

**CENTRO UNIVERSITÁRIO DE ANÁPOLIS – UniEVANGÉLICA
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO**

**DESENVOLVIMENTO DO APLICATIVO DONAR PARA AMPLIAR O VOLUME DE
DOAÇÕES DE SANGUE NO BRASIL**

**GABRIEL LEITE DIAS
MATHEUS LIMA DE ALBUQUERQUE**

**ANÁPOLIS
2018**

**GABRIEL LEITE DIAS
MATHEUS LIMA DE ALBUQUERQUE**

**DESENVOLVIMENTO DO APLICATIVO DONAR PARA AMPLIAR O VOLUME DE
DOAÇÕES DE SANGUE NO BRASIL**

Trabalho de Conclusão de Curso II apresentado como requisito parcial para a obtenção de grau do curso de Bacharelado em Engenharia de Computação do Centro Universitário de Anápolis – UniEVANGÉLICA.

Orientador(a): Prof. Me. Millys Fabrielle Araujo Carvalhaes.

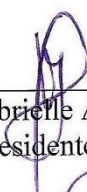
Anápolis
2018

GABRIEL LEITE DIAS
MATHEUS LIMA DE ALBUQUERQUE

**DESENVOLVIMENTO DO APLICATIVO DONAR PARA AMPLIAR O VOLUME DE
DOAÇÕES DE SANGUE NO BRASIL**

Trabalho de Conclusão de Curso II apresentado como requisito parcial para a obtenção de grau do curso de Bacharelado em Engenharia de Computação do Centro Universitário de Anápolis – UniEVANGÉLICA.

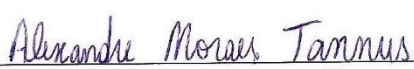
Aprovado(a) pela banca examinadora em 11 de dezembro de 2018, composta por:



Millys Fabrielle Araújo Carvalhaes
Presidente da Banca



Luciana Nishi
Prof(a). Convidado(a)



Alexandre Moraes Tannus
Prof(a). Convidado(a)

Agradecimentos

Primeiramente gostaríamos de agradecer a Deus pela sabedoria durante todo o decorrer do curso, ao nosso orientador, o professor Me. Millys Fabrielle Araujo Carvalhaes, que esteve presente em todas as etapas do desenvolvimento deste projeto, assim como aos demais professores que desempenharam um excelente trabalho durante o decorrer de todo a graduação.

As nossas respectivas famílias, que estiveram presentes, em todos os momentos difíceis, nos apoiando e dando forças com palavras de apoio e superação durante o desenrolar dos últimos cinco anos.

Gostaríamos de agradecer, ainda, aos nossos colegas de sala, que apoiaram a realização deste trabalho, em especial: Érico Junior de Moraes, Josimar Candido Narciso, Lucas Andrade da Silva, Samuel de Souza Silva e Vitor Augusto, que participaram ativamente no desenvolvimento deste projeto, sugerindo ideias, melhorias e novas perspectivas.

Ressaltando, ainda, o imenso prazer em passar esses cinco anos na companhia dos nossos colegas. Foram cinco anos de aprendizagem, alegria e obviamente alguns momentos difíceis nessa trajetória, mas esses nos tornaram alunos melhores e certamente um profissional mais preparado para o mercado de trabalho.

“Sic Parvis Magna”.¹

(Sir Francis Drake)

¹ Grandeza, de pequenos começos.

RESUMO

Este relato de experiência objetiva descrever as etapas de construção de um aplicativo *mobile*, assim como um módulo de gerenciamento *web*, visando o aumento do índice de doações de sangue no Brasil. O foco dessa aplicação é conectar um doador a um local de doação como: hemocentros, bancos de sangue e locais afins, levando em consideração a distância entre doador e o local de doação. A conexão será feita por meio de campanhas que serão divulgadas pelas redes sociais por intermédio dos próprios usuários da aplicação e de informativos que serão disponibilizados aos doadores buscando a conscientização e disseminação dos direitos atribuídos aos mesmos. O intuito é fazer com que esses usuários tenham conhecimento dos direitos assegurados a eles por lei, quando se disponibilizam a realizar o procedimento de doação, contribuindo para a fidelização dos mesmos e permitindo um melhor controle de qualidade do material recolhido. Espera-se, com o uso do aplicativo pela comunidade em geral, a maximização do aumento no número de doações de sangue para os níveis recomendados pela Organização Mundial da Saúde (OMS), além de conscientizar as pessoas sobre a importância desse ato. E, assim, efetivar o uso da tecnologia como uma ferramenta social de fundamental importância na tentativa de salvar vidas.

Palavras-chave: Aplicativo Mobile. Doações de Sangue. Doadores de Sangue. Redes Sociais. Veículo Social.

ABSTRACT

This objective experience report describes the steps of building a mobile application, as well as a web management module, aiming to increase the blood donation index in Brazil. The focus of this application is to connect a donor to a donation site, such as: blood centers, blood banks and related sites, taking into account the distance between donor and donation local. The connection will be made through campaigns that will be disseminated through social networks through the users of the application itself and information that will be made available to donors seeking awareness and dissemination of the rights assigned to them. The intention is to make these users aware of the rights granted to them by law, when they are available to carry out the donation procedure, contributing to their loyalty and allowing a better quality control of the collected material. The use of the application by the community in general is expected to maximize the increase in the number of blood donations to levels recommended by the World Health Organization (WHO), as well as make people aware of the importance of this act. In addition, this, make the use of technology as a social tool of fundamental importance in the attempt to save lives.

Keywords: Blood donations. Blood donors. Mobile Application. Social networks. Social Vehicle.

LISTA DE ILUSTRAÇÕES

Figura 1 - Passos do TDD	22
Figura 2 - Processo de Construção da Documentação	34
Figura 3- Priorização das Features	35
Figura 4 - Protótipo da Tela Inicial	36
Figura 5 - Protótipo da Tela de Perfil	36
Figura 6 - Protótipo da Tela de Campanhas	36
Figura 7 - Protótipo da Tela de Informações	36
Figura 8 - Protótipo da Tela Locais para Doação	37
Figura 9 - Protótipo do Dashboard de Gerenciamento	37
Figura 10 - Protótipo da Tela de Campanhas Ativas	38
Figura 11 - Protótipo da Tela de Registro de Doação	38
Figura 12 - Teste Automatizado de Inserção de Campanhas	39
Figura 13 - Pipelines de Integração Contínua	40
Figura 14 - Estruturação do Projeto Ionic	41
Figura 15 - Estruturação do Projeto Web	41
Figura 16 - Model das Campanhas	42
Figura 17 – View Tela de Login.....	42
Figura 18 - Controller da Tela de Login.....	43
Figura 19 - Component para Listagem de Hemocentros	44
Figura 20 - Função para injeção do Component	44
Figura 21 - Provider Hemocentros	45

LISTA DE ABREVIATURAS E SIGLAS

ANATEL	Agência Nacional de Telecomunicações
API	<i>Application Programming Interface</i>
BaaS	<i>Backend as a service</i>
CASE	<i>Computer-Aided Software Engineering</i>
CI	<i>Continuous Integration</i>
CSS	<i>Cascading Style Sheets</i>
DI	<i>Dependency Injection</i>
DB	<i>Database</i>
GPU	<i>Graphics Processing Unit</i>
HEMOPA	Hemocentro do Estado do Pará.
HTML	<i>HyperText Markup Language</i>
IOHA	Instituto Onco-Hematológico de Anápolis
JSON	<i>JavaScript Object Notation</i>
OS	Ordem de Serviço
KG	Quilograma
ML	Mililitro
MVC	<i>Model-View-Controller</i>
MVP	<i>Minimum Viable Product</i>
NPM	<i>Node Package Manager</i>
OMS	Organização Mundial da Saúde
OPAS	Organização Pan-Americana da Saúde
PHP	<i>Personal Home Page</i>
RAD	<i>Rapid Application Development</i>
SGBD	Sistema de Gerenciamento de Banco de Dados
SQL	<i>Structured Query Language</i>
TDD	<i>Test-Driven Development</i>
URL	<i>Uniform Resource Locator</i>
WWW	<i>World Wide Web</i>
XML	<i>Extensible Markup Language</i>
XP	<i>eXtreme Programming</i>
ZF2	<i>Zend Framework 2</i>

SUMÁRIO

1	INTRODUÇÃO	12
2	FUNDAMENTAÇÃO TEÓRICA	15
2.1	Doação de sangue.....	15
2.1.1	<i>Leis de incentivo a doação</i>	16
2.2	Impacto das mídias e redes sociais.....	17
2.2.1	<i>Inovação tecnológica</i>	18
2.3	Desenvolvimento de software	19
2.3.1	<i>Desenvolvimento ágil</i>	20
2.3.1.1	Manifesto ágil.....	20
2.3.1.2	Gerenciamento com SCRUM.....	21
2.3.1.3	Desenvolvimento Orientado a Teste	22
2.3.2	<i>Desenvolvimento web</i>	24
2.3.2.1	PHP.....	24
2.3.2.2	Web responsivo	25
2.3.2.3	Frameworks de desenvolvimento web	25
2.3.2.3.1	Laravel.....	27
2.3.2.3.1.1	XAMPP	28
2.3.3	<i>Desenvolvimento mobile</i>	28
2.3.3.1	Frameworks de desenvolvimento Mobile	29
2.3.3.1.1	Ionic.....	30
2.3.4	<i>Banco de dados</i>	31
2.3.4.1	Firestore	32
3	DESENVOLVIMENTO	33
3.1	Elicitação de requisitos.....	33
3.2	Delimitação de escopo	33
3.3	Definição da arquitetura e criação de protótipos.....	34
3.4	Desenvolvimento e validação do MVP	38
3.5	Definição e implementação de testes	39
3.6	Codificação da solução.....	40
3.6.1	<i>Model</i>	41
3.6.2	<i>View</i>	42
3.6.3	<i>Controller</i>	42
3.6.4	<i>Component</i>	43
3.6.5	<i>Provider</i>	45
3.7	Gerenciamento dos dados.....	45

3.8 Integração entre módulos	46
4 CONSIDERAÇÕES FINAIS	47
REFERÊNCIAS BIBLIOGRÁFICAS	48

1 INTRODUÇÃO

A tecnologia apresenta constante crescimento e evolução nas mais diversas áreas da sociedade, sendo que, na área da saúde não é diferente. A tecnologia, facilita a obtenção de informações sobre os pacientes, assim como, a prevenção de doenças e a difusão de melhores hábitos (PIZZO, 2015).

Uma possível área para aplicação tecnológica é no processo de doação de sangue. De acordo com Brasil (2015), apenas 1,8% da população brasileira doa sangue, números esses aquém dos recomendados que seriam de 3 a 5 por cento da população (Organização Mundial da Saúde - OMS, 2012; PEREIRA *et al.*, 2016).

A doação de sangue é algo que requer atenção da população. Segundo a Organização Pan-Americana de Saúde (OPAS, 2014), a maior dificuldade em aumentar o volume de doações reside na falta de informações sobre o processo e também na falta de conscientização sobre sua importância. Segundo a OMS (2013, apud SILVA; KUPEK; PERES, 2013, p. n.p.)

O uso de sangue e hemoderivados desempenha importante papel na prática médica contemporânea. Países e regiões onde são realizados maior número de procedimentos médicos avançados, como cirurgia cardíaca, transplantes e quimioterapia, apresentam maior demanda por sangue.

Uma única doação permite que até quatro vidas possam ser salvas (BRASIL, 2017). Todavia, conforme já frisado, o volume de doações no Brasil é bem inferior aos índices recomendados, sendo que uma porcentagem de cerca de 40% das doações são realizadas para fins de reposição, quando algum familiar ou pessoa do vínculo social sofrem um acidente ou algum problema de saúde (BRASIL, 2015). Nesse sentido, o Brasil, apesar de coletar o maior volume em termos absolutos de sangue na América Latina, doa proporcionalmente menos quando comparado aos outros países da região.

A OMS (2012) alerta que as doações voluntárias e não remuneradas precisam aumentar rapidamente em mais da metade do países para garantir um suprimento confiável de sangue para os pacientes, visto que, doações voluntárias regulares promovem melhor controle sobre o sangue coletado.

Com base nesses dados, percebe-se a existência de uma lacuna entre as organizações que recebem as doações e os doadores de sangue, esse problema norteou o desenvolvimento deste trabalho com o seguinte questionamento: como a tecnologia pode ser aplicada no desenvolvimento de uma solução com o intuito de mudar o quadro de doações de sangue, enquanto, ferramenta de conscientização para a população.

Levando em consideração o cenário apresentado, o desenvolvimento de uma solução *mobile*, com o objetivo de auxiliar no aumento do número de doações e esclarecer concepções equivocadas que a população possui sobre o processo de doação, permitirá alcançar uma grande parcela da população, pois, em contraponto ao déficit de doações, a quantidade de aparelhos celulares em funcionamento no Brasil em outubro de 2018 era de cerca de 233,35 milhões, média de 1,12 aparelhos para cada brasileiro a época. (BRASIL, 2018a).

Diante desses números, integrar uma aplicação *mobile* às principais redes sociais, proverá uma ferramenta eficiente de divulgação de campanhas e fidelização de doadores. As redes sociais exercem grande influência na vida e também na opinião das pessoas, pois,

As redes sociais existem em todos os lugares e podem ser formadas por pessoas ou organizações que partilham valores e objetivos comuns. Não são limitadas a uma estrutura hierárquica ou meio e podem estar na escola, no trabalho, na música, na política e até mesmo na família (CIRIBELI e PAIVA, 2011).

