

## **1º Cenário:**

### **1.1 Materiais de Apoio**

**1.1.1** Identificação da Documentação: Documentação Oficial do próprio Marketplace (APIs), requisitos de negócio e funcionais, especificações técnicas da integração. Encontrei em:

[https://developers.mercadolivre.com.br/pt\\_br/produto-sincronizacao-de-publicacoes](https://developers.mercadolivre.com.br/pt_br/produto-sincronizacao-de-publicacoes)

**1.1.2** Análise da Documentação: Buscar palavras-chave, elaborar tópicos, definir prioridade, entender o fluxo do processo, analisar endpoints e payloads, leitura crítica dos requisitos.

**1.1.3** Mapeamento dos Requisitos: Será feita em conjunto com a tarefa anterior, analisando os tópicos, requisitos e elaborando uma matriz de rastreabilidade que relacione os dados.

**1.1.4** Utilização de Ferramentas: Para organização e administração, pode-se utilizar Jira ou Trello. Para o teste de API, pode-se utilizar Postman. Para automatização dos testes, pode-se utilizar Selenium (a ferramenta).

### **1.2 Abrangência dos Testes**

**1.2.1** Funcionalidades: Deverão ser testadas todas as funcionalidades, por exemplo integrações (estoque, pedidos, faturamento, anúncios). Também sincronização do estoque, faturamento e envio de notas, atualização de preços, importação de pedidos (se tiver).

#### **1.2.2 Casos de Uso**

##### **1.2.2.1 Estoque:**

- > Sincronização bem-sucedida de estoque ao alterar a quantidade;
- > Falha ao sincronizar estoque com valor inválido (ex.: negativo);
- > Teste com múltiplos updates simultâneos;

##### **1.2.2.2 Anúncios**

- > Criação de novo anúncio via integração;
- > Atualização de título, descrição e imagens;
- > Falha por obrigatoriedade dos campos;

##### **1.2.2.3 Pedidos**

- > Importação de pedido válido;
- > Pedido duplicado;
- > Pedido com item fora de estoque;

##### **1.2.2.4 Faturamento**

- > Geração automática de nota fiscal com dados corretos;
- > Falha por CNPJ inválido ou dados fiscais incompletos;

##### **1.2.2.5 Preço**

- > Alteração de preço com sucesso;
- > Validação de limites de preço (mínimo e máximo);

**1.2.3** Priorização dos Testes: Deverão ser testadas primeiro as rotinas que geram mais impacto no sistema, ou seja, aquelas que são mais genéricas e mais usadas, para evitar o máximo de gargalos possíveis. Também considerar custos, priorizar o que o que poderia gerar mais prejuízo, se for o caso. Muito provavelmente: Pedidos, Estoque e Faturamento.

### **1.3 Execução dos Testes**

**1.3.1** Ambiente de Teste: Ambiente de Homologação;

**1.3.2** Dados de Teste: Dados fictícios de clientes, produtos, códigos de produtos (SKU), valores, CNPJ e demais dados para nota fiscal. Pode ser usado um script SQL pronto para facilitar a inserção dos dados.

**1.3.3 Ferramentas de Automação:** Postman para as APIS, Selenium para os testes (estes têm interface).

**1.3.4 Registro de Resultados:** Planilhas, relatórios ou JIRA.

## **2º Cenário:**

### **2.1 Documentação e Materiais de Apoio**

**2.1.1 Identificação da Documentação:** Documentação oficial da API Bling, requisitos, especificações técnicas da integração. Encontrei em:

<https://developer.bling.com.br/bling-api#introdu%C3%A7%C3%A3o>

**2.1.3 Mapeamento dos Requisitos:** Será feito analisando os tópicos, requisitos e elaborando uma matriz de rastreabilidade que relacione os dados.

**2.1.4 Utilização de Ferramentas:** Para organização e administração, pode-se utilizar Jira ou Trello. Para o teste de API, pode-se utilizar Postman. Para automatização dos testes, pode-se utilizar Selenium, TestRail ou Zephyr.

