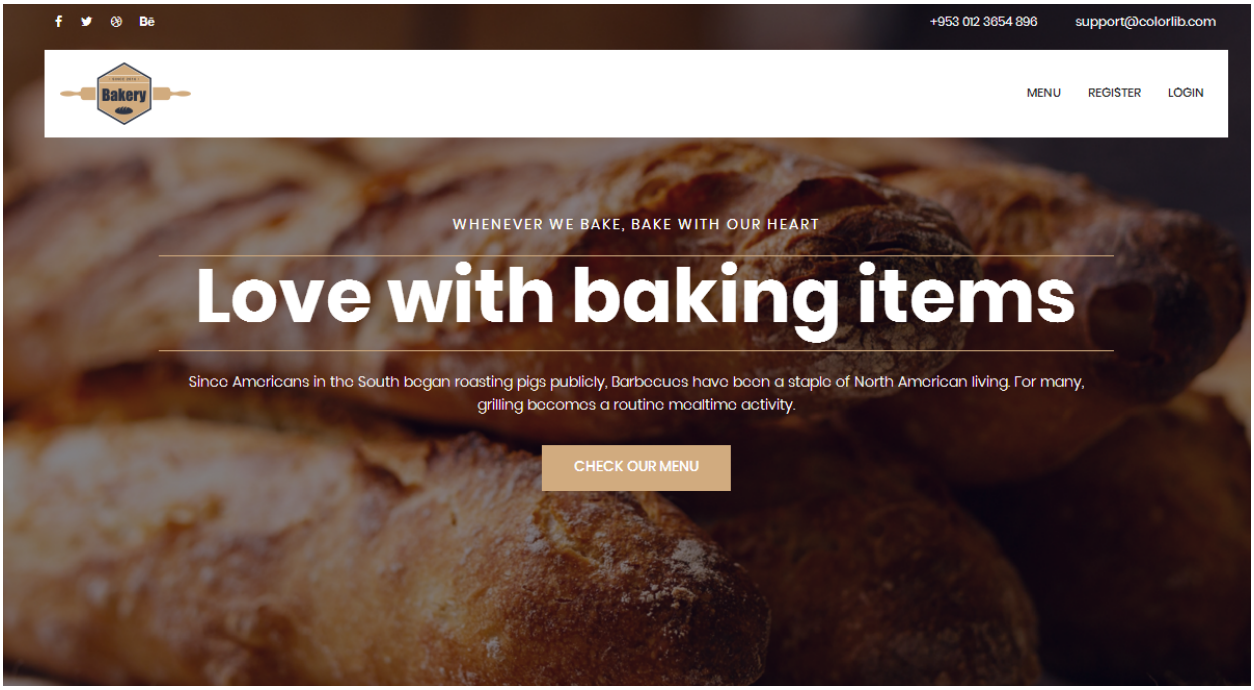
UAlg DEEI
UNIVERSIDADE DO ALGARVE
FACULDADE DE CIÊNCIAS E TECNOLOGIA

IPM – Exame Ép. Normal
16.01.2023 09:00 – 12:00

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 de 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 uma lista das encomendas do utilizador;
6. **"Message**" é uma página auxiliar para mostrar mensagens (dado);

É dado acesso a um conjunto de templates HTML construídos em *Bootstrap* realizados pela empresa de web design colorlib.com. É obrigatório em qualquer circunstância utilizar os templates dados.

Solicita-se ao aluno a realização do site web no *Framework VueJS*, apenas das páginas acima descritas.

Deve ser UTILIZADO o seguinte mapeamento entre URLs e páginas:

```
const routes = [
  {
    path: '/',
    component: Home
  },
  {
    path: '/menu',
    component: Menu
  },
  {
    path: '/register',
    component: Register
  },
  {
    path: '/login',
    component: Login
  },
  {
    path: '/orders',
    component: MyOrders
  },
  {
    path: '/message',
    component: Message
  },

]
```

No ficheiro "zip" que é distribuído encontra um esboço das páginas requeridas.

