|  |  |  |  |
| --- | --- | --- | --- |
|  | | Curso de TeSP em  Programação de Sistemas de Informação  Projeto em  Sistemas de Informação (PSI) | |
|  |  | |  |
| **Ano Letivo 2021/2022** |  | | **2º Ano, 1º Semestre** |
|  | | | |
| **Projeto em Sistema de Informação** | | | |



**Relatório de Especificação do SI**

**Healthyfi**

|  |  |
| --- | --- |
|  |  |
| **Grupo:** PL1 | **Data de entrega:** 2022-01-10 |
| **Nº** 2201087 | Pedro Miguel Miranda Lourenço |
| **Nº** 2201093 | Tiago Micael Dias Gil |
|  |  |

ÍNDICE

[1 Introdução 5](#_Toc92711368)

[2 Definição da Lógica de Negócio 6](#_Toc92711369)

[3 Funcionamento da API 7](#_Toc92711370)

[3.1 FirstController 7](#_Toc92711371)

[3.2 UserController 7](#_Toc92711372)

[3.3 ProfileController 8](#_Toc92711373)

[3.4 WorkedtimeController 8](#_Toc92711374)

[3.5 MealController 9](#_Toc92711375)

[3.6 CartController 9](#_Toc92711376)

[3.7 PaymentController 9](#_Toc92711377)

[3.8 ReviewController 10](#_Toc92711378)

[3.9 ReservationsController 11](#_Toc92711379)

[3.10 SalesController 11](#_Toc92711380)

[3.11 EmpsalesController 12](#_Toc92711381)

[3.12 TableController 12](#_Toc92711382)

# Introdução

O projeto *Healthyfi* tem como finalidade a criação de uma aplicação Web para gestão de *back-office* e *front-office* e uma aplicação android para a gestão de um restaurante. A ambição do projeto é melhorar e agilizar o funcionamento de restaurantes modernos a partir de meios digitais, eliminar por completo um intermediário entre o cliente e a cozinha, e permitir que o cliente faça o seu pedido baseado no número de calorias e macro-nutrientes que pretende consumir.

O back-office será utilizado para gerir a cozinha e a parte da administração, o front-office é utilizado apenas para fornecer informação ao cliente. A aplicação android tem como finalidade ser utilizada pelo cliente para várias tarefas como por exemplo o check-in no restaurante e fazer pedidos a mesa.

A aplicação Web vai ser desenvolvida na unidade curricular de Plataformas para Sistemas de Informação (PLSI), a aplicação android vai ser desenvolvida na unidade curricular de Acesso Móvel a Sistemas de Informação (AMSI).

A API (*Application Programming Interface*) que vai ser utilizada para fazer a comunicação entre as aplicações e a base de dados vai ser desenvolvida na unidade curricular de Serviços e Interoperabilidade de Sistemas (SIS).

# Definição da Lógica de Negócio

Deverá ser criado um sistema informático composto por uma aplicação web, uma aplicação móvel e uma API para permitir a comunicação entre as duas aplicações com a base de dados.

A aplicação web será desenvolvida na framework Yii2 com as linguagens de programação *PHP, CSS, HTML,* possibilitando a utilização da mesma em qualquer sistema operativo.

A aplicação móvel será desenvolvida para sistemas android na linguagem de programação JAVA, garantindo a compatibilidade com 94.1% dos dispositivos android.

O sistema dirige-se a todos os restaurantes que pretendam uma solução de gestão *all-in-one* que permita, em simultâneo, gerir a frente de casa, cozinha, stocks, funcionários e ainda a interação com o cliente. Desta forma evitasse assim a necessidade de intermediários, agilizando todas as operações no local de trabalho.

O sistema encontra-se repartido em 2 plataforma: Plataforma Web e uma Plataforma Móvel. A plataforma web terá 3 *acessos* possíveis, cliente, administração e cozinha, sendo que as zonas de administração e cozinha estarão protegidas por credenciais autorizadas. Na página principal do website apenas será apresentada informação para os clientes. A plataforma móvel terá 2 *acessos, o de cliente e de staff, aqui também o* login de staff estará protegido por credencias autorizadas.

Na plataforma web o acesso de cliente será composto pelas seguintes funcionalidades (nenhuma autenticação requerida):

* Qualquer pessoa poderá na página principal consultar a ementa para o dia da semana que escolher, poderá também visualizar as reviews; elaborar uma review apenas será possível através da aplicação móvel; também irá estar disponível uma secção de informações sobre o restaurante e uma outra sobre eventos;
* Na área de administração do restaurante será possível fazer alterações à ementa; aceder a vários dados estatísticos sobre as refeições (número de almoços e jantares, pratos mais pedidos) e de gestão financeira do estabelecimento; e possibilidade de responder às críticas dos clientes.

