

**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CURSO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

JULIO CESAR MOTA DA SILVA

**APLICATIVO DE CHAT PARA APRENDIZADO DE IDIOMAS VINCULADO AO
FACEBOOK PARA DISPOSITIVOS ANDROID**

TRABALHO DE CONCLUSÃO DE CURSO

**PATO BRANCO
2015**

JULIO CESAR MOTA DA SILVA

**APLICATIVO DE CHAT PARA APRENDIZADO DE IDIOMAS VINCULADO AO
FACEBOOK PARA DISPOSITIVOS ANDROID**

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Diplomação, do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Tecnólogo.

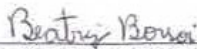
Orientador: Profa. Beatriz Terezinha Borsoi

**PATO BRANCO
2015**

ATA Nº: 269

DEFESA PÚBLICA DO TRABALHO DE DIPLOMAÇÃO DO ALUNO JULIO CESAR MOTA DA SILVA.

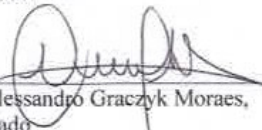
Às 17:00 hrs do dia 25 de junho de 2015, Bloco V da UTFPR, Câmpus Pato Branco, reuniu-se a banca avaliadora composta pelos professores Beatriz Terezinha Borsoi (Orientadora), Adriano Serckumecka (Convidado) e Alessandro Graczyk Moraes (Convidado), para avaliar o Trabalho de Diplomação do aluno Julio Cesar Mota da Silva, matrícula 1066862, sob o título **Aplicativo de Chat para aprendizagem de idiomas vinculados ao Facebook para dispositivos Android**; como requisito final para a conclusão da disciplina Trabalho de Diplomação do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, COADS. Após a apresentação o candidato foi entrevistado pela banca examinadora, e a palavra foi aberta ao público. Em seguida, a banca reuniu-se para deliberar considerando o trabalho **APROVADO**. Às 18:00 hrs foi encerrada a sessão.




Prof. Beatriz Terezinha Borsoi, Dr.
Orientadora



Prof. Adriano Serckumecka, M.Sc.
Convidado



Prof. Alessandro Graczyk Moraes,
Convidado



Prof. Soelaine Rodrigues Ascari, M.Sc.
Coordenador do Trabalho de Diplomação



Prof. Edilson Pontarolo, Dr.
Coordenador do Curso

RESUMO

SILVA, Julio César Mota da. Aplicativo de chat para aprendizado de idiomas vinculado ao Facebook para dispositivos Android. 2015. 53 f. Monografia de Trabalho de Conclusão de Curso - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

A Internet como uma rede mundial de computadores tem facilitado a globalização do comércio e da cultura. Em termos de comércio, no sentido que a compra e a venda de bens e serviços não mais estarem restringidas por fronteiras físicas e/ou necessidade de deslocamento físico do comprador ao vendedor ou vice-versa. Em termos de cultura, os vínculos com pessoas das mais diferentes regiões do planeta trazem a necessidade de uso e mesmo de desenvolvimento de tecnologias, procedimentos e outros que visem facilitar a comunicação. A necessidade de aprender outros idiomas torna-se cada vez mais evidente na sociedade atual. E isso ocorre pelos mais diversos motivos que vão da interação entre pesquisadores, realização de comércio ao entretenimento. As redes sociais, a exemplo do Facebook, têm como uma de suas finalidades promoverem interação entre as pessoas e com o uso da Internet o seu alcance é global. Uma forma de aprender outros idiomas é por meio da conversação e sendo o Facebook uma rede social tão ampla, torna-se um ambiente propício à interação com pessoas dos mais diversos idiomas. Considerando a abrangência do Facebook e dos recursos oferecidos, verificou-se a possibilidade de desenvolver um aplicativo que pudesse ser utilizado a partir de dispositivos com Android para a prática de conversação em idiomas. O aplicativo permite selecionar pessoas a partir de interesses comuns facilitando, assim, a conversação.

Palavras-chave: Aplicativos móveis. Chat para aprendizado de idiomas. Aplicativos para Android.

ABSTRACT

SILVA, Julio César Mota da. Chat application for language learning linked to Facebook for Android devices. 2015. 53 f. Monografia de Trabalho de Conclusão de Curso - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

The internet as a worldwide computer network has facilitated the globalization of commerce and culture. In terms of trade, in the sense that the sale and purchase of goods and services are no longer constrained by physical boundaries and/or need for physical displacement of the buyer to the seller or vice versa. In terms of culture, the bond with the people of the remotest regions of the globe brings the need for ways to facilitate communication (dialogue) between people. The need to learn other languages becomes increasingly evident in today's society. This occurs for several reasons ranging from interaction between researchers, conducting commerce to entertainment. Social networks, like Facebook, have as one of its purposes to promote interaction between people and with the use of Internet their scope is global. One way to learn other languages is through conversation and being the Facebook a social network so wide, it becomes an environment conducive to interaction with people from many different languages. Considering the scope of Facebook and resources offered, there was the possibility of developing an application that could be used from Android devices for the practice of conversation in languages. The application allows you to select people from common interests, thus facilitating the conversation.

Palavras-chave: Mobile applications. Chat. Android applications.

LISTA DE FIGURAS

Figura 1 – Mercado de <i>smartphones</i>	12
Figura 2 – Dados do mercado de sistemas operacionais para <i>smartphones</i>	12
Figura 3 – Aplicações móveis nativas, baseadas em web e híbridas.....	17
Figura 4 – Modelos de padrão de projetos MVC e PAC-TG.....	19
Figura 5 - Tela para procurar tutor aleatório.....	24
Figura 6 - Tela para seleção de um tutor específico	24
Figura 7 - Busuu	25
Figura 8 - Diagrama de entidades e relacionamentos do banco de dados	27
Figura 9 - Tela de login	30
Figura 10 - Permissão Facebook	31
Figura 11 - Seleção de idioma	32
Figura 12 - Idioma inserido é carregado	33
Figura 13 - Exclusão de idioma	34
Figura 14 - Cadastro de interesses	35
Figura 15- Tela para procurar novo amigo.....	36
Figura 16 - Conversa com amigo.....	36
Figura 17 - Interesses comuns	37
Figura 18 - Lista de amigos.....	38
Figura 19 - Tela de gerenciamento de usuários do servidor Openfire.....	38
Figura 20 - Métodos Web Service	41

LISTA DE QUADROS

Quadro 1 – Graph API Root Nodes	21
Quadro 2 – Ferramentas e tecnologias utilizadas	23
Quadro 3 - Requisitos funcionais	27
Quadro 4 - Campos da tabela Usuarios	28
Quadro 5 - Campos da tabela Idiomas	28
Quadro 6 - Campos da tabela Interesses	28
Quadro 7 - Campos da tabela IdiomasUsuarios	29
Quadro 8 - Campos da tabela UsuariosInteresses	29
Quadro 9 - Campos da tabela Amizades	29

LISTAGEM DE CÓDIGO

Listagem 1 - Criação de usuário no servidor Openfire	39
Listagem 2 - Envio e recebimento de mensagens	40
Listagem 3 - Criação de usuário no web service	42
Listagem 4 - Layout do ListView	42
Listagem 5 - Adapter do ListView de idiomas	44
Listagem 6 - Arquivo build.gradle do Android.....	45
Listagem 7 - Chamada da GraphAPI do Facebook.....	46
Listagem 8 - Classe de conexão dos bancos de dados do Web Service.....	47
Listagem 9 - Método para buscar nova conversa do Web Service.....	48

LISTA DE TERMOS ESTRANGEIROS

Adapter – Um objeto que age como ponte entre um *AdapterView* e os dados subjacentes para a referida *view*.

EditText – componente de texto editável do Android.

Flag – Mecanismo lógico que armazena um valor verdadeiro ou falso (booleano).

ListView – componente que exibe uma lista de itens com rolagem do Android.

Request – Uma chamada a um *Web Service*.

Spinner – componente de menu suspenso do Android.

Thread – É uma divisão de processos de um programa. Nessa divisão, um processo se divide em um conjunto de duas ou mais tarefas que podem ser executadas simultaneamente.

Token – segmento de texto ou símbolo que pode ser manipulado por um analisador sintático, que fornece um significado ao texto.

LISTA DE ABREVIATURAS E SIGLAS

API	<i>Application Programming Interface</i>
CDMA	<i>Code Division Multiple Access</i>
CRM	<i>Customer Relationship Management</i>
CSS	<i>Cascading Style Sheets</i>
SDK	<i>Software Development Kit</i>
GPS	<i>Global Position System</i>
GSM	<i>Global System for Mobile communications</i>
HTML	<i>Hypertext Markup Language</i>
MVC	<i>Model-View-Controller</i>
PAC-TG	<i>Presentation–Abstraction–Controller Transformer/Gateway</i>
TI	Tecnologia de Informação
URL	<i>Uniform Resource Locators</i>

SUMÁRIO

1 INTRODUÇÃO.....	11
1.1 CONSIDERAÇÕES INICIAIS	11
1.2 OBJETIVOS.....	13
1.2.1 Objetivo Geral.....	14
1.2.2 Objetivos Específicos.....	14
1.3 JUSTIFICATIVA	14
1.4 ESTRUTURA DO TRABALHO	15
2 DESENVOLVIMENTO DE APLICAÇÕES PARA DISPOSITIVOS MÓVEIS E REDES SOCIAIS.....	16
2.1 DISPOSITIVOS MÓVEIS	16
2.2 REDES SOCIAIS	19
2.1.1 Facebook.....	20
3 MATERIAIS E MÉTODO	22
3.1 MATERIAIS.....	22
3.2 MÉTODO	23
4 RESULTADO	26
4.1 ESCOPO DO SISTEMA.....	26
4.2 MODELAGEM DO SISTEMA.....	26
4.3 APRESENTAÇÃO DO SISTEMA	30
4.4 IMPLEMENTAÇÃO DO SISTEMA	38
5 CONCLUSÃO.....	49
REFERÊNCIAS.....	51

1 INTRODUÇÃO

Este capítulo apresenta as considerações iniciais, os objetivos e a justificativa da realização deste trabalho. No final do capítulo é apresentada a organização do texto por meio de uma breve apresentação dos seus capítulos.

1.1 CONSIDERAÇÕES INICIAIS

Em 2013, o número de celulares alcançou 6.8 bilhões representando 96% do total da população (GLOBAL ICT DEVELOPMENTS, 2014; ABRIL, 2014), se considerado um aparelho por habitante. Com a possibilidade de os aplicativos *web* serem mais responsivos, as páginas *web* tornaram-se flexíveis o suficiente para trabalhar adequadamente com uma ampla variedade de resoluções que podem ser controladas e ajustadas à tela, tornando-se adequadamente visíveis em aparelhos celulares, *tablets* ou *desktops* (KARADIMCE; BOGATINOSKA, 2014). Considerando que atualmente os *smartphones* estão se tornando mais rápidos, com telas maiores e processadores múltiplos, eles também estão permitindo realizar atividades costumeiramente destinadas aos computadores pessoais seja no campo da educação, negócio ou doméstico (NAGESH; CAICEDO, 2012).

Alguns dos dispositivos móveis (*smartphones* e *tablets*, por exemplo) mais populares incluem (PAVLIC; PAVLIC; JOVANOVIĆ, 2012): iPhone, iPad, Eee Pad, Blueberry, Slate PC. Esses mesmos autores destacam que no campo dos *smartphones* quatro sistemas operacionais são atualmente dominantes no mercado: Symbian (Nokia), Android (Google), iOS (Apple) e Windows Mobile (Microsoft).

De acordo com estudo realizado pelo Gartner Group (2011), o sistema operacional que dominará em 2015 é o Android, com praticamente 50% do mercado. A Figura 1 apresenta o histórico e a tendência para o ano de 2015 para os quatro sistemas operacionais citados.

