

UTFPR-UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Bacharelado em Engenharia de Software - 6º Período

DISCIPLINA: *Oficina de Integração 2*

PROFESSOR: *Rogerio Santos Pozza*

Documento de Projeto de Software

ArqHub

Felipe Seolin Bento

Rafael Fernandes Marques

Elian Marcos Veloso

Wirlley Oliveira Delfino

Cornélio Procópio

2019

1 Introdução	2
Contexto	3
Justificativa	3
Proposta	4
Organização do Documento	5
2 Descrição Geral do Sistema	5
Objetivos (Gerais e Específicos)	5
Limites e Restrições	6
Descrição dos Usuários do Sistema	6
3 Desenvolvimento do Projeto	8
Tecnologias e ferramentas	8
Metodologia de desenvolvimento	11
Cronograma previsto	15
4 Requisitos do Sistema	16
Requisitos Funcionais	16
Requisitos Não-funcionais	17
Diagramas de Casos de Uso	18
5 Análise do Sistema	19
Modelo do Banco de Dados	19
Diagrama de Classes	20
Diagrama de Atividades	21
6 Implementação	23
Protótipos de Telas	23
Descrição do código	26
7 Considerações Finais	28
8 Bibliografia	29

1 Introdução

- **Contexto**

Construir uma nova casa e achar o projeto correto pode ser uma das principais decisões que uma pessoa pode tomar em sua vida. A escolha de um projeto adequado oferece alguns passos iniciais importantes para garantir que está se tomando a decisão correta. O projeto auxilia no sentido de que tudo é analisado harmoniosamente, projetado tendo em vista a melhor forma possível de aproveitar os espaços, criar acessibilidade e facilidade de uso aos ambientes, assim como ter uma aparência elegante e apresentável da casa.

Adquirir um imóvel no Brasil requer de um caro investimento e é algo que necessita de planejamento e estabilidade financeira. Falando-se do processo para adquirir projetos residenciais, um dos desafios enfrentados é a burocracia, serviços de baixa qualidade, alto custo e pouca estruturação tecnológica. Tudo isso torna esse contexto ainda mais desafiador. Esse momento é a oportunidade que as novas tecnologias têm para ganhar o mercado e proporcionar aos brasileiros uma nova experiência e novas oportunidades.

No âmbito de projetos arquitetônicos, as plantas baixas são um elemento fundamental de marketing. Auxiliam potenciais compradores a entender o layout da casa, e concluir se é o que eles realmente estão procurando. Pode-se conseguir tudo isso negociando com um arquiteto que fará o projeto da casa usando as melhores técnicas construtivas e elaborando um projeto sobre medida para atender às necessidades de moradia do cliente. O projeto da construção de uma casa é uma das partes mais importantes, e pode ser muito caro dependendo do caso. Fazer uma casa sem um projeto de qualidade poderá resultar em numerosos problemas no resultado final da obra.

Atualmente, ao construir um imóvel geralmente procura-se profissionais que serão responsáveis pela elaboração do projeto, que na maioria dos casos são arquitetos ou engenheiros. O arquiteto é encarregado pela elaboração do projeto arquitetônico e tem uma compreensão um pouco mais completa sobre a melhor maneira de usufruir o espaço. O engenheiro é exigido para o projeto estrutural, hidráulico e elétrico da casa. O investimento em um empreendimento completo e personalizado varia de acordo com o grau de complexidade do serviço e a qualificação do profissional contratado. Entretanto existem alguns casos em que o indivíduo não pode contratar um arquiteto por questões diversas ou por algum empecilho. Em muitas cidades não se tem oportunidade de acesso a um arquiteto, seja pelo custo ou até pela inexistência de oferta do ofício. É preciso também levar em consideração as famílias de menor poder aquisitivo e de cidades remotas.

- **Justificativa**

As vendas online permitem que os projetos sejam mais produtivos e lucrativos. O projetista se concentraria nos projetos que realmente exigem um tratamento exclusivo. Podemos usar como exemplo o modelo de negócio usado para venda de automóveis. Escolhe-se um modelo básico e então é adicionado as características desejadas. Pode-se incluir aros novos, assentos de couro, etc. Não há razão para “redesenhar” o carro.

Para muitas projetistas, o grande lucro de um projeto seria muito mais atraente do que se envolver com a venda de um produto on-line de baixo custo que requer um volume de vendas maior. Porém, um projeto arquitetônico popular pode ser revendido um número ilimitado de vezes em um site especializado. Ou você pode configurar um site básico do Mercado Livre e vendê-los diretamente para você, porém é mais difícil obter resultados nos mecanismos de pesquisa quando comparado a um site especializado no domínio.

Além disso, com um site pode-se começar a comercializar para todos os públicos. Mais pessoas teriam acesso a projetos inovadores e sustentáveis. Ao mesmo tempo, os projetistas teriam mais tempo e recursos para inovar e desenvolver ainda mais as práticas de construção de casas sustentáveis. Ao permitir que múltiplos projetistas exponham seu projeto, oferecemos aos compradores uma variedade de opções que eles não poderiam obter de forma tão fácil. Ao trazer, em grande número, vendedores e compradores, criamos um efeito de rede que leva a uma comunidade cada vez maior.

- **Proposta**

O sistema proposto tem como objetivo transformar a maneira que o brasileiro compra e vende projetos arquitetônicos. Iremos tornar a experiência de comprar e vender projetos mais ágil, transparente e barata. A plataforma terá projetos arquitetônicos prontos e disponibilizará eles na internet para a venda. Esses projetos podem ser vendidos a pessoas que nunca usariam seus serviços porque não moram na sua região. Isso permite alcançar uma soma maior de potenciais compradores.

Os vendedores podem gerenciar seus projetos, adicionando novos, atualizando ou removendo conforme necessário. Os projetos serão expostos com representação visual através de sua planta baixa, fotos profissionais ou renderizações, elevações frontais, laterais e traseiras e informações completas sobre o projeto. Existem diversos recursos proporcionados pela computação que permitem um efeito ainda mais impressionante para os clientes, como a realidade virtual e a possibilidade de fazer uma maquete eletrônica 3D.

Ainda há oportunidades para o projetista aumentar seu lucro através da personalização de projetos existentes. Os requisitos de personalização mais básicos são o ajuste para o terreno e leis locais. Uma personalização mais avançada pode incluir diferenças em acabamentos ou um quarto maior por exemplo. Desta forma o projetista aproveita o mesmo projeto várias vezes, sem ter que dedicar muitas horas para cada cliente.

A maior parte do valor obtido nas vendas irá para o projetista. O site ficaria com uma porcentagem para manter e desenvolver a plataforma, bem como para atrair novos compradores.

- **Organização do Documento**

No primeiro tópico é apresentada uma introdução geral ao sistema proposto. Apresenta-se o contexto atual, seguido pela justificativa e proposta de projeto. No segundo tópico é feita uma descrição mais detalhada do sistema e usuários, explicando seus objetivos, limites e restrições. No terceiro tópico é apresentado a metodologia e cronograma de desenvolvimento, e as ferramentas que serão utilizadas. No quarto tópico é feita a especificação de requisitos do projeto, apresentando os requisitos funcionais e não-funcionais do sistema e diagramas de caso de uso. No quinto tópico é apresentada a análise do sistema com base no que já foi feito anteriormente no tópico 4.

2 Descrição Geral do Sistema

- Objetivos (Gerais e Específicos)**

O objetivo geral do projeto é fornecer uma interface e um meio de comunicação para projetistas, para que estes possam mostrar os seus trabalhos realizados para todos que se interessarem, como possíveis compradores.

