

# Desenho e Implementação

## Conteúdos

<b>Desenho e Implementação</b>	<b>1</b>
<b>1 Introdução</b>	<b>1</b>
1.1 Sumário executivo	1
1.2 Controlo de versões	2
<b>2 Arquitetura do sistema</b>	<b>2</b>
2.1 Objetivos gerais	2
2.2 Requisitos com impacto na arquitetura	3
2.3 Decisões tomadas e fundamentação	3
2.4 Arquitetura proposta	3
2.4.1 Arquitetura lógica da solução	3
2.4.2 Interação entre módulos	4
2.4.3 Integrações com sistemas externos	4
<b>3 Ambiente de desenvolvimento e incrementos</b>	<b>5</b>
3.1 Arquitetura de instalação	5
3.2 Tecnologias de desenvolvimento	5
3.3 Incremento desenvolvido	5
<b>4 Histórias e critérios de aceitação</b>	<b>6</b>
4.1 Caraterização das <i>Personas</i> representativas	6
4.2 Histórias para a 1ª iteração da Construção ( <i>Construction #1</i> )	7
4.3 Automação de testes de aceitação	7
<b>5 Referências e recursos</b>	<b>7</b>

# 1 Introdução

## 1.1 Sumário executivo

Este relatório apresenta os resultados da segunda iteração da *Elaboration* e da fase de *Construction*, adaptada do método OpenUP, em que se constrói o produto ao longo de várias iterações. Os principais requisitos com impacto na arquitetura prendem-se com a segurança, rapidez de resposta por parte do sistema e gestão dos dados dos utilizadores.

Nesta iteração, consideramos prioritário implementar as bases de dados, bem como melhorar o backend do sistema visto que torna-se necessário de gerir os vários registos na webapp de forma eficiente e segura.

## 1.2 Controlo de versões

Quando?	Responsável	Alterações significativas
16/01	Diogo Amaral	2.3,4.1,4.2,3.3
16/01	José Luis Costa	2.1, 2.3, 2.4, 3.1, 3.2
16/01	Guilherme Pereira	3.3,4.3, 4.2, 4.1
15/01	Maria Cunha	2.2, 3.3, 4.1, 4.2

# 2 Arquitetura do sistema

## 2.1 Objetivos gerais

O objetivo principal desta arquitetura é criar uma aplicação versátil para a gestão por parte do restaurante, estafeta e a criação de pedidos por parte do cliente para a plataforma, através de uma webapp acessível para todas as plataformas.

Para o funcionamento desta arquitetura destacamos os seguintes objetivos:

- A plataforma tem que estar acessível em todos os dispositivos, de forma a que todos os clientes a possam utilizar.
- A webapp vai integrar uma plataforma de pagamentos eletrónicos através de um cartão bancário.
- Tanto o restaurante como o estafeta têm que aceitar um contrato específico durante o registo em que garantem um conjunto de restrições e taxas sobre os pedidos e, no caso dos restaurantes, uma garantia de qualidade dos produtos.
- O restaurante tem a possibilidade de alterar o seu menu disponível ao público, em função da disponibilidade dos produtos.

Para a utilização da aplicação disponibilizamos 1 conta de teste para cada tipo de utilizador:

- Cliente -> email: [luiscosta@gmail.com](mailto:luiscosta@gmail.com) | pwd: c1
- Restaurante -> email: [deti@ua.pt](mailto:deti@ua.pt) | pwd: c1
- Estafeta -> email: [afonso@estafeta.pt](mailto:afonso@estafeta.pt) | pwd: c1

## 2.2 Requisitos com impacto na arquitetura

Requisitos	Descrição
RDes.1	Garantir que todas as transacções MB demoram menos de 1 minuto.
RDes.3	Confirmação dos dados/quantidades da compra antes do pagamento
RSeg.1	Todos os dados introduzidos(cartão multibanco, morada, nome...) são confidenciais e protegidos.
RSeg.2	Todos os dados que o utilizador quiser guardar são armazenados na base de dados.
RSeg.3	Não existe tolerância para falhas, tanto de segurança como para armazenamento de dados.
RSisEx.2	Utilização de uma base de dados.

## 2.3 Decisões tomadas e fundamentação

Tendo em conta os objetivos para a arquitetura, e os requisitos levantados na Análise, foram tomadas as seguintes decisões para a implementação da webapp:

- Frontend implementado com html,css e javascript, utilizando libraries como o bootstrap 4. Foram utilizadas estas linguagens pois são a base do desenvolvimento web atual e a grande versatilidade e comunidade ativa no bootstrap.
- O backend da aplicação foi desenvolvido com flask em python, principalmente pelo conhecimento prévio da linguagem e framework em questão, mas também pela facilidade de programação. A biblioteca flask de python permite uma fácil interação com a webapp, tanto para a troca de informação, como para a sua apresentação na interface do utilizador.
- Para o armazenamento de toda a informação relativa à aplicação, foi utilizado uma base de dados desenhada em sqlite, e posteriormente utilizada com uma biblioteca de python, sqlite3, de forma a que a interação com a biblioteca flask fosse intuitiva.
- Utilização de um framework de pagamentos eletrónicos regulado por uma entidade externa (ifthenpay), que garante segurança, e uma documentação detalhada do funcionamento da api.
- Para a troca de informação utilizamos o modelo de transmissão de informação JSON, pela sua simplicidade e bom suporte na linguagem de programação em questão (python).

