**DECLARAÇÃO DE ESCOPO**



**PROJETO APROVAÇÕES**

**API**

**Abril / 2021**

**ÍNDICE**

1 – OBJETIVO DA CONTRATAÇÃO 3

2 – EXPECTATIVAS COM A CONTRATAÇÃO 3

3 – CONTEXTO DO PROJETO 3

4 – GLOSSÁRIO 4

5 – DESENVOLVIMENTO 4

6 – SWAGGER 4

7 – CONTROLLERS 4

7.1 – AUTH 4

7.2 – HOME 5

7.3 – APROVAÇÕES 5

7.4 – SIMULAÇÃO 6

7.5 – PROTOCOLO 6

7.6 – COTAÇÕES 6

**1 – OBJETIVO DA CONTRATAÇÃO**

Captar desenvolvedor externo para implementar as API’s em REST, utilizando o framework .NET Core 3.1 na linguagem C#.

# 2 – EXPECTATIVAS COM A CONTRATAÇÃO

|  |  |
| --- | --- |
| **Ordem** | **Descrição** |
| 1 | Redução de prazo de entrega do projeto |
| 2 | Software com a interface dentro dos padrões especificados |
| 3 | Expansão da base de desenvolvedores freelancer |
| 4 | Otimização da matriz de investimentos do projeto |

**3 – CONTEXTO DO PROJETO**

Esta API será desenvolvida para fornecer dados à um aplicativo mobile desenvolvido em Flutter. O projeto foi planejado para suprir a demanda de embarcadores que analisam os fretes que suas empresas realizam e, para isso, eles farão aprovações em seus celulares para garantir uma entrega mais efetiva aos seus clientes, seja pelo preço ou pelo prazo de entrega.

**4 – GLOSSÁRIO**

**Cotação:** Ato ou efeito de cotar, de estabelecer preço, valor. No caso, valor de frete.

**Ocorrência:**  Registro de algo que ocorreu no frete. Ex.: Cliente ausente, não foi possível realizar a entrega.

**Protocolo:** Registro de um pedido de frete na plataforma.

**3 – DESENVOLVIMENTO**

O software a ser implementado será uma API RESTFUL construída em .NET Core 3.1 WebAPI na linguagem C#.

A arquitetura da aplicação seguirá os conceitos da Onion Architecture, sendo dividida em 4 camadas (2 camadas de infra com contextos diferentes, 1 camada de domínio e 1 de apresentação).

Para a autenticação na API utilizaremos o JWT.

Os bancos de dados já existem e devem ser espelhados nas camadas de infra utilizando o Scaffold do Entity Framework CORE. Qualquer alteração no banco de dados deve ser requisitada para a equipe responsável ou contratante.

É necessário a utilização do Swagger para documentar a API.

Será disponibilizado ao desenvolvedor um documento para a consulta de convenções de endpoint separadamente e também um projeto Scrum no software Jira para o acompanhamento das Sprints semanais.

É necessário utilizar a língua inglesa como padrão no projeto.

**6 – SWAGGER**

A documentação da API é necessária para o desenvolvedor front-end ter como base as informações principais dos endpoints.

Serão exigidos:

* Descrição do endpoint;
* Payload;
* Retornos da API.

**7 – Controllers**

**7.1 – Auth**

A controller Auth é responsável pela autenticação do usuário. Nela, teremos somente o endpoint “login” que receberá as credenciais de acesso e fornecerá o JWT para o front-end com claims de permissão de acesso.

**7.2 – Home**

A controller Home será responsável por retornar a quantidade de registros na lista de cada aprovação disponível para o usuário aprovar. Exemplo: A aprovação de canal vermelho tem 3 registros para o usuário logado aprovar. Será retornado somente o número 3. Essa regra se aplica a todas as aprovações do menu.

**7.3 – Aprovações**

A controller aprovações será responsável por filtrar e listar cada tipo de aprovação com suas respectivas informações. Também serão realizadas aprovações, reprovações, solicitação de devolução, solicitação de reentrega e revalidação de ocorrência.

* Tipos de aprovação: Aprovação canal vermelho, aprovação coleta infrutífera, aprovação frete elevado, aprovação ocorrência, aprovação peso protocolo, revalidação ocorrência;
* Filtragem: Terá no payload 5 campos para que o usuário filtre as cotações para sua análise e outro endpoint separado com 2 campos no payload;
* Listagem: Com base nos filtros e no tipo de aprovação que o usuário está acessando pelo payload, serão listados os protocolos com suas respectivas informações;
* Aprovar: Endpoint específico para aprovar uma solicitação. Pode-se aprovar mais de uma solicitação;
* Reprovar: Endpoint específico para reprovar uma solicitação. Pode-se aprovar mais de uma solicitação;
* Solicitar devolução: Endpoint específico para solicitar a devolução de uma carga. Pode-se solicitar a devolução de mais de uma solicitação. Será exigido ao usuário informar se haverá custo extra;
* Solicitar reentrega: Endpoint específico para solicitar a reentrega de uma carga. Pode-se solicitar a reentrega de mais de uma solicitação. Será exigido ao usuário informar se haverá custo extra.
* Revalidação de ocorrência: Endpoint específico para revalidar a ocorrência cadastrada em uma solicitação. Pode-se revalidar a ocorrência de mais de uma solicitação.

**7.4 – Simulação**

A controller de simulação será responsável por listar as simulações de frete para os protocolos que são canal vermelho e escolher uma nova transportadora para realizar o frete. O retorno terá o nome da transportadora, a valor do frete, o prazo, o veículo que será utilizado para transportar e o nível de serviço. O usuário poderá selecionar uma transportadora da lista e, se isso ocorrer, a solicitação será reprovada automaticamente e a simulação escolhida pelo usuário será a escolhida para realizar o frete.

**7.5 – Protocolo**

A controller protocolo terá somente um endpoint. Nele, será retornado todas as informações referentes ao protocolo selecionado pelo usuário.

Teremos o seguinte endpoint:

* (GET) detalhe-protocolo/{id}

Será fornecido ao front-end informações detalhadas do protocolo selecionado. Serão 43 campos no JSON e 6 listas com no máximo 8 campos.

**7.6 – Cotações**

A controller cotações será responsável por listar, detalhar e aprovar cotações.

Teremos os seguintes endpoints:

* (GET) Listar solicitações:

Será listado as solicitações pendentes de aprovação (com o status “em aprovação”). Será retornado o código da solicitação, a cidade de origem, a cidade de destino do frete e a data de criação da solicitação;

* (GET) Detalhar solicitação/{id}:

Será fornecido ao front-end as informações detalhadas da solicitação. Serão 19 campos no JSON e uma lista com 5 campos detalhando o item transportado;

* (PUT) Alterar data limite solicitacao/{id}:

Será recebido a nova data limite para cotar e uma string dizendo o motivo da alteração da data limite;

* (GET) Listar cotacoes por solicitacao/{solicitationCode}:

Será listado as cotações de determinada solicitação com 6 campos no retorno JSON;

* (PUT) Aprovar cotacao/{id}:

Será aprovada a cotação recebida. Além de id, poderá receber o campo de justificativa de forma opcional;

* (PUT) Cancelar solicitacao/{id}:

Será cancelada a solicitação recebida. Além de id, poderá receber o campo de justificativa de forma opcional;