### **2.2 Abrangência dos Testes**

**2.2.1 Funcionalidades:** Testar sincronização e atualização dos produtos, estoque, pedidos, verificar faturamento (preços, notas fiscais), e emissão de relatórios.

**2.2.3 Priorização dos Testes:** Deverão ser testadas primeiro as rotinas que geram mais impacto no sistema, ou seja, aquelas que são mais genéricas e mais usadas, para evitar o máximo de gargalos possíveis. Também considerar custos, priorizar o que o que poderia gerar mais prejuízo, se for o caso. Muito provavelmente: Pedidos, Estoque e Faturamento.

### **2.3 Execução dos Testes**

**2.3.2 Dados de Teste:** Dados fictícios de clientes, produtos, códigos de produtos (SKU), valores, CNPJ e demais dados para nota fiscal. Pode ser usado um script SQL pronto para facilitar a inserção dos dados.

**2.3.3 Ferramentas de Automação:** Postman para as APIS, Selenium para os testes (estes têm interface).

**2.3.4 Registro de Resultados:** Planilhas, relatórios ou JIRA.

## **3º Cenário:**

-> Verificar Documentação, disponível em: API Docs >> Mercado Livre >> Introdução >> Sincronização e modificação de publicações;

-> Pesquisar principais palavras relacionadas, por exemplo: Estoque, atualização, situação, produtos, pausado...

-> Testar criando um anúncio de teste, com o estoque do produto = 1;

-> Verificar como é feita a alteração da situação;

-> Realizar uma compra de teste, zerando o estoque;

-> Verificar o que ocorreu com a situação, se acontecer o mesmo problema relatado, é provável que o erro esteja no envio da API;

Após analisar a documentação, foi constatado que o erro pode estar no envio dos dados pela API, onde a quantidade não está sendo enviada para o marketplace, e quando a quantidade do estoque estiver como "0", não está sendo adicionado o subestado "out\_of\_stock", apenas o estado "paused";

## Capturas de tela abaixo, deste trecho da documentação:

**Paused:** pode se apresentar de forma automática (out of stock) ou por decisão do usuário.

- out of stock: o item foi **pausado** por falta de estoque e será automaticamente ativado quando for restituído, para evitar isso, o item deverá ser **pausado** com estoque =1.
- picture downloading pending: o item foi **pausado** até que a imagem seja baixada corretamente e será ativado quando isso aconteça.

**pausado:** **pausa** sua publicação. Uma vez **pausado**, o produto não poderá ser visualizado pelos outros usuários do Mercado Livre, mas não será encerrado e poderá ser reativado depois.

Exemplo:

```
curl -X PUT -H 'Authorization: Bearer $ACCESS_TOKEN' -H 'Content-Type: application/json' -H 'Accept: application/json' -d
{
  "status": "paused"
}
https://api.mercadolibre.com/items/$ITEM_ID
```

**ativo:** reativa um produto previamente **pausado**.

Exemplo:

```
curl -X PUT -H 'Authorization: Bearer $ACCESS_TOKEN' -H 'Content-Type: application/json' -H 'Accept: application/json' -d
{
  "status": "active"
}
https://api.mercadolibre.com/items/$ITEM_ID
```



### Notas:

O valor diferencia entre letras maiúsculas e minúsculas e deve ser enviado em letras minúsculas.