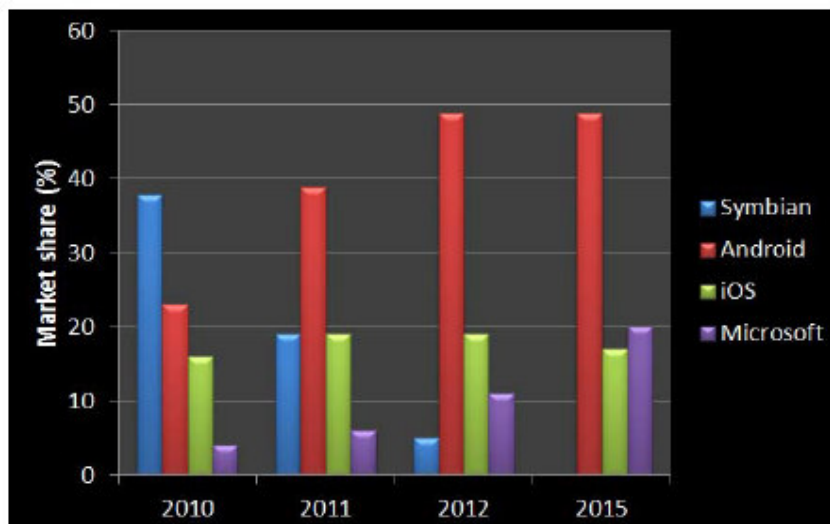


Figura 1 – Mercado de *smartphones*
 Fonte: Gartner (2011, p. 1).

Dados da IDC divulgados em HAMANN (2015) indicam que no final de 2014 o sistema operacional Android alcançou 84,7% do mercado de sistemas operacionais para *smartphones*. Esses dados podem ser visualizados por meio do gráfico apresentado na Figura 2.

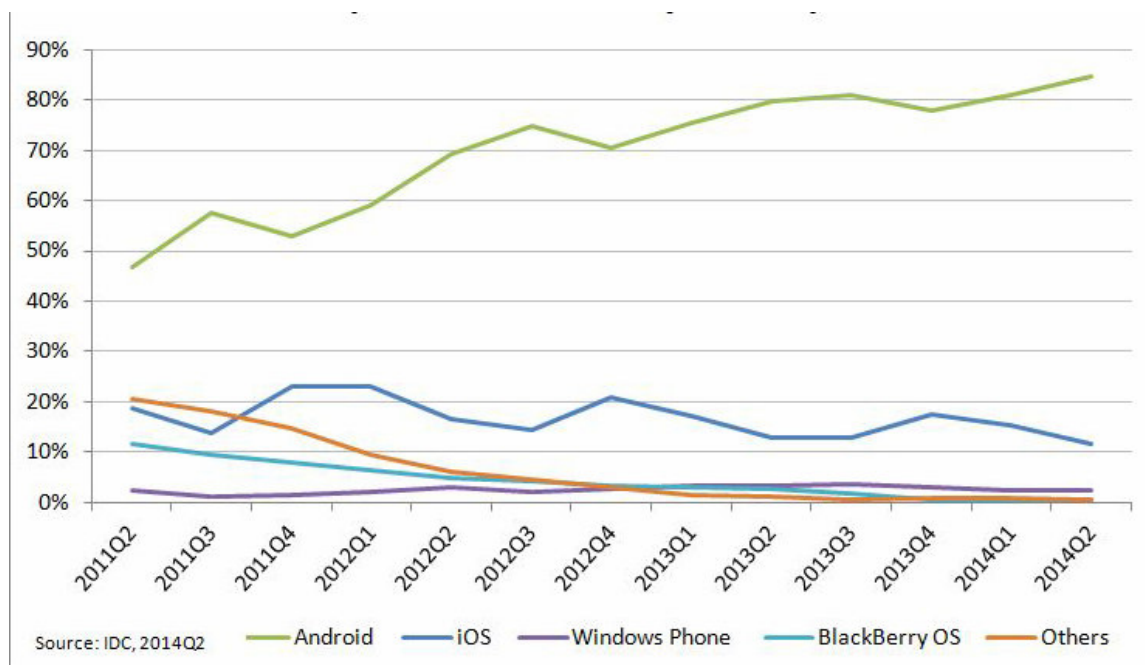


Figura 2 – Dados do mercado de sistemas operacionais para *smartphones*
 Fonte: Hamann (2015, p. 1).

As tecnologias móveis tais como os *smartphones* estão definindo uma nova geração de consumidores e de aplicações de negócio (TENG; HELPS, 2010). Esses autores destacam que a partir da *web*, com uso de *email* para acesso remoto a *Customer Relationship Management* (CRM), não há limite para o que pode ser realizado com dispositivos móveis.

Para os profissionais de Tecnologia de Informação (TI) esse nível sem precedentes de mobilidade é tanto uma oportunidade como um desafio (TENG; HELPS, 2010). A adoção de tecnologias móveis pode ser um processo complexo, não somente por causa do grande número de tipos de dispositivos e de diversos sistemas operacionais existentes (como, por exemplo, BlackBerry OS, Windows Mobile, Symbian OS e iPhone OS), mas, também, porque envolve diversos sistemas *backend* e uma variedade de tecnologias sem fio, como, *Code Division Multiple Access* (CDMA), *Global System for Mobile communications* (GSM), Wi-Fi, WiMax, 3G e 4G networks.

Juntamente com as tecnologias de comunicação, representada pela Internet e amplamente consolidada pelo uso de dispositivos móveis, está a globalização de idiomas. Pessoas de línguas distintas com a possibilidade de trocar ideias. Arelado a isso está a necessidade de conhecer idiomas estrangeiros. Não é mais suficiente, para o mundo dos negócios, expressar-se apenas no idioma nativo. As empresas realizam transações comerciais internacionais, os sistemas computacionais são desenvolvidos visando internacionalização, as oportunidades de estudo e trabalho ignorando, atualmente, as fronteiras físicas entre países.

Visando contribuir para a proficiência em línguas estrangeiras por meio da conversação e utilizando o potencial da Internet e dos dispositivos móveis, associados à rede social Facebook verificou-se a oportunidade de desenvolver um aplicativo móvel para auxiliar na identificação de pessoas nativas de outros idiomas para conversação.

1.2 OBJETIVOS

O objetivo geral está relacionado ao resultado principal que é esperado da realização deste trabalho. E os objetivos específicos complementam o objetivo geral em termos de funcionalidades do sistema.

1.2.1 Objetivo Geral

Implementar um aplicativo computacional para promover conversas visando aperfeiçoamento de idiomas.

1.2.2 Objetivos Específicos

Facilitar, por meio de um aplicativo para dispositivos móveis, que pessoas aprendendo algum idioma tenham contato com falantes nativos ou fluentes, praticando, dessa maneira a conversação.

Desenvolver um aplicativo na plataforma Android que vinculado ao Facebook auxilie no aprendizado de línguas estrangeiras por meio de conversação.

1.3 JUSTIFICATIVA

Uma das dificuldades encontradas ao aprender um idioma é ter uma prática de fala ou escrita constante. Visando auxiliar no aprendizado de conversação de um idioma, o sistema desenvolvido permitirá conversas com usuários com gostos parecidos e que tem domínio do idioma escolhido. Facilitando, assim, a interação. Nesse tipo de conversa há também um aumento do vocabulário, já que o usuário não irá se prender a um assunto apenas.

A plataforma *mobile* foi escolhida, pois hoje as pessoas passam mais tempo utilizando *smartphones* do que vendo TV. Em 2012, pesquisa da Meta Análise indicava que as pessoas passavam mais tempo na Internet do que assistindo televisão (METAANALISE, 2012) e pesquisa da UOL (UOL TECNOLOGIA, 2015) indica que é maior o número dos que usam *tablets* em detrimento de assistir televisão. E, além dessa predominância dos dispositivos móveis, a escolha de Android decorre de o mesmo ser o sistema operacional *mobile* presente na maioria dos *smartphones*.

Para fazer o cadastro dos usuários será utilizado o Facebook. Esse aplicativo já permite a integração do perfil dos usuários com aplicativos Android. As vantagens de utilizar essa rede social é que o cadastro do usuário na aplicação é rápido e evita perfis falsos. O

Facebook também permite acesso às opções curtir, fazendo com que seja possível filtrar usuários com gostos semelhantes.

1.4 ESTRUTURA DO TRABALHO

Este texto está organizado em capítulos. O Capítulo 2 apresenta o referencial teórico. No Capítulo 3 são apresentados os materiais e o método utilizados para o desenvolvimento do trabalho. No Capítulo 4 está o resultado da realização do trabalho e no Capítulo 5 a conclusão.

2 DESENVOLVIMENTO DE APLICAÇÕES PARA DISPOSITIVOS MÓVEIS E REDES SOCIAIS

Este capítulo apresenta o referencial teórico que fundamenta conceitualmente a proposta deste trabalho que se refere a um aplicativo móvel para conversação vinculado ao Facebook.

2.1 DISPOSITIVOS MÓVEIS

Dispositivos móveis, dentre os quais estão os *smartphones* e os *tablets*, estão se tornando parte essencial da vida das pessoas como a mais efetiva e conveniente ferramenta de comunicação não delimitada pelo tempo e pelo espaço (DEV; BAISHNAB, 2014). Usuários de dispositivos móveis acumulam experiências ricas de serviços disponibilizados por diversas aplicações móveis, que executam nos dispositivos e/ou em servidores remotos por meio de redes sem fio. O progresso rápido da computação móvel torna-se uma importante tendência tecnológica no desenvolvimento de Tecnologia de Informação (TI) bem como nos campos de comércio eletrônico e da indústria (SATYANARAYANAN, 2010). Porém, os dispositivos móveis estão enfrentando muitos desafios em termos de recursos (por exemplo: tempo de uso da carga da bateria e sua vida útil, a capacidade de armazenamento do dispositivo e a largura de banda demandada pelos aplicativos) e comunicação (como, por exemplo, mobilidade e segurança) (SATYANARAYANAN, 1996).

Os recursos limitados desses dispositivos impedem significativamente a melhoria da qualidade de serviço (DEV; BAISHNAB, 2014). Esses mesmos autores ressaltam que, por outro lado, avanços no campo de computação baseada em rede e aplicações sob demanda têm levado a um crescimento explosivo de modelos de aplicação, tais como a computação em nuvem, software como serviço, redes de comunidades e lojas *web*, dentre outros.

Aplicações móveis podem ser genericamente classificadas em três categorias (NAGESH; CAICEDO, 2012): aplicações móveis nativas, aplicações móveis baseadas na *web* e aplicações móveis híbridas. Considerando as diferenças entre as três tecnologias, cada abordagem tem suas vantagens e desvantagens (KARADIMCE; BOGATINOSKA, 2014). A Figura 3 apresenta o relacionamento entre essas três categorias de aplicações para dispositivos móveis. Por meio dessa Figura verifica-se que as aplicações móveis híbridas estão entre as aplicações nativas e as aplicações móveis baseadas em *web*.

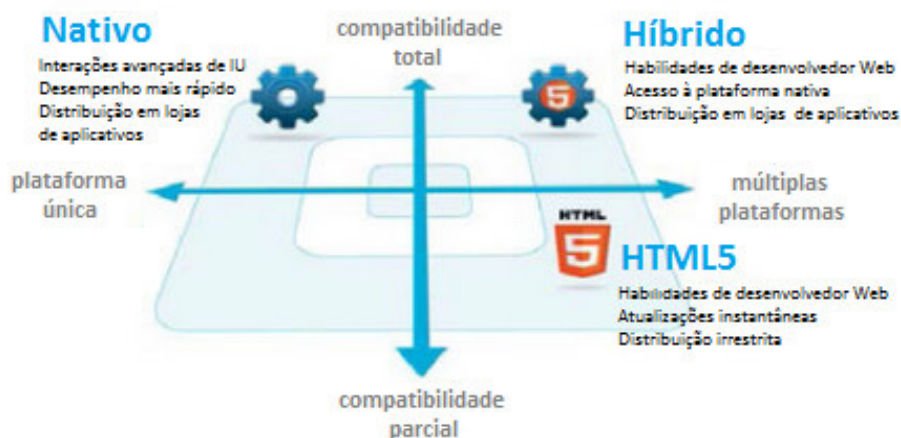


Figura 3 – Aplicações móveis nativas, baseadas em web e híbridas

Fonte: Jern (2013) apud Karadimce e Bogatinoska (2014, p. 688).

