APLICAÇÃO DE MONITORIZAÇÃO

Desenvolvimento de Aplicações Empresariais

SÍNTESE

Este projeto tem como objetivo implementar e testar uma aplicação empresarial de monitorização de embalagens inteligentes.

Miguel Venâncio Crespo - 2222046
Bernardo José Mendes Lopes - 2222048
José Miguel Delgado - 2222049

Índice

Introdução

Aplicação de Monitorização

A Aplicação de Monitorização permite recolher informação de sensores de monitorização integrados nas embalagens de alguns produtos ou encomendas. Para além de receber informação dos sensores a Aplicação de Monitorização, faz a comunicação, e dá resposta a todas as funcionalidades dos diversos sistemas, exceto ao Sistema de E-Commerce, que para fins académicos será simulado internamente no nosso ambiente de desenvolvimento back-end.

Sistema de E-Commerce

Refere-se à plataforma de comércio eletrónico, onde o cliente realiza a compra.

Está ligado à Logística, indicando quando o cliente faz uma encomenda, as informações dos produtos e do utilizador são enviadas para a logística, para serem tratadas posteriormente pela Aplicação de Monitorização.

Sistema de Apoio ao Cliente

Representa o Serviço de Apoio ao Cliente, que se liga diretamente à Aplicação de Monitorização. Aqui, o cliente pode consultar o estado, a localização entre outros detalhes das suas encomendas, bem como, fazer o cancelamento das encomendas que ainda estão numa fase de processamento.

O Cliente poderá consultar o histórico de todas as encomendas, “por entregar”, “em processamento”, “entregue” e “cancelada” sendo que pode receber alertas quando existe um inconveniente com as suas encomendas por entregar.

O Cliente terá acesso a um histórico de leituras feitas pelos sensores de uma determinada categoria, que abrange todas as suas encomendas.

Sistema de Logística

Responsável por fazer a associação de um sensor a um determinado volume de uma encomenda existente, fazer a criação de uma nova encomenda, o que implica a escolha de todos os produtos da encomenda e a seleção do utilizador.

Neste sistema é onde é feita a leitura do sensor que ficará associado a um volume, podem consultar-se os detalhes de um determinado volume e de uma encomenda especifica. Como extra decidimos que cada volume de uma encomenda terá sempre associado um sensor de GPS, pois os clientes e os gestores conseguirão acompanhar de forma visual a geolocalização dos volumes da mesma.

O Sistema de Logística permite recolher a lista de encomendas com o estado “por entregar”, para que quando uma determinada encomenda chega ao seu ponto de destino, possa alterar o seu estado para “entregue” e informar a Aplicação de Monitorização para fazer a atualização dessa informação, bem como, a desativação dos sensores associados ao(s) volume(s) dessa mesma encomenda.

Sistema Operacional

Interage com a Aplicação de Monitorização para gerir e supervisionar detalhadamente as operações do sistema, como a monitorização de volumes, sensores e encomendas.

Será possível ver alertas rigorosos das diversas encomendas em distribuição, com a possibilidade de verificar a última leitura registada pelos sensores de uma categoria especifica e ainda ativos.

Sistema de Sensores

O Sistema de Sensores monitoriza aspetos como temperatura, localização, entre outros dados, e enviam essa informação diretamente para a Aplicação de Monitorização, assegurando que as encomendas estão nas condições corretas durante o transporte. Com a constante monitorização dos dados dos sensores, a cada leitura enviada o sensor perde 1% de bateria, logo quando o sistema deteta um sensor com um nível baixo de bateria, informa a Aplicação de Monitorização para que o mesmo seja desativado.

