UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ESPECIALIZAÇÃO EM TECNOLOGIA JAVA

JOHN CHARLES DA SILVA

# APLICATIVO ANDROID PARA DIVULGAÇÃO E ORGANIZAÇÃO DE EVENTOS SELETIVOS

MONOGRAFIA DE ESPECIALIZAÇÃO

PATO BRANCO
2017

JOHN CHARLES DA SILVA

# APLICATIVO ANDROID PARA DIVULGAÇÃO E ORGANIZAÇÃO DE PROCESSOS SELETIVOS, CONCURSOS E AFINS

Monografia de especialização apresentado ao Curso de Especialização em Tecnologia Java, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Especialista.

Orientador: Prof. Robison Cris Brito

PATO BRANCO
2017

**MINISTÉRIO DA EDUCAÇÃO**
Universidade Tecnológica Federal do Paraná
Câmpus Pato Branco
Departamento Acadêmico de Informática
Curso de Especialização em Tecnologia Java

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

## TERMO DE APROVAÇÃO

APLICATIVO ANDROID PARA DIVULGAÇÃO E ORGANIZAÇÃO DE EVENTOS SELETIVOS

por

JOHN CHARLES DA SILVA

Este trabalho de conclusão de curso foi apresentado em 19 de dezembro de 2017, como requisito parcial para a obtenção do título de Especialista em Tecnologia Java. Após a apresentação o candidato foi arguido pela banca examinadora composta pelos professores Andreia Scariot Beulke e Vinicius Pegorini. Em seguida foi realizada a deliberação pela banca examinadora que considerou o trabalho aprovado.

_____
Robison Cris Brito
Prof. Orientador (UTFPR)

_____
Andreia Scariot Beulke
Banca (UTFPR)

_____
Vinicius Pegorini
Banca (UTFPR)

_____
Robison Cris Brito
Coordenador da IV Especialização em Tecnologia Java

A Folha de Aprovação assinada encontra-se na Coordenação do Curso.

## AGRADECIMENTOS

Agradeço a dedicação dos professores do Curso de Especialização em Tecnologia Java, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, cujo ensino proporcionou a aquisição dos conhecimentos necessários para que o projeto apresentado nesta monografia se tornasse realidade.

# RESUMO

SILVA, John Charles da. Aplicativo Android para divulgação e organização de Processos Seletivos, Concursos e Afins. 2017. 36f. Monografia (Trabalho de Especialização) – Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2017.

O objetivo deste trabalho foi desenvolver uma solução para o processo de recrutamento, seleção, triagem, encaminhamento, treinamento e comunicação envolvido no controle de eventos sazonais remunerados (concursos, processos seletivos, entre outros). Foi, inicialmente, um estudo para que fossem definidadas as funcionalidades e necessidades que o sistema deveria possuir. Concluiu-se que a melhor forma de atender estas necessidades seria o desenvolvimento de uma solução para dispositivos móveis. Optou-se pelo uso do Android Studio para o desenvolvimento focado na plataforma Android. Utilizando, ainda, as ferramentas providas pela Firebase SDK como solução de armazenamento e análise, assim como a notificação de mensagens. Como resultado deste trabalho foi possível criar um aplicativo que contemplou os requisitos levantados.

**Palavras-chave**: Dispositivos Móveis. Java. Android. Firebase.

## ABSTRACT

SILVA, John Charles da Silva. MOBILE APPLICATION FOR EVENT CONTROL. 2017. 36f. Monografia (Trabalho de especialização) – Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2017.

The objective of this work was to develop a solution for the recruitment, selection, sorting, routing, training and communication process involved in the control of remunerated seasonal events (competitions, selective processes, etc). A study was initially carried out to define the functionalities and needs that the system should have. It was defined that the best way to meet these needs would be to develop a solution for mobile devices. It was then decided to use Android Studio to focus on the Android platform using the tools provided by the Firebase SDK as a storage, analysis and notification solution.



**Keywords**: App. Java. Android. Firebase.

**LISTA DE FIGURAS**

## LISTA DE QUADROS

## LISTAGENS DE CÓDIGOS

## LISTA DE SIGLAS

SGBD Sistema de Gerenciamento de Banco de Dados

WORA "Write once, run anywhere" ou "Escreva uma vez, execute em qualquer lugar", é o slogan da Sun MicroSystems para exemplificar os benefícios multi-plataforma da Linguagem Java.

JVM Java Virtual Machine.

**SUMÁRIO**

# 1 INTRODUÇÃO

## 1.1 CONSIDERAÇÕES INICIAIS

Os processos seletivos (concursos, vestibulares, entre outros) possuem como objetivo avaliar candidatos que disputam um número limitado de vagas. Estes processos seletivos existem tanto na esfera pública como na privada e são caracterizados por serem realizados em curtos períodos de tempo (geralmente, um dia).

As pessoas envolvidas na organização, aplicação, correção e demais tarefas necessárias para realização de um destes eventos, em geral possuem um comprometimento limitado ao dia da aplicação, sem vínculo empregatício com o órgão que geriu o processo.

Existem diversas funções a serem ocupadas nestes eventos e algumas podem ser consideradas como a ponta final do processo, sendo elas: os chefes de sala, fiscais aplicadores, fiscais volantes e apoios. Estas funções, na maioria das vezes, representam a maior parte da mão de obra envolvida no processo.

A divulgação para que pessoas possam se candidatar a trabalhar nestes eventos fica a cargo dos coordenadores de locais, que não possuem nenhuma ferramenta para tal tarefa e as realizam básicamente fazendo uso de algumas mídias sociais, divulgação “boca a boca”, emails, entre outras. Como este processo de divulgação não é centralizado, pode apresentar falhas como mal entendidos e atrasos na comunicação.

Feita a divulgação, o passo seguinte é a seleção das pessoas para o preenchimento do quadro de trabalho. Nesta fase faz-se necessário um levantamento do histórico pessoal de cada um, no qual informações essenciais, tais como o número de eventos dos quais ela já participou e quais funções exerceu, além das avaliações e observações pregressas determinariam quais vagas esta pessoa estaria apta a preencher. Além disso, fatores como a ordem em que estas pessoas manifestaram interesse em trabalhar também constitue um ponto importante no momento em que as funções são designadas. O correto controle das quantidades de vagas e das pessoas que as preencheram é fundamental para que não sobre ou falte pessoal no dia da aplicação do evento. Este controle, atualmente, é feito básicamente em planilhas eletrônicas ou similares.

Seguindo a fase seguinte do processo seria a contínua comunicação entre a coordenação do evento e as pessoas selecionadas para participarem do mesmo. Avisos importantes, hórarios de reuniões, etc precisam ter garantia de ser entregues e lidos para que o processo se desenvolva com sucesso.

Atualmente, esta comunicação é feita utilizando-se de email, porém, sem uma plataforma específica de comunicação para o evento, mensagens enviadas podem ser perdidas e, consequentemente, não respondidas, o que gera problemas para o processo. Para facilitar esta comunicação, optou-se no desenvolvimento de um software para dispositivos móveis.

O primeiro telefone móvel surgiu em 1973 (MÜLLER, 2017), mas apenas na década de 80 tornaram-se viáveis comercialmente, ainda que de uso acessível apenas as pessoas de maior poder aquisitivo. Porém o seu uso original, fazer ligações telefônicas, alterou-se substancialmente nas ultimas décadas com o advento de novas tecnologias, forçando uma mudança de percepção em relação a este dispositivo. É muito raro encontrar, nos dias atuais, quem não possua ou ao menos já não tenha usado um smartphone. O uso destes aparelhos não para de crescer no Brasil, em 4 anos, o número de pessoas que usam esse tipo de celular no Brasil subiu 3,5 vezes, passando de 14% em 2012 para 62% em 2016 (UOL, 2017).

O Android é um sistema operacional *open source* baseado em Linux e inicialmente desenvolvido pela Android Inc, com apoio da Google, que posteriormente comprou a empresa. O Android foi projetado para dispositivos móveis touchscreen, como smartphones e tablets. O primeiro telefone com Android foi vendido em outubro de 2008 (OLHAR DIGITAL, 2013). A escolha do Android justificou-se pela forte presença deste no mercado de dispositivos móveis, englobando, atualmente, mais de 85% dos usuários conforme, pode ser visto na Figura 16 em Anexos.

## 1.2 OBJETIVOS

O objetivo do projeto apresentado nesta monografia é suprir a necessidade de automação do controle de eventos, gerenciando os passos envolvidos no processo dos mesmo, por meio do uso de *smartphones* tanto pelos colaboradores, bem como dos responsáveis pela coordenação do mesmo.

