Developing an Android-Based Mobile App to Solve Transportation Issues

https://doi.org/10.3991/ijim.v16i21.36063

Ramiz Salama(✉), Azhar Abdul Jabbar
Department of Computer Engineering, Near East University, Nicosia, Turkey
ramiz.salama@neu.edu.tr

Abstract—Transportation is extremely important in society's day-to-day operations. A major portion of the working class in most cities go to work and school regularly. Commuting can be used for a variety of reasons, including relaxation, dining out, and other activities. Land transportation, which includes the use of both individual and commercial cars, is the most frequent mode of transportation in North Cyprus, particularly in emerging cities... Commercial transportation, on the other hand, has a significant cost benefit over having a personal vehicle. Taxis are becoming increasingly popular, particularly in the cities of Nicosia, Famagusta, and Karenina. Before, the institutions providing such services were unable to reach as many people as they would have liked, particularly in rural areas. Given the widespread use of smartphones in this area, there is an urgency to reach more customers while taking care of the existing ones. Some businesses in North Cyprus have implemented mobile applications to be used by their consumers, but North Cab Taxi is still a work in progress. Instead, the company uses a Web-based cab reservation system. North Cab has realized the necessity for a smartphone version of its cab booking service for it to keep a competitive step above its competition.

Keywords—android system, web server, mobile development, transportation problems, save time

1 Introduction

Transportation plays a vital role in the day-to-day activities of society. In most communities, a large fraction of the working population commutes to work and study daily [1-5]. Commuting May not only be for business purposes but also for relaxation, shopping, and other social activities. Of all the means of transportation, land transport, comprising of the use of vehicles – both private and commercial [6-10].

A key advantage of commercial transportation over owning a personal vehicle is that it is less expensive [11-15]. Taxi services are becoming prominent. Before this time, the companies offering these services have not been able to reach out to as many people as they would have desired especially in remote locations. However, given the widespread adoption of mobile devices (particularly smartphones) in this region, there is a pressing need to reach out to more customers and also better cater to existing ones [16-20].

Some of the companies have already deployed mobile applications to better serve their customers. But some companies are yet to successfully deploy a mobile app to support their services. Instead, the company relies on its Web-based taxi booking system. To maintain a competitive edge with their competitors, Some Companies have recognized the need for a mobile version of their taxi booking service [21-24].

The goal of this research is the creation of a smartphone application for the Development of an Android-based mobile app for solving the problems of transportation.

2 Building the application

This section presents the process of building the proposed technology. The details are outlined in the ensuing steps.

→ Open android studio
→ I entered my project details & my planned domain name
→ Then I went to console firebase.google.com to start a new project there.
→ I chose com.rentin.northcab

On the left side, select the project tab
→ North Cab>app>src>main>java> com.rentin.northcab >mainactivity
→ Then select the gradle tab from the right side of the android studio and wait for it to load.
→ Northcab>tasks>android>signing report.
→ Then under the command tab, copy SHA1

Figure 1 exhibits the output from the codes entered up to this point.

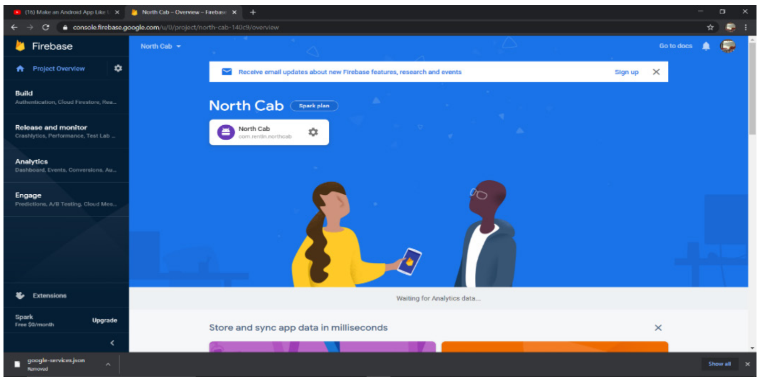
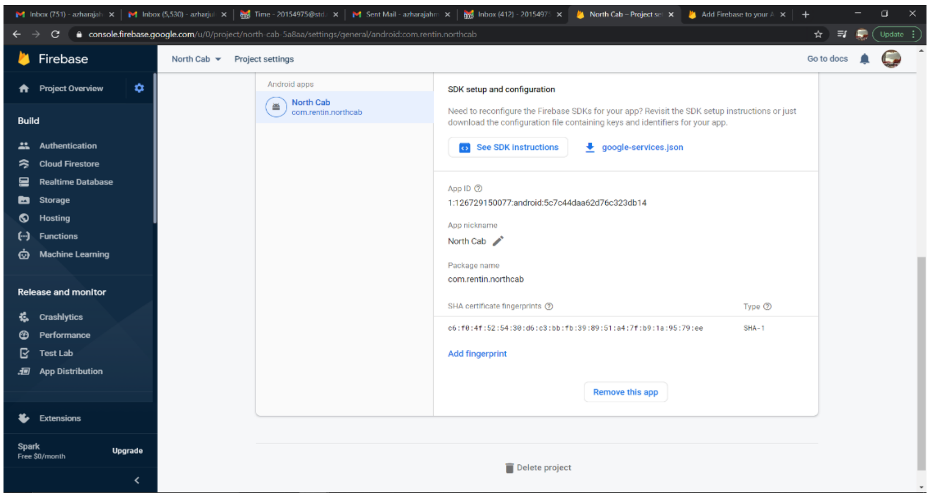

(16) Make an Android App Like U
North Cab – Overview – Firebase
console.firebase.google.com/u/0/project/north-cab-140c9/overview
Firebase
Project Overview
Build
Authentication, Cloud Firestore, Rea...
Release and monitor
Crashlytics, Performance, Test Lab ...
Analytics
Dashboard, Events, Conversions, Au...
Engage
Predictions, A/B Testing, Cloud Mes...
Extensions
Spark
Free $0/month
Upgrade
North Cab
Go to docs
Receive email updates about new Firebase features, research and events
Sign up
North Cab
Spark plan
North Cab
com.rentin.northcab
Waiting for Analytics data...
Store and sync app data in milliseconds
google-services.json
Removed
Show all

Fig. 1. Configuration of firebase

2.1 Adding firebase to my android application

To add firebase to the android application, the following commands were entered.
→ Android package name:- com.rentin.northcab
→ optional nickname:- North Cab.
SHA1 :- C6:FO:4F:52:54:30:D6:C3:BF:FB:39:89:51:A4:7F:B9:1A:95:79:EE
→ Download google services json and move it to the Android Studio>Project > Northcab>app
→ Go to android>gradle script>build.gradle(module app & project)
→ Go under dependancies and create a new classpath
→ Follow the instructions in firebase
→ I used Northcab140C9 project
→com.rentin.northcab
After entering these codes, there is a need to restart the app to prevent any errors. Figure 2 exhibits the output after entering the commands up to this stage.


SDK setup and configuration

Need to reconfigure the Firebase SDKs for your app? Revisit the SDK setup instructions or just download the configuration file containing keys and identifiers for your app.

App ID
1:126729150077:android:5c7c44daa62d76c323db14

App nickname
North Cab

Package name
com.rentin.northcab

SHA certificate fingerprints	Type
c6:f0:4f:52:54:30:d6:c3:bb:fb:39:89:51:a4:7f:b9:1a:95:79:ee	SHA-1

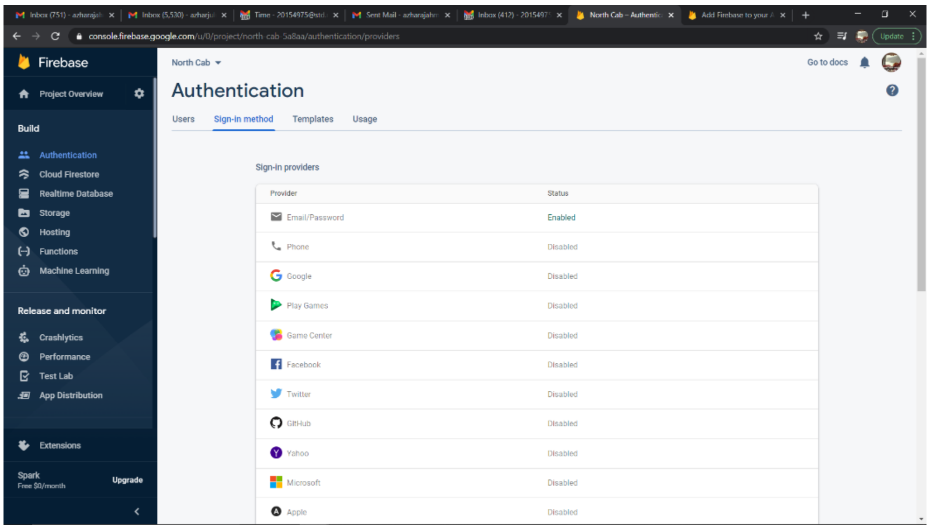
Fig. 2. SDK setup of application

2.2 User login and registration

For login and registration, the following codes were entered.

