

23544 - Ana Beatriz Machado Carvalho

23548 - Ana Margarida Maia Pinto

23552 - Diana Alexandra da Costa Dinis

Saúde On Line

Aplicação web com base de dados em Postgres

Programação Web

Professor Doutor Patrícia Leite

Programação de Bases de Dados

Professor Doutor Joaquim Gonçalves

Licenciatura em Engenharia Informática
Médica

2022/2023

Índice

1. Introdução.....	1
2. Análise e projeto do sistema.....	2
2.5 Objetivos do projeto.....	2
2.2. Definição de negócio.....	3
2.3 Definição dos requisitos	5
2.4 Planeamento inicial do projeto	6
2.5 Mockup	9
2.5.1. Página principal.....	9
2.5.2. Signup e Login de doente e médico	11
2.5.2. Página principal do utente.....	13
2.5.3. Página principal do medico.....	15
3. Desenvolvimento da base de dados.....	17
3.1 Tabelas e relações	17
3.2 CRUD	17
3.2.1. Create.....	18
3.2.2. Read.....	20
3.2.3. Update.....	20
3.2.4. Delete.....	22
3.3. Views	23
4. Desenvolvimento da página web e API.....	25
4.1. Modal	25
4.3 Acordeão	27
4.3 API.....	28
5. Conclusão	29
6. Webgrafia	30

Índice de Figuras

Figura 1 - Diagrama de contexto da Saúde On Line.....	3
Figura 2 - Diagrama BPMN	4
Figura 3 - Arquitetura técnica da aplicação.....	7
Figura 4 - Diagrama entidade relação.....	8
Figura 5 - Área de signup.....	9
Figura 6 - Como funcionam os formulários	9
Figura 7 - Bloco introdutório às especialidades	10
Figura 8 - Informação mais detalhada sobre as especialidades, com os médicos	10
Figura 9 – Testemunhos.....	10
Figura 10 - Signup do médico	11
Figura 11 - Signup doente.....	12
Figura 12 – Login	12
Figura 13 - Formulário do doente	13
Figura 14 - Consultas possíveis para o doente.....	13
Figura 15 – Ver as consultas do doente	13
Figura 16 - Informações abertas do historico	14
Figura 17 - Informações abertas dos exames e prescrições	14
Figura 18 - Consultas futuras do doente	15
Figura 19 - Formulários por responder	15
Figura 20 - Formulário não respondido aberto	15
Figura 21 - Ver o histórico de formulários já preenchidos e consultas já dadas do médico.....	15
Figura 22 - Exemplo do histórico de formulários do médico	16
Figura 23 - Consultas futuras do médico	16

Índice de Tabelas

Tabela 1 - Regras de negócio	5
Tabela 2 - Pressupostos da aplicação.....	6
Tabela 3 - Restrições da aplicação	6
Tabela 4 - Requisitos não funcionais da aplicação.....	6

Siglas e Acrónimos

API - Application Programming Interface

CRUD - Create Read Update Delete

CSS – Cascading style sheets

HTML – Hypertext Markup Language

JS – JavaScript

SOL – Saúde On Line

1. Introdução

Este trabalho final enquadra-se nas unidades curriculares de Programação Web e Programação de Bases de Dados, lecionadas, respetivamente, pelos docentes Patrícia Leite e Joaquim Gonçalves do curso Engenharia Informática Médica do Instituto Politécnico do Cávado e do Ave. O objetivo deste será a realização de uma aplicação web de gestão de uma clínica médica Saúde On Line. Este trabalho irá incorporar os conteúdos lecionados nas duas disciplinas, assim como algum conhecimento extra de construção de APIs.

A base de dados será construída em Postgres, visto que é o utilizado em aulas e a página web será montada com recurso a html, css e js, assim como bootstrap. Para a API vai ser utilizado o Node.js.

2. Análise e projeto do sistema

Neste capítulo do trabalho estará todo o processo de análise e projeto da aplicação web a desenvolver.

2.1 Objetivos do projeto

Nesta parte estará exposto o objetivo do negócio e o seu enquadramento.

O objetivo do projeto é a realização de uma aplicação web para uma clínica médica “Saúde On Line”.