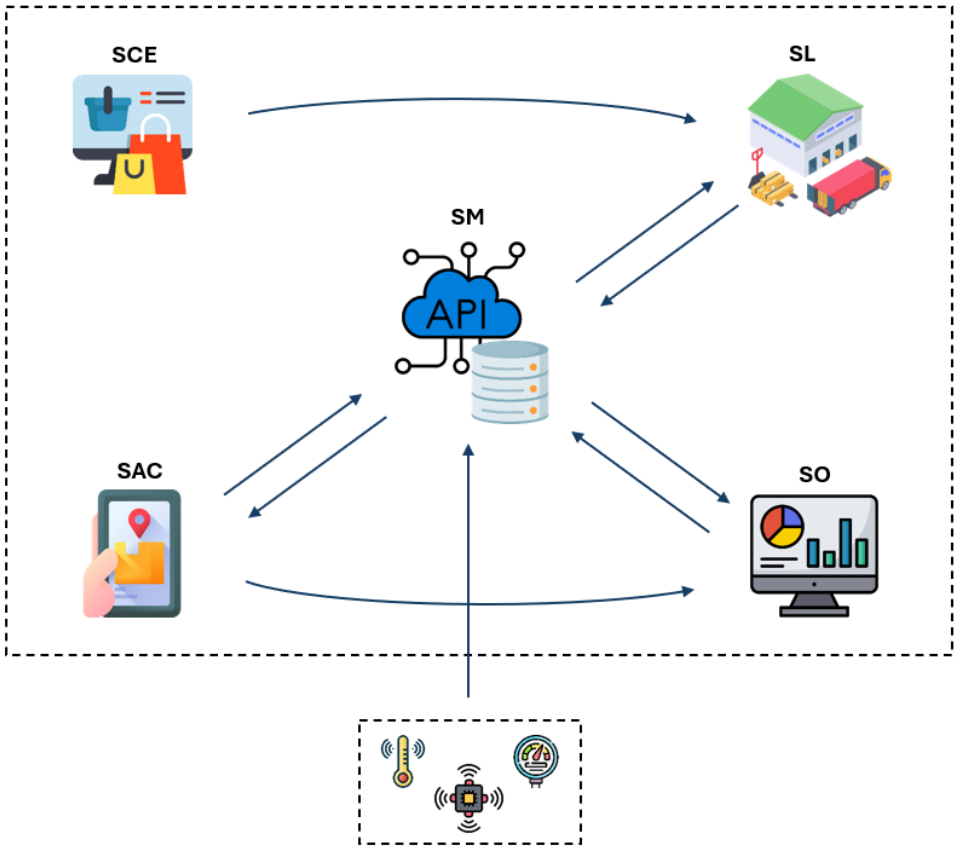
Mapa Lógico dos Sistemas

A Figura 1 - Mapa Lógico dos Sistemas ilustra todos os sistemas presentes no cenário da empresa AmazonJBM, como o Sistema de Monotorização (SM), Sistema de Comercio Eletrónico (SCE), Sistema de Logística (SL), Sistema Operacional (SO), Sistema de Apoio ao Cliente (SAC), e os sensores que estarão fora da empresa, bem como as respetivas comunicações entre eles. Na realização deste projeto apenas iremos desenvolver os sistemas que comunicam diretamente com o sistema de monotorização.

No centro podemos observar o nosso sistema de monotorização, nele estará presenta a API juntamente com a base de dados da empresa. A API dará suporte à integração dos sistemas, permitindo que os mesmos comuniquem entre si de forma eficiente e fluida, utilizando o protocolo HTTP. A base de dados da empresa guardará os diversos dados da mesma, como os dados dos utilizadores, todas as encomendas, os produtos existentes, os tipos de sensores, etc.

Na figura, também podemos observar os sensores que estarão localizados fora da empresa e que terão comunicação direta com o sistema de monitorização, este, por sua vez, será responsável por armazenar os dados capturados pelos sensores e por notificar os diferentes sistemas sobre quaisquer alterações nesses dados. Na prática a simulação dos sensores será realizada através de uma página front-end, nela será possível alterar os dados dos mesmos para efeitos de teste.

Figura 1 - Mapa Lógico dos Sistemas



Extras Desenvolvidos

Os sistemas integram diversas funcionalidades extras de forma melhorar o projeto como um todo. Entre estas funcionalidades destacam-se a importação de dados através de ficheiros CSV, a implementação do frontend com design responsivo para todos os tipos e tamanhos de ecrãs, a criação de alguns endpoints que ajudam a criação de novos produtos, de novos tipos de embalagens e de novos sensores e a possibilidade dos gestores e clientes conseguirem ver a localização exata das embalagens que contenham um sensor de GPS.