Dessa forma temos como objetivos específicos:

- Facilitar a comunicação entre comprador e vendedor.
- Divulgar trabalhos já realizados.
- Possibilidade de reaproveitar completamente ou parcialmente trabalhos anteriores.
- Aumentar o número de clientes para um projetista.
- Gerenciamento de projetos arquitetônicos.

- Limites e Restrições**

Limites

- O sistema não implementará o pagamento, pois isto nos deixaria responsáveis por várias questões legais que possam acontecer.
- De início, o sistema não tem planos para construir um chat, sendo assim as mensagens se dariam por envio de e-mails. Isto foi decidido

Restrições

- O software será feito para os navegadores: Google Chrome, Mozilla Firefox, Safari e Opera em suas versões mais recentes, sendo assim as versões mais antigas e outros navegadores podem apresentar instabilidades.
- O software terá como foco o uso em desktop.

- **Descrição dos Usuários do Sistema**

Projetista - Este usuário é responsável por adicionar projetos, fornecer informações de contato, fornecer informações profissionais

Visitante - Este usuário não precisa de login para acessar o sistema ou interagir com este, sua função principal é observar e ser espectador das informações fornecidas por um projetista. Sendo que o usuário pode se interessar pelo projeto e/ou projetista e entrar em contato com este por meio de mensagens.

3 Desenvolvimento do Projeto

• **Tecnologias e ferramentas**

Nesta seção são listadas todas as tecnologias e ferramentas que utilizamos no decorrer do projeto, sendo que cada uma se adequa a uma categoria.

Gerenciamento de Projeto:

- **Google Drive:** é um serviço de armazenamento e sincronização de arquivos em nuvem, que abriga várias ferramentas como: o Google Docs, Google Slides, Google Sheets e um leque de aplicações.
- **Taiga:** é uma plataforma de gerenciamento de projetos para desenvolvedores e designers ágeis e gerentes de projeto que desejam uma ferramenta eficiente. Devido às limitações da plataforma, no caso o máximo de três integrantes por projeto, foi decidido utilizar o **Github Projects** como alternativa, pois oferece uma experiência similar e satisfatória à equipe.

Edição de Documento:

- **Google Docs:** é um editor de texto online, onde o documento fica armazenado na nuvem do Google Drive, sua maior proposta é o compartilhamento e contribuição em um documento.
- **Google Sheets:** é um editor de planilhas online, no qual é possível compartilhar e colaborar em grupo para um projeto. Traz consigo várias funcionalidades, como o salvamento automático em nuvem e Insights em tempo real, conforme a interação do usuário na plataforma.
- **Google Slides:** é um editor de slides e exibidor de apresentações online. Assim como os outros produtos Google, traz salvamento automático em nuvem e permite a colaboração e compartilhamento em grupos.
- **Word 2018 | versão 1802:** é um produto Microsoft, já consagrado no mercado. Tanto que quando se fala de editor texto, já é diretamente associado com este software. Atualmente, possui várias funcionalidades, como trabalho em equipe, suporte online, integração com o OneDrive e entre outros.

Prototipação:

- **Quant-ux:** é um software online para prototipação. Além de possuir a possibilidade de carregar imagens, também se pode construir as interações de uma tela para a outra através do controle de fluxo. O software traz a oportunidade de se fazer testes de usabilidade, por meio de tarefas entregues aos usuários e aprender com os dados coletados, como mapas de calor.

IDEs e editores:

- **Visual Studio Code | versão 1.26:** é um editor de código-fonte desenvolvido pela Microsoft e com colaboradores, pois o código é aberto (*open source*). Possui suporte à depuração, versionamento (git), IntelliSense, que é um auto preenchimento inteligente, já trazendo consigo a documentação e ajuda a programação a ficar mais intuitiva, entre tantas funcionalidades.
- **PhpStorm | versão 2019.1:** O PhpStorm é uma IDE da Jetbrains que dá suporte a linguagens web, principalmente o PHP e os seus Frameworks, por isso trabalha bem com Symfony, Laravel, Zendo, CakePHP e outros.

Versionamento:

- **Git | versão 2.18.0:** é um sistema de versionamento e controle de versões distribuído de arquivos. Através do software, pode-se desenvolver em conjunto e de maneira paralela, até no mesmo arquivo, sem se preocupar com a possibilidade de algo ser sobreescrito ou apagado.

Repositório:

- **GitHub:** é um serviço de hospedagem de projetos, servindo como um repositório remoto. É capaz de gerenciar e controlar versões de forma distribuída. Oferece suporte à implantação (deploy), colaboração, gerenciamento de projeto e outros.

Linguagens

- **SQL | SQL:2008:** *Structured Query Language*, ou Linguagem de Consulta Estruturada em tradução para o português, é uma linguagem amplamente utilizada, sendo inspirada pela álgebra relacional. Está presente em banco de

dados relacionais, tanto para consulta como para a administração e modificação da estrutura do banco.

- **Laravel | 5.7:** Um framework PHP que oferece várias ferramentas que auxiliam na programação.

Sistema Gerenciador de Banco de Dados (SGBD) e Modelagem

- **PostgreSQL | versão 10.5:** é um sistema gerenciador de banco de dados objeto-relacional, que usa da linguagem SQL estendida e combinada com várias outras funcionalidades, além de ser um projeto de código aberto (opensource).
- **PG Admin 4 | versão 3.2:** é uma ferramenta opensource de administração e desenvolvimento para PostgreSQL. Usa de elementos de interface para melhor experiência do usuário. Trata de questões como multi-user e web deployment (implantação) e traz dashboards.
- **Vertabelo:** é uma ferramenta online para modelagem de banco de dados relacionais, permite gerar o código sql por meio de engenharia reversa e também modificar o banco a partir de um script. Além disso, traz consigo a possibilidade de compartilhamento e colaboração em um projeto.
- **brModelo | versão 3.2:** ferramenta de código aberto para a diagramação de modelo de banco de dados relacionais, permitindo a visualização em modelo lógico e modelo conceitual.
- **Astah UML 7.2.0/1ff236 | versão 37:** é uma ferramenta UML para modelagem de diagramas. A ferramenta possui várias funcionalidades, entre essas: comparação de diagramas, modelagem colaborativa, integração com códigos-fonte, desenvolvimento em equipe, modificações no ambiente por plug-ins e APIs, entre tantos outros.

- **Metodologia de desenvolvimento**

A metodologia que será utilizada é o FDD - “Feature Driven Development”. O motivo desta escolha, está equiparado a algumas escolhas anteriores, na qual esbarramos em alguns obstáculos, estas escolhas anteriores foram, incremental, Scrum, XP. Porém, em algum ponto da metodologia havia algum obstáculo, até que chegamos ao FDD, que nada mais é, do que a junção destas três metodologias mencionadas, e realizado uma adaptação das mesmas.

De acordo com DEVMedia, o FDD busca o desenvolvimento por funcionalidade, ou seja, por um requisito funcional do sistema. É prático para o trabalho com projetos iniciais ou projetos com codificações existentes. Apesar de ter algumas diferenças entre o FDD e o XP, é possível utilizar as melhores práticas de cada metodologia. O FDD atua muito bem em conjunto com o Scrum, pois o Scrum atua no foco do gerenciamento do projeto e o FDD atua no processo de desenvolvimento.

