EVENT ADVERTISING AND REGISTERING APPLICATION

J COMPONENT PROJECT REPORT
Review - III

Submitted by

SOUMI HAZRA (17BCE0446)
AYUSH SHARMA (18BCE0172)

Submitted to

PROF. AKILA VICTOR MA'AM

in partial fulfillment for the award of the degree of

B. Tech

in

Computer Science and Engineering

Vellore-632014, Tamil Nadu, India

School of Computer Science and Engineering

October, 2019

ACKNOWLEDGEMENT

We would firstly like to acknowledge our Professor Ma’am AKILA VICTOR for throughout helping us out in the project with his knowledge and experience. We would also like to thank our batch mates for exchanging ideas with us which actually helped a lot in creative thinking and exploring in the topic.

We are thankful to the VIT administration for giving us this golden opportunity of trying out our own ideas on a semester level and getting a chance to execute it.

We thank the entire staff of VIT who have directly or indirectly contributed to the system for its proper functioning and giving students the freedom to analyze things and projecting their creative minds in the limelight.

ABSTRACT

Events play an important role in our society. Any happening or an activity can be referred as an event. Individuals often find they lack the expertise and time to plan events themselves. Independent planners are needed to step in and give these special events the attention they deserve. In the current scenario, planning an event requires a lot of patience and hustle bustle right from deciding the theme to deciding venue and events. Lots of factors need to be considered while making each decision.

This project intent to solve the problems of propagating news and information, and also alleviate the problem of traditional event managing procedures such as lots of paper work, or long queue at the registration desk. The objective of this project is to develop an android application which provides interesting news and events. Moreover, users will be able to manage their event participation, such as reserving their seats in events, registering at the event site, and so on.

AIM

The application that we have created in this project aims at overcoming the difficulties of traditional event advertising and registering issues that often leads to a confusion of which event one has registered for and also of which event one can attend. Our application makes such tasks of event advertising and registering for any event extremely systematic and simple, that is, it is very user-friendly so one feels like attending nearby events in a hassle free manner.

INTRODUCTION

Smartphone is a common computational device that possessed by the most of people nowadays, which is the inspiration to create an application that its information can be easily reached anywhere, any time. In addition, it would be difficult to manage all event registration manually, because it will take a long time for a long queue of customers to sign their name at the registrations table, also a lot of document to handle. Furthermore, people nowadays prefer convenience for their life. In other words, it is harder for users to open the website than click on an application in their smartphones.

This application focuses on solving problems of event registration and management and also providing news, information of events, and project ideas. First of all, users will be able to reserve and manage their event participation via this application. Additionally, this application provides significant information and news of many interesting events from the event provider.

This application system consists of two main components which are front-end system and back-end system as shown in figure 1.

• The Front-end System is the information-displaying section which queries the data from the remote database and also able to send data to be stored in the database. Moreover, the staff side front-end system will send the participant information to the server to verify their identity.

• The Back-end System is the database management section which always interacts with the front-end system. Additionally, it will send the required data to the front-end system whenever the request is sent.

Event scheduling is a fest management system shaped to provide all the necessary manipulations and maintenance of data related to fest conducted by the different departments of various colleges. It includes the management of coordinators and volunteers who manage the event, sponsors who provide the basic funding to the fest and the participants. The main objective of the scheduling is to avoid the overlapping of venues and time for each event. This is done by Genetic Algorithm usage. The budgeting, account of expenses and managing it is also a tedious job in event scheduling.

EXISTING METHODOLOGY

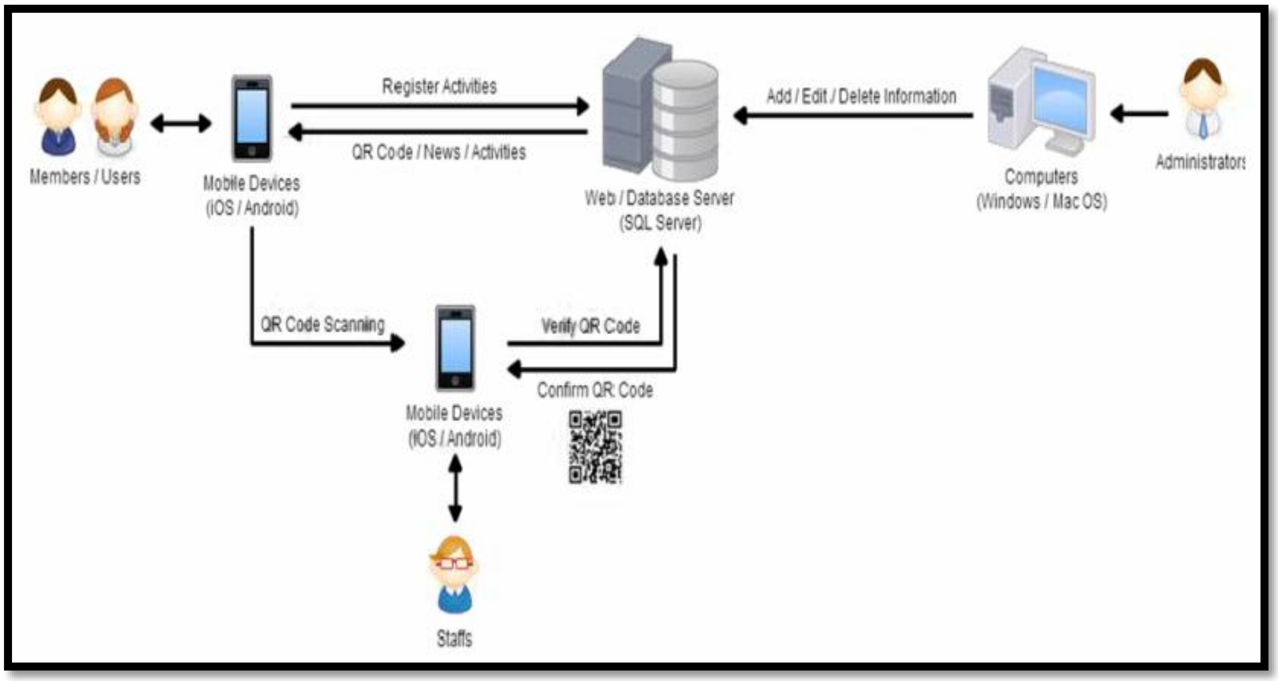
Event calendar apps and websites are the existing methodologies which are used for event management purposes. But this methodology do not offer a platform to promote one's event. It only allows user to add and edit events in which he likes to participate rather than giving him a wide options of event available in the campus for registration at that particular time.

These event calendar app uses CalDAV integration which is an extension of WebDAV that provides a standard for clients to access calendar information on a remote server. Google provides a CalDAV interface that you can use to view and manage calendars using the CalDAV protocol.

All of these existing apps are using SQL database which our not favourable for our app as it include following drawbacks:

- Hard to maintain temporal tables
- Need a schema nomenclature to make it easy to other people to look at your database
- If used as a backend for applications its takes a while to update whereas firebase updates immediately.

PROPOSED METHODOLOGY


Members / Users
Mobile Devices (iOS / Android)
Register Activities
QR Code / News / Activities
Web / Database Server (SQL Server)
Add / Edit / Delete Information
Computers (Windows / Mac OS)
Administrators
QR Code Scanning
Verify QR Code
Confirm QR Code
Mobile Devices (iOS / Android)
Staffs

✓ENTIRE SYSTEM

This application consists of six main modules which are Authentication System, Member Management System, News Management System, Event Management System, Project Ideas Management System, and Administrator Management System.

1. Authentication System: provides security to the whole system by allowing only authorized members to have the right to utilize the preserved features.

- Member Registration System: a system which allows user to be able to register and become a member.
- Log-in System: a system which allows member to be able to log into the application and access the preserved area.
- FIREBASE METHODOLOGY USED FOR CREATING USERS:

  We have used mongo dB as our database to store data in the form of Jason file. We have created a user creation system which will create user with email and password credentials which can be used for login purposes.

