DOCUMENTO DE ARQUITETURA DE SOFTWARE

Versão 1.4

Universidade Joaquim Nabuco
Bacharelado em Sistemas de Informação

Equipe de Desenvolvimento

Claudiomildo Ventura

Leonardo Belas

Lucas Wallis

William Andrey

SOFTWORK / 2019

Paulista – PE

Histórico de Revisões

Versão	Data	Autor	Descrição
1.0	28/04/2019	Lucas Wallis	Elaboração do documento
1.1	01/05/2019	Lucas Wallis	Avaliações e modificações gerais do documento
1.2	20/05/2019	Claudiomildo Ventura	Avaliações e modificações gerais do documento
1.3	20/06/2019	Claudiomildo Ventura	Avaliações e modificações gerais do documento
1.4	30/09/2019	Lucas Wallis	Avaliações e modificações gerais do documento

Índice

1. Introdução 1
1.1. Finalidade 1
1.2. Escopo 1
1.3. Definições, Acrônicos E Abreviações 1
2. Representação Arquitetural 1
2.1. View 2
2.2. Control 2
2.3. Negócios 2
2.4. Model 2
3. Metas E Restrições Da Arquitetura 2
3.1. Usabilidade 3
3.2. Segurança 3
3.3. Desempenho 3
3.4. Manutenabilidade 3
3.5. Portabilidade 3
4. Visão Da Arquitetura 3
4.1. Visão Lógica 4
4.2. Visão Do Caso De Uso 4
4.3. Visão Geral 4
5. Visão Lógica ..... Error! Bookmark not defined.
6. Visão De Implantação 4
7. Visão Do Diagrama De Classe 5
8. Visão De Dados 5
9. Tamanho E Desempenho 5
10. Qualidade 6
11. Padrões Adotados 6
11.1. Padrão De Nomeclatura 6

1. INTRODUÇÃO

1.1. Finalidade

Este documento detalha de forma especifica os aspectos arquiteturais que foram adotados para o desenvolvimento do projeto Staff Fitness. Desta forma os desenvolvedores terão as informações necessárias para a construção do sistema.

1.2. Escopo

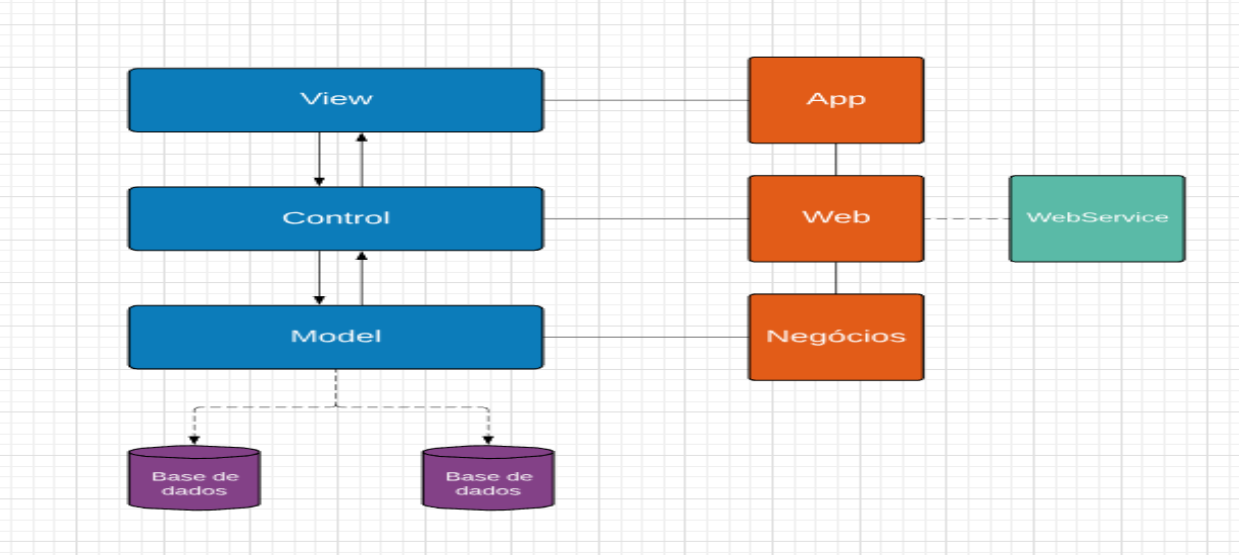
Este documento visa explanar o modelo de arquitetura da aplicação Staff Fitness. Esta arquitetura será utilizada na implementação dos casos de uso da aplicação.

