Quem Indica? (QI): Uma proposta de rede social para compartilhamento de experiênciais acadêmicas e profissionais[1]

Karla Pereira do Carmo, Lucas de Paiva Rosa Gaspar, Maria Isabel de Sousa Carneiro & Rafael Souza Pires

***Abstract* — This paper aims to present a new social network proposal called Quem Indica? (QI), which aims to offer professionals, students and institutions and effective way to share and consult academic or professional experiences experienced during life. Related works that served as inspiration for the creation of this social network and technologies that enabled its construction will also be explored in order to substantiate it.**

***Index Terms* — Social Networks, Node.js, Neo4j, Angular 6.**

***Resumo* — Este trabalho tem como objetivo apresentar uma nova proposta de rede social intitulada Quem Indica? (QI), que visa oferecer a profissionais, estudantes e instituições uma forma breve mas eficaz de compartilhamento e consulta de experiências acadêmicas ou profissionais vivenciadas durante a vida. Trabalhos relacionados que serviram de inspiração para criação desta rede social e tecnologias que viabilizaram a sua construção também serão exploradas a fim de fundamentá-la.**

***Palavras chave* — Redes Sociais, Node.js, Neo4J, Angular 6.**

I. INTRODUÇÃO

Desde os primórdios da humanidade a raça humana buscou se organizar em grupos, seja para se manterem mais seguros, expandirem suas conquistas ou trocarem experiências sobre algo em comum. Inicialmente, utilizavam de gestos, sons e expressões para se comunicarem uns com os outros, mas com o passar dos tempos, essa comunicação foi sendo aprimorada, dando origem à linguagem verbal.

A evolução constante também trouxe aos seres humanos outras técnicas extraordinárias: a de relatar fatos por meio de desenhos e também por meio da escrita, meios estes de suma importância para transmissão de acontecimentos e experiências dentro de um grupo e através de gerações.

Com o advento e avanço da tecnologia, as redes sociais se tornaram ferramentas indispensáveis para conectar pessoas de vários extremos do planeta. Segundo Boyd, redes sociais nada mais são que serviços disponíveis na rede mundial de computadores que permitem indivíduos contruírem perfis públicos ou semi-públicos para que possam compartilhar experiências com outros usuários que também fazem uso do mesmo serviço [1]. São inúmeros os diferentes perfis de redes sociais que atualmente se encontram disponíveis para uso na *Internet*, mas dentre elas, as redes voltadas às relações pessoais, educacionais e profissionais se destacam.

Graças ao grande volume de informações geradas pelas pessoas nas redes sociais, muitas empresas ou grupos de trabalho hoje em dia conseguem identificar possíveis candidatos para contribuírem com seu negócio ou atividade casual sem ao mesmo conhecê-los pessoalmente. Isso é muito importante, pois aumenta significativamente o raio de alcance de quem procura novos parceiros. Porém, muitas vezes essa consulta por informações detalhadas das experiências de um indivíduo pode demandar uma sequência de várias pesquisas em diferentes redes sociais, e que em alguns casos, pode até ser confundido como "invasão de privacidade".

Baseando-se na necessidade de oferecer mais uma alternativa *online*, porém mais prática de comunicação interpessoal e interempresarial entre indivíduos, este trabalho apresenta uma nova alternativa para compartilhamento de experiências do cotidiano que, engajam de alguma forma, o compartilhamento de experiências profissionais e/ou acadêmicas. Intitulada Quem Indica?, ou de forma abreviada, QI, esta plataforma tem como objetivo permitir que pessoas compartilhem de forma prática e breve experiências profissionais ou acadêmicas que viveram com um determinado indivíduo por meio de breves depoimentos, possibilitando assim, que suas habilidades e experiências vividas sejam facilmente acessíveis por outras empresas e/ou grupos de trabalho que buscam por novos colaboradores.

Em sequência este artigo está organizado da seguinte forma: Na seção II são apresentados trabalhos relacionados, plataformas ou redes sociais que, de alguma forma, buscam facilitar o compartilhamento de experiências acadêmicas e profissionais entre as pessoas. Na seção III serão descritas as particularidades das ferramentas e tecnologias que foram utilizadas para possibilitar a criação do sistema QI. Na seção IV é abordado o princípio de operação e funcionamento da rede social proposta, bem como detalhes importantes de utilização da mesma. Por fim, na seção V, são apresentadas as conclusões do trabalho.

II. REDES SOCIAIS E PLATAFORMAS RELACIONADAS

Com o avanço da tecnologia e o aumento de usuários com acesso à *Internet*, diversas redes sociais surgiram para atender

[1] Trabalho de Conclusão de Curso apresentado ao Instituto Nacional de Telecomunicações como parte dos requisitos para obtenção do Grau de Bacharel em Engenharia da Computação. Orientador: Prof. Me. Renzo Paranaíba Mesquita. Trabalho aprovado em 24/11/2018.

o público empresarial e acadêmico, com intuito de substituir currículos e portfólios tradicionais pela mídia *online*.

Redes sociais como *LinkedIn* [2], Catho [3], *BusinessFriend* [4] e BocaABoca [5] são exemplos populares entre o público desse mercado. Todas elas oferecem a facilidade de permitir que as pessoas compartilhem experiências profissionais ou acadêmicas entre si, porém, cada uma possui as suas próprias particularidades.

Fundada nos Estados Unidos em meados de 2003 e disponível em diversas línguas, o *Linkedin* ostenta do título de maior rede social profissional do planeta.[2] Nesta rede, cada usuário ou até mesmo empresas possuem uma lista de "conexões", que nada mais são que pessoas na qual elas conhecem, tenham apreço ou interesse profissional. Por meio dela é permitido recomendar pessoas, vagas de emprego e também compartilhar de aptidões desenvolvidas durante a carreira por meio de comentários e textos de outros usuários. Em suma, o *LinkedIn* funciona como um estreitador de laços entre empregados e empregadores de diferentes áreas.

A empresa Catho é uma plataforma brasileira fundada em 1996 e que funciona como um classificado *online* de currículos e vagas. Sua missão é "ajudar pessoas a se realizarem profissionalmente, empresas a serem mais produtivas e o Brasil a prosperar". A Catho tem o total controle dos anúncios de suas vagas, sendo prudente em todos os processos de elaboração do seu serviço. No processo de cadastro das vagas é feito o reconhecimento de todos os dados da empresa para garantir consistência e veracidade na divulgação da oportunidade. Quaisquer anúncios disponibilizadas pela plataforma são passados por uma revisão para correção de possíveis erros, sejam eles ortográficos ou impróprios, garantindo um alto padrão no serviço prestado. A Catho fornece uma base de dados para vários tipos de vagas em todo o Brasil, e permite que um usuário envie seu currículo para uma vaga específica pelo próprio *website* da plataforma. Apesar de se apresentar como uma plataforma de alta confiabilidade e popular no Brasil, seus serviços são pagos.

A rede social QI, proposta neste trabalho, utiliza da opinião de pessoas para destacar pontos acadêmicos e profissionais positivos de outras pessoas. A plataforma *BocaABoca* faz algo muito parecido, porém, com produtos e serviços. O aplicativo possui empresas cadastradas na sua base de dados e o usuário pode indicá-las a seus amigos e familiares. Se uma venda for fechada, a pessoa que indicou esta empresa será remunerada por isto. A plataforma aproveitou da confiança que as pessoas têm em seus contatos pessoais quando se trata de uma indicação, visto que as pessoas se sentem muito mais confortáveis em comprar produtos que outras pessoas já aprovaram. E o ato de bonificar um cliente ao indicar o produto ou solução para outras pessoas pode fazer com que ele compre cada vez mais da mesma empresa e que continue a recomendar os produtos da empresa a mais pessoas, o que gera uma certa fidelidade. Assim, acabam beneficiando-se todas as partes envolvidas.

O *BusinessFriend* é uma rede social profissional que foi lançada em 2015 que tem a premissa de ser uma rede social de identidade social única, ou seja, que possui todos os meios de comunicação necessários para administrar a vida profissional dos usuários em uma só plataforma. Essa identidade única surgiu devido ao problema percebido em que um professional geralmente tem que utilizar cerca de cinco aplicativos/plataformas diferentes para conseguir atender todas as suas necessidades. Com o *BusinessFriend* os usuários podem fazer chamadas de vídeo, mandar mensagens instantâneas, tem até 2GB de armazenamento gratuito na nuvem, compartilhamento de conteúdo com colegas, entre vários outros benefícios em um ambiente simples e amigável que contempla todas essas funções. Segundo o vice-presidente da *BusinessFriend*, o aplicativo foi gerado para pessoas que possuem síndrome de comunicação dispersa (SCS), por isso, o aplicativo fornece todos os meios de comunicações possíveis, para que o usuário consiga ter maior controle sobre suas mensagens, *e-mails* e dados.

