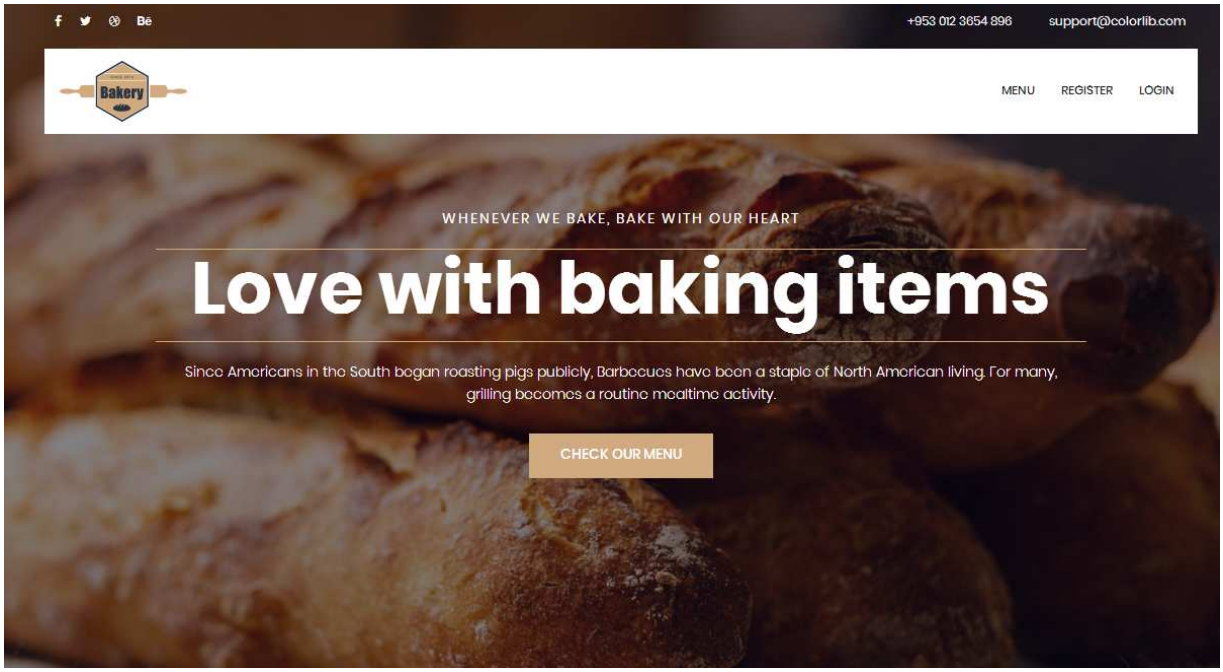
UAlg DEEI
UNIVERSIDADE DO ALGARVE
FACULDADE DE CIÊNCIAS E TECNOLOGIA

DAW – Exame EXEMPLO

00.02.2021 00:00 – 00:00

LER O ENUNCIADO ATÉ AO FIM ANTES DE COMEÇAR!

O objectivo do exame é construir um site web suportado por uma base de dados relacional. Informação sobre a base de dados encontra-se em ANEXO.



O site web consiste no portal de uma padaria que permite a utilizadores registados fazer encomendar online pizzas e outros produtos. O site tem as seguintes páginas:

1. "**Home**" é a página de rosto do site;
2. "**Menu**" é a página principal do site;
3. "**Register**" para registo de utilizadores;
4. "**Login**" para o login de utilizadores;
5. "**My Orders**" mostra as encomendas realizadas pelo utilizador;

É dado acesso a um conjunto de templates HTML construídos em *Bootstrap* pela empresa de web design colorlib.com.
Solicita-se ao aluno a realização do site web em *Symfony* e *Twig*, apenas das páginas acima descritas.