1.3. Definições, Acrônicos e abreviações

Artefato de glossário.

2. REPRESENTAÇÃO ARQUITETURAL

O padrão escolhido para o desenvolvimento do projeto foi o modelo em camadas. O estilo em camadas organiza o sistema em camadas hierárquicas.



2.1. View

Neste secão será processada as entradas e saídas das operações realizadas pelo usuário. Além disso, esta camada mantém uma relação com o usuário e a camada de comunicação. É responsável por usar as informações modeladas para produzir interfaces de apresentação conforme a necessidade.

2.2. Control

Esta camada tem o objetivo de prover a comunicação entre a interface gráfica e as operações relacionadas a requisições do usuário. Está camada será como uma porta de acesso que autentica e autoriza o usuário a estabelecer uma comunicação direta com a aplicação. Interpreta as requisições submetidas pelo usuário e traduz em comandos que são enviados para o (Model) e/ou para a (View).

2.3. Negócios

Esta camada promoverá a interação do usuário com a aplicação no sentido de executar as atividades disponíveis ao usuário. Desta maneira, proporcionará a modificação e recuperação de informações pertinentes aos processos de negócio. Sendo assim, serão estabelecidas classes básicas para representar as entidades envolvidas no processo que foram especificadas nos casos de uso desse projeto. Como também, serão estabelecidas coleções que reunirão as classes básicas do sistema com o intuito de permite manipulações dessas classes. Por exemplo, permite atualização, consulta requisição etc. Além disso, serão estabelecidas classes que controlem o fluxo principal e secundário de atividades do sistema.

2.4. Model

Esta camada tem o objetivo de estabelecer o armazenamento persistente dos dados do sistema como também de criar mecanismos de tratá-los e de recuperá-los. Desta forma, é a camada que gerencia as transações e que se relaciona com a camada de negócio.

3. METAS E RESTRIÇÕES DA ARQUITETURA

As metas e restrições estabelecidas para o projeto Staff Fitness se encontram no documento de requisitos, no que se refere aos requisitos não funcionais. Desta forma, visando à

qualidade da aplicação, foram estabelecidos os seguintes critérios que estão especificados logo abaixo.

3.1. Usabilidade

O sistema deve ser de fácil compreensão e utilização por parte dos usuários.

3.2. Segurança

A aplicação deve ser confiável. Sendo assim, deve oferecer proteção de tal maneira que não cause danos financeiros ou doloso aos usuários. Desta forma, deverá ser resistente a intrusões acidentais ou intencionais. Além disso, deve oferecer persistência dos dados armazenados no sistema. Por tanto, é fundamental o comprometimento com a integridade dos dados.

3.3. Desempenho

O sistema deverá estar disponível 24 horas, e 7 dias por semana, capaz de fornecer serviços úteis a qualquer momento que for solicitado. Deverá garantir o fornecimento correto dos serviços esperado pelo usuário. Além disso, o tempo de processamento das operações não poderão ultrapassar 30 segundos.

3.4. Manutenabilidade

A aplicação deve oferecer facilidade de ser mantido através de manutenções. Sendo assim, estaremos seguindo os padrões durante a sua produção. Como também, garantir que novos requisitos possam ser incluídos quando houver necessidade.

3.5. Portabilidade

A aplicação será capaz de rodar em plataformas que os seus usuários estão acostumados a utilizar. Atualmente, a aplicação *mobile* está sendo desenvolvido na versão para dispositivos *Android*.

4. VISÃO DA ARQUITETURA

Os tipos de visões aqui descritas servem para esclarecer quais ferramentas e métodos serão utilizados no desenvolvidos do projeto Staff Fitness.