O FDD possui cinco processos básicos:

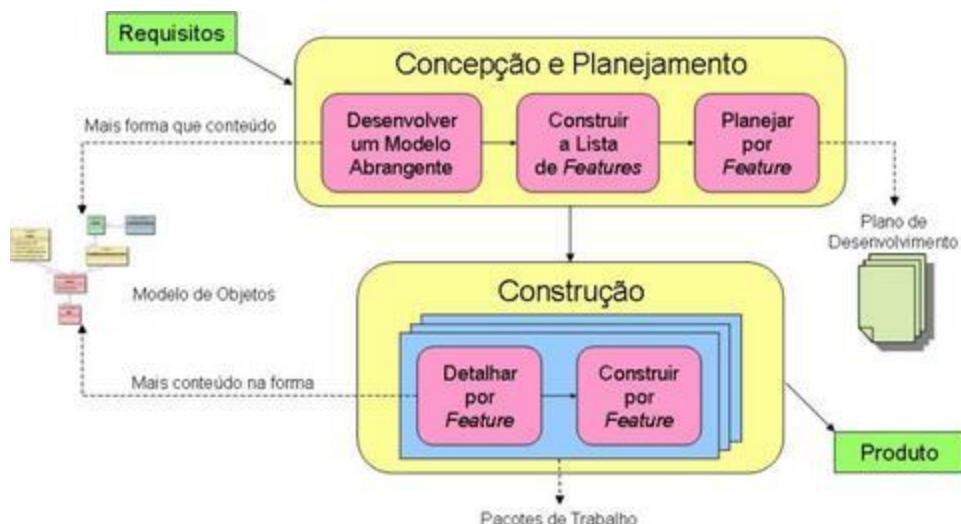


Figura 01 : Processos do FDD

De acordo com a figura 1, abaixo serão especificados cada processo e suas etapas, segundo Scott W. Ambler :

Desenvolvimento de modelo abrangente (Análise orientada por objetos)

Critério de entrada: os especialistas no domínio de negócio, os programadores líderes e o arquiteto foram selecionados.

Formar a equipe de modelagem: composta de membros permanentes da área do domínio do negócio e de desenvolvimento

Estudo dirigido sobre o domínio: especialista no domínio do negócio apresenta uma visão geral do domínio será modelada

Estudar a documentação: a equipe estuda os documentos de referência ou de requisitos disponíveis

Desenvolver o modelo: Selecionar um modelo proposto ou compor um modelo de acordo com a combinação de ideias da equipe.

Refinar o modelo de projeto abrangente: frequentemente o modelo é atualizado, de acordo com a necessidade de tal iteração do projeto.

Verificação: realiza-se uma autoavaliação ou uma avaliação interna através da participação ativa dos especialistas no domínio.

Construção de lista de funcionalidades (Decomposição funcional)

Critério de entrada: Os especialistas no domínio de negócios, os programadores líderes e o arquiteto foram selecionados.

Formar a equipe da lista de funcionalidades: é composta por programadores-líderes da equipe de modelagem do processo 1.

Construir a lista de funcionalidades: identificar o conjunto de funcionalidade usando o conhecimento adquirido no processo 1. Exemplo: pesquisar sala, adicionar usuário, etc.

Verificação: Realiza-se uma autoavaliação ou uma avaliação interna através da participação ativa dos membros da equipe de modelagem. Quando necessária, uma avaliação pode ser feita pedindo-se aos especialistas no domínio do negócio da equipe de modelagem ou ao negócio (usuários) que confirmem ou esclareçam as questões que afetam a lista de funcionalidades.

Critério de saída: uma lista de áreas; para cada área, uma lista de atividades de negócio dentro daquela área; para cada passo da atividade de negócio, uma funcionalidade que satisfaça ao passo.

Planejar por funcionalidade (Planejamento incremental)

Critério de entrada: o processo construir a lista de funcionalidades foi completado.

Formar a equipe de planejamento: composta pelo gerente de desenvolvimento e programadores líderes.

Determinar sequência de desenvolvimento (cronograma e requisitos): A equipe de planejamento deve atribuir uma data (mês e ano apenas) para o término de cada atividade de negócio. A identificação da atividade de negócio e a data de término (e dessa forma a sequência de desenvolvimento) é baseada em:

- Dependências entre as funcionalidades em termos de classes envolvidas;
- Distribuição da carga de trabalho entre os proprietários das classes;
- Complexidade das funcionalidades a serem implementadas;
- Adiantamento das atividades de negócio de alto risco ou complexidade;
- Consideração de qualquer marco externo (visível) do projeto, como versões “beta”, demonstrações,

pontos de verificação e “todos os produtos” que satisfaçam tais marcos.

Critérios de saída: O resultado do processo é o plano de desenvolvimento, consistindo em:

- Atividades de negócio com datas de término (mês e ano);
- Programadores líderes atribuídos a atividades de negócio;
- Áreas com datas de término (mês e ano), derivadas da data do último término de suas respectivas

atividades de negócio;

- Lista das classes e seus respectivos desenvolvedores proprietários (a lista de proprietários de classes).

Detalhe por funcionalidade (Desenho orientado a objetos);

Critérios de entrada: o processo de planejamento foi completado.

Desenvolver diagramas: produzir o pacote de projeto (design) para a funcionalidade (desenvolver os diagramas).

Escrever prefácios de classes e métodos: o proprietário das classes escreve os prefácios de classe e métodos para cada item definido pela funcionalidade e pelo(s) diagrama(s) de sequência.

Isto inclui tipos de parâmetros, tipos de retorno, exceções e mensagens. Uma vez que cada desenvolvedor completou sua tarefa, o programador líder gera a documentação da API usando <sua ferramenta> e a submete para publicação na intranet do projeto

Critérios de saída: O resultado do processo é um Pacote de Projeto (Design) inspecionado com sucesso. O pacote de projeto consiste em:

- Uma capa com comentários, que completa e descreve o pacote de projeto de tal forma a ser suficiente para futuros revisores;
- Os requisitos referenciados (se houver) na forma de documentos e de todos os memorandos de

confirmação relacionados, e documentação de apoio;

- O(s) diagrama(s) de sequência;
- Alternativas de projeto (design) (se houver);
- O modelo de objetos com classes, métodos e atributos novos/atualizados;
- A saída gerada pela <sua ferramenta> para os prefácios de classes e métodos, criados ou modificados por esse projeto (design);
- Lista de tarefas e agendamentos para itens de ação nas classes afetadas para cada membro da equipe.

Construção por funcionalidade (Programação e teste orientado a objetos).

Critério de entrada: o processo detalhar por funcionalidade foi completado.

Implementar classes e métodos: Os proprietários de classes implementam os itens necessários para satisfazer aos requisitos de suas classes para esta funcionalidade.

Teste de unidade: Os proprietários de classes testam seus códigos para certificar que todos os requisitos de suas classes foram satisfeitos. O programador líder determina quais testes de unidade no nível da equipe de funcionalidades são necessários (se houver). Isto é, se algum teste envolvendo as classes desenvolvidas para esta funcionalidade é exigido.

Promover à versão atual: As classes somente podem ser promovidas para a versão atual (build) após uma inspeção de código com sucesso. O programador líder, monitora as classes sendo promovidas individualmente, através de informações dos desenvolvedores, e é o ponto de integração para a funcionalidade inteira.

Critérios de saída: o resultado do processo é:

- Classes e/ou métodos que passaram na inspeção de código com sucesso;
- Classes que foram promovidas à versão atual (build);
- O término de uma função com valor para o cliente (funcionalidade).

A metodologia FDD, contém alguns papéis que ficam assim definidos:

Gerente de Projeto: é o líder administrativo e financeiro do projeto: tem a palavra final no que se trata de escopo, cronograma e recursos do projeto.

Gerente de Desenvolvimento: é o líder nas atividades diárias do desenvolvimento. É responsável por resolver qualquer tipo de conflito que venha a ocorrer dentro da equipe.

Arquiteto-chefe: é responsável pela modelagem do projeto. Deve conduzir as sessões de modelagem de forma que a equipe de desenvolvedores possam contribuir na construção do sistema.

Programadores-chefe: é o líder dos pequenos times de análise, modelagem e desenvolvimento de novas funcionalidades.