As aplicações híbridas combinam o melhor das aplicações nativas e dos recursos *web*. Elas criam uma ponte entre o navegador e as *Applications Programming Interface* (API) dos dispositivos. Assim, as aplicações móveis híbridas usufruem de todas as vantagens das características que os *smartphones* e *tablets* oferecem (KARADIMCE; BOGATINOSKA, 2014).

As aplicações móveis nativas são construídas para um dispositivo móvel específico e seu respectivo sistema operacional. Essas aplicações podem tomar vantagem das novidades e inovações das tecnologias de acessibilidade em dispositivos móveis e pode ser integrada com aplicações *onboard* tais como calendário, contatos e *email* ou *hardware onboard*, tais como, *Global Position System* (GPS) e câmera e aplicativos como *email*, calendário, contatos, galerias de imagens e vídeos, gerenciador de arquivos e áreas de *widjets* na tela inicial (LIONBRIDGE, 2012).

Aplicações móveis nativas são capazes de trabalhar em vários dispositivos, embora versões específicas possam ser necessárias (KARADIMCE; BOGATINOSKA, 2014). Esses autores ressaltam, porém, que, por outro lado, uma aplicação *web* móvel normalmente é baixada de um servidor *web* cada vez que é executada, embora aplicações construídas com *Hypertext Markup Language 5* (HTML 5) podem também ser executadas nos dispositivos móveis de forma *offline* (LIONBRIDGE, 2012).

Kim, Chan e Gupta (2007) ressaltam que os serviços oferecidos pela Internet Móvel podem ser classificados em: comércio, comunicação e conteúdo. Comércio varia do banco móvel ao *ticket* eletrônico para a compra de produtos físicos, enquanto o *e-mail* e os serviços interativos, como bate-papo, são considerados serviços de comunicação. Conteúdo inclui

downloads, notícias, atualizações do trânsito ou das bolsas de valores, bem como outros serviços sensíveis ao tempo ou baseados na localização. Outra classificação é fornecida por Reuver, Ongena e Bouwman (2013) que separam os aplicativos em: entretenimento (*downloads* de jogos e músicas), básicos (*e-mail*, navegação por páginas de conteúdo, mecanismos de busca) e transacionais (reserva e compra de serviços e de produtos).

Para Karadimce e Bogatinoska (2014), o processo de desenvolvimento de aplicações *web* móveis é relativamente barato, fácil e rápido. E esses autores ressaltam que, ao mesmo, customizações no desenvolvimento de aplicativos para dispositivos móveis específicos são, geralmente, essenciais. Os desenvolvedores escrevem em última instância parte do código em HTML, *Cascading Style Sheets* (CSS) e JavaScript e reusam esse código entre dispositivos diferentes ao invés de reescrever a aplicação inteira para cada plataforma móvel.

Aplicações desenvolvidas para plataformas móveis, por exemplo, iPhone, Android ou Blackberry, tem sido foco intenso de negócio, mercado e interesse técnico. Aplicações móveis destinadas para pessoas ou empresas, têm sido focadas na migração de aplicações populares já utilizadas pelos usuários (RADIA *et al.*, 2012).

A ênfase do desenvolvimento de aplicativos baseados em plataformas tem estado no desenvolvimento eficiente de geradores de código para aplicações destinadas a plataformas específicas. Os desenvolvedores de cada plataforma específica têm oferecido um modelo ou padrão de projeto para o desenvolvimento de aplicações que podem ser portadas para determinada plataforma (RADIA *et al.*, 2012).

O modelo ou padrão de projetos *Model-View-Controller* (MVC) tem sido usado para desenvolvimento de aplicações para dispositivos móveis. No MVC, o *model* (modelo) encapsula os dados e a lógica de negócio (código da aplicação) enquanto que o *controll* (controlador) captura as entradas dos usuários e atualiza o modelo, que, então, atualiza a *view* (visão) para apresentar os resultados para o usuário. Uma vez que a visão e o controlador estão fortemente acoplados, por um lado representado a interface para o usuário e por outro a aplicação funcional (o modelo), os desenvolvedores frequentemente encontram dificuldade para as especificações no modelo MVC de forma que a aplicação suporte a multiplicidade de plataformas existentes atendendo a especificação de ser independente de plataforma (RADIA *et al.*, 2012).

O modelo *Presentation-Abstraction-Controller Transformer/Gateway* (PAC-TG) é uma alternativa ao MVC para o desenvolvimento de aplicações móveis independentes de plataforma (RADIA *et al.* 2012). Nesse modelo o controle provê uma ligação (*link*) entre a abstração (modelo) e os componentes de apresentação. Vários transformadores e *transcoders*

suportam customização da interface para determinadas plataformas de destino, enquanto *gateways* de protocolos permitem integração facilitada a vários tipos de interface de comunicação para entradas, saídas e sensores (RADIA et al. 2012). O gerador de código, após a composição da aplicação a partir de um conjunto de blocos de construção (botões, barras, campos de texto), mapeia essa especificação para a execução em uma plataforma móvel (GOOGLE, 2011). A Figura 4 apresenta uma adaptação do modelo PAC-TG juntamente com o MVC, proposta por Radia *et al.* (2012).

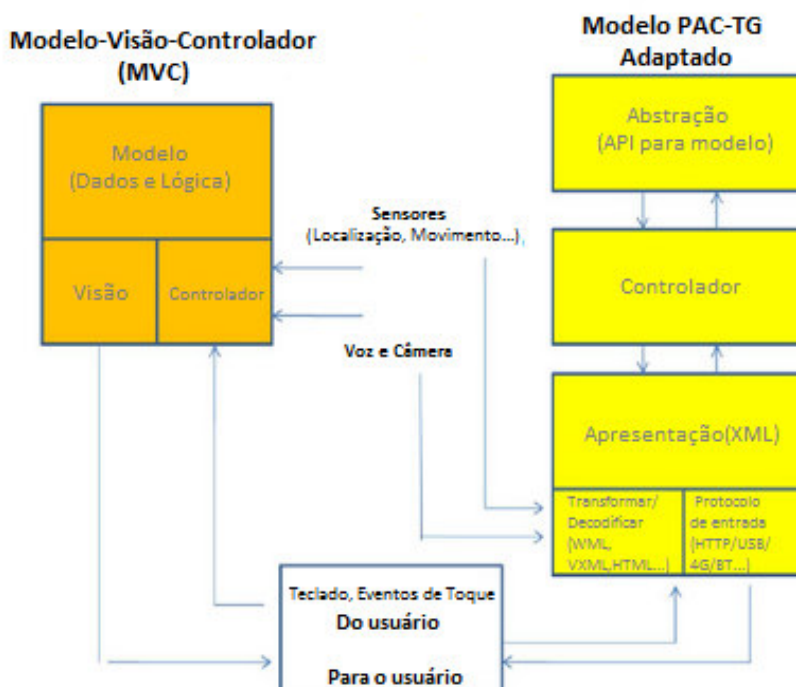


Figura 4 – Modelos de padrão de projetos MVC e PAC-TG

Fonte: Radia et al. (2012, p. 842).

2.2 REDES SOCIAIS

No contexto da Internet, as redes sociais podem ser definidas como locais (sites ou sítios) em que pessoas se conectam umas às outras por meio de laços sociais baseados em afinidades, interesses em comum ou em amizades existentes no mundo real (PANTOJA, 2012). Geralmente, esses interesses são descritos em perfis (*profiles*) que podem ser públicos, dependendo do grau de privacidade escolhido pelo usuário.

As redes sociais são primariamente focadas em interações entre pessoas e possuem um grande potencial de agir como plataformas para o compartilhamento de dados de contexto obtidos por meio de sensores disponíveis nos dispositivos dos usuários, possibilitando assim a criação de sensores sociais de usuários e de seus contatos na rede social (ROSSI *et al.*, 2011). Existem diversas redes sociais, embora algumas poucas sejam de conhecimento e uso muito amplo. Em Lista de Redes Sociais (2015) é possível encontrar uma listagem dessas redes.

A seguir é apresentado sobre a rede social Facebook por ser a utilizada para a execução do aplicativo desenvolvido como resultado deste trabalho.

2.1.1 Facebook

O conteúdo da plataforma Facebook é disponibilizado por meio da Graph API (2015) que permite a leitura e a escrita de dados. O paradigma utilizado é o da orientação a objetos, sendo que objetos são perfis de usuários, amigos, fotos, *posts*, entre outros (PANTOJA, 2012).

A Graph API (2015), disponível em <https://graph.facebook.com> apresenta uma visão do grafo social do Facebook, representando de forma uniforme os objetos e as conexões entre eles (por exemplo, as relações entre amigos, conteúdos compartilhados e *tags* de fotos). Cada objeto no grafo social tem um identificador único e é possível acessar suas propriedades por meio de *Uniform Resource Locators* (URL) bem formadas. O Quadro 1 apresenta os nós principais da Graph API.

Node	Descrição
/achievement	Representa uma pessoa ganhando uma conquista de jogo.
/achievement-type	Um tipo de conquista de jogo.
/album	Um álbum de fotos.
/app	Um aplicativo do Facebook.
/app-link-host	Um objeto de <i>host</i> app link individual criado por um aplicativo.
/comment	Um único comentário.
/conversation	Uma conversa de mensagens do Facebook entre uma pessoa e uma página do Facebook.
/debug_token	Este terminal retorna metadados sobre um determinado token de acesso.
/domain	Um domínio web solicitado por Facebook Insights.
/event	Um evento do Facebook.
/friendlist	O nome de um grupo de amigos criado por uma pessoa no Facebook.
/group	Um grupo do Facebook.
/group-doc	Um documento em um grupo do Facebook.
/link	Um link compartilhado no Facebook.
/message	Uma única mensagem no Facebook Messenger.

/milestone	Um marco para uma página do Facebook.
/notification	Uma notificação individual do Facebook não lida.
/object/comments	Comentários em um objeto.
/object/insights	Informações e métricas para um objeto.
/object/likes	Curtidas para um objeto.
/object/sharedposts	Compartilhamentos deste objeto.
/offer	Uma oferta publicada por uma página do Facebook.
/offsite-pixel	Uma conversão de objeto de pixel.
/order	Um pedido criado por um aplicativo usando pagamentos do Facebook.
/page	Uma Página do Facebook.
/payment	Um único pagamento.
/photo	Uma foto.
/place_tag	Uma instância de uma pessoa sendo marcada em um lugar ou uma foto, vídeo, post, status ou link.
/post	Uma entrada individual em um feed. Feeds podem ser encontrados em usuários, páginas, aplicativos e grupos.
/profile	Um perfil, que pode ser um usuário, página, grupo ou evento.
/request	Uma solicitação de aplicativo recebida por uma pessoa. Uma solicitação pode ser enviada por um aplicativo ou outra pessoa.
/review	Uma análise de um aplicativo do Facebook.
/status	Uma mensagem de status em um feed de um perfil.
/test-user	Um usuário de teste para um aplicativo do Facebook
/thread	Um tópico de mensagem no Facebook Messenger.
/user	Uma pessoa.
/video	Um vídeo.
URL	Compartilhamentos, app links e objetos Open Graph para uma URL.

Quadro 1 – Graph API Root Nodes

Fonte: Facebook Graph API (2015, p. 1).

A Graph API permite o acesso à informação que está disponível em relação a um determinado objeto do Facebook e que é pública. No entanto, para obter informações adicionais ou mesmo enviar um *post* em nome do usuário, é necessário obter um *token* de acesso que deverá ser usado em toda a requisição subsequente (PANTOJA, 2012).

