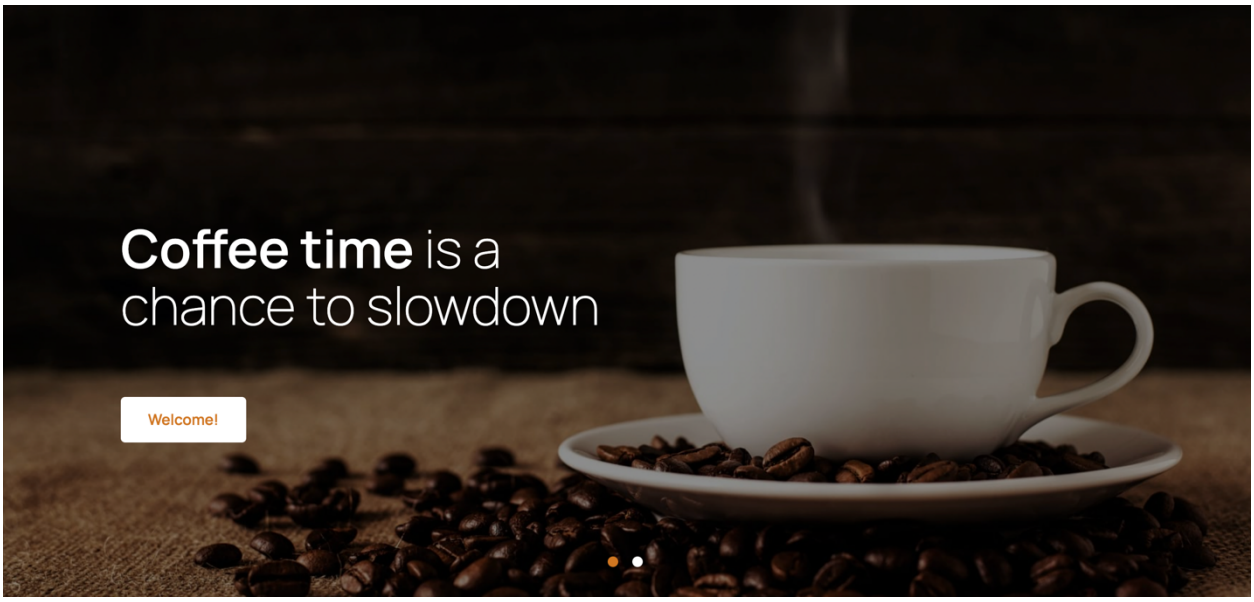
UAlg DEEI
UNIVERSIDADE DO ALGARVE
FACULDADE DE CIÊNCIAS E TECNOLOGIA

IPM – Exame Ép. Recurso

24.01.2024 09:00 – 12:00

LER O ENUNCIADO ATÉ AO FIM ANTES DE COMEÇAR!

O objectivo do exame é construir um site web em programação "front-end" usando o *Framework VueJS 3*. O site é 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 cafetaria que permite a utilizadores registados fazer reservas de mesas.

Pode ver uma implementação do site no URL

http://daw.deei.fct.ualg.pt/~a555554/dist/

O site tem as seguintes páginas/vistas:

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. "**Book**" permite ao cliente registado fazer uma reserva de mesa;
6. "**MyBookings**" permite ao cliente registado ver as suas reservas;
7. "**Message**" permite enviar mensagens (**dado**);

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

É dado acesso a um conjunto de templates HTML construídos em *Bootstrap* fornecidos pela empresa de web design. A adaptação dos templates ao framework VueJS **não pode alterar** o layout e o estilo!

Solicita-se ao aluno a realização do site web em *Framework VueJS 3*, apenas das páginas acima descritas.