Além das redes sociais ou plataformas voltadas para a área profissional, destacam-se também plataformas voltadas para a área acadêmica, como o *Docsity* [6] e o Lattes [7].

O *Docsity* é um acervo completo de esquemas, exercícios, notas de aula, resumos, teses e outros tipos de documentos que são indispensáveis no dia a dia de qualquer estudante. Além do material fornecido (pelos próprios estudantes que participam da rede), os usuários podem tirar dúvidas com outros usuários da comunidade. Ainda há um *blog* em que se encontra diversos fatos curiosos e dicas de estudo, de empregos e intercâmbios. Essa rede social surgiu devido a indignação de um estudante, Riccardo Ocleppo, de ter tanta dificuldade em reunir materiais de ajuda e de dificuldade em conseguir se comunicar com outros estudantes de outras cidades e até mesmo países. Assim, *Docsity* teve sua jornada iniciada em 2010 e conta com auxílio de estudantes ao redor do mundo que colaboram com seus trabalhos. Uma das suas particularidades é que o usuário recebe pontos por cada ação realizada dentro da rede, como postar os próprios materiais, responder as perguntas de outras pessoas ou ter um arquivo baixado por várias pessoas. Esses pontos são acumulados e permitem que o usuário consiga baixar mais documentos e subir no *ranking* para obter outros benefícios exclusivos dentro da plataforma.

O Lattes, diferentemente das plataformas já apresentadas anteriormente, não é uma rede social, mas sim "uma plataforma de integração de bases de dados de currículos, de grupos de pesquisa e de Instituições de ensino em um único sistema de informações" [7] Desenvolvido por alunos das universidades Federais de Santa Catarina e de Pernambuco, juntamente com outros profissionais, o Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) lançou a plataforma em agosto de 1999, após anos de tentativas de criar um padrão de base de currículos para ser utilizado no âmbito do Ministério da Ciência e Tecnologia. Brasileiro. Para incentivar o uso dessa padronização de base de currículos nacionais, o CNPq tornou a licença do *software* gratuita e começou a fornecer consultoria para a implementação em outros países. O Lattes também foi implementado em outros países da América Latina, como a Colômbia, Equador, Chile, Peru e Argentina. A plataforma também exibe dados e estatísticas de todos os currículos cadastrados. As estatísticas são divididas por região, instituições, sexo ou idade. A sua visibilidade vem aumentando devido aos diversos acordos

institucionais que a CNPq possui. Estes acordos, além de aumentar a visibilidade da plataforma, também traz benefícios as instituições, já que estas conseguem extrair diversos dados relativos aos seus pesquisadores.

III. Tecnologias Utilizadas

A. *Node.js*

O *Node.js* é uma plataforma de código aberto que possibilita a execução de código *JavaScript* fora do navegador. Esta plataforma utiliza do interpretador do navegador *Google Chrome*, o V8, que é de propriedade do *Google*, e foi desenvolvida utilizando as linguagens de programação *C*, *C++* e *JavaScript* [8].

O *Node.js* é amplamente utilizado ao redor do mundo para os mais diversos tipos de aplicações como aplicações de *back-end*, construções de *Application Programming Interfaces* (APIs), aplicações *front-end* e *full-stack* além de outras áreas. A Figura 1 mostra as principais áreas de desenvolvimento nas quais o *Node.js* é utilizado.

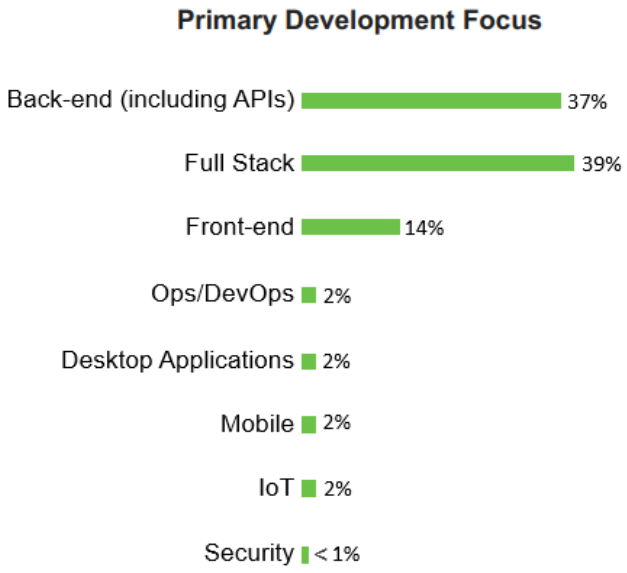

Figura 1: Principais áreas de desenvolvimento com *Node.js* [9]

Atualmente, o *Node.js* é utilizado em diversas empresas ao redor do globo, tais como: *IBM*, *Netflix*, *Uber* e *PayPal*.

Segundo uma pesquisa feita pelo *GitHub*, o *JavaScript* é a linguagem que possui o maior número de contribuidores em repositórios públicos e privados, devido ao grande número de desenvolvedores de *JavaScript* ao redor do mundo. O *Node.js* se torna um grande atrativo, pois abre portas para o desenvolvimento de aplicações escaláveis sem a necessidade de aprender outra sintaxe [9].

De acordo com a *Node.js Foundation*, a fundação mantenedora da plataforma, o *JavaScript* é uma das linguagens que mais possui materiais disponíveis e de qualidade para aprender a programar, conforme ilustra a Figura 2..

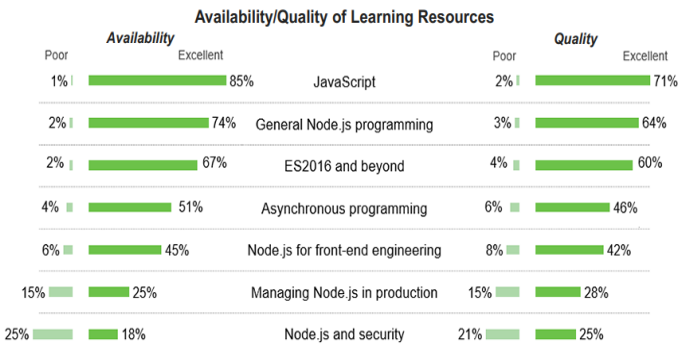

Figura 2: Porcentagem de disponibilidade e qualidade dos materiais para aprendizado de diferentes linguagens e técnicas relacionadas [9].

Outro grande atrativo da plataforma, é a sua usabilidade em empresas devido ao aumento de produtividade, a redução de custos no projeto e ao aumento da performance geral da aplicação. A Figura 3 ilustra os principais pontos impactados ao se utilizar o *Node.js.*

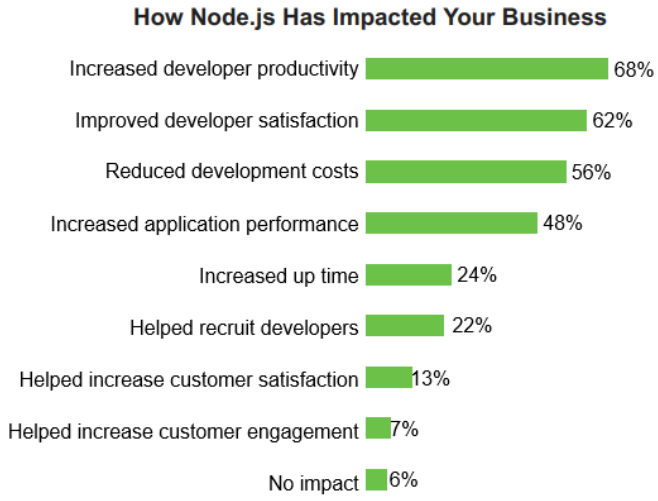

Figura 3: Impacto das áreas de desenvolvimento ao se utilizar do *Node.js* [9].

