Documento de Arquitetura de Software

|  |
| --- |
| **ZikaFoot 2020** |
| **Gestor do Projeto** |
| Ejziel Sampaio Santos |
| ejziels@gmail.com |

|  |
| --- |
| **Objetivo deste Documento** |
| Este documento tem como objetivo descrever as principais decisões de projeto tomadas pela equipe de desenvolvimento e os critérios considerados durante a tomada destas decisões. Suas informações incluem aparte de *hardware* e *software* do sistema. |

Sumário

[1. INTRODUÇÃO 3](#_Toc14351453)

[1.1 Finalidade 3](#_Toc14351454)

[1.2 Escopo 3](#_Toc14351455)

[1.3 Definições, Acrônimos e Abreviações 3](#_Toc14351456)

[1.4 Referências 4](#_Toc14351457)

[2. REPRESENTAÇÃO ARQUITETURAL 4](#_Toc14351458)

[3. REQUISITOS E RESTRIÇÕES ARQUITETURAIS 5](#_Toc14351459)

[4. VISÃO DE CASOS DE USO 5](#_Toc14351460)

[4.1 Casos de Uso significantes para a arquitetura 5](#_Toc14351461)

[5. VISÃO LÓGICA 6](#_Toc14351462)

[5.1 Visão Geral – pacotes 6](#_Toc14351463)

[6. VISÃO DE IMPLEMENTAÇÃO 7](#_Toc14351464)

[6.1 Cadastro de Time [001] 7](#_Toc14351465)

[6.1.1 Diagrama de Sequência 7](#_Toc14351466)

[6.2 Cadastro de Jogador [002] 8](#_Toc14351467)

[6.2.1 Diagrama de Sequência 8](#_Toc14351468)

[6.3 Listar Elenco [003] 9](#_Toc14351469)

[6.3.1 Diagrama de Sequência 9](#_Toc14351470)

[6.4 Jogar Amistoso [101] 10](#_Toc14351471)

[6.4.1 Diagrama de Sequência 10](#_Toc14351472)

[7. DIAGRAMA DE CLASSES 11](#_Toc14351473)

[8. VISÃO DE IMPLANTAÇÃO 12](#_Toc14351474)

# INTRODUÇÃO

## Finalidade

Este documento fornece uma visão arquitetural abrangente do Sistema de Gerenciamento Escolar usando diversas visões de arquitetura 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.

O documento irá adotar uma estrutura baseada na visão “4+1” de modelo de arquitetura [KRU41].



Figura 1 – Arquitetura 4+1

## Escopo

Este Documento de Arquitetura de Software se aplica ao ZikaFoot 2020, que será desenvolvido por Ejziel Sampaio.

## Definições, Acrônimos e Abreviações

QoS – Quality of Service, ou qualidade de serviço. Termo utilizado para descrever um conjunto de qualidades que descrevem as requisitos não-funcionais de um sistema, como performance, disponibilidade e escalabilidade [QOS].

## Referências

[KRU41]: The “4+1” view model of software architecture, Philippe Kruchten, November 1995, <http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/Pbk4p1.pdf>

[QOS] <https://docs.oracle.com/cd/E19636-01/819-2326/6n4kfe7dj/index.html>

# REPRESENTAÇÃO ARQUITETURAL

Este documento irá detalhar as visões baseado no modelo “4+1” [KRU41], utilizando como referência os modelos definidos na MDS. As visões utilizadas no documento serão:

|  |  |  |
| --- | --- | --- |
| Visão | Público | Área |
| Lógica | Analistas | Realização dos Casos de Uso |
| Processo | Integradores | Performance, Escalabilidade, Concorrência |
| Implementação | Programadores | Componentes de Software |
| Implantação | Gerência de Configuração | Nodos físicos |
| Caso de Uso | Todos | Requisitos funcionais |
| Dados | Especialistas em dados  Administradores de dados | Persistência de dados |

Tabela 1 – Visões, Público, Área e Artefatos da MDS

# REQUISITOS E RESTRIÇÕES ARQUITETURAIS

Esta seção descreve os requisitos de software e restrições que tem um impacto significante na arquitetura.

|  |  |
| --- | --- |
| Requisito | Solução |
| Linguagem | JAVA 1.8 |
| Plataforma | Windows e Linux |
| Segurança | O sistema não possui aspectos específicos relacionados à segurança. |
| Persistência | O sistema utiliza o servidor de banco de dados MySQL. |

Tabela 2 – Exemplo de requisitos e restrições

# VISÃO DE CASOS DE USO

Esta seção lista as especificações centrais e significantes para a arquitetura do sistema.