Importação de dados através de CSV

A funcionalidade de importação de dados permite carregar grandes volumes de informação de forma rápida e segura. Através de ficheiros CSV é possível importar listas completas de encomendas, de volumes, de embalagens, de produtos, de utilizadores, entre outros, reduzindo o risco de erro manual e acelerando o processo de configuração.

Frontend responsivo

A implementação de um frontend com design responsivo permite que o site se adapte a todos os tipos e tamanhos de ecrãs, isso irá trazer diversas vantagens aos trabalhadores da empresa como por exemplo, os trabalhadores da logística podem utilizar tablets para facilitar o seu trabalho, melhorando a eficiência e a mobilidade no processo de gestão de encomendas, volumes, embalagens e sensores, com acesso rápido e prático á aplicação.

Criação de dados em tempo real

A possibilidade de criar novos produtos, tipos de embalagens e sensores facilita o trabalho dos trabalhadores da logística e aumenta a eficiência da empresa. Isso permite a adição de novos dados em tempo real, sem a necessidade de alterações manuais no sistema de monitorização, garantindo agilidade e flexibilidade.

Especificação da API REST

1. Um utilizador não autenticado **efetua o login** através do protocolo HTTP, verbo **POST**, para o sítio:
   /backend/api/auth/login

   O corpo do **pedido** recebido tem o seguinte formato JSON:

   ```
   {
    "username": "Miguel",
    "password": "123"
   }
   ```

   Após o login, em caso de **sucesso**, a resposta **devolvida** por este recurso será um token JWT (Json Web Token) que o utilizador deve usar para autenticação e autorização, precisando do mesmo para aceder aos diversos endpoints da API.

   Em caso de **insucesso**, a resposta **devolvida** por este recurso retorna um código de status HTTP, 401 Unauthorized.

2. Um utilizador autenticado pode ver a sua informação pessoal através do protocolo HTTP, verbo GET, para o sítio:
   /backend/api/auth/user

   A resposta **devolvida** retorna toda a informação relevante sobre o utilizador que fez o pedido:

   ```
   {
    "username": "Miguel",
    "name": "Smigueli",
    "email": "miguel@gmail.com",
    "role": "Logista"
   }
   ```

3. Um utilizador autenticado **visualiza todas as suas encomendas** através do protocolo HTTP, verbo **GET**, para o sítio:
   /**backend/api/encomenda**

   A resposta **devolvida** retorna todas as encomendas e segue o seguinte formato JSON:


```
[
 {
  "id": 2,
  "username": "Bernardo",
  "estado": "EmProcessamento",
  "data_expedicao": null,
  "data_entrega": null
 },
 {
  "id": 1,
  "username": "Bernardo",
  "estado": "PorEntregar",
  "data_expedicao": "2025-01-17T14:33:27.253195",
  "data_entrega": null
 }
]
```

4. Um utilizador autenticado, **faz um pedido para ver os detalhes de uma encomenda,** através do protocolo HTTP, verbo **GET**, para o sítio:
   /backend/api/encomenda/{id}

   A resposta **devolvida** retorna uma determinada encomenda de um cliente específico e segue o seguinte formato JSON:


```
{
 "id": 1,
 "username": "Bernardo",
 "estado": "PorEntregar",
 "data_expedicao": "2025-01-17T14:33:27.253195",
 "data_entrega": null,
 "volumes": [
  {
   "id": 1,
   "entregue": false,
   "embalagems": [
    {
     "id": 1,
     "produto": {
      "id": 2,
      "nome": "Pão Integral"
     },
     "sensores": [
      {
       "id": 1,
       "valor": "2",
       "tipoId": 1,
       "tipoNome": "Temperatura",
       "estado": "ativo",
       "bateria": 84,
       "valMax": 30,
       "valMin": 10,
       "timeStamp": "2025-01-18T11:30:46.770582143",
```

```
            "embalagemId": 1
          }
        ],
        "quantidade": 1,
        "idTipoEmbalagem": 1,
        "tipoEmbalagem": "Isotérmica"
      }
    ]
}
```


5. Um utilizador autenticado, **pesquisa as encomendas por um determinado estado** através do protocolo HTTP, verbo **GET**, para o sítio:
   /backend/api/encomenda/estado/{estado}

