RideOn - Android App to Share Ride

Charan Lellaboyena, Harikanth Ghanta

Department of Electrical and Computer Engineering
Stony Brook University
Stony Brook, USA

***Abstract*— Mobile Cloud Computing is widely accepted as a concept that can significantly improve the user experience when accessing mobile services. By removing the limitations of mobile devices with respect to storage and computing capabilities and providing a new level of security by a centralized maintenance of security-critical software for e.g. mobile payment applications, it is expected that it will find broad acceptance on the business as well as consumer side. Research indicates [1] that Mobile Cloud Computing will additionally help to make visions of context services become reality. However, Mobile Cloud Computing concepts rely on an always-on connectivity and will need to provide a scalable and - when requested - quality mobile access.This surge opens up a huge opportunity to process and analyse data and provide insights from this crowd sourced data. Our system deals with using the power of smart phones along with cloud computing to help the user create a ride by entering the details of ride and can also join rides which other users created and also record the data of the ride like distance and time.**

***Keywords*— Mobile, Cloud computing, Android, Reminders, Location, Calendar, Notification, Firebase SDK, XML, Smartphone applications**

I. INTRODUCTION

The concept of Mobile Cloud Computing (MCC) intends to make the advantages of Cloud Computing available for mobile users but will provide additional functionality to the "cloud" as well. Mobile Cloud Computing (MCC) will help to overcome limitations of mobile devices in particular of the processing power and data storage. It might also help to extend the battery life by moving the execution of commutation-intensive application "to the cloud". However, a significant gain in battery stand-by time will require that the wireless connectivity for the MCC operation is at least as energy-efficient as the state of the art. MCC is also seen as a potential solution for the fragmented market of mobile operating systems with currently eight major operating systems.

Over the course of the last decade rapid advancements in the field of smartphones and cloud computing have created various opportunities which were impossible before. Nowadays smartphones have become ubiquitous and an increasing uptake of cloud computing will make it easier for individuals to consciously work together on crowd sourcing data. But, with all right tools lying around there is a gap between cloud computing and crowd sourced data. Particularly one opportunity is gathering data related to events of the bicycle rides a person wishes to go and share it with the community of other users of the app and encourages others to join the ride. This application encourages community in going rides together enjoying nature maintaining good health and improving social life. The normal way of going to a ride is just asking your friends in your known circle to go on a ride together and planning it whenever they are free. But this process has many hurdles as all of your friends may to be free at the same time. By using our Android application you can ask the entire outer world of your friends circle about going on a ride and can post your ride timings whenever you are free and let others join your ride.

You can also share your distance travelled and time taken to achieve it to your friends and encourage them to go on a ride to break your record.

Section II gives an overview of design architecture used Section III gives a brief description about the system, section IV gives an idea of the user interface and the different options available for the user. Section V deals with the Firebase Real Time Database implementation. Section VI deals with the E-Mail dispatcher and Section VII deals with the calendar event creator and Section VIII deals with the User Permissions and Section XI deals with User Statistics that are stored which helps user to share their leaderboards.

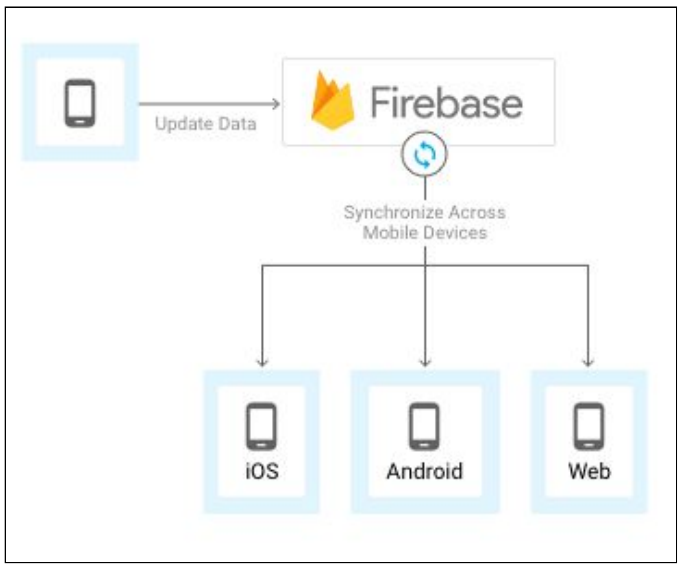
II. DESIGN

Most mobile apps and games need a backend service for things that can't be done solely on-device, such as sharing and processing data from multiple users, or storing large files. In our application we need to store the login credentials, data of the user entered while using the app. Selecting a design pattern for the application is the

most crucial part of the project. Building a backend service [10] for a mobile app is similar to building a web-based service, with some additional requirements:

- Limit on-device data storage
- Synchronize data across multiple devices
- Handle the offline case gracefully
- Send notifications and messages
- Minimize battery drain

The pattern used is Firebase, a two-tier architecture, in which mobile app and Firebase both manipulate the data directly. This creates important differences in how security and data validation are handled.



III.IMPLEMENTATION

This Android application [2] is divided into individual modules that enable user authentication, user interface, email-dispatcher [4], add calendar event, user statistics, cloud based database storage, distance calculator, data processing etc. All these modules work together to achieve our goal of providing user data and analytics. The primary system implementation is to authenticating users to sign in the application create a ride, storing the details of the created ride in the cloud [3], receive e-mail notifications of other created rides, join rides created by others, adding events of upcoming joined rides to the calendar [6], starting the ride and storing the statistics of ride like time taken and distance travelled. Our application has the following components:

i. User Interface
ii. Firebase Realtime Database
iii. E-mail dispatcher (smtp host)
iv. Calendar event creator
v. User permissions
vi. User statistics

IV.USER INTERFACE

The user interface acts as the bridge between the user and the cloud database. This interface provides the user with different features like user access, interface to schedule rides, join rides, view user statistics.

A. Login Activity

As privacy is required to protect the data of the application we used Firebase Authentication [4] to securely sign in the users with their existing e-mail, facebook and Gmail accounts. In this layout, the user gets to login with an existing an account, create a new account and also update password. The authentication of users is implemented using Firebase by Google. We created a Firebase Auth API [3] instance. This instance is linked with the Login button. The API instance captures the entered email and password and sends it to the cloud database which then compares it with the list of already registered users. In a case where the user is not registered it prompts the user to create an account. In order to create a new account there is a button to create a new account that redirects to the Signup activity.

B. Main_Activity

After successful login, the home activity displays to the user with a set of buttons to interact with the app. The buttons are “CreateRide”, “ShowRides”. Also, the user is provided an option to logout from the application.

C. CreatRide Activity

Once the CreateRide button is pressed in the Main_Activity, the CreateRide activity is initiated and it contains edittext views to enter the ridename, source, destination, time and date of the ride you wanted to create. At the end of the list a CreateNewRide button is present.

D. JoinRides Activity

JoinRidesActivity is created when you press the CreateNewRide button in CreateRide Activity. In this activity an email is sent to everyone whose details are stored in our Firebase database by E-mail dispatcher that a new ride is created.

Now a new listview is inflated where we have a list of rides which were created by other users. You can check

the ride if you are interested in any of them. You can also see the details of the ride by clicking the textview of the ride. If you check any of the rise checker_eventlistener is called and you will be added to the ride in the database. Now you will be notified with an e-mail [5] about the details of the ride and a calendar reminder event [6] will be created in your mobile which notifies you before the ride gets started. you also have start the ride option in this layout and you can start the ride by clicking it. It has a dynamic timer and distance calculator which is calculated by the location services implemented in the application.

V. FIREBASE REALTIME DATABASE

The Cloud Database [2] we used in this application is provided by Google. It is called as the Firebase Realtime Database. We store user credentials, user ride details and statistics in this realtime database.

First we have to add the android project to the firebase by creating new firebase project in firebase console. After this an instance of this database is created by calling .getInstance() method.

1. **User Class:** We store credentials of the user in the database by creating an instance of this class and pushing it to the database.
2. **RideDetails Class:** We store all the details of created ride such as name, source, destination, timestamp by creating an instance of this class and pushing it to the database reference.
3. **RideDetailsMap Class:** All the data of a user and to which ride he has joined is stored in this instance.

By using the above mentioned instances we retrieve and store the data according to the requirement of the application.

VI. E-MAIL DISPATCHER (SMTP HOST)

SMTP is the protocol (method) used to send email between mail servers, and by your email software to submit outgoing email. POP and IMAP are methods used to read mail stored on a mail server. We used sun.android.mail API [5] for sending an e-mail to the users when required in the application.