Lista de casos de uso do sistema:

* **Cadastrar Time [001]**
* **Cadastrar Jogador [002]**
* **Listar Elenco [003]**
* **Jogar Amistoso [101]**
* **Iniciar Amistoso [102]**

## Casos de Uso significantes para a arquitetura

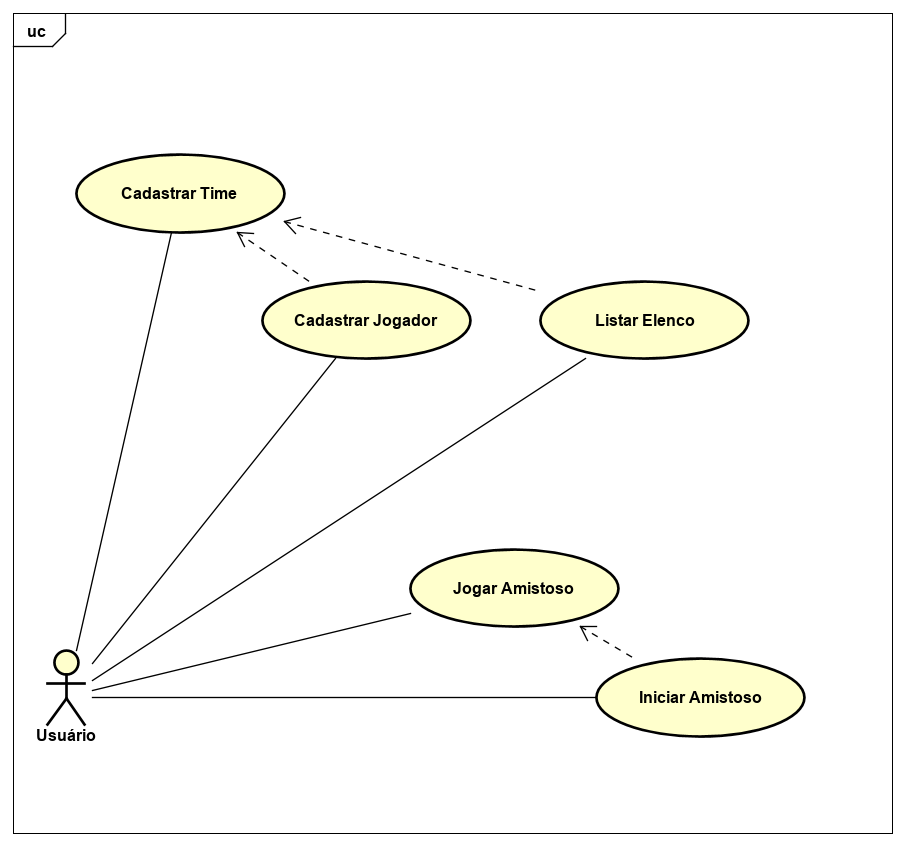


Figura 2 – Diagrama com os casos de uso significativos e atores

# VISÃO LÓGICA

No projeto será utilizado o padrão de projeto MVC, que é utilizado em muitos projetos devido à arquitetura que possui, o que possibilita a divisão do projeto em camadas muito bem definidas. Cada uma delas, o Model, o Controller e a View, executa o que lhe é definido e nada mais do que isso. A utilização do padrão MVC trás como benefício isolar as regras de negócios da lógica de apresentação, a interface com o usuário. Isto possibilitará que futuramente o projeto possa ser desenvolvido para outras plataformas com maior facilidade.