   A resposta **devolvida** por este recurso retorna o seguinte formato JSON:


```
[
 {
   "id": 4,
   "username": "Bernardo",
   "estado": "Cancelada",
   "data_expedicao": null,
   "data_entrega": null
 },
 {
   "id": 6,
   "username": "Tendeiro",
   "estado": "Cancelada",
   "data_expedicao": null,
   "data_entrega": null
 }
]
```

6. Um utilizador autenticado **vê os alertas de uma determinada encomenda** com um inconveniente, através do protocolo HTTP, verbo **GET**, para o sítio:
   /backend/api/encomenda/{id}/alerta

   A resposta devolvida por este recurso seguinte formato JSON:


```
[
 {
  "id": 13,
  "id_volume": 1,
  "id_embalagem": 1,
  "mensagem": "Pão Integral - Temperatura excedeu o limite mínimo de 10!",
  "valor": "2",
  "data": "2025-01-17T14:58:10.47196",
  "id_encomenda": 1
 },
 {
  "id": 12,
  "id_volume": 1,
  "id_embalagem": 1,
  "mensagem": "Pão Integral - Temperatura excedeu o limite mínimo de 10!",
  "valor": "9",
  "data": "2025-01-17T14:56:53.417413",
  "id_encomenda": 1
 }
]
```

7. Um utilizador autenticado **altera os estado de uma encomenda** através do protocolo HTTP, verbo **PATCH**, para o sítio:
   /backend/api/encomenda/{id}

   O corpo do pedido **recebido** por este recurso segue o seguinte formato JSON:

   ```
   {
     "estado": "Entregue"
   }
   ```

   Em caso de **sucesso**, a resposta **devolvida** por este recurso retorna um código de status utilizado pelo HTTP:

   Response: status 200 OK

   Em caso de **insucesso**, a resposta **devolvida** por este recurso retorna um código de status HTTP adequado, acompanhado por uma mensagem descritiva do erro ocorrido.

8. Um utilizador autenticado, **associa um sensor a uma embalagem** através do protocolo HTTP, verbo **POST**, para o sítio:
   /backend/api/embalagem/{id}/sensor

   O corpo do pedido **recebido** por este recurso segue o seguinte formato JSON:

   ```
   {
     "id": 55,
     "valor": 90,
     "tipoId": 4,
     "valMin": 10,
     "valMax": 40,
     "bateria": 72
   }
   ```

   Em caso de **sucesso**, a resposta **devolvida** por este recurso retorna um código de status utilizado pelo HTTP:

   Response: status 201 Created

   Em caso de **insucesso**, a resposta **devolvida** por este recurso retorna um código de status HTTP adequado, acompanhado por uma mensagem descritiva do erro ocorrido.

9. Um utilizador autenticado**, verifica a última leitura dos sensores de um determinado tipo, das encomendas**, através do protocolo HTTP, verbo **GET**, para o sítio: **/backend/api/sensor/{tipo-sensor}**

   A resposta devolvida por este recurso seguinte formato JSON:


```
[
 {
  "id": 21,
  "valor": "39.74906316836962: -8.81280859823362",
  "tipoNome": "GPS",
  "estado": "ativo",
  "bateria": 100,
  "timeStamp": "2025-01-17T14:43:56.548163",
  "id_encomenda": 15,
  "id_volume": 16,
  "id_embalagem": 17
 },
 {
  "id": 24,
  "valor": "39.73440231964457: -8.821080620077632",
  "tipoNome": "GPS",
  "estado": "ativo",
  "bateria": 100,
  "timeStamp": "2025-01-17T14:44:15.843562",
  "id_encomenda": 15,
  "id_volume": 17,
  "id_embalagem": 18
 }
]
```

10. Um utilizador autenticado, pode **ver os detalhes de um volume,** através da utilização do protocolo HTTP, verbo **GET**, para o sítio:
    /backend/api/volume/{id}

    A resposta **devolvida** por este recurso seguinte formato JSON:


```
{
 "id": 1,
 "entregue": false,
 "embalagems": [
  {
   "id": 1,
   "produto": {
    "id": 2,
    "nome": "Pão Integral"
   },
   "sensores": [
    {
     "id": 1,
     "valor": "2",
     "tipoId": 1,
     "tipoNome": "Temperatura",
     "estado": "ativo",
     "bateria": 84,
     "valMax": 30,
     "valMin": 10,
     "timeStamp": "2025-01-18T12:03:14.81230513",
     "embalagemId": 1
    }
   ],
   "quantidade": 1,
   "idTipoEmbalagem": 1,
   "tipoEmbalagem": "Isotérmica"   }   ]
```