  Below is the model code for the used methodology:-

```
auth.createUserWithEmailAndPassword(email, password)
        .addOnCompleteListener(this) { task ->
            if (task.isSuccessful) {
                // Sign in success, update UI with the signed-in user's information
                Log.d(TAG, "createUserWithEmail:success")
                val user = auth.currentUser
                updateUI(user)
            } else {
                // If sign in fails, display a message to the user.
                Log.w(TAG, "createUserWithEmail:failure", task.exception)
                Toast.makeText(baseContext, "Authentication failed.",
                        Toast.LENGTH_SHORT).show()
                updateUI(null)
            }
            // ...
        }
```

- **LOGIN METHODOLOGY USED:**

CODE FOR THE LOGIN METHODOLY USED

```
mAuth.signInWithEmailAndPassword(emailid, pswd)
            .addOnCompleteListener(MainActivity.this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    if (task.isSuccessful()) {
                        FirebaseUser user = mAuth.getCurrentUser();
                        String uid = user.getUid();
                        final DatabaseReference emid =
mDatabase.child("deptCheck").child(uid);
                        emid.addValueEventListener(new ValueEventListener() {
                            @Override
                            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

                                String check = dataSnapshot.getValue().toString();
                                if (check.equals("0")){
                                    startActivity(new
Intent(MainActivity.this,StudAccount.class));
                                }
                                else{
                                    startActivity(new
Intent(MainActivity.this,clubAcc.class));
                                }
                            }

                            @Override
                            public void onCancelled(@NonNull DatabaseError databaseError) {

                            }

                        });


                    } else {
                        Toast.makeText(MainActivity.this, "Authentication failed.",
Toast.LENGTH_SHORT).show();
                    }
                }
            });
}
```

2. Member Management System: manages members in the back-end system and personal profile adjustment for front-end users.

3. News Management System: manages information of event news in the back-end system and show it in the front-end.

4. Event Management System: a system which manages information of an event.

o Show Event: a system which shows event in the interface.

o Add Event: a system which allows administrator to be able to add a new event into the database.

o Edit Event: a system which allows administrator to edit event information in the database.

o Delete Event: a system which allows administrator to be able to delete event

information in the database.

o Event Registration System: a system which allows member to be able to reserve their seats in the upcoming events.

- MODEL CODE FOR SHOW EVENT METHODOLGY:-

```
• @Override
  protected void onStart() {
      super.onStart();
      FirebaseRecyclerAdapter<Model, ViewHolder> firebaseRecyclerAdapter =
              new FirebaseRecyclerAdapter<Model, ViewHolder>(
                      Model.class,
                      R.layout.row,
                      ViewHolder.class,
                      mRef
              ) {
                  @Override
                  protected void populateViewHolder(ViewHolder viewHolder, Model model, int i)
  {

  viewHolder.setDetails(getApplicationContext(),model.getName(),model.getDescription(),model.ge
  tEvent_url(),model.getLogo_url(),model.getRegistration_Link());
                  }
              };

      mRecyclerView.setAdapter(firebaseRecyclerAdapter);
  }

• public void setDetails(Context ctx,String title, String description, String image,String
  Logo,String Registration_Link){

      TextView mTitleTv= mView.findViewById(R.id.rTitleTv);
      TextView mDetailTv = mView.findViewById(R.id.rDescriptionTv);
      TextView mRegistration = mView.findViewById(R.id.rRegistration);
      ImageView mImageIv= mView.findViewById(R.id.rImageView);
      ImageView mLogoTv= mView.findViewById(R.id.rLogoView);
      mTitleTv.setText(title);
      mDetailTv.setText("About Event:- "+description);
      mRegistration.setText( Registration_Link);
      Picasso.with(ctx).load(image).into(mImageIv);
      Picasso.with(ctx).load(Logo).into(mLogoTv);
  }
```

- **MODEL CODE FOR ADDING EVENT METHODOLGY:-**

```
FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
    String uid = user.getUid();
    database = FirebaseDatabase.getInstance();
    mDatabase = database.getReference();
    final DatabaseReference emid = mDatabase.child("club").child(uid);
    ///
    final DatabaseReference ch1 = mDatabase.child("club").child("eventdisplay");
    final DatabaseReference ch2 = emid.child("logo_url");
    //String logou;


    ///
    final DatabaseReference emid2 = emid.child("events");
    String eventName1 = String.valueOf(System.currentTimeMillis());
    final DatabaseReference events = emid2.child(eventName1);
    //
    final DatabaseReference evedis = ch1.child(eventName1);
    //
    final DatabaseReference logo = events.child("event_url");
    final DatabaseReference logo1 = evedis.child("event_url");
    events.child("Description").setValue(description);
    //
    evedis.child("Description").setValue(description);
    events.child("Registration_Link").setValue(googleFormLink);
    evedis.child("name").setValue(eventName);
    events.child("name").setValue(eventName);
        evedis.child("EventUID").setValue(eventName1);
        events.child("EventUID").setValue(eventName1);

    ch2.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            logou = dataSnapshot.getValue().toString();
            evedis.child("Logo_url").setValue(logou);
            events.child("Logo_url").setValue(logou);
        }

        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {
            logou = "error";
        }
    });


    evedis.child("Registration_Link").setValue(googleFormLink);
    mStorageRef = FirebaseStorage.getInstance().getReference(uid);
    mStorageRe = mStorageRef.child("Events");

    final StorageReference fileReference = mStorageRe.child(eventName + "." +
getFileExtension(mImageUri));
    mUploadTask = fileReference.putFile(mImageUri)
            .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
                    //Toast.makeText(clubSignup.this, "Upload successful",
Toast.LENGTH_SHORT).show();
                    ////
                    fileReference.getDownloadUrl().addOnSuccessListener(new
OnSuccessListener<Uri>() {
                        @Override
                        public void onSuccess(Uri uri) {
                            logo.setValue(uri.toString());
                            logo1.setValue(uri.toString());
                        }
                    });
                    ////


//logo.setValue(taskSnapshot.getMetadata().getReference().getDownloadUrl().toString());
                    Toast.makeText(Event.this, "CLUB EVENT SUCCESSFULLY REGISTERED!!",
Toast.LENGTH_LONG).show();
                    //FirebaseAuth.getInstance().signOut();
                    startActivity(new Intent(Event.this, clubAcc.class));
                }
            })
            .addOnFailureListener(new OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception e) {
                    Toast.makeText(Event.this, e.getMessage(), Toast.LENGTH_SHORT).show();
                }
            });
}
}
```

- **MODEL CODE FOR EDDITING THE EVENTS BY CLUBS:-**

```
save.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String description= desc.getText().toString();
        String googleFormLink= link.getText().toString();
        String eventName= name.getText().toString();
        if (TextUtils.isEmpty(description) || TextUtils.isEmpty(googleFormLink)||
TextUtils.isEmpty(eventName)){
            Toast.makeText(EditEvent.this, "Fill up the details completely!!",
Toast.LENGTH_SHORT).show();
        }
        else {
            if (!URLUtil.isValidUrl(googleFormLink)) {
                Toast.makeText(EditEvent.this, "Enter the correct registration link with https or
http initials.", Toast.LENGTH_SHORT).show();
            }
            else{
            if (mImageUri != null) {
                mStorageRef = FirebaseStorage.getInstance().getReference(uid);
                mStorageRe = mStorageRef.child("Events");
                String newEventName = String.valueOf(System.currentTimeMillis());
                final StorageReference fileReference = mStorageRe.child(newEventName + "." +
getFileExtension(mImageUri));
                mUploadTask = fileReference.putFile(mImageUri)
                        .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
                            @Override
                            public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
                                //Toast.makeText(clubSignup.this, "Upload successful",
Toast.LENGTH_SHORT).show();
                                ////
                                fileReference.getDownloadUrl().addOnSuccessListener(new
OnSuccessListener<Uri>() {
                                    @Override
                                    public void onSuccess(Uri uri) {
                                        emid.child("event_url").setValue(uri.toString());
                                        emid2.child("event_url").setValue(uri.toString());
                                    }
                                });

                            }
                        })
                        .addOnFailureListener(new OnFailureListener() {
                            @Override
                            public void onFailure(@NonNull Exception e) {
                                Toast.makeText(EditEvent.this, e.getMessage(),
Toast.LENGTH_SHORT).show();
                            }
                        });
            }

            emid.child("Description").setValue(description);
            emid.child("Registration_Link").setValue(googleFormLink);
            emid.child("name").setValue(eventName);
            emid2.child("Description").setValue(description);
            emid2.child("Registration_Link").setValue(googleFormLink);
            emid2.child("name").setValue(eventName);
            Toast.makeText(EditEvent.this, "Information saved successfully",
Toast.LENGTH_SHORT).show();
            startActivity(new Intent(EditEvent.this, clubAcc.class));

        }
        }



    }
});
```