Database usage.
→ Go to project overview
Build > Authentication > sign-in method.
→ I just used email and password for now.
→ go to the Real-time Database tab
https://north-cab-5a8aa-defant-rtdb.firebese.io.com
→ go to Data and Click on
north-cab-

default-rtdb:null
→ Name; Users, click the '+' button to
add one more for the Drivers.
→ Press add with value- True
→ Add another user as a customer and set the value to true
→ This is to authenticate the id of the users otherwise we wouldn't be able to know if they are drivers or customers.



console.firebase.google.com/u/0/project/north-cab-5a8aa/authentication/providers

Authentication

Sign-in providers

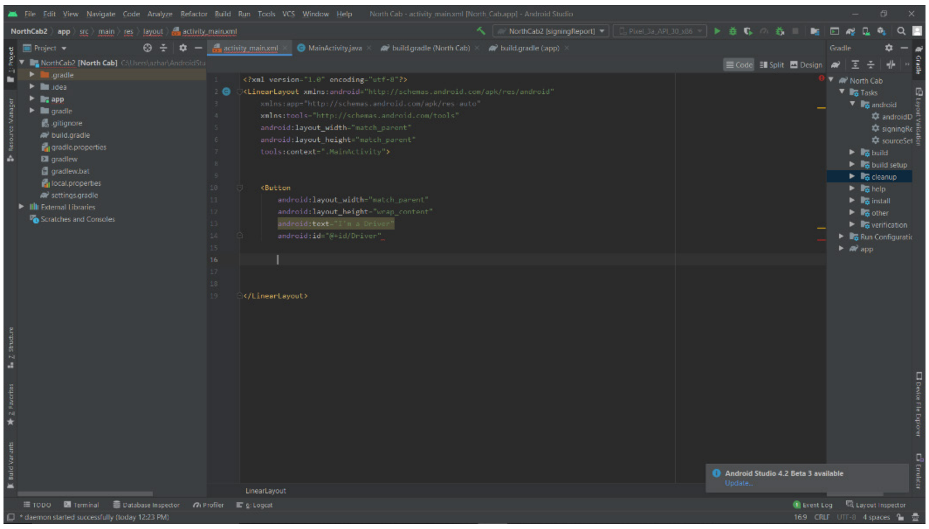
Provider	Status
Email/Password	Enabled
Phone	Disabled
Google	Disabled
Play Games	Disabled
Game Center	Disabled
Facebook	Disabled
Twitter	Disabled
GitHub	Disabled
Yahoo	Disabled
Microsoft	Disabled
Apple	Disabled

Fig. 3. Authentication and sign-in

Now in Android Studio
→ Switch from sign-in report to app. Then go to the main activity tab.
→ go to activity-main.xml
→ Scroll to the right and these tabs will appear "code" "Split" "Design". Go to code and in line 2, change the constraint layout to a Linear layout.
→ change in tools: context="com.rentin.north.cab.mainactivity"

```
<Button
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Im a Driver"
        android:id = "@+id/Driver"/>

<Button
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Im a Customer"
        android:id = "@+id/Customer"/>
```

After these codes were entered, the code for the main activity was set. Figure 4 presents the output for the main activity code.



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">


    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="I'm a Driver"
        android:id="@+id/Driver"



</LinearLayout>
```

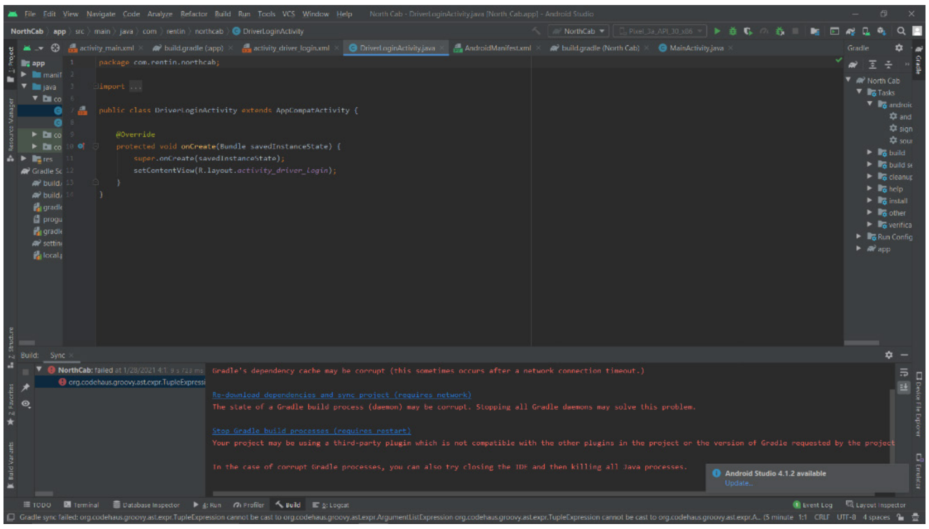
Fig. 4. Setting up code in main activity

Now switch to the "Main activity, java" and code the buttons we created in the activity here under the public class.

```
                           private button mDriver,mCustomer;
  → under the context view
                          mDriver =
(Button)findviewbyld(R.id.driver);
                          mCustomer =
(Button)findviewbyld(R.id.Customer);

  → Under click listener
```

```
mDriver.setonClickListener(newView.Ch.Clicklistener)
                           {

                           @overide
                           public void on Clickch(View V) {
                                           Intent intent = new
intent

(mainActivity.this,DriverLoginActivity.class);

                           start Activity(intent);
                           finish();
                           return;
```

Figure 5 shows the output for setting up the driver login activity code.



```
package com.rentin.northcab;

import ...

public class DriverLoginActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_driver_login);
    }
}
```

```
NorthCab: failed at 1/28/2021 4:1 9 s 723 ms
  org.codehaus.groovy.ast.expr.TupleExpressi

Gradle's dependency cache may be corrupt (this sometimes occurs after a network connection timeout.)

Re-download dependencies and sync project (requires network)
The state of a Gradle build process (daemon) may be corrupt. Stopping all Gradle daemons may solve this problem.

Stop Gradle build processes (requires restart)
Your project may be using a third-party plugin which is not compatible with the other plugins in the project or the version of Gradle requested by the project

In the case of corrupt Gradle processes, you can also try closing the IDE and then killing all Java processes.

Gradle sync failed: org.codehaus.groovy.ast.expr.TupleExpression cannot be cast to org.codehaus.groovy.ast.expr.ArgumentListExpression org.codehaus.groovy.ast.expr.TupleExpression cannot be cast to org.codehaus.groovy.ast.expr.A... (5 minute
```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
North Cab - DriverLoginActivity.java [North_Cab.app] - Android Studio
NorthCab app src main java com rentin northcab DriverLoginActivity
activity_main.xml build.gradle (app) activity_driver_login.xml DriverLoginActivity.java AndroidManifest.xml build.gradle (North Cab) MainActivity.java
Android Studio 4.1.2 available
Update...
TODO Terminal Database Inspector 4: Run Profiler Build 6: Logcat Event Log Layout Inspector

Fig. 5. Setting up driver login activity

→ Scroll to left most tab and go to app>java>northcab

Left click new activity and select an empty activity and rename the package name to “Driver login activity” & Click finish.

*I am going to create everything first for the Driver and then transfer it to the customer because they are almost similar.

→ Go to the activity_driver_login.xml tab and see the second line and change the layout from constraint to linear and then

Below the tools context line, we will create two <Edittext for the email and password

```
<Edittext
      android:layout_width = "match_parent"
      android:layout_height = "wrap_content"
      android:text = "Email"
      android:id = "@+id/Email"/>

<Edittext
      android:layout_width = "match_parent"
      android:layout_height = "wrap_content"
      android:text = "Password"
      android:id = "@+id/Password"/>

<Button
      android:layout_width = "match_parent"
      android:layout_height = "wrap_content"
      android:text = "Login"
      android:id = "@+id/Login"/>

<Button
      android:layout_width = "match_parent"
      android:layout_height = "wrap_content"
      android:text = "Registration"
      android:id = "@+id/Registration"/>
```

Go to DriverLoginActivity.java

Under public class….

```
private Edittext mEmail,mPassword;
            private button mLogin,mRegistration;
```

→Go to MainActivity.java

```
mEmail = (Edittext) findviewbyld(R.id.Email);
mPassword = (Edittext) findviewbyld(R.id.Password);

mLogin = (Edittext) findviewbyld(R.id.Login);
mRegistration = (Edittext) findviewbyld(R.id.Registration);
```

→I then made sure all the libraries from the firebase documentation are added to my build gradle: app tab as follows;
Implementation platform (‘com.google.firebase:firebase-bom:26.3.0’)
Implementation ‘com.google.firebase:firebase-auth:20.0.2’
Implementation ‘com.google.firebase:firebase-database:19.6.0’

```
→Now under set contextview

    mAuth=FirebaseAuth.getInstance();

    firebaseAuthlistener = new FirebaseAuth.AuthstateListener() {


    @override

     Public void onAuthStateChanged(@NonNull FirebaseAuth) {




→ //import the firebaseUser class here

      FirebaseUser user= FirebaseAuth.getInstance().getCurrentUser;

//This variable above Stores Information of the current Logged in Users

     If (user!=null) {

     Intent intent = new intent
            (DriverLoginActivity.this,.class);
```

//The class area is empty because we did not create that class yet