As APIs de acesso à base dados **são dadas**, bem como **é dada** uma VueJS "store" que contém todas as "acções" AJAX de acesso às APIs

PRELIMINARES

- **VUEJS**

Faça o download de uma instalação base do VueJS adaptada para este exame em[1]

https://github.com/jmatbastos/EXAMEVue2/archive/refs/heads/master.zip

e expanda o ficheiro `EXAMEVue2-master.zip` para dentro pasta

`C:\XAMPP\htdocs`

- Se necessário mude o nome da pasta de `EXAMEVue2-master` para `EXAMEVue2`

- **VUE STORE**

Na pasta `C:\XAMPP\htdocs\EXAMEVue2\src\store` encontra o código de todas as acções necessárias para acesso AJAX à base de dados.

Nos ficheiros "`categories.js`", "`products.js`", "`orders.js`", "`user.js`" **actualize** os URLs

`http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/...`

e substitua "12345" pelo seu nº de aluno

- **API E CREDENCIAIS DE ACESSO À BASE DE DADOS**

Faça login com o PuTTY no servidor daw2 (IP 10.10.23.184)

```
a12345@daw2:~$ cd public_html
a12345@daw2:~/public_html$
git clone https://github.com/jmatbastos/EXAMEVue2.git EXAMEVue2
```

Abra o ficheiro `EXAMEVue2/api/db.php`

```
a12345@daw2:~/public_html$ cd EXAMEVue2/api
a12345@daw2:~/public_html/EXAMEVue2/api$ nano db.php
```

[1] em alternativa se tem a aplicação git instalada no seu PC/portátil:

```
C:\XAMPP\htdocs> git clone https://github.com/jmatbastos/EXAMEVue2.git
```

e actualize com as suas credenciais de acesso à base de dados

```
$hostname = "localhost";
$db_name = "db_a12345";
$db_user = "a12345";
$db_passwd = "PASS";
```

(substitua '12345' pelo seu nº de aluno e 'PASS' pela sua password de acesso à base de dados...)

- **ATUALIZAÇÃO DA BASE DE DADOS**

Atualize a sua base de dados no servidor de produção com o comando

```
a12345@daw2:~/public_html/EXAMEVue2/api$
mysql -ua12345 -pPASS db_a12345 < ~/public_html/EXAMEVue2/api/db.SQL
```

Encontra no **APÊNDICE 1** a descrição da estrutura da base de dados, bem como um método alternativo de criação das tabelas.

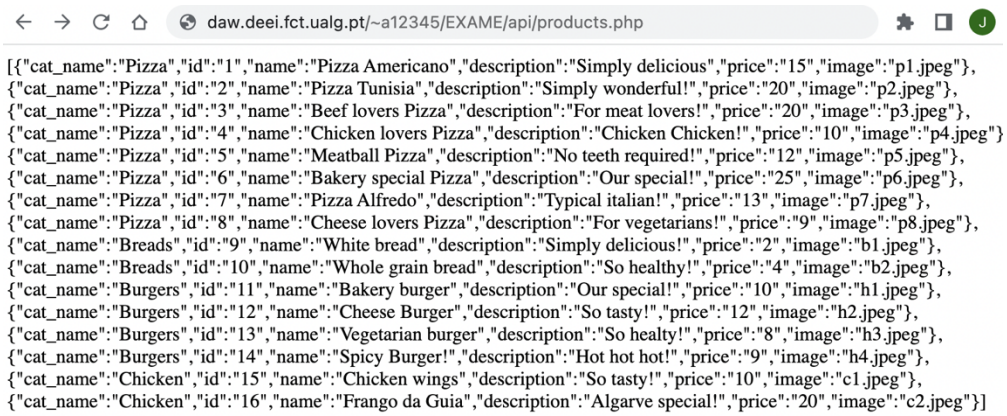
TESTE O FUNCIONAMENTO DA API E O ACESSO À BASE DE DADOS

