Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

JÔNATAS DOMINGUES DE ALMEIDA CHIZZOLINI

DESENVOLVIMENTO DE APLICAÇÃO *"IDEAS ON START.UP"*: UMA ABORDAGEM UTILIZANDO NODE.JS, ANGULAR JS E MONGO DB

Assis/SP
2016

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

JÔNATAS DOMINGUES DE ALMEIDA CHIZZOLINI

DESENVOLVIMENTO DE APLICAÇÃO *"IDEAS ON START.UP"*: UMA ABORDAGEM UTILIZANDO NODE.JS, ANGULAR JS E MONGO DB

Projeto de pesquisa apresentado ao curso de Análise e Desenvolvimento de Sistemas do Instituto Municipal de Ensino Superior de Assis – IMESA e a Fundação Educacional do Município de Assis – FEMA, como requisito parcial à obtenção do Certificado de Conclusão.

Discente: Jônatas D. de Almeida Chizzolini
Orientador: Dr. Almir Rogério Camolesi

Assis/SP
2016

FICHA CATALOGRÁFICA

CHIZZOLINI, Jônatas Domingues de Almeida.
Desenvolvimento de Aplicação "Ideas on Start.UP": uma abordagem utilizando Node.JS, Angular JS e Mongo DB. / Jônatas Domingues de Almeida Chizzolini. Fundação Educacional do Município de Assis – FEMA – Assis, 2016.
Número de páginas.

1. Startup. 2. Node.JS. 3. Angular JS. 4. Mongo DB.

CDD:
Biblioteca da FEMA

DESENVOLVIMENTO DE APLICAÇÃO “IDEAS ON START.UP”: UMA ABORDAGEM UTILIZANDO NODE.JS, ANGULAR JS E MONGO DB

JÔNATAS DOMINGUES DE ALMEIDA CHIZZOLINI

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação de Análise e Desenvolvimento de Sistemas, avaliado pela seguinte comissão examinadora:

Orientador: _____
Dr. Almir Rogério Camolesi

Examinador: _____
Esp. Domingos de Carvalho Vilella Junior

Assis/SP
2016

DEDICATÓRIA

Dedico este trabalho de conclusão de curso à minha família, ao meu pai Gilberto, à minha mãe Vera Lúcia, e ao meu irmão Samuel, pelo incentivo e dedicação à minha pessoa. Da mesma forma, dedico-o à minha esposa Giseli, pelo seu fundamental apoio e respaldo às minhas jornadas de estudo.

AGRADECIMENTOS

Agradeço a Deus por me abençoar durante toda minha e me capacitar de modo a ter a condição de trabalhar e estudar. Agradeço especialmente pela oportunidade de poder concluir mais essa importante etapa da minha vida que se trata da conclusão do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas.

Agradeço à minha família, aos meus pais, meu irmão, e à minha esposa por terem me apoiado e sustentado durante os anos de curso, e especialmente na etapa da concepção e desenvolvimento do Trabalho de Conclusão de Curso. Por vezes, todos tiveram que ser insistentes para que eu perseverasse, mas com isso me garantiram a força necessária para não esmorecer durante o processo.

Agradeço também ao Corpo Docente do Departamento de Informática da FEMA, em especial ao meu Professor Orientador Dr. Almir Rogério Camolesi pela paciência, confiança e respaldo no processo de desenvolvimento do TCC, e aos Professores Doutores Alex Poletto, Luiz Carlos Begosso, e Luiz Ricardo Begosso pelos conselhos, e direcionamento durante o curso, bem como por me auxiliar no Processo da implementação da Bolsa de Estudos do Programa CsF para um ano de estudo no Canadá.

Gostaria de agradecer aos amigos que me auxiliaram durante os anos do curso: Gabriel Galli, e Leonardo Bastelli. Além deles gostaria de agradecer aos amigos Flávia Salmeron pelos conselhos e encorajamento, bem como ao Flávio Queiroga e ao Leandro Ikezili pelo auxílio que me deram no que diz respeito à concepção e revisão técnica do projeto.

“Estou convencido que metade do que separa os empreendedores bem-sucedidos de todo o resto é a pura perseverança. ”

Steve Jobs

RESUMO

Este texto versa sobre a realidade das startups no atual contexto da sociedade, descrevendo sobre a sua importância e os seus desafios. Dado este contexto, a proposta desta monografia é o desenvolvimento de um sistema que sirva como uma plataforma de compartilhamento de ideias e experiências entre pessoas que se interessam por empreendedorismo e querem iniciar ou investir numa startup.

O aplicativo será disponibilizado em plataforma web (em formato de *website*), dando aos usuários a capacidade de compartilhar as suas experiências e ler as demais histórias compartilhadas no Portal, bem como permite que usuários se conectem entre si de acordo com os seus interesses, e ainda permite que as empresas sejam beneficiadas por um modelo de *crowdfunding* (financiamento coletivo) no qual as melhores histórias passam a ser eletivas a esta condição.

Para este desenvolvimento, utiliza-se o modelo de desenvolvimento web conhecido por *Full Stack Web* Development no qual um único desenvolvedor conhece toda a cadeia de desenvolvimento. As tecnologias empregadas foram: Node.JS (*server side*), JavaScript (*client e server side*), Angular JS, HTML, CSS, e Bootstrap (*client side*), e para o Banco de Dados usou-se o Mongo DB.

Palavras-chave: Startup; Node.JS; Angular JS; Mongo DB.

ABSTRACT

This text refers to the startup's current context in the society, describing its importance and challenges. Given this context, the proposal of this monography is to develop a system which serves as a platform of sharing ideas and experiences among people which have interest about entrepreneurship and want to begin or invest in a startup.

The app will be provided as a web platform (in a website format), giving the users the capacity of sharing experiences and read others stories shared in the Portal, as well as allowing that users connect among themselves according to their interests, and also allowing that companies can be benefited by a crowdfunding model in which the best stories become elective to this condition.

For this development, it was used the Full Stack Web Development model in which one single developer knows all the development chain. The Technologies used were: Node.JS (server side), JavaScript (client and server side), Angular JS, HTML, CSS, and Bootstrap (client side), and for the Database was used the Mongo DB.

Keywords: Startup; Node.JS; Angular JS; Mongo DB.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API – Application Programming Interface

APK – Android Package

CRUD – Create, Read, Update, and Delete

CSS – Cascading Style Sheets

HTML – Hyper Text Markup Language

JSS – JavaScript

JSON – JavaScript Object Notation

MVC – Model View Controller

NoSQL – Not Only Structure Query Language

SDK – Software Development Kit

SQL – Structure Query Language

SUMÁRIO

1. INTRODUÇÃO

No contexto econômico e social atual, no qual as grandes companhias estão tendo que se reinventar, buscando uma constante melhoria de processos para aumentar a produtividade e diminuir custos, se apresentam para as pessoas do meio a oportunidade e o desafio do empreendedorismo[1].

O mundo tem passado por momentos de acentuada instabilidade econômica, combinados de uma maneira quase que discrepante com uma evolução tecnológica sem presentes na história da humanidade.

Estes elementos são essenciais para a formação do contexto propício para a expansão do cenário de startups de forma tão acelerada como se tem visto e isso em escala global.

Evidente que existem algumas regiões onde a presença e o desenvolvimentos dessas empresas têm maior relevância e onde parece ser ainda mais propício para esse crescimento. Como exemplo mais destacado internacionalmente pode-se citar o Vale do Silício situado no estado da Califórnia nos Estados Unidos da América. No Brasil vê-se o aparecimento de startups nas mais diversas regiões, especialmente no Sul, Sudeste, Centro-Oeste e em alguns Estados do Nordeste. Porém a concentração maior ainda tem sido em São Paulo e no Rio de Janeiro, conforme publicação do Programa Startup Brasil do Ministério da Ciência, Tecnologia e Inovação.[2]

[1] http://www.empreendedoresweb.com.br/negocios-que-crescem-em-tempos-de-crise/; http://www.tudomudou.com/2011/12/15/empreendedorismo-numa-crise-economica-o-melhor-emprego-podera-ser-aquele-que-cria-para-si-proprio/

[2] http://startupbrasil.org.br/wp-content/uploads/2014/11/book_demoday_startupbrasil_2014.pdf

Neste movimento de empresas que começam pequenas e têm por objetivo um rápido crescimento através da concepção e desenvolvimento de produtos e serviços inovadores, encontra-se vários exemplos de sucesso, a saber: SpaceX (empresa do ramo da astronomia focado no lançamento de foguetes tripulados e não-tripulados para transporte de cargas e pessoas até a estação espacial internacional – MIR, ou até o espaço sideral), a Tesla (companhia voltada para a criação e produção de carros elétricos de alta-performance e desempenho dispondo de todas as vantagens que um veículo elétrico pode possuir), ambas do visionário sul-africano Elon Musk e sediadas em Los Angeles, Califórnia, EUA[3]; há ainda a Uber (companhia que atua no setor de serviços, promovendo uma revolução na forma como as pessoas se deslocam nos grandes centros, passando a depender cada vez menos de automóveis particulares, e promovendo uma concorrência com o serviço tradicional dos taxistas fazendo com que a qualidade geral do serviço aumente, beneficiando os usuários de táxis e do próprio Uber), de propriedade de Travis Kalanick, e sediada em San Francisco, Califórnia, EUA; como exemplos brasileiros cabe citar o Easytaxi, o 99 táxis, Ifood, e especialmente o Nubank (companhia classificada como sendo uma Fintech, atuando no mercado financeiro empregando alta tecnologia e extrema inovação no que diz respeito à maneira como o serviço é prestado para os seus clientes), sediada no bairro de Pinheiros em São Paulo, e na mesma linha o Banco Original (primeiro banco brasileiro 100% digital no qual os clientes dependem apenas do seu smartphone para realizar a abertura e manutenção da sua conta bancária, não tendo nunca a necessidade de visitar uma agência bancária para a obtenção de qualquer serviço necessário) sediado na Avenida das Nações Unidas nas margens do Rio Pinheiros em São Paulo.

[3] VANCE, Ashlee. "Elon Musk - Como o Ceo Bilionário da Spacex e da Tesla Está Moldando Nosso Futuro". Rio de Janeiro: Intrínseca, 2015.

Acima citamos uma ínfima parcela de casos de sucesso de empresas que começaram com um grande sonho e objetivo a ser conquistado e que estão conseguindo fazê-lo. Porém é preciso dizer que startups e pessoas querendo empreender são cada vez mais comuns e existem infinitos outros casos de sucesso.

Porém, da mesma forma que existem as histórias que deram e dão certo, há aquelas em que as companhias não conseguem se estabelecer da maneira conforme planejado pelos seus idealizadores. E os motivos para essa inviabilidade de permanência no mercado são inúmeros e não tão divulgados quanto nas ocasiões em que o contrário ocorre. Por essa razão, muitos dos desafios e percalços que serão enfrentados pelos empreendedores não são conhecidos previamente.

Ainda compondo esse quebra-cabeça das startups há ainda o papel primordial do capital necessário para fazer girar as engrenagens da empresa e tornar as ideias reais. Aí aparece a figura do grande investidor ávido por informações e sempre buscando as melhores e mais inovadoras companhias para poder aportar o seu dinheiro e fazê-lo render. Porém, o grande investidor não é o único a desempenhar o papel do financiador das startups, visto que o financiamento dos novos produtos e serviços pode também ser feito por pessoas comuns que se identificam com uma ou outra ideia é investem às vezes uma pequena parcela de dinheiro que por vezes não chega nem a 50 dólares, mas que com essa pequena contribuição passam a se sentir “donas” do projeto. Essa prática, conhecida por *crowdfunding* é cada dia mais comum e adotada internacionalmente, contando inclusive com a existência de sites especializados em viabilizar isso.

1.1. OBJETIVOS

Os objetivos deste trabalho são o de desenvolver uma aplicação *web* em formato de website que sirva como portal de informações, histórias e experiências de startups bem-sucedidas e também daquelas que falharam, além de servir como um ponto de *networking* (rede de contatos) entre os interessados em inovação, e ainda como um repositório bastante seletivo para financiamento compartilhado das melhores histórias/empresas, o *crowdfunding* (financiamento coletivo). Para o desenvolvimento desta aplicação serão utilizados HTML, Bootstrap, CSS, JavaScript, e *frameworks* de JavaScript como o Angular JS e o Node.JS. Para armazenar as informações será construído um modelo não relacional de banco de dados utilizando-se o MongoDB.

Com isso pretende-se desenvolver uma aplicação bastante funcional, estável, facilmente escalável e com alta disponibilidade para uma quantidade crescente de acessos tão logo esses acessos forem se avolumando. Como veremos à frente, as tecnologias empregadas são desenvolvidas de forma a conferir as características despejadas para o website.

Além da aplicação web desenhada para ser acessada pelos mais variados navegadores comercias disponíveis atualmente (Microsoft Edge, Google Chrome, Mozilla Firefox, Safari, Opera e outros) e também para acesso nos navegadores similares, porém a partir de dispositivos móveis, pretende-se desenvolver uma aplicação móvel a partir do código criado previamente para a versão web, com o auxílio dos frameworks de desenvolvimento *mobile,* Ionic[4] e Cordova[5].

[4] http://ionicframework.com/docs/guide/preface.html
[5] https://cordova.apache.org/

Ao término do presente estudo espera-se atestar a viabilidade das tecnologias empregadas para o desenvolvimento da aplicação, além de compreender melhor sobre o mercado e o contexto das startups e do *networking* entre o seus profissionais e idealizadores.

1.2. FINALIDADE

Este projeto destina-se a estudar e promover a disseminação de informações de nível elementar sobre novas tecnologias como o Node.JS[6], o Angular JS[7], o banco de dados MongoDB e os frameworks para desenvolvimento móvel Ionic e Cordova. Paralelo a isso, tem-se ainda como finalidade atestar a eficácia destas tecnologias no atendimento das demandas de uma aplicação Web que deve começar pequena, mas se expandir com o passar do tempo. Neste ponto poderá ser verificado sua aderência à necessidade de escalabilidade.

Há ainda a intenção de demonstrar que o conhecimento das tecnologias empregadas neste trabalho pode habilitar ao profissional desenvolvedor conhecer toda a cadeia do processo de construção de uma aplicação Web, desde o *front-end* até o *back-end* da mesma. Nesse ponto deve-se apresentar o conceito de *Full stack Web Developer* (desenvolvedor que atua tanto com *front-end*, quanto com *back-end*).

Por fim, esta aplicação foi idealizada afim de criar um canal centralizado de informação, debate e interação e interessados em empreender nestes dias em que a informação se mostra como um dos recursos mais fundamentais para a economia global em que vivemos. Ao término do seu desenvolvimento, o que pode eventualmente

[6] https://nodejs.org/en/
[7] https://angularjs.org/

ultrapassar os limites do presente projeto, espera-se ter uma aplicação capaz de entregar essas funcionalidades de maneira a entusiasmar os usuários que venham a utilizá-la.

1.3. PÚBLICO ALVO

As pessoas a que esta aplicação se destina são aquelas que querem conhecer mais sobre o universo das startups, aquelas que querem fazer pesquisas de mercado mais assertivas antes de empreender num negócio próprio. Pessoas que já estão no mercado, mas querem fortalecer a experiência da sua companhia podem ser interessar pelo website afim da possibilidade de poder se conectar com outros profissionais de qualquer parte do planeta.

A aplicação destina-se ainda àqueles que querem encontrar uma destinação para investimento do seu dinheiro. Essas pessoas poderão encontrar as ideias mais discutidas e debatidas do website e a partir de então poderão participar da discussão e até mesmo financiar as startups detentoras das melhores ideias armazenadas no repositório.

Este projeto pode ainda ser interessante para estudantes e profissionais que queiram conhecer e aprimorar seus conhecimentos nas tecnologias empregadas no desenvolvimento da aplicação e estudadas através do texto que vem a seguir.

1.4. ESTRUTURA DO TRABALHO

Esta monografia está organizada e dividida em capítulos e seções que devem cobrir os elementos fundamentais das tecnologias escolhidas para o desenvolvimento, e elementos mais protocolares que regem sobre a execução do trabalho e o fluxo do mesmo,

bem como sobre uma descrição mais detalhada da implantação do sistema e uma conclusão oportuna.

O primeiro capítulo consiste na Introdução do trabalho (o presente capítulo), enquanto o segundo capítulo deve versar sobre as tecnologias empregadas para a implementação da aplicação proposta.

O terceiro capítulo será utilizado para detalhar a proposta da aplicação a ser desenvolvida, incluindo a especificação do sistema e diagramas de UML (Casos de Uso, Diagrama de Atividades, Diagrama de Sequências, Diagrama de Classes, e Diagrama de Modelagem de Entidades e Relacionamento). E este mesmo capítulo será utilizado para conferir uma visão mais gerencial acerca do desenrolar do projeto. Nele estará descrita a metodologia usada no desenvolvimento do projeto. Será demonstrado o WBS (Work Breakdown Structure), e modelos de documentos que serão usados para dar uma visão de status do andamento das atividades do projeto. Para essa finalidade de gestão do projeto será utilizada a aplicação online *"Asana"* (ferramenta *online* de Gestão de Projetos).

O quarto capítulo mostrará detalhes do desenvolvimento da aplicação, mostrando trechos dos códigos e capturas de tela que sejam relevantes para a ilustração de como as tecnologias utilizadas servem como suporte para o desenvolvimento do sistema.