O *JavaScript* possui uma natureza assíncrona, ou seja, um processo não aguarda o outro terminar para começar a executar. Isso provê ao *Node.js* uma base para manejar múltiplas requisições e atendê-las de forma simultânea. Isso é feito através do *Event Loop*, ou repetição de eventos. O *Event Loop* é uma repetição infinita que verifica se existem novos eventos a serem tratados em sua fila de eventos. Ao receber um evento, o *Event Loop* o executa e passa o seu retorno para uma função, chamada de *callback*, e busca pelo próximo evento na fila [10]. Os eventos são enviados à fila do *Event Loop*, através *do Event Emmiter*, ou emissor de eventos. Este é um paradigma conhecido como Programação Orientada A Eventos (POE), ilustrado na Figura 4.

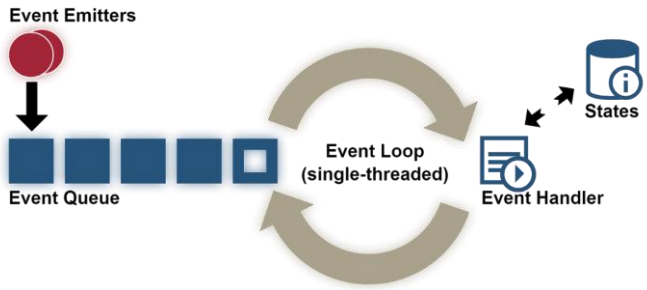

Figura 4: Representação da Programação Orientada A Eventos (POE). [10]

O *Node.js* conta com um conjunto de módulos que auxiliam no desenvolvimento de aplicações, tais como *Hypertext Transfer Protocol* (HTTP), responsável por receber requisições e enviar as respostas, o *File System* (FS) que permite a interação com arquivos do sistema, o OS que possibilita a interação com o sistema operacional, entre muitos outros.

O *Node Package Manager* (NPM) é o gerenciador de pacotes do *Node.js* que permite que desenvolvedores consigam facilmente compartilhar pacotes com funções *JavaScript* com outros desenvolvedores. Pacotes como *Lodash* que possui funções utilitárias e corriqueiras para acelerar o desenvolvimento, *Express* que facilita a criação de APIs, pacotes de conexão com banco de dados entre outros, podem ser encontrados no NPM [10].

B. Neo4j

O *Neo4j* [11] é um banco de dados não relacional (*NoSQL*) orientado a grafos lançado em 2010 pela empresa *Neo Technology*. Sua implementação é feita em Java e distribuída para outras linguagens juntamente com o *Cypher*, a linguagem oficial de consultas do *Neo4j*, que possibilita a criação, modificação e busca de dados em uma estrutura baseada em grafos. O modelo de grafos é uma estrutura de dados composta por nós (vértices) ligados por meio de relacionamentos (arestas), permitindo que cada nó tenha suas próprias propriedades, ou seja, atributos armazenados com seus respectivos valores.

Este tipo de banco de dados é altamente recomendado para criações de novas redes sociais, funcionando muito bem também em sistemas que envolvem inteligência artificial, aprendizado de máquina, *big data* e sistemas de telecomunicações. Um dos motivos que optou-se por utilizar do *Neo4j* para armazenamento dos dados da rede social proposta é que ele possui uma maneira inovadora de busca de dados, oferece alta performance de leitura e gravação comparado a alguns bancos de dados relacionais, além de fornecer proteção da integridade dos dados [11].

A eficiência no tempo das consultas do *Neo4j* é justificada pelos relacionamentos entre os nós, que possuem registros usados para organizar por tipo e direção os dados. Quando ocorre uma operação de busca ou junção, o banco acessa diretamente apenas uma parcela dos nós conectados, retirando do caminho da sua busca nós desnecessários. A Figura 5 ilustra de forma gráfica um exemplo de uma pequena base de dados criada no *Neo4j*. Nela é possível notar como os dados são organizados e como os relacionamentos são relevantes no momento de realizar buscas específicas.

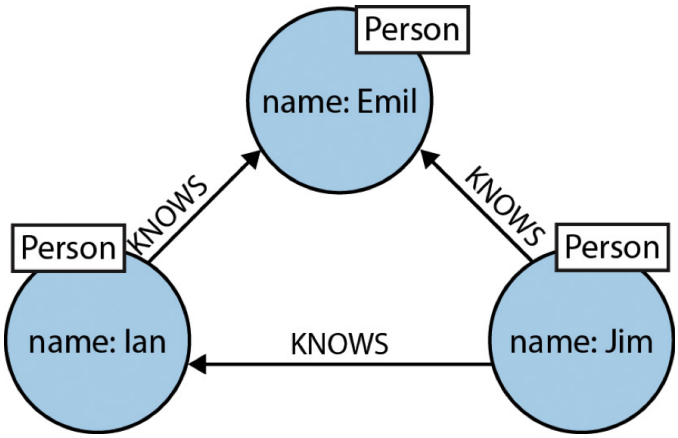

Figura 5: Exemplo de base de dados no *Neo4j* [11]

A linguagem *Structured Query Language* (*SQL*), popularmente usada em bancos de dados relacionais, opera sobre tabelas e registros. No *Cypher*, linguagem de consulta de dados do *Neo4J*, utiliza-se de componentes chamados *labels*, que possuem a mesma finalidade das tabelas dos bancos relacionais, porém, os nós equivalem aos registros no *SQL* e as relações entre esses nós, chamadas de *relationship*, equivalem aos relacionamentos em bancos relacionais. A sintaxe das duas linguagens são parecidas pois são compostas por cláusulas (palavras reservadas). Por exemplo, a cláusula *MATCH* no *Cypher* equivale ao *SELECT* do *SQL*, na qual, é responsável por organizar um padrão de busca que traga os nós e relacionamentos solicitados, a cláusula *SET* é usada em ambas para alterar a propriedade de uma relação ou nó, a cláusula *DELETE* é usada para apagar um nó ou relação inteira, entre várias outras similaridades. Todos os relacionamentos em *Cypher* são unidirecionais, apontando o sentido da relação através do símbolo ← ou → para representar a direção do relacionamento.

Dependendo do tipo de busca, no quesito redigibilidade o *Cypher* pode produzir resultados muito superiores a buscas em SQL. Na Figura 6 é ilustrada uma busca complexa na linguagem *SQL* e na Figura 7 é apresentada a mesma busca, porém, na linguagem *Cypher*.

```
SELECT name FROM Person
LEFT JOIN Person_Department
  ON Person.Id = Person_Department.PersonId
LEFT JOIN Department
  ON Department.Id = Person_Department.DepartmentId
WHERE Department.name = "IT Department"
```

Figura 6: Exemplo de busca em SQL [12].

```
MATCH (p:Person)-[:WORKS_AT]->(d:Dept)
WHERE d.name = "IT Department"
RETURN p.name
```

Figura 7: Exemplo de mesma busca realizada na Figura 6, porém, fazendo uso do *Cypher* [12].

Apesar de conter um contexto diferente em relação a outros bancos de dados mais populares no mercado, o *Neo4j* é um banco de dados fácil de aprender, pois oferece uma interface de usuário que inter-relaciona com a plataforma em tempo real, exibindo visualmente os relacionamentos com os dados, conforme foi ilustrado na Figura 5.

C. Angular 6

O *Angular 6* (também chamado de *Angular 2+*) é um *framework* para desenvolvimento de aplicações *web* que trabalha junto das tecnologias *Hypertext Markup Language* (HTML), *Cascading Style Sheets* (CSS) e *TypeScript*. Mantida pela *Google*, foi escrito em *TypeScript*, uma linguagem que utiliza de recursos do *EcmaScript 5 e do EcmaScript 6*, tornando-a de fácil entendimento para desenvolvedores *JavaScript*. O *TypeScript* traduz seu código para *JavaScript*, e depois o compila [13].

Esse poderoso *framework* utiliza do conceito de *Single Page Aplicattion* (SPA), aplicações de única página, o que faz com que a aplicação carregue novos *templates* sem que a tela inteira seja desenhada novamente, trocando apenas a seção que deve ser mudada, por meio de um módulo chamado *router*, que se encarrega de alterar tais áreas. Além disso , o *Angular* possui outros módulos como o *HttpClient*, que realiza requisições *Hypertext Transfer Protocol* (HTTP), e os *observables*, que faz a troca de mensagens entre fornecedores e consumidores de dados.

O *framework* ainda conta com injeção de dependências, criação de módulos, *components*, *services*, *templates* e diretivas, para tornar mais simples o desenvolvimento de uma aplicação que possa precisar de bom desempenho.