A partir do seu browser preferido vá ao seguinte URL

`https://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/products.php`

(substitua '12345' pelo seu nº de aluno)

Deverá receber dados no formato JSON:



TESTE A INSTALAÇÃO DO FRAMEWORK

- Execute o comando

```
C:\XAMPP\htdocs\EXAMEVue2> npm install[2]
```

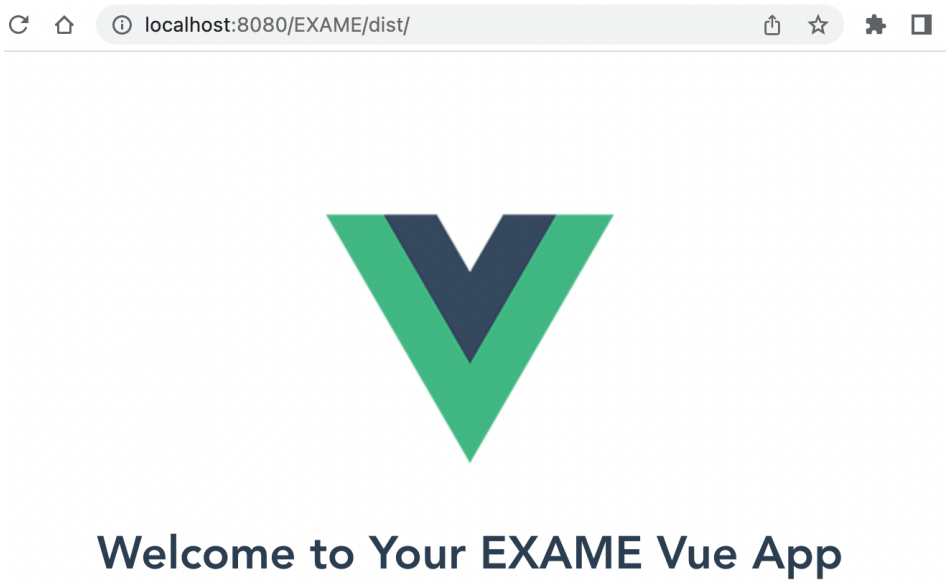
- Execute o comando

```
C:\XAMPP\htdocs\EXAMEVue2> npm run serve
```

A partir do seu browser preferido vá ao seguinte URL

http://localhost:8080/EXAMEVue2/dist/

Deverá receber uma página web de boas-vindas



[2] tem de ter instalado node.js versão 16.18.0 ou 16.18.1

FUNCIONALIDADE “HOME”