Por fim, o trabalho se encerra com um capítulo conclusivo para validar as proposições e os resultados alcançados. De acordo com a conclusão do trabalho, o projeto poderá vir a se expandir com a implantação de mais funcionalidades ou então poderá ser estagnado ou ser revisto para uma eventual continuidade.

2. TECNOLOGIAS UTILIZADAS PARA O DESENVOLVIMENTO

A aplicação *Ideas on Start.UP* será desenvolvida fundamentalmente com o uso do JavaScript, suportado do lado do servidor pelo *framework* Node.JS, e do lado do *front-end* será utilizado o *framework* Angular JS afim de garantir o desenvolvimento de uma aplicação *single page*, além Bootstrap, CSS, e HTML para garantir o *layout*.

Afim de migrar a aplicação da plataforma web também para uso em dispositivos móveis serão usados os *frameworks* Ionic e Cordova.

Para o armazenamento das informações constantes da aplicação será utilizado o banco não-relacional MongoDB, o qual será apresentado com maiores detalhes a seguir.

2.1. JAVASCRIPT

O JavaScript foi desenvolvido inicialmente pela equipe do Netscape no ano de 1995 inicialmente com o nome de Livescript. Na sequência dos fatos, o desenvolvimento da linguagem ganhou um importante impulso quando a Netscape estabeleceu uma parceria com a Sun Microsystems para continuar esse projeto. Foi nesse momento que a linguagem ganhou o nome de JavaScript, isso já em 1996.[8]

E então, em novembro de 1996, a Netscape submeteu a tecnologia para o ECMA Internacional para que eles pudessem cuidar da especificação de padronização da linguagem. A partir de então não somente o navegador Netscape poderia utilizar o JavaScript para interpretar e renderizar o conteúdo das páginas web, mas também os outros navegadores disponíveis no mercado naquele momento.

[8] https://www.w3.org/community/webed/wiki/A_Short_History_of_JavaScript

O JavaScript é uma linguagem de programação bastante popular no meio dos desenvolvedores web. É importante destacar que essa não é a principal linguagem no desenvolvimento de páginas para a Internet, mas trata-se de um elemento fundamental para conferir dinamismo para as referidas páginas. O HTML e o CSS não conseguem executar determinados efeitos ou possibilitar determinados resultados esperados e é para esses pontos que o JavaScript é utilizado na internet.

Algumas das aplicações que o JavaScript pode ter são:

- **Interatividade:** permite que o a página e o seu conteúdo não sejam estáticos como costuma ser quando feitos somente com HTML. A página passa a ter maior interatividade com o usuário.
- **Validação de formulários:** capaz de realizar validação de dados digitados em formulários com regras de negócio ou dados previamente armazenados no banco de dados.
- **Controle de comportamento da página:** Os scripts podem controlar o comportamento de toda a página ou então de alguns dos elementos que constam da mesma de acordo com alguma interação do usuário (clique de *mouse*, por exemplo) ou então de acordo outros disparadores por exemplo um *timer*.
- **Personalização de aparência da página pelo usuário:** Permite que o usuário latere configurações de visualização da página (cor e tamanho da fonte, cor do plano de fundo, etc).
- **Dinamismo de conteúdo:** capaz de alterar parte do conteúdo da página sem ter a necessidade de recarregar a página toda para que as alterações de conteúdo sejam exibidas.

Além dessas aplicações, as principais características do JavaScript são:

- ***Client side*:** essa linguagem foi desenvolvida para ser executada do lado do cliente em sistemas de arquitetura *client-server*.
- **Interpretada:** normalmente é interpretada no navegador do computador do cliente, não tendo a necessidade de ser previamente compilada.
- ***Case Sensitive*:** esta linguagem é capaz de diferenciar letras maiúsculas de minúsculas, então é necessário respeitar essas diferenciações no momento de nomear e chamar variáveis, por exemplo.
- **Fracamente tipada:** não se define o tipo de dados que a variável armazena no momento em que esta é declarada. Com isso pode-se armarzenar qualquer tipo de dados numa mesma variável.
- **Dinamicamente tipada:** a variável passa a ser do tipo *string* assim que um conteúdo desse tipo é atribuído à variável, ou então passa a ser *integer* assim que um conteúdo desse tipo é armazenado na variável.
- **Estruturada ou Orientada à Objeto:** a linguagem foi incialmente criada seguindo o paradigma de programação estruturado, porém atualmente suporta também o paradigma de Orientação à Objetos (OO) extremamente popular e difundido entre os programadores da atualidade.
- **Acionamento:** os scripts podem ser acionados automaticamente por algum evento ou então são acionados diretamente quando encontrados pelo navegador assim que esse estiver lendo a página HTML.

2.2. NODE.JS

O Node.JS é na realidade um *framework* JavaScript capaz de fazer com que a linguagem JavaScript possa ser também executada e interpretada do lado do servidor em sistemas de arquitetura *client-server*.[9]

O Node.JS foi desenvolvido incialmente no ano de 2009 por Ryan Dahl. Neste mesmo ano o Node.JS foi apresentado na JSConfEuropa. A ideia de Dahl para criar o Node.JS surgir ao observar a barra de progresso de *upload* do Flickr. O navegador não sabia quanto do arquivo tinha sido enviado e tinha que ficar requisitando essa informação continuamente do Web server.

Citando o autor William Moraes: “O Node.JS é uma poderosa plataforma para aplicações, para construir fácil e rapidamente aplicações de rede escaláveis, e que utiliza a *engine* JavaScript *open source* de alta performance do navegador Google Chrome, o motor V8, que é escrita em C++”.

O Node.Js diferencia-se por exemplo da arquitetura do PHP, tendo em vista que neste o PHP funciona como linguagem de *server side* e tem o Apache como servidor Web, enquanto que no caso do Node.JS, o servidor Web e a aplicação *server side* são ambos escritos fazendo uso da linguagem JavaScript.

O Node.JS é construído sob uma arquitetura baseada em eventos e capaz de trabalhar com requisições assíncronas, não bloqueante. Isso faz com que o sistema seja leve e bastante eficiente. Essas são soluções ideais para resolver problemas de tráfego intenso de usuários e requisições na rede em que o sistema esteja sendo executado.

[9] http://www.ibm.com/developerworks/cloud/library/cl-nodejscloud/index.html

Por meio dessas características podemos escolher o Node.JS como o *framework* de desenvolvimento quando queremos garantir alta disponibilidade e escalabilidade para o sistema desenvolvido.

2.2.1. Instalação e documentação do Node.JS

Para instalar o Node.JS na máquina basta que o programador acesse o Portal oficial do *framework*, no endereço https://nodejs.org/. Uma vez neste endereço, basta escolher a versão desejada e o arquivo de instalação compatível com o sistema operacional da máquina e seguir o passo a passo da instalação. Depois disso, convém que o profissional configure as variáveis de ambiente da máquina para que o Node.JS possa ser executado a partir de qualquer diretório do computador. Para esta tarefa de configuração das variáveis de ambiente, existem diversos tutoriais em vídeo ou escritos que podem ser acessados a partir de uma rápida pesquisa no Google.

No mesmo portal a partir do qual a instalação é efetuada, pode-se encontrar uma extensa documentação em inglês que discorre sobre os mais variados pontos de interesse para aqueles que farão uso no Node.JS em seus projetos de desenvolvimento, tais como, por exemplo, os módulos, os eventos, os erros, dentre outros.

2.2.2. Programação síncrona e assíncrona

Conforme visto previamente, o Node.JS utiliza-se do benefício do desenvolvimento com requisições assíncronas ao servidor da aplicação.

A diferença que caracteriza o padrão síncrono e assíncrono é que no padrão síncrono uma operação ocorre após o término da outra. Com isso, a aplicação não executa nenhuma linha de código antes que a primeira requisição tenha sido finalizada totalmente.

Esse é um comportamento também conhecido como “bloqueante”. Já no padrão assíncrono, as operações podem começar independente do término das operações anteriores. Com isso as requisições têm autonomia uma em relação as outras e as linhas de código vão sendo executadas pela aplicação sem seres bloqueadas.

Para a garantia do sucesso da execução das requisições (métodos e/ou funções), todas têm como parâmetro uma função de *call-back* para que não se percam em meio a execução das demais linhas do código que contenham outros métodos e funções que também já foram iniciados.

2.2.3. Orientação a eventos

No navegador existem diversos eventos, tais como o clique do *mouse* em algum elemento, o pressionar de uma tecla, entre outros. No Node.Js os eventos podem ser o recebimento parcial ou total de algum conteúdo num *streaming*, ou então o aviso de uma conexão com sucesso em um banco de dados, ou então justamente o contrário, a falha na conexão com um banco, entre outros.

Nesse contexto de Orientação a Eventos, um objeto (*Subject*) possui uma lista de dependências (*Observers*) e os avisa automaticamente quando alguma mudança de estado acontece.

Essa característica trabalha de maneira complementar à primeira que diz respeito à programação assíncrona, viabilizando a organização das requisições e as respostas para as funções *call-backs*.

2.2.4. Outras características

Além do já apresentado, o Node.Js incorpora ainda outros elementos presentes no JavaScript, tais como Orientação a Objetos, e programação funcional. Isso porque os códigos são escritos em JavaScript e se apoiam no Node.JS para garantir a sua execução do lado do servidor.

Com isso o Node.JS, demonstra-se como sendo um *framework* bastante eficaz e funcional ao se valer do JavaScript suportando-o como uma linguagem de programação para ser desenvolvida e utilizada também do lado do servidor.

2.2.5. Gerenciador de Pacotes Node (*Node Package Manager* – NPM)

Um elemento essencial e de muita relevância no contexto do Node.JS é o NPM (Node Package Manager), ou Gerenciador de Pacotes Node, em português. Trata-se de um repositório gratuito de módulos com incontáveis funções já implementadas em JavaScript.

Funciona como um serviço gratuito e disponível para todos aqueles que possuem o Node.JS instalado nas suas máquinas. Pelo fato de dispor de muitas soluções já implementadas, os desenvolvedores diminuem muito o tempo de desenvolvimento pois encontram muitas coisas que precisam dentro do NPM e podem facilmente incorporá-las ao seu projeto com uma simples instalação do módulo necessário.

Para fazer uma instalação de um módulo existente no NPM, basta que o desenvolvedor abra o *Prompt* de comando ou o Windows PowerShell (se estiver uma máquina com o Windows como sistema operacional), ou então o Terminal (caso esteja com o Linux, ou o Mac OS como sistema operacional). Dentro do Terminal, basta digitar

npm install nome_do_módulo

Feito isso, o npm instalará o módulo solicitado dentro da pasta *node_modules* que será criada automaticamente dentro do diretório em que o comando foi digitado.

Convém dizer que todo o conteúdo de módulos disponíveis para serem baixados está disponível *online* no seguinte endereço https://www.npmjs.com/. Neste portal é possível pesquisar os módulos, ler a documentação existente para cada um deles, comparar os diversos módulos que possuem finalidades similares, etc. Neste portal ainda, é possível que o desenvolver faça o seu cadastro e passe a colaborar com a comunidade enviando os seus códigos e disponibilizando-os como módulos do NPM.

Vale destacar ainda a existência de um menu que traz a documentação do próprio NPM para que os seus usuários possam tirar as suas dúvidas e aprender melhor sobre o funcionamento do repositório.

Tendo todos esses atributos, verifica-se que o NPM é uma ferramenta extremamente útil para aqueles que fazem uso do Node.JS como *framework* de desenvolvimento, facilitando e otimizando o seu tempo de codificação. No entanto, é importante que o profissional seja bastante criterioso na pesquisa pelos módulos que serão instalados e utilizados no projeto. É importante que sejam levados em consideração se o módulo vem sendo mantido ou não, o número de *issues* abertos para o módulo e o número de *issues* que foram encerrados com sucesso, avaliação de testes de performance para diferentes módulos (se disponível), dentre outros elementos. Ao considerar estes pontos, o desenvolvedor terá mais condições de acertar na escolha do melhor módulo a ser incorporado ao seu projeto.

2.2.6. Encerramento da análise do Node.JS

O Node.JS, tal qual foi apresentado até então consiste numa importante tecnologia que proporciona a possibilidade do JavaScript ser usado para desenvolvimento do lado do servidor.

Porém, além disso, a concepção do projeto inicial do Node.JS visava criar um *framework* que primasse pela eficiência do desenvolvimento, e pela garantia de uma aplicação que fosse escalável e com altíssima taxa de disponibilidade (em função da característica de programação assíncrona e de orientação a eventos).

2.3. ANGULAR JS

O Angular JS é outro *framework* JavaScript, sendo que este foi desenvolvido e lançado inicialmente no ano de 2010 pelo Google e continua sendo mantido até o presente pela mesma companhia. Vale destacar que neste ano de 2016 está sendo apresentada a 2ª versão do *framework* denominado como Angular JS 2 (o qual não é objeto de estudo do presente trabalho)[10].

O Angular JS auxilia na criação de aplicações web dinâmicas, especialmente no que é conhecido como *single page application*. Nesse sentido, podemos construir uma aplicação na qual os recursos são carregados dinamicamente conforme a necessidade. A ideia é que toda a complexidade do processamento da aplicação seja executada na máquina cliente, deixando para o servidor a responsabilidade de se comunicar com o banco

[10] https://techcrunch.com/2016/09/14/google-launches-final-release-version-of-angular-2-0/

de dados. O Angular JS auxilia nesta tarefa através do conceito de “injeção” das páginas conforme a necessidade da exibição destas para o usuário.[11]

O Angular funciona grande parte em cima do uso de Diretivas (*Directives*, em inglês) que são elementos como *tags* que são incorporados junto ao código HTML da página. Essas diretivas servem para customizar e dar as funcionalidades desejadas aos elementos sobre os quais se quer trabalhar. Como exemplo dessas, pode-se citar, por exemplo: ng-repeat, ng-view, ng-bind, entre outros. Detalhes destes serão abordados oportunamente no capítulo sobre a implantação do sistema.

2.4. MONGO DB

O Mongo DB é um banco de dados não relacional, também conhecido como NoSQL. Esta sigla vem do inglês *Not only SQL*, traduzindo-se para Não Somente SQL. Isso significa que essa arquitetura permite organizações mais complexas dos dados e informações armazenados, mas também permite buscas baseadas no modelo SQL tradicional.[12] Tal característica garante um banco mais versátil e capaz de armazenar informações de maneira mais simples e até mesmo intuitiva para o desenvolvedor.[13]

Os Bancos de Dados do tipo NoSQL podem ser estruturados como sendo orientados a documentos, ou então podem ser formados por composições do tipo Chave-Valor, dentre outras alternativas para estruturação.

---

[11] http://matheusazzi.com/single-page-apps/
[12] http://www.informit.com/articles/article.aspx?p=2247310
[13] http://www.developer.com/db/mongodb-nosql-solution-advantages-and-disadvantages.html

No caso do Mongo DB, este é estruturado pelo modelo de Orientação a Documentos. Os dados e informações de uma mesma entidade são armazenados em um documento, e um conjunto de documentos compõe uma coleção. Os objetos de dados que são armazenados num documento são gravados no formato JSON ou BSON.

Os documentos são localizados dentro da coleção através de um id único. Além disso, um documento pode conter sub-documentos, o que normalmente um banco de dados relacionais seria armazenado em uma tabela separada.

As principais vantagens e diferenciais oferecidos pelo Mongo DB são:

- Armazenamento da informação em documentos do tipo JSON;
- Alta Performance, tendo em vista que o Mongo DB se trata de um dos Bancos de Dados com maior performance da atualidade. Isso é importante num contexto de muitos acessos simultâneos, como num site da internet por exemplo;
- Alta Disponibilidade, devido ao modelo de replicação adotado;
- Alta Escalabilidade: a estrutura do Mongo DB torna fácil escalar o banco de maneira horizontal, alocando os dados em múltiplos servidores.

2.5. IONIC E CORDOVA

Nesta seção serão abordados os *frameworks* para desenvolvimento *mobile*, Ionic e Cordova. Essas duas tecnologias são relevantes no contexto de desenvolvimento pois auxiliam neste trabalho conferindo maior eficiência no processo, além de serem responsáveis por disponibilizar a aplicação para multiplataformas.

