MANUAL DE DESARROLLADOR

DentalApp

INDICE

DESCRIPCIÓN DE LA APLICACIÓN

Dental App es una aplicación móvil para sistema operativo Android a partir de la versión 4.1, la cual tiene como principal objetivo llevar el control de citas de los pacientes de una clínica dental además de permitir la visualización de los servicios ofrecidos por la misma y un aspecto a resaltar es que esta se encuentre al alcance de cualquier cliente de la clínica por lo cual el sitio le permite registrarse.

.

REQUISITOS DE LA APLICACIÓN

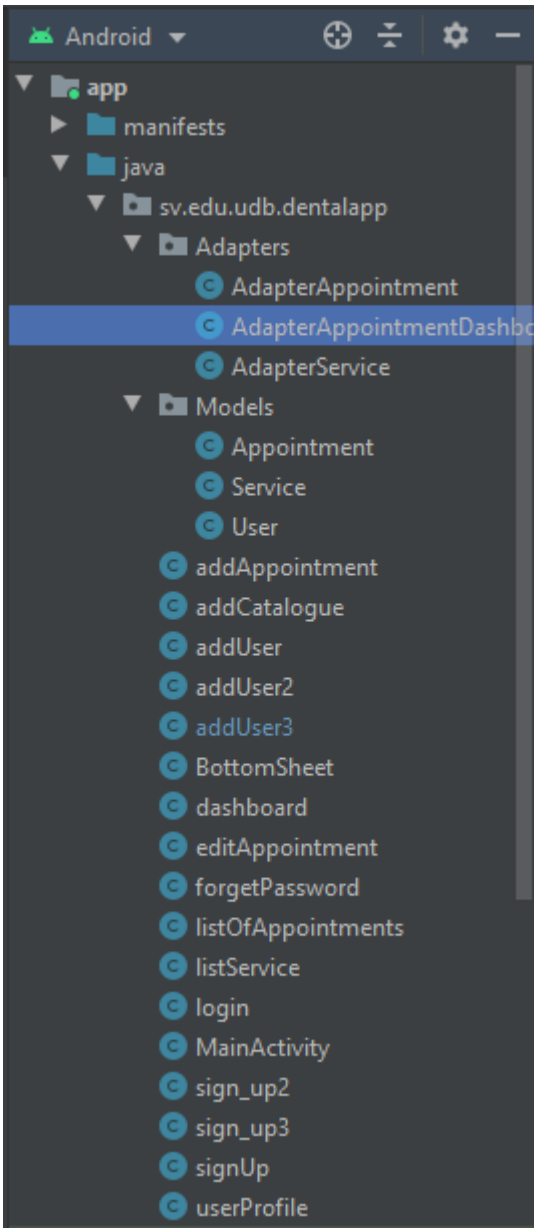
Para el correcto funcionamiento de la aplicación DentalApp se pide como requisito lo siguiente:

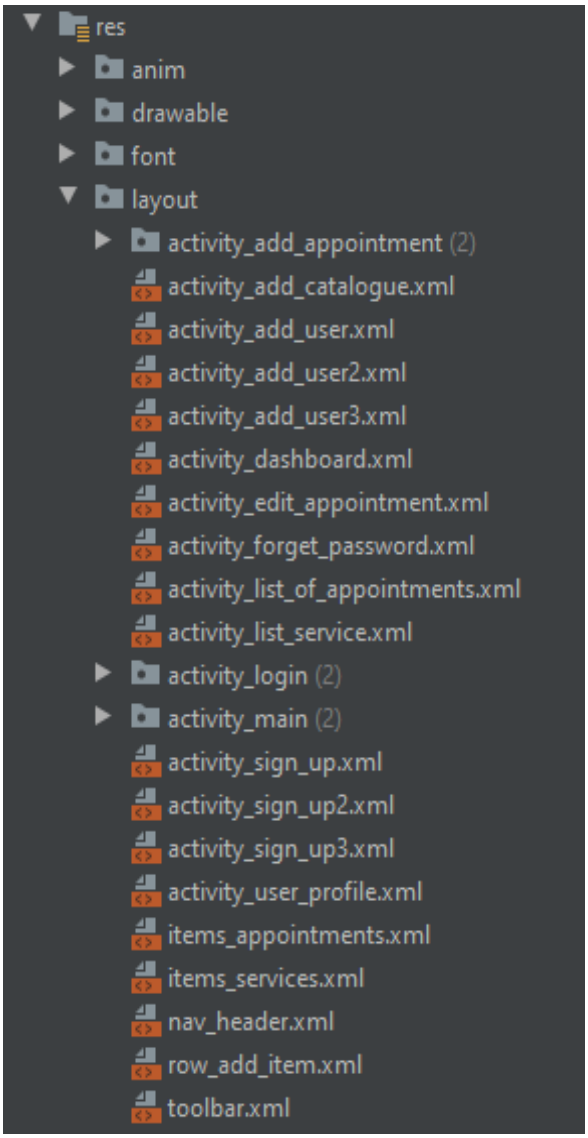
- Dispositivo con sistema operativo Android 4.1 o superiores.
- 1 GB de memoria RAM
- 512 MB de almacenamiento
- Conexión a internet

Para el correcto desarrollo de la aplicación DentalApp se piden los siguientes requisitos:

- Android Studio 4.1.3
- Acceso consola del proyecto DentalApp en Firebase (Administrador de base de datos de la app)

A continuación, se muestra la estructura del proyecto en Android llamado “DentalApp”. Al lado izquierdo se muestran las clases “Java” que se han utilizado en la programación de las diferentes pantallas que componen “DentalApp”. Al lado derecho se muestran los diferentes layouts (pantallas) que se han diseñado para “DentalApp”





DISEÑO DE PANTALLAS

A continuación, se muestra el diseño de cada una de las pantallas de la aplicación. Se ha colocado el nombre de la pantalla, el nombre el layout “nombrepantalla.xml” y la imagen con su respectivo id, con el cual se identifica a cada uno de los componentes dentro de cada clase “java” de programación de cada pantalla.

INICIO DE SESIÓN

normal\activity_login.xml

#	Id (Diseño)
1	edtUser
2	edtPass
3	recuperarcontra
4	btnLogin
5	BtnGoogle

Ilustración 2: Pantalla de Inicio de Sesión

REGISTRO

activity_sign_up.xml

Ilustración 3: Pantalla 1 de registro

#	Id (Diseño)
1	Registrername
2	Registrerlastname
3	Registrerphone
4	btnNext
5	btnLogin
6	btnBack

activity_sign_up2.xml

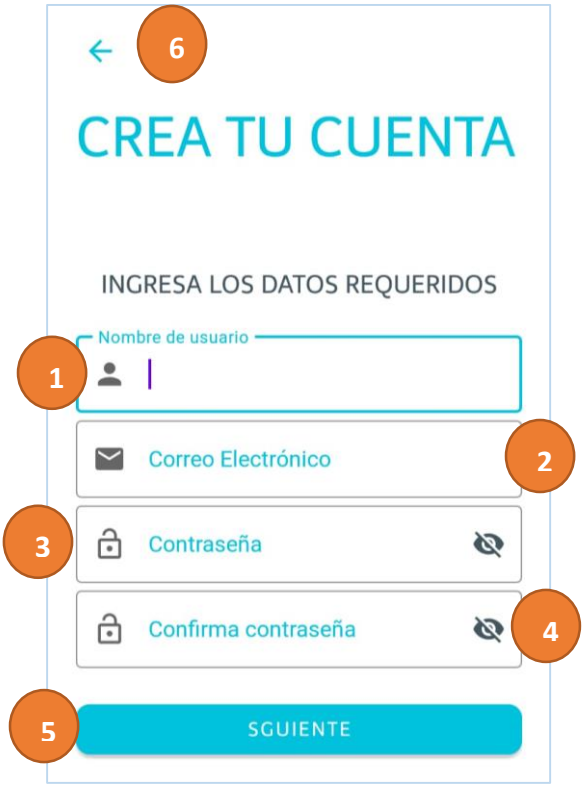


Ilustración 4: Pantalla 2 de registro

#	Id (Diseño)
1	RegistrerUser
2	RegistrerEmail
3	RegistrerPassword
4	RegistrerPasswordAgain
5	btnLogin
6	btnNext

activity_sign_up3.xml

#	Id (Diseño)
1	rbtnMale, rbtnFemale
2	DPfecha
3	btnNext

Ilustración 5: Pantalla final de registro

AGENDAR CITA

normal\activity_add_appointment.xml

Ilustración 6: Pantalla para agendar cita

#	Id (Diseño)
1	user
2	btnSelectTime_Date
3	hour_date
4	observation
5	listServicesAppointment
6	txtvServiceRequired
7	btnaddAppointment
8	btncancel

