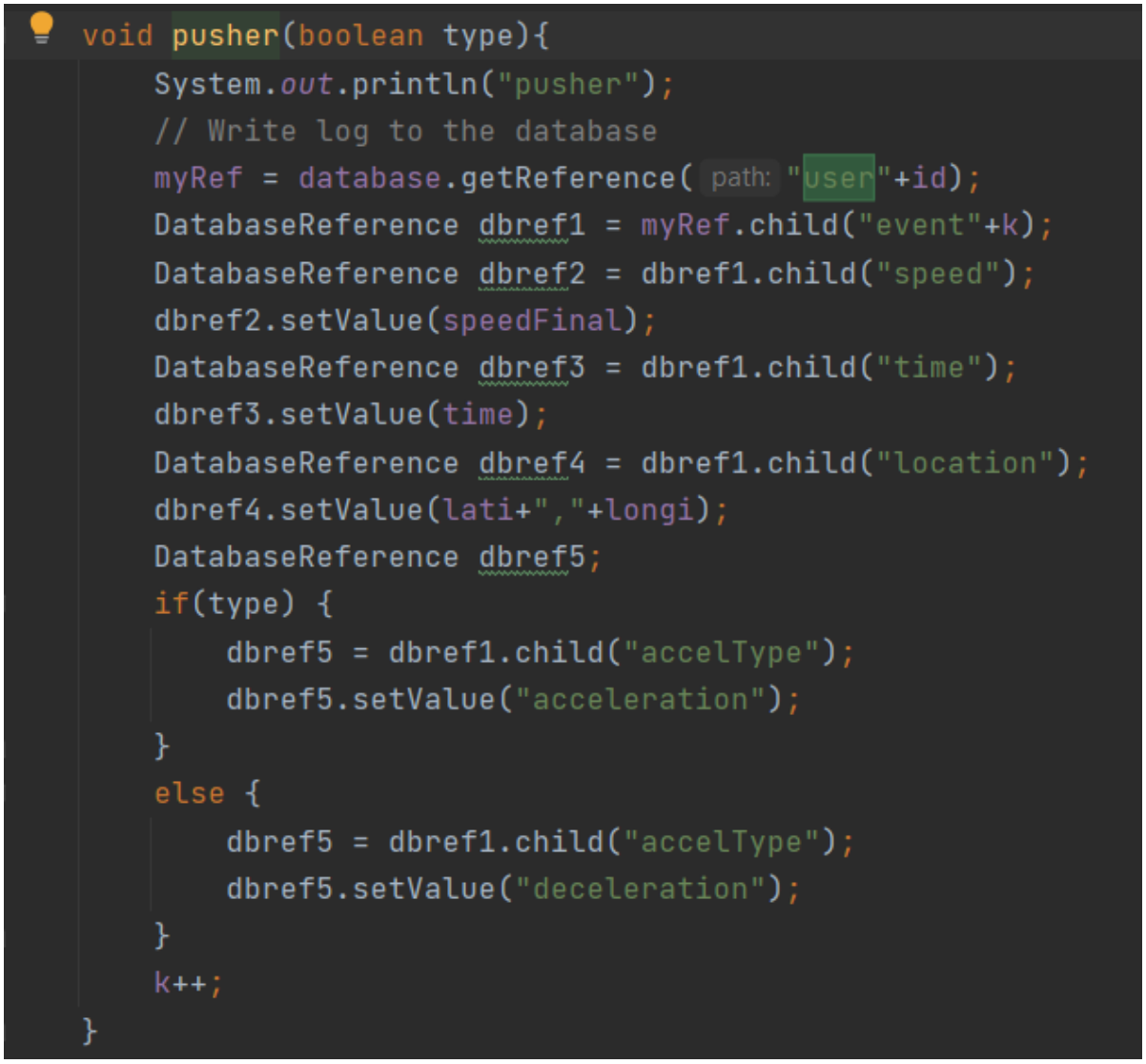
Έπειτα βλέπουμε αν υπάρχει Location , εάν όχι πετάμε στον χρήστη το αντίστοιχο μήνυμα λάθους, και παίρνουμε την ταχύτητα με την βοήθεια της getSpeed. Καταγράφουμε επίσης και το latitude και το longitude καθώς και την ημερομηνία και την ώρα που καταγράψαμε την ταχύτητα .Ύστερα καλούμε την μέθοδο checkAccel().



```
void checkAccel() {
    int Du=speedStarting - speedFinal;
    if (Du > 10){
        txtView.setTextColor(Color.GREEN);
        Toast.makeText( context: this,  text: "Accelerated from "+String.valueOf(speedFinal)+"km/h to "
                +String.valueOf(speedStarting)+"km/h", Toast.LENGTH_SHORT).show();
        pusher( type: true);
        speedFinal = speedStarting;
    }
    else if (Du < -10){
        txtView.setTextColor(Color.RED);
        Toast.makeText( context: this,  text: "Decelerated from "+String.valueOf(speedStarting)+"km/h to "
                +String.valueOf(speedFinal)+"km/h", Toast.LENGTH_SHORT).show();
        pusher( type: false);
        speedFinal = speedStarting;
    }
}
```