O Ionic é um SDK completo para desenvolvimento de aplicações *mobile* multiplataforma, desenvolvido a partir de 2013, tendo sido construído sobre o Angular JS e o Cordova, conforme pode-se observar na figura abaixo.[14]

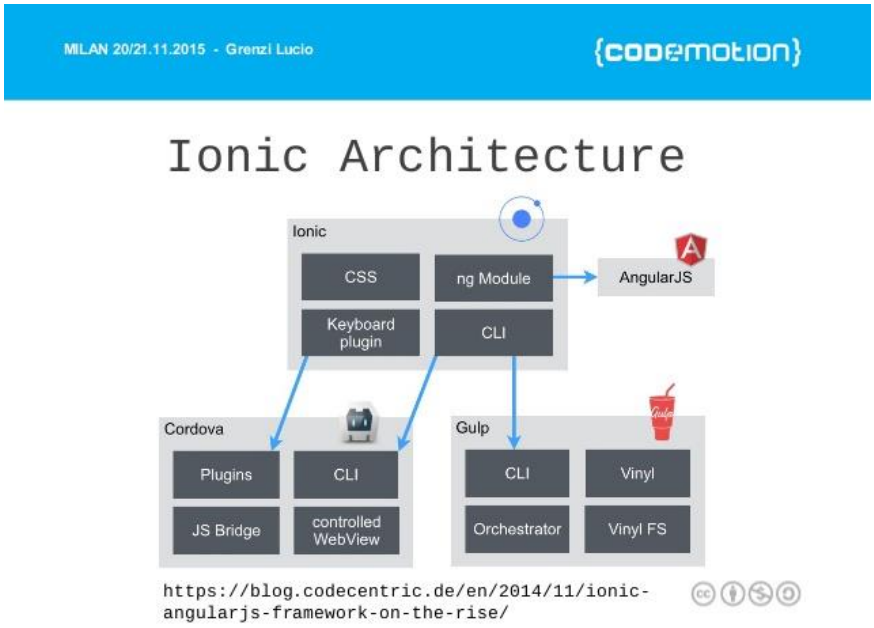

Figura 1: Arquitetura do Ionic[15]

Esta tecnologia é responsável por auxiliar na criação de aplicativos para plataformas Android, iOS, ou Windows Phone. Além disso, pode auxiliar na transformação de uma página HTML com CSS, em uma aplicação para as mesmas plataformas mencionadas previamente. Isso é feito através do acoplamento de algumas *tags* próprias da linguagem junto ao código da página em questão. Com o uso do Ionic, é possível ter acesso a funcionalidades nativas dos aparelhos que estão sendo utilizados (sejam *tablets* ou *smartphones*). Isso significa que a aplicação pode, por exemplo, ter acesso à câmera do dispositivo, usando-a para fazer fotos, ou então pode ser acionada por gestos de

[14] http://www.slideshare.net/Codemotion/lucio-grenzi-use-ionic-framework-to-develop-mobile-application
[15] http://image.slidesharecdn.com/ionic-framework-151124145509-lva1-app6891/95/lucio-grenzi-use-ionic-framework-to-develop-mobile-application-15-638.jpg?cb=1448377119

deslizamento de dedos na tela do dispositivo. Enfim, com o uso do Ionic é possível criar aplicações que, para o usuário, proporcionem uma experiência de aplicação criada através de codificação nativa nas linguagens Java (Android) ou Objective-C (iOS).

O Cordova é um *framework* similar ao Ionic, servindo de suporte para este e também proporcionando o acesso a funcionalidades nativas dos dispositivos. Para se acessar essas funcionalidades é preciso fazer uso das *tags* $cordova disponíveis e documentadas no portal http://ngcordova.com.

Vale ressaltar que o Cordova é um produto já um pouco mais antigo sendo a continuação do framework conhecimento previamente como PhoneGap. O Cordova é de propriedade da Adobe Systems desde o ano de 2011.[16]

2.6. HTML, BOOTSTRAP E CSS

Esta seção é dedicada a discorrer brevemente sobre as três tecnologias que servem como suporte para a construção do sistema e sua disponibilização para uso em formato de página web. São elas o HTML, o CSS e o Boostrap.

O Hyper Text Markup Language (HTML) desenvolvida pelo físico britânico Tim Berners-Lee desde o final do ano de 1990, tendo sido lançada oficialmente no ano de 1993. É importante destacar que em 1889, Tim Berners-Lee escreveu um memorando propondo a utilização de um sistema de internet baseado em hipertexto no qual lançou mão dos principais pontos que viriam a formar a futura linguagem.[17] A partir deste documento já podia notar o que viria pela frente com a linguagem já estruturada.

[16] http://www.adobe.com/aboutadobe/pressroom/pressreleases/201110/AdobeAcquiresNitobi.html
[17] https://www.w3.org/History/1989/proposal.html

O HTML é usado essencialmente para a publicação de conteúdo na web, seja texto, imagens, vídeo ou áudio, entre outras mídias (FERREIRA, EIS, 2014).

Trata-se de uma linguagem que visa a formação de uma grande rede de conteúdo e informação através de “nós” que ligam um documento a outro. Esses nós são os conhecidos links nos quais clicamos para navegar pelas páginas na internet.

O HTML é uma linguagem de marcação (as tags) que os navegadores usam para interpretar e compor os conteúdos a serem exibidos na tela do usuário, desde textos, imagens, até vídeos e outros formatos de mídia comunicativa.

A linguagem vem evoluindo constante desde o seu primeiro lançamento sendo que os seus principais marcos foram: HTML 2.0 publicado em 1995; HTML 3.2 publicado em 1997; HTML 4.0 publicado em 1998, e finalmente o HTML 5 publicado em 2014 como uma recomendação do Consórcio W3C (World Wide Web Consortium que é o responsável global pela manutenção e padronização da linguagem e da World Wide Web).[18]

O CSS, por sua vez teve o seu desenvolvimento a partir da ideia de Tim Berners-Lee (criador do HTML), de que a questão do estilo das páginas deveria ficar a cargo do próprio navegador e não do código HTML. Foi então que Håkon Wium Lie, propôs inicialmente o CSS no ano de 1994, quando trabalhava juntamente com Tim Berners-Lee no CERN (European Organization for Nuclear Research)[19]

[18] https://www.w3.org/2014/10/html5-rec.html.en
[19] https://www.w3.org/Style/LieBos2e/history/

3. ESPECIFICAÇÃO DO SISTEMA E PLANO DO PROJETO

Este sistema foi concebido para servir primeiramente como um portal de interação entre pessoas que se interessam pelo universo das *Startups*. O sistema denominado por "*Ideas on Start.Up*" consiste então num *website* em que o acesso pode se dar de forma controlada (por autenticação), ou não controlada.

O acesso controlado garante que todas as funcionalidades disponibilizadas pela ferramenta sejam utilizadas pelo usuário autenticado. No caso do acesso não controlado por sua vez, o visitante terá para si um cenário com restrição para algumas determinadas funções.

A seguir demonstradas as funções do sistema através de diagramas UML tais como: Casos de Uso, Diagrama de Sequência, e Diagramas de Atividades. Espera-se demonstrar também um pouco da arquitetura do sistema através dos Diagramas de Entidade-Relacionamento e Diagramas de Classe.

3.1. LISTA DE EVENTOS

Para o atendimento dos principais requisitos do sistema, elenca-se a seguinte lista de eventos conforme segue:

1. Efetuar registro no site
2. Efetuar *login* no *site*
3. Compartilhar ideias/histórias
4. Procurar ideias/histórias

5. Realizar comentários
6. Encontrar pessoas
7. Financiar Startups
8. Manter usuários
9. Manter ideias/histórias
10. Manter área de financiamentos
11. Manter comentários

3.2. CASOS DE USO

Na figura encontra-se descrito o cenário genérico dos casos de uso concernente ao Administrador do *website*.

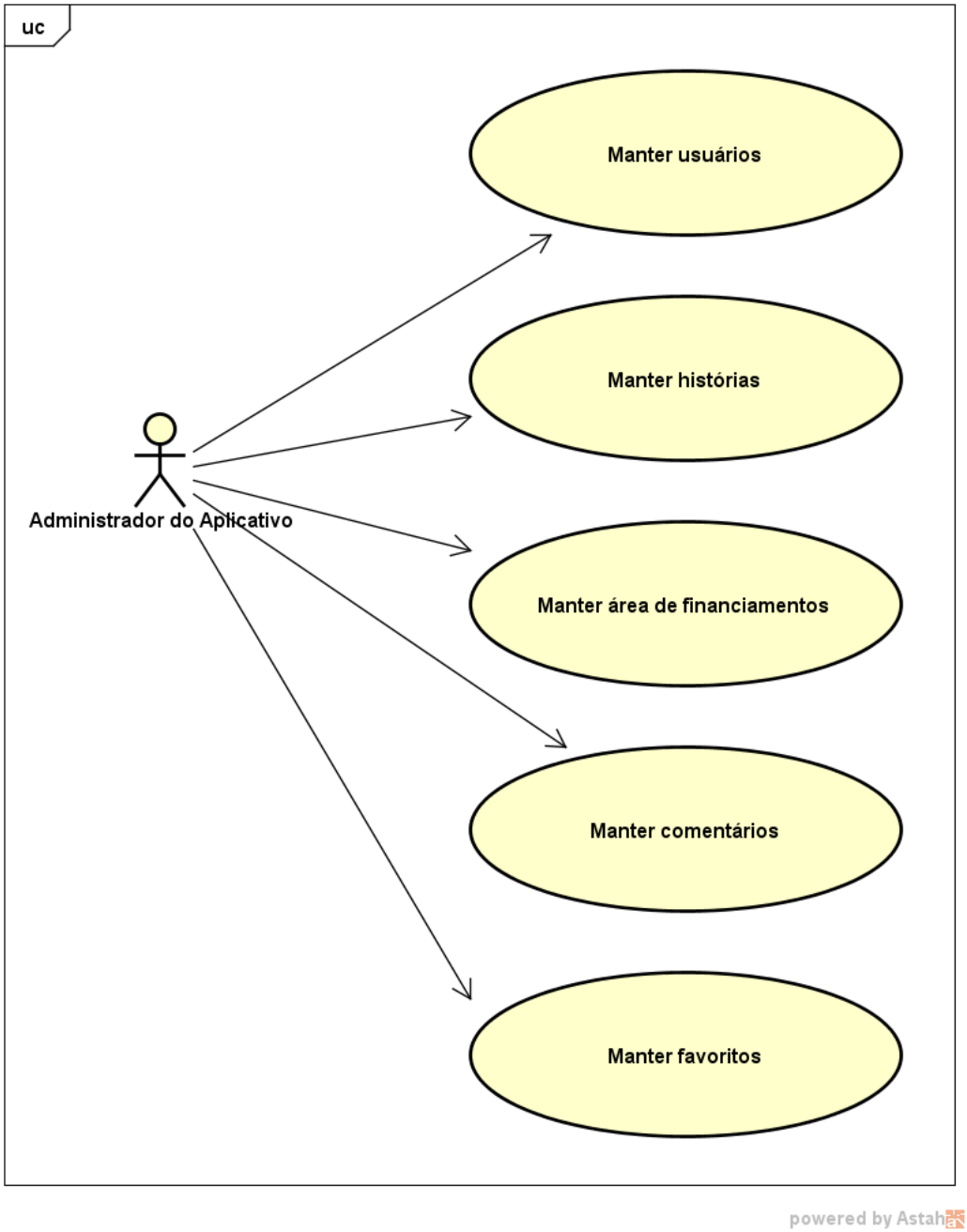

Figura 2: Caso de Uso do Administrador do *Website*

A próxima figura exibe os Casos de Uso do Usuário do Sistema que o acessa pelo *website*.

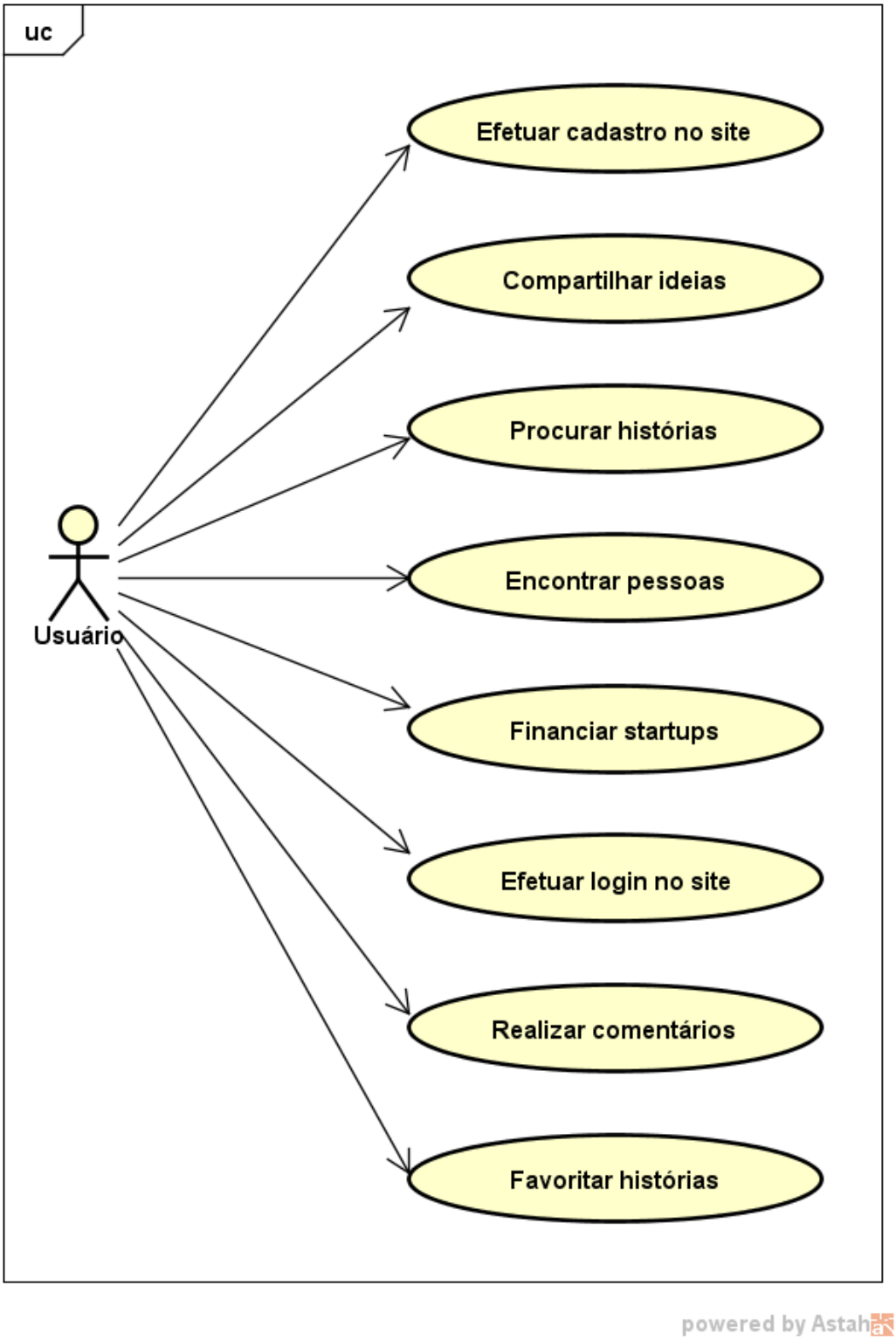

Figura 3: Caso de Uso do Usuário do *Website*

3.3. DIAGRAMA DE ATIVIDADES

Os diagramas a seguir mostram de forma processual as atividades que o usuário pode executar ao acessar o site da aplicação. Vale destacar que o diagrama do cenário completo das atividades é composto por cinco fluxogramas sendo que os fluxogramas do 2 ao 5 são continuações do primeiro, conforme explicitado oportunamente em cada legenda.

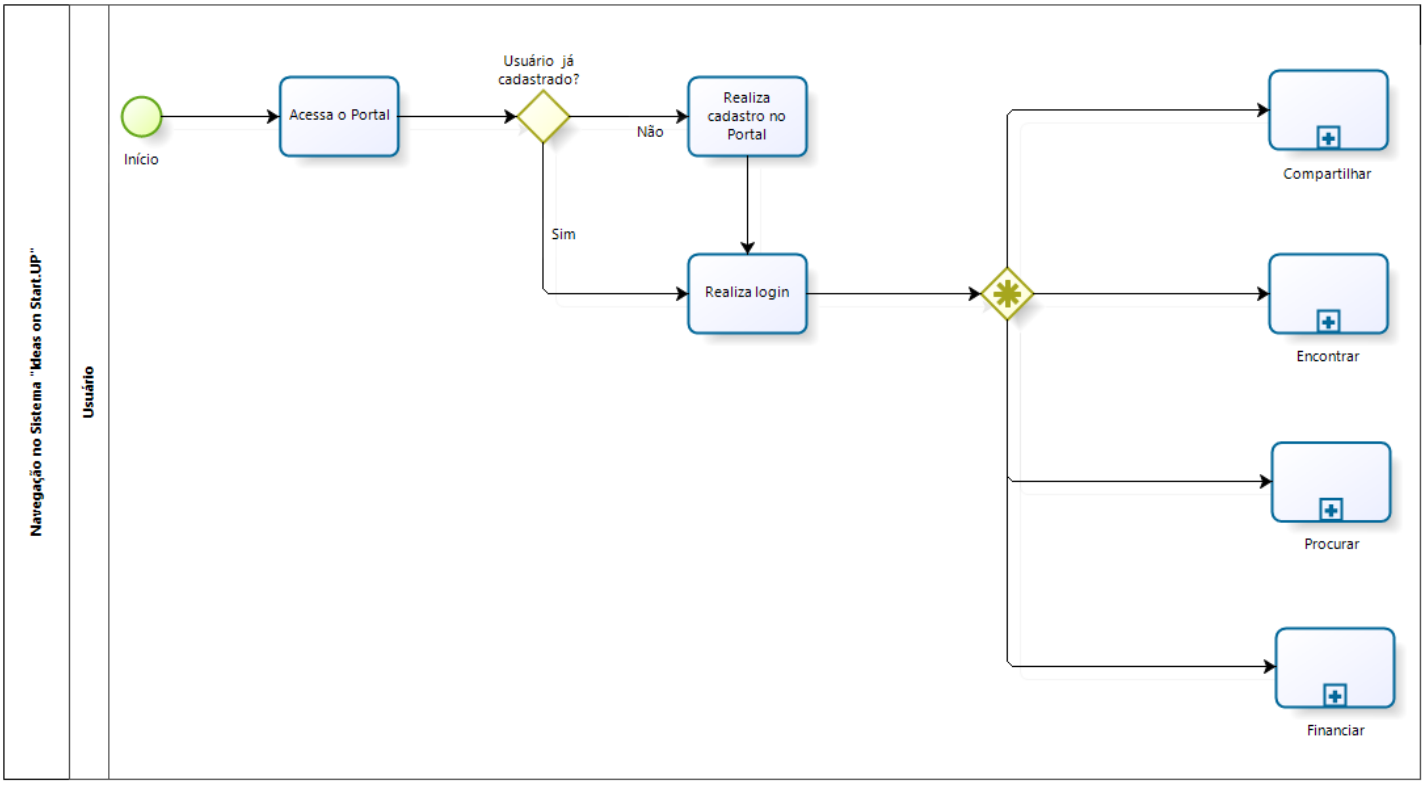

Figura 4: Fluxograma: Navegação no Sistema "*Ideas on Start.UP*"

O próximo fluxograma demonstra o usuário compartilhando uma História/Ideia no Portal.