Deve ser usado 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: '/book',
    component: Book
  },
  {
    path: '/mybookings',
    component: MyBookings
  },

  {
    path: '/message',
    component: Message
  },

]
```

PRELIMINARES

- **VUEJS**

Faça o download de uma instalação base do VueJS 3 adaptada para este exame em[1]

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

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

`C:\XAMPP\htdocs`

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

- **VUE STORE**

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

Nos ficheiros "`beverages.js`", "`bookings.js`","`categories.js`", "`user.js`" **atualize** os URLs

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

e substitua "12345" pelo seu nº de aluno

- **PATH RAIZ DO SITE**

Atualize o "publicPath" no ficheiro

`C:\XAMPP\htdocs\RECURSO\vite.config.js`

`base: '/~a12345/EXAME/dist/'`

e substitua '12345' pelo seu nº de aluno

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

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

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

Abra o ficheiro

`C:\XAMPP\htdocs\RECURSO\api\db.php`

e **atualize com as suas credenciais** de acesso à base de dados

```
$hostname = "10.10.23.184";
$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...)

Com o seu programa SCP preferido (scp, FileZilla, WinSCP, etc..) **faça o upload**

da pasta

- **"api"**

para dentro da pasta "RECURSO"[2] no servidor `10.10.23.184`[3]

- **ATUALIZAÇÃO DA BASE DE DADOS**

Faça login com o PuTTY no servidor daw2 (IP 10.10.23.184) **e atualize a sua base de dados** no servidor de produção:

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

[2] A pasta "RECURSO" já foi previamente criada. Se a pasta não existe, faça login com o PuTTY no servidor `10.10.23.184` e crie uma pasta com o nome "`RECURSO`" dentro da pasta "`public_html`":

```
a12345@daw2:~$ cd public_html
a12345@daw2:~/public_html$ mkdir RECURSO
```

[3] Também pode fazer o upload para o servidor 10.10.23.27

OPCIONAL: Método alternativo de atualização da base de dados:

Use o software **phpMyAdmin** disponível no URL

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

e **importe** o ficheiro

`C:/XAMPP/htdocs/RECURSO/api/db.SQL`

Desative a checkbox "enable foreign key checks".

Encontra no **ANEXO 1** a descrição da estrutura da base de dados.

Encontra no **ANEXO 2** a descrição das APIs que permitem o acesso à base de dados

TESTE A INSTALAÇÃO DAS API

A partir do seu browser preferido vá ao seguinte URL

`http://daw.deei.fct.ualg.pt/~a12345/RECURSO/api/beverages.php`

(substitua "12345" pelo seu número de aluno)

Deverá receber uma página web com dados no formato JSON:

[{"cat_id":"1","cat_name":"Coffee","id":"1","name":"Single Cup Brew","description":"Fresh Brewed coffee and steamed milk","price":"20","image":"middle.png"},{"cat_id":"1","cat_name":"Coffee","id":"2","name":"Caff\u00e9 Americano","description":"Fresh Brewed coffee","price":"14","image":"middle1.png"},{"cat_id":"1","cat_name":"Coffee","id":"3","name":"Caramel Macchiato","description":"Fresh Caramel","price":"11","image":"middle2.png"},{"cat_id":"1","cat_name":"Coffee","id":"4","name":"Standard black coffee","description":"Fresh black ","price":"8","image":"middle3.png"},{"cat_id":"1","cat_name":"Coffee","id":"5","name":"Black Eyed Andy","description":"Fresh black ","price":"23","image":"middle2.png"},{"cat_id":"1","cat_name":"Coffee","id":"6","name":"Coffee of the Day","description":"Fresh coffee","price":"22","image":"middle3.png"},{"cat_id":"1","cat_name":"Coffee","id":"7","name":"Cuban Shot","description":"Cream cuban","price":"14","image":"middle.png"},{"cat_id":"1","cat_name":"Coffee","id":"8","name":"Caf\u00e9 Latte","description":"Fresh latte","price":"11","image":"middle1.png"},{"cat_id":"2","cat_name":"Hot Beverages","id":"9","name":"Caramel Macchiato","description":"Fresh macchiato","price":"20","image":"middle3.png"},{"cat_id":"2","cat_name":"Hot Beverages","id":"10","name":"Coffee of the Day","description":"Fresh coffee","price":"50","image":"middle2.png"},{"cat_id":"2","cat_name":"Hot Beverages","id":"11","name":"Caff\u00e9 Americano","description":"Fresh americano","price":"23","image":"middle.png"},{"cat_id":"2","cat_name":"Hot Beverages","id":"12","name":"Filtered Coffee","description":"Fresh filtered","price":"50","image":"middle1.png"},{"cat_id":"2","cat_name":"Hot Beverages","id":"13","name":"Cappuccino coffee","description":"Fresh coffee","price":"10","image":"middle2.png"},{"cat_id":"2","cat_name":"Hot Beverages","id":"14","name":"Cafe latte","description":"Cream latte","price":"10","image":"middle1.png"},{"cat_id":"2","cat_name":"Hot Beverages","id":"15","name":"Espresso coffee","description":"Cream espresso","price":"15","image":"middle3.png"},{"cat_id":"2","cat_name":"Hot Beverages","id":"16","name":"Ice\/Cold Coffee","description":"Super iced","price":"18","image":"middle.png"},{"cat_id":"3","cat_name":"Cold Beverages","id":"17","name":"Single Cup Brew","description":"Single cream","price":"20","image":"middle.png"},{"cat_id":"3","cat_name":"Cold Beverages","id":"18","name":"Caff\u00e9 Americano","description":"Cream coffee","price":"14","image":"middle1.png"},{"cat_id":"3","cat_name":"Cold Beverages","id":"19","name":"Caramel Macchiato","description":"Fresh caramel","price":"11","image":"middle3.png"},{"cat_id":"3","cat_name":"Cold Beverages","id":"20","name":"Standard black coffee","description":"Standard black","price":"8","image":"middle2.png"},{"cat_id":"3","cat_name":"Cold Beverages","id":"21","name":"Black Eyed Andy","description":"Super black andy","price":"23","image":"middle1.png"},{"cat_id":"3","cat_name":"Cold Beverages","id":"22","name":"Coffee of the Day","description":"Cimbalino well made!","price":"22","image":"middle3.png"},{"cat_id":"3","cat_name":"Cold Beverages","id":"23","name":"Cuban Shot","description":"Best bica in town","price":"14","image":"middle2.png"},{"cat_id":"3","cat_name":"Cold Beverages","id":"24","name":"Caf\u00e9 Latte","description":"Best \"meia de leite\" in town!","price":"11","image":"middle.png"}]

TESTE A INSTALAÇÃO DO FRAMEWORK

- Execute o comando

```
C:\XAMPP\htdocs\RECURSO> npm install
```

- Execute o comando

```
C:\XAMPP\htdocs\RECURSO> npm run dev
```

A partir do seu browser preferido vá ao seguinte URL

http://localhost:8080/~a12345/RECURSO/dist/

Deverá receber uma página web de boas-vindas

1. [3 valores] FUNCIONALIDADE “HOME”

A funcionalidade “Home” é página de rosto do site

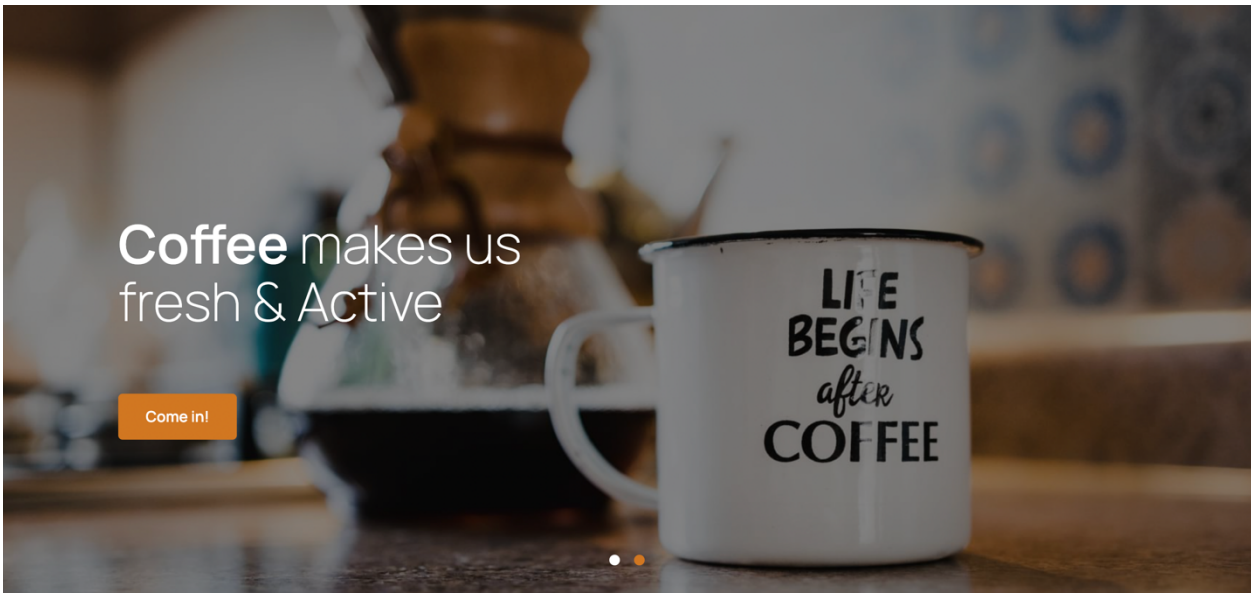
Construa a vista para esta página no ficheiro

