Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia

Eco-Data Management

Ana Cristina Jesus 20211383

Programação Orientada a Objetos
Docente: Miguel Bugalho
2021/2022

Sumário Executivo

Enquadramento

Cork Modular é uma empresa que tem o conceito de funcionalidade e sustentabilidade, distinguindo-se de outros móveis e empresas pela a sua modularidade.

Com a construção de prateleiras, mesas laterais, unidade de armazenamento e outros módulos com apenas dois elementos de cortiça e peças de conexão que podem ser feitas e montadas pelo cliente.

A marca foi fundada em 2016 pela designer Irena Übler, com um estúdio localizado no Porto, em Portugal. Utilizam principalmente a cortiça, um material nobre e amigo do ambiente produzido inteiramente em Portugal. Este material é tradicional de Portugal, e é combinada com tiras têxteis para dar um toque final único.

Diagrama de Classes

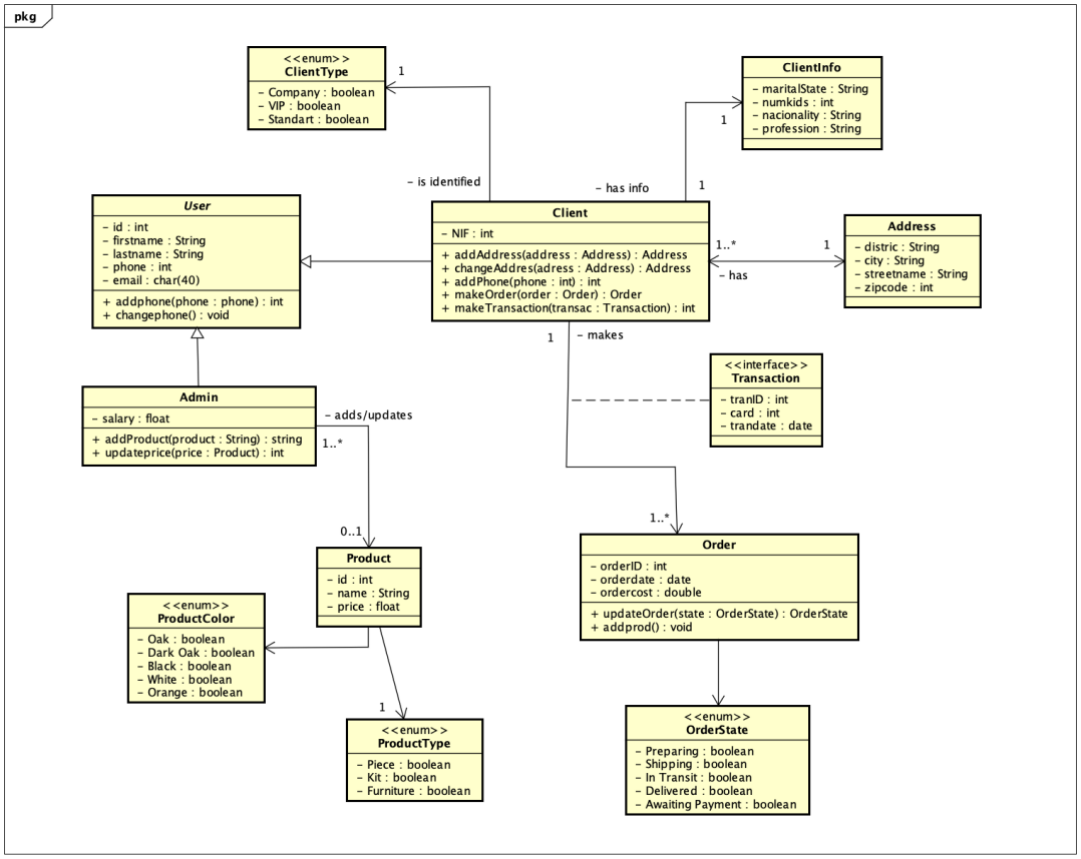
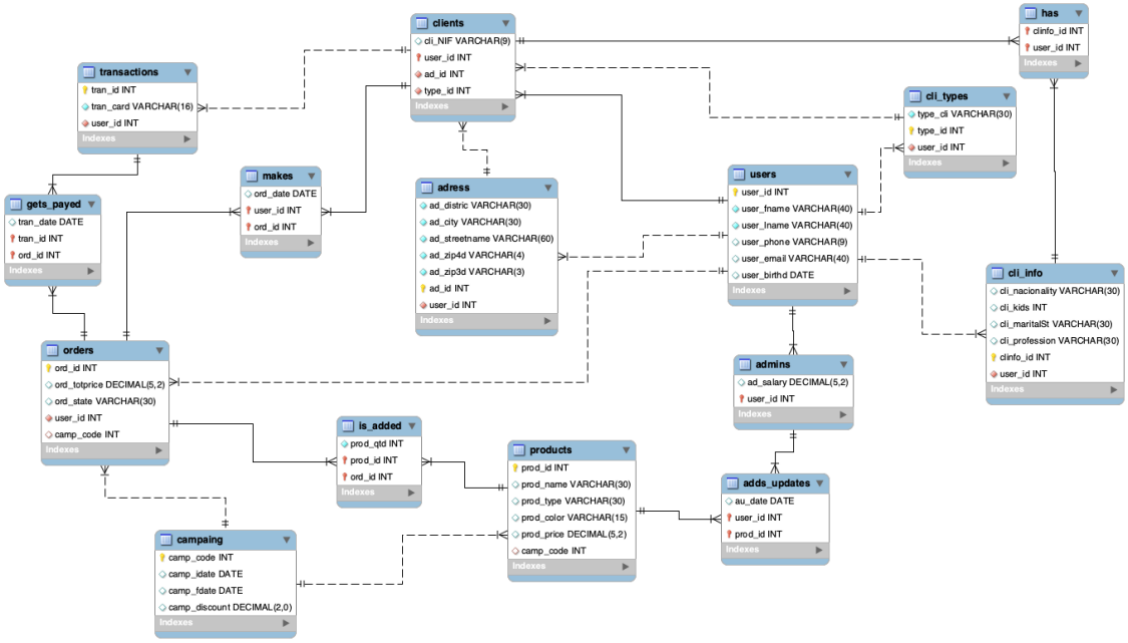