EDITAR CITA

activity_edit_appointment.xml

Ilustración 7: Pantalla para editar cita

#	Id (Diseño)
1	user
2	btnSelectTime_Date
3	hour_date
4	observation
5	listServices
6	txtvServiceRequired
7	btnaddAppointment
8	btncancel

AÑADIR CATÁLOGO

activity_add_catalogue.xml

Ilustración 8: Pantalla para agregar servicios

#	Id (Diseño)
1	txtServiceName
2	txtDescription
3	txtPrice
4	txtTime
5	btnAddCatalogue
6	btnCancel

CATÁLOGO DE SERVICIOS

activity_list_service.xml

Ilustración 9: Pantalla para ver servicios

#	Id (Diseño)
1	listServices
2	btnback

LISTA DE CITAS

activity_list_of_appointments.xml

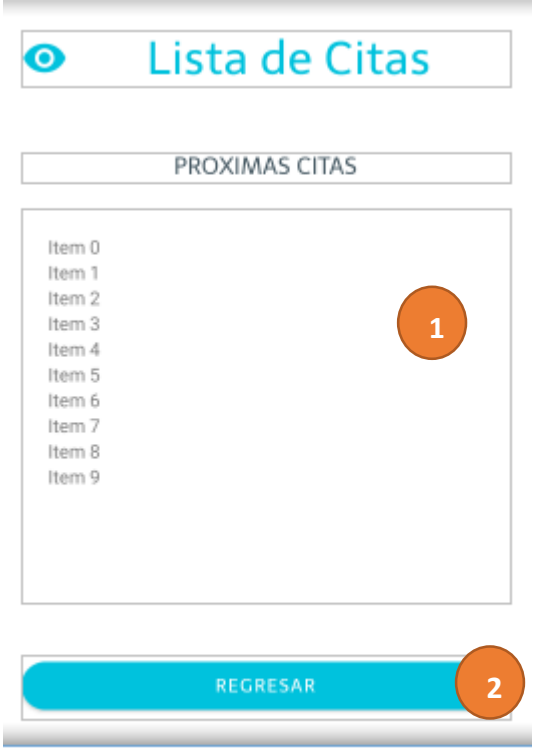


Ilustración 10: Pantalla para ver citas

#	Id (Diseño)
1	listAppointments
2	btnback

RECUPERAR CONTRASEÑA

activity_forget_password.xml

Ilustración 11: Pantalla para recuperar contraseña

#	Id (Diseño)
1	rcorreo
2	rbtn
3	btnBack

CREAR USUARIO

activity_add_user.xml

Ilustración 12: Pantalla para agregar usuarios

#	Id (Diseño)
1	btnBack
2	txtName
3	txtLastName
4	txtPhone
5	btnNext

activity_add_user2.xml

Ilustración 13: Pantalla para agregar usuarios

#	Id (Diseño)
1	btnBack
2	txtAddUserEmail
3	txtAddUserPassword
4	txtAddUserUsuario
5	btnNext

activity_add_user3.xml

Ilustración 14: Pantalla para agregar usuarios

#	Id (Diseño)
1	btnBack
2	rbtnAddUserMale
3	rbtnAddUserFemale
4	rbtnQuee
5	rdbTypeAdmin
6	rdbTypeClient
7	fechaAddUser
8	btnNext

EDITAR CITA

activity_edit_appointment.xml

Ilustración 15: Pantalla de Editar citas

#	Id (Diseño)
1	userEdit
2	hour_dateEdit
3	observationEdit
4	ServicesAppointmentEdit
5	btnEditAppointment
6	btncancelEdit

PERFIL DE USUARIOS

activity_user_profile.xml

#	Id (Diseño)
1	txtNameProfile
2	txtLasNameProfile
3	txtEmailProfile
4	txtPasswordProfile
5	txtPhoneProfile
6	btnUpdate

Ilustración 16: Pantalla del Perfil de Usuario

MENÚ LATERAL

activity_dashboard.xml

Ilustración 17: Menú lateral

PROGRAMACIÓN DE PANTALLAS (JAVA CLASS)

A continuación, se muestran el código utilizado para cada pantalla diseñada (layout). Se coloca el nombre, y unas capturas con la muestra del código que se ha programado para la funcionalidad de "DentalApp".

Login.java

En esta clase se programa la vista del inicio de sesión de usuarios, ya sean administradores o clientes de la clínica dental. Se puede iniciar sesión con el correo de Gmail con ayuda de la autenticación de Firebase o ingresando las credenciales en los inputs establecidos. Además de tener los métodos para acceder a la vista de olvido de contraseña.

```
public class login extends AppCompatActivity {
    private Button BtnLogin,btnLogin;
    private EditText edtUser, edtPassword;
    private GoogleSignInClient mGoogleSignInClient;
    private TextView recuperacion;
    private FirebaseAuth mAuth;
    int RC_SIGN_IN = 1;
    String TAG = "GoogleSignIn";


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN);
        setContentView(R.layout.activity_login);
        mAuth = FirebaseAuth.getInstance();
        BtnLogin= findViewById(R.id.BtnGoogle);
        btnLogin=findViewById(R.id.btnLogin);
        edtUser=findViewById(R.id.edtUser);
        edtPassword=findViewById(R.id.edtPass);
        recuperacion =findViewById(R.id.recuperarcontra);

        GoogleSignInOptions gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
                .requestIdToken(getString(R.string.default_web_client_id))
                .requestEmail()
                .build();
        mGoogleSignInClient = GoogleSignIn.getClient( activity: this, gso);
```

```
        BtnLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { signIn(); }
        });

        btnLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { loginUserAccount(); }
        });
        recuperacion.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent( packageContext: login.this, forgetPassword.class);
                startActivity(intent);
                finish();

            }
        });
    }
```

```
//LOGIN GOOGLE

private void signIn() {

    Intent signInIntent = mGoogleSignInClient.getSignInIntent();
    startActivityForResult(signInIntent, RC_SIGN_IN);
}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    // Result returned from launching the Intent from GoogleSignInApi.getSignInIntent(...);
    if (requestCode == RC_SIGN_IN) {
        Task<GoogleSignInAccount> task = GoogleSignIn.getSignedInAccountFromIntent(data);
        try {
            // Google Sign In was successful, authenticate with Firebase
            GoogleSignInAccount account = task.getResult(ApiException.class);
            Log.d(TAG,  msg: "firebaseAuthWithGoogle:" + account.getId());
            firebaseAuthWithGoogle(account.getIdToken());
        } catch (ApiException e) {
            // Google Sign In failed, update UI appropriately
            Log.w(TAG,  msg: "LOGIN DE GOOGLE FALLIDO", e);
            // ...
        }
    }

}
```

```
private void firebaseAuthWithGoogle(String idToken) {

    AuthCredential credential = GoogleAuthProvider.getCredential(idToken,  accessToken: null);
    mAuth.signInWithCredential(credential)
            .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    if (task.isSuccessful()) {
                        // Sign in success, update UI with the signed-in user's information
                        Log.d(TAG,  msg: "CREDENCIALES EXITOSAS");


                        if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.LOLLIPOP) {

                            Intent dashboardActivity = new Intent( packageContext: login.this,dashboard.class);
                            startActivity(dashboardActivity);
                            login.this.finish();

                        }} else {
                        // If sign in fails, display a message to the user.
                        Log.w(TAG,  msg: "signInWithCredential:failure", task.getException());

                    }
                }
            });
}
```

```
//LOGIN CON CORREO Y CONTRASEÑA

    private void loginUserAccount() {

        String email, password;
        email = edtUser.getText().toString();
        password = edtPassword.getText().toString();

        if (TextUtils.isEmpty(email)) {
            Toast.makeText(getApplicationContext(),  text: "Please enter email...", Toast.LENGTH_LONG).show();
            return;
        }
        if (TextUtils.isEmpty(password)) {
            Toast.makeText(getApplicationContext(),  text: "Please enter password!", Toast.LENGTH_LONG).show();
            return;
        }

        mAuth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task) {
                        if (task.isSuccessful()) {
                            Toast.makeText(getApplicationContext(),  text: "¡Login Exitoso!", Toast.LENGTH_LONG).show();


                            Intent intent = new Intent( packageContext: login.this, dashboard.class);
                            startActivity(intent);
                        }
```

```
                else {
                    Toast.makeText(getApplicationContext(), text: "No se logró iniciar sesión. Intente más tarde", Toast.LENGTH_LONG).show();

                }
            }
        });

}
```

```
    public void forgetPassword(View view ){
        Intent intent = new Intent( packageContext: this,forgetPassword.class);
        startActivity(intent);
        finish();
    }

    public void signUp(View view){
        Intent intent = new Intent( packageContext: this, signUp.class);
        startActivity(intent);
        finish();
    }
}
```

signUp.java

En esta clase se programa el primer paso del registro de usuarios tipo cliente. En la sesión de diseño, se muestran los id de los inputs o elementos utilizados en cada pantalla. En este caso se programa el ingreso de los datos personales básicos. Además del método para pasar del paso 1 al paso 2.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_sign_up);
    //Components hooks
    imvback = findViewById(R.id.btnBack);
    txtvTitle = findViewById(R.id.txtvTitle);
    btnLogin = findViewById(R.id.btnLogin);
    btnNext = findViewById(R.id.btnNext);
    edtname = findViewById(R.id.Registrername);
    edtlastname = findViewById(R.id.Registrerlastname);
    edtphone = findViewById(R.id.Registrerphone);
}
```