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

SUGESTÃO: Adapte o template “`index.html`” fornecido pela empresa de web design. Encontra este template na pasta “demo” (`C:\XAMPP\htdocs\RECURSO\demo`).

Pode ver o template no browser no URL

`file:///C:/XAMPP/htdocs/RECURSO/demo/index.html`



“Welcome!/Come in!” é um link (“/menu”) que permite mostrar a página com o menu da cafetaria

NOTA IMPORTANTE:

- **os links** para os recursos estáticos css, fonts e javascript utilizados no template (`<link rel="stylesheet" href="... "> <script src="... ">`) **estão já colocados** no ficheiro "`index.html`" (**dado**)

- **os recursos estáticos** utilizados no template (images, css, fonts, js) **estão já colocados** em pastas com o mesmo nome (images, css, etc) dentro da pasta "`public`" (**dado**)

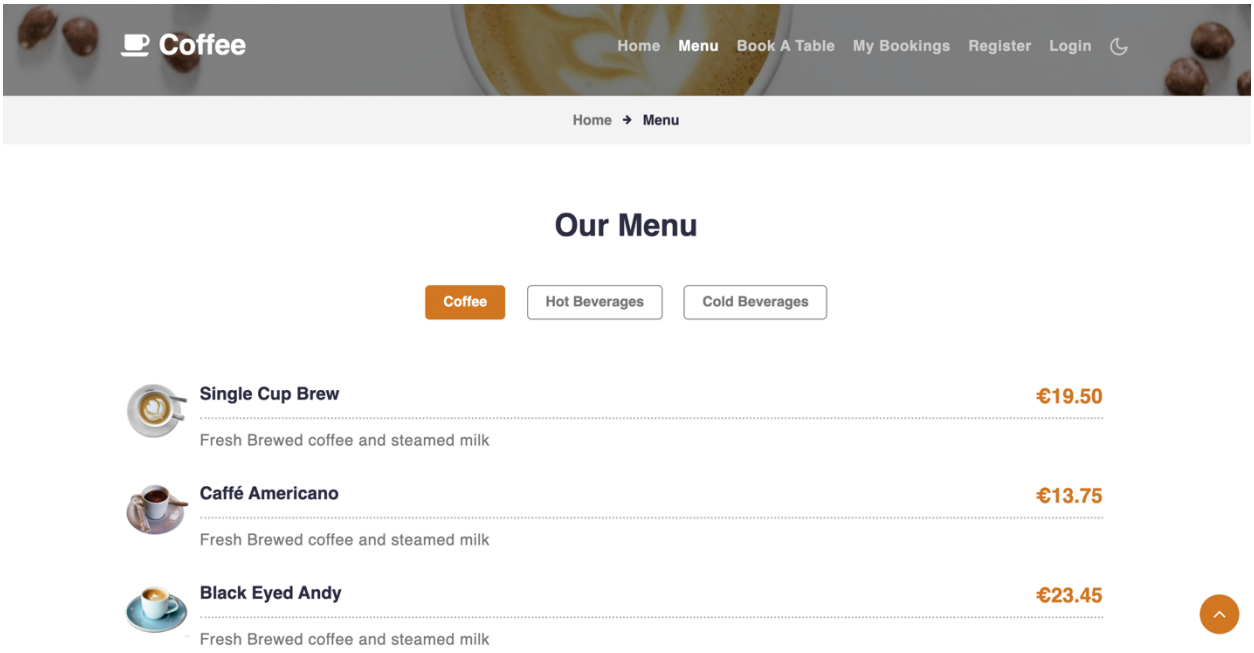
- Prefere construir o site com as imagens na pasta `src/assets`? Terá de copiar a pasta `demo/images` para dentro da pasta `assets`.