O processo de autenticação e autorização da API do Facebook utiliza o protocolo OAuth 2.0 e envolve três etapas (FACEBOOK, 2015): a) autenticação do usuário: garante que o usuário é quem ele diz ser (*login* no Facebook); b) autorização da aplicação social: garante que o usuário esteja ciente exatamente de quais dados e recursos está disponibilizando para ele; c) autenticação da aplicação: garante que a aplicação está agindo em nome do usuário.

3 MATERIAIS E MÉTODO

Este capítulo apresenta os materiais e o método utilizados para a realização deste trabalho. Os materiais estão relacionados às tecnologias e ferramentas utilizadas e o método apresenta a sequência das principais atividades realizadas.

3.1 MATERIAIS

O Quadro 1 apresenta as ferramentas e as tecnologias que foram utilizadas para modelar e implementar o sistema.

Ferramenta / Tecnologia	Versão	Referência	Finalidade
Java Platform (JDK)	8u40	http://www.oracle.com/	Linguagem de programação
Microsoft Visual Basic 2013	06177-004-0444002-02144	https://www.visualstudio.com/en-us/downloads/download-visual-studio-vs.aspx	Linguagem de programação
Android <i>Software Development Kit</i> (SDK)	24.0.2	http://developer.android.com/sdk	Kit de desenvolvimento de software do Android
Android Studio	1.2.2	http://developer.android.com/sdk	Ambiente de desenvolvimento.
Graph API	v2.2	https://developers.facebook.com/docs/graph-api	API de integração com Facebook
Microsoft Visual Studio Professional 2013	Version 12.0.31101.00 Update 4	https://www.visualstudio.com/en-us/downloads/download-visual-studio-vs.aspx	Ambiente de desenvolvimento
Facebook Android SDK	3.23.0	https://developers.facebook.com/docs/android	Kit de desenvolvimento de software do Facebook
OpenSSL	0.9.8	https://www.openssl.org/	Biblioteca de criptografia

Openfire	3.10.0	http://www.igniterealtime.org/projects/openfire/	Servidor Jabber
Smack	4.1.0	https://www.igniterealtime.org/projects/smack/	Biblioteca XMPP
IIS 8	6.2	https://www.microsoft.com/pt-br/download/details.aspx?id=34679	Servidor Web

Quadro 2 – Ferramentas e tecnologias utilizadas

3.2 MÉTODO

Por tratar-se de um aplicativo com uma quantidade reduzida de requisitos e sem necessidade de armazenamento de muitos dados não houve a necessidade de realizar modelagem ou um planejamento extenso da implementação.

O levantamento dos requisitos foi realizado a partir da análise de dois aplicativos com funcionalidades semelhantes: o Cambly e o Busuu.

O Cambly, disponível em <https://www.cambly.com>, é um aplicativo pago. Por meio desse aplicativo onde é possível manter conversas com tutores de idiomas por meio de chamadas de vídeo. Para iniciar uma nova conversa basta clicar no botão "Pratique Inglês" e com essa ação será solicitado um tutor aleatório ou caso o usuário queira um tutor específico deve-se selecionar a aba "Orientadores". A Figura 5 apresenta a tela para a solicitação de um tutor aleatório.

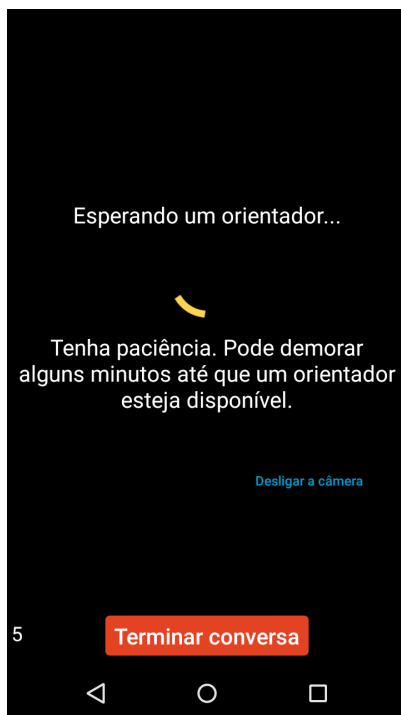


Figura 5 - Tela para procurar tutor aleatório

Na Figura 6 está a tela que é apresentada pelo aplicativo Cambly quando da solicitação de escolha de um tutor específico.

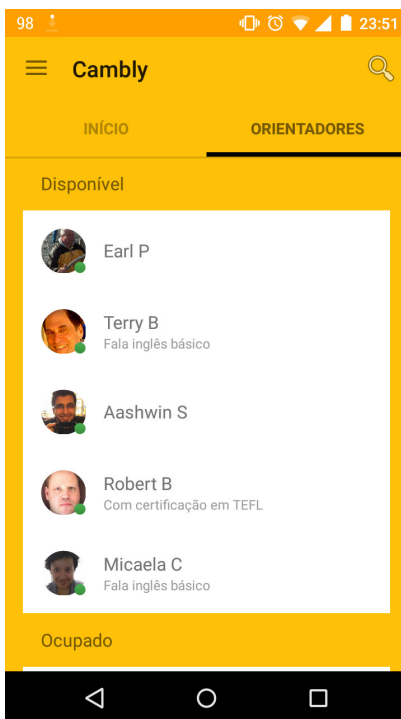


Figura 6 - Tela para seleção de um tutor específico

O Busuu, disponível em Site: <https://www.busuu.com>, é um site e aplicativo que possibilita aos usuários selecionar idiomas que falam, idiomas que gostariam de aprender e procurar amigos para a conversação. A interação com esses amigos ocorre por meio de conversas ou pela correção de exercícios.



The screenshot shows the 'Buscar usuários do busuu' (Search Busuu users) interface. At the top, there is a blue navigation bar with the Busuu logo, a flag icon, a 'Mude para Premium!' button, and user profile, message, and notification icons. Below the navigation bar, the search form is titled 'Buscar usuários do busuu'. It contains several input fields and dropdown menus: 'Nome do usuário:' (empty text input), 'Usuários falando:' (dropdown menu with 'Inglês' selected), 'Usuários aprendendo:' (dropdown menu with 'Português' selected), 'Sexo' (dropdown menu with 'Todos' selected), 'Usuários de:' (dropdown menu with 'Todos' selected), and 'Faixa de idade:' (dropdown menu with '16 - 29' selected). To the right of these fields are two buttons: 'Redefinir' (blue) and 'Procurar' (green). Below the search form, there is a link: 'Convidar seus amigos do facebook a utilizar o busuu'. At the bottom of the page, there is a blue footer bar with copyright information '2008-2015 © Busuu Ltd.', links for 'Baixar aplicações móveis', 'Ajuda', 'Sugestões', and 'Mostrar mais', and social media icons for Twitter, Facebook, VK, Google+, YouTube, and a 'Blog' link.

Figura 7 - Busuu

Pela tela do aplicativo Busuu verifica-se que é possível, além de idioma, indicar outros requisitos para a busca de usuários para como faixa de idade e gênero. Esse aplicativo permite o convite a usuários do Facebook.

A partir da definição dos requisitos, as tabelas do banco de dados foram criadas e o aplicativo foi implementado utilizando as ferramentas e tecnologias indicadas no Quadro 2.

4 RESULTADO

Este capítulo apresenta o resultado deste trabalho que o desenvolvimento de um aplicativo para dispositivos móveis com o objetivo de auxiliar no aprendizado de idiomas.

4.1 ESCOPO DO SISTEMA

O aplicativo desenvolvido é um sistema de *chat* para dispositivos Android com o objetivo de auxiliar no aprendizado de idiomas. O aplicativo permite o usuário logar-se por meio de sua conta no Facebook, cadastrar os idiomas que fala e quais deseja aprimorar ou aprender. Com base nesses idiomas e nos dados do seu perfil, como opções curtir, o sistema sugere uma conversa com usuários que tenham interesses semelhantes (os dados de interesse são obtidos do perfil) e que os idiomas sejam os mesmos, mas visando sempre combinar usuários com língua nativa ou fluente com os que pretendem aprender a referida língua.

Durante uma conversa é possível abandoná-la e procurar outra combinação ou adicionar como amigo, permitindo, nesse caso, que os usuários conversem em outro momento.

4.2 MODELAGEM DO SISTEMA

O Quadro 3 apresenta a listagem dos requisitos funcionais identificados para o aplicativo.

Identificação	Nome	Descrição
RF01	Cadastrar usuário	Um usuário é cadastrado no sistema no momento em que ele faz <i>login</i> no Facebook pela primeira vez.
RF02	Cadastrar idioma fala	Cadastrar idiomas que o usuário domina (fluência) na fala. O grau de fluência (definir se domina) é critério do usuário quem define. O aplicativo não verifica o conhecimento do usuário no referido idioma.
RF03	Cadastrar idioma aprender	Cadastrar idiomas que o usuário deseja aprender (praticar conversação).

RF04	Cadastrar interesse	Cadastrar interesses. Os interesses são os assuntos que o usuário tem interesse em debater ou conversar sobre.
RF05	Procurar conversa	Buscar na base de dados um usuário que tenha pelo menos um idioma em comum e o máximo de interesses iguais.
RF06	Carregar lista de contatos	Carregar lista com os amigos do usuário.
RF07	Enviar mensagem	Enviar mensagem para o usuário com o <i>chat</i> aberto
RF08	Receber mensagem	Receber mensagem do usuário com o <i>chat</i> aberto
RF09	Listar idiomas em comuns	Quando no perfil do amigo, listar todos os idiomas em comum.
RF10	Listar Interesses em comum	Quando no perfil do amigo, listar todos os interesses em comum.

Quadro 3 - Requisitos funcionais

A Figura 8 apresenta o diagrama de entidades e relacionamentos do banco de dados da aplicação. O banco de dados é bastante simples, tendo como entidades: usuários (participantes dos *chats* de conversação), idiomas (que fala ou pretende aprender) e interesses (definem o perfil do usuário). Há, ainda, duas tabelas de relacionamento entre essas tabelas principais. Uma delas é a tabela que relaciona idiomas e usuários e a outra é a tabela que relaciona usuários e interesses.

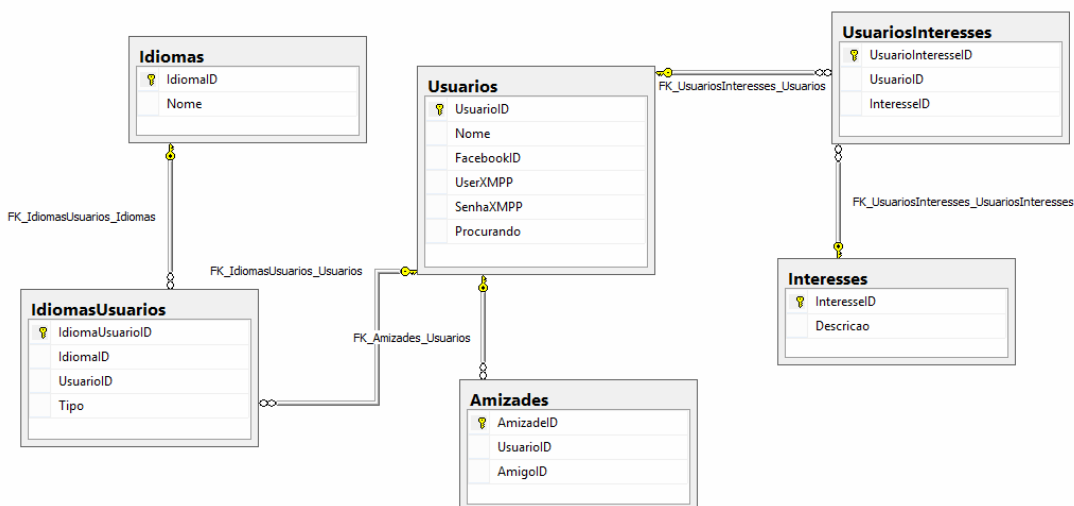


Figura 8 - Diagrama de entidades e relacionamentos do banco de dados

A seguir são descritos os campos das tabelas que compõem a Figura 5. O Quadro 4 apresenta os campos da tabela Usuarios. Usuários são as pessoas que possuem acesso ao sistema.

