Tchê Linux

Tecnologia assistiva para deficientes visuais no uso de transporte público

Nome: Everton Luis de Almeida

Apresentação Pessoal

- Formado em tecnologia em analise e desenvolvimento de sistemas no IFRS – Rio Grande;
- Atuo na área de desenvolvimento de software a mais de 2 anos;
- Mais de 300hrs de cursos back-end (Java, .NET, Node.js, entre outros);
- Mais de 200hrs de cursos front-end (HTML, CSS, JAVASCRIPT, entre outros);
- Mais de 100hrs de cursos de inovação, infra estrutura e gestão;
- Faço parte da organização do Tchelinux a 1 ano;
- Palestrante a 2 anos;

Introdução

O diálogo entre os seres vivos que constituem uma sociedade não se limita simplesmente à leitura e à escrita tradicional, pois há diversos meios de um ser se comunicar com outro. Esses meios ocorrem devido ao fato de haver indivíduos com limitações físicas, cuja forma pela qual eles se expressam é restrita.

Introdução

Pensando nisso e com grande avanço da tecnologia, foram surgindo diversos facilitadores para ajudar essas pessoas com necessidades especiais. E assim surgiu o termo tecnologia assistiva.

Tecnologia Assistiva (TA)

É um termo ainda novo, utilizado para identificar todo o arsenal de recursos e serviços que contribuem para proporcionar ou ampliar habilidades funcionais de pessoas com deficiência e, consequentemente promover vida independente e inclusão (BERSCH, 2013).

Tecnologia Assistiva (TA)

TA para deficientes visuais

Virtual Vision
Acessibilidade para pessoas com deficiência visual

Talk Back

JAWS

NVDA

DOS VOX

TA para deficientes visuais

Esta é uma versão exclusiva para deficiente visuais

CittaMobi acessibilidade

moovit

TAGPOINT

wayfindr community

Beacons

São pequenos aparelhos de proximidade que emitem informações, por meio da tecnologia *bluetooth*, diretamente aos *smartphones* cadastrados.

Funcionamento geral do *Beacon*

➢***Universal Unique Identifier* (UUID):** É uma sequência de 16 *bytes* usada para diferenciar um grande grupo de *beacons* relacionados.

➢***Mor*:** É uma sequência de 2 *bytes* usada para distinguir um subconjunto menor de *beacons* dentro de um grupo maior.

➢***Minor*:** É uma sequência de 2 *bytes* destinada a identificar *beacons* individuais.

➢**Tx Potência:** É usada para determinar a distância do emissor (*Beacon*), também conhecido como farol.

Objetivo

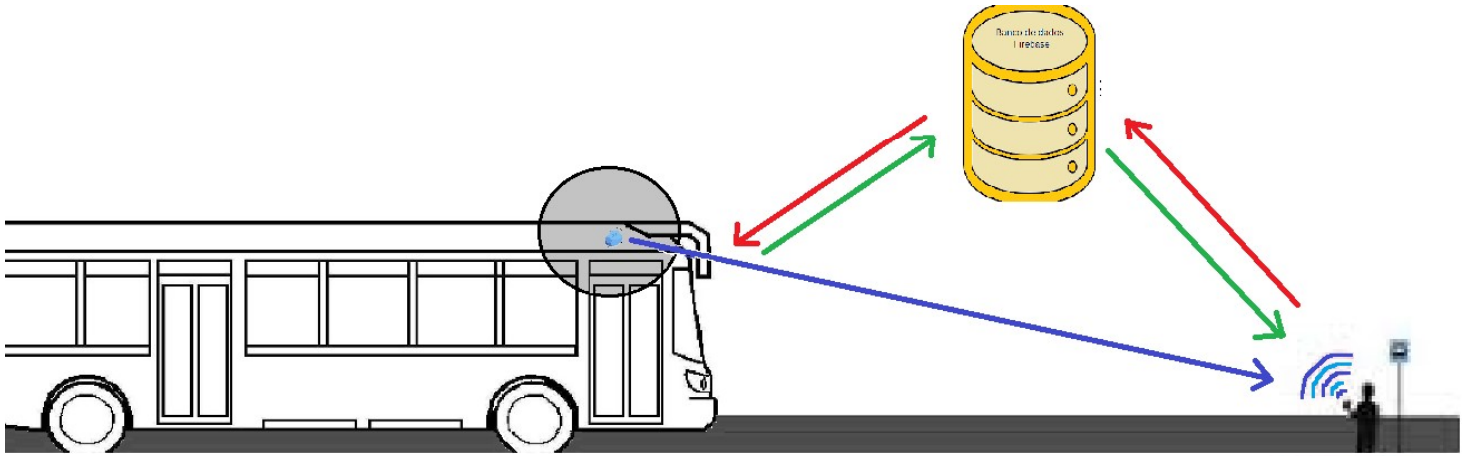
Ensinar a desenvolver uma tecnologia assistiva que facilite o uso de transporte coletivo para deficientes visuais através do uso do dispositivo *beacon* associado a um aplicativo acessível a deficientes visuais.

Objetivo Específicos

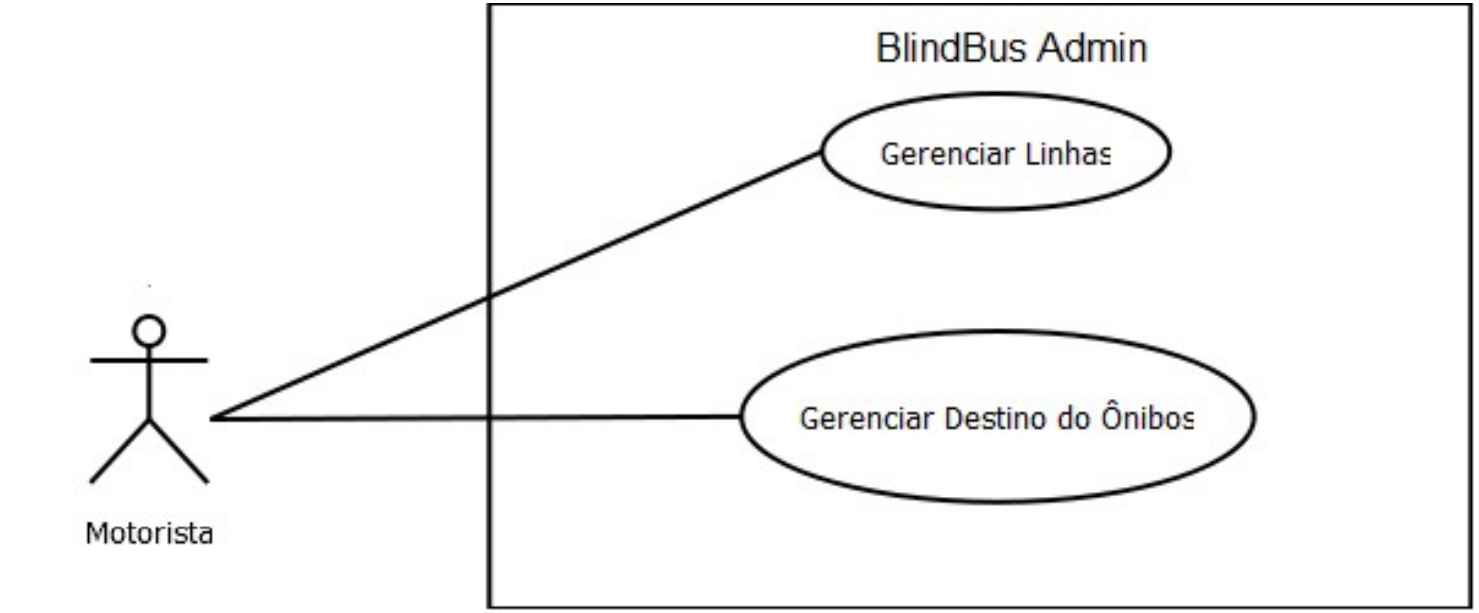
- Ensinar o desenvolvimento de um aplicativo para utilização de *beacon*.
- Ensinar o desenvolvimento de um aplicativo acessível para deficientes visuais.
- Fazer a integração das informações geradas.

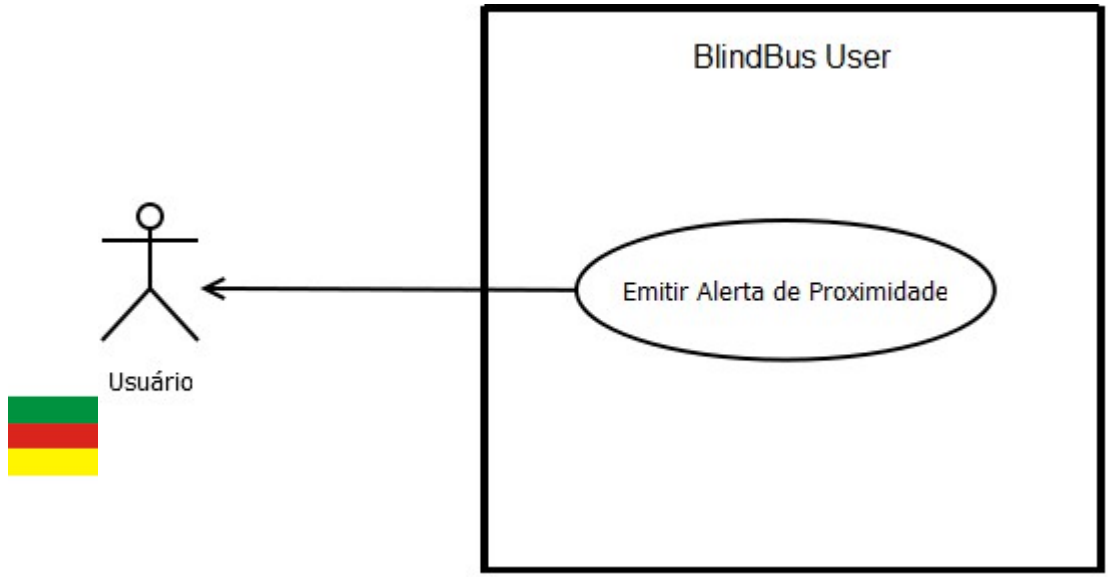
Modelagem

O presente trabalho propõe o BlindBus, uma tecnologia assistiva para auxiliar a mobilidade dos deficientes visuais, auxiliando-os no uso do transporte público. Ele é composto por dois aplicativos e um *beacon* que transmite informações.