## 2.4 Arquitetura proposta

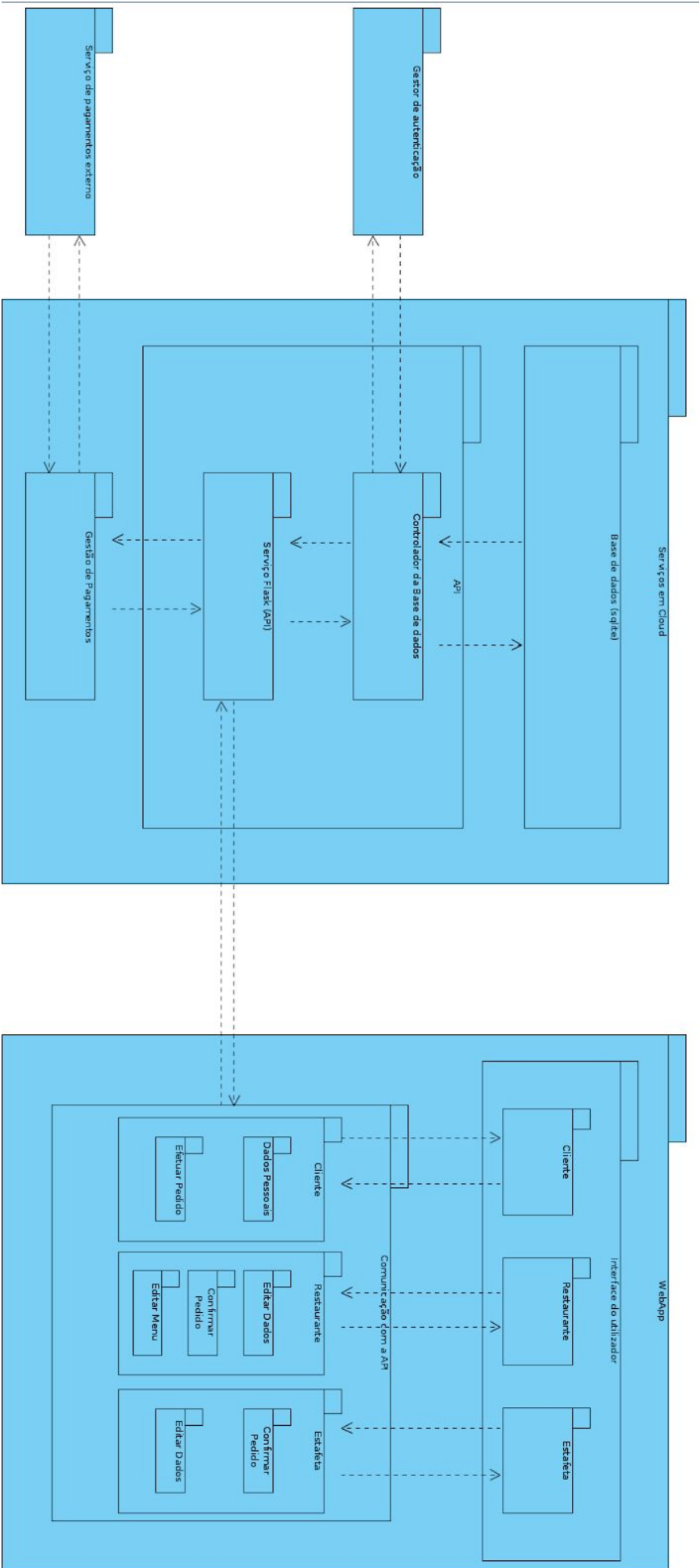
### 2.4.1 Arquitetura lógica da solução

**Serviço em Cloud** - O serviço em cloud representa todo o backend da aplicação que vai permitir as funcionalidades da webapp. Neste serviço, a componente mais importante é a API, que vai permitir o processamento e redirecionamento dos vários pedidos, não só da webapp, mas também de todas as entidades externas como o gestor de autenticação e o serviço de pagamento.

Salientamos também a relevância do controlador da base de dados responsável pela atualização, manutenção e segurança de todos os dados presentes na base de dados.

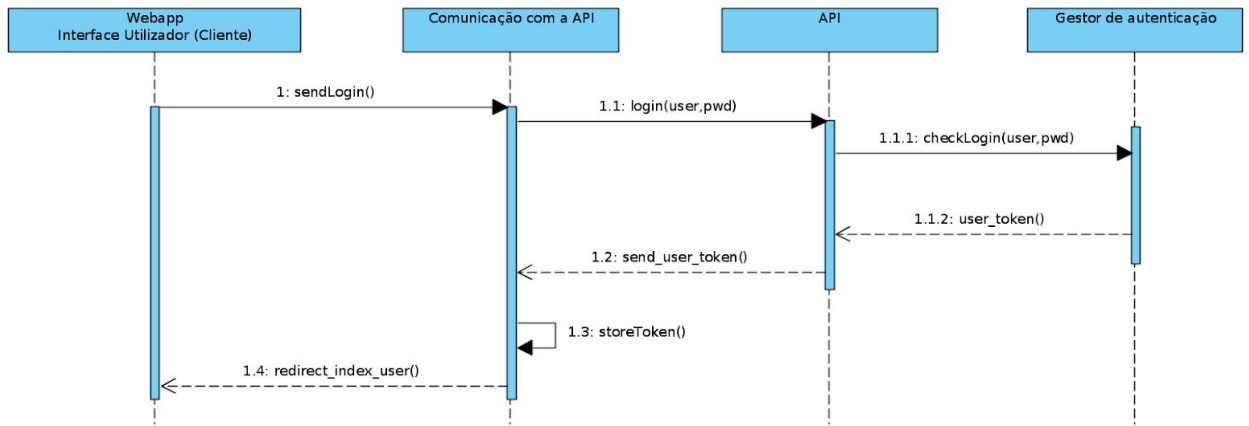
**Webapp** - Para o desenvolvimento desta componente, subdividimos as funcionalidades em 2 grandes componentes:

- **Interface do utilizador:** Toda a estrutura e funcionalidades de representação de informação sobre a qual todos os utilizadores do sistema vão interagir. Cada tipo de utilizador (Cliente, Estafeta, Restaurante) vai dispor de diferentes componentes e funcionalidades adaptadas aos requisitos do utilizador em questão.
- **Comunicação com a API:** Este módulo tem como objetivo a interação segura com a API, sendo que, como todos os tipos de utilizadores tem um conjunto de funcionalidades características/específicas, existem diferentes tipos de interações exclusivas para cada tipo de utilizador, de forma a que a comunicação com API seja segura (ex: Efetuar pedido pelo Cliente, editar o menu pelo restaurante, etc).

