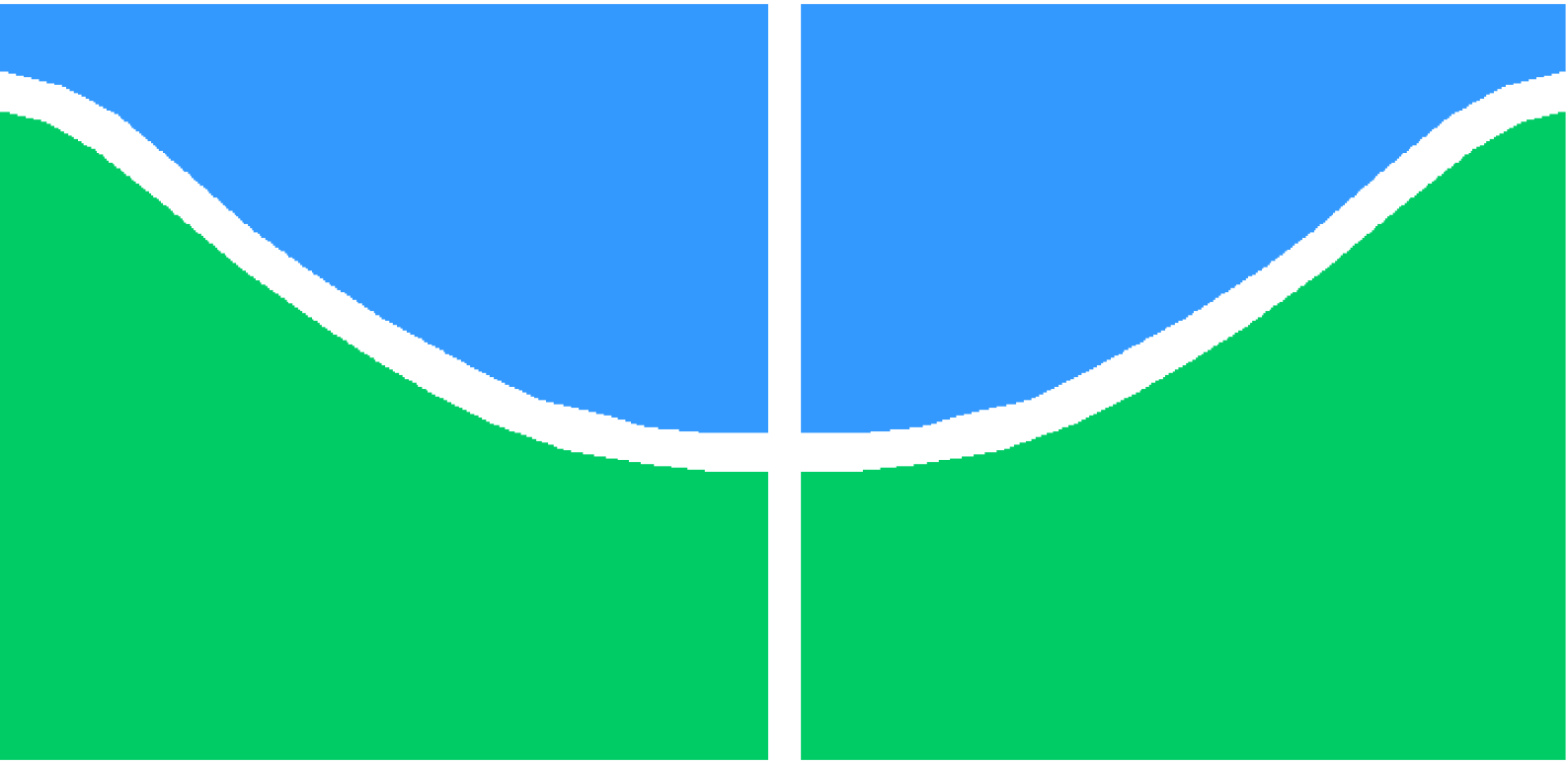

Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia de Software

Anansi - Um aplicativo móvel para identificação de famílias da ordem Araneae

Autor: Pedro Henrique Sales Ferreira
Orientador: Professor Dr. Wander Cleber Pereira
Coorientador: Professor Dr. Paulo César Motta

Brasília, DF
2018

Pedro Henrique Sales Ferreira

Anansi - Um aplicativo móvel para identificação de famílias da ordem Araneae

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software .

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Professor Dr. Wander Cleber Pereira

Coorientador: Professor Dr. Paulo César Motta

Brasília, DF

2018

Pedro Henrique Sales Ferreira
Anansi - Um aplicativo móvel para identificação de famílias da ordem Araneae / Pedro Henrique Sales Ferreira . – Brasília, DF, 2018-
63 p. : il. (algumas color.) ; 30 cm.

Orientador: Professor Dr. Wander Cleber Pereira

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2018.
Anansi - Um aplicativo móvel para identificação de famílias da ordem Araneae

CDU

Pedro Henrique Sales Ferreira

Anansi - Um aplicativo móvel para identificação de famílias da ordem Araneae

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software .

Trabalho aprovado. Brasília, DF, 13 de dezembro de 2018.

Professor Dr. Wander Cleber Pereira
Orientador

Professor Dr. Paulo César Motta
Coorientador

Professor Dr Edson Alves da Costa Júnior
Convidado 1

Professora Rejane Maria da Costa Figueiredo
Convidado 2

Brasília, DF
2018

Agradecimentos

Agradeço ao meu pai Valdenez e a minha mãe Francisca, por sempre terem colocado a educação e a formação moral e humana dos filhos em primeiro lugar. Muito mais que pais e facilitadores de minhas conquistas, foram grandes amigos e exemplos de vida. Agradeço também a minha irmã Éricka, que sempre me foi exemplo de retidão e dedicação.

Agradeço imensamente a minha noiva, Camila, por sempre ter sido para mim braço direito e esquerdo nos momentos pesarosos, companheira quando em momentos de luta e um farol nos momentos de escuridão. Obrigado por todo o incentivo para que eu pudesse alcançar meus objetivos.

Agradeço ainda meus orientadores, Wander Cleber e Paulo César, por se disporem a me auxiliar e me guiar na execução do presente trabalho.

Acima de tudo agradeço a Deus, pois sem Ele nada me seria possível.

"*A felicidade só é verdadeira quando compartilhada.* "

Christopher McCandless

Resumo

Como forma de incentivar a pesquisa e difundir o conhecimento de forma informatizada, prática e interativa a respeito das espécies brasileiras, o presente trabalho tem como proposta um meio de consulta sistematizada por famílias de aranhas do bioma Cerrado, por meio de um software multi-plataforma, desenvolvido com *framework Ionic 3* e API em *Java*, utilizando princípios, técnicas e práticas da metodologia ágil *Scrum* adaptada ao desenvolvimento solo com entrega contínua. Como resultado, tem-se o aplicativo Anansi, para consulta de família de aranhas por meio de nomenclatura ou por características visuais e morfológicas.

Palavras-chave: *aplicativo. multi-plataforma. scrum. metodologia ágil. biodiversidade. aracnídeos. aranhas.*

Abstract

As a way to encourage research and disseminate knowledge in a computerized, practical and interactive way regarding Brazilian species, the present work has as a proposal a systematized consultation for families of spiders from Cerrado biome, through a multi-platform software, developed with Ionic 3 framework and API in Java using principles, techniques and practices of the agile methodology Scrum adapted to the solo development with continuous delivery. As a result, the Anansi application is used to query the spider family by means of nomenclature or by visual and morphological characteristics.

Key-words: *application. multi-platform. scrum. agile methodology. biodiversity. arachnids. spiders.*

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

TCC	Trabalho de Conclusão de Curso
API	*Application Programming Interface*
CTFB	Catálogo Taxonômico da Fauna do Brasil
XP	*Extreme Programming*
PSP	*Personal Software Process*
SDK	*Software Development Kit*
C&T	Ciência e Tecnologia
API	*Application Programming Interface*
REST	*Representational State Transfer*
IoC	*Inversion of Control*
IC	Integração Contínua
APP	Aplicativo
WSCA	*World Spider Catalog*
MVC	*Model View Controller*

Sumário

1 Introdução

Tendo em vista a vastidão de exemplares existentes na biodiversidade brasileira, o conhecimento da população a respeito das espécies nativas ainda é escasso. Os meios para consulta e pesquisa, mesmo que em grande parte digitalizados, ainda se dão de forma não trivial e de difícil acesso para leigos no assunto.

1.1 Contexto

De acordo com o Ministério do Meio Ambiente, o Brasil abriga a maior biodiversidade do planeta, sendo mais de 20% do número total de espécies da Terra. Ele ocupa o posto de principal nação entre os 17 países megadiversos (ou de maior biodiversidade) (MMA, 2018). Atualmente são conhecidas 117.445 espécies válidas de animais no Brasil, a sua enorme maioria artrópodes (cerca de 85%, com quase 94.000 espécies) (JBRJ, 2016).

A sistematização dos dados a respeito da fauna e flora mundial está ocorrendo de maneira gradual. Há muitos documentos já digitalizados e publicados, bem como mecanismos de catalogação, como o *Catálogo Taxonômico da Fauna do Brasil (CTFB)* (JBRJ, 2016). Porém, esta divulgação científica ainda é realizada majoritariamente de uma forma mecânica e estática, sendo por vezes necessário dedicar muito tempo para pesquisa e mineração de dados nas mais diversas plataformas e catálogos.

O aplicativo Anansi (nome da lenda africana figurada como um deus aranha) tratada no presente trabalho tem como proposta disponibilizar de maneira informatizada as informações científicas básicas e essenciais a respeito de exemplares da ordem *Araneae* (aranhas), tendo em vista a pluralidade e a grande importância desses indivíduos no âmbito biológico, visando rápida identificação e classificação de indivíduos. Tais informações serão dispostas em formato catálogo inteligente, permitindo buscas tanto por categoria taxonômica como a partir de características morfológicas observáveis, de forma a possibilitar consultas tanto para usuários leigos no assunto como especialistas, garantindo a obtenção de informações a respeito da importância médica, nocividade, comportamento, incidências e morfologia, entre outras.

1.2 Justificativa

Mesmo com a difusão tecnológica de informações a partir do advento da internet e o aprimoramento dos meios de pesquisas, o conhecimento acerca da biodiversidade permanece incompleto, principalmente nos trópicos, onde ainda há um grande número de

espécies a serem descritas e muito pouco conhecidas no que diz respeito à sua distribuição (WHITTAKER et al., 2005). A maior parte do conhecimento atual sobre diversidade e distribuição geográfica de organismos é baseada em alguns táxons, particularmente animais vertebrados (OLIVEIRA, 2011a). Alguns grupos de animais, como os artrópodes, vem sendo negligenciados em estudos taxonômicos (DINIZ-FILHO; MARCO; HAWKINS, 2010), embora eles englobem a maior parte da diversidade animal.

Os aracnídeos (Filo *Arthropoda*, Subfilo *Chelicerata*) constituem a segunda maior classe do reino animal (superado apenas pelos insetos), e o maior e mais importante grupo de predadores terrestres, com cerca de 110 mil espécies divididas em 11 ordens, sendo a maior parte composta pela ordem dos ácaros e carrapatos. Os demais aracnídeos, cujos representantes mais conhecidos são as aranhas e os escorpiões, são mensurados em cerca de 60 mil espécies, com cerca de 5 a 10 mil espécies ocorrendo no Brasil, sendo registradas até o momento cerca de mil espécies de aranhas somente no Cerrado (MOTTA, 2010).

Os aracnídeos representam os principais animais peçonhentos do Brasil, sendo responsáveis entre os anos de 2000 e 2012 por 65% dos acidentes com animais peçonhentos em todo o território nacional. Apesar do alto número de acidentes com aracnídeos, a mortalidade é baixa, sendo inferior até a de insetos (proporcionalmente picadas de aranha matam menos que cerdas de lagartas) (MOTTA, 2010).

Porém, informações científicas relevantes a respeito destes indivíduos, como riscos oferecidos, distribuição, morfologia básica e peçonha são desconhecidos pela grande maioria da população, e o acesso a informações deste tipo não são encontradas com facilidade na literatura, ainda mais pela complexidade no processo de identificação do animal. De tal forma, a informatização para distribuição destas informações, traria para a população um meio de acesso rápido ao banco de conhecimento científico de espécies, possibilitando agilidade no atendimento em casos de acidentes com estes indivíduos e também o esclarecimento de dúvidas, podendo ainda contribuir com a consolidação e expansão do conhecimento científico, por meio de contribuições fotográficas e georreferenciais.

1.3 Problema de Pesquisa

A divulgação científica, informações como distribuição, comportamento, nocividade e importância médica dos aracnídeos não são de conhecimento comum, e o acesso a esse tipo de informação não é realizado de maneira prática e/ou dinâmica, carecendo de pesquisas e consultas (as vezes minuciosas a depender da espécie que se procura) a vasta bibliografia existente, não sendo ainda esta pública em sua totalidade.

Mesmo sendo esta bibliografia em parte digitalizada, o processo investigativo para extração de informações se dá de forma onerosa mesmo para pesquisadores e especialistas, devido a decentralização dessas informações, levando ao desinteresse do público

leigo em se aprofundar no assunto, e sendo muitas vezes demasiadamente demorado para pesquisadores o processo taxonômico.

A questão de pesquisa que será discutida durante este trabalho é *"Como promover o incentivo à pesquisa científica sobre espécies da Ordem Araneae por meio de um aplicativo móvel?"*.

1.4 Objetivos

O presente trabalho tem como objetivo geral a realização de um estudo acerca da problemática da difusão prática e interativa de informações sobre a biodiversidade, levantando a proposta de implementação de um software multiplataforma para catalogação sistematizada de espécies, a fim de proporcionar praticidade na pesquisa científica de espécies, garantindo maior acessibilidade de informações para a população.

Os objetivos Específicos são:

- criar uma aplicação mobile para consulta de famílias de espécies;
- implementar de mecanismo consulta inteligente, baseado em chaveamento, para possibilitar identificação visual de espécies;
- identificar melhorias de usabilidade de software nas ferramentas de buscas taxonômicas existentes;
- realizar levantamento de fatores comuns de interesse popular a respeito da morfologia de exemplares da ordem *Araneae*;
- apresentar informações de interesse comum acerca das espécies consultadas.

2 Referencial Teórico

Neste capítulo serão apresentados os conceitos centrais abordados no presente trabalho, a fim de contextualizar a problemática e fundamentar a proposição de solução.

2.1 Ciências Biológicas

2.1.1 Divulgação Científica

O conhecimento científico se expande de forma muito rápida devido o dinamismo da ciência e da tecnologia e, por isso, as pessoas precisam se atualizar constantemente (BERTOLETTI, 2003). Essa rapidez no desenvolvimento da ciência e tecnologia faz com que o conhecimento dos cidadãos se distancie do conhecimento produzido pela ciência (KEMPER, 2008).

