

# Relatório e documentação do 1º projeto de AS

Realizado por: Ricardo Antunes, 98275

## Introdução

Para a cadeira de Arquitetura de Software, foi proposto a criação de um portal de saúde que permite aos clientes visualizar e gerir os seus registos médicos. A aplicação diferencia os níveis de acesso entre clientes e utilizadores do helpdesk, com códigos especiais para elevar temporariamente o acesso dos utilizadores do helpdesk. A aplicação é construída com um frontend React e um backend ASP.NET Core e é integrada ao OpenTelemetry para observabilidade. Foi utilizada a *template* React and ASP.NET Core fornecida pelo Visual Studio.

## Funcionalidades

### Autenticação:

Os clientes e os utilizadores do helpdesk podem fazer login na aplicação. Caso não tenham uma conta, podem registar-se.

A autenticação é necessária para aceder aos seus perfis, onde será apresentada os respetivos dados dependendo do tipo de utilizador.

### Gestão de registos médicos:

Os clientes podem visualizar e editar todos os seus dados pessoais.

Os utilizadores do helpdesk podem visualizar os dados dos clientes bem como os seus registos médicos e editar campos específicos com permissões especiais. Caso não tenham permissões especiais (código de acesso) apenas podem editar campos não mascarados.

### Códigos de Acesso Especiais:

Os clientes podem fornecer códigos especiais ao helpdesk para este poder obter acesso temporário a campos mascarados do cliente, podendo assim editar qualquer campo.

# Arquitetura

A aplicação segue uma arquitetura cliente-servidor, onde o frontend React interage com o backend ASP.NET Core por meio de APIs RESTful. O frontend é responsável por fornecer uma interface de utilizador intuitiva e responsiva, construída usando componentes React. A autenticação e o estado do utilizador são controlados por uma variável de estado chamada `loggedInUser`, que armazena os dados do utilizador autenticado. Essa variável é atualizada durante o login e o registo, e é utilizada para determinar o estado de autenticação e exibir o respetivo perfil do utilizador (cliente ou helpdesk).

As solicitações HTTP para o backend são tratadas pelos controllers, que lidam com operações como autenticação, acesso aos dados do(s) cliente(s) e atualizações de informações do cliente. O backend comunica com a base de dados para aceder e armazenar informações de clientes.

A integração com o OpenTelemetry é realizada no backend para fornecer recursos avançados de monitorização, rastreamento e registo de eventos importantes.

Para garantir a funcionalidade correta do mask/unmask de dados, foi incluído um projeto de testes utilizando a estrutura NUnit.

Em baixo, é exibida a Figura 1 que representa a arquitetura da aplicação.