Proprietários de código/classe (Desenvolvedores): os “Donos” de Classe trabalham orientados pelo programador Chefe executando as tarefas de modelar, codificar, testar e documentar.

Especialistas do Domínio (negócio): os Especialistas do domínio podem ser usuários, clientes, patrocinadores ou analistas de negócio. Sua responsabilidade é de adquirir e transmitir informações a respeito do funcionamento dos requisitos do sistema.

Em relação ao nosso projeto, definimos os papéis de cada integrante do grupo, da seguinte forma mencionada abaixo:

	Gerente de Projeto	Gerente de Desenvolvimento	Arquiteto	Programadores-chefe	Proprietários	Especialistas do Domínio
Elian					X	
Felipe		X		X	X	
Rafael	X				X	X
Wirlley			X	X	X	

Tabela 1: Divisão de papéis na metodologia FDD.

- **Cronograma previsto**

Cronograma feito com base na metodologia FDD, e seus processos. Também pensando nas entregas necessárias da matéria:

Processos	Atividade(s)	Responsável(is)	Data
1ª Desenvolver um modelo abrangente	Critérios de entrada, formar a equipe de modelagem, estudo dirigido sobre o domínio, estudar a documentação, desenvolver o modelo, refinar o modelo de projeto abrangente, verificação/auto-avaliação	Elian Felipe Rafael Wirlley	12/03/2019- 15/03/2019
2ª Construir a Lista de Funcionalidades	Critérios de entrada, formar a equipe da lista de funcionalidades, construir a lista de funcionalidades, verificação/auto-avaliação, critérios de saída.	Elian Felipe Rafael Wirlley	16/03/2019- 20/03/2019
3ª Planejar por funcionalidade	Critérios de entrada, formar a equipe de planejamento, determinar sequência de desenvolvimento (cronograma e requisitos)	Elian Felipe Rafael Wirlley	21/03/2019- 24/03/2019
4ª Detalhar por funcionalidade	Critérios de entrada, desenvolver diagramas, escrever prefácios de classes e métodos, critérios de saída.	Elian Felipe Rafael Wirlley	25/03/2019- 09/04/;2019
5ª Construir por funcionalidade	Critério de entrada, implementar classes e métodos, teste de unidade, promover à versão atual, critérios de saída.	Elian Felipe Rafael Wirlley	10/04/2019- 04/06/2019

Tabela 2: Cronograma.

4 Requisitos do Sistema

• Requisitos Funcionais

Abaixo seguem os requisitos funcionais que o sistema deverá ter, ou seja, o quê, como e com quem o software deve ser executado. Cada requisito descreve uma funcionalidade do software, assim como alguma restrição para que sua integridade e eficiência na execução permaneçam intactos.

Cada funcionalidade seguirá conforme cada prioridade a seguir:

Essencial – deve ser implementado para que o sistema funcione.

Importante – sem este requisito o sistema pode funcionar, mas não da maneira esperada.

Desejável – este tipo de requisito não compromete o funcionamento do sistema.

ID	Funcionalidade	Prioridade	Relacionados
RF01	O sistema deve permitir que o projetista faça upload de seu portfólio de projetos.	Essencial	
RF02	O sistema deve permitir o cadastramento de usuário.	Essencial	
RF03	O sistema deve permitir recuperação de acesso.	Essencial	
RF04	O sistema deve permitir que um usuário visitante filtre todas os projetos conforme seu interesse.	Essencial	RF08, RF09, RF10
RF05	O sistema deve apresentar uma área específica explicando como é feita a entrega do projeto após a compra.	Desejável	
RF06	O projetista deve ter acesso a funções de adicionar, editar, excluir e listar seus projetos.	Essencial	RF01
RF07	O visitante deve ser capaz de enviar uma mensagem, do tipo	Essencial	

	e-mail, para solicitar o orçamento do projetista.		
RF08	O sistema deve separar os projetos por categorias.	Importante	RF04
RF09	O sistema deve exibir ao usuário a quantidade de cada tipo de cômodos do imóvel.	Desejável	RF04
RF10	O sistema deve exibir dados do tamanho do terreno referente aos projetos.	Essencial	RF04

Tabela 3: Requisitos Funcionais.

- Requisitos Não-funcionais**

Abaixo seguem os requisitos não-funcionais que o sistema deverá possuir. Cada requisito descreve uma característica do software, especificando como as funcionalidades deverão ser entregues ao usuário.

Cada requisito não-funcional seguirá conforme cada categoria a seguir:

ID	Requisito	Categoria
RNF01	O sistema deve ser capaz de funcionar perfeitamente nos navegadores: Google Chrome, Mozilla Firefox, Safari e Opera.	Portabilidade
RNF02	O sistema deve ser capaz de funcionar perfeitamente nas últimas versões dos navegadores anteriormente mencionados.	Portabilidade
RNF03	O sistema deverá ser implementado utilizando o framework PHP Laravel	Implementação
RNF04	Cada projeto deve possuir um código identificador único.	Usabilidade
RNF05	O componente de login deve ser implementado utilizando a API Google Sign-In.	Implementação

RNF06	O sistema deverá aceitar apenas imagens do tipo jpeg, jpg, png e svg;	Implementação
-------	---	---------------

Tabela 4: Requisitos Não-Funcionais

- **Diagramas de Casos de Uso**

Os casos de uso descritos abaixo, simulam a interação entre algum usuário com o sistema.

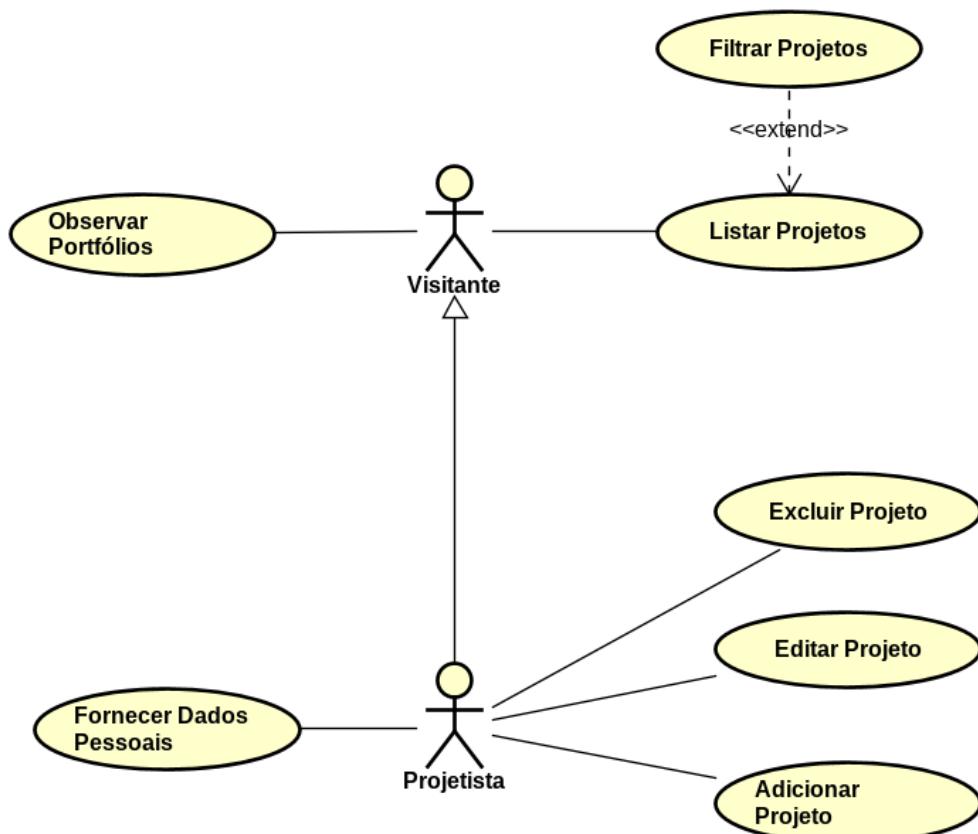


Figura 02: Diagrama de casos de uso - funções gerais dos usuários

5 Análise do Sistema

Nesta seção, está incluído o esquema relacional do banco de dados, diagrama de classes e diagramas de atividades do sistema.

