

Gonçalo Almeida, aluno Nº 2020218868

Guilherme Branco, aluno Nº 2020216924

João Santos, aluno Nº 2020218995

**Plataforma de Vendas Online**

Projeto elaborado no âmbito da unidade curricular de Bases de Dados

Maio de 2022

Índice

[1. Manual do Utilizador 2](#_Toc103953257)

[a. Descrição do sistema 2](#_Toc103953258)

[b. Níveis de acesso dos utilizadores 2](#_Toc103953259)

[c. *Endpoints* disponíveis 2](#_Toc103953260)

[ Para todos os utilizadores 2](#_Toc103953261)

[ Para administradores 6](#_Toc103953262)

[ Para vendedores 6](#_Toc103953263)

[ Para compradores 6](#_Toc103953264)

[2. Manual de Instalação 7](#_Toc103953265)

[a. Pré-requisitos 7](#_Toc103953266)

[ Software necessário 7](#_Toc103953267)

[ Python packages 7](#_Toc103953268)

[b. Processo de instalação 7](#_Toc103953269)

[c. Desinstalação 7](#_Toc103953270)

[3. Pormenores da implementação 8](#_Toc103953271)

[4. Plano de desenvolvimento 8](#_Toc103953272)

# Manual do Utilizador

## Descrição do sistema

Esta plataforma permite a comercialização de diferentes tipos de equipamentos eletrónicos (“computers”, “televisions” e “smartphones”). Cada produto é vendido por uma empresa específica e é caracterizado por um identificador único (id) e outros atributos específicos do produto além dos genéricos “name”, “stock”, “description” e “price”.

Sempre que há atualização dos detalhes de um produto é criada uma nova versão que mantém os detalhes não alterados e os novos, sendo assim criado um histórico das versões anteriores.

## Níveis de acesso dos utilizadores

O tipo de utilizador com maior poder hierárquico é o admin, com permissões de moderação e gestão da plataforma. Apenas um admin pode criar vendedores (“sellers”) e outros admins e criar campanhas promocionais. Outros tipos de utilizadores também têm acesso a operações exclusivas:

Um user do tipo “seller” pode adicionar e atualizar os produtos que vende.

Um user do tipo “buyer” pode efetuar compras, deixar rating/feedback de um produto que tenha comprado e subscrever a campanhas promocionais.

Qualquer utilizador pode consultar estatísticas da plataforma relativas a vendas e campanhas promocionais.

## *Endpoints* disponíveis

### Para todos os utilizadores

#### Registo de utilizadores

Qualquer pessoa pode registar-se como “buyer” sem ter realizado *login*. Apenas os administradores podem criar “sellers” e outros admins.

**POST** [**http://localhost:8080/dbproj/user**](http://localhost:8080/dbproj/user)

Payload JSON para “buyer”:

{

    "username": "username",

    "email": "mail@mail.com",

    "password": "password",

    "type": "buyers",

    "nif": 123456789,

    "home\_addr": "Addr St. 20"

}

Payload JSON para “seller”:

{

    "username": "username",

    "email": "mail@mail.com",

    "password": "password",

"type": "sellers",

    "nif": 123456789,

    "shipping\_addr": "Addr St. 30"

}

Payload JSON para “admin”:

{

    "username": "username",

    "email": "mail@mail.com",

    "password": "password",      "type": "admins"

}

#### Autenticação

Fornecendo os dados de *login* irá receber um *token* de autenticação válido durante 20 minutos que deve ser enviado em pedidos seguintes no *header* “Authorization” com o valor “Bearer {colocar *token* aqui}”.

**PUT** [**http://localhost:8080/dbproj/user**](http://localhost:8080/dbproj/user/)

Payload JSON:

{

    "username": "username",

    "password": "password"

}

Resultado:

{

    "status": 200,

    "token": {token}

}

#### Consultar informações de um produto

Permite obter o nome, o stock, a descrição, o histórico de preços por versão, a média do rating e os comentários feitos ao produto com ID {product\_id}.

**GET** [**http://localhost:8080/dbproj/product/{product\_id}**](http://localhost:8080/dbproj/product/%7bproduct_id%7d)

Resultado:

{

    "results": {

        "comments": [

            "Comment 1",

            "Comment 2",

[...]

