ParkingEaze

Database connections functionality update

This document provides the information regarding the functionality of the database in the android mobile application. The demonstration has been held during the online class session for the database connection. The database is stored and saved in the firebase for the android application.

Grading for this milestone:

/1 Participated: Online session

/4 Online demo

/1 Addresses connection to enterprise wireless (v.s. home/open Wi-Fi/hotspot)

/1 Database configuration mentioned

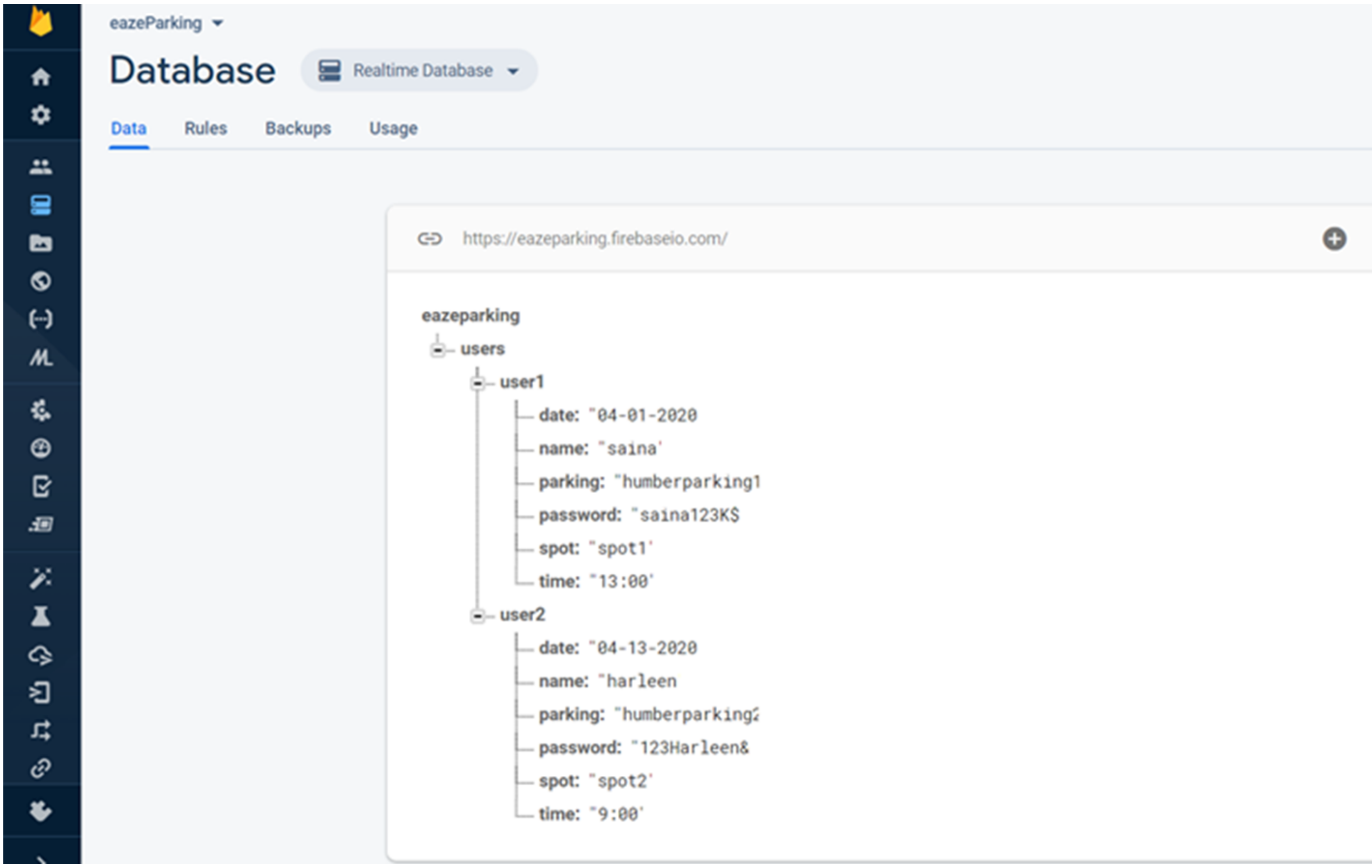
/1 Security considered

/2 Unit and production testing considerations

Database configuration

Firebase Implementation for our database

A firebase project is created using a username and password then firebase is added with the android application project in the android studio. The data is stored in the JSON format along with providing an offline facility of data storage by enabling the disk persistence. After creating firebase project, the android application can be registered to it. Firebase also supports to focus on performance and implementation issues by repairing bugs precisely from its backend solution.However,our group have experienced many problems in order to connect to firebase.Eventually,we are able to display the users in the firebase , who are logged in the application.