A fim de aproximar o público geral das descobertas e avanços da ciência, são produzidas divulgações científicas, que podem ser definidas como “o uso de processos e recursos técnicos para a comunicação da informação científica e tecnológica ao público em geral ”. Nesse sentido, divulgação supõe a tradução de uma linguagem especializada para uma leiga, visando a atingir um público mais amplo (ALBAGLI, 1996).

Com a diversificação dos meios de comunicação, houve um aumento de divulgações científicas em relação às décadas anteriores. A “revolução tecnológica ”possibilitou avanços no campo da informação, tornando divulgações científicas ainda mais acessíveis ao público, excedendo os limites da universidade (PECHULA, 2007).

Porém, é interessante notar a escassa divulgação científica que tem sido realizada pelas unidades de pesquisa e instituições públicas de ensino superior, mesmo após o Projeto de Lei 1120/07, de 2007, aprovado na Comissão de Educação e Cultura da Câmara dos Deputados, que as obriga a publicarem suas produções técnicas e científicas na internet (TORRESI; PARDINI; FERREIRA, 2012).

DUARTE (2004) afirma que as instituições públicas que atuam na área de C&T são atores sociais mantidos pela sociedade que devem dar um retorno dos recursos públicos investidos, contribuindo para a evolução da sociedade, por meio da divulgação do conhecimento produzido e da inserção da C&T no cotidiano das pessoas.

De tal forma, é inegável a importância da tecnologia na divulgação científica, agindo como facilitador e vinculador do acesso a informação. Sob um olhar mais atento, podemos afirmar que hoje, mais do que nunca, está estabelecida uma relação profunda entre a sociedade e os domínios do conhecimento científico e tecnológico (FIRME; SILVA,

2016).

BUSTAMANTE (1997) afirma:

> "O papel que, hoje em dia, a ciência e a tecnologia desempenham em nossa sociedade é tão profundo e extenso que se torna difícil conceber um único âmbito de atividade em que não estejam presentes ou, ainda, não modifiquem, substancialmente, atitudes, comportamentos, formas de relação, ou não propunham novas formas de fazer, de pensar e sentir e não ponham em questão valores tradicionalmente assumidos."

2.1.2 Biologia das Aranhas (*Araneae*)

Há muitas lacunas no que diz respeito à divulgação científica sobre aranhas. Afinal, conforme afirma Foelix (2011), o fato de normalmente as pessoas não gostarem de aranhas influencia no pouco conhecimento que têm sobre elas.

As aranhas (ordem *Araneae*) estão distribuídas em todo o mundo e conquistaram todos os ecossistemas, com exceção do ar e o mar aberto. A maioria das aranhas são relativamente pequenas, medindo de 2 a 10 milímetros de corpo. Porém algumas grandes tarântulas podem ter de 80 a 90 milímetros de corpo, sendo os machos quase sempre menores e com menor expectativa de vida que as fêmeas (FOELIX, 2011).

As aranhas muitas vezes confundidas com insetos, são diferenciadas por possuírem o corpo dividido em duas partes (cefalotórax e abdômen) e seis pares de apêndices no cefalotórax: um par de quelíceras (órgão para inocular veneno), um par de pedipalpos (apêndices semelhantes às pernas que nos machos são modificados em órgãos copuladores) e quatro pares de patas, como mostrado nas Figuras 1 e 2. A teia é depositada pelas fiandeiras, localizadas no final do abdômen. (MOTTA, 2010)

Todas as aranhas são carnívoras e produzem seda, muitas especializadas em construção de armadilhas, enquanto outros caçam suas vítimas. Os insetos constituem a principal fonte de presas para as aranhas, mas alguns outros artrópodes são frequentemente consumidos também, além de pequenos répteis, anfíbios e mamíferos (FOELIX, 2011).

A grande maioria das aranhas possui veneno pouco tóxico. No Brasil, apenas 20 espécies provocam acidentes de importância médico-sanitária. Estas espécies pertencem ao grupo das *Araneomorfas* e podem ser divididas em dois subgrupos: o primeiro inclui os gêneros Phoneutria (aranha-armadeira) e *Latrodectus* (viúva-negra), ambos com peçonha neurotóxica (atinge o sistema nervoso), e o segundo grupo, representado pela *Loxosceles* (aranha-marrom), com peçonha necrosante (SILVA; TIBURCIO; CORREIA; AQUINO, 2005). A maioria das espécies não oferece riscos aos seres humanos, como, por exemplo, a aranha-caranguejeira, que por seu aspecto e tamanho causa aversão a muitas pessoas, mas não produz veneno prejudicial ao homem. Portanto, não é possível considerar a aparência de uma aranha para julgá-la peçonhenta ou não (SOUZA; SILVA; SANTOS, 2011).

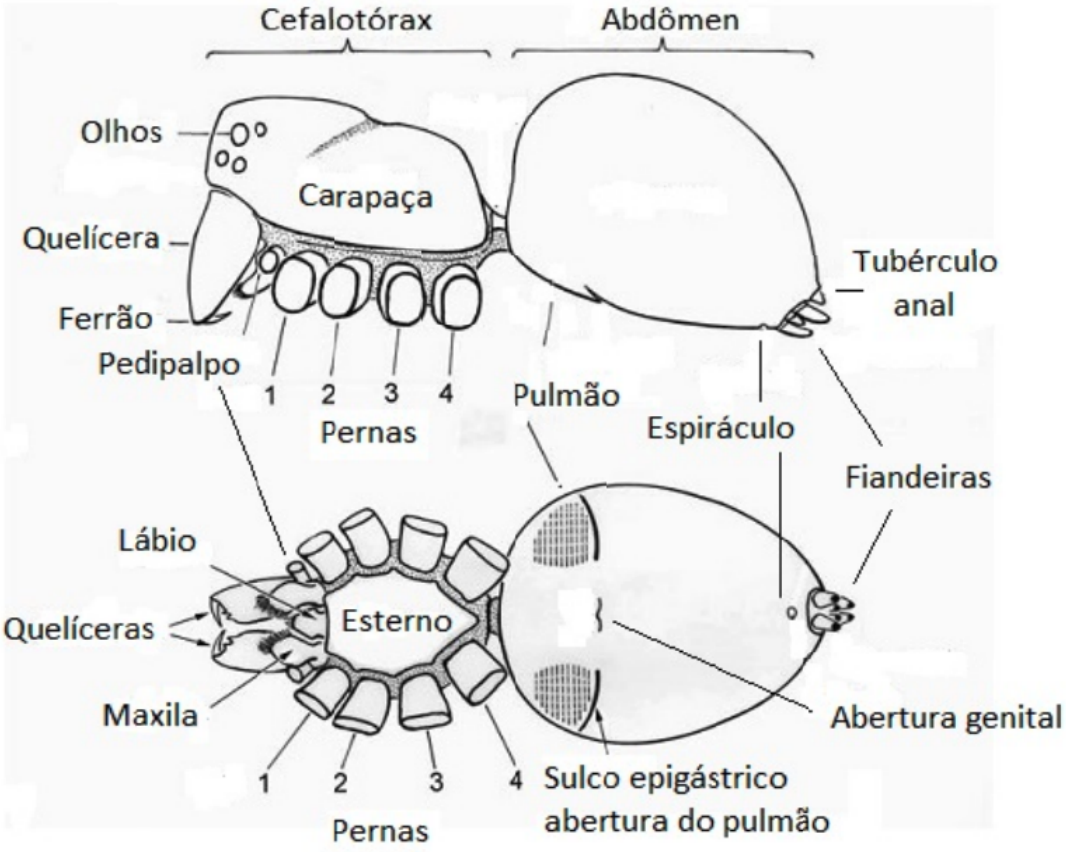

Figura 1 – Morfologia Externa de uma Aranha.

Fonte: (MOTTA, 2010)

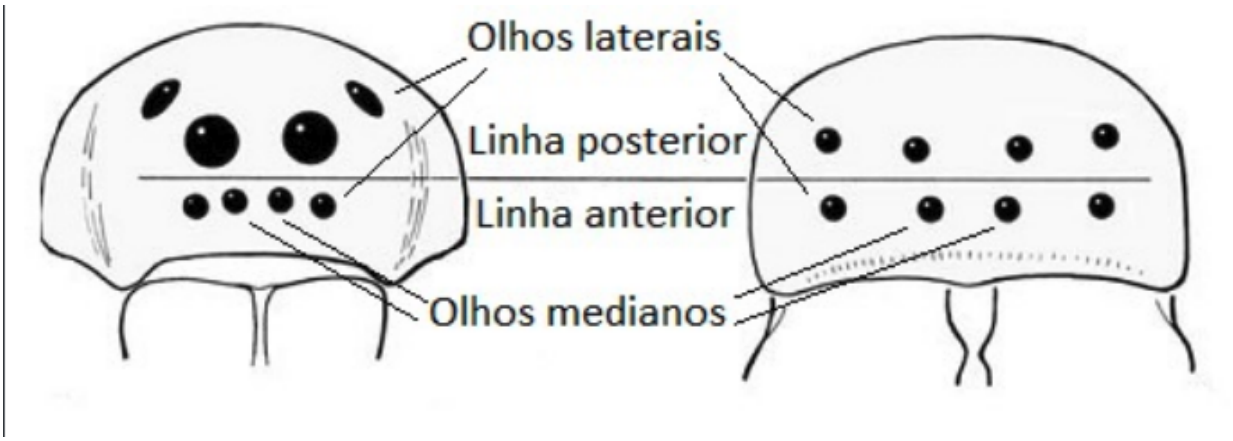

Figura 2 – Morfologia dos olhos de uma Aranha.

Fonte: (MOTTA, 2010)

Apesar da grande diversidade da região tropical, a fauna de aranhas da região ainda não é bem conhecida. Estima-se, por exemplo, que de 60% a 70% dos espécimes de aranhas orbitelas neotropicais disponíveis em coleções representem espécies novas, o que evidencia a falta de conhecimento sobre o grupo. O percentual de espécies novas neotropicais é

certamente ainda maior para grupos de aranhas menos conspícuas, presentes em habitats pouco estudados, como a serrapilheira e o dossel principalmente na Amazônia brasileira (OLIVEIRA, 2011b).

Diante da importância ecológica e médica dos aracnídeos, faz-se necessário esclarecer mitos existentes sobre esses animais e proporcionar à comunidade informações sobre as medidas preventivas necessárias para evitar acidentes domésticos por aracnídeos e proliferação destes, bem como informar que esses animais, apesar de apresentarem espécies perigosas ao ser humano, auxiliam no equilíbrio do meio ambiente.

2.2 Engenharia de Software

2.2.1 Metodologias Ágeis

Durante a evolução dos processos de Engenharia de Software, a indústria se baseou nos métodos tradicionais de desenvolvimento, que definiram por muitos anos os padrões para criação de software nos meios acadêmico e empresariais (FILHO, 2008).

As metodologias ágeis para desenvolvimento são uma resposta às metodologias tradicionais, mais focadas em documentação e produção de artefatos (SOARES, 2004). O termo "Metodologias Ágeis" se tornou popular em 2001, quando dezessete especialistas em processos de desenvolvimento de software estabeleceram princípios comuns compartilhados pelos métodos *Scrum* (SCHWABER; SUTHERLAND, 2014) e *Extreme Programming (XP)* (BECK; TOKAR, 2000).

Foram então estabelecidos 12 princípios e a publicação do "Manifesto Ágil", que tem como principais valores a valorização de indivíduos e de seu trabalho conjunto, o software como produto e a resposta a mudanças em detrimento do cumprimento a risca de um plano traçado e documentos prescritos. (FOWLER; HIGHSMITH, 2017):

Os 12 princípios apontados por Kalermo e Rissanen (KALERMO; RISSANEN, 2002) são visam priorizar a satisfação do cliente, sendo flexível com possíveis mudanças e proporcionando entregas frequentes do software funcionando. Procura proporcionar a interação entre indivíduos com objetivo transmissão de conhecimento de forma eficiente e também a simplicidade, preocupado com a melhora contínua de técnicas design e arquitetura.

As metodologias, então chamadas de ágeis, propõem a obtenção de resultados práticos em um período menor do que a indústria de software estava acostumada, tirando o foco do processo e o colocando no produto. Nestas, o desenvolvimento é iterativo, ou seja, acontece em ciclos (iterações), que têm como objetivo produzir e integrar partes do software. Cada iteração pode durar desde alguns meses até poucas horas, conforme a metodologia escolhida e as habilidades da equipe. Dessa forma, o processo se torna

flexível para acomodar mudanças funcionais e de prioridade durante a construção do sistema. No fim de cada ciclo, o software pode ser entregue ao cliente para que o restante do desenvolvimento seja direcionado pelo seu *feedback* (FILHO, 2008).

2.2.2 Scrum

Scrum é um *framework* Ágil, simples e leve, utilizado para a gestão do desenvolvimento de produtos complexos imersos em ambientes complexos. É embasado no empirismo e utiliza uma abordagem iterativa e incremental para entregar valor com frequência e, assim, reduzir os riscos do projeto. Alguns dos benefícios da tecnologia são (SABBAGH, 2014):

- entregas frequentes de retorno ao investimento dos clientes;
- redução dos riscos do projeto;
- maior qualidade no produto gerado;
- mudanças utilizadas como vantagem competitiva;
- visibilidade do progresso do projeto;
- redução do desperdício;
- aumento de produtividade.

No *Scrum*, os projetos são divididos em ciclos chamados de *Sprints*, como pode ser visto na Figura 3, que representa um período dentro do qual um conjunto de atividades deve ser executado (TELES V. M.; MELLO, 2018).

No *Product Backlog* são mantidas as funcionalidades a serem implementadas no projeto. No início de cada *Sprint*, faz-se uma *Sprint Planning Meeting*, que se trata de uma reunião de planejamento na qual o *Product Owner* prioriza os itens do *Product Backlog* e a equipe seleciona as atividades que será capaz de implementar durante a *Sprint* que se inicia. As tarefas alocadas em uma *Sprint* são transferidas do *Product Backlog* para o *Sprint Backlog* (KOHN, 2018).

Normalmente, a cada dia de uma *Sprint*, a equipe faz uma breve reunião, chamada *Daily Scrum*, com o objetivo de disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho do próximo dia de atividades.

Ao final de uma *Sprint*, as funcionalidades implementadas são apresentadas em uma *Sprint Review Meeting* e se faz uma retrospectiva da *Sprint* , onde a equipe parte para o planejamento do próximo *Sprint*. Assim, reinicia-se o ciclo (TELES V. M.; MELLO, 2018).

