**TASK API-01 Análise e Condições de Teste**

**Vitor Colombo Nunes**

**Condições de Teste**

## LOGIN

**POST/login**

**CT01** – Verificar login com email e password válidos cadastrados no banco {email:”admin@qa.com.br” password: “admin”} Response: 200. message: "Login realizado com sucesso".

**CT02** – Verificar login com email inválido e password válido {email: “teste.com.br”, password: “teste”} Response: 400. email: “deve ser um email válido".

**CT03** - Verificar login com email e password válidos não cadastrados no banco {email: “teste@qa.com.br”, password: “teste”} Response: 401. message: "Email e/ou senha inválidos".

## USUÁRIOS

**GET/usuarios**

**CT04** - Verificar listagem de usuários com usuários cadastrados no banco {} Response: 200. Response body retorna para cada usuário nome, email, password, id e se o usuário é administrador.

**CT05** - Verificar listagem de usuários sem usuários cadastrados no banco {} Response: 200. Response body não retorna nada.

**POST/usuários**

**CT06** – Verificar um cadastro de usuário com dados válidos {nome: "Fulano da Silva", email: "beltrano@qa.com.br", password: "teste", administrador: "true"} Response 201. message: "Cadastro realizado com sucesso".

**CT07** – Verificar um cadastro de usuário com email inválido {nome: "Fulano da Silva", email: "beltranoqa.com.br", password: "teste", administrador: "false"} Response 400. email: "email deve ser um email válido".

**CT08** – Verificar um cadastro de usuário com email que já está sendo usado {nome: "Fulano da Silva", email: "beltrano@qa.com.br", password: "teste", administrador: "false"} Response 400. message: "Este email já está sendo usado".

**GET/usuários/\_id**

**CT09** – Verificar a busca de um usuário com id válido {\_id: “0uxuPY0cbmQhpEz1”} Response 200. Response body retorna nome, email, password, id e se o usuário é administrador.

**CT10** – Verificar a busca de um usuário com id inválido {\_id : “sdasdY0cbmQhpEz1”}. Response 400. message: "Usuário não encontrado"

**DELETE/usuários/\_id**

**CT11** – Verificar excluir usuário com id válido {\_id: 0uxuPY0cbmQhpEz1} Response 200. message: "Registro excluído com sucesso".

**CT12** – Verificar excluir usuário com id inválido {\_id: “aaaaaaaaaaaaa”} Response 400. message: "Nenhum registro excluído".

**CT13** - Verificar excluir usuário com id válido que possui um carrinho de compras {\_id: “0uxuPY0cbmQhpEz1”} Response body retorna message: "Não é permitido excluir usuário com carrinho cadastrado", idCarrinho: "DQuUXJZGx0iGyRJQ"

**PUT/usuários/\_id**

CT14 - Verificar editar usuário com id não cadastrado path{\_id: 666666666} Response 201. message: "Cadastro realizado com sucesso", \_id: "KocHFfiWGEnrCyfq".

**CT15** - Verificar editar usuário com id válido path{\_id: KocHFfiWGEnrCyfq} Response 200. message: "Registro alterado com sucesso".

**CT16** - Verificar editar usuário com id válido e novo email já cadastrado path{\_id: “KocHFfiWGEnrCyfq”} Response 400. message: " Este email já está sendo usado"

**CT31**- Verificar editar usuário com inputs válidos e usuário sem permissões de admin path{\_id: “uCF8Zc1r2jbrkG8t”} body{nome: "admin", email:[aministrador1@qa.com.br](mailto:aministrador1@qa.com.br), password: "admin", administrador: "true" } Response 200. message: "Cadastro realizado com sucesso", \_id: " uCF8Zc1r2jbrkG8t ".

## PRODUTOS

**GET/produtos**

**CT17** - Verificar listagem de produtos com produtos cadastrados no banco {} Response: 200. Response body retorna para cada produto nome, preco, descrição, quantidade, id.

**CT18** - Verificar listagem de usuários sem usuários cadastrados no banco {} Response: 200. Response body não retorna nada.

**DELETE/produtos/\_id**

**CT19** - Verificar delete de produtos com id válido e produto não adicionado em um carrinho. {\_id: “FLAe2zR55Dx8NQTu”} Response: 200. Response body recebe message: “Registro excluído com sucesso”.

**CT20** – Verificar delete de produtos com id válido e produto adicionado em um carrinho. {\_id: “K6leHdftCeOJj8BJ”} Response: 400. Response body recebe message: “Não é permitido excluir produto que faz parte de carrinho” e um array com os ids dos carrinhos que possuem o produto.

**CT21** – Verificar delete de produtos com id de produto não cadastrado. {\_id: “asdasdasd”} Response: 200. Response body recebe message: “Nenhum registro excluído”.

**CT28** – Verificar delete de produtos usuário sem permissões de admin. {\_id: “sssssss”} Response: 403. Response body recebe message: “Rota exclusiva para administradores”.

**POST/produtos**

**CT22** – Verificar a criação de um produto com os dados válidos. {"nome":"teste MX", "preco": 11, "descricao": "teste descricao", "quantidade": 381} Response: 201. No body é retornado o id do produto criado junto de uma message:” Cadastro realizado com sucesso”.

**CT23** - Verificar a criação de um produto com nome já existente em outro produto. {"nome":"teste MX", "preco": 122, "descricao": "descricao", "quantidade": 1} Response: 400. No body é retornada uma message: “Já existe um produto com esse nome”.

**CT24** – Verificar a criação de um produto com autenticação expirada. {"nome":"teste 2", "preco": 312, "descricao": "teste 2", "quantidade": 3} Response: 401. É recebido como resposta message: “Token de acesso ausente, inválido, expirado ou usuário do token não existe mais”.