```
                  start Activity(intent);
                  finish();
                  return;
}
```

*If the is not null, the AuthStateListener is called upon when the user logs out.
So we have to make sure that the user is logged in to move forward.

→ Now we are going to create Actual Buttons for Login & Registration
→ Go to DriverLoginActivity.java, under everything, add

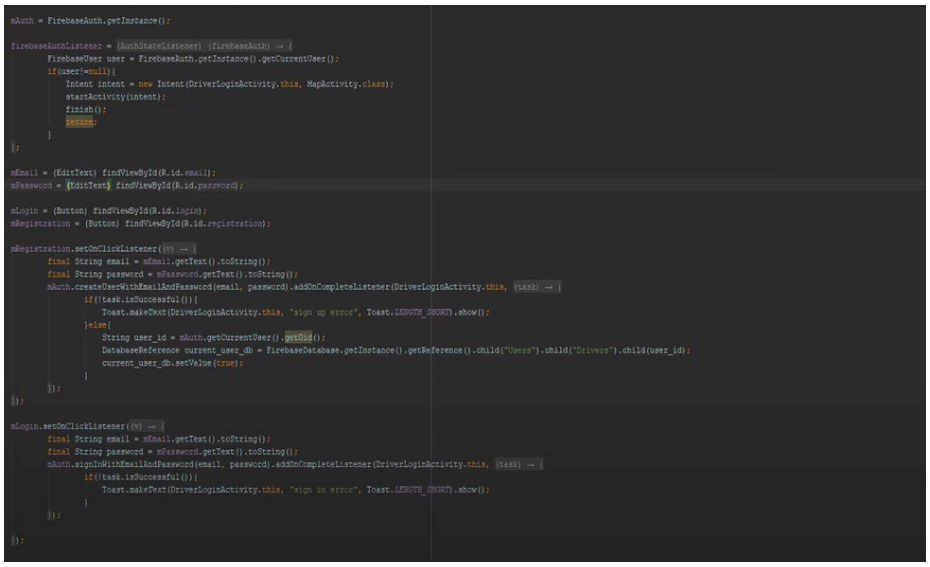
```
          mRegistration.setOnClickListener(new
View.onClickListener() {
                      @override
                       public void onClick(View V) {
                                  final String Email =
mEmail .getText().toString();
                                  final String Password =
mPassword .getText().toString();
```

→ Now I am going to create a user and for that, I will use mAuth.
→ In DriverLoginActivity.java, under final add

```
  mAuth.createUserwithEmailandPassword(Email,Password)
.addonCompleteListener(DriverLoginActivity.this,new
onCompleteListener
  <Auth Result> () {
                   @override
                   public void onComplete (@NonNull Task
<Auth Result>task) {
                             if(!task.isSuccessful()) {

Toast.makeText(DriverLoginActivity.this , “sign up error”
,
                                 Toast. LENGTH-SHORT)
.show();
  }
                             else {


  String user-id = mAuth.getcurrentUser().get Uid();

  DatabaseReference curret-user-db =
FirebaseDatabase.getInstance()
  .getReference() .child(“Users”) .child(“Drivers”)
.child(user-id);

  Current-user-db .setValue(true);



  }
```

→ if this is successful, the user will appear in authentication in firebase
→ I also had to save some information in the database. I had to put the id of the users(Drivers) in the drivers' child in firebase

Figure 6 presents the output of the code for setting up the Login Credentials page in the Driver Section.



```java
mAuth = FirebaseAuth.getInstance();

firebaseAuthListener = (AuthStateListener) (firebaseAuth) → {
        FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
        if(user!=null){
            Intent intent = new Intent(DriverLoginActivity.this, MapActivity.class);
            startActivity(intent);
            finish();
            return;
        }
};

mEmail = (EditText) findViewById(R.id.email);
mPassword = (EditText) findViewById(R.id.password);

mLogin = (Button) findViewById(R.id.login);
mRegistration = (Button) findViewById(R.id.registration);

mRegistration.setOnClickListener((v) → {
        final String email = mEmail.getText().toString();
        final String password = mPassword.getText().toString();
        mAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(DriverLoginActivity.this, (task) → {
                if(!task.isSuccessful()){
                    Toast.makeText(DriverLoginActivity.this, "sign up error", Toast.LENGTH_SHORT).show();
                }else{
                    String user_id = mAuth.getCurrentUser().getUid();
                    DatabaseReference current_user_db = FirebaseDatabase.getInstance().getReference().child("Users").child("Drivers").child(user_id);
                    current_user_db.setValue(true);
                }
        });
});

mLogin.setOnClickListener((v) → {
        final String email = mEmail.getText().toString();
        final String password = mPassword.getText().toString();
        mAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(DriverLoginActivity.this, (task) → {
                if(!task.isSuccessful()){
                    Toast.makeText(DriverLoginActivity.this, "sign in error", Toast.LENGTH_SHORT).show();
                }
        });

});
```

Fig. 6. Setting up login credentials page in driver section

→ Add

```
    mLogin.setOnClickListener(new View.onClickListener() {
           @override
           public void onClick(View V) {
                  final String Email = mEmail .getText().toString();
                  final String Password = mPassword .getText().toString();
                  mAuth.signinwithEmailandPassword(Email,Password);

.addonCompleteListener(DriverLoginActivity.this,new onCompleteListener
<Auth Result> () {
         @override
         public void onComplete (@NonNull Task <Auth Result>task) {
                if(!task.isSuccessful()) {
                  Toast.makeText(DriverLoginActivity.this , "sign in error" ,
                  Toast. LENGTH-SHORT) .show();
```

→ Now under Everything add,

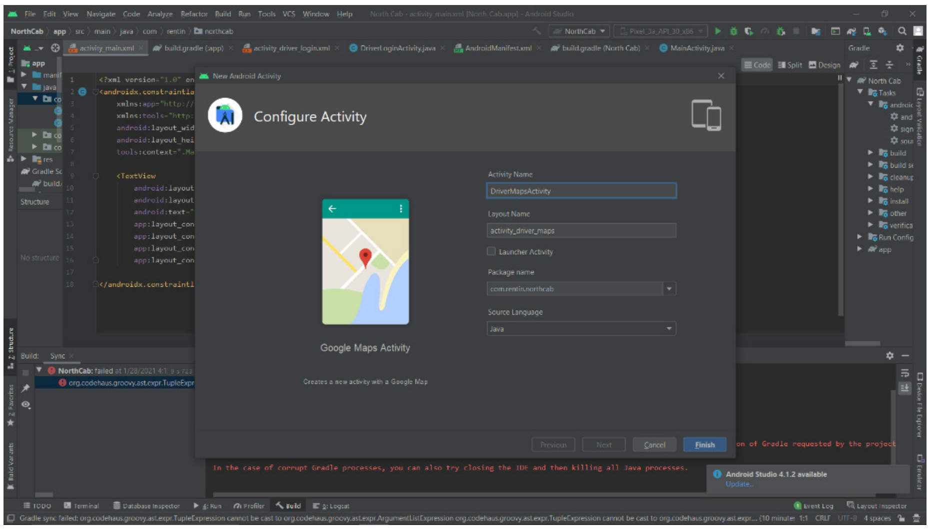
```
                @override
                Protected void onStart() {
                         Super.on Start();
                         mAuth.add AuthStateListener(FirebaseAuthListener);
                }
```

```
                              @override
                             Protected void onStop() {
                                         Super.on
Stop();
                                         mAuth.remove
AuthStateListener(FirebaseAuthListener);
                             }
```

→ Now on the left under Android>app>java>com.rentin.northcab, Create a new activity by right-clicking there.New>Activity>Empty Activity after we move on till the login and registration is complete.

→ I named it Map Activity, for now, I will change it later

Figure 7 presents the output for the code entered for Configuration of Map Activity.



Activity Name: DriverMapsActivity
Layout Name: activity_driver_maps
Launcher Activity: ☐
Package name: com.rentin.northcab
Source Language: Java

```
In the case of corrupt Gradle processes, you can also try closing the IDE and then killing all Java processes.
```

Fig. 7. Configuration of map activity

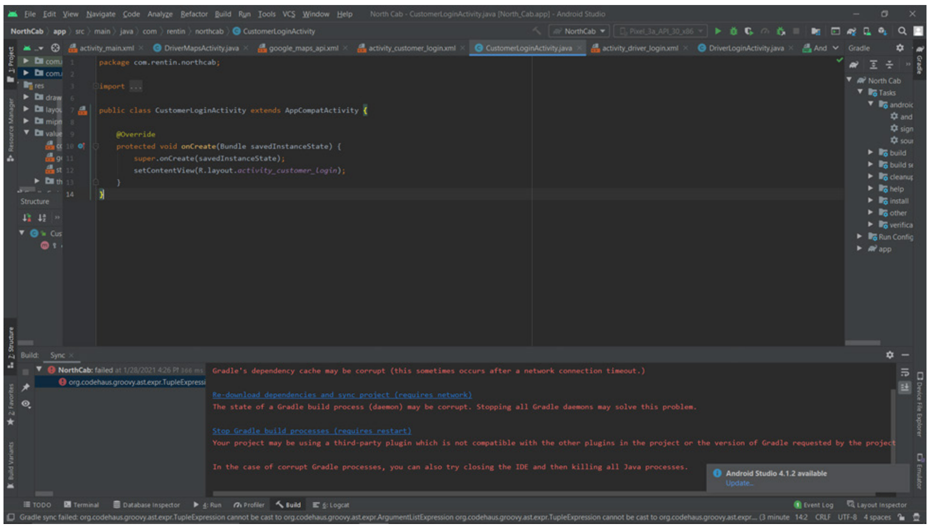
→ Now in DriverLoginActivity.java, in firebaseAuthListener in Intent line, change it to

Intent intent = new Intent(DriverLoginActivity.this,MapActivity.class);

→ Now Create another Empty Activity, this time for customers/passengers. We will name it CustomerLoginActivity

→ Go to the activity customer login.xml tab and see the second line and change the layout from constraint to linear and then below the tool's context line, we will create two <Edittext for the email and password

Figure 8 presents the output for the code for Configuration of Customer Login Activity.