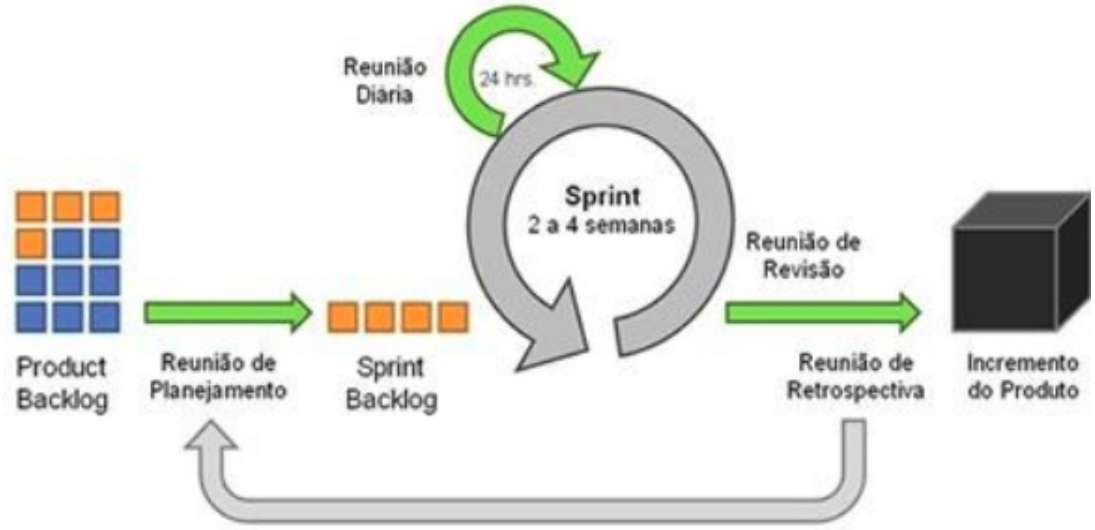

Figura 3 – Ciclo Scrum

Fonte: Autor

2.2.3 Scrum Solo

O *Scrum* Solo, criado por Pagotto, Fabri, Lerario e Goncalves (2016b) na Universidade Tecnológica Federal do Paraná, é uma customização do processo *Scrum* voltada para o desenvolvimento individual de software. O *framework* criado integra as boas práticas do *Scrum* com as prerrogativas delineadas pelo PSP, visando garantir qualidade e agilidade na produção de software (GUOPING; SHAO; ZHANG, 2010).

Figura 4 – Escopo do Scrum Solo.

Fonte: (PAGOTTO; FABRI; LERARIO; GONCALVES, 2016a)

O *Scrum Solo* possui características semelhantes ao *Scrum* tradicional. O *Product Backlog* e o *Sprint Backlog* ocorrem de maneira idêntica em ambos os *frameworks*. No *Scrum* Solo, ao final de cada *Sprint*, de forma equivalente ao *Scrum*, deve ser entregue uma versão funcional do software com novas funcionalidades e podem existir, quando

necessário, reuniões de orientação entre o grupo de validação e o desenvolvedor.

Figura 5 – Fluxo do Processo Scrum Solo.

Fonte: (PAGOTTO; FABRI; LERARIO; GONCALVES, 2016a)

As atividades realizadas no *Scrum Solo* são (GUOPING; SHAO; ZHANG, 2010) (PAGOTTO; FABRI; LERARIO; GONCALVES, 2016a):

- *Requeriment*: tem como objetivo definir o escopo do produto, caracterizar o cliente do produto e definir o *Product Backlog*.
- *Sprint*: tem como objetivo desenvolver o conjunto de itens selecionados a partir do *Sprint Backlog* em duração máxima de uma semana.
- *Deployment*: tem como objetivo disponibilizar o produto para uso do cliente.
- *Management*: tem como objetivo planejar, monitorar e controlar o desenvolvimento do produto.

Os atores envolvidos são (PAGOTTO; FABRI; LERARIO; GONCALVES, 2016a):

- *Product Owner*: caracterizado como proprietário do produto.
- Desenvolvedor Individual: responsável por executar o processo e construir o produto.
- Orientador: caracterizado como um consultor que conhece a fundo o processo.
- Grupo de Validação: possíveis usuários do produto gerado.

2.2.4 Entrega Contínua

A realização de entregas manuais de software pode expor a organização e o usuário final a riscos e gerar um grande desconforto para quem está implantando um sistema em produção (ALMEIDA, 2015).

A Entrega Contínua é um conjunto de práticas com o objetivo de garantir que um novo código esteja apto para ser disponibilizado em ambiente de produção. No entanto, o *deploy* em ambiente de produção não é automático, sendo uma decisão de negócio a ser aprovada previamente (DIAS, 2017). Trata-se de velocidade de mercado, que indica um ciclo de resposta mais curto, onde você falha mais rápido, corrige mais rápido, ajusta mais rápido e obtém êxito mais rápido, e com um tempo de geração de valor mais rápido.

Alguns aspectos positivos da Entrega Contínua são: redução dos custos e do tempo de entrega ao cliente, redução dos riscos, aumento da qualidade do produto e foco na automação (PHILLIPS, 2013).

2.2.5 Testes Automatizados de Software

Controlar a qualidade de sistemas de software é um grande desafio devido à alta complexidade dos produtos e às inúmeras dificuldades relacionadas ao processo de desenvolvimento, que envolve questões humanas, técnicas e negociais. Idealmente, os sistemas de software devem não só fazer corretamente o que o cliente precisa, mas também fazê-lo de forma segura, eficiente e escalável, além de serem flexíveis, possibilitando fácil manutenção e evolução (BERNARDO; KON, 2008).

O teste de software é uma das principais atividades realizadas para melhorar a qualidade de um produto em desenvolvimento. Seu principal objetivo é revelar a presença de erros no software o mais cedo possível no ciclo de desenvolvimento, buscando minimizar o custo da correção dos mesmos (FANTINATO et al., 2005).

A execução manual de um caso de teste é rápida e efetiva, mas a execução e repetição de um vasto conjunto de testes manualmente é uma tarefa muito dispendiosa e cansativa. É normal e compreensivo que os testadores não verifiquem novamente todos os casos a cada mudança significativa do código, e é deste cenário que surgem os erros de software, trazendo prejuízo para as equipes de desenvolvimento, que perdem muito tempo para identificar e corrigir os erros, além de prejuízo para o cliente que, entre outros problemas, sofre com constantes atrasos nos prazos combinados e com a entrega de um software de qualidade duvidosa (BERNARDO; KON, 2008).

A automação do teste consiste em repassar para o computador tarefas de teste de software que seriam realizadas manualmente, sendo feita geralmente por meio do uso de ferramentas de automação de teste (FANTINATO et al., 2005). São programas ou *scripts* simples que exercitam funcionalidades do sistema, fazendo verificações automáti-

cas nos efeitos colaterais obtidos. A grande vantagem desta abordagem é que todos os casos de teste podem ser facilmente e rapidamente repetidos a qualquer momento e com pouco esforço. A reprodutibilidade dos testes permite simular identicamente e inúmeras vezes situações específicas, garantindo que passos importantes não serão ignorados por falha humana e facilitando a identificação de um possível comportamento não desejado (BERNARDO; KON, 2008).

2.2.6 Desenvolvimento de Aplicação Mobile

Com a crescente evolução das tecnologias da informação e a progressiva miniaturização dos componentes, um novo paradigma de processamento e armazenamento de informações surgiu, conhecido como Computação Móvel (RODRIGUES; PRADO, 2014). Segundo El-Kassas, A.Abdullah, H.Yousef e M.Wahba (2015), o desenvolvimento de aplicativos móveis se diferencia do desenvolvimento de outros tipos de software por possuir particularidades e restrições, tais como:

- Experiência de uso: é importante que as aplicações sejam simples e que possuam uma interface amigável para atender às expectativas dos usuários. Caso contrário, devido ao grande número de aplicações disponíveis, o mesmo pode ser descartado e substituído.
- Ecossistema heterogêneo: O desenvolvimento de aplicações móveis se encontra em um contexto heterogêneo devido aos diferentes sistemas operacionais e à grande quantidade de dispositivos distintos, com variações de poder computacional, configurações de hardware e tamanhos de tela. Essas singularidades devem ser consideradas no desenvolvimento de aplicativos, o que pode implicar na necessidade de diferentes versões de uma mesma aplicação.
- Manutenção: plataformas mobile passam por constantes atualizações que podem afetar aplicativos já desenvolvidos, a ponto de torná-los inutilizáveis ou causarem desconforto aos usuários. Para que a aplicação continue a funcionar corretamente, são necessárias frequentes manutenções e atualizações.

Para o desenvolvimento de aplicações móveis, existem duas formas de implementação: nativa e híbrida.(LIM, 2015). Aplicações nativas são desenvolvidas com o uso de ferramentas e linguagens de programação específicas para determinada plataforma, usando o SDK e *frameworks* providos por ela. Os aplicativos ficam vinculados a esse ambiente, executando apenas nos dispositivos da plataforma alvo (EL-KASSAS; A.ABDULLAH; H.YOUSEF; M.WAHBA, 2015).

Aplicativos híbridos são desenvolvidos utilizando ferramentas de desenvolvimento web em junção com elementos nativos (SERRANO; HERNANTES; GALLARDO, 2013),

sendo em essência aplicações web empacotadas em um aplicativo nativo, segundo Stark (2010).

2.3 Aplicações Afins e Similares

Atualmente, com a ascensão da tecnologia não somente na base científica, mas também na divulgação da ciência, surgem aplicações com o intuito de prover informações ao público leigo, descentralizando o conhecimento dos meios tradicionais.

Assim, temos aplicações de divulgação científica voltadas para as áreas Biológicas, por exemplo. Uma delas, que inclusive alimentará e consistirá parte da base de dados da aplicação descrita no presente trabalho, é o *WSCA - World Spider Catalog*. O WSCA é uma associação suíça, mantida pelo *Natural History Museum of Bern*, para a promoção de pesquisas taxonômicas e sistemáticas em aracnologia e campos relacionados, coletando toda a literatura sobre aracnologia relevante para pesquisa e torna-a acessível (NMBE, 2018).

Existem ainda aplicações para identificação de espécies, como o *TriadoDex*, criado por Maxwell Almeida, Douglas Rocha, Andrey Andrade e Jainaine Abrantes. Este tem objetivo de facilitar a identificação morfológica de triatomíneos (*Hemiptera: Reduviidae*) por meio de chaves dicotômicas, fornecendo informações sobre os descritores, distribuição geográfica, tamanho, habitats e importância médica (GURGEL-GONçALVES; ALMEIDA; BATISTA; ROCHA, 2017). E também o *LutzoDex*, para identificação de fiebotomíneos (*Diptera: Phlebotominae*), sendo um aplicativo do mesmo autor e seguindo os mesmos princípios do *TriadoDex* (ROCHA; ALMEIDA; BATISTA; ANDRADE, 2015).

3 Metodologia

Para a execução do projeto foram feitas escolhas de tecnologias baseadas em critérios que corroborassem para o sucesso da implementação. Para implementação da API *webservice*, foi escolhida a linguagem *Java*, por ser uma linguagem consolidada no mercado, sendo a quinta linguagem de programação mais utilizada na produção de software (atrás somente de *JavaScript, HTML, CSS, SQL*) (AGRELA, 2018), garantindo maior manutenibilidade para futuras implementações.

Para o *frontend*, foi escolhido o *framework Ionic* por permitir o desenvolvimento hibrido (*Android e iOS*), mesmo que a princípio a aplicação seja desenvolvida com foco em *Android*.

Para metodologia de desenvolvimento de software, foi escolhida a metodologia Ágil *Scrum* principalmente pela sua adaptabilidade, velocidade de desenvolvimento, transparência e melhoria contínua.

3.1 Metodologia de Desenvolvimento

Nessa seção será abordada a definição do processo de desenvolvimento, bem como os recursos e ferramentas utilizados no projeto, esclarecendo as escolhas de ferramentas e os métodos de implantação. Também é apresentada a arquitetura e a o modelo de dados desenvolvidos.

3.1.1 Scrum

Como afirma SOARES (2004), as melhores metodologias para modelagem de aplicações baseadas na Internet são as ágeis, pelo dinamismo do ambiente web. Levando em consideração esta informação e as vantagens explicitadas em 2.2.2, foi escolhida pelo autor deste TCC para o desenvolvimento do software de catalogação e difusão de dados de espécies uma adaptação da metodologia ágil *Scrum*, adotando algumas características do *Scrum Solo*, elucidado em 2.2.3.

3.1.1.1 Processo

Foi desenvolvido para guiar a execução do presente projeto o processo das *Sprints*, ilustrado na Figura 6, contemplando quatro fases:

- Elicitação de Requisitos, onde são levantadas e/ou refinadas as necessidades que a aplicação deve suprir, bem como a especificação das histórias a serem desenvolvidas,

por meio das técnicas *Brainstorm* e entrevista.

- Planejamento, fase onde as histórias são priorizadas planejadas
- Desenvolvimento, etapa em que o código fonte é desenvolvido, bem como os testes unitários, responsáveis por testar pontualmente a lógica do código.
- Implantação, etapa em que é realizada a integração do código desenvolvido na *Sprint* e a validação junto ao usuário final. Também aqui são colhidas as métricas de qualidade do software e registradas as possíveis melhorias e correções.

3.1.1.2 Papéis

Uma vez que o presente trabalho é desenvolvido de forma individual, os papéis da metodologia *Scrum* foram mesclados com da metodologia *Scrum Solo* e distribuídos da seguinte maneira: o responsável pelo desenvolvimento da aplicação, Pedro Sales, assume o papel de Time de Desenvolvimento (Desenvolvedor Individual no *Scrum Solo*) e *Scrum Master*. O papel de *Product Owner* é atribuído ao Professor Dr. Paulo César, conhecedor da área de aplicação da solução, e eventualmente por Pedro Sales. O papel de Orientador é atribuído ao Professor Dr. Wander Cleber Pereira.

3.1.1.3 Sprints

Para o presente projeto, as *Sprints* são divididas com duração de duas semanas. As atividades inerentes a *Sprint* são:

- Planejamento de *Sprint*: Nessa atividade são selecionados os itens do *Product Backlog* a serem desenvolvidas durante o período da *Sprint*. É realizada no primeiro dia da *Sprint*;
- Revisão da *Sprint*: Nessa atividade são apresentadas e discutidas junto ao *Product Owner* as histórias de usuário implementadas durante a *Sprint*. É realizada ao final da *Sprint*;

3.1.2 Hardware

Para o desenvolvimento desse projeto foi utilizado um notebook para desenvolvimento, implementação e implantação de código, testes funcionais e não funcionais, integrações e documentação. Foi também utilizado um *smartphone* para validação e análise de produto final. As especificações dos mesmos seguem abaixo.