### 1.2.2 OBJETIVOS ESPECIFICOS

- Estabelecer uma forma de divulgação dos eventos;
- Realizar o cadastro de interessados;
- Fazer a seleção de pessoal;
- Criar um canal de comunicação entre as partes;

# 2. FERRAMENTAS, TECNOLOGIAS E PROCEDIMENTOS

A ênfase deste capítulo está em reportar o que foi utilizado para implementar o aplicativo proposto neste trabalho.

## 2.1 FERAMENTAS E TECNOLOGIAS

As ferramentas e tecnologias utilizadas neste projeto são listadas no Quadro 1.

| Ferramenta / Tecnologia | Versão | Disponível em | Aplicação |
|---|---|---|---|
| gliffy.com | N/A | www. gliffy.com | Modelagem do sistema |
| Yuml | N/A | www.yuml.com | Diagramas de Uso de Caso |
| Java | 1.8.0 | www.java.com | Linguagem para desenvolvimento da aplicação |
| Gson | 2.8.2 | code.google.com | Biblicoteca para manipulação de JSON |
| MaskFormatter | 1.0 | https://github.com/rtoshiro/MaskFormatter | Biblioteca para formatação de entrada de dados. |
| Firebase | 11.4.2 | firebase.google.com | Conjunto de bibliotecas para autenticação, comunicação, armazenamento de dados e analise |
| SQLite | 3.19.4 | Nativo, pré-instalado no aparelho | SGBD |
| Android Studio | 3.1 | developer.android.com | *Integrated Development Environment* (IDE) para desenvolvimento da aplicação |

**Quadro 1 – Tecnologias e ferramentas utilizadas na modelagem e na implementação do aplicativo**

### 2.1.1 GLYFFY

Para a diagramação e modelagem do sistema optou-se pelo uso do Gliffy (www.gliffycom) . A Gliffy foi fundada em 2005 por Chris Kohlhardt e Clint Dickson em

São Francisco, Califórnia, e é a primeira aplicação gráfica de negócios online do mundo. A Gliffy facilita a criação, compartilhamento e colaboração em uma ampla gama de diagramas, permitindo que as pessoas se comuniquem com mais clareza, promovam a inovação, melhorem as decisões e trabalhem de forma mais eficaz. Embora não seja uma solução gratuíta, ela possui um período de testes que foi suficiente para uso neste projeto. A opção por esta solução baseou-se na sua facilidade e disponibilidade. Na Figura 1 é possível observar uma tela da ferramenta.

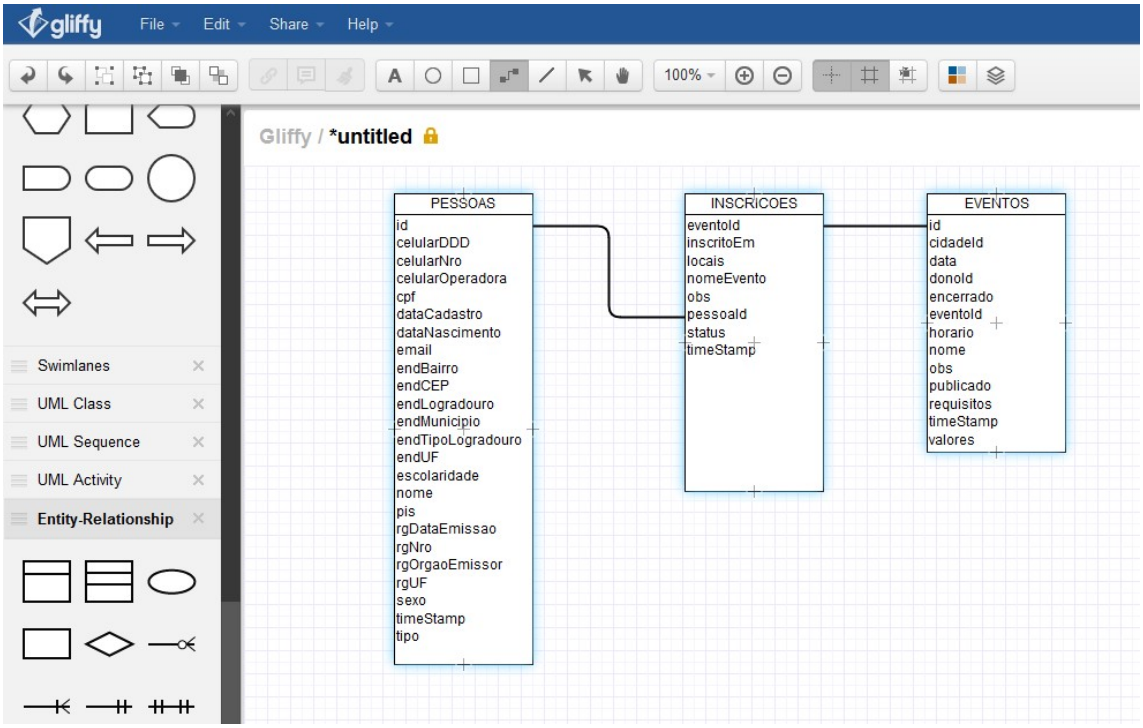


**Figura 1 Ambiente de modelagem do Gliffy**

### 2.1.2 yUML

O yUML é uma ferramenta online gratuita para a criação rápida de diagramas de caso de uso de forma programada utilizando uma linguagem simplificada sem a necessidade de ferramentas de desenho. Na Figura 2 é mostrada uma tela da ferramenta

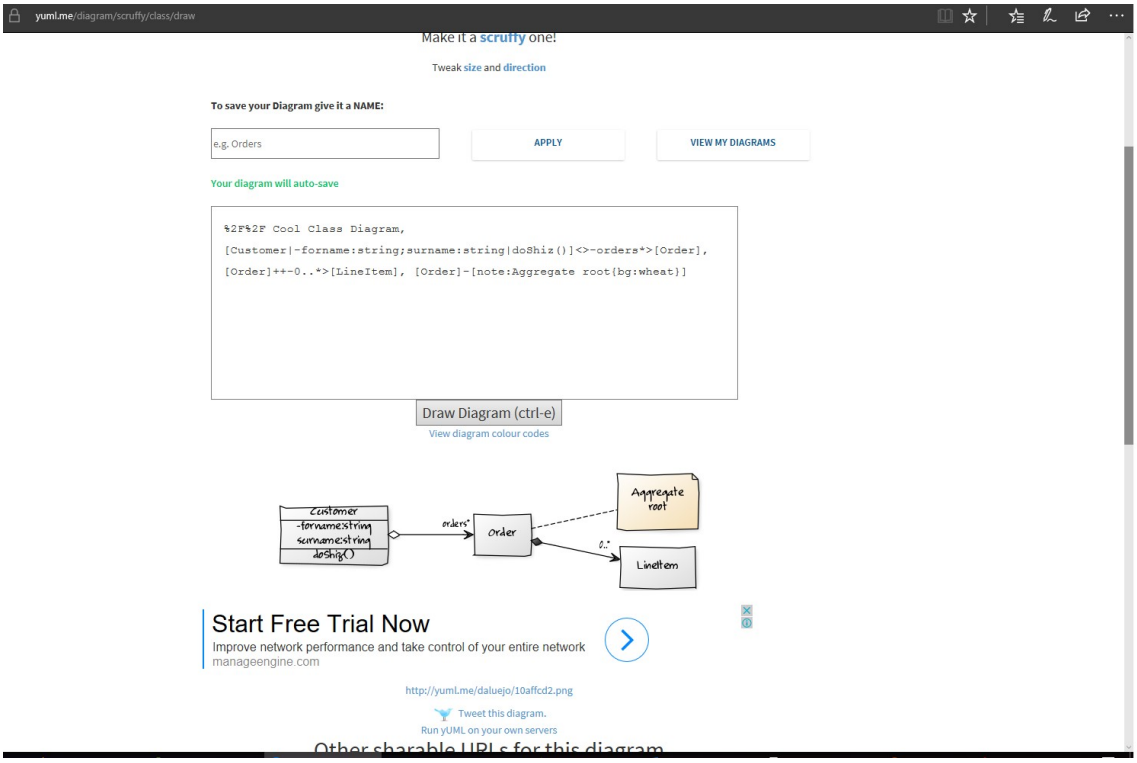


**Figura 2 Exemplo de utilização do yUML.**

### 2.1.3 JAVA

Java é uma linguagem de programação de computador de uso geral, baseada em classes e orientada a objetos Seu objetivo é permitir que os desenvolvedores de aplicativos "escrevam uma vez, execute em qualquer lugar" (WORA) , o que significa que o código Java compilado pode ser executado em todas as plataformas que suportam Java sem a necessidade de recompilação. Os aplicativos Java geralmente são compilados para bytecode que podem ser executados em qualquer máquina virtual Java (JVM), independentemente da arquitetura do computador. O Java tornou-e uma das linguagens de programação mais populares em uso, particularmente para aplicações web cliente-servidor, com mais de 9 milhões de desenvolvedores. O Java foi originalmente desenvolvido por James Gosling na Sun Microsystems (que desde então foi adquirido pela Oracle Corporation) e lançado em 1995 como um componente central da plataforma Java da Sun Microsystems. O idioma deriva muito da sua sintaxe de C e C ++

### 2.1.4 GSON

Gson é uma biblioteca Java que pode ser usada para converter objetos Java em sua representação JSON. Ele também pode ser usado para converter uma String JSON em um objeto Java equivalente. Gson é um projeto de código aberto e se encontra hospedado em http://code.google.com/p/google-gson. Gson pode trabalhar com objetos Java arbitrários, incluindo objetos pré-existentes os quais você não possua código-fonte.