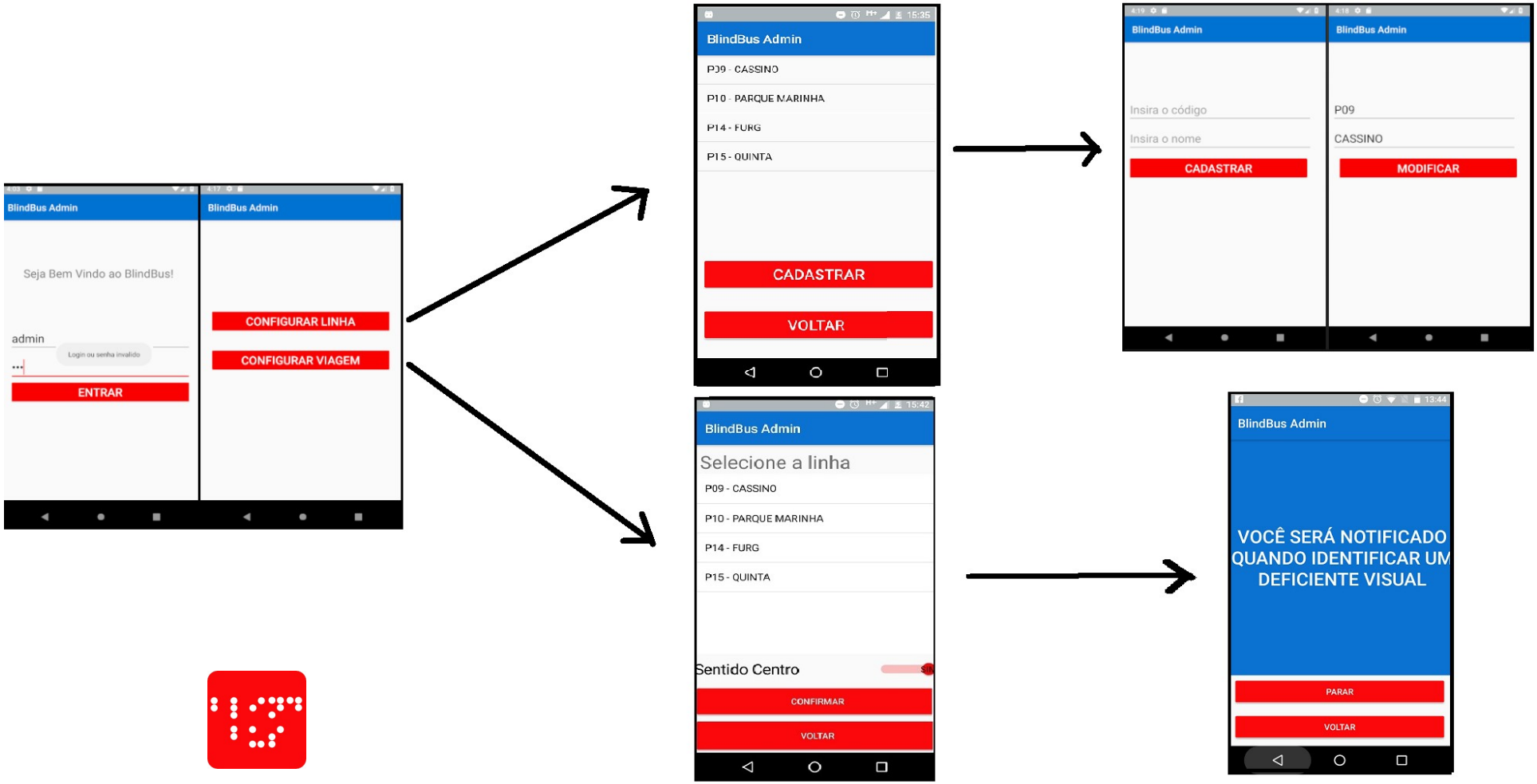
Caso de Uso


BlindBus Admin
Gerenciar Linhas
Gerenciar Destino do Ônibos
Motorista


BlindBus User
Emitir Alerta de Proximidade
Usuário

Ferramentas e Tecnologias Utilizadas

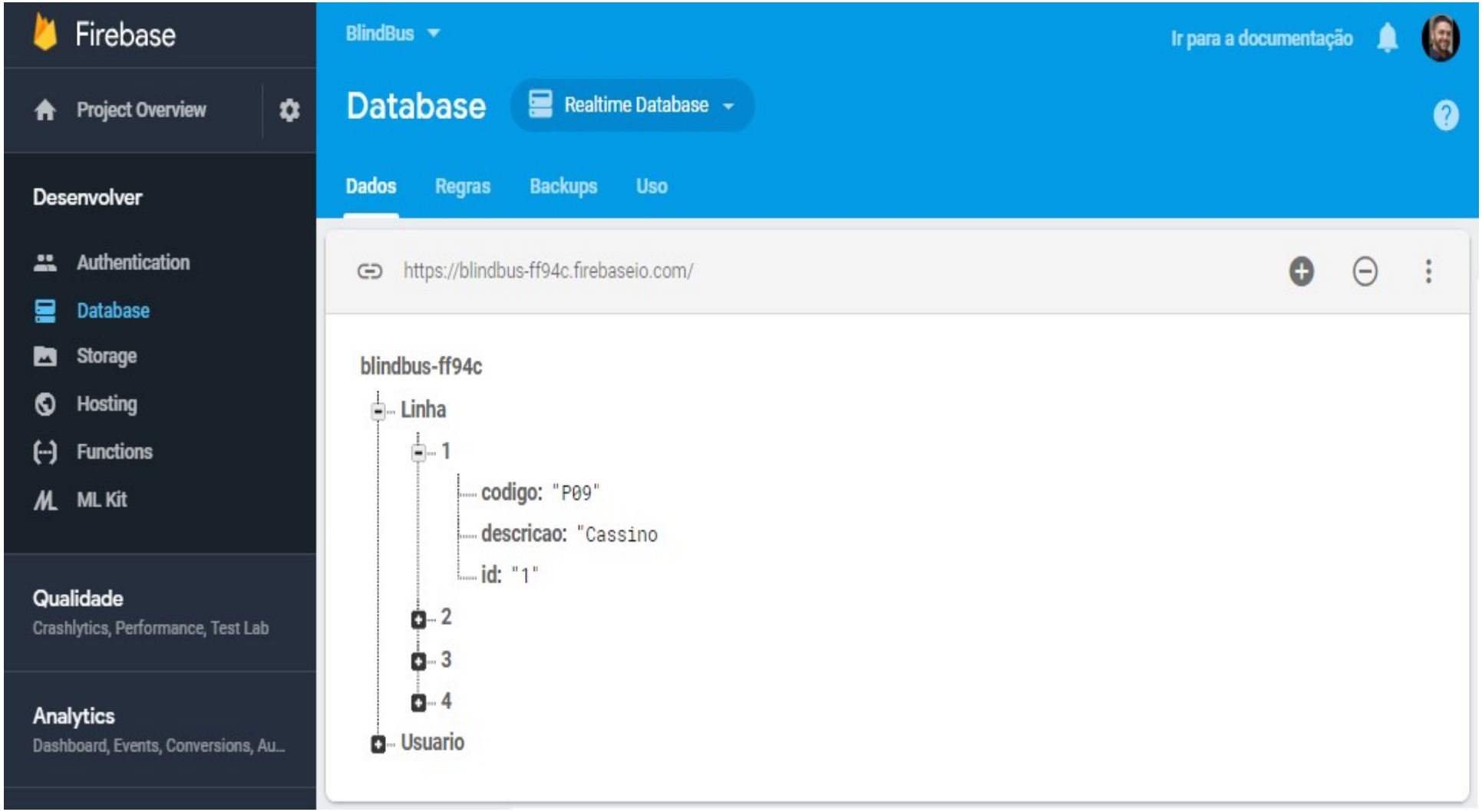
BlindBus Admin



BlindBus User

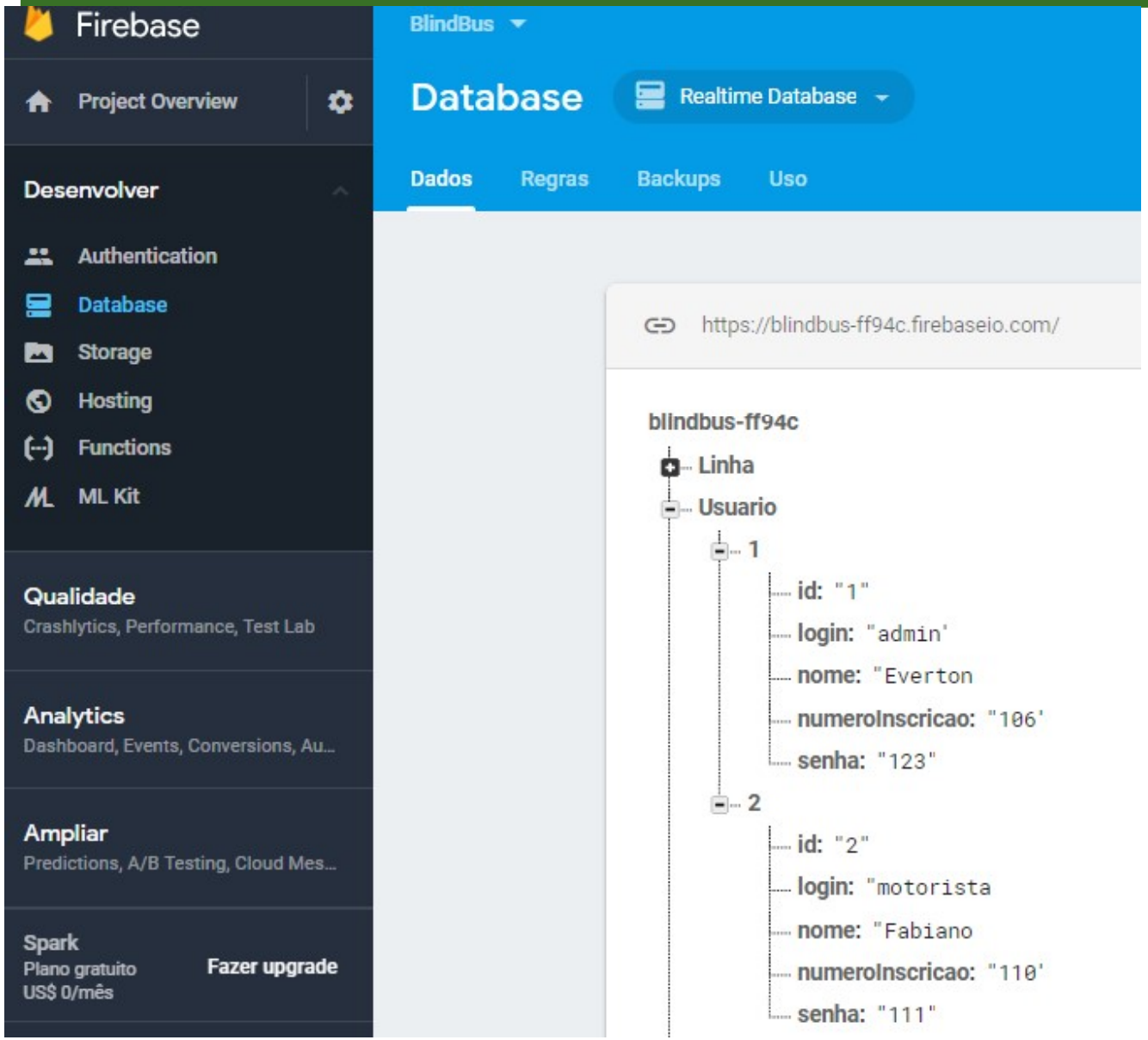
Firebase SGBD

Sistema de Gerenciamento de Banco de Dados



Firebase SGBD

Sistema de Gerenciamento de Banco de Dados



Comunicação Serviço Externo

1 Connect your app to Firebase

Connected

2 Add the Realtime Database to your app

Add the Realtime Database to your app

3 Configure Firebase Database Rules

The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to. By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up Authentication, you can configure your rules for public access. This does make your database open to anyone, even people not using your app, so be sure to restrict your database again when you set up authentication.

4 Write to your database

Retrieve an instance of your database using getInstance() and reference the location you want to write to.

```
// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference("message");

myRef.setValue("Hello, World!");
```

You can save a range of data types to the database this way, including Java objects. When you save an object the responses from any getters will be saved as children of this location.

Ambas aplicações

Comunicação Serviço Externo

5 Read from your database

To make your app data update in realtime, you should add a ValueEventListener to the reference you just created.

The onDataChange() method in this class is triggered once when the listener is attached and again every time the data changes, including the children.

```
// Read from the database
myRef.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        // This method is called once with the initial value and again
        // whenever data at this location is updated.
        String value = dataSnapshot.getValue(String.class);
        Log.d(TAG, "Value is: " + value);
    }

    @Override
    public void onCancelled(DatabaseError error) {
        // Failed to read value
        Log.w(TAG, "Failed to read value.", error.toException());
    }
});
```

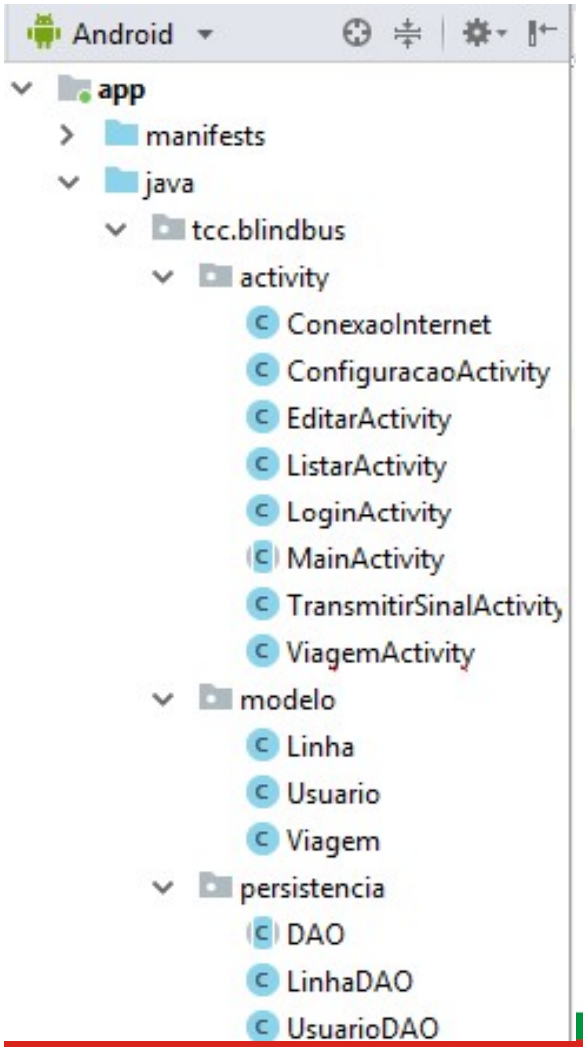
6 Optional: Configure ProGuard

7 Prepare for Launch

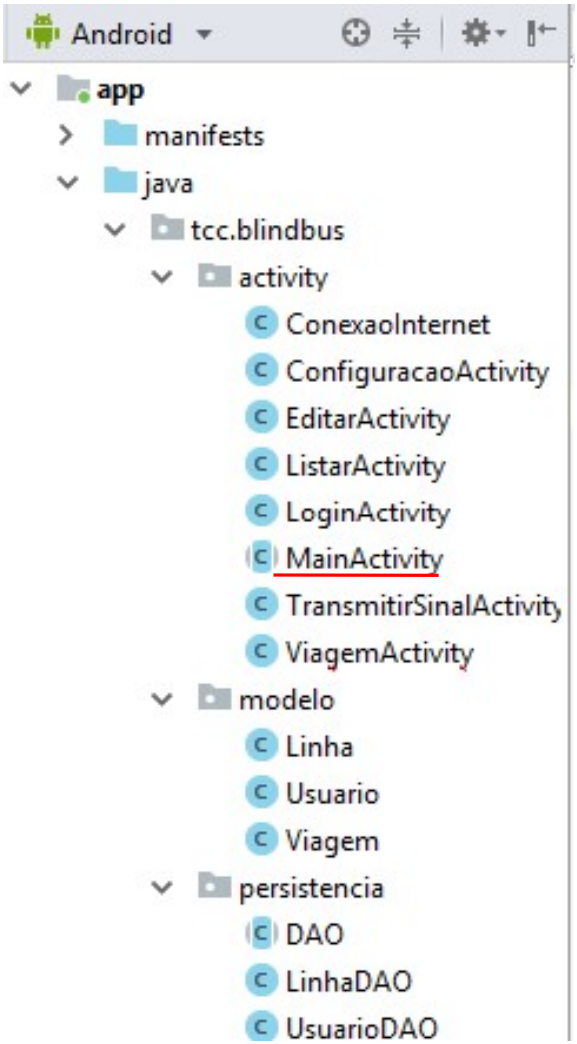
Before launching your app, we recommend walking through our launch checklist to make sure your app is ready to go!