```
//To make the transition to sign_up2 activity
public void nextScreen1(View view){
    Intent intent = new Intent(getApplicationContext(), sign_up2.class);

    intent.putExtra( name: "name",edtname.getText().toString());
    intent.putExtra( name: "lastname",edtlastname.getText().toString());
    intent.putExtra( name: "phone",edtphone.getText().toString());

    Pair[] pairs = new Pair[3];
    pairs[0] = new Pair<View, String>(imvback, "btnBack_transition");
    pairs[1] = new Pair<View, String>(txtvTitle, "title_transition");
    pairs[2] = new Pair<View, String>(btnNext, "btnLogin_transition");
    if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.LOLLIPOP) {
        ActivityOptions options = ActivityOptions.makeSceneTransitionAnimation( activity: signUp.this, pairs);
        startActivity(intent,options.toBundle());
    }else{
        startActivity(intent);
        finish();
    }

}
```

```
public void toLogin(View view){
    Intent intent = new Intent( packageContext: this, login.class);
    startActivity(intent);
    finish();
}
```

sign_up2.java

En esta clase se programa el primer paso del registro de usuarios tipo cliente. En la sesión de diseño, se muestran los id de los inputs o elementos utilizados en cada pantalla. En este caso se programa el ingreso de los datos para iniciar sesión. Además del método para pasar del paso 2 al paso 3, y de regresar al paso 1.

```
private ImageView imvback;
private TextView txtvTitle;
private Button btnNext;
private EditText edtuser, edtemail,edtpassword,edtpasswordAgain;
String name, lastname,phone;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_sign_up2);
    //Components hooks
    imvback = findViewById(R.id.btnBack);
    txtvTitle = findViewById(R.id.txtvTitle);
    btnNext = findViewById(R.id.btnNext);
    edtuser = findViewById(R.id.RegistrerUser);
    edtemail = findViewById(R.id.RegistrerEmail);
    edtpassword = findViewById(R.id.RegistrerPassword);
    edtpasswordAgain = findViewById(R.id.RegistrerPasswordAgain);
```

```
    Bundle bundle = getIntent().getExtras();
    name = bundle.getString( key: "name");
    lastname = bundle.getString( key: "lastname");
    phone = bundle.getString( key: "phone");
}

//To make the transition to sign_up2 activity
public void nextScreen2(View view){

    if(edtpassword.getText().toString().isEmpty() || edtemail.getText().toString().isEmpty()) {
        Toast.makeText(getApplicationContext(), text: "Rellena los campos",Toast.LENGTH_LONG).show();}
    else
    {
        if(edtpassword.getText().length()<=6){Toast.makeText(getApplicationContext(), text: "La contraseña debe contener mas de 6 caracteres",Toast.LENGTH_LONG).show();}
        else{
            Intent intent = new Intent( packageContext: this, sign_up3.class);

            intent.putExtra( name: "name",name);
            intent.putExtra( name: "lastname",lastname);
            intent.putExtra( name: "phone",phone);
            intent.putExtra( name: "user",edtuser.getText().toString());
            intent.putExtra( name: "email",edtemail.getText().toString());
            intent.putExtra( name: "password",edtpassword.getText().toString());
            startActivity(intent);
            finish();
        }

    }
}
```

Externally added files can be added to Git
View Files Always Add Don't Ask Again

```
    public void toSignUp1(View view){
        Intent intent = new Intent( packageContext: this, signUp .class);
        startActivity(intent);
        finish();
    }
}
```

sign_up3.java

En esta clase se programa el primer paso del registro de usuarios tipo cliente. En la sesión de diseño, se muestran los id de los inputs o elementos utilizados en cada pantalla. En este caso se programa el ingreso de los datos personales básicos del usuario. Además del método para regresar al paso 2.

```
private String name, lastname,phone,email,password, user, date, gender;
private RadioButton radioM, radioF;
DatePicker fecha;

FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference refUsuarios = database.getReference( path: "Usuarios");
FirebaseAuth mAuth;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    mAuth = FirebaseAuth.getInstance();

    setContentView(R.layout.activity_sign_up3);
    radioM = findViewById(R.id.rbtnMale);
    radioF = findViewById(R.id.rbtnFemale);
    fecha = findViewById(R.id.DPfecha);

    if (radioM.isChecked()==true) gender = radioM.getText().toString();
    if (radioF.isChecked()==true) gender = radioF.getText().toString();
```

```
    Bundle bundle = getIntent().getExtras();
    name = bundle.getString( key: "name");
    lastname = bundle.getString( key: "lastname");
    phone = bundle.getString( key: "phone");
    email = bundle.getString( key: "email");
    password = bundle.getString( key: "password");
    user = bundle.getString( key: "user");
    date = String.valueOf(fecha.getDayOfMonth()) + "-"+String.valueOf(fecha.getMonth())+"-"+String.valueOf(fecha.getYear());
}
```

```
public void createUserAccount(View view){

    mAuth.createUserWithEmailAndPassword(email,password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    User u = new User(name,lastname,phone,user,email,password,date,gender);
                    refUsuarios.push().setValue(u);
                    Toast.makeText(getApplicationContext(), text: "Usuario Registrado",Toast.LENGTH_LONG).show();
                    Intent intent = new Intent( packageContext: sign_up3.this,login.class);
                    startActivity(intent);
                    finish();
                }
                else {
                    Toast.makeText(getApplicationContext(), text: "Fallo de registro, intente de nuevo",Toast.LENGTH_LONG).show();
                }
            }
        });
}
```

```
    public void toSignUp2(View view){
        Intent intent = new Intent( packageContext: this, sign_up2 .class);
        startActivity(intent);
        finish();
    }
}
```

MainActivity.java

En esta clase se programa la pantalla principal, con la cual inicia la app, donde se encuentra el logo de la aplicación, el nombre y el logan. Se encuentra el método para la animación inicial de la app.

```
    //variables for animation
    Animation topAnim, bottomAnim;
    //components' variables
    ImageView imgvAppLogo;
    TextView txtvAppName, txtvSlogan;
    //variable to pass actyvity
    private static  int SPLASH_SCREEN = 4000;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN);
        setContentView(R.layout.activity_main);
        //Animations
        topAnim = AnimationUtils.loadAnimation( context: this, R.anim.top_animation);
        bottomAnim = AnimationUtils.loadAnimation( context: this, R.anim.bottom_animation);
        imgvAppLogo = (ImageView) findViewById(R.id.imgvAppLogo);
        txtvAppName = (TextView)findViewById(R.id.txtvAppName);
        txtvSlogan = (TextView)findViewById(R.id.txtvSlogan);
        //setting animation to components
        imgvAppLogo.setAnimation(topAnim);
        txtvAppName.setAnimation(bottomAnim);
        txtvSlogan.setAnimation(bottomAnim);
```

```
        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                Intent intent = new Intent(getApplicationContext(), login.class);
                Pair[] pairs = new Pair[3];
                pairs[0] = new Pair<View, String>(imgvAppLogo, "logo_transition");
                pairs[1] = new Pair<View, String>(txtvAppName, "logoText_transition");
                pairs[2] = new Pair<View, String>(txtvSlogan, "slogan_transition");

                if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.LOLLIPOP) {
                    ActivityOptions options = ActivityOptions.makeSceneTransitionAnimation( activity: MainActivity.this, pairs);
                    startActivity(intent, options.toBundle());
                    finish();
                }else{
                    startActivity(intent);
                    finish();
                }
            }
        }, SPLASH_SCREEN);
    }
}
```

listService.java

En esta clase se programa la lista de servicios que se muestran en el catálogo de servicios en un recyclerview.