5. Administrator Management System: manages the right of administrators according to the priority.

- **SECURITY METHODOLGY USED:-**

SECURITY CODE FOR USER ACCESS:

```
// Grants a user access to a node matching their user ID
service firebase.storage {
  match /b/{bucket}/o {
    // Files look like: "user/<UID>/path/to/file.txt"
    match /user/{userId}/{allPaths=**} {
      allow read, write: if request.auth.uid == userId;
    }
  }
}
```

SECURITY CODE FOR SUPER ADMIN ACCESS:

```
// Access to files through Firebase Storage is completely disallowed.
// Files may still be accessible through Google App Engine or GCS APIs.
service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write: if false;
    }
  }
}
```

✓ ADVANTAGES

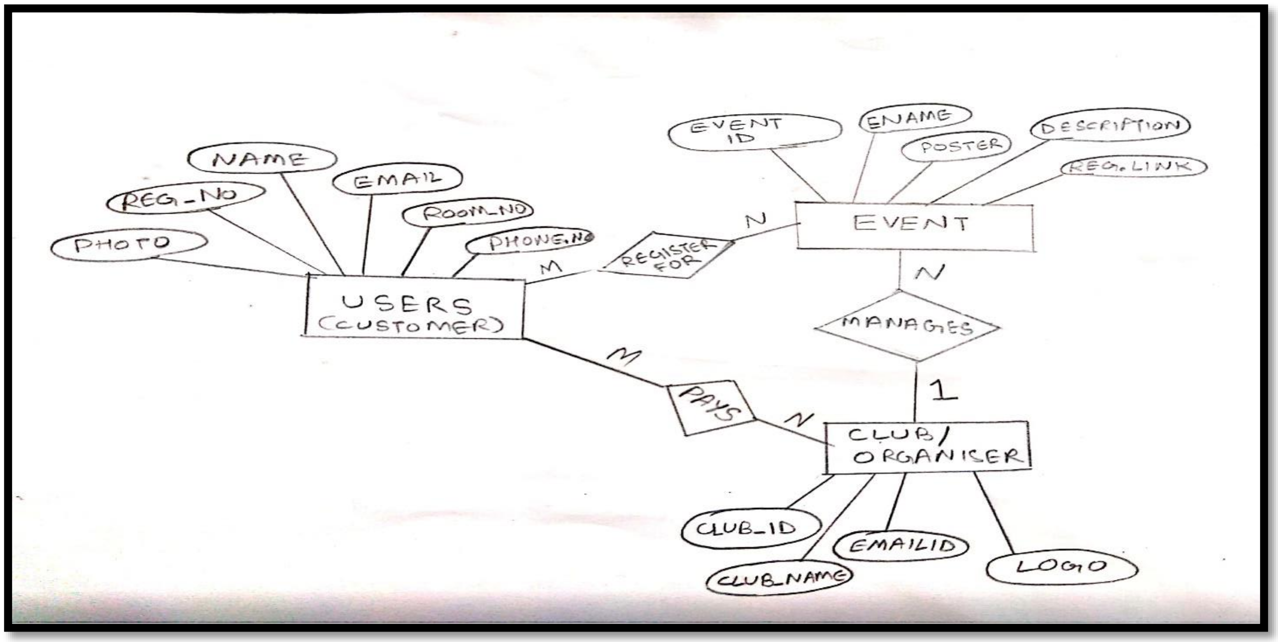
To avoid all the disadvantages in existing model we are creating an event manager app which is both user and organizer favorable. The interface is easy to use. To enhance our DBMS performance we are going to use firebase storage cloud which has following benefits:

- It uses JSON (Javascript Object Notation) storage, means no barrier between data and objects
- simple serialization of app state
- 3-way data binding via Angularfire
- minimal setup
- easy access to data, files, auth, and more
- no server infrastructure needed to power apps with data
- massive storage size potential
- real time
- pretty much the most advanced hosted BaaS solution
- highly secure
- serverless

✓ DISADVANTAGES

- In current app there is no security key deployed yet to cross examine a spammer under clubs section. It can be deployed by giving a unique key to each club via E-mail followed by verification.
- Since we are using firebase as a backend service for our app it will take time to download images from the storage cloud if users load increases on server.
- It do not schedule an event rather it only helps to promote it and also help in registration process.

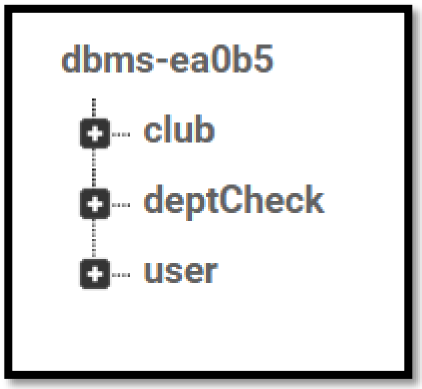
ER DIAGRAM



EXPLANATION OF ER DIAGRAM CORRESPONDING TO OUR DATABASE:

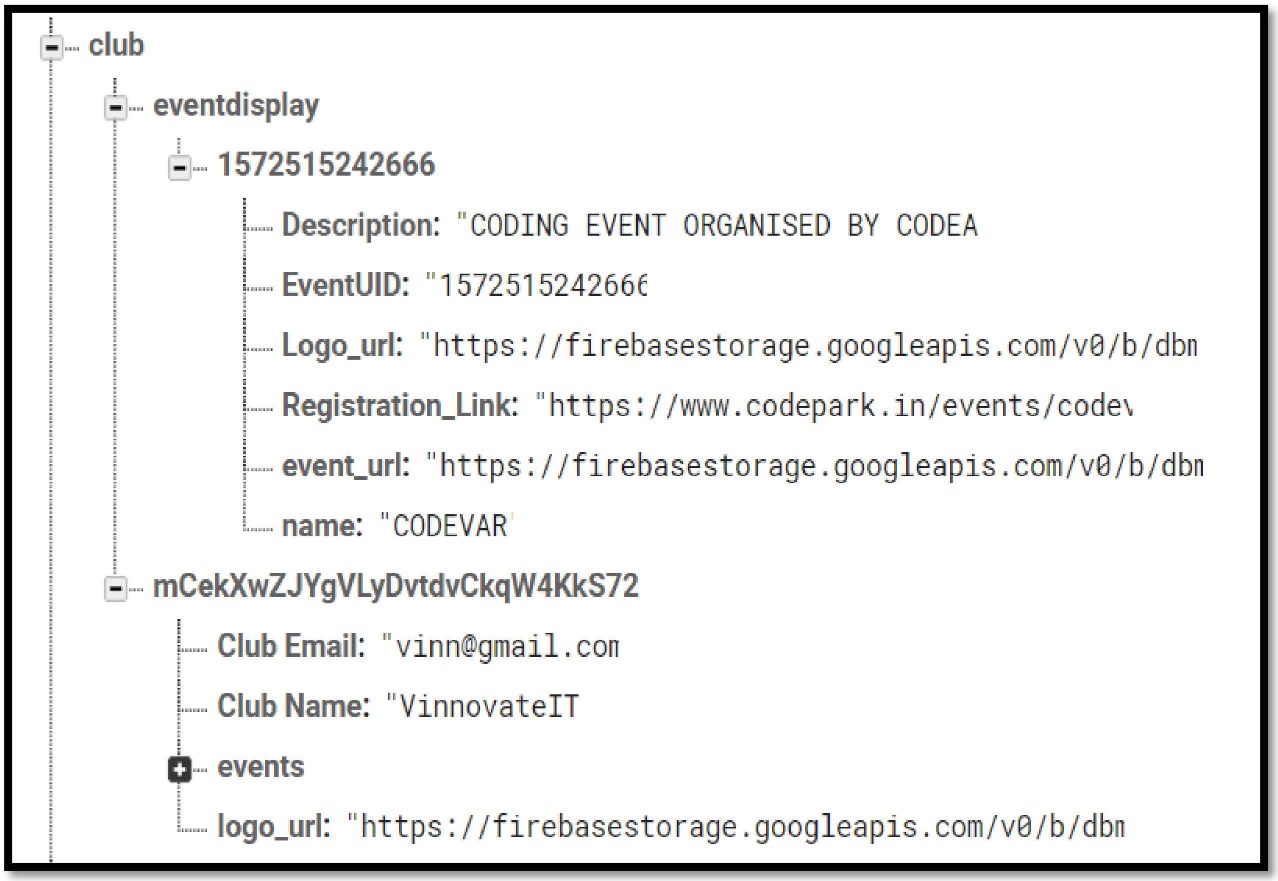
- **ATTRIBUTES INVOLVED:**

Whenever a new user registers himself either into club or student category his details get into the respective attribute in a real time.



