4. IMPLEMENTASI SISTEM

Pada bab ini akan dijelaskan implementasi sistem yang telah dirancang dari bab sebelumnya. Implementasi ini dibagi menjadi dua bagian yaitu implementasi pada aplikasi *mobile* berbasis android dan aplikasi *web*. Aplikasi *mobile* akan digunakan oleh *user* sebagai pencari maupun penyedia jasa yang dibuat menggunakan Android Studio dengan bahasa pemrograman java. Aplikasi *web* akan digunakan oleh *admin* untuk mengelola data yang dibuat dengan javascript

4.1. Implementasi Awal Pembuatan Aplikasi

Pembuatan aplikasi android sangat dibutuhkan beberapa komponen-komponen untuk dapat menjalankan aplikasi dengan baik seperti *permission* dan *gradle*. Berikut adalah penjelasannya:

4.1.1. Android Manifest

Android Manifest menggunakan *permission* yang berfungsi untuk menggunakan data dan fitur yang dimiliki oleh android. Berikut adalah implementasi dari Android Manifest.xml dapat dilihat pada segmen 4.1.

Segmen 4.1 Android Manifest

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

4.1.2. Gradle

Gradle adalah *tool* untuk mengotomatisasi pembangunan proyek secara tersistem. Gradle seperti *library* dimana pengguna tinggal menggunakannya secara otomatis. Berikut adalah gradle yang digunakan dapat dilihat pada Segmen 4.2.

Segmen 4.2 Gradle

```
//circular image
compile 'de.hdodenhof:circleimageview:2.2.0'
//google maps
compile 'com.google.android.gms:play-services:11.0.2'
compile 'com.google.android.gms:play-services-maps:11.0.2'
compile 'com.google.android.gms:play-services-location:11.0.2'
//firebase
compile 'com.google.firebase:firebase-core:11.0.2'
compile 'com.google.firebase:firebase-auth:11.0.2'
compile 'com.google.firebase:firebase-database:11.0.2'
compile 'com.google.firebase:firebase-storage:11.0.2'
compile 'com.firebaseui:firebase-ui:0.6.2'
compile 'com.google.android.gms:play-services-auth:11.0.2'
compile 'com.google.firebase:firebase-messaging:11.0.2'
compile 'com.squareup.retrofit2:retrofit:2.3.0'
compile 'com.squareup.retrofit2:converter-gson:2.3.0'
//crop & image
compile 'com.theartofdev.edmodo:android-image-cropper:2.6.+'
compile 'com.squareup.picasso:picasso:2.5.2'
//send email
compile files('libs/activation.jar')
compile files('libs/additionnal.jar')
compile files('libs/mail.jar')
```

4.2. Implementasi Aplikasi Android

Dalam sistem ini, aplikasi *mobile* ini digunakan oleh pencari jasa dan penyedia jasa. Pada bagian ini akan dijelaskan implementasi program pada fitur-fitur yang penting saja. Berikut adalah Tabel 4.1 yang berisi fungsi-fungsi dari fitur aplikasi.

Tabel 4.1 Tabel Implementasi pengguna aplikasi android

No.	Nama Fitur	Segmen	*Keterangan*	Pemilik Fitur	
				Pencari	Penyedia
1.	*Login*	4.3	Digunakan untuk membatasi penggunaan aplikasi	V	V
2.	Registrasi	4.4	Digunakan untuk melakukan pendaftaran akun baru	V	V
3.	Profil	4.5	Digunakan untuk melihat informasi tentang pengguna	V	V
4.	Pencarian Proyek	4.6	Digunakan untuk pencarian proyek yang dibuat oleh pencari jasa		V

5.	Pembuatan Proyek	4.7	Digunakan untuk pembuatan proyek untuk mencari penyedia jasa	V	
6.	*Project List*	4.8	Digunakan untuk menampilkan data-dataproyek apa saja yang sedang atau sudah dijalankan	V	V
7.	*Chat*	4.9	Digunakan untuk mengirim pesan antara pencari jasa dan penyedia jasa	V	V
8.	*Bid*	4.10	Digunakan untuk melakukan penawaran harga oleh penyedia jasa sesuai dengan maksimum budget yang dimiliki oleh pencari jasa		V
9.	Konfirmasi Pembayaran	4.11	Digunakan untuk mengirim bukti transaksi kepada pihak admin untuk meminta konfirmasi agar dapat dilanjutkan ke proses kerja		V
10.	Portofolio *Watermark*	4.12	Digunakan untuk penyedia jasa mengupload gambar portofolio mereka dan menampilkannya dengan watermark	V	
11.	*Rating* dan *Review*	4.13	Digunakan untuk pencari jasa melakukan penilaian terhadap kualitas kerja penyedia jasa		V
12.	Notifikasi	4.14	Digunakan untuk mendapatkan notifikasi	V	V

Dalam sistem ini, *website* ini digunakan oleh admin untuk mengelola data-data seperti pekerjaan, proyek, dan transaksi. Berikut adalah Tabel 4.2 yang berisi fungsi-fungsi dari fitur *website*.

Tabel 4.2 Tabel implementasi *website admin*

No	Nama Fitur	Segmen	*Keterangan*
1.	*Manage* Jasa	4.15	Digunakan untuk mengelola data-data pekerjaan
2.	*Manage* Proyek	4.16	Digunakan untuk mengelola data-data proyek yang dibuat oleh pencari jasa
3	*Manage* Transaksi	4.17	Digunakan untuk mengelola data-data transaksi dari pencari maupun penyedia jasa

4.3. Implementasi Aplikasi Untuk Sisi Pengguna (Android)

Implementasi aplikasi dimulai dengan penerapan aplikasi dari sisi pengguna. Pengguna yang dimaksud adalah pengguna aplikasi yaitu pencari dan penyedia jasa. Pada awal pengguna akan melakukan akses aplikasi pengguna melakukan *login* menggunakan *email* dan *password* dan button untuk menuju halaman registrasi dan lupa *password*. Setelah *login* berhasil maka pengguna memasuki halaman *home*. Setelah itu, pengguna dapat mengakses fitur-fitur yang ada sesuai dengan jenis akun yang dibuat saat registrasi yaitu pencari atau penyedia jasa. Berikut adalah fungsi-fungsi dari fitur:

4.3.1. Login

Login digunakan untuk pengecekan berdasarkan *email* dan *password* pada *database* Firebase. Sebelum melakukan login pengguna harus memverifikasi *email* terlebih dahulu untuk dapat melewati proses *login* ini. Selain itu pengguna harus memasukkan data *email* dan *password* dengan benar. Setelah pengguna berhasil memasukkan *email* dan *password* dengan benar maka akan diarahkan ke

halaman selanjutnya berdasarkan jenis akun yang dibuatnya yaitu sebagai pencari atau penyedia jasa. Untuk lebih jelasnya dapat dilihat pada Segmen 4.3.