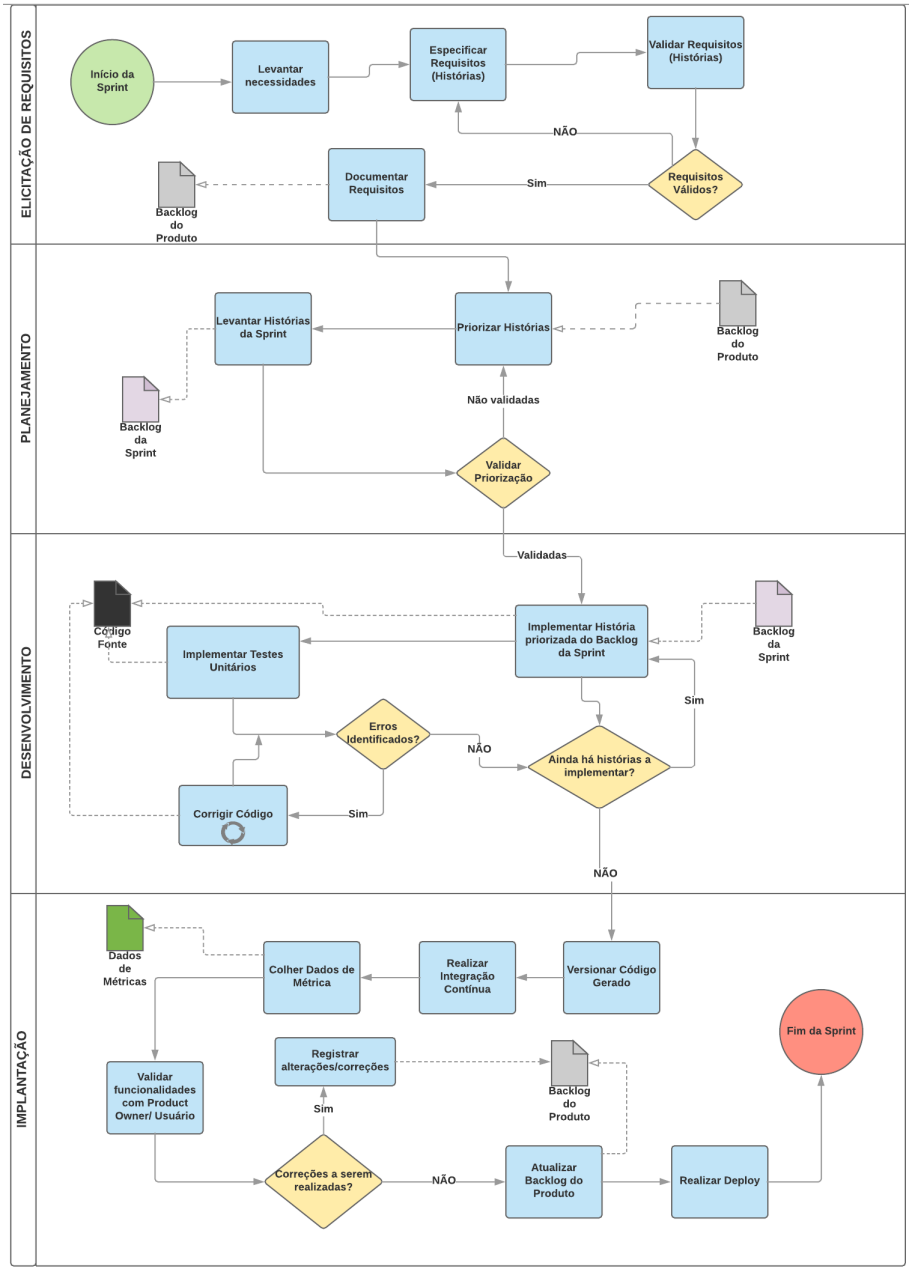

Figura 6 – Processo de Desenvolvimento.

Fonte: Autor

Tabela 1 – Equipamentos utilizados

Tipo	**Sistema Operacional**	**Modelo**
Computador	Linux Fedora 27	Inspirion 14
Smartphone	Android 6.0.1	Umidigi Plus E

3.1.3 Ferramentas Utilizadas

Nessa sessão serão descritas as diversas ferramentas utilizadas nas diferentes fases do processo de desenvolvimento da aplicação.

3.1.3.1 Linguagens e Frameworks

Para o desenvolvimento da parte mobile da aplicação, é utilizado o *framework Ionic 3*. Este é um *framework* criado em 2012 pela *Drifty Co.* baseado em *Angular* e *Cordova*, contando com uma gama de *plugins* e funcionalidades nativas para desenvolvimento de aplicações de dispositivos móveis, visando o desenvolvimento de apps híbridas e de rápido e fácil desenvolvimento. A linguagem de programação utilizada pelo *Ionic 3* é *TypeScript 2.2*, uma linguagem fortemente tipada, que possibilita a aplicação de orientação a objetos em aplicações *JavaScript*. O *TypeScript* é compilado e transformado em *JavaScript* (MICROSOFT, 2017). Para a escrita do código foi utilizado o Visual Studio Code.

Junto com o desenvolvimento do aplicativo foi desenvolvida em paralelo a camada de serviços, uma API *RESTful* escrita na linguagem de programação Java (JDK 8), utilizando *frameworks* do *Spring 5.0*. A IDE utilizada para o desenvolvimento da API foi o IntelliJ IDEA.

O *Spring* é um *framework open source* para a plataforma *Java* criado por Rod Johnson. Trata-se de um *framework* não intrusivo, baseado nos padrões de projeto injeção de dependência e inversão de controle (IoC) (SPRING, 2018).

Para a realização dos testes unitários da API *Java* da aplicação, é utilizado o *framework JUnit 4.12* . *JUnit* é um *framework* que facilita o desenvolvimento e execução de testes unitários em código *Java* (JUNIT, 2018).

3.1.3.2 Métricas

Para as métricas de código-fonte da API *Java* e para a aplicação mobile é utilizado o *CodeClimate v0.82.0*, uma ferramenta de análise estática de qualidade de código, criada por *Bryan Helmkamp* para análise de código *Ruby* e posteriormente evoluída para análise em linguagens *Python, Java, JavaScript* entre outros (CODECLIMATE, 2018).

3.1.3.3 Integração e Entrega Contínua

Na Integração Contínua do projeto é utilizado o *TravisCI*, um serviço web de IC na nuvem integrado com o *GitHub*. O *Travis CI* suporta a integração com ferramentas externas, como analisadores de cobertura ou analisadores estáticos. Na integração que o *Travis* realiza com o *GitHub*, a cada *commit* ou *pull request* o *Travis* executa seu *job* e marca visualmente se o ambiente foi comprometido ou não, gerando *badges* que informam o status do projeto. Ele possui ainda um sistema de notificações, que a cada *build* construída é enviado um e-mail informando o status da *build*. O *Travis* suporta vários tipos de linguagem como *C#, Java, Ruby, Python, Haskell*, entre outras (TRAVISCI, 2018).

É utilizado também o *Heroku*, uma plataforma que para criar, monitorar, entregar e escalar aplicativos. Normalmente utilizado para o *deploy* de aplicações *web* , a plataforma também hospeda serviços na nuvem, facilitando a arquitetura de aplicativo-API (HEROKU, 2018).

3.1.4 Arquitetura

3.1.4.1 Arquitetura da API Java

A arquitetura do presente projeto, como pode ser visto na Figura 7, segue um modelo adaptado do MVC (*Model View Controller*) em sua API *Java*, sendo as funcionalidades disponibilizadas em uma camada de controladora de serviços, acessível via requisição anônima *RESTfull* pela aplicação mobile (*View* externa). Esta camada de controladora se comunica com uma camada de serviços, fazendo a intermediação com o modelo de dados implementado. Por fim, a *model*, espelhada no modelo de dados implementado, é utilizada pelas *Repositorys* para acesso ao banco de dados em memória (IMDB).

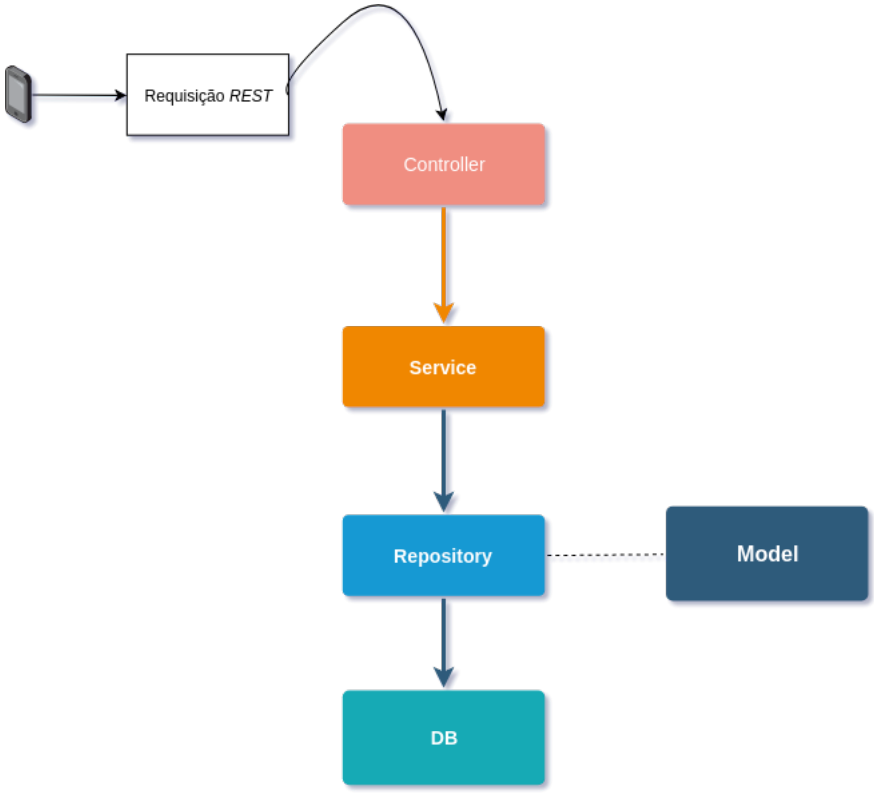

Figura 7 – Modelo Arquitetural da Aplicação.

Fonte: Autor

3.1.4.2 Arquitetura do App Ionic

A arquitetura de pacotes da aplicação Ionic pode ser representada no modelo ilustrado na Figura 9:

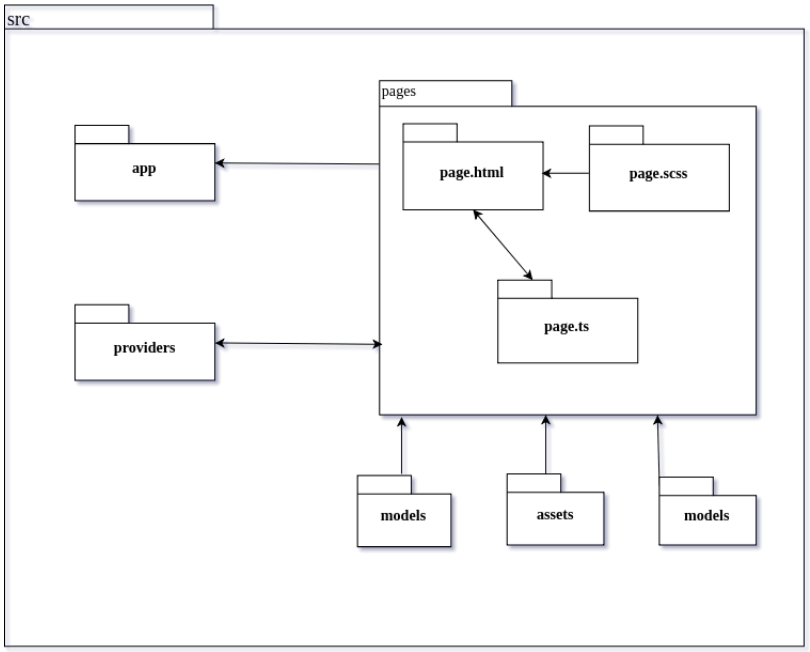

Figura 8 – Diagrama de pacotes.

Fonte: Autor

- **src**: é o diretório fonte da aplicação, onde os principais recursos serão salvos;
- **app**: onde o framework reúne todos os recursos a serem compilados;
- **pages**: contém as páginas do aplicativo. Cada página é composta de HTML, SCSS, que compõem o visual da página e o Typescript que é onde está a parte lógica;
- **providers**: contém os métodos que fazem a comunicação com a API;
- **models**: guarda os modelos de entidades utilizadas na aplicação;
- **assets**: possui arquivos de imagens utilizados pela aplicação;
- **utils**: contém utilitários utilizados pela aplicação.

3.1.5 Modelo de Dados

O modelo de dados utilizado na aplicação é representado pelo modelo de banvo de dados ilustrado na Figura 9

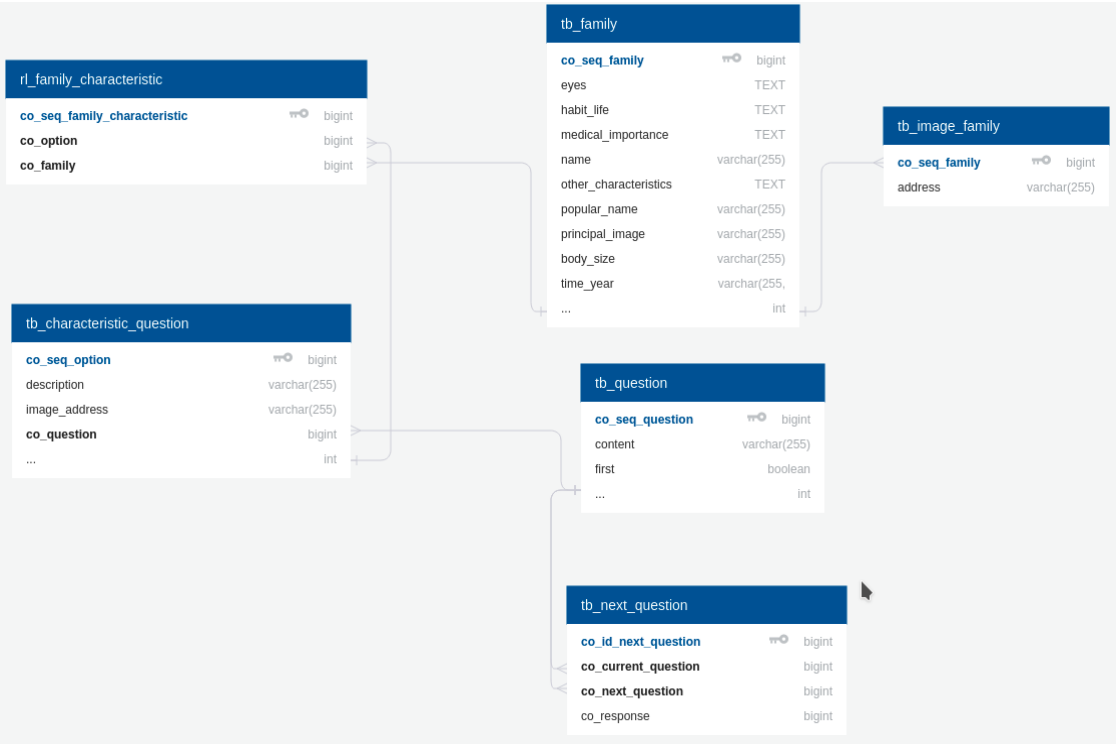

Figura 9 – Modelo de Banco de Dados.