        ],

        "description": "generic product description",

        "name": "This Product’s Name",

        "prices": [

            "{price value} - YYYY-MM-DD HH:MM:SS",

"{price value} - YYYY-MM-DD HH:MM:SS",

[...]

        ],

        "rating": "{média de classificação}",

        "stock": {número de artigos disponíveis}

    },

    "status": 200

}

#### Deixar comentário/pergunta

Permite deixar um comentário/pergunta em relação a um produto com ID {product\_id}.

**POST** [**http://localhost:8080/dbproj/questions/{product\_id}**](http://localhost:8080/dbproj/questions/%7bproduct_id%7d)

É possível também responder a uma pergunta com ID {parent\_question\_id} no produto, usando a seguinte alternativa:

**POST** [**http://localhost:8080/dbproj/questions/{product\_id}/{parent\_question\_id}**](http://localhost:8080/dbproj/questions/%7bproduct_id%7d/%7bparent_question_id%7d)

Payload JSON:

{

    "question": "{texto do comentário}"

}

Resultado:

{

    "status": 200,

    "results": {ID do comentário}

}

#### Obter estatísticas mensais de vendas dos últimos 12 meses

Permite obter os detalhes das vendas mensais dos últimos 12 meses: o número de encomendas e o seu custo total para cada mês.

**GET** [**http://localhost:8080/proj/report/year**](http://localhost:8080/proj/report/year)

Resultado:

{

    "results": [

        {

            "month": "MM-YYYY",

            "orders": {nº de compras},

            "total\_value": "{valor total}"

        },

        {

            "month": "MM-YYYY",

            "orders": {nº de compras},

            "total\_value": "{valor total}"

        },

        [...]

    ],

    "status": 200

}

#### Obter estatísticas de descontos aplicados por campanha

Permite obter uma lista das campanhas com a sua informação: o número de cupões emitidos, utilizados, e o valor total dos descontos aplicados.

**GET** [**http://localhost:8080/dbproj/report/campaign**](http://localhost:8080/dbproj/report/campaign)

Resultado:

{

    "results": [

        {

            "campaign\_id": {ID da campanha},

            "generated\_coupons": {nº de cupões gerados},

            "total\_discount\_value": {valor total de descontos aplicados},

            "used\_coupons": {nº de cupões usados}

        },

[...]

    ],

    "status": 200

}

#### Consultar notificações

Permite ao utilizador consultar as notificações que lhe são dirigidas.

**GET** [**http://localhost:8080/dbproj/inbox**](http://localhost:8080/dbproj/inbox)

Resultado:

{

    "results": [

        [

            "WEEKDAY, DD MONTH YYYY HH:MM:SS GMT",

            {ID da notificação},

            "{Texto da notificação}"

        ],

        [

            "WEEKDAY, DD MONTH YYYY HH:MM:SS GMT",

            {ID da notificação},

            "{Texto da notificação}"

        ],

[...]

    ],

    "status": 200

}

### Para administradores

#### Criar campanha promocional

Permite criar uma campanha, tendo em conta que nunca pode existir mais que uma campanha ativa. Os compradores devem subscrever à campanha dentro do intervalo no qual está ativa para obterem cupões válidos durante 30 dias.

**POST** [**http://localhost:8080/dbproj/campaign**](http://localhost:8080/dbproj/campaign)

Payload JSON:

{

    "description": "{descrição da campanha}",

    "date\_start": "YYYY-MM-DD",

    "date\_end": "YYYY-MM-DD",

    "coupons": {nº de cupões},

    "discount": {% de desconto}

}

### Para vendedores

#### Adicionar um produto

Permite colocar um produto para venda, sendo que é necessário fornecer todos os seus detalhes.

**POST** [**http://localhost:8080/dbproj/product**](http://localhost:8080/dbproj/product)

Payload JSON:

{

    "description": "{descrição}",

    "type": "{tipo de produto}",

    "price": {preço unitário},

    "stock": 6{nº de itens em stock},

    "name": "{nome do produto}",

    {“attribute”: “value” para outros detalhes}

}

Resultado:

{

    "results": "{ID do produto}",

    "status": 200

}

#### Atualizar informação de um produto

Permite atualizar os detalhes de um produto com ID {product\_id}. Os novos detalhes são adicionados à base de dados com um novo atributo de versão.