## Visão Geral – pacotes

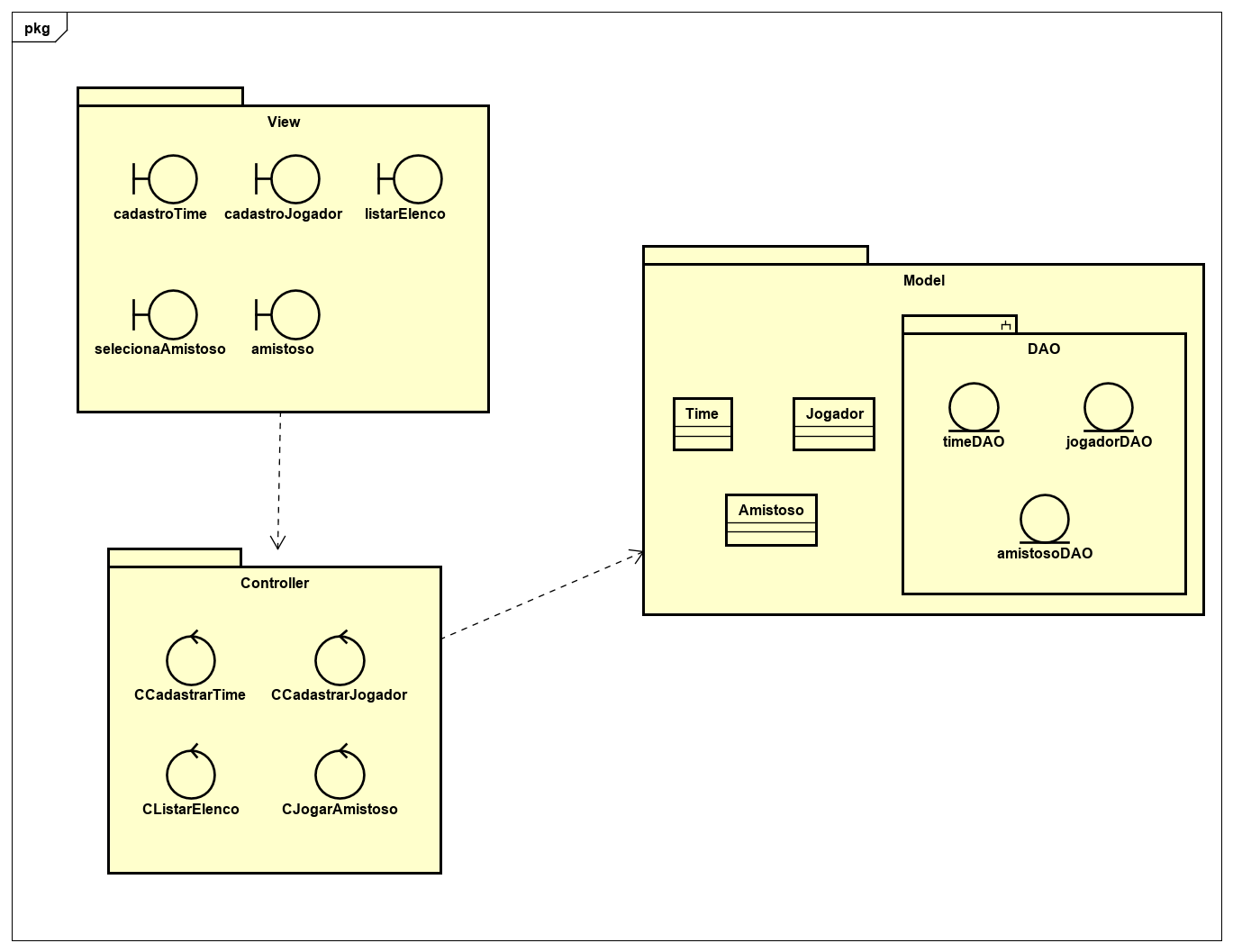


Figura 3 – Diagrama de Pacotes da Aplicação

* **Controlador (Controller)** interpreta as entradas do mouse ou do teclado enviado pelo usuário e mapeia essas ações do usuário.
* **Modelo (Model)** gerencia um ou mais elementos de dados, responde a perguntas sobre o seu estado e responde a instruções para mudar de estado.
* **Visão (View)** gerencia a área retangular do display e é responsável por apresentar as informações para o usuário através de uma combinação de gráficos e textos.

# VISÃO DE IMPLEMENTAÇÃO

Um diagrama de sequência é uma espécie de diagrama de interação, pois descreve como, e em qual ordem, um grupo de objetos trabalha em conjunto. No projeto foram especificados os seguintes diagramas de sequência para cada caso de uso.