```
ArrayList<Service> listServices;
RecyclerView recycler;
AdapterService adapter;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_list_service);

    recycler = (RecyclerView) findViewById(R.id.listServices);

    recycler.setLayoutManager(new LinearLayoutManager( context: this,LinearLayoutManager.VERTICAL, reverseLayout: false));
    listServices = new ArrayList<Service>();

    FirebaseDatabase database = FirebaseDatabase.getInstance();
    adapter = new AdapterService(listServices);
    recycler.setAdapter(adapter);
    database.getReference( path: "Services").addValueEventListener(new ValueEventListener() {
```

```
        database.getReference( path: "Services").addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot datasnapshot) {
                listServices.removeAll(listServices);
                for(DataSnapshot snapshot : datasnapshot.getChildren()){
                    Service s = snapshot.getValue(Service.class);
                    listServices.add(s);
                }
                adapter.notifyDataSetChanged();
            }

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

            }
        });
    }

    public void back(View view){
        Intent intent = new Intent( packageContext: listService.this, dashboard.class);
        startActivity(intent);
    }
}
```

forgetPassword.java

En esta clase se programa la vista de “recuperar contraseña”, con ayuda de Firebase se realiza este proceso. Se declara la variable de correo electrónico para obtener la recuperación con Firebase.

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_forget_password);
        recorreo = findViewById(R.id.rcorreo);
        recuperarbtn = findViewById(R.id.rbtn);
        recuperarbtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                sendEmail(recorreo.getText().toString());

            }
        });


    }

    @Override
    public void onBackPressed(){

        super.onBackPressed();
        Intent intent = new Intent( packageContext: forgetPassword.this, login.class);
        startActivity(intent);
        finish();
    }
```

Externally added files can be added to Git
View Files Always Add Don't Ask Again

```
    public void sendEmail(String email){
        FirebaseAuth auth = FirebaseAuth.getInstance();
        String emailaddress = email;

        auth.sendPasswordResetEmail(emailaddress).addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                if (task.isSuccessful()){
                    Toast.makeText( context: forgetPassword.this,  text: "Correo enviado", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent( packageContext: forgetPassword.this, login.class);
                    startActivity(intent);
                    finish();
                }
                else {
                    Toast.makeText( context: forgetPassword.this,  text: "Correo invalido", Toast.LENGTH_SHORT).show();
                }
                }
        });
    }

    //back to Login
    public void backLogin(View view){
        Intent intent = new Intent( packageContext: this, login.class);
        startActivity(intent);
        finish();
    }
}
```

Externally added files can be added to Git
View Files Always Add Don't Ask Again

editAppointment.java

En esta clase se configura el “editar” o modificar la información de las citas de cada usuario.

```
private ListView listView;
private ArrayList<String> services;
TextView txtvServiceRequired;
//date & time picker's variables
TextView hour_date;
Button btnSelectTime_Date;

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

    //hooks
    listView= findViewById(R.id.listServices);
    txtvServiceRequired = findViewById(R.id.txtvServiceRequired);
    hour_date = findViewById(R.id.hour_date);
    btnSelectTime_Date = findViewById(R.id.btnSelectTime_Date);

    hour_date.setInputType(InputType.TYPE_NULL);

    //Aqui se obtendrian los servicios del catalogo
    services = new ArrayList<String>();
    services.add("Servicio 1");
    services.add("Servicio 2");
    services.add("Servicio 3");
    services.add("Servicio 4");
    services.add("Servicio 5");
```

Externally added files can be added to Git
View Files Always Add Don't Ask Aga

```
    ArrayAdapter<String> adapter = new ArrayAdapter<>( context: this, android.R.layout.simple_list_item_activated_1,services);
    listView.setAdapter(adapter);

    //Get item selected
    listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> adapterView, View view, int position, long id) {
            txtvServiceRequired.setText(services.get(position));
        }
    });

    btnSelectTime_Date.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) { showDateTimeDialog( hour_date); }
    });
}
```

```
    //Time & Date Picker's method
    private void showDateTimeDialog(final TextView date_time_in) {
        final Calendar calendar=Calendar.getInstance();
        DatePickerDialog.OnDateSetListener dateSetListener=new DatePickerDialog.OnDateSetListener() {
            @Override
            public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
                calendar.set(Calendar.YEAR,year);
                calendar.set(Calendar.MONTH,month);
                calendar.set(Calendar.DAY_OF_MONTH,dayOfMonth);

                TimePickerDialog.OnTimeSetListener timeSetListener=new TimePickerDialog.OnTimeSetListener() {
                    @Override
                    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
                        calendar.set(Calendar.HOUR_OF_DAY,hourOfDay);
                        calendar.set(Calendar.MINUTE,minute);

                        SimpleDateFormat simpleDateFormat=new SimpleDateFormat( pattern: "dd-MM-yy HH:mm");

                        date_time_in.setText(simpleDateFormat.format(calendar.getTime()));
                    }
                };

                new TimePickerDialog( context: editAppointment.this,timeSetListener,calendar.get(Calendar.HOUR_OF_DAY),calendar.get(Calendar.MINUTE), is24HourView: false).show();
            }
        };

        new DatePickerDialog( context: editAppointment.this,dateSetListener,calendar.get(Calendar.YEAR),calendar.get(Calendar.MONTH
    }
}
```

Externally added files can be added to Git
View Files Always Add Don't Ask Again

dashboard.java

En esta clase se configura el menú lateral de la aplicación, programando cada clasificación u opción que tiene cada usuario para navegar dentro de la aplicación.

```
//Menu Variables
DrawerLayout drawerLayout;
NavigationView navigationView;
Toolbar toolbar;
private FirebaseAuth mAuth;
private TextView NombreUsuario;

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

    NombreUsuario=findViewById(R.id.txtUsuario);

    //FIrebase
    mAuth=FirebaseAuth.getInstance();
    FirebaseUser currentUser= mAuth.getCurrentUser();
    NombreUsuario.setText(currentUser.getDisplayName());

    //Hooks
    drawerLayout = findViewById(R.id.drawer_layout);
    navigationView = findViewById(R.id.nav_view);
    toolbar = findViewById(R.id.toolbar);

    //Toolbar
    setSupportActionBar(toolbar);
```

Externally added files can be added to
View Files Always Add Don't Ask

```
    //Navigation Drawer Menu
    navigationView.bringToFront();
    ActionBarDrawerToggle toggle = new ActionBarDrawerToggle( activity: this, drawerLayout, toolbar,"Abrir Menu", "Cerrar Menu");
    drawerLayout.addDrawerListener(toggle);
    toggle.syncState();

    navigationView.setNavigationItemSelectedListener(this);
    navigationView.setCheckedItem(R.id.nav_hom);
}

@Override
public void onBackPressed() {
    if(drawerLayout.isDrawerOpen(GravityCompat.START)){
        drawerLayout.closeDrawer(GravityCompat.START);
    }else{
        super.onBackPressed();
    }

}
public void SignOut(){
    mAuth.signOut();
    Intent intent = new Intent(getApplicationContext(), login.class);
    startActivity(intent);
    finish();
}
```

```
//go to other acts
@Override
public boolean onNavigationItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()){
        case R.id.nav_hom:
            break;
        case R.id.nav_config:
            break;
        case R.id.nav_help:
            break;
        case R.id.nav_addAppointment:
            Intent intent= new Intent( packageContext: dashboard.this, addAppointment.class);
            startActivity(intent);
            finish();
            break;
```

```
            case R.id.nav_seeAppointments:
                Intent intent2 = new Intent( packageContext: dashboard.this, listOfAppointments.class);
                startActivity(intent2);
                break;
            case R.id.nav_addServices:
                Intent intent3 = new Intent( packageContext: dashboard.this, addCatalogue.class);
                startActivity(intent3);
                break;
            case R.id.nav_seeServices:
                Intent intent4 = new Intent( packageContext: dashboard.this, listService.class);
                startActivity(intent4);
                break;
            case R.id.navSesion:
                SignOut();
                break;
        }
        drawerLayout.closeDrawer(GravityCompat.START);
        return true;
    }
}
```