11. Um utilizador autenticado, **associa um volume a uma encomenda,** através da utilização do protocolo HTTP, verbo **POST**, para o sítio:
/backend/api/encomenda/{id}/volume

O corpo do pedido **recebido** por este recurso segue o seguinte formato JSON:


```
{
  "id": 110,
    "embalagens": [
     {
      "id": 110,
      "produto": {
       "id": 1
      },
      "tipo": 1,
      "quantidade": 8
     },
     {
      "id": 111,
      "produto": {
       "id": 5
      },
      "tipo": 4,
      "quantidade": 8
     }
    ]
}
```


Em caso de **sucesso**, a resposta **devolvida** por este recurso retorna um código de status utilizado pelo HTTP:

Response: status 201 Created

Em caso de **insucesso**, a resposta **devolvida** por este recurso retorna um código de status HTTP adequado, acompanhado por uma mensagem descritiva do erro ocorrido.

12. Um utilizador autenticado, **cria uma encomenda** através do protocolo HTTP, verbo **POST**, para o sítio:

    /backend/api/encomenda

    O corpo do pedido **recebido** por este recurso segue o seguinte formato JSON:


```
{
 "id": 100,
 "username": "Bernardo",
 "volumes": [
  {
   "id": 100,
   "embalagens": [
    {
     "id": 105,
     "produto": {
      "id": 1
     },
     "tipo": 1,
     "quantidade": 8
    },
    {
     "id": 106,
     "produto": {
      "id": 5
     },
     "tipo": 2,
     "quantidade": 8
    }
   ]
  }
 ]
}
```


    Em caso de **sucesso**, a resposta **devolvida** por este recurso retorna um código de status utilizado pelo HTTP:

    Response: status 201 Created

    Em caso de **insucesso**, a resposta **devolvida** por este recurso retorna um código de status HTTP adequado, acompanhado por uma mensagem descritiva do erro ocorrido.

13. No Sistema de Logística para a criação de uma encomenda, é necessário **recolher a informação de todos os produtos,** através da utilização do protocolo HTTP, verbo **GET**, para o sítio:
    /backend/api/produto

    A resposta **devolvida** por este recurso seguinte formato JSON:


```
[
 {
  "id": 1,
  "nome": "Maçã",
  "categoria": "Alimentos"
 },
 {
  "id": 6,
  "nome": "Fones",
  "categoria": "Tv e Som"
 },
 {
  "id": 7,
  "nome": "Martelo",
  "categoria": "Ferramentas"
 }
]
```

14. No Sistema de Logística para a associar um sensor a um volume, é necessário **saber os tipos de sensores,** através da utilização do protocolo HTTP, verbo **GET**, para o sítio:
/backend/api/sensor/tipo

A resposta **devolvida** por este recurso seguinte formato JSON:


```
[
 {
  "id": 1,
  "tipo": "Temperatura"
 },
 {
  "id": 2,
  "tipo": "Aceleração"
 }
]
```


15. No Sistema de Logística para a criação de uma encomenda, é necessário **recolher a informação de todos os clientes**, através da utilização do protocolo HTTP, verbo **GET**, para o sítio:
/backend/api/cliente

A resposta **devolvida** por este recurso seguinte formato JSON:


```
[
 {
  "username": "Bernardo"
 },
 {
  "username": "Carvalho"
 }
]
```

16. Um utilizador autenticado, **faz um pedido para visualizar o histórico de alertas de um sensor específico**, através do protocolo HTTP, verbo **GET**, para o sítio: **/backend/api/sensor/{id}/alerta**