Segmen 4.3 *Login*

```
mLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        strEmail = mEmail.getText().toString();
        strPassword = mPassword.getText().toString();
        pd.setMessage("Please Wait ...");
        pd.show();
        if (strEmail.equals("") || strPassword.equals("")){
            Toast.makeText(context,"Email / Password
Kosong",Toast.LENGTH_SHORT).show();
            pd.dismiss();
        }
        else{

mAuth.signInWithEmailAndPassword(strEmail,strPassword).addOnCompleteListener(new
OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    final FirebaseUser user = mAuth.getCurrentUser();
                    if (!task.isSuccessful()){
                        Toast.makeText(context,"Wrong Email /
Password",Toast.LENGTH_SHORT).show();
                        pd.dismiss();
                        mAuth.signOut();
                    }
                    else{
                        try {
                            if(user.isEmailVerified()){
                                userID = user.getUid();
                                Common.currentToken =
FirebaseInstanceId.getInstance().getToken();

mReference.child("Users").child(userID).child("token_id").setValue(Common.currentT
oken);

mReference.child(getString(R.string.users)).child(userID)
                                        .addValueEventListener(new
ValueEventListener() {
                                    @Override
                                    public void onDataChange(DataSnapshot
dataSnapshot) {
                                        if (dataSnapshot.exists()) {

mReference.child("Users").child(userID).child("password").setValue(strPassword);
                                            String account;
                                            account =
dataSnapshot.child(getString(R.string.account)).getValue().toString();
                                            ArrayList<String> mStringList = new
ArrayList<>();
                                            mStringList.add(account);
                                            if (mStringList.contains("Pencari")) {
                                                Toast.makeText(context,
"Authentication Success", Toast.LENGTH_SHORT).show();
                                                startActivity(new Intent(context,
HomePencariActivity.class));
                                                finish();
                                                pd.dismiss();
                                            } else if
(mStringList.contains("Penyedia")) {
                                                Toast.makeText(context,
"Authentication Success", Toast.LENGTH_SHORT).show();
                                                startActivity(new Intent(context,
HomePenyediaActivity.class));
                                                finish();
                                                pd.dismiss();
```

```
                                        }
                                    }
                                }
                                @Override
                                public void onCancelled(DatabaseError
databaseError) {
                                }
                            });
                        }
                        else{
                            Toast.makeText(context,"Verification Your
Email",Toast.LENGTH_SHORT).show();
                            pd.dismiss();
                            mAuth.signOut();
                        }
                    }
                    catch (NullPointerException e){
                    }
                }
            }
        });
    }
}
});
```

4.3.2. Registrasi

Registrasi digunakan untuk pengguna melakukan pendaftaran untuk dapat menggunakan aplikasi ini. Dalam fungsi ini pengguna harus mengisi nama, *email valid*, *password*, jenis akun sebagai pencari atau penyedia jasa dan foto profil yang bersifat *optional*. Saat sukses registrasi setiap pengguna akan memiliki data-data yang sama tetapi semua data nantinya tidak akan dimunculkan seperti contoh adalah data *rating*. *Rating* disini hanya dimunculkan pada akun yang berjenis penyedia jasa karena *rating* dapat diberikan oleh pencari jasa untuk penilaian kepada penyedia jasa yang dipilihnya untuk menyelesaikan pekerjaanya. Untuk lebih jelasnya dapat dilihat pada Segmen 4.4.

Segmen 4.4 Registrasi

```
private void register_new_user(){
    strEmail = mEmail.getText().toString();
    strPassword = mPassword.getText().toString();
    strName = mName.getText().toString();
    strStatus = "My Information";
    iJumlahproyek = 0;
    iJumlahMenang = 0;
    dRating = 0.0;
    if (strEmail.equals("") || strEmail.equals("") || strName.equals("") ||
strAccount.equals("")){
        Toast.makeText(this,"Harap Isi Semua Data",Toast.LENGTH_SHORT).show();
    }
    else {
        firebase_method.register_new_email(strEmail,strPassword);
    }
}

private void setupFirebaseAuthentication(){
    mAuth = FirebaseAuth.getInstance();
    mAuthListener = new FirebaseAuth.AuthStateListener() {
```

```
        @Override
        public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
            FirebaseUser user = mAuth.getCurrentUser();

            if(user!=null){
                userId = mAuth.getCurrentUser().getUid();

                mReference.addListenerForSingleValueEvent(new ValueEventListener()
{
                    @Override
                    public void onDataChange(DataSnapshot dataSnapshot) {

firebase_method.send_new_user_data(strName,strEmail,strPassword,strAccount,strStat
us,iJumlahproyek,iJumlahMenang,0,dRating,"Default","");
                        select_image();
                        Toast.makeText(RegisterActivity.this,"Registration
Success",Toast.LENGTH_SHORT).show();
                        mAuth.signOut();
                    }
                    @Override
                    public void onCancelled(DatabaseError databaseError) {}
                });
                finish();
            }
        }
    };
}
```

4.3.3. Profil

Profil digunakan untuk pencari jasa dapat melihat informasi-informasi mengenai penyedia jasa seperti *record* proyek yang telah diselesaikan dan juga sebaliknya penyedia jasa dapat melihat informasi-informasi yang dimiliki pencari jasa. Namun yang membedakan profil pencari dan penyedia jasa adalah dimana pencari jasa hanya berisikan satu halaman tentang informasi detail pencari jasa tersebut sedangkan penyedia jasa memiliki halaman tambahan dimana untuk menampilkan *rating* dan *review* atas pekerjaan yang pernah diselesaikan. Untuk lebih jelasnya dapat dilihat pada Segmen 4.5.

Segmen 4.5 Profil

```
mReference.child(getString(R.string.users)).child(value)
        .addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                if (dataSnapshot.exists()) {
                    String name, image, jumProyek, information;
                    name =
dataSnapshot.child(getString(R.string.name)).getValue().toString();
                    image =
dataSnapshot.child(getString(R.string.profile_image)).getValue().toString();
                    jumProyek =
dataSnapshot.child(getString(R.string.jumlahproyek)).getValue().toString();
                    information =
dataSnapshot.child(getString(R.string.status)).getValue().toString();
                    mName.setText(name);
                    mProject.setText(jumProyek);
                    mInformation.setText(information);
                    ArrayList<String> mStringList = new ArrayList<>();
                    mStringList.add(image);
                    if (mStringList.contains("Default")) {
```

```
                    } else {
                        Picasso.with(mContext).load(image).into(mProfile_image);
                    }
                }
            }
            @Override
            public void onCancelled(DatabaseError databaseError) {

            }
        });
```

4.3.4. Pencarian Proyek

Pada pencarian proyek digunakan oleh penyedia jasa untuk mencari proyek sesuai dengan pekerjaan, kota dan *budget* yang diinginkan. Setelah memilih data-data yang ada kemudian data tersebut akan dimasukkan kedalam *list*. Kemudian dari *list* tersebut dapat di klik untuk menuju ke detail informasi tentang proyek tersebut. Untuk lebih jelasnya dapat dilihat pada Segmen 4.6.