## 4º Cenário:

Executar os seguintes testes:

### Rotina Consultar Pessoas >> Ação Alterar Pessoa >> Aba Geral

#### Campo Nome

- > Verificar se o campo permite valor null, ou em branco, ou espaço (ex. '');
- > Verificar se aceita caracteres especiais (ex. #);
- > Verificar se aceita números;
- > Verificar se tem quantidade máxima de caracteres permitidos;
- > Verificar se o campo salva maiúsculas e minúsculas distintas;
- > Avançar para o próximo campo utilizando tab, para verificar se obriga o preenchimento;
- > Inserir dados e clicar Enter, para verificar se foca/avança para o próximo campo;

#### Campo E-mail

- > Verificar se o campo exige o formato usuario@dominio.extensao;
- > Verificar se obriga que tenha o caracter "@" presente na informação digitada;
- > Verificar se permite números antes e depois do "@";
- > Verificar se permite caracteres especiais antes e depois do "@";
- > Verificar se permite deixar em branco e confirmar;

- > Avançar para o próximo campo utilizando tab, para verificar se o comportamento do campo muda;
- > Inserir dados e clicar Enter, para verificar se foca/avança para o próximo campo;

### **Campo Data de Nascimento**

- > Verificar se permite inserir anos anteriores (ex. 01/01/1500);
- > Verificar se permite informar data inválida (ex. 45/26/2987);
- > Verificar se permite datas futuras (ex. 01/01/2030);
- > Verificar se permite inserir dados errados através de CTRL + C + V;
- > Verificar se permite selecionar data através do botão seletor;
- > Verificar se permite inserir letras ou caracteres especiais;
- > Avançar para o próximo campo utilizando tab, para verificar se o comportamento do campo muda;
- > Inserir dados e clicar Enter, para verificar se foca/avança para o próximo campo;

## **Rotina Consultar Pessoas >> Ação Alterar Pessoa >> Aba Contatos**

### **Número de Telefone**

- > Verificar se aplica a máscara conforme digita os números;
- > Verificar se aplica a máscara utilizando CTRL + C + V;
- > Verificar se obriga inserir DDD;
- > Verificar se obriga inserir código de área (ex. +55 - Brasil);

## **Rotina Consultar Pessoas >> Ação Alterar Pessoa >> Aba Endereços**

### **Endereço (com campos para rua, cidade, estado e CEP)**

- > Verificar se preenche os demais campos a partir da inserção do CEP;
- > Verificar se os dados obtidos estão corretos, através de pesquisa na internet;
- > Verificar se permite alterar os campos, e se sim, se permite inserir local errado;

### **Campo CEP**

- > Verificar se aplica a máscara;
- > Verificar se limita a quantidade de caracteres para 9 (ex. 89188-000);
- > Verificar se obriga o preenchimento dos demais campos também, ou verificar qual o comportamento desejado dos outros campos a partir deste;
- > Verificar se permite valor null, ou em branco, ou espaço (ex. ' ');
- > Verificar se permite letras ou caracteres especiais;

### **Campo Rua**

- > Sugerir alteração para "logradouro", se for o caso;
- > Verificar se permite valor null, ou em branco, ou espaço (ex. ' ');
- > Verificar se aceita caracteres especiais (ex. #);
- > Verificar se aceita números (ex. Rua 15 de Novembro);
- > Verificar se tem quantidade máxima de caracteres permitidos;
- > Verificar se o campo salva maiúsculas e minúsculas distintas;
- > Avançar para o próximo campo utilizando tab, para verificar se obriga o preenchimento;

-> Inserir dados e clicar Enter, para verificar se foca/avança para o próximo campo;

### **Campo Cidade**

- > Verificar se condiz com o CEP;
- > Verificar se permite valor null, ou em branco, ou espaço (ex. ' ');
- > Verificar se aceita caracteres especiais (ex. #);
- > Verificar se aceita números;
- > Verificar se tem quantidade máxima de caracteres permitidos;
- > Verificar se o campo salva maiúsculas e minúsculas distintas;
- > Avançar para o próximo campo utilizando tab, para verificar se obriga o preenchimento;
- > Inserir dados e clicar Enter, para verificar se foca/avança para o próximo campo;

### **Campo Estado**

- > Verificar se condiz com o CEP;
- > Verificar se exibe as opções da lista corretamente;
- > Avançar para o próximo campo utilizando tab, para verificar se obriga o preenchimento;