Fonte: Autor

Os dados que alimentam a aplicação foram extraídos de planilhas de pesquisa fornecidas pelo professor Motta (2010) da Universidade de Brasília, baseadas em sua obra Aracnídeos do Cerrado.

4 Resultados

O presente capítulo tem por objetivo apresentar e discutir os resultado obtidos com a aplicação do processo definido utilizando as ferramentas apresentadas anteriormente. Serão abordadas as funcionalidades desenvolvidas no *Software*, bem como suas telas e seu mecanismo de funcionamento. Ao final serão apresentadas as métricas colhidas e a cobertura de testes final da API Java.

4.1 Aplicação desenvolvida

Foi desenvolvido neste trabalho um aplicativo para consulta por famílias de aracnídeos por meio de filtro de nomenclatura ou por características observáveis aplicando chaveamento politômico. De tal modo, a aplicação possui duas funcionalidades de pesquisa distintas e uma funcionalidade de detalhamento de resultado.

4.1.1 Consulta por Nomenclatura

A funcionalidade de Consulta por Nomenclatura diz respeito a um mecanismo implementado de consulta simples por nome de famílias de aracnídeos (inicialmente das regiões de Cerrado Brasileiro) ou por seu nome popular. A consulta foi implementada de forma responsiva, de maneira a filtrar dinamicamente resultados em tempo de digitação pelo usuário, como mostrado na Figura 10. Na lista de resultados é apresentado o nome da família em destaque, seguido pelos nomes populares cadastrados e uma imagem de exemplificação, podendo esta ser amplificada ao toque para melhor visualização.

Inicialmente, caso não seja digitado o nome no filtro de pesquisa, é apresentada a lista completa de 16 famílias cadastradas na base de dados. O mecanismo é apresentado da Figura 11.

4.1.2 Consulta por Chave Politômica

O segundo mecanismo de consulta da aplicação é baseado em chaveamento politômico. Neste o usuário busca a família de aracnídeos baseando-se em características observáveis, tais como local de ocorrência, tamanho, cor, dentre outros aspectos que podem ser cadastrados na base de dados.

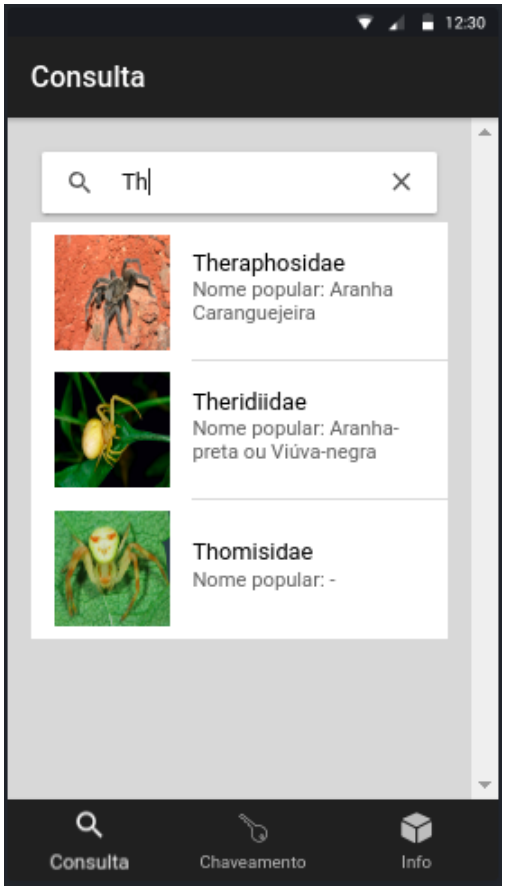

Figura 10 – Tela de Consulta.

Fonte: Autor

4.1.2.1 Estratégia de Consulta por Chaveamento Politômico

A estratégia implementada para a Consulta por Chaveamento Politômico é baseada em fluxos de questões. Cada questão cadastrada possui duas ou mais alternativas de respostas, que correspondem a características associadas às famílias de aracnídeos. Para cada alternativa de questões é possível realizar um encadeamento com a próxima pergunta a ser realizada ao usuário, de maneira a manter um fluxo que faça sentido no que diz respeito às características informadas. O modelo de fluxos de perguntas é representado na Figura 12.

A primeira questão realizada é caracterizada como um índice (podendo haver mais de um, como exemplificado na Figura 12), a partir da qual se inicia um fluxo de questões a serem realizadas. A medida que as opções são escolhidas, a característica correspondente à alternativa é guardada e então apresentada a nova questão, linkada a alternativa selecionada. Além das opções cadastradas para as questões, existe uma alternativa comum entre todas as questões, que é "Não vi/ não sei". Ao selecionar esta alternativa, o fluxo atual de questões é quebrado e então buscado um novo índice para o fluxo de questões. Caso não exista um outro índice cadastrado que ainda não tenha sido apresentado, e a quantidade de famílias correspondentes ainda seja elevada, é executada uma consulta por questões

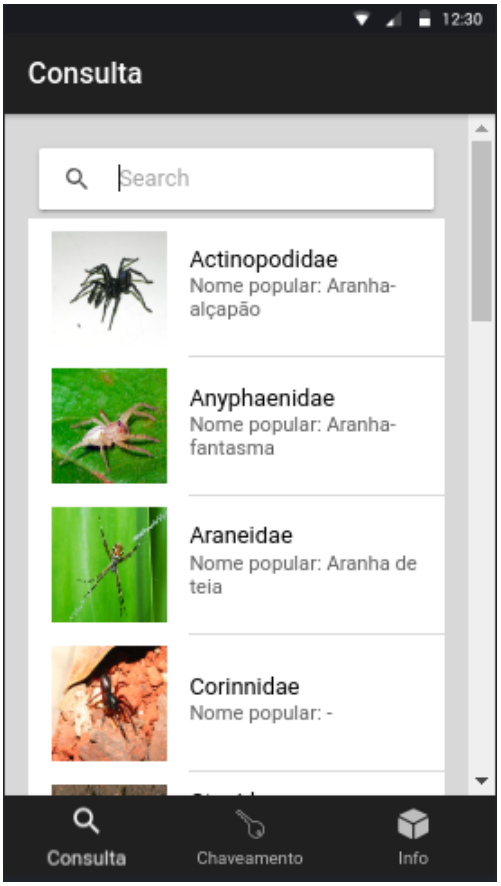

Figura 11 – Tela de Listagem.

Fonte: Autor

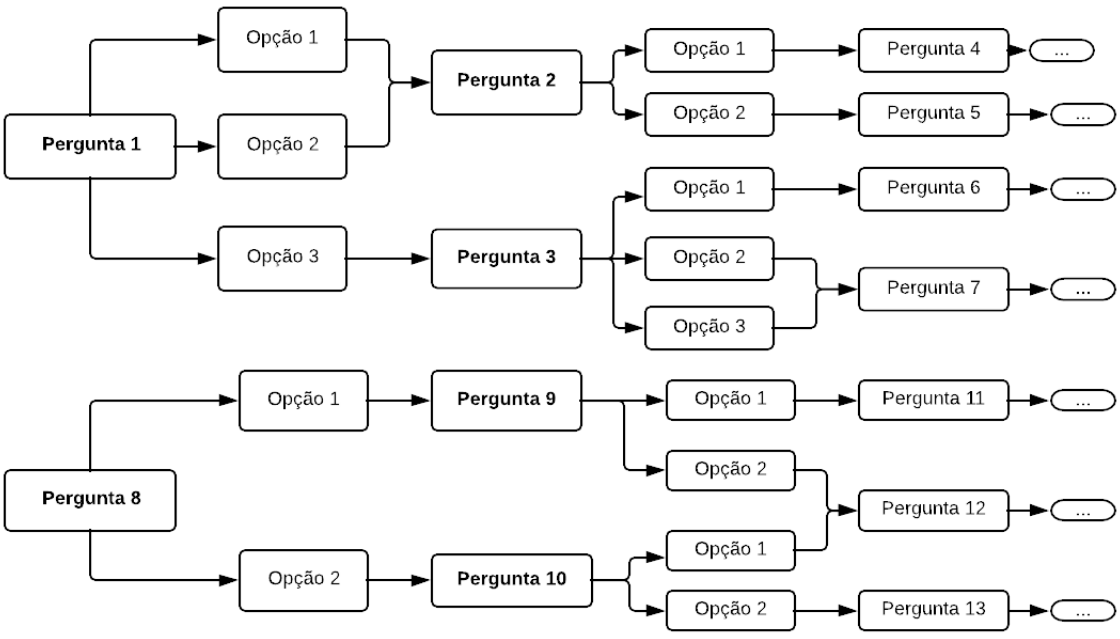

Figura 12 – Fluxo Lógico de Questões.

Fonte: Autor

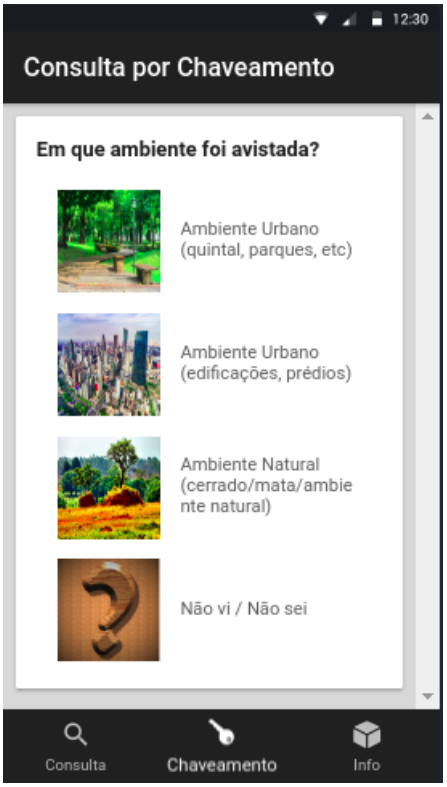

Figura 13 – Tela de Consulta por Chaveamento. Fonte: Autor

Fonte: Autor

aleatórias, a fim de se obter maior quantidade de características, porém impedindo que sejam apresentadas questões repetidas.

A cada resposta selecionada, é realizada uma consulta na base de dados por famílias de aracnídeos que correspondam às características guardadas até o momento. Aqui se encontra o primeiro mecanismo de parada. Caso seja encontrado somente um correspondente para as características, acabam-se as perguntas e a família encontrada é apresentado como sugestão.

Antes de serem apresentadas as alternativas de questões, é executada uma validação das alternativas. Caso a possível alternativa corresponda a uma característica que, se adicionada às características guardadas, não retorne nenhuma família correspondente, esta é retirada da lista de opções. Caso todas as alternativas sejam retiradas, é buscada uma questão raíz ou aleatória.

O segundo mecanismo de parada é quando esgotam-se as questões cadastradas no banco (atualmente 8). Ao chegar neste ponto, são apresentadas todas as famílias que possuem as características informadas até então.

A qualquer momento deste fluxo, o usuário pode selecionar a opção "Recomeçar", a fim de reiniciar a identificação.

4.1.3 Detalhamento

Seja por meio da Consulta por Nomenclatura ou pela Consulta por chave Politômica, ao final do fluxo o usuário pode acessar o detalhamento do resultado, ilustado na Figura 14 tendo informações a respeito da família de aracnídeos. Neste detalhamento são apresentadas as seguintes informações:

- **Nome Popular:** Nomes pelos quais é popularmente reconhecida;
- **Importância Médica:** Riscos a saúde que as espécies da família podem causar;
- **Hábito de Vida:** Informações de hábito de vida das espécies da família, como períodos em que se expõem e tipo de ambiente onde comumente são avistadas;
- **Tamanho:** Tamanho médio do corpo das espécies em fase adulta, desconsiderando as patas;
- **Olhos:** Disposição dos olhos das espécies, fundamentais para a identificação destas;
- **Época do Ano:** Períodos do ano em que são mais facilmente avistadas;
- **Outras Características:** Outras informações relevantes a respeito das espécies da família;
- **Fotos:** Fotos de exemplares.

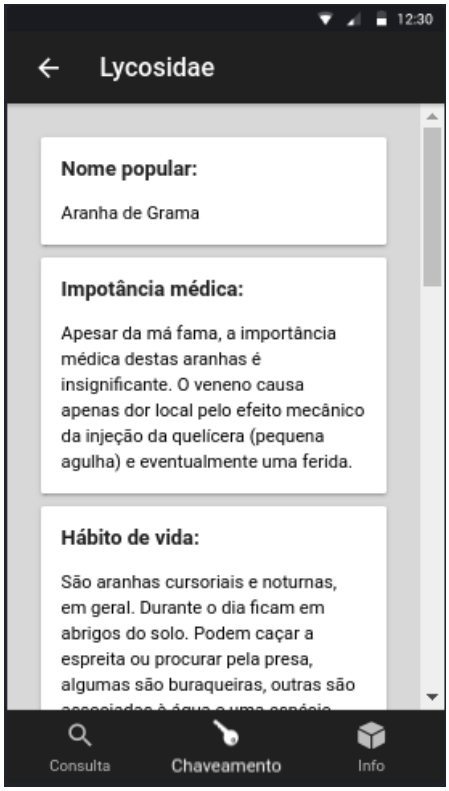

Figura 14 – Tela de Detalhamento.

Fonte: Autor

4.1.4 Informações

Na Tela de Informações, Figura 15, é apresentada a logo do aplicativo e informações importantes (fornecidas pelo Instituto de Biologia da Universidade de Brasília, por meio do professor Paulo César) acerca da relevância de características para a identificação visual de famílias de aracnídeos. Também são apresentadas informações sobre o desenvolvimento e o contexto geral da aplicação.

4.2 Testes Unitários

A fim de se manter a integridade e o pleno funcionamento dos mecanismos da ferramenta, foram implementados testes unitários na API Java, utilizando JUnit (JUNIT, 2018). Ao final do desenvolvimento da aplicação, a cobertura de testes da aplicação estava com 100% para os principais módulos da aplicação: Controller e Service, que foram priorizados em detrimento do módulo Model, por conterem mecanismos fundamentais para o funcionamento da aplicação, enquanto os métodos da Model são responsáveis majoritariamente por recuperar valores do modelo. Os dados foram extraídos pela ferramenta nativa *Coverage* do IntelliJ.