### 2.1.5 MASKFORMATTER

MaskFormatter (https://github.com/rtoshiro/MaskFormatter) é uma biblioteca Java que pode ser utilizada para formatação de Strings usando máscaras, podendo ser inclusive utilizada com TextWatcher (https://developer.android.com/reference/android/text/TextWatcher.html), sendo este um componente visual da plataforma Android.

### 2.1.6 FIREBASE

O Firebase (firebase.google.com) é uma plataforma de desenvolvimento para aplicativos móveis e web desenvolvida inicialmente pela empresa de mesmo nome em 2011 e adquirida pelo Google em 2014. A partir de sua aquisição a plataforma passou a integrar-se com diversos serviços da empresa e tornou-se uma plataforma unificada para desenvolvedores móveis, oferecendo uma ampla gama de serviços tais como Analytics, Cloud Messaging, Auth, Realtime Database, Storage, etc. Na Figura 3 pode-se ver o ambiente online da ferramenta Firebase.

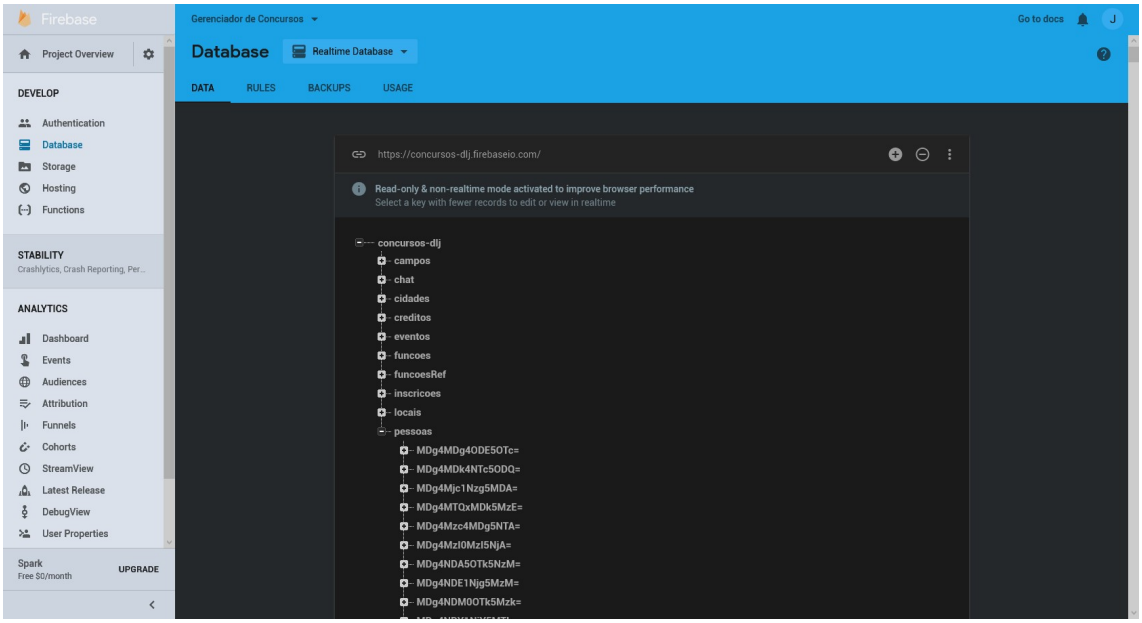


**Figura 3 Ambiente do Firebase**

### 2.1.7 SQLite

**SQLite** é uma biblioteca em linguagem C que implementa um banco de dados SQL embutido. Programas que usam a biblioteca SQLite podem ter acesso a banco de dados SQL sem executar um processo SGBD separado.

O SQLite é um banco de dados pequeno, rápido e embutido. O que o torna popular é a combinação do mecanismo de banco de dados e da interface em uma única biblioteca, bem como a capacidade de armazenar todos os dados em um único arquivo. Sua funcionalidade está entre o MySQL eo PostgreSQL, porém é mais rápido que os dois bancos de dados (CHRIS NEWMAN, 2004)

### 2.1.8 ANDROID STUDIO

O Android Studio é o ambiente gratuito de desenvolvimento integrado (IDE) oficial para o desenvolvimento de aplicativos Android e é baseado no IntelliJ IDEA_. Além do editor de código e das ferramentas de desenvolvedor avançados do IntelliJ, o Android Studio oferece ainda mais recursos para aumentar sua produtividade na criação de aplicativos Android, como:

- Um sistema de compilação flexível baseado no Gradle;
- Um emulador rápido com inúmeros recursos;
- Um ambiente unificado para você poder desenvolver para todos os dispositivos Android;
- Instant Run para aplicar alterações a aplicativos em execução sem precisar compilar um novo APK;
- Modelos de códigos e integração com GitHub para ajudar a criar recursos comuns dos aplicativos e importar exemplos de código;
- Ferramentas e estruturas de teste cheias de possibilidades;
- Ferramentas de verificação de código suspeito para detectar problemas de desempenho, usabilidade, compatibilidade com versões e outros;
- Compatibilidade com C++ e NDK;
- Compatibilidade embutida com o Google Cloud Platform, facilitando a integração do Google Cloud Messaging e do App Engine;

A Figura 4 apresenta a tela do ambiente do Android Studio

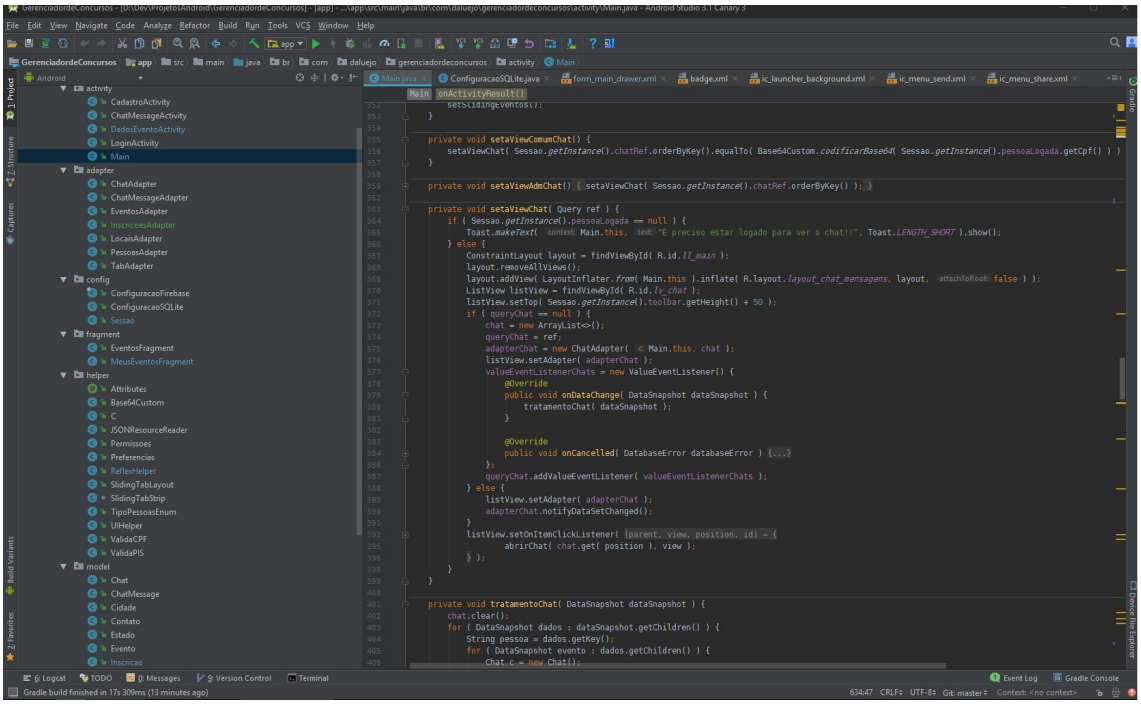

**Figura 4 Ambiente do Android Studio**

# 3 RESULTADOS

Este capítulo apresenta o resultado da realização deste trabalho que é um sistema para divulgação e organização de eventos seletivos.