Segmen 4.6 Pencarian proyek

```
Query mQuery =
mReference.child("ListProject").orderByChild("jasa_kota_budget").equalTo(value);
mQuery.addChildEventListener(new ChildEventListener() {
    @Override
    public void onChildAdded(final DataSnapshot dataSnapshot, String s) {
        if (dataSnapshot.exists()) {
            getkey = dataSnapshot.getKey();
            int size = (int) dataSnapshot.getChildrenCount();
            if (size != 0) {
                mTersedia.setText("Tap untuk melihat detail proyek");
            }

mReference.child("ListProject").child(getkey).addValueEventListener(new
ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot2) {
                    if (dataSnapshot2.exists()) {
                        String
id_proyek,id_jasa,id_pencari,id_penyedia,id_kota,kategori_budget,
                                judulproyek,
                                statusproyek,jasa_kota, jasa_budget, kota_budget,
jasa_kota_budget, budget, namapencari, id_bid;
                        id_proyek =
dataSnapshot2.child("id_proyek").getValue().toString();
                        id_jasa =
dataSnapshot2.child("id_jasa").getValue().toString();
                        id_pencari =
dataSnapshot2.child("id_pencari").getValue().toString();
                        id_kota =
dataSnapshot2.child("id_kota").getValue().toString();
                        kategori_budget =
dataSnapshot2.child("kategori_budget").getValue().toString();
                        statusproyek =
dataSnapshot2.child("statusproyek").getValue().toString();
                        jasa_kota =
dataSnapshot2.child("jasa_kota").getValue().toString();
                        jasa_budget =
dataSnapshot2.child("jasa_budget").getValue().toString();
                        kota_budget =
dataSnapshot2.child("kota_budget").getValue().toString();
```

```
                        jasa_kota_budget =
dataSnapshot2.child("jasa_kota_budget").getValue().toString();
                        if (statusproyek.equals("Menunggu Bid") ||
statusproyek.equals("Menunggu Kontestan")) {
                            mProjectList.add(new DataProject(id_proyek, id_jasa,
id_pencari, id_kota, kategori_budget, statusproyek, jasa_kota, jasa_budget,
                                    kota_budget, jasa_kota_budget));

                            mAdapter = new ListResult(getApplicationContext(),
mProjectList);
                            mResultlistview.setAdapter(mAdapter);

                        } else {
                            mTersedia.setText("Proyek tidak ditemukan");
                        }
                    }
                }
                @Override
                public void onCancelled(DatabaseError databaseError) {
                }
            });
        }
    }
    @Override
    public void onChildChanged(DataSnapshot dataSnapshot, String s) {}
    @Override
    public void onChildRemoved(DataSnapshot dataSnapshot) {}
    @Override
    public void onChildMoved(DataSnapshot dataSnapshot, String s) {}
    @Override
    public void onCancelled(DatabaseError databaseError) {}
});
```

4.3.5. Pembuatan Proyek

Pembuatan proyek dibagi menjadi empat *template* yang sudah disediakan. Setiap *template* memiliki beberapa perbedaan untuk pengisian data yang diminta seperti pemilihan lokasi pada google maps, jenis produk dan durasi acara dalam proyek tersebut yang tidak semuanya ada pada setiap *template*. Setelah pencari jasa mengisi semua data yang diminta maka data-data tersebut akan dimasukkan kedalam *database* dan bagi penyedia jasa yang sudah melakukan pengelolaan data pada tersedia layanan jasa dan kota maka akan mendapatkan notifikasi sesuai yang sudah dibuat. Untuk lebih jelasnya dapat dilihat pada Segmen 4.7.