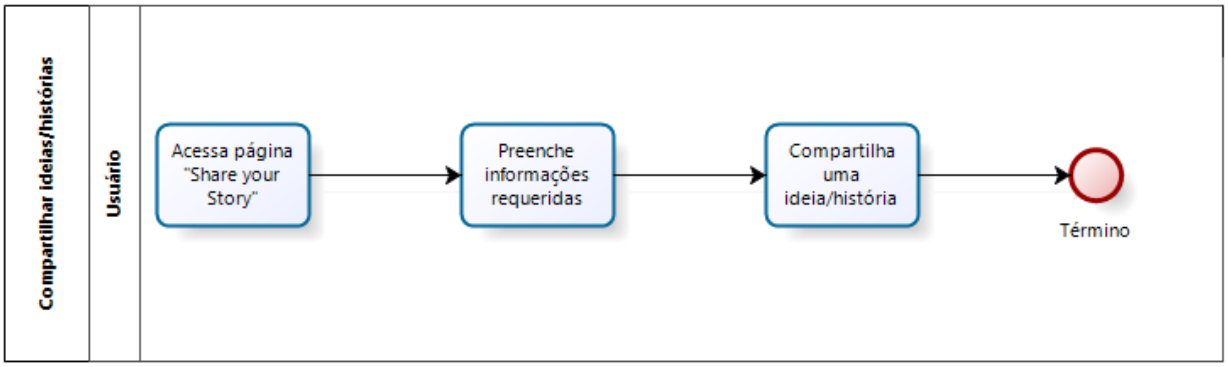

Figura 5: Fluxograma: Compartilhar Ideias/Histórias

O próximo fluxograma demonstra o usuário encontrando pessoas para formar parcerias e trabalhar junto.

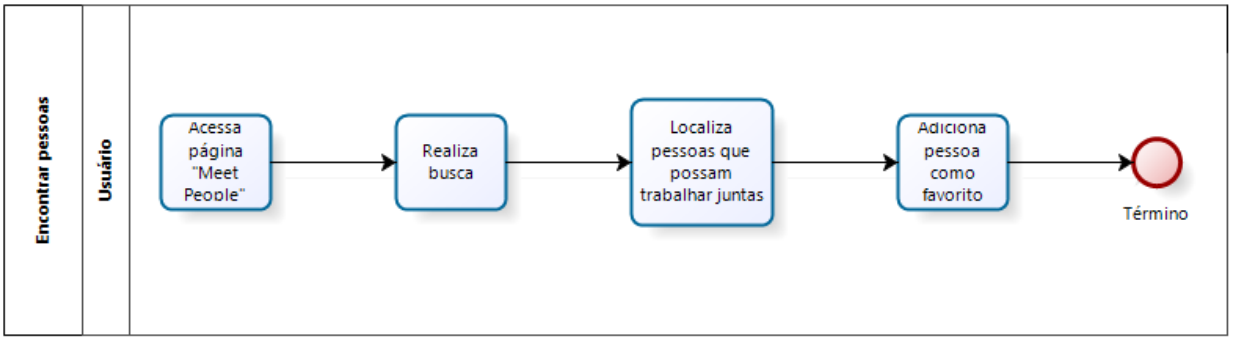

Figura 6: Fluxograma: Encontrar Pessoas

O próximo fluxograma demonstra o usuário procurando Histórias e Ideias, com a possibilidade de favoritar, ou comentar nessas histórias.

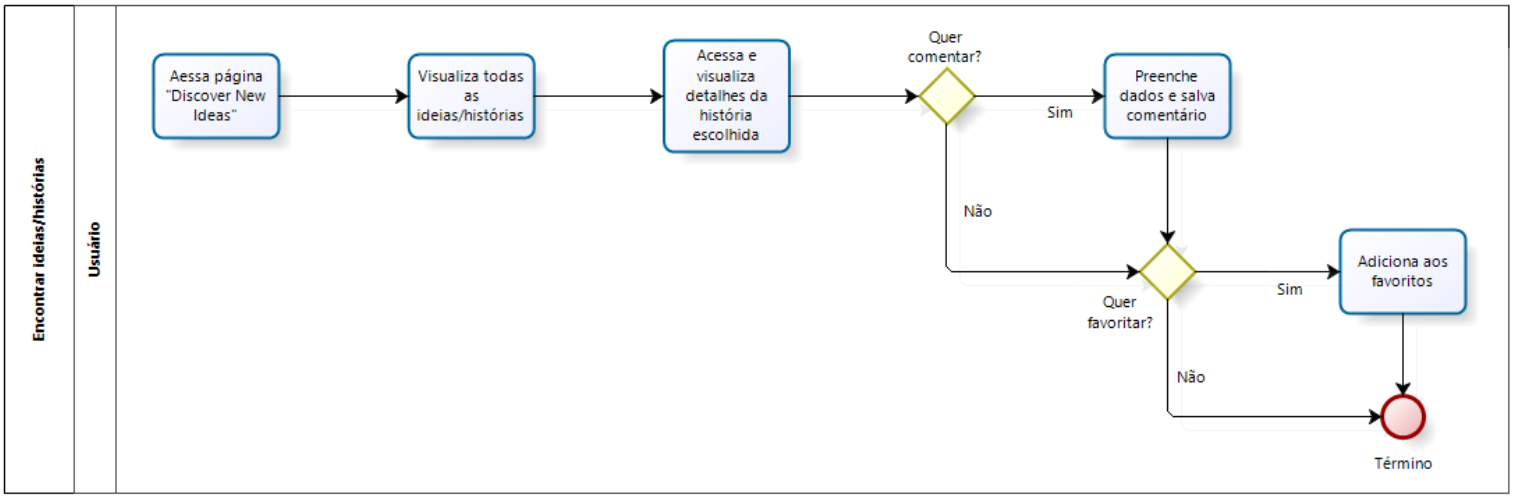

Figura 7: Fluxograma: Encontrar Ideias/Histórias

O próximo fluxograma demonstra o usuário localizando *Startups* que possam ser alvo de investimento e investindo nas mesmas.

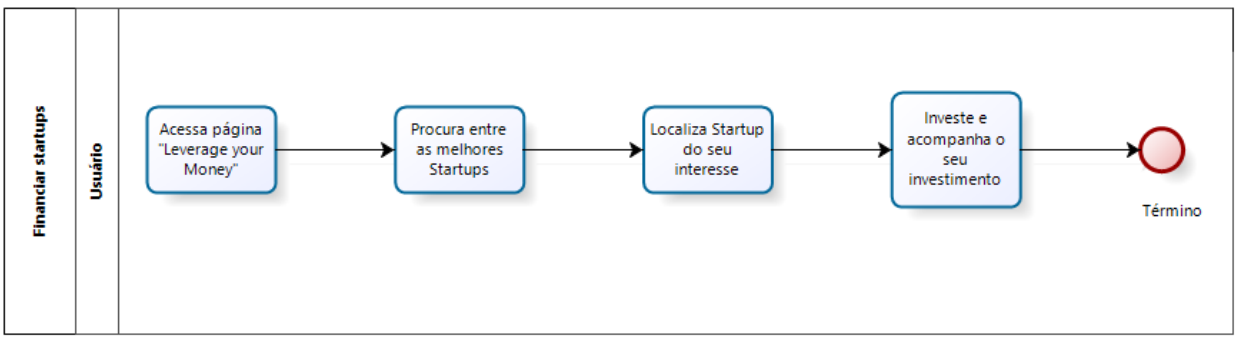

Figura 8: Fluxograma: Financiar Startups

3.4. DIAGRAMA DE CLASSES

A seguir encontra-se o Diagrama de Classes, expondo os principais atributos e os métodos que cada um dos objetos vão dispor dentro do sistema, afim de prover funcionalidade e fácil navegabilidade para os usuários.

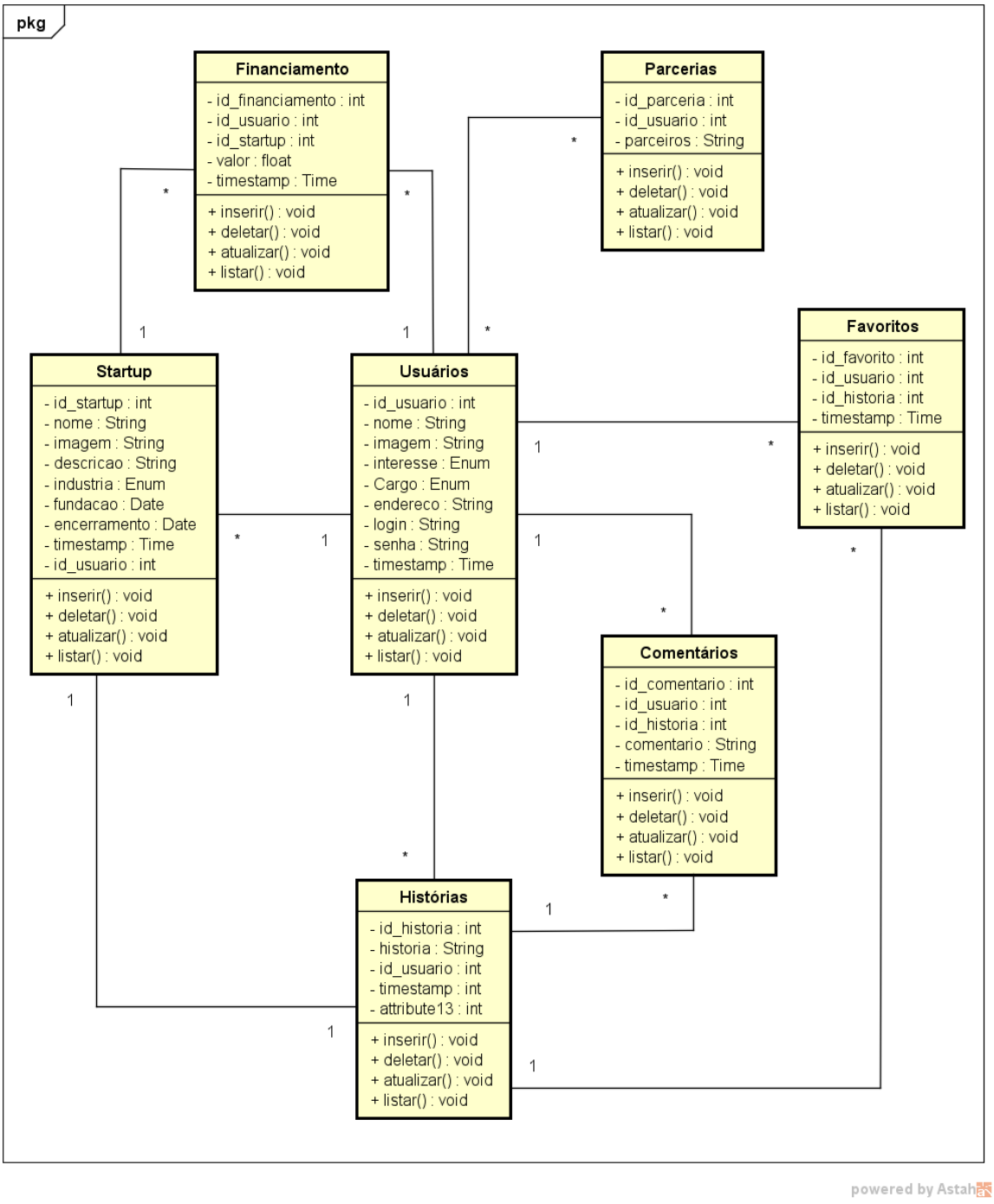

Figura 9: Diagrama de Classes do Sistema

3.5. MODELO DE ENTIDADE E RELACIONAMENTO

A seguir encontra-se o Modelo de Entidade e Relacionamento que serve para representar graficamente o fluxo e o armazenamento dos dados da aplicação. Neste modelo utilizou-se a notação “Pé de Galinha” para ilustrar a maneira tal qual dá-se o relacionamento dos dados do sistema. Vale enfatizar que este modelo gráfico tem apenas o objetivo de documentar o fluxo e armazenamento da informação por parte do sistema, tendo em vista que o mesmo se utiliza de um Banco de Dados NoSQL (modelo que não prima pelo relacionamento entre tabelas).

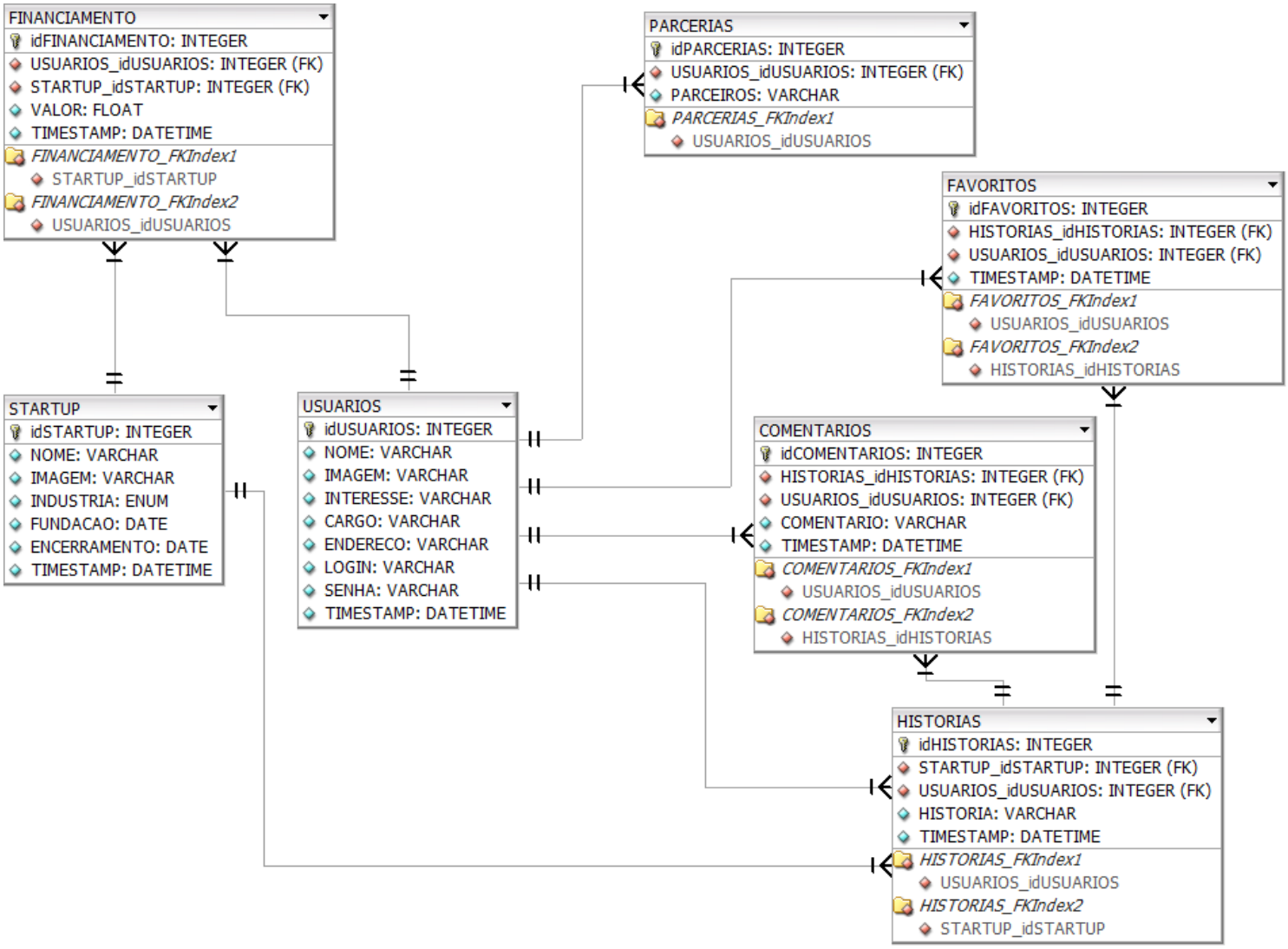

Figura 10: Modelo Entidade e Relacionamento do Sistema "*Ideas on Start.UP*”

3.6. PLANO E ESTRUTURA DO PROJETO

Para uma melhor gestão dos entregáveis desse projeto foram utilizados três t*emplates* de Gerenciamento de Projetos, sendo eles:

- WBS (*Work Breakdown Strucuture*), em português EAP, Estrutura Analítica de Projetos
- Diagrama de Sequenciamento de Atividades
- Cronograma, desenvolvido com o auxílio do Excel, a partir dos entregáveis previstos na Estrutura Analítica de Projeto (EAP). O plano de gerenciamento de tempo será reavaliado quinzenalmente, verificando a aderência ao planejamento inicial.