## 3.1 ESCOPO DO SISTEMA

O aplicativo desenvolvido neste trabalho busca apresentar um meio mais acessível e centralizado para a divulgação de eventos sazonais bem como ajudar na organização e seleção de pessoal para aplicação e na contínua comunicação entre os mesmos e os organizadores do evento. Para que estas tarefas sejam realizadas o aplicativo possui dois tipos de perfis diferentes, que são  o coordenador de eventos e o inscrito.

Para iniciar o processo, após feito o *download* do app no site, o usuário deverá realizar o seu cadastro, no qual irá escolher o seu perfil, que definirá quais dados serão obrigatórios. A maior parte dos eventos requer um cadastro mais completo, incluindo números de documentos que serão validados pelo app. Após realizado o cadastro, na próxima inicialização, bastará  que usuário informar o seu CPF. Quando o perfil for definido pelo administrador do sistema como coordenador, opcionalmente poderá ser cadastrada uma senha como forma de segurança, visto que o CPF é uma informação pública.

A sequência do processo no app seguirá, a partir deste ponto funções distintas: ao coordenador de evento será dada a opção de criar eventos, gerenciar inscrições enviar mensagens e responder *chats*; ao participante será mostrado uma lista de eventos disponíveis, o *status* das inscrições que ele realizou, listagem de mensagens recebidas e um *chat* para conversar diretamente com o coordenador dos eventos que ele irá participar. Todos os dados entrados no app serão persistidos utilizados o Firebase Realtime Database.

### 3.1.1 MODELAGEM DO SISTEMA

Os requisitos levantados para a execução do projeto estão listados no Quadro 2.

| Identificação | Nome | Descrição |
|---|---|---|
| 01 | Realizar o cadastro do | O coordenador realiza o cadastro do evento, definindo o horário, |

| | | |
|---|---|---|
| | Evento | local, requisitos e demais informações. |
| 02 | Realizar o cadastro dos interessados | Cada pessoa interessada em participar do evento deverá realizar o seu cadastro pessoal, preenchendo ps dados necessários. |
| 03 | Realizar a inscrição no evento | Para participar do evento, os interessados deverão acessar os eventos e registrar sua inscrição. |
| 04 | Avaliação da inscrição | O coordenador faz a avaliação e define se a inscrição foi ou não aceita utilizando os critérios definidos para o evento, bem como a avaliação do histórico pregresso do inscrito, quando houver. |

**Quadro 2 – Requisitos funcionais**

Com base nos requisitos funcionais foi criado o diagrama de caso de uso mostrado na Figura 5.

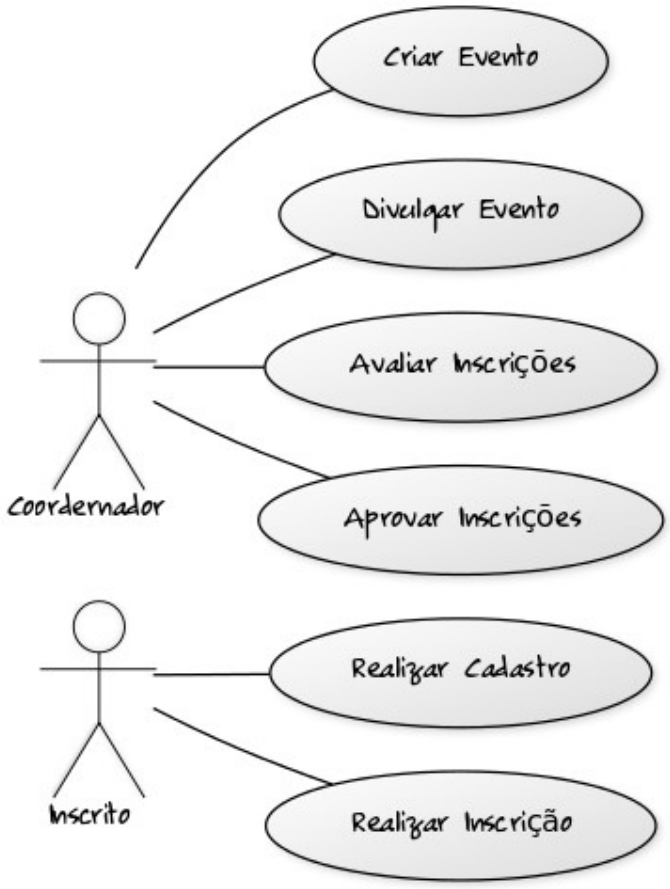


**Figura 5 Diagrama de Caso de Uso**

Após o cadastro, o usuário com perfil de oordenador poderá criar um evento e divulga-lo, para a lista de pessoas cadastradas no sistema, ou seja, pessoas que participaram

de outros eventos. Também é de responsabilidade do coordenador avaliar a inscrição dos candidatos que desejam participar do evento e aprovar estas inscrições.

Aos inscritos, que são as pessoas interessadas em trabalhar nos eventos, é possível realizar o cadastro, verificar a lista de eventos disponíveis, receber notificação sobre novos eventos cadastrados por coordenador, realizar a inscrição nos eventos.

Na Figura 6 é possível visualizar o modelo de funcionamento do app. O mesmo funciona localmente, em um aparelho Android, porém, cada funcionalidade requer comunicação com o servidor Firebase, que será responsável por integrar os dados dos diferentes aparelhos Android que executam a aplicação.

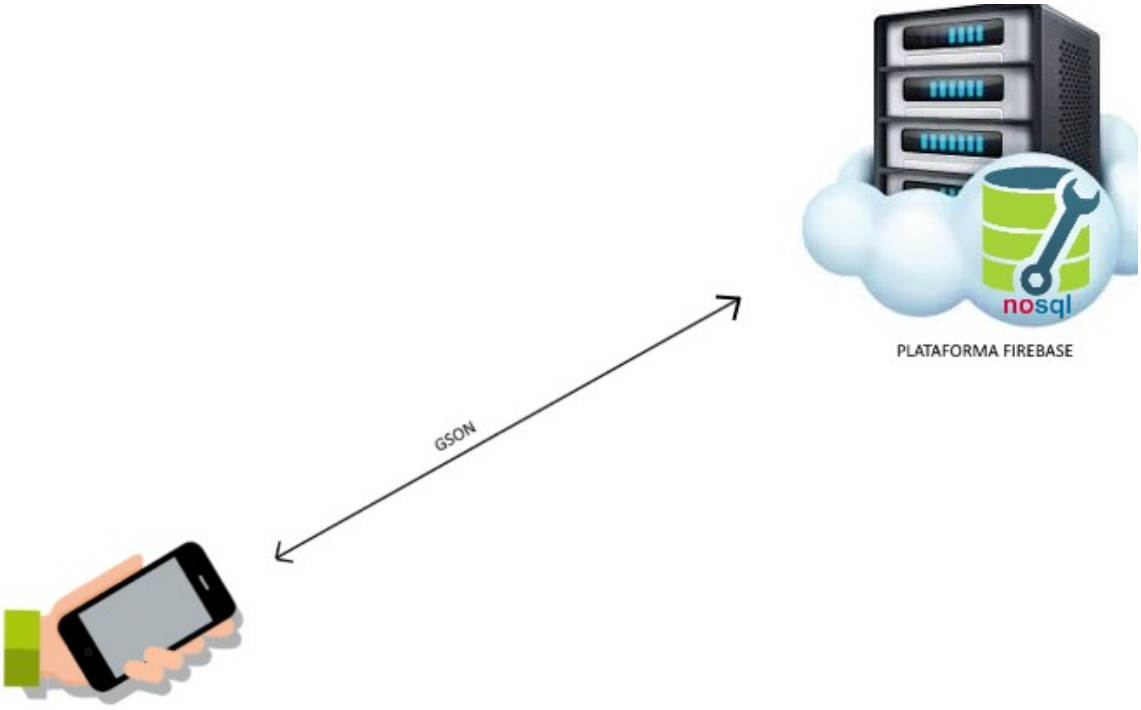


**Figura 6 Funcionamento de integração com o servidor.**
**Fonte: Autoria Própria**

Para a comunicação entre o Android e o Firebase foi utilizada o biblioteca GSON da Google, que permite converter os dados trafegados no padrão JSON.

## 3.3 APRESENTAÇÃO DO SISTEMA

O app possui dois tipos de perfis de usuários, os coordenadores de evento e os interessados em trabalhar no evento. Ao iniciar o app, será apresentado as opções de autenticar por meio do CPF e da realização do cadastro, caso ainda não possua um. Optou-se por não utilizar uma senha de acesso para o app. Após o cadastro, o administrador do sistema poderá alterar diretamente no banco de dados da aplicação, dando perfil de Coordenador a um determinado CPF, neste caso, ao logar com este CPF, o coordenador terá a opção de cadastrar uma senha, por questões de segurança.

A Figura 7 apresenta a tela inicial de login.