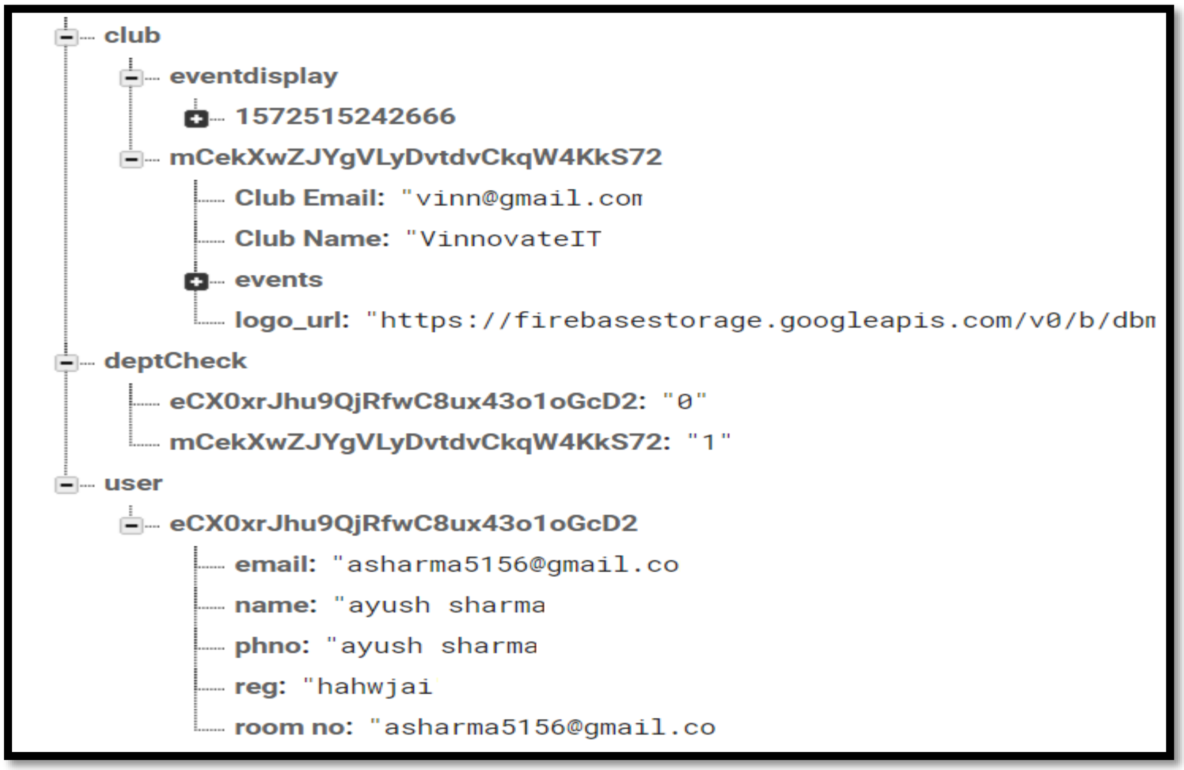
```
dbms-ea0b5
├─ club
├─ deptCheck
└─ user
```

- **RELATION BETWEEN CLUB AND EVENT:**



```
club
  eventdisplay
    1572515242666
      Description: "CODING EVENT ORGANISED BY CODEA
      EventUID: "157251524266(
      Logo_url: "https://firebasestorage.googleapis.com/v0/b/dbn
      Registration_Link: "https://www.codepark.in/events/code\
      event_url: "https://firebasestorage.googleapis.com/v0/b/dbn
      name: "CODEVAR
  mCekXwZJYgVLyDvtdvCkqW4KkS72
    Club Email: "vinn@gmail.con
    Club Name: "VinnovateIT
    events
    logo_url: "https://firebasestorage.googleapis.com/v0/b/dbn
```

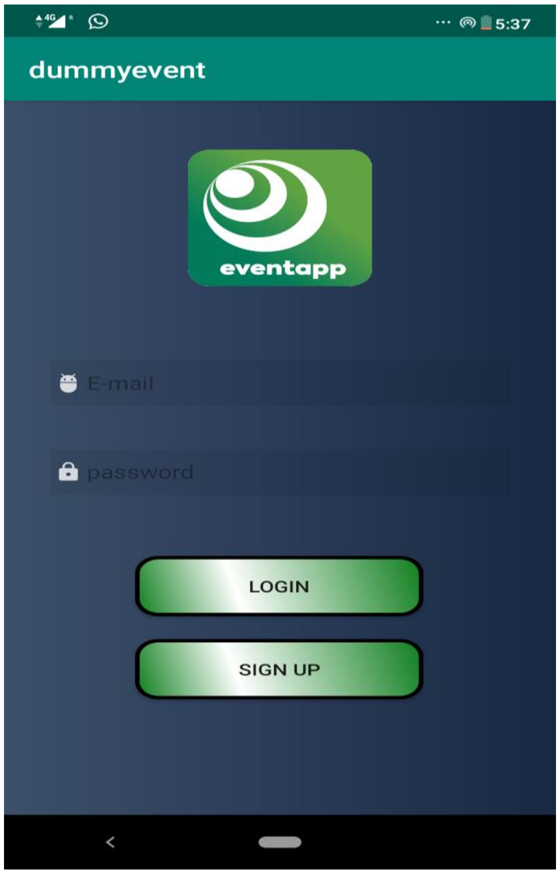
- **RELATION BETWEEN USER AND CLUB**



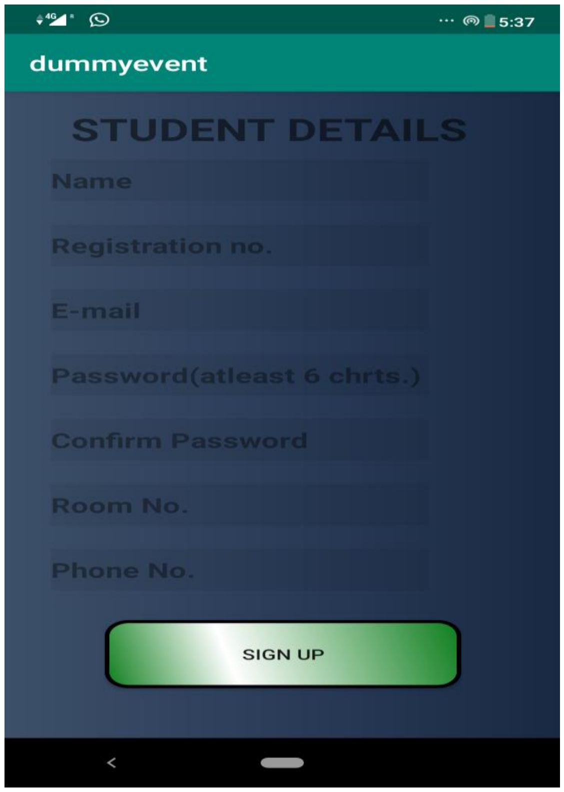
```
club
    eventdisplay
        1572515242666
    mCekXwZJYgVLyDvtdvCkqW4KkS72
        Club Email: "vinn@gmail.com
        Club Name: "VinnovateIT
        events
        logo_url: "https://firebasestorage.googleapis.com/v0/b/dbm
deptCheck
    eCX0xrJhu9QjRfwC8ux43o1oGcD2: "0"
    mCekXwZJYgVLyDvtdvCkqW4KkS72: "1"
user
    eCX0xrJhu9QjRfwC8ux43o1oGcD2
        email: "asharma5156@gmail.co
        name: "ayush sharma
        phno: "ayush sharma
        reg: "hahwjai
        room no: "asharma5156@gmail.co
```

