List of changes on both: 1) Web API  
 2) Android  
 3) Database

1. Usage and Changes in the Web API (Cloud9 IDE to other android connector)
2. Usage and changes in the Android
3. Usage and the changes in the Database
4. Difficulties along the way
5. How to download and use the system (android requirements and why we did that)
6. User’s Guide

**What had changed since Iteration-1?**

There are three main area to investigate for this project. Web API, Android and database. These three components come together to complete an Android Application in basics. For this project, the changes from Iteration-1 and the complications of those changes will be expleained throughout the report.

**List of Changes:**

1. **Wep API**

We encountered a problem with the Bilkent Transportation Unit, therefore we needed to change our data and while changing our data, we needed to use a different Web API to connect a different kind of database.

1. **Android Studio**

In the first iteration, we haven’t implemented Map and the bus selection functions. In this iteration we have implemented the Map page and the bus selection as a drawable menu as an Android widget. We also have organized the PHP interfaces according to the new tool we are using now, Firebase.

1. **Database**

In the first iteration, we were planning to gather the data from schools transportation database but we had a problem with this data, therefore we have created a function to assign GPS values to the busses in order to track them. The old databases except user database are deleted.

**Changes in the Project in Detail**

There are 2 main differences between Iteration-1 and Iteration-2. First one is the MapActivity java class that contains the Google Maps widget and the drawable menu for a specific bus selection. The other difference is the usage of GPS data. We had a little problem with the Bilkent Transportation unit about gathering the GPS data. The details of this problem will be held in the “Changes in Database” section.

**Changes in Database**

As one of the main difference and the problem, this change affected the whole project in both Android and Web API sides. Our problem was that the GPS data of busses are either incomplete or lost due to driver problems. The GPS devices in busses are manually turned on or off. In addition, Bilkent busses are not owned by the school, they are rental. So some of them serve another school in the morning and in the afternoon or vice versa. That’s why drivers must control their devices each time their shuttle starts and turn the devices off each time they end their work with Bilkent.

In the first iteration we thought that this is a problem we can solve, but appearatnly, it was not. Drivers just do not turn their devices on or off. That is why we decided to delete the GPS database from our project and come up with a function that sends the simulated GPS data to the GoogleMaps object periodically. We have placed the function in MapActivity and made necessary changes so we do not have to use GPS data table in our database. These changes consist of deleting related functions and interface from the Android Studio.

**Changes in Web API**

We also had a little problem with our database connector, PHP unit. The service provider Cloud9 was bought by AWS (Amazon Web Services) and the integrity packages that we use (Redbeans and Slim) are no longer available to Cloud9. So we changed our Web Service provider and moved our project to Firebase. A database that is especially created for Android applications. This system has both database and web service in itself and there is no need for any further usage of external PHP library or source. Many big application today use Firebase, such as Alibaba.com, Trivago, Shazam...

**Changes in Android Studio**

First change here is the deletion and organization of the necessary classes and functions that are related with Cloud9 and GPS data table from our database. After these changes, we have added the MapActivity that contains GoogleMaps object and a drawable menu for selecting a specific bus. The function for getting the GPS data is connected to GoogleMaps object inside the MapActivity class.

**BAHADIR MAP ACTİVİTY ANLAT (NASIL IMPLEMENT ETTIN, UYELIK AÇMA KEY GIRME OLAYLARI FALAN ANLAT ABI DOLDUR BURAYI :D)**

**Difficulties Along the Way**

First of all, from the beginning, it was all new to us. The Android Studio, Cloud9 Web Service and MySQL database management. Alog the way, we have learned all about them and learned how to combine them to create an application with database and all. After first iteration, it was hard to change our web service from Cloud9 to Firebase, since Cloud9 is a service to be written in PHP and Firebase is connected with Java. We spent some time learning 2 different Web APIs.

Another difficulty was dealing with the GPS data problem. Since we could not get the data from school. We had to generate our own location. For this generation, we thought of two ways. Gathering the GPS data by recording the locaiton in each 2-4 seconds while going to Tunus or Sıhhiye, or creating the route from school to Tunus stop, then transforming this path to a graph, then a function, then generate locations by increasing the variants.

Creating the Map activity also had its difficulties. We relied on Google Maps fort he map functionality. However, Google Maps isn’t something one can just import and use or add to xml as a fragment. There had to be several xml’s dedicated to Google Maps info, for example to use Google Maps in any project one has to get a token from Google and add it to a xml dedicated for the token etc.