2. [5 valores] FUNCIONALIDADE “MENU”

A funcionalidade “Menu” é página principal do site e contem um menu/lista dos cafés disponíveis nesta cafetaria.

Construa a vista `Menu.vue` para esta página.

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



Utilize os métodos

```
beveragesStore.getBeveragesDB(),
beveragesStore.getBeveragesByCat(cat_id)
```

para fazer o download de uma lista dos cafés existentes na tabela “beverages” da base de dados

NOTA IMPORTANTE:

Note que o menu é controlado por três botões que permitem **mostrar à vez** cada uma das três categorias de bebidas existentes na tabela “beverages”. Estão claramente identificadas no template as três secções correspondentes às três categorias (content1, content2, content3).

O código javascript responsável pela tabulação já se encontra implementado: **apenas é necessário preencher o conteúdo das secções com as categorias de bebidas provenientes da base de dados.**

- **Pode considerar estáticas as três categorias de bebidas descritas nos botões**.

- Para cada bebida mostre a imagem, o nome, a descrição, e o preço (“name”, “image”, "description", "price" existentes na tabela “beverages”)

- “Register”, “Login” são links (“/register”, “/login”), apenas visíveis caso o cliente seja anonimo. Caso o cliente tenha feito login, transformam-se em “Logout”, “Book a table” (“/logout”, “/book”), e fica visível um outro link "My Bookings" ("/mybookings"), bem como aparece o texto “Welcome user” onde *user* é o login do cliente. Esta funcionalidade **já está implementada no componente "Header.vue" (dado)**

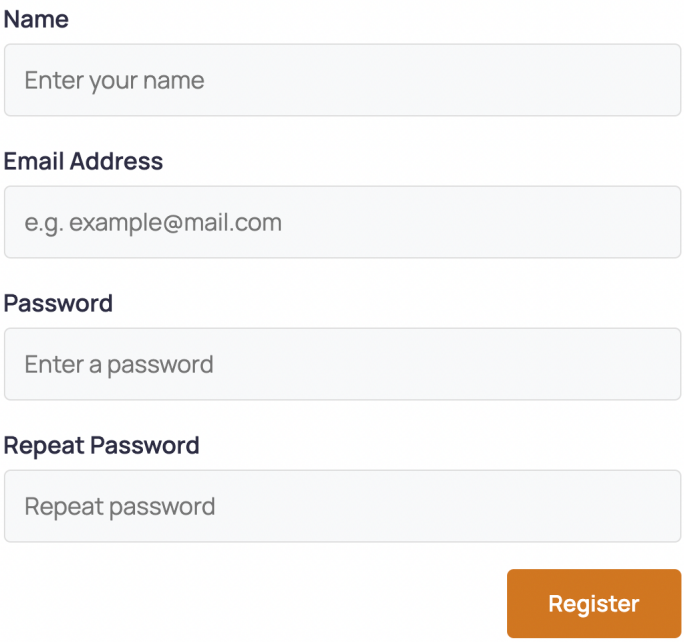
- São **dados dois componentes**: "Header.vue" e "Footer.vue"

3. [2 valores] FUNCIONALIDADE “REGISTER”

A funcionalidade “Register” permite registar um cliente.

Construa a vista `Register.vue` para esta página.