- **Modelo do Banco de Dados**

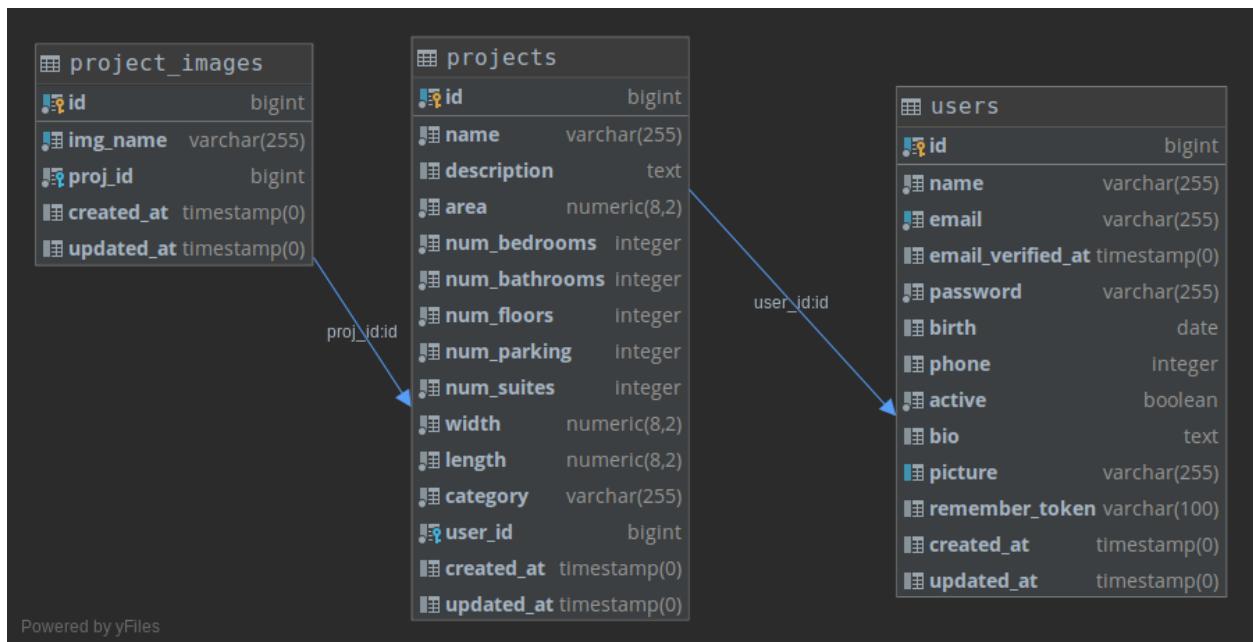


Figura 03: Esquema Relacional do Banco de Dados

- **Diagrama de Classes**

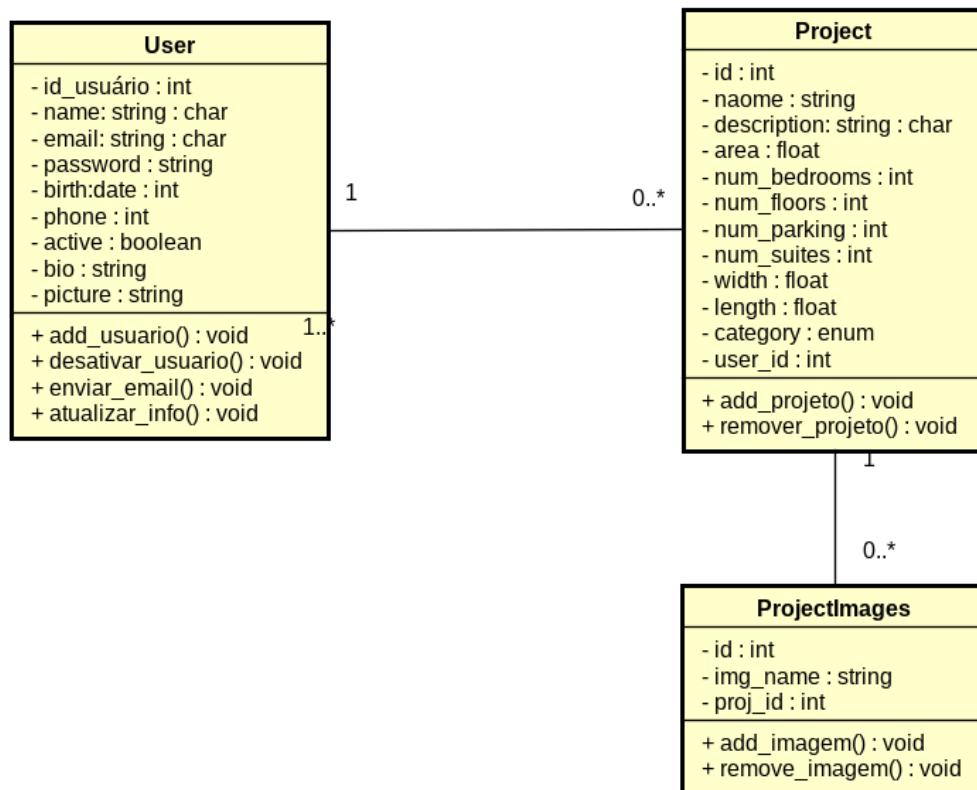


Figura 04: Diagrama de Classes.