O Brasil é o país que mais utiliza as redes sociais na América Latina, tendo cerca de 62% da população como usuários ativos, com tempo médio de utilização de 3h39m (KEMP, 2018). Fundamentando-se nessas informações, as redes sociais se mostram de fundamental importância no processo de fidelização de doadores, sendo ferramentas ideais para vinculação de campanhas, assim como, canal de conscientização dos doadores no que diz respeito ao processo de doação e às garantias ofertadas aos que se dispõem a realizar esse procedimento solidário.

Dessa forma, este trabalho busca o desenvolvimento de uma aplicação, que permita aos bancos de sangue a criação, gerenciamento e divulgação de campanhas de doação de sangue, assim como, a disseminação de informações quanto o processo de doação e os benefícios disponibilizados aos doadores.

Para tanto, o objetivo geral deste projeto é o desenvolvimento de um aplicativo *mobile* e um módulo de gerenciamento *web*, para disseminação de informações e campanhas, na busca pela melhoria do quadro de doações de sangue no Brasil.

Almejando a realização do objetivo geral, são dispostos como objetivos específicos desse processo de desenvolvimento a realização de pesquisas bibliográficas e em loco, para obtenção de dados quantitativos para o projeto; a avaliação dos resultados obtidos, definindo etapas de desenvolvimento e homologação da solução e pôr fim a descrição das etapas do processo de obtenção de conhecimento e construção da solução.

Para alcançar o proposto foram desenvolvidas as seguintes atividades: a) Estudo e análise referente ao quadro de doações de sangue no Brasil; b) Estudo e definição de requisitos para o sistema; c) Documentação e desenvolvimento de artefatos pertinentes ao sistema; d) Validação da proposta junto ao hemocentro de Anápolis, por intermédio, do produto mínimo viável (MVP); e) Obtenção de ideias para melhoria e aperfeiçoamento do sistema); f) Definição e implementação dos testes, em conformidade com a metodologia de desenvolvimento orientada a testes (TDD); g) Desenvolvimento do módulo *mobile* focado no atendimento dos doadores e do módulo web voltado para o gerenciamento dos locais de coleta de sangue; h) Integração entre os módulos.

Sendo a metodologia desta pesquisa, quanto aos objetivos, restringida a um estudo bibliográfico e experimental, bibliográfico ao procurar estabelecer um paralelo entre a situação em que o país se encontra e as possíveis áreas de melhoria no que concerne a doação de sangue ao realizar pesquisas nas principais organizações que lidam com saúde e experimental ao implementar uma solução tecnológica com base nos resultados obtidos na etapa bibliográfica (WAZLAWICK, 2009).

O resultado deste processo é exposto a seguir, estando estruturado em: Fundamentação Teórica; Desenvolvimento e Considerações Finais.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda os conceitos que envolvem o desenvolvimento do trabalho de conclusão de curso, fazendo uma revisão bibliográfica sobre o tema para a aquisição do conhecimento necessário para o desenvolvimento da aplicação proposta.

2.1 Doação de sangue

A medicina não para de avançar. Com o passar dos anos são descobertas novidades para o tratamento de doenças, e até mesmo a cura delas. Mas mesmo com este avanço, nunca se descobriu uma forma de substituir o sangue humano. Quem, por algum motivo, precisa de uma transfusão, está completamente dependente de doações (BRASIL, 2018b).

A doação de sangue é um procedimento no qual o doador, por livre e espontânea vontade, tem seu sangue coletado, seja, por banco de sangue ou hemocentro, para uso futuro em transfusões de sangue (OPAS, 2014). Essa atividade é de fundamental importância para o funcionamento de hospitais, centros de saúde ou estabelecimentos afins.

O Brasil, apesar de coletar o maior volume em termos absolutos de sangue na América Latina, doa proporcionalmente menos do que outros países da região (OPAS, 2014). De acordo com o Instituto de Biociências da Universidade de São Paulo (2005, p.69), o Brasil necessita diariamente de 5.500 bolsas de sangue e, para suprir essa demanda, precisa de um número muito maior de doadores.

Antes de realizar a doação de sangue, o candidato a doador deve ter consciência que o receptor de seu sangue, ou derivados do mesmo, será uma pessoa fragilizada por alguma enfermidade ou doença. Portanto, o potencial doador deverá estar em boas condições físicas e desde a véspera, ou no mínimo quatro horas antes da doação, não poderá consumir alimentos muito gordurosos nem bebidas alcoólicas. Deverá ter se alimentando com comidas leves e saudáveis como, por exemplo, sucos, frutas, verduras e legumes, pois a ingestão de alimentos pesados pode interferir nos testes de triagem sanguínea e/ou prejudicar o receptor.

No processo de doação existem alguns requisitos que visam proteger a saúde do doador: ele deve ter entre 18 e 65 anos, ter o peso proporcional à altura e estar com aparência saudável. Além disso, o doador também passará por um teste de anemia e, caso se constate alguma doença, não poderá realizar a doação para não agravar sua condição de saúde (BRASIL, 2018c).

De acordo com o Banco de Sangue da Universidade Federal de Santa Catarina (2017), mesmo que não existam efeitos colaterais na doação de sangue, em alguns casos, o doador pode apresentar ansiedade, queda de pressão e náuseas, mas raramente algo mais grave, sendo que

os hemocentros possuem equipes especializadas para atender os doadores caso ocorram problemas durante o procedimento.

Durante a doação, a quantidade de sangue que pode ser coletada é de 9ml por quilo do doador, sendo que o mesmo deve ter no mínimo 50kg, podendo assim, fazer a doação padrão de 450ml de sangue sem causar prejuízo ao seu organismo. Após realizada a doação o organismo precisa de algum tempo para repor o sangue, sendo que o plasma é repostado em algumas horas, as plaquetas em alguns dias e as hemácias em alguns meses. Por esse motivo, a doação de sangue só deve ser realizada a cada 90 dias para os homens e a cada 120 dias para as mulheres (BRASIL, 2018c).

Além da pessoa que vai receber o sangue, o doador também melhora a sua saúde após a doação, De acordo com Melhor (2018), doar sangue pode melhorar a saúde cardiovascular, reduzir o risco de câncer, permitindo ainda, que o doador acompanhe regularmente sua saúde.

2.1.1 Leis de incentivo a doação

A constituição brasileira proíbe qualquer tipo de remuneração para a doação de sangue, conforme explicado na lei nº 10205, de 21 de março de 2001. É proibida remuneração ao doador pela doação de sangue (BRASIL, 2001). Entretanto, existem medidas que garantem benefícios aos doadores, sendo elas, definidas pelo governo de cada estado.

Caso o estado ou município possua algum incentivo, os estabelecimentos devem respeitar esse direito e garantir os mesmos, porém o doador deve ter em mãos a carteirinha de doador, documento que atesta a pessoa como um doador.

Alguns benefícios concedidos aos doadores incluem: meia entrada em eventos culturais, artísticos e esportivos, preferência em filas, inscrição gratuita em concursos públicos, entre outros (BRASIL, 2001). Está sendo avaliada a possibilidade de se ampliar os dias de folga aos quais os doadores dispõem, visando aumentar a curto prazo o percentual de doadores para no mínimo 2,3% (BRASIL, 2016).

Outros tipos de incentivo, do ponto de vista legal, devem ser estudados e analisados. Embora seja uma ação importante que ajuda muitas vidas, infelizmente, a maioria das pessoas tende a pensar nos benefícios, partindo dessa premissa ações que incentivam a fidelização de doadores são de fundamental importância, pois, contribuem para o aumento do volume de doações e permitem melhorar a qualidade do material coletado, conforme defende OPAS (2014) doações regulares permitem um melhor controle de qualidade em relação ao sangue coletado.

No estado do Pará, em 2016, foi realizada uma campanha para atrair mais doadores, a mesma contou com três dias de ações voltadas diretamente para os que se dispusessem a realizar

doações, dentre as atividades realizadas podem ser destacadas, festivais de música, oferta de serviços de beleza, oficina de customização de camisetas, distribuição de brindes, palestras, entre outras. Sendo que essas ações levaram a um aumento de adesão de 13% para 34%, um aumento significativo, quando pensamos, na quantidade de vidas salvas, graças aos frutos dessa ação.

Outro exemplo de ação, vem sendo realizada pelo Ministério da Saúde desde de 2013, a campanha “Torcedor Sangue Bom”, que reúne torcedores de diversos times do futebol brasileiro para a realização de doações sendo promovida por meio das redes sociais, sendo as doações direcionadas para os serviços de hemoterapia locais e para as respectivas agremiações esportivas da região de enfoque da campanha.

Ações sociais que promovam a doação de sangue existem e apresentam resultados satisfatórios, entretanto, precisam ser executadas com uma frequência maior e de maneira mais eficiente para que possam gerar melhorias significativas no quadro de doações.

2.2 Impacto das mídias e redes sociais

As redes sociais exercem grande influência na vida e na opinião das pessoas. Os conteúdos, compartilhados, podem impactar na maneira de se vestir, de como se portar, ditar novos valores para as pessoas e até mesmo moldar uma geração, principalmente, nas faixas etárias mais jovens, que representam grande parte dos usuários ativos nesse ambiente (CIRIBELI e PAIVA, 2011).

O índice de usuários ativos nas redes sociais é bem elevado. No primeiro trimestre de 2018, o *Instagram* possuía 800 milhões de usuários, o *Facebook* 2 bilhões e o *Whatsapp* 1,3 bilhões (KEMP, 2018, p.59 e *seg.*).

Entretanto, não é somente na vida dos jovens que as redes sociais geram impacto, no dia a dia das empresas, dos profissionais, das famílias e dos casais, as redes sociais são utilizadas não apenas para o compartilhamento de conteúdo, mas como veículo de interação e comunicação entre pessoas. O *Whatsapp* é um exemplo disso, já que esse aplicativo ultrapassou a barreira entre ser um mero aplicativo de mensagens e se tornou parte integrante da vida das pessoas (ACIOLI, 2016).

A transformação promovida pelas redes sociais é evidente, Kemp (2018), ressalta que um terço dos matrimônios norte-americanos já é proveniente das redes sociais, ou seja, elas podem ser utilizadas para diversas finalidades, em especial, para causas sociais.

As campanhas digitais no Brasil, segundo dados da Nielsen Media Research (NIELSEN, 2016) conseguem atingir, em média, 78% das impressões dentro do perfil demográfico previsto para a audiência. Existem centros de coleta de sangue em todos os estados do Brasil, entretanto,

poucos apresentam projetos no âmbito digital, como o realizado, por estudantes do primeiro semestre de medicina da Universidade de Santa Catarina no interior de São Paulo, que desenvolveu o aplicativo Time do Sangue que visa reunir uma equipe de doadores quando alguém necessitar de uma doação (PEREIRA; BASTOS, 2009).

Levando em consideração os dados estatísticos citados acima, percebemos que o alcance a usuários poderá ser feito de uma forma consistente e com boas chances de alcançar o objetivo proposto.

2.2.1 *Inovação tecnológica*

O processo de inovação tecnológica representa uma série de procedimentos necessários para que se implementem melhorias ou se desenvolva um novo processo produtivo, produto ou serviço.

De acordo com Venki (2017), existem oito passos necessários em uma inovação tecnológica, são eles:

- Pesquisa básica: ocorre normalmente apenas em grandes empresas normalmente dos setores farmacêuticos e tecnologia da informação, que tem por objetivo manter departamentos de pesquisa se atualizando no que mais pode impactar a suas operações;
- Pesquisa aplicada: especificar alguma necessidade do mercado que possa representar uma oportunidade de negócio;
- Desenvolvimento: momento em que será desenvolvido um protótipo da solução a ser entregue;
- Engenharia: definição dos materiais, formas adequadas, estocagem e transporte dos materiais;
- Fabricação: definição da melhor maneira de se entregar a solução para o cliente final seguindo os padrões necessários;
- Marketing: descobrir como o produto será aceito no mercado (testes de aceitação e mercado);
- Promoção: lançamento da solução;
- Melhoria contínua: Sempre deve-se encontrar formas de melhorar e aprimorar os produtos sempre com o objetivo de agregar mais valor.

Um exemplo interessante de inovação tecnológica e que serve de argumento para o alcance da solução Donar são os *smartphones*. O tempo para ter acesso às informações é muito rápido, principalmente com o uso dos *smartphones*. De acordo com uma pesquisa de Brasil

(2018a) a quantidade de aparelhos celulares em funcionamento em outubro de 2017 era de cerca 242,1 milhões, uma média de 1,17 aparelhos para cada brasileiro, assim sendo, uma aplicação *mobile* tem um mercado consolidado para se desenvolver.

Este trabalho, atentou-se a todas as fases, acima mencionadas, mas possui enfoque principalmente nas de pesquisa, desenvolvimento e engenharia, entretanto, a questão do *marketing* e da melhoria continua, já está em estudo.

2.3 Desenvolvimento de software

No que concerne ao desenvolvimento de software, percebe-se que ocorreu uma evolução em comparação com as décadas anteriores, novas tecnologias, novos métodos que proporcionam uma maneira melhor para a realização do mesmo, entretanto, os benefícios dessa evolução aumentam o índice de exigências que devem ser atendidas, pois, caso não se respeite as mesmas, os possíveis clientes serão perdidos para organizações capazes de suportar suas necessidades (BAYLÃO; OLIVEIRA, 2012).

A utilização de computadores nas diversas áreas do conhecimento humano tem caracterizado uma crescente demanda por soluções computadorizados. Tal processo é composto por fases, entre as quais, se destacam o levantamento e análise de requisitos, o projeto, a implementação e pôr fim a realização dos testes.