O alicerce de uma aplicação em *Angular* são os *NgModules*, conhecidos como módulos. Os módulos são conjuntos de blocos de códigos, que definem um contexto da aplicação. Toda aplicação possui um módulo principal, que é inicializado ao se iniciar o aplicativo. Eles são compostos por um conjunto de *components*, que fazem uso de *services*. Ao se iniciar um módulo, o mesmo inicializa o seu componente raiz, como mostrado na Figura 8.

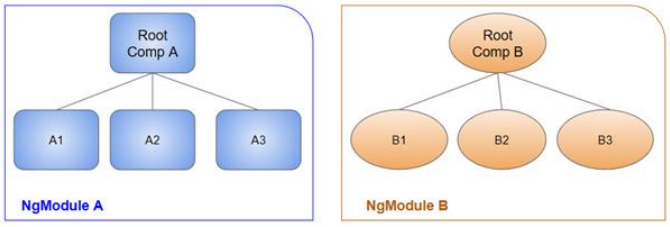

Figura 8: Conjunto de componentes do Angular6. [13]

Os *Components* controlam um conjunto de telas, chamado de *view*, e é responsável por manipular o *Document Object Model (DOM)*, ou árvore de componentes de uma aplicação. Os *Components* são classes que possuem a lógica do componente da aplicação, e os *Templates* mostram os dados de uma certa seção da aplicação. Tais integrantes do sistema, controlam as *views* através da ligação de dados e eventos. A ligação de dados e a de eventos se encarregam de inserir os valores no DOM e de recuperar as entradas do usuário para executar alguma ação. A Figura 9, exemplifica como é feita a comunicação entre um *Template* e um *Component*.

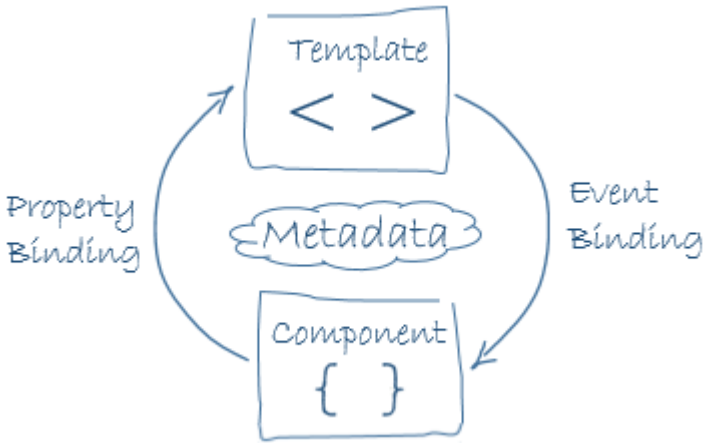

Figura 9: Comunicação entre um *template* e um *component* [13].

Os *Templates* são muito parecidos com *tags* normais de HTML, contudo, contém uma sintaxe que altera o HTML de acordo com o estado da aplicação.

O *Angular* suporta uma ligação bidirecional entre os *Components* e os *Templates* através de marcações nos componentes HTML, conforme ilustra a Figura 10.

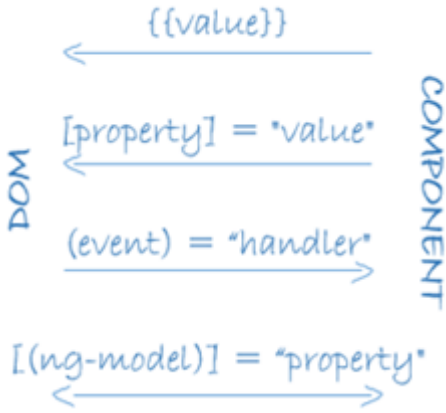

Figura 10: Marcações entre um *template* e um *component* [13].

Os *Services*, ou serviços, são classes com um propósito bem definido, que possui um conjunto de funções e valores que o aplicativo precisa. Não são ligados a um componente específico, podendo ser reutilizados por diversos componentes, evitando a duplicação de código. Os *Components* devem conter apenas métodos de ligação de

dados e de eventos, e possuem os *Services* como aliados, delegando-os para funções como requisições de dados para um servidor, validação de formulários, tratamento de dados, entre outras funções. Os *Services* são classes injetáveis, que utiliza de uma técnica chamada injeção de dependências. A Figura 11 mostra como deve ser feita tal injeção.

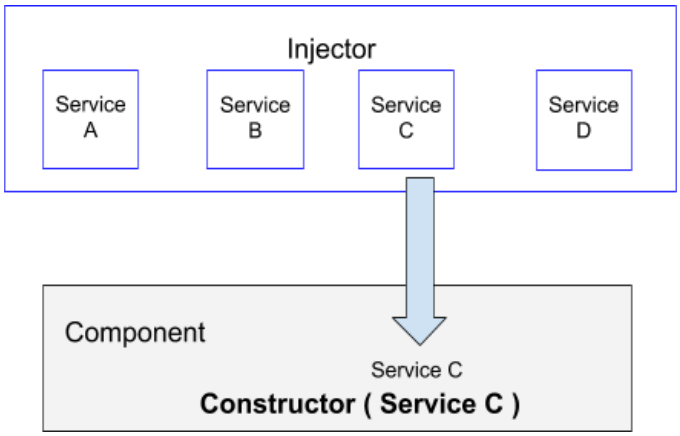

Figura 11: Arquitetura de Injeção de Dependências. [13]

A Injeção de Dependências (ID) é responsável por fornecer os *Services* ou qualquer outra dependência necessária aos *Components*. Ela não é responsável por funções como validação de entrada ou requisição de dados, mas seu objetivo é apenas prover as necessidades dos *Components*. O *Injector* é o principal mecanismo da ID, ele é criado pelo *Angular* para servir toda a aplicação e faz uso de um *Provider*, que o informa o local onde encontrar cada dependência.

Esse conjunto de blocos torna o Angular um *framework* poderoso e de grande aplicabilidade para o desenvolvimento de aplicações *web*. A Figura 12 mostra um panorama completo da arquitetura do *Angular*.

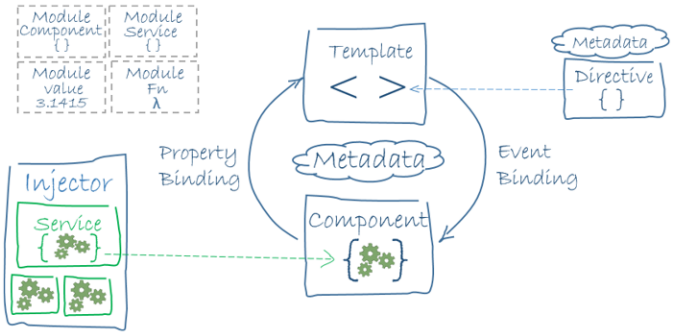

Figura 12: Panorama da arquitetura do Angular. [13]

IV. A PLATAFORMA QUEM INDICA? (QI)

Neste tópico é explicado em detalhes o princípio de implementação e funcionamento da rede social QI.

A. *Implementação*

Na Figura 13 é ilustrada a interação entre as tecnologias utilizadas para criação da plataforma QI. Usuários com interesses profissionais ou acadêmicos poderão usufruir da rede social por meio de páginas *web* desenvolvidas em *Angular 6*, que por trás dos bastidores, dependendo da ação do usuário, poderão enviar requisições *HTTP* do *tipo get, post, delete e post* para o servidor do *Node.js*. O método *get* solicita algum recurso ou objeto para o servidor, o *post* é usado para enviar um dado ou formulário *HTML* ao servidor, o *delete* informa por meio do *Uniform Resource Locator* (URL) o objeto a ser deletado e o *put* serve para criar ou modificar algum objeto do servidor. Assim, através da conexão entre o *Node.js* e o banco de dados *Neo4j* são executadas buscas para trazer as informações solicitadas, que são retornadas ao *front-end* (páginas *web*) por meio de um documento em formato *Javascript Object Notation (JSON)*.

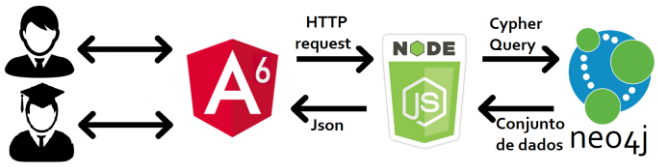

Figura 13: Diagrama de Funcionamento da rede social Quem Indica?.