PRELIMINARES

A. Faça login por ssh (com o PuTTY, por exemplo) no servidor com o IP 10.10.23.184

```
a12345@daw2:~$
```

B. Faça download para a pasta "PRACTICE"[1], do código do site web

```
a12345@daw2:~$ cd public_html

a12345@daw2:~/public_html$
git clone git://github.com/jmatbastos/PRACTICE.git PRACTICE
```

C. Complete a instalação

```
a12345@daw2:~/public_html$ cd PRACTICE
a12345@daw2:~/public_html/PRACTICE$ composer install
```

D. Crie a sua cópia da base de dados

```
a12345@daw2:~/public_html/PRACTICE$
mysql -u a12345 --password=****** db_a12345 < database.SQL
```

Substitua "a12345" pelo seu login e "******" pela password de acesso à sua base de dados

E. Actualize o ficheiro .env

Altere o ficheiro ".env" para utilizar as credenciais da sua base de dados.

```
a12345@daw2:~/public_html/PRACTICE$ nano .env
DATABASE_URL=mysql://a12345:******@10.10.23.184:3306/db_a12345?serverVersion=15.1
```

NOTAS:

- Se **não** se recorda da password da sua base de dados, recupere-a com o comando

```
a12345@daw2:~$ /usr/local/bin/mysql-db
```

[1] a pasta "EXAME" é criada automaticamente pelo comando `git clone`

- Se tiver dificuldade em criar a base de dados em linha de commando pode utilizar o acesso web phpMyAdmin e os comandos SQL descritos no APÊNDICE

- No caso de haver algum conflito com uma tabela já existente na sua base de dados, mude o nome à tabela existente *ou* apague a tabela existente

- A aplicação deve correr obrigatoriamente na área pessoal do aluno na pasta "PRACTICE" no servidor web do departamento disponibilizado para o efeito: http://daw.deei.fct.ualg.pt

- O controlador principal tem que estar no ficheiro "`BakeryController.php`"

- A classe com as funções de acesso à base de dados tem que estar no ficheiro "`Bakery_modelController.php`"

- **Opcional!** Caso use o componente **Doctrine ORM** a restrição acima é levantada e pode utilizar um ficheiro para cada modelo que representa uma tabela da base de dados

- **Opcional!** Pode utilizar **bin/console** para "scaffolding" das funcionalidades de registo e autenticação de utilizadores, bem como para a criação de formulários.

- **Opcional!** Pode utilizar **Doctrine SQL Query Builder** para acessar a base de dados.

A página de entrada no site tem que ser

http://daw.deei.fct.ualg.pt/~a12345/PRACTICE/public/index.php/bakery

Deve ser considerado uma **SUGESTÃO** o seguinte mapeamento entre URLs e controladores:

```
* @Route("/bakery name="bakery")
* @Route("/menu/{category_id?}", name="menu")
* @Route("/register", name="register")
* @Route("/login", name="login")
* @Route("/logout", name="logout")
* @Route("/order/{product_id}", name="order")
* @Route("/orders", name="myOrders")
```

FUNCIONALIDADE “HOME”