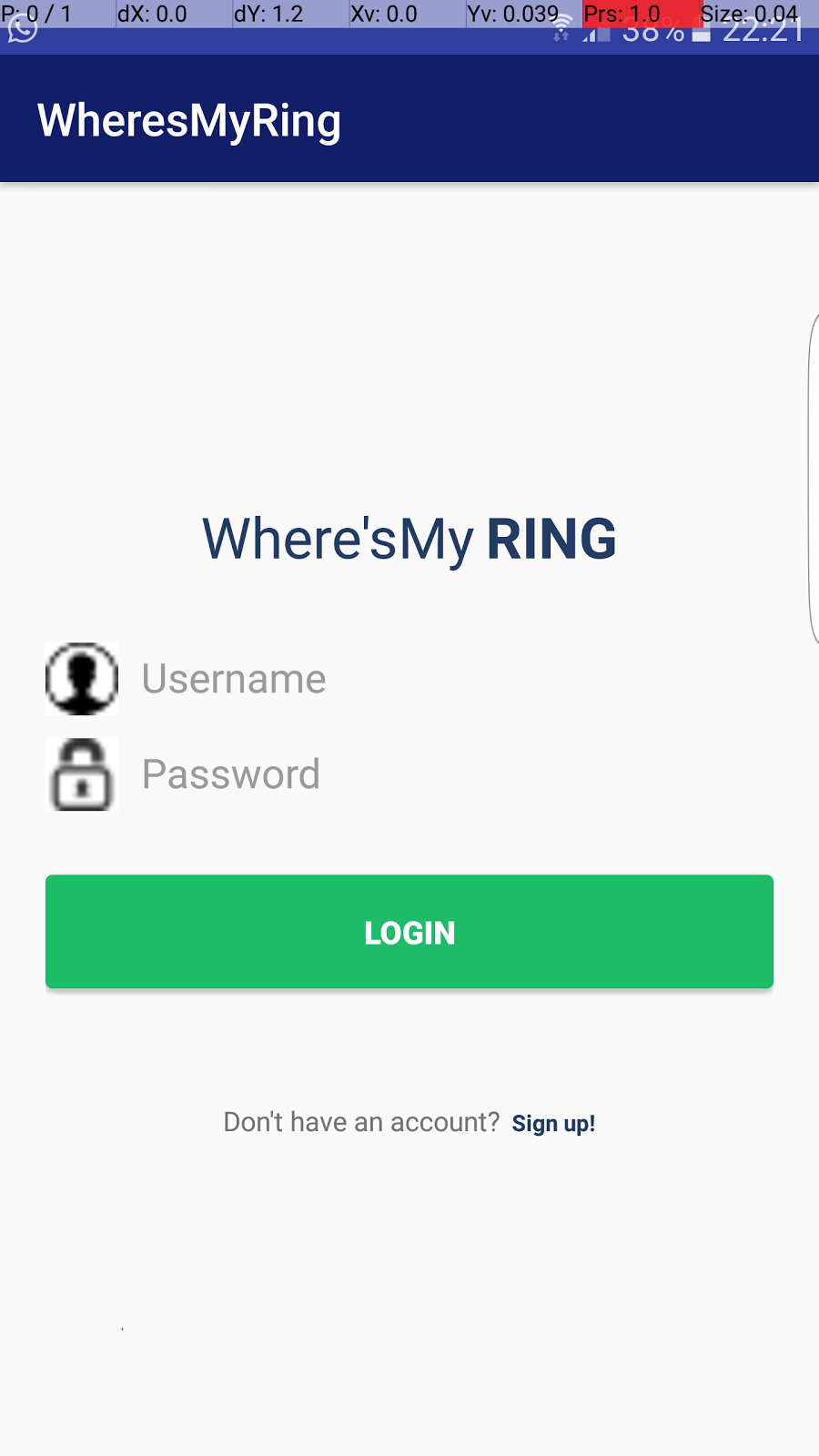
Since Google Maps also means that we will use location, that meant we had to get permissions for Network and Fine Location. However, since Android 5, dangerous permissions can not be only declared in the Manifesto file, they also have to be asked in runtime to the user directly. Thus we had to dedicate a whole utility class just to get the permission and troubleshoot if denied.

There also was the problem of people outside Bilkent being able to use this service and see our rings. We solved the issue by adding a login and signup feature in which when signing up only emails within the bilkent.edu.tr will be accepted. A verification email will be send to the new user to ratify the signup procedure and only after that the user can use the app.