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



Utilize os métodos
`userStore.userExistsDB(user)` e
`userStore.addUserDB()`

para registar um novo cliente na base de dados

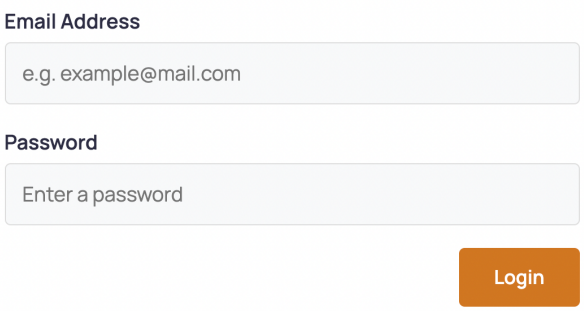
- Se o email ainda não existe na base de dados, regista o cliente na tabela “users”. Envia uma mensagem de sucesso ao cliente. Re-direccione para a vista “Menu”.
- Se o email já existe na base de dados deve aparecer uma mensagem de erro no formulário

4. [3 valores] FUNCIONALIDADE “LOGIN”

A funcionalidade “Login” permite autenticar um cliente.

Construa a vista `Login.vue` para esta página.

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



Utilize o método
`userStore.loginUserDB(user)`

para fazer o login do cliente e registá-lo na "store"

- em caso de sucesso no login:
  - Envia uma mensagem de sucesso ao cliente. Re-direcciona para a vista “Menu”
  - No menu o link “Login” transforma-se no link “Logout” e
  - No menu o link “Register” desaparece
  - Aparece no menu o texto “Welcome user!” (onde “user” é o nome do cliente registado)
  - No menu aparece um novo link “Book a table”
  - No menu aparece um novo link “My Bookings”
- em caso de insucesso no login
  - Mostra no formulário uma mensagem de erro “Wrong credentials”.

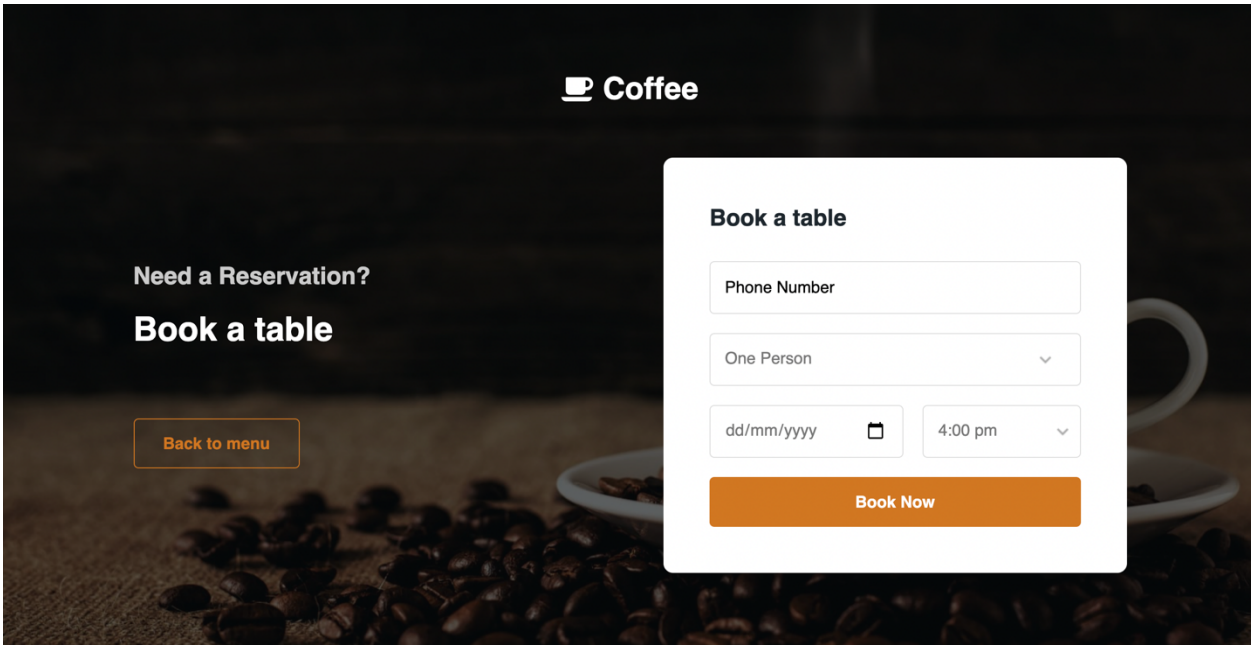
A funcionalidade "Logout" que encerra a sessão é **dada**: está implementada no componente "Header.vue"**.**

5. [4 valores] FUNCIONALIDADE “Book a table”

A funcionalidade “Book” permite ao cliente registado fazer uma reserva na cafetaria.
NOTA: Não é permitido ao cliente fazer mais do que uma reserva por dia[4].