Ambas aplicações

Implementação: BlindBus Admin

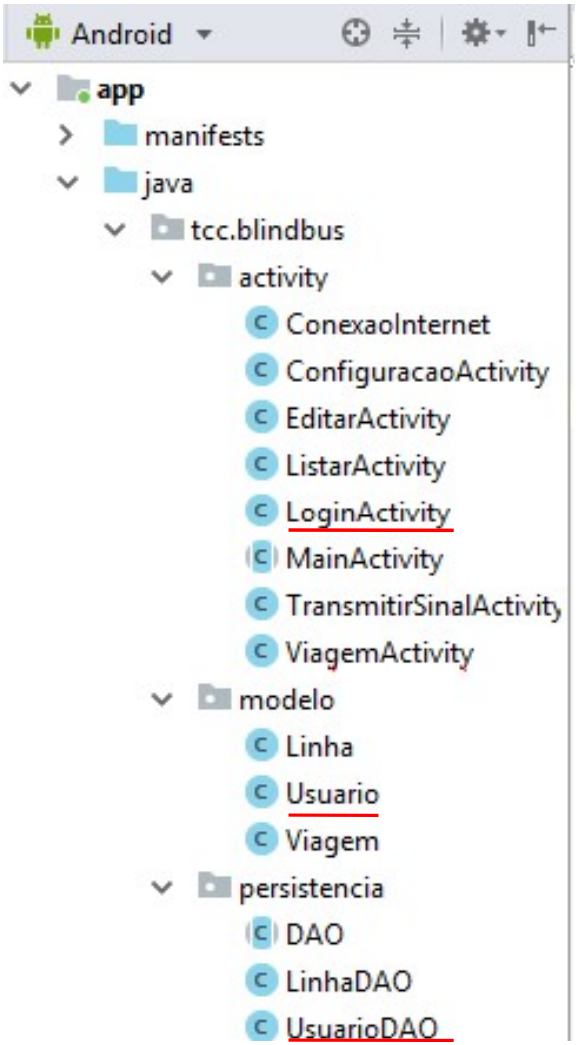


Implementação: BlindBus Admin




```
public abstract class MainActivity extends AppCompatActivity {
    protected UsuarioDAO daoUsuario;
    protected LinhaDAO daoLinha;
    protected Usuario usuario;
    protected Linha linha;
    protected static String idUsuarioGlobal;
    protected FirebaseDatabase firebaseDatabase;
    protected DatabaseReference databaseReference;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }
    protected void inicializarFirebase(Context context) {
        FirebaseApp.initializeApp(context);
        firebaseDatabase = FirebaseDatabase.getInstance();
        databaseReference = firebaseDatabase.getReference();
    }
    protected boolean verificaConexao() {
        boolean conectado;
        ConnectivityManager conectivtyManager = (ConnectivityManager)
                getSystemService(Context.CONNECTIVITY_SERVICE);
        if (conectivtyManager.getActiveNetworkInfo() != null
                && conectivtyManager.getActiveNetworkInfo().isAvailable()
                && conectivtyManager.getActiveNetworkInfo().isConnected()) {
            conectado = true;
        } else {
            conectado = false;
        }
        return conectado;
    }

}
```

Implementação: BlindBus Admin



Classe LoginActivity

```
package tcc.blindbus.activity;
databaseReference.child("Usuario").addValueEventListener(new ValueEventListener(){
    @Override
    public void onDataChange(DataSnapshot dataSnapshot){
        for (DataSnapshot objSnapshot:dataSnapshot.getChildren()){
            usuario = objSnapshot.getValue(Usuario.class);
            daoUsuario.inserirUsuario(new Usuario(usuario.getNome(),usuario.getLogin(),
                    usuario.getSenha(),usuario.getNumeroInscricao()));
            usuario = new Usuario();
        }
    }
});
```

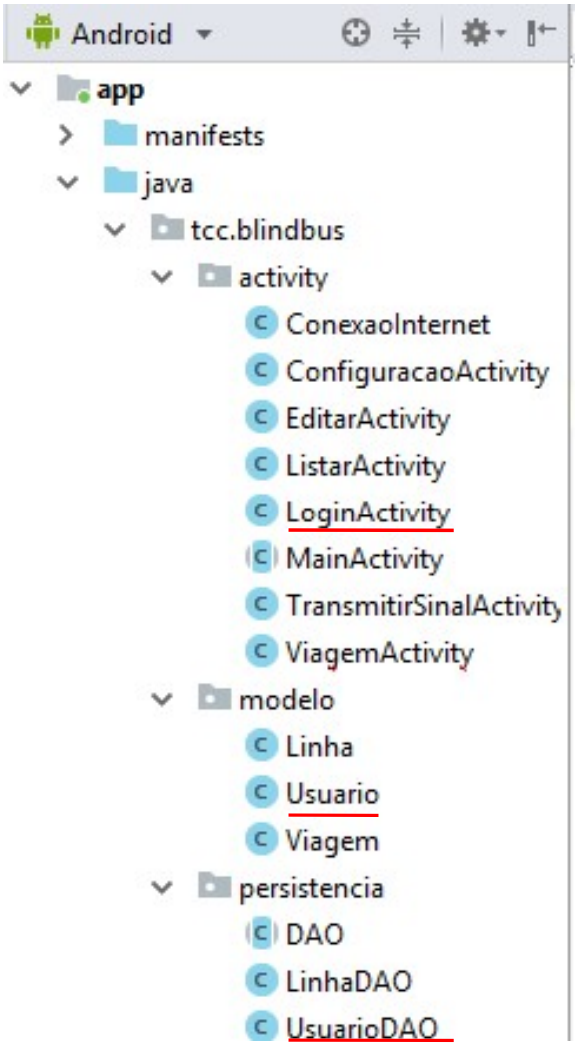
Classe UsuarioDAO

```
package tcc.blindbus.persistencia;
public void inserirUsuario(Usuario usuario)
{
    ContentValues values = new ContentValues();
    values.put("nome",usuario.getNome());
    values.put("login",usuario.getLogin());
    values.put("senha",usuario.getSenha());
    values.put("numeroInscricao",usuario.getNumeroInscricao());
    inserir(values);
}
```

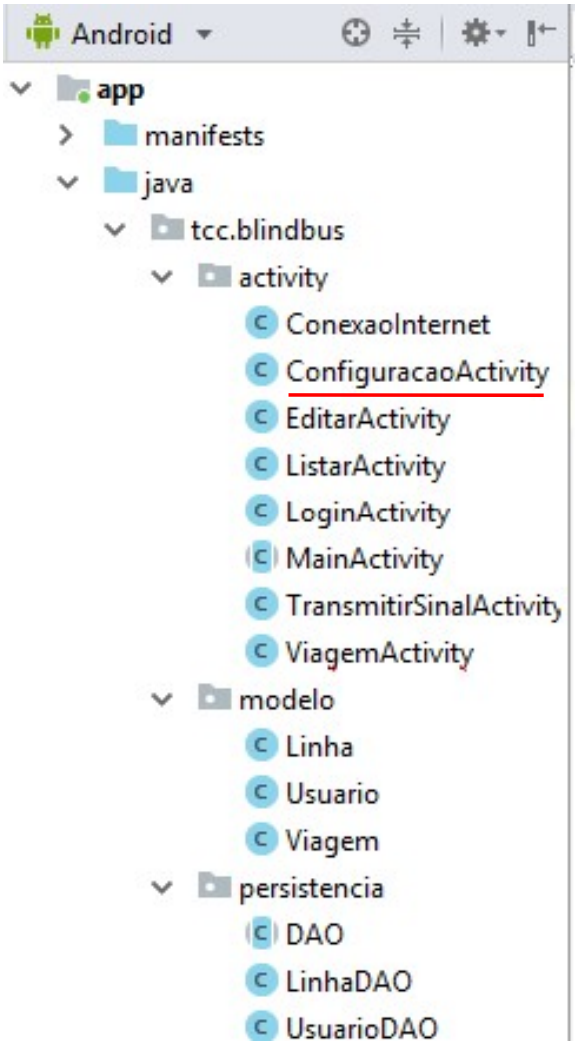
Classe DAO

```
protected void inserir(ContentValues values) {
    getWritableDatabase().insert(getTableName(), nullColumnHack: null,values);
}
```