Segmen 4.7 Pembuatan proyek

```
mCreateProject.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        strJudul = mJudulProyek.getText().toString();
        strBudget = mBudget.getText().toString();
        strDetail = mDetail.getText().toString();
        final String statusProyek = "Menunggu Bid";
        SimpleDateFormat dateFormatGmt = new SimpleDateFormat("dd/MM/yyyy");
        dateFormatGmt.setTimeZone(TimeZone.getTimeZone("GMT+7"));
        dateNow = dateFormatGmt.format(new Date());
        String[] separated = dateNow.split("/");
        tglnow = Integer.valueOf(separated[0]);
        blnnow = Integer.valueOf(separated[1]);
        thnnow = Integer.valueOf(separated[2]);
        if (strJudul.equals("") || strBudget.equals("") || strDetail.equals("") ||
```

```
cid == 0 || yearDeadline == 0 || yearButuh == 0 ) {
            Toast.makeText(TemplateProjectSatuActivity.this, "Please Input All
Data", Toast.LENGTH_SHORT).show();
        }
        else {
            budget = Integer.valueOf(strBudget);
            if (budget<500000){
                kategoribudget = "1";
            }
            else if (budget>=500000 && budget<1000000){
                kategoribudget = "2";
            }
            else if (budget>=1000000 && budget<1500000){
                kategoribudget = "3";
            }
            else if (budget>=1500000 && budget<2000000){
                kategoribudget = "4";
            }
            else {
                kategoribudget = "5";
            }
            jasa_kota_budget = value+cjid+kategoribudget;
            jasa_kota = value+cjid;
            jasa_budget = value+kategoribudget;
            kota_budget = cjid+kategoribudget;
            if (thnnow <= yearDeadline && thnnow <= yearButuh) {
                if (yearDeadline <= yearButuh) {
                    if (blnnow <= monthDeadline && blnnow <= monthButuh) {
                        if (monthDeadline <= monthButuh) {
                            if (tglnow <= dayDeadline && tglnow <= dayButuh) {
                                if (dayDeadline <= dayButuh) {
                                    //Masukin Firebase
                                    final String key =
mDatabase.getReference("Project").push().getKey();
                                    final DataProjectSatu dataProjectSatu = new
DataProjectSatu(key,userID,"", "",strJudul,

strDetail,tglDeadline,tglDibutuhkan,"",statusProyek,budget);

mReference.child("Project").child(key).setValue(dataProjectSatu);

                                    final DataProject dataProjectPencari = new
DataProject(key,value,userID,cjid,kategoribudget,jasa_kota,jasa_budget,kota_budget,
jasa_kota_budget,statusProyek);

mReference.child("ListProject").child(key).setValue(dataProjectPencari);

Toast.makeText(TemplateProjectSatuActivity.this, "Project Created",
Toast.LENGTH_SHORT).show();
                                    //Notifikasi
                                    final String idcity = cjid;
                                    Query mQuery =
mReference.child("AvailableJob").orderByChild("id_job").equalTo(value);
                                    mQuery.addChildEventListener(new
ChildEventListener() {
                                        @Override
                                        public void onChildAdded(DataSnapshot
dataSnapshot, String s) {
                                            if (dataSnapshot.exists()) {
                                                String getkey =
dataSnapshot.getKey();

mReference.child("AvailableJob").child(getkey).addValueEventListener(new
ValueEventListener() {
                                                    @Override
                                                    public void
onDataChange(DataSnapshot dataSnapshot2) {
                                                        if (dataSnapshot2.exists())
{
                                                            final String
temppenyedia = dataSnapshot2.child("id_penyedia").getValue().toString();
                                                            Query mQuery2 =
mReference.child("AvailableCity").orderByChild("id_city").equalTo(idcity);
```

```
mQuery2.addChildEventListener(new ChildEventListener() {
                                                                    @Override
                                                                    public void
onChildAdded(DataSnapshot dataSnapshot3, String s) {
                                                                        if
(dataSnapshot3.exists()) {
                                                                            final
String getkeycity = dataSnapshot3.getKey();

mReference.child("AvailableCity").child(getkeycity).addValueEventListener(new
ValueEventListener() {

@Override
                                                                                public
void onDataChange(DataSnapshot dataSnapshot4) {
                                                                                    if
(dataSnapshot4.exists()) {

final String temppenyediacity =
dataSnapshot4.child("id_penyedia").getValue().toString();

if (temppenyediacity.equals(temppenyedia)) {

mReference.child("Users").child(temppenyediacity).addValueEventListener(new
ValueEventListener() {

@Override

public void onDataChange(DataSnapshot dataSnapshot5) {

if (dataSnapshot5.exists()) {

String token_id = dataSnapshot5.child("token_id").getValue().toString();

mService = Common.getFCMClient();
Notification notification = new Notification(key, "Proyek");

Sender sender = new Sender(token_id, notification);

mService.sendNotification(sender).enqueue(new Callback<MyResponse>() {

@Override

public void onResponse(Call<MyResponse> call, Response<MyResponse> response) {

if (response.body().success == 1) {

}

}

@Override

public void onFailure(Call<MyResponse> call, Throwable t) {
}
});
}

}

@Override

public void onCancelled(DatabaseError databaseError) {

}

});

} else {

}
                                                                            }
                                                                        }
```

```
@Override
                                                                        public
void onCancelled(DatabaseError databaseError) {
                                                                        }
                                                                    });
                                                                }
                                                            }
                                                            @Override
                                                            public void
onChildChanged(DataSnapshot dataSnapshot, String s) {}
                                                            @Override
                                                            public void
onChildRemoved(DataSnapshot dataSnapshot) {}
                                                            @Override
                                                            public void
onChildMoved(DataSnapshot dataSnapshot, String s) {}
                                                            @Override
                                                            public void
onCancelled(DatabaseError databaseError) {}
                                                        });
                                                    }
                                                }
                                                @Override
                                                public void
onCancelled(DatabaseError databaseError) {}
                                            });
                                        }
                                    }
                                    @Override
                                    public void onChildChanged(DataSnapshot
dataSnapshot, String s) {}
                                    @Override
                                    public void onChildRemoved(DataSnapshot
dataSnapshot) {}
                                    @Override
                                    public void onChildMoved(DataSnapshot
dataSnapshot, String s) {}
                                    @Override
                                    public void onCancelled(DatabaseError
databaseError) {}
                                });


Toast.makeText(TemplateProjectSatuActivity.this, "Project Created",
Toast.LENGTH_SHORT).show();
                                Intent intent = new
Intent(TemplateProjectSatuActivity.this, HomePencariActivity.class);
                                intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK
| Intent.FLAG_ACTIVITY_NEW_TASK);
                                startActivity(intent);

                            } else {

Toast.makeText(TemplateProjectSatuActivity.this, "Tanggal Deadline/Dibutuhkan Tidak
Sesuai", Toast.LENGTH_SHORT).show();
                            }
                        } else {
                            Toast.makeText(TemplateProjectSatuActivity.this,
"Tanggal Invalid", Toast.LENGTH_SHORT).show();
                        }
                    } else {
                        Toast.makeText(TemplateProjectSatuActivity.this, "Bulan
Deadline/Dibutuhkan Tidak Sesuai", Toast.LENGTH_SHORT).show();
                    }
                } else {
                    Toast.makeText(TemplateProjectSatuActivity.this, "Bulan
Invalid", Toast.LENGTH_SHORT).show();
                }
            } else {
                Toast.makeText(TemplateProjectSatuActivity.this, "Tahun
Deadline/Dibutuhkan Tidak Sesuai", Toast.LENGTH_SHORT).show();
            }
        } else {
            Toast.makeText(TemplateProjectSatuActivity.this, "Tahun Invalid",
```

```
Toast.LENGTH_SHORT).show();
                }
            }
        }
});
```

4.3.6. Daftar Proyek

Daftar proyek adalah halaman dimana menampilkan data proyek-proyek apa saja yang dimiliki oleh pengguna baik pencari maupun penyedia jasa. Daftar proyek ini bertujuan agar mempermudah pengguna jika mengeklik proyek tersebut maka akan melihat secara detail tentang proyek tersebut. Perbedaan yang dimiliki antara pencari dan penyedia jasa adalah berdasarkan data dari *database*. Untuk pencari jasa *query* berdasarkan *database* Project dengan id yang dimiliki oleh pencari jasa. Sedangkan *query* penyedia jasa adalah berdasarkan *database.* BidKontes dengan id yang dimiliki oleh penyedia jasa. Untuk lebih jelasnya dapat dilihat pada segmen *program* 4.8.

Segmen 4.8 Daftar proyek

```
Query mQuery =
mReference.child("ListProject").orderByChild("id_pencari").equalTo(value);
mQuery.addChildEventListener(new ChildEventListener() {
    @Override
    public void onChildAdded(DataSnapshot dataSnapshot, String s) {
        if (dataSnapshot.exists()){
            final int size = (int) dataSnapshot.getChildrenCount();
            getkey = dataSnapshot.getKey();

mReference.child("ListProject").child(getkey).addListenerForSingleValueEvent(new
ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot2) {
                    if (dataSnapshot2.exists()) {
                        if (size != 0) {
                            mTersedia.setText("Tap untuk melihat lebih detail");
                        } else {
                            mTersedia.setText("Data proyek anda tidak ditemukan");
                        }

                        id_proyek =
dataSnapshot2.child("id_proyek").getValue().toString();
                        id_jasa =
dataSnapshot2.child("id_jasa").getValue().toString();
                        id_pencari =
dataSnapshot2.child("id_pencari").getValue().toString();
                        id_kota =
dataSnapshot2.child("id_kota").getValue().toString();
                        kategori_budget =
dataSnapshot2.child("kategori_budget").getValue().toString();
                        statusproyek =
dataSnapshot2.child("statusproyek").getValue().toString();
                        jasa_kota =
dataSnapshot2.child("jasa_kota").getValue().toString();
                        jasa_budget =
dataSnapshot2.child("jasa_budget").getValue().toString();
                        kota_budget =
dataSnapshot2.child("kota_budget").getValue().toString();
```

```
                        jasa_kota_budget =
dataSnapshot2.child("jasa_kota_budget").getValue().toString();

                        mProjectList.add(new DataProject(id_proyek, id_jasa,
id_pencari, id_kota, kategori_budget, statusproyek, jasa_kota, jasa_budget,
                                kota_budget, jasa_kota_budget));

                        mAdapter = new ListProjectPencari(getApplicationContext(),
mProjectList);
                        mResultlistview.setAdapter(mAdapter);

                    }
                }
                @Override
                public void onCancelled(DatabaseError databaseError) {
                }
            });
        }
    }
    @Override
    public void onChildChanged(DataSnapshot dataSnapshot, String s) {}
    @Override
    public void onChildRemoved(DataSnapshot dataSnapshot) {}
    @Override
    public void onChildMoved(DataSnapshot dataSnapshot, String s) {}
    @Override
    public void onCancelled(DatabaseError databaseError) {}
});
```