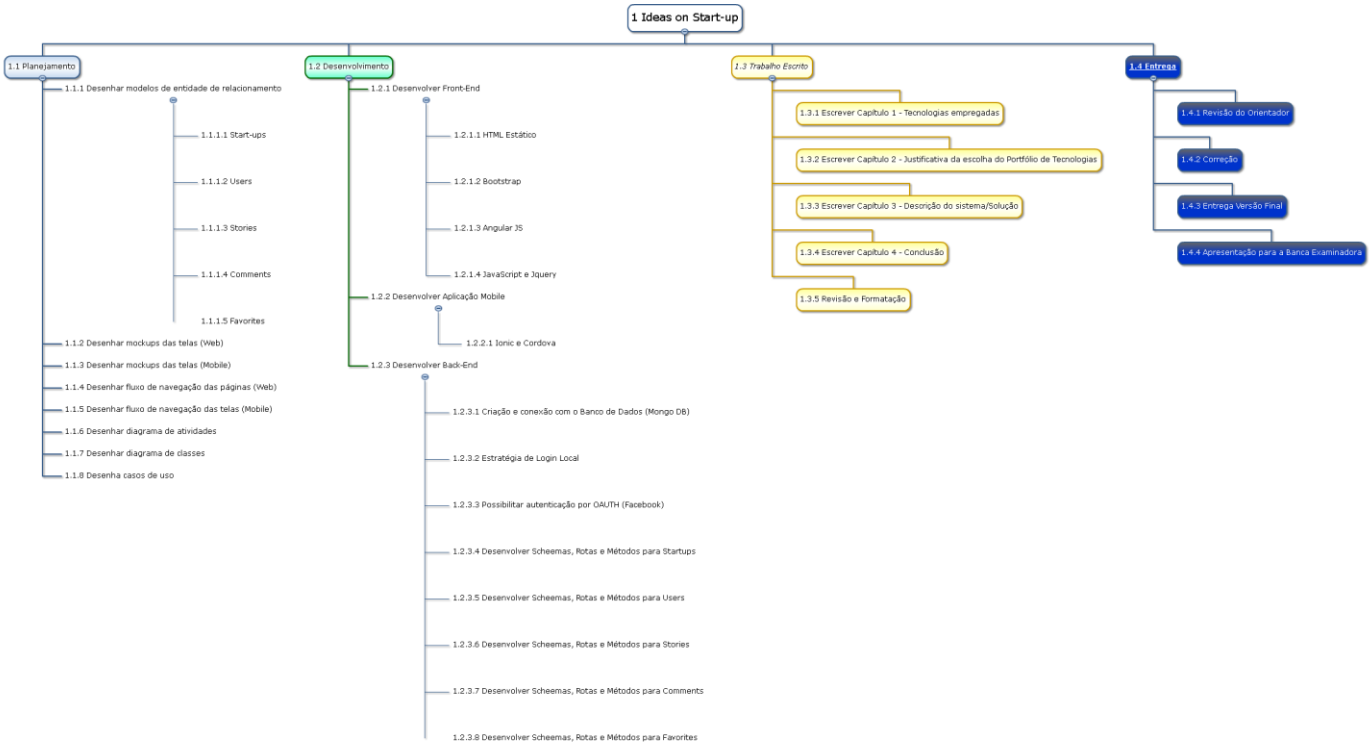

Figura 11: *Work Breakdown Structure* (WBS)

ID	Nome da tarefa	Início	Término	Duração
1	**Ideas On Start-up**	**06/05/2016**	**22/08/2016**	**108**
1.1	**Planejamento**	06/05/2016	30/05/2016	24
1.1.1	Desenhar Modelo de Entidades e Relacionamento	06/05/2016	11/05/2016	5
1.1.1.1	Startups	12/05/2016	13/05/2016	1
1.1.1.2	Users	14/05/2016	15/05/2016	1
1.1.1.3	Stories	16/05/2016	17/05/2016	1
1.1.1.4	Comments	18/05/2016	19/05/2016	1
1.1.1.5	Favorites	20/05/2016	21/05/2016	1
1.1.2	Desenhar Mockups das telas (Web)	22/05/2016	25/05/2016	3
1.1.3	Desenhar Mockups das telas (Mobile)	22/05/2016	25/05/2016	3
1.1.4	Desenhar Fluxo de navegação das páginas (Web)	25/05/2016	30/05/2016	5
1.1.5	Desenhar Fluxo de navegação das páginas (Mobile)	25/05/2016	30/05/2016	5
1.1.6	Desenhar Diagrama de Atividades	25/05/2016	30/05/2016	5
1.1.7	Desenhar Diagrama de Classes	25/05/2016	30/05/2016	5
1.1.8	Desenhar Casos de Uso	25/05/2016	30/05/2016	5
1.2	**Desenvolvimento**	01/06/2016	28/07/2016	57
1.2.1	Desenvolver Front-End	01/06/2016	30/06/2016	29
1.2.1.1	HTML Estático	01/06/2016	20/06/2016	19
1.2.1.2	Bootstrap	01/06/2016	20/06/2016	19
1.2.1.3	Angular JS	01/06/2016	20/06/2016	19
1.2.1.4	JavaScript e Jquery	01/06/2016	20/06/2016	19
1.2.2	Desenvolver Aplicação Mobile	01/06/2016	20/06/2016	19
1.2.1.1	Ionic e Cordova	21/06/2016	01/07/2016	10
1.2.3	Desenvolver Back-End	01/07/2016	28/07/2016	27
1.2.3.1	Criação e conexão com o Banco de Dados (Mongo DB)	01/07/2016	28/07/2016	27
1.2.3.2	Estratégia de Login Local	01/07/2016	28/07/2016	27
1.2.3.3	Possibilitar autenticação por OAUTH (Facebook)	01/07/2016	28/07/2016	27
1.2.3.4	Desenvolver Scheemas, Rotas e Métodos para Startups	01/07/2016	28/07/2016	27
1.2.3.5	Desenvolver Scheemas, Rotas e Métodos para Users	01/07/2016	28/07/2016	27
1.2.3.6	Desenvolver Scheemas, Rotas e Métodos para Stories	01/07/2016	28/07/2016	27
1.2.3.7	Desenvolver Scheemas, Rotas e Métodos para Comments	01/07/2016	28/07/2016	27
1.2.3.8	Desenvolver Scheemas, Rotas e Métodos para Favorites	01/07/2016	28/07/2016	27
1.3	**Trabalho Escrito**	01/07/2016	20/07/2016	19
1.3.1	Escrever Introdução	01/07/2016	04/07/2016	3
1.3.2	Escrever Capítulo 1 - Tecnologias Empregadas	05/07/2016	10/07/2016	5
1.3.3	Escrever Capítulo 2 - Justificativa da escolha do Portfólio de Tecnologias	11/07/2016	13/07/2016	2
1.3.4	Escrever Capítulo 3 - Descrição do sistema/Solução	14/07/2016	17/07/2016	3
1.3.5	Escrever Capítulo 4 - Conclusão	18/07/2016	20/07/2016	2
1.3.6	Revisão e Formatação	18/07/2016	20/07/2016	2
1.4	**Entrega**	01/08/2016	22/08/2016	21
1.4.1	Revisão do Orientador	01/08/2016	10/08/2016	9

1.4.2	Correção	01/08/2016	10/08/2016	9
1.4.3	Entrega Versão Final	20/08/2016	22/08/2016	2
1.4.4	Apresentação para a Banca Examinadora	22/08/2016	22/08/2016	0

Tabela 1: Cronograma de atividades

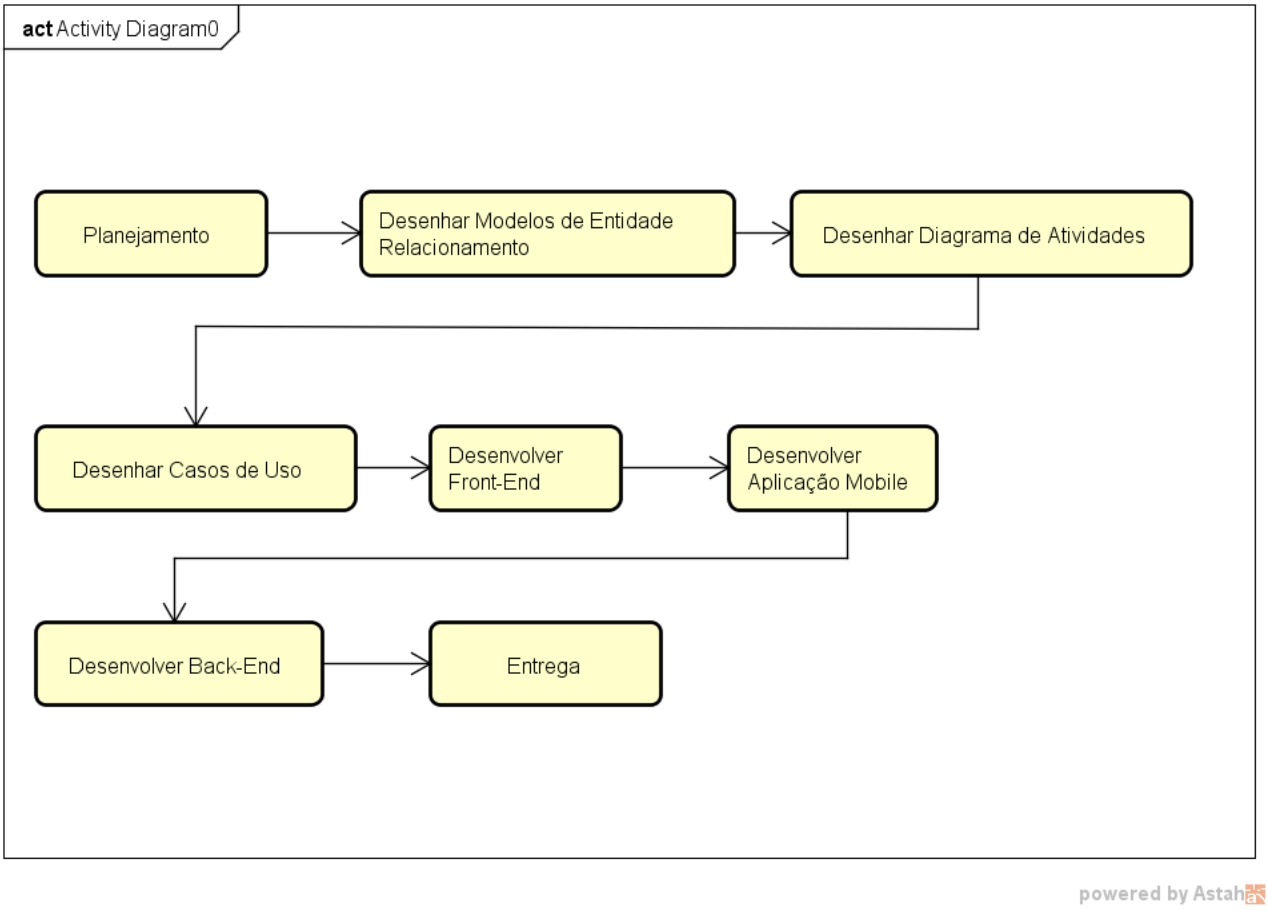

Figura 12: Diagrama de Sequenciamento de Atividades

4. DESENVOLVIMENTO DA APLICAÇÃO

Para a realização desse desenvolvimento foi utilizado o Editor de Texto para códigos conhecido como *Sublime Text* 2, conforme vê-se na figura abaixo:

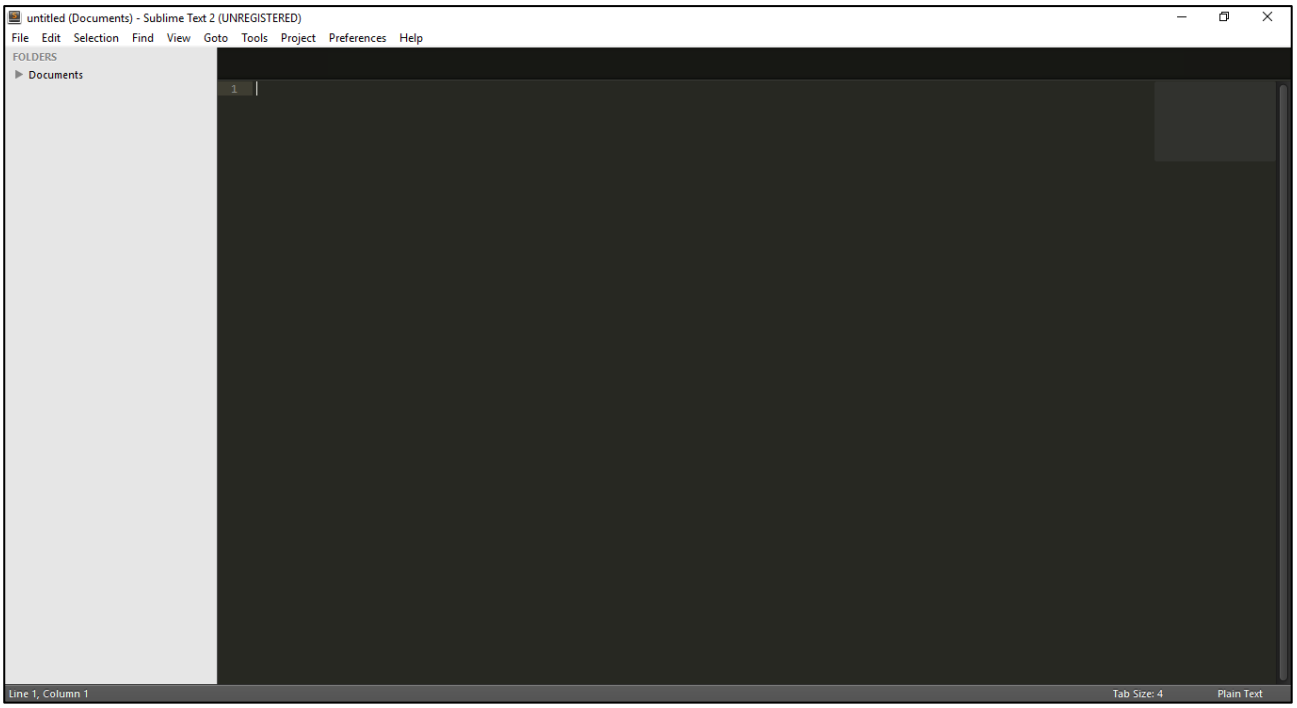

Figura 13: *Sublime Text* 2 Editor

Trata-se de um editor bastante eficiente e de código aberto que pode ser usado para escrever códigos JavaScript, HTML, CSS, entre outros. Bastante útil e eficaz para um rápido desenvolvimento, sem necessidade de configurações especiais de ambiente, bastando o seu download e posterior instalação descomplicada com o auxílio do *setup wizard*.

Para o desenvolvimento do projeto propriamente dito foram utilizados alguns frameworks que automatizam tarefas e tornam o desenvolvimento mais acelerado e mais abstrato. Essas ferramentas auxiliam no *quick start* de desenvolvimento do projeto e permitem que o desenvolvedor possa crescer a partir de uma base sólida já desenvolvida.

Neste sentido utilizou-se o *Gulp*[20] para auxiliar no desenvolvimento da interface de Front-End. O *Gulp* é uma ferramenta que automatiza tarefas muito popular ao lado do Grunt (ferramenta que tem o mesmo propósito).[21]

O *Gulp* auxilia no processo de construção da aplicação JavaScript, executando tarefas necessárias como teste em diferentes navegadores, compilação de CSS, e "minificação" de arquivos.

Além do *Gulp*, utilizou-se também o *LoopBack*[22] que auxilia na tarefa de criação e exposição das APIs em Rest. Com o auxílio desta ferramenta é possível criar modelos e expô-los para servirem como serviços para ser consumidos via web.

O *LoopBack* pode ser instalado fazendo uso do *Prompt* de Comando do Microsoft Windows ou pelo Terminal do Linux ou do Mac OS. Para baixa-lo e instalá-lo deve-se digitar **$ npm install -g strongloop** no Terminal (Linux, Mac OS) ou **npm install -g strongloop** (Microsoft Windows), de modo a instalá-lo globalmente na máquina do desenvolvedor.