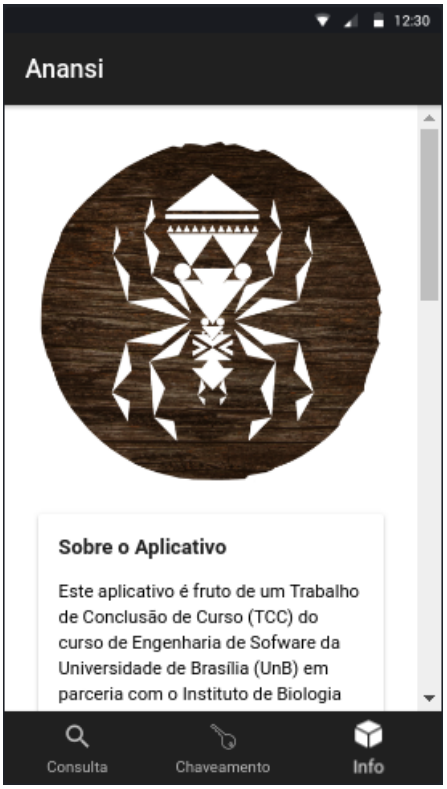

Figura 15 – Tela de Informacoes.

Fonte: Autor

Overall Coverage Summary

Package	Class, %	Method, %	Line, %
all classes	100% (11/ 11)	42,3% (33/ 78)	40,7% (55/ 135)

Coverage Breakdown

Package ▲	Class, %	Method, %	Line, %
br.com.anansi	100% (1/ 1)	50% (1/ 2)	33,3% (1/ 3)
br.com.anansi.controller	100% (3/ 3)	100% (10/ 10)	100% (14/ 14)
br.com.anansi.model	100% (5/ 5)	21,4% (12/ 56)	14,1% (12/ 85)
br.com.anansi.service	100% (2/ 2)	100% (10/ 10)	84,8% (28/ 33)

Figura 16 – Cobertura de Testes.

Fonte: IntelliJ IDEA

4.3 Métricas de Código Fonte

As métricas foram extraídas da API Java e da aplicação Ionic pela ferramenta CodeClimate. Na Figura 17 e Figura 18 são apresentadas exemplos de telas de relatório da ferramenta.

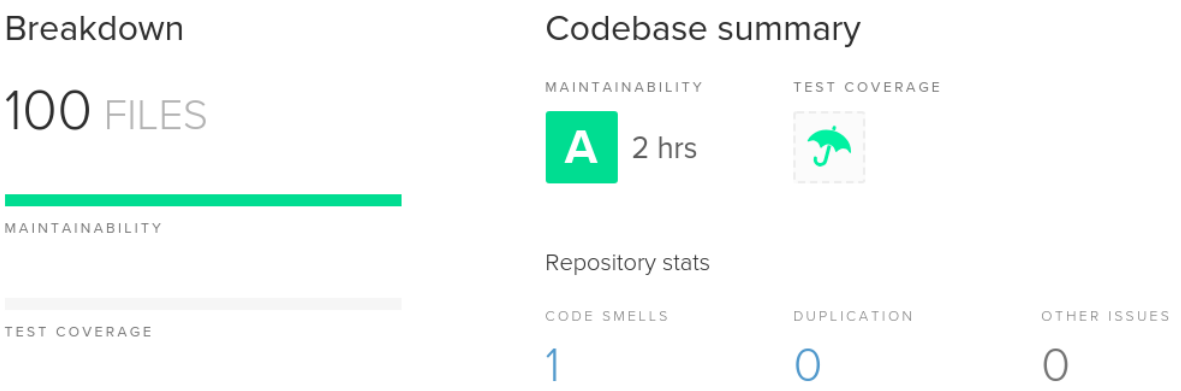

Figura 17 – Métricas Extraídas da API Java.

Fonte: CodeClimate

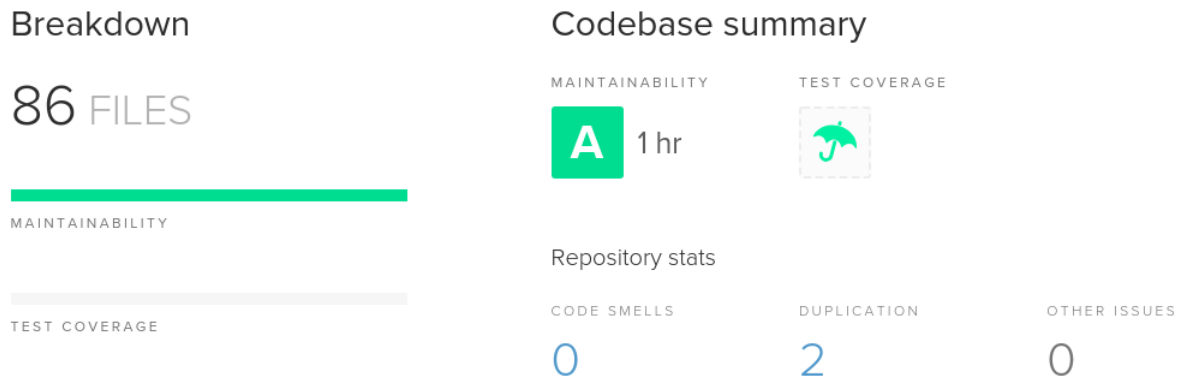

Figura 18 – Métricas Extraídas do APP Ionic. Fonte: CodeClimate

Fonte: CodeClimate

4.3.1 Integração Contínua

Por meio da ferramenta Travis CI foi realizada a Integração e Entrega Contínua da API Java. A ferramenta realiza a validação do pacote gerado, emitindo notificação a respeito da integridade do pacote após alterações realizadas na *master*, *branch* principal do repositório Git.

Na Figura 19 é possível visualizar um relatório de *Build* bem sucedida.

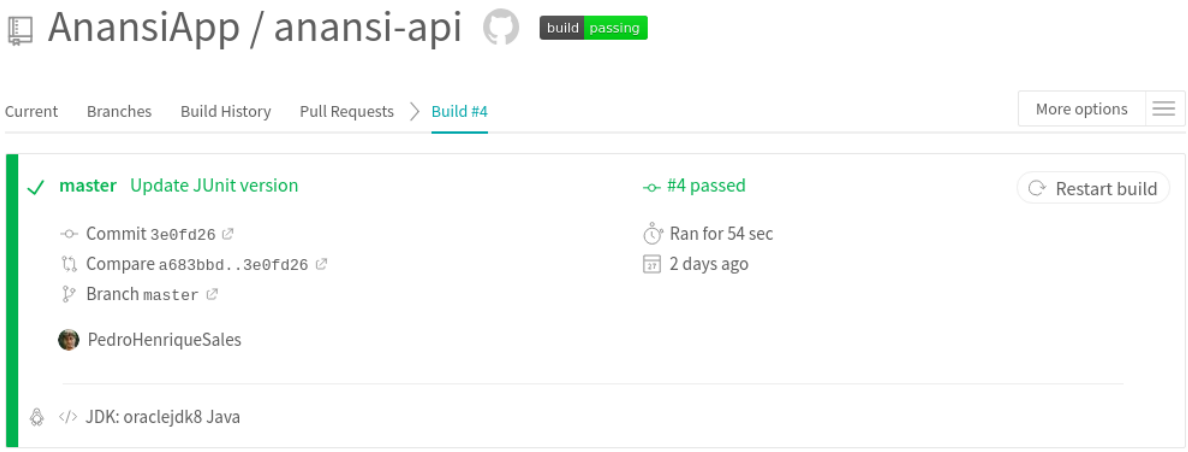

Figura 19 – Build do TravisCI.

Fronte: TravisCI

4.3.2 Deploy

O *deploy* da API Java é efetuado no Heroku, onde a mesma fica disponibilizada para acesso, podendo então ser utilizada pela aplicação mobile.

A Figura 20 exibe um relatório de atividades de deploys na ferramenta.

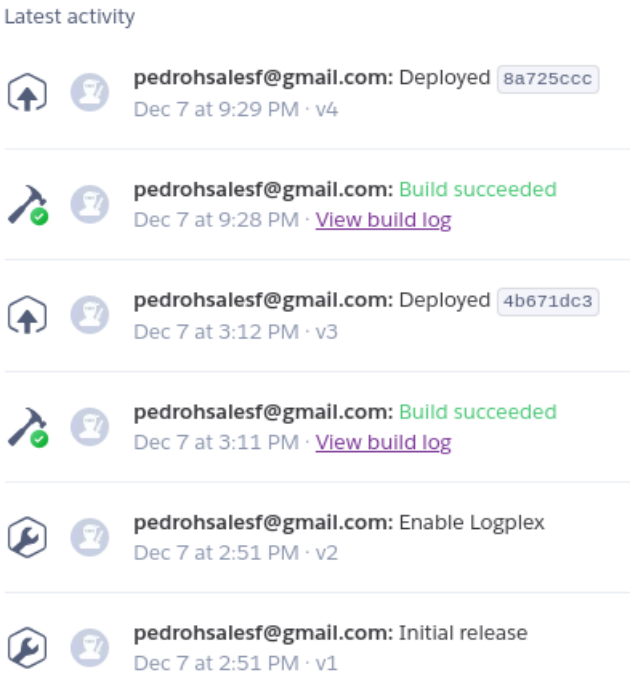

Figura 20 – Relatório de Atividades Heroku.

Fonte: Heroku

4.3.3 Requisitos mínimos

Os requisitos mínimos necessários para instalação do aplicativo Anansi são: Android versão 4.1 ou superior, 22MB livres em disco e 500MB de memória RAM.

5 Conclusões

5.1 Considerações Finais

Utilizando conhecimento em Engenharia de Software, tais como desenvolvimento com processo ágil, métricas de código, integração e entrega contínua, testes de software entre outros, foi possível realizar um trabalho interdisciplinar, unindo a Engenharia de Software com a Biologia, corroborando na construção de um aplicativo para consulta e identificação de famílias de aracnídeos, inicialmente do bioma Cerrado.

A utilização da tecnologia Java foi um facilitador do desenvolvimento, uma vez que essa é amplamente difundida, sendo possível encontrar materiais de apoio com considerável facilidade. Outro benefício da escolha é que esta também era uma linguagem de conhecimento do desenvolvedor.

A curva de aprendizagem na linguagem Ionic foi relativamente rápida. Porém o desconhecimento da mesma pelo desenvolvedor acarretou um atraso no inicio do processo de desenvolvimento, tornando uma primeira entrega do produto mais demorada que o previsto. Entretanto, a utilização da linguagem proporcionou aprendizado de grande valia em desenvolvimento mobile multi-plataforma. Mesmo o foco do desenvolvimento da aplicação sendo voltada para plataforma Android, o resultado final pode facilmente ser disponibilizado também nas plataformas Ios e Windows Phone.

As ferramentas de apoio (TravisCI, Heroku, CodeClimate, Git, JUnit, entre outras) associadas a metodologia adotada possibilitaram um desenvolvimento mais ágil e conciso, favorecendo ao cumprimento dos prazos estipulados.

O desenvolvimento de uma aplicação para uma área completamente distinta também proporcionou uma experiência de aprendizado grandiosa. A área de Biologia é de fundamental importância para a sociedade, e promover a divulgação científica acerca de aracnídeos, classe dos artrópodes tão comum porém tão pouco conhecidas pelo público leigo, foi um trabalho prazeroso de ser executado.

5.2 Trabalhos Futuros

Como trabalhos futuros para a aplicação, podem ser listadas as seguintes evoluções pretendidas para o sistema:

- **Ambiente Web de Administração:** A implementação de uma ferramenta de administração, a ser utilizada pelo Instituto de Biologia da UnB, a fim de realizar a

manutenção e cadastramento das informações das famílias disponíveis para consulta na aplicação, bem como a criação de novos fluxos de questões, possibilitando maior assertividade na consulta por chaves politômicas. A API já encontra-se apta para a modelagem arquitetural desta nova funcionalidade.

- **Funcionalidade de mapeamento de espécies:** A fim de promover além da divulgação científica, mas também a produção de dados científicos, planeja-se implementar uma ferramenta para mapeamento controlado de espécies, a fim de se gerar um banco de geolocalização de espécies.

- **Validação na Integração e Deploy:** Inserção de novas políticas de controle de qualidade de código, a fim de proporcionar entregas no repositório com alta qualidade.

Referências

AGRELA, L. **Estas são as 20 linguagens de programação mais usadas**. 2018. Disponível em: <https://exame.abril.com.br/tecnologia/estas-sao-as-20-linguagens-de-programacao-mais-usadas/>. Acesso em: 05 junho 2018. Citado na página 26.

ALBAGLI, S. **Divulgação científica: informação científica para a cidadania?** [S.l.]: Revista Ibict, 1996. Citado na página 16.

ALMEIDA, A. Entrega contínua: Um estudo de caso para automatização do fluxo de implantação do sistema de integra. 2015. Citado na página 23.

BECK, K.; TOKAR, I. **Extreme Programming Das Manifest**. München: Addison Wesley, 2000. Citado na página 19.

BERNARDO, P. C.; KON, F. A importância dos testes automatizados. Engenharia de Software Magazine, 2008. Citado 2 vezes nas páginas 23 e 24.

BERTOLETTI, J. J. **Museu de Ciências e Tecnologia da PUC-RS**. 2003. Disponível em: <http://www.comciencia.br/dossies-1-72/reportagens/cultura/cultura16.shtml>. Acesso em: 01 junho 2018. Citado na página 16.

BUSTAMANTE, J. A integração da ciência, tecnologia e sociedade: o grande desafio da educação no século xxi. Revista Educação Brasileira, 1997. Citado na página 17.

CODECLIMATE. **About CodeClimate**. 2018. Disponível em: <https://docs.codeclimate.com/>. Acesso em: 10 junho 2018. Citado na página 29.

DIAS, R. R. **Diferenças entre Integração, deploy e entrega contínua**. 2017. Disponível em: <https://www.4linux.com.br/diferencas-entre-integracao-deploy-e-entrega-continua/>. Acesso em: 12 junho 2018. Citado na página 23.

DINIZ-FILHO, J. A. F.; MARCO, P. D.; HAWKINS, B. A. **Defying the curse of ignorance: perspectives in insect macroecology and conservation biogeography. Insect Conservation and Diversity**. [S.l.: s.n.], 2010. Citado na página 14.

DUARTE, J. Da divulgação científica à comunicação. Associação Brasileira de Jornalismo Científico, 2004. Citado na página 16.

EL-KASSAS, W. S.; A.ABDULLAH, B.; H.YOUSEF, A.; M.WAHBA, A. Taxonomy of cross-platform mobile applications development approaches author links open overlay panel. Ain Shams Engineering Journal, 2015. Citado na página 24.

FANTINATO, M. et al. Autotest – um framework reutilizável para a automação de teste funcional de software. researchgate, 2005. Citado na página 23.