Δημιουργούμε την μεταβλητή Du (διαφορά ταχύτητας) όπου σύμφωνα με την τιμή της , μεγαλύτερη ή μικρότερη από 10, βλέπουμε εάν το όχημα επιταχύνει ή επιβραδύνει.

Μόλις καταλάβουμε τον τύπο της επιτάχυνσης του οχήματος καλούμε την μέθοδο pusher().

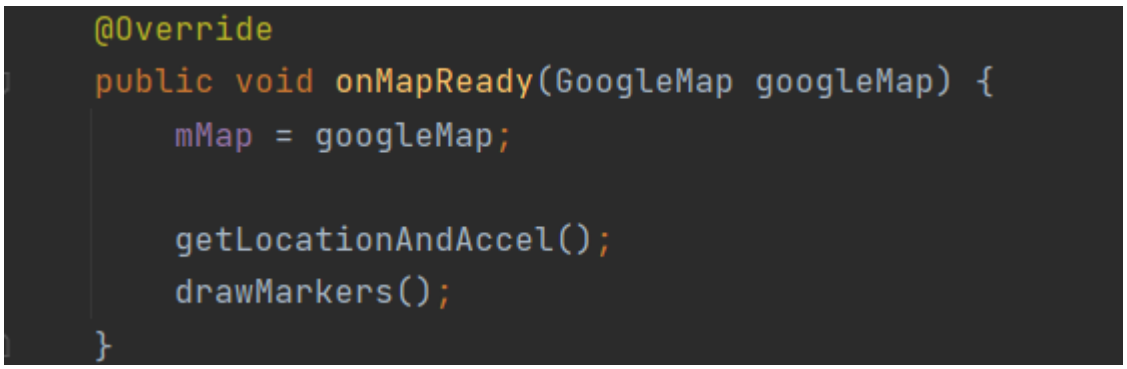


```
void pusher(boolean type){
    System.out.println("pusher");
    // Write log to the database
    myRef = database.getReference( path: "user"+id);
    DatabaseReference dbref1 = myRef.child("event"+k);
    DatabaseReference dbref2 = dbref1.child("speed");
    dbref2.setValue(speedFinal);
    DatabaseReference dbref3 = dbref1.child("time");
    dbref3.setValue(time);
    DatabaseReference dbref4 = dbref1.child("location");
    dbref4.setValue(lati+","+longi);
    DatabaseReference dbref5;
    if(type) {
        dbref5 = dbref1.child("accelType");
        dbref5.setValue("acceleration");
    }
    else {
        dbref5 = dbref1.child("accelType");
        dbref5.setValue("deceleration");
    }
    k++;
}
```

Η μέθοδος pusher είναι υπεύθυνη για το ανέβασμα των δεδομένων στην βάση μας με το μοναδικό id του χρήστη και τα στοιχεία του κάθε συμβάντος (event k).

1.2 Παρουσίαση δεδομένων στον χάρτη.

Δημιουργούμε ένα νέο mapsActivity με το κουμπί map.

