Desafio de NodeJS 2022.2

Juiz de Fora, 2022

SUMÁRIO

Especificação dos Requisitos de Software

Projeto: Desafio de NodeJS - 2022.2	
Data de criação: 30/08/2022	**Versão:** 1.0.0

Histórico de Modificação			
Data	**Versão**	**Descrição**	**Autor(es)**
30/08/2022	1.0.0	Início da elaboração do documento	Lucas de Oliveira Varino

1. Introdução:

1.1. Propósito:

O propósito do Documento de Especificação de Requisitos é delinear os requisitos do software a ser construído, descrevendo suas funcionalidades e características. O público alvo do documento são clientes, gerentes e desenvolvedores do projeto.

1.2. Escopo:

O projeto consiste em construir uma API para um sistema de Empresas Júnior, onde seja possível cadastrar membros, cargos e departamentos.

1.3. Definições e abreviações:

- **RF:** requisito funcional;
- **RNF:** requisito não funcional.

1.4. Visão geral do documento:

- **Seção 2 - Descrição Geral:** apresenta uma visão geral do sistema, especificando a perspectiva do produto e detalhamento do escopo do sistema através da discretização das funções do produto. Além disso, são explicitadas as características gerais dos usuários do produto e as restrições que poderão limitar as possibilidades de desenvolvimento.

- **Seção 3 - Requisitos funcionais:** apresentação de todos os requisitos funcionais do sistema. Descreve as principais ações do produto, considerando a aceitação e processamento das entradas e o processamento e geração das saídas.

- **Seção 4 - Requisitos não funcionais:** apresentação de todos os requisitos não funcionais do sistema. Descreve todos os aspectos qualitativos do sistema, explicitando os detalhes de facilidade de uso, manutenibilidade, confiabilidade, desempenho, segurança, distribuição, adequação a padrões e requisitos de hardware e software.

2. Descrição geral:

2.1. Funções do produto:

- CRUD de Membros
- CRUD de Departamentos
- CRUD de Cargos
- Login
- Documentação da API com Swagger

3. Requisitos funcionais:

Requisito 01 / RF001 - CRUD de Membros (40%)	
1. Descrição	O sistema deverá ter um gerenciamento de membros, sendo possível criar, visualizar, editar e deletar um membro. A rota de visualizar deve ser pública (qualquer pessoa pode ver os membros). As demais rotas podem ser acessadas apenas por pessoas que estão logadas no sistema. Utilize middlewares para fazer essa verificação. O membro é também o usuário que acessa o sistema.
2. Dados	• Id <Int> • Nome <String> • Email <String> • Senha <String> • Aniversário <Date> • Departamento <Departamento[ ]> • Cargo <Cargo>

Requisito 02 / RF002 - CRUD de Departamentos (10%)	
1. Descrição	O sistema deverá ter um gerenciamento de departamentos, sendo possível criar, visualizar, editar e deletar um departamento. A rota de visualizar deve ser pública (qualquer pessoa pode ver os departamentos). As demais rotas podem ser acessadas apenas por pessoas que estão logadas no sistema. Utilize middlewares para fazer essa verificação.
2. Dados	• Id <Int> • Nome <String>

Requisito 03 / RF003 - CRUD de Cargos (10%)	
1. Descrição	O sistema deverá ter um gerenciamento de cargos, sendo possível criar, visualizar, editar e deletar um cargo. A rota de visualizar deve ser pública (qualquer pessoa pode ver os cargos). As demais rotas podem ser acessadas apenas por pessoas que estão logadas no sistema. Utilize middlewares para fazer essa verificação.
2. Dados	• Id <Int> • Nome <String>

Requisito 04 / RF004 - Login (30%)	
1. Descrição	O sistema deve ter Login de membros no sistema, utilizando token JWT como forma de realizar o login.

Requisito 05 / RF005 - Documentação da API (Swagger) (10%)	
1. Descrição	O sistema deverá ter a documentação detalhada de todos os endpoints da API utilizando o Swagger.
2. Dados	• Todas as rotas devem ser documentadas • As rotas POST, PUT e DELETE devem ter os dados que precisam ser enviados na documentação da API.

4. Orientações:

- O relacionamento da aplicação com o banco de dados pode ser feito utilizando TypeORM ou Prisma.
- O banco de dados utilizado fica a critério do desenvolvedor, mas priorize MySQL, PostgreSQL e MongoDB.
- O login é OBRIGATÓRIO que seja feito com JWT.
- O sistema é uma API, então não é necessário fazer um front-end para consumí-lo, utilize o Insomnia ou o Postman para fazer as requisições.
- Caso tenha dificuldades com os códigos HTTP, acesse HTTP Cats ou HTTP Status Dogs
- Fica a critério do desenvolvedor a adição de bibliotecas ao sistema.
- Será disponibilizado um template para iniciar o desenvolvimento, mas o desenvolvedor pode optar por fazer quaisquer alterações na estrutura.
- Qualquer dúvida entre em contato com o gerente responsável pelo documento de requisitos ou com o diretor de projetos
- As porcentagens de cada requisito estão no título de cada tabela acima. Os membros que não estão no departamento de projetos devem entregar pelo menos 50% até a data final de entrega.
- Link para repositório base do desafio Desafio NodeJs - Github

5. Requisitos não funcionais:

Requisito não funcional / RNF001 - Códigos HTTP	
1.Descrição	O sistema deve enviar para o front o código HTTP certo para cada ocasião.

Requisito não funcional / RNF001 - Mensagens de Erro	
1.Descrição	O sistema deve enviar para o front mensagens de erro intuitivas e detalhadas, para serem devidamente tratadas.