A engenharia de software surgiu para resolver os problemas da crise (MOTA; LIMA; ROMANO, 2011). Baseando-se em modelos industriais, foram criadas as metodologias de desenvolvimento que organizavam de uma maneira profissional o modo de desenvolver sistemas. Destacam-se como exemplos dessas metodologias: cascata, incremental, espiral, prototipação, *Rapid Application Development* (RAD) e a metodologia ágil, sendo essa última melhor tratada na seção 2.3.1.

As metodologias acima mencionadas, são divididas em etapas, acompanhadas de documentos para especificar ao cliente o que será desenvolvido, visando sempre à qualidade do produto final. Sendo que, a documentação em um projeto de software permite a criação de um entendimento único, no que diz respeito, ao que deve ser desenvolvido, evitando assim imprecisões, pois, segundo Sommerville (2011), a imprecisão na especificação de requisitos é o motivo de diversos problemas de engenharia de software.

O processo de desenvolvimento de software é bastante mutável, conforme afirmam Mainart e Santos (2010, p. n.p., *apud* SILVA; CARVALHO, 2014, p.131), como, por exemplo, o surgimento de novos requisitos, sejam funcionais ou não funcionais, por parte do cliente. É necessário alterar a documentação e o produto em si, frequentemente ocorrendo casos de

fracasso ao final do período de tempo estipulado nas fases iniciais do projeto e com isso gerando impacto nas etapas finais.

Cinco fatores críticos podem ser identificados em um projeto para determinar o quanto seu desenvolvimento deve ser ágil ou dirigido por plano (tradicional), conforme defendem Boehm e Turner (2002), sendo eles, tamanho do projeto, a criticidade, o dinamismo do ambiente de desenvolvimento com relação a alterações, os recursos humanos (pessoal) envolvidos e, por fim, os fatores culturais.

2.3.1 *Desenvolvimento ágil*

No final da década de 90, surgiram os métodos denominados ágeis, que após a publicação do manifesto ágil em 2001, obtiveram grande impulso. O desenvolvimento ágil de software é caracterizado pelo seu caráter adaptativo e orientado a diversos tipos de projetos. Segundo Oliveira, Rocha e Vasconcelos (2004)

Os Processos Ágeis de Desenvolvimento compartilham a premissa de que o cliente aprende sobre suas necessidades, na medida em que é capaz de manipular o sistema que está sendo produzido e, com base no feedback do sistema, ele reavalia as suas necessidades e prioridades, gerando mudanças que devem ser incorporadas ao software.

Essa técnica é recomendada quando o projeto é caracterizado por:

- Mudanças constantes;
- Requisitos passíveis de alterações;
- Equipe pequena;
- Recodificação não impacta em aumento de custo;
- Data de entrega acarreta em um alto custo;
- Desenvolvimento rápido é fundamental.

As metodologias ágeis propõem a obtenção de resultados práticos em tempo inferior às utilizadas pelos processos tradicionais utilizados pelo mercado de desenvolvimento de software, elas pretendem fazer isso tirando o foco do processo e colocando-o no produto. Dessa forma, os métodos ágeis se propõem a dispensar ou modificar as etapas e a forma como os envolvidos realizam suas tarefas (BASSI FILHO, 2008).

2.3.1.1 Manifesto ágil

Segundo Soares (2004), o manifesto ágil é uma declaração de princípios que mostram que os processos, ferramentas e o planejamento tem importância secundária quando comparados com os indivíduos e interações durante o processo de desenvolvimento.

Figueiredo (2016) sumariza alguns princípios:

- A maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado;
- *Stakeholders* e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto;
- O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face.

Tendo como base os conceitos e exemplificações sobre os princípios do desenvolvimento ágil, o projeto foi gerenciado por meio do *framework Scrum*, voltado para o trabalho com metodologias ágeis sendo o desenvolvimento orientado a testes pela metodologia ágil TDD que será explorada posteriormente.

2.3.1.2 Gerenciamento com SCRUM

Pode-se dizer que o *Scrum* é leve, fácil de entender, no entanto, muito difícil de dominar. O *Scrum* vem sendo usado desde o início dos anos 90 e é interessante destacar que ele não se classifica como processo ou uma técnica, em vez disso, é um *framework* onde você pode aplicar vários processos e técnicas (SUTHERLAND; SCHWABER, 2017).

Scrum é um *framework* para o desenvolvimento ágil de softwares, em que, diferentes times se relacionam com diferentes papéis, artefatos e regras, cada um desses componentes desempenha uma função específica e de grande importância para o sucesso do *framework* que propõe que as pessoas resolvam problemas complexos e adaptativos enquanto de uma forma produtiva e criativa entregam produtos com o mais alto valor possível. (SUTHERLAND; SCHWABER, 2017).

De acordo com Sutherland e Schwaber (2017), o *Scrum* é fundamentado em teorias empíricas de controle de processo, ou seja, o conhecimento vem da experiência referente a projetos anteriormente desenvolvidos. Existem três pilares que apoiam a implementação de controle de processos empíricos: transparência, inspeção e adaptação.

Os projetos no *Scrum* são divididos em ciclos, que representam períodos, comumente chamados de *sprints*, em que um conjunto de atividades devem ser realizadas. As funcionalidades a serem implementadas em um projeto são mantidas em uma lista conhecida como *Product Backlog* (SUTHERLAND; SCHWABER, 2017). Os itens dessa lista são priorizados pelo *Product Owner* e selecionados para execução pela equipe a cada início de *sprint* em reuniões de planejamento, chamadas de *Sprint Planning Meeting*.

Ao final de um *sprint*, a equipe apresenta as funcionalidades implementadas em uma *Sprint Review Meeting* (SUTHERLAND; SCHWABER, 2017). Então, antes de iniciar uma nova *sprint*, é realizada uma *Sprint Retrospective* para que a equipe possa identificar o que foi bom no projeto, o que pode ser melhorado e as dificuldades enfrentadas pelo time.

2.3.1.3 Desenvolvimento Orientado a Teste

O desenvolvimento orientado a testes, do inglês, *test driven development* (TDD) é definido por Rocha (2012), como uma técnica de desenvolvimento ágil de software que se baseia em um ciclo curto de repetições. A ideia por trás dessa metodologia é que primeiro se deve escrever os testes para posteriormente o código.

No TDD o desenvolvedor escreve um caso de teste automatizado que define uma melhoria desejada ou uma nova funcionalidade, implementa o código que será validado e realiza refatorações até que o mesmo, esteja em conformidade com os padrões aceitáveis (ROCHA, 2012).

A figura 1, ilustra os passos do desenvolvimento utilizando TDD:

Figura 1 - Passos do TDD



Fonte: *Objective Solutions*, 2016

O planejamento dos testes permite a identificação dos itens e funcionalidades que deverão ser testados, o TDD se inicia com a criação do teste, sendo esse, chamado de teste falho, pois a funcionalidade ainda não existe, desta forma, o teste deve retornar um erro, assim sendo, é realizada a implementação de uma solução que satisfaça ao teste, sendo essa construída da forma mais simples possível, visando simplesmente que a funcionalidade passe no teste,

atingindo esse patamar, o código é então refatorado fazendo uso de boas práticas de programação e engenharia buscando a criação de um código otimizado, esse processo é repetido até que todas as funcionalidades tenham sido atendidas.

Rocha (2012) destaca, que o TDD é utilizado em metodologias como o *Extreme Programming* (XP), sendo, uma das técnicas que auxiliam na melhoria de qualidade do processo de desenvolvimento, o tornando mais eficiente. O desenvolvimento orientado por testes proporciona uma visão mais ampla do que deve ser realizado pelo desenvolvedor, pois antes de criar a funcionalidade, deve-se criar um teste para a mesma.

Os testes são utilizados para facilitar no entendimento do projeto, clareando ideias em relação ao que se deseja do código (FREEMAN; PRYCE, 2012). A criação de testes unitários ou de componentes é parte crucial para o TDD.

No TDD os componentes são testados para garantir que operem corretamente. Cada componente é testado independentemente, de forma, a não interagir com os demais componentes do sistema. Os componentes podem ser entidades simples, tais como funções ou classes de objetos, ou podem ser grupos coerentes dessas entidades (PRESSMAN, 2011).

Freeman e Pryce (2012) ressaltam, que os testes devem seguir o modelo *F.I.R.S.T*

- *Fast* - Rápidos: testam apenas uma unidade;
- *Isolated* - Isolados: testam individualmente as unidades e não sua integração;
- *Repeatable* - Repetíveis: com resultados de comportamento constante;
- *Self-verifying* - Auto verificáveis: verificam automaticamente o sucesso ou falha;
- *Timely - Pontual*: o teste deve ser oportuno.

Os tipos de testes executados no TDD são os testes de unidade também conhecidos como testes unitários. O teste de unidade é um meio de garantir que a execução daquele trecho mínimo de código esteja correta. Com isso para cada unidade, é criado um teste, obtendo um padrão de sucesso (SILVEIRA *et al.*, 2012).

Os principais benefícios dessa metodologia são, em conformidade com Rocha (2012), o entendimento sobre o que deve ser desenvolvido, ocasionando na redução de problemas, pois já foram testadas as unidades do sistema e a possibilidade de se realizar análises de tempo empregado para a realização dos testes tornando o processo de refatoração mais dinâmico, contribuindo para a melhoria contínua.

Tendo em vista tais conceitos, a aplicação dessa técnica possibilita gastar bem menos recursos com a resolução de problemas, entendo de maneira ampla como o sistema deve funcionar e que ferramentas são necessárias para sua construção.

2.3.2 Desenvolvimento web

Nos últimos anos, a *World Wide Web* (*www*), tornou-se presente em muitas situações da vida de milhares de pessoas em todo o mundo, ultrapassando todos os outros desenvolvimentos tecnológicos da história (GINIGE; MURUGESAN, 2001).

As aplicações web recebem este nome porque são executados na internet. Ou seja, os dados ou os arquivos são processados e armazenados dentro da web. Estes aplicativos geralmente não precisam ser instalados no seu computador.

O conceito de aplicação web está relacionado com o armazenamento na nuvem. Toda a informação é guardada de forma permanente em grandes servidores de internet e estes enviam a nossos dispositivos ou computadores os dados requeridos neste momento, deixando uma cópia temporária dentro deles, existe uma diversidade de linguagens para desenvolvimento *web*, por exemplo, *Ruby*, *Java*, *Phyton*, *Pearl*, *.NET*, o foco desse trabalho será no *Hypertext Preprocessor*, popularmente conhecido, como PHP.

2.3.2.1 PHP

Segundo o The PHP Group (2017), desenvolvedor da linguagem, o PHP pode ser definido como uma linguagem de *script open source* de uso geral, especialmente adequada para o desenvolvimento *web* e que pode ser embutida dentro do código *HTML*, tendo sido criada em 1994 por Rasmus Lerdorf.

O PHP, como qualquer outra linguagem, possui vantagens e desvantagens, conforme lista Araújo (2009), as principais vantagens, são:

- Fácil aprendizado;
- Velocidade;
- Robustez;
- Multiplataforma.

Como desvantagens Araújo (2009), destaca:

- Compatibilidade entre versões;
- Documentação incompleta;
- Suporte a datas.

A versão mais recente foi lançada em 2017, com alguns ajustes sendo realizados no ano de 2018, essa versão trouxe consigo diversas melhorias focadas principalmente na remoção de funcionalidades depreciadas e na melhoria da consistência da linguagem (THE PHP GROUP, 2017). Sendo que o advento da *web 2.0* tornou necessário pensar na *web* de maneira responsiva.

2.3.2.2 Web responsivo

Surgindo em meados de 2012, o design responsivo segundo Zemel (2013), veio com o objetivo de revolucionar o desenvolvimento de interfaces, trazendo consigo conceitos não somente de reposicionamento de conteúdo, mas também uma nova forma de explorar a grande quantidade de dispositivos existentes no mercado atualmente.

Considerar esse aspecto em um projeto é muito importante, pois ponderar sobre a adaptação do sistema em diversos dispositivos e telas é pensar além das limitações de um *browser desktop* e considerar páginas com flexibilidade que suportem todo tamanho de tela, qualquer tipo de resolução, interfaces com *touch* ou *mouse*. Zemel (2013), afirma que pensar responsivamente é repensar a *web* para o futuro.

2.3.2.3 Frameworks de desenvolvimento web

A grande quantidade de linguagens de programação disponíveis no mercado, permitiu o surgimento de um número equiparável de *frameworks* para atender as necessidades das mesmas, para efeitos de comparação essa proposta vai expor as características dos *frameworks* das linguagens *Java*, *PHP* e *.NET* que segundo Lacerda (2013), são as mais utilizadas para o desenvolvimento de aplicações *web*.

Em relação a linguagem *Java*, um framework que tem conquistado bastante destaque no mercado é o *Spring framework*. Esse *framework* segundo Walls (2011), é utilizado para desenvolver aplicações que são baseadas em técnicas de *Dependency Injection* (DI) e *Aspect Oriented Programming*, cujo objetivo é tornar o desenvolvimento das aplicações *Java* mais simples e com melhor desempenho.

As vantagens mais relevantes do *Spring Framework*, conforme sumariza Johnson (2005), são:

- Disponibilização de camadas de abstração sobre as mais diversas tecnologias, facilitando o processo de desenvolvimento;
- Apesar de ser um *framework* que contempla todas as áreas das aplicações empresariais, não tenta solucionar problemas para os quais já existem soluções comprovadas;
- Facilita a realização de testes, o que muito favorece a qualidade e robustez das aplicações.

