Bota Fora
Documento de Arquitetura de Software

Versão 1.1

Autores

Francine Dias
Gabriela Trevisan
Guilherme Barth
Matheus Hahn dos Santos
Paula Adriana Knob
Tais Rabello

Histórico de Revisões

Data	Versão	Descrição	Autor
09/09/2019	1.0	Primeira versão	Matheus Hahn
09/09/2019	1.1	Ajustes de índice, escrita e diagrama de classes	Paula A. Knob

Sumário

1. Introdução

1.1 Finalidade

Este documento oferece uma visão geral abrangente do sistema, usando diversas visões arquiteturais para representar diferentes aspectos do sistema. O objetivo deste documento é capturar e comunicar as decisões arquiteturais significativas que foram tomadas em relação ao sistema.

1.2 Escopo

Este documento de Arquitetura de Software se aplica ao sistema Bota Fora.

1.3 Definições, Acrônimos e Abreviações

- **HTTP** – Hypertext Transfer Protocol

2. Representação da Arquitetura

Esta seção descreve qual é a arquitetura de software do sistema atual e como ela é representada. Nas Visões de Casos de Uso, Lógica, do Processo, de Implantação e de Implementação, este documento enumera as visões necessárias e, para cada uma delas, explica os tipos de elementos do modelo que contém.

3. Metas e Restrições de Arquitetura

Esta seção descreve os requisitos de software e os objetivos que têm um impacto significativo na arquitetura, como proteção, segurança, privacidade, uso de um produto desenvolvido internamente e adquirido pronto para ser usado, portabilidade, distribuição e reutilização.

Existem algumas restrições de requisito e de sistema principais que têm uma relação significativa com a arquitetura. São elas:

- O sistema estará disponível via internet.
- A linguagem de desenvolvimento utilizada para o sistema será JavaScript.
- O Framework Ionic será utilizado para desenvolver o aplicativo híbrido.
- O Aplicativo será hospedado no Firebase.
- O banco de dados escolhido será o Firebase Database.

4. Visão de Casos de Uso

Esta seção lista os casos de uso ou cenários do modelo de casos de uso que representam uma funcionalidade central e significativa do sistema final ou se têm uma ampla cobertura de arquitetura, ou seja, que experimenta vários elementos de arquitetura e enfatizam ou ilustram um determinado ponto complexo da arquitetura.

Os Casos de Uso do sistema Bota Fora são listados a seguir. Os Casos de Uso em negrito são destacados por que são muito importantes para a arquitetura:

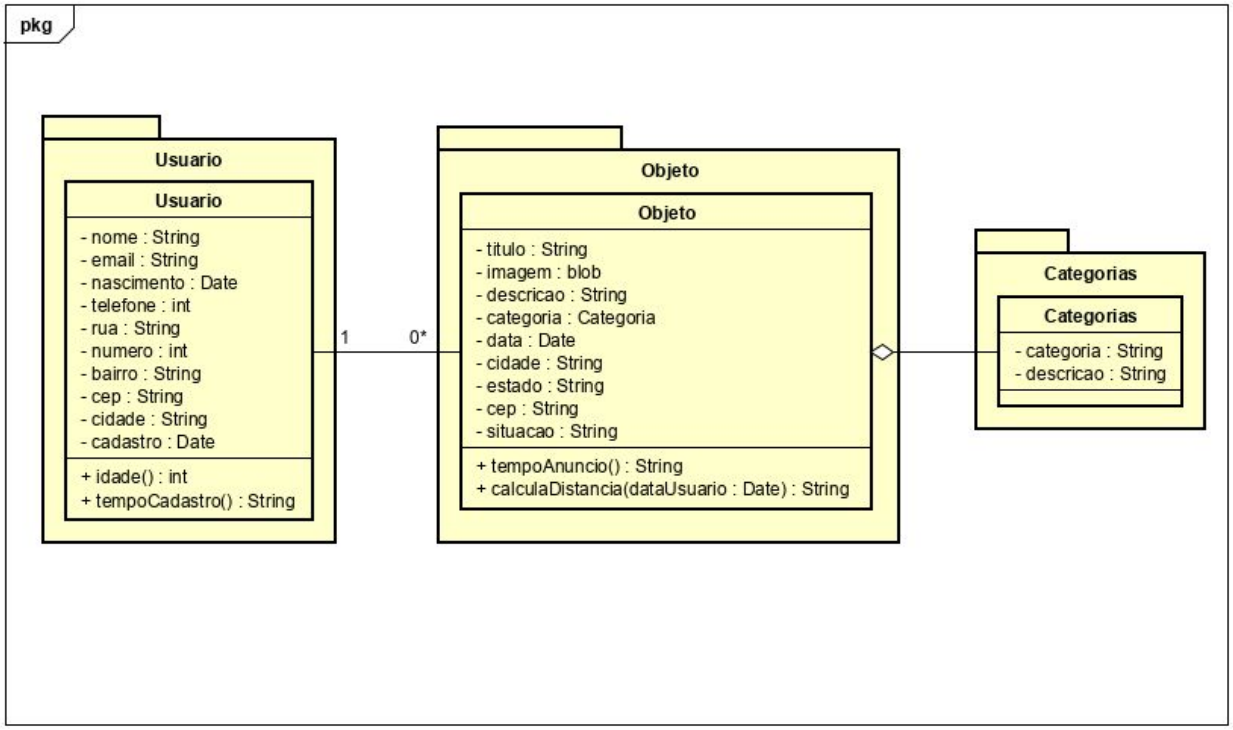
- **Criar Conta;**
- Preencher cadastro;
- **Demonstrar interesse em item;**
- Contatar doador;
- **Cadastrar item;**
- Atualizar Item;
- Notificar interessados;
- Mudar status do item;