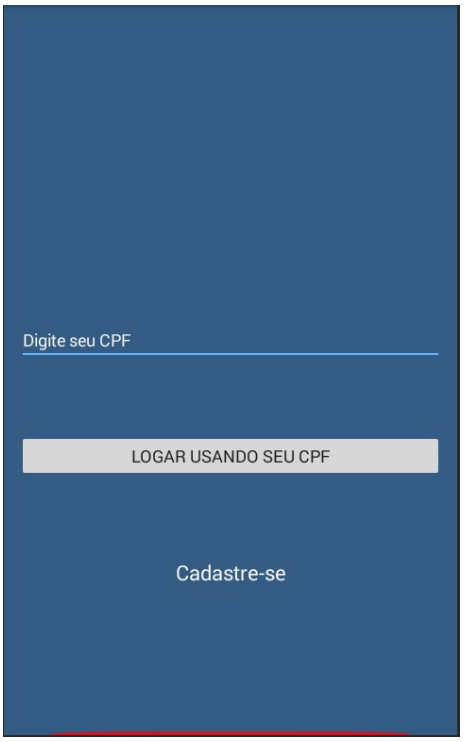


**Figura 7 Tela de Login**

### 3.3.1 ACESSANDO O SISTEMA COMO COORDENADOR

Ao acessar o app como coordenador será apresentado ao usário uma listagem dos eventos cadastrados, como mostrado na Figura 8. Nesta mesma tela existe a opção de incluir um novo evento.

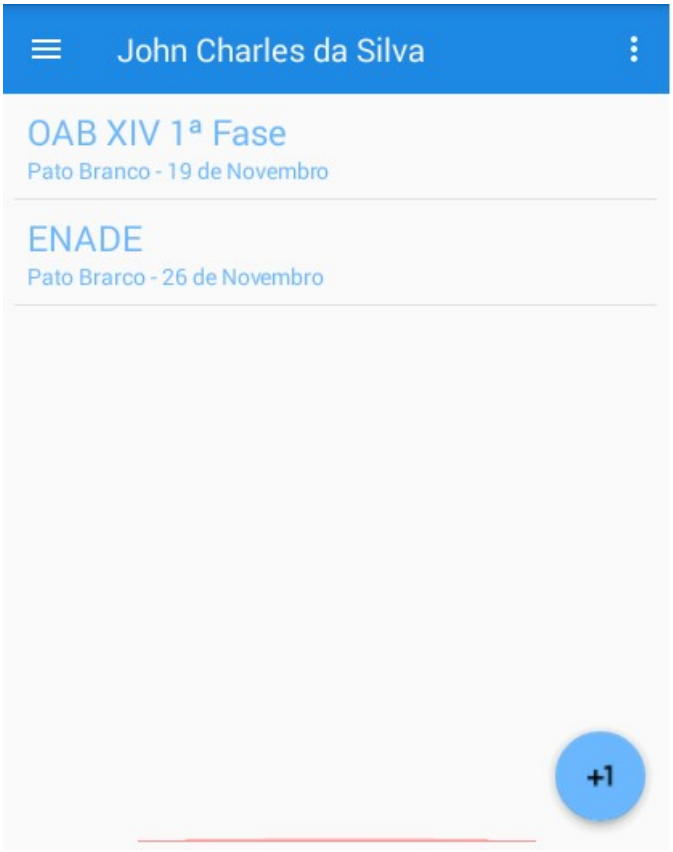


**Figura 8 Eventos Cadastrados**

Ao escolher cadastrar um novo evento clicando no botão com o sinal +1 o usuário será direcionado para a tela no qual os dados do novo evento deverão ser digitados, conforme mostrado na Figura 9.

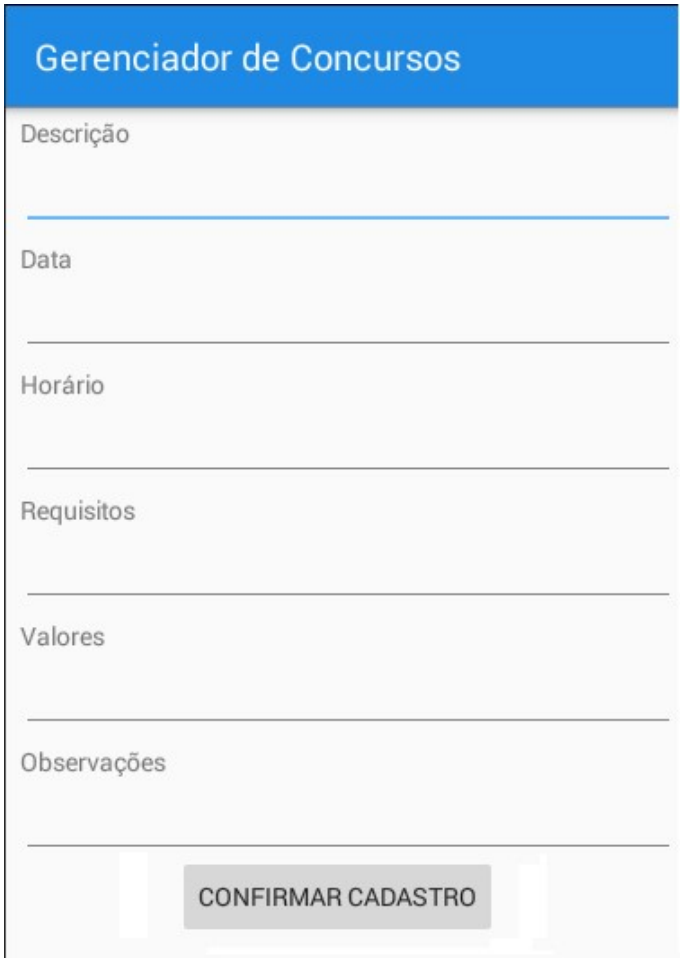


**Figura 9 Inserindo um novo evento**

Após efetuar o cadastro de um novo evento, todos os usuários do sistema poderão visualizá-lo. Quando o aplicativo for acessado será disponibilizada a opção de realizar a inscrição para trabalhar neste evento. O coordenador poderá utilizar o *console* do Firebase para disparar uma notificação a todos os usuários tornando, assim, mais rápido o preenchimento das funções.

O coordenador pode acessar a listagem contendo todos os inscritos em todos os eventos, atuais e passados, conforme visto na Figura 10.


**Figura 10 Listagem de Inscritos**

Ao acessar um evento, o coordenador irá visualizar a listagem dos inscritos, podendo acessar o histórico destes inscritos (por meio do toque longo na inscrição) e alterar o *status* da inscrição (por meio do toque curto na inscrição), como pode ser visto na Figura 11.