A funcionalidade “Home” é página de rosto do site. Encontra no URL

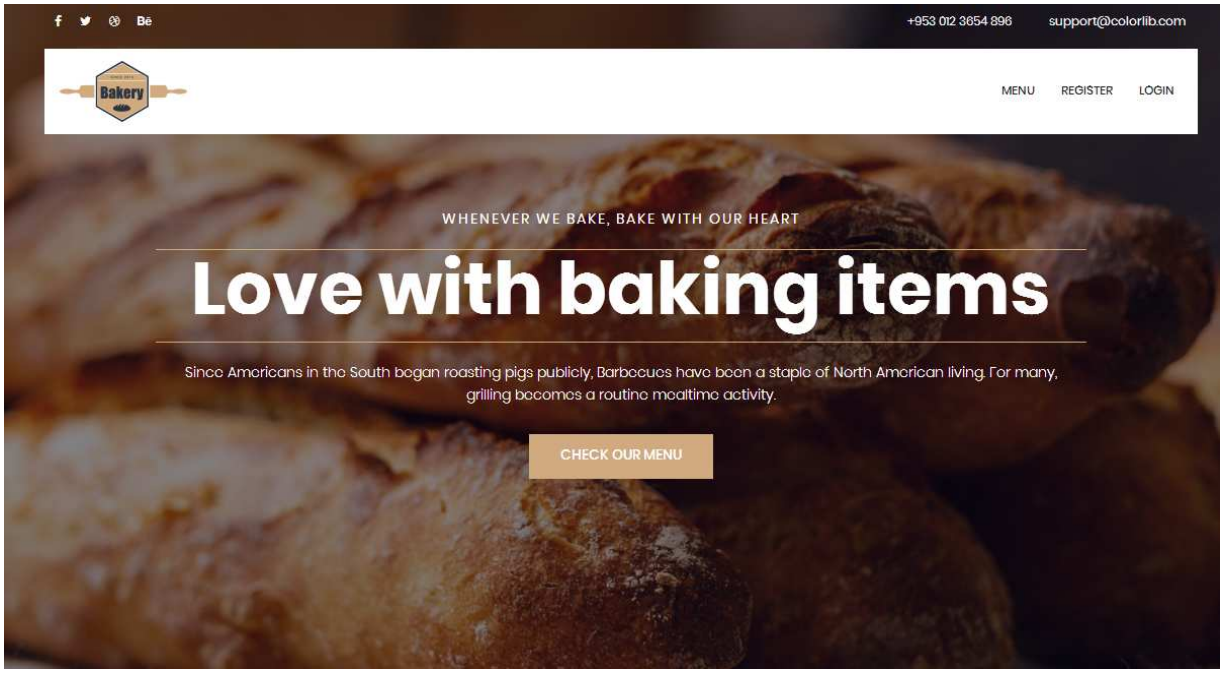
`http://daw.deei.fct.ualg.pt/~a12345/PRACTICE/demo/home.html`

uma demonstração do funcionamento do site.

1. **[3 valores]**

Construa o template Twig para esta página.

SUGESTÃO: Adapte o template “`home.html`” fornecido pela empresa de web design. Encontra este template na pasta “demo”.



Construa o método no controlador `BakeryController.php` responsável por gerar a página de rosto do site

NOTA:

- seu site deve ser portável; utilize as funções `asset()` e `path()` para gerar os hyperlinks locais!

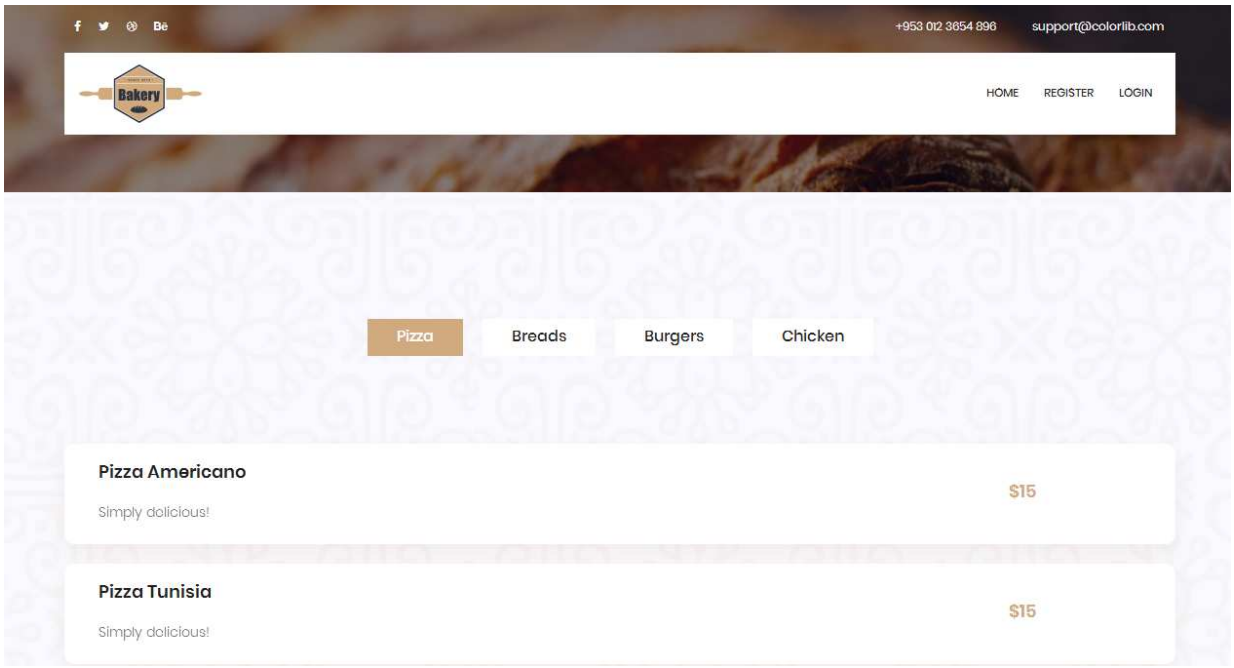
FUNCIONALIDADE “MENU”