```java
package com.rentin.northcab;

import ...

public class CustomerLoginActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_customer_login);
    }
}
```

```
NorthCab: failed at 1/28/2021 4:26 PM 366 ms
  org.codehaus.groovy.ast.expr.TupleExpressi

Gradle's dependency cache may be corrupt (this sometimes occurs after a network connection timeout.)

Re-download dependencies and sync project (requires network)
The state of a Gradle build process (daemon) may be corrupt. Stopping all Gradle daemons may solve this problem.

Stop Gradle build processes (requires restart)
Your project may be using a third-party plugin which is not compatible with the other plugins in the project or the version of Gradle requested by the project

In the case of corrupt Gradle processes, you can also try closing the IDE and then killing all Java processes.

Android Studio 4.1.2 available
Update...

Gradle sync failed: org.codehaus.groovy.ast.expr.TupleExpression cannot be cast to org.codehaus.groovy.ast.expr.ArgumentListExpression org.codehaus.groovy.ast.expr.TupleExpression cannot be cast to org.codehaus.groovy.ast.expr... (3 minute
```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
North Cab - CustomerLoginActivity.java [North_Cab.app] - Android Studio
NorthCab › app › src › main › java › com › rentin › northcab › CustomerLoginActivity
activity_main.xml
DriverMapsActivity.java
google_maps_api.xml
activity_customer_login.xml
CustomerLoginActivity.java
activity_driver_login.xml
DriverLoginActivity.java
Build: Sync
TODO
Terminal
Database Inspector
4: Run
Profiler
Build
6: Logcat
Event Log
Layout Inspector

Fig. 8. Configuration of customer login activity

```
<Edittext
      android:layout_width = "match_parent"
      android:layout_height = "wrap_content"
      android:text = "Email"
      android:id = "@+id/Email"/>

<Edittext
      android:layout_width = "match_parent"
      android:layout_height = "wrap_content"
      android:text = "Password"
      android:id = "@+id/Password"/>


<Button
      android:layout_width = "match_parent"
      android:layout_height = "wrap_content"
      android:text = "Login"
      android:id = "@+id/Login"/>

<Button
      android:layout_width = "match_parent"
```

```
    android:layout_height = “wrap_content”
          android:text = “Registration”
          android:id = “@+id/Registration”/>

           If (user!=null) {

           Intent intent = new intent
                  (CustomerLoginActivity.this,MapActivity.class);

               start Activity(intent);
               finish();
               return;
    }
```

→ Go to CustomerLoginActivity.java, under everything,add

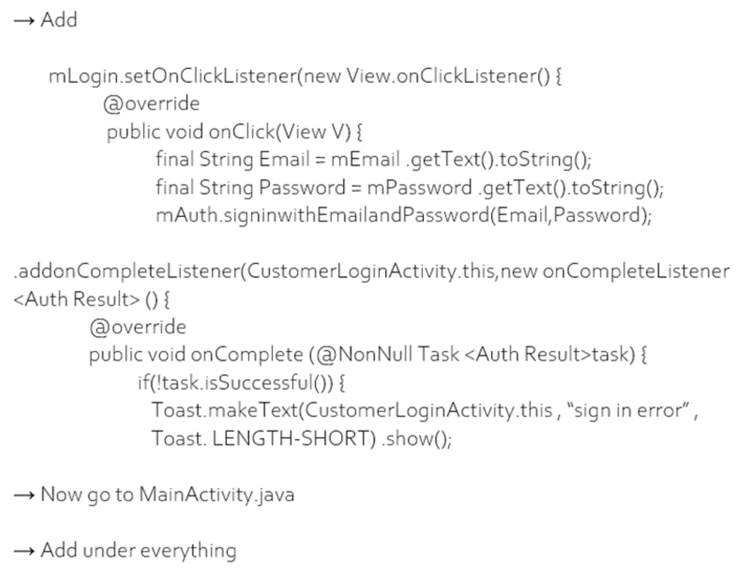
```
        mRegistration.setOnClickListener(new View.onClickListener() {
               @override
                public void onClick(View V) {
                      final String Email = mEmail .getText().toString();
                      final String Password = mPassword .getText().toString();
```

→ Now I am going to create a user and for that, I will use mAuth.
→ In CustomerLoginActivity.java, under final add

```
  mAuth.createUserwithEmailandPassword(Email,Password)
.addonCompleteListener(CustomerLoginActivity.this,new
onCompleteListener
  <Auth Result> () {
                   @override
                   public void onComplete (@NonNull Task
<Auth Result>task) {
                              if(!task.isSuccessful()) {

Toast.makeText(CustomerLoginActivity.this , “sign up
error” ,
                                 Toast. LENGTH-SHORT)
.show();
  }



     else {

  String user-id = mAuth.getcurrentUser().get Uid();
  DatabaseReference curret-user-db =
FirebaseDatabase.getInstance()
  .getReference() .child("Users") .child("Customers")
.child(user-id);

  Current-user-db .setValue(true);


  }
```

→ if this is successful, the user will appear in authentication in firebase

→ I also had to save some information in the database. I had to put the id of the users(Customers) in the drivers' child in firebase



→ Add

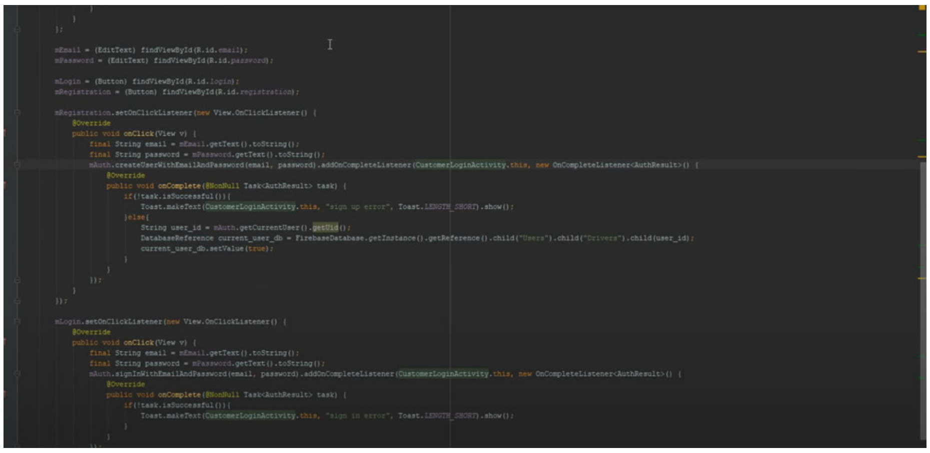
```
    mLogin.setOnClickListener(new View.onClickListener() {
          @override
          public void onClick(View V) {
                final String Email = mEmail .getText().toString();
                final String Password = mPassword .getText().toString();
                mAuth.signinwithEmailandPassword(Email,Password);

 .addonCompleteListener(CustomerLoginActivity.this,new onCompleteListener
 <Auth Result> () {
          @override
          public void onComplete (@NonNull Task <Auth Result>task) {
               if(!task.isSuccessful()) {
                 Toast.makeText(CustomerLoginActivity.this , "sign in error" ,
                 Toast. LENGTH-SHORT) .show();
```

→ Now go to MainActivity.java

→ Add under everything

```
  mCustomer.setonClickListener(newView.Ch.Clicklistener)
                     {

                     @overide
                     public void on Clickch(View V) {
                                  Intent intent = new
intent