A aplicação permitirá criar um sistema de diagnóstico e marcação de consultas inovador, onde o utente apenas precisará de preencher um formulário com os seus sintomas que será avaliado por algum médico da especialidade requerida, sendo marcada uma consulta se necessário.

A aplicação deverá ser capaz de suportar funcionalidades como:

- Login e Sign up de médicos e utentes na aplicação;
- Utente preencher formulários, podendo consultar todos que já respondeu ou apenas os que ainda não foram respondidos;
- Médicos aceder aos formulários ainda não respondidos da sua especialidade e marcar consultas, assim como poder consultar todos os formulários já respondidos por ele;
- Prescrição de exames e medicação;
- Marcar nova consulta a partir de outra;
- Utente consultar as suas prescrições de exames e medicações, assim como consultas antigas, as que ainda estão por vir e conforme a especialidade;
- Médico consultar todas as suas consultas, também as consultas antigas, as que ainda estão por vir, todas as consultas que deu a um doente;

Este novo sistema permitirá uma maior facilidade ao acesso de consultas, sem que os utentes precisem de se deslocar para uma consulta de triagem à unidade clínica, provocando algum embaraço na unidade. Assim, podendo o médico estar no conforto da sua casa, pode responder aos formulários dos doentes, fazendo a triagem conforme os sintomas descritos no formulário. Durante ou após a consulta, poderá colocar observações da consulta, assim como prescrever exames e medicamentos.

Na Figura 1 está presente o diagrama de contexto da aplicação web:

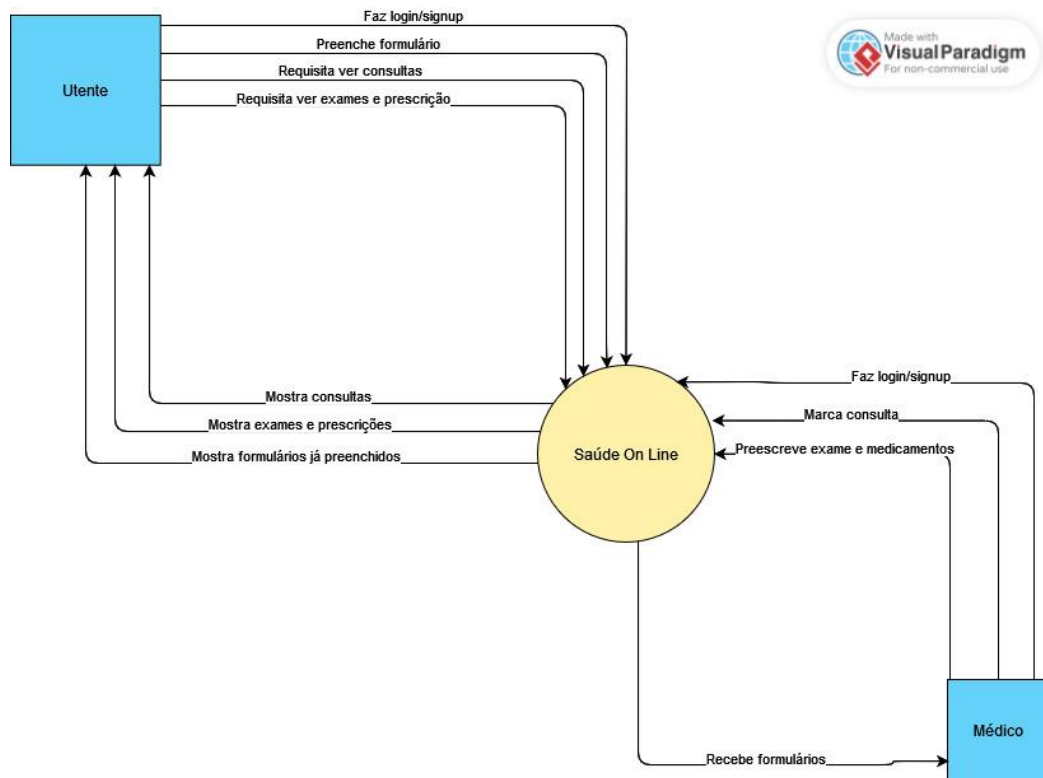


Figura 1 - Diagrama de contexto da Saúde On Line

2.2. Definição de negócio

O projeto terá os seguintes processos de negócio, estando o diagrama BPMN disponível na Figura 2:

- Quando o utente necessitar de consulta, faz login na aplicação web da SOL. Se já tiver conta, entra na sua página inicial. Caso não tenha, terá de fazer o signup;
- Ao entrar, na página terá o formulário para preencher com os seus sintomas e, mais em baixo, terá as opções de ver os outros formulários já preenchidos, as suas prescrições e exames e consultas;
- O médico inicia sessão na mesma aplicação web, caso tenha conta. Se não, fará o login;
- Terá acesso aos formulários não respondidos da sua especialidade e poderá consultar cada uma das suas consultas, as ainda não dadas e as já dadas;
- Após avaliar um formulário, decide se marca consulta ou não, escolhendo a data e a hora;
- Após a consulta, o médico poderá então adicionar observações e prescrever exames e/ou medicamentos;
- O utente poderá consultar essas prescrições.

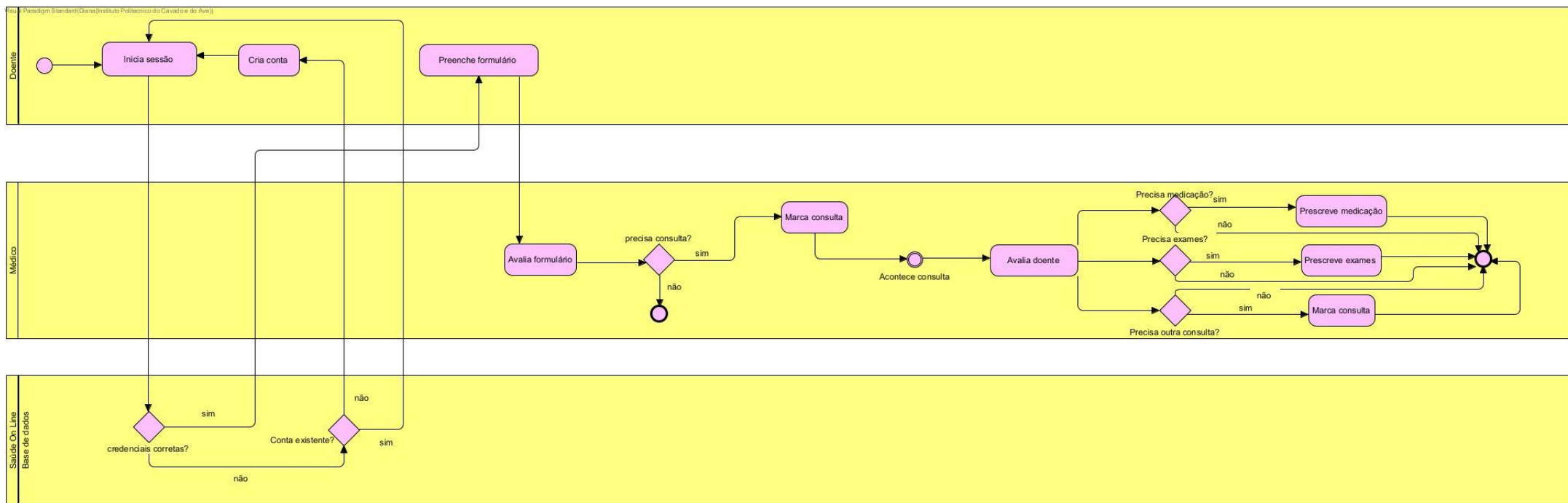


Figura 2 - Diagrama BPMN

As classes têm relações entre si. Um utente preenche um ou mais formulários, sendo que este apenas pode ser preenchido por um utente e visionado por um médico, que poderá visionar mais do que um formulário. A partir do formulário, o médico pode ou não dar uma consulta que pode ser dada por apenas um médico. Uma consulta pode ou não originar outra consulta e, dentro desta, pode ser ou não prescrito uma ou várias prescrições e passar ou não um ou vários exames.