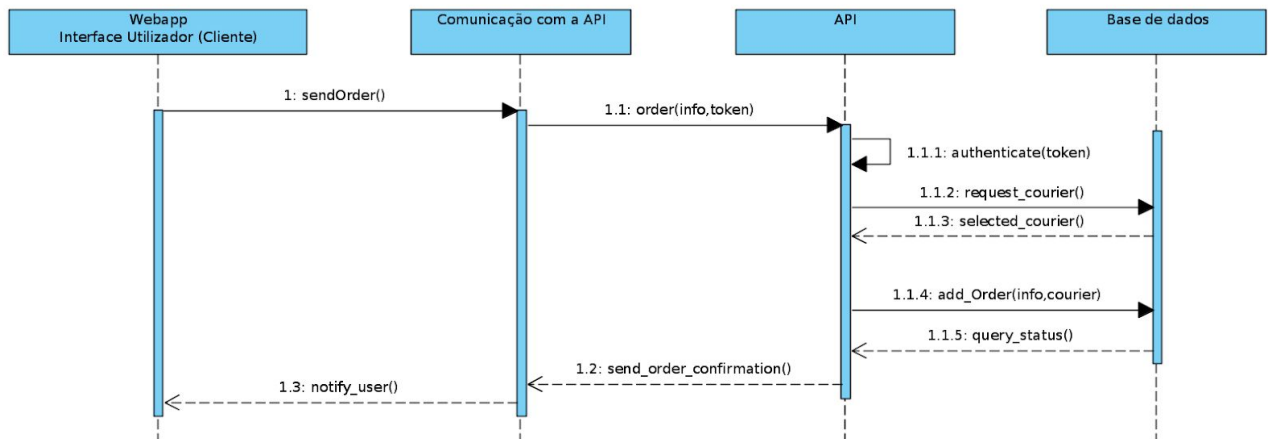


## 2.4.2 Interação entre módulos

### 2.4.2.1 Autenticação bem sucedida na aplicação

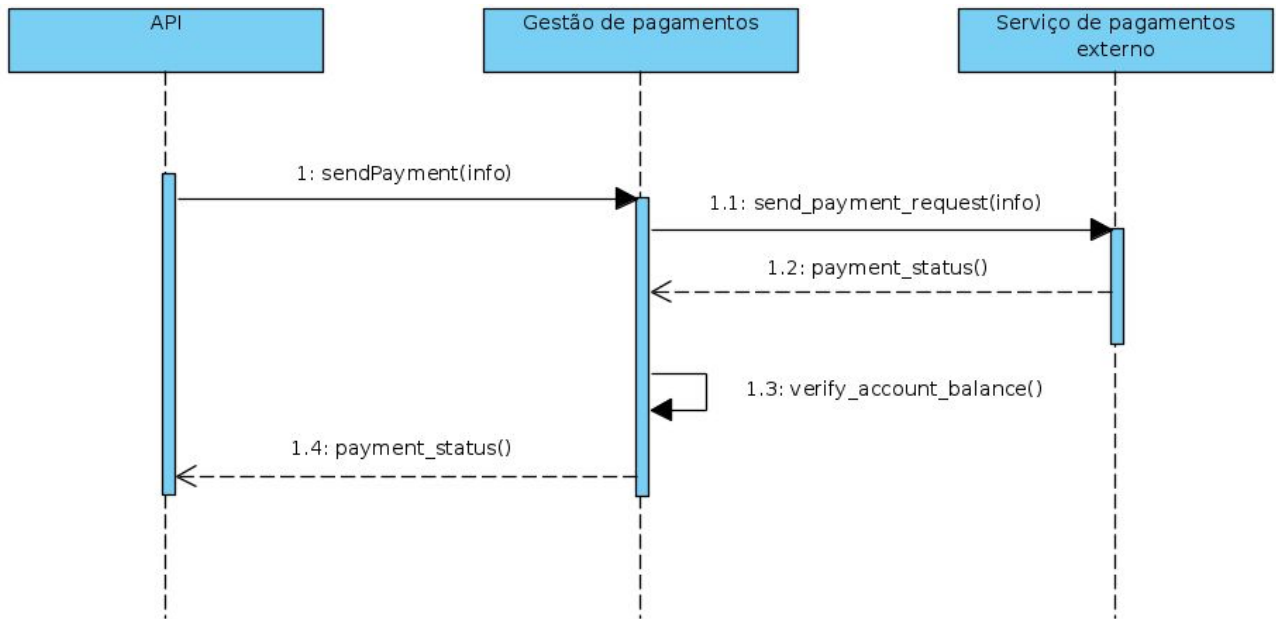


### 2.4.2.2 CaU 1 - Efetuar Pedido (Cliente)



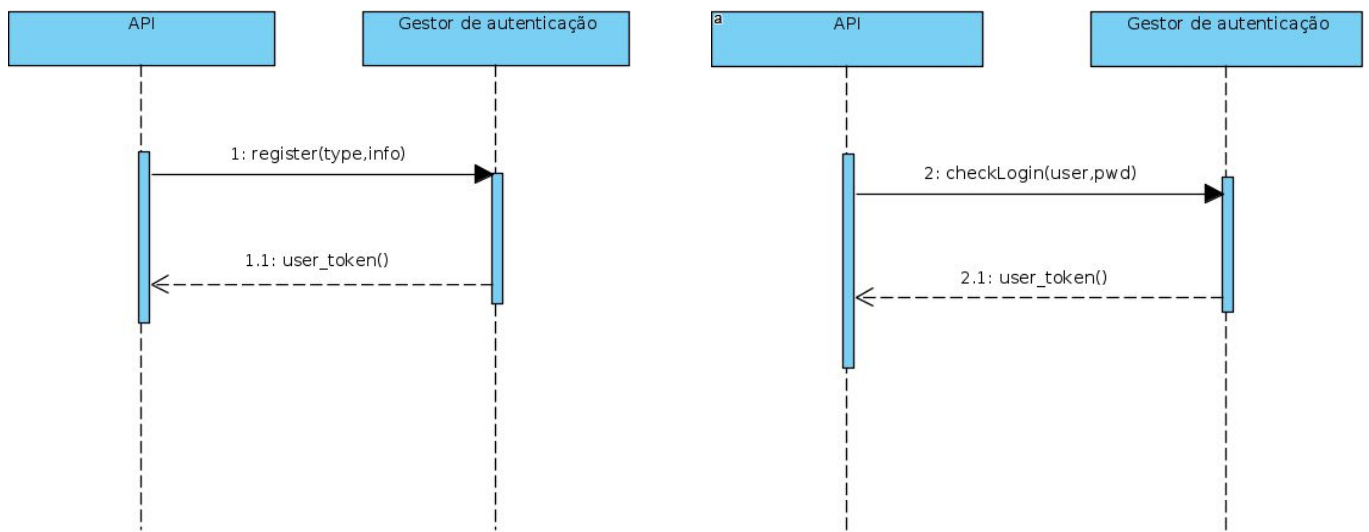
## 2.4.3 Integrações com sistemas externos

### 2.4.3.1 Integração com o sistema de pagamentos



Para o sistema de comunicação com o serviço de pagamentos externo, ilustramos um processo de pagamento processado pelo sistema, em que este envia um pedido no formato JSON para a entidade externa que se encarrega de realizar e confirmar o pagamento. Se a entidade externa confirmar o pagamento, então procederemos a uma verificação do saldo na conta bancária da empresa.

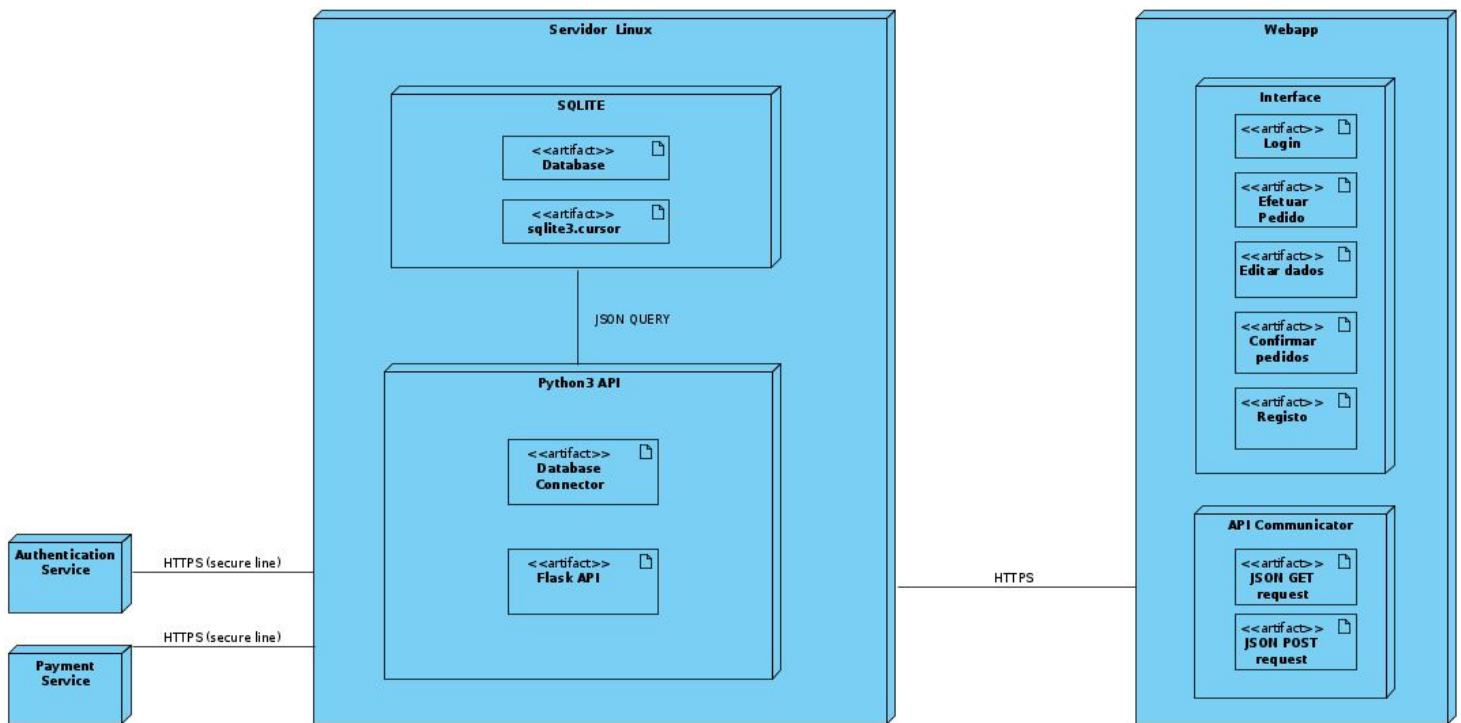
### 2.4.3.2 Integração com o sistema de autenticação



Para o sistema de comunicação com o gestor de autenticação, ilustramos o exemplo do login e registo de um utilizador na plataforma. Para autenticar/registar o utilizador, enviamos os dados via um pedido POST (codificado em JSON) para a api do gestor de autenticação, a sua resposta indica o token que identifica o utilizador e o seu tempo de sessão (codificado no token).

## 3 Ambiente de desenvolvimento e incrementos

### 3.1 Arquitetura de instalação



O serviço encontra-se alocado num servidor Linux (Ubuntu Server 18.4), onde corre tanto a api do sistema como a base de dados. O servidor Linux encontra-se configurado de forma a oferecer um deployment simples e rápido através da sincronização com o repositório git do website. Todas as comunicações externas, tanto para a plataforma de pagamento como para a de autenticação são asseguradas sobre uma ligação segura, através de pedidos no formato JSON. A webapp representa a aplicação que executa no computador do utilizador que, como mencionado anteriormente, comunica com a API do servidor.