Mesmo possuindo vantagens significantes, Johnson (2005) ressalta, que o *Spring* possui algumas desvantagens:

- É uma plataforma muito extensa e complexa. Esta situação pode aumentar consideravelmente a dimensão das aplicações;
- Nas primeiras versões, as configurações eram todas feitas através de ficheiros XML, o que aumentava o tamanho destes e a sua quantidade. Nas versões mais recentes passaram a poder ser feitas em *Java*, entretanto, continua a haver uma grande proliferação de arquivos XML.

Outro exemplo de *framework* Java é o *Apache Struts 2*, definido por Brown (2008), como um *framework open source* para o desenvolvimento de aplicações *web*, que implementa o padrão *model-view-controller* (MVC), com o objetivo de responder as crescentes solicitações *web*.

Brown (2008) sumariza, como principais características, desse *framework*:

- Arquitetura simples quando comparada com outros frameworks;
- Simplicidade nos processos de configuração, pois, utiliza o princípio de *Convention over Configuration*, que permite reduzir significativamente os ficheiros de configuração;
- Comunidade extensa o que pode ser benéfico para a obtenção de ajuda e apoio para os desenvolvedores.

No que concerne aos aspectos negativos, Brown (2008), destaca:

- A documentação existente não é extensa e não está bem organizada;
- O *framework* tem perdido visibilidade em relação a outros mais recentes ou que disponibilizam abordagens diferentes.

Quando lidamos com um projeto em *PHP*, Allen (2009) destaca, o *Laravel*, contemplado posteriormente em um tópico à parte, e o *Zend Framework 2 (ZF2)*, um *framework open source*, desenvolvido pela Zend Technologies, em conjunto com programadores independentes voltado para o desenvolvimento de aplicações *web* utilizando *PHP* e o padrão *MVC*.

As principais vantagens desse *framework* conforme lista, Allen (2009), são:

- O *framework* é composto por vários componentes independentes uns dos outros, que cobrem a maioria dos elementos necessários para o desenvolvimento de uma aplicação *web*;
- Embora seja muito extenso, a sua modularidade facilita a aprendizagem.

No que diz respeito as desvantagens, Allen (2009) ressalta:

- Não está otimizado para o desenvolvimento de pequenas aplicações;

- Embora seja possível excluir os componentes não utilizados, os programadores geralmente não o realizam, resultando no aumento das dimensões das aplicações.

Em relação aos *frameworks* da linguagem *.NET* pode-se destacar o *ASP.Net MVC* que segundo Galloway (2012), é um *framework open source* para o desenvolvimento de aplicações *web* que segue os princípios do padrão *MVC*. Sua primeira versão foi lançada em 2009 e desde o seu lançamento, diversas atualizações têm sido desenvolvidas, sendo as mais recentes a versão 4.0 e a 4.5.

As principais vantagens do *framework*, são:

- Arquitetura *MVC*, que permite que as interações dos utilizadores, sigam um fluxo natural onde as alterações na camada de dados atualizam os componentes da camada de apresentação;
- Maior controle do código *HTML*, o resultado final é um código mais simples facilitando a organização dos elementos visuais. (ESPOSITO, 2010; PALERMO *et al.*, 2012; FREEMAN; SANDERSON, 2011).

As principais desvantagens deste *framework*, são:

- O processo de aprendizagem ser mais demorado e exigir mais dos programadores por se tratar de uma abordagem de mais baixo nível. (ESPOSITO, 2010; PALERMO *et al.*, 2012; FREEMAN; SANDERSON, 2011).

2.3.2.3.1 Laravel

Para a implementação do módulo de gerenciamento *web*, dentre as opções analisadas, optou-se pela utilização do *framework Laravel* em conjunto com o *Bootstrap*.

O Laravel é um framework para desenvolvimento ágil em PHP, sendo ele livre e de código aberto, seu objetivo é proporcionar que seja possível trabalhar de uma forma estruturada e rápida. Fornecendo todas as ferramentas para que seja possível começar a programar o que for necessário sendo simples e bem intuitivo (ROBERTO, 2017).

As principais vantagens do *Laravel*, conforme destaca Roberto (2017), são, a documentação da linguagem, o fato de utilizar o *Composer* para gerenciar as suas dependências e a utilização do sistema de rotas que facilita o monitoramento de *URLs* digitadas no navegador, entretanto, para o seu perfeito funcionamento alguns requisitos devem ser atendidos, sendo que o XAMPP pode fornece-los, conforme será exposto na secção 2.3.2.3.1.1.

O *Bootstrap* é um dos frameworks mais populares para desenvolvimento *front-end*, facilita o desenvolvimento de páginas responsivas e *templates*, desde a criação de protótipos simples para provas de conceito, por exemplo, como até plataformas completas.

Originalmente criado como um projeto paralelo dentro da organização do *Twitter*, é um *framework* que está em constante evolução, ou seja, sua aceitação no mercado está em constante crescimento e novas melhorias ainda estão em desenvolvimento (BOOTSTRAP, 2018).

2.3.2.3.1.1 XAMPP

O XAMPP é uma distribuição para desenvolvedores contendo um pacote com os principais servidores de código aberto do mercado, incluindo *FTP*, banco de dados *MySQL* e *Apache* com suporte às linguagens PHP e Perl (HIGA, 2012). Com o XAMPP, é possível executar sistemas como *WordPress* e *Drupal* localmente, facilitando e agilizando o desenvolvimento, visto que, o conteúdo é armazenado em uma rede local o acesso aos arquivos é facilitado.

O XAMPP está disponível para quatro sistemas operacionais: Windows, Linux, Mac OS X e Solaris. Caso não queira instalá-lo diretamente no seu computador, basta extrair o arquivo compactado, de preferência para a pasta raiz do seu disco rígido ou um local de fácil acesso e executar o aplicativo que inicia os servidores.

2.3.3 Desenvolvimento mobile

A implementação de um módulo *mobile* em um sistema é uma etapa que requer bastante análise tanto pela complexidade dos requisitos, como, pela grande variedade de dispositivos existentes no mercado e que devem ser atendidos no sistema. Segundo Martins, Antônio e Oliveira (2013), a construção de uma aplicação para dispositivos móveis e a escolha do tipo de aplicativo é um ponto muito importante no processo de desenvolvimento do projeto. Ao iniciar o projeto é necessário realizar uma análise minuciosa e estratégica sobre a plataforma, sistemas, produtos e arquiteturas que deverão ser utilizadas.

A indústria dos dispositivos móveis não segue um padrão na construção de seus produtos. Dessa forma um mesmo smartphone, por exemplo, pode conter recursos como câmera e sensores de diferentes fabricantes, aumentando ainda mais a diversidade de requisitos que deve ser atendida pelos novos aplicativos.

Gonçalves (2017) defende que a variedade de dispositivos móveis e de sistemas operacionais existentes impõe um desafio aos desenvolvedores, pois estes devem criar softwares capazes de funcionar adequadamente em todos estes equipamentos, Gasparotto

(2017) aponta que é fundamental saber escolher as ferramentas com as quais se vai trabalhar no desenvolvimento, para se obter produtividade, manutenibilidade e testabilidade, por exemplo.

Os principais tipos de aplicativos móveis a serem considerados no início de um projeto são: Aplicativos *Web Apps* (sites móveis) que consistem em soluções feitas para a *web* formatadas para serem acessadas através do *browser* dos dispositivos móveis; aplicativos móveis nativos que consistem em soluções desenvolvidas para um determinado tipo de dispositivo e sistema operacional; e aplicativos híbridos que consistem em aplicações desenvolvidas com a junção do *Web* com o nativo.

Gasparotto (2017) defende que o desenvolvimento de aplicativos híbridos está em constante evolução, o que é justificado pela necessidade cada vez maior de entregar a mesma solução para diferentes plataformas. Gonçalves (2017) aponta que o desenvolvimento de aplicativos híbridos pode ser a melhor alternativa, para contornar a diversidade de dispositivos móveis que existe no mercado, reduzindo custos com tempo e recursos humanos.

O nome híbrido é dado pela junção de código nativo para empacotamento e distribuição do aplicativo com o código não nativo (*HTML, CSS e JavaScript*), responsável pelo visual e funcionalidades da aplicação.

2.3.3.1 Frameworks de desenvolvimento Mobile

Dentre os *frameworks* existentes para o desenvolvimento de aplicações *mobile* este trabalho irá explorar o *Xamarin* e o *Ionic*.

A primeira versão do *Xamarin* foi anunciada em julho 2001, por Miguel de Icaza e Nat Friedman, para fornecer suporte e desenvolver softwares, para um projeto em desenvolvimento de Icaza, o *GNOME*, sendo na época um projeto *open source* denominado *Mono*.

Em abril de 2011 a empresa mantenedora do projeto *Mono* foi adquirida pela *Attachmate Group* e os funcionários responsáveis pelo projeto foram demitidos, em maio do mesmo ano, Icaza anunciou que o projeto continuaria com o time demitido pela *Attachmate*, mas sendo renomeado como *Xamarin*, sendo que em fevereiro de 2016 o projeto foi adquirido pela Microsoft, por um valor especulado entre US\$ 400 milhões e US\$ 500 milhões.

O *Xamarin* é uma plataforma que visa amenizar a necessidade de desenvolver códigos separados para cada sistema móvel existente no mercado, tornando possível criar aplicativos usando uma mesma linguagem de programação, C# (*C-Sharp*) (PROCEDI, 2016).

O *Xamarin* possibilita o desenvolvimento de aplicações nativas, através de um compilador que gera o código nativo para a plataforma desejada, trazendo vantagens como desempenho, *user interface* ideal e utilização direta dos recursos disponíveis no dispositivo,

sendo o diferencial do *framework* não se basear em uma interface própria e sim na construção de interfaces que respeitem os conceitos de interface da cada plataforma alvo, em outras palavras a aplicação é a mesma, mas o usuário a enxerga diferente em virtude da plataforma utilizada. (GASPAROTTO, 2015).

Embora seja um excelente *framework*, o *Xamarin* não é completamente compatível com as plataformas alvo, sendo limitado a cerca de 75%, sendo que os 25% restantes devem ser implementados em separado, o que torna a curva de aprendizagem não tão baixa, quanto de outras soluções encontradas no mercado, mesmo que utilizando C# para a realização de tal implementação, existe a necessidade do desenvolvedor, aprender conceitos relacionados aos sistemas *iOS* e *Android*. (GASPAROTTO, 2015).

2.3.3.1.1 Ionic

O *Ionic* foi criado pela *Drifty Co.* em 2013, é uma estrutura de desenvolvimento de aplicativos móveis HTML5, com o objetivo de construir uma estrutura que se concentrasse no desempenho com os padrões modernos da *web* (GONÇALVES, 2017). Desde o seu lançamento, foram criadas algumas versões, com mudanças significativas entre a primeira versão e a versão atual. Em 2015 foram criados mais de 1,3 milhões de aplicativos com o *framework*. Várias plataformas são suportadas pelo mesmo, com ênfase no *Android* da versão 4.1 em diante e no *iOS* a partir da versão 7.

O *Ionic* apresenta uma alta aceitação no mercado, devido a preocupação de sua equipe com a otimização e performance das aplicações desenvolvidas com ele, assim como, uma comunidade ativa e disposta a auxiliar iniciantes no *framework* (GONÇALVES, 2017).

As aplicações híbridas, desenvolvidas com o *Ionic* possuem o seu desempenho melhorado, devido a utilização do *Angular* ao invés do *JQuery*, sendo o *Angular* alavanca as transições CSS transformando-as para animações afim de aproveitar melhor a GPU e otimizar o tempo de processamento do dispositivo.

Todo o código desenvolvido em uma aplicação híbrida é executado dentro de um recurso chamado *WebView*, que funciona como um *browser*, esse recurso é executado quando a aplicação híbrida é requisitada pelo usuário, sendo que essa forma de execução não fica explícita para o usuário, deixando a impressão de que o aplicativo se comporta da mesma forma que um aplicativo nativo (GONÇALVES, 2017).

2.3.4 Banco de dados

Um banco de dados pode ser definido, como uma coleção de dados inter-relacionados, representado informações sobre um domínio específico, sendo essa uma definição genérica, em outras palavras,

Um banco de dados é uma coleção de dados, os mesmos são fatos que podem ser gravados e que possuem um significado implícito. Por exemplo, considere nomes, números telefônicos e endereços de pessoas que você conhece (ELMASRI; NAVATHE, 2006; KORTH, 1994).

Existem diferentes tipos de banco de dados, como por exemplo, relacional, multimídia, não relacional, entre outros. Esse trabalho será focado nos bancos de dados não relacionais, cuja grande motivação é resolver os problemas de escalabilidade dos bancos tradicionais, entretanto, será realizado um comparativo para melhor compreensão da adoção desse modelo de banco, com o banco de dados relacional *SQLite*.

No modelo de banco de dados relacional, os dados são organizados na forma de uma tabela onde as colunas representam as informações que a tabela irá conter e as linhas de cada coluna os dados, sendo que nesse modelo de banco de dados é necessário conhecer a estrutura do banco, pois, a mesma deve ser definida antes de ser possível realizar qualquer inserção no banco (DIAS NETO, 2011). Para os propósitos deste trabalho, se destaca no modelo relacional o *SQLite*.