Diagrama da Base de Dados



Cenários

1. Um cliente entra no site/aplicação e é lhe mostrado a sua informação, é depois levada para a lista de produtos disponível, com o seu id e o resto das suas informações como o seu nome, cor e preço. Se o cliente desejar fazer uma encomenda, introduz o id do produto desejado (apenas um) e insere a quantidade. E assim é lhe dado o id da encomenda.

2. Um administrador entra no site/aplicação e é lhe mostrado o seu perfil. Se pretender adicionar um produto, é lhe levado para a lista de produtos e para adicionar insere apenas o nome, a cor, o tipo de produto e o preço. Quando acaba, clica *enter*, e o produto automaticamente é inserido na base de dados com um novo id, e na parte do cliente é lhe mostrado esse novo produto com essas informações.

Eco-Data Management-
Documentação
REST

Recurso Users (api/users)

Listar todos os users
Devolve a lista de todos os users

/api/users/?fname:=texto&lname=:text **(get)**

Parâmetros:
fname (optional): Os users mostrados têm de ter o texto neste parâmetro no primeiro nome de user
lname (optional): Os users mostrados têm de ter o texto neste parâmetro no último nome de user

Sucesso (200):
[
{"id": 1, "phone": "929455500","salary": "900.00","name": "Quaresma Nazário","age": 23},
{"id": 2, "phone": "965551683","salary": "850.00","name": "Serena Leitão","age": 27},
…]

Erro:
500: Erro de Servidor

Exemplo:

```
let f_name = document.getElementById("fname").value;
let l_name = document.getElementById("lname").value;
let u_type = await $.ajax({
    url: "/api/users?fname="+f_name+"&lname="+l_name,
    method: "get",
    dataType: "json"
});
```

Obter informação de um user (individual)
Identifica o tipo de user

/api/users/id **(get)**

Parâmetros:
Id (obrigatório): número que identifica o utilizador, todos têm um diferente de cada um (único)

Sucesso (200):
{"id":13,"phone":"935555750","name":"Cael Olivares","age":24}

Erro:
500: Erro de Servidor
404: O user não foi encontrado {“msg”: “User not found for that id”}

Exemplo:

```
let u_type = await $.ajax({
    url: "/api/users/"+Users_id,
    method: "get",
    dataType: "json"
});
```

Fazer autenticação de um user
A partir do id e do email, caso o utilizador exista, devolve a informação desse utilizador, incluindo informação privada.