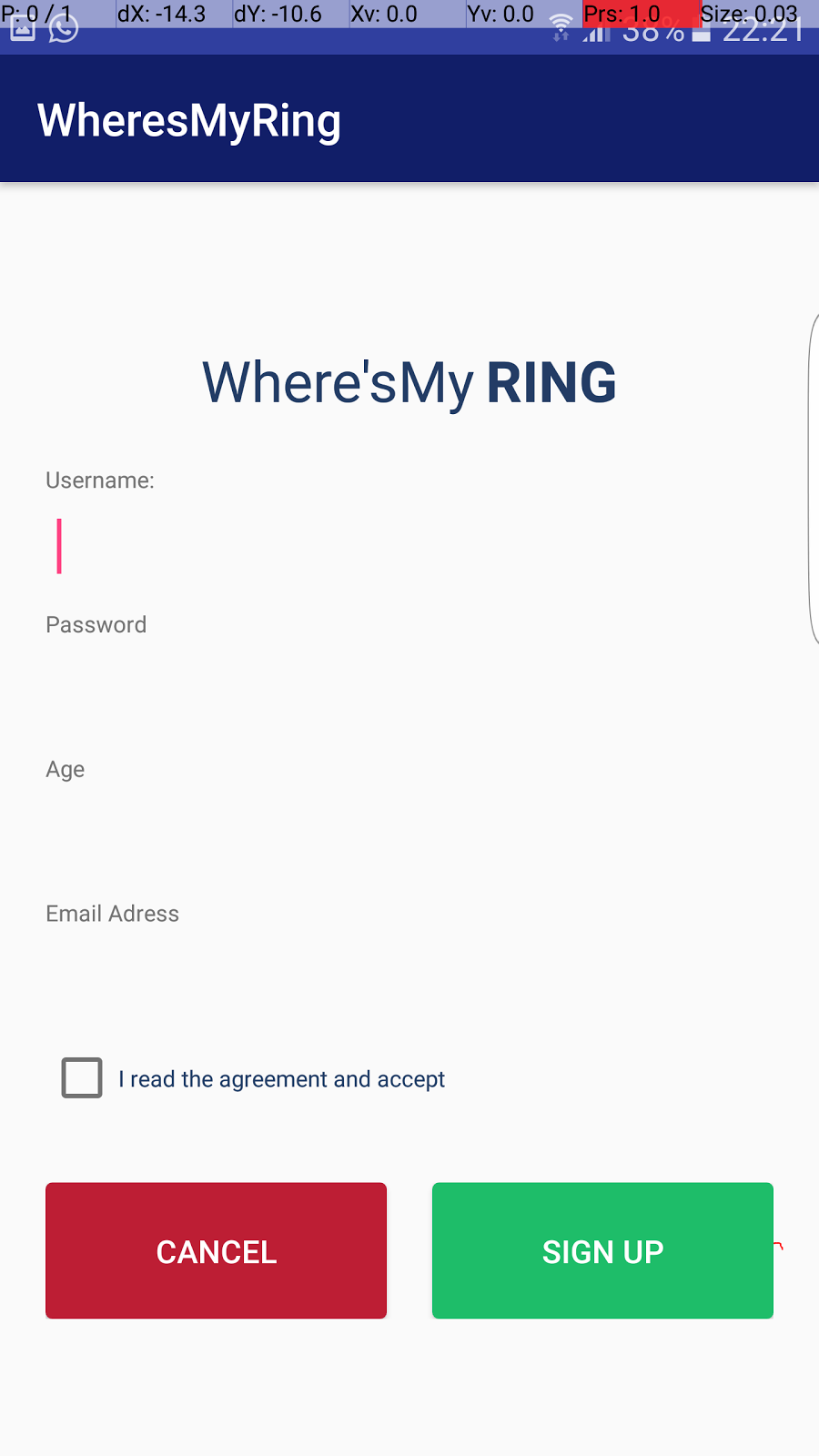
**Completion of the Project**

The project is not fully complete though. Our drawable menu for selectiong busses is not fully ready yet because of the last minute changes we had to make about the web service and database. We have given the description and how to use the drawable menu in the User’s Manual but the implementation of it will be done later.

**User Manual and Interface**



**Login Page:** 1 time activity in which user has to login to verify his/her credentials. After which the login procedure will not be repeated and the user will be directed to Map Page at the start. If the user does not have an account set they will click the Sign Up text to be directed to the signup page.



**Signup page:** One time sign up to make sure user is from Bilkent. The user will be authenticated as a Bilkenter from the email in which if it doesn't belong to the @bilkent.edu.tr domain it will not be accepted. Once signup is completed a verification email will be send and sign up will be completed (not implemented yet). If the user clicks sign up with sufficient info like accept the contract, good password, etc. then the user will be directed to the login page with a toast (short notification in Android) saying that sign up was successful. If they click cancel or back they will return to the login page with a toast notifying that the signup procedure has failed.