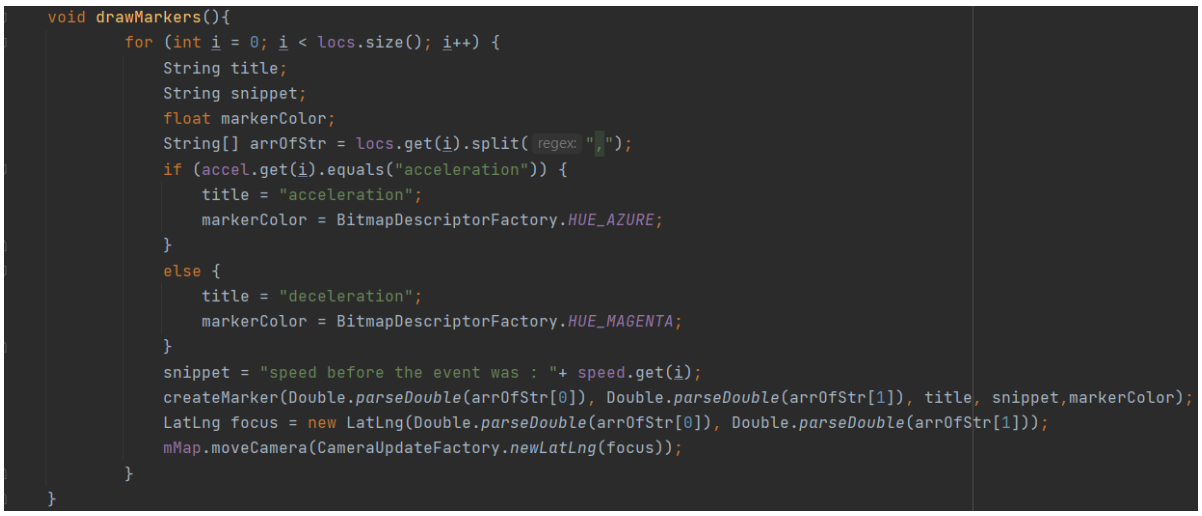


```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    getLocationAndAccel();
    drawMarkers();
}
```

Στην onMapReady καλούμε την getLocationAndAccel() η οποία δημιουργεί ένα arrayList με τα Location των συμβάντων , ένα ακόμα για τον τύπο της επιτάχυνσης και τελικά ένα για την ταχύτητα πριν από το κάθε συμβάν.

```
void getLocationAndAccel() {
    FirebaseDatabase.getInstance("https://gpsaccelerometer-99b4c-default-rtdb.firebaseio.com/").getReference().
            addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            for (DataSnapshot snapshot : dataSnapshot.getChildren()) {
                Object user = snapshot.getValue(Object.class);
                for(DataSnapshot snapshot1 : snapshot.getChildren())
                {
                    String loc = String.valueOf(snapshot1.child("location").getValue());
                    String accl = String.valueOf(snapshot1.child("accelType").getValue());
                    String speedo = String.valueOf(snapshot1.child("speed").getValue());
                    locs.add(loc);
                    accel.add(accl);
                    speed.add(speedo);
                }
                System.out.println(locs);
                drawMarkers();
            }
        }
        @Override
        public void onCancelled(DatabaseError error) {

        }
    });
```

Έπειτα καλούμε την drawMarkers() η οποία χρησιμοποιείται για να προσθέσουμε Markers στο activity μας.



```
void drawMarkers(){
        for (int i = 0; i < locs.size(); i++) {
            String title;
            String snippet;
            float markerColor;
            String[] arrOfStr = locs.get(i).split( regex: ",");
            if (accel.get(i).equals("acceleration")) {
                title = "acceleration";
                markerColor = BitmapDescriptorFactory.HUE_AZURE;
            }
            else {
                title = "deceleration";
                markerColor = BitmapDescriptorFactory.HUE_MAGENTA;
            }
            snippet = "speed before the event was : "+ speed.get(i);
            createMarker(Double.parseDouble(arrOfStr[0]), Double.parseDouble(arrOfStr[1]), title, snippet,markerColor);
            LatLng focus = new LatLng(Double.parseDouble(arrOfStr[0]), Double.parseDouble(arrOfStr[1]));
            mMap.moveCamera(CameraUpdateFactory.newLatLng(focus));
        }
}
```

Ύστερα καλούμε την createMarker() περνώντας τα δεδομένα latitude , longitude , tittle , snipper , markerColor.

Το latitude longitude String location split

Το latitude και το longitude το παίρνουμε από το String location κάνοντας split στο κομμα(,).