4.1. DESENVOLVIMENTO *BACK-END*

Uma vez instalado, o sistema pode ser inicializado através da digitação do comando **slc loopback** no terminal de preferência do usuário, conforme figura abaixo:

[20] http://gulpjs.com/
[21] http://imasters.com.br/desenvolvimento/por-que-usar-gulp/?trace=1519021197&source=single
[22] https://loopback.io/

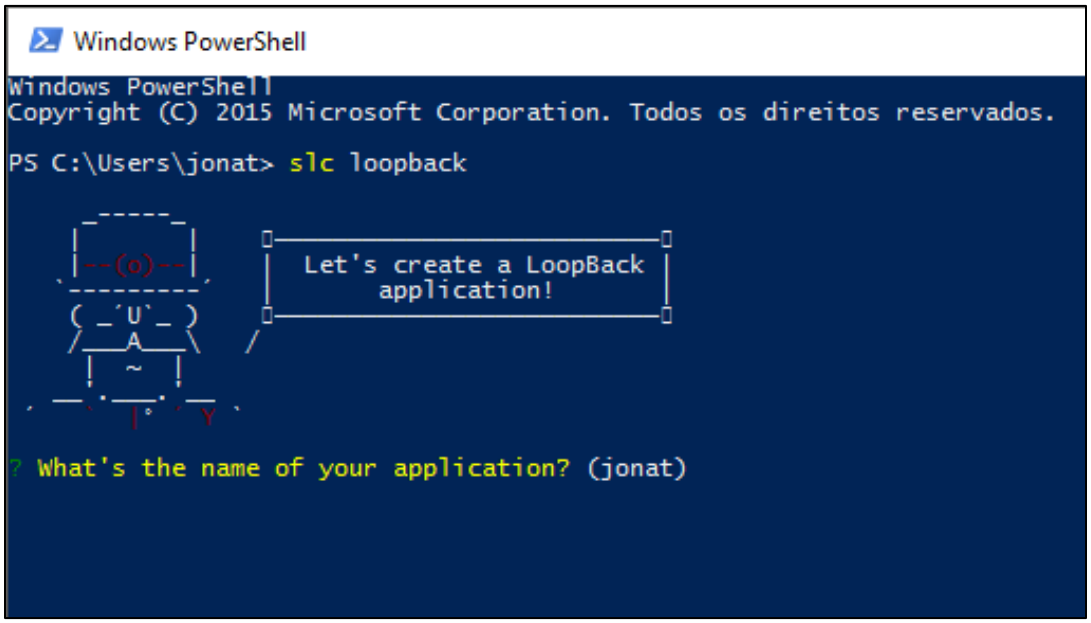

Figura 14: Tela Inicial da Aplicação *LoopBack*

A partir daí o *LoopBack* exibe uma série de perguntas a partir das quais ele fará a configuração mínima para a aplicação e a exposição das APIs. Ao término deste procedimento já se tem a aplicação disponível para receber requisições via web. Este procedimento é muito simples e é completado em poucos minutos. Na sequência deve-se criar os modelos de dados (*scheemas*), conceder as permissões de acesso a cada um dos métodos dos modelos, os possíveis relacionamentos entre os modelos de dados. Essas são as configurações básicas para a aplicação ser disponibilizada e estar funcional para suportar o desenvolvimento *front-end*, porém além dessas configurações outras ainda podem ser feitas para garantir a autenticação à aplicação utilizando-se de sistemas terceiros como por exemplo o Facebook, ou Google Plus. É possível também habilitar a validação do e-mail do usuário, através do envio de um e-mail de confirmação para este no momento em que este se cadastra na aplicação.

Depois desta configuração inicial, o LoopBack terá criado a estrutura inicial básica da aplicação *Rest* conforme figura a seguir:

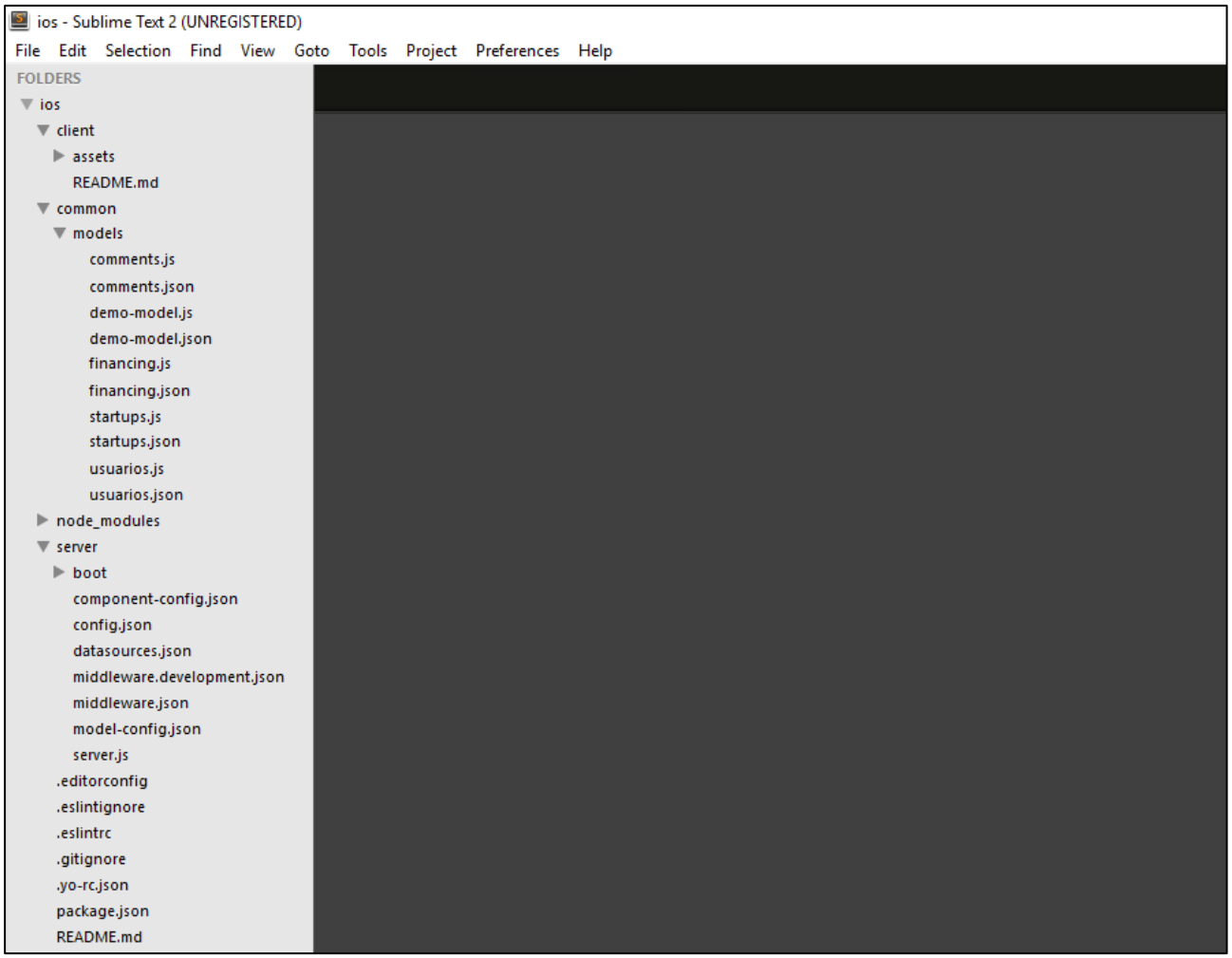

Figura 15: Estrutura da aplicação criada com o auxílio do *LoopBack*

Assim, para a criação dos modelos de dados é preciso digitar **slc loopback:model** no *prompt* de comando do sistema operacional. Feito isso, basta seguir definindo as questões postas pelo sistema. Neste momento, deve-se decidir pelo nome do modelo, em qual banco de dados será armazenado o modelo, se ele deve ser exposto via API *Rest*, pode ainda escolher se o modelo é apenas utilizado internamente dentro do servidor ou será usado na aplicação cliente, além de criar os atributos do modelo (além de definir as características destes atributos). Abaixo segue figura que demonstra o passo-a-passo detalhado conforme descrito.

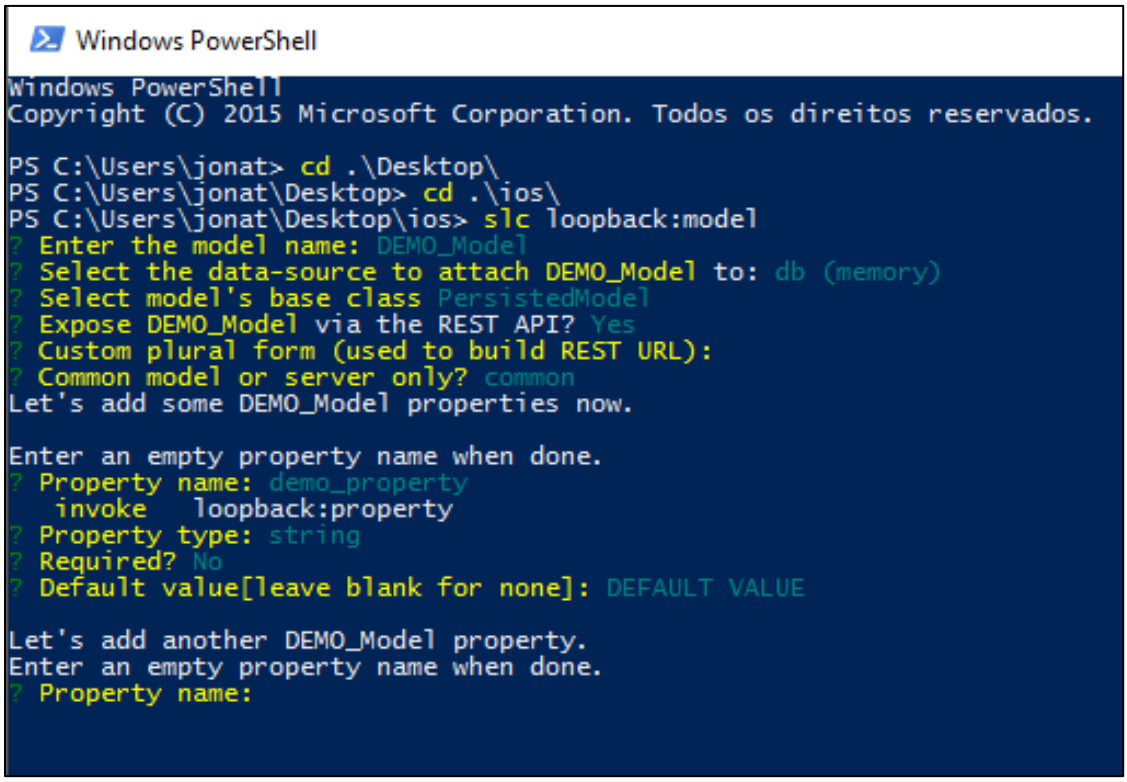

Figura 16: Criação de Modelos de Dados no *LoopBack*

O resultado desta configuração de modelo pelo *LoopBack* gera o seguinte código que pode ser usado para referência e pequenos ajustes na aplicação:

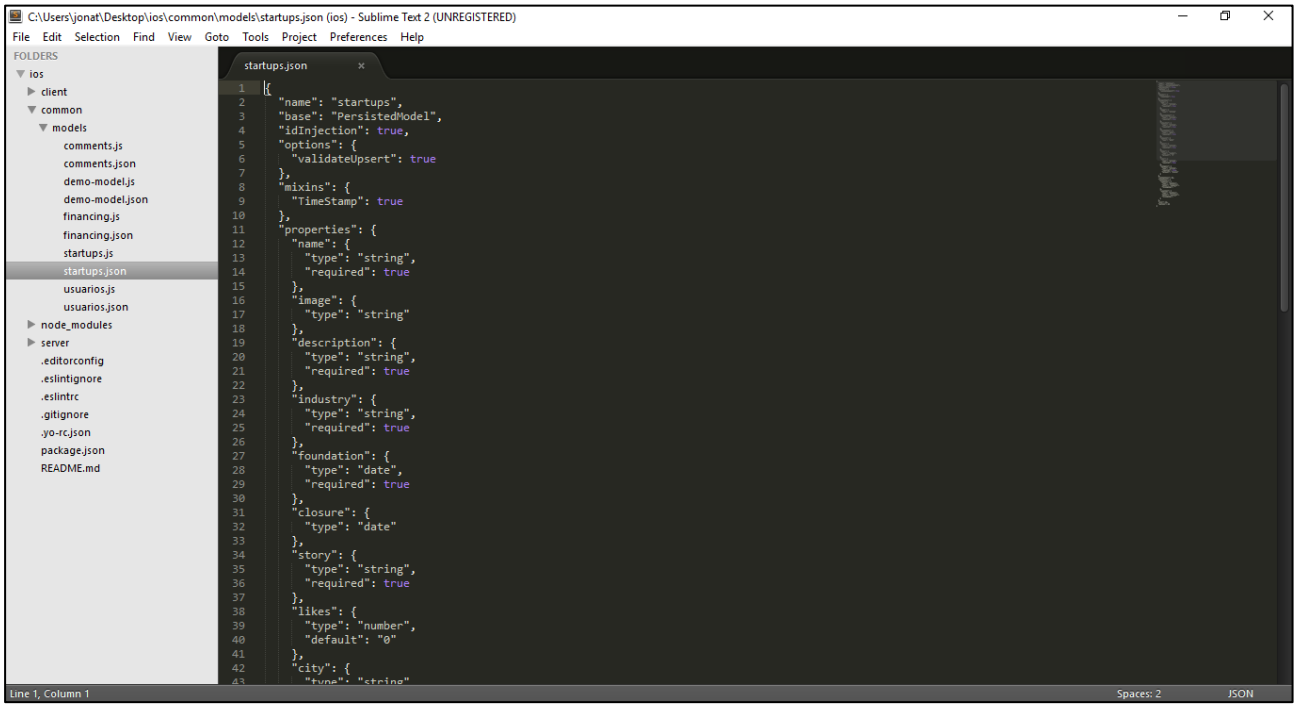

Figura 17: Arquivo JSON do Modelo de Dados

Depois da criação dos modelos, é preciso criar a conexão com o banco de dados. Para tanto é preciso digitar o comando **slc loopback:datasource** no *prompt* e seguir com as instruções dadas pela ferramenta. Segue ilustração do comando "*slc loopback:datasource*":

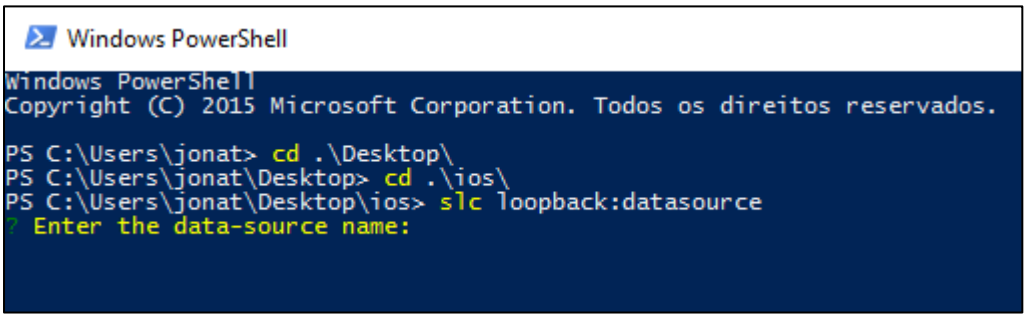

Figura 18: Criação da Conexão com o Banco de Dados no *LoopBack*

A seguir é possível observar o código da configuração da Conexão com o Banco de Dados e os ajustes que são feitos para tornar essa conexão plenamente funcional:

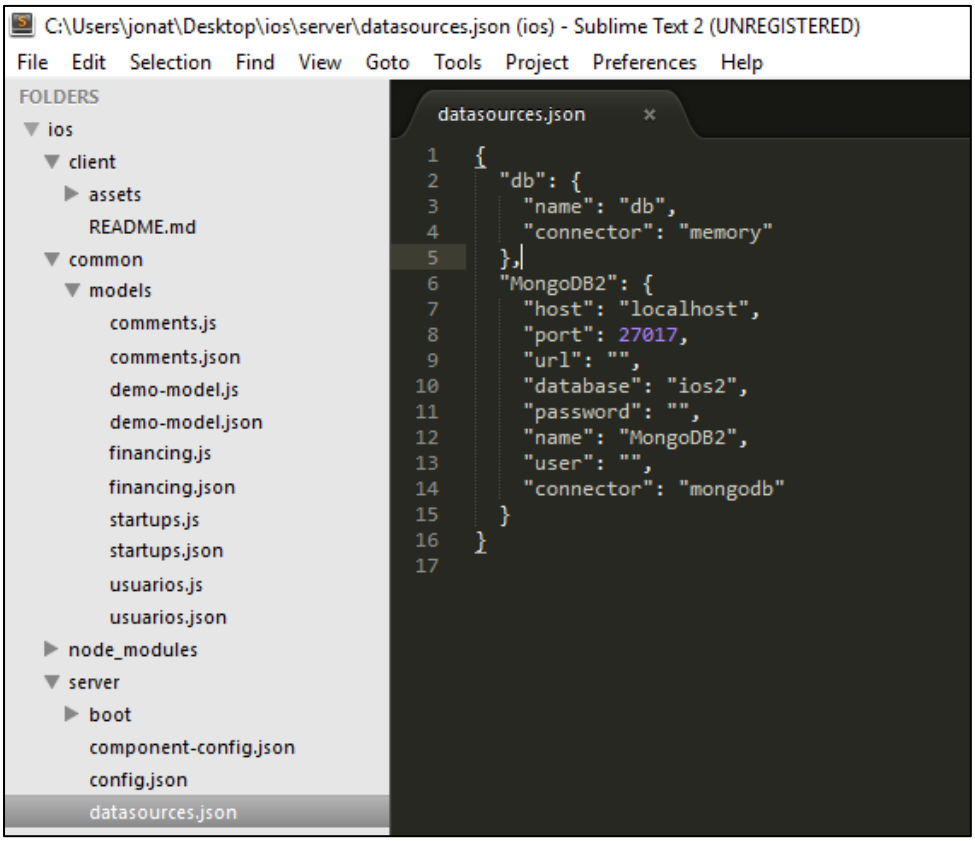

Figura 19: Conexão com o Banco de Dados

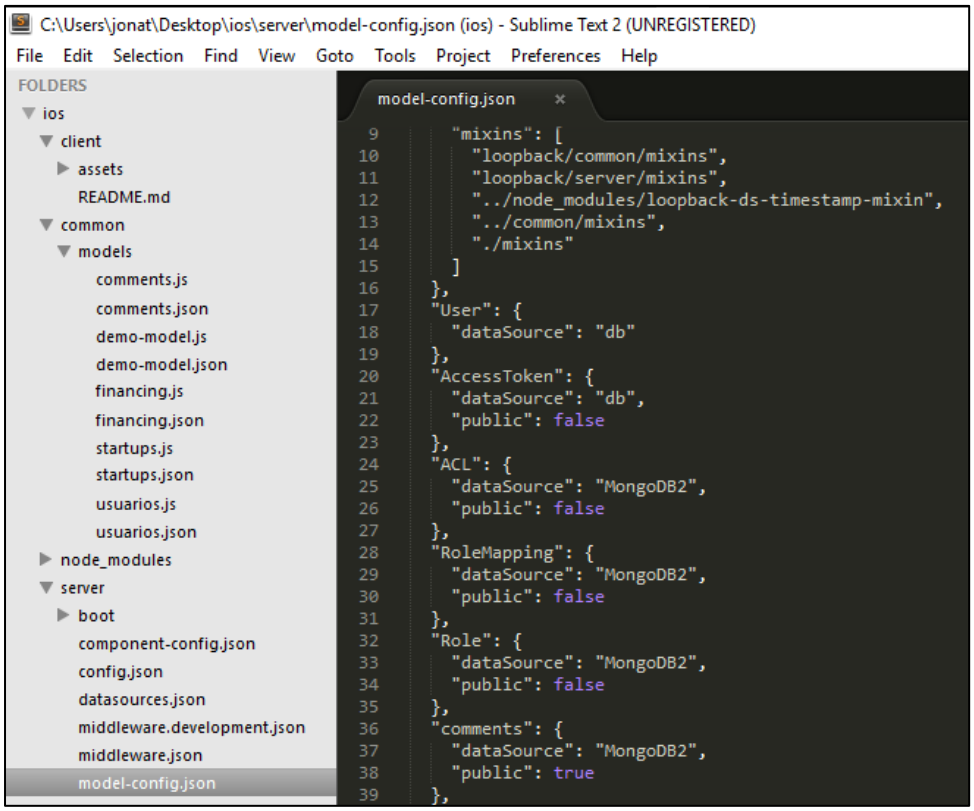

Figura 20: Apontamento da Conexão com o Banco dentro dos Modelos De Dados

Para a execução da tarefa do estabelecimento dos relacionamentos que se mostram necessários (com o auxílio do *LoopBack*), no *prompt* deve-se digitar o seguinte comando: **slc loopback:relation** e seguir respondendo às perguntas que surgirão na tela a fim de criar o relacionamento entre os modelos de dados desejados. Segue ilustração do processo na figura abaixo:

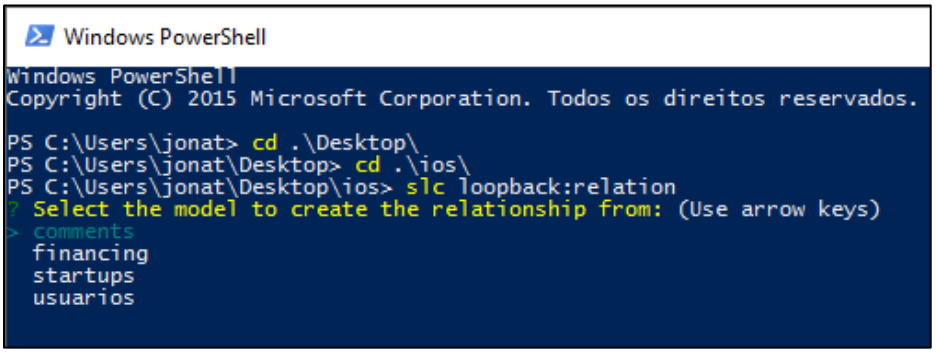

Figura 21: Criação de relacionamento entre Modelos de Dados no *LoopBack*

Neste momento é importante enfatizar o significado de relacionamento entre os modelos de dados. Nos bancos de dados relacionais convencionais tais como os comerciais Oracle, e SQL Server, e os *open source* MySQL[23] e PostgreSQL[24]. No que tange os bancos de dados originalmente não relacionais, como por exemplo o Mongo DB, no caso estudado, é possível fazer relacionamento entre os documentos (aninhando um documento dentro de outro). Vale também explicar a arquitetura essencial do Mongo DB: este é composto pelos Bancos de Dados, seguido pelas Coleções que são as estruturas que os Documentos (os registros em si).

É mister destacar que todos comandos digitados no *prompt* de comando devem ser executados dentro do diretório onde encontra-se a aplicação *loopBack* com a qual se está trabalhando. Caso essa diretriz não seja seguida, o *loopBack* retornará um erro, conforme figura a seguir:

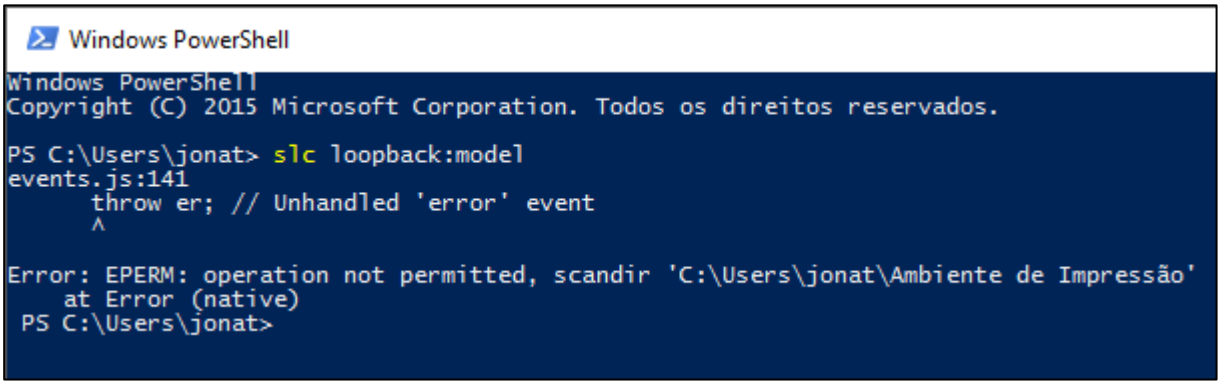

Figura 22: Erro exibido quando os comandos são executados em diretórios errados

Enquanto cada uma das configurações é realizada, é possível, em tempo real, navegar pela aplicação para validar as suas funcionalidades e o comportamento dos serviços, a conexão e a persistência dos dados no Banco de Dados. Para tanto, é preciso subir o Banco de Dados e a aplicação criada por meio do comando **node .** no *prompt* de

[23] https://www.mysql.com/
[24] https://www.postgresql.org/

comando e no diretório onde está a aplicação. Seguem figuras que ilustram este procedimento:

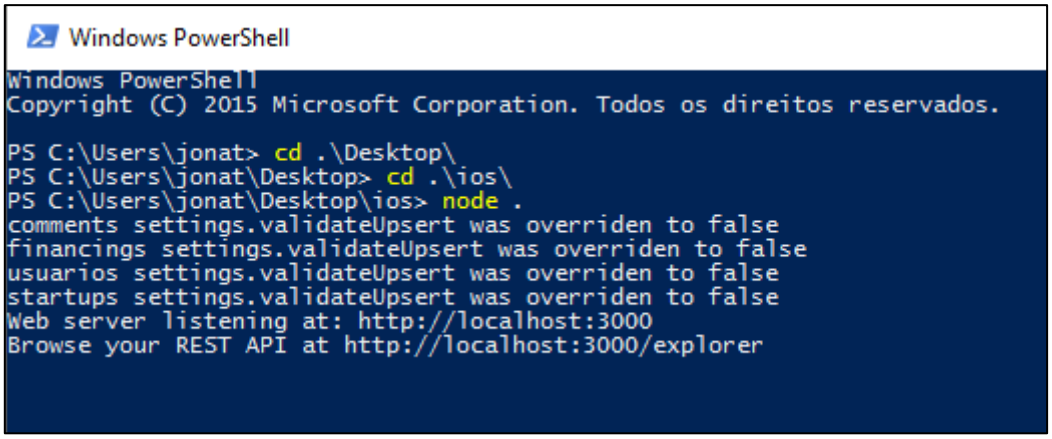

Figura 23: Iniciando a aplicação com o comando “node .”

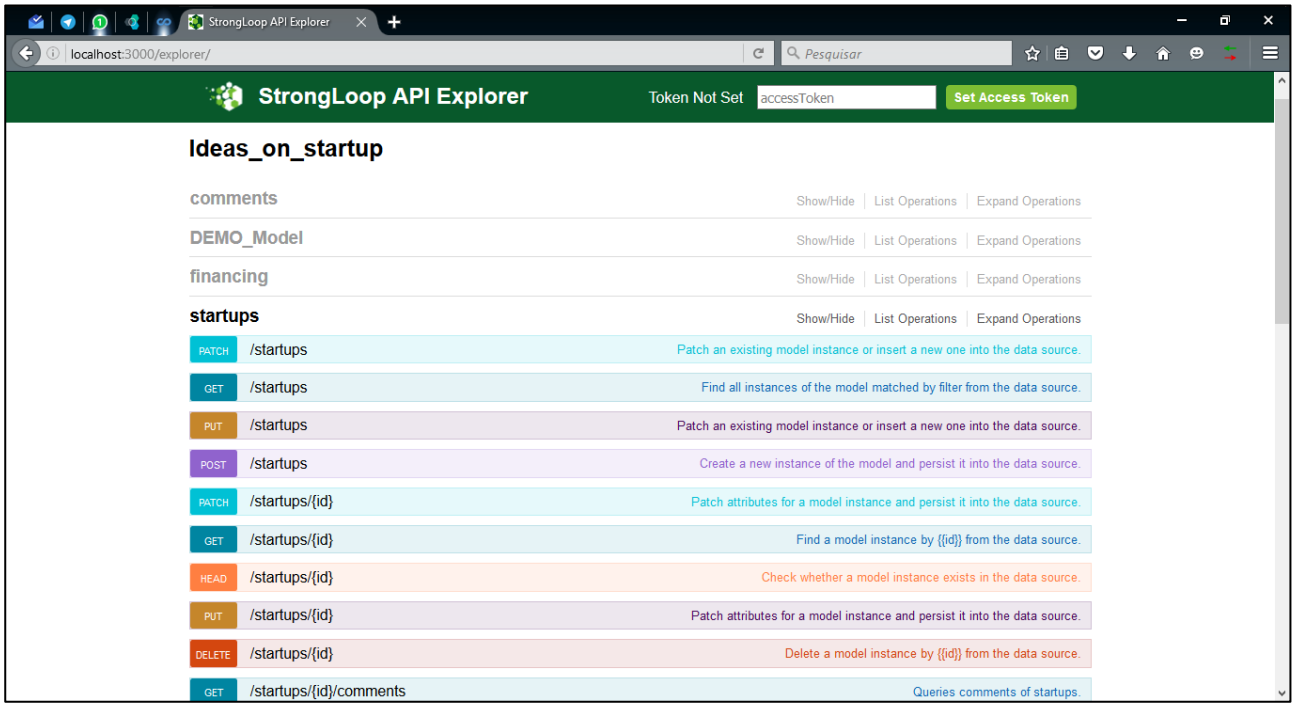

Figura 24: Interface gráfica para validação dos Métodos e Modelos

4.2. DESENVOLVIMENTO *FRONT-END*

O desenvolvimento da aplicação front-end foi efetuado utilizando-se o *framework* Angular JS. Com isso criou-se uma estrutura *Model-View-Controller* (MVC). Nesse sentido

as *Views* servem para criar a parte visual da aplicação, os *models* servem para se comunicar com o Banco de dados, e finalmente os *controllers* têm a finalidade de controlar as ações executadas nas páginas.

A seguir, é possível verificar detalhes dos elementos que compõem a aplicação criada como base para avaliação das tecnologias em questão.

A aplicação Angular JS tem início com a diretiva “ng-app” colocada na página index.html, conforme detalhe da figura abaixo. Ainda na página index.html existe a chamada do *controller* principal da aplicação, o “AppController” e o apontamento do local da *view* onde serão injetadas todas as *views* do sistema.

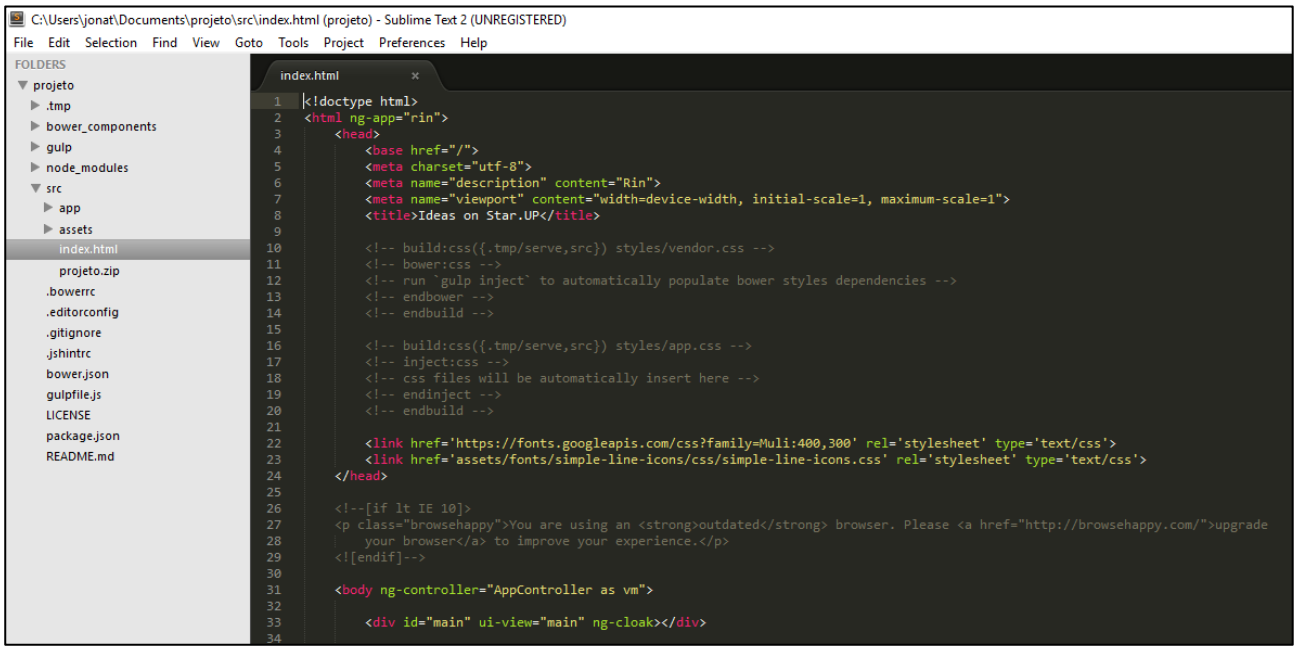

Figura 25: Detalhes da página index.html

Para garantir o apontamento dos *controllers* de cada uma das páginas e a injeção correta das *views* de acordo com os cliques feitos pelo usuário nas aplicações, é preciso configurar as rotas da aplicação num arquivo de configuração conforme exibido na figura 26 e “sobreposto” pelos arquivos de módulos (que contém as rotas) de cada uma das páginas da aplicação conforme exemplificado na figura 27.