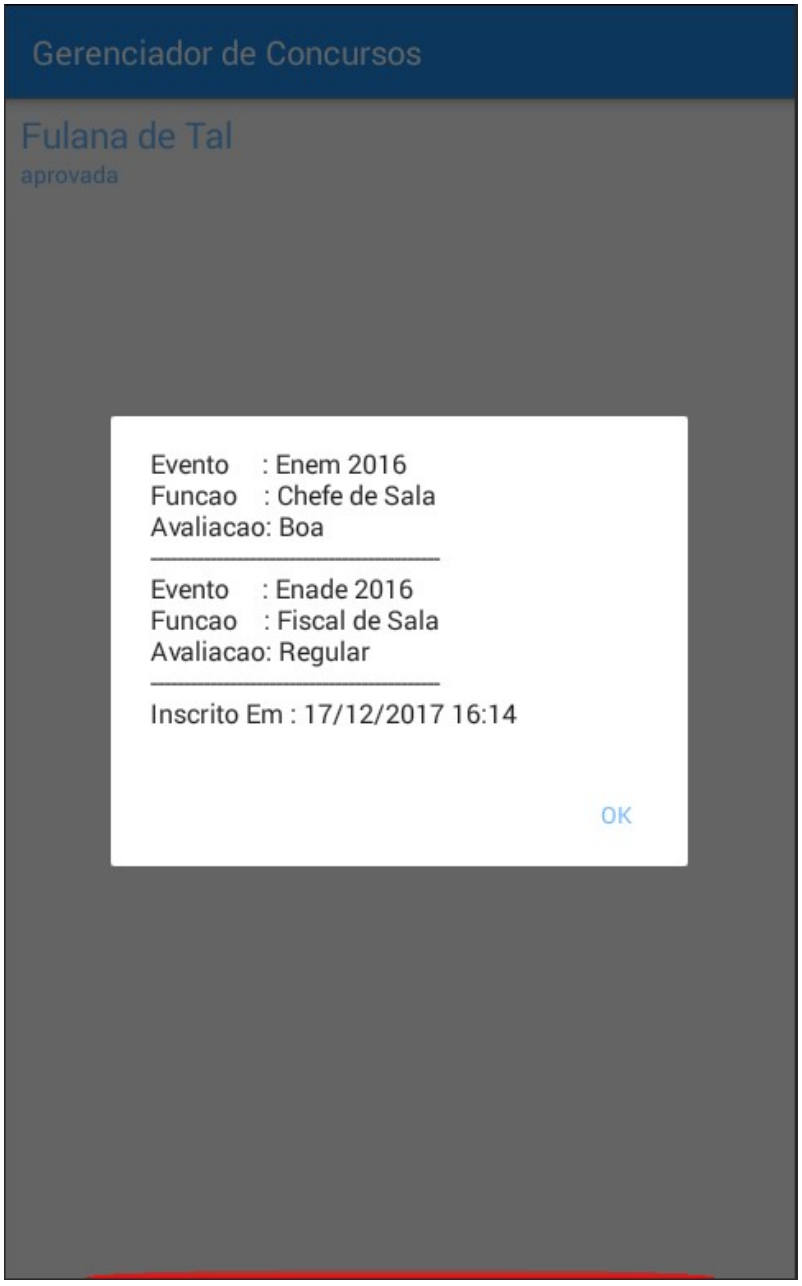


**Figura 11 - Visualizando o Histórico do Inscrito**

### 3.3.2 ACESSANDO O SISTEMA COMO INSCRITO

Ao realizar o acesso com o perfil de inscrito, o usuário será direcionado para uma listagem dos eventos atualmente disponíveis, semelhante a tela do coordenador, com o diferencial que a listagem estará dividida entre eventos disponíveis e eventos ele optou por se inscrever juntamente com o *status* da sua inscrição. A Figura 12 mostra uma montagem destas duas listagens lado a lado.

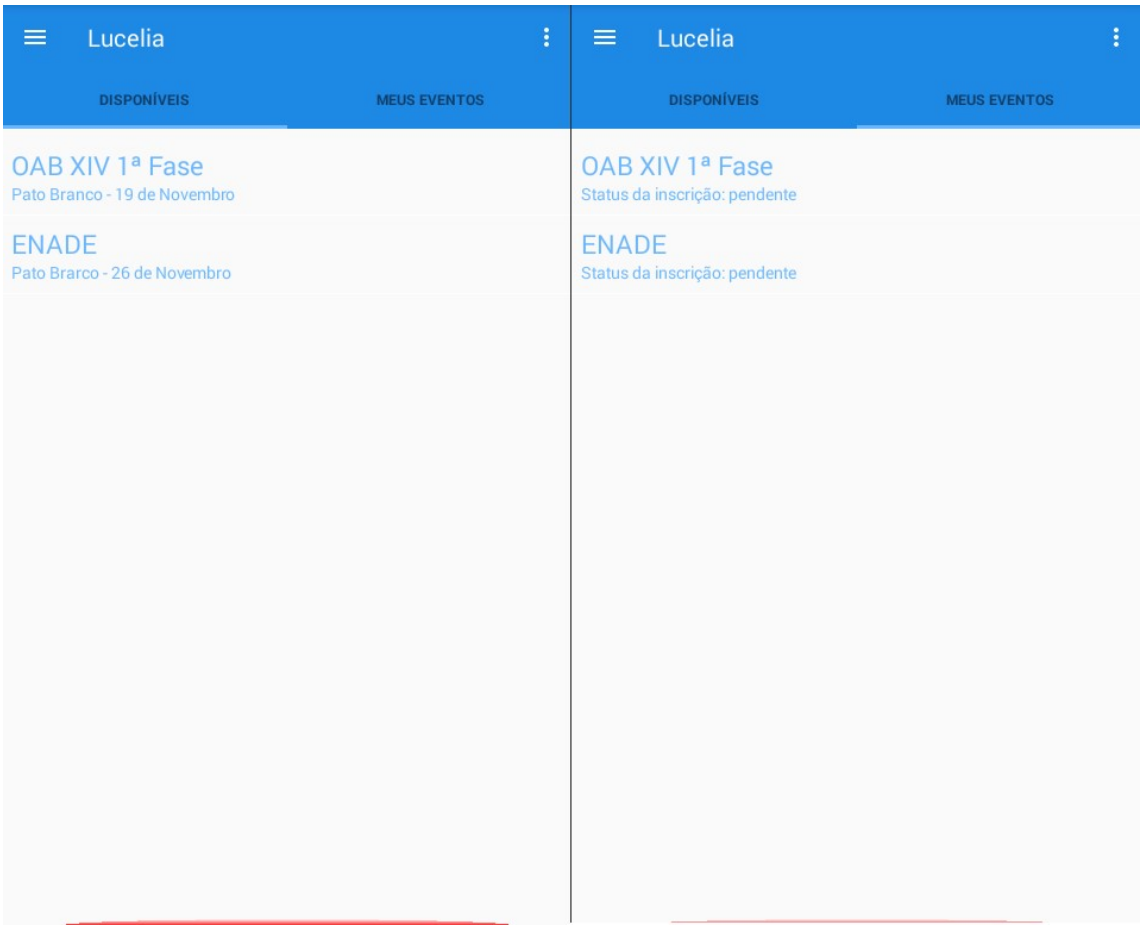


**Figura 12 Eventos e Inscrições**

### 3.3.1 CHAT

Ambos os perfis possuirão acesso ao Chat, recurso que irá funcionar como meio de comunicação entre o coordenador e os inscritos. Este recurso é mostrado na Figura 13.

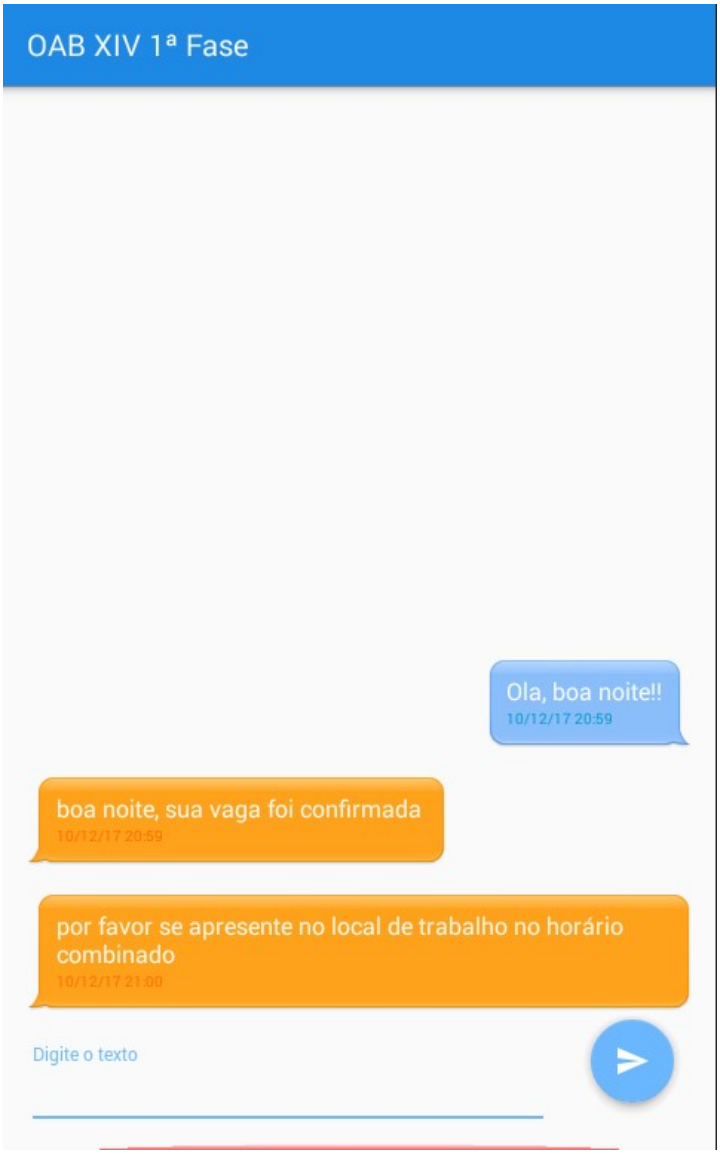


**Figura 13 Chat entre o Coordenador e Inscrito**

O usuário inscrito terá a opção de conversar apenas com o coordenador do evento, não tendo acesso aos demais candidatos inscritos para o evento, por questão de sigilo. Ao coordenador será disponbilizada a opção de conversar com todos os inscritos aprovados. A Figura 14 mostra como é apresentado a listagem de chats ao coordenador.