O produto encontra-se implementado no seguinte endereço:

[http://websiteams.ddns.net/site.github.io/Micro-Site/Versao\\_3/index.html](http://websiteams.ddns.net/site.github.io/Micro-Site/Versao_3/index.html)

Para aceder à plataforma em questão deve clicar no botão de login na barra de navegação.

### 3.2 Tecnologias de desenvolvimento

Para a implementação deste projeto, foram utilizados as seguintes linguagens de programação / Frameworks:

- **Flask (python3):** Utilizamos o flask através do python3 que vai integrar o módulo da API (Esquema no ponto 2.4.1). Utilizamos o servidor flask não só para o processamento dos pedidos GET e POST efetuados pela Web app, mas também para interação com o módulo de comunicação com a base de dados, quando necessário.
- **SQLite:** Para a criação da base de dados utilizamos sqlite, e para a sua utilização uma biblioteca de python3 (sqlite3), para a introdução dinâmica de dados na mesma.
- **Javascript:** A utilização de javascript neste projeto foi maioritariamente para a comunicação com a api, por parte da webapp (Esquema no ponto 2.4.1), mas também para oferecer alguma funcionalidade dinâmica na interface do utilizador. Toda a informação apresentada dinamicamente na interface do utilizador, provém maioritariamente de pedidos e envios (GET e POST) de informação para a api, através desta linguagem.



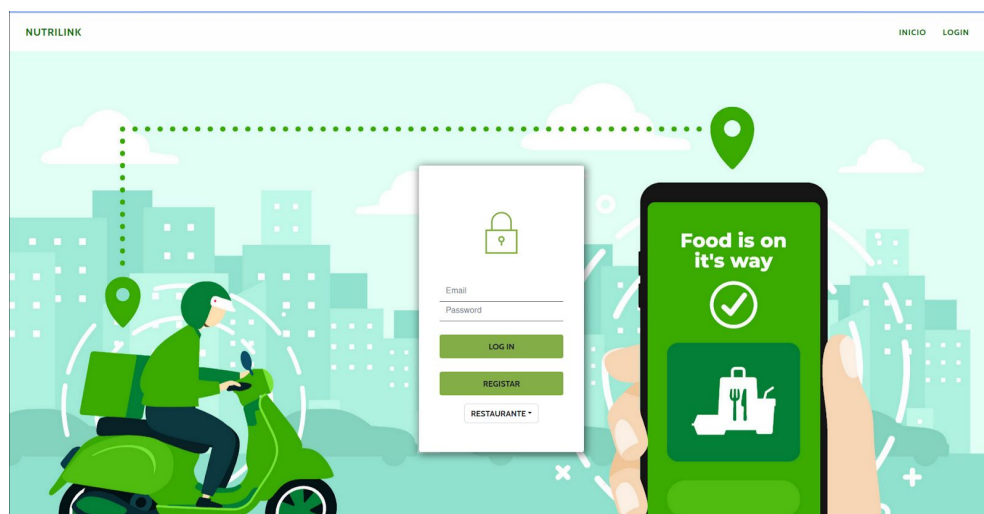
- **HTML/CSS/Bootstrap:** Estas linguagens foram utilizadas para toda a implementação da interface do utilizador (frontend da aplicação).

A comunicação entre os diversos componentes deste projeto é realizada através do modelo de transmissão de informação JSON. Todos os pedidos realizados entre o módulo da API e o módulo de comunicação com a API são codificados neste formato, mas também a comunicação com o sistema externo para o processamento de pagamentos e autenticações, tornando o processamento de dados simples e eficiente

### 3.3 Incremento desenvolvido

#### 3.3.1 Registar

Para o exemplo iremos considerar o registo de um restaurante. Partindo da página de login, o tipo de conta a ser criada é escolhida pelo dropdown menu (neste caso, é escolhido *Restaurante*) e é pressionado o botão *Registar* para continuar com o registo, sendo redirecionado para a página de registo.



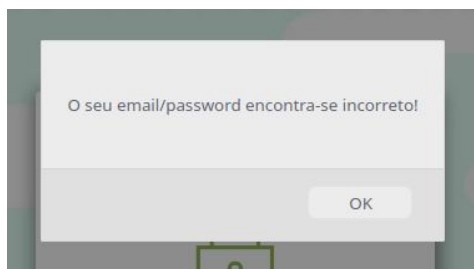
Na página de registo é necessário completar todos os campos, desde de dados pessoais a informações adicionais, dependendo do tipo de usuário a ser registado.

Preenchendo todos os campos do registo serão levados para a página principal do usuário agora registado. No caso de o usuário ser um Restaurante, na sua página principal poderá editar a sua ementa e aceitar pedidos feitos por clientes .

### 3.3.2 Logar-se

Para o exemplo iremos considerar o login de um estafeta. Partindo da página de login, o tipo de conta a ser acedida é escolhida pelo dropdown menu (neste caso, é escolhido *Estafeta*) e é pressionado o botão *Login* para confirmar o login, sendo redirecionado para a página principal do usuário caso o login esteja sido efetuado corretamente.

Caso os dados de login fornecidos estejam incorretos um pop up aparece avisando que o login não foi efetuado.



### 3.3.3 Alterar definições

Para o exemplo iremos considerar que um cliente pretende adicionar um novo método de pagamento. Partindo da página de login, o usuário faz o login e seguidamente será redirecionado para a sua página principal.

No menu no topo da página há um botão *Definições* que o levará para a página de edição das suas definições. Nesta página apareceram todos os dados associados à sua conta de usuário que poderá editar.