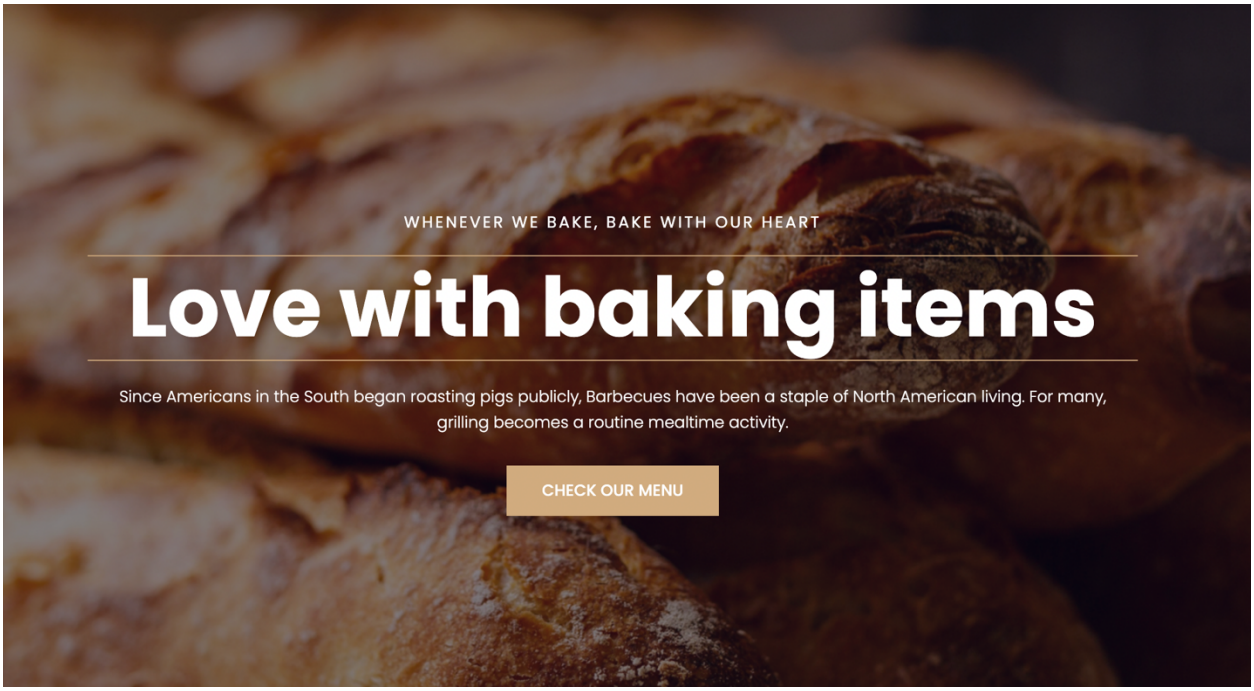
A funcionalidade “Home” é página de entrada do site.

1. **[3 valores]**

Adapte o template “`index.html`” fornecido pela empresa de web design.
Encontra este template na pasta “`demo`”. Pode ver este template em funcionamento no URL

```
http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/demo/index.html
```

(substitua 12345 pelo seu nº de aluno...)



Construa o template no ficheiro

```
C:\XAMPP\htdocs\EXAMEVue2\src\views\Home.vue
```

"CHECK OUR MENU" é um link para a página/vista "menu"

NOTA IMPORTANTE:

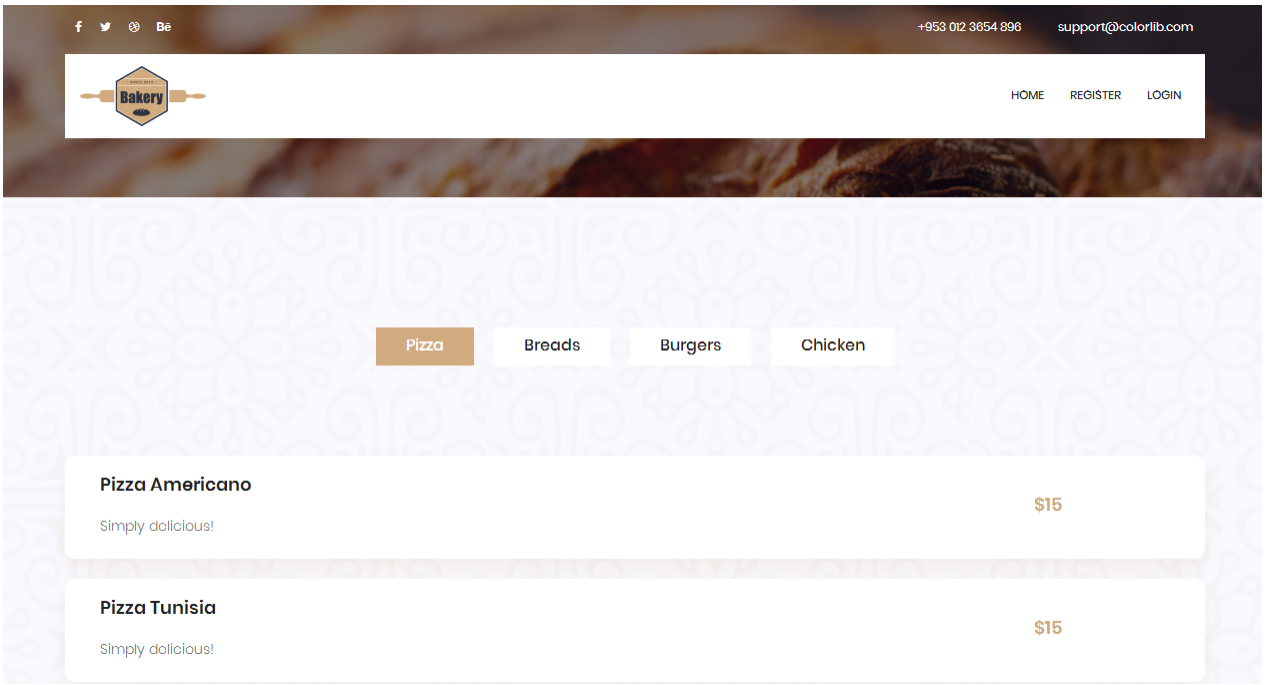
- os hyperlinks para os recursos css e javascript utilizados no template (`<link rel="stylesheet" href="... "> <script src="...">`) estão já colocados no ficheiro "`public/index.html`" (dado)
- os recursos estáticos utilizados no template (img, css,fonts, js) estão já colocados em pastas com o mesmo nome (img, css, etc) dentro da pasta "`public`" (dado)