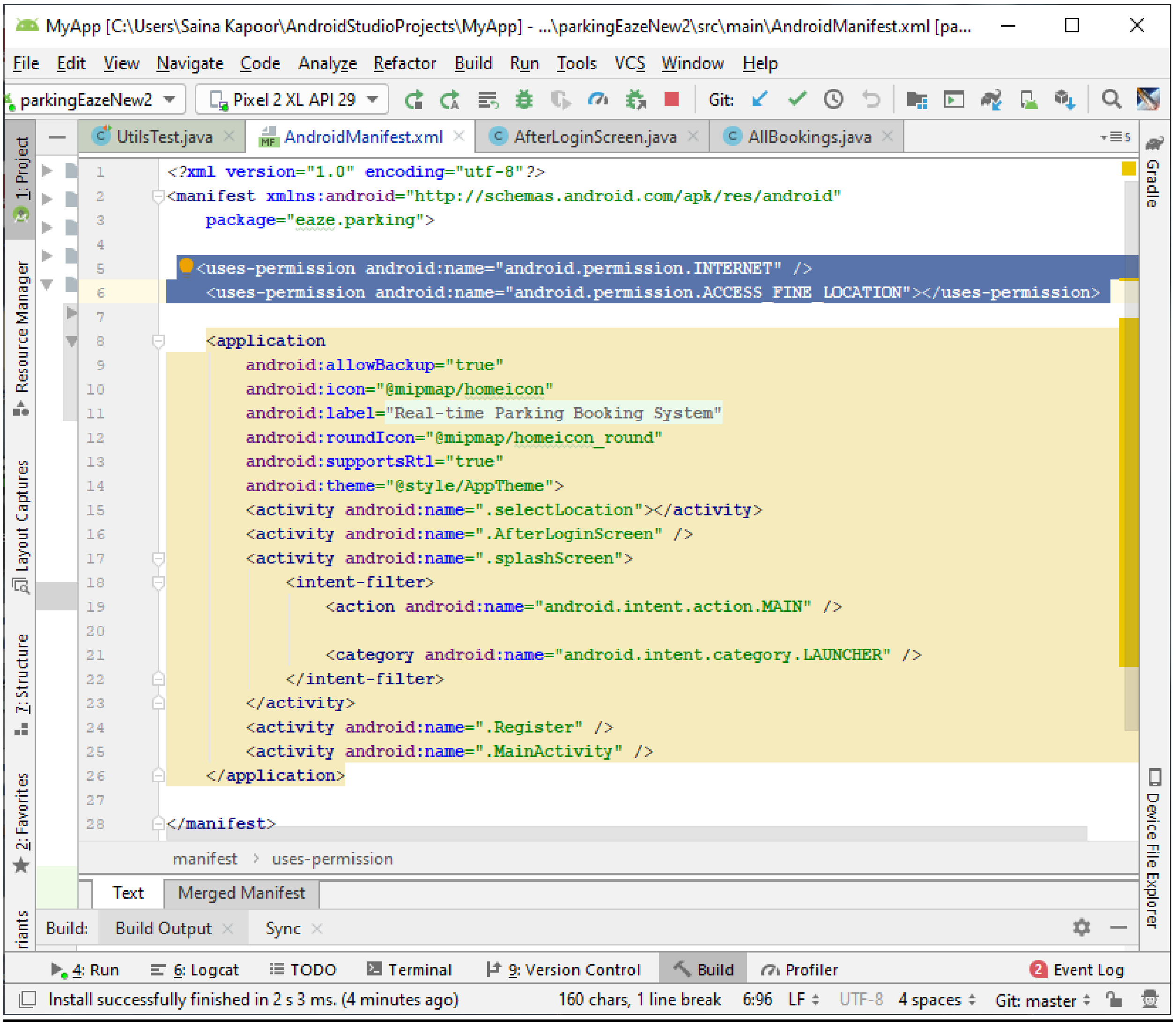
```
https://eazeparking.firebaseio.com/

eazeparking
 ⊟- users
     ⊟- user1
         ├─ date: "04-01-2020
         ├─ name: "saina'
         ├─ parking: "humberparking1
         ├─ password: "saina123K$
         ├─ spot: "spot1'
         └─ time: "13:00'
     ⊟- user2
         ├─ date: "04-13-2020
         ├─ name: "harleen
         ├─ parking: "humberparking2
         ├─ password: "123Harleen&
         ├─ spot: "spot2'
         └─ time: "9:00'
```

Figure 1 The information stored in the firebase is displayed

Addresses connection to enterprise wireless

The connection is established using the wireless network which is WIFI. The application uses the internet using the runtime permission feature in the android studio which is provided in the AndroidManifest.xml file. The connection can be established in order to access the internet.