Campo	Tipo	Nulo	Chave Primária	Chave estrangeira	Observações
UsuarioID	int	Não	Sim	Não	
Nome	nvarchar(50)	Não	Não	Não	
FacebookID	nvarchar(50)	Não	Não	Não	
UserXMPP	nvarchar(50)	Não	Não	Não	
Procurando	tinyint	Não	Não	Não	

Quadro 4 - Campos da tabela Usuarios

O Quadro 5 apresenta os campos da tabela Idiomas. Esses dados se referem aos idiomas cadastrados na base de dados e que serão escolhidos pelo usuário como idioma que domina (fala) e que pretende aprender (praticar).

Campo	Tipo	Nulo	Chave Primária	Chave estrangeira	Observações
IdiomaID	int	Não	Sim	Não	
Nome	nvarchar(50)	Não	Não	Não	

Quadro 5 - Campos da tabela Idiomas

No Quadro 6 estão os campos da tabela interesses. Os interesses compõem o perfil do usuário e auxiliam na indicação de outros usuários com os quais ele poderia manter conversação. Por exemplo: um usuário indica futebol como interesse e indica inglês como idioma fluente; outro usuário indica que quer aprender inglês e também futebol como interesse. O sistema sugerirá conversa (*chat*) entre essas duas pessoas. O primeiro filtro de pesquisa é o idioma de fluência e de interesse de aprendizado. A pesquisa é refinada pelos interesses que é um segundo filtro. Não havendo interesses comuns é tomado como base apenas o idioma.

Campo	Tipo	Nulo	Chave Primária	Chave estrangeira	Observações
InteresseID	int	Não	Sim	Não	
Descricao	nvarchar(50)	Não	Não	Não	

Quadro 6 - Campos da tabela Interesses

Os campos da tabela de IdiomasUsuarios (Quadro 7) representam um idioma falado por um usuário. O campo tipo representa o tipo de idioma que o usuário tem fluência: 1 representa idioma que fala e 2 representa o idioma que quer aprender.

Campo	Tipo	Nulo	Chave Primária	Chave estrangeira	Observações
IdiomaUsuarioID	int	Não	Sim	Não	
IdiomaID	int	Não	Não	Sim	Da tabela Idiomas
UsuarioID	int	Não	Não	Sim	Da tabela Usuarios
Tipo	tinyint	Não	Não	Não	

Quadro 7 - Campos da tabela IdiomasUsuarios

O Quadro 8 apresenta os campos da tabela UsuariosInteresses. Os interesses são utilizados para buscar (filtrar) usuários para conversação.

Campo	Tipo	Nulo	Chave Primária	Chave estrangeira	Observações
UsuarioInteresseID	int	Não	Sim	Não	
UsuarioID	int	Não	Não	Sim	Da tabela Usuarios
InteresseID	int	Não	Não	Sim	Da tabela Interesses

Quadro 8 - Campos da tabela UsuariosInteresses

No Quadro 9 estão os campos da tabela Amizades. Essa tabela contém o usuário e o amigo ao qual esse usuário está vinculado. Sempre que uma amizade é cadastrada dois registros são inseridos. Cada usuário do sistema deve ter o seu próprio registro da amizade.

Campo	Tipo	Nulo	Chave Primária	Chave estrangeira	Observações
AmizadeID	int	Não	Sim	Não	
UsuarioID	int	Não	Não	Sim	Da tabela Usuarios
AmigoID	int	Não	Não	Não	

Quadro 9 - Campos da tabela Amizades

4.3 APRESENTAÇÃO DO SISTEMA

Ao iniciar o aplicativo apresenta a tela da Figura 9 que tem o botão do Facebook para fazer *login*. Dessa maneira, o usuário vinculará sua conta ao aplicativo.

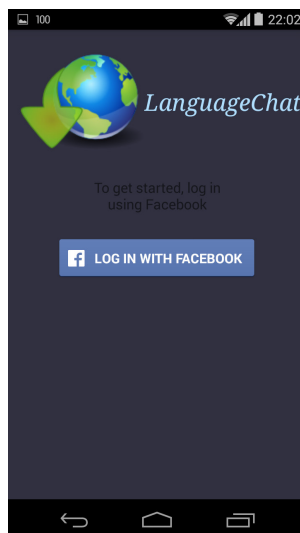


Figura 9 - Tela de login

Caso o usuário já tenha o Facebook instalado no *smartphone* será apresentada a tela que permitirá que o aplicativo use as informações do Facebook, como mostra na Figura 10. Caso contrário, será apresentada a tela para que o usuário digite seu *login* e senha da rede social.



Figura 10 - Permissão Facebook

Assim que o *login* é realizado o usuário é redirecionado para a página inicial (Figura 11), que contém sua foto de perfil e nome. Essas duas informações são buscadas da sua conta do Facebook e são apresentadas as opções de adicionar os idiomas que fala e quer aprender. Abaixo de cada uma dessas opções é mostrada uma listagem de idiomas para o usuário cadastrar seus interesses. Ao lado do nome está um ícone de uma lupa (região circulada na Figura 11). Esse ícone é um botão que redireciona para a tela de procurar uma nova conversa. Há, ainda, um ícone que representa esquematicamente de três pessoas (região circulada no canto superior direito da Figura 11), ao clicar nesse botão o usuário é redirecionado para a sua lista de amigos.

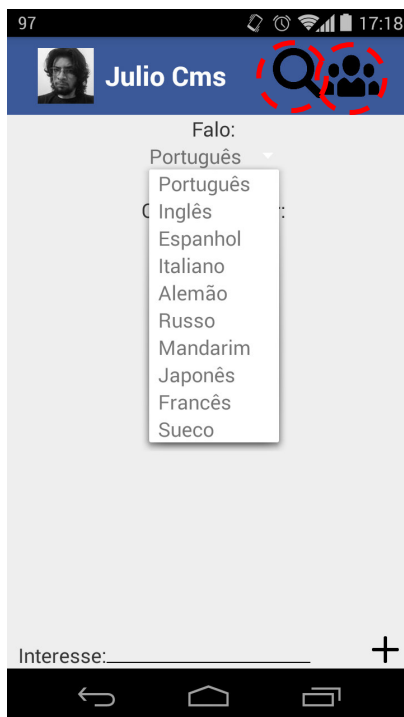


Figura 11 - Seleção de idioma

Assim que o usuário clicar nos símbolos “+” ele poderá adicionar um idioma que fala ou que pretende aprender. No primeiro clique é carregado um *spinner* com os idiomas cadastrados na base como mostra na Figura 4. Assim que o botão é novamente clicado o idioma é vinculado ao usuário e carregado em um *ListView* personalizado com bandeira e nome do idioma. O ícone de uma lixeira também é apresentado, sinalizando que basta clicar para excluí-lo, como mostrado na Figura 12. Também é apresentada uma mensagem informando que a adição foi realizada com sucesso.

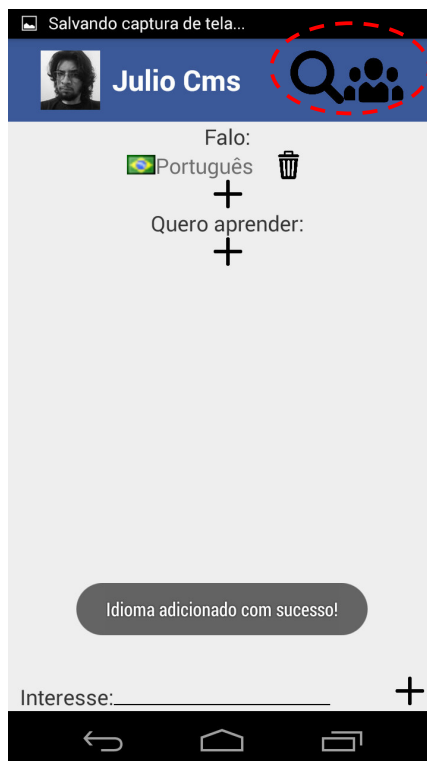


Figura 12 - Idioma inserido é carregado

Ao clicar em algum dos itens que compõem o *ListView* é apresentada uma mensagem questionando se o usuário deseja excluir o idioma selecionado. Clicando em “Sim” é realizada a exclusão e o *ListView* é recarregado com a atualização da exclusão realizada. Caso a escolha seja por cancelar, o sistema permanece no estado anterior à realização da ação de exclusão. A Figura 13 apresenta a tela de mensagem questionando sobre a confirmação ou cancelamento da exclusão.

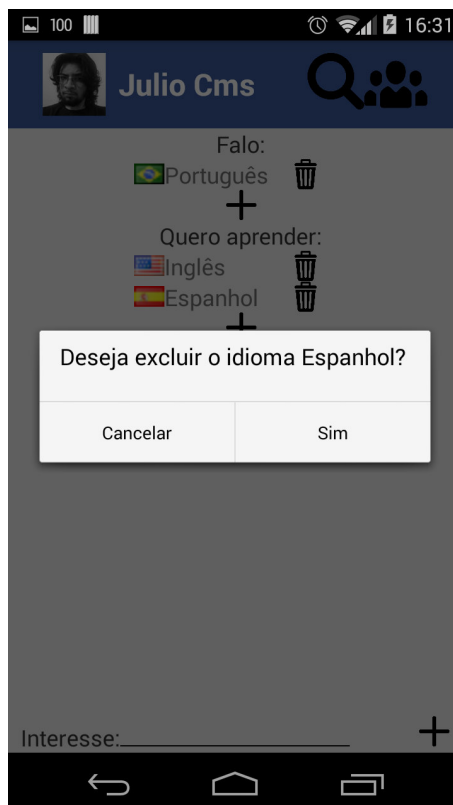


Figura 13 - Exclusão de idioma

Na Figura 14 é possível visualizar que ao selecionar o *EditText* de interesse, o usuário tem a opção de escrever o que deseja que seja utilizado como filtro para procurar interesses semelhantes. Assim que o usuário clicar no símbolo de “+” ao lado, o interesse é adicionado na base e vinculado ao usuário. Caso o referido interesse já tenha sido cadastrado (pelo próprio usuário ou outro) é apenas realizado o vínculo com o interesse já cadastrado. Em seguida é apresentada uma mensagem informando da adição do interesse informado no banco de dados.

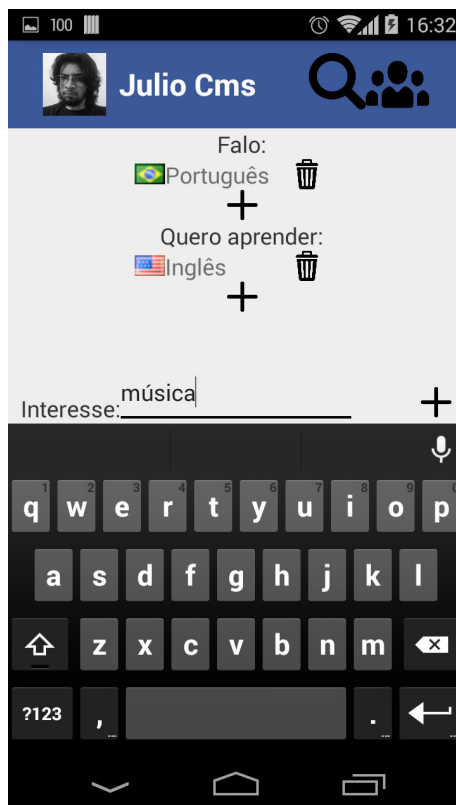


Figura 14 - Cadastro de interesses

Quando o usuário clicar no botão com formato de lupa é realizado o redirecionamento para a tela apresentada na Figura 15. Nessa tela é apresentada uma mensagem “Clique para procurar uma nova conversa”. Assim que clicado na lupa, o sistema faz uma busca de outro usuário que possua o relacionamento definido de idioma e que possua interesses em comum e, ainda, que também esteja procurando uma conversa no momento. Caso a busca não tenha resultado (não sejam encontrados registros que atendem aos critérios de pesquisa) é mostrada uma mensagem informado para que procure novamente. Caso a busca tenha resultado, o usuário é redirecionado para a tela apresentada da Figura 12 para que possa iniciar uma conversa.