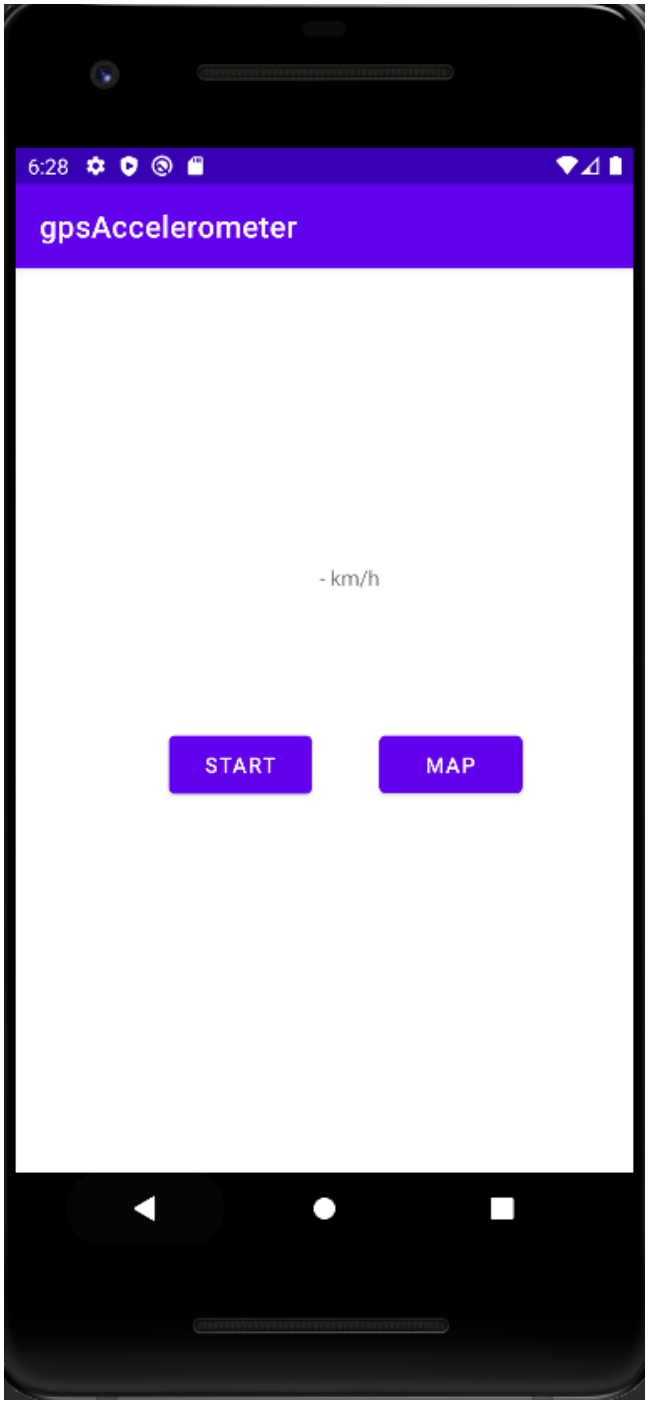
```
protected Marker createMarker(double latitude, double longitude, String title, String snippet,float color) {

    return mMap.addMarker(new MarkerOptions()
            .position(new LatLng(latitude, longitude))
            .anchor( v: 0.5f,  v1: 0.5f)
            .title(title)
            .snippet(snippet)
            .icon(BitmapDescriptorFactory.defaultMarker(color)));
}
```

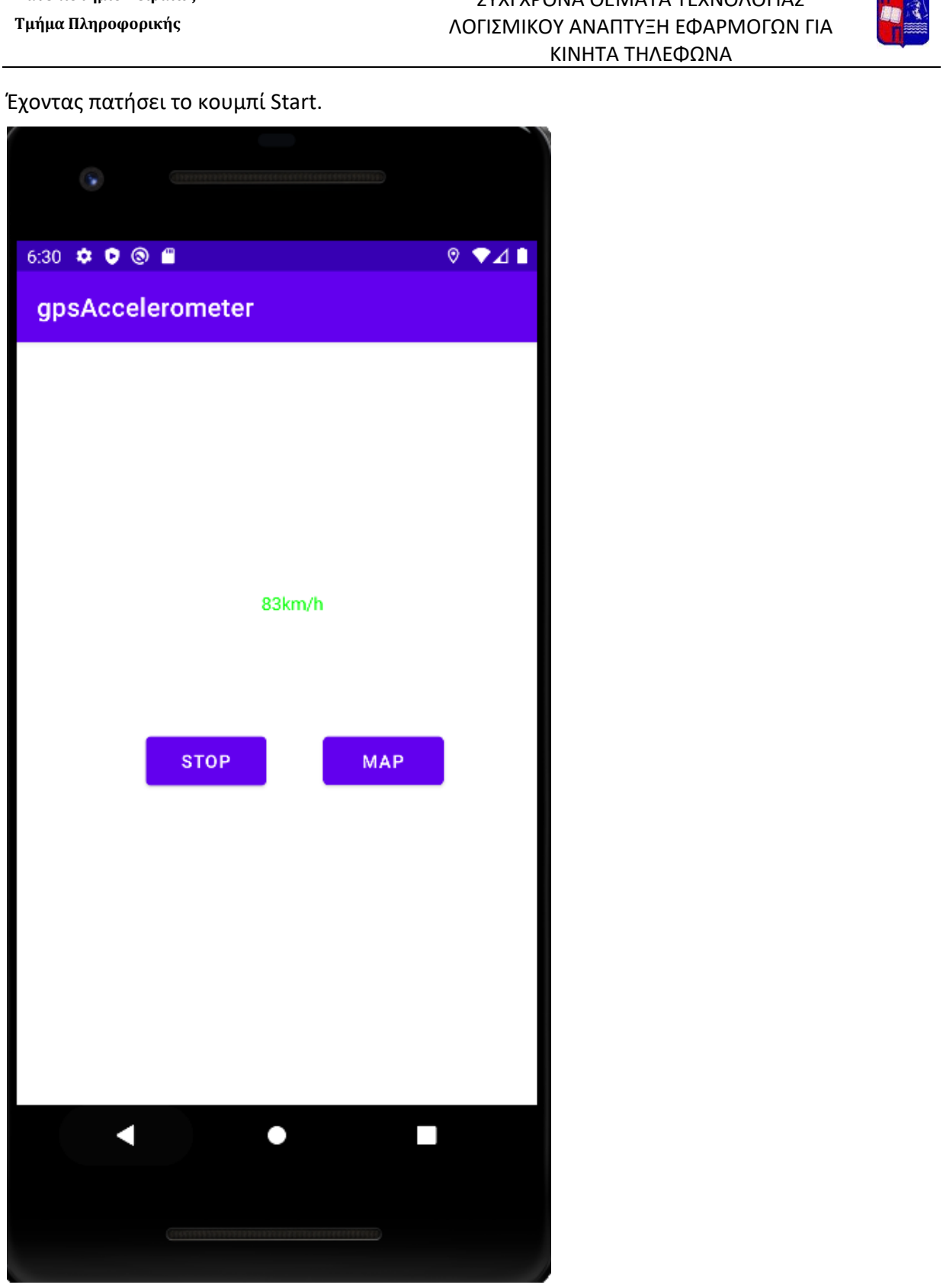
Η createMarker δημιουργεί τα Markers που ζητήσαμε .