    A resposta **devolvida** por este recurso segue o seguinte formato JSON**:**


```
[
 {
   "id": 13,
   "id_volume": 1,
   "id_embalagem": 1,
   "mensagem": "Pão Integral - Temperatura excedeu o limite mínimo de 10!",
   "valor": "2",
   "data": "2025-01-17T14:58:10.47196",
   "id_encomenda": 1
 },
 {
   "id": 12,
   "id_volume": 1,
   "id_embalagem": 1,
   "mensagem": "Pão Integral - Temperatura excedeu o limite mínimo de 10!",
   "valor": "9",
   "data": "2025-01-17T14:56:53.417413",
   "id_encomenda": 1
 }
]
```

17. Um utilizador autenticado, **faz um pedido para receber os alertas de todas as encomendas por entregar**, através do protocolo HTTP, verbo **GET**, para o sítio: **/backend/api/encomenda/alerta**

    A resposta **devolvida** por este recurso segue o seguinte formato JSON**:**


```
[
 {
    "id": 10,
    "mensagem": "Aspirador - Pressão Atmosférica excedeu o limite máximo de 9!",
    "id_sensor": 4,
    "valor": "10",
    "id_encomenda": 1,
    "username": "Bernardo",
    "time_stamp": "2025-01-17T14:55:04.345368"
  },
  {
   "id": 7,
   "mensagem": "Gelados - Temperatura excedeu o limite mínimo de 1!",
   "id_sensor": 33,
   "valor": "-2",
   "id_encomenda": 11,
   "username": "Carvalho",
   "time_stamp": "2025-01-17T14:54:03.507681"
  }
]
```

18. Um utilizador autenticado no Sistema Operacional, **faz um pedido para receber as coordenadas de cada volume de uma encomenda**, através do protocolo HTTP, verbo **GET**, para o sítio:
    /backend/api/encomenda/{id}/coordenada

    A resposta devolvida por este recurso segue o seguinte formato JSON

```
[
 {
   "volumeId": 2,
   "produtoNome": "Chave de Fenda",
   "coordenadas": "40.111: -73.12"
 }
]
```

19. O Sistema de Sensores terá acesso a **todos os sensores ativos**, através do protocolo HTTP, verbo **GET**, para o sítio:
    /backend/api/sensor

    A resposta **devolvida** por este recurso segue o seguinte formato JSON:


```
[
 {
  "id": 1,
  "valor": "2",
  "tipoNome": "Temperatura",
  "estado": "ativo",
  "bateria": 84,
  "timeStamp": "2025-01-17T14:58:10.471803",
  "idEmbalagem": 1,
  "idVolume": 1,
  "idEncomenda": 1,
  "valMax": 30,
  "valMin": 10
 },
 {
  "id": 5,
  "valor": "40.111: -73.12",
  "tipoNome": "GPS",
  "estado": "ativo",
  "bateria": 90,
  "timeStamp": "2025-01-17T14:56:43.41873",
  "idEmbalagem": 4,
  "idVolume": 2,
  "idEncomenda": 1,
  "valMax": null,
  "valMin": null
 } ]
```

20. O sensor quando deteta um baixo nível de bateria (menor que 2%), **altera o seu estado para inativo** através do protocolo HTTP, verbo **PATCH** para o sítio: /**backend/api/sensor/{id}**

    Em caso de **sucesso**, a resposta **devolvida** por este recurso tem o seguinte formato JSON:

    ```
    {
     "id": 2,
     "valor": "11",
     "tipoNome": "Temperatura",
     "estado": "inativo",
     "bateria": 90,
     "timeStamp": "2025-01-18T12:33:29.226205204"
    }
    ```

21. Um sensor **cria uma nova leitura** através do protocolo HTTP, verbo **POST** para o sítio: **/backend/api/sensor**

    O corpo o **pedido** recebido tem o seguinte formato JSON:

    ```
    {
     "id_sensor": 1,
     "bateria": 72,
     "valor": "40"
    }
    ```

    Em caso de **sucesso**, a resposta **devolvida** por este recurso retorna um código de status utilizado pelo HTTP:

    Response: status 200 OK

    Em caso de **insucesso**, a resposta **devolvida** por este recurso retorna um código de status HTTP adequado, acompanhado por uma mensagem descritiva do erro ocorrido.