FUNCIONALIDADE “MENU”

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

2. [7 valores]

Adapte o template “`menu.html`” fornecido pela empresa de web design.
Encontra este template na pasta “`demo`”. (Pode ver este template em funcionamento no URL `http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/demo/menu.html`)



Construa o template e o controlador no ficheiro

`C:\XAMPP\htdocs\EXAMEVue2\src\views\Menu.vue`

Sugere-se que o conteúdo "Header" e "Footer" deste template, que é comum aos outros templates, seja colocado nos componentes

`C:\XAMPP\htdocs\EXAMEVue2\src\components\Header.vue` (dado em parte)
`C:\XAMPP\htdocs\EXAMEVue2\src\components\Footer.vue` (dado)

O controlador utiliza a acção

```
this.$store.dispatch('products/getProductsFromDB')
```

para fazer o download de todos os produtos existentes na base de dados. Pode considerar que as categorias existentes no site são estáticas.

O menu apresenta por defeito os produtos da categoria "Pizza".

Para cada produto apresente o nome, a descrição, e o preço.

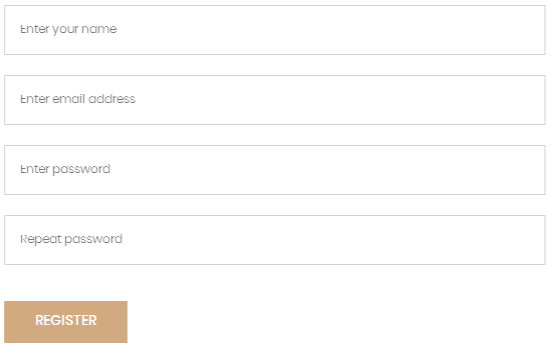
NOTA IMPORTANTE: Este template já contém javascript que controla o funcionamento dos "tabs" "Pizza", "Breads", etc. Se desejar pode continuar a utilizar esta funcionalidade, ou pode em alternativa reprogramar a funcionalidade tabular em VueJS.

FUNCIONALIDADE “REGISTER”

A funcionalidade “Register” permite registar um utilizador.

3. [2 valores]

Adapte o template “`register.html`” fornecido pela empresa de web design. Encontra este template na pasta “`demo`”. Pode ver este template em funcionamento no URL `http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/demo/register.html`.



O template deve ser colocado no ficheiro

`C:\XAMPP\htdocs\EXAMEVue2\src\views\Register.vue`

O controlador do formulário valida os dados introduzidos.