EmailDispatcher Class: In this class we have entered a template where it sends the newly created ride details to all the user who already had logged in once. We fetch the details stored in the firebase database by calling the required instances described in the previous section and append it to the template and send it to the users.

We also use this class to send a confirmation e-mail to the user if he joins a new ride created by other user.

VII. CALENDAR EVENT CREATOR

We used android.calendar.events API [6] in the application to save the ride as an event in the calendar that user has joined.

EventsCalendar Class: In this class we retrieve the timestamp of the ride and details of the ride from the database and create an event in the calendar of the user. This helps in reminding the user that a ride is scheduled at the start of that ride.

VIII. USER PERMISSIONS

To maintain security [11] for the system and users, Android requires apps to request permission before the apps can use certain system data and features. Depending on how sensitive the area is, the system may grant the permission automatically, or it may ask the user to approve the request.

Listing the user permisssions used in the application.

1. android.permission.INTERNET to access Internet
2. android.permission.ACCESS_NETWORK_STATE,
3. android.permission.READ_CALENDAR to read the calendar,
4. android.permission.WRITE_CALENDAR to write into calendar, android.permission.ACCESS_FINE_LOCATION to get the exact location

Enabled the sensors TYPE_ACCELEROMETER, TYPE_LINER_ACCELARATION in the application which are used to measure the speed of the user.

IX. USER STATISTICS

We use the G.P.S service in the Smartphone to calculate the coordinates of a user location when he starts the ride. From these coordinates we find the distance travelled from the starting point to the present location and display it dynamically.We also start the timer once the user starts his ride to track his time taken to complete the ride.

X. CHALLENGES

The implementation required cloud services and

heavily relies on them. It took some time to understand how Firebase Database stores the data. To store and retrieve the data like the user credentials, details of created ride, user id's of users joined in a particular ride is challenging as we have to store each of the above mentioned data each in separate child in JSON file of the database. Implementing E-mail dispatcher and CalendarEvents creator is challenging and interesting.

To make the application appealing user interface should be butter smooth and attractive. It was challenging to add the styles and colors to the layout and buttons and background as we see in the app.

XI. FUTURE WORK

There is a lot of scope to improve this application as we can add some tons of features which helps connecting socially and riding together long destinations. We can add live locations of the users who started the same ride together which gives us a real feel of racing together like in computer racing games. We can also connect it to the smartwatch and can display the speed required to beat the person next to you and store realtime health statistics like pulse rate to the application.

XI. CONCLUSION

In this application, we have designed an app that encourages the user to create and join rides and can travel to long destination with a group of people and can make new friends. Share the statistics like time taken for the ride and distance travelled to friends and family and can encourage them by challenging them with user's results. With the help of Firebase cloud services authentication of users is done without any hassles and backend of the application is implemented without any extra infrastructure.

XII. ACKNOWLEDGEMENT

This work is completed under the guidance of Professor Shan Lin as part of coursework in ESE 545 - Mobile Cloud Computing in Spring 2017. We thank him for the guidance and support which drove the interest in application development.

REFERENCES

[1] *Mobile Cloud Computing - Next Generation Browsers Widgets SIM Network-as-a-Service and Platform-as-a-Service 3Q*, 2009, [online] Available: http://www.abiresearch.com/research/1003385.

[2]https://developer.android.com/reference/packages.html.

[3] https://firebase.google.com/docs/

[4]https://www.udacity.com/course/firebase-in-a-weekend-by-google-android--ud0352

[5]https://developers.google.com/gmail/api/quickstart/android

[6]https://developer.android.com/guide/topics/providers/calendar-provider.html

[7]http://www.json.org/

[8] Kuo-chu Wu, Wei-y Liu, Shiow-yang Wu, "Development model and environment for dynamic mobile cloud services", *Cloud Computing Congress (APCloudCC) 2012 IEEE Asia Pacific*, pp. 31-36, 2012.

[9] Le Guan, Xu Ke, Meina Song, Junde Song, "A Survey of Research on Mobile Cloud Computing", *Computer and Information Science (ICIS) 2011 IEEE/ACIS 10th International Conference on*, pp. 387-392, 2011.

[10]https://cloud.google.com/solutions/mobile/mobile-app-backend-services

[11]https://developer.android.com/guide/topics/permissions/index.html