Na plataforma móvel terá dois tipos de autorizações: clientes e *staff.*

Através desta plataforma o *staff* poderá registar as entradas e saídas de expediente assim como outros dados referentes ao seu trabalho.

Os clientes poderão através da aplicação fazer reservas de mesa no estabelecimento e pedidos de entrega ao domicílio ou de *“take-away”.* Já no local, o cliente fará o registo da sua chegada através de leitura de “*QR-code*”. Efetuado o registo poderá fazer o pedido para a sua mesa através da ementa na aplicação. O pagamento poderá ser feito através da *app* ou *MBWay.* Após pagamento poderá fazer uma crítica ao serviço prestado. Pretende-se que apenas seja possível deixar uma crítica após o cliente experienciar o serviço de modo a combater falsas críticas sejam elas positivas ou negativas. O cliente terá ainda disponível na aplicação o seu histórico de pedidos e as críticas que efetuou.

# Funcionamento da API

## FirstController

Este controlador é responsável pelas funções de login e de registo de novos utilizadores

Apenas suporta métodos POST

|  |  |
| --- | --- |
| **Endpoint** | **api/first/login** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"email\":\"admin@hoje.pt\",\"password\":\"12345678\"}" http://localhost/healthify/backend/web/api/first/login |
| **Endpoint** | **api/first/register** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"username\":\"curlTeste\",\"email\":\"curl@test.pt\",\"password\":\"12345678\"}" <http://localhost/healthify/backend/web/api/first/register> |
| **Endpoint** | **api/first/register** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"username\":\"curlTeste\",\"email\":\"curl@test.pt\",\"password\":\"12345678\"}" <http://localhost/healthify/backend/web/api/first/register> |

## UserController

Através deste método podemos eliminar utilizadores

|  |  |
| --- | --- |
| **Endpoint** | api/user |
| **Método** | **DELETE** |
| **cURL** | curl -i -X DELETE http://localhost/healthify/backend/web/api/user/26 |

## ProfileController

Com este controlador podemos criar, ver, alterar ou apagar informações dos perfis de utilizador

|  |  |
| --- | --- |
| **Endpoint** | api/profile |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"nif\":\"987654321\",\"name\":\"Pedro\",\"cellphone\":\"987654321\",\"street\":\"fewf\",\"door\":\"21\",\"city\":\"Leiria\",\"userid\":\"9\"}" http://localhost/healthify/backend/web/api/profile |
| **Método** | **GET** |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/profile/2 |
| **Método** | **PUT** |
| **cURL** | curl -i -X PUT -H "Content-Type: application/json" -d "{\"nif\":\"987654321\",\"name\":\"TestePut\",\"cellphone\":\"987654321\",\"street\":\"fewf\",\"door\":\"21\",\"city\":\"Leiria\",\"userid\":\"9\"}" <http://localhost/healthify/backend/web/api/profile/2> |
| **Método** | **DELETE** |
| **cURL** | curl -i -X DELETE http://localhost/healthify/backend/web/api/profile/5 |

## WorkedtimeController

Este controlador aceita apenas dois métodos: GET e POST, devolve e registas os dias trabalhos pelo empregado

|  |  |
| --- | --- |
| **Endpoint** | api/workedtime |
| **Método** | **GET** – devolve todos os dias trabalhados pelo colaborador |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/workedtime/workedtime/1 |
| **Método** | **POST** – aceita apenas o id e regista o dia, hora de entrada e saída |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/workedtime/attendance/1 |

## MealController

Este controlador, para ser usado apenas para devolver pedidos GET, estando as outras funções desativadas no próprio, podemos pedir todas as refeições ou apenas uma adicionando o id da refeição

|  |  |
| --- | --- |
| **Endpoint** | api/meal |
| **Método** | **GET** – devolve todas as refeições introduzidas na ementa do restaurante |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" <http://localhost/healthify/backend/web/api/meal> |
| **Método** | **GET** – devolve apenas a refeição pretendida |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/meal/1 |

## CartController

Com este controlador podemos criar, ver, alterar ou apagar informações dos perfis de utilizador

|  |  |
| --- | --- |
| **Endpoint** | api/cart |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"userprofilesid\":1,\"mealsid\":1,\"sellingprice\":13.50,\"itemquantity\":2}" http://localhost/healthify/backend/web/api/cart |
| **Método** | **GET** |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/cart/1 |
| **Método** | **PUT** |
| **cURL** | curl -i -X PUT -H "Content-Type: application/json" -d "{\"itemquantity\":5}" http://localhost/healthify/backend/web/api/cart/2 |
| **Método** | **DELETE** |
| **cURL** | curl -i -X DELETE http://localhost/healthify/backend/web/api/cart/2 |

## PaymentController

Este controlador aceita um método POST com id de utilizador e número de cartão de pagamento

O número de cartão tem de ser um número de cartão valido permitido pelo sistema.

|  |  |
| --- | --- |
| **Endpoint** | api/payment |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"id\":1,\"mealsid\":1,\"reviewtext\":"estava\ boa\",\"rating\":5}" http://localhost/healthify/backend/web/api/review/createreview |

## ReviewController

Este controlador possui 1 método personalizado, “Fromuser” devolve todas as reviews feitas por um utilizador.