Construa a vista `Book.vue` para esta página.

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



Utilize o método `bookingsStore.newBookingDB(reservation)` para fazer a reserva da mesa na tabela “bookings”.

- No caso de sucesso, envie a mensagem "Success, your booking is registered". Re-direccione para a vista "Menu".

- Caso o cliente já tenha uma reserva para esse dia deve aparecer essa informação no formulário, dando oportunidade ao cliente de escolher outro dia.

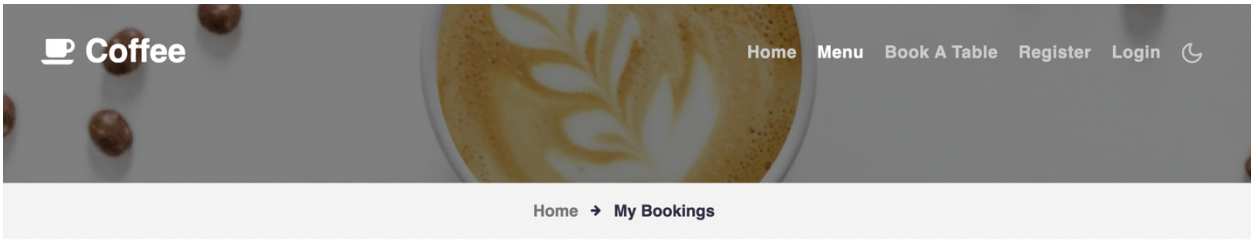
[4] Existe uma `UNIQUE KEY (`date`,`user_id`)` na tabela "bookings"

6. [3 valores] FUNCIONALIDADE “MY Bookings”

A funcionalidade “My Bookings” permite ao cliente registado consultar todas as reservas de mesa que efectuou

Construa a vista `MyBookings.vue` para esta página.

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



My Bookings

Table for one person **Date: 2022-01-01 Time: 4:00 pm**

Reservation date: 2022-01-01

Table for 4 people **Date: 2022-01-01 Time: 4:00 pm**

Reservation date: 2022-01-01

Utilize o método `bookingsStore.getMybookingsDB(client_id)`

para fazer uma lista das reservas existentes na tabela “bookings” da base de dados, em que “client_id” corresponde ao "id" do cliente que fez login.

BUILD/UPLOAD

Confirme/Atualize o “publicPath” no ficheiro

`C:\XAMPP\htdocs\RECURSO\vite.config.js`

com o conteúdo

`base: '/~a12345/EXAME/dist/'`

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

Execute o comando

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

Faça o upload com scp/WinSCP/FileZilla **das pastas**

- **"src"**
- **"dist"**

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

Teste o funcionamento do site no URL

http://daw.deei.fct.ualg.pt/~a12345/RECURSO/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

O acesso à base de dados MySQL pode ser feita em linha de comando (substitua "12345" pelo seu número de aluno)

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

ou ainda utilizando o software **phpMyAdmin** disponível no URL

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

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

CREATE TABLE `categories` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


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

INSERT INTO `categories` VALUES (1,'Coffee'),(2,'Hot
    Beverages'),(3,'Cold Beverages');

--
-- Table structure for table `beverages`
--

CREATE TABLE `beverages` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) DEFAULT NULL,
  `price` smallint(6) DEFAULT NULL,
  `cat_id` int(11) NOT NULL,
  `description` varchar(255) DEFAULT NULL,
  `image` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `cat_id` (`cat_id`),
  FOREIGN KEY (`cat_id`) REFERENCES `categories` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
--
-- Dumping data for table `beverages`
--