/api/users/login **(post)**

Dados:
{ “User_Id”: 1, ”email”: “m.andrade99@gmail.com” }

Sucesso (200):
{"id":23,"phone":"922555991","name":"Catarina Rijo","age":54}

Erro:
500: Erro de Servidor
404: O user não foi encontrado {“msg”: “Wrong email or Id, try again”}

Exemplo:

```
let login = {
    user_id = document.getElementById("User_id").value,
    email = document.getElementById("email").value
    }
    let user = await $.ajax({
      url: "/api/users/login",
      method: "post",
      data: JSON.stringify(login),
      dataType: "json",
      content: "application/json"
    });
    sessionStorage.setItem("User  id",Users.user  id);
    window.location = "profile.html";
```

Recurso Clientes (api/users/clients)

Listar todos os clientes
Devolve a lista de todos os clientes com algumas das suas informações

/api/users/?fname:=texto&lname=:text **(get)**

Parâmetros:
fname (optional): Os clientes mostrados têm de ter o texto neste parâmetro no primeiro nome de cliente
lname (optional): Os clientes mostrados têm de ter o texto neste parâmetro no último nome de cliente

Sucesso (200):

[
{"id":11,"phone":"922555636","name":"HenriqueAreosa","age":35},
{"id":12,"phone":"929255584","name":"Edgar Quinaz","age":51},
{"id":13,"phone":"935555750","name":"Cael Olivares","age":24}
}
…]

Erro:
500: Erro de Servidor

Exemplo:

```
let f_name = document.getElementById("fname").value;
let l_name = document.getElementById("lname").value;
let client = await $.ajax({
    url: "/api/users/client?fname="+f_name+"&lname="+l_name,
    method: "get",
    dataType: "json"
});
```

Obter informação de um cliente (individual)
Devolve a informação de um cliente

/api/users/clients/:id **(get)**

Parâmetros:
Id (obrigatório): número que identifica o utilizador, todos têm um diferente de cada um (único)

Sucesso (200):

{"id":42,"phone":"921555810","name":"Maria Jesus","age":53}

Erro:
500: Erro de Servidor
404: O user não foi encontrado {“msg”: “Client not found for that id”}

Exemplo:

```
let client = await $.ajax({
    url: "/api/users/client"+client_id,
    method: "get",
    dataType: "json"
});
```

Recurso Admin (api/users/admin)

Listar todos os admin
Devolve a lista de todos os admin com algumas das suas informações

/api/users/admin/?fname:=texto&lname=:text **(get)**

Parâmetros:
fname (optional): Os users mostrados têm de ter o texto neste parâmetro no primeiro nome de user
lname (optional): Os users mostrados têm de ter o texto neste parâmetro no último nome de user

Sucesso (200):
[
{"id":1,"phone":"929455500","salary":"900.00","name":"Quaresma Nazário","age":23},
{"id":2,"phone":"965551683","salary":"850.00","name":"Serena Leitão","age":27}
…]

Erro:
500: Erro de Servidor

Exemplo:

```
let f_name = document.getElementById("fname").value;
let l_name = document.getElementById("lname").value;
let admin = await $.ajax({
    url: "/api/users/admin?fname="+f_name+"&lname="+l_name,
    method: "get",
    dataType: "json"
});
```

Obter informação de um admin (individual)
Devolve a informação de um admin

/api/users/admin/:id **(get)**

Parâmetros:
Id (obrigatório): número que identifica o utilizador, todos têm um diferente de cada um (único)

Sucesso (200):
{"id":4,"phone":"922555778","salary":"870.00","name":"Lurdes Raposo","age":28}

Erro:
500: Erro de Servidor
404: O user não foi encontrado {“msg”: “Admin not found for that id”}

Exemplo:

```
let f_name = document.getElementById("fname").value;
let l_name = document.getElementById("lname").value;
let admin = await $.ajax({
    url: "/api/users/admin/"+admin_id,
    method: "get",
    dataType: "json"
});
```

Recurso Produtos (api/products)

Listar todos os produtos
Devolve a lista de todos os produtos

/api/products **(get)**

Sucesso (200):
[
{"id":1,"info":"Textile Strap; null; 0.0€; "},
{"id":2,"info":"Textile Strap; null; 0.0€; "},
{"id":3,"info":"Textile Strap; null; 0.0€; "},
{"id":4,"info":"Textile Strap; null; 0.0€; "},
...]