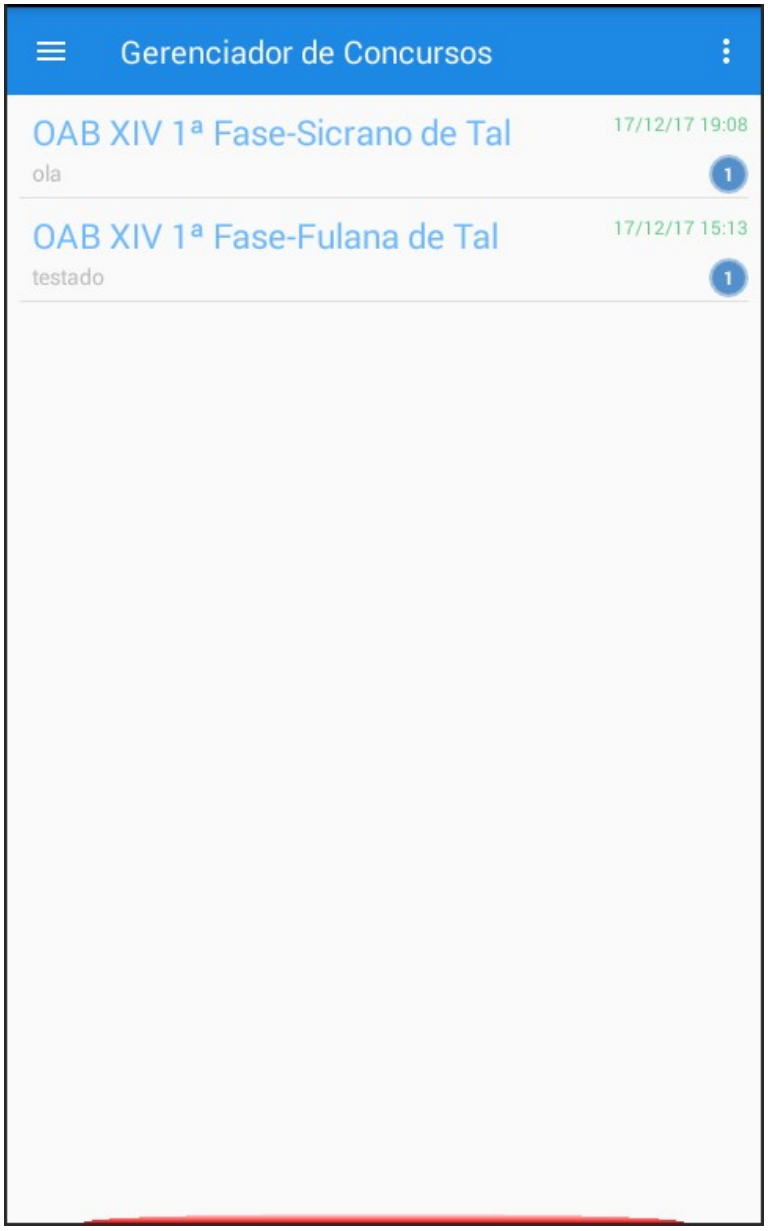


**Figura 14 Listagem de Chats**

## 3.4 ESTRUTURA DO PROJETO

Na Figura 15, pode-se visualizar a estrutuda do projeto e código parcial da classe Main.java.

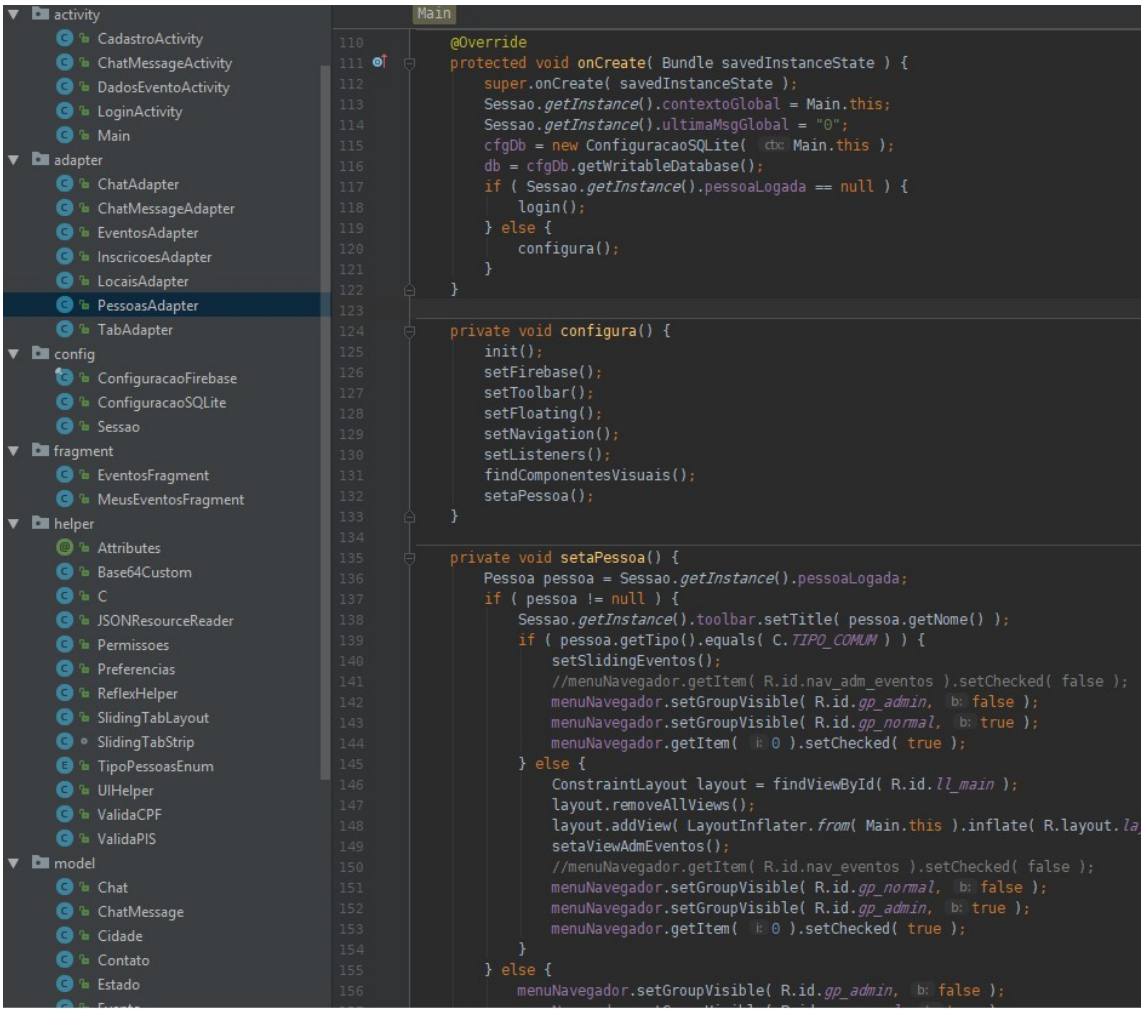


**Figura 15 Estrutura do Projeto no Android Studio**

Um dos pontos chave para o desenvolvimento do app foi a definição do backend, que faria a persistência dos dados e a comunicação entre as partes envolvidas. Inicialmente a ideia seria utilizar um servidor java hospedado no Amazon Web Service (AWS). Porém esta opção apresentaria um custo mensal que não se justificaria diante do uso que seria feito. Nesse momento surgiu a ideia de utilizar a plataforma Firebase, disponibilizada pela Google e que possui um plano sem custo para projetos com baixo consumo de recursos. O maior desafio no uso desta plataforma apresentou-se na utilização do

recurso de banco de dados fornecido, chamado de Firebase Realtime Database. Este recurso trata-se de um banco de dados NoSQL o qual armazena os dados no formato JSON podendo ser sincronizados em tempo real com todos os clientes conectados (Laurence Moroney, 2017). Este formato de banco de dados possui diferenças marcantes em relação aos bancos de dados relacionais tradicionais e apresentou diversos desafios na sua implementação. Notadamente a imposibilidade de realizar consultas mais complexas apresenta um grande desafio quando se deseja velocidade e baixo tráfego de dados.