4.3.7. Chat

Chat dapat dilakukan oleh sesama pengguna yaitu pencari dan penyedia jasa. Dalam fitur *chat* ini saat pengguna sudah mengirimkan pesan makasan tersebut akan masuk ke dalam *database* dengan dua jenis yaitu "*PrivateChat"* dan "*Chat*". *Database* pada PrivateChat berfungsi untuk menampilkan isi pesan sedangkan *Chat* akan berfungsi untuk menampilkan pengguna siapa saja yang terkait dengan *chat* tersebut pada daftar *chat* Untuk lebih jelasnya dapat dilihat pada Segmen 4.9.

Segmen 4.9 *Chat*

```
    //Untuk mengirim chat
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            EditText input = (EditText) findViewById(R.id.inputmessagep);
            String key = mDatabase.getReference("PrivateChat").push().getKey();
            //Masukin database untuk isi chat
            DataPrivateChat dataPrivateChatPengirim = new
DataPrivateChat(pengirim,target,input.getText().toString(),"Pengirim");

mReference.child("PrivateChat").child(pengirim).child(target).child(key).setValue(
dataPrivateChatPengirim);
            DataPrivateChat dataPrivateChatPenirima = new
DataPrivateChat(pengirim,target,input.getText().toString(),"Penerima");

mReference.child("PrivateChat").child(target).child(pengirim).child(key).setValue(
dataPrivateChatPenirima);
            //Masukin database untuk list chat
            DataChat dataChat1 = new DataChat(pengirim,target,"0");
```

```
mReference.child("Chat").child(pengirim).child(target).setValue(dataChat1);
            DataChat dataChat2 = new DataChat(target,pengirim,"1");

mReference.child("Chat").child(target).child(pengirim).setValue(dataChat2);
            input.setText("");

            //kirim notifikasi
            mReference.child("Users").child(target).addValueEventListener(new
ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot2) {
                    if (dataSnapshot2.exists()) {
                        String token_id =
dataSnapshot2.child("token_id").getValue().toString();
                        mService = Common.getFCMClient();
                        Notification notification = new Notification(target,
pengirim);
                        Sender sender = new Sender(token_id, notification);
                        mService.sendNotification(sender).enqueue(new
Callback<MyResponse>() {
                            @Override
                            public void onResponse(Call<MyResponse> call,
Response<MyResponse> response) {
                                if (response.body().success == 1) {
                                }
                            }

                            @Override
                            public void onFailure(Call<MyResponse> call, Throwable
t) {
                            }
                        });
                    }
                }
                @Override
                public void onCancelled(DatabaseError databaseError) {}
            });
        }
    });
    displayMessage();
}

//Untuk menampilkan isi chat
private void displayMessage(){
    ListView listofMessage = (ListView) findViewById(R.id.list_of_messagep);
    adapter = new
FirebaseListAdapter<DataPrivateChat>(this,DataPrivateChat.class,R.layout.list_item
_chat,mReference.child("PrivateChat").child(userID).child(target)) {
        @Override
        protected void populateView(View v, DataPrivateChat model, int position) {
            TextView messageText,namaUser,messageTime;
            messageText = (TextView) v.findViewById(R.id.message_text);
            namaUser = (TextView) v.findViewById(R.id.message_user);
            messageTime = (TextView) v.findViewById(R.id.message_time);
            if(model.getStatususer().equals("Pengirim")){
                namaUser.setText("You");
                namaUser.setTextColor(Color.parseColor("#adc83e"));
            }
            else {
                namaUser.setText(namatarget);
            }

            messageText.setText(model.getMessageText());
            messageTime.setText(DateFormat.format("dd-MM-yyyy
(HH:mm:ss)",model.getMessageTime()));
        }
    };
    listofMessage.setAdapter(adapter);
}
```

4.3.8. Bid

Bid dilakukan oleh penyedia jasa untuk mengajukan tawaran harga terbaik mereka dengan menyesuaikan maksimum *budget* yang dimiliki oleh pencari jasa. Penyedia jasa dapat melakukan *bidding* kurang dari sama dengan maksimum *budget.* Jika pengguna melakukan *bid* lebih dari maksimum *budget* maka akan memunculkan pesan bahwa *bid* yang dilakukan berlebih. Jika telah selesai melakukan *bid*, selama pencari belum melakukan pemilihan maka penyedia jasa dapat melakukan *edit* terhadap *bid* yang mereka ajukan. Untuk lebih jelasnya dapat dilihat pada segmen *program* 4.8.

Segmen 4.10 *Bid*

```
AlertDialog.Builder builder = new AlertDialog.Builder(PenyediaShowProject.this);
builder.setTitle("Masukkan Bid Anda");
input = new EditText(PenyediaShowProject.this);
input.setInputType(InputType.TYPE_CLASS_NUMBER);
builder.setView(input);

builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        m_Text = input.getText().toString();
        if (m_Text.equals("")) {
            Toast.makeText(PenyediaShowProject.this, "Bid tidak boleh kosong",
Toast.LENGTH_SHORT).show();
        } else {
            bidnum = Integer.parseInt(m_Text);
            if (bidnum > bidmax) {
                Toast.makeText(PenyediaShowProject.this, "Bid melebihi budget
pencari", Toast.LENGTH_SHORT).show();
            }

            else {
                bidnum = Integer.parseInt(m_Text);
                NumberFormat numberFormat =
NumberFormat.getNumberInstance(Locale.US);
                String numberAsString = numberFormat.format(bidnum);

                String key = mDatabase.getReference("Bid").push().getKey();
                DataBid dataBid = new DataBid(key, idproyek, idpencari, userID,
penyedia_proyek, "Menunggu", "",bidnum);
                mReference.child("BidKontes").child(key).setValue(dataBid);
                Toast.makeText(PenyediaShowProject.this, "Bid Berhasil",
Toast.LENGTH_LONG).show();
                mMybid.setText("Bid Saya: Rp. " + numberAsString);

                //Kirim Notifikasi

mReference.child("Users").child(idpencari).addValueEventListener(new
ValueEventListener() {
                    @Override
                    public void onDataChange(DataSnapshot dataSnapshot5) {
                        if (dataSnapshot5.exists()) {
                            String token_id =
dataSnapshot5.child("token_id").getValue().toString();

                            String key =
mDatabase.getReference("Notifikasi").push().getKey();
                            DataNotifikasi dataNotifikasi = new
DataNotifikasi(key, value, idpencari, "Anda mendapatkan bid baru");
```

```
mReference.child("Notifikasi").child(idpencari).child(key).setValue(dataNotifikasi
);

                            mService = Common.getFCMClient();
                            Notification notification = new Notification(idproyek,
"Proyek Bid");
                            Sender sender = new Sender(token_id, notification);
                            mService.sendNotification(sender).enqueue(new
Callback<MyResponse>() {
                                @Override
                                public void onResponse(Call<MyResponse> call,
Response<MyResponse> response) {
                                    if (response.body().success == 1) {
                                    }
                                }

                                @Override
                                public void onFailure(Call<MyResponse> call,
Throwable t) {
                                }
                            });
                        }
                    }
                    @Override
                    public void onCancelled(DatabaseError databaseError) {}
                });

                Intent nextPage = new Intent(PenyediaShowProject.this,
SearchProjectActivity.class);
                nextPage.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK |
Intent.FLAG_ACTIVITY_NEW_TASK);
                startActivity(nextPage);
            }

        }
    }
});
builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        dialog.cancel();
    }
});
builder.show();
```