(mainActivity.this,CustomerLoginActivity.class);

                     start Activity(intent);
                     finish();
                     return;
```

Figure 9 presents the output for the code for Setting Up Credentials for Customer Login Activity and Input.



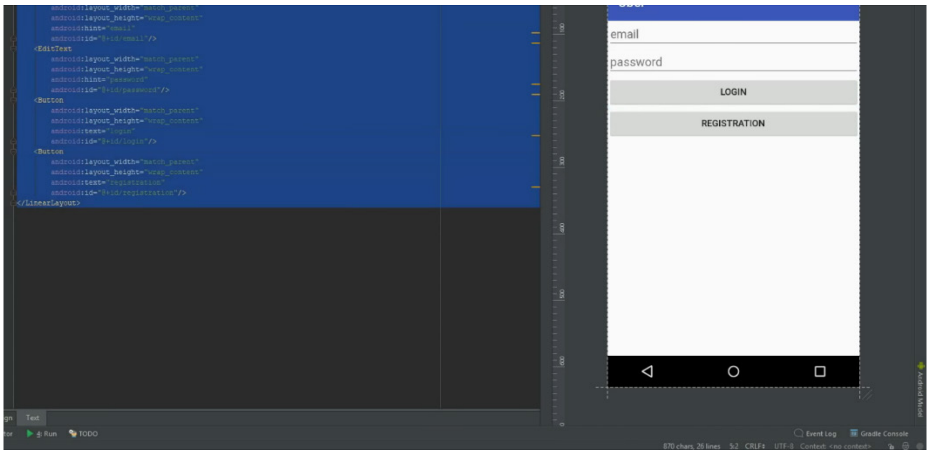
```
            }
        }
    };

    mEmail = (EditText) findViewById(R.id.email);
    mPassword = (EditText) findViewById(R.id.password);

    mLogin = (Button) findViewById(R.id.login);
    mRegistration = (Button) findViewById(R.id.registration);

    mRegistration.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            final String email = mEmail.getText().toString();
            final String password = mPassword.getText().toString();
            mAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(CustomerLoginActivity.this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    if(!task.isSuccessful()){
                        Toast.makeText(CustomerLoginActivity.this, "sign up error", Toast.LENGTH_SHORT).show();
                    }else{
                        String user_id = mAuth.getCurrentUser().getUid();
                        DatabaseReference current_user_db = FirebaseDatabase.getInstance().getReference().child("Users").child("Drivers").child(user_id);
                        current_user_db.setValue(true);
                    }
                }
            });
        }
    });

    mLogin.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            final String email = mEmail.getText().toString();
            final String password = mPassword.getText().toString();
            mAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(CustomerLoginActivity.this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    if(!task.isSuccessful()){
                        Toast.makeText(CustomerLoginActivity.this, "sign in error", Toast.LENGTH_SHORT).show();
                    }
                }
```

Fig. 9. Setting up credentials for customer login activity and input

2.3 Compile and test

For compilation and testing, the following codes were entered.
→ Login as a driver using the emulator
User: driver@driver.com
Password: 123456
→ Now go to Firebase and you can see in a database under drivers' child, I got an id and we can see when we click the Authentication, the user that we just created got authenticated
→ I still didn't create a logout button, so I had to uninstall the app and install it again every time. I wanted to create a user and this time I wanted to log in as a Customer/Passenger
→ Login as a customer using the emulator
User: customer@customer.com
Password: 123456

Figure 10 presents the output for entering the code for using the Emulator to test out Login Page.



```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="email"
        android:id="@+id/email"/>
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="password"
        android:id="@+id/password"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="login"
        android:id="@+id/login"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="registration"
        android:id="@+id/registration"/>
</LinearLayout>
```

Fig. 10. Using the emulator to test out login page

2.4 Google Maps API setup

The main activity here is going to be a map in which the customer can call a northcab and see the location of the Driver and Pickup Location and also for the driver to know where the pickup Location would be. For this, we need to use the Google Maps API.

→ Search for Google API in google and click on the first link
→ Go to console on the top right
→ Empty Screen and go to notifications and we see Create Project: Northcab
→ On the top left, we can see three parallel lines, Click and select API Manager>Dashboard>Enable API>Google Maps Android API>Enable>Create Credentials>Get your Credentials and then we can get our API key
→ Click Done
→ The API key will inform google that Northcab is using the maps.

Figure 11 exhibits the output after entering the code for setting up the project.

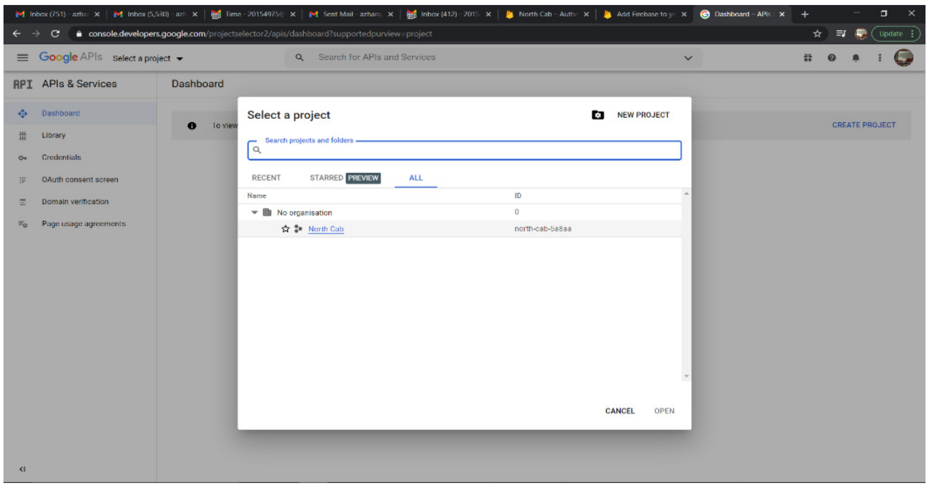
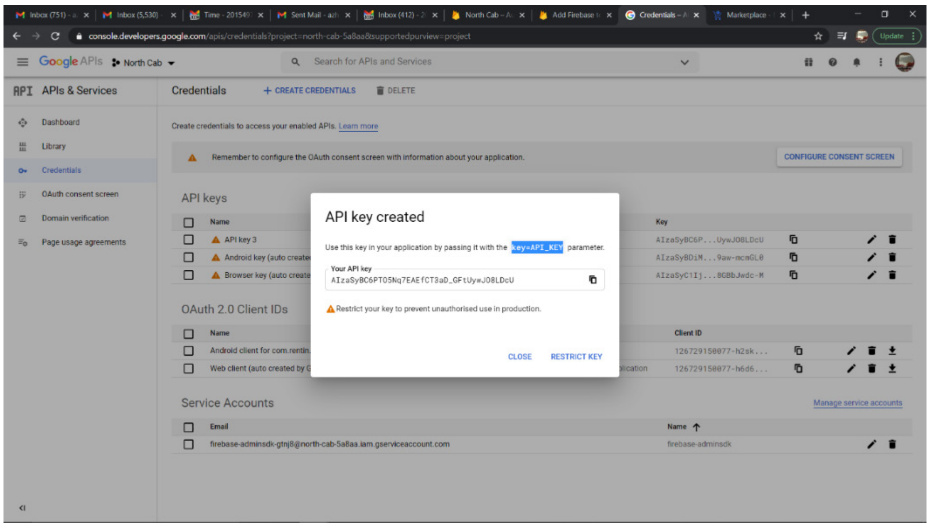


Fig. 11. Setting up project

Figure 12 exhibits the output after entering the code for setting up the API key.



Create credentials to access your enabled APIs. Learn more

Remember to configure the OAuth consent screen with information about your application.

API keys

Name	Key
API key 3	AIzaSyBC6P...UywJ08LDcU
Android key (auto create	AIzaSyBDiM...9aw-mcmGL0
Browser key (auto create	AIzaSyC1Ij...8GBbJwdc-M

OAuth 2.0 Client IDs

Name	Client ID
Android client for com.rentin	126729150077-h2sk...
Web client (auto created by G	126729150077-h6d6...

Service Accounts

Email	Name
firebase-adminsdk-gtnj8@north-cab-5a8aa.iam.gserviceaccount.com	firebase-adminsdk

API key created

Use this key in your application by passing it with the key=API_KEY parameter.

Your API key
AIzaSyBC6PTO5Nq7EAEfCT3aD_GFtUywJ08LDcU

Restrict your key to prevent unauthorised use in production.

Fig. 12. Setting up API key

API key

Figure 13 presents the output after entering the code for Project Error and Traffic Analysis.

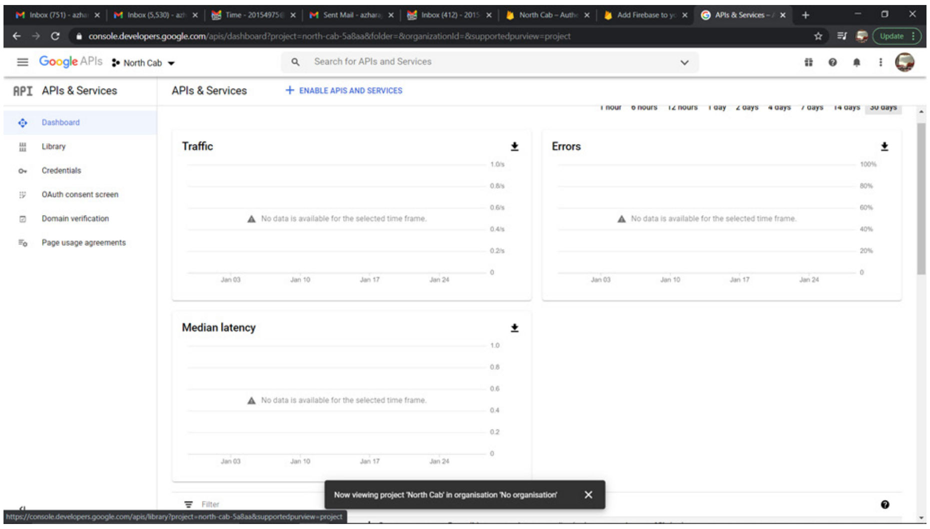
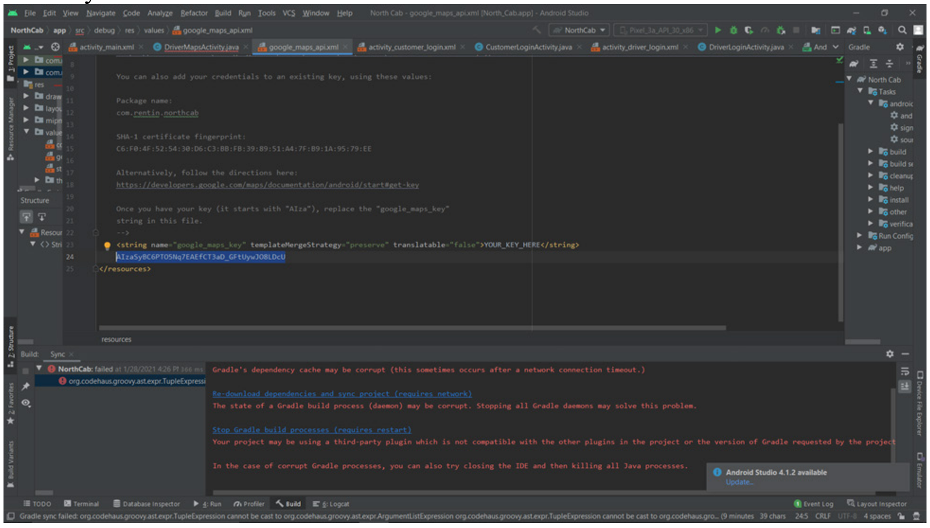


Fig. 13. Project error and traffic analysis

Figure 14 presents the output after entering the code for Configuration of API Key in Android Studio.

API Key in Android Studio