**CT27** - Verificar a criação de um produto logado com usuário sem permissões de admin. {"nome":"teste 2", "preco": 312, "descricao": "teste 2", "quantidade": 3} Response: 403. É recebido como resposta message: “Rota exclusiva para administradores”.

**GET/produtos/\_id**

**CT25** - Verificar listagem de produtos com id válido. {\_id: “K6leHdftCeOJj8BJ”} Response: 200. Response body retorna nome, preco, descrição, quantidade e id do produto com o id de entrada.

**CT26** - Verificar listagem de produtos com id não cadastrado no banco. {\_id: “asdasdasdasd”} Response: 400. Reposta com message: “Produto não encontrado”.

**PUT/produtos/\_id**

**CT29** - Verificar a edição de um produto estando logado com usuário sem permissões de admin. {"nome":"teste edicao", "preco": 32, "descricao": "teste 3", "quantidade": 3} Response: 403. É recebido como resposta message: “Rota exclusiva para administradores”.

**CT30** - Verificar a edição de um produto estando logado com usuário com permissões de admin e nome já existente em outro produto cadastrado. {nome:"testeduplicado", preco: 2, descricao: "teste 3", quantidade: 3} Response: 400. É recebido como resposta message: “Já existe produto com esse nome”.

**CT32** - Verificar a edição de um produto estando logado com usuário com permissões de admin e id não encontrado no cadastro. {nome:"teste novo produto", preco: 222, descricao: "teste cria na att", quantidade: 3} Response: 201. É recebido no body como resposta message: “Cadastro realizado com sucesso” junto do id gerado para o novo produto.

## CARRINHOS

**GET/carrinhos**

**CT33** - Verificar listagem de carrinhos com carrinhos em aberto cadastrados{} Response: 200. Response body retorna para cada carrinho um array de produtos com quantidade, id e precoUnitario, além do precoTotal e quantidadeTotal do carrinho, idUsuario, id(carrinho).

**CT34** - Verificar listagem de carrinhos com carrinhos em aberto cadastrados usando o idUsuario como filtro. {idUsuario: “0uxuPY0cbmQhpEz1”} Response: 200. Response body retorna para o carrinho do usuário filtrado com um array de produtos com quantidade, id e precoUnitario, além do precoTotal e quantidadeTotal do carrinho, idUsuario, id(carrinho).

**POST/carrinhos**

**CT33** - Verificar criação de carrinhos com usuário não possuindo um carrinho{produtos:[{idProduto: “BeeJh5lz3k6kSIzA”, quantidade: 1},{idProduto: “KMcy5ydcimk2EU33”, quantidade: 1}]} Response: 201. Response body retorna message: “Cadastro realizado com sucesso” junto do id do carrinho.

**CT34** - Verificar criação de carrinhos com usuário já possuindo um carrinho{produtos:[{idProduto: “BeeJh5lz3k6kSIzA”, quantidade: 1}]} Response: 400. Response body retorna message: “Não é permitido ter mais de 1 carrinho”.

**CT35** - Verificar criação de carrinhos com usuário não possuindo um carrinho e quantidade de produto maior que o estoque disponível{produtos:[{idProduto: “BeeJh5lz3k6kSIzA”, quantidade: 100000}]} Response: 400. Response body retorna message: “Produto não possui quantidade suficiente” junto das informações do item em questão(idProduto, quantidade, quantidadeEstoque, index).

**GET/carrinhos/\_id**

**CT36** - Verificar listagem de carrinho pelo id válido{\_id: DQuUXJZGx0iGyRJQ} Response: 200. Response body retorna um array de produtos com quantidade, id e precoUnitario, além do precoTotal e quantidadeTotal do carrinho, idUsuario, id(carrinho) para o carrinho com o id passado.

**CT37** - Verificar listagem de carrinho pelo id inválido{\_id: asdasdasdasd} Response: 400. Response body retorna message: “Carrinho não encontrado”.

**DELETE/carrinhos/concluir-compra**

**CT38** – Verificar deletar carrinho a partir da conclusão da compra sem possuir carrinho relacionado ao idUsuario. {}. Response: 200. Response body retorna message: “Não foi encontrado carrinho para esse usuário”.

**CT39** – Verificar deletar carrinho a partir da conclusão da compra possuindo carrinho relacionado ao idUsuario. {}. Response: 200. Response body retorna message: “Registro excluído com sucesso”.

**DELETE/carrinhos/cancelar-compra**

**CT40** – Verificar deletar carrinho a partir do cancelamento da compra sem possuir carrinho relacionado ao idUsuario. {}. Response: 200. Response body retorna message: “Não foi encontrado carrinho para esse usuário”.

**CT41** – Verificar deletar carrinho a partir do cancelamento da compra possuindo carrinho relacionado ao idUsuario. {}. Response: 200. Response body retorna message: “Registro excluído com sucesso. Estoque dos produtos reabastecido”.

## Revisão

As respostas obtidas por **get/usuarios** e **get/usuarios/\_id** mostram o password dos usuários.

Nas requisições em **put/usuarios** está sendo permitido que o usuário, mesmo sem permissões de administrador, possa alterar o campo de administrador.

Em **get/produtos/\_id** e **get/usuarios/*id*** *o* erro ao não encontrar poderia ser trocado por 404 no lugar de do recebido (400).

Nas requisições de **delete/usuarios/\_id** o código de status poderia ser 204 em vez de 200.

Em **post/login** entrando com um email inválido, a resposta não retorna dentro de message, mas sim em email.

Existe um carrinho que não possui dono pré-cadastrado no sistema

Preço total do carrinho e preço do produto não deveriam ser integer para poderem receber números com maior precisão.