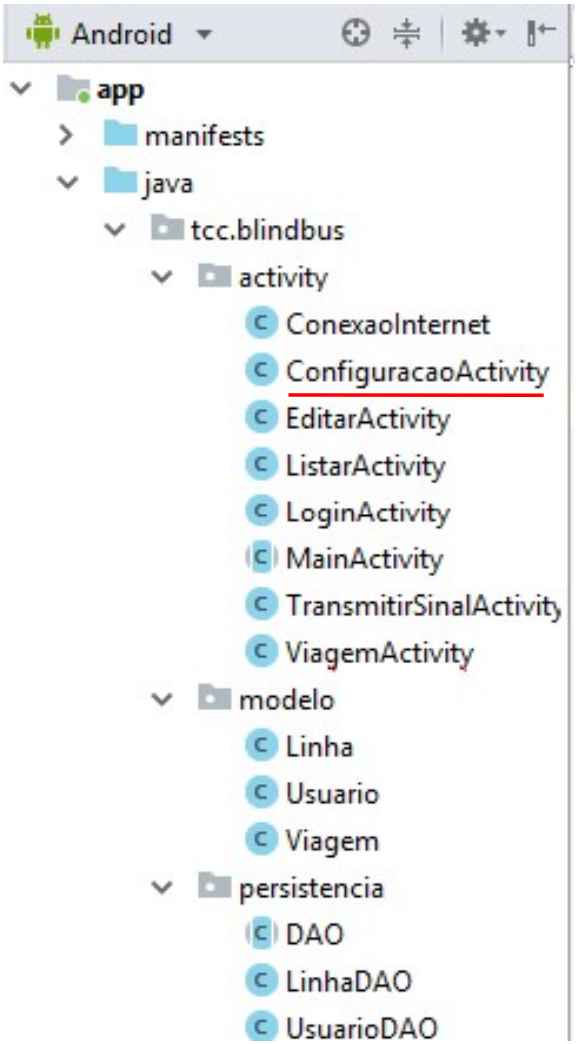
Implementação: BlindBus Admin



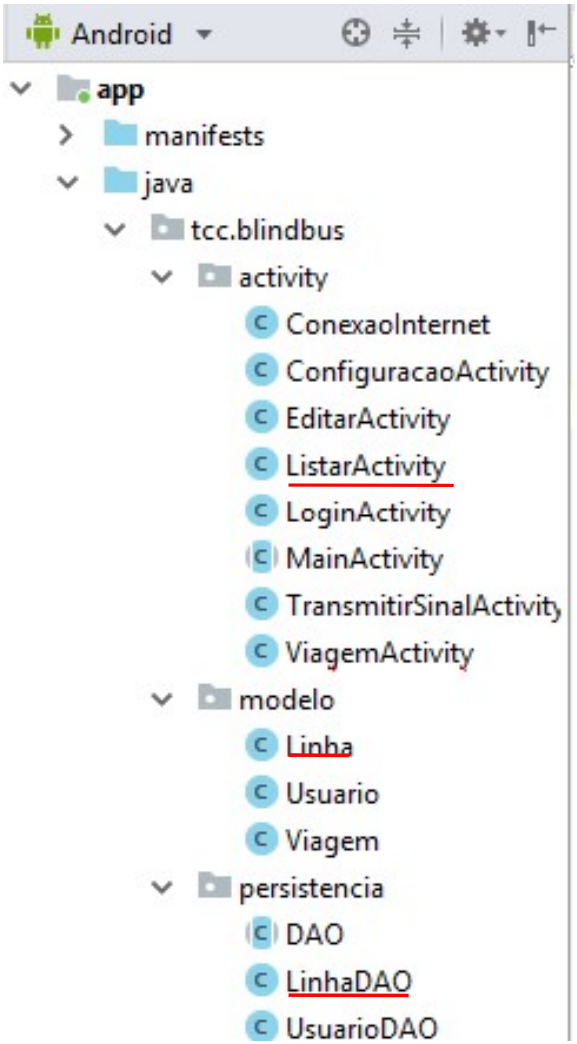
Implementação: BlindBus Admin



Implementação: BlindBus Admin

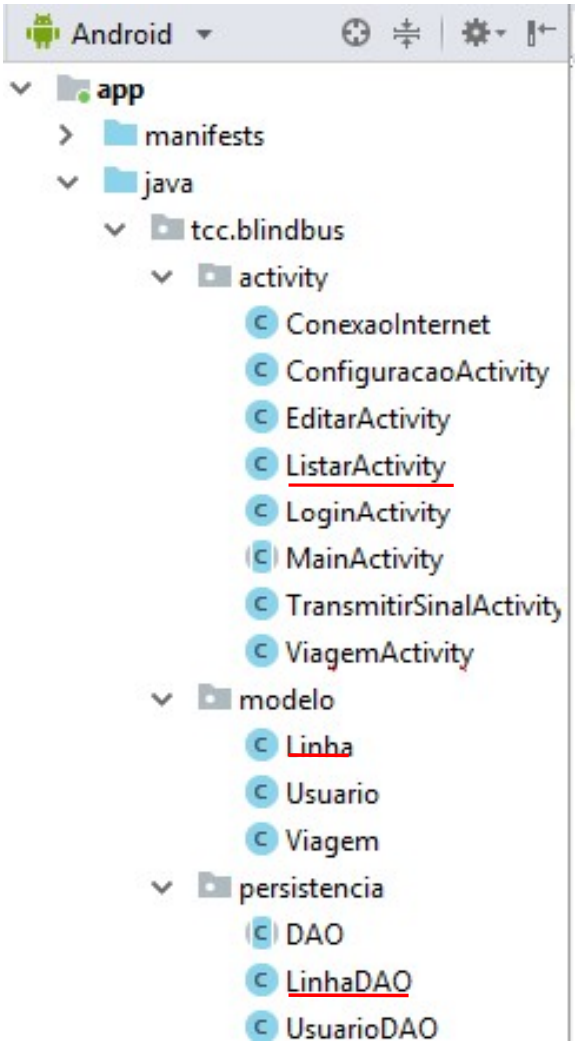


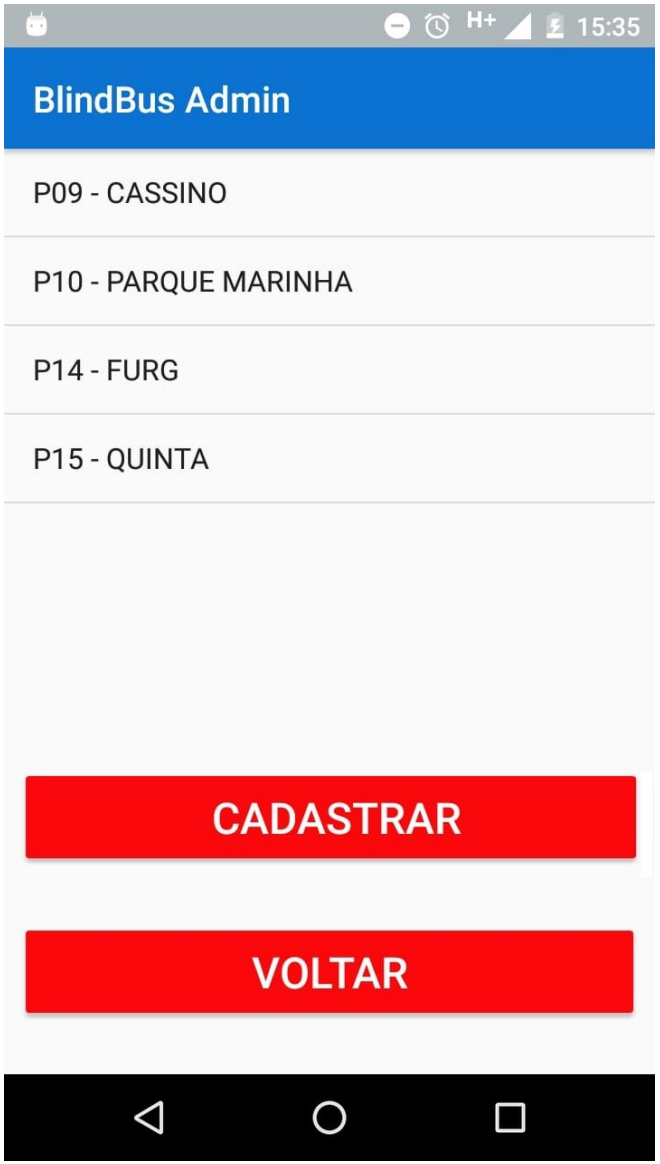
Implementação: BlindBus Admin



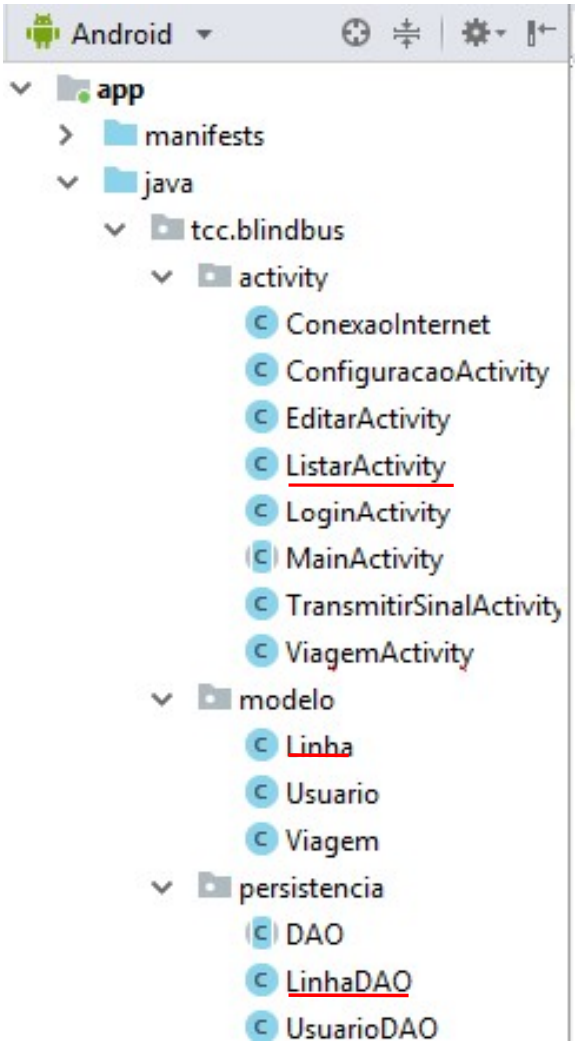

```
public class ListarActivity extends MainActivity {
    ListView lista;
    ArrayList<Linha> listview_Linha;
    Linha linha;
    ArrayAdapter adapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {...}

    @Override
    protected void onResume() {
        super.onResume();
        carregarLinha();
    }
    public void carregarLinha(){
        daoLinha = new LinhaDAO( context: ListarActivity.this);
        listview_Linha = daoLinha.getLista();
        daoLinha.close();
        if (listview_Linha != null){
            adapter = new ArrayAdapter<Linha>( context: ListarActivity.this,
                    android.R.layout.simple_list_item_1, listview_Linha);
            lista.setAdapter(adapter);
        }
    }
}
```