A funcionalidade “Menu” é página principal do site e contem uma lista dos produtos disponíveis na padaria.

2. [5 valores]

Construa o template Twig para esta página.

SUGESTÃO: Adapte o template “`menu.html`” fornecido pela empresa de web design. Encontra este template na pasta “demo”



Construa o método `menu(cat_id = FALSE)` no controlador `BakeryController.php` e a query correspondente no controlador `Bakery_modelController.php` responsável por fazer uma lista dos produtos existentes na tabela “products” da base de dados

- As categorias de produtos (“Pizza”, “Breads”, ...) encontram-se na tabela “categories” da base de dados;

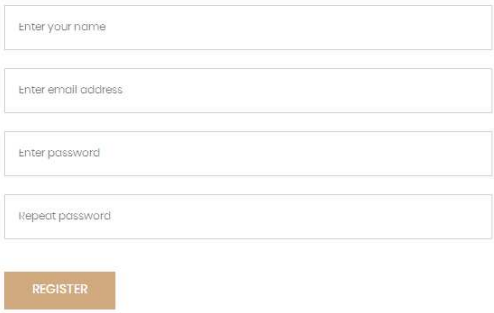
- Na página “menu”, as categorias de produtos (“Pizza”, “Breads”, ...) são hyperlinks com dados embutidos (“/menu/1”, “/menu/2” etc) que permitem mostrar uma lista de produtos da categoria seleccionada
- A página de entrada no site, por omissão, mostra todos os produtos
- Preencha a lista com os campos “name”, “description”, “price” existentes na tabela “products”
- Ignore os campos vazios, quer na tabela “categories”, quer na tabela “products”

FUNCIONALIDADE “REGISTER”

A funcionalidade “Register” permite registar um utilizador.

3. [2.5 valores]
Construa o template Twig para esta página.

SUGESTÃO: Adapte o template “`register.html`” fornecido pela empresa de web design. Encontra este template na pasta “demo”.



Construa os métodos `register()` e `register_action()` no controlador `BakeryController.php`[2]

- Se todos os campos obrigatórios não estão preenchidos, redirecciona novamente para o formulário com uma mensagem de erro. A validação dos dados deve ser feita utilizando o componente Symfony "Validator".

- Se todos os campos obrigatórios estão correctamente preenchidos, e o email ainda não existe na base de dados, regista o utilizador na tabela “users” utilizando a query correspondente no controlador `Bakery_modelController.php`.

A password_digest é a hash utilizando o algoritmo MD5 da password: `substr(md5($_POST['pass1_utilizador']),0,32)`. Pode utilizar a função MySQL `NOW()`ou a função PHP `date("Y-m-d H:i:s")` para actualizar os campos `created_at`, `updated_at` da tabela “users”

[2] Pode combinar os dois métodos `register()` e `register_action()` num só método

FUNCIONALIDADE “LOGIN” & “LOGOUT”

A funcionalidade “Login” permite autenticar um utilizador.

4. [2.5 valores]
Construa o template Twig correspondente.