Erro:
500: Erro de Servidor

Exemplo:

```
let products = await $.ajax({
    url: "/api/products",
    method: "get",
    dataType: "json"
});
```

Obter informação de um produto (individual)
Devolve a informação de um cliente

/api/products/:id **(get)**

Parâmetros:
Id (obrigatório): número que identifica o utilizador, todos têm um diferente de cada um (único)

Sucesso (200):
{"id":5,"info":"Textile Strap; null; 0.0€; "}

Erro:
500: Erro de Servidor
404: O user não foi encontrado {“msg”: “Product not found for that id”}

Exemplo:

```
let products = await $.ajax({
    url: "/api/products/"+prod_id,
    method: "get",
    dataType: "json"
});
```

Inserir um produto
Adiciona um produto

/api/products **(post)**

Dados:

```
{
    "name" : "Laptop Support",
    "price" : "19.99",
    "color" : "Oak",
    "type" : "Accessories"
}
```

Sucesso (200):

```
{
    "id": 56,
    "name": "Laptop Support",
    "price": 19.99,
    "color": "Oak",
    "type": "Accessories",
    "info": "Laptop Support; Accessories; 19.99€; "
}
```

Erro:
500: Erro de Servidor

Recurso Encomenda (api/orders)

Listar todos as encomendas
Devolve a lista de todos os clientes com algumas das suas informações

/api/orders **(get)**

Sucesso (200):

```
[
  {"id":1,"orderInfo":"Ordered at 2021-11-05","cost":19.9},
  {"id":2,"orderInfo":"Ordered at 2021-11-05","cost":13.79},
  {"id":3,"orderInfo":"Ordered at 2021-11-05","cost":20.5},
…]
```

Erro:
500: Erro de Servidor

Exemplo:

```
let orders = await $.ajax({
    url: "/api/orders",
    method: "get",
    dataType: "json"
});
```

Obter informação de uma encomenda
Devolve a informação de uma encomenda

/api/orders/:id **(get)**

Parâmetros:
Id (obrigatório): número que identifica o utilizador, todos têm um diferente de cada um (único)

Sucesso (200):
{"id":5,"orderInfo":"Ordered at 2021-11-05","cost":20.5}

Erro:
500: Erro de Servidor
404: O user não foi encontrado {"msg": "Order not found for that id"}

Exemplo:

```
let orders = await $.ajax({
    url: "/api/orders/"+order_id,
    method: "get",
    dataType: "json"
});
```

Fazer uma encomenda
Adiciona um produto

/api/orders **(post)**

Dados:

```
{
    "user": 43,
    "value": 19.9,
    "state": "Awaiting Payment",
    "ordDate": "2022-05-25",
    "product": "43",
    "qtd": 1,
    "orderInfo": "Ordered at 2022-05-25",
    "cost": 19.9
}
```

Sucesso (200):

```
{
    "id": 32,
    "user": 43,
    "value": 19.9,
    "state": "Awaiting Payment",
    "ordDate": "2022-05-25",
    "product": "43",
    "qtd": 1,
    "orderInfo": "Ordered at 2022-05-25",
    "cost": 19.9
}
```

Erro:
500: Erro de Servidor

Recurso Campanha (api/campaigns)

Listar todos as campanhas
Devolve a lista de todas as campanhas

/api/campaigns **(get)**

Sucesso (200):
[
{"id":1,"discount":1.0,"campaindDate":"Campaign starts: 2017-11-24 and ends at 2017-11-24"},
{"id":2,"discount":0.0,"campaindDate":"Campaign starts: 2017-12-26 and ends at 2017-12-28"}
…]

Erro:
500: Erro de Servidor

Exemplo:

```
let campaign = await $.ajax({
      url: "/api/campaigns",
      method: "get",
      dataType: "json"
   });
```