Externally added files can be added to Git
View Files Always Add Don't Ask Ag

addCatalogue.java

En esta clase se declaran los métodos que se utilizan para agregar los servicios al catálogo, siempre que sea un usuario administrador.

```
public static FirebaseDatabase database = FirebaseDatabase.getInstance();
public static DatabaseReference refServicios = database.getReference( path: "Services");
EditText edtName, edtDescription, edtPrice, edtTime;

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

    edtName = findViewById(R.id.txtServiceName);
    edtDescription = findViewById(R.id.txtDescription);
    edtPrice = findViewById(R.id.txtPrice);
    edtTime = findViewById(R.id.txtTime);
}

public void back(View view){
    Intent intent = new Intent( packageContext: addCatalogue.this, dashboard.class);
    startActivity(intent);
}
```

```
    public void CleanInputs(){
        edtName.setText("");
        edtDescription.setText("");
        edtPrice.setText("");
        edtTime.setText("");
    }

    public void add(View view) {
        String nombre = edtName.getText().toString();
        String descripcion = edtDescription.getText().toString();
        String precio = edtPrice.getText().toString();
        String tiempo = edtTime.getText().toString();

            Service servicio = new Service();
            servicio.setName(nombre);
            servicio.setDescription(descripcion);
            servicio.setPrice(precio);
            servicio.setTime(tiempo);
            refServicios.push().setValue(servicio);
            CleanInputs();

    }
}
```

addAppointment.java

En esta clase se programa el "insertar" de las citas de los usuarios. Se han empleado alrededor de 5 métodos para el agregar citas para cada usuario. Algunos de estos son para obtener la información de cada elemento donde se ha seleccionado.

```
public class addAppointment extends AppCompatActivity {

    ArrayList<Service> listServices;
    RecyclerView recycler;
    AdapterService adapter;
private Service servicio;
    TextView txtvServiceRequired;
    //date & time picker's variables
    TextView hour_date;
    Button btnSelectTime_Date , btnAddAppointment;
    EditText edtUser, edtDescription;

    public static FirebaseDatabase database = FirebaseDatabase.getInstance();
    public static DatabaseReference refAppointments = database.getReference( path: "Appointments");
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_appointment);
        //hooks
        recycler = findViewById(R.id.listServicesAppointment);
```

```
    txtvServiceRequired = findViewById(R.id.txtvServiceRequired);
    hour_date = findViewById(R.id.hour_date);
    btnSelectTime_Date = findViewById(R.id.btnSelectTime_Date);
    btnAddAppointment = findViewById(R.id.btnaddAppointment);
    hour_date.setInputType(InputType.TYPE_NULL);
    edtUser = findViewById(R.id.user);
    edtDescription = findViewById(R.id.observation);



    recycler.setLayoutManager(new LinearLayoutManager( context: this,LinearLayoutManager.VERTICAL, reverseLayout: false));
    listServices = new ArrayList<Service>();
    adapter = new AdapterService(listServices);
    adapter.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            txtvServiceRequired.setText(listServices.get(recycler.getChildAdapterPosition(v)).getName());
        }
    });
    recycler.setAdapter(adapter);
```

```
    database.getReference( path: "Services").addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot datasnapshot) {
            listServices.removeAll(listServices);
            for(DataSnapshot snapshot : datasnapshot.getChildren()){
                Service s = snapshot.getValue(Service.class);
                listServices.add(s);
            }
            adapter.notifyDataSetChanged();
        }

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

        }
    });
```

```
    btnSelectTime_Date.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) { showDateTimeDialog( hour_date); }
    });

    btnAddAppointment.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) { AddAppointment(); }
    });

}
```

```
//Time & Date Picker's method
private void showDateTimeDialog(final TextView date_time_in) {
    final Calendar calendar=Calendar.getInstance();
    DatePickerDialog.OnDateSetListener dateSetListener=new DatePickerDialog.OnDateSetListener() {
        @Override
        public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
            calendar.set(Calendar.YEAR,year);
            calendar.set(Calendar.MONTH,month);
            calendar.set(Calendar.DAY_OF_MONTH,dayOfMonth);

            TimePickerDialog.OnTimeSetListener timeSetListener=new TimePickerDialog.OnTimeSetListener() {
                @Override
                public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
                    calendar.set(Calendar.HOUR_OF_DAY,hourOfDay);
                    calendar.set(Calendar.MINUTE,minute);

                    SimpleDateFormat simpleDateFormat=new SimpleDateFormat( pattern: "dd-MM-yy HH:mm");

                    date_time_in.setText(simpleDateFormat.format(calendar.getTime()));
                }
            };

            new TimePickerDialog( context: addAppointment.this,timeSetListener,calendar.get(Calendar.HOUR_OF_DAY),calendar.get(Calendar.MINUTE), is24HourView: false).show();
        }
    };

    new DatePickerDialog( context: addAppointment.this,dateSetListener,calendar.get(Calendar.YEAR),calendar.get(Calendar.MONTH),calendar.get(Calendar.DAY_OF_MONTH)).show();
}
```

```
    public void back(View view){
        Intent intent = new Intent( packageContext: addAppointment.this, dashboard.class);
        startActivity(intent);
    }

    public void AddAppointment(){
        String usuario = edtUser.getText().toString();
        String descripcion = edtDescription.getText().toString();
        Appointment A = new Appointment(usuario,descripcion,hour_date.getText().toString(),txtvServiceRequired.getText().toString());
        refAppointments.push().setValue(A);
        Toast.makeText(getApplicationContext(), text: "Cita Realizada",Toast.LENGTH_LONG).show();
        CleanInputs();
    }

    public void CleanInputs(){
        txtvServiceRequired.setText("Selección");
        hour_date.setText("");
        edtDescription.setText("");
        edtUser.setText("");
    }
}
```

addUser.java

En esta clase se programa el “agregar” un nuevo usuario, que tendrá acceso a la aplicación. En esta clase se programa el primer paso, en el cual se especifican: nombre, apellido y número de teléfono. Luego se encuentra el botón ‘siguiente’ que permite avanzar hacia el siguiente paso.

```
public class addUser extends AppCompatActivity {

    private String tipo, usuario;
    private TextView type, user;
    private EditText Nombre,Apellido,Telefono;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_user);
        Bundle bundle = getIntent().getExtras();
        tipo = bundle.getString( key: "type");
        usuario = bundle.getString( key: "user");

        Nombre = findViewById(R.id.txtName);
        Apellido = findViewById(R.id.txtLastName);
        Telefono = findViewById(R.id.txtPhone);
        type = findViewById(R.id.txtUserAccount);
        user = findViewById(R.id.txtTypeUserAccount);
        type.setText(tipo);
        user.setText(usuario);
    }
```

```
    public void nextAddUser(View view){
        Intent intent = new Intent( packageContext: addUser.this, addUser2.class);
        intent.putExtra( name: "name",Nombre.getText().toString());
        intent.putExtra( name: "apellido",Apellido.getText().toString());
        intent.putExtra( name: "telefono",Telefono.getText().toString());
        intent.putExtra( name: "type",tipo);
        intent.putExtra( name: "user",usuario);
        startActivity(intent);
    }
    //Back to dashboard
    public void backDash1(View view){
        Intent intent = new Intent( packageContext: addUser.this, dashboard.class);
        intent.putExtra( name: "type",tipo);
        intent.putExtra( name: "user",usuario);
        startActivity(intent);
    }
}
```

addUser2.java

En esta clase se programa el segundo paso del “agregar” un nuevo usuario, que tendrá acceso a la aplicación. En esta clase se obtienen los datos de: correo, contraseña y usuario. Además, se permite avanzar al siguiente paso.

```java
public class addUser2 extends AppCompatActivity {

    private EditText edtEmail,edtPassword,edtUser;
    private String usuario, tipo, nombre,apellido,telefono;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_user2);

        Bundle bundle = getIntent().getExtras();
        usuario = bundle.getString( key: "user");
        tipo = bundle.getString( key: "type");
        nombre = bundle.getString( key: "name");
        apellido = bundle.getString( key: "apellido");
        telefono = bundle.getString( key: "telefono");

        edtEmail = findViewById(R.id.txtAddUserEmail);
        edtPassword = findViewById(R.id.txtAddUserPassword);
        edtUser = findViewById(R.id.txtAddUserUsuario);
    }
```

```java
public void nextAddUser3(View view){
    if(edtEmail.getText().toString().isEmpty() || edtPassword.getText().toString().isEmpty()){
        Toast.makeText(getApplicationContext(), text: "Rellena los campos",Toast.LENGTH_LONG).show();
    }

    else {
        if(edtPassword.getText().length()<=6){
            Toast.makeText(getApplicationContext(), text: "La contraseña debe contener mas de 6 caracteres",Toast.LENGTH_LONG).show();
        }
        else{
            Intent intent = new Intent( packageContext: addUser2.this,addUser3.class);
            intent.putExtra( name: "name",nombre);
            intent.putExtra( name: "apellido",apellido);
            intent.putExtra( name: "telefono",telefono);
            intent.putExtra( name: "type",tipo);
            intent.putExtra( name: "user",usuario);
            intent.putExtra( name: "email",edtEmail.getText().toString());
            intent.putExtra( name: "password",edtPassword.getText().toString());
            intent.putExtra( name: "usuario",edtUser.getText().toString());
            startActivity(intent);
            finish();
        }
    }
}
```

```
    public void backAddUser1(View view){
        Intent intent = new Intent( packageContext: addUser2.this, addUser.class);
        intent.putExtra( name: "type",tipo);
        intent.putExtra( name: "user",usuario);
        startActivity(intent);
    }
}
```