O *SQLite* é um banco de dados simples, que implementa uma biblioteca em C, sendo que seus tipos de dados são limitados, suportando os tipos *text*, *integer* e *real*, todos os outros tipos devem ser convertidos em um desses campos antes de serem salvos. Contudo, o *SQLite* não valida se os valores adicionados nas colunas são dos tipos definidos (CORDEIRO, 2013).

Apesar de simples e limitado, ele é um banco de dados leve e rápido, tornando-o perfeito para dispositivos *Android* ao contrário de outros bancos de dados que ficam em servidores, como *Oracle* e *Microsoft SQL Server* (CORDEIRO, 2013).

O *SQLite* utiliza uma classe de implementação *SQL*, porém nem todas as funções estão configuradas na mesma, o banco não possui nenhuma licença de uso, sendo disponibilizado pelos autores, ou seja, qualquer pessoa pode baixar as fontes no site, compilar, modificar, executar e utilizar sem problema algum, entretanto, Cordeiro (2013) ressalta que em alguns casos os usuários devem adquirir uma licença de uso para podem utilizar o *SQLite*, sendo os mesmos sumarizados abaixo:

- Utilização em instituições privadas;

- Utilização em instituições que não reconhecem o autor;
- Caso se opte pela não publicação da licença de uso para os usuários;
- Caso o departamento legal informe a necessidade de se adquirir a licença.

No modelo não relacional ou *NoSql*, as informações são dispostas de maneira a fornecer uma melhor performance, sendo agrupadas em um registro, tal registro dispensa a necessidade de definição de uma estrutura para o banco, não sendo necessário ainda a construção de diferentes tabelas para ser formar a informação o que ocorre no modelo relacional, a informação estará em sua totalidade no mesmo registro (IANNI, 2012). Dentre os bancos de dados não relacionais, pode-se destacar ainda os que operam em tempo real, nessa categoria se destaca, o *firebase*, que em virtude dessa característica foi selecionado para ser o banco de dados desse projeto, no módulo *mobile*.

2.3.4.1 Firebase

O *Firebase Realtime Database* é um banco de dados *NoSQL*, hospedado na nuvem, os dados são armazenados como *JSON* e sincronizados em tempo real com todos os clientes conectados que compartilham uma mesma instância (GOOGLE, 2018).

O *Firebase* realiza sincronização de dados sempre que os mesmos são alterados, permitindo que os dispositivos conectados recebam atualizações em questão de milissegundos, criando experiências imersivas e colaborativas sem a necessidade de se preocupar com códigos de rede, os dados permanecem responsivos mesmo quando o aparelho está fazendo uso da aplicação off-line, os dados são mantidos em disco e sincronizados no instante em que a conexão é restabelecida.

As informações no banco podem ainda ser validadas por uma série de regras definidas pelo administrador da aplicação, permitindo validar os tipos de dados sendo fornecidos, assim como, delimitar níveis de acesso.

3 DESENVOLVIMENTO

Este capítulo descreve os passos realizados para a implementação do sistema Donar, utilizando como base para a construção do mesmo a metodologia TDD, juntamente com as orientações do *framework Scrum* para gerenciamento de projetos.

3.1 Elicitação de requisitos

Visando a obtenção de requisitos para a aplicação, realizou-se uma pesquisa bibliográfica com enfoque na problemática da doação de sangue no Brasil, onde em um primeiro momento o foco da proposta era no desenvolvimento de um sistema *mobile*, entretanto, a medida em que o projeto foi sendo refinado constatou-se a necessidade de um sistema para gerenciamento das atividades realizadas nos locais de doação, essa necessidade originou a reestruturação da proposta em um sistema modular contendo uma implementação *mobile* para os doadores e um sistema de gerenciamento *web* para os locais de doação.

As informações acima, foram obtidas por intermédio de pesquisas em diferentes fontes, dentre elas, Anatel, Organização Mundial da Saúde, Organização Pan-Americana da Saúde e o Ministério da Saúde e demais órgãos de saúde, permitindo a geração de dados quantitativos para o projeto.

3.2 Delimitação de escopo

Durante o desenvolvimento de uma atividade, independente do meio onde é realizada, surgem conflitos, sejam por consequência de diferentes ideologias, pontos de vista ou mesmo formas de realizar um processo, frente a essa realidade, faz-se de sumária importância a estruturação de medidas que garantam um entendimento comum sobre o que deve ser realizado para o bom andamento de um projeto.

No que diz respeito ao desenvolvimento de sistemas o entendimento quanto ao que deve ser realizado pode ser obtido por meio da documentação, sendo essa, uma forma de proteger a equipe e o processo, neste trabalho os documentos norteadores do projeto, são: o documento de visão, arquitetura e o de requisitos, Apêndices A, B e C, respectivamente, estando dispostos nesses finalidade do projeto, escopo, visão geral, assim como, descrições quanto as restrições e características a serem atendidas.

O processo de construção da documentação, assim como, as demais etapas deste trabalho, foram conduzidas em conformidade com as orientações do *framework Scrum*, sendo as atividades estruturadas, por meio dos quadros de atividades da ferramenta *Trello*, buscando atender o princípio de transparência do *framework* ao tornar visível aos envolvidos no processo as características que devem ser atendidas para o sucesso da proposta.

O processo de construção da documentação e a delimitação de tarefas encontrasse exposto na figura 2, estando as mesmas dispostas em documentos implementados, atividades a serem realizadas, atividades em andamento e atividades concluídas, permitindo uma visão dinâmica do sistema, contribuindo, para o alinhamento de perspectivas quanto ao projeto e seus objetivos.

Figura 2 - Processo de Construção da Documentação

Documentação

Documento de Requisitos

Termo de Abertura do Projeto

Montagem da Estrutura Analítica do Projeto

Tabela de Funções do Sistema Círculo

Membros	Documentação	Negociação	Desenvolvimento	Testes	Treinamento
Erick Junior	X		X	X	
Gabriel Leite	X	X	X	X	
Joimar Cândido			X	X	
Matheus Lima	X	X	X	X	X
Lucas Andrade	X	X	X	X	X
Rafael de Souza			X	X	

Designação das Funções dos membros da equipe

Construir o Produto Backlog

Definir o Sprint Backlog

A fazer

Desenvolver o fluxograma do sistema

Desenvolver o Diagrama de caso de uso

Levantamento de Requisitos

Desenvolver o Diagrama de Classes

Criar esquema de acesso as funções

Implementar as Regras de Negócio do sistema

Desenvolvimento da API

Desenvolver Casos de Teste

Em Andamento

Validação dos Requisitos coletados

Concluído

+ Adicionar um cartão

Fonte: Os Autores

3.3 Definição da arquitetura e criação de protótipos

Determinada a esquematização do sistema, foi realizada uma reunião para definição da arquitetura do sistema, através da qual se obtém o conhecimento sobre os componentes do sistema e seus inter-relacionamentos, essa reunião culminou na escolha do *framework Ionic* para o módulo *mobile* e da linguagem *PHP* para o *web*, ambos sendo estruturados em conformidade com a arquitetura MVC, a escolha pelas mesmas foi guiada, por aceitação mercadológica, comparativos com outras tecnologias, assim como, pela experiência dos membros da equipe.

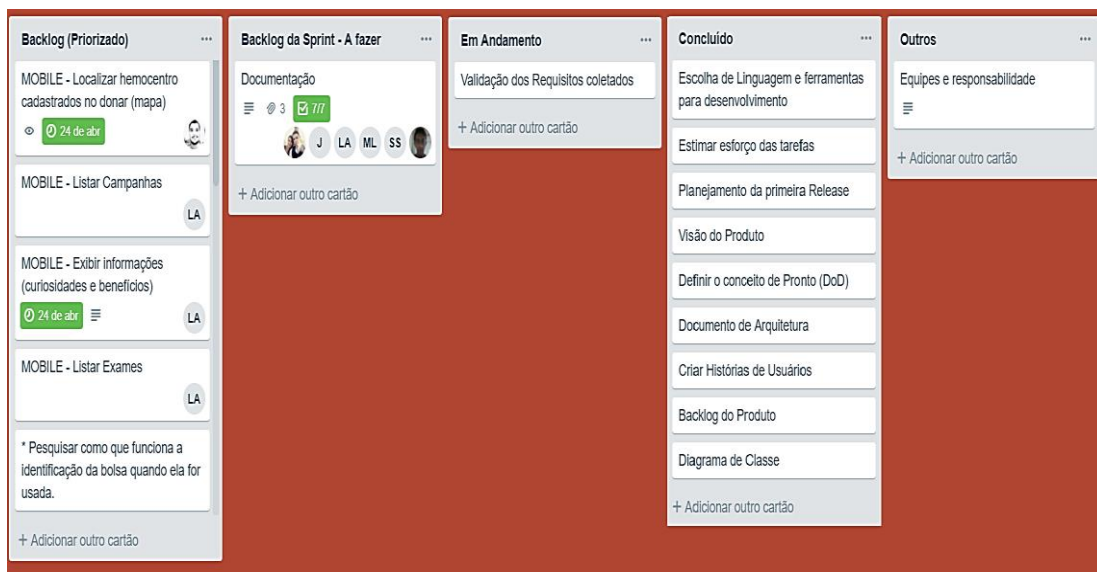
Para a estruturação e desenvolvimento do projeto foi escolhida a metodologia orientada a testes (TDD), sendo a mesma, escolhida por proporcionar uma visão mais ampla quanto ao que deverá ser realizado pelo desenvolvedor, pois, antes de codificar a funcionalidade esse deve entender o problema e definir os testes.

Tendo a metodologia sido definida, optou-se, pelo gerenciamento do trabalho por *Scrum*, visto, ser um *framework* capaz de suportar e adaptar-se a utilização de diferentes metodologias, levando em consideração as habilidades e o conhecimento dos membros componentes da equipe de desenvolvimento.

As demais escolhas quanto a arquitetura guiaram a definição de restrições, características e indicadores de qualidade para o sistema, estando a descrição detalhada dessas escolhas elucidada no documento de arquitetura, Apêndice B.

Com arquitetura e requisitos definidos, realizou-se o processo de planejamento das *sprints* (ciclos de atividades), definindo o tamanho e estimativa de horas para o desenvolvimento das mesmas, tendo como resultado final desse processo a priorização e estruturação das *features* a serem desenvolvidas, constituindo o *backlog*, uma lista de requisitos a serem atendidos durante o decorrer de um projeto, sendo que esses deverão ser entregues ao proprietário do sistema, em algum momento no tempo, conforme exposto na figura 3.

Figura 3- Priorização das Features



Fonte: Os Autores

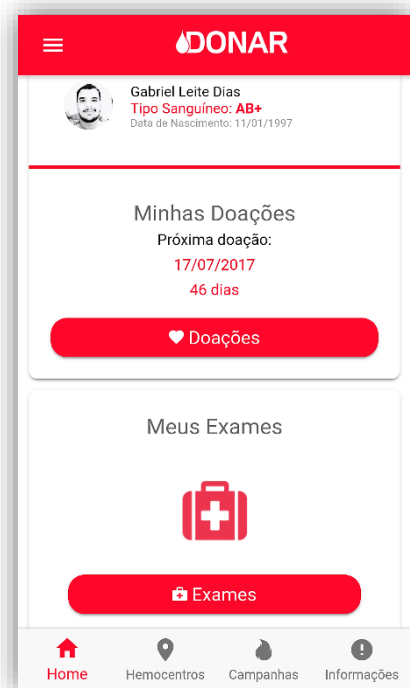
Estando o projeto estruturado a equipe optou pela construção do modelo MVP, conceituado como uma técnica que consiste em desenvolver um protótipo de uma versão mínima com as funcionalidades necessárias para o seu funcionamento, por meio do qual, é possível validar aspectos comportamentais e fluxos de execução do sistema (FONSCECA, 2017). A implementação do MVP permite as possíveis partes interessadas, validar e realizar adequações de maneira mais dinâmica a solução, para sua construção foram prototipadas as telas da aplicação, sendo que para o *mobile* foram idealizadas em conformidade com os recursos do *Ionic*, figuras 4, 5, 6, 7 e 8.