SUGESTÃO: Adapte o template “`register.html`” fornecido pela empresa de web design. Encontra este template na pasta “demo”

Construa os métodos `login()` e `login_action()`no controlador `BakeryController.php`[3] `e a` query correspondente no controlador `Bakery_modelController.php`

- em caso de sucesso no login:
  - regista no array da sessão os dados “id” e “name” do utilizador retirados da base de dados
  - re-direcciona para o portal de entrada no site
  - na barra de navegação o hyperlink “Login” transforma-se no hyperlink “Logout” e o hyperlink “Register” transforma-se no texto “Welcome user!” (onde “user” é o nome do utilizador registado) na página “menu”
  - na barra de navegação da página “menu” aparece um novo hyperlink "My Orders"
  - na página "menu" fica visível à direita de cada produto um hyperlink “Order” com o id do produto embutido (/order/1, /order/2 etc)
- em caso de insucesso
  - re-direcciona novamente para a página `login`.
  - Envia uma mensagem de erro “Wrong email or password”

No controlador `BakeryController.php` construa o método `logout()`que encerra a sessão

[3] Se desejar pode combinar os dois métodos `login()` e `login_action()` num só método

FUNCIONALIDADE “ORDER”

A funcionalidade “Order” permite registar o pedido do utilizador.

4. [2 valores]

Construa o método `order(product_id)`no controlador `BakeryController.php` e a query correspondente no controlador `Bakery_modelController.php`, responsáveis pelo registo do pedido do utilizador na base de dados

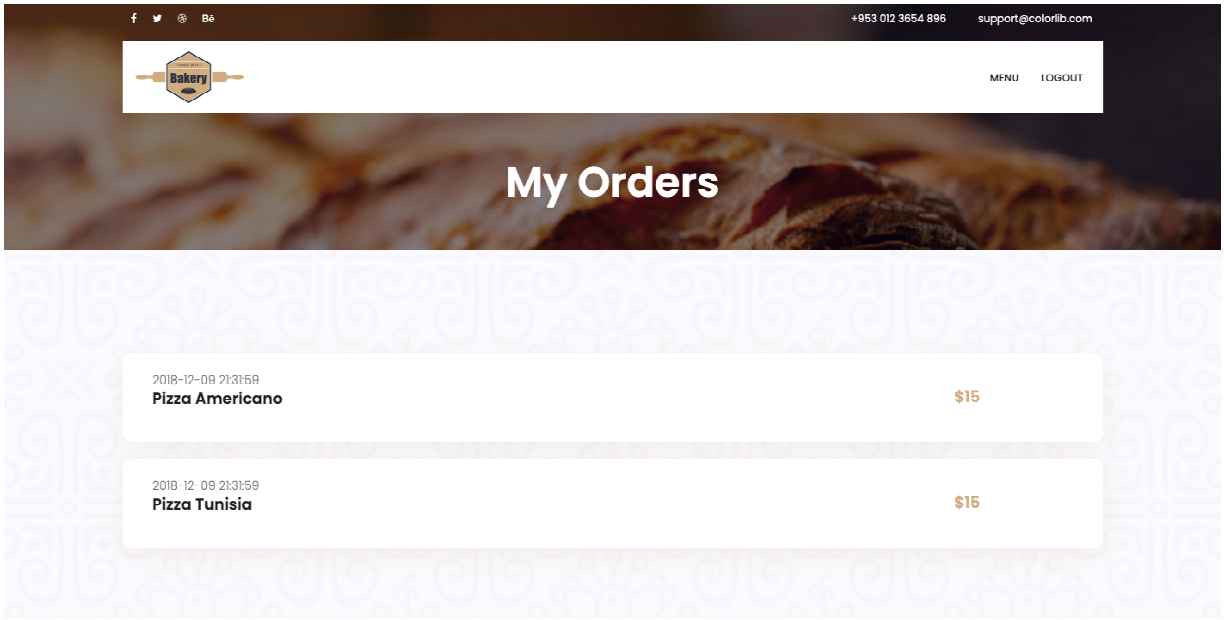
FUNCIONALIDADE “MY ORDERS”

A funcionalidade “My Orders” permite ao utilizador registado mostrar uma lista com os produtos que foram pedidos.

4. [3 valores]

Construa o template Twig correspondente.