**PUT** [**http://localhost:8080/dbproj/product/{product\_id}**](http://localhost:8080/dbproj/product/%7bproduct_id%7d)

Payload JSON:

{

    {pares "detail\_name": "{novo valor}"}

}

### Para compradores

#### Efetuar uma compra

Permite realizar uma compra. Se for incluído o ID de um cupão, o seu desconto é aplicado.

**POST** [**http://localhost:8080/dbproj/order**](http://localhost:8080/dbproj/order)

Payload JSON:

{

    "cart": [

        {

            "product\_id": {ID do produto},

            "quantity": {nº de unidades}

        },

        {

            "product\_id": {ID do produto},

            "quantity": {nº de unidades}

        },

[...]

    ],

    "coupon": {ID do cupão} (opcional)

}

#### Deixar rating/feedback

Permite atribuir um *rating* de 1 a 5 a um produto com ID {product\_id} que tenha sido comprado.

**POST** [**http://localhost:8080/dbproj/**](http://localhost:8080/dbproj/)**rating/{product\_id}**

Payload:

{

    "rating": {valor de 1 a 5},

    "comment": "{comentário do rating"

}

#### Subscrever campanha

Permite obter acesso a um cupão de desconto da campanha com ID {campaign\_id}. Os cupões obtidos têm um prazo de 30 dias.

**PUT** [**http://localhost:8080/dbproj/subscr**](http://localhost:8080/dbproj/subscr)**ibe/{campaign\_id}**

Resultado:

{

    "results": {

        "coupon\_id": {ID do cupão},

        "expiration\_date": "YYYY-MM-DD HH:MM:SS"

    },

    "status": 200

}

# Manual de Instalação

## Pré-requisitos

Nesta secção é apresentada uma lista de pré-requisitos de instalação que devem ser atendidos antes da instalação.

### Software necessário

1. PostgresSQL versão 14
2. *Python* versão 3.9+
3. Postman versão 9.18.2

### 

### Python packages

Para a utilização do *script* “api.py” são necessários os seguintes packages:

1. Flask versão 2.1.2;
2. psycopg2-binary versão 2.9.3;
3. PyJWT versão 2.3.0;
4. cryptography versão 37.0.2;

## Processo de instalação

Após veriricar que reúne todos os requisitos, pode proceder à instalação. Para criar a DB deve executar o *script* “dbproj\_create.sql” na linha de comandos da seguinte forma:

psql -h localhost -U postgres -f dbproj\_create.sql postgres

Este *script*:

* Cria a DB com o nome “dbproj”;
* Cria um utilizador com o nome “projuser” e a password “projuser”;
* Cria as tabelas e funções;
* Insere alguns dados-exemplo nas tabelas.

## Desinstalação

Para desinstalar a DB “dbproj” e remover o utilizador “projuser”, execute o *script* “dbproj\_drop.sql”:

psql -h localhost -U postgres -f dbproj\_drop.sql postgres

# Pormenores da implementação

## Segurança da conexão à base de dados

A palavra-passe usada para aceder à base de dados encontra-se encriptada usando uma chave fernet fornecida num ficheiro. É desencriptada em cada momento de conexão.

O utilizador criado para aceder à base de dados tem permissões limitadas: apenas pode conectar-se e realizar as operações *select, update* e *insert*, que são as necessárias para as funcionalidades implementadas. Foi também certificado que novos utilizadores criados não adquirem automaticamente permissões.

## Autenticação de utilizadores

As *passwords* dos utilizadores são armazenadas na base de dados de forma encriptada.

A autenticação é realizada através de *tokens* JWT. Ao realizar o *login* é gerada uma *token* com o ID de utilizador, usando uma chave associada à aplicação Flask, com o momento de criação definido e com uma audiência definida como o nome de sessão da aplicação (o que torna a *token* descodificável apenas por essa sessão). É também definido um momento de expiração para a *token*, pelo que uma sessão de *login* de um utilizador é válida por 20 minutos.

(ER e Relational data models)

(descrições mais detalhadas do q fizemos)

(EXPLICAR TRIGGERS IMPLEMENTADOS)

# Plano de desenvolvimento

(cronograma, que tarefas foram feitas por cada um, horas de cada um)