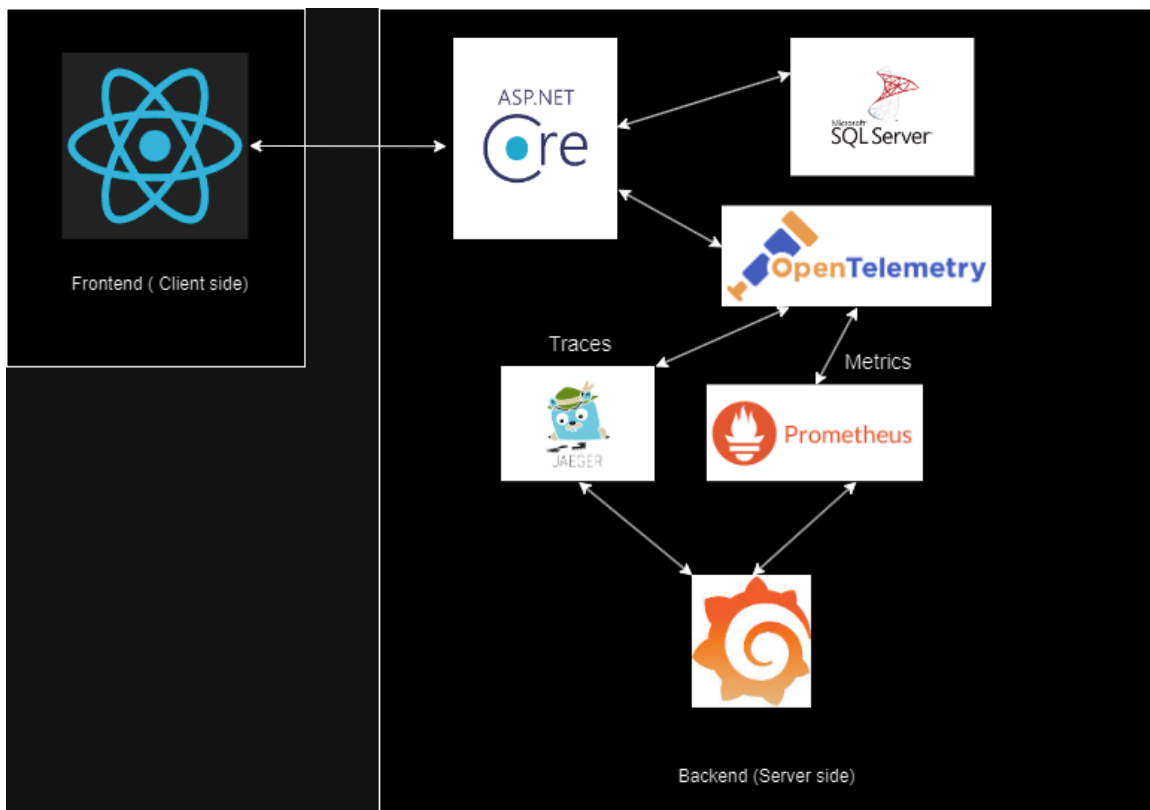


Figura 1: Arquitetura da aplicação.

## Integração com OpenTelemetry

A aplicação foi integrada com o OpenTelemetry para proporcionar uma monitorização abrangente e detalhada das operações e interações dentro do sistema. Por meio dessa integração, foram implementadas métricas, traces e logs, permitindo uma observabilidade eficaz das atividades e comportamentos do sistema.

As métricas (Figura 2) foram implementadas para rastrear o número de tentativas de login feitas pelos utilizadores (clientes e helpdesks) fornecendo insights valiosos sobre os padrões de uso e identificando possíveis ameaças à segurança, como ataques de força bruta. Essas métricas são essenciais para entender o comportamento dos utilizadores e garantir a segurança do sistema.

Para coletar e monitorizar as métricas, foi utilizado o Prometheus como exportador. Os dados coletados são posteriormente visualizados e analisados no Grafana, onde foi criado um dashboard personalizado.

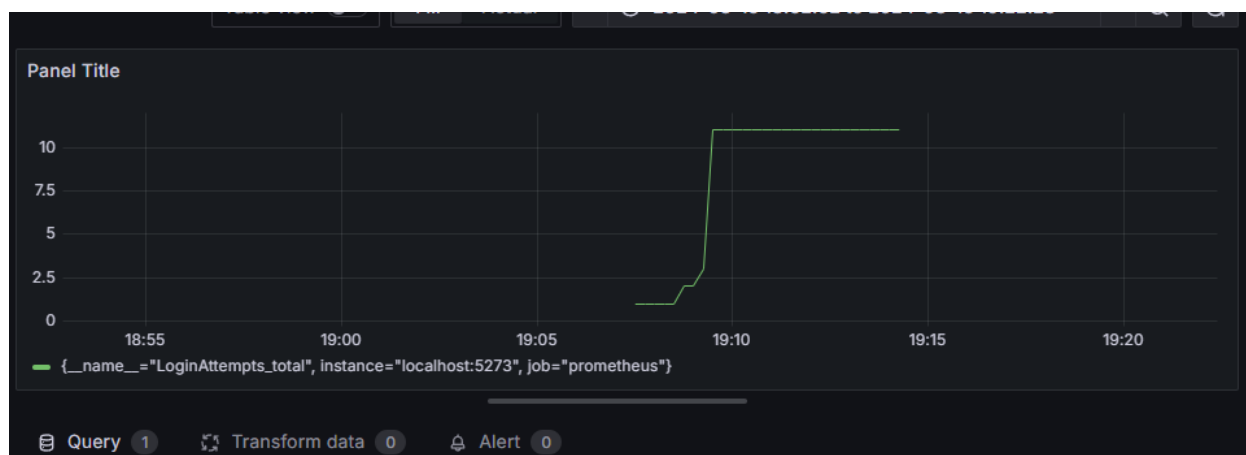


Figura 2: Métrica que regista o número de tentativas de login.