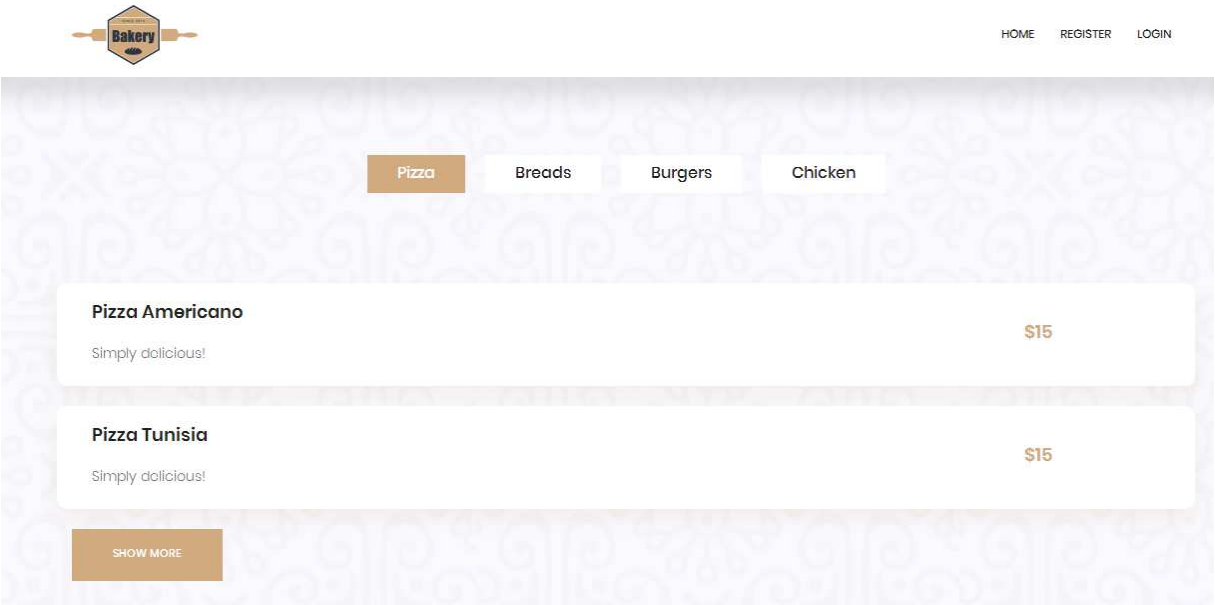
SUGESTÃO: Adapte o template “`myOrders.html`” fornecido pela empresa de web design. Encontra este template na pasta “demo”



Construa o método `myOrders()`no controlador `BakeryController.php` e a query correspondente no controlador `Bakery_modelController.php`

FUNCIONALIDADE “SHOW MORE”

A funcionalidade “Show More” permite mostrar os produtos da padaria em incrementos de dois.



5. [2 valores]

Quando a página “Menu” é carregada é feito o *download* de uma pagina HTML com apenas os primeiros dois produtos de uma determinada categoria .

- Altere o método no controlador `BakeryController.php` e a query correspondente em `Bakery_modelController.php`, para a página HTML inicial apenas conter os dois primeiros produtos de uma determinada categoria que constam da tabela “products”
- Altere o template e adicione um hyperlink “SHOW MORE” para uma função em AJAX que, sempre que se clica no hyperlink, mostra mais dois produtos.
- A informação deve ser transferida do servidor para o browser em JSON
- Construa o método no controlador `BakeryController.php` (e a query correspondente em `Bakery_modelController.php`) que é o interlocutor no servidor da função AJAX

SUGESTÃO: Tem à sua disposição elementos de layout (botões etc) em “`elements.html`” fornecido pela empresa de web design. Encontra este template na pasta “demo”.

NOTAS:

- Caso tenha trabalhado no seu portatil, **é obrigatório fazer o upload de todos os ficheiros** para a pasta “PRACTICE” no seu site web pessoal

```
/users/a12345/public_html/PRACTICE
```

(onde 12345 é o seu número de aluno). Utilize scp (Linux) ou WinSCP (Windows) ou FileZilla (Windows e MAC) para fazer a cópia. As permissões dos ficheiros devem ser octal 640 (`rw- r-- ---`).

- NÃO faça o upload de pastas! Se precisar de criar uma pasta faça-o no servidor com o comando

```
a12345@daw2:~$ mkdir nome_da_pasta
```

Verifique que o site fica operacional.