- **Diagrama de Atividades**

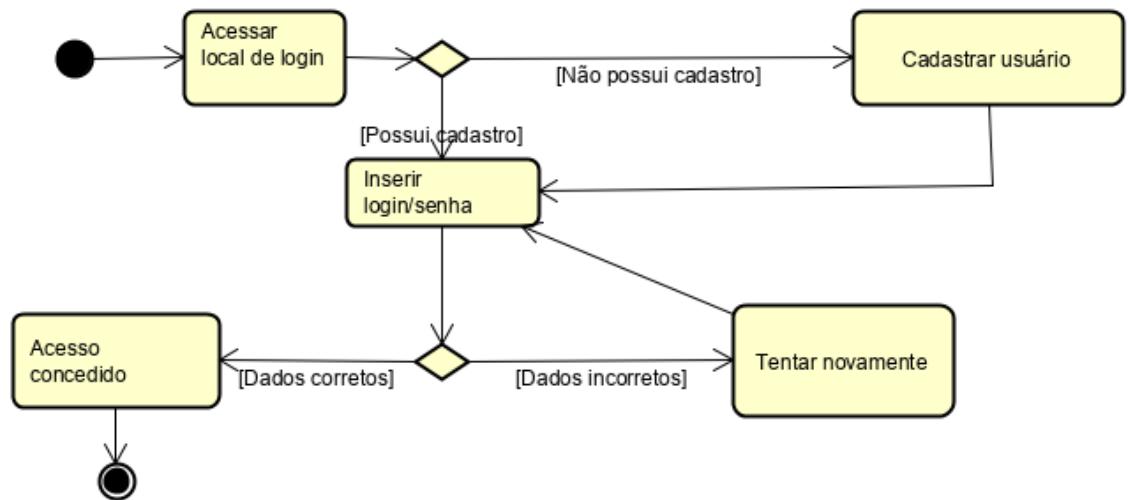


Figura 05: Diagrama de Atividades - login do usuário projetista

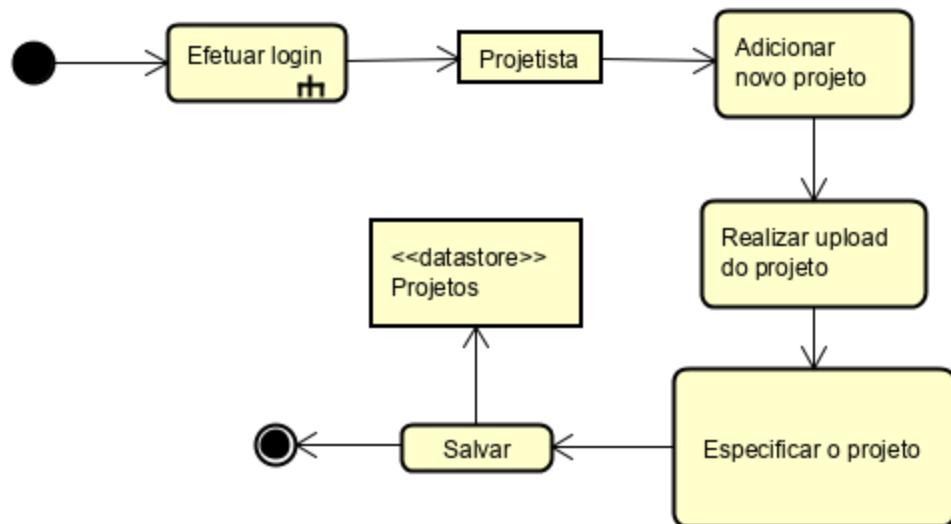


Figura 06: Diagrama de Atividades - Usuário projetista, adicionando um novo projeto

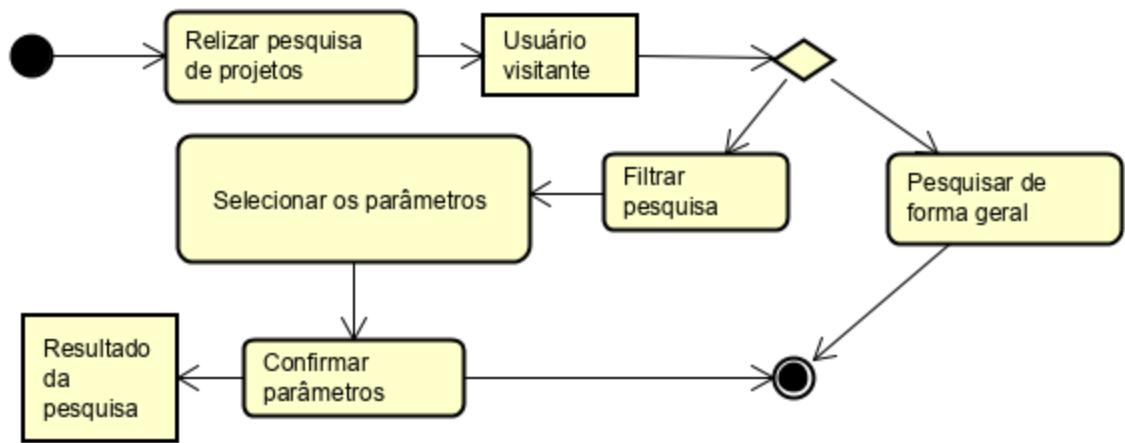


Figura 07: Diagrama de Atividades - Usuário visitante, realizando pesquisa de um projeto

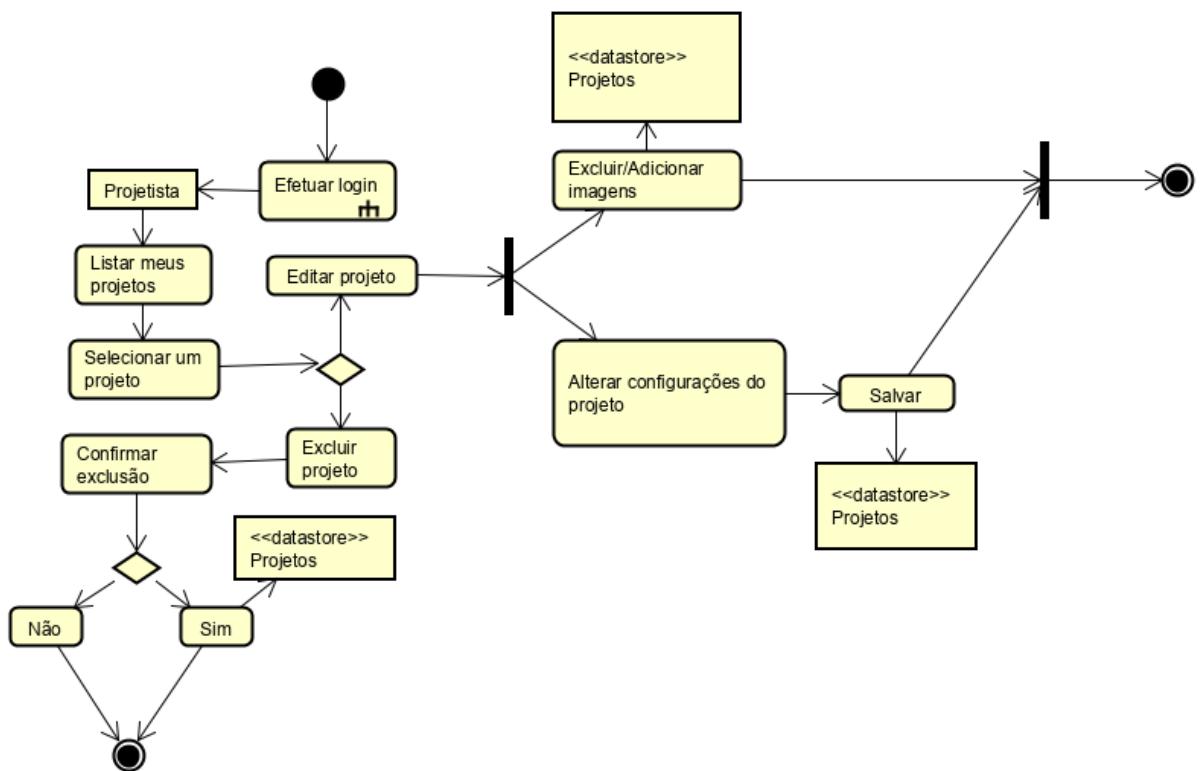


Figura 08: Diagrama de Atividades - Usuário projetista, listando/excluindo/alterando um projeto

6 Implementação

Protótipos de Telas



Figura 09: Protótipo de tela inicial do projeto

A Figura 09 faz referência a tela inicial, onde o usuário poderá encontrar um menu, com todas as opções de interação do sistema, além de uma barra de pesquisa para procurar projetos. Ao fundo teria um carrossel com as imagens de alguns projetos arquitetônicos.

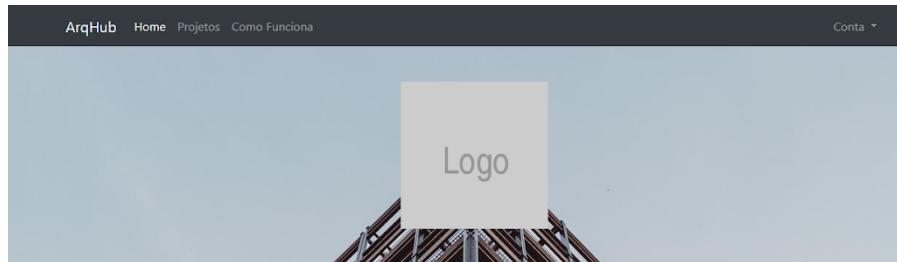
A screenshot of the project list on the initial prototype. At the top, there is a header "Os mais pesquisados" with some placeholder text below it. Below this, there are four cards, each representing a project. Each card has a small thumbnail image of a house, the project name "Nome Projeto", a detailed description in Portuguese, and a blue "Go somewhere" button at the bottom. The descriptions are in Portuguese and mention terms like "concreto", "área", "projeto", "projeto arquitetônico", and "projeto de casa".