DADOS

Primeiro Nome:

Luis

Ultimo Nome:

Costa

Email:

luiscosta@gmail.com

Password

c1l2

Telemóvel:

967215556

Data de Nascimento:

09/09/1989

ENDEREÇOS

CASA ▾

Nome da morada:

Nome da morada

Endereço:

Endereço

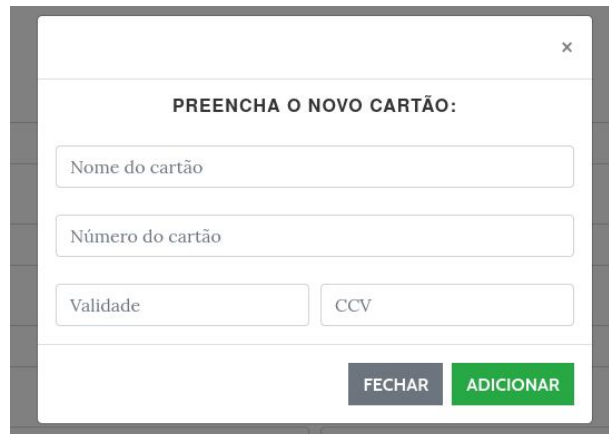
Código postal:

Código postal

Cidade:

Cidade

No drop down menu na seção de Pagamento seleciona-se a opção *Adicionar Cartão* e preencher os campos que aparecem no pop up.



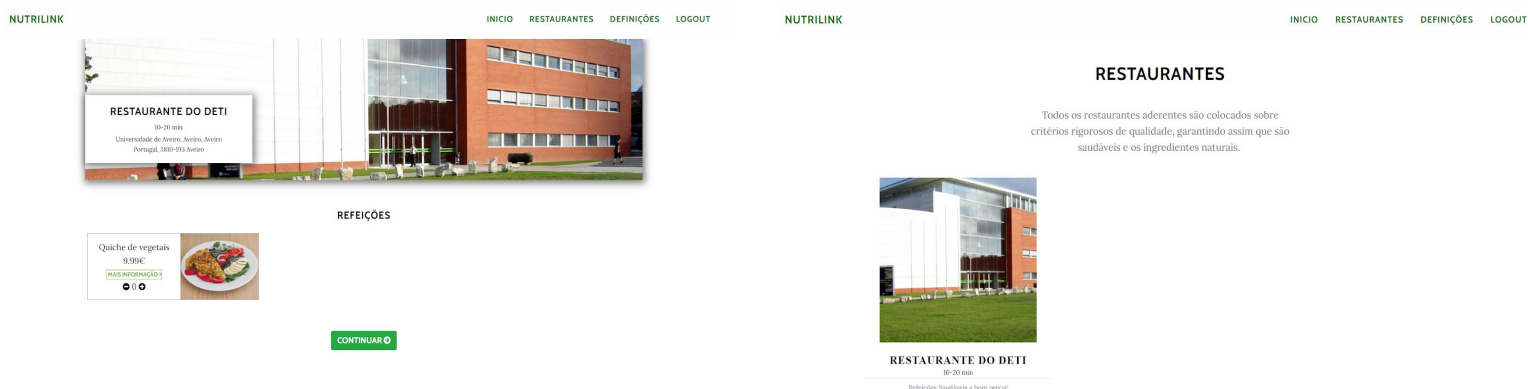
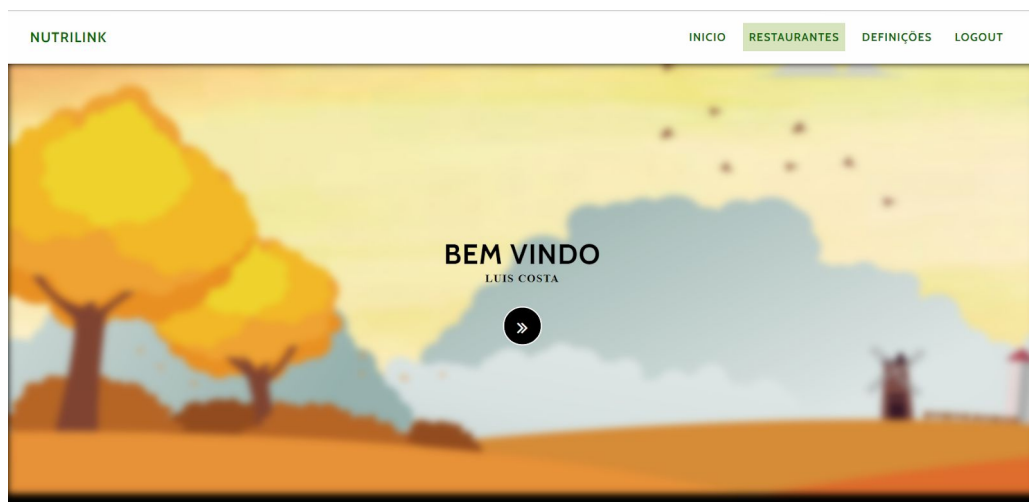
A modal window titled "PREENCHA O NOVO CARTÃO:" with a close button (X) in the top right corner. It contains four input fields: "Nome do cartão", "Número do cartão", "Validade", and "CCV". At the bottom right, there are two buttons: "FECHAR" (grey) and "ADICIONAR" (green).

Ao clicar em *Adicionar* o cartão agora criado será adicionado ao usuário.

### 3.3.4 Fazer um pedido

Para o exemplo iremos considerar que um cliente pretende fazer um novo pedido de refeição. Partindo da sua página inicial página inicial, o usuário irá a seção dos restaurantes e seguidamente será redirecionado a página que contém a lista de restaurantes disponíveis.

Seguidamente na página dos restaurantes vai escolher aquele que pretender, e será levado para a página específica desse restaurante. Após selecionar a quantidade de refeições que pretende, será redirecionado para a página de processamento do pagamento e confirmação de pedido, onde uma vez introduzidos os dados corretos será apresentado uma mensagem de conclusão de pedido e redirecionado para a página inicial.