Os atributos das classes são os seguintes, estando presente na figura o modelo de domínio:

Utente: Nome (*string*), número telemóvel (*int*), nif (*int*), email (*string*), concelho (*int*), distrito (*int*);

Médico: Nome (*string*), especialidade (*int*), email (*string*);

Formulário: Descrição (*string*), especialidade (*int*);

Consulta: Horário (*timestamp*), observações (*string*);

Exame: Nome (*string*);

Prescrição: Nome (*string*).

2.3 Definição dos requisitos

Na Tabela 1 está presente as regras de negócio

RN1	Os utentes apenas podem preencher um formulário caso estejam com sessão iniciada.
RN2	Um utilizador (médico e utente) não pode ter mais do que uma consulta agendada no mesmo horário.
RN3	Os utilizadores devem permissões adequadas para acessar e visualizar diferentes tipos de informação.
RN4	O sistema deve manter um registo do histórico de consultas de cada utente, incluindo data, médico responsável e observações.
RN5	Os médicos devem ter acesso ao histórico de consultas dos pacientes para análise e referência durante o atendimento.

Tabela 1 - Regras de negócio

Na Tabela 2 estão os pressupostos:

PC1	Os utilizadores terão acesso à internet.
PC2	A aplicação pressupõe que terá uma base de dados para armazenar as informações dos utentes, consultas, médicos e outros dados provenientes das consultas.

PC3	Os médicos terão acesso às informações médicas dos utentes.
PC4	Os utilizadores cumpram as suas responsabilidades ao utilizar a aplicação, como fornecer as informações corretas.
PC5	Os utentes poderão sempre ir às consultas agendadas.

Tabela 2 - Pressupostos da aplicação

Na Tabela 3 estão as restrições:

R1	A aplicação deve ser compatível com todos os browsers.
R2	A aplicação limita-se a determinados campos médicos ou áreas de atuação específicas.
R3	Não se pode receitar duas vezes o mesmo medicamento nem exame em uma consulta
R4	Um formulário apenas pode dar origem a uma consulta

Tabela 3 - Restrições da aplicação

Na Tabela 4 estão os requisitos não funcionais:

RNF1	A interface da aplicação deve ser intuitiva e de fácil utilização.
RNF2	A aplicação deve ser capaz de suportar o aumento do número de utilizadores
RNF3	A aplicação deve ter um bom desempenho em diferentes plataformas
RNF4	O estilo visual do produto ou aplicativo deve ser consistente em todas as páginas e elementos de interface do utilizador, podendo incluir uma paleta de cores, fontes e ícones.
RNF5	O código da aplicação deve ser bem documentado e seguir boas práticas de desenvolvimento para facilitar a compreensão e a realização de alterações.

Tabela 4 - Requisitos não funcionais da aplicação

2.4 Planeamento inicial do projeto

A arquitetura técnica é uma representação da estrutura e organização de um sistema de software. É criada durante a fase de design do desenvolvimento de software, após a compreensão dos requisitos do sistema e das necessidades dos utilizadores. Ela é projetada para garantir que o sistema seja eficiente, de fácil manutenção e atenda aos objetivos de negócio.

O Front-End é responsável pela interface com o usuário e pela apresentação visual do sistema. No desenvolvimento do front-end, vai ser utilizado HTML, CSS e JS para desenvolver as páginas, com o auxílio da biblioteca de CSS Bootstrap.

O Front-End comunica com a base de dados por meio de uma API em Node.js, enviando solicitações para obter ou enviar dados. O back-end recebe as solicitações, interage com a base de dados e envia as respostas de volta ao front-end. A base de dados é responsável pelo armazenamento persistente dos dados do sistema. Foi escolhido o PostgreSQL pois foi o utilizado durante as aulas.

Na Figura 3 está presente o diagrama da arquitetura técnica da aplicação:

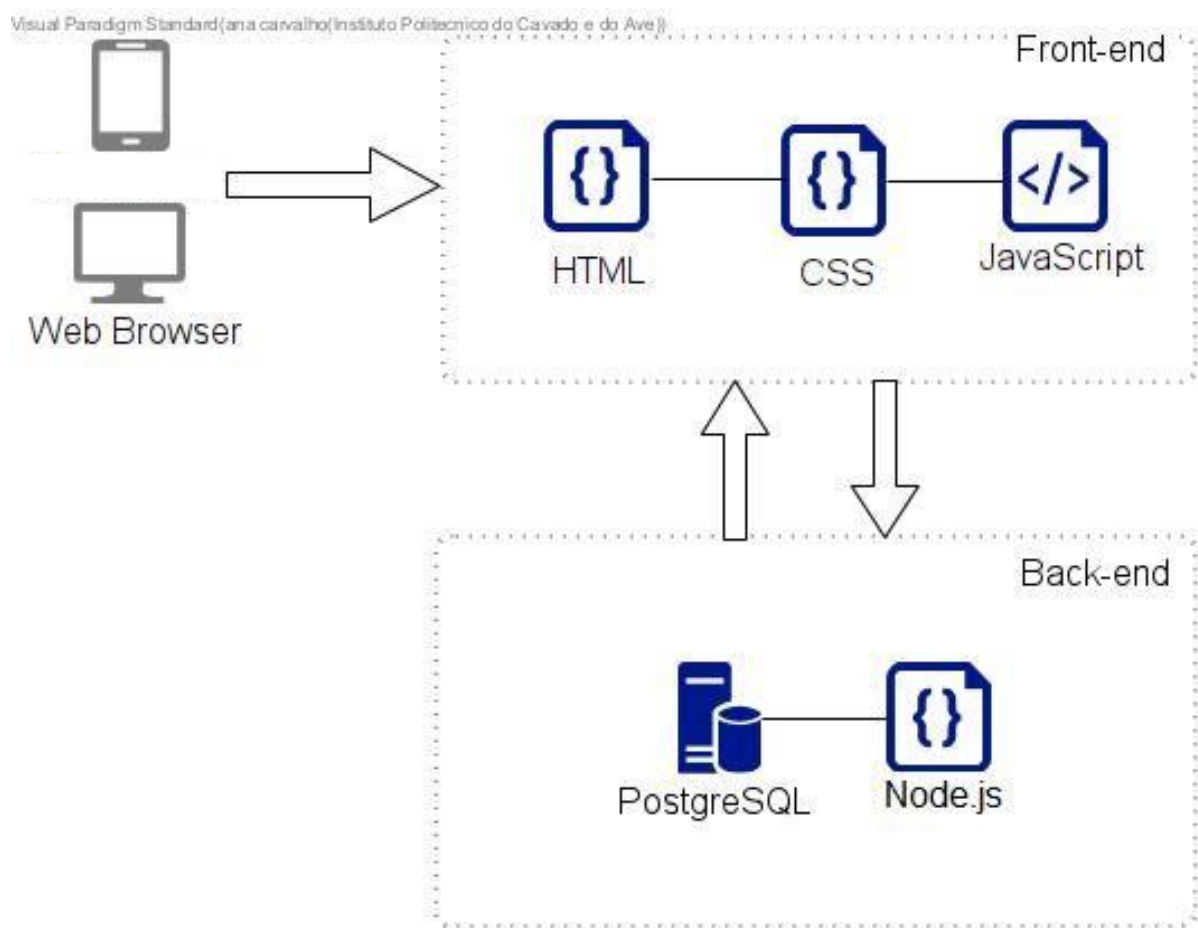


Figura 3 - Arquitetura técnica da aplicação

Já o diagrama de entidade relação da base de dados está presente na Figura 4:

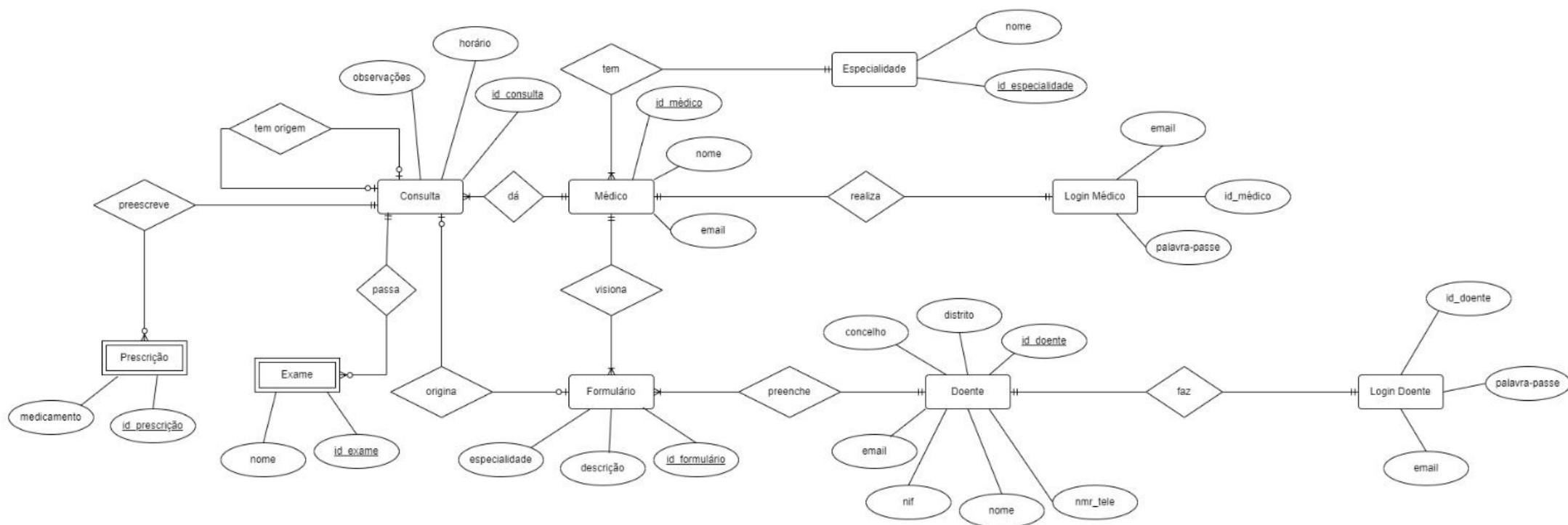


Figura 4 - Diagrama entidade relação

2.5 Mockup

Por fim, resta apenas o mockup das páginas, que vai da Figura 5 à Figura 23.

2.5.1. Página principal

A página principal terá a secção de se registar para o utente e para o médico (Figura 5), também contendo os passos para utilizar o site (Figura 6). Terá um pequeno resumo das especialidades oferecidas, mostrando os médicos de cada especialidade (Figura 7 e Figura 8) e alguns testemunhos a contar a experiência que tiveram com a clínica(Figura 9):



Figura 5 - Área de signup

Como é que funciona?

Tire as suas dúvidas sobre saúde com os melhores médicos e especialistas.

- 1 Preencha o formulário que depois será avaliado por um médico.
- 2 Vai ser capaz de as suas consultas e as que já foram realizadas.
- 3 Também vai ser capaz de verificação das suas prescrições e exames.



Figura 6 - Como funcionam os formulários



Dermatologia

Especialidade médico-cirúrgica dedicada ao estudo, diagnóstico e tratamento das doenças de pele, cabelo, unhas e mucosas.

[Ver médicos](#)



Psiquiatria

Especialidade médica que se dedica à prevenção, diagnóstico e tratamento de perturbações mentais, emocionais ou comportamentais.

[Ver médicos](#)

Figura 7 - Bloco introdutório às especialidades

Dermatologia

Especialidade médico-cirúrgica dedicada ao estudo, diagnóstico e tratamento das doenças de pele, cabelo, unhas e mucosas.

Equipa Médica:

José Figueiras
Manuel Lima
Maria Lourenço
Rita Castro

Figura 8 - Informação mais detalhada sobre as especialidades, com os médicos

Estava realmente preocupada com alguns sintomas que estava a enfrentar há semanas e decidi experimentar a "Saúde On-line". Fiquei impressionada com a rapidez com que recebi uma resposta de um médico experiente e agora estou no caminho certo para uma solução.

- Maria Gonçalves

Finalmente, consegui marcar uma consulta com um especialista que entendia minha situação e ofereceu um plano de tratamento adequado. Não posso expressar minha gratidão o suficiente por essa plataforma!

- Diogo Silva

Eu não tenho um médico de família e deparei-me com alguns problemas de saúde persistentes. Quando me deparei com este site, percebi que era a solução perfeita para mim. Agora, finalmente estou a receber o cuidado necessário e a minha saúde melhorou significativamente.