Figura 15- Tela para procurar novo amigo

Como mostra na Figura 16, é possível enviar mensagens ao novo usuário. As mensagens sendo trocadas (enviadas e recebidas) são apresentadas na tela à medida que elas são enviadas pelos dois usuários participando da conversação.

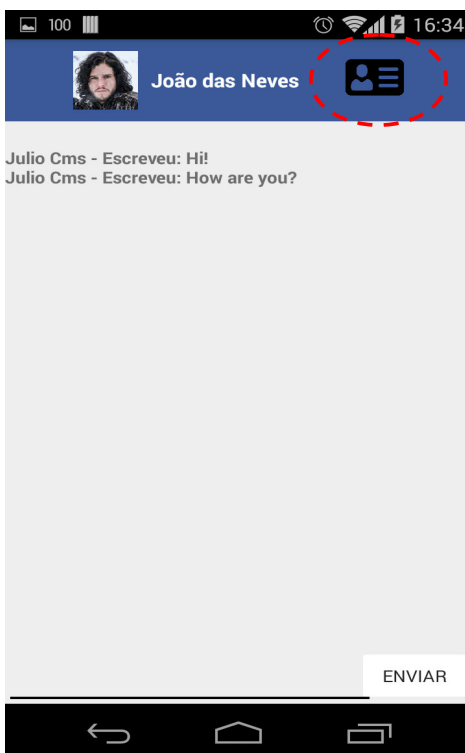


Figura 16 - Conversa com amigo

Na tela apresentada na Figura 17 há um ícone que se assemelha a um cartão de visitas. Ao clicar nesse ícone o usuário é redirecionado para uma página com a foto e nome do amigo. Logo abaixo é possível visualizar em dois *ListView* os idiomas e os interesses comuns, como mostra na Figura 18.

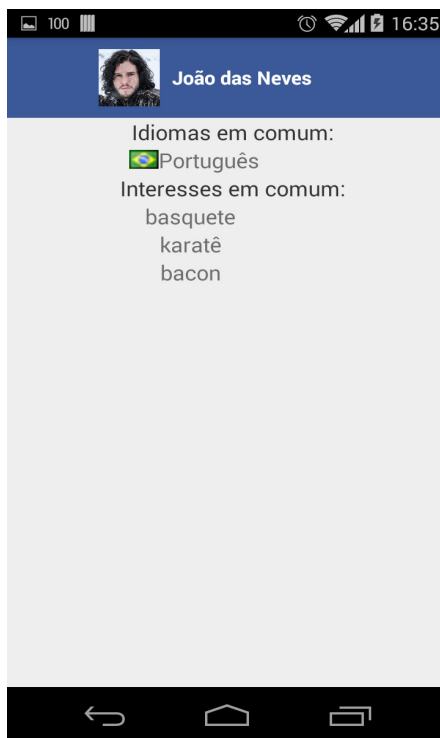


Figura 17 - Interesses comuns

Caso o usuário volte à tela inicial, por meio do próprio botão de voltar do Android, e selecione o ícone de lista de amigos, o sistema o redireciona para a tela mostrada na Figura 18. Nessa tela são apresentados todos os amigos que o usuário tem em um *ListView* personalizado que contém a foto e o nome. Ao clicar em qualquer um dos itens o usuário é redirecionado para uma conversa com o amigo selecionado, a mesma tela da Figura 16.

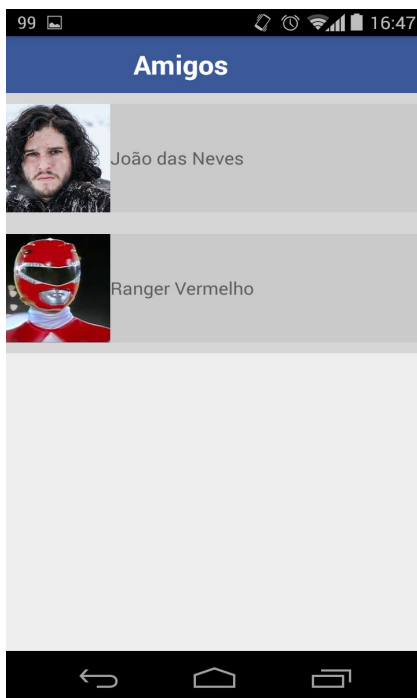


Figura 18 - Lista de amigos

4.4 IMPLEMENTAÇÃO DO SISTEMA

Para fazer envio de mensagens foi utilizado um servidor Jabber, que gerencia usuários e mensagens. Na Figura 19, é possível ver a tela de gerenciamento, mais especificamente na aba “Users”, na qual são listados os usuários cadastrados. Esses usuários são cadastrados pelo aplicativo assim que o usuário faz *login* pelo seu Facebook.

openfire

Openfire 3.10.0
Logged in as admin - [Logout](#)

Server **Users/Groups** Sessions Group Chat Plugins

Users Groups

User Summary
Create New User
User Search
Pesquisa de Usuários

User Summary

Total Users: 5 -- Sorted by Username -- Users per page: 100

Online	Username	Name	Groups	Created	Last Logout	Edit	Delete
	109277229411013		None	Jun 19, 2015			
	1405452566451804		None	Jun 20, 2015	1 hour, 3 minutes		
	794017887301866		None	Jun 13, 2015	1 hour, 3 minutes		
	admin	JulioCMS	None	May 23, 2015	1 day, 8 hours, 38 minutes		
	teste	teste	None	May 23, 2015			

Server | Users/Groups | Sessions | Group Chat | Plugins

Built by [Jive Software](#) and the [IgniteRealtime.org](#) community

Figura 19 - Tela de gerenciamento de usuários do servidor Openfire

Dentro do sistema Android foi criada a classe SmackCon. Essa classe faz o gerenciamento de conexão da aplicação com o servidor Openfire. Na Listagem 1 está o código para a criação de um novo usuário na aplicação. Esse método recebe como parâmetro um usuário e senha, cria a configuração de acesso ao servidor usando um objeto `XMPPTCPConnectionConfiguration`, instância um objeto `AbstractXMPPConnection` passando para ele a configuração criada e o objeto se conecta ao servidor, em seguida é declarado uma `AccountManager` passando o servidor. Esse objeto permite que um novo usuário seja criado. Após usar seu método `createAccount` com os parâmetros é criado o novo usuário no servidor, retornando um booleano (valor `true`) caso consiga.

```
public boolean CriarUsuarioSmack(String sUser, String sSenha){
    XMPPTCPConnectionConfiguration.Builder config =
    XMPPTCPConnectionConfiguration.builder();

    config.setSecurityMode(ConnectionConfiguration.SecurityMode.disabled)
    ;
    config.setServiceName("languagechat");
    config.setHost(ip);
    config.setPort(5222);

    AbstractXMPPConnection con = new
    XMPPTCPConnection(config.build());

    try{
        con.connect();
        AccountManager accountManager =
    AccountManager.getInstance(con);
        accountManager.sensitiveOperationOverInsecureConnection(true);
        accountManager.createAccount(sUser, sSenha);
        return true;
    } catch (IOException e){
        con = null;
        return false;
    } catch (SmackException e){
        con = null;
        return false;
    } catch (XMPPException e){
        con = null;
        return false;
    }
}
```

Listagem 1 - Criação de usuário no servidor Openfire

Para enviar e receber uma mensagem é necessário conectar ao servidor. Na Listagem 2 está o código de como isso é feito. Esse método também está na classe SmackCon. Após a

conexão é declarado um `ChatManager` e um `Chat`, ambos objetos da biblioteca `Smack`. O `ChatMessageListener` é responsável por receber mensagens e o objeto `Chat` consegue enviá-las.

```

public String EnviarMensagem(String sUser, String sSenha, String
sMensagem, String sUserSend){
    StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder().permitAll().build();
    StrictMode.setThreadPolicy(policy);
    XMPPConnectionConfiguration.Builder config =
XMPPConnectionConfiguration.builder();
    config.setUsernameAndPassword(sUser, sSenha);
    config.setServiceName("languagechat");
    config.setHost(ip);
    config.setPort(5222);

    AbstractXMPPConnection con = new
XMPPConnection(config.build());
    try{
        con.connect();
        con.login();

        ChatManager chatManager = ChatManager.getInstanceFor(con);
        Chat newchat = chatManager.createChat(sUserSend +
"@languagechat", new ChatMessageListener() {
            @Override
            public void processMessage(Chat chat, Message message) {
                System.out.println("Received message: " + message);
            }
        });

        try {
            System.out.println("Message sent: " + sMensagem);
            newchat.sendMessage(sMensagem);
        }
        catch (Exception e) {
            System.out.println("Error Delivering block");
        }

    } catch (IOException e){
        con = null;
        return "";
    } catch (SmackException e){
        con = null;
        return "";
    } catch (XMPPException e){
        con = null;
        return "";
    } finally {
        return sRetorno ;
    }
}

```

Listagem 2 - Envio e recebimento de mensagens

Para realizar operações de banco de dados é utilizado um *Web Service* que contém todas as operações necessárias. Esse *web service* é publicado em um servidor IIS. Na Figura 17 é possível verificar todos os métodos criados.