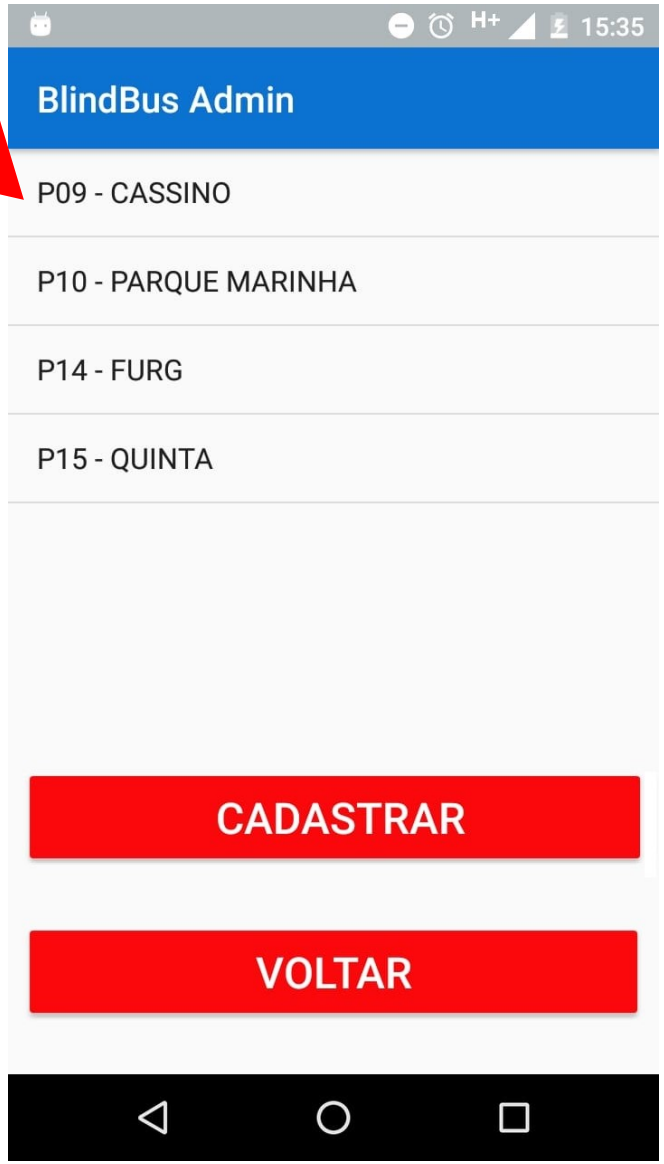
Implementação: BlindBus Admin



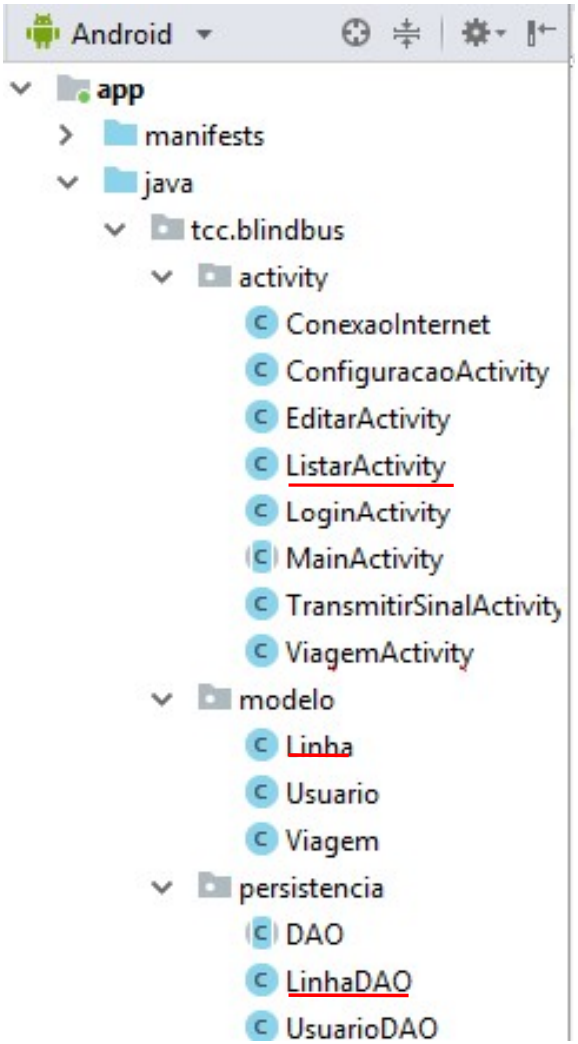


Implementação: BlindBus Admin

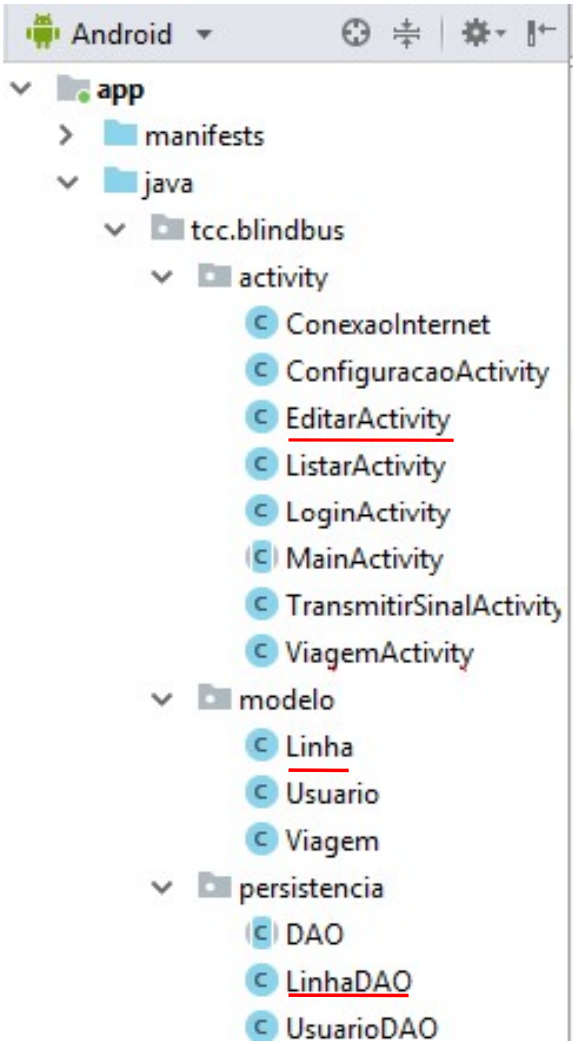




Implementação: BlindBus Admin

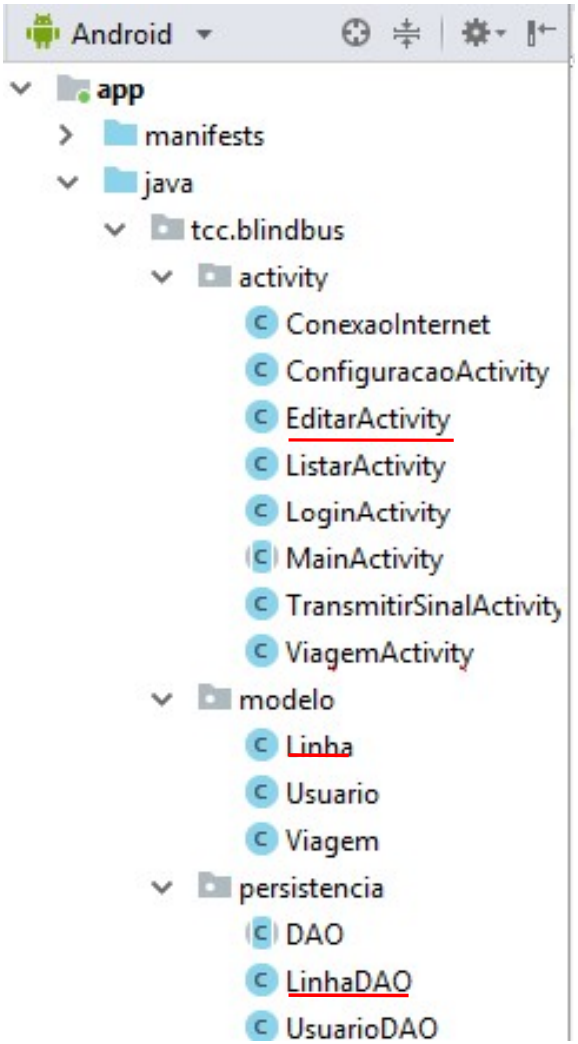


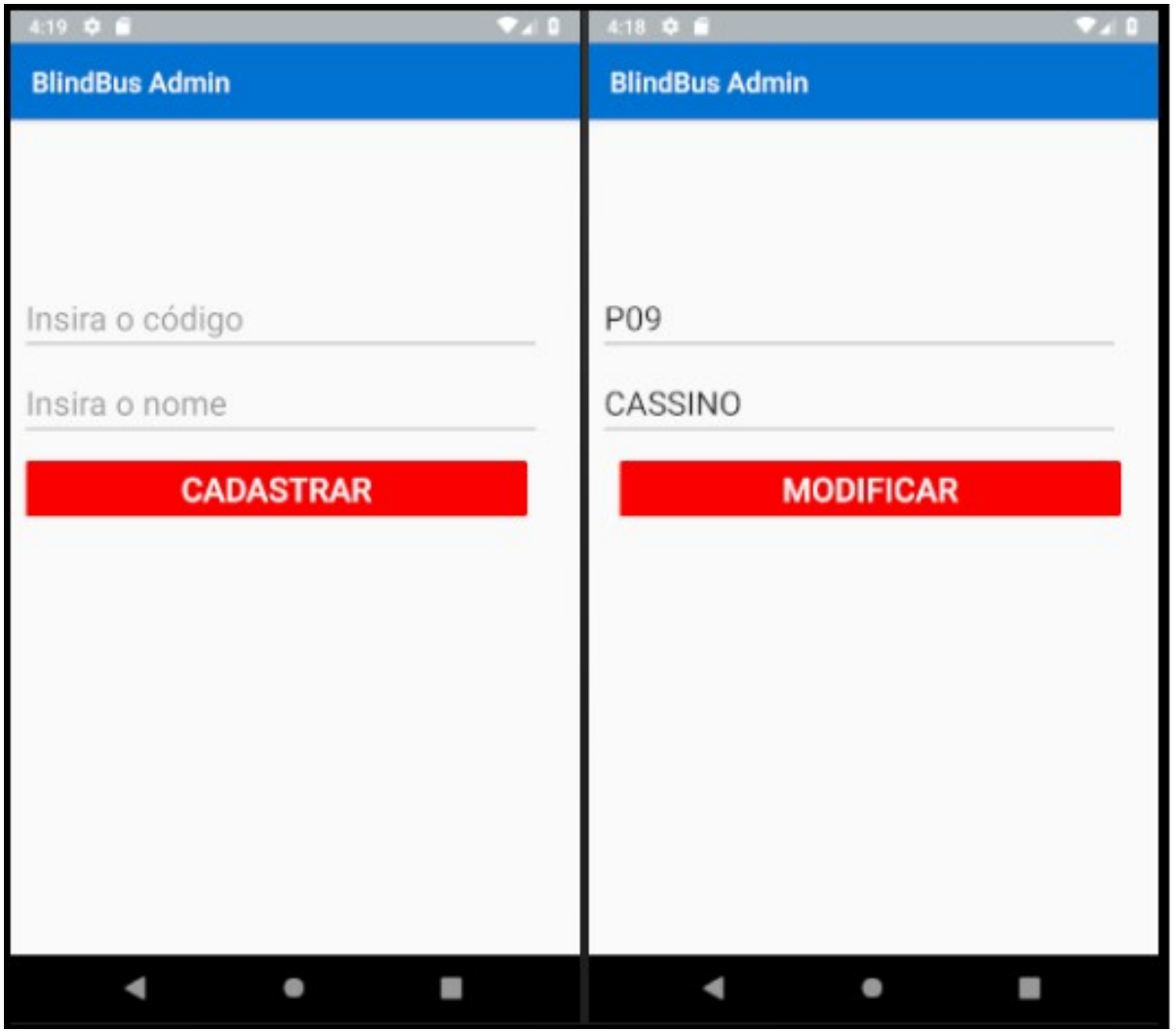
Implementação BlindBus Admin



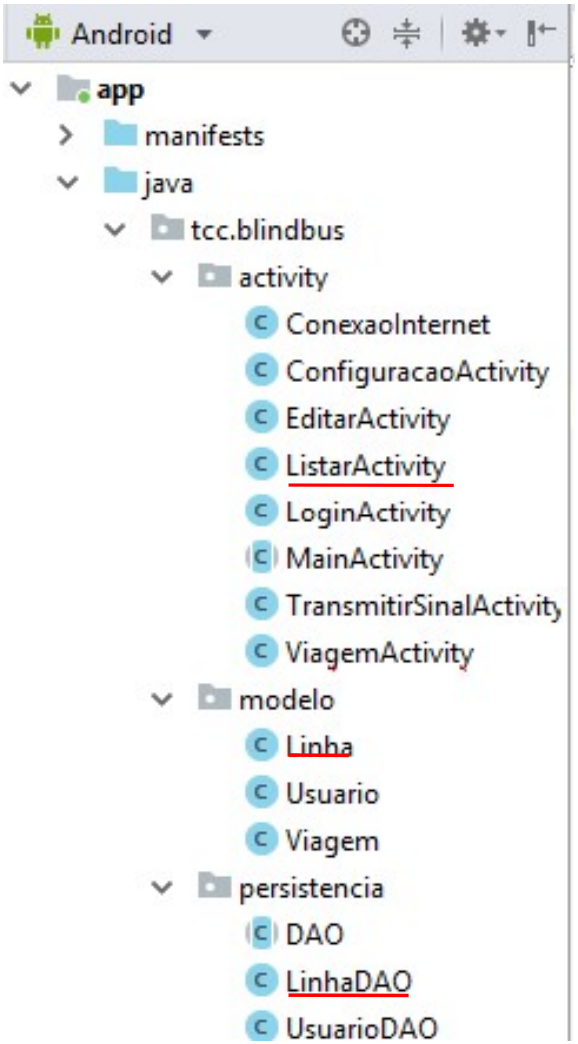

```
public class EditarActivity extends MainActivity {
    EditText editText_Descricao, editText_Codigo;
    Button btn_Poliform;
    Linha alterarLinha, linha;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_editar);
        linha = new Linha();
        daoLinha = new LinhaDAO( context: EditarActivity.this);
        Intent intent = getIntent();
        alterarLinha = (Linha) intent.getSerializableExtra( name: "linha-escolhida");
        editText_Descricao = (EditText) findViewById(R.id.editText_Descricao);
        editText_Codigo = (EditText) findViewById(R.id.editText_Codigo);
        btn_Poliform = (Button) findViewById(R.id.btn_Poliform);
        if (alterarLinha !=null){
            btn_Poliform.setText("Modificar");
            editText_Descricao.setText(alterarLinha.getDescricao());
            editText_Codigo.setText(alterarLinha.getCodigo());
            linha.setId(alterarLinha.getId());
        }else{
            btn_Poliform.setText("Cadastrar");
        }
        btn_Poliform.setOnClickListener(new View.OnClickListener() {
            Intent intent2 = new Intent( packageContext: EditarActivity.this, ListarActivity.class);
            @Override
            public void onClick(View v) {
                if(btn_Poliform.getText().toString().equals("Cadastrar")){
                    daoLinha.inserirLinha(linha);
                }else{
                    daoLinha.alterarLinha(linha);
                }
                daoLinha.close();
                intent2.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK | Intent.FLAG_ACTIVITY_NEW_TASK);
                startActivity(intent2);
            }
        });

    }
```