Na Listagem 1 é apresentado um exemplo de recuperação de dados utilizando o Firebase. Nesse exemplo, foi feita a recuperação das mensagens do chat. Neste código em particular não foi utilizado um indexador, pois optou-se pela não persistência das mensagens localmente.

```
private void carregarMensagens() {
    chat = new ArrayList<>();
    adapterChat = new ChatMessageAdapter( getBaseContext(), chat );
    listView.setAdapter( adapterChat );
    if ( Sessao.getInstance().pessoaLogada.getTipo().equalsIgnoreCase( C.TIPO_COMUM ) ) {
        queryChat = Sessao.getInstance().chatRef.child( Base64Custom.codificarBase64(
Sessao.getInstance().pessoaLogada.getCpf() ) ).child( chatAtual.getEventoId() ).child(
"mensagens" );
        queryChat.addValueEventListener( new ValueEventListener() {
            @Override
            public void onDataChange( DataSnapshot dataSnapshot ) {
                chat.clear();
                for ( DataSnapshot dados : dataSnapshot.getChildren() ) {
                    ChatMessage cm = new ChatMessage();
                    cm.setHorario( dados.getKey().toString() );
                    cm.setTexto( dados.getValue().toString() );
                    chat.add( cm );
                    Preferencias.getInstance().set( "CHATS."
                                    + Base64Custom.codificarBase64(
Sessao.getInstance().pessoaLogada.getCpf() )
                                    + chatAtual.getEventoId()
                                    + C.CHATULTIMAMSG
                            , cm.getHorario() );
                }
                Preferencias.getInstance().set( "CHATS."
                                + Base64Custom.codificarBase64(
Sessao.getInstance().pessoaLogada.getCpf() )
                                + chatAtual.getEventoId()
                                + ".PEND"
                        , "0" );
                adapterChat.notifyDataSetChanged();
            }

            @Override
```

```
            public void onCancelled( DatabaseError databaseError ) {
            }
        } );
        listView.setOnItemClickListener( new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick( AdapterView<?> parent, View view, int position, long id )
{

            }
        } );
    } else {
        queryChat = Sessao.getInstance().chatRef.child( chatAtual.getPessoa() ).child(
chatAtual.getEventoId() ).child( "mensagens" );
        queryChat.addValueEventListener( new ValueEventListener() {
            @Override
            public void onDataChange( DataSnapshot dataSnapshot ) {
                chat.clear();
                for ( DataSnapshot dados : dataSnapshot.getChildren() ) {
                    ChatMessage cm = new ChatMessage();
                    cm.setHorario( dados.getKey().toString() );
                    cm.setTexto( dados.getValue().toString() );
                    chat.add( cm );
                    Preferencias.getInstance().set( "CHATS."
                                    + Base64Custom.codificarBase64(
Sessao.getInstance().pessoaLogada.getCpf() )
                                    + chatAtual.getEventoId()
                                    + C.CHATULTIMAMSG
                            , cm.getHorario() );
                }
                Preferencias.getInstance().set( "CHATS."
                                + Base64Custom.codificarBase64(
Sessao.getInstance().pessoaLogada.getCpf() )
                                + chatAtual.getEventoId()
                                + ".PEND"
                        , "0" );
                adapterChat.notifyDataSetChanged();
            }

            @Override
            public void onCancelled( DatabaseError databaseError ) {
            }
        } );
        listView.setOnItemClickListener( new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick( AdapterView<?> parent, View view, int position, long id )
{

            }
        } );
```

```
    }
}
```

**Listagem 1 –ChatMessageActivity.java**

Algumas tabelas, porém, poderiam trafegar grandes quantidades de dados pela rede e, assim sendo, preferiu-se optar pela uma outra abordagem usando-se persistência local e indexadores para otimizar o tráfego. Na Listagem 2 pode-se observar um exemplo de recuperação de dados utilizando-se um indexador. Na Listagem 2 é mostrado um trecho parcial do código responsável pela recuperação dos cadastros de pessoas, o qual fica disponível para o coordenador do evento para consultas.

```
queryAdmPessoas = Sessao.getInstance().pessoasRef.orderByChild( "timeStamp"
).startAt( ultimoTimeStamp );
queryAdmPessoas.addValueEventListener( new ValueEventListener() {
    @Override
    public void onDataChange( DataSnapshot dataSnapshot ) {
        for ( DataSnapshot dados : dataSnapshot.getChildren() ) {
            int existente = - 1;
            Pessoa pessoa = dados.getValue( Pessoa.class );
            for ( Pessoa p : admPessoas ) {
                if ( pessoa.getCpf().equals( p.getCpf() ) ) {
                    existente = admPessoas.indexOf( p );
                }
            }
            ContentValues values = new ContentValues();
            values.put( "nome", pessoa.getNome() );
            values.put( "tipo", pessoa.getTipo() );
        <......>
            if ( existente > -1 ) {
                admPessoas.set( existente, pessoa );
                db.update( "pessoa", values, "cpf=?", new String[]{
pessoa.getCpf() } );
            }
            else {
                admPessoas.add( pessoa );
                db.insertWithOnConflict( "pessoa", null, values,
SQLiteDatabase.CONFLICT_IGNORE );
            }
```

```
        }
        adapterAdmPessoas.notifyDataSetChanged();
    }
```

**Listagem 2 – Main.java**

## 4 CONSIDERAÇÕES FINAIS

O objetivo deste trabalho foi demonstrar o desenvolvimento de um projeto destinado a auxiliar na coordenação dos envolvidos em eventos, como concursos, processos seletivos e outros. A linguagem escolhida para o projeto foi Java com foco em aplicativos móveis.

Utilizou-se de vários recursos disponibilizados pela plataforma Firebase e concluí-se que este possui um excelente custo benefício para desenvolvedores que não necessitam ou não podem investir em uma estrutura mais avançada e de maior custo.

As dificuldades encontradas foram nas mudanças de paradigma da programação *destktop* para a programação para dispositivos móveis, em especial o *design* de telas e a utilização de banco de dados NoSQL.

De modo geral o proposto como objetivo neste projeto pode ser alcançado, visto que a maior necessidade apresentada para as pessoas envolvidas nos eventos é a existência de um canal mais prático e dinâmico de comunicação que acabou se concretizando.

Por fim, diversas melhorias e implementações ainda podem ser desenvolvidas no app, bem como ampliar os testes para garantir maior compatibilidade com os diversos aparelhos disponíveis no mercado.

**REFERÊNCIAS**

MORONEY, Laurence. **The Firebase Realtime Database**. Apress, Berkeley, CA 2017.

NEWMAN, Chris. SQLite Developer's Library. Sams Indianapolis, IN, USA 2004.

UOL. **Uso de smartphones cresce 3,5 vezes no Brasil em quatro anos**, 2017. Disponível em: <https://tecnologia.uol.com.br/noticias/redacao/2017/02/28/uso-de-smartphones-cresce-35-vezes-no-brasil.htm>. Acesso em: 27 nov 2017.

MÜLLER, Leonardo. **Há exatamente 44 anos, Motorola demonstrava primeiro celular da História**, 2017. Disponível em : <https://www.tecmundo.com.br/motorola/115512-ha-exatamente-44-anos-motorola-demonstrava-primeiro-celular-historia.htm>. Acesso em: 01 dez 2017.

OLHAR DIGITAL, Redação. **Primeiro celular com Android completa cinco anos. Relembre a história**, 2013**.** Disponível em:<https://olhardigital.com.br/noticia/lancamento-do-primeiro-celular-com-android-completa-cinco-anos-relembre/37760>. Acesso em: 26 nov 2017.

ANEXOS

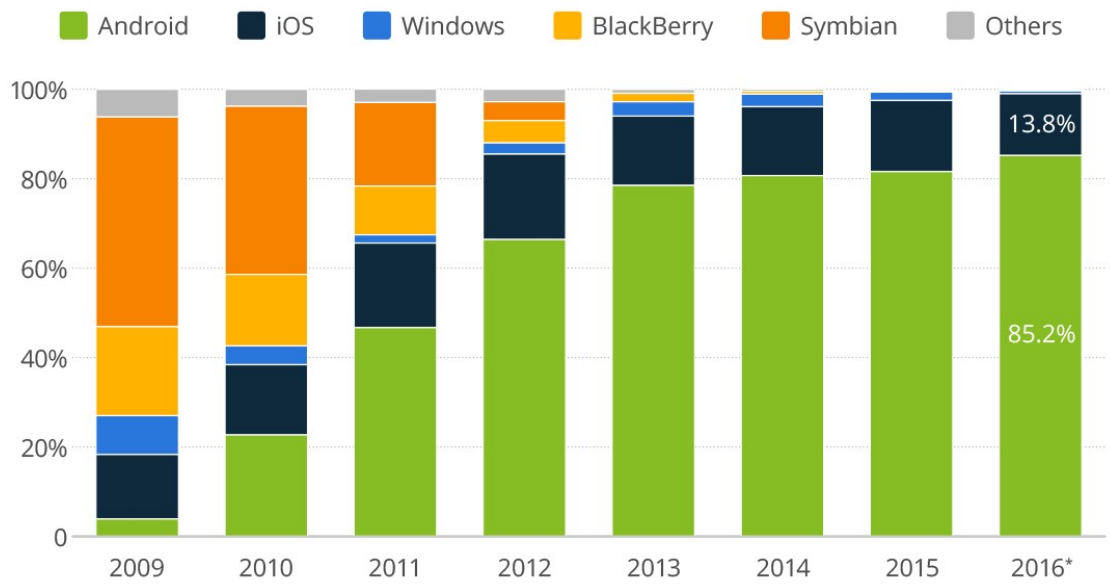


**Figura 16 Porcentagem de uso por Sistema Operacional. Fonte: Gartner Group, Janeiro 2017**