```
You can also add your credentials to an existing key, using these values:

Package name:
com.rentin.northcab

SHA-1 certificate fingerprint:
C6:F0:4F:52:54:30:D6:C3:BB:FB:39:89:51:A4:7F:B9:1A:95:79:EE

Alternatively, follow the directions here:
https://developers.google.com/maps/documentation/android/start#get-key

Once you have your key (it starts with "AIza"), replace the "google_maps_key"
string in this file.
-->
<string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">YOUR_KEY_HERE</string>
AIzaSyBC6PTO5Nq7EAEfCT3aD_GFtUywJO8LDcU
</resources>
```

```
NorthCab: failed at 1/28/2021 4:26 PM 366 ms
org.codehaus.groovy.ast.expr.TupleExpressi

Gradle's dependency cache may be corrupt (this sometimes occurs after a network connection timeout.)

Re-download dependencies and sync project (requires network)
The state of a Gradle build process (daemon) may be corrupt. Stopping all Gradle daemons may solve this problem.

Stop Gradle build processes (requires restart)
Your project may be using a third-party plugin which is not compatible with the other plugins in the project or the version of Gradle requested by the project

In the case of corrupt Gradle processes, you can also try closing the IDE and then killing all Java processes.

Gradle sync failed: org.codehaus.groovy.ast.expr.TupleExpression cannot be cast to org.codehaus.groovy.ast.expr.ArgumentListExpression org.codehaus.groovy.ast.expr.TupleExpression cannot be cast to org.codehaus.gro... (9 minutes 39 chars 24:5 CRLF UTF-8 4 spaces
```

Fig. 14. Configuration of API key in android studio

2.5 Testing Google Maps API and user login and registration

Figure 15 presents the output after entering the code for Testing Google Maps API and User Login and Registration in Emulator.

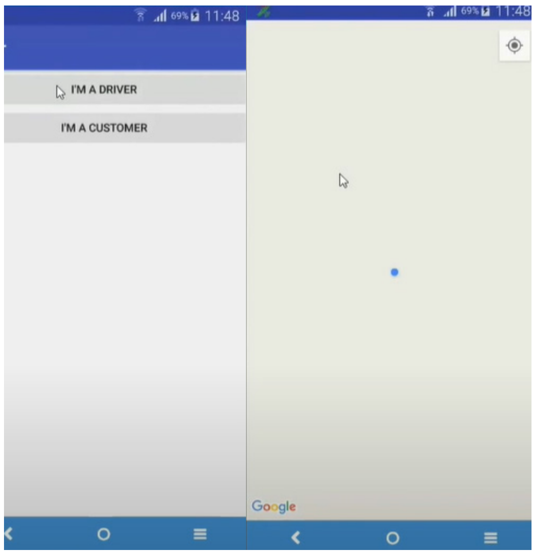


Fig. 15. Testing Google Maps API and user login and registration in emulator

2.6 Code split design

Figure 16 presents the output after entering the code for Configuring Code, Split & Design.

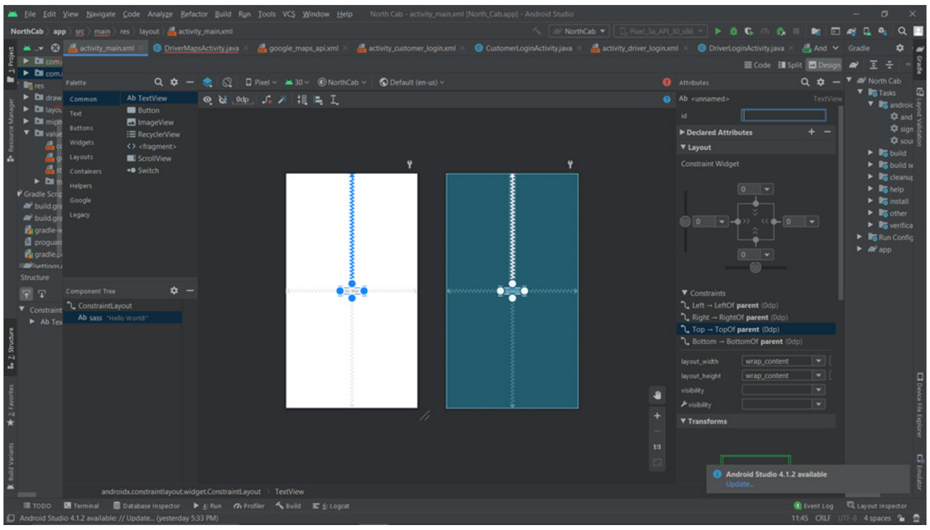
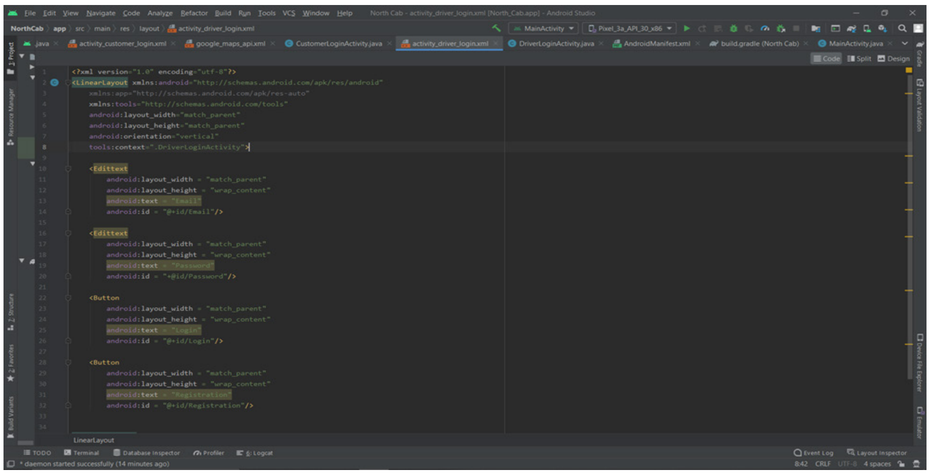

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
activity_main.xml
DriverMapsActivity.java
google_maps_api.xml
activity_customer_login.xml
CustomerLoginActivity.java
activity_driver_login.xml
DriverLoginActivity.java
Code Split Design
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
Ab TextView
Button
ImageView
RecyclerView
<fragment>
ScrollView
Switch
Component Tree
ConstraintLayout
Attributes
Declared Attributes
Layout
Constraint Widget
Constraints
Left → LeftOf parent (0dp)
Right → RightOf parent (0dp)
Top → TopOf parent (0dp)
Bottom → BottomOf parent (0dp)
layout_width wrap_content
layout_height wrap_content
visibility
Transforms
Android Studio 4.1.2 available
androidx.constraintlayout.widget.ConstraintLayout TextView
TODO Terminal Database Inspector Run Profiler Build Logcat
Event Log Layout Inspector

Fig. 16. Configuring code, split & design

2.7 Adding buttons

To add buttons to the application, the following codes exhibited in Figure 17 were entered.

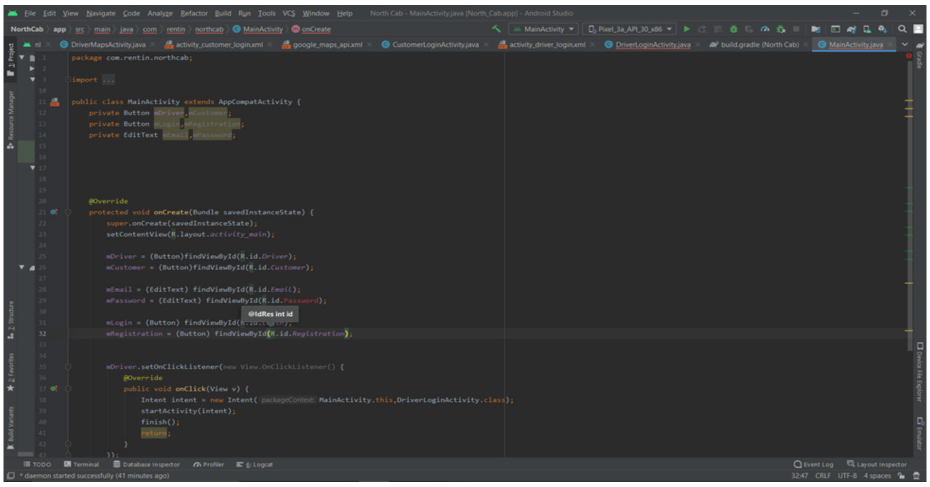