5. Visão Lógica

Esta seção descreve as partes significativas do ponto de vista da arquitetura do modelo de design, como sua divisão em subsistemas e pacotes. Além disso, para cada pacote significativo, ela mostra sua divisão em classes e utilitários de classe. Apresenta as classes significativas do ponto de vista da arquitetura e descreve suas responsabilidades, bem como alguns relacionamentos, operações e atributos de grande importância.

5.1 Divisão em Pacotes

Nesta seção serão apresentadas as camadas da arquitetura proposta. Serão descritas as responsabilidades de cada camada quais tecnologias devem ser aplicadas a cada uma delas.



5.1.1 *Pacote Usuário*

Classe Usuario	
Descrição	Abriga os usuários (um usuário é tanto recebedor quanto doador)
Relações	
Responsabilidades	Saber o nome do usuário Saber o email do usuário para autenticação Saber se o usuário tem maioridade Saber o contato telefônico do usuário Saber o endereço completo do usuário Saber a data de cadastro do usuário
Métodos	idade(): retorna a idade do usuário tempoCadastro(): retorna há quanto tempo o usuário é cadastrado

Atributos	nome: nome do usuário email: email de cadastro nascimento: data de nascimento do usuário telefone: contato do usuário rua: rua do endereço do usuário número: número do endereço do usuário bairro: bairro do endereço do usuário cep: CEP do endereço do usuário cidade: cidade do endereço do usuário cadastro: data em que o usuário se cadastrou

5.1.2 *Pacote Objeto*

Classe Objeto	
Descrição	Armazena os objetos que estão sendo doados
Relações	Depende da classe Categorias.
Responsabilidades	Saber o "nome" do objeto Saber uma descrição mais detalhada do objeto Ter fotos do objeto Saber a localização atual do objeto Saber a categoria à qual o objeto pertence Calcular a quanto tempo o objeto foi cadastrado Calcular a distância do objeto em relação ao usuário Saber a disponibilidade da doação
Métodos	tempoAnuncio(): calcula há quanto tempo o objeto foi cadastrado calculaDistancia(): retorna a distância da localização do objeto e do usuário que está considerando a doação
Atributos	titulo: nome do objeto, será usado como título do anúncio imagem: fotos do objeto descricao: detalhes do objeto categoria: tipo de objeto data: data em que foi cadastrado cidade: cidade onde o objeto está localizado estado: estado onde o objeto está localizado cep: CEP de onde o objeto está localizado situação: se está disponível, reservado ou se já foi doado

5.1.3 Pacote Categorias

Classe Categoria	
Descrição	Abriga os tipos de móveis
Relações	
Responsabilidades	Saber o os nomes das categorias Saber uma descrição detalhada das categorias
Métodos	

Atributos	categoria: tipo de objeto descricao: detalhes da categoria

6. Visão de Implantação

Como estamos utilizando o Framework Ionic para implementar o aplicativo híbrido, os seguintes dispositivos/hardwares serão suportados para mobile:

- Android: Versão 4.4+;
- iOS: versão 10+;

Os seguintes browsers também são suportados:

- Chrome;
- Safari;
- Edge;
- Firefox;
- IE: versão 11+;

7. Visão de Implementação

A visão de implementação aborda a arquitetura sobre a perspectiva do projeto estrutural dos componentes do sistema, como o sistema e cada um dos seus componentes serão organizados em termos de diretórios e como o sistema, será empacotado para publicação (deployment).

A visão de implementação retrata a composição física da implementação em termos de subsistemas e elementos (diretórios, arquivos, dados, executáveis). Geralmente essas camadas de implementação se ajustam às camadas definidas na visão lógica.

7.1 Camadas

Nesta seção serão apresentadas as camadas da arquitetura proposta bem como as responsabilidades de cada uma delas.