PEDIDO

ENTREGA/PAGAMENTO

CONFIRMAÇÃO

PEDIDO

RESTAURANTE DO DETI

Dados:

Nome: Luis Costa

Tel: 967215556

Resumo:

1x Quiche de vegetais - 9€

Taxa de entrega - 1€

Total: 10€

>

### 3.3.5 Adicionar a ementa

Para o exemplo iremos considerar que um cliente com conta do tipo restaurante faz login e na sua página de restaurante seleciona o botão de editar ementa e na ementa pode selecionar o botão de adicionar produto ou remover uma refeição já existente, após preencher os campos pretendidos e selecionar uma imagem a refeição esta é adicionada a ementa.

RESTAURANTE DO DETI

Universidade de Aveiro, Aveiro, Portugal, Aveiro

3300-803 Aveiro

PEDIDOS PENDENTES

Pedido #4

MAIS INFORMACÃO

Pedido #5

MAIS INFORMACÃO

Pedido #6

MAIS INFORMACÃO

Pedido #7

MAIS INFORMACÃO

Pedido #8

MAIS INFORMACÃO

Pedido #9

MAIS INFORMACÃO

EDITAR EMENTA

RESTAURANTE DO DETI

19-20 000


Universidade de Aveiro, Aveiro, Portugal, 3300-803 Aveiro

REFEIÇÕES

Quiche de vegetais

9.99€

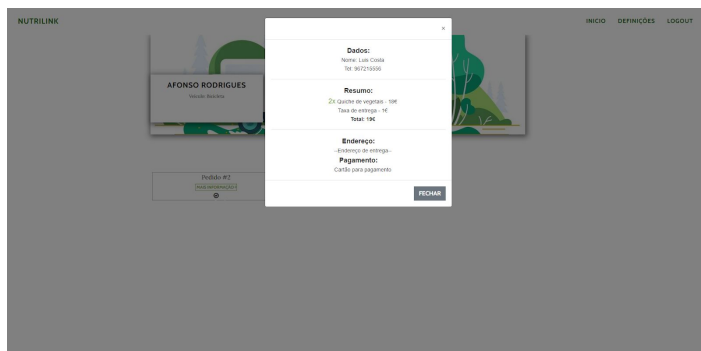
MAIS INFORMACÃO



ADICIONAR PRODUTO

### 3.3.6 Verificar um pedido por parte do estafeta

Para o exemplo iremos considerar que um estafeta faz login e na sua página de estafeta encontra os vários pedidos feitos por clientes, para ver as informações de cada pedido, o estafeta tem de clicar na opção “mais informações”, se o pretender aceitar, seleciona o botão com uma seta.



## 4 Histórias e critérios de aceitação

### 4.1 Caracterização das *Personas* representativas

#### *Persona 1: Artur*

Artur tem 32 anos e trabalha numa empresa informática em Aveiro, mais especificamente a criar WebSites e posteriormente a manter a manutenção destes. Também por causa do seu trabalho ele passa muito do seu tempo sentado e com maus hábitos de alimentação então para este novo ano ele decidiu criar uma meta, ter uma vida mais saudável.

#### *Persona 2: João*

O João de 26 trabalha em part time jobs desde que entrou no mundo do trabalho há 6 anos, optou por nunca seguir uma carreira estável (trabalho fixo) pois ele gosta de horários flexíveis para poder dedicar-se aos seus hobbies preferidos e também passar mais tempo com a sua noiva. Graças ao elevado número de trabalhos diferentes por o qual ele já passou, o João tem facilidade em adaptar-se a qualquer trabalho.

#### *Persona 3: Raquel*

A Raquel é proprietária de um restaurante de família. Ela como muitos outros proprietários de estabelecimentos de restauração tem sido desafiada por a situação da pandemia o que a fez procurar por serviços que a ajudassem a expandir o seu negócio de forma a gerar maior lucro. Como empreendedora e aventureira nas áreas do desporto a Raquel precisa de tempo longe do restaurante para poder frequentar os seus hobbies de forma a não pôr em causa o desempenho do restaurante.

## 4.2 Histórias para a 1ª iteração da Construção (*Construction #1*)

### **O Artur faz registo na webapp**

Sendo o Artur, um novo utilizador do nutrilink,

Quero poder registar-me na aplicação

De modo a poder utilizar os seus serviços disponíveis através da página do cliente

#### **Cenário 1:** Registo com sucesso

Dado que estou na página inicial da (link da página)

E seleciono o botão de login

Então sou levado para a página do registo, onde insiro os dados que me são pedidos.

Quando acabo de inserir os dados, são válidos.

Então sou levado para a página inicial do utilizador.

#### **Cenário 2:** Registo sem sucesso

Dado que estou na página inicial da (link da página)

E seleciono o botão de login

Então sou levado para a página do registo, onde insiro os dados que me são pedidos.

Quando acabo de inserir os dados, não são válidos.

Então sou levado para a página inicial aparecerá uma mensagem de erro.

### **O Artur faz login na webapp**

Sendo o Artur, um utilizador já com conta criada,

Quero poder entrar na aplicação

De modo a realizar o pedido com a comida escolhida do restaurante escolhido

#### **Cenário 1:** Login com sucesso

Dado que estou na página principal do site, introduzo o meu email e a minha password

Quando seleciono o botão do login

Então a página entra na página principal do usuário

#### **Cenário 2:** Login sem sucesso

Dado que estou na página principal do site, introduzo o meu email e a minha password

Quando seleciono o botão do login

Então a página mostra um erro de username/password errado

#### **O Artur faz um pedido**

Sendo o Artur um utilizador com conta criada  
Quero poder fazer um pedido de uma refeição a um restaurante  
De modo a receber essa refeição em casa

##### **Cenário 1:** pedido com sucesso

Dado que estou na página inicial do utilizador  
Quando seleciono o restaurante  
Então sou levado para a página do restaurante  
Quando seleciono o restaurante  
E insiro a informação de pagamento  
Então sou avisado que o pedido foi bem sucedido

#### **A Raquel adiciona refeições a ementa**

Sendo a Raquel, um utilizador com uma conta de um restaurante criada,  
Quero adicionar uma refeição a ementa  
De modo a atualizar os possíveis clientes das refeições disponíveis.