```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".DriverLoginActivity">

    <Edittext
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Email"
        android:id = "@+id/Email"/>

    <Edittext
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Password"
        android:id = "+@id/Password"/>

    <Button
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Login"
        android:id = "@+id/Login"/>

    <Button
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Registration"
        android:id = "@+id/Registration"/>
```

Fig. 17. Adding of buttons in driver login activity

2.8 Creating login and registration

Figure 18 shows the output of the code for finalizing the Login/Registration Tab



```java
package com.rentin.northcab;

import ...

public class MainActivity extends AppCompatActivity {
    private Button mDriver,mCustomer;
    private Button mLogin,mRegistration;
    private EditText mEmail,mPassword;




    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mDriver = (Button)findViewById(R.id.Driver);
        mCustomer = (Button)findViewById(R.id.Customer);

        mEmail = (EditText) findViewById(R.id.Email);
        mPassword = (EditText) findViewById(R.id.Password);
                                     @IdRes int id
        mLogin = (Button) findViewById(R.id.Login);
        mRegistration = (Button) findViewById(R.id.Registration);


        mDriver.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent( packageContext: MainActivity.this,DriverLoginActivity.class);
                startActivity(intent);
                finish();
                return;
            }
        });
```

Fig. 18. Finalizing login/registration tab

2.9 Testing

Figure 19 is the output for the code entered for testing the app via emulator for both Driver/Customer.

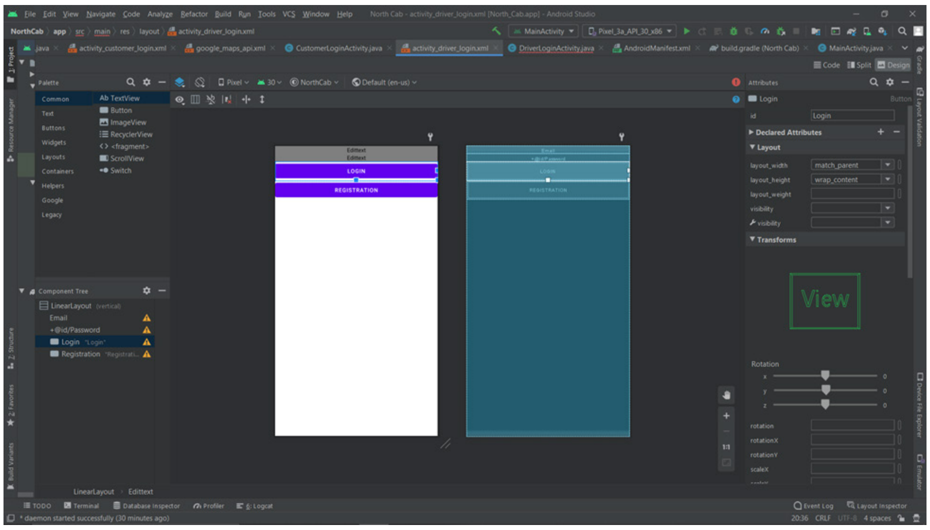
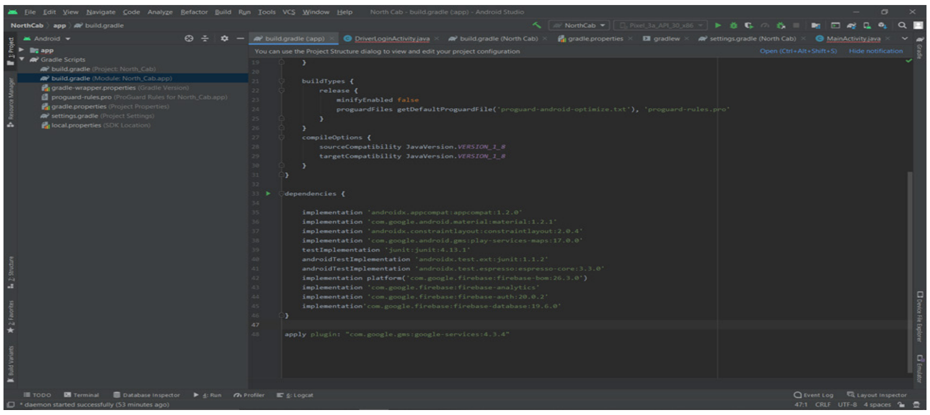


Fig. 19. Testing the app via emulator for both driver/customer

2.10 Gradle dependencies

Figure 20 shows the output code for Gradle Dependancies used in Android Studio for Northcab.

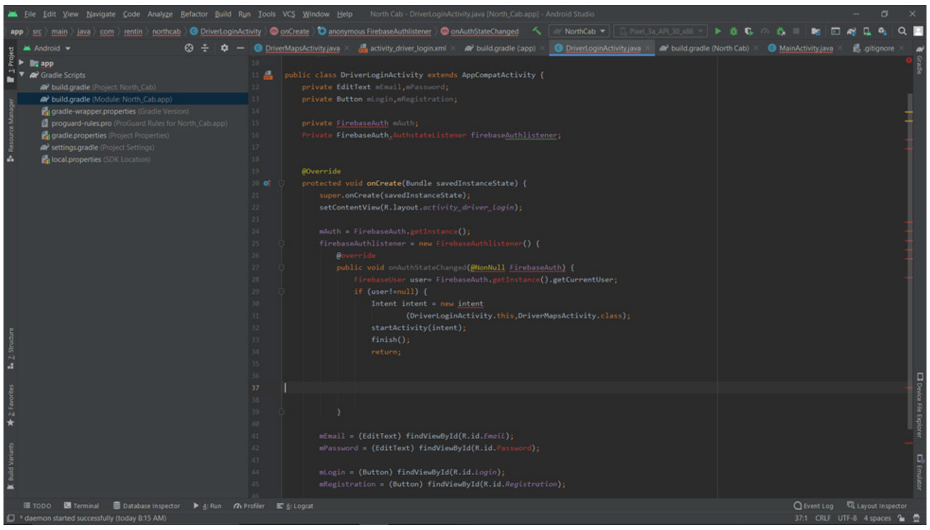


```
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {

    implementation 'androidx.appcompat:appcompat:1.2.0'
    implementation 'com.google.android.material:material:1.2.1'
    implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
    implementation 'com.google.android.gms:play-services-maps:17.0.0'
    testImplementation 'junit:junit:4.13.1'
    androidTestImplementation 'androidx.test.ext:junit:1.1.2'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'
    implementation platform('com.google.firebase:firebase-bom:26.3.0')
    implementation 'com.google.firebase:firebase-analytics'
    implementation 'com.google.firebase:firebase-auth:20.0.2'
    implementation'com.google.firebase:firebase-database:19.6.0'
}

apply plugin: "com.google.gms:google-services:4.3.4"
```

Fig. 20. Gradle dependancies used in android studio for Northcab

2.11 Firebase authentication codes

Figure 21 exhibits the output for firebase Authentication codes used in NorthCab.

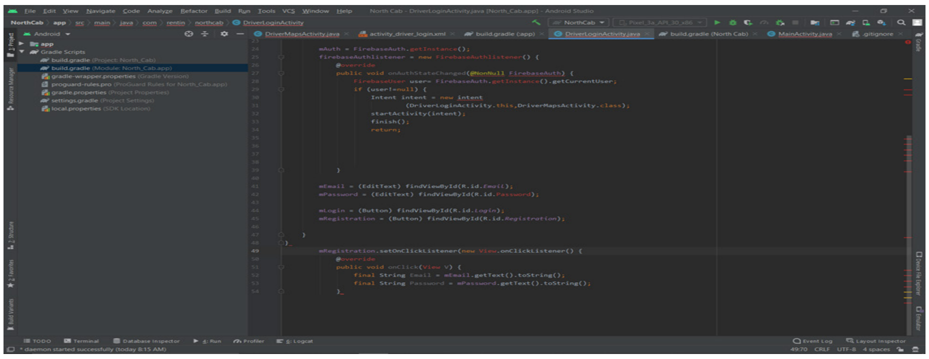


```
public class DriverLoginActivity extends AppCompatActivity {
    private EditText mEmail,mPassword;
    private Button mLogin,mRegistration;

    private FirebaseAuth mAuth;
    Private FirebaseAuth.AuthstateListener firebaseAuthlistener;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_driver_login);

        mAuth = FirebaseAuth.getInstance();
        firebaseAuthlistener = new FirebaseAuthlistener() {
            @override
            public void onAuthStateChanged(@NonNull FirebaseAuth) {
                FirebaseUser user= FirebaseAuth.getInstance().getCurrentUser;
                if (user!=null) {
                    Intent intent = new intent
                            (DriverLoginActivity.this,DriverMapsActivity.class);
                    startActivity(intent);
                    finish();
                    return;




            }

        mEmail = (EditText) findViewById(R.id.Email);
        mPassword = (EditText) findViewById(R.id.Password);

        mLogin = (Button) findViewById(R.id.Login);
        mRegistration = (Button) findViewById(R.id.Registration);
```

Fig. 21. Firebase authentication codes used in NorthCab

2.12 Some errors faced in building the app

Figure 22 shows the errors faced in the development of the NorthCab application.