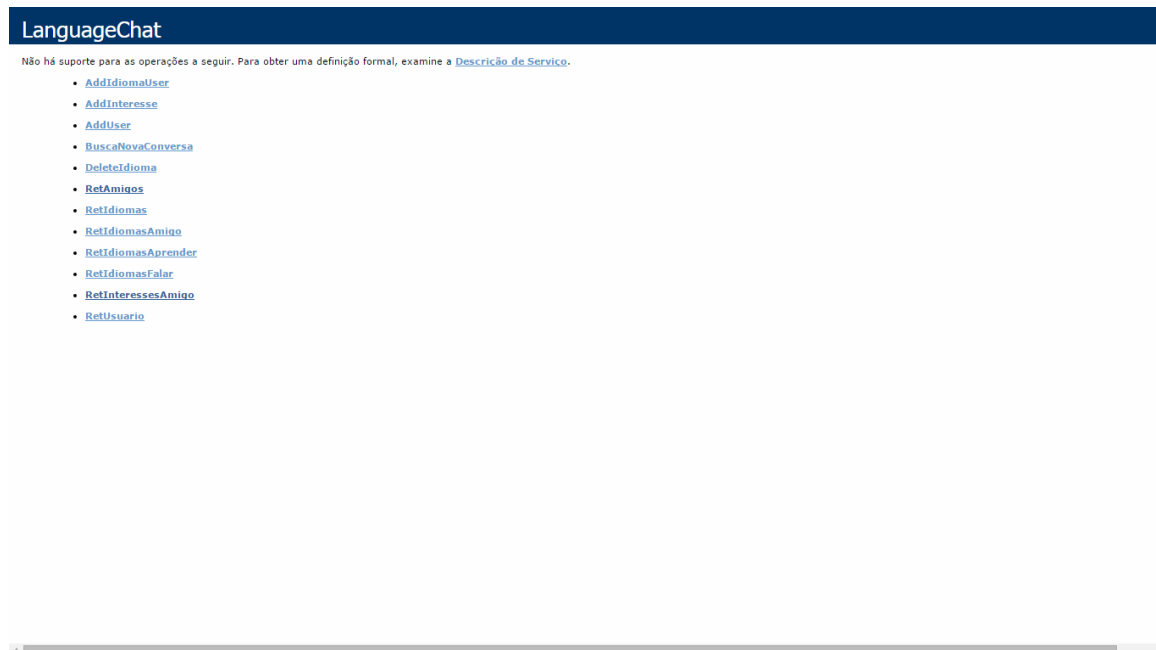


Figura 20 - Métodos Web Service

Para conectar neste *Web Service* é utilizada a classe *WSCon* criada no Android (código na Listagem 3). Neste caso, a classe tem a função *CriaUsuario*, que recebe como parâmetro o id do Facebook e o Nome. Utilizando a biblioteca *KSOAP2* é feito um *request* do método no *Web Service*, logo após o resultado que é um XML é passado para o objeto *Usuário*. Nesse caso são retornadas as informações *UsuarioID*, *Nome*, *FacebookID*, *UserXMPP* e *SenhaXMPP*. Esse objeto retornado é o utilizado para criação do usuário no servidor *Openfire*, pois esse método do *Web Service* é quem cria um usuário e senha para serem utilizados. Essa informação já fica armazenada no servidor.

```
public Usuario CriaUsuario(String sFacebookID, String sNome) throws
IOException, XmlPullParserException{
    String methodname = "AddUser";
    String SOAP_ACTION = NAMESPACE + methodname;
    SoapObject request = new SoapObject(NAMESPACE, methodname);
    Usuario usuario = new Usuario();
    request.addProperty("sFacebookID", sFacebookID);
    request.addProperty("sNome", sNome);

    SoapSerializationEnvelope envelope =
getSoapSerializationEnvelope(request);
    HttpTransportSE ht = getHttpTransportSE();
```

```

    ht.call(SOAP_ACTION, envelope);
    SoapObject resultsString = (SoapObject)envelope.getResponse();

    for(int i = 0; i < resultsString.getPropertyCount(); i++){
        SoapObject objSoap = (SoapObject)resultsString.getProperty(i);
        usuario.UsuarioID =
Integer.parseInt(objSoap.getProperty(0).toString());
        usuario.Nome = objSoap.getProperty(1).toString();
        usuario.FacebookID = objSoap.getProperty(2).toString();
        usuario.UserXMPP = objSoap.getProperty(3).toString();
        usuario.SenhaXMPP = objSoap.getProperty(4).toString();
    }

    return usuario;
}

```

Listagem 3 - Criação de usuário no web service

Dentro da aplicação existem componentes *ListView* que são personalizados. Um exemplo desse componente é o que carrega os idiomas do usuário. Para fazer isso é necessário criar um leiaute específico para o componente, como apresentado na listagem 4. Para popular esse componente é necessário setar um *adapter*.

```

<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:gravity="center_horizontal">

    <ImageView
        android:id="@+id/imgidioma"
        android:layout_width="48px"
        android:layout_height="48px"
        android:layout_gravity="center"/>

    <TextView
        android:id="@+id/nomeidioma"
        android:layout_width="200px"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:textSize="18sp"
    />

    <ImageView
        android:id="@+id/imgdelete"
        android:layout_width="48px"
        android:layout_height="48px" />

</LinearLayout>

```

Listagem 4 - Layout do ListView

O *adapter* é uma classe, como mostrado na Listagem 5, que recebe um *Arraylist* de objetos *IdiomaFalar*. Por meio desse *Arraylist* os componentes são preenchidos, após isso basta setar esse *adapter* no *ListView*.

```
public class IdiomaFalarAdapter extends BaseAdapter {
    private Context context;
    private ArrayList<IdiomaFalar> idiomas;

    public IdiomaFalarAdapter(Context context, ArrayList<IdiomaFalar>
idiomas) {
        this.context = context;
        this.idiomas = idiomas;
    }

    @Override
    public long getItemId(int position) {
        IdiomaFalar idioma = idiomas.get(position);
        return idioma.IdiomaID;
    }

    @Override
    public Object getItem(int position) {
        return idiomas.get(position);
    }

    @Override
    public int getCount() {
        return idiomas.size();
    }

    @Override
    public View getView(int position, View convertView, ViewGroup
parent) {
        LayoutInflater inflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        View v = convertView;
        if(v == null){
            v = inflater.inflate(R.layout.idiomas_detalhes,
parent, false);
        }
        IdiomaFalar idioma = idiomas.get(position);

        TextView textNome = (TextView)
v.findViewById(R.id.nomeidioma);
        textNome.setText(idioma.Nome);

        ImageView img = (ImageView) v.findViewById(R.id.imgidioma);
        switch (idioma.IdiomaID){
            case 1:
                img.setImageResource(R.drawable.br);
                break;
            case 2:
                img.setImageResource(R.drawable.us);
                break;
        }
    }
}
```

```

        case 3:
            img.setImageResource(R.drawable.es);
            break;
        case 4:
            img.setImageResource(R.drawable.it);
            break;
        case 5:
            img.setImageResource(R.drawable.de);
            break;
        case 6:
            img.setImageResource(R.drawable.ru);
            break;
        case 7:
            img.setImageResource(R.drawable.cn);
            break;
        case 8:
            img.setImageResource(R.drawable.jp);
            break;
        case 9:
            img.setImageResource(R.drawable.fr);
            break;
        case 10:
            img.setImageResource(R.drawable.se);
            break;
        default:
            break;
    }

    ImageView imgDelete = (ImageView)
v.findViewById(R.id.imgdelete);
    imgDelete.setImageResource(R.drawable.delete);

    return v;
}
}

```

Listagem 5 - Adapter do ListView de idiomas

Para o desenvolvimento foram utilizadas bibliotecas externas. A Listagem 6 mostra o arquivo build.gradle do Android. Por meio desse arquivo é possível adicionar essas bibliotecas automaticamente, bastando apenas defini-las no arquivo.

```

apply plugin: 'com.android.application'

android {
    compileSdkVersion 21
    buildToolsVersion "21.1.2"

    defaultConfig {
        applicationId "com.julio.languagechat"
        minSdkVersion 14
        targetSdkVersion 21
        versionCode 1
    }
}

```

```

        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-
android.txt'), 'proguard-rules.pro'
        }
    }
}

repositories {
    maven {
        url 'https://oss.sonatype.org/content/repositories/snapshots'
    }
    maven {
        url 'http://ksoap2-android.googlecode.com/svn/m2-repo'
    }
    mavenCentral()
}

dependencies {
    compile 'com.android.support:appcompat-v7:22.0.0'
    compile 'com.facebook.android:facebook-android-sdk:3.23.0'

    compile "org.igniterealtime.smack:smack-android:4.1.0"
    // Optional for XMPPTCPConnection
    compile "org.igniterealtime.smack:smack-tcp:4.1.0"
    // Optional for XMPP-IM (RFC 6121) support (Roster, Threaded
Chats, ...)
    compile "org.igniterealtime.smack:smack-im:4.1.0"
    // Optional for XMPP extensions support
    compile "org.igniterealtime.smack:smack-extensions:4.1.0"
    //KSOAP
    compile 'com.google.code.ksoap2-android:ksoap2-android:3.4.0'
}

```

Listagem 6 - Arquivo build.gradle do Android.

Para fazer a busca de dados do usuário no Facebook é chamada a Graph API. Na Listagem 7 está um exemplo de como é realizada essa chamada.

```

private void makeMeRequest(final Session session) {
    // Make an API call to get user data and define a
    // new callback to handle the response.

    Request request = Request.newMeRequest(session,
        new Request.GraphUserCallback() {
            @Override
            public void onCompleted(GraphUser user, Response
response) {
                // If the response is successful
                if (session == Session.getActiveSession()) {
                    if (user != null) {

```

```

        // Set the id for the ProfilePictureView
        // view that in turn displays the profile
picture.
profilePictureView.setProfileId(user.getId());
        // Set the Textview's text to the user's
name.
        userNameView.setText(user.getName());
    }
}
if (response.getError() != null) {
    // Handle errors, will do so later.
}
});
request.executeAsync();
}

```

Listagem 7 – Chamada da GraphAPI do Facebook

O *Web Service* foi desenvolvido em VB.Net. Ele contém apenas duas classes. A classe *Conexão* que tem os métodos de acesso ao banco de dados e a classe *LanguageChat.aspx.vb* que contém os métodos. Na listagem 8 estão os métodos criados para conexão e para trabalhar com os dados do banco de dados.

```

Public Shared Function Executar(ByVal sSQL As String) As Boolean
    Try
        Dim con As New SqlConnection(getConnectionString())
        con.Open()
        Dim cExecuta As New SqlCommand(sSQL, con)
        cExecuta.ExecuteNonQuery()
        Return True
    Catch ex As Exception
        Throw New Exception(ex.Message)
    End Try
End Function

Public Shared Function RetCampo(ByVal sTabela As String, ByVal sCampo As String,
ByVal sFiltro As String, Optional ByRef transaction As SqlTransaction = Nothing) As
String
    Dim dt As DataTable
    Dim sSQL As String = "Select Top 1 {0} From {1} Where {2}"

    dt = RetDataTable(String.Format(sSQL, sCampo, sTabela, sFiltro), transaction)
    If dt.Rows.Count > 0 Then
        Return dt.Rows(0)(0).ToString()
    Else
        Return ""
    End If
End Function

Public Shared Function RetDataTable(ByVal sSql As String, Optional ByRef
transaction As SqlTransaction = Nothing) As Data.DataTable
    Dim db As SqlDatabase = getDatabase()

    Dim cmd As DbCommand = db.GetSqlCommand(sSql)

```

```

cmd.CommandTimeout = 300

Try
  If transaction Is Nothing Then
    Return db.ExecuteDataSet(cmd).Tables(0)
  Else
    Return db.ExecuteDataSet(cmd, transaction).Tables(0)
  End If
Catch ex As SqlException

  Throw New Exception(ex.Message)

Catch ex As Exception
  Throw ex
End Try
End Function

Private Shared Function getDatabase() As SqlDatabase
  Return
Microsoft.Practices.EnterpriseLibrary.Data.Sql.SqlDatabase(getConnectionString)
End Function

Private Shared Function getConnectionString() As String
  Return
ConfigurationManager.ConnectionStrings("LanguageChatConnection").ConnectionString
End Function