##### **Cenário 1:** Adição com sucesso

Dado que estou na página do restaurante  
E preencho os campos de informação da refeição  
Quando seleciono o botão para adicionar refeição  
Então sou redirecionado para a página do restaurante já atualizada com a nova refeição

#### **O João recebe um pedido**

Sendo o João um utilizador com uma conta de estafeta criada  
Quer confirmar um pedido já entregue  
De modo a notificar a aplicação que já fez o pedido

##### **Cenário 1:** confirmar o pedido

Dado que estou na página do estafeta  
Quando seleciono o botão de confirmar pedido  
Então o pedido é removido da página do estafeta

## **4.3 Automação de testes de aceitação**

Para realizar os testes de aceitação foi usado o Selenium IDE(Integrated Development Environment) que é uma ferramenta que serve para o desenvolvimento de testes por exemplo em aplicações web.

Estes testes encontram-se na pasta teste que vai junto na entrega do projeto.

Foram realizados testes para as histórias apresentadas na secção anterior.

É importante referir que os testes foram feitos no browser google chrome, pois no firefox há problemas com as pop-up windows o que faz com que alguns testes não funcionem corretamente nesse browser.

## Fazer Login na webapp

Para testar o login como cliente inserimos uma conta já criada(email:luiscosta@gmail.com, password:c1 na página de login que nos leva para a página inicial do utilizador.

## Teste de adicionar ementa

Project: TESTES\*

Tests ▾ +

Search tests... Q

http://websitemams.ddns.net/site.github.io/Website\_Funcional/Versao\_1/login.html

	Command	Target	Value
ADICIONAR_EMENTA			
FAZER_PEDIDO			
LOGIN			
REGISTAR*			
Receber_Pedido			
7	click	css=.login-dark	
8	click	id=pwd	
9	type	id=pwd	c112
10	send keys	id=pwd	\$(KEY_ENTER)
11	click	css=fa-arrow-circle-right	
12	click	css=.text-uppercase	
13	wait for element visible	id=name	70000
14	click	id=name	
15	type	id=name	COMIDA
16	wait for element present	id=preco	50000
17	click	id=preco	
18	type	id=preco	9
19	wait for element present	id=desc	50000
20	click	id=desc	

Para testar adicionar uma ementa começamos por dar login numa conta do tipo restaurante já criada (email:deti@ua.pt,password:c1) em seguida na página do restaurante adicionamos uma ementa preenchendo os campos pedidos.

É de notar que com este teste não testamos adicionar uma imagem pois assim o teste não correria em computadores diferentes.

## Teste para fazer pedidos

Project: TESTES\*

Tests ▾ +

Search tests... Q

http://websitemams.ddns.net/site.github.io/Website\_Funcional/Versao\_1/login.html

	Command	Target	Value
ADICIONAR_EMENTA			
FAZER_PEDIDO			
LOGIN			
REGISTAR*			
Receber_Pedido			
6	send keys	id=pwd	\$(KEY_ENTER)
7	click	css=.nav-item:nth-child(2) > .nav-link	
8	click	css=.img-fluid	
9	click	xpath=//i[@id='1']//2	
10	choose cancel on next confirmation		
11	click	id=botaoConfirm	
12	webdriver choose cancel on visible confirmation		
13	click	id=botaoConfirm	
14	click	css=.ml-auto:nth-child(1) > .fa	
15	mouse over	css=#form-content-1 .ml-auto	
16	mouse out	css=#form-content-1 .ml-auto	
17	click	css=#form-content-1 .ml-auto	
18	click	css=#sendOrder > .fa	

Para testar adicionar uma ementa começamos por dar login numa conta já criada, em seguida na página do utilizador selecionamos um restaurante já existente e em seguida escolhemos a refeição já existente e preenchemos os campos de confirmação de pedido.



## Teste para registar

Project: TESTES\*

Tests	Command	Target	Value
ADICIONAR_EMENTA	37 type	id=pais	Portugal
FAZER_PEDIDO	38 click	id=estado	
LOGIN	39 type	id=estado	Aveiro
REGISTAR*	40 click	css=.btn:nth-child(2) > .fa	
Receber_Pedido	41 click	id=nomeCartao	
	42 type	id=nomeCartao	Cartao UBER
	43 click	id=numeroCartao	
	44 type	id=numeroCartao	1234567891123456
	45 click	id=validadeCartao	
	46 type	id=validadeCartao	12/12
	47 click	id=ccvCartao	
	48 type	id=ccvCartao	705
	49 // click	css=#lastButton .fa	

Para testar o registo na página de login escolhemos o botão de registar e preenchemos os campos.

## Teste para receber pedidos

Project: TESTES\*

Tests	Command	Target	Value
ADICIONAR_EMENTA	1 open	http://websitemams.ddns.net/site.github.io/Website_Funcional/Versao_1/login.html	
FAZER_PEDIDO	2 set window size	1366x719	
LOGIN	3 click	css=.login-dark	
REGISTAR*	4 click	id=email	
Receber_Pedido	5 click	id=check	
	6 click	linkText=Estafeta	
	7 click	id=email	
	8 click	id=email	
	9 type	id=email	afonso@estafeta.pt
	10 click	id=pwd	
	11 type	id=pwd	c112
	12 send keys	id=pwd	\$(KEY_ENTER)

Para testar a situação em que o estafeta recebe pedidos para entregar começamos por dar login numa conta do tipo estafeta já criada (email:afonso@estafeta.pt,password:c1) onde somos levados para a página dos pedidos.

Numa fase inicial dos testes também era clicado no botão de confirmação de entrega do pedido, mas acabou-se por remover da versão final do teste, pois isto implicaria que havia sempre pedidos novos para ele entregar cada vez que era corrido o teste.

## 5 Referências e recursos

Ifthenpay manual de programador -> [https://www.ifthenpay.com/downloads/ifmb/Manual\\_Ifmb.pdf](https://www.ifthenpay.com/downloads/ifmb/Manual_Ifmb.pdf)

selenium faq -> <https://www.selenium.dev/selenium-ide/docs/en/introduction/faq>