addUser3.java

En esta clase se programa el último paso del "agregar" un nuevo usuario, que tendrá acceso a la aplicación. En esta clase se obtienen los datos de: genero de usuario, tipo de usuario y la fecha de nacimiento. Además, se encuentra el botón de agregar usuario.

```
public class addUser3 extends AppCompatActivity {

    FirebaseDatabase database = FirebaseDatabase.getInstance();
    DatabaseReference refUsuarios = database.getReference( path: "Usuarios");
    FirebaseAuth mAuth;

    private String tipo, usuario, name, lastname, phone, email, password, gender, date, type, user;
    private RadioButton rbtnAdmin, rbtnClient, rbtnMale, rbtnFemale;
    private DatePicker fechaNac;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_user3);
        mAuth = FirebaseAuth.getInstance();

        Bundle bundle = getIntent().getExtras();
        tipo = bundle.getString( key: "type");
        usuario = bundle.getString( key: "user");
        name = bundle.getString( key: "name");
        lastname = bundle.getString( key: "apellido");
        phone = bundle.getString( key: "telefono");
        email = bundle.getString( key: "email");
        password = bundle.getString( key: "password");
        user = bundle.getString( key: "usuario");
```

```
        rbtnAdmin = findViewById(R.id.rdbTypeAdmin);
        rbtnClient = findViewById(R.id.rdbTypeClient);
        rbtnMale = findViewById(R.id.rbtnAddUserMale);
        rbtnFemale = findViewById(R.id.rbtnAddUserFemale);
        fechaNac = findViewById(R.id.fechaAddUser);
    }

    public void backAddUser2(View view){
        Intent intent = new Intent( packageContext: addUser3.this, addUser2.class);
        intent.putExtra( name: "name",name);
        intent.putExtra( name: "apellido", lastname);
        intent.putExtra( name: "telefono",phone);
        intent.putExtra( name: "type",tipo);
        intent.putExtra( name: "user",usuario);
        startActivity(intent);
    }
```

```
//To create user account
public void createUserAccount(View view){

    if (rbtnAdmin.isChecked()==true) type = "Admin";
    if (rbtnClient.isChecked()==true) type ="cliente";

    if (rbtnMale.isChecked()==true) gender = "Masculino";
    if (rbtnFemale.isChecked()==true) gender = "Femenino";

    date = String.valueOf(fechaNac.getDayOfMonth()) + "-"+String.valueOf(fechaNac.getMonth())+"-"+String.valueOf(fechaNac.getYear());

    mAuth.createUserWithEmailAndPassword(email,password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if(task.isSuccessful()){
                User u = new User(name,lastname,phone,user,email,password,date,gender,type);
                u.setKey(UUID.randomUUID().toString());
                refUsuarios.push().setValue(u);
                Toast.makeText(getApplicationContext(), text: "Usuario Registrado",Toast.LENGTH_LONG).show();
                Intent intent = new Intent( packageContext: addUser3.this, dashboard.class);
                intent.putExtra( name: "type",tipo);
                intent.putExtra( name: "user",usuario);
                startActivity(intent);
            }
            else {
                Toast.makeText(getApplicationContext(), text: "Fallo de registro, intente de nuevo",Toast.LENGTH_LONG).show();
            }
        }
    });
}
```

editAppointment.java

En esta clase se programa la opción “editar” citas según el usuario. Las opciones a modificar son: la hora de la cita, el servicio y agregar una breve descripción de la cita. Luego se presentan los botones de “editar” y “cancelar”.

```
public class editAppointment extends AppCompatActivity {

    ArrayList<Service> listServices;
    RecyclerView recycler;
    AdapterService adapter;
    Service servicio;
    Appointment appointment;
    TextView txtvServiceRequired, typeuser, User;
    //date & time picker's variables
    TextView hour_date;
    Button btnSelectTime_Date, btnEditAppointment;
    EditText edtUser, edtDescription;
    String date, description,key,service,user;

    com.google.android.material.textfield.TextInputLayout mask;
    private String tipoUsuario;
    private String usuario;

    public static FirebaseDatabase database = FirebaseDatabase.getInstance();
    public static DatabaseReference refAppointments = database.getReference( path: "Appointments");
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_edit_appointment);

        Bundle bundle = getIntent().getExtras();
        tipoUsuario = bundle.getString( key: "type");
        usuario = bundle.getString( key: "usuario");
        date = bundle.getString( key: "date");
        description = bundle.getString( key: "description");
        key = bundle.getString( key: "key");
        service = bundle.getString( key: "service");
        user = bundle.getString( key: "user");

        //hooks
        recycler = findViewById(R.id.listServicesAppointmentEdit);
        typeuser = (TextView) findViewById(R.id.txtTypeUserEdit);
        mask = findViewById(R.id.maskEdit);
        User = (TextView) findViewById(R.id.txtUserAEdit);


        txtvServiceRequired = findViewById(R.id.txtvServiceRequiredEdit);
        hour_date = findViewById(R.id.hour_dateEdit);
        btnSelectTime_Date = findViewById(R.id.btnSelectTime_Date);
        btnEditAppointment = findViewById(R.id.btnEditAppointment);
        hour_date.setInputType(InputType.TYPE_NULL);
        edtUser = findViewById(R.id.userEdit);
        edtDescription = findViewById(R.id.observationEdit);
```

```
    edtDescription.setText(description);
    hour_date.setText(date);
    txtvServiceRequired.setText(service);
    typeuser.setText(tipoUsuario);
    User.setText(usuario);
    edtUser.setText(user);
    edtUser.setEnabled(false);

    recycler.setLayoutManager(new LinearLayoutManager( context: this,LinearLayoutManager.VERTICAL, reverseLayout: false));
    listServices = new ArrayList<Service>();
    adapter = new AdapterService(listServices);
    adapter.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            txtvServiceRequired.setText(listServices.get(recycler.getChildAdapterPosition(v)).getName());
        }
    });
    recycler.setAdapter(adapter);
```

```
    database.getReference( path: "Services").addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot datasnapshot) {
            listServices.removeAll(listServices);
            for(DataSnapshot snapshot : datasnapshot.getChildren()){
                Service s = snapshot.getValue(Service.class);
                listServices.add(s);
            }
            adapter.notifyDataSetChanged();
        }

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

        }
    });

    btnSelectTime_Date.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) { showDateTimeDialog( hour_date); }
    });
}
```

```
public void back(View view){
    Intent intent = new Intent( packageContext: editAppointment.this, dashboard.class);
    intent.putExtra( name: "type",tipoUsuario);
    intent.putExtra( name: "user",usuario);
    startActivity(intent);
}
```

```
//Time & Date Picker's method
private void showDateTimeDialog(final TextView date_time_in) {
    final Calendar calendar=Calendar.getInstance();
    DatePickerDialog.OnDateSetListener dateSetListener=new DatePickerDialog.OnDateSetListener() {
        @Override
        public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
            calendar.set(Calendar.YEAR,year);
            calendar.set(Calendar.MONTH,month);
            calendar.set(Calendar.DAY_OF_MONTH,dayOfMonth);

            TimePickerDialog.OnTimeSetListener timeSetListener=new TimePickerDialog.OnTimeSetListener() {
                @Override
                public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
                    calendar.set(Calendar.HOUR_OF_DAY,hourOfDay);
                    calendar.set(Calendar.MINUTE,minute);

                    SimpleDateFormat simpleDateFormat=new SimpleDateFormat( pattern: "dd-MM-yy HH:mm");

                    date_time_in.setText(simpleDateFormat.format(calendar.getTime()));
                }
            };

            new TimePickerDialog( context: editAppointment.this,timeSetListener,calendar.get(Calendar.HOUR_OF_DAY),calendar.get(Calendar.MINUTE), is24HourView: false).show();
        }
    };

    new DatePickerDialog( context: editAppointment.this,dateSetListener,calendar.get(Calendar.YEAR),calendar.get(Calendar.MONTH),calendar.get(Calendar.DAY_OF_MONTH)).show();
}
```

```
public void EditAppointment(View view){
    String Usuario="";
    if(tipoUsuario.equals("cliente")){Usuario = usuario;}
    else{
        Usuario = edtUser.getText().toString();
    }

    String descripcion = edtDescription.getText().toString();
    Appointment A = new Appointment(Usuario,descripcion,hour_date.getText().toString(),txtvServiceRequired.getText().toString(), key);
    refAppointments.child(key).setValue(A);
    Toast.makeText(getApplicationContext(), text: "Cita Modificada",Toast.LENGTH_LONG).show();
    Intent intent = new Intent( packageContext: editAppointment.this, dashboard.class);
    intent.putExtra( name: "type",tipoUsuario);
    intent.putExtra( name: "user",usuario);
    startActivity(intent);
}
```