```

Listagem 8 - Classe de conexão dos bancos de dados do Web Service

Um exemplo de método criado no *Web Service* está na Listagem 9. Esse é o método criado para buscar uma nova conversa. É passado o id do usuário como parâmetro. E o método seta o *flag* do usuário no banco para Procurando = 1. Assim é possível saber que o usuário está em busca de uma nova conversa. São realizadas 10 tentativas, com uma *thread* prolongando cada tentativa em 1 segundo, antes de realizar uma nova tentativa é verificado se o *flag* continua como 1. Isso porque um usuário já pode ter encontrado e definido como amigo. Caso ainda não tenha ocorrido essa localização, é realizada a consulta que procura um usuário que tenha idiomas semelhantes e o máximo de interesses iguais possíveis. Se encontrados, é criada uma “amizade” e marcado o usuário encontrado como 0, ou seja, na próxima iteração da chamada do outro usuário ele saberá que já foi encontrado. Caso em alguma iteração seja verificado que um usuário o marcou como amigo, a estrutura de repetição (*for*) é interrompida e é identificada a última amizade feita. Caso um “amigo” tenha sido localizado, o seu id é retornado. Em todos os casos, o usuário é marcado como 0 na *flag* de procurando, para indicar que já não está disponível.


```

<WebMethod()> _
Public Function BuscaNovaConversa(ByVal nUsuarioID As Integer) As XmlDocument
    Dim dt As DataTable
    Dim sRetorno As String = "<usuario><id>0</id></usuario>"

    Dim sSQL As String =
        "Select Top 1 (UsuarioID) From " & _
        "(Select Nome,UsuarioID, " & _
        "(Select          Count(*)          From          UsuariosInteresses          Where
UsuariosInteresses.InteresseID In (Select aux.InteresseID From UsuariosInteresses aux
where UsuarioID = " & nUsuarioID & ")) as CountAtiv " & _
        "From Usuarios " & _
        "Where " & _
        "Procurando = 1 And Usuarios.UsuarioID Not In (Select AmigoID From Amizades
Where UsuarioID = " & nUsuarioID & ") And Usuarios.UsuarioID <> " & nUsuarioID & ")
as Consulta Order By CountAtiv Desc "

    Dim sSQL2 As String = "UPDATE Usuarios Set Procurando = 1 Where UsuarioID = " &
& nUsuarioID
    Conexao.Executar(sSQL2)

    For i = 0 To 10
        Dim lContinua As Boolean = Val(Conexao.RetCampo("Usuarios", "Procurando",
"UsuarioID = " & nUsuarioID))
        If lContinua Then
            dt = Conexao.RetDataTable(sSQL)
            If dt.Rows.Count > 0 Then
                Dim nAmigoID As Integer = 0
                sRetorno = "<usuario>" & vbCrLf
                For Each dr As DataRow In dt.Rows
                    nAmigoID = dr("UsuarioID")
                    sRetorno &= "<id>" & dr("UsuarioID") & "</id>" & vbCrLf
                Next
                sRetorno &= "</usuario>" & vbCrLf
                Dim sSQL3 As String = "INSERT INTO Amizades (UsuarioID, AmigoID)
VALUES (" & nUsuarioID & "," & nAmigoID & ")"
                Conexao.Executar(sSQL3)
                sSQL3 = "UPDATE Usuarios Set Procurando = 0 Where UsuarioID = " &
nAmigoID
                Conexao.Executar(sSQL3)
                Exit For
            Else
                Threading.Thread.Sleep(1000)
            End If
        Else
            Dim nAmigoID As Integer = Val(Conexao.RetCampo("Amizades", "Top 1
(AmigoID)", "Where UsuarioID = " & nUsuarioID & " order by AmizadeID desc"))
            sRetorno = "<usuario>" & vbCrLf
            sRetorno &= "<id>" & nAmigoID & "</id>" & vbCrLf
            sRetorno &= "</usuario>" & vbCrLf
        End If
        i += 1
    Next
    sSQL2 = "UPDATE Usuarios Set Procurando = 0 Where UsuarioID = " & nUsuarioID
    Conexao.Executar(sSQL2)
    Dim oXML As New XmlDocument()
    oXML.LoadXml(sRetorno)
    Return oXML
End Function

```

Listagem 9 - Método para buscar nova conversa do Web Service

5 CONCLUSÃO

O trabalho de conclusão de curso com foco no desenvolvimento de um aplicativo Android foi uma excelente oportunidade de aprendizado, pois permitiu trabalhar com tecnologias com as quais o autor possuía muito pouco conhecimento. Foi uma oportunidade de usar ferramentas e tecnologias diferentes das aplicadas no dia a dia de trabalho que é a VB.Net. Além das linguagens novas, o fato de ser um aplicativo *mobile* requer uma maneira diferente para realizar tarefas das que são realizadas para o desenvolvimento de uma aplicação *web*. Contudo, há um custo em termos de tempo e dificuldade em trabalhar com tecnologias com as quais se tem pouco domínio, como a linguagem Java para Android. Esse custo é compensado porque é transformado em aprendizado.

Para desenvolver o *Web Service* foi utilizado o VB.Net. Apesar do conhecimento nessa tecnologia, o desenvolvimento do aplicativo foi uma oportunidade de criar tudo desde o início, além de ser necessário tratar conexões com banco de dados. No ambiente de trabalho, essas funcionalidades já estão desenvolvidas, sendo necessário apenas implementar algo ou fazer correções. Para fazer a publicação do *Web Service* foi utilizado o IIS. Foi necessário aprender como configurar esse servidor *web* e também como fazer publicações nele.

Para o desenvolvimento direto com o Android foi necessário fazer muitas pesquisas e foram encontradas muitas dificuldades. E alguns vícios de implementação, provenientes de desenvolvimento com outras linguagens, acabaram atrapalhando. Muitas vezes ocorreram erros que tomaram muito tempo resolver. No início a dificuldade para popular componentes no Android foi grande, porém após um tempo foi possível perceber que apesar de trabalhoso existem muitas opções para personalizar esses componentes.

Além dessas linguagens citadas foi utilizado o SQL Server. O uso do SQL Server ocorreu sem muitos problemas. Foi trabalhoso utilizar a biblioteca Smack e o servidor Jabber Openfire. A documentação, principalmente da biblioteca Smack, é um pouco confusa, muitas vezes contendo aspectos que já não eram utilizados na versão mais atualizada. Foi desafiador, mas possível encontrar a maneira correta de uso. Diversas vezes a comunidade *online*, incluindo os fóruns, exerceu um papel crucial na solução das dúvidas.

A utilização do SDK do Facebook foi um pouco mais simples, pois a própria plataforma oferece muitos exemplos e documentação atualizada.

A realização deste trabalho de conclusão de curso foi desafiadora, pois pela primeira vez foi necessário realizar o ciclo completo de desenvolvimento de uma aplicação: dos

requisitos à implantação. Isso exigiu paciência e pesquisa e por isso foi extremamente proveitoso: pelo teste de habilidades e pelo aprendizado.

Como trabalhos futuros, visando complementar as funcionalidades implementadas e melhorar essas funcionalidades, sugere-se a possibilidade de inclusão de mais de um usuário em uma mesma conversa e a possibilidade de realização de chamadas por vídeo. Possibilitando, assim, o treinamento (aprendizagem) da pronúncia no referido idioma. Para complementar as funcionalidades destaca-se o aperfeiçoamento da interface utilizando as diretrizes (*guidelines*) da Google para o desenvolvimento de interface para aplicativos móveis. Atualmente, orientações sobre o desenvolvimento utilizando essas diretrizes podem ser obtidas nos seguintes endereços: <https://developers.google.com/webmasters/mobile-sites/get-started/?hl=en> e <https://support.google.com/webmasters/answer/6101188?hl=pt-BR>.

REFERÊNCIAS

ABRIL. **Mundo terá quase 7 bilhões de celulares em uso até o final de 2014, diz estudo.** Disponível em: <<http://veja.abril.com.br/noticia/vida-digital/mundo-tera-quase-7-bilhoes-de-celulares-em-uso-ate-o-final-de-2014-diz-estudo/>>. Acesso em: 08 fev. 2015.

DEV, Dipayan; BAISHNAB, Krishna Lal. **A review and research towards mobile cloud computing.** In: 2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), 2014, p. 252 – 256.

FACEBOOK GRAPH API. **Facebook Graph API reference.** Disponível em: <<https://developers.facebook.com/docs/reference/api/>>. Acesso em: 11 fev. 2015.

FACEBOOK. Facebook Authentication Guide. Disponível em: <<https://developers.facebook.com/docs/authentication>>. Acesso em: julho de 2011. 3.2.1

GARTNER GROUP. **Gartner says android to command nearly half of worldwide smartphone operating system market by year-end 2012.** 2011. Disponível em: <<http://www.gartner.com/newsroom/id/1622614>>. Acesso em: 08 fev. 2015.

GLOBAL ICT DEVELOPMENTS. **Report on Global mobile-cellular subscriptions by ITU Statistics.** Disponível em: <www.itu.int/ITU-D/ict/statistics/explorer/index.html>. Acesso em: 05 fev. 2015.

GOOGLE APP INVENTOR, 2011. Disponível em: <<http://beta.appinventor.mit.edu/about/>>. Acesso em: 20 fev. 2015.

HAMANN, Renan. **iOS, Android e Windows Phone: números dos gigantes comparados.** Disponível em: <<http://www.tecmundo.com.br/sistema-operacional/60596-ios-android-windows-phone-numeros-gigantes-comparados-infografico.htm>>. Acesso em: 06 abr. 2015.

JERN, Magnus. **How to choose the right technology? Native, HTML5 and more...**, 2013. Disponível em: <http://www.goldengekko.com/?blog_post=mobile-strategy-handbook-chapter-6-how-to-choose-the-right-technology-native-html5-and-more>. Acesso em: 25 fev. 2015.

KARADIMCE, Aleksandar; BOGATINOSKA, Dijana Capeska. **Using hybrid mobile applications for adaptive multimedia content delivery.** 37th International Convention Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014, p. 686 – 691.

KIM, Hee-Woong; CHAN, Hock C.; GUPTA, Sumeet. Value-based Adoption of Mobile Internet: An empirical investigation. **Decision Support Systems**, v.43, n.1, p.11-126, 2007.

LIONBRIDGE, **Mobile Web Apps vs Mobile Native Apps: How to Make the Right Choice**, 2012. Disponível em: <http://www.lionbridge.com/files/2012/11/Lionbridge-WP_MobileApps2.pdf>. Acesso em: 08 fev. 2015.

LISTA DE REDES SOCIAIS. 2015. Disponível em: <http://pt.wikipedia.org/wiki/Lista_de_redes_sociais>. Acesso em 09 fev. 2015.

METAANALISE. **Brasileiro passa mais tempo na internet do que vendo TV**, 2012. Disponível em: <http://www.metaanalise.com.br/inteligenciademercado/index.php?option=com_content&view=article&id=6630:brasileiro-passa-mais-tempo-na-internet-do-que-vendo-tv&catid=9:pesquisas&Itemid=359>. Acesso em: 23 fev. 2015.

NAGESH, Anirudh; CAICEDO, Carlos E. **Cross-platform mobile application development**. In: 10th Annual Conference on Telecommunications and Information Technology. Disponível em: <http://www.academia.edu/4233075/Cross-Platform_Mobile_Application_Development>. Acesso em: 26 fev. 2015.

PANTOJA, Victor. **Um framework para integracao entre aplicacoes móveis e redes sociais**. Dissertação (mestrado). Programa de Pós-Graduação em Informática do Departamento de Informática da PUC-Rio. Pontifícia Universidade Católica do Rio de Janeiro - PUC-Rio. 2012. Disponível em: <http://www.maxwell.vrac.puc-rio.br/Busca_etds.php?strSecao=resultado&nrSeq=21094@1>. Acesso em: 20 fev. 2015.

PAVLIĆ, Daniel; PAVLIĆ, Mile; JOVANOVIĆ, Vladan. **Future of Internet technologies**. In: 35th International Convention of Information Communication Technology, Electronics and Microelectronics (MIPRO 2012), May 21-25, 2012, p. 1377-1371.

RADIA, Nimish; ZHANG, Ying; TATIPAMULA, Mallik; MADISETTI, Vijay K. **Next-generation applications on cellular networks: trends, challenges, and solutions**, IEEE v. 100, n. 4, p. 841 – 854, 2012

REUVER, Mark de; ONGENA, Guido; BOUWMAN, Harry. Should mobile Internet be an extension to the fixed web? Fixed-mobile reinforcement as mediator between context of use and future use, 2013. **Telematics and Informatics**, v. 30, n.2, p. 111-120.

ROSI, Alberto, MAMEI, Marco; ZAMBONELL, Franco; DOBSON, Simon; STEVENSON, Graeme; YE, Juan. **Social sensors and pervasive services: approaches and perspectives**. In: IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011, p. 525 – 530.

SATYANARAYANAN, Mahadev. **Fundamental challenges in mobile computing**. In: 5th annual ACM symposium on Principles of distributed computing, p. 1-7, May 1996.

SATYANARAYANAN, Mahadev. **Mobile computing: the next decade**. In: 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond (MCS), June 2010, v. 15, n. 2, p. 2-10.

TENG; Chia-Chi, HELPS, Richard. **Mobile application development: essential new directions for IT**. 2010 Seventh International Conference on Information Technology, 471 - 475.

UOL Tecnologia. **Tablets já roubam público da TV, diz pesquisa**. 215. Disponível em: <<http://anid.com.br/site/index.php/ultimas/43-tablets-roubam-publico-tv.html>>. Acesso em: 23 fev. 2015.