## Cadastro de Time [001]

### Diagrama de Sequência

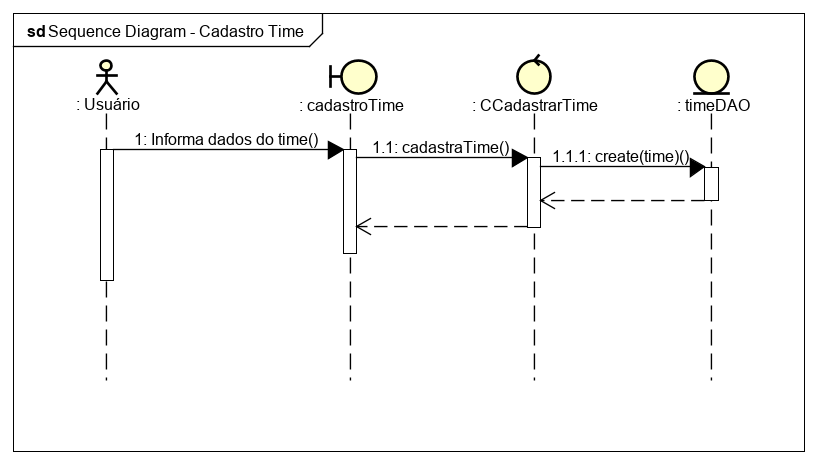


Figura 4 – Diagrama de Sequência

## Cadastro de Jogador [002]

### Diagrama de Sequência

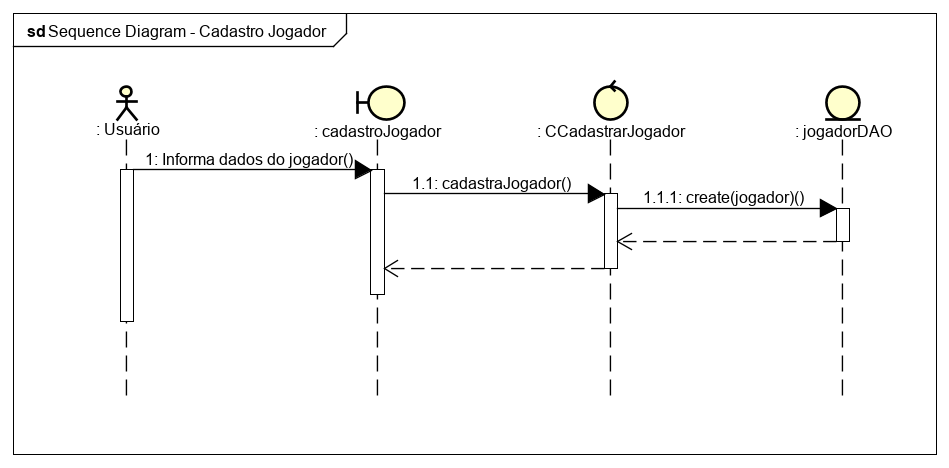


Figura 5 – Diagrama de Sequência

## Listar Elenco [003]

### Diagrama de Sequência

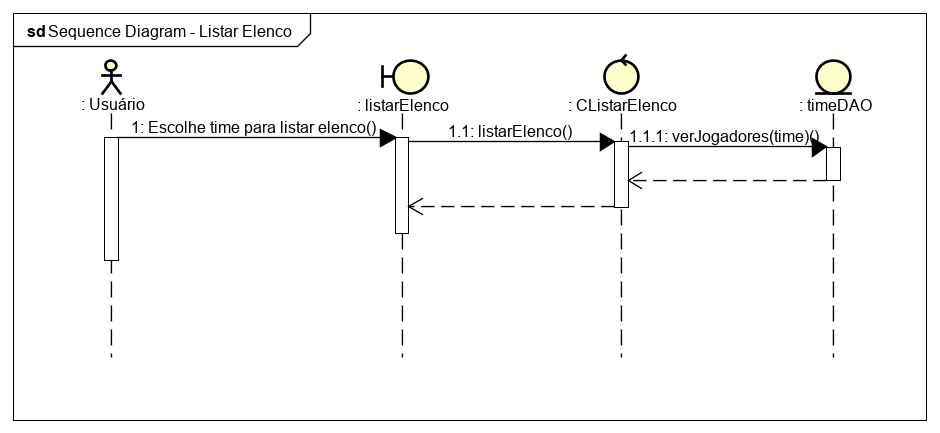


Figura 6 – Diagrama de Sequência

## Jogar Amistoso [101]

### Diagrama de Sequência

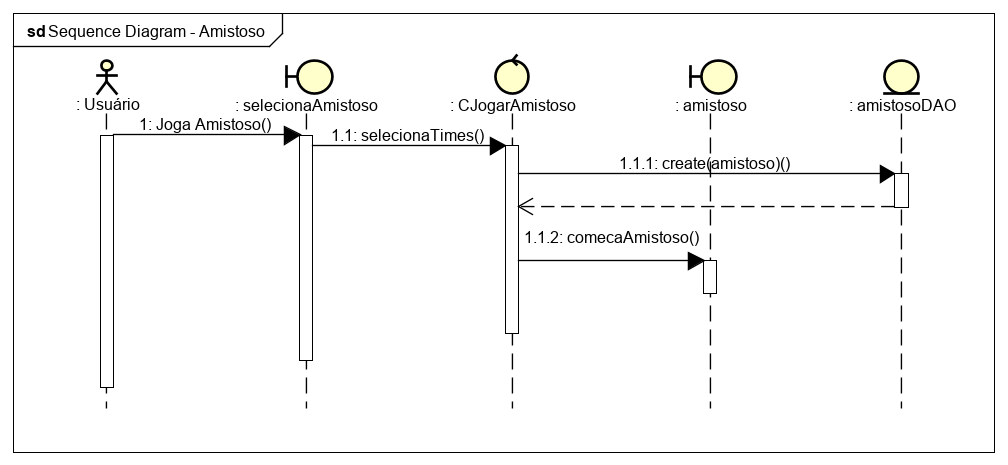


Figura 7 – Diagrama de Sequência