A Figura 14 ilustra um diagrama de casos de uso de todas as funcionalidades do sistema. De forma geral, nesta rede social será possível gerenciar: usuários, grupos, pedidos de amizade, depoimentos, amigos e *login*, cada um deles com suas particularidades específicas. As funcionalidades do diagrama de casos de uso são explicadas de forma detalhada no princípio de funcionamento da plataforma, na subseção *B* desta seção.

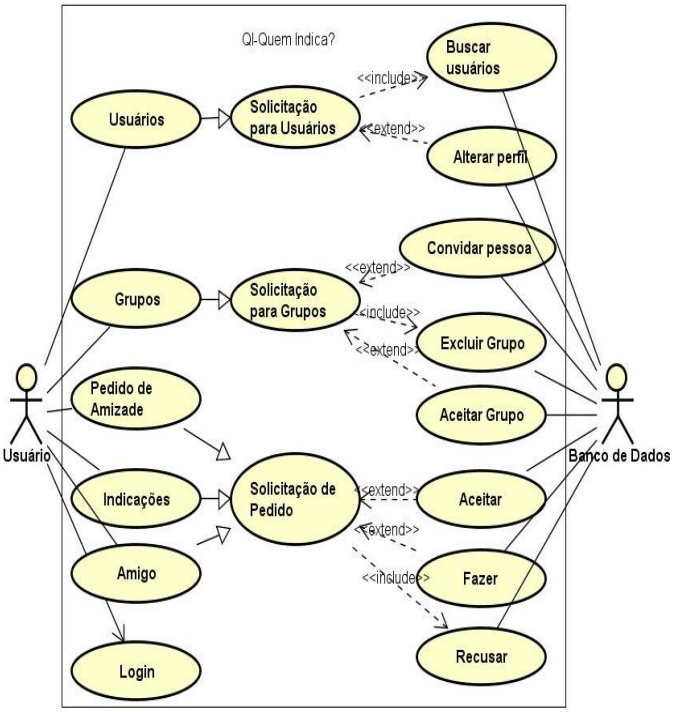

Figura 14: Diagrama de Casos de Uso.

Todas as possíveis ações que podem ser realizadas dentro desta rede social foram encapsuladas em uma *Application Program Interface* (API) criada inteiramente pelos autores, a fim de facilitar e organizar as chamadas ao *back-end* da aplicação. As funções da API podem ser verificadas nas Tabelas I,II,III,IV e V presentes no Apêndice A deste trabalho, que de forma geral, podem ser chamadas utilizando do formato ilustrado pela Figura 15. A parte 1 correponde ao caminho de onde se encontra publicada a API para uso, a parte 2 corresponde aos diferentes tipos de funções que podem ser chamadas, a parte 3 corresponde às subfunções, e por fim, na parte 4, são colocados os parâmetros que são passados na chamada das subfunções.

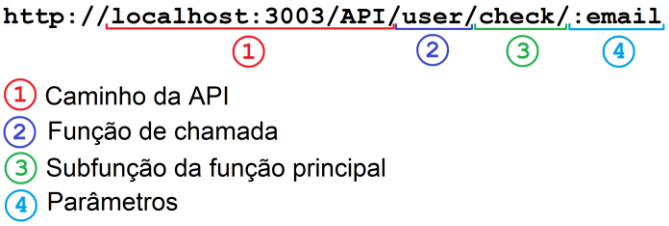

Figura 15: Princípio de funcionamento da API desenvolvida.

B. *Funcionamento*

Para fazer parte da rede social inicialmente é necessário realizar um cadastro. Um cadastro é feito de forma simples: basta o usuário preencher o formulário com o nome, profissão, local, *e-mail* e senha. Após realizar o cadastro, o usuário conseguirá fazer o *login* na rede informando seu *e-mail* e senha, como ilustrado na Figura 16.

Figura 16: Cadastro e *login*.

A plataforma disponibiliza um perfil público para empresas ou recrutadores que desejarem acessar um perfil de usuário, através de uma URL específica, referente ao usuário desejado. Devido a este fato, é possivel vizualizar o perfil sem realizar o *login*. Tal flexibilidade de acesso implica em alterações na página de acordo com o estado da aplicação, por isso, dois modelos diferentes de barra lateral podem aparecer, como ilustrado na Figura 17. A barra é usada como guia, para ajudar na navegação das funcionalidades disponibilizadas como: buscar pessoas, verificar o perfil de usuário, vizualizar amizades, indicações, grupos e a opção de sair da rede social.

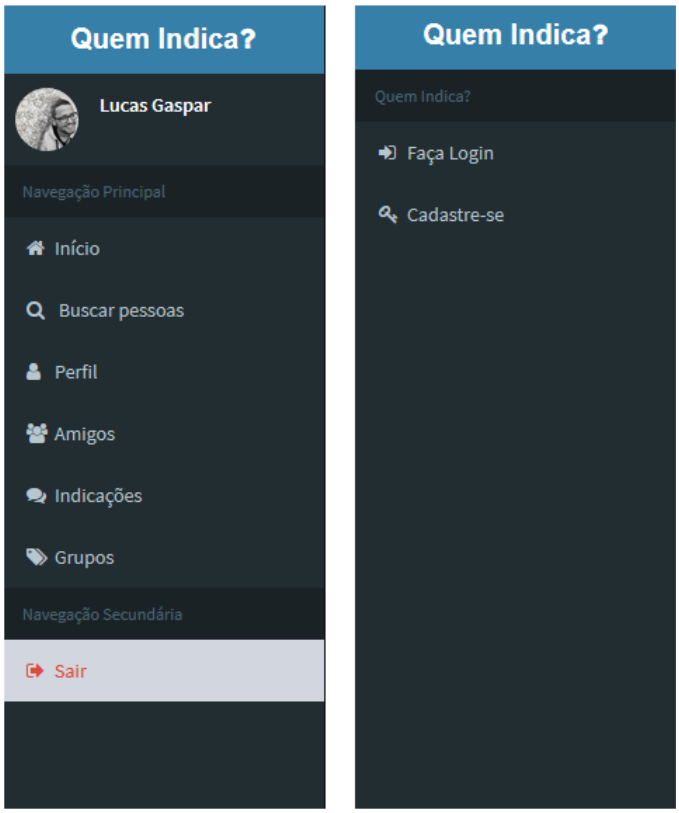

Figura 17: Barras laterais. A barra da esquerda aparece quando o usuário deseja fazer uso de vários recursos da rede social e a barra da direita para consultas rápidas à ela.

Após se conectar à rede social, o usuário será redirecionado para a tela inicial. Nesta tela estará disponível informações como números de indicações, grupos e amigos referentes ao usuário, além de disponibilizar o numero de indicações pendentes, solicitações de amizades recebidas e convites para participação de novos grupos, através de cartões como ilustrado na Figura 18.

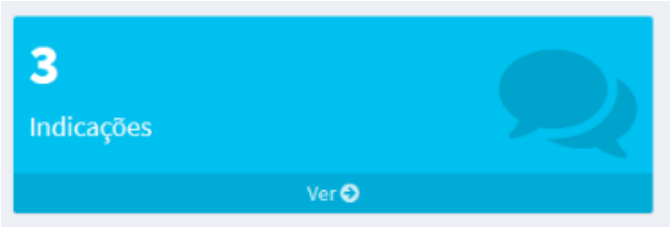

Figura 18: Cartão informativo.

Como mencionado anteriormente, uma funcionalidade de busca é oferecida aos usuários da rede social para que colegas

sejam encontrados e novas conexões possam ser criadas. A busca pode ser realizada pelo nome ou *e-mail*, preenchido em um campo de digitação. A busca retorna os resultados em cartões, como na Figura 19, mostrando foto, nome e profissão de novos usuários. Ao clicar no cartão, a plataforma redireciona para o perfil do usuário selecionado.

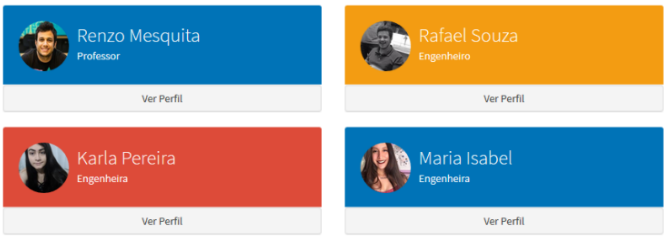

Figura 19: Resultado da busca de amigos.