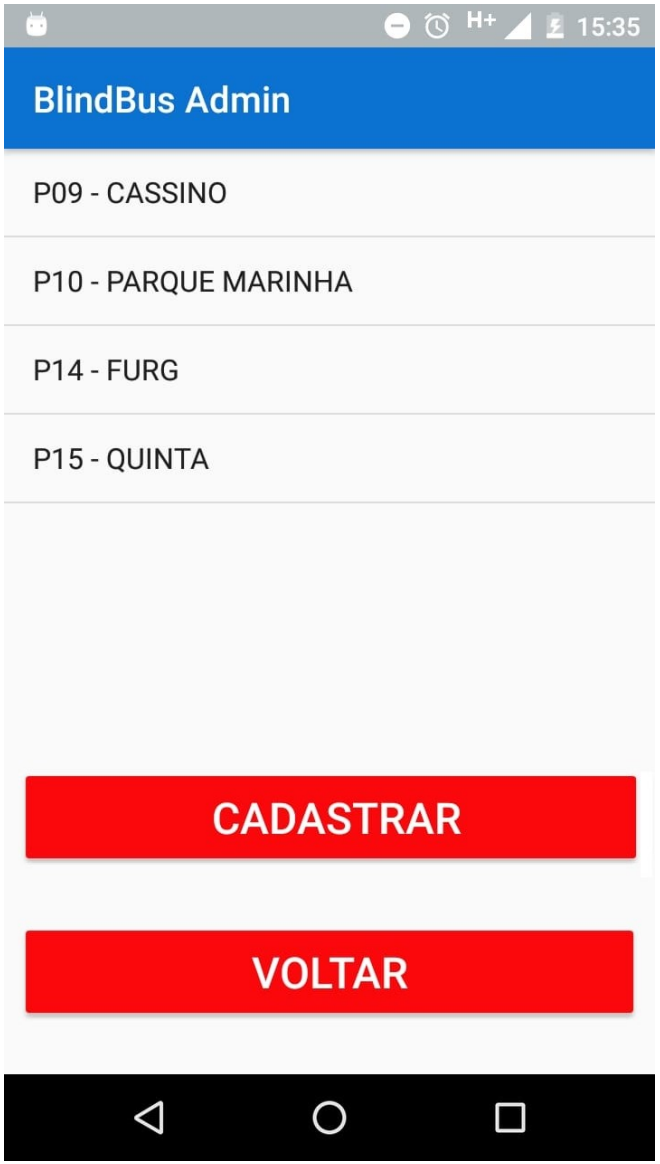
Implementação: BlindBus Admin



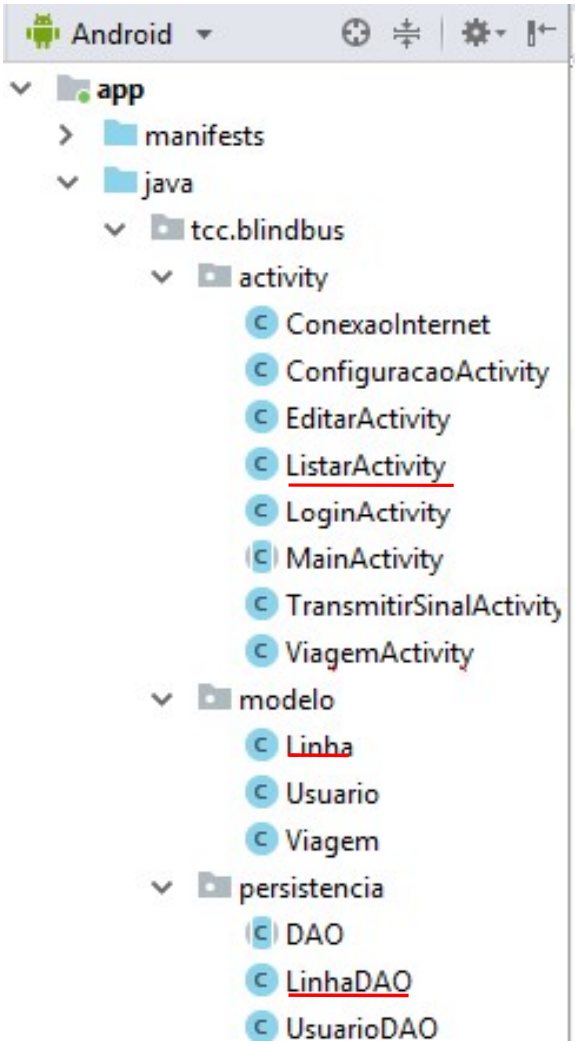


Implementação: BlindBus Admin

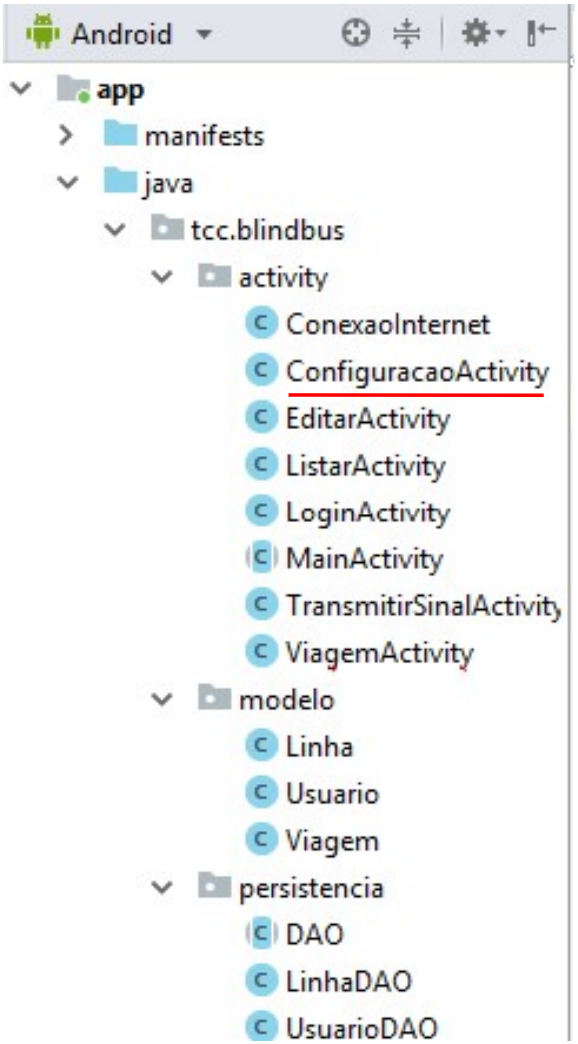




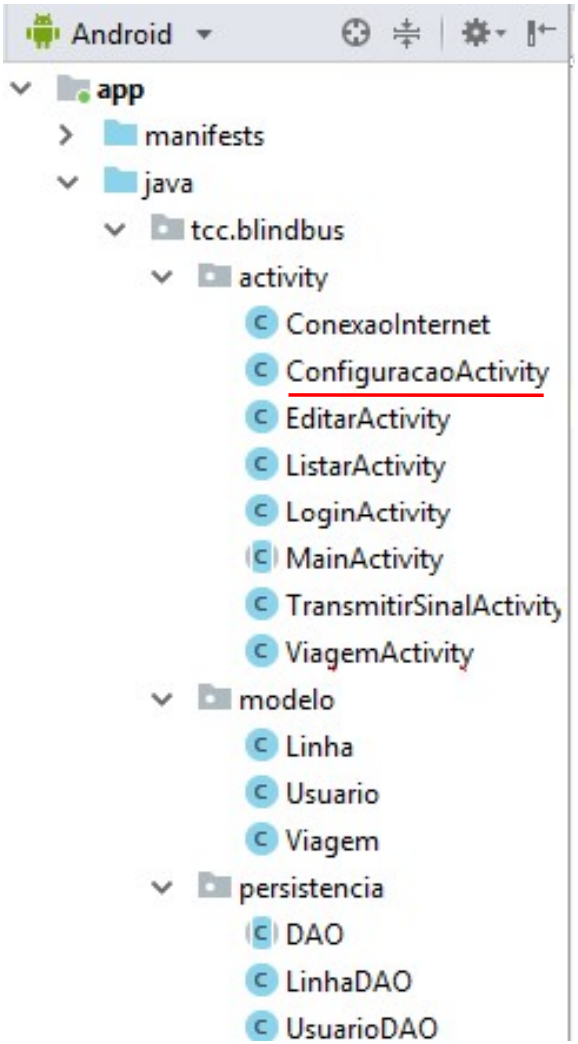
Implementação: BlindBus Admin



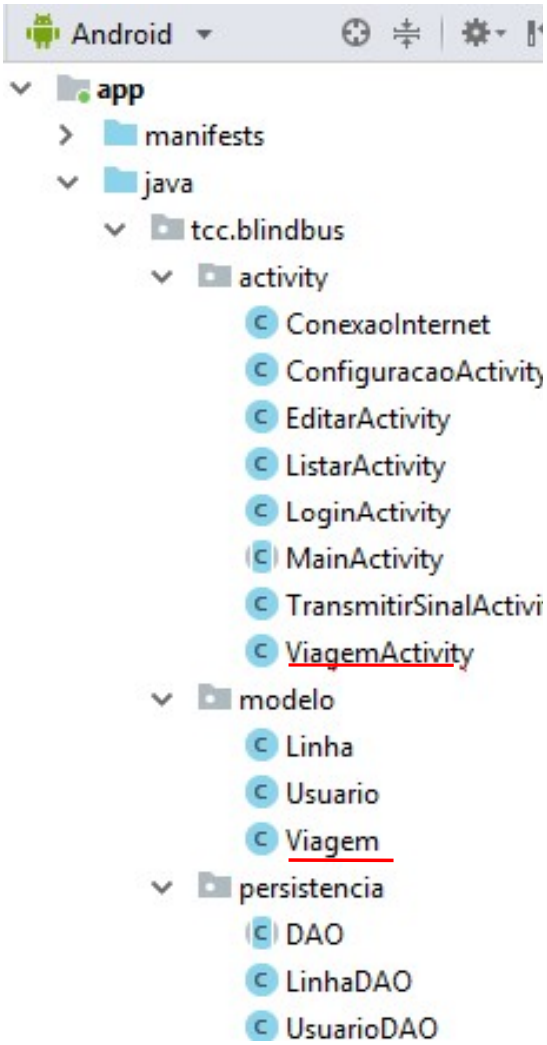
Implementação: BlindBus Admin



Implementação: BlindBus Admin

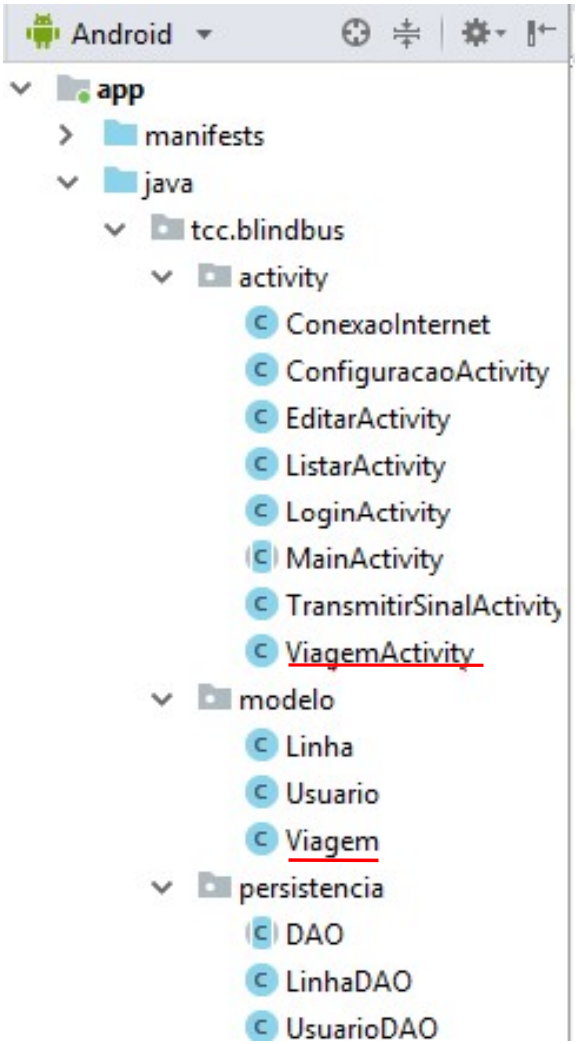


Implementação BlindBus Admin

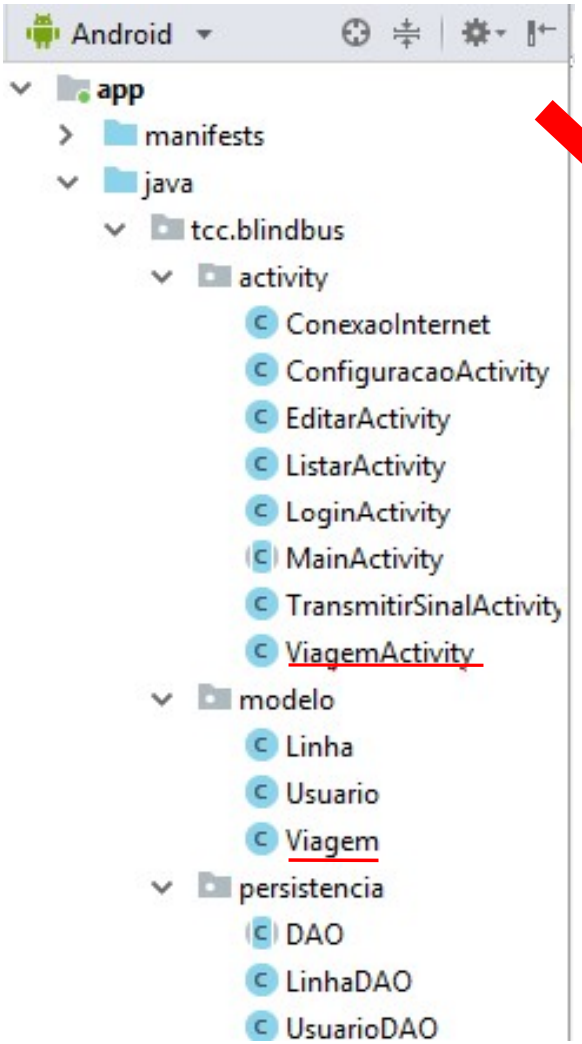


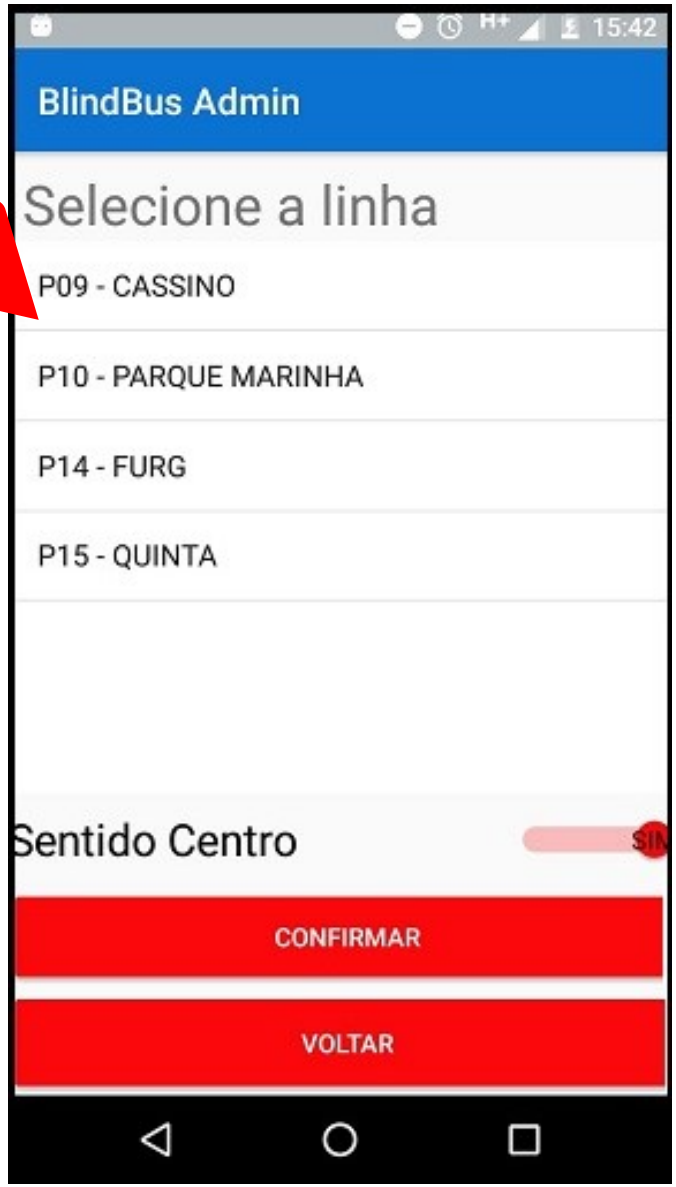

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_viagem);
    Button btCadastrar = (Button) findViewById(R.id.btCadastrar);
    Button btVoltar = (Button) findViewById(R.id.btVoltar);
    daoUsuario = new UsuarioDAO( context: this);
    inicializarFirebase( context: ViagemActivity.this);
    usuario =  daoUsuario.carregaUsuario(idUsuarioGlobal);
    btCadastrar.setOnClickListener((v) → {
            if(viagemSelecionada.getLinha()!=null) {
                Intent intent = new Intent( packageContext: ViagemActivity.this,
                        TransmitirSinalActivity.class);
                Switch switchSentidoCentro = (Switch) findViewById(R.id.switchSentido);
                if (switchSentidoCentro.isChecked()) {
                    viagemSelecionada.setSentido("CENTRO");
                } else {
                    viagemSelecionada.setSentido("BAIRRO");
                }
                viagemSelecionada.setInformacoesUsuario(usuario.getNumeroInscricao() +
                        " " + usuario.getNome());
                Viagem viagem = new Viagem();
                viagem.setInstanceBeacon("0x000005000453");
                viagem.setNameSpace("0x699ebc80e1f311e39a0f");
                viagem.setLinha(viagemSelecionada.getLinha());
                viagem.setSentido(viagemSelecionada.getSentido());