Os Traces (Figura 3) foram implementados para monitorizar as interações dos utilizadores com dados sensíveis, como acesso ou atualização de registos médicos. Para coletar e visualizar os traces, foi utilizado o Jaeger como servidor de rastreamento. Os dados recolhidos são posteriormente analisados e exibidos no Grafana.

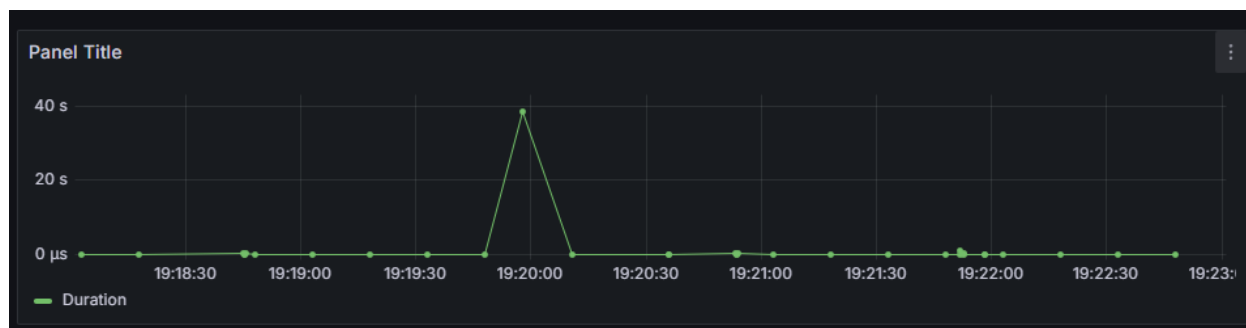


Figura 3: Traces – atualização de registos médicos.

Para além disso, foram implementados logs (Figura 4 e 5), que são exportados para a console, para registar eventos importantes, como tentativas de login, de validação de códigos de acesso pela equipe de suporte técnico e tentativas não autorizadas de acesso a campos mascarados. Esses logs são essenciais para identificar possíveis falhas de segurança ou anomalias no sistema.

```
info: AsProj1.Server.Controllers.LoginController[0]
      Login bem sucedido!!!!!!!!!!!!!!!!!!!!!!
Activity.TraceId:      d13a9a9c5ab736dc70b4b9ac00e3797c
Activity.SpanId:      62001ccd8f459702
Activity.TraceFlags:   Recorded
Activity.ActivitySourceName: Microsoft.AspNetCore
Activity.DisplayName:  POST api/Login
Activity.Kind:         Server
Activity.StartTime:    2024-03-15T18:09:09.7737596Z
Activity.Duration:     00:00:00.8167060
Activity.Tags:
  server.address: localhost
  server.port: 7095
  http.request.method: POST
  url.scheme: https
  url.path: /api/Login
  network.protocol.version: 2
  user_agent.original: Mozilla/5.0 (Windows NT 10.0; Win64; x
  http.route: api/Login
  http.response.status_code: 200
Resource associated with Activity:
  service.name: AsProj1.Server
  service.instance.id: f7dde690-e5f2-4dfb-8ac2-c8a751ec631c
  telemetry.sdk.name: opentelemetry
  telemetry.sdk.language: dotnet
  telemetry.sdk.version: 1.7.0
```

Figura 4: Log tentativa de login.

```
info: AsProj1.Server.Controllers.LoginController[0]
      Código de acesso incorreto!!!!!!
Activity.TraceId:      5ec1a3195ecfe3373195fef926f1e0f2
Activity.SpanId:      917a27d81b797452
Activity.TraceFlags:   Recorded
Activity.ActivitySourceName: Microsoft.AspNetCore
Activity.DisplayName:  POST api/CheckAccessCode
Activity.Kind:         Server
Activity.StartTime:    2024-03-15T18:12:59.3961353Z
Activity.Duration:     00:00:00.1737624
Activity.Tags:
  server.address: localhost
  server.port: 7095
  http.request.method: POST
  url.scheme: https
  url.path: /api/CheckAccessCode
  network.protocol.version: 2
  user_agent.original: Mozilla/5.0 (Windows NT 10.0; Win64;
  http.route: api/CheckAccessCode
  http.response.status_code: 400
Resource associated with Activity:
  service.name: AsProj1.Server
  service.instance.id: f7dde690-e5f2-4dfb-8ac2-c8a751ec631c
  telemetry.sdk.name: opentelemetry
  telemetry.sdk.language: dotnet
  telemetry.sdk.version: 1.7.0
```

Figura 5: Log validação dos códigos de acesso.