Figura 4 - Protótipo da Tela Inicial



Fonte: Os Autores

Figura 5 - Protótipo da Tela de Perfil



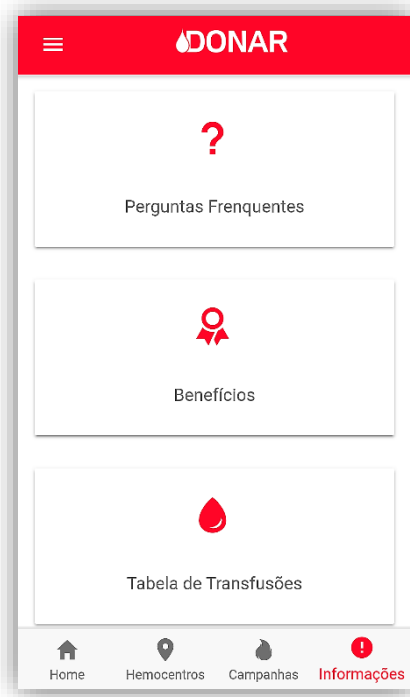
Fonte: Os Autores

Figura 6 - Protótipo da Tela de Campanhas



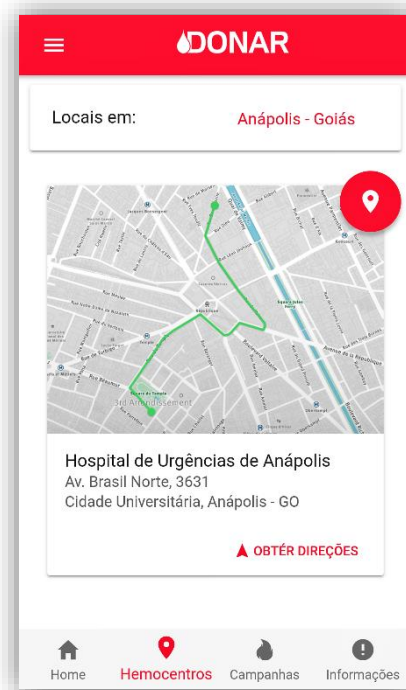
Fonte: Os Autores

Figura 7 - Protótipo da Tela de Informações



Fonte: Os Autores

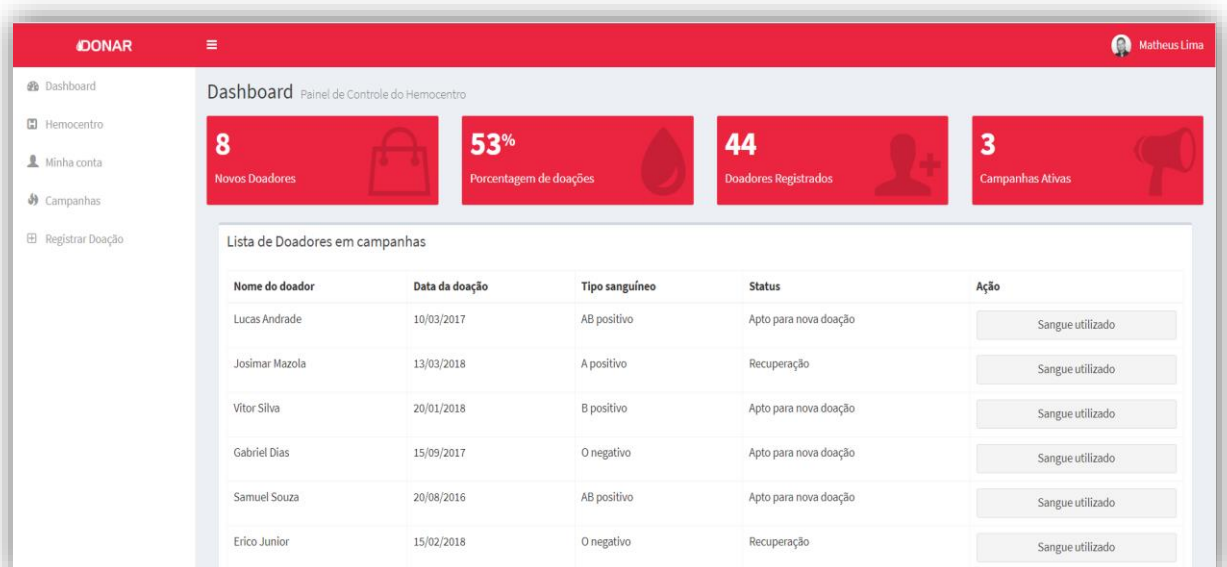
Figura 8 - Protótipo da Tela Locais para Doação



Fonte: Os Autores

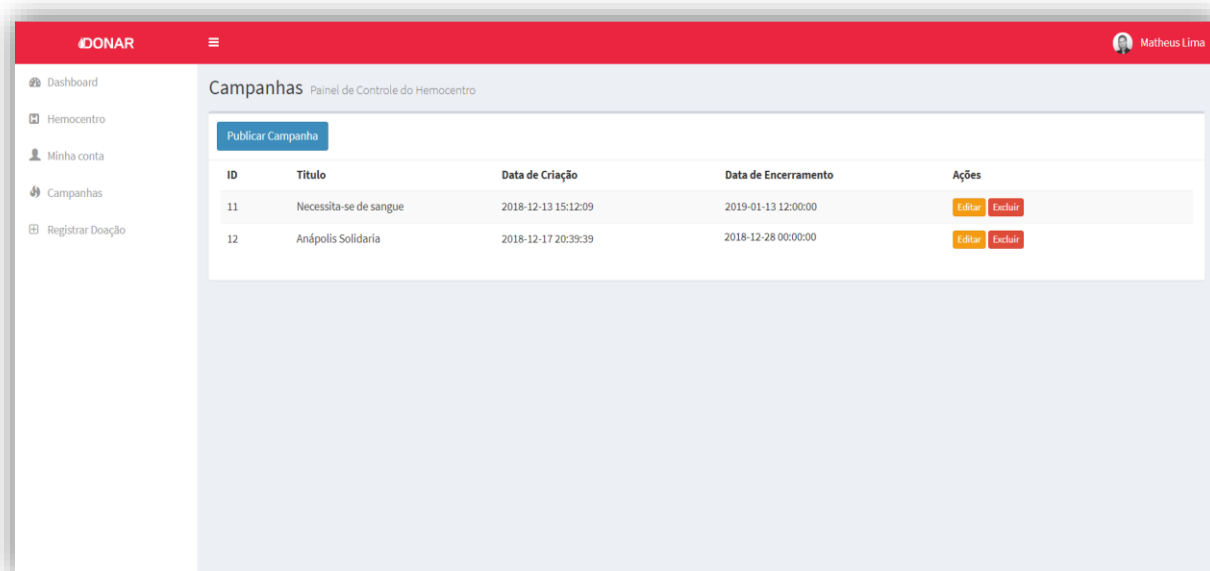
Enquanto, o módulo *web* teve seu MVP construído em conformidade com o recursos disponíveis no *framework bootstrap*, sendo realizado para tanto um estudo das principais funções que poderiam gerar valor agregado aos locais de coleta de sangue, levando em consideração ainda sua integração com o módulo *mobile*, sendo o resultado deste processo originado os protótipos contemplados na figuras 9, 10 e 11.

Figura 9 - Protótipo do Dashboard de Gerenciamento



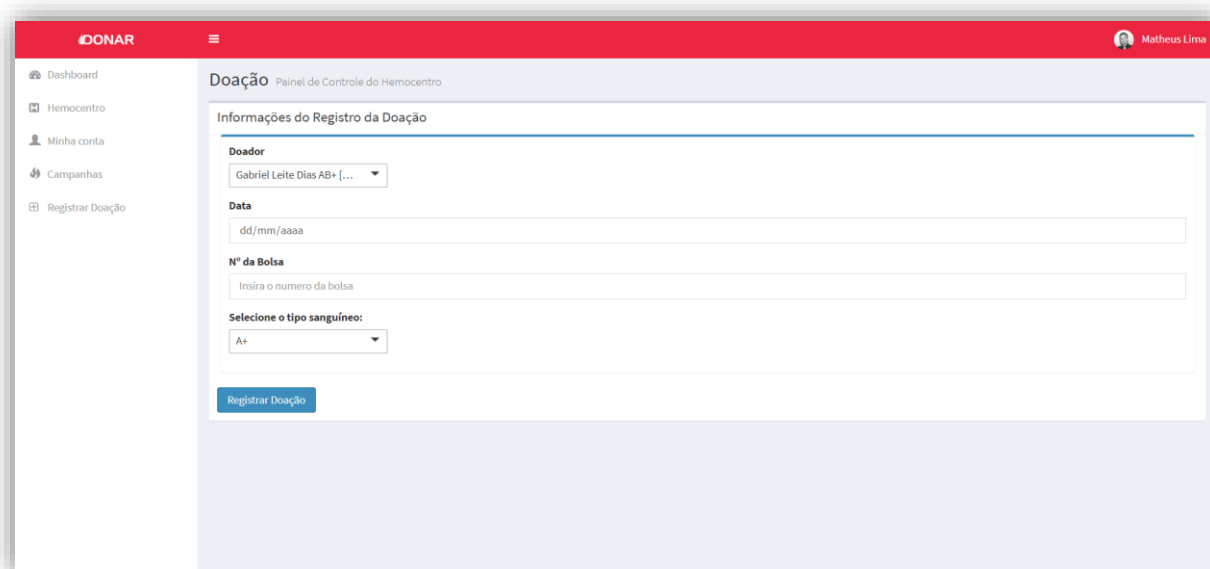
Fonte: Os autores

Figura 10 - Protótipo da Tela de Campanhas Ativas



Fonte: Os Autores

Figura 11 - Protótipo da Tela de Registro de Doação



Fonte: Os autores

3.4 Desenvolvimento e validação do MVP

Construídos os protótipos de tela, no período de julho a setembro, foram implementados os modelos MVP da proposta, sendo realizada uma visita ao Instituto Onco-Hematológico de Anápolis (IOHA), para validação dos mesmos, sendo a responsável pelo processo a supervisora de doações, Cleonice Maria Garcia de Souza, que além de validar os modelos, sugeriu

melhorias e complementou informações sobre o processo de doação de sangue, gerando como resultado o documento de validação, Apêndice E.

3.5 Definição e implementação de testes

Uma vez que, o modelo MVP foi validado, foram determinados os aspectos metodológicos para a construção da solução, sendo que esses não foram aplicados ao MVP, por se tratar de uma versão mínima executável do sistema, para definição da metodologia foram selecionadas as que melhor se adaptavam a projetos ágeis, optando ao final pela aplicação da metodologia TDD, visto que, corrobora para uma melhor qualidade do código desenvolvido, assim como, uma melhor compreensão do escopo por parte da equipe de programação.

O detalhamento da abordagem dos testes segue no plano de testes, Apêndice D, onde, foram definidos testes automatizados por meio do *Gitlab CI*, os testes para o módulo *web* foram codificados, por intermédio, do *PHPUnit* e os do *mobile* pelo *karma* juntamente com o *selenium*, ferramentas de teste incorporadas ao angular, adaptadas pela comunidade do *Ionic*, podendo ser utilizadas no mesmo, a figura 12 demonstra a codificação de um teste, enquanto, a figura 13 contempla as *pipelines* de integração continua no *Gitlab CI*, lembrando que devido a aplicação de TDD ao projeto os testes foram realizados antes do processo de implementação da solução.

Figura 12 - Teste Automatizado de Inserção de Campanhas

```
public function InsertCampaignTest(Request $request)
{
    $campController = new CampaignController();

    $request->titulo = "Nova Campanha";
    $request->texto = "Texto Nova Campanha";
    $request->tipoSang = array("A", "AB");

    $result = $campController->insertData($request);

    $this->assertNotNull($result);

    return $result;
}
```

Figura 13 - Pipelines de Integração Contínua

Status	Commit ID	Author	Branch	Job Name	Duration	Time Ago
falhou	#35521514	by donation	master -> 2d5999b	donation	00:04:18	1 week ago
bem sucedido	#31609635	by developer	developer -> 85d7dc8c	Tests	00:04:10	1 month ago
bem sucedido	#31090325	by donation	master -> 85d7dc8c	Tests	00:05:27	1 month ago
falhou	#31089902	by donation	master -> 7198397d	Campaign test	00:05:35	1 month ago
falhou	#31089458	by donation	master -> d1a723c0	Change Campaign insert feed...	00:06:23	1 month ago

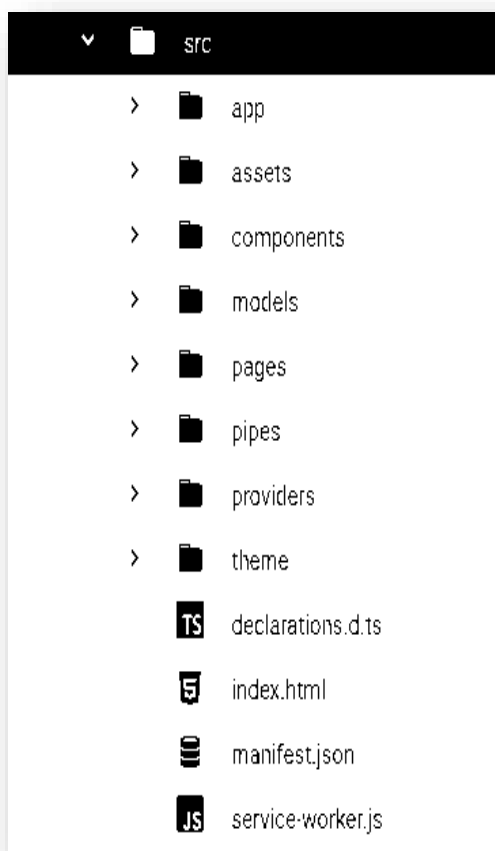
Fonte: Os autores

Com as *pipelines* de integração contínua e os testes configurados, se iniciou o processo de desenvolvimento.

3.6 Codificação da solução

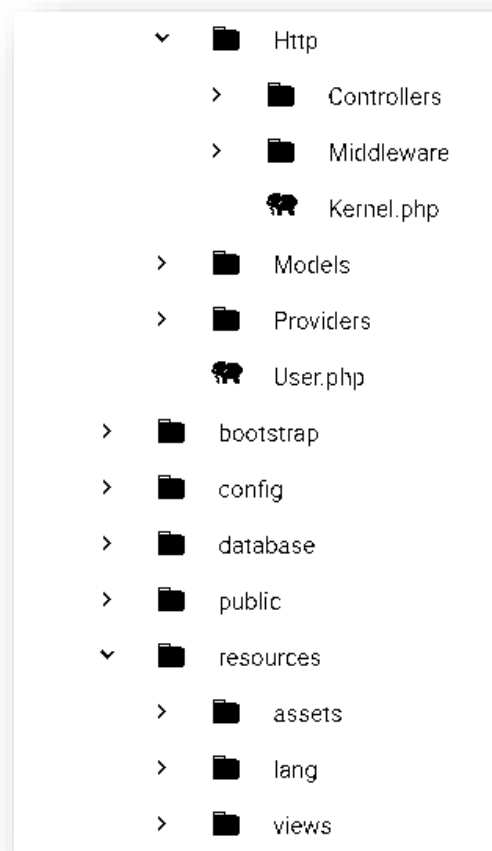
As etapas de desenvolvimento do módulo *mobile* foram realizadas, por meio, do *framework Ionic*, o qual permite desenvolver aplicações multiplataforma. O módulo *Web*, por sua vez, foi implementado em PHP com auxílio do *framework Laravel*, sendo que, ambos os *frameworks*, foram estruturados em conformidade com o padrão arquitetural MVC, a estruturação dos projetos é representada a seguir, por meio das figuras 14 e 15, vale ressaltar, que o *Laravel* trouxe ao módulo PHP uma implementação para controle de rotas, assim como a existente no *framework Ionic* que herda essa característica do angular.