                viagem.setInformacoesUsuario(viagemSelecionada.getInformacoesUsuario());
                viagem.setClienteEncontrado("NAO");
                databaseReference.child("Viagem/0x000005000453").setValue(null);
                databaseReference.child("Viagem").child(viagem.getInstanceBeacon()).setValue(viagem);
                startActivity(intent);
            }else{
                alert(getString(R.string.viagem_nao_selecionada));
```

Implementação: BlindBus Admin

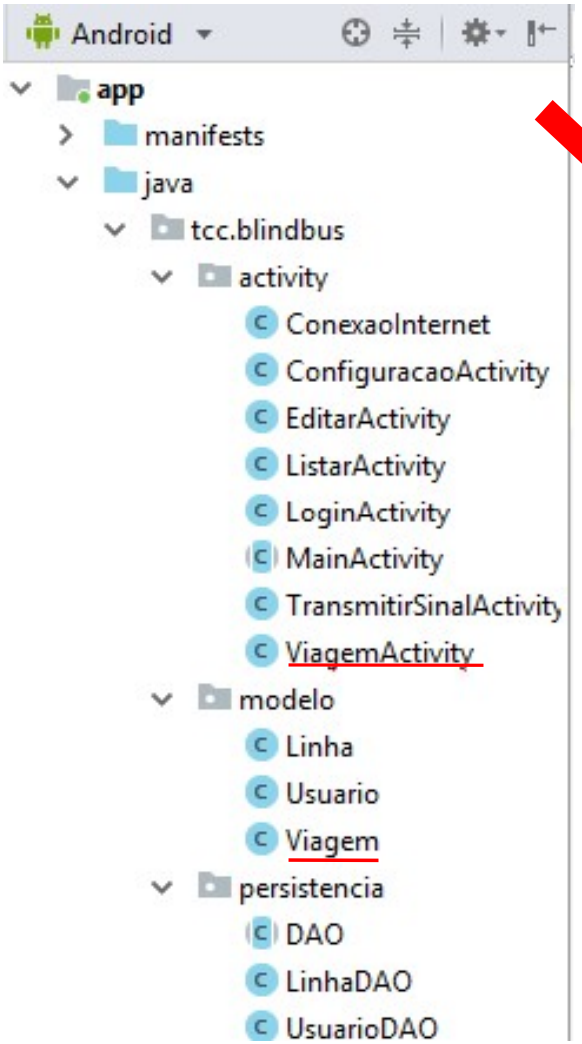


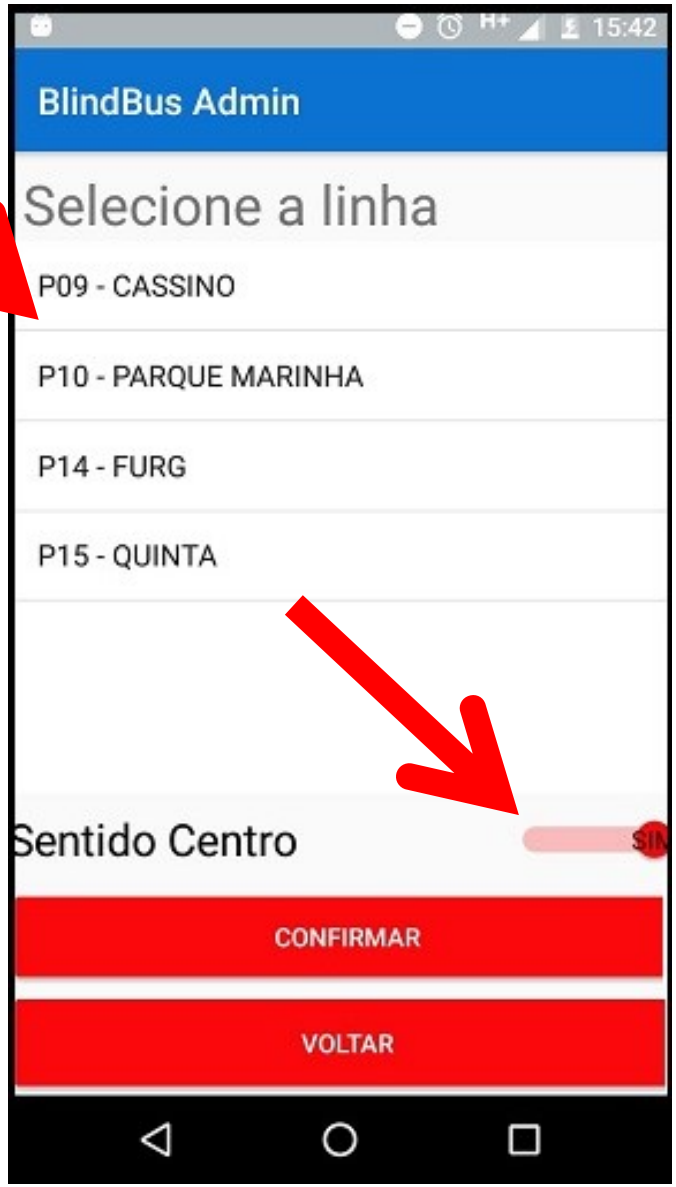
Implementação: BlindBus Admin



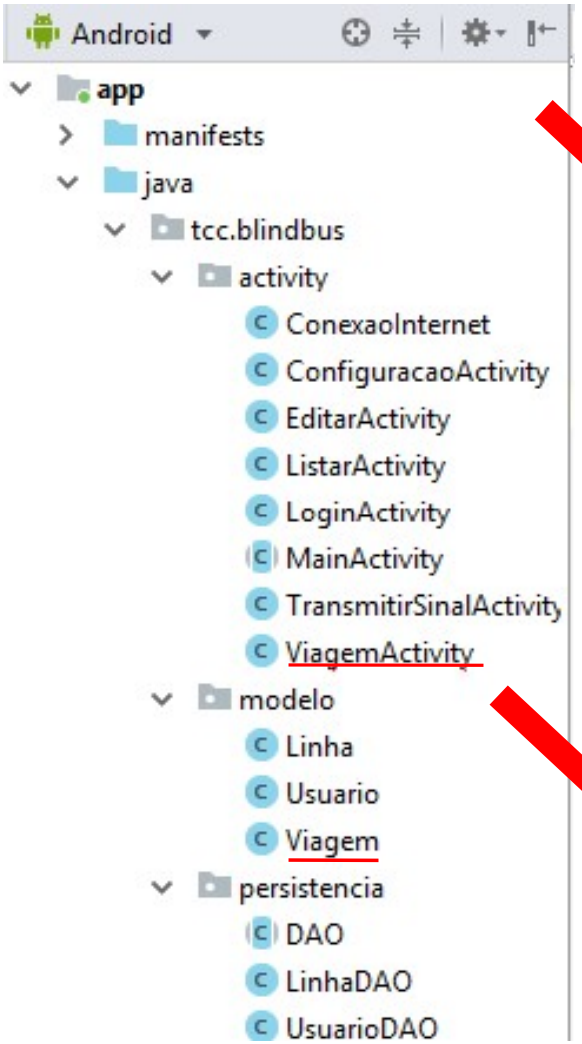


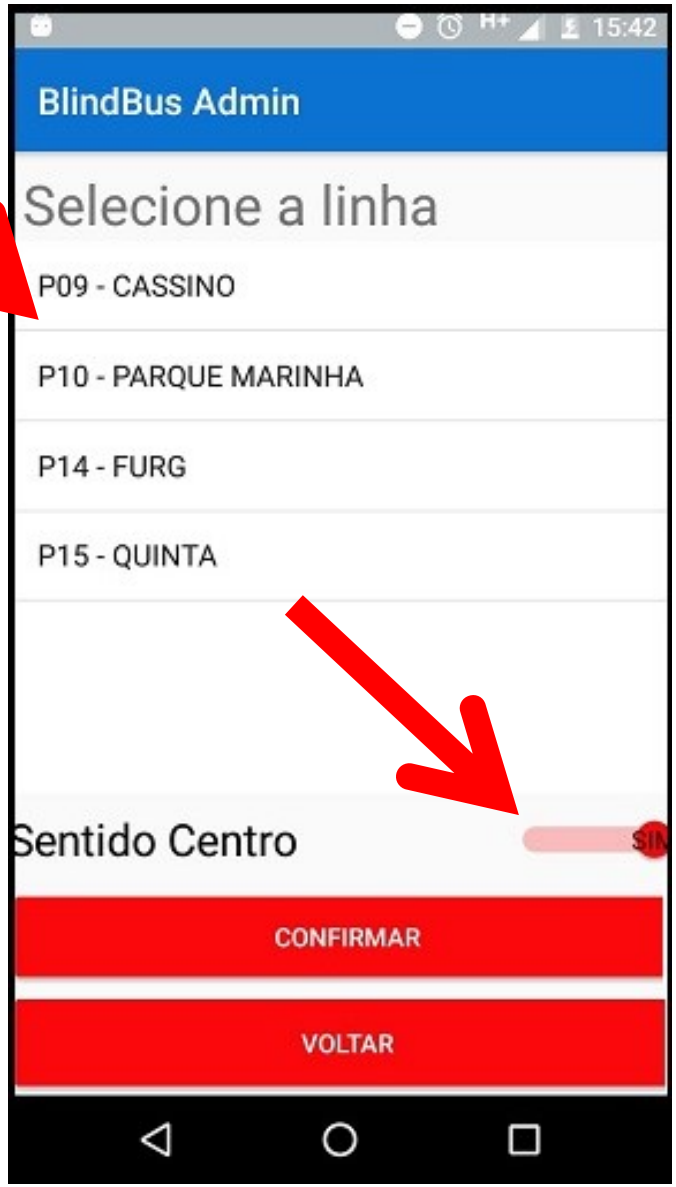
Implementação: BlindBus Admin



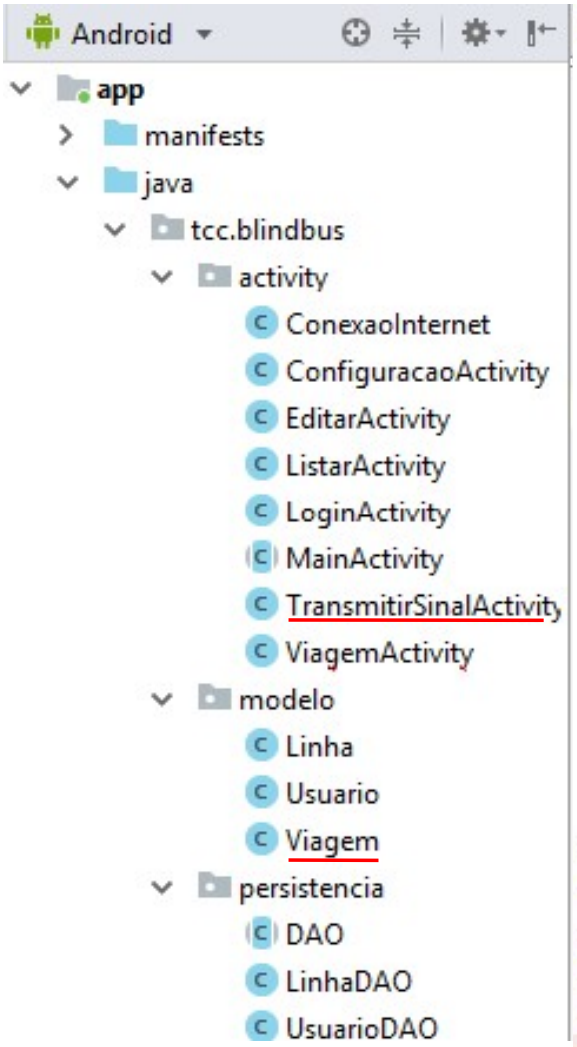


Implementação: BlindBus Admin



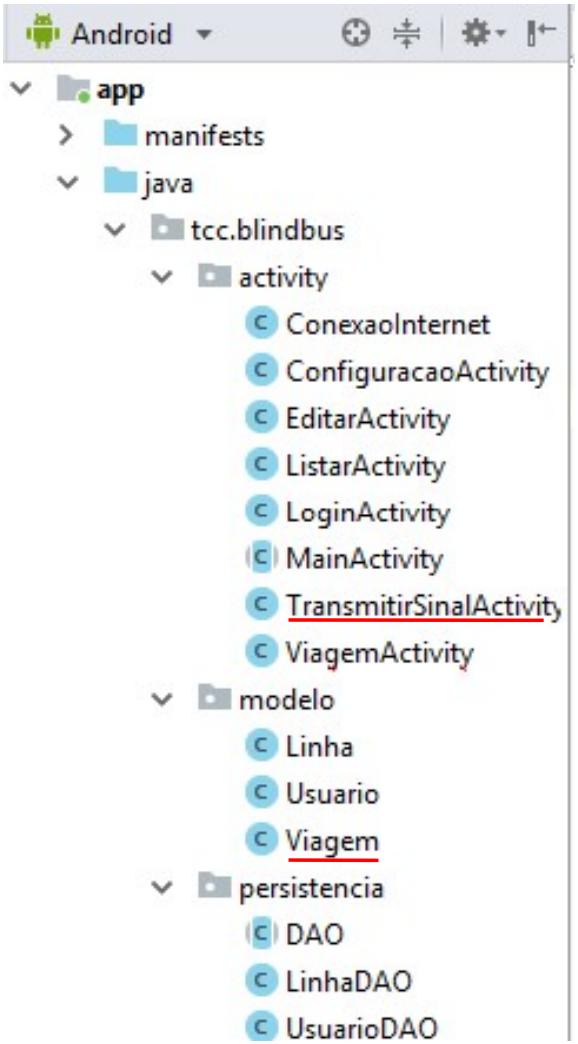


Implementação: BlindBus Admin

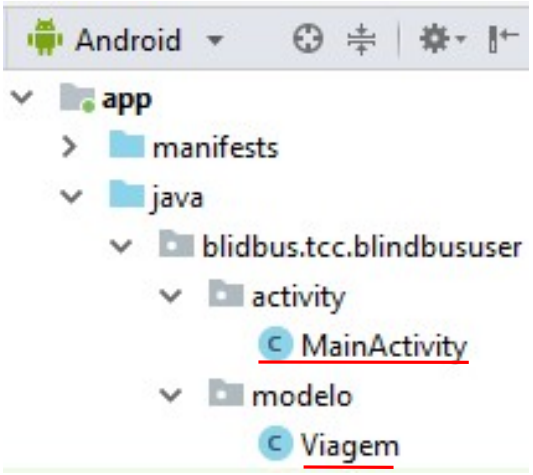



```
public void lerViagem() {
    lerConexao();
    databaseReference.child("Viagem/0x000005000453").addValueEventListener(new
        ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            Viagem viagemRecebida = dataSnapshot.getValue(Viagem.class);
            if(viagemRecebida!=null && !travaFuncao) {
                if(viagemRecebida.getClienteEncontrado().equals("SIM")) {
                    alert("Deficiente identificado");
                    atualizaViagem(viagemRecebida);
                    playSound();
                    piscar();
                }
            }
        }
        @Override
        public void onCancelled(DatabaseError databaseError) {
        }
    });
}
public void atualizaViagem(Viagem viagem) {...}
private void alert(String message) {...}
private void playSound() {...}
private void piscar(){...}
private void lerConexao() {...}
```

Implementação: BlindBus Admin



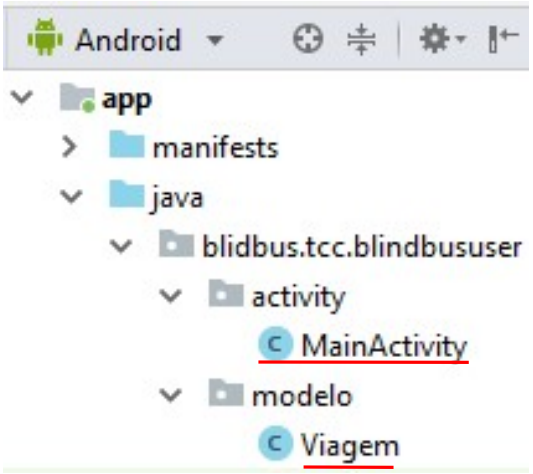
Implementação: BlindBus User



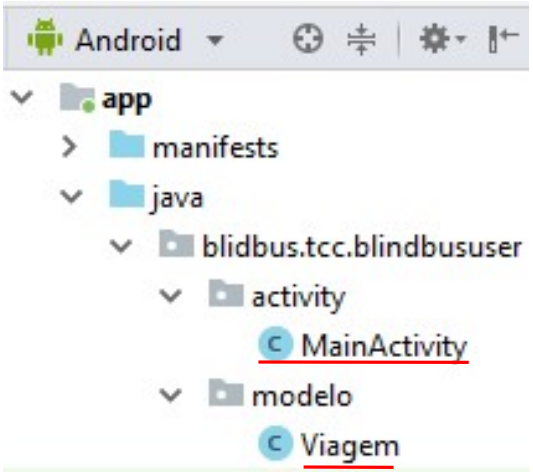