FILHO, D. L. B. Experiências com desenvolvimento ágil. USP, 2008. Citado 2 vezes nas páginas 19 e 20.

FIRME, R. do N.; SILVA, P. do N. Divultação científica: Analisando modelos de comunicação da ciência e tecnologia e implicações para o letramento científico e tecnoloógico. Extensio - Revista Eletrônica de Extensão, 2016. Citado na página 17.

FOELIX, R. F. **Biology of Spiders**. [S.l.]: Oxford University Press, 2011. Citado na página 17.

FOWLER, M.; HIGHSMITH, J. **Manifesto for Agile Software Development**. 2017. Disponível em: <https://http://agilemanifesto.org/>. Acesso em: 30 abril 2018. Citado na página 19.

GUOPING, R.; SHAO, D.; ZHANG, H. Scrum-psp: Embracing process agility and discipline. 17th Asia Pacific Software Engineering Conference (APSEC'10), 2010. Citado 2 vezes nas páginas 21 e 22.

GURGEL-GONçALVES, R.; ALMEIDA, M. R. de; BATISTA, J. A. de S.; ROCHA, D. de A. New interactive key for species identification of the subfamily triatominae (hemiptera, reduviidae). XXV Congresso Brasileiro de Parasitologia, 2017. Citado na página 25.

HEROKU. **About Heroku**. 2018. Disponível em: <https://devcenter.heroku.com/articles/how-heroku-works>. Acesso em: 10 junho 2018. Citado na página 30.

JBRJ, C. **Catálogo Taxonômico da Fauna do Brasil (CTFB)**. 2016. Disponível em: <http://fauna.jbrj.gov.br/fauna/listaBrasil/PrincipalUC/PrincipalUC.do?lingua=pt>. Acesso em: 03 abril 2018. Citado na página 13.

JUNIT. **About JUnit**. 2018. Disponível em: <https://junit.org/junit5/docs/current/user-guide/>. Acesso em: 10 junho 2018. Citado 2 vezes nas páginas 29 e 39.

KALERMO, J.; RISSANEN, J. **Agile software development in theory and practice.** 2002. Disponível em: <https://jyx.jyu.fi/handle/123456789/12345/>. Acesso em: 25 abril 2018. Citado na página 19.

KEMPER, A. A evolução biológica e as revistas de divulgação científica : potencialidades e limitações para o uso em sala de aula. Dissertação (Mestrado em Educação)-Universidade de Brasília, 2008. Citado na página 16.

KOHN, M. **Scrum**. 2018. Disponível em: <http://www.desenvolvimentoagil.com.br/scrum/>. Acesso em: 18 abril 2018. Citado na página 20.

LIM, S.-H. Experimental comparison of hybrid and native applications for mobile systems. International Journal of Multimedia and Ubiquitous Engineering, 2015. Citado na página 24.

MICROSOFT. **About TypeScript**. 2017. Disponível em: <https://www.typescriptlang.org/>. Acesso em: 02 junho 2018. Citado na página 29.

MMA, B. **Biodiversidade Brasileira**. 2018. Disponível em: <http://www.mma.gov.br/biodiversidade/biodiversidade-brasileira>. Acesso em: 03 abril 2018. Citado na página 13.

MOTTA, P. C. **Aracnídeos do Cerrado**. [S.l.]: Technical Books Editora, 2010. Citado 4 vezes nas páginas 14, 17, 18 e 33.

NMBE. **The World Spider Catalog Association WSCA**. 2018. Disponível em: <https://wsc.nmbe.ch/wsca/>. Acesso em: 20 junho 2018. Citado na página 25.

OLIVEIRA, U. Diversidade e biogeografia de aranhas do brasil: Esforço amostral, riqueza potencial e Áreas de endemismo. Brasport Editora, 2011. Citado na página 14.

OLIVEIRA, U. de. Diversidade e biogeografia de aranhas do brasil: Esforço amostral, riqueza potencial e áreas de endemismo. Programa de Pós-Graduação em Ecologia, Conservação e Manejo da Vida Silvestre, 2011. Citado na página 19.

PAGOTTO, T.; FABRI, J. A.; LERARIO, A.; GONCALVES, J. A. **Desenvolvimento Ágil - Scrum**. 2016. Disponível em: <https://scrumsolo.wordpress.com/>. Acesso em: 23 abril 2018. Citado 2 vezes nas páginas 21 e 22.

_____. Scrum solo. IEEE, 2016. Citado na página 21.

PECHULA, M. R. **A ciência nos meios de comunicação de massa: divulgação de conhecimento ou reforço do imaginário social?** [S.l.]: Ciência&Educação, 2007. Citado na página 16.

PHILLIPS, A. **5 things you need to know about Continuous Delivery software development.** 2013. Disponível em: <https://www.networkworld.com/article/2169405/tech-primers/5-things-you-need-to-know-about-continuous-delivery-software-development.html/>. Acesso em: 12 junho 2018. Citado na página 23.

ROCHA, D. de A.; ALMEIDA, M. R. de; BATISTA, J. A. de S.; ANDRADE, A. J. de. Lutzodex – android mobile application for identification of brazilian sand flies (diptera: phlebotominae). XXV Congresso Brasileiro de Parasitologia, 2015. Citado na página 25.

RODRIGUES, L.; PRADO, A. F. de. Desenvolvimento de aplicações móveis com serviços restful e html5. Revista TIS, 2014. Citado na página 24.

SABBAGH, R. **Scrum - Gestão Ágil para Projetos de Sucesso**. [S.l.]: Casa do Código, 2014. Citado na página 20.

SCHWABER, K.; SUTHERLAND, J. Der scrum guide. 2014. Citado na página 19.

SERRANO, N.; HERNANTES, J.; GALLARDO, G. Mobile web apps. IEEE Software, 2013. Citado na página 24.

SILVA, S. T. da; TIBURCIO, I. C. S.; CORREIA, G. Q. C.; AQUINO, R. C. T. de. **Escorpiões, Aranhas e Serpentes: aspectos gerais e espécies de interesse médico no Estado de Alagoas**. [S.l.]: EDUFAL, 2005. Citado na página 17.

SOARES, M. D. S. Comparação entre metodologias Ágeis e tradicionais para o desenvolvimento de software. INFOCOMP Journal of Computer Science,, 2004. Citado na página 26.

SOARES, M. dos S. Metodologias Ágeis extreme programming e scrum para o desenvolvimento de software. Revista Eletrônica de Sistemas de Informação, 2004. Citado na página 19.

SOUZA, A. M. B.; SILVA, I. M. B.; SANTOS, Y. G. Estudo etnológico: Importância médica dos aracnídeos (arachnida: araneae, scorpiones) e sua relação com a comunidade de caetité - ba. X Congresso de Ecologia do Brasil, 2011. Citado na página 17.

SPRING. **About Spring**. 2018. Disponível em: <https://spring.io/>. Acesso em: 02 junho 2018. Citado na página 29.

STARK, J. **Building iPhone Apps with HTML, CSS, and JavaScript**. [S.l.]: O'Reilly Media, 2010. Citado na página 25.

TELES V. M.; MELLO, L. **Desenvolvimento Ágil - Scrum**. 2018. Disponível em: <http://www.desenvolvimentoagil.com.br/scrum/>. Acesso em: 18 abril 2018. Citado na página 20.

TORRESI, S. I. C. de; PARDINI, V. L.; FERREIRA, V. F. Sociedade, divulgação científica e jornalismo científico. Quim. Nova, Vol. 35, No.3, 447, 2012. Citado na página 16.

TRAVISCI. **About TravisCI**. 2018. Disponível em: <https://travis-ci.org/>. Acesso em: 10 junho 2018. Citado na página 30.

WHITTAKER, R. et al. **Conservation biogeography: assessment and prospect. Diversity and Distributions**. [S.l.: s.n.], 2005. Citado na página 14.

A Backlog do Produto

O Backlog do Produto é composto pelas Histórias de Usuário e Histórias técnicas elicitadas e desenvolvidas durante as Sprints. Estas representam necessidades levantadas reais para a aplicação, levantadas pelo PO e desenvolvedor.

A.1 Histórias de Usuário:

Na Tabela 2 são apresentadas as Histórias de Usuário elicitadas durante as Sprints. As Histórias de Usuário representam as necessidades levantadas pelo usuário, que culminam em funcionalidades da aplicação

Tabela 2 – Histórias de Usuário

História de Usuário	**Implementada**
Eu, como usuário, gostaria visualizar famílias de aracnídeos listadas em ordem alfabética com imagens, para que possa identifica-las facilmente.	Sim
Eu, como usuário, gostaria de consultar famílias de espécies pelo nome, para encontrar uma família específica mais facilmente.	Sim
Eu, como usuário, gostaria de realizar a identificação de um espécime por características visuais, a fim de obter informações sobre espécies as quais não sei o nome.	Sim
Eu, como usuário, gostaria de visualizar os dados e informações de cada família de aracnídeos, a fim de me informar sobre.	Sim
Eu, como usuário, gostaria de ser orientado sobre a utilização do aplicativo, a fim de mais facilmente utilizá-lo.	Sim
Eu, como usuário, gostaria de consultar famílias de espécies pelo seu nome popular, para encontrar uma família específica mais facilmente.	Sim

Dentre as histórias elicitadas, todas foram concluídas no decorrer das Sprints.

A.2 Histórias Técnicas:

Na Tabela 3 são apresentadas as Histórias Técnicas elicitadas durante as Sprints. Estas representam as necessidades técnicas do projeto, levantadas principalmente pelo desenvolvedor, em acordo com o PO.

Dentre as histórias elicitadas, todas foram concluídas no decorrer das Sprints.

Tabela 3 – Histórias Técnicas

História Técnica	**Implementada**
Eu, como desenvolvedor, gostaria de a configuração do ambiente de desenvolvimento da aplicação mobile e API Java, a fim de iniciar a implementação do sistema.	Sim
Eu, como desenvolvedor, gostaria de realizar o versionamento do código, a fim de manter maior controle das versões da aplicação.	Sim
Eu, como desenvolvedor, gostaria de separar a lógica da aplicação em uma API, a fim de desacoplar o backend do frontend.	Sim
Eu, como desenvolvedor, gostaria de configurar a integração contínua da API, a fim de validar as builds geradas.	Sim
Eu, como desenvolvedor, gostaria de configurar o deploy automático da API, a fim de facilitar a sua publicação.	Sim
Eu, como desenvolvedor, gostaria de configurar a coleta dinâmica de métricas da API, a fim de manter a qualidade do código	Sim
Eu, como desenvolvedor, gostaria de configurar a coleta dinâmica de métricas do aplicativo, a fim de manter a qualidade do código	Sim

B Diagrama de Classes da API

Na Figura 21 é possível visualizar o diagrama de classes resultante da API Java.

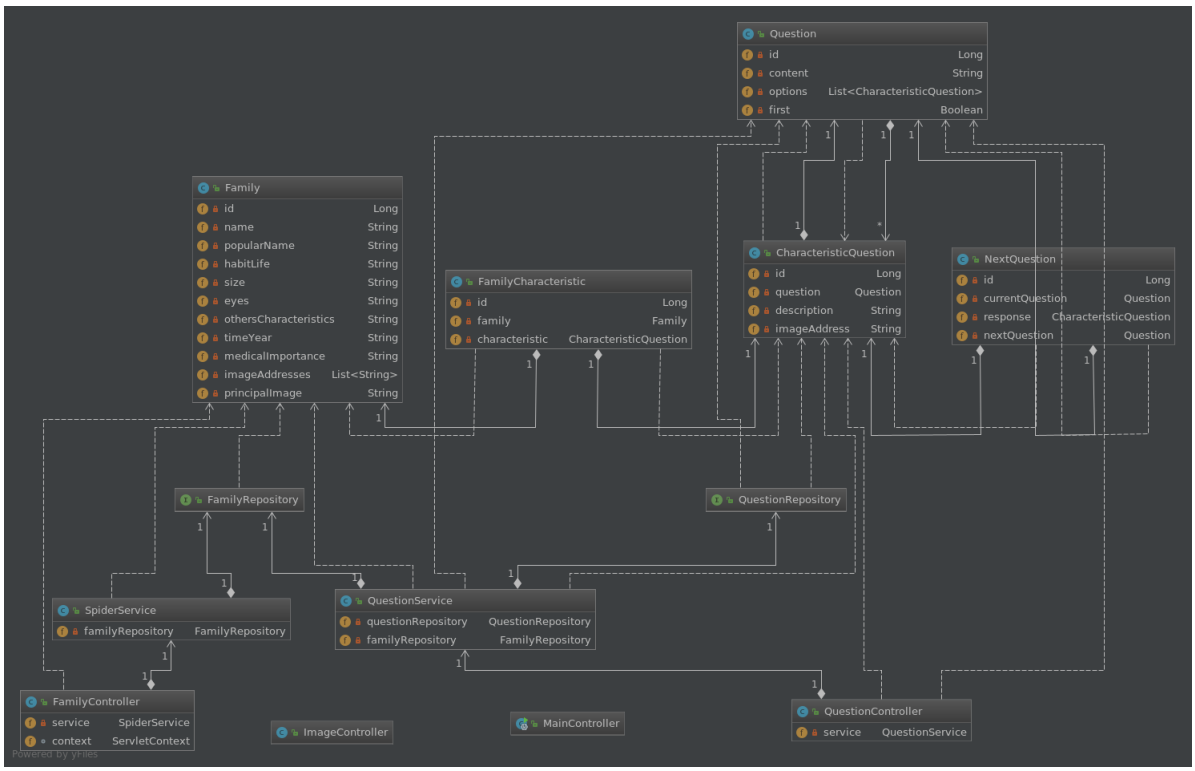

Figura 21 – Diagrama de Classes.

Fonte: IntelliJ IDEA

C Telas

C.1 Consulta por nome

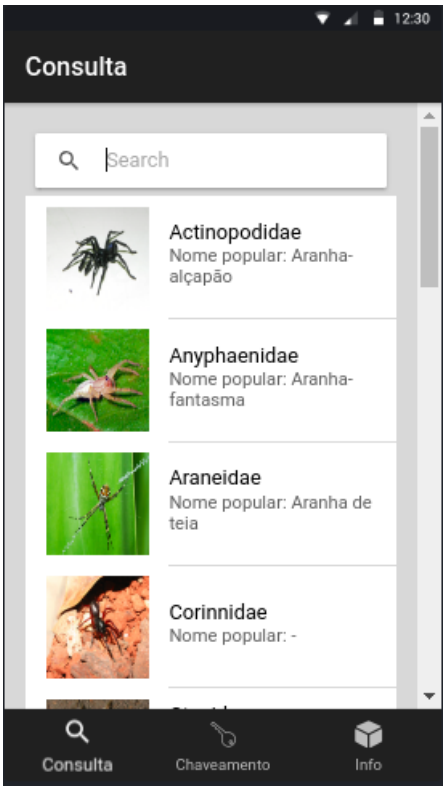

Figura 22 – Tela de Listagem.