Nome Projeto	Nome Projeto	Nome Projeto
Loreum ipsum dolor, sit amet consectetur adipiscing elit. Nullam, commodi maxime vitae corrupti provident laborum mesquunt etem defectus, quod, minus velit nobis sint a reprehenderit nulla at cum natione distinctio.	Loreum ipsum dolor sit amet consectetur adipiscing elit. Nullat eum esse, officia adipisci cum officiis nam tenetur, aliquid animi ullam totam sint cumque dolobus deleniti, suscipit voluptas. Facere, sunt tenetur lor.	Loreum ipsum dolor sit amet consectetur adipiscing elit. Nullat eum esse, officia adipisci cum officiis nam tenetur, aliquid animi ullam totam sint cumque dolobus deleniti, suscipit voluptas. Facere, sunt tenetur!
Go somewhere	Go somewhere	Go somewhere

Figura 10: Protótipo de listagem na tela inicial

A Figura 10 faz referência às listagens de projetos que estarão na tela inicial do projeto, de modo a destacar os últimos modelos que foram disponibilizados e também os mais acessados.

As Figuras 11, 12 e 13 mostram os protótipos da tela feitos já no início do desenvolvimento do sistema, utilizando *bootstrap*. O projeto final utilizou o mesmo *layout*, com apenas algumas pequenas modificações.



Section Heading

Lorem ipsum dolor sit amet, consectetur adipisicing elit.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Aliquid, suscipit, rerum quos facilis repellat architecto commodi officia atque nemo facere eum non illo voluptatem quae delectus odit vel itaque amet.

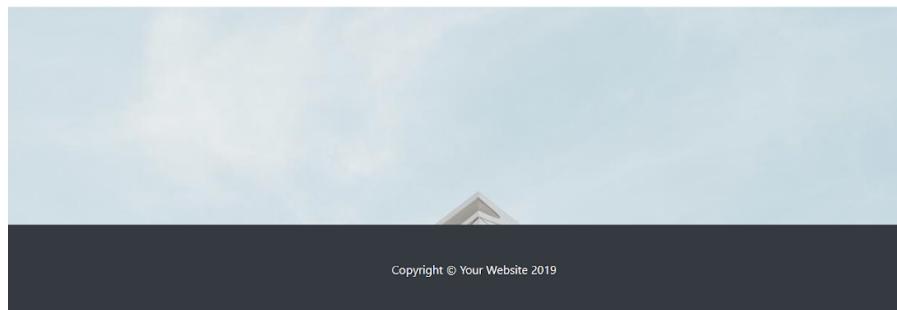


Figura 11: Protótipo atualizado de tela inicial

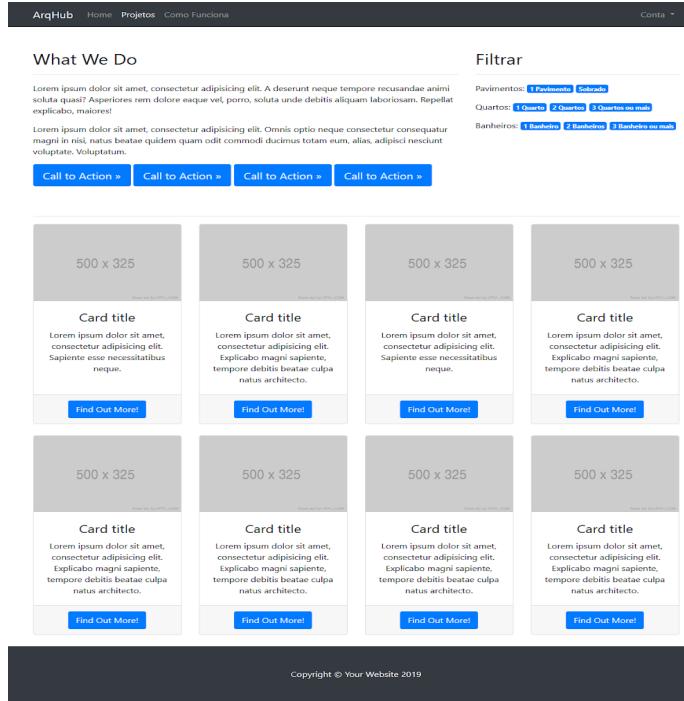


Figura 12: Protótipo atualizado de tela de projetos

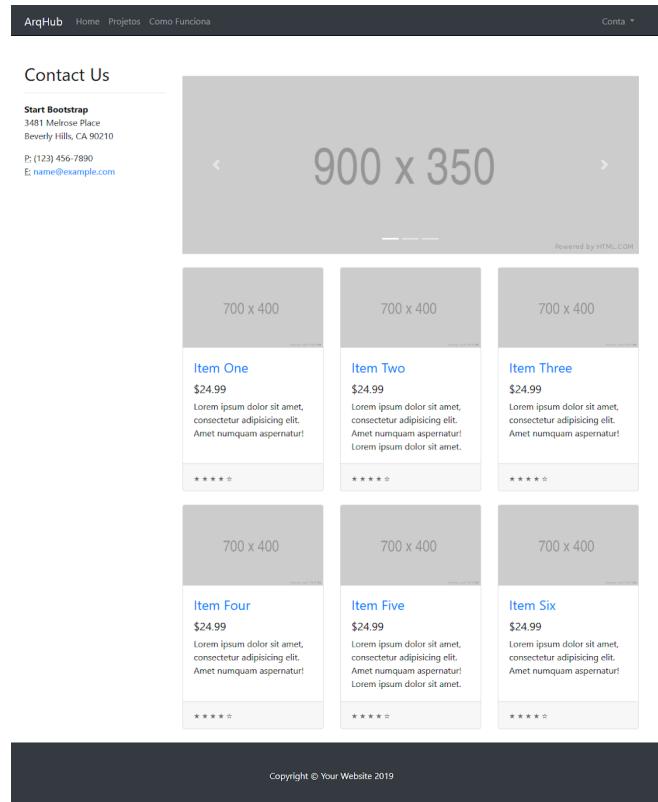


Figura 13: Protótipo atualizado de tela pessoal

Descrição do código

Muitas vezes, grandes aplicações a serem desenvolvidas possuem um alto grau de complexidade, com isso tem-se quase o uso indispensável de um padrão, que tem como objetivo facilitar o desenvolvimento. Esses padrões, tornam o desenvolvimento mais flexível, promovendo o reuso de componentes que já se mostraram consistentes em outras aplicações, e um fluxo padronizado de dados, que facilitar manutenção de todo o software.

Um padrão muito utilizado quando há constante interação do software com o usuário, é o padrão chamado MVC. Esse padrão possui três componentes, como pode ser observado na Figura 11, onde o componente View, é a parte visual com que o usuário interage, sendo o componente que é exibido na tela, já o componente Model representa toda a base de dados, que é refletida na view, e por fim, o Controller que define como a view deve interagir com a model.