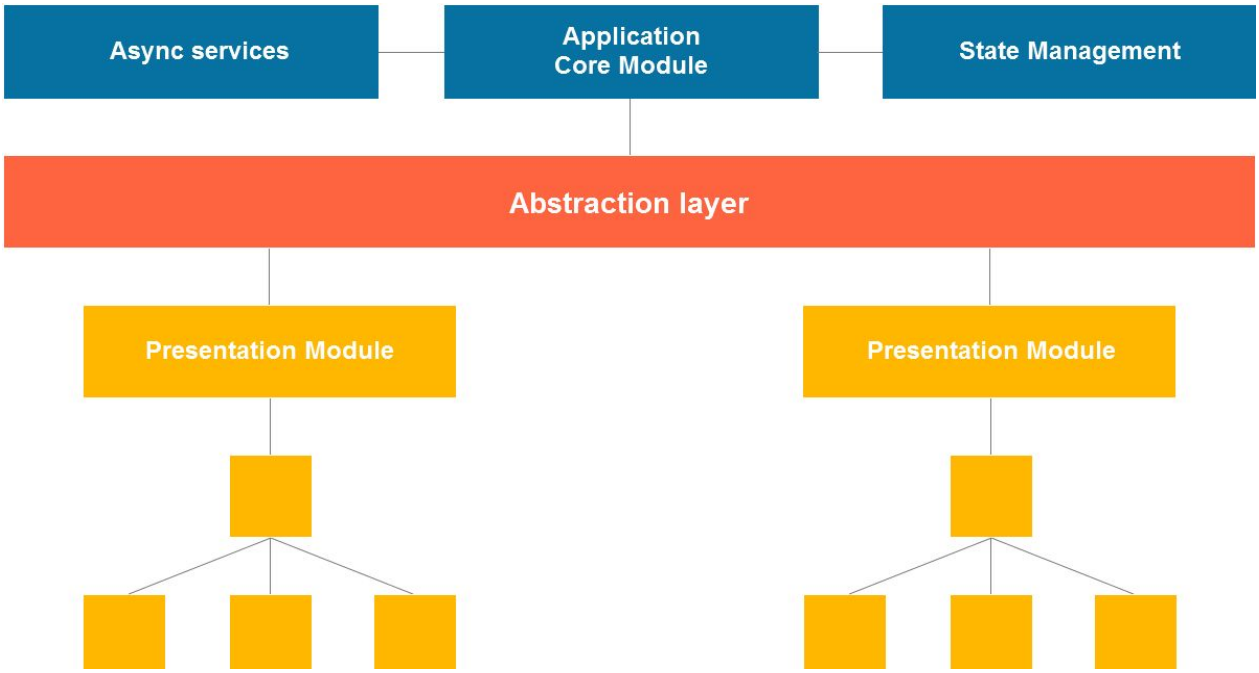


Figura 1. Design de alto nível em módulos

7.1.1 Application Core Module

A camada do core contém a lógica da aplicação, manipulação de dados, comunicação com APIs, entre outros. Cada uma das camadas é separada por módulos e organizada por features. A comunicação é feita através de serviços assíncronos, podendo ter n serviços para n APIs.

7.1.2 Abstraction Layer

A camada de abstração provê acesso ao core da aplicação para que a camada de apresentação e o core possam ser desacoplados, logo torna testes e a reutilização mais fáceis.

7.1.3 Presentation Module

A camada de apresentação são módulos independentes que incluem todas as suas dependências(e.g. configurações, estado) necessárias do core através da camada de abstração para facilitar a reutilização e testes individuais.

7.2 Soluções utilizadas

7.2.1 Ionic

Ionic framework é um kit de ferramentas de UI móvel, gratuito e de código aberto para o desenvolvimento de aplicativos híbridos de alta qualidade para iOS, Android e web com apenas uma base de código.
O Ionic Framework concentra-se na experiência do usuário front-end e na interação da interface do usuário de um aplicativo (controles, interações, gestos, animações). É fácil de aprender e integra-se bem a outras bibliotecas ou frameworks, como Angular. Atualmente, o Ionic Framework possui integração oficial com o Angular, mas o suporte ao Vue e React está em desenvolvimento.
Um dos principais benefícios do Ionic é possuir diversos componentes de UI que aceleram o desenvolvimento e se adaptam de acordo com a plataforma.

7.2.2 **Angular**

Angular é um framework de aplicações web de código aberto e front-end baseado em *TypeScript* liderado pelo Google. Possui uma grande e ativa comunidade de desenvolvedores e empresas que utilizam e ajudam no desenvolvimento. Assim com o Ionic, também é multiplataforma e possui uma ótima velocidade e performance comparado com os outros frameworks disponíveis no mercado.

7.2.3 **Firebase**

Firebase é uma plataforma de desenvolvimento de aplicações web e mobile desenvolvido pela Firebase em 2011 e depois adquirido pelo Google em 2014. Segundo a própria Firebase, possui mais de 1,5 milhões de apps rodando e utilizando ativamente a plataforma.
Firebase te dá funcionalidades como controle de autenticação, hospedagem, database, mensagens, relatórios de erros e muitas outras funcionalidades sem necessidade de lidar com a infraestrutura para que o time possa focar no desenvolvimento e nos usuários.