Ao acessar o perfil de um usuário específico, apenas a parte central será alterada e as informações do usuário buscado serão carregadas. Nesta tela será exibido a quantidade de indicações que um usuário recebeu, de amigos que ele possui e de grupos pelo qual fez ou faz parte, conforme ilustra a Figura 20. O perfil também apresenta cartões de amigos, como ilustra a Figura 21, grupos (Figura 22) e indicações (Figura 23). Para não fazer pedidos muito pesados ao servidor, foi usado uma paginação de resultados, sendo retornado apenas cinco indicações, amigos e grupos. Caso o número dessas informações for maior do que cinco, o usuário pode solicitar mais cinco. Nesta tela, também é exibido um botão para solicitar amizade ou desfazer a amizade.

Figura 20: Informações de um usuário.

Figura 21: Amigos de um usuário.

Figura 22: Grupos de um usuário.

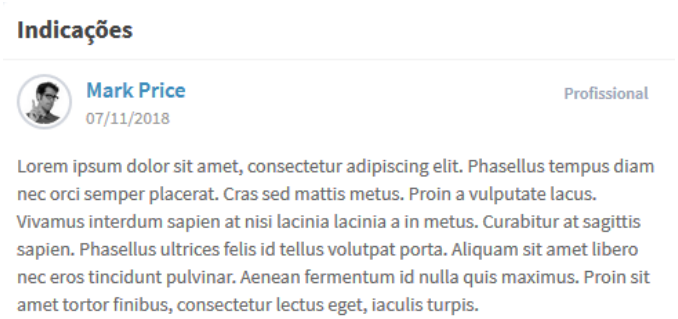

Figura 23: Exemplo de uma indicação recebida por um usuário.

Antes de usuários se tornarem amigos, será necessário que um pedido de amizade seja aceito. Na tela de amigos, além de apresentar os amigos que um usuário já possui, ainda é exibido uma lista de pedidos de amizade pendentes, conforme ilustra a Figura 24, e uma lista de solicitações requeridas pelo usuário, conforme Figura 25.

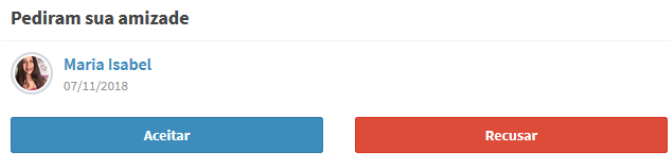

Figura 24: Solicitação recebida de amizade.

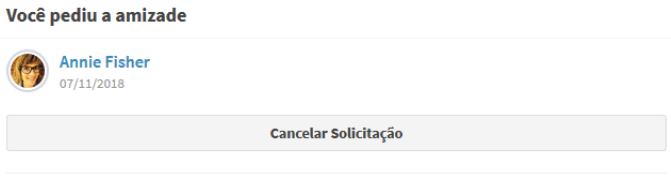

Figura 25: Solicitação enviada de amizade.

A rede social também permite que usuários participem de grupos, destinados para pessoas de um mesmo ciclo, ou assuntos em comum. Para se criar um novo grupo, basta clicar com o símbolo + no canto inferior direito da tela, onde aparecerá um campo para preencher as informações do grupo, como nome e tipo, disponibilizado como profissional ou acadêmico, conforme ilustrado pela Figura 26. Na tela de grupos o usuário poderá aceitar ou recusar a solicitação de um dono de grupo, como mostra a Figura 27, além de visualizar os grupos ao qual pertence, ilustrado pela Figura 28.

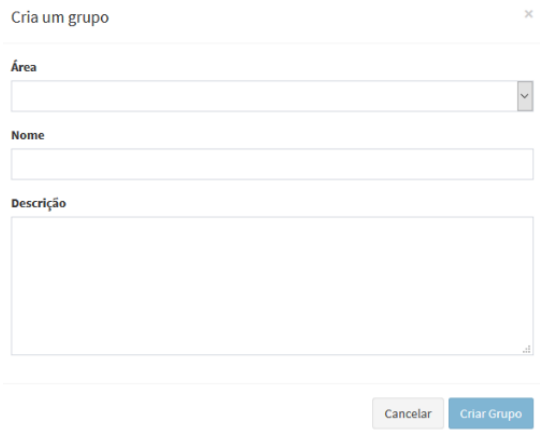

Figura 26: Criação de grupo.

Figura 27: Convite para participação de um grupo.

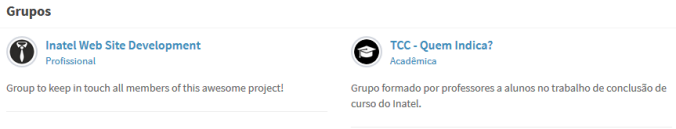

Figura 28: Grupos de um usuário.

Ao selecionar um grupo será possivel visualizar os seus detalhes e seus membros, como ilustra a Figura 29. Há um botão de "Sair do Grupo", caso o usuário não queira mais participar. Apenas o criador do grupo poderá convidar membros, ao clicar no botão para enviar o convite de participação do grupo, será exibido um buscador de membros do QI e o criador poderá pesquisar o usuário pelo nome. Assim, será exibido opções de membros para serem convidados para o grupo. Este processo é ilustrado pela Figura 30.

Figura 29: Detalhes de um grupo.

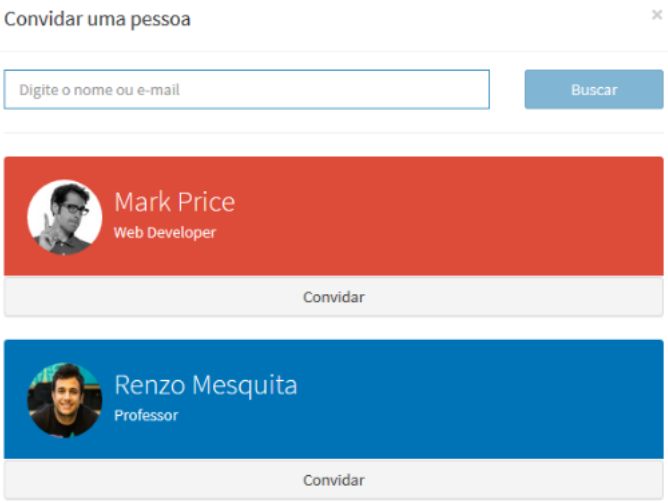

Figura 30: Convidar membros para um grupo.

Além da tela inicial de grupos de cada usuário, é possível o usuário aceitar ou recusar uma solicitação para participar de um grupo através do perfil do grupo específico. Será exibido um botão de "Aceitar convite" e outro de "Recusar convite", caso o dono do grupo tenha adicionado o usuário no grupo. Este processo é ilustrado na Figura 31.

Após a criação de um vínculo de amizade entre dois usuários, será aberta opções de funcionalidades entre ambos, como escrever uma indicação. Para escrever uma indicação para um amigo, basta entrar em seu perfil e clicar no botão redondo vermelho no canto inferior direito da tela, onde aparecerá uma caixa para preencher com as informações, conforme ilustra a Figura 32. Lembrando que tanto grupos quanto indicações podem ser classificadas de duas formas: profissional ou acadêmico.

Figura 31: Convite de solicitação com detalhes.

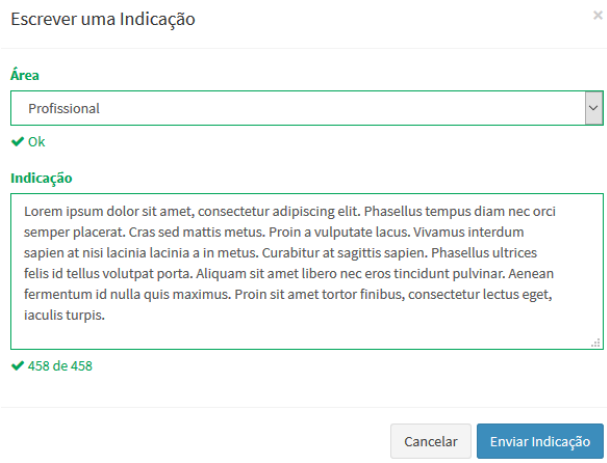

Figura 32: Escrever uma indicação.