<u>SCREENSHOTS</u>

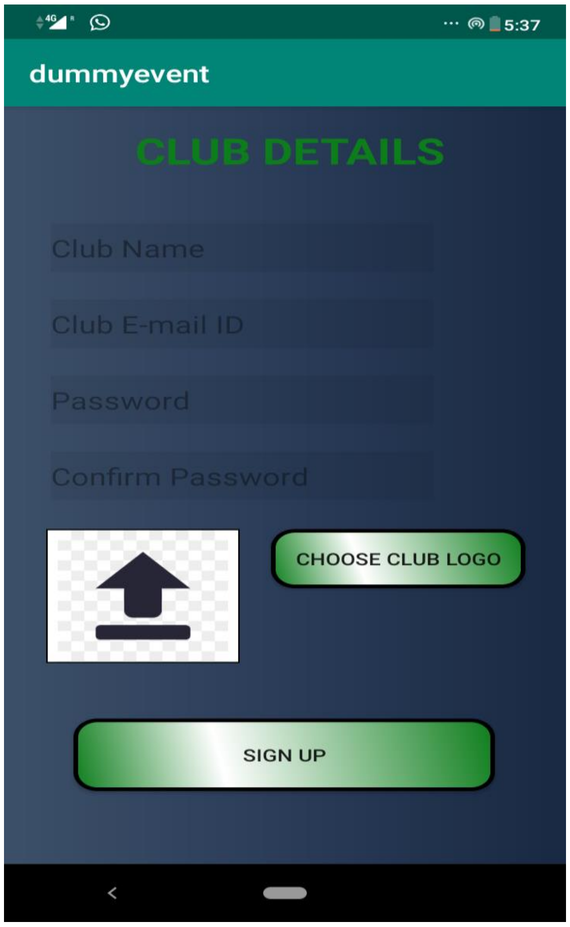
- **LOGIN ACTIVITY**



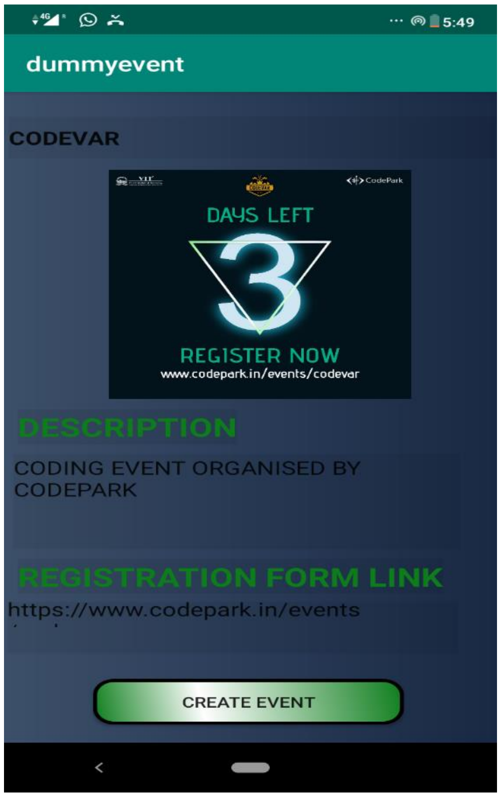
- **STUDENT SIGNUP ACTIVITY**



- **CLUB SIGNUP ACTIVITY**



- **EVENT CREATION ACTIVITY**


dummyevent

CODEVAR

DESCRIPTION

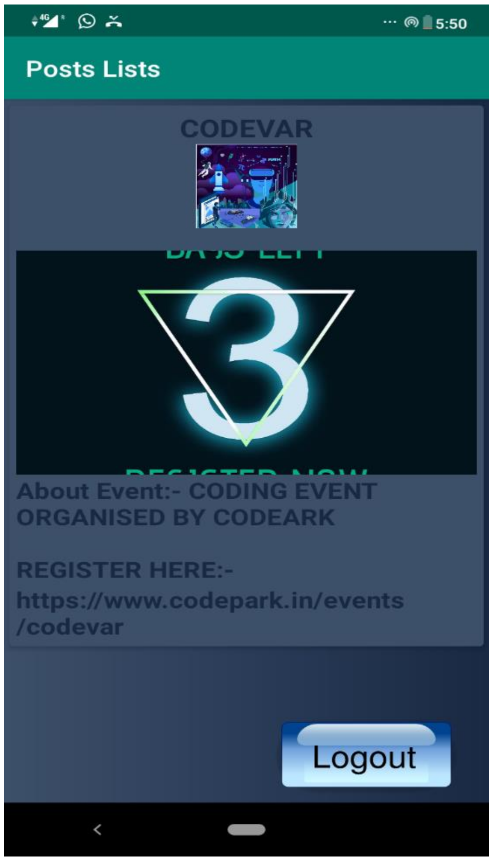
CODING EVENT ORGANISED BY CODEPARK

REGISTRATION FORM LINK

https://www.codepark.in/events

CREATE EVENT

- **STUDENT ACCOUNT AND REGISTRATION ACTIVITY**


Posts Lists

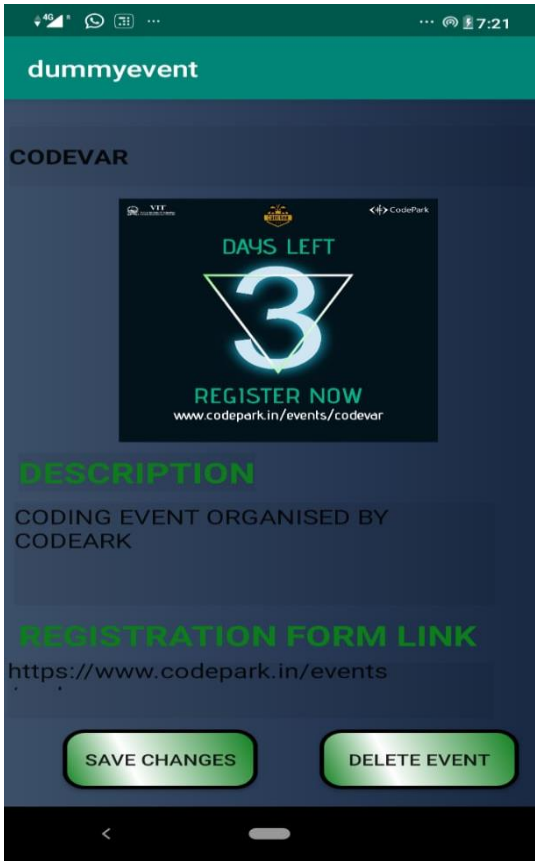
CODEVAR

About Event:- CODING EVENT ORGANISED BY CODEARK

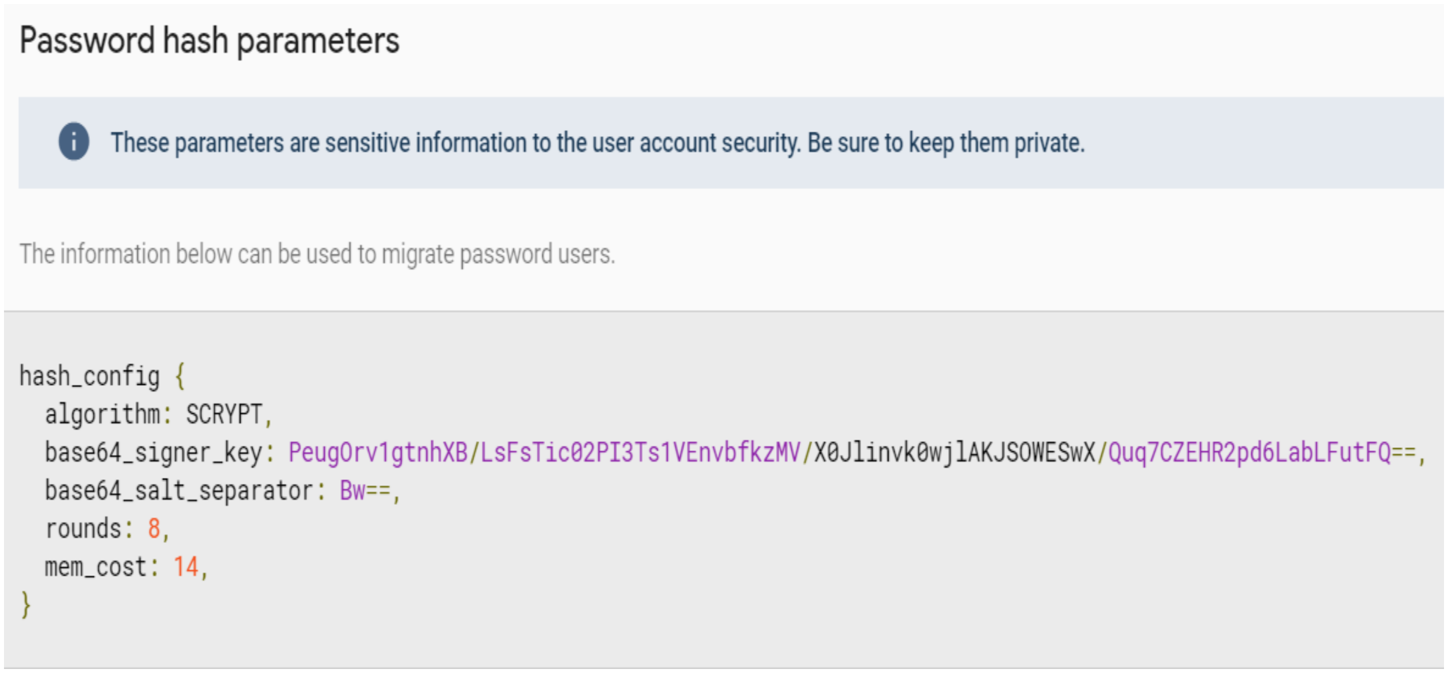