# DIAGRAMA DE CLASSES

Diagramas de classes estão entre os tipos mais úteis de diagramas UML pois mapeiam de forma clara a estrutura de um determinado sistema ao modelar suas classes, seus atributos, operações e relações entre objetos. No sistema foi mapeada a seguinte relação entre as classes.

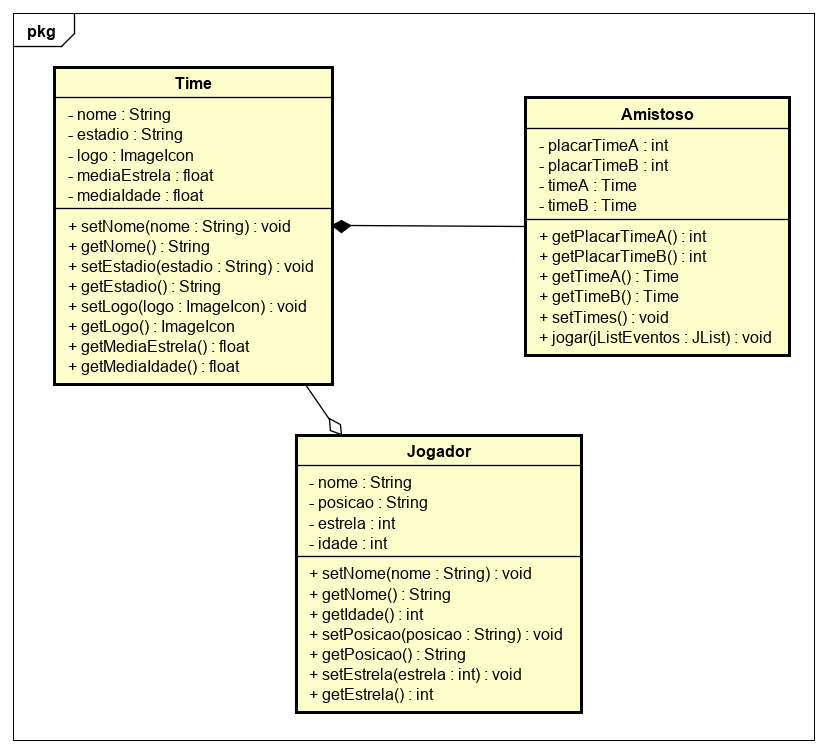


Figura 8 – Diagrama de Classes

# VISÃO DE IMPLANTAÇÃO

Descreve os nodos físicos, as configurações e os artefatos que serão implantados.

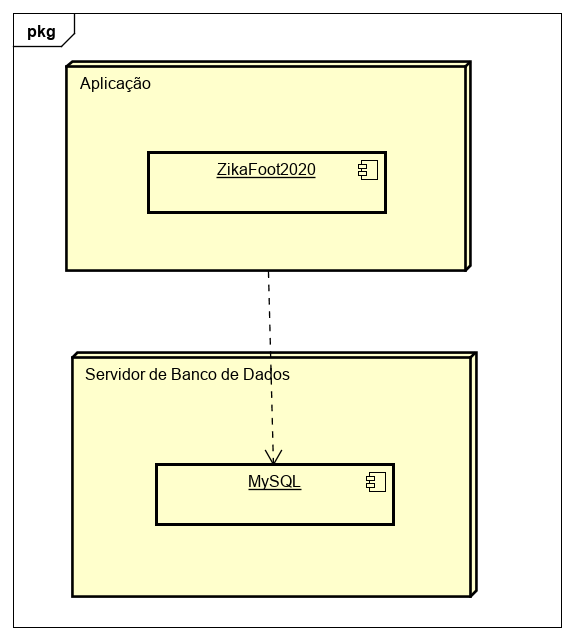


Figura 9 – Diagrama de Implantação Java