No modelo deste método situado em “backend\api\models\review” esta inserido o *publish* do *mosquitto* , assim sempre que é feita uma nova review por um utilizador é enviado para todos os utilizadores da aplicação. O publish foi implementado com o método “*afterSave()*”.

|  |  |
| --- | --- |
| **Endpoint** | api/review |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"userprofilesid\":1,\"mealsid\":1,\"sellingprice\":13.50,\"itemquantity\":2}" http://localhost/healthify/backend/web/api/cart |
| **Método** | **GET** |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/review/1 |
| **Método** | **PUT** |
| **cURL** | curl -i -X PUT -H "Content-Type: application/json" -d "{\"review\":\"Optima\"}" http://localhost/healthify/backend/web/api/review/1 |
| **Método** | **DELETE** |
| **cURL** | curl -i -X DELETE <http://localhost/healthify/backend/web/api/review/1> |
| **Endpoint** | api/review/fromuser |
| **Método** | **GET** |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/review/fromuser/1 |

## ReservationsController

Este controlador possui todos os métodos de *CRUD* e contem ainda um método *GET* extra para retornar todas as reviews de um determinado utilizador, o método post foi também substituído por um que nos possibilita fazer as verificações que pretendemos antes de adicionar uma nova entrada na base de dados

|  |  |
| --- | --- |
| **Endpoint** | api/review |
| **Método** | **GET** |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/reservations/21 |
| **Método** | **PUT** |
| **cURL** | curl -i -X PUT -H "Content-Type: application/json" -d "{\"reservedtime\":\"almoco\"}" http://localhost/healthify/backend/web/api/reservations/21 |
| **Método** | **DELETE** |
| **cURL** | curl -i -X DELETE <http://localhost/healthify/backend/web/api/reservations/21> |
| **Endpoint** | api/reservations/reserved |
| **Método** | **GET** |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" <http://localhost/healthify/backend/web/api/reservations/reserved/2> |
| **Endpoint** | api/reservations/new |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"reservedday\":\"2022-01-28\",\"reservedtime\":\"almoco\",\"tableid\":1,\"userprofilesid\":2}" http://localhost/healthify/backend/web/api/reservations/new |

## SalesController

Este controlador apenas é usado para retornar todas as compras feitas por um determinado utilizador

|  |  |
| --- | --- |
| **Endpoint** | api/sales/sold |
| **Método** | **GET** |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/sales/sold/1 |

## EmpsalesController

É um controlador que usa o mesmo modelo “*sales”* que o controlador acima referido, enquanto o “*SalesController*” *é destinado a compras feitas pela aplicação o “EmpsalesController”* é destinado a uso pelos funcionários do restaurante quando registarem os pedidos dos utilizadores

|  |  |
| --- | --- |
| **Endpoint** | api/empsales |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" -d "{\"precototal\":\"15.00\",\"discount\":\"0.00\",\"paidamount\":\"15.00\",\"paymentmethod\":\"card\",\"paymentstate\":\"not\",\"userprofilesid\":3}" http://localhost/healthify/backend/web/api/empsales |
| **Método** | **GET** |
| **cURL** | curl -i -X GET -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/empsales/1 |
| **Método** | **PUT** |
| **cURL** | curl -i -X PUT -H "Content-Type: application/json" -d "{\"paymentstate\":\"paid\"}" http://localhost/healthify/backend/web/api/empsales/9 |
| **Método** | **DELETE** |
| **cURL** | curl -i -X DELETE http://localhost/healthify/backend/web/api/empsales/8 |

## TableController

É o controlador que possibilita a gestão de mesas do restaurante, quando o cliente chega para se sentar à mesa e feito um scan de um código QR que comunica com a *API*  para estabelecer o estado de ocupação da mesa, como tal este controlador possui 3 métodos: Ocupar, Libertar e Reservar.

|  |  |
| --- | --- |
| **Endpoint** | api/tables/reservar |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/tables/reservar/2 |
| **Endpoint** | api/tables/ocupar |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/tables/ocupar/2 |
| **Endpoint** | api/tables/libertar |
| **Método** | **POST** |
| **cURL** | curl -i -X POST -H "Content-Type: application/json" http://localhost/healthify/backend/web/api/tables/libertar/2 |