REGISTER HERE:- https://www.codepark.in/events/codevar

Logout

- **EDIT AND DELETE EVENT ACTIVITY**



<u>SECURITY RULES FOR ENCRYPTING PASSWORD</u>



Password hash parameters

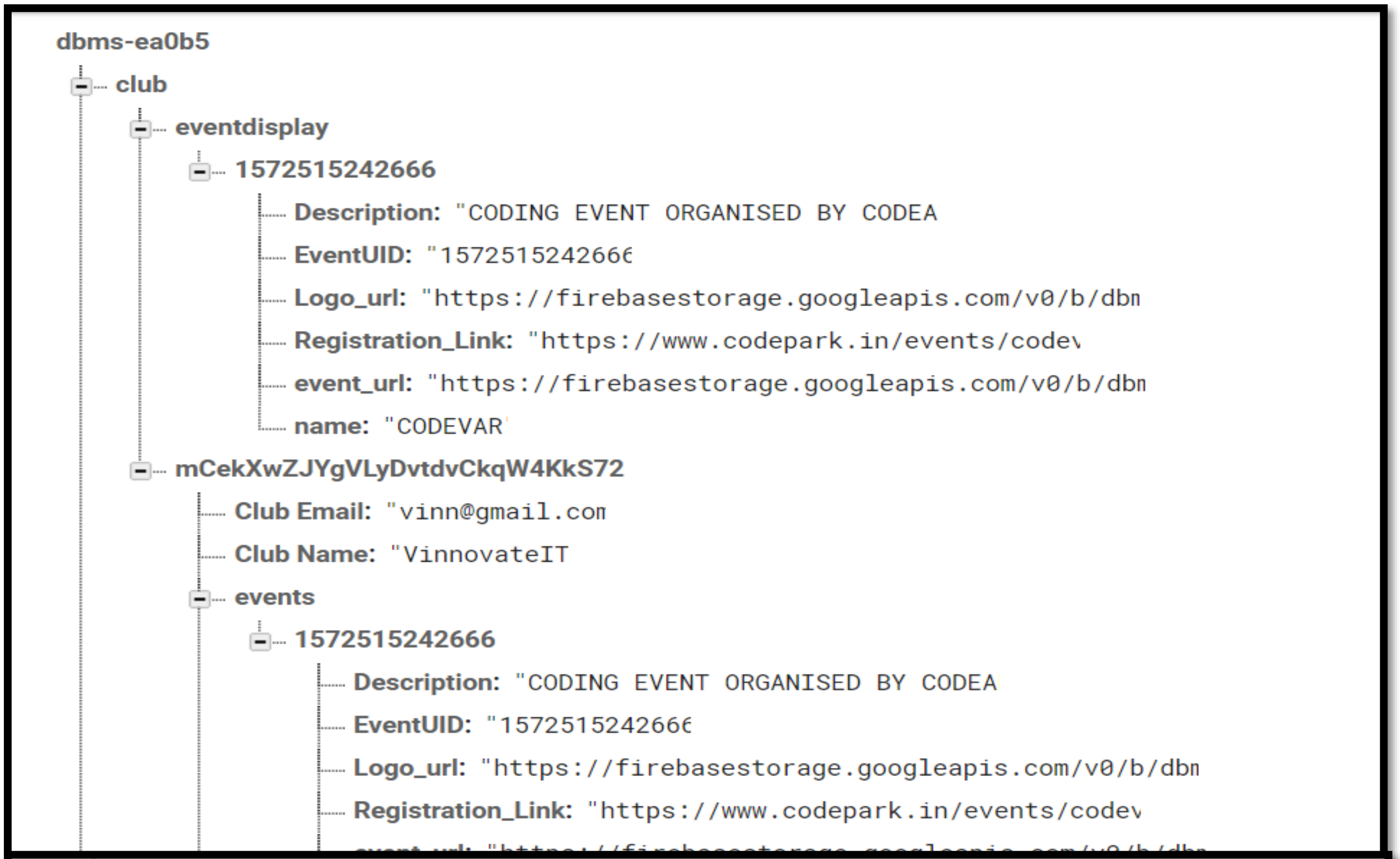
These parameters are sensitive information to the user account security. Be sure to keep them private.

The information below can be used to migrate password users.

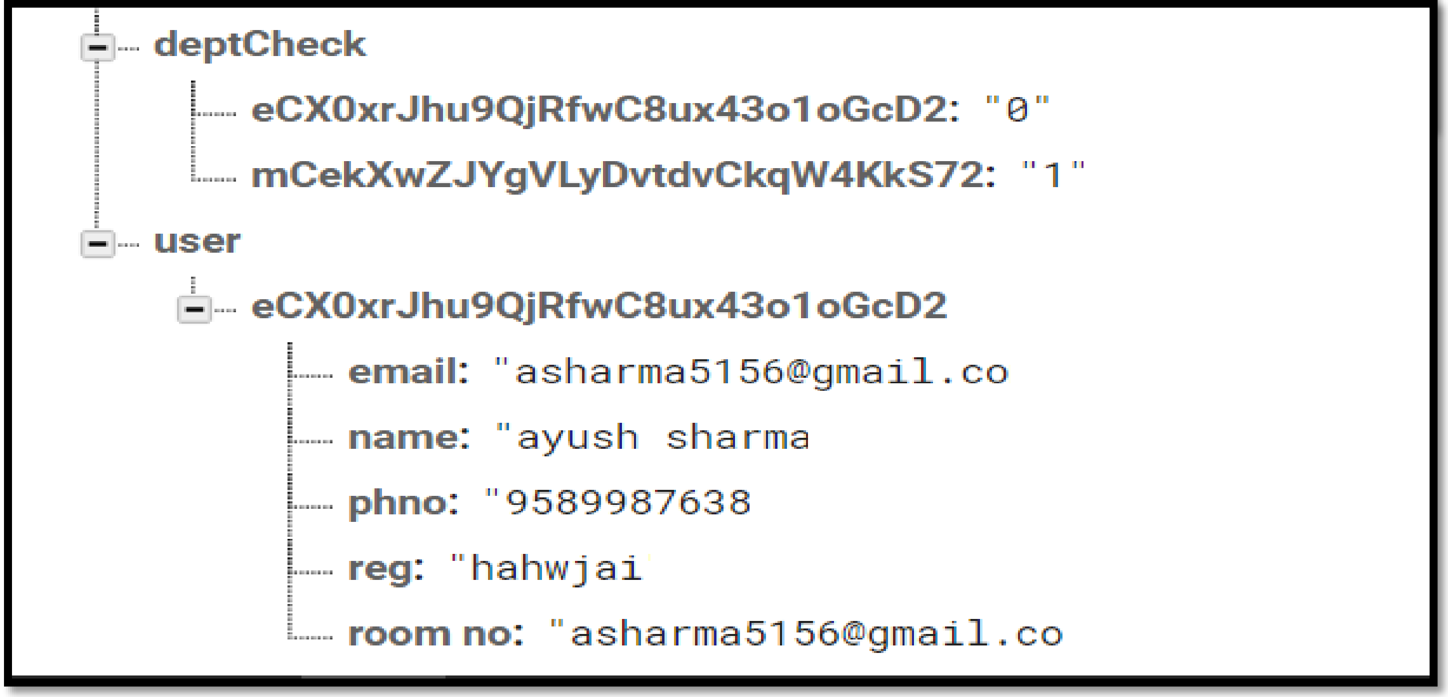
```
hash_config {
  algorithm: SCRYPT,
  base64_signer_key: PeugOrv1gtnhXB/LsFsTic02PI3Ts1VEnvbfkzMV/X0Jlinvk0wjlAKJSOWESwX/Quq7CZEHR2pd6LabLFutFQ==,
  base64_salt_separator: Bw==,
  rounds: 8,
  mem_cost: 14,
}
```

DATABASE SCREENSHOTS

Json file screenshots:

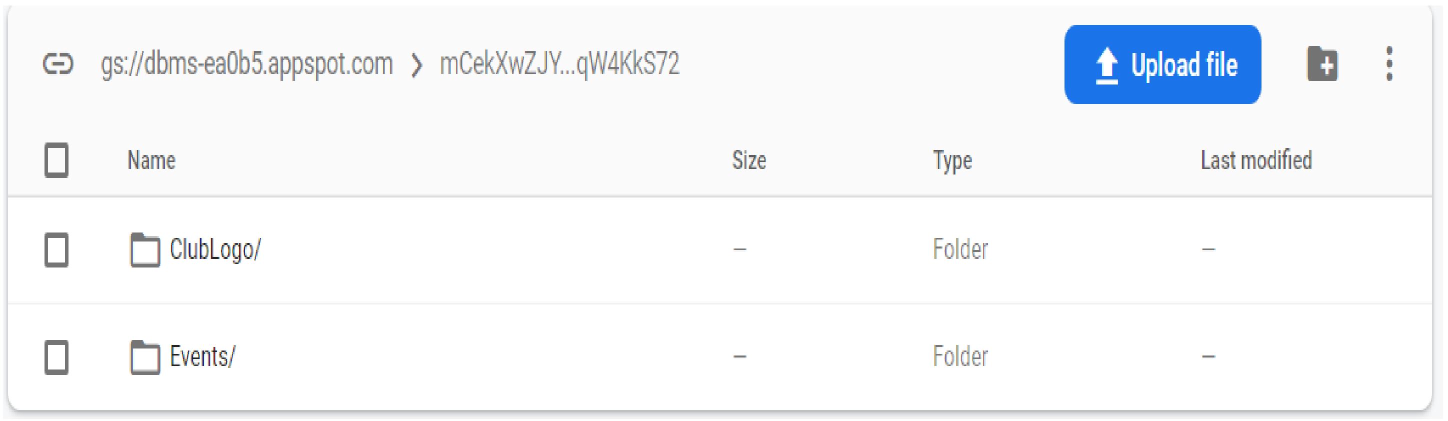


```
dbms-ea0b5
  club
    eventdisplay
      1572515242666
        Description: "CODING EVENT ORGANISED BY CODEA
        EventUID: "1572515242666
        Logo_url: "https://firebasestorage.googleapis.com/v0/b/dbm
        Registration_Link: "https://www.codepark.in/events/codev
        event_url: "https://firebasestorage.googleapis.com/v0/b/dbm
        name: "CODEVAR
    mCekXwZJYgVLyDvtdvCkqW4KkS72
      Club Email: "vinn@gmail.com
      Club Name: "VinnovateIT
      events
        1572515242666
          Description: "CODING EVENT ORGANISED BY CODEA
          EventUID: "1572515242666
          Logo_url: "https://firebasestorage.googleapis.com/v0/b/dbm
          Registration_Link: "https://www.codepark.in/events/codev
          event_url: "https://firebasestorage.googleapis.com/v0/b/dbm
```



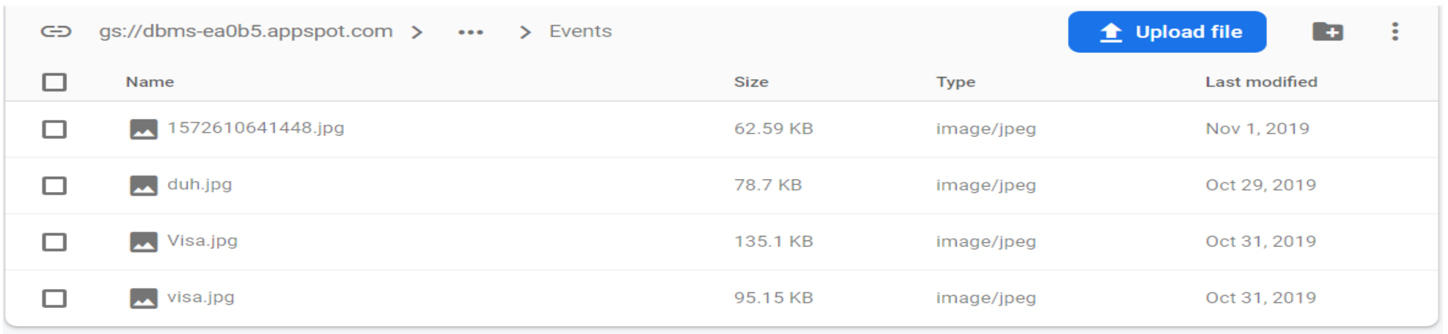
```
  deptCheck
    eCX0xrJhu9QjRfwC8ux43o1oGcD2: "0"
    mCekXwZJYgVLyDvtdvCkqW4KkS72: "1"
  user
    eCX0xrJhu9QjRfwC8ux43o1oGcD2
      email: "asharma5156@gmail.co
      name: "ayush sharma
      phno: "9589987638
      reg: "hahwjai
      room no: "asharma5156@gmail.co
```

IMAGE STORAGE IN CLOUD:-

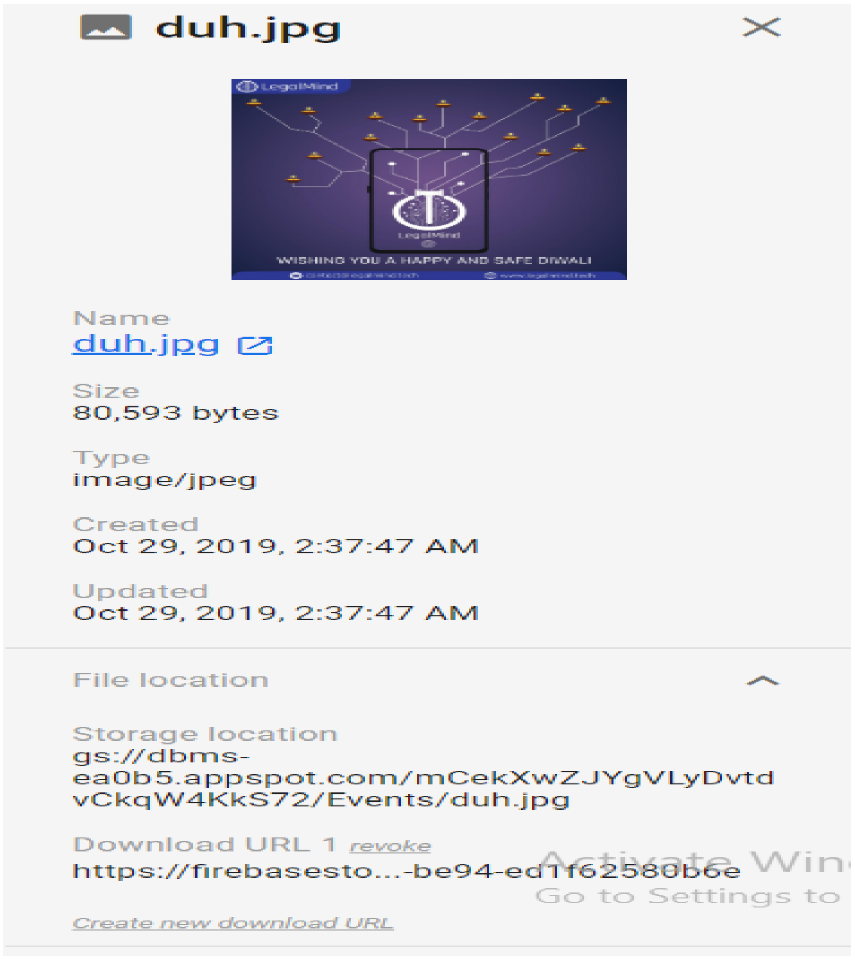


Name	Size	Type	Last modified
ClubLogo/	–	Folder	–
Events/	–	Folder	–

IMAGES OF EVENT POSTERS:-



Name	Size	Type	Last modified
1572610641448.jpg	62.59 KB	image/jpeg	Nov 1, 2019
duh.jpg	78.7 KB	image/jpeg	Oct 29, 2019
Visa.jpg	135.1 KB	image/jpeg	Oct 31, 2019
visa.jpg	95.15 KB	image/jpeg	Oct 31, 2019



Name
duh.jpg

Size
80,593 bytes

Type
image/jpeg

Created
Oct 29, 2019, 2:37:47 AM

Updated
Oct 29, 2019, 2:37:47 AM

File location

Storage location
gs://dbms-ea0b5.appspot.com/mCekXwZJYgVLyDvtdvCkqW4KkS72/Events/duh.jpg

Download URL 1 revoke
https://firebasesto...-be94-ed1f62580b6e

Create new download URL

ANALYSIS

- ## PROJECT SCOPE AND DESCRIPTION

Event promoter app is meant to reduce a tedious job of promoting and advertising events in colleges (VIT). This will replace other options like desk duties in colleges for promoting club events. This will reduce money scams, will save time and space and will also help the club to promote on a larger e-platform.

This application focuses on solving problems of event registration and management and also providing news, information of events, and project ideas. First of all, users will be able to reserve and manage their event participation via this application. Additionally, this application provides significant information and news of many interesting events from the event provider.

- ## HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Interfaces

The application is intended to be a stand-alone, single-user system. The application will run on an Android mobile device or an Android emulator. No further hardware devices or interfaces will be required.

Software Interfaces

The software will run on the Android operating system, specifically version 4.1 (Jellybean) and above.

Communications Interfaces

The application shall communicate with the various databases and software services via API function calls. Because the application will be written in Java, Java functions will make these calls to the APIs. The exact formats and protocols for incoming and outgoing messages should be abstracted by the APIs.

- ## APPLICATION'S DETAIL ANALYSIS

Our event app functionality can be explained with a series of application functions. It provides a platform for the clubs or event organizers to promote their event on our app. This is how it works:
A club can register in our app under which it will be asked to enter some basic information about the club and a club logo. Once the club is registered it can add and modify events. They also have options to delete the event. Under adding event section they can add the name of the event, its description, Registration URL and they can also upload a poster of event which will be displayed in students account.
Similarly a student or a user can register himself under student section by filling up basic details. After registration if a user login he will be directed to an event page where all the events of all the event organizers will be displayed in an image card which will contain name of the event, club logo, event poster description of event and registration link of the event. When users click on the event image card they will be directed to the registration page of the app and there they can successfully register the event.

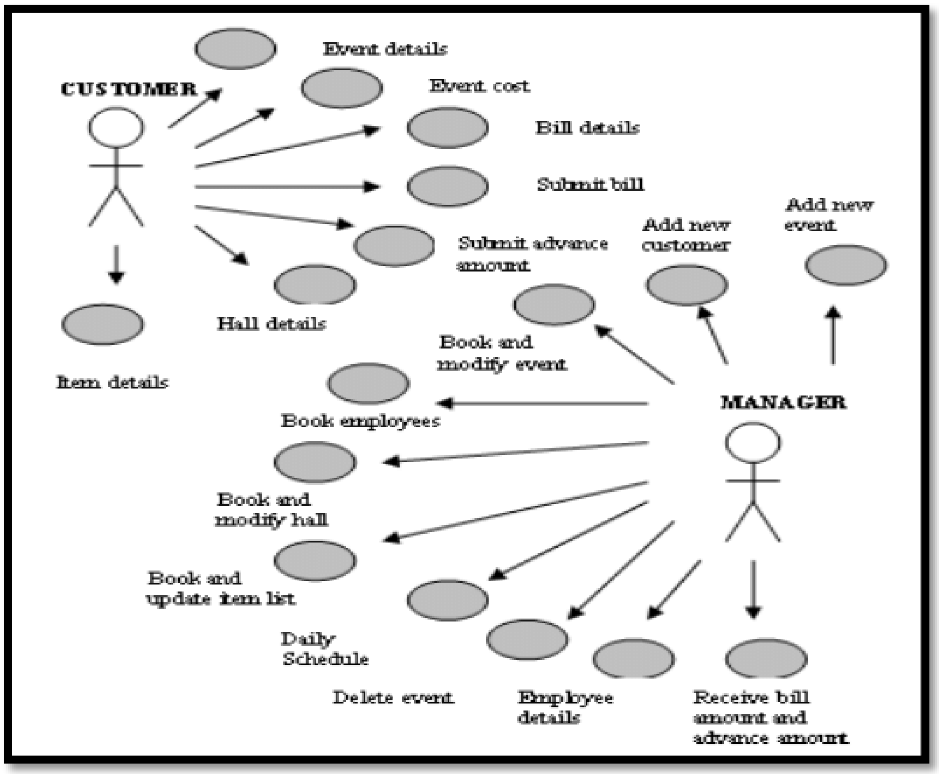
- ## APP BACKEND AND DATABASE ANALYSIS

  We are using firebase for our app as a real time backend. We used the firebase cloud computing for storing data and images that are being uploaded in the app by the club or event organizers. All the club and students details are storing in firebase database. The database is based on mongo db. And all the data is represented in the form of Json trees. When the club and chapters add any poster it gets store d into firbase storage which is a cloud where images get stores in real time.

CONCLUSION

This application focuses on solving the problems of propagating news and information, also developing the application to satisfy users' desire. Moreover, this application will provide significant information of events in order to be easily reached by users and will be able to manage their event participation. Additionally, this application can be used from everywhere, anytime. More importantly, integrating unique user id will provide more convenience to handle events because it able to complete authentication in one step.

FUTURE WORK



For future work we are planning to convert our event promotion and registering app into a full-fledged event management app which will bring one more organization into play “VIT” (or the organizing source or the organization/college) along with user and clubs. For future use we include various features in our app such as hall booking and availability, Bill payments, catering, event scheduling etc.

We will bring the administration or the college into our app through which we will give them the facility to permit and provide halls for organizing events. They can accept the payments from customers (clubs) and can assign them hall according to their demands and in turn the clubs and chapters will take registration and bill payment on the same app.

REFERENCES

[1]. System and method for creation and management of advertising inventory using metadata
- R Steelberg, C Steelberg - US Patent App.

[2]. Advertising allocation and impact of advertising on event ticket sales: Which product, where, and when
- MY Kang - International Journal of Market Research

[3]. Platform for mobile advertising and persistent microtargeting of promotions
- RC Lewis, GD Mandyam

[4]. Seamless authentication for an application development platform
- MD Wood

[5]. A statistical approach to participant selection in location-based social networks for offline event marketing
- Y Liu, A Liu, X Liu, X Huang

[6]. Displaying mobile advertising based on determining user's physical activity from mobile device sensor data
- T Phan

[7]. Apparatus for exchanging advertising media and method for the same
- LEE Ang

[8]. Method and intelligent system for generating a predictive outcome of a future event
- E Heredia, D Sivalingam, RAI Sapna

[9]. https://stackoverflow.com/

[10]. https://firebase.google.com/docs/android