Adicione o código necessário para:

- Em caso de insucesso
  - o formulário apresenta os campos correctos já preenchidos e o motivo do insucesso no registo
- Em caso de sucesso
  - o controlador utiliza as acções `this.$store.dispatch('user/userExists',user)` para verificar se o utilizador já existe na base de dados e, se não existir,

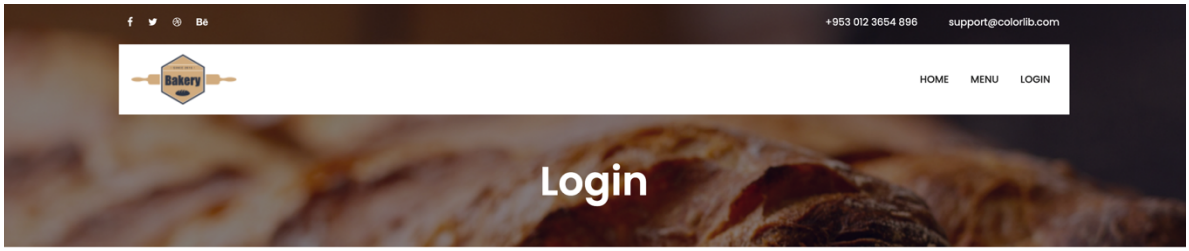
`this.$store.dispatch('user/addUser')` para inserir o novo utilizador na base de dados

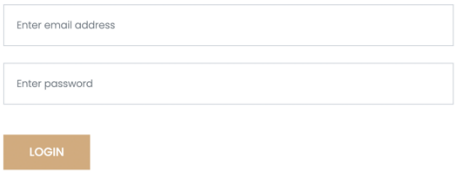
- o controlador redireciona para a "vista" `Message.vue`, que mostra a mensagem “Welcome! You can now login". A "vista" `Message.vue` redireciona automaticamente para a página do menu do site passados 2 segundos

FUNCIONALIDADE “LOGIN” & “LOGOUT”

A funcionalidade “Login” permite autenticar um utilizador.

4. [3 valores]
Adapte o template “`register.html`” fornecido pela empresa de web design. Encontra este template na pasta “`demo`”. Pode ver este template em funcionamento no URL `http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/demo/login.html`.





O template e o controlador deverão encontrar-se em

`C:\XAMPP\htdocs\EXAMEVue2\src\views\Login.vue`

Adicione o código necessário para utilizar a "acção" `this.$store.dispatch('user/loginUser', user)`:

- Em caso de sucesso, guardar na "store" no objecto "user" o id, nome, e email do utilizador validado
- Em caso de sucesso, os links no componente "Header" devem mudar de "Login" "Register" para "My Orders" e "Logout".
- Em caso de sucesso deve aparecer um botão "Order" ao lado direito do produto
- Em caso de sucesso utilizar a "vista" `Message.vue`, com a mensagem `"Welcome back " + this.$store.getters['user/getUser'].name + "!"`
- Em caso de insucesso no formulário de login é apresentada a mensagem de erro “Login failed: wrong email or password”.

Utilize `this.$store.commit('user/logoutUser')` para fazer logout do utilizador

FUNCIONALIDADE “My Orders”

A funcionalidade “My Orders” permite mostrar os produtos da padaria que o utilizador comprou até ao momento

5. [5 valores]