Assim, esta integração com o OpenTelemetry permite que a aplicação aproveite ao máximo as capacidades de observabilidade oferecidas pela framework.

## Base de dados

A base de dados foi projetada para armazenar e gerir informações relacionadas com os clientes, seus registos médicos e credenciais de autenticação.

Para tal, foram criadas 3 tabelas, `dbo.Users`, `dbo.Clients`, `dbo.MedReport`.

Na figura 6 está representada a estrutura da base de dados.

### `dbo.Users`

A autenticação dos utilizadores foi implementada manualmente e é mantida através da tabela `dbo.Users`.

Esta tabela é utilizada para guardar as credenciais do utilizador como também para identificar o tipo de utilizador, ou seja, se é cliente ou helpdesk.

- UserId: Chave primária usada para identificar utilizadores.
- UserMail: Email do utilizador.
- PasswordHash: Hash da password do utilizador.
- UserType: Indica o tipo de utilizador, como cliente ou helpdesk.

## dbo.Clients

Tabela dbo.Clients armazena os dados pessoais dos clientes.

- ClientId: Chave primária usada para identificar clientes.
- FullName: Nome do cliente.
- EmailAddress: Endereço de email do cliente.
- PhoneNumber: Número de telefone do cliente, mascarado com uma função de masking default.
- MedicalRecordNumber: Número de registo médico único para cada cliente. Foi aplicada a restrição Unique.
- AccessCodeHash: Hash do código de acesso do cliente.

## dbo.MedReport

Tabela dbo.MedReport armazena os registos médicos associados a cada cliente.

- MedReportId: Chave primária para a tabela MedReport, que identifica cada relatório medico
- MedicalRecordNumber: Chave estrangeira que faz referência ao MedicalRecordNumber da tabela dbo.Clients.
- DiagnosisDetails: Detalhes de diagnósticos de um cliente, mascarado com uma função de masking default.
- TreatmentPlan: Plano de tratamento de um cliente, mascarado com uma função de masking default.

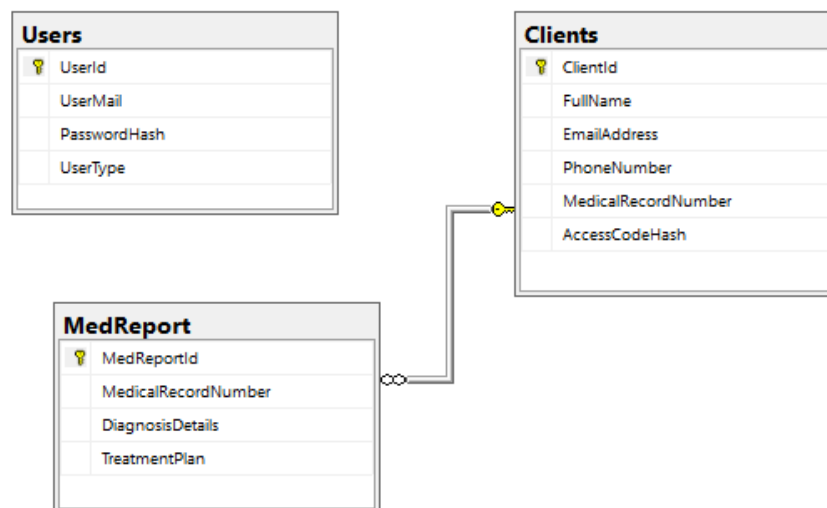


Figura 6: Diagrama da base de dados.

## Column masking and access controls

A implementação de *column masking* e *access controls* na aplicação desempenha um papel fundamental na garantia de segurança dos dados. Essas medidas são essenciais para proteger informações confidenciais, controlando quem pode acessá-las e como esses dados são apresentados.

Essas medidas foram aplicadas ao nível da base de dados, onde foram adicionadas máscaras às colunas das tabelas que contêm informações privadas.