<uses-permission android: name="android. permission. INTERNET"/>



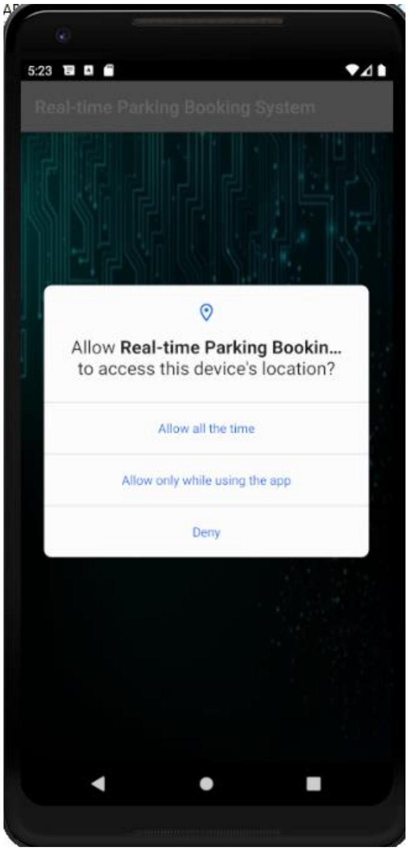
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="eaze.parking">

   <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"></uses-permission>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/homeicon"
        android:label="Real-time Parking Booking System"
        android:roundIcon="@mipmap/homeicon_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".selectLocation"></activity>
        <activity android:name=".AfterLoginScreen" />
        <activity android:name=".splashScreen">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".Register" />
        <activity android:name=".MainActivity" />
    </application>