4.3.9. Konfirmasi Pembayaran

Konfirmasi pembayaran disini adalah untuk pencari jasa mengirimkan bukti transfer berupa foto kepada pihak pemilik aplikasi. Data tersebut nanti akan dimasukkan kedalam *database* Transaksi. Kemudian setelah mengirim selesai pencari akan menunggu konfirmasi dari admin untuk menentukan apakah pembayaran yang dilakukan benar atau tidak. Untuk lebih jelasnya dapat dilihat pada Segmen 4.11.

Segmen 4.11 Konfirmasi Pembayaran

```
mListbid.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        if (statusproyek.equals("Transaksi")) {

mReference.child("Project").child(value).child("statusproyek").setValue("Menunggu
```

```
Admin");
            Intent photoPickerIntent = new Intent(Intent.ACTION_PICK);
            photoPickerIntent.setType("image/*");
            startActivityForResult(photoPickerIntent, PICK_IMAGE);
        }
    }
    )
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == RESULT_CANCELED) {
        return;
    }
    if (requestCode == PICK_IMAGE) {
        if (data != null) {
            imageUri = data.getData();
            select_image();
        }
    }
}
private void select_image(){
    final String key = mDatabase.getReference("Transaksi").push().getKey();
    StorageReference imagePath =
mStorage.child("Transaksi").child(userID).child(imageUri.getLastPathSegment());
    imagePath.putFile(imageUri).addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
        @Override
        public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
            String image_firebase_uri = taskSnapshot.getDownloadUrl().toString();
            String strProfileImage = image_firebase_uri;

            String key = mDatabase.getReference("Transaksi").push().getKey();
            SimpleDateFormat dateFormatGmt = new SimpleDateFormat("dd/MM/yyyy
(HH:mm:ss)");
            dateFormatGmt.setTimeZone(TimeZone.getTimeZone("GMT+7"));
            String dateNow = dateFormatGmt.format(new Date());
            DataRekening dataRekening = new DataRekening(key, idproyek, idbid,
userID, idpenyedia,"", dateNow, strProfileImage, "Pencari");
            mReference.child("Transaksi").child(key).setValue(dataRekening);
            Toast.makeText(PencariShowProject.this, "Transaksi sudah dikirim harap
tung konfirmasi dari admin", Toast.LENGTH_SHORT).show();
        }
    });
}
```

4.3.10. Portofolio Watermark

Portofolio *watermark* berfungsi untuk melindungi foto-foto yang dimiliki penyedia jasa agar tidak diambil dengan sembarangan. Sebelum proses ini dilakukan penyedia mengupload portofolionya dan kemudian masuk ke dalam

storage dan *datbase* di firebase. Untuk melihat portofolionya penyedia atau pencari jasa harus mengeklik terlebih dahulu foto yang ingin dilihat setelah itu akan berpindah ke halaman dimana halaman tersebut akan menampilkan foto dengan *watermark*. Dalam fungsi ini ada dua bitmap yaitu foto portofolio milik penyedia dan gambar logo *watermark* aplikasi. Foto dan logo tersebut akan di download kemudian akan diubah kedalam bitmap. Setelah itu foto dan logo tersebut akan dijadikan satu dengan canvas dalam bentuk bitmap kemudian ditampilkan. Untuk lebih jelasnya dapat dilihat pada Segmen 4.12.

Segmen 4.12 Portofolio *watermark*

```
Bundle extras = getIntent().getExtras();
    if (extras != null) {
        value = extras.getString("key");
        idpenyedia = extras.getString("idpenyedia");

mReference.child("Portofolio").child(idpenyedia).child(value).addValueEventListene
r(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            portofolioimage =
dataSnapshot.child("portofolio_image").getValue().toString();

            Thread thread = new Thread(new Runnable(){
                public void run() {
                    try {
                        ByteArrayOutputStream bytes = new ByteArrayOutputStream();
                        URL url = new URL(portofolioimage);
                        HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
                        connection.setDoInput(true);
                        connection.connect();
                        InputStream input = connection.getInputStream();
                        bitmapimage = BitmapFactory.decodeStream(input);
                        bitmapimage.compress(Bitmap.CompressFormat.PNG, 100,
bytes);


                        watermarkimage =
BitmapFactory.decodeResource(getResources(), R.drawable.logowatermark);
                        watermarkimage.compress(Bitmap.CompressFormat.PNG, 100,
bytes);

                        int w = bitmapimage.getWidth();
                        int h = bitmapimage.getHeight();
```

```
                        resultbitmap = Bitmap.createBitmap(w, h,
bitmapimage.getConfig());
                        Canvas canvas = new Canvas(resultbitmap);
                        canvas.drawBitmap(bitmapimage, 0, 0, null);
                        canvas.drawBitmap(watermarkimage, 0, h/2, null);
                        resultbitmap.compress(Bitmap.CompressFormat.PNG, 100,
bytes);

                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                mPortofolio.setImageBitmap(resultbitmap);
                            }
                        });

                    } catch (IOException e) {
                        e.printStackTrace();
                        Toast.makeText(ImageWatermark.this, "Failed!",
Toast.LENGTH_SHORT).show();
                    }
                }
            });
            thread.start();
        }
        @Override
        public void onCancelled(DatabaseError databaseError) {
        }
    });
}
```

4.3.11. Rating dan Review

Rating dan *review* dapat diberikan pencari jasa kepada penyedia jasa atas penilaian pekerjaan yang sudah dilakukan. *Rating* dan *review* akan aktif jika status proyek dalam status pekerjaan selesai saja. Untuk lebih jelasnya dapat dilihat pada segmen *program* 4.13.