- Caso tenha problemas como seu código, lembre-se que pode consultar o log do servidor web com o comando

```
a12345@daw2:~$ tail -f /var/log/apache2/error.log
```

ANEXO 1 Acesso à base de dados MySQL

- O acesso à base de dados MySQL pode ser feito utilizando o software **phpMyAdmin** disponível no URL

- http://daw.deei.fct.ualg.pt/phpMyAdmin
- http://10.10.23.184/phpMyAdmin

ou em linha de comando (substitua “12345” pelo seu número de aluno)

```
a12345@daw2:~$mysql -u a12345 -p db_a12345
```

ANEXO 2 : estrutura da base de dados

```
--
-- Table structure for table `users`
--

CREATE TABLE IF NOT EXISTS `users` (
 `id` int(11) NOT NULL auto_increment,
 `name` varchar(255) default NULL,
 `email` varchar(255) default NULL,
 `created_at` datetime NOT NULL,
 `updated_at` datetime NOT NULL,
 `password_digest` varchar(255) default NULL,
 `remember_digest` varchar(255) default NULL,
 `admin` tinyint(1) default NULL,
 `activation_digest` varchar(255) default NULL,
 `activated` tinyint(1) default NULL,
 `activated_at` datetime default NULL,
 `reset_digest` varchar(255) default NULL,
 `reset_sent_at` datetime default NULL,
 `roles` longtext COMMENT '(DC2Type:json)',
 PRIMARY KEY (`id`),
 UNIQUE KEY (`email`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Table structure for table `categories`
--

CREATE TABLE IF NOT EXISTS `categories` (
 `id` int(11) NOT NULL auto_increment,
 `name` varchar(255) default NULL,
 `description` varchar(255) default NULL,
 `image` varchar(255) default NULL,
 PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `categories`
--

INSERT INTO `categories` VALUES
(1,'Pizza',NULL,NULL),(2,'Breads',NULL,NULL),(3,'Burgers',NULL,NULL),(4,'C
hicken',NULL,NULL);

--
-- Table structure for table `products`
--

CREATE TABLE IF NOT EXISTS `products` (
 `id` int(11) NOT NULL auto_increment,
 `cat_id` int(11) NOT NULL,
 `name` varchar(255) default NULL,
 `description` varchar(255) default NULL,
 `price` int(5) default NULL,
 `image` varchar(255) default NULL,
```

```
 PRIMARY KEY (`id`),
 CONSTRAINT FOREIGN KEY (`cat_id`) REFERENCES `categories` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `products`
--

INSERT INTO `products` VALUES (1,1,'Pizza Americano','Simply
delicious',15,NULL),(2,1,'Pizza Tunisia','Simply
wonderful!',20,NULL),(3,1,'Beef lovers Pizza','For meat
lovers!',20,NULL),(4,1,'Chicken lovers Pizza','Chicken
Chicken!',10,NULL),(5,1,'Meatball Pizza','No teeth
required!',12,NULL),(6,1,'Bakery special Pizza','Our
special!',25,NULL),(7,1,'Pizza Alfredo','Typical
italian!',13,NULL),(8,1,'Cheese lovers Pizza','For
vegetarians!',9,NULL),(9,2,'White bread','Simply
delicious!',2,NULL),(10,2,'Whole grain bread','So
healthy!',4,NULL),(11,3,'Bakery burger','Our
special!',10,NULL),(12,3,'Cheese Burger','So
tasty!',12,NULL),(13,3,'Vegetarian burger','So
healty!',8,NULL),(14,3,'Spicy Burger!','Hot hot
hot!',9,NULL),(15,4,'Chicken wings','So tasty!',10,NULL),(16,4,'Frango da
Guia','Algarve special!',20,NULL);


--
-- Table structure for table `orders`
--


CREATE TABLE IF NOT EXISTS `orders` (
`id` int(11) NOT NULL AUTO_INCREMENT,
`user_id` int(11) NOT NULL,
`product_id` int(11) NOT NULL,
`created_at` datetime NOT NULL,
 PRIMARY KEY (`id`),
 CONSTRAINT FOREIGN KEY (`user_id`) REFERENCES `users` (`id`),
 CONSTRAINT FOREIGN KEY (`product_id`) REFERENCES `products` (`id`)
 ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```