SISLAR

Documento de Arquitetura de Software

Versão 1.0

Histórico da Revisão

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** |
| 15/09/2017 | 1.0 | Descrição da arquitetura do sistema | Renato Aguiar |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Índice

1. Introdução 3

1.1 Objetivo 3

1.2 Escopo 3

1.3 Definições, Acrônimos e Abreviações 3

1.4 Referências 3

1.5 Visão Geral 3

2. Representação Arquitetural 3

3. Restrições e Metas Arquiteturais 3

4. Visão de Casos de Uso 3

5. Visão Lógica 3

5.1 Visão Geral 3

5.2 Pacotes de Design Significativos do Ponto de Vista da Arquitetura 3

5.3 Realizações de Casos de Uso 3

6. Visão de Processos 3

7. Visualização da Implementação 3

8. Visão da Implementação 3

8.1 Visão Geral 3

8.2 Camadas 3

9. Visão de Dados (opcional) 3

10. Tamanho e Desempenho 3

11. Qualidade 3

Documento de Arquitetura de Software

# Introdução

Através deste documento serão abordados os aspectos da Arquitetura do sistema SISLAR, mostrando seu objetivo, qual o escopo de abrangência, além de mostrar a representação arquitetural do sistema, junto com casos de usos, tipos de restrições, dentre outros.

## Objetivo

Este documento fornece uma visão arquitetural abrangente do sistema, usando diversas visões de arquitetura para representar diferentes aspectos do sistema. Ele pretende capturar e transmitir as decisões arquiteturas significativas que foram tomadas em relação ao sistema.

## Escopo

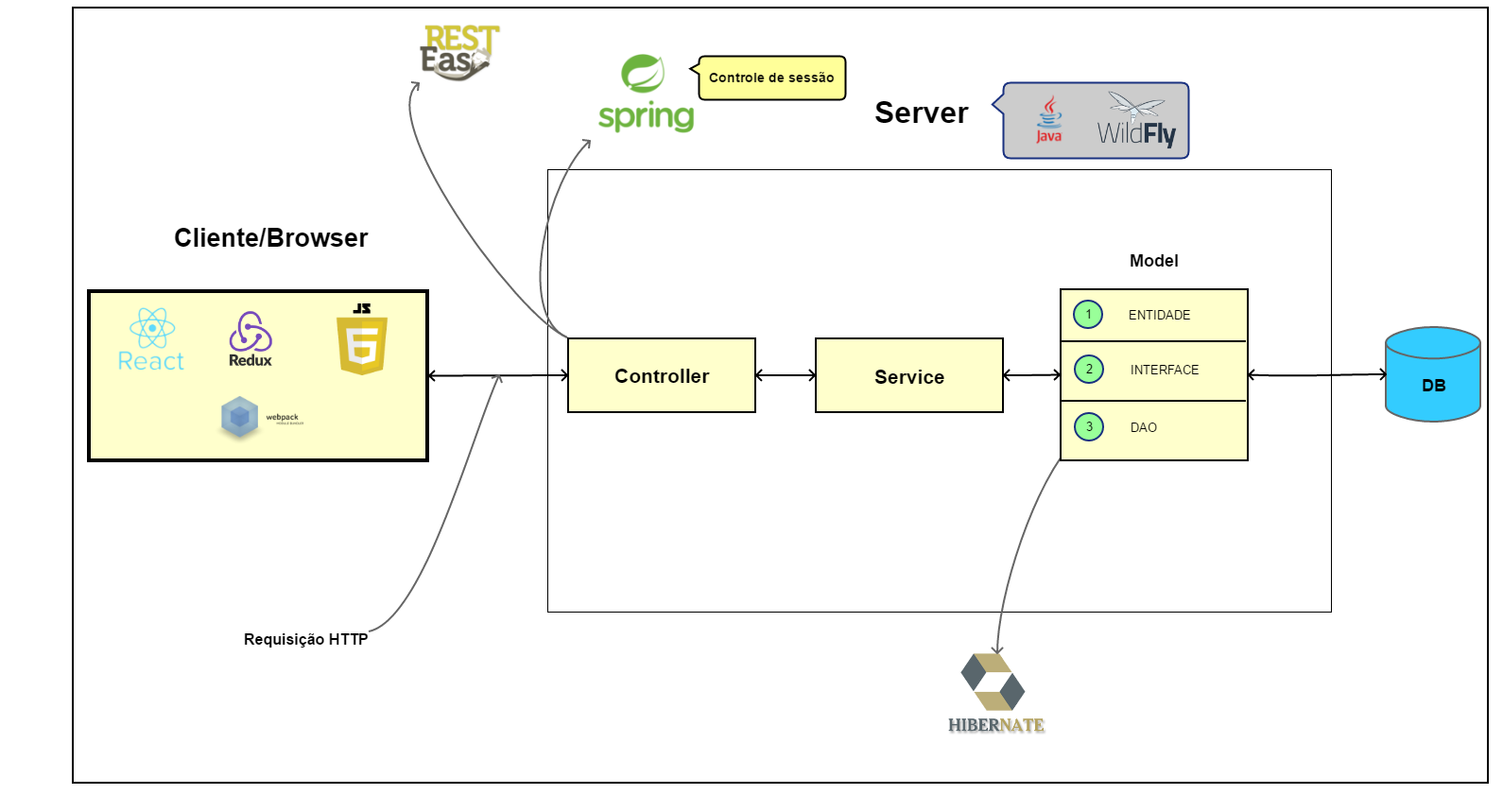
Este documento tem como escopo apresentar uma visão/representação da arquitetura do sistema.