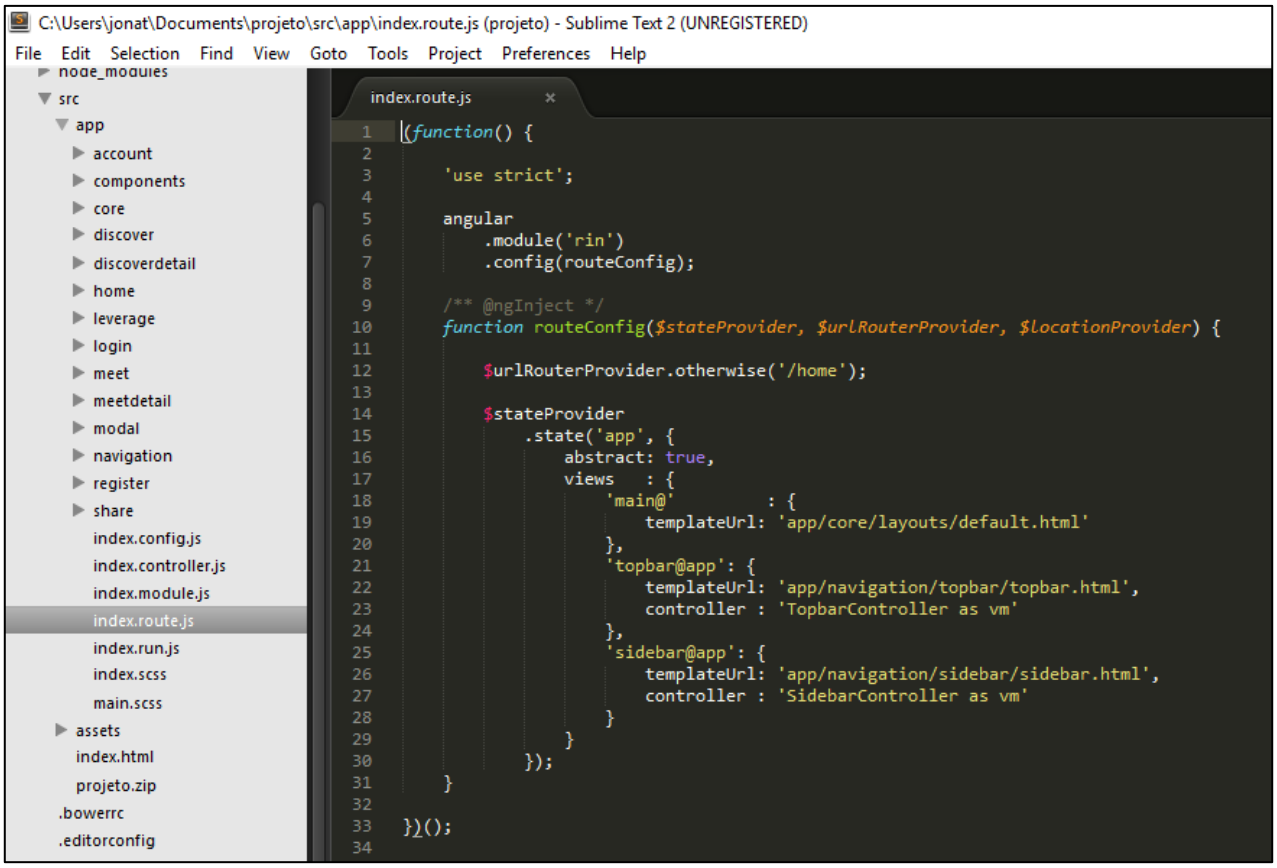

Figura 26: Detalhes do arquivo index.route.js

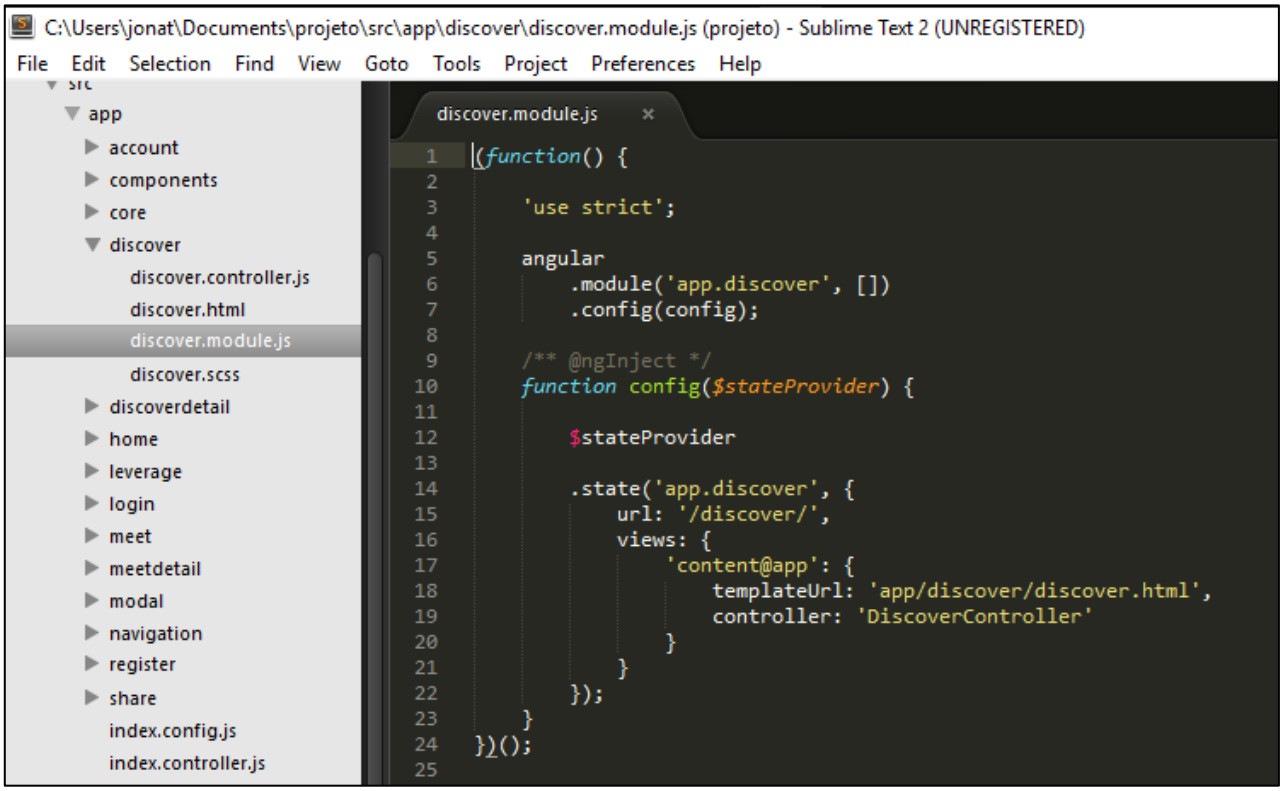

Figura 27: Detalhes do arquivo discover.module.js

A aplicação reconhece qual *view* exibir e qual *controller* utilizar de acordo com o endereço da URL digitado pelo usuário. Este define o estado da aplicação que será ativado e isso faz com que o acionamento da página e a execução dos *controllers* ocorram. A passagem da informação sobre qual estado sendo executado é feita através do elemento $stateProvider[25], sendo transmitido a partir dos arquivos JavaScripts de cada uma das seções da aplicação (*"Discover New Ideas"*, *"Meet People"*, por exemplo) para o arquivo principal de roteamento, o index.route.js (visto acima).

Para que a aplicação conheça todos os estados possíveis de serem executados, criou-se um arquivo no qual todos estão elencados para o sistema, o index.module.js, vide figura abaixo:

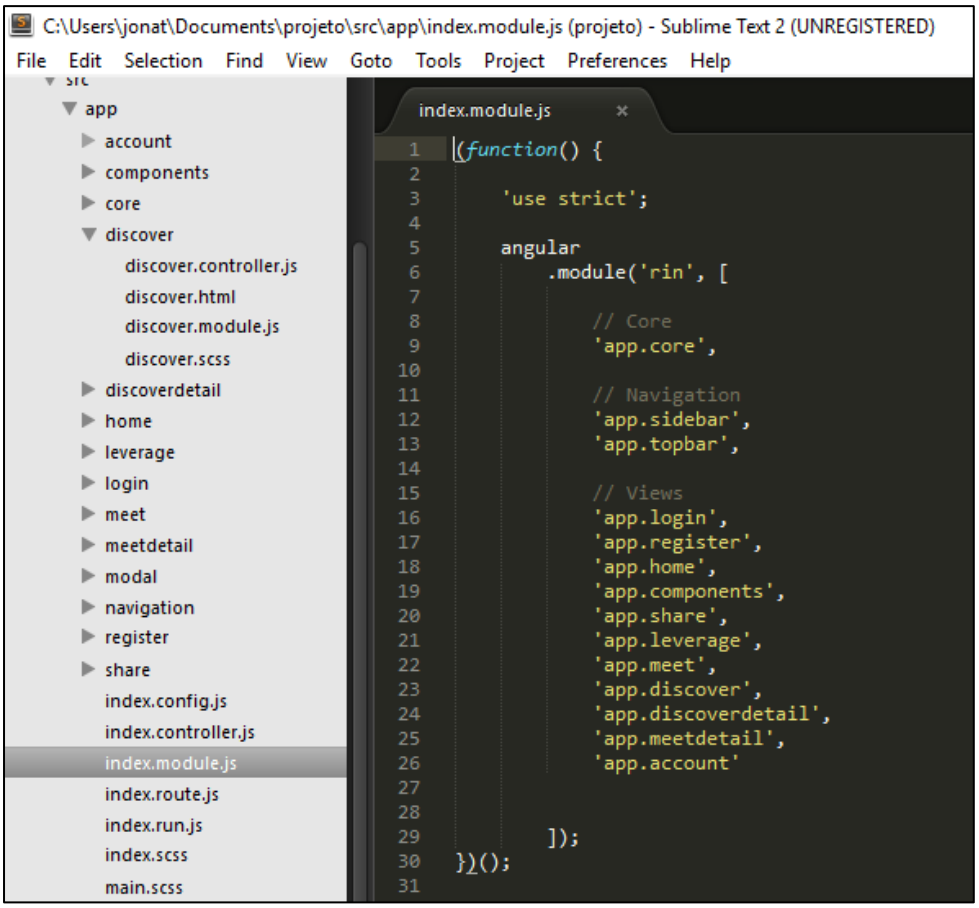

Figura 28: Detalhes do arquivo index.module.js

[25] https://www.sitepoint.com/write-modular-code-angular-ui-router-named-views/

E para encerrar a demonstração do desenvolvimento do *front-end* do sistema *“Ideas on Start.UP”*, criado para testar e complementar o estudo do Node.JS, Angular JS, e Mongo DB, exibe-se um arquivo controller de uma das seções da aplicação que contém a codificação dos serviços que se conectam aos métodos expostos pelo *back-end* do sistema, de modo a conseguirem executar o CRUD (do inglês *“Create, Read, Update, and Delete”*, cuja tradução em português significa literalmente “Criar, Ler, Atualizar, e Apagar”) dos dados no Mongo DB; além dos controles dos elementos da página, tais como os botões e os seus comportamentos, além da exibição de dados na página, recebidos pela parte do serviço, bem como outras ações e comportamentos. Vide figura a seguir:

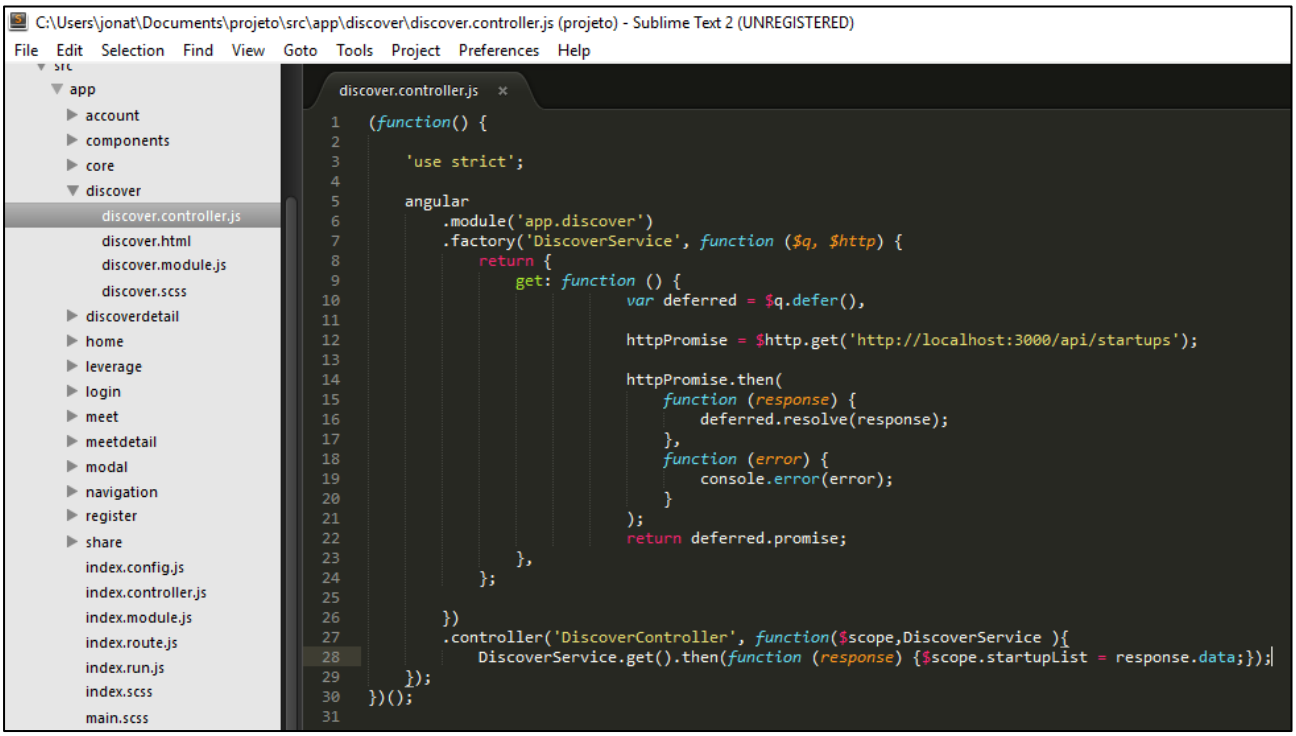

Figura 29: Detalhes do arquivo discover.controller.js

Com isso encerra-se a demonstração do desenvolvimento realizado para aplicação prática dos conceitos aprendidos a partir do estudo do Angular JS, Node.JS, e MongoDB.

5. CONCLUSÃO

Ao término deste trabalho verificou-se que as tecnologias estudadas, a saber: Angular JS, Node.JS e MongoDB, são bastante eficientes no que diz respeito ao processo de desenvolvimento, e da mesma forma, são bastante performáticas no tocante ao desempenho que proporciona para a aplicação que está sendo executada na Web.

Vale destacar ainda a possibilidade de todo o desenvolvimento ser realizado com o uso da linguagem JavaScript, tanto no *front-end*, quanto no *back-end*. Isso em função do uso do *framework* do Node.JS. Trata-se de um importante benefício que os desenvolvedores que conhecem tal linguagem passam a dispor.

Além disso, o desenvolvedor dispões de importantes ferramentas que auxiliam durante as atividades de desenvolvimento, tais como *Loopback Server*, *Gulp*, *Node Express Generator*, entre outras que são muito úteis e aceleram o tempo de desenvolvimento.

Tendo estes pontos em mente, conclui-se que vale o estudo e o aprimoramento nestas tecnologias afim de ganhar vantagem competitiva ao poder desenvolver mais rapidamente, e ao poder criar aplicações que sejam mais eficientes para os usuários.

Com relação à aplicação a qual foi desenvolvida a fim de estudar e demonstrar as tecnologias estudadas, verifica-se a importância de prosseguir trabalhando na mesma em decorrência da relevância do seu tema. Sendo assim, pretende-se prosseguir no seu desenvolvimento afim de torna-la mais funcional para os possíveis usuários, além de possibilitar desenvolvimento e crescimento profissional para o seu autor.

REFERÊNCIAS BIBLIOGRÁFICAS

ABERNETHY, Michael. O que é exatamente o Node JS? Disponível em: < http://imasters.com.br/artigo/22016/javascript/o-que-exatamente-e-o-nodejs/ >. Acesso em: 08.jul.2016.

AZZI, Matheus. Single Page Apps com AngularJS. Disponível em: < http://matheusazzi.com/single-page-apps/ >. Acesso em: 10. set. 2016.

CHAPMAN, Stephen. What Is JavaScript? Disponível em: < http://javascript.about.com/od/reference/p/javascript.htm >. Acesso em: 11.jul.2016.

CLARK, Scott. Web-based Mobile Apps of the Future Using HTML 5, CSS and JavaScript. Disponível em: < http://www.htmlgoodies.com/beyond/article.php/3893911/Web-based-Mobile-Apps-of-the-Future-Using-HTML-5-CSS-and-JavaScript.htm >. Acesso em: 04. jul. 2016.

DAYLEY, Brad. Introducing NoSQL and MongoDB. Disponível em: < http://www.informit.com/articles/article.aspx?p=2247310 >. Acesso em: 13.jul.2016

FERREIRA, Elcio; EIS, Diego. HTML5: Curso W3C Escritório Brasil. Disponível em: < http://www.w3c.br/pub/Cursos/CursoHTML5/html5-web.pdf >. Acesso em: 04. jul. 2016.

GIFT, Noah; JONES, Jeremy. Use Node.js as a full cloud environment development stack Disponível em: < http://www.ibm.com/developerworks/cloud/library/cl-nodejscloud/index.html >. Acesso em: 08.jul.2016.

GOTTSCHALK, Harold. Ionic: What it is, and why we use it. Disponível em: < http://www.softstackfactory.com/blog/2016/4/7/ionic-what-it-is-and-why-we-use-it >. Acesso em 13.jul.2016

GRENZI, Lucio. Use Ionic framework to develop mobile application. Disponível em: < http://www.slideshare.net/Codemotion/lucio-grenzi-use-ionic-framework-to-develop-mobile-application >. Acesso em 13.jul.2016.

KALAN, Matt. Rapid Development and Performance By Transitioning from RDBMSs to MongoDB. Disponível em: < https://www.mongodb.com/presentations/mongodb-chicago-benefits-using-mongodb-over-rdbms >. Acesso em: 12. set. 2016.

LIE, Håkon Wium; BOS, Bert. Cascading Style Sheets, designing for the Web. Disponível em: < https://www.w3.org/Style/LieBos2e/history/ >. Acesso em: 04. jul. 2016.

MORAES, William Bruno. "Construindo aplicações com Node.JS". São Paulo: Novatec Editora, 2015.

MUNZLINGER, Elizabete. Introdução à Tecnologia Web. Disponível em: < http://www.elizabete.com.br/site/Outros/Entradas/2011/2/20_Tecnologia_Web_files/01-JS-HistoricoCaracteristicas.pdf >. Acesso em: 11.jul.2016.

NARAYANAN, Uma. MongoDB NoSQL Solution: Advantages and Disadvantages. Disponível em: < http://www.developer.com/db/mongodb-nosql-solution-advantages-and-disadvantages.html >. Acesso em: 12. set. 2016.

PEREIRA, Caio Ribeiro. Node.JS. Disponível em: < http://udgwebdev.com/nodejs/ >. Acesso em: 09.jul.2016.

VALENTE, Kaio. Porque utilizar AngularJS no seu próximo projeto. Disponível em: < https://tasafo.wordpress.com/2014/11/26/porque-utilizar-angularjs-no-seu-proximo-projeto/ >. Acesso em: 10.jul.2016.

VANCE, Ashlee. "Elon Musk - Como o Ceo Bilionário da Spacex e da Tesla Está Moldando Nosso Futuro". Rio de Janeiro: Intrínseca, 2015.

ZAKAS, Nicholas C. "JavaScript de Alto Desempenho". São Paulo: Novatec Editora, 2010.