2 Συνοπτική επεξήγηση με εικόνες

2.1 Κώδικας της άσκησης

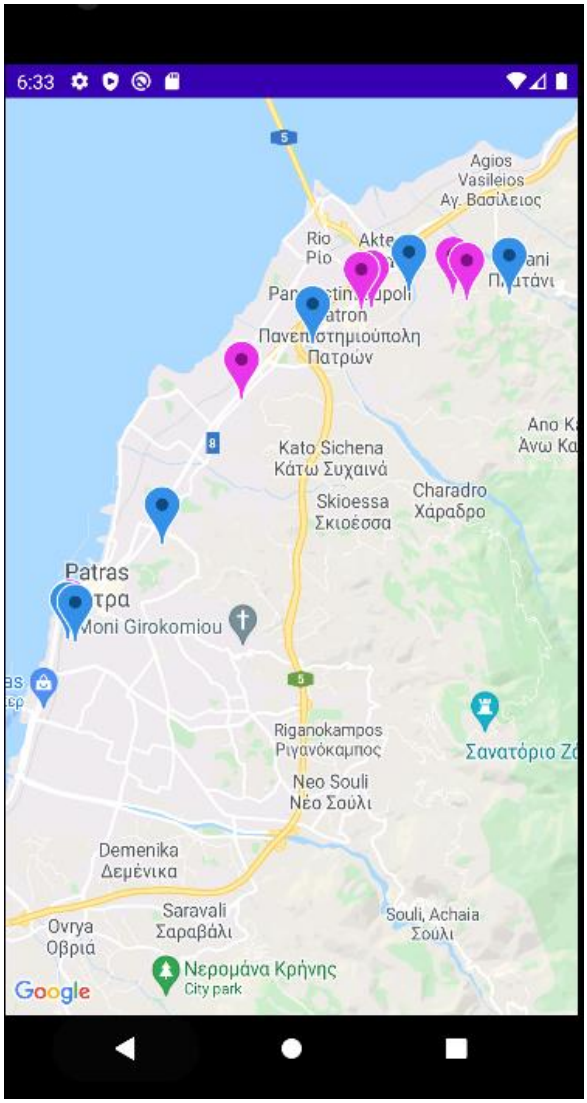
Αρχική της εφαρμογής.





Έχοντας πατήσει το κουμπί Start.

Τελικός έχοντας πατήσει το κουμπί MAP.



Με μπλε μπορούμε να δούμε τα σημεία που έχουμε επιταχύνει και με κόκκινο τα σημεία που έχουμε επιβραδύνει . Επιπροσθέτως αν πατήσουμε πάνω σε κάποιο από τα σημεία μας λέει τις εξής πληροφορίες (τύπος επιτάχυνσης και ταχύτητα επιτάχυνσης πριν από το συμβάν).