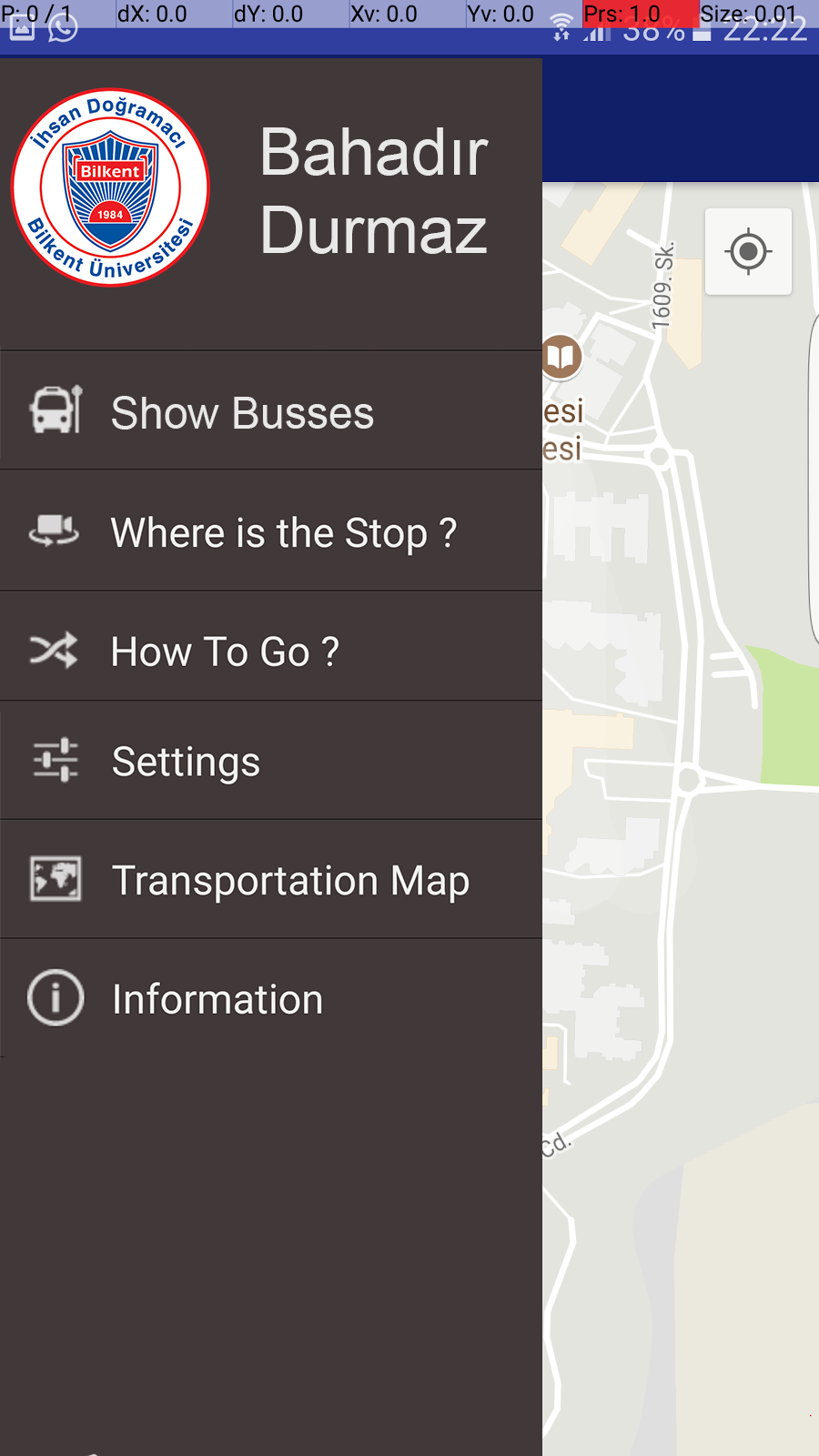
**Map Page:** This page will show all rings according to their whereabouts and the stops on a Google Maps client. After the signup and login procedures this will be the main page user will interact. If the user hasn’t clicked on the location button, we will be shown this location with 3 of the stops, marked. If clicked on them we can see their names:



If the user clicks the location button the camera will zoom to their location and the closest stops will become visible.

Though not yet implemented when finished the closest moving busses of Bilkent will be displayed on the map with the user being able to click on each of them to see info like ETA (we will get this using Google Map’s API), license plate, where it’s heading. User will be able see their location by clicking on the position button.

The 3 parallel lines on the left uppermost corner when clicked will open a navigation bar with options:



**Navigation Bar:** Though we are still in Map Activity, we can see further functionality when we click to the 3 parallel lines on the left uppermost corner in the form of a drawable navigation bar.

When clicked on show busses, all the busses currently in motion will be visible in the map and the camera will zoom in or out to cover all the busses. User can click on an individual bus to get ETA, where it’s heading and general information.

When clicked on how to go, a popup will allow the user to enter where they want to go and map will calculate the route.

When clicked on Settings, settings page will open for general personalization settings, etc…

When clicked on transportation map; user location, all the busses and stops will be visible and the camera will zoom out accordingly.

When clicked on information, a popup will appear with info on the app.