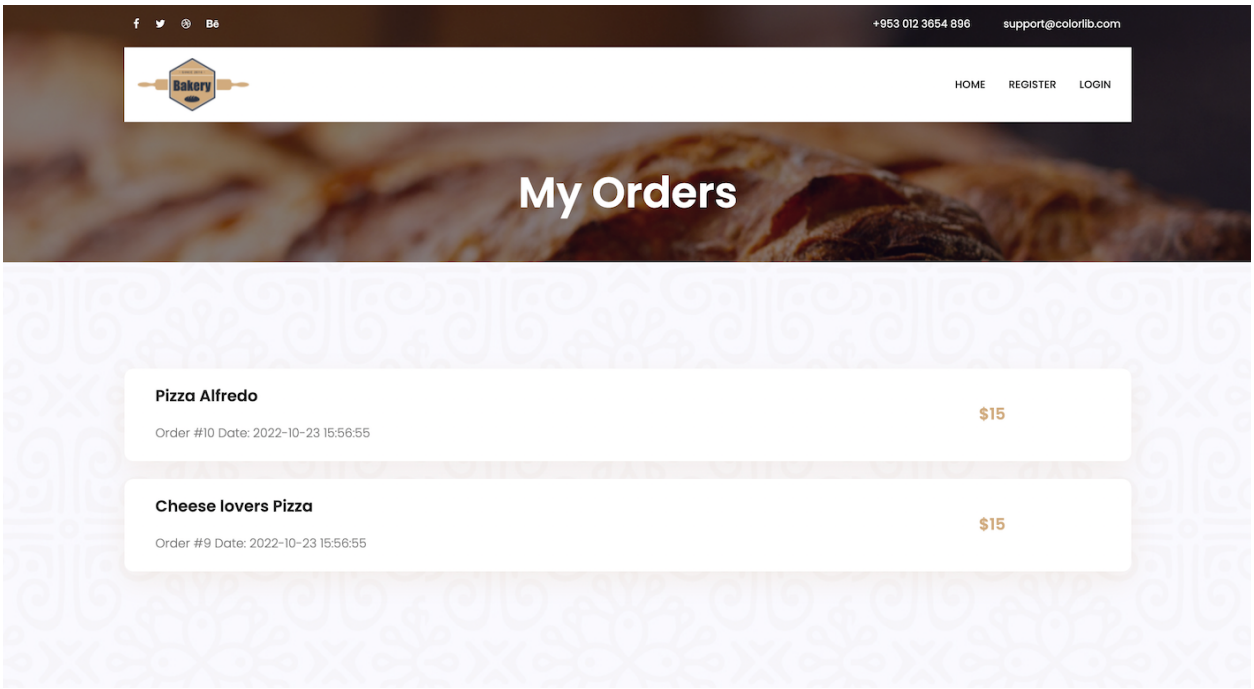
No ficheiro

`C:\XAMPP\htdocs\EXAMEVue2\src\views\Menu.vue`

utilize a acção `this.$store.dispatch('orders/newOrder', newOrder)` para guardar na base de dados uma compra do utilizador, sempre que este carregar no botão "Order".

Utilize a "vista" `Message.vue`, com a mensagem "Order success!"

Adapte o template “`myOrders.html`” fornecido pela empresa de web design. Encontra este template na pasta “`demo`”. Pode ver este template em funcionamento no URL `http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/demo/myOrders.html`.



Construa o template e o controlador no ficheiro

```
C:\XAMPP\htdocs\EXAMEVue2\src\views\MyOrders.vue
```

O controlador utiliza a acção

```
this.$store.dispatch('orders/getMyOrdersFromDB',
this.$store.getters['user/getUser'].id)
```

para fazer o download da base de dados dos produtos comprados pelo utilizador.

Para cada produto apresente o nome, o id e a data da encomenda, e o preço.

UPLOAD

Actualize o “publicPath” no ficheiro

`C:\XAMPP\htdocs\EXAMEVue2\vue.config.js`
com o conteúdo

```
// vue.config.js
module.exports = {
  publicPath: '/~a12345/EXAMEVue2/dist/'
}
```

(substitua '12345' pelo seu nº de aluno!)

Execute o comando

```
C:\XAMPP\htdocs\EXAMEVue2> npm run build
```

Faça o upload com scp/WinSCP/FileZilla das pastas

- `"src"`
- `"dist"`

para a pasta “EXAMEVue2” no seu site web pessoal no servidor de produção

Teste o funcionamento do site no URL

http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/dist/

(substitua '12345' pelo seu nº de aluno...)

NÃO espere pelo último minuto do exame para fazer o upload!!

NÃO faça upload da pasta "node_modules"!!