- Paula Barros

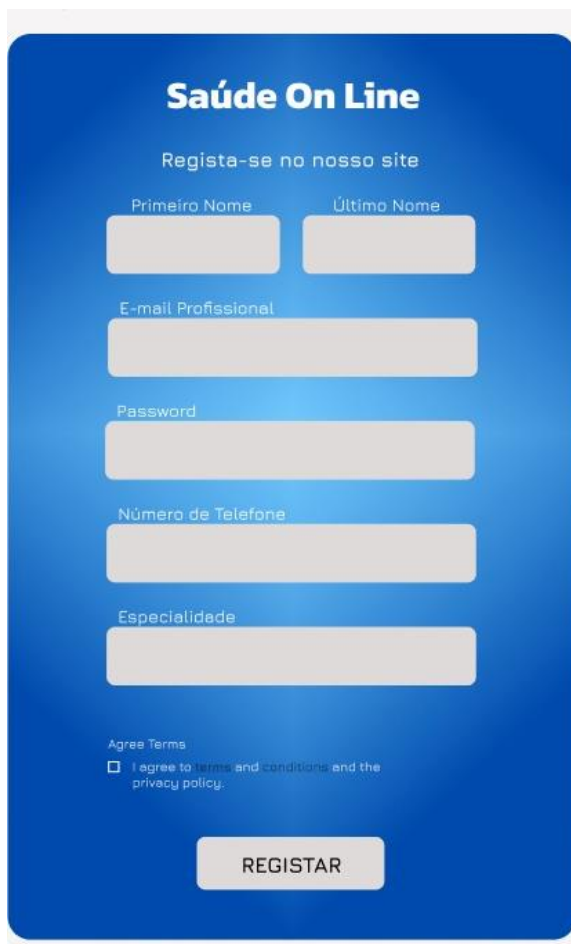
Depois de uma consulta médica frustrante, onde não senti que minhas preocupações foram levadas a sério, decidi experimentar esse site. Recebi um feedback rápido e, em pouco tempo, consegui marcar uma consulta com um médico atencioso e comprometido.

- Leandro Pereira

Figura 9 – Testemunhos

2.5.2. Signup e Login de doente e médico

Para ter acesso a todas as funcionalidades da aplicação, é necessário que os utilizadores iniciem sessão ou criem conta. O formulário de sign up do médico (Figura 10) vai ser diferente do signup do doente (Figura 11). O login (Figura 12) do médico e do doente pede o email e a password definida por eles.



The image shows a digital form titled "Saúde On Line" with the subtitle "Regista-se no nosso site". The form is set against a blue gradient background. It contains several input fields: "Primeiro Nome" and "Último Nome" (two separate boxes), "E-mail Profissional", "Password", "Número de Telefone", and "Especialidade". Below these fields is a section for "Agree Terms" with a checkbox and the text "I agree to terms and conditions and the privacy policy." At the bottom of the form is a large, light-colored button labeled "REGISTAR".

Figura 10 - Signup do médico



Saúde On Line

Regista-se no nosso site

Primeiro Nome

Último Nome

Data de Nascimento

E-mail

Password

Número de Telefone

NIF

Concelho

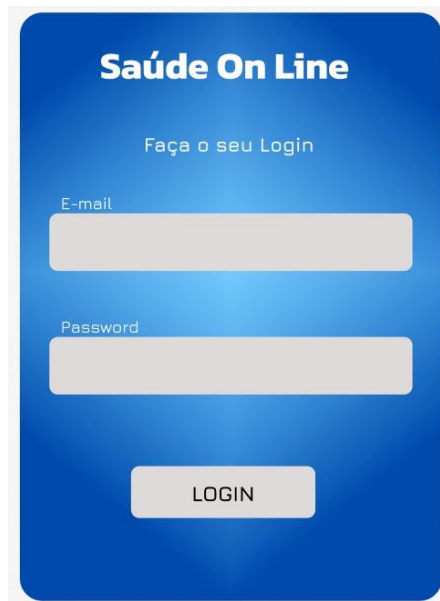
Distrito

Agree Terms
☐ I agree to terms and conditions and the privacy policy.

or login

REGISTAR

Figura 11 - Signup doente



Saúde On Line

Faça o seu Login

E-mail