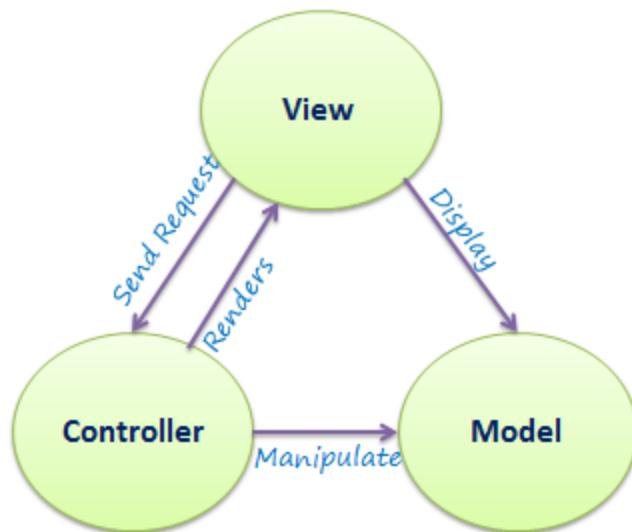


Figura 14: Representação do fluxo de dados no padrão MVC

Outro recurso utilizado visando diminuir a complexidade e facilitar o desenvolvimento de softwares, é a utilização de frameworks, que consiste em um conjunto de funcionalidades genéricas que já se mostraram sólidas e eficazes com o intuito de reduzir a quantidade de códigos escrito pelos desenvolvedores.

Neste trabalho, o framework utilizado é o Laravel, que utiliza o padrão MVC para o desenvolvimento da aplicação web. O componente View contém um mecanismo chamado Blade, onde o html e o php são compilados juntos, possibilitando a construção de toda parte visual responsável pela interação com o usuário. Para a criação dos componentes de Model e Controller, o Laravel disponibiliza o Artisan, que nada mais é que uma ferramenta de linha de comando, capaz de criar ambos os componentes com uma grande quantidade de funções prontas, que facilita a redução de codificação a ser escrita, como por exemplo, as ORMs para o banco de dados nos Models ou as middlewares para os Controllers. A arquitetura do projeto utilizando Laravel pode ser vista na imagem 15.

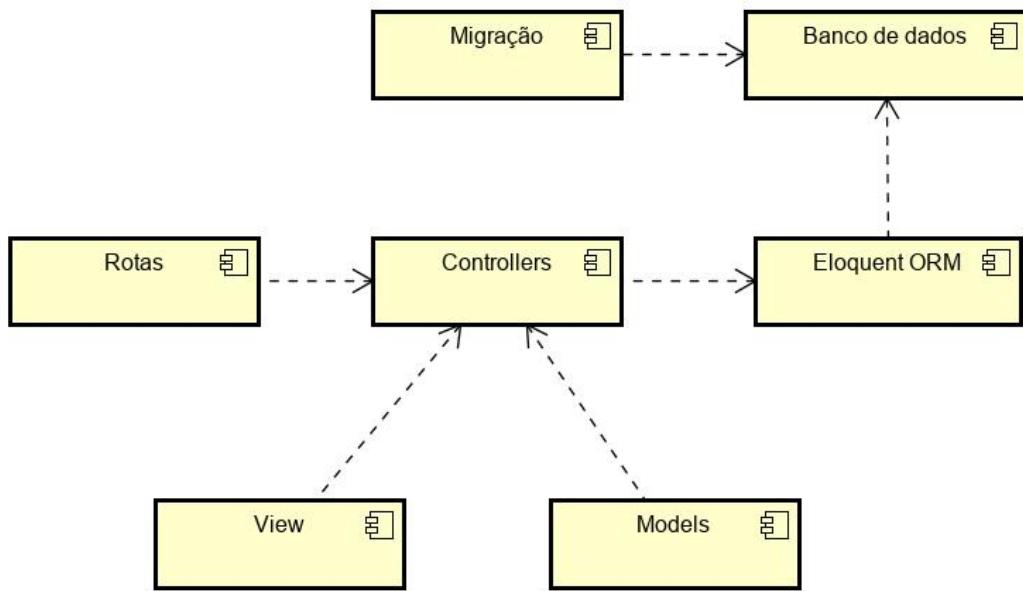


Figura 15: Componentes do projeto

Como estamos utilizando o github, disponível em: <https://github.com/felipeseolin/project-arqhub>, nosso projeto está com seu código aberto para toda a comunidade, de modo que qualquer aluno da disciplina possa acessar, baixar e tentar utilizar o sistema. Também utilizamos o Heroku para subirmos em um servidor online gratuito, sendo que pode ser acessado pela seguinte url: <https://project-arqhub.herokuapp.com>.

7 Considerações Finais

Com o início do projeto, podemos observar que este ramo é pouco explorado atualmente, sendo assim as oportunidades para o crescimento são maiores. As opções presentes no mercado se tornam irrelevantes principalmente por não apresentarem a possibilidade de que qualquer projetista exponha seu trabalho no website, restringindo-se a projetos de empresas específicas.

Notamos também, que o projeto poderia ser evoluído ao ponto de tornarmos a plataforma uma espécie de portfólio online com características de redes sociais, sendo um tanto parecido com o GitHub e o Behance, mas voltado para projetos arquitetônicos. Este é um dos pontos que pode ser melhor explorado em trabalhos futuros.

Outra possível melhoria é nos filtros para listagem de projetos. Pode-se fazer uma pesquisa em sites semelhantes para observar quais são as características mais utilizadas para a listagem e utilizar estas em nosso sistema. Sendo uma possibilidade, a adoção de um framework javascript como: Angular, ReactJS ou Vue e banco de dados como o Mongo, que caracterizariam <https://github.com/felipeseolin/project-arqhubuma> melhoria consideravelmente grande e mais adequada, porém que não foram adotadas devido a equipe não possuir experiência e já estar aprendendo várias novas tecnologias.

Nosso projeto foi direcionado pela metodologia FDD, como já descrito na seção 3 deste documento, e esta metodologia em resumo, se diz sobre planejar e construir por funcionalidades. No projeto e produção das funcionalidades não tivemos grandes empecilhos, a não ser na parte de login: inicialmente foi previsto a implementação de login com a API do Google, mas por questões de prazos das entregas este componente foi implementado com a biblioteca Laravel Auth, estabelecendo-se sessões e todas as outras características necessárias.

O cronograma estipulado inicialmente pode ser concluído. Todas as fases foram monitoradas e acompanhadas pelos membros da equipe no Github para evitar imprevistos e preparar planos de contingência, caso fosse necessário. A implementação conseguiu satisfazer os requisitos especificados durante o projeto, garantindo que as principais funcionalidades e aplicabilidades do sistema sejam satisfeitas. Os requisitos funcionais foram implementados em sua totalidade, por sua vez os requisitos não funcionais foram todos atendidos, com exceção do RN05, que foi deixado como não essencial, pois como descrito anteriormente, a sua implementação poderia atrasar o projeto e seu sucesso, sendo possível adicionar tal requisito com certa facilidade no futuro.

Por fim, vale a pena ressaltar que o projeto está bem modularizado, sendo que quaisquer alterações, como a adoção de um framework front-end, a troca do um banco de dados ou a mudança do programa para servir como uma API, não seria muito trabalhoso.

8 Bibliografia

ROCHA, Fábio Gomes. **Introdução ao FDD:** Feature Driven Development. 2013. Disponível em: <<https://www.devmmedia.com.br/introducao-ao-fdd-feature-driven-development/27971>>. Acesso em: 30 mar. 2019.

Faculdade de Engenharia da Universidade do Porto. **Metodologia Ágil: Feature Driven Development.** Disponível em: <https://paginas.fe.up.pt/~aaguiar/es/artigos%20finais/es_final_22.pdf>. Acesso em: 30 mar. 2019.

AMBLER, Scott W.. **Feature Driven Development (FDD) and Agile Modeling.** Disponível em: <<http://agilemodeling.com/essays/fdd.htm>>. Acesso em: 30 mar. 2019.

ArqHub. 2019. Disponível em: <<https://project-arqhub.herokuapp.com>>. Acesso em: 26 jun. 2019.

Project ArqHub. 2019. Disponível em: <<https://github.com/felipeseolin/project-arqhub>>. Acesso em: 26 jun. 2019.