Figura 14 - Estruturação do Projeto Ionic



Fonte: Os Autores

Figura 15 - Estruturação do Projeto Web



Fonte: Os autores

A seguir serão descritos os principais aspectos do processo de implementação, sendo que por ambos os projetos serem estruturados em conformidade com a metodologia MVC, serão explicados somente os aspectos gerais de desenvolvimento, com os respectivos códigos de cada implementação.

3.6.1 Model

Camada responsável por armazenar os dados e notificar a *view* e o *controller* quando há uma mudança em seu estado. Estas notificações permitem que a *view* produza saídas atualizadas e que o controller altere o conjunto de comandos disponíveis, contendo ainda abstrações das entidades do sistema, conforme exemplificado pela figura 16.

Figura 16 - Model das Campanhas

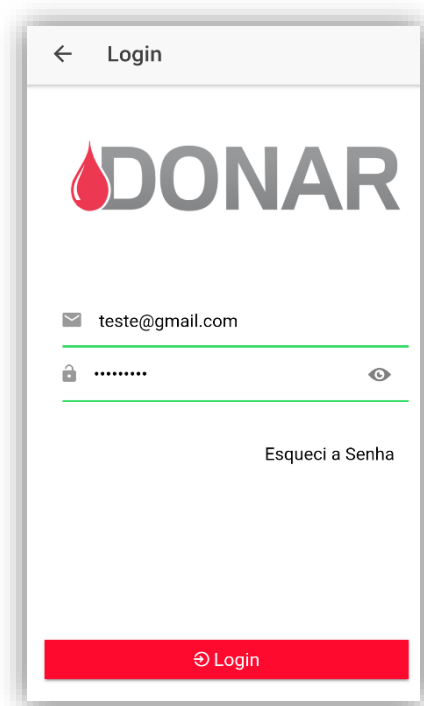
```
export class Campaign{  
  id: string;  
  title: string;  
  description: string;  
  image: string;  
}
```

Fonte: Os Autores

3.6.2 View

Camada de interação com o usuário. A *view* faz a exibição dos dados, sendo por meio de *html* ou *xml*. A figura17 exemplifica, uma das *views* do sistema *mobile*, contendo a tela de login da aplicação.

Figura 17 – View Tela de Login



Fonte: Os Autores

3.6.3 Controller

É o responsável por receber todas as requisições do usuário. Seus métodos chamados *actions* são responsáveis por uma página, controlando qual *model* usar e qual *view* será mostrado ao usuário. Conforme demonstrado na figura 18, que contém o processo de validação do *login*, caso o mesmo seja bem sucedido realiza o redirecionamento para o *dashboard*.

Figura 18 - Controller da Tela de Login

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Hemocentro;
use Illuminate\Support\Facades\Auth;

class DashboardController extends Controller
{
    public function __construct()
    {
        | | $this->middleware('auth');
    }

    public function index()
    {
        | | $hemocentro = Hemocentro::where('user', Auth::id())->first();
        | | return view('hemocentro.dashboard', ['hemocentro'=>$hemocentro]);
    }
}
}
```

Fonte: Os Autores

3.6.4 Component

O *component* é uma diretiva de configuração simplificada. Enquanto novas estruturas e funcionalidades encorajam o uso de boas práticas de codificação, isto por si só não é suficiente para a produção de um bom código, os componentes permitem que trechos simples de código possam ser reutilizados em diferentes partes da aplicação sem a necessidade de se copiar o código, conforme demonstrado nas figura 19 e 20, que exemplificam o processo de injeção de um *component* para exibição das informações dos locais de doação cadastrados no sistema.

Figura 19 - *Component* para Listagem de Hemocentros

```

<div>
  <ion-card>
    
    <ion-fab right top>
      <button ion-fab>
        <ion-icon name="pin"></ion-icon>
      </button>
    </ion-fab>
    <ion-item>
      <h2>{{bloodCenter.name}}</h2>
      <p>{{bloodCenter.address}}, {{bloodCenter.number}} </p>
      <p>{{bloodCenter.neighborhood}}, {{bloodCenter.city}} - {{bloodCenter.state}}</p>
    </ion-item>
    <ion-item>
      <button ion-button icon-start clear item-end>
        <ion-icon name="navigate"></ion-icon>
        Obtér Direções
      </button>
    </ion-item>
  </ion-card>
</div>

```

Fonte: Os Autores

Figura 20 - Função para injeção do *Component*

```

import { Component, Input } from '@angular/core';

import { BloodCenter } from '../models/blood-center';

@Component({
  selector: 'blood-centers',
  templateUrl: 'blood-centers.html'
})
export class BloodCentersComponent {

  @Input()
  bloodCenter: BloodCenter;

}

```

Fonte: Os autores

3.6.5 Provider

De maneira resumida, é uma camada de encapsulamento para as regras de negócio de uma determinada operação em um sistema, provendo uma abstração do processo de conexão, a figura 21 exemplifica o processo de *crud* por meio de *providers*, onde por meio de uma referência acessa o banco de dados e realiza as operações na mesma.

Figura 21 - Provider Hemocentros