INSERT INTO `beverages` VALUES (1,'Single Cup Brew',20,1,'Fresh
    Brewed coffee and steamed milk','middle.png'),(2,'Caffé
    Americano',14,1,'Fresh Brewed
    coffee','middle1.png'),(3,'Caramel Macchiato',11,1,'Fresh
    Caramel','middle2.png'),(4,'Standard black
    coffee',8,1,'Fresh black ','middle3.png'),(5,'Black Eyed
    Andy',23,1,'Fresh black ','middle2.png'),(6,'Coffee of the
    Day',22,1,'Fresh coffee','middle3.png'),(7,'Cuban
    Shot',14,1,'Cream cuban','middle.png'),(8,'Café
    Latte',11,1,'Fresh latte','middle1.png'),(9,'Caramel
    Macchiato',20,2,'Fresh macchiato','middle3.png'),(10,'Coffee
    of the Day',50,2,'Fresh coffee','middle2.png'),(11,'Caffé
    Americano',23,2,'Fresh
    americano','middle.png'),(12,'Filtered Coffee',50,2,'Fresh
    filtered','middle1.png'),(13,'Cappuccino coffee',10,2,'Fresh
    coffee','middle2.png'),(14,'Cafe latte',10,2,'Cream
    latte','middle1.png'),(15,'Espresso coffee',15,2,'Cream
    espresso','middle3.png'),(16,'Ice/Cold Coffee',18,2,'Super
    iced','middle.png'),(17,'Single Cup Brew',20,3,'Single
    cream','middle.png'),(18,'Caffé Americano',14,3,'Cream
    coffee','middle1.png'),(19,'Caramel Macchiato',11,3,'Fresh
    caramel','middle3.png'),(20,'Standard black
    coffee',8,3,'Standard black','middle2.png'),(21,'Black Eyed
    Andy',23,3,'Super black andy','middle1.png'),(22,'Coffee of
    the Day',22,3,'Cimbalino well
    made!','middle3.png'),(23,'Cuban Shot',14,3,'Best bica in
    town','middle2.png'),(24,'Café Latte',11,3,'Best \"meia de
    leite\" in town!','middle.png');


--
-- Table structure for table `clients`
--

CREATE TABLE `clients` (
  `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 `email` (`email`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Table structure for table `bookings`
--

CREATE TABLE `bookings` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `created_at` datetime NOT NULL,
  `date` date NOT NULL,
  `time` varchar(255) NOT NULL,
  `phone` int(11) DEFAULT NULL,
  `n_persons` varchar(255) NOT NULL,
  `client_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `client_id` (`client_id`),
  UNIQUE KEY `date` (`date`,`client_id`),
  FOREIGN KEY (`client_id`) REFERENCES `clients` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

ANEXO 2. Descrição dos métodos disponíveis na API de 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

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

ou utilizando a extensão "THUNDER " do Visual Studio Code, ou utilizando o site https://postman.com

Tabela clients

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

Tabela categories

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

Tabela beverages

- **Todos as bebidas:**
  GET http://daw.deei.fct.ualg.pt/~a12345/RECURSO/api/beverages.php
  retorna: JSON string [{"cat_id":"1",cat_name":"Coffee","id":"1","name":"Single Cup Brew","description":"Fresh Brewed coffee and steamed milk", ...}, ...]

Tabela bookings

- **Todas as reservas do cliente:**
  GET http://daw.deei.fct.ualg.pt/~a12345/RECURSO/api/bookings.php?client_id=35
  retorna: JSON string [{"id":"1","created_at":"2021-12-03 18:20:31","client_id":"35","date":"2021-12-03", ...}, ...] ou null
- **Registo de uma nova reserva do cliente:**
  POST http://daw.deei.fct.ualg.pt/~a12345/RECURSO/api/bookings.php
  Body: JSON string {"client_id":"19","date":"2021-12-03","time":"4:00 pm", "phone":"99999999", "n_persons":"6"}
  retorna: JSON string {"id":"1","created_at":"2021-12-03 18:20:31",client_id":"19","date":"2021-12-03","time":"4:00 pm", "phone":"99999999", "n_persons":"6"}