Segmen 4.13 *Rating* dan *review*

```
buttonratingreview.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        reviewtext = review.getText().toString();
        if (rating == 0 || reviewtext.equals("")){
            Toast.makeText(context,"Harap isi rating dan
review",Toast.LENGTH_SHORT).show();
        }
        else {

mReference.child("Users").child(valuepencari).addListenerForSingleValueEvent(new
```

```
ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot2) {
                    if (dataSnapshot2.exists()) {
                        String jumlahproyek =
dataSnapshot2.child("jumlahproyek").getValue().toString();
                        String jumlahrating =
dataSnapshot2.child("rating").getValue().toString();
                        int jumproyek = Integer.parseInt(jumlahproyek);
                        Double jumrating = Double.parseDouble(jumlahrating);
                        double ratinguser = ((jumrating*(jumproyek-
1))+rating)/jumproyek;//(jumrating + rating) / jumproyek;

mReference.child("Users").child(valuepenyedia).child("rating").setValue(ratinguser
);

mReference.child("Project").child(valueproyek).child("statusproyek").setValue("Pro
yek Selesai");

                        String key =
mDatabase.getReference("Review").child(valuepenyedia).push().getKey();
                        DataReview dataReview = new DataReview(valuepenyedia,
valuepencari, reviewtext, rating);

mReference.child("Review").child(valuepenyedia).child(key).setValue(dataReview);
                        Toast.makeText(RatingReview.this, "Terima kasih telah
menggunakan jasa ini", Toast.LENGTH_SHORT).show();

                        Intent nextPage = new Intent(RatingReview.this,
HomePencariActivity.class);
                        nextPage.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK |
Intent.FLAG_ACTIVITY_NEW_TASK);
                        startActivity(nextPage);
                    }
                }
                @Override
                public void onCancelled(DatabaseError databaseError) {
                }
            });
        }
    }
});
```

4.3.12. Notifikasi

Notifikasi ini bertujuan agar pengguna mendapatkan pemberitahuan tentang sesuatu yang terjadi seperti *chat*, proses *bid* dan banyak lagi. Dalam fungsi ini dibuatlah dua string dimana akan menentukan tujuan halaman notifikasi jika di klik yaitu cekpage dan cekid. Cekpage berfungsi sebagai menentukan halaman tujuan sedangkan cekid adalah *value* yang berisikan id untuk mendapatkan data yang ada pada halaman tersebut. Tetapi untuk fitur *chat* cekpage akan berisikan id penerima pesan dan cekid berisi id pengirim pesan. Untuk lebih jelasnya dapat dilihat pada segmen *program* 4.14.

Segmen 4.14 Notifikasi

```
mReference.child("ListProject").child(cekid).addValueEventListener(new
ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
```

```
        if (dataSnapshot.exists()) {
            String idjasa = dataSnapshot.child("id jasa").getValue().toString();

mReference.child("Jasa").child(idjasa).addListenerForSingleValueEvent(new
ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot2) {
                    if (dataSnapshot2.exists()) {
                        String namajasa =
dataSnapshot2.child("nama").getValue().toString();
                        String template =
dataSnapshot2.child("template").getValue().toString();

                        Uri alarmSound =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);

                        if (template.equals("1")) {
                            nextPage = new Intent(MyFirebaseMessaging.this,
PenyediaShowProject.class);
                            nextPage.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK |
Intent.FLAG_ACTIVITY_NEW_TASK);
                            nextPage.putExtra("key",cekid);
                            nextPage.putExtra("iduser",userID);
                            nextPage.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        } else if (template.equals("2")) {
                            nextPage = new Intent(MyFirebaseMessaging.this,
PenyediaShowProject2.class);
                            nextPage.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK |
Intent.FLAG_ACTIVITY_NEW_TASK);
                            nextPage.putExtra("key",cekid);
                            nextPage.putExtra("iduser",userID);
                            nextPage.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        } else if (template.equals("3")) {
                            nextPage = new Intent(MyFirebaseMessaging.this,
PenyediaShowProject3.class);
                            nextPage.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK |
Intent.FLAG_ACTIVITY_NEW_TASK);
                            nextPage.putExtra("key",cekid);
                            nextPage.putExtra("iduser",userID);
                            nextPage.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        } else if (template.equals("4")) {
                            nextPage = new Intent(MyFirebaseMessaging.this,
PenyediaShowProject4.class);
                            nextPage.putExtra("key",cekid);
                            nextPage.putExtra("iduser",userID);
                            nextPage.putExtra("cekpage","show");
                            nextPage.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        }

                        PendingIntent pendingIntent =
PendingIntent.getActivity(MyFirebaseMessaging.this, 0, nextPage,
PendingIntent.FLAG_ONE_SHOT);
                        NotificationCompat.Builder builder = new
NotificationCompat.Builder(MyFirebaseMessaging.this)
                                .setSmallIcon(R.mipmap.ic_launcher_round)
                                .setSound(alarmSound)
                                .setContentTitle(namaproyek)
                                .setContentText("Membutuhkan Jasa " + namajasa)
                                .setAutoCancel(true)
                                .setContentIntent(pendingIntent);

                        NotificationManager notificationManager =
(NotificationManager) getSystemService(NOTIFICATION_SERVICE);
                        notificationManager.notify(0, builder.build());
                    }
                }
                @Override
                public void onCancelled(DatabaseError databaseError) {
                }
            });
        }
    }
    @Override
    public void onCancelled(DatabaseError databaseError) {
```

```
    }
});
```

4.4. Implementasi Aplikasi Admin (Website)

Selanjutnya implementasi website dimana hanya bisa diakses oleh admin untuk mengelola data-data yang ada. Dalam implementasi ini *website* menggunakan *Javascript* agar dapat terhubung dengan *database* Firebase. Berikut adalah penjelasan mengenai implementasi *website*:

4.4.1. Manage Pekerjaan

Pada halaman ini admin melakukan pengelolaan data jasa pekerjaan apa saja yang tersedia dalam aplikasi. Dalam proses ini admin cukup memasukkan id pekerjaan, nama pekerjaan dan jenis *template*. Untuk lebih jelasnya dapat dilihat pada Segmen 4.15.