```
@Injectable()
export class BloodCenterProvider {
  reference;
  constructor(private af: AngularFireDatabase) {
    this.initialize();
  }
  private initialize(){
    this.reference = this.af.database.ref('/bloodCenter/');
  }
  save(place: BloodCenter){
    let id;
    if(place.id != undefined){
      id = place.id;
    }else{
      id = this.reference.push().key;
      place.id = id;
    }
    this.reference.child(id).update(place);
  }
  delete(place: BloodCenter){
    return this.reference.child(place.id).remove();
  }
}
```

Fonte: Os autores

3.7 Gerenciamento dos dados

No que concerne ao banco de dados optou-se por uma solução em tempo real (*Firebase*) para o módulo *mobile*, ficando esse responsável, pelo gerenciamento de usuários, replicação de campanhas e distribuição de informativos ao usuário, visto que, sendo uma solução em tempo real é capaz de replicar de maneira imediata alterações no banco, capacidade que o torna a escolha ideal para criação, difusão e replicação de campanhas ou ações, em situações emergenciais ou de alta prioridade.

Para o banco de dados *Web*, foi definido o uso do *MySQL*, sendo as informações advindas do *firebase* replicadas para o mesmo, visando, manter a consistência das mesmas nos diferentes módulos que compõem o sistema, sendo responsável em primazia, por armazenar informações sobre os locais de doação, resultados dos exames realizados, assim como, os dados das doações recebidas, fornecendo ao *firebase* a posição dos locais de doação, os *links* para *download* de exames e notificando o mesmo, quando uma bolsa de sangue for utilizada, sendo

3.8 Integração entre módulos

Com os módulos *mobile* e *web* desenvolvidos, o processo de integração entre os mesmos foi iniciado, sendo que o *firebase*, fornece a estrutura necessária para que possam ser realizadas requisições externas a base de dados, característica definida como, Backend as a service (BaaS), contextualizada como,

Um modelo de serviço de computação em nuvem que serve como middleware fornecendo aos desenvolvedores maneiras de conectar seus aplicativos da Web e móveis a serviços em nuvem por meio de interfaces de programação de aplicativos (API). Em comparação com outros modelos de serviço no ambiente de computação em nuvem. (TECHOPEDIA, 2018).

Com a configuração de acesso realizada, o módulo *web* se tornou capaz de centralizar as informações dos usuários da aplicação *mobile*, podendo ainda gravar na base do *firebase* informações sobre doações, exames, campanhas e utilização do sangue.

4 CONSIDERAÇÕES FINAIS

A realização desse projeto, constituiu uma experiência de grande importância, seja de uma perspectiva profissional, como pessoal. Ao desenvolver a ferramenta proposta, verificou-se que existem diversas tecnologias disponíveis no mercado, sendo que, o processo de escolha das ferramentas trouxe a equipe maturidade para consolidar a proposta em um produto de fato, tendo esse plena capacidade de se desenvolver e em um estado de maior maturidade, ser lançado no mercado.

No início, do processo a equipe apresentou certa dificuldade em alinhar ideias com fluxo de trabalho, esses percalços, puderam ser superados, mediante, o aumento de comunicação e a definição do processo, mudanças que a longo prazo levaram a equipe a um status de alinhamento dos objetivos, tornando-se capaz de realizar as atividades propostas, fazendo uso do *framework* Scrum de maneira eficiente, o que denota a importância da comunicação no processo de desenvolvimento de sistema.

Compreendeu-se ainda a importância de se preparar para realizar incrementos as funções à medida em que os requisitos sofrem alterações, compreendendo se tratar de um processo natural no desenvolvimento de um sistema e buscando formas de realizar esse processo de maneira a causar o menor impacto possível, principalmente, em virtude da utilização do TDD no projeto que requer pensar nos testes antes do desenvolvimento, contribuindo para um melhor entendimento quanto ao deveria ser construído, assim como que necessidades deveriam ser primeiramente atendidas para satisfazer o teste e posteriormente como melhorar a solução encontrada durante a etapa de refatoração do código, permitindo ainda, entender de forma clara o processo de integração contínua.

Por fim, com o que já foi atingindo, espera-se conseguir parcerias que viabilizem a implantar do sistema no mercado, objetivando que mais pessoas possam ser ajudadas, assim como, hemocentros e bancos de sangue possam melhorar o seu processo de gerenciamento, tendo na tecnologia uma aliada no processo de melhoria do processo, servindo ainda, como veículo social na preservação da vida das pessoas envolvidas direta ou indiretamente com o processo de doação de sangue.

REFERÊNCIAS BIBLIOGRÁFICAS

ACIOLI, Thiago. **Qual o impacto das redes sociais nas nossas vidas?** 2016. Disponível em: <<https://medium.com/publicitariossc/qual-o-impacto-das-redes-sociais-nas-nossas-vidas-9fe7faf7c83b>>. Acesso em: 13 mar. 2018.

ALLEN, R.; LO, N; BROWN, S., Zend Framework in Action, Manning, 2009

ANDRADE, Ana Paula; VIANA, Phil. **Criação e Geração de Planos de Teste de Software.** 2012. Disponível em: <https://www.ibm.com/developerworks/br/local/rational/criacao_geracao_planos_testes_software/index.html>. Acesso em: 04 mai. 2018.

ARAÚJO, Fabrício, Vantagens e Desvantagens do PHP. Disponível em: <<https://www.inforlogia.com/vantagens-e-desvantagens-do-php/>> Acesso em 26 nov. 2018.

BANCO DE SANGUE DA UNIVERSIDADE FEDERAL DE SANTA CATARINA (Brasil). **Perguntas Frequentes.** 2017. Disponível em: <<http://bancodesangue.paginas.ufsc.br/perguntas-frequentes/>>. Acesso em: 18 jun. 2018.

BASSI FILHO, Dairton Luiz. **Experiências com desenvolvimento ágil.** 170 f. Dissertação (Mestrado em Ciência da Computação) - Instituto de Matemática e Estatística da Universidade de São Paulo. USP, São Paulo, 2008.

BAYLÃO, Andre Luis da Silva; OLIVEIRA, Victor Miranda de. **Impacto da Evolução Tecnológica na Gestão Empresarial.** Resende: Associação Educacional Dom Bosco, 2012. Disponível em: <<https://www.aedb.br/seget/arquivos/artigos15/14922205.pdf>>. Acesso em: 01 jun. 2018.

BOEHM, B. W; TURNER, R. **Balancing Agility and Discipline.** Boston; Addison Wesley, 2002.

BOOTSTRAP. **Sobre.** 2018. Disponível em: <<https://getbootstrap.com/docs/4.1/about/overview/>>. Acesso em: 17 jun. 2018.

BROWN, D., DAVIS, C. M., STANLICK, S., Struts 2 in Action, Manning, 2008

BRASIL. AGÊNCIA NACIONAL DE TELECOMUNICAÇÕES. **Brasil registra 233,35 milhões de linhas móveis em outubro sustentado pelas pequenas operadoras.** 2018. Disponível em: <<http://www.anatel.gov.br/dados/acessos-telefoniamovel>>. Acesso em: 12 mar. 2018.

_____. Constituição (2001). **Lei nº 10205**, de 21 de março de 2001. Brasília, 21 mar. 2001. Disponível em: <http://www.planalto.gov.br/ccivil_03/leis/leis_2001/110205.htm>. Acesso em: 20 mai. 2018.

_____. MINISTÉRIO DA SAÚDE. **Doação de Sangue:** Caminhos do Sangue do doador até

quem precisa. 2018. Disponível em: <<http://portalms.saude.gov.br/saude-de-a-z/doacao-de-sangue/caminhos-do-sangue>>. Acesso em: 11 nov. 2018.

_____. MINISTÉRIO DA SAÚDE. **Ministério da Saúde Lança Campanha para Incentivar Doação Regular de Sangue.** 2015. Disponível em: <<http://portalsaude.saude.gov.br/index.php/cidadao/principal/agencia-saude/28742-ministerio-da-saude-lanca-campanha-para-incentivar-doacao-regular-de-sangue>>. Acesso em: 12 fev. 2018.

_____. MINISTÉRIO DA SAÚDE. **Uma Doação Pode Salvar até Quatro Vidas.** 2017. Disponível em: <<http://www.brasil.gov.br/saude/2017/11/uma-doacao-de-sangue-pode-salvar-ate-quatro-vidas>>. Acesso em: 8 dez. 2017.

_____. SECRETÁRIA DA SAÚDE. **Requisitos básicos para doação de sangue.** 2018. Disponível em: <http://www.saude.sp.gov.br/resources/ses/perfil/gestor/aceso-rapido/grupo-de-sangue-componentes-e-derivados-hemorrede/doacao-de-sangue-informacoes-requisitos-e-cuidados/requisitos_basicos_doacao_de_sangue.pdf>. Acesso em: 15 jun. 2018.

_____. SENADO FEDERAL. **Aumentar o número de dias de folga para doação de sangue e em contrapartida reduzir o número de feriados.** 2016. Disponível em: <<https://www12.senado.leg.br/ecidadania/visualizacaoideia?id=45642>>. Acesso em: 04 fev. 2018.

CIRIBELI, João Paulo; PAIVA, Victor Hugo Pereira. **Redes e mídias sociais na internet: realidades e perspectivas de um mundo conectado.** 2011. Disponível em: <<http://fumec.br/revistas/mediacao/article/viewFile/509/504>>. Acesso em: 22 mar. 2018.

CORDEIRO, Fillipe. **Guardando Dados com SQLite.** 2013. Disponível em: <https://www.androidpro.com.br/blog/armazenamento-de-dados/sqlite/#O_que_e_o_SQLite>. Acesso em: 14 jun. 2018.

DIAS NETO, Arilo Cláudio. **Bancos de Dados Relacionais.** 2011. Disponível em: <<https://www.devmedia.com.br/bancos-de-dados-relacionais/20401>>. Acesso em: 14 jun. 2018.

ELMASRI, Ramez; NAVATHE, Shamkant B. (Ed.). **Fundamentals of Database Systems.** 4. ed. São Paulo: Pearson Education, 2006. Disponível em: <http://www.rclick.com.br/prime/BD/Sistema_de_banco_de_dados_Navathe.pdf>. Acesso em: 05 mai. 2018.

ESPOSITO D., Programming Microsoft ASP.NET MVC, Microsoft Press, 2010.

FIGUEIREDO, Eduardo. **Manifesto Ágil.** 2016. Disponível em: <http://homepages.dcc.ufmg.br/~figueiredo/disciplinas/aulas/metodos-ageis-manifesto_v02.pdf>. Acesso em: 14 jun. 2018.

FREEMAN, A., SANDERSON, S., ASP.NET MVC 3 Framework, 3.^a Ed., Apress, 2011.

FREEMAN, Steve; PRYCE, Nat. **Desenvolvimento de Software Orientado a Objetos, Guiado por Testes**. 1. Ed. Rio de Janeiro: Alta Books, 2012.

FONSECA, Fernando Josepetti. **PSI 2617 Inovação em Engenharia**. 2017. Disponível em: <https://edisciplinas.usp.br/pluginfile.php/4160716/mod_resource/content/2/PSI%202617%20MVP.pdf>. Acesso em: 21 set. 2018.

GALLOWAY, L., HAACK, P., WILSON, B., ALLEN, K., **Professional ASP.NET MVC 4**, Wiley, 2012.

GASPAROTTO, Henrique Machado. **Desenvolvimento multiplataforma com Xamarin**. 2015. Disponível em: <<https://www.devmedia.com.br/desenvolvimento-multiplataforma-com-xamarin/33467>>. Acesso em: 17 jun. 2018.

_____. **Xamarin, Ionic e Cordova: Conheça o que são e as principais diferenças**. 2017. Disponível em: <<https://www.devmedia.com.br/xamarin-ionic-e-cordova-conheca-o-que-sao-e-as-principais-diferencas/37690>>. Acesso em: 17 jun. 2018.

GINIGE, Athula; MURUGESAN, San. (2001). **Web engineering: An introduction**. Multimedia, IEEE. 8. Ed. p. 14 - 18.

GONÇALVES, Allan Jheyson Ramos. Desenvolvimento de Aplicativos Híbridos com o Ionic Framework. In: COMPUTAÇÃO, Sociedade Brasileira de. **Livro Anais: Artigos e Minicursos**. Teresina: Livro Eletrônico., 2017. p. 500-515.

GOOGLE. **Firestore Realtime Database**. 2018. Disponível em: <<https://firebase.google.com/docs/database/?hl=pt-br>>. Acesso em: 22 abr. 2018.

IANNI, Vinicius. **Introdução aos bancos de dados NoSQL**. 2012. Disponível em: <<https://www.devmedia.com.br/introducao-aos-bancos-de-dados-nosql/26044>>. Acesso em: 17 jun. 2018.

HIGA, Paulo. **O que é XAMPP e para que serve**. 2012. Disponível em: <<https://www.techtudo.com.br/dicas-e-tutoriais/noticia/2012/02/o-que-e-xampp-e-para-que-serve.html>>. Acesso em: 24 out. 2018.

INSTITUTO DE BIOCÊNCIAS DA UNIVERSIDADE DE SÃO PAULO (Brasil). **Sangue Fluido da Vida**. 2005. Disponível em: <<http://www.ib.usp.br/microgene/files/biblioteca-18-PDF.pdf>>. Acesso em: 15 out. 2017.

JOHNSON, R.; HOELLER, J; ARENDSSEN, A; RISBERG, T.; Sampaleanu, C., **Professional Java Development with the Spring Framework**, Wiley Publishing, Inc., 2005

KEMP, Simon. **Digital in 2018: World's Internet Users Pass the 4 Billion Mark**. 2018. Disponível em: <<https://wearesocial.com/blog/2018/01/global-digital-report-2018>>. Acesso em: 21 fev. 2018.

KOOMEN, Tim; POL, Martin. **Test Process Improvement: A Practical Step-by-Step Guide to Structured Testing**. Boston: Addison-wesley Longman Publishing Co., 1999.

KORTH, H.F. e SILBERSCHATZ, A.; Sistemas de Bancos de Dados, Makron Books, 2a. edição revisada, 1994.

LACERDA, José António Vieira. **Análise de Frameworks de Desenvolvimento Web para a Conversão de uma Aplicação Desktop.** 2013. Disponível em: <<http://repositorio.ipv.pt/handle/20.500.11960/1404>>. Acesso em: 17 jun. 2018.

MARTINS, C.; ANTÔNIO, A.; OLIVEIRA, C. A. **Os desafios para a mobilização de aplicações baseadas em plataforma Web.** Catalão: Enacomp, 2013. P. 294-300.

MELHOR, Saúde. **Doar sangue faz bem? Quais os riscos?** 2018. Disponível em: <<https://www.saudemelhor.com/doar-sangue-faz-bem-quais-riscos/>>. Acesso em: 24 abr. 2018.

MOTA, Rodrigo L. M.; LIMA, Pablo B. S.; ROMANO; Breno L. Um modelo para definição de metodologia de desenvolvimento de software baseado em pessoas. Departamento de Computação e Matemática - Instituto de Ciências Exatas - Universidade Federal de Itajubá (UNIFEI), Itajubá - MG, 2011. Disponível em: <<http://eati.info/eati/2011/anais/artigos/91321.pdf>>. Acesso em: 18 nov. 2017.

MYRES, G. F. The Art of Software Testing. Ed. John Wiley & Sons, Inc. New Jersey, 2004.

NIELSEN MEDIA RESEARCH. **Nielsen: 78% dos Anúncios Digitais no Brasil Conseguem Atingir o Público-alvo.** 2016. Disponível em: <<http://www.nielsen.com/br/pt/insights/news/2016/Nielsen-78-por-cento-dos-anuncios-digitais-no-Brasil-conseguem-atingir-o-publico-alvo.html>>. Acesso em: 26 fev. 2018.

OBJECTIVE SOLUTIONS. Testes Automatizados de Software. 2016. Disponível em: <<https://www.objective.com.br/testes-automatizados-de-software/>>. Acesso em: 14 dez. 2018.

OLIVEIRA, S. R. B; ROCHA, T. A.; VASCONCELOS, A. M. L. (2004). **Adequação de Processos para Fábricas de Software**, Anais do Simpósio Internacional de Melhoria de Processos de Software – SIMPROS, São Paulo.

OPAS: SUMINISTRO DE SANGRE PARA TRANSFUSIONES EM LOS PAÍSES DE LATINOAMÉRICA Y DEL CARIBE. Ed. 2014.

ORGANIZAÇÃO MUNDIAL DA SAÚDE. Organização das Nações Unidas. **More voluntary blood donations essential: On World Blood Donor Day, WHO calls for more people to be heroes – donating blood regularly.** 2012. Disponível em: <http://www.who.int/mediacentre/news/releases/2012/blood_donation_20120614/en/>. Acesso em: 21 set. 2017.

PALERMO, J et al. ASP.NET MVC 4 in Action Manning, 2012;

PEREIRA, Jefferson Rodrigues et al. **Doar ou não doar, eis a questão: uma análise dos fatores críticos da doação de sangue.** 2016. Disponível em: <https://www.scielo.org/scielo.php?script=sci_arttext&pid=S1413-81232016000802475&lang=pt#>. Acesso em: 05 fev. 2018.

PEREIRA, Thiago Silveira; BASTOS, João Luiz. **Doação de sangue entre estudantes de medicina da Universidade Federal de Santa Catarina.** 2009. Disponível em: <<http://www.acm.org.br/acm/revista/pdf/artigos/743.pdf>>. Acesso em: 15 abr. 2018.

PIZZO, Vladimir Ribeiro. **Sua Saúde: Tecnologia digital facilita a vida de pacientes e médicos.** 2015. Disponível em: <<https://hospitalsiriolibanes.org.br/sua-saude/Paginas/tecnologia-digital-facilita-vida-pacientes-medicos.aspx>>. Acesso em: 12 out. 2018.

PRESSMAN, Roger S. **Engenharia de Software: Uma abordagem Profissional.** 7. Ed. Porto Alegre: Bookman, 2011.

PROCEDI, Lisandro. **Avaliação do framework Xamarin.Forms para desenvolvimento de aplicativos móveis multiplataforma, criando uma aplicação real.** 2016. Disponível em: <<http://www.lume.ufrgs.br/handle/10183/150981>>. Acesso em: 17 jun. 2018.

ROBERTO, João. **O que é Laravel? Porque usá-lo.** 2017. Disponível em: <<https://medium.com/joaorobertopb/o-que-é-laravel-porque-usá-lo-955c95d2453d>>. Acesso em: 12 fev. 2018.

ROCHA, Fabio Gomes. **Introdução ao desenvolvimento guiado por teste (TDD) com JUnit.** 2012. Disponível em: <<https://www.devmedia.com.br/introducao-ao-desenvolvimento-guiado-por-teste-tdd-com-junit/26559>>. Acesso em: 17 jun. 2018.

SILVA, Kéllyson Gonçalves da; CARVALHO, William Chaves de Souza. **Análise do paradigma híbrido na indústria de software.** 2014. Disponível em: <<http://perquirere.unipam.edu.br/documents/23456/612187/Análise+do+paradigma+híbrido+na+indústria+de+software.pdf>>. Acesso em: 15 abr. 2018.

SILVA, Rafael Mariano Gislon da; KUPEK, Emil; PERES, Karen Glazer. **Prevalência de doação de sangue e fatores associados em Florianópolis, Sul do Brasil:** estudo de base populacional. 2013. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0102-311X2013001000017>. Acesso em: 20 mar. 2018.

SILVEIRA, Paulo et al. **Introdução à arquitetura e design de software: uma visão sobre a plataforma Java.** Rio de Janeiro: Campus, 2012.

SOARES, Michel dos Santos. **Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software.** 2004. Disponível em: <<http://infocomp.dcc.ufla.br/index.php/INFOCOMP/article/view/68>>. Acesso em: 17 jun. 2018.

SOMMERVILLE, Ian. **Engenharia de Software.** 9. Ed. São Paulo: Pearson Prentice Hall, 2011.

SUTHERLAND, Meet Jeff; SCHWABER, Meet Ken. **The Scrum Guide.** 2017. Disponível em: <<https://www.scrumguides.org/scrum-guide.html>>. Acesso em: 15 dez. 2017.

TECHOPEDIA. Backend as a Service: O que significa Backend as a Service. 2018. Disponível em: <<https://www.techopedia.com/definition/29428/backend-as-a-service-baas>>. Acesso em: 17 dez. 2018.

THE PHP GROUP, História do PHP. Disponível em: <https://secure.php.net/manual/pt_BR/history.php.php>. Acesso em 27 out. 2018.

VENKI. **Processo de inovação tecnológica e suas 8 etapas.** 2017. Disponível em: <<http://www.venki.com.br/blog/processo-de-inovacao/>>. Acesso em: 05 abr. 2018.

WALLS, C., Spring in Action, 3.^a Ed., Manning, 2011

WAZLAWICK, Raul S. Metodologia de Pesquisa para Ciência da Computação. 1. ed. Rio de Janeiro: Elsevier, 2009.

ZEMEL, Tércio. **Web Design Responsivo:** páginas adaptáveis para todos os dispositivos. São Paulo: Casa do Código, 2013.