Na tela de indicações, ficarão disponíveis as indicações de que um usuário recebeu, conforme ilustra a Figura 33. Após recerber uma indicação, o usuário pode aceitá-la ou recusá-la através de cartões, conforme ilustra a Figura 34, podendo também cancelar indicações que foram enviadas a outros usuários e que ainda não foram aceitas, conforme a Figura 35.

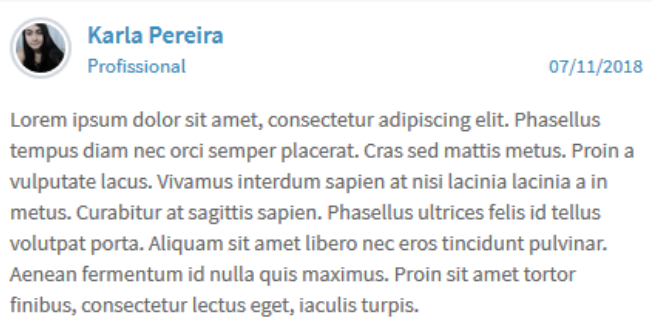

Figura 33: Indicação de um usuário.

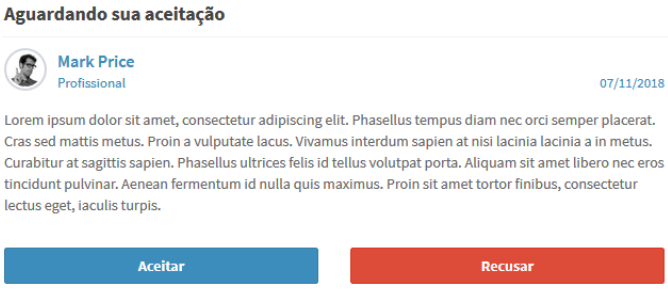

Figura 34: Indicação aguardando aceitação.

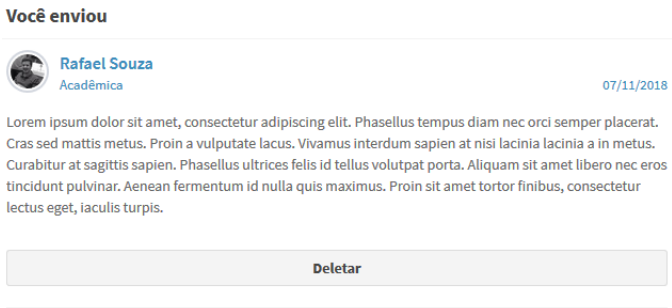

Figura 35: Indicação enviada e aguardando aceitação.

V. Conclusão

A rede social QI tem como objetivo permitir que seus usuários compartilhem experiências acadêmicas e profissionais de maneira simples e objetiva, funcionando como uma espécie de portifólio ou currículo para interesses interpessoais e interpresariais. Isso só foi possível graças ao estudo e integração das diferentes tecnologias apresentadas.

Dentre das dificuldade encontradas ao desenvolver a rede, a maior delas foi aprender a modelar e usar do banco de dados *Neo4j*, que apresenta um paradigma completamente diferente dos bancos baseados em SQL, no qual geralmente é mais utilizado e aprendido nas faculdades.

Como melhorias para a plataforma, sugere-se criar um sistema de *feedback* para cada depoimento, em que amigos de ambos usuários pudessem deixar uma reação positiva, como um “like”, ou uma reação negativa, como “dislike”, para transmitir veracidade de uma determinada aptidão de uma pessoa, também permitindo comentários, que possibilitam essa

mesma ideia. Para tornar a rede social mais interativa, seria colocado um calendário de eventos, com intuitos profissionais ou acadêmicos, para os usuários interagirem uns com os outros, como por exemplo, um reencontro de um grupo ou alguma outra ocasião, com informações descritivas como assunto, data e horário e podendo este, ser público ou privado para os usuários. Outra ideia seria acrescentar uma *timeline* para cada usuário, a fim de representar uma sequência de eventos vivenciados por cada um de forma linear e organizada de forma cronológica.

Por fim, a rede social ainda não se encontra ativa para o público. Ela precisa ser testada e validada em uma larga escala de usuários a fim de comprovar a sua eficácia na hora de permitir o rápido compartilhamento de experiências acadêmicas e profissionais entre seus usuários.

REFERÊNCIAS

[1] danah m. boyd, Nicole B. Ellison; Social Network Sites: Definition, History, and Scholarship, *Journal of Computer-Mediated Communication*, Volume 13, Issue 1, 1 October 2007, Pages 210–230

[2] Linkedin - disponível em https://www.linkedin.com/about - acessado em 7 de Setembro de 2018.

[3] Catho - disponível em https://www.catho.com.br - acessado em 7 de Setembro de 2018.

[4] BusinessFriend - disponível em https://www.businessfriend.com/privacy- acessado em 20 de Outubro de 2018.

[5] BocaaBoca - disponível em https://www.bocaaboca.digital/ - acessado em 20 de Outubro de 2018.

[6] Docsity - disponível em https://www.docsity.com/pt/pag/o-projeto/ - acessado em 04 de Setembro de 2018;

[7] Lattes - disponível em http://lattes.cnpq.br/ - acessado em 10 de Outubro de 2018;

[8] Codecademy - https://www.codecademy.com/articles/what-is-node - acessado em 7 de Novembro de 2018;

[9] Node.js - disponível em https://nodejs.org/en/user-survey-report/ - acessado em 7 de Novembro de 2018;

[10] Tmeoliveira GitHub - disponível em http://tmeloliveira.github.io/acbook-nodejs/lecture1.html - acessado em 7 de Novembro de 2018;

[11] Neo4J - disponível em https://neo4j.com/ - acessado em 7 de Outubro de 2018.

[12] Medium - disponível em https://medium.com/accendis-tech/uma-gentil-introdu%C3%A7%C3%A3o-ao-uso-de-banco-de-dados-orientados-a-grafos-com-neo4j-ca148df2d352 - acessado em 7 de Outubro de 2018.

[13] Angular IO - disponível em https://angular.io/guide/architecture- acessado em 5 de Novembro de 2018;

Karla Pereira do Carmo nasceu em Baependi, Minas Gerais, em 21 de julho de 1995. Possui título de técnica em informática pelo Instituto Federal de Educação, Ciência e Tecnologia – Campus Inconfidentes. Atualmente é graduanda em Engenharia de Computação no Instituto Nacional de Telecomunicações e estagiária na área de RF de telecomunicações na INATEL/Ericsson. Já participou da organização estudantil para promover eventos relacionados ao curso de Computação, participa da comunidade MCode do Inatel, no qual tem como objetivo incentivar a participação de mulheres na área de tecnologia e desenvolvimento e já auxiliou a equipe Raster ARMY na FETIN 2017 na área de desenvolvimento web.

Em 2017 atuou como estagiária no antigo Laboratório de Jogos do Inatel na área de pesquisa e desenvolvimentos de jogos 2D. Tem interesse nas áreas de GameDev, Redes de Computadores e Radiofrequência.

Lucas de Paiva Rosa Gaspar nasceu em São Paulo, São Paulo, em 22 de setembro de 1995. Atualmente é graduando em Engenharia de Computação no Instituto Nacional de e estagiário na área de desenvolvimento de software no INATEL/Ericsson. Foi monitor de Cálculo I, Álgebra Linear e Geometria Analítica, participou da organização estudantil para promover eventos relacionados ao curso de Computação, fez intercâmbio na Irlanda no WIT – Waterford Institute od Tecnhnology, onde cursou as matérias de Desenvolvimento de Jogos.

Em 2017 atuou como bolsista de IC – Iniciação Científica pela Fapemig no antigo Laboratório de Jogos do Inatel. Tem interesse nas áreas de Desenvolvimento de Sistemas Distribuídos e Jogos.

Maria Isabel de Sousa Carneiro nasceu em São José dos Campos, São Paulo, em 11 de novembro de 1994. Possui título de técnica em informática pela Escola Técnica Professor Everardo Passos de São José dos Campos. Atualmente é graduanda em Engenharia de Computação no Instituto Nacional de Telecomunicações e estagiária da área de suprimentos na EMBRAER. Já participou da organização estudantil para promover eventos relacionados ao curso de Computação, participou durante um ano da CP2eJr (Empresa Junior do Inatel), também participou do programa governamental Ciência sem Fronteiras nos anos de 2015 e 2016.