## Visão Geral

Em uma visão geral esse documento vai fornecer informações arquiteturais do sistema, quais são as restrições e metas a serem alcançadas, exemplificação de casos de usos, qual a visão lógica para o sistema.

Abordando pontos como qualidade, visão e implementação.

# Representação Arquitetural



# Restrições e Metas Arquiteturais

|  |  |
| --- | --- |
| Linguagens | Java e React |
| Servidor de aplicações | Wildfly 10.0 |
| Segurança | Spring Security |
| Persistência | Hibernate e JPA |

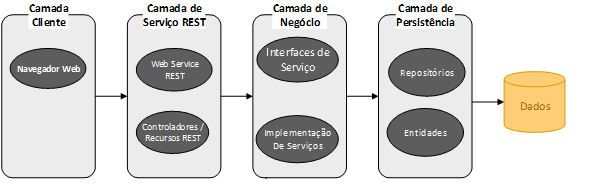
# Visão de Casos de Uso

* Efetuar Login;
* Cadastrar Funcionário;
* Editar Funcionário;
* Apagar Funcionário;
* Escala de Funcionários;
* Check-list enfermeira;
* Anexar documentos;
* Cadastrar Idoso;
* Editar Idoso;
* Apagar Idoso;
* Adicionar Relatório Diário Idoso;
* Editar Relatório Diário Idoso;
* Consultar Relatório Diário Idoso;

## Visão Lógica

## As principais classes do sistema serão aquelas que irão prover a comunicação entre o cliente e servidor, através de filters e controller, o cliente enviará a requisição para o servidor, o filter avaliará se é um usuário e dado válido, sendo aprovado a requisição procede e chega no serviço rest, que neste caso será um conjunto de controllers, a função chamada no controller será responsável por chamar o servisse correto para executar aquela ação, caminhando assim até a camada de persistência e persistindo ou recuperando dados.

## Visão Geral



## Pacotes de Design Significativos do Ponto de Vista da Arquitetura

[Para cada pacote significativo, inclua uma subseção com o respectivo nome, uma breve descrição e um diagrama com todos os pacotes e classes significativos nele contidos.

Para cada classe significativa no pacote, inclua o respectivo nome, uma breve descrição e, opcionalmente, uma descrição de algumas das suas principais responsabilidades, operações e atributos.]

## Realizações de Casos de Uso

[Esta seção ilustra o funcionamento do software, apresentando algumas realizações (ou cenários) de casos de uso selecionadas e explica como os diversos elementos do modelo de design contribuem para a respectiva funcionalidade.]

# Visão de Processos

[Esta seção descreve a decomposição do sistema em processos leves (encadeamentos simples de controle) e processos pesados (agrupamentos de processos leves). Organize a seção em grupos de processos que se comunicam ou interagem. Descreva os modos principais de comunicação entre processos, como transmissão de mensagens e interrupções.]

# Visão da Implementação

[Esta seção descreve a estrutura geral do modelo de implementação, a divisão do software em camadas e subsistemas no modelo de implementação e todos os componentes significativos do ponto de vista da arquitetura.]

## Visão Geral

[Esta subseção nomeia e define as diversas camadas e o seu conteúdo, as regras que determinam a inclusão em uma camada específica e as fronteiras entre as camadas. Inclua um diagrama de componentes que mostre os relacionamentos entre as camadas. ]

## Camadas

[Para cada camada, inclua uma subseção com o respectivo nome, uma lista dos subsistemas localizados na camada e um diagrama de componentes.]

# Visualização da Implantação

[Esta seção descreve uma ou mais configurações da rede física (hardware) na qual o software é implantado e executado. Ela é uma visão do Modelo de Implantação. No mínimo, para cada configuração, ela deve indicar os nós físicos (computadores, CPUs) que executam o software e suas interconexões (barramento, LAN, ponto a ponto, etc.) Inclui também um mapeamento dos processos da **Visualização do Processo** sobre os nós físicos.]

# Visão de Dados

[Uma descrição da perspectiva de armazenamento de dados persistentes do sistema. Esta seção será opcional se os dados persistentes forem poucos ou inexistentes ou se a conversão entre o Modelo de Design e o Modelo de Dados for trivial.]

# Tamanho e Desempenho

Volume:

1. Número estimado de usuários: Inicialmente 25, podendo crescer exponencialmente;
2. Número de acesso diários: Aproximadamente 400 logins efetuados para 25 usuários;
3. Tempo de sessão de um usuário: 30 minutos;

Performance:

1. Disponibilidade do sistema: O sistema deve estar disponível 24x7;
2. O tempo de resposta das requisições não devem ultrapassar o tempo definido por cada uma no documento de requisitos;

# Qualidade

* Escalabilidade: O sistema estará preparado para a escalabilidade de acessos que possam crescer com o tempo e com o aumento de usuários acessando;
* Disponibilidade: O sistema ficará disponível 24x7, com exceções de pequenas paradas para correções ou upgrades;
* Segurança: Usuário só acessará o sistema se estiver autenticado, o controle de sessão será feito pelo Spring security;
* Confiabilidade: Os dados serão protegidos e também possuirá backups para o caso de eventual perda de dados.