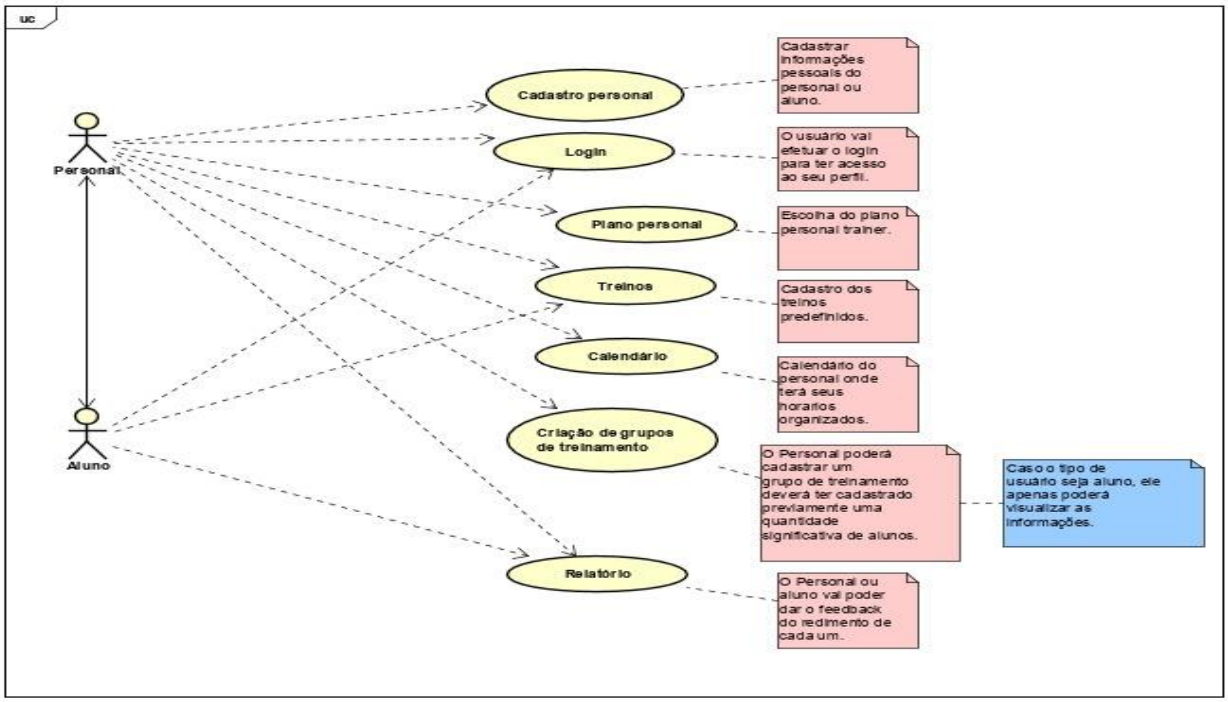
4.1. Visão Lógica

A aplicação Staff Fitness será desenvolvida utilizando as ferramentas *React-Native* para o desenvolvimento da aplicação mobile. O *React JS* para o desenvolvimento da aplicação Web. O *Adonis JS* sendo utilizado como *API* pra estabelecer o trafego e consumo de informações. E o banco de dados utilizado será o *PostgreSQL*.

4.2. Visão do Caso de Uso

As especificações dos casos de uso estão descritas no documento de caso de uso.