</manifest>
```

```
Install successfully finished in 2 s 3 ms. (4 minutes ago)     160 chars, 1 line break   6:96   LF   UTF-8   4 spaces   Git: master
```

Figure 2 The AndroidManifest.xml file in android studio

Consideration of security:

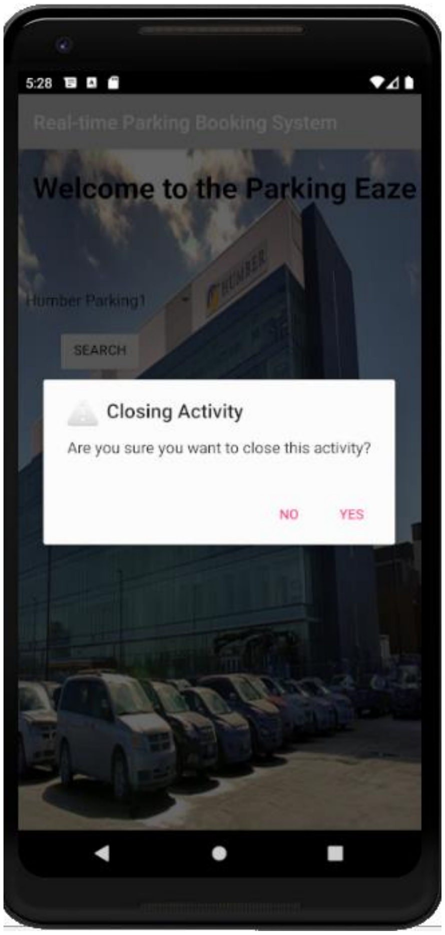
The security of the application used by the user is enhanced as shown in figure3 and Figure4 below. When the user opens the application, a dialog box appears which prompts a message whether the user wants to provide the location access to the application that follows user's privacy. It also assures and reminds them if they want to exit without the submission of any changes when the user clicks exit option. Hence,instead of instantly closing the app, a dialog box is prompted.


5:23
Real-time Parking Booking System

Allow **Real-time Parking Bookin...** to access this device's location?

Allow all the time
Allow only while using the app
Deny

Figure 3 Asking permission to use the current location before seeing the nearby parking lots.



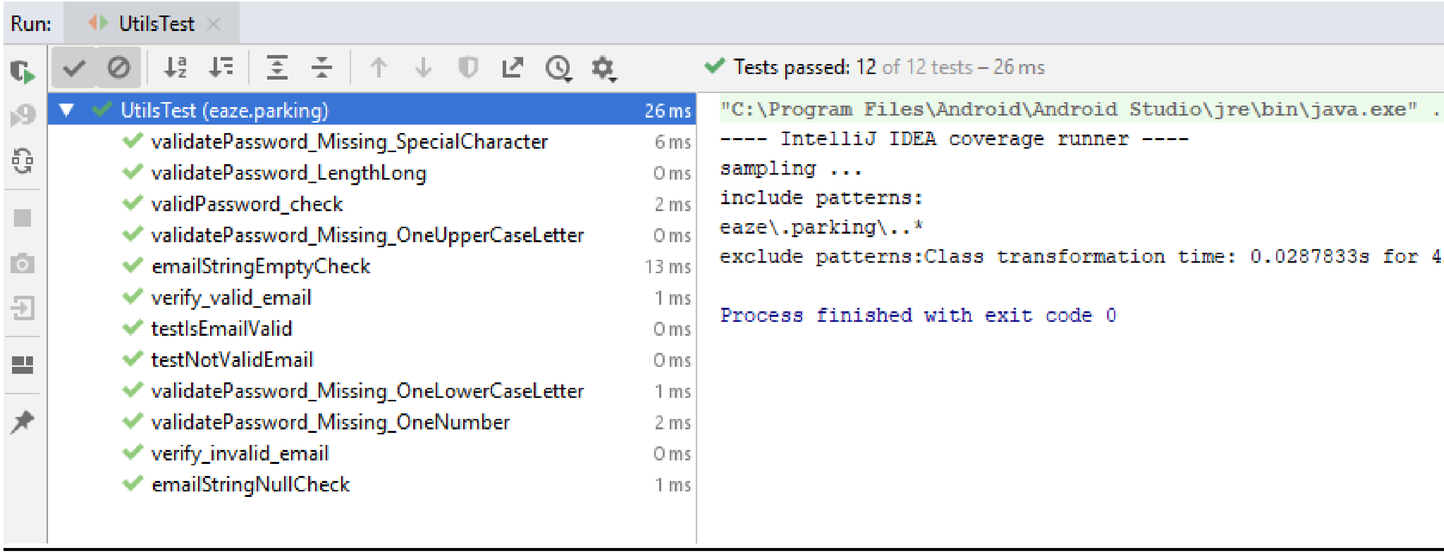
Closing Activity

Are you sure you want to close this activity?

Figure 4 Asking the user if the user wants to leave without submitting their changes.

Unit Testing

In the figure below, the testing is done for software validation code of some cases. The test cases are implemented and tested step by step which are eventually passed. Performing unit tests is intended to be uncomplicated, generally the tests are written in the structure of functions that will decide whether a returned value equals the value you were anticipating when you entered the function in the software code.



✓ Tests passed: 12 of 12 tests – 26 ms

```
▼ ✓ UtilsTest (eaze.parking)                          26 ms
    ✓ validatePassword_Missing_SpecialCharacter        6 ms
    ✓ validatePassword_LengthLong                      0 ms
    ✓ validPassword_check                              2 ms
    ✓ validatePassword_Missing_OneUpperCaseLetter      0 ms
    ✓ emailStringEmptyCheck                           13 ms
    ✓ verify_valid_email                               1 ms
    ✓ testIsEmailValid                                 0 ms
    ✓ testNotValidEmail                                0 ms
    ✓ validatePassword_Missing_OneLowerCaseLetter      1 ms
    ✓ validatePassword_Missing_OneNumber               2 ms
    ✓ verify_invalid_email                             0 ms
    ✓ emailStringNullCheck                             1 ms
```

```
"C:\Program Files\Android\Android Studio\jre\bin\java.exe" .
---- IntelliJ IDEA coverage runner ----
sampling ...
include patterns:
eaze\.parking\..*
exclude patterns:Class transformation time: 0.0287833s for 4

Process finished with exit code 0
```

Figure 5 The test cases for the validation are tested and all of them are passed

Production Testing

The application connected to the firebase and once the application is complete and if it is eligible to be launched on the google play store, then the users can download it from there based on their requirement. The hardware consists of the sensors and effectors which comply differently according to the production testing.

- **UtilsTest.java contains testing functions from the code and the code can be found through this link**
- https://github.com/NavkiranKaur/ParkingEaze/blob/master/code%20files/UtilsTest.java

- **The main activity java file that contains firebase connection code can be found using this link below**
- https://github.com/NavkiranKaur/ParkingEaze/blob/master/code%20files/MainActivity.java