Rafael Souza Pires nasceu em São Bento do Sapucaí, São Paulo, em 21 de abril de 1995. Cursou o primário no Colégio Santa Ângela em Paraisópolis, Minas Gerais. Atualmente é graduando em Engenharia de Computação no Instituto Nacional de Telecomunicações e estagiário na área de desenvolvimento de software no INATEL/Ericsson., também participou de um intercambio para Alemanha no ano de 2018.

APÊNDICE A – TABELAS DE FUNÇÕES DISPONIBILIZADAS PELA API DESENVOLVIDA

TABELA I
FUNÇÕES DA API VOLTADAS A USUÁRIOS. NA COLUNA "CHAMADA" DESTA E DAS OUTRAS TABELAS, O PREFIXO http://localhost:3003/API ANTECEDE AS CHAMADAS.

USUARIO			
TIPO	CHAMADA	PARAMETROS	RETORNO
GET	/user/all/:pag	pag	Retorna todos os nós paginando com 5 componentes por pagina
GET	/user/search/:string/:id	ID	Retorna os nós que contém String no nome ou no E-mail
GET	/user/get_relation/:id1/:id2	id de usuário search, e id do usuário a ser encontrado	Retorna a relação entre dois nós
GET	/user/:token	token	Retorna o nó de acordo com o Token
GET	/user/get_by_id/:id	id do nó	Retorna o nó de acordo com o ID passado
GET	/user/check/:email	E-mail	Retorna um nó com e-mail passado como parâmetro, checando se o usuário foi cadastrado
GET	/user/get_friend_solicitations_recieved/:id	ID	Retorna o numero de solicitações de amizade feita ao nó do ID passado
GET	/user/get_friend_solicitations_send/:id	ID	Retorna o numero de solicitações enviadas por um Nó
GET	/user/get_depo_solicitations/:id	ID	Retorna o numero de solicitações de depoimentos feitas a um nó
GET	/user/get_group_invitations/:id	ID	Retorna o numero de convites de participação de grupo feitas a um nó
GET	/user/get_friend_number/:id	ID	Retorna o numero de amigos do nó
GET	/user/get_depo_number/:id	ID	Retorna o numero de depoimentos do nó
GET	/user/get_written_depo_number/:id	ID	Retorna o numero de depoimentos escritos por um nó
GET	/user/get_group_number/:id	ID	Retorna o numero de grupos que o nó faz parte
GET	/user/get_created_group_number/:id	ID	Retorna o numero de grupos que um nó criou
GET	/user/get_asked_group_number/:id	ID	Retorna o numero de grupos que o nó pediu para participar

TABELA I (CONTINUAÇÃO)

POST	/user/	ID, nome, e-mail, foto, profissão, local são passados no corpo da request	Retorna o nó criado no Banco
PUT	/user/:id	ID, nome, e-mail, foto, profissão, local são passados no corpo da request	Altera as informações do nó no Banco pelo ID
DELETE	/user/:id	ID	não tem retorno, pois deleta o no do banco

TABELA II
FUNÇÕES DA API VOLTADAS AOS DEPOIMENTOS

DEPOIMENTO			
TIPO	CHAMADA	PARAMETROS	RETORNO
GET	/depoimento/has_depo/:id/:pag	ID, pag	Retorna todos os depoimentos de um nó, juntamente com seu autor, paginando de cinco em cinco
GET	/depoimento/has_depo/:idUser/:idDepo	IdUser, IdDepo	Retorna um depoimento e seu autor
GET	/depoimento/waiting_confirmation/:id/:pag	ID, pag	Retorna todos os depoimentos e seus específicos autores, que precisam ser aceitos, paginando de cinco em cinco
GET	/depoimento/waiting_confirmation/:idUser/:idDepo	IdUser, IdDepo	Retorna um depoimento pendente e seu especifico autor
GET	/depoimento/wrote/:id/:pag	ID, pag	Retorna todos depoimentos escrito pelo nó, juntamente com seus especificos destinatários, paginando de cinco em cinco
POST	/depoimento/:id1/:id2	ID1, ID2	Cria um depoimento entre dois nós
PUT	/depoimento/:idUser/:idDepo	IdUser, IdDepo	Atualiza a relação "WAITING_CONFIRMATION" para uma relacao "HAS_DEPO"
DELETE	/depoimento/has_depo/:idUser/:idDepo	IdUser, IdDepo	Deleta um depoimento com a relação "HAS_DEPO"(depoimento já aceito)
DELETE	/depoimento/has_depo/:id	ID	Deleta todos os depoimentos com a relacao "HAS_DEPO" de um nó
DELETE	/depoimento/waiting_confirmation/:idUser/:idDepo	IdUser	Deleta um depoimento com a relação "WAITING_CONFIRMATION" de um nó
DELETE	/depoimento/waiting_confirmation/:id	ID	Deleta todos os depoimentos com a relacao "WAITING_CONFIRMATION" de um nó
DELETE	/depoimento/wrote/:idUser/:idDepo	IdUser, IdDepo	Deleta um depoimento escrito pelo nó
DELETE	/depoimento/wrote/:id	idUser	Deleta todos os depoimentos escritos pelo nó

TABELA III
FUNÇÕES DA API VOLTADAS AOS GRUPOS

GRUPOS			
TIPO	CHAMADA	PARAMETROS	RETORNO
GET	/groups/:idU/:pag	idU, pag	Retorna todos os grupos dos quais o nó é membro paginando de cinco em cinco
GET	/groups/members/:idG/:pag	idG, pag	Retorna todos os nós de um grupo paginando de cinco em cinco
GET	/groups/ask_membership/:idG/:pag	idG, pag	Retorna todos os pedidos de participação de um grupo paginando de cinco em cinco
GET	/groups/invite/:idU/:pag	idU, pag	Retorna todos os convites de participação de grupo recebidos por um nó
POST	/groups/:idU	idU, nome, descrição, area, idCreator, data	Método responsável pela criação de um grupo
POST	/groups/invite/:idG/:idI	idG, idI	Método que convida um nó para um grupo
POST	/groups/ask_membership/:idG/:idU	idG, idU	Método para pedir a participação de um grupo
PUT	/groups/ask_membership/:idG/:idU	idG, idU	Método que aceita um pedido de participação
PUT	/groups/invite/:idG/:idI	idG, idU	Método para um convidado aceitar um convite de um grupo
DELETE	/groups/:idG/:idU	idG, idU	Método responsável por apagar as relações entre grupo e um nó desde que essa relação seja uma das seguintes: INVITED, ASKED_MEMBERSHIP ou IS_MEMBER
DELETE	/groups/:idG	idG	Método responsável por deletar o grupo e todas as relações da mesma

TABELA IV
FUNÇÕES DA API VOLTADAS AOS PEDIDOS DE AMIZADE

ASKED_AS_FRIEND			
TIPO	CHAMADA	PARAMETROS	RETORNO
GET	/asked_as_friend/:id/:pag	ID, pag	Retorna todas solicitações feitas a um nó, paginando de cinco em cinco
GET	/asked_as_friend/:id1/:id2	ID1, ID2	Retorna a relação "ASKED_AS_FRIEND" entre dois nós
PUT	/asked_as_friend/:id1/:id2	ID1, ID2, date	Transforma uma relação tipo "ASKED_AS_FRIEND" para uma "IS_FRIEND" entre 2 nós
POST	/asked_as_friend/:id1/:id2	ID1, ID2, date	Cria uma relação "ASKED_AS_FRIEND" entre dois nós
DELETE	/asked_as_friend/:id1/:id2	ID1, ID2	Deleta a relação "ASKED_AS_FRIEND" entre dois nós
DELETE	/asked_as_friend/:id	ID	Deleta todas as relações "ASKED_AS_FRIEND" de um nó

TABELA V
FUNÇÕES DA API VOLTADAS A FUNÇÃO “É AMIGO”

IS_FRIEND			
TIPO	CHAMADA	PARAMETROS	RETORNO
GET	/is_friend/:id/:pag	ID, pag	Retorna todas as relações "IS_FRIEND" de um nó
GET	/is_friend/:id1/:id2	ID1, ID2	Retorna a relação "IS_FRIEND" entre dois nós
POST	/is_friend/:id1/:id2	ID1, ID2, date	Cria uma relação "IS_FRIEND" entre dois nós
DELETE	/is_friend/:id	ID	Deleta todas as relações "IS_FRIEND" de um nó
DELETE	/is_friend/:id1/:id2	ID1, ID2	Deleta a relação "IS_FRIEND" entre dois nós