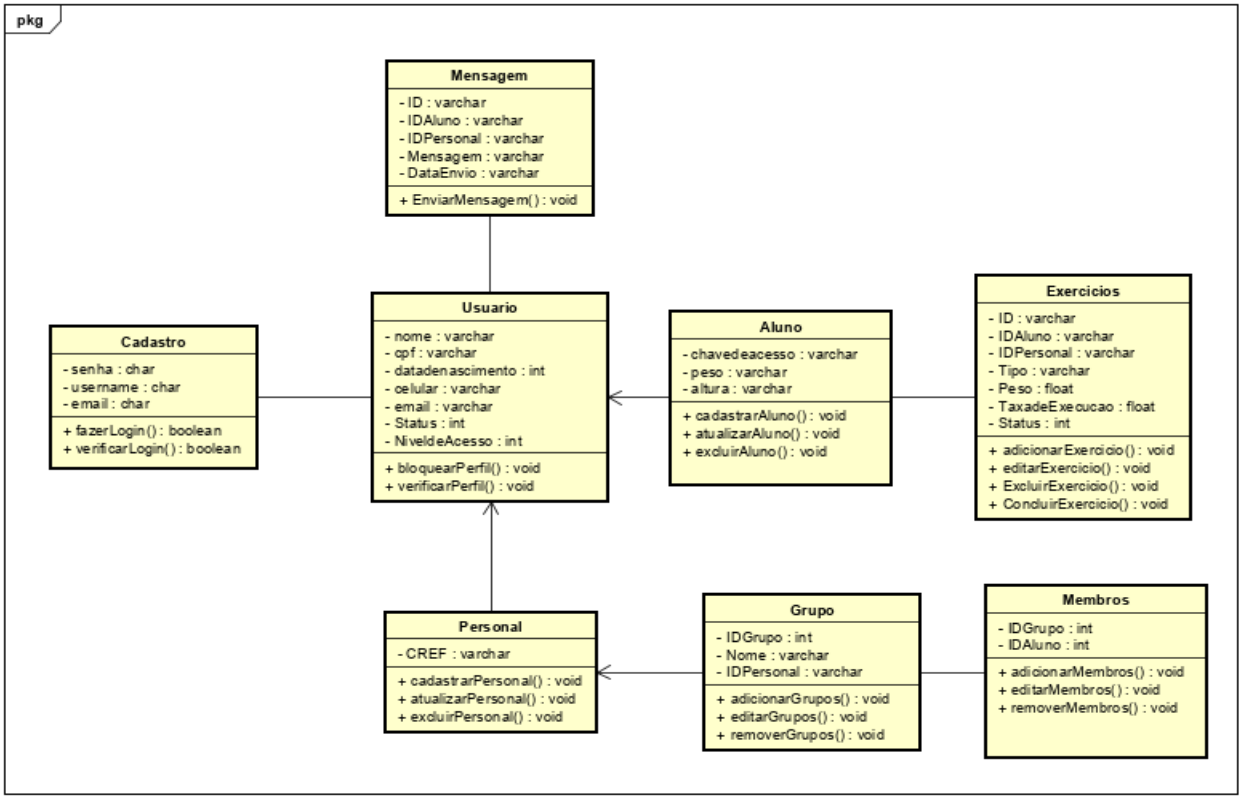
4.3. Visão Geral



5. VISÃO DE IMPLANTAÇÃO

Para utilizar a aplicação Staff Fitness é necessário possuir dispositivo com acesso à internet e navegadores *Web* que ofereça condições de acesso, sendo eles (*Mozilla, Internet Explorer e Chrome*).

6. VISÃO DO DIAGRAMA DE CLASSE



7. VISÃO DE DADOS

O repositório das informações referente a aplicação será executado e servido por uma *API* de comunicação remota para manter a persistência e integridade dos dados.

8. TAMANHO E DESEMPENHO

A complexidade do desenvolvimento da aplicação é considerada de altíssima grau, devido à necessidade de interação com a aplicação *web* e *mobile,* em que as duas estarão trabalhando simultaneamente.

O desempenho poderá ser considerado alto devido ao uso de altos cálculos matemáticos, em contrapartida o desempenho gráfico é mediano, mas mesmo assim ocasionando um baixo consumo de recursos computacionais. Foi definido um tempo de resposta máximo como restrição do desempenho desejado.

9. QUALIDADE

A arquitetura da aplicação permite que novas funcionalidades possam ser desenvolvidas e adicionadas ao modelo atual de arquitetura sem dificuldades, garantindo a sua escalabilidade de forma satisfatória.

10. PADRÕES ADOTADOS

10.1. Padrões de Nomeclatura

- Nomes de classes sempre terão a primeira letra maiúscula.
  Ex. Cliente.

- Nomes de variáveis serão sempre minúsculos, porém se for um nome composto ele será escrito sem espaço entre as palavras e a primeira letra do segundo nome deverá ser maiúscula.
  Ex. nome, nomeCliente.

- Nomes de métodos seguirão o mesmo padrão das variáveis.
  Ex. inserirCliente.

- Nomes de constantes deverão ser completamente maiúsculos e caso for composto deve ser separado por underline “_”.
  Ex. TABELA_USUARIO