```
public class MainActivity extends AppCompatActivity
        implements View.OnClickListener, BeaconConsumer,RangeNotifier {
    protected final String TAG = MainActivity.this.getClass().getSimpleName();
    private static final int PERMISSION_REQUEST_COARSE_LOCATION = 1;
    private static final int REQUEST_ENABLE_BLUETOOTH = 1;
    private static final long DEFAULT_SCAN_PERIOD_MS = 1000;
    private static final String ALL_BEACONS_REGION = "AllBeaconsRegion";
    String instanceBeacon = "";
    BeaconManager mBeaconManager;
    String verificaRepeticaoMensagem = "";
    private Region mRegion;
    FirebaseDatabase firebaseDatabase;
    DatabaseReference databaseReference;
    Viagem viagem = new Viagem();
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    getStartButton().setOnClickListener(this);
    getStopButton().setOnClickListener(this);
    mBeaconManager = BeaconManager.getInstanceForApplication(this);
    //Corrigir um protocolo de beacon, Eddystone neste caso
    mBeaconManager.getBeaconParsers().add(new BeaconParser().setBeaconLayout(BeaconParser.EDDYSTONE_UID_LAYOUT));
    ArrayList<Identifier> identifiers = new ArrayList<>();
    mRegion = new Region(ALL_BEACONS_REGION, identifiers);
    inicializarFirebase( context: MainActivity.this);
}
```

Implementação: BlindBus User



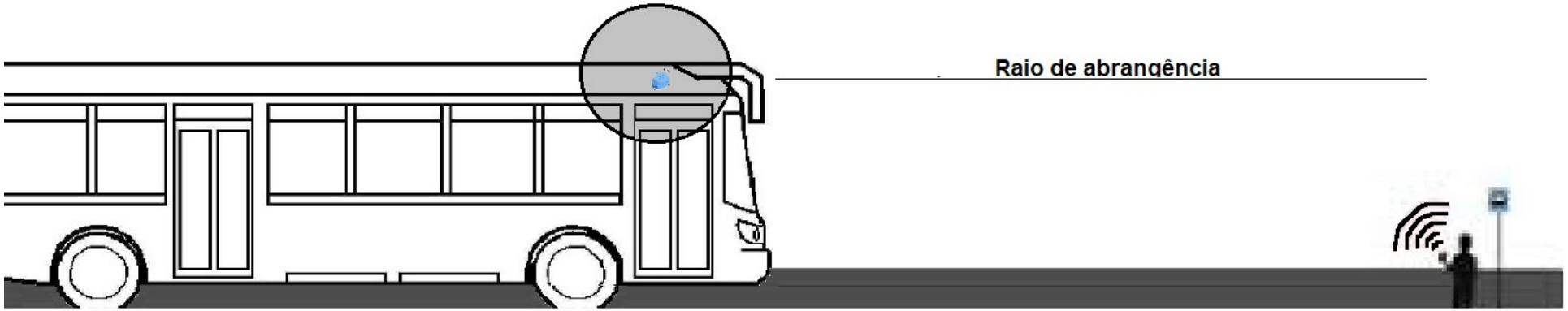
Implementação: BlindBus User



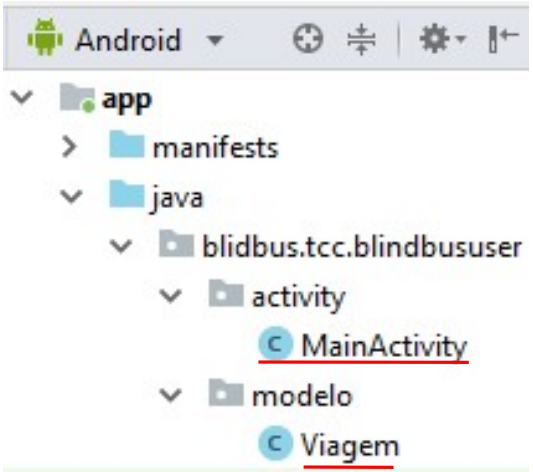

```
@Override
public void onBeaconServiceConnect() {
    try {
        // Comece a procurar por beacons que correspondam ao objeto Last Region, incluindo
        // atualiza a distância estimada
        mBeaconManager.startRangingBeaconsInRegion(mRegion);
        alert("Buscando ônibus");
    } catch (RemoteException e) {
        Log.d(TAG, msg: "Ocorreu um erro ao começar a procurar ônibus " + e.getMessage());
    }
    mBeaconManager.addRangeNotifier(this);
}
```

```
private void startDetectingBeacons() {
    // Definir um período de verificação
    mBeaconManager.setForegroundScanPeriod(DEFAULT_SCAN_PERIOD_MS);
    // Link para o serviço de beacons. Obter um retorno de chamada quando estiver pronto para ser usado
    mBeaconManager.bind( consumer: this);
    // Desativar o botão Iniciar
    getStartButton().setEnabled(false);
    getStartButton().setAlpha(.5f);
    // Ativar o botão Parar
    getStopButton().setEnabled(true);
    getStopButton().setAlpha(1);
}
```

Implementação: BlindBus User

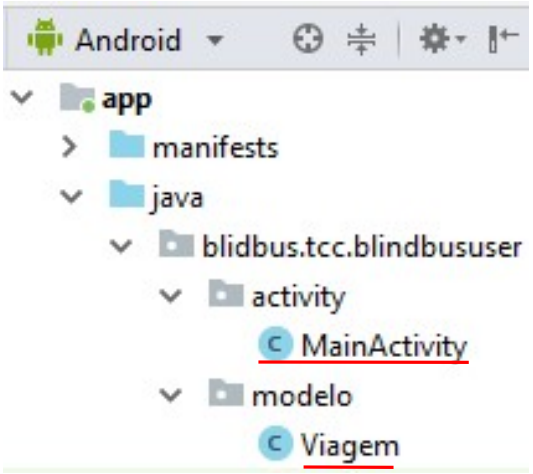


Implementação: BlindBus User




```
@Override
public void didRangeBeaconsInRegion(Collection<Beacon> beacons, Region region) {
    for (Beacon beacon : beacons) {
        instanceBeacon = beacon.getId2().toString();
        lerViagem();
        if(instanceBeacon.equals(viagem.getInstanceBeacon())){
            atualizaViagem(viagem);
            if(verificaRepeticaoMensagem.equals(viagem.getLinha()+" "+viagem.getSentido())){
                try {
                    new Thread().sleep( millis: 10000);
                    alert(viagem.getLinha()+" "+viagem.getSentido());
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }else{
                verificaRepeticaoMensagem = viagem.getLinha()+" "+viagem.getSentido();
                alert(viagem.getLinha()+" "+viagem.getSentido());
            }
            viagem = new Viagem();
        }
    }
}
public void atualizaViagem(Viagem viagem) {
    viagem.setClienteEncontrado("SIM");
    databaseReference.child("Viagem/"+viagem.getInstanceBeacon()).setValue(null);
    databaseReference.child("Viagem").child(viagem.getInstanceBeacon()).setValue(viagem);
}
```

Implementação: BlindBus User



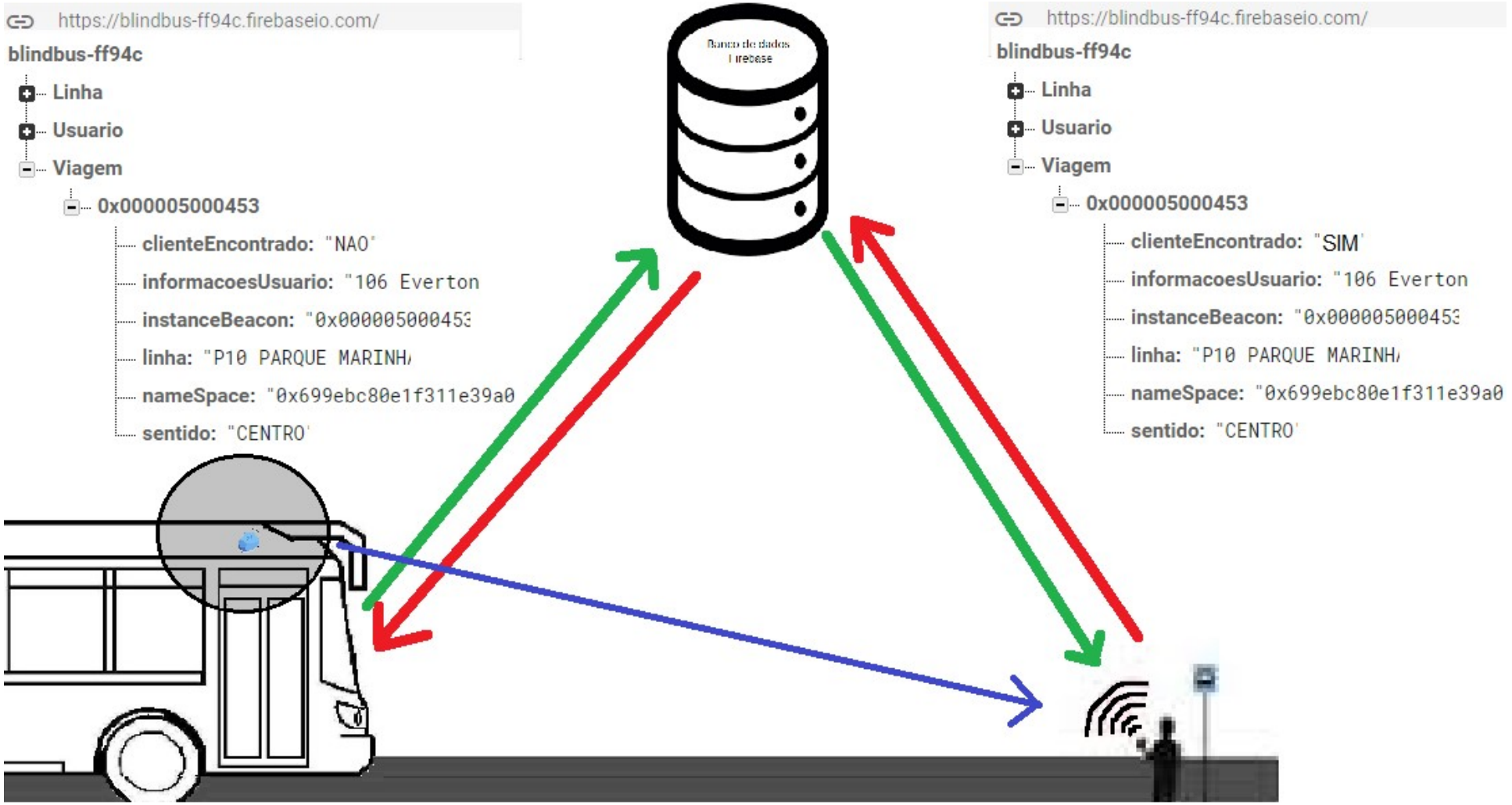
Comunicação Internet

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

```
protected boolean verificaConexao() {
    boolean conectado;
    ConnectivityManager conectivtyManager = (ConnectivityManager)
            getSystemService(Context.CONNECTIVITY_SERVICE);
    if (conectivtyManager.getActiveNetworkInfo() != null
            && conectivtyManager.getActiveNetworkInfo().isAvailable()
            && conectivtyManager.getActiveNetworkInfo().isConnected()) {
        conectado = true;
    } else {
        conectado = false;
    }
    return conectado;
}
```

Ambas aplicações

Comunicação Serviço Externo



Ambas aplicações

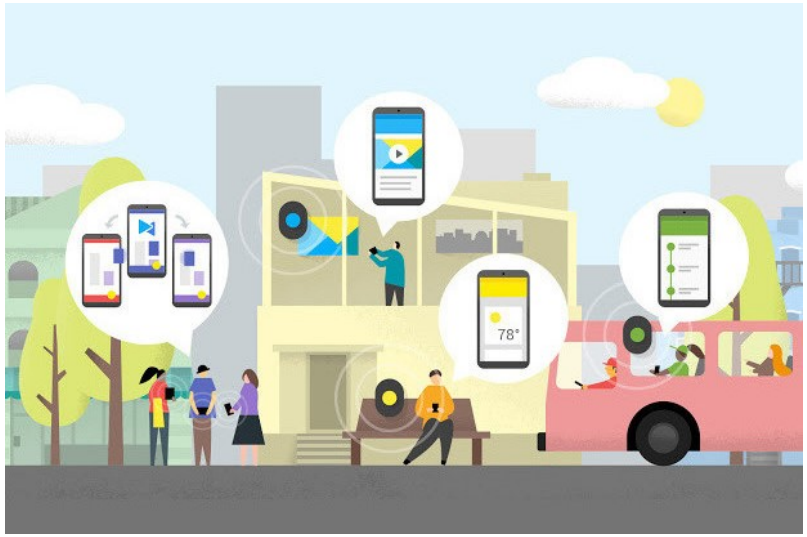
Comunicação Serviço Interno

```
import android.content.ContentValues;
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
public abstract class DAO extends SQLiteOpenHelper {
    public static final String DATABASE ="bdblindbus";
    public static final int VERSION = 7;
    public DAO(Context context) { super(context, DATABASE, factory: null, VERSION); }
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(getUpgradeTableScript());
        db.execSQL(getCreateTableScript());
    }
    protected void inserir(ContentValues values) {
        getWritableDatabase().insert(getTableName(), nullColumnHack: null,values);
    }
    protected abstract String getTableName();
    protected abstract String getCreateTableScript();
    protected abstract String getUpgradeTableScript();
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL(getUpgradeTableScript());
    }
}
```

Somente BlindBus Admin

Comunicação Beacon

```
import org.altbeacon.beacon.Beacon;
import org.altbeacon.beacon.BeaconConsumer;
import org.altbeacon.beacon.BeaconManager;
import org.altbeacon.beacon.BeaconParser;
import org.altbeacon.beacon.Identifier;
import org.altbeacon.beacon.RangeNotifier;
import org.altbeacon.beacon.Region;
public class MainActivity extends AppCompatActivity implements View.OnClickListener, BeaconConsumer, RangeNotifier {
    private static final int REQUEST_ENABLE_BLUETOOTH = 1;
    private static final long DEFAULT_SCAN_PERIOD_MS = 1000;
    private static final String ALL_BEACONS_REGION = "AllBeaconsRegion";
    BeaconManager mBeaconManager;
    private Region mRegion;
```



Somente BlindBus User

Comunicação Beacon

```
@Override
public void onBeaconServiceConnect() {
    try {
        mBeaconManager.startRangingBeaconsInRegion(mRegion);
        alert(getString(R.string.ligando_busca_de_beacons));
    } catch (RemoteException e) {
        Log.d(TAG,  msg: "Ocorreu um erro ao começar a procurar ônibus " + e.getMessage());
    }
    mBeaconManager.addRangeNotifier(this);
}
@Override
public void didRangeBeaconsInRegion(Collection<Beacon> beacons, Region region) {
    for (Beacon beacon : beacons) {
        instanceBeacon = beacon.getId2().toString();
    }
}
private void stopDetectingBeacons() {
    try {
        mBeaconManager.stopMonitoringBeaconsInRegion(mRegion);
        alert(getString(R.string.desligando_busca_de_beacons));
    } catch (RemoteException e) {
        Log.d(TAG,  msg: "Ocorreu um erro ao parar de procurar ônibus " + e.getMessage());
    }
    mBeaconManager.removeAllRangeNotifiers();
    mBeaconManager.unbind( consumer: this);
}
```

Somente BlindBus User

Links Importantes

- Integração android e beacon:
  https://altbeacon.github.io/android-beacon-library/.
- Conheça o Android Studio:
  https://developer.android.com/studio/intro/index.html?hl=pt-br
- Slides e projeto:
  https://github.com/evertondealmeida/BlindBus

Tchê Linux

Tecnologia assistiva para deficientes visuais no uso de transporte público

Obrigado...