BottomSheet.java

En esta clase se programa la opción “Eliminar” una cita. Antes de eliminarla, se ha programado una notificación de confirmación de acción.

```
public class BottomSheet extends BottomSheetDialogFragment {
    FirebaseDatabase database;
    DatabaseReference refAppointment;
    String key, tipo = "", usuario = "", date,description,service,User;
    Button btnUpdate, btnDelete;

    public BottomSheet(Appointment appointment, String type, String user) {
        database = FirebaseDatabase.getInstance();
        refAppointment = database.getReference();
        key = appointment.getKey();
        date = appointment.getDate();
        description = appointment.getDescription();
        service = appointment.getService();
        User = appointment.getUser();
        tipo = type;
        usuario = user;
    }
```

```
@Nullable
@Override
public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable  Bundle savedInstanceState) {

    View view = inflater.inflate(R.layout.row_add_item,container,  attachToRoot: false);
    btnUpdate = view.findViewById(R.id.btnUpdate);
    btnDelete = view.findViewById(R.id.btnDelete);

    btnUpdate.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(getContext(), editAppointment.class);
            intent.putExtra( name: "date",date);
            intent.putExtra( name: "description",description);
            intent.putExtra( name: "key",key);
            intent.putExtra( name: "service",service);
            intent.putExtra( name: "user", User);
            intent.putExtra( name: "type",tipo);
            intent.putExtra( name: "usuario",usuario);
            startActivity(intent);
        }
    });
```

```
    btnDelete.setOnClickListener(new View.OnClickListener(){
        @Override
        public void onClick(View v) {
            AlertDialog.Builder ad = new AlertDialog.Builder(getContext());
            ad.setMessage("¿Está seguro de eliminar la cita?").setTitle("Confirmación");

            ad.setPositiveButton( text: "Sí", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    refAppointment.child("Appointments").child(key).removeValue();
                    Toast.makeText(getContext(), text: "Cita eliminada",Toast.LENGTH_LONG).show();
                }
            });

            ad.setNegativeButton( text: "No", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(getContext(), text: "Operacion cancelada", Toast.LENGTH_LONG).show();
                }
            });
            ad.show();
        }
    });

    return view;
}

}
```

ListOfAppointments.java

En esta clase se programa la opción “Ver” citas por usuario. En ella se acumulan las citas que tiene programadas el usuario, es decir su historial. Hace referencia al modelo llamado *Appointments.*

```
public class listOfAppointments extends AppCompatActivity {

    ArrayList<Appointment> listAppointments;
    RecyclerView recycler;
    AdapterAppointment adapter;
    private String tipoUser, user;
    TextView type, Usuario;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_list_of_appointments);

        Bundle bundle = getIntent().getExtras();
        tipoUser = bundle.getString( key: "type");
        user = bundle.getString( key: "user");

        recycler = (RecyclerView) findViewById(R.id.listAppointments);
        type = (TextView) findViewById(R.id.txtTipoUsuario);
        Usuario = (TextView) findViewById(R.id.txtuser);
        type.setText(tipoUser);
        Usuario.setText(user);

        recycler.setLayoutManager(new LinearLayoutManager( context: this,LinearLayoutManager.VERTICAL, reverseLayout: false));
        listAppointments = new ArrayList<Appointment>();
```

```
        FirebaseDatabase database = FirebaseDatabase.getInstance();
        adapter = new AdapterAppointment(listAppointments);
        recycler.setAdapter(adapter);

        if(tipoUser.equals("cliente")){
            Query query = database.getReference( path: "Appointments").orderByChild("user").equalTo(user);
            query.addValueEventListener(new ValueEventListener() {
                @Override
                public void onDataChange(@NonNull DataSnapshot datasnapshot) {
                    listAppointments.removeAll(listAppointments);
                    for(DataSnapshot snapshot : datasnapshot.getChildren()){
                        Appointment a = snapshot.getValue(Appointment.class);
                        listAppointments.add(a);
                    }
                    adapter.notifyDataSetChanged();
                }

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

                }
            });
        }
```

```
        else{
            database.getReference( path: "Appointments").addValueEventListener(new ValueEventListener() {
                @Override
                public void onDataChange(@NonNull DataSnapshot datasnapshot) {
                    listAppointments.removeAll(listAppointments);
                    for(DataSnapshot snapshot : datasnapshot.getChildren()){
                        Appointment a = snapshot.getValue(Appointment.class);
                        listAppointments.add(a);
                    }
                    adapter.notifyDataSetChanged();
                }

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

                }
            });
        }


    }
```

```
    public void back(View view){
        Intent intent = new Intent( packageContext: listOfAppointments.this, dashboard.class);
        intent.putExtra( name: "type",tipoUser);
        intent.putExtra( name: "user",user);
        startActivity(intent);
    }
}
```

UserProfile.java

En esta clase se muestra el perfil con la información básica del usuario. Además, le permite modificar información como el nombre, apellido, correo electrónico, contraseña y el número de teléfono.

```
public class userProfile extends AppCompatActivity {
    FirebaseDatabase database;
    ImageButton btnBack;
    Button btnUpdate;
    private TextView userp, typeuserp;
    private EditText edtNombre, edtApellido, edtTelefono, edtContra,edtCorreo;
    private String tipo, usuario, nombres, apellidos, telefono, contra,fecha,genero, user,key;

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

        //hooks
        btnBack = findViewById(R.id.btnBackDash);
        btnUpdate = findViewById(R.id.btnUpdate);

        Bundle bundle = getIntent().getExtras();
        tipo = bundle.getString( key: "type");
        usuario = bundle.getString( key: "user");

        userp=(TextView)findViewById(R.id.userp);
        typeuserp=(TextView)findViewById(R.id.typeuserp);
        userp.setText(usuario);
        typeuserp.setText(tipo);
```

```
edtNombre = findViewById(R.id.txtNameProfile);
edtApellido = findViewById(R.id.txtLasNameProfile);
edtTelefono = findViewById(R.id.txtPhoneProfile);
edtContra = findViewById(R.id.txtPasswordProfile);
edtCorreo = findViewById(R.id.txtEmailProfile);

database = FirebaseDatabase.getInstance();
Query query = database.getReference( path: "Usuarios").orderByChild("email").equalTo(usuario);
query.addValueEventListener(new ValueEventListener() {
    int count = 0;
    String TipoUsuario ="";
    String correo ="";
```

```
    public void onDataChange(@NonNull DataSnapshot datasnapshot) {

        for(DataSnapshot snapshot: datasnapshot.getChildren()){
            User u = snapshot.getValue(User.class);
            fecha = u.getBirthday();
            correo = u.getEmail();
            genero = u.getGender();
            nombres = u.getName();
            apellidos = u.getLastname();
            contra = u.getPassword();
            telefono = u.getPhone();
            user = u.getUser();
            key = u.getKey();
            count++;
        }
        edtNombre.setText(nombres);
        edtApellido.setText(apellidos);
        edtCorreo.setText(correo);
        edtContra.setText(contra);
        edtTelefono.setText(telefono);
    }

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

    }
});
```

```
    public void updateData(View view){
        User u = new User();
        u.setBirthday(fecha);
        u.setEmail(edtCorreo.getText().toString());
        u.setGender(genero);
        u.setName(edtNombre.getText().toString());
        u.setLastname(edtApellido.getText().toString());
        u.setPassword(edtContra.getText().toString());
        u.setPhone(edtTelefono.getText().toString());
        u.setType(tipo);
        u.setUser(user);
        u.setKey(key);
        DatabaseReference refUsuarios = database.getReference( path: "Usuarios");
        refUsuarios.child(u.getKey()).setValue(u);
        Toast.makeText( context: this, text: "Datos Actualizados", Toast.LENGTH_LONG).show();
    }

    public void backDash(View view){
        Intent intent = new Intent( packageContext: userProfile.this, dashboard.class);
        intent.putExtra( name: "type",tipo);
        intent.putExtra( name: "user",usuario);
        startActivity(intent);
    }
}
```