ANEXO 1. Estrutura da base de dados

A estrutura da base de dados pode ser criada com

- na shell do servidor de produção (IP 10.10.23.184) com o comando

```
a12345s@daw2:~$
mysql -ua12345 -pPASS db_a12345 < ~/public_html/EXAMEVue2/api/db.SQL
```

- com o "tab" "IMPORT" em

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

onde `db.SQL` é o ficheiro que contem a estrutura da base de dados: (encontra este ficheiro em `C:\XAMPP\htdocs\EXAMEVue2\api\db.SQL`)

```
--
-- Table structure for table `customers`
--

CREATE TABLE `customers` (
 `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,
 PRIMARY KEY (`id`),
 UNIQUE KEY `index_users_on_email` (`email`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

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

```
CREATE TABLE `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,N
ULL),(4,'Chicken',NULL,NULL);

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

CREATE TABLE `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 `orders` (
  `id` int NOT NULL AUTO_INCREMENT,
  `customer_id` int NOT NULL,
  `product_id` int NOT NULL,
  `created_at` datetime NOT NULL,
  `price` int DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `customer_id` (`customer_id`),
  KEY `product_id` (`product_id`),
  CONSTRAINT `orders_ibfk_1` FOREIGN KEY (`customer_id`)
REFERENCES `customers` (`id`),
  CONSTRAINT `orders_ibfk_2` FOREIGN KEY (`product_id`)
REFERENCES `products` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3;
```

ANEXO 2. Descrição dos métodos disponíveis na API de acesso à base de dados

Na pasta `C:\XAMPP\htdocs\EXAMEVue2\api` encontra o código de todas as APIs necessárias para acesso à base de dados.

Todos os dados recebidos da API ou enviados à API estão no formato JSON. Pode testar os métodos disponíveis com o comando "curl", por exemplo[3]

```
curl -X GET 'http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/customers.php?email=jbastos@ualg.pt'
curl -X POST 'http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/customers.php' -d '{"name":"Mary
Stevens","email":"stevens@gmail.com", "password":"sWd356"}'
```

ou utilizando o site https://postman.com

Tabela customers

- **Verifica se um utilizador já se encontra registado na base de dados:**
  GET http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/customers.php?email=jbastos@ualg.pt
  retorna: JSON string {"id":"1","name":"Jose Bastos","email":"jbastos@ualg.pt"} ou null
- **Registo de um utilizador:**
  POST http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/customers.php
  Body: JSON string {"name":"Jose Bastos","email":"jbastos@ualg.pt","password":"segredo"}
  retorna: JSON string {"id":"1","name":"Jose Bastos","email":"jbastos@ualg.pt"} ou null
- **Login de um utilizador:**
  GET http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/customers.php?email=jbastos@ualg.pt&password="segredo"
  retorna: JSON string {"id":"1","name":"Jose Bastos","email":"jbastos@ualg.pt"} ou null

[3] os métodos que utilizam o comando HTTP "GET" podem ser testados diretamente no browser!

Tabela categories

- **Todas as categorias:**
  GET http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/categories.php
  retorna: JSON string [{"id":"1","name":"Pizza","description":"", ...}, ...]

Tabela products

- **Todos os produtos:**
  GET http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/products.php
  retorna: JSON string [{"cat_name":"SUSHI","id":"1","name":"Salmon Roll","description":"Simply delicious!", ...}, ...]

Tabela orders

- **Todas as ordens do utilizador:**
  GET http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/orders.php?customer_id=35
  retorna: JSON string [{"id":"1","created_at":"2021-12-03 18:20:31","product_price":"102","product_name":"Salmon Roll", ...] ou null
- **Registo de uma nova ordem do utilizador:**
  POST http://daw.deei.fct.ualg.pt/~a12345/EXAMEVue2/api/orders.php
  Body: JSON string {"customer_id":"19","product_id":"3","product_price":"10"}
  retorna: JSON string {"id":"5", "customer_id":"3", "product_id":"5", created_at":"2021-12-03 18:20:31","price":"30"}