Além disso, foram criados dois tipos de utilizadores na aplicação: clientes e helpdesks. Cada tipo de utilizador possui diferentes permissões de acesso aos dados. Os clientes têm acesso completo aos dados e podem editá-los conforme necessário. Por outro lado, os helpdesks têm acesso restrito e só podem visualizar informações sensíveis se fornecerem corretamente o código de acesso dado pelo cliente.

A lógica de controle de acesso começa quando um utilizador é registado no sistema. Por padrão, ao criar um utilizador, é definido como um cliente.

Uma vez autenticado com sucesso, o utilizador é redirecionado para o seu perfil, onde são exibidos os seus dados. Esse processo é realizado através de uma chamada a um endpoint específico da API, na qual o email do utilizador autenticado é enviado como parâmetro.

O endpoint, inicialmente começa por determinar o tipo de utilizador com base no email fornecido. Em seguida, é invocado um stored procedure, no qual ocorre a verificação do tipo de utilizador e a recuperação dos dados correspondentes consoante o tipo de permissões que o utilizador tenha.

Além da autenticação e da obtenção dos dados dos utilizadores, é relevante destacar que, quando um utilizador é helpdesk, inicialmente, ele apenas pode editar os campos não mascarados, pois não possui o código de acesso do cliente. Isso significa que ele não tem permissão para editar outros campos. No entanto, quando o código de acesso é inserido corretamente, o helpdesk obtém permissão para editar todos os campos. Todo esse processo ocorre por meio de um endpoint, que recebe a indicação se o utilizador inseriu corretamente ou não o código do cliente, juntamente com os dados atualizados do mesmo. Em seguida, um stored procedure é chamado, que, com base nesses argumentos, concede permissões necessárias ao utilizador para atualizar os dados do cliente, seguido pela revogação das mesmas.

## Casos de teste

Para comprovar o funcionamento correto do mask/unmasking, são fornecidas as Figuras 7 a 12.

Nas figuras 7 e 8 pode-se visualizar o perfil de um utilizador do tipo cliente. Este utilizador consegue visualizar e editar todos os seus campos.

**Bem-vindo!**

+ Adicionar Dados    Logout

Name: Ricardo Antunes  
Email: ricardo@ua.pt  
Phone Number: 939291145  
Medical Record Number: 200100221

Diagnosis Details	Treatment Plan
N/A	N/A

✎

Figura 7: Perfil do cliente.

Name:

Phone Number:

Save    Cancel

Figura 8: Edição do cliente.

Na figura 9, pode observar-se o perfil de um utilizador Helpdesk. Este utilizador consegue visualizar os dados dos clientes, mas com campos mascarados.

**Bem-vindo!**

Logout

Name	Email	Phone number	MedicalRecordNumber	Diagnosis Details	Treatment Plan	Options
Ricardo Antunes	ricardo@ua.pt	xxxx	200100221	xxxx	xxxx	
Tiago Coelho	tiago@ua.pt	xxxx	194195201	xxxx	xxxx	

Figura 9: Perfil do helpdesk.

A figura 10, demonstra o Helpdesk a tentar editar os dados de um cliente, sem ter inserido o código de acesso desse cliente, ou seja, apenas consegue editar os campos não mascarados.

X

Name:

Tiago Coelho

Medical record number:

194195201

Salvar

Figura 10: Edição de dados do helpdesk sem código de acesso.

Na figura 11 e 12, observa-se que o utilizador já inseriu o código de acesso do cliente “Tiago Coelho”. Assim, é apresentado todos os dados não mascarados sendo possível a edição dos mesmos.





Bem-vindo!						
						Logout
Name	Email	Phone number	MedicalRecordNumber	Diagnosis Details	Treatment Plan	Options
Ricardo Antunes	ricardo@ua.pt	xxxx	200100221	xxxx	xxxx	 
Tiago Coelho	tiago@ua.pt	965427411	194195201	N/A	N/A	 

Figura 11: Código de acesso do cliente Tiago Coelho

X

Name:

Tiago Coelho

Email:

tiago@ua.pt

Phone number:

965427411

Medical Record Number:

194195201

Diagnosis Details:

Treatment plan:

Salvar

Figura 12: Edição de dado