MODELS

Appointment.java

En este modelo, se almacenan las variables que capturan la información que se ingresa sobre las citas. Para posteriormente, añadirlas en una lista en la clase llamada “ListOfAppointments”.

```
public class Appointment implements Serializable {
    private String key;
    private String user;
    private String description;
    private String date;
    private String service;

    public Appointment(String user, String description, String date, String service,String key) {
        this.user = user;
        this.description = description;
        this.date = date;
        this.service = service;
        this.key = key;
    }

    public Appointment() {
    }

    public String getKey() { return key; }

    public void setKey(String key) { this.key = key; }

    public String getUser() { return user; }

    public void setUser(String user) { this.user = user; }
```

```
    public String getDescription() { return description; }

    public void setDescription(String description) { this.description = description; }

    public String getDate() { return date; }

    public void setDate(String date) { this.date = date; }

    public String getService() { return service; }

    public void setService(String service) { this.service = service; }
}
```

Service.java

En este modelo, se almacenan las variables que capturan la información que se ingresa sobre los servicios. Para posteriormente, añadirlas en una lista en la clase llamada “ListService”.

```
public class Service {
    private String name;
    private String description;
    private String time;
    private String price;

    public Service() {
    }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public String getDescription() { return description; }

    public void setDescription(String description) { this.description = description; }

    public String getTime() { return time; }

    public void setTime(String time) { this.time = time; }

    public String getPrice() { return price; }

    public void setPrice(String price) { this.price = price; }
}
```

User.java

En este modelo, se almacenan las variables que capturan la información que se ingresa sobre los usuarios. Para posteriormente, referenciarlas en userProfile.

```
public class User {
    private String key;
    private String name;
    private String lastname;
    private String phone;
    private String user;
    private String email;
    private String password;
    private String birthday;
    private String gender;
    private String type;

    public User(String name, String lastname, String phone, String user, String email, String password, String birthday, String gender, String type) {
        this.name = name;
        this.lastname = lastname;
        this.phone = phone;
        this.user = user;
        this.email = email;
        this.password = password;
        this.birthday = birthday;
        this.gender = gender;
        this.type = type;
    }

    public User() {
    }
```

```
    public String getKey() { return key; }

    public void setKey(String key) { this.key = key; }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public String getLastname() { return lastname; }

    public void setLastname(String lastname) { this.lastname = lastname; }

    public String getPhone() { return phone; }

    public void setPhone(String phone) { this.phone = phone; }

    public String getUser() { return user; }

    public void setUser(String user) { this.user = user; }

    public String getEmail() { return email; }

    public void setEmail(String email) { this.email = email; }

    public String getPassword() { return password; }

    public void setPassword(String password) { this.password = password; }
```

```
    public String getBirthday() { return birthday; }

    public void setBirthday(String birthday) { this.birthday = birthday; }

    public String getGender() { return gender; }

    public void setGender(String gender) { this.gender = gender; }

    public String getType() { return type; }

    public void setType(String type) { this.type = type; }
}
```

ADAPTERS

AdapterAppointment.java

```
public class AdapterAppointment extends RecyclerView.Adapter<AdapterAppointment.ViewHolderAppointment> {
    ArrayList<Appointment> listAppointments;

    public AdapterAppointment(ArrayList<Appointment> listAppointments){this.listAppointments = listAppointments;}
    @Override
    public ViewHolderAppointment onCreateViewHolder(ViewGroup parent, int viewType){
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.items_appointments, root: null, attachToRoot: false);
        return new ViewHolderAppointment(view);
    }

    @Override
    public void onBindViewHolder(ViewHolderAppointment holder, int position){
        holder.asignarDatos(listAppointments.get(position));
    }
```

```
    @Override
    public int getItemCount() { return listAppointments.size(); }


    public class ViewHolderAppointment extends RecyclerView.ViewHolder{
        TextView usuario, descripcion,fecha, servicio;

        public ViewHolderAppointment(View itemView){
            super(itemView);

            usuario = (TextView) itemView.findViewById(R.id.item_userAppointment);
            descripcion = (TextView) itemView.findViewById(R.id.item_descriptionAppointment);
            fecha = (TextView) itemView.findViewById(R.id.item_dateAppointment);
            servicio = (TextView) itemView.findViewById(R.id.item_serviceAppointment);
        }

        public void asignarDatos(Appointment a){

            usuario.setText("Usuario: "+a.getUser());
            descripcion.setText("Nota: "+a.getDescription());
            fecha.setText("Fecha: "+a.getDate());
            servicio.setText("Servicio: " +a.getService());

        }
    }
}
```

AdapterService.java

```java
public class AdapterService extends RecyclerView.Adapter<AdapterService.ViewHolderService> implements View.OnClickListener{
    ArrayList<Service> listServices;
    private View.OnClickListener listener;
    public AdapterService(ArrayList<Service> listservices) { this.listServices = listservices; }

    @Override
    public ViewHolderService onCreateViewHolder(ViewGroup parent, int viewType){
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.items_services, root: null, attachToRoot: false);
        view.setOnClickListener(this);
        return new ViewHolderService(view);
    }

    @Override
    public void onBindViewHolder(ViewHolderService holder, int position){
        holder.asignarDatos(listServices.get(position));
    }
```

```java
    @Override
    public int getItemCount() { return listServices.size(); }

    public void setOnClickListener(View.OnClickListener listener) { this.listener = listener; }

    @Override
    public void onClick(View v) {
    if(listener != null)
    {
        listener.onClick(v);
                }
    }
```

```java
    public class ViewHolderService extends RecyclerView.ViewHolder{
        TextView nombre, descripcion,precio, tiempo;

        public ViewHolderService(View itemView){
            super(itemView);

            nombre = (TextView) itemView.findViewById(R.id.item_name);
            descripcion = (TextView) itemView.findViewById(R.id.item_description);
            precio = (TextView) itemView.findViewById(R.id.itemPrice);
            tiempo = (TextView) itemView.findViewById(R.id.item_time);
        }

        public void asignarDatos(Service s){

            nombre.setText(s.getName());
            descripcion.setText(s.getDescription());
            precio.setText("Precio: $"+s.getPrice());
            tiempo.setText("Duracion: " +s.getTime() +" minutos");

        }
    }
}
```

AdapterAppointmentDashboard.java

```java
public class AdapterAppointmentDashboard extends RecyclerView.Adapter<AdapterAppointmentDashboard.ViewHolderAppointment> implements View.OnClickListener{

    ArrayList<Appointment> listAppointments;
    private View.OnClickListener listener;
    public AdapterAppointmentDashboard(ArrayList<Appointment> listAppointments){
        this.listAppointments = listAppointments;
    }
    @Override
    public ViewHolderAppointment onCreateViewHolder(ViewGroup parent, int viewType){
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.items_appointments_dasboard, root: null, attachToRoot: false);
        view.setOnClickListener(this);
        return new ViewHolderAppointment(view);
    }

    @Override
    public void onBindViewHolder(ViewHolderAppointment holder, int position){
        holder.asignarDatos(listAppointments.get(position));
    }
```

```java
    @Override
    public int getItemCount() { return listAppointments.size(); }

    public void setOnclickListener(View.OnClickListener listener){this.listener = listener;}

    @Override
    public void onClick(View v) {
        if(listener != null)
        {
            listener.onClick(v);
        }
    }
    public class ViewHolderAppointment extends RecyclerView.ViewHolder{
        TextView usuario, descripcion,fecha, servicio;

        public ViewHolderAppointment(View itemView){
            super(itemView);

            usuario = (TextView) itemView.findViewById(R.id.userAppointmentDashboard);
            descripcion = (TextView) itemView.findViewById(R.id.descriptionAppointmentDashboard);
            fecha = (TextView) itemView.findViewById(R.id.dateAppointmentDashboard);
            servicio = (TextView) itemView.findViewById(R.id.hourAppointmentDashboard);
        }
```

```java
        public void asignarDatos(Appointment a){

            usuario.setText("Usuario: "+a.getUser());
            descripcion.setText("Nota: "+a.getDescription());
            fecha.setText("Fecha: "+a.getDate());
            servicio.setText("Servicio: " +a.getService());


        }
    }

}
```