22. Um utilizador autenticado, **recebe os tipos de embalagens e os sensores que as embalagens necessitam** através do protocolo HTTP, verbo **GET** para o sítio: **/backend/api/embalagem/tipo**

    A resposta **devolvida** por este recurso segue o seguinte formato JSON:


```
[
 {
  "id": 1,
  "tipo": "Isotérmica",
  "tipoSensorDTO": [
   {
    "tipo": "Temperatura"
   }
  ]
 },
 {
  "id": 3,
  "tipo": "Metalica",
  "tipoSensorDTO": [
   {
    "tipo": "Aceleração"
   },
   {
    "tipo": "Pressão Atmosférica"
   },
   {
    "tipo": "GPS"
   }
  ]
 },
 {
  "id": 4,
  "tipo": "Cartao",
  "tipoSensorDTO": [
   {
    "tipo": "GPS"
   }
  ]
 }
]
```

23. Um utilizador autenticado**, desassocia um sensor de uma embalagem** através do protocolo HTTP verbo DELETE para o sítio:
/backend/api/embalagem/{id}/sensor/{id}

Em caso de **sucesso**, a resposta **devolvida** por este recurso retorna um código de status utilizado pelo HTTP:

Response: status 200 OK

Em caso de **insucesso**, a resposta **devolvida** por este recurso retorna um código de status HTTP adequado, acompanhado por uma mensagem descritiva do erro ocorrido.

24. Um utilizador autenticado, **cria um novo tipo de embalagem** através do protocolo HTTP verbo POST para o sítio:
/backend/api/embalagem/{id}/tipo

O corpo o **pedido** recebido tem o seguinte formato JSON:

```
{
 "id": "1",
 "tipo": "E_TPP",
 "tipos_sensores": [
  { "id": 1 },
  { "id": 4 },
  { "id": 3 }
 ]
}
```

25. Um utilizador autenticado, **recebe as categorias dos produtos** através do protocolo HTTP verbo GET para o sítio:

    /backend/api/categoria

    A resposta **devolvida** por este recurso segue o seguinte formato JSON:


```
[
 {
  "id": 1,
  "nome": "Alimentos"
 },
 {
  "id": 2,
  "nome": "Tv e Som"
 },
 {
  "id": 3,
  "nome": "Ferramentas"
 },
 {
  "id": 4,
  "nome": "Bebidas"
 },
 {
  "id": 5,
  "nome": "Eletrodomésticos"
 },
 {
  "id": 6,
  "nome": "Vestuário"
 },
 {
  "id": 7,
  "nome": "Educação"
 },
 {
  "id": 8,
  "nome": "Desporto"
 }
]
```

26. Um utilizador autenticado, **cria um novo produto** através do protocolo HTTP verbo **POST** para o sítio:
    /backend/api/produto

    O corpo do **pedido** recebido tem o seguinte formato JSON:


```
{
  "id": 53,
  "nome": "Sapatos",
  "id_categoria": "6"
}
```


27. Um utilizador autenticado, **faz um pedido para ver o histórico de leituras de um sensor**, através do protocolo HTTP, verbo **GET**, para o sítio:
    /backend/api/sensor/{id}/leitura

    O corpo do pedido **recebido** por este recurso segue o seguinte formato JSON:


```
[
 {
   "id_sensor": 1,
   "bateria": 70,
   "valor": "10.9",
   "timeStamp": "2025-01-17T16:28:50.516119"
 },
 {
   "id_sensor": 1,
   "bateria": 72,
   "valor": "13.5",
   "timeStamp": "2025-01-17T16:28:45.5094"
 },
]
```

28. Um utilizador autenticado, **cria um novo tipo de sensor** através do protocolo HTTP verbo **POST** para o sítio:
    /backend/api/sensor/tipo

    O corpo o **pedido** recebido tem o seguinte formato JSON:

    ```
    {
      "id": "10",
      "tipo": "Humidade"
    }
    ```

29. Um utilizador autenticado **entrega um volume foi entregue,** através do protocolo HTTP, verbo **PATCH**, para o sítio:
    /backend/api/volume/{id}

    Em caso de **sucesso**, a resposta **devolvida** por este recurso retorna um código de status utilizado pelo HTTP:

    Response: status 200 OK

    Em caso de **insucesso**, a resposta **devolvida** por este recurso retorna um código de status HTTP adequado, acompanhado por uma mensagem descritiva do erro ocorrido.