Fonte: Autor

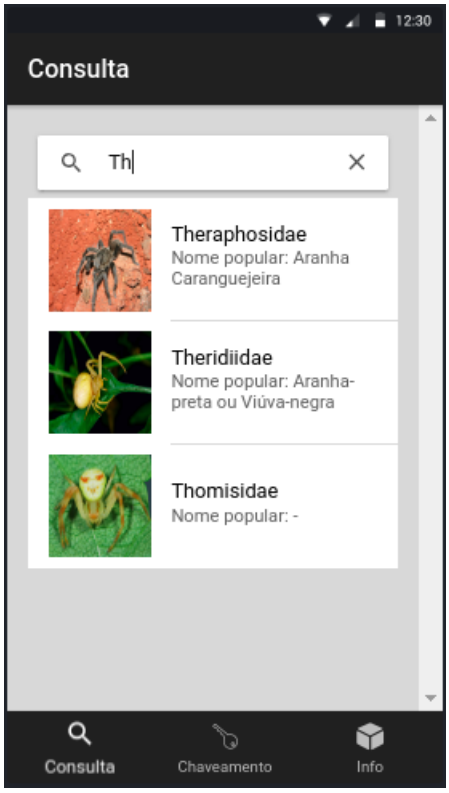

Figura 23 – Telas da Consulta por Nome.

Fonte: Autor

C.2 Telas da Consulta Chaveada

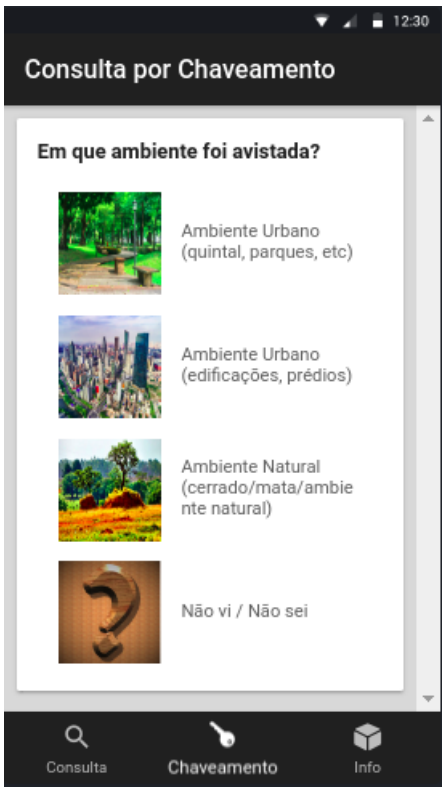

Figura 24 – Tela de Consulta por Chaveamento.

Fonte: Autor

Figura 25 – Tela de Consulta por Chaveamento.

Fonte: Autor

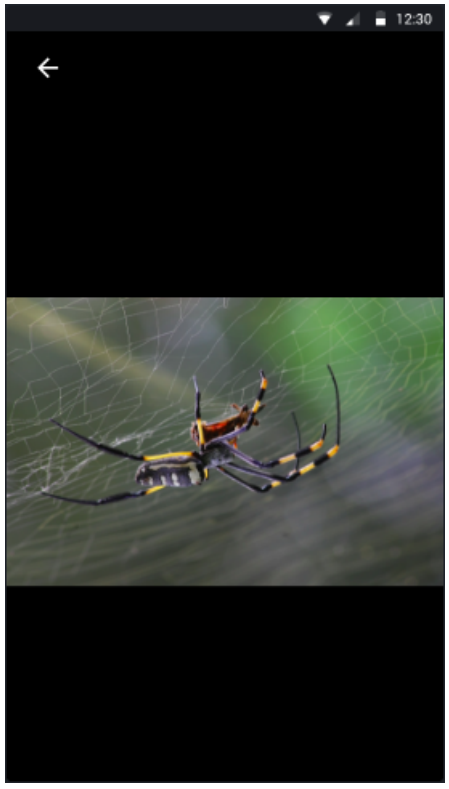

Figura 26 – Tela de Ampliação de Imagem de Característica.

Fonte: Autor

C.3 Tela de sugestões

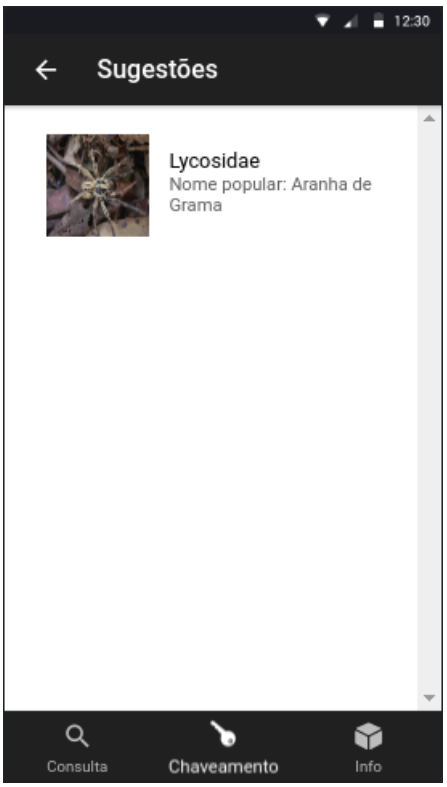

Figura 27 – Tela de Sugestões.

Fonte: Autor

C.4 Telas de Detalhamento

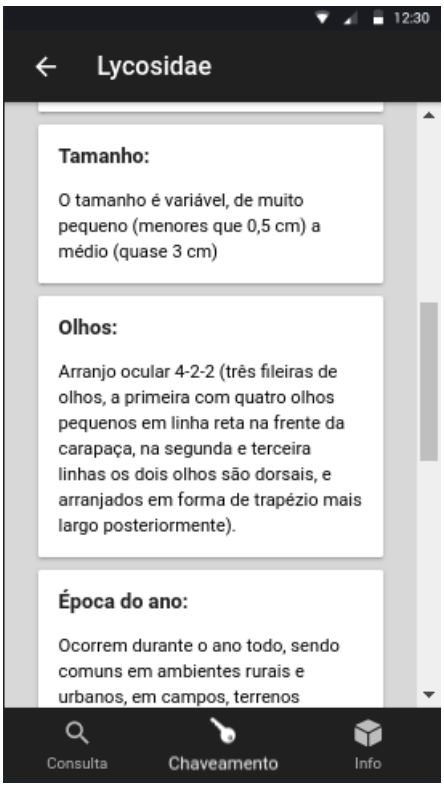

Figura 28 – Tela de Detalhamento.

Fonte: Autor

Figura 29 – Tela de Detalhamento.

Fonte: Autor

C.5 Telas de Informações. Fonte: Autor

Figura 30 – Tela de Informações.

Fonte: Autor

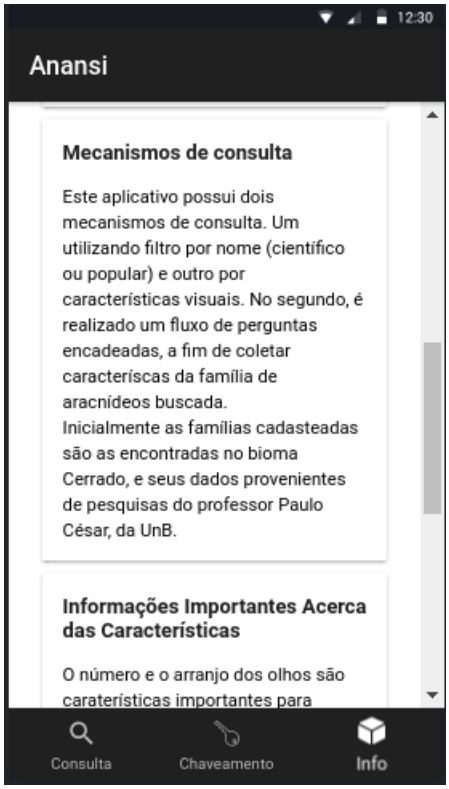

Figura 31 – Tela de Informações.

Fonte: Autor

D Cronograma

Para guiar o desenvolvimento do projeto, foi elaborado inicialmente um cronograma de planejamento do trabalho escrito, para a primeira entrega, e um cronograma referente ao desenvolvimento da aplicação, na segunda entrega. O Cronograma elaborado é apresentado a seguir:

	Nome	Duração	Início	Fim
1	**01/2016 - TCC 1**	89dias?	**06/03/2018**	**06/07/2018**
2	Definição do Escopo do Projeto	23dias?	06/03/2018	05/04/2018
3	Elaboração do capítulo de Introdução	12dias?	06/04/2018	23/04/2018
4	Pesquisa e desenvolvimento do Referencial Teórico	25dias?	08/05/2018	11/06/2018
5	**Elaboração da Metodologia**	25dias?	**14/05/2018**	**15/06/2018**
6	Definição da metodologia de desenvolvimento	6dias?	14/05/2018	21/05/2018
7	Definição da arquitetura da aplicação	10dias?	14/05/2018	25/05/2018
8	Definição das tecnologias de desenvolvimento	14dias?	22/05/2018	08/06/2018
9	Elaboração do processo de desenvolvimento	5dias?	11/06/2018	15/06/2018
10	Formatação e revisão do trabalho	7dias?	18/06/2018	26/06/2018
11	Mapeamento e definição de riscos	4dias?	27/06/2018	02/07/2018
12	Revisão final da primeira entrega	4dias?	03/07/2018	06/07/2018
13	**02/2016 - TCC 2**	70dias?	**20/08/2018**	**23/11/2018**
14	**Desenvolvimento da aplicação**	60dias?	**20/08/2018**	**09/11/2018**
15	Sprint 1	10dias?	20/08/2018	31/08/2018
16	Sprint 2	10dias?	03/09/2018	14/09/2018
17	Sprint 3	10dias?	17/09/2018	28/09/2018
18	Sprint 4	10dias?	01/10/2018	12/10/2018
19	Sprint 5	10dias?	15/10/2018	26/10/2018
20	Sprint 6	10dias?	29/10/2018	09/11/2018
21	Atualização da doumentação	10dias?	12/11/2018	23/11/2018
22	Revisão e Entrega final do projeto	12dias?	23/10/2018	07/11/2018

Figura 32 – Cronograma de Execução.

Fonte: Autor

Entretanto, devido a curva de aprendizagem de tecnologias e dificuldades encontradas referentes a agenda, impactando muitas vezes na cumprimento adequado das etapas definidas do processo, ocorreram alterações no cronograma da segunda entrega, de modo a possibilitar o andamento das atividades planejadas. O cronograma final da segunda entrega é apresentado na Figura x (segunda entrega).

02/2016 - TCC 2	**69dias?**	**10/09/2018**	**13/12/2018**
Desenvolvimento da aplicação	**65dias?**	**10/09/2018**	**07/12/2018**
Sprint 1	**10dias?**	**10/09/2018**	**21/09/2018**
Eu, como desenvolvedor, gostaria de a configuração do ambiente de desenvolvimento da aplicação mobile e API Java, a fim de iniciar a implementação do si...	5dias?	10/09/2018	14/09/2018
Eu, como desenvolvedor, gostaria de realizar o versionamento do código, a fim de manter maior controle das versões da aplicação.	5dias?	17/09/2018	21/09/2018
Sprint 2	**10dias?**	**24/09/2018**	**05/10/2018**
Eu, como desenvolvedor, gostaria de separar a lógica da aplicação em uma API, a fim de desacoplar o backend do frontend	8dias?	24/09/2018	03/10/2018
Eu, como usuário, gostaria visualizar famílias de aracnídeos listadas em ordem alfabética com imagens, para que possa identifica-las facilmente	3dias?	03/10/2018	05/10/2018
Sprint 3	**10dias?**	**08/10/2018**	**19/10/2018**
Eu, como usuário, gostaria de realizar a identificação de um espécime por características visuais, a fim de obter informações sobre espécies as quais não se...	8dias?	08/10/2018	17/10/2018
Eu, como usuário, gostaria de consultar famílias de espécies pelo nome, para encontrar uma família específica mais facilmente	3dias?	17/10/2018	19/10/2018
Sprint 4	**10dias?**	**22/10/2018**	**02/11/2018**
Eu, como desenvolvedor, gostaria de configurar a coleta dinâmica de métricas da API, a fim de manter a qualidade do código	4dias?	22/10/2018	25/10/2018
Eu, como desenvolvedor, gostaria de configurar a coleta dinâmica de métricas do aplicativo, a fim de manter a qualidade do código	3dias?	24/10/2018	26/10/2018
Eu, como usuário, gostaria de realizar a identificação de um espécime por características visuais, a fim de obter informações sobre espécies as quais não se...	3dias?	26/10/2018	30/10/2018
Eu, como usuário, gostaria de visualizar os dados e informações de cada família de aracnídeos, a fim de me informar sobre	5dias?	29/10/2018	02/11/2018
Sprint 5	**10dias?**	**05/11/2018**	**16/11/2018**
Eu, como usuário, gostaria de realizar a identificação de um espécime por características visuais, a fim de obter informações sobre espécies as quais não se...	10dias?	05/11/2018	16/11/2018
Eu, como usuário, gostaria de visualizar os dados e informações de cada família de aracnídeos, a fim de me informar sobre	10dias?	05/11/2018	16/11/2018
Eu, como usuário, gostaria de ser orientado sobre autilização do aplicativo, a fim de mais facilmente utilizá-lo.	5dias?	12/11/2018	16/11/2018
Eu, como desenvolvedor, gostaria de configurar o deploy automático da API, a fim de facilitar a sua publicação.	4dias?	12/11/2018	15/11/2018
Sprint 6	**15dias?**	**19/11/2018**	**07/12/2018**
Eu, como usuário, gostaria de realizar a identificação de um espécime por características visuais, a fim de obter informações sobre espécies as quais não se...	15dias?	19/11/2018	07/12/2018
Eu, como usuário, gostaria de consultar famílias de espécies pelo seu nome popular, para encontrar uma família específica mais facilmente.	6dias?	28/11/2018	05/12/2018
Eu, como desenvolvedor, gostaria de configurar o deploy automático da API, a fim de facilitar a sua publicação.	10dias?	26/11/2018	07/12/2018
Eu, como desenvolvedor, gostaria de configurar a integração contínua da API, a fim de validar as builds geradas.	8dias?	28/11/2018	07/12/2018
Eu, como usuário, gostaria de consultar famílias de espécies pelo seu nome popular, para encontrar uma família específica mais facilmente	5dias?	03/12/2018	07/12/2018
Atualização da doumentação	3dias?	07/12/2018	11/12/2018
Revisão e Entrega final do projeto	3dias?	11/12/2018	13/12/2018

Figura 33 – Cronograma da Segunda Entrega.

Fonte: Autor