```
        mAuth = FirebaseAuth.getInstance();
        firebaseAuthlistener = new FirebaseAuthlistener() {
            @override
            public void onAuthStateChanged(@NonNull FirebaseAuth) {
                FirebaseUser user= FirebaseAuth.getInstance().getCurrentUser;
                if (user!=null) {
                    Intent intent = new intent
                            (DriverLoginActivity.this,DriverMapsActivity.class);
                    startActivity(intent);
                    finish();
                    return;




            }

        mEmail = (EditText) findViewById(R.id.Email);
        mPassword = (EditText) findViewById(R.id.Password);

        mLogin = (Button) findViewById(R.id.Login);
        mRegistration = (Button) findViewById(R.id.Registration);

    }
}
        mRegistration.setOnClickListener(new View.onClickListener() {
            @override
            public void onClick(View V) {
                final String Email = mEmail.getText().toString();
                final String Password = mPassword.getText().toString();
            }
```

Fig. 22. Some of the errors we faced in the development of NorthCab

3 Conclusion

This study showed the process of designing and developing a mobile application to ensure that the possible deployment of this system is progressive and that it reaches its maximum capabilities in the nearest future. The following will be considered in future work: Usability evaluation of the mobile application to elicit feedback from users that will help to further improve the quality of the mobile application.

The capabilities of the system will be extended by incorporating a payment module into the app to cater to users who prefer to use cards for transactions, all of these efforts will be geared towards developing the application.

4 References

[1] Xing, H., Stuart, C., Spence, S., & Chen, H. (2021). Alternative fuel options for low carbon maritime transportation: Pathways to 2050. Journal of Cleaner Production, 297, 126651. https://doi.org/10.1016/j.jclepro.2021.126651

[2] Priyanka, E. B., Thangavel, S., Madhuvishal, V., Tharun, S., Raagul, K. V., & Krishnan, S. (2021). Application of integrated IoT framework to water pipeline transportation system in smart cities. In Intelligence in Big Data Technologies—Beyond the Hype (pp. 571-579). Springer, Singapore. https://doi.org/10.1007/978-981-15-5285-4_57

[3] Yang, J., Fei, L., Zhang, X., Ma, X., Luo, K. H., & Shuai, S. (2021). Improved pseudopotential lattice Boltzmann model for liquid water transport inside gas diffusion layers. International Journal of Hydrogen Energy, 46(29), 15938-15950. https://doi.org/10.1016/j.ijhydene.2021.02.067

[4] Antropova, M. Y., Vlasov, A. A., & Kasyanenko, E. F. (2019). Mobile technologies in educational process Chinese universities. New Trends and Issues Proceedings on Humanities and Social Sciences, 6(5), 1–7. https://doi.org/10.18844/prosoc.v6i5.4367

[5] Bianco, N. D.., Giaconi, C., Gison, G., D'Angelo, I., & Capellini, S. A. (2021). Inclusion at the University through technology: A case study in Italy. International Journal of Special Education and Information Technologies, 7(1), 01–15. https://doi.org/10.18844/jeset.v7i1.6793

[6] Çelik, Özgür, & Yavuz, F. (2018). The effect of using mobile applications on literal and contextual vocabulary instruction. International Journal of Learning and Teaching, 10(2), 126–136. https://doi.org/10.18844/ijlt.v10i2.3407

[7] Meena, M., Divyanshu, K., Kumar, S., Swapnil, P., Zehra, A., Shukla, V., ... & Upadhyay, R. S. (2019). Regulation of L-proline biosynthesis, signal transduction, transport, accumulation, and its vital role in plants during variable environmental conditions. Heliyon, 5(12), e02952. https://doi.org/10.1016/j.heliyon.2019.e02952

[8] Qureshi, K. N., & Abdullah, A. H. (2013). A survey on intelligent transportation systems. Middle-East Journal of Scientific Research, 15(5), 629-642.

[9] Golob, T. F., & Regan, A. C. (2001). Impacts of information technology on personal travel and commercial vehicle operations: research challenges and opportunities. Transportation Research Part C: Emerging Technologies, 9(2), 87-121. https://doi.org/10.1016/S0968-090X(00)00042-5

[10] Pelletier, M. P., Trépanier, M., & Morency, C. (2011). Smart card data use in public transit: A literature review. Transportation Research Part C: Emerging Technologies, 19(4), 557-568. https://doi.org/10.1016/j.trc.2010.12.003

[11] Moore, M. D. (2003). Personal air vehicles: a rural/regional and intra-urban on-demand transportation system. Journal of the American Institute of Aeronautics and Astronautics (AIAA), 2646, 1-20.

[12] Lempert, R. J., Preston, B., Charan, S. M., Fraade-Blanar, L., & Blumenthal, M. S. (2021). The societal benefits of vehicle connectivity. Transportation research part D: transport and environment, 93, 102750. https://doi.org/10.1016/j.trd.2021.102750

[13] Ratnakar, R. R., Gupta, N., Zhang, K., van Doorne, C., Fesmire, J., Dindoruk, B., & Balakotaiah, V. (2021). Hydrogen supply chain and challenges in large-scale LH2 storage and transportation. International Journal of Hydrogen Energy, 46(47), 24149-24168. https://doi.org/10.1016/j.ijhydene.2021.05.025

[14] Adewumi, A., Odunjo, V., & Misra, S. (2015, December). Developing a mobile application for a taxi service company in Nigeria. In 2015 International Conference on Computing, Communication, and Security (ICCCS) (pp. 1-5). IEEE. https://doi.org/10.1109/CCCS.2015.7374204

[15] Siau, K., & Shen, Z. (2003). Mobile communications and mobile services. International Journal of Mobile Communications, 1(1-2), 3-14. https://doi.org/10.1504/IJMC.2003.002457

[16] Villamizar, M., Garcés, O., Castro, H., Verano, M., Salamanca, L., Casallas, R., & Gil, S. (2015, September). Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud. In 2015 10th Computing Colombian Conference (10CCC) (pp. 583-590). IEEE. https://doi.org/10.1109/ColumbianCC.2015.7333476

[17] Ozcan, M. S., & Kert, S. B. (2020). Use of mobile applications in collocation teaching. Cypriot Journal of Educational Sciences, 15(5), 1176–1190. https://doi.org/10.18844/cjes.v15i5.4629

[18] Dağyar, M., Kasalak, G., & Sezgin, E. (2020). What do primary school students think about mobile programming education? " ˜Developing my own mobile game". World Journal on Educational Technology: Current Issues, 12(4), 258–277. https://doi.org/10.18844/wjet.v12i4.5179

[19] Zhou, S. L., Jia, X., Skinner, S. P., Yang, W., & Claude, I. (2021). Lessons on mobile apps for COVID-19 from China. Journal of Safety Science and Resilience, 2(2), 40-49. https://doi.org/10.1016/j.jnlssr.2021.04.002

[20] González-Cabañas, J., Cuevas, Á., Cuevas, R., & Maier, M. (2021). Digital contact tracing: Large-scale geolocation data as an alternative to Bluetooth-based apps failure. Electronics, 10(9), 1093. https://doi.org/10.3390/electronics10091093

[21] Uzunboylu, H., Ozcinar, Z., Kolotushkin, S., Kalugina, O., & Zulfugarzade, T. (2019). Research and trends in technology and gifted child: Results of a content analysis. International Journal of Emerging Technologies in Learning (iJET), 14(22), 56-69. https://doi.org/10.3991/ijet.v14i22.11751

[22] Hassan, A., Yakubu, M. B., Bulama, M., & Shitu, A. A. (2018). A customer perspective on infrastructure & legislative effects to use mobile banking apps in Nigeria. Global Journal of Information Technology: Emerging Technologies, 8(3), 102–113. https://doi.org/10.18844/gjit.v8i3.4050

[23] Karakaya, S. (2022). Mechanical design of a differential drives mobile robot platform. Global Journal of Computer Sciences: Theory and Research, 12(1), 01–11. https://doi.org/10.18844/gjcs.v12i1.6508

[24] Karaca, A. (2020). Innovative technologies and living spaces; Updated living standards according to the evolution of homo sapiens. New Trends and Issues Proceedings on Advances in Pure and Applied Sciences, (12), 91–108. https://doi.org/10.18844/gjpaas.v0i12.4990

5 Authors

Ramiz Salama is with Department of Computer Engineering, AI and Robotics Institute, Research Center for AI and IoT, Near East University, Nicosia, Mersin 10, Turkey (email: ramiz.salama@neu.edu.tr).

Azhar Abdul Jabbar is with Department of Computer Engineering, AI and Robotics Institute, Research Center for AI and IoT, Near East University, Nicosia, Mersin 10, Turkey.

Article submitted 2022-09-14. Resubmitted 2022-10-13. Final acceptance 2022-10-14. Final version published as submitted by the authors.