Segmen 4.15 *Manage* pekerjaan

```
<script>
          var tblJob = document.getElementById('tbl_job_list');
          var databaseRef = firebase.database().ref('Jasa/');
          var rowIndex = 1;
          databaseRef.once('value', function(snapshot){
            snapshot.forEach(function(childSnapshot){
              var childKey = childSnapshot.key;
              var childData = childSnapshot.val();
              var row = tblJob.insertRow(rowIndex);
              var cellId = row.insertCell(0);
              var cellName = row.insertCell(1);
              var cellTemplate = row.insertCell(2);
              cellId.appendChild(document.createTextNode(childKey));
              cellName.appendChild(document.createTextNode(childData.nama));

        cellTemplate.appendChild(document.createTextNode(childData.template));

              rowIndex = rowIndex + 1;
            });
          });

          function addupdate_job() {
            var nama = document.getElementById('nama').value;
            var job_id = document.getElementById('job_id').value;
            var template = document.getElementById('template').value;
            var data = {
```

```
            nama: nama,
            template: template
          }
          var updates = {};
          updates['/Jasa/'+job_id] = data;
          firebase.database().ref().update(updates);
          reload_page();
        }

        function delete_job(){
          var job_id = document.getElementById('job_id').value;
          firebase.database().ref().child('/Jasa/'+job_id).remove();
          alert('Delete Sukses');
        }

        function reload_page() {
          window.location.reload();
        }
</script>
```

4.4.2. Manage Proyek

Pada halaman ini admin dapat memutuskan apakah proyek tersebut layak untuk di *post* atau tidak. Jika *post* yang dilakukan oleh pencari jasa tidak layak untuk ditampilkan maka *admin* cukup memasukkan id proyek tersebut dan kemudian menghapusnya. Untuk lebih jelasnya dapat dilihat pada Segmen 4.16.

Segmen 4.16 *Manage* proyek

```
      <script>
        var tblJob = document.getElementById('tbl_job_list');
        var databaseRef = firebase.database().ref('Project/');
        var rowIndex = 1;
        databaseRef.once('value', function(snapshot){
          snapshot.forEach(function(childSnapshot){
            var childKey = childSnapshot.key;
            var childData = childSnapshot.val();
            var row = tblJob.insertRow(rowIndex);
            var cellidproyek = row.insertCell(0);
            var cellidpencari = row.insertCell(1);
            var celljudulproyek = row.insertCell(2);
            var celldetailproyek = row.insertCell(3);
            var cellstatusproyek = row.insertCell(4);
            var celljenisproyek = row.insertCell(5);
            var cellbudget = row.insertCell(6);
```

```
        cellidproyek.appendChild(document.createTextNode(childData.id_proyek));

        cellidpencari.appendChild(document.createTextNode(childData.id_pencari));

        celljudulproyek.appendChild(document.createTextNode(childData.judulproyek))
        ;

        celldetailproyek.appendChild(document.createTextNode(childData.detailproyek
        ));

        cellstatusproyek.appendChild(document.createTextNode(childData.statusproyek
        ));

        celljenisproyek.appendChild(document.createTextNode(childData.jenisproyek))
        ;
              cellbudget.appendChild(document.createTextNode(childData.budget));
              rowIndex = rowIndex + 1;
            });
          });

          function delete_project(){
            var projectId = document.getElementById('projectId').value;
            firebase.database().ref().child('/ListProject/'+projectId).remove();
            var updatesProyek = {};
            updatesProyek['/Project/'+projectId+'/statusproyek'] = "Batal";
            firebase.database().ref().update(updatesProyek);
            alert('Delete Sukses');
          }
          function reload_page() {
            window.location.reload();
          }
</script>
```

4.4.3. Manage Transaksi

Pada halaman terdapat dua halaman yang berbeda yaitu *manage* transaksi pencari dan *manage transaksi* penyedia ini admin melakukan pengecekan terhadap transaksi-transaksi yang dilakukan oleh pencari jasa maupun penyedia jasa. Yang membedakkan adalah pencari jasa akan berbentuk foto bukti transfer sedangkan penyedia jasa dalam bentuk identitas rekening yang ada. Jika proses transaksi berjalan dengan baik maka admin hanya memasukkan id bid dan id proyek untuk mengubah status tersebut agar dapat ke proses pengerjaan proyek. Untuk lebih jelasnya dapat dilihat pada Segmen 4.17.

Segmen 4.17 *Manage* transaksi

```
<script>
  var tblJob = document.getElementById('tbl_job_list');
  var databaseRef =
firebase.database().ref('Transaksi/').orderByChild('statustransaksi').equal
To('Pencari');
  var rowIndex = 1;

  databaseRef.once('value', function(snapshot){
    snapshot.forEach(function(childSnapshot){
      var childKey = childSnapshot.key;
      var childData = childSnapshot.val();
      var row = tblJob.insertRow(rowIndex);
      var cellidtransaksi = row.insertCell(0);
      var cellidproyek = row.insertCell(1);
      var cellidbid = row.insertCell(2);
      var cellidpencari = row.insertCell(3);
      var cellbuktitrans = row.insertCell(4);
      var celltanggalkirim = row.insertCell(5);
      var image = new Image();
      image.src = childData.buktitrans;
      image.width = 300;

cellidtransaksi.appendChild(document.createTextNode(childData.id_transaksi)
);

cellidproyek.appendChild(document.createTextNode(childData.id_proyek));
      cellidbid.appendChild(document.createTextNode(childData.id_bid));

cellidpencari.appendChild(document.createTextNode(childData.id_pencari));
      cellbuktitrans.appendChild(image);

celltanggalkirim.appendChild(document.createTextNode(childData.tanggalkirim
));
      rowIndex = rowIndex + 1;
    });
  });
  function terima_transaksi(){
    var ProyekId = document.getElementById('ProyekId').value;
    var BidId = document.getElementById('BidId').value;
    var TransaksiId = document.getElementById('TransaksiId').value;
    if (TransaksiId != '') {
      var updatesTransaksi = {};
      updatesTransaksi['/Transaksi/'+TransaksiId+'/statustransaksi'] =
"Finish";
      firebase.database().ref().update(updatesTransaksi);
    }
    if (BidId != '') {
```

```
            var updatesBid = {};
            updatesBid['/BidKontes/'+BidId+'/statusbid'] = "Bekerja";
            firebase.database().ref().update(updatesBid);
          }
          if (ProyekId != '') {
            var updatesProyek = {};
            updatesProyek['/Project/'+ProyekId+'/statusproyek'] = "Bekerja";
            firebase.database().ref().update(updatesProyek);
            var updatesProyek2 = {};
            updatesProyek2['/ListProject/'+ProyekId+'/statusproyek'] = "Bekerja";
            firebase.database().ref().update(updatesProyek2);
          }
          reload_page();
        }
        function tolak_transaksi(){
          var ProyekId = document.getElementById('ProyekId').value;
          var BidId = document.getElementById('BidId').value;

          if (TransaksiId != '') {
            var updatesTransaksi = {};
            updatesTransaksi['/Transaksi/'+TransaksiId+'/statustransaksi'] =
      "Finish";
            firebase.database().ref().update(updatesTransaksi);
          }
          if (BidId != '') {
            var updatesBid = {};
            updatesBid['/Bid/'+BidId+'/statusbid'] = "Batal";
            firebase.database().ref().update(updatesBid);
          }
          if (ProyekId != '') {
            var updatesProyek = {};
            updatesProyek['/Project/'+ProyekId+'/statusproyek'] = "Batal";
            firebase.database().ref().update(updatesProyek);
          }
          reload_page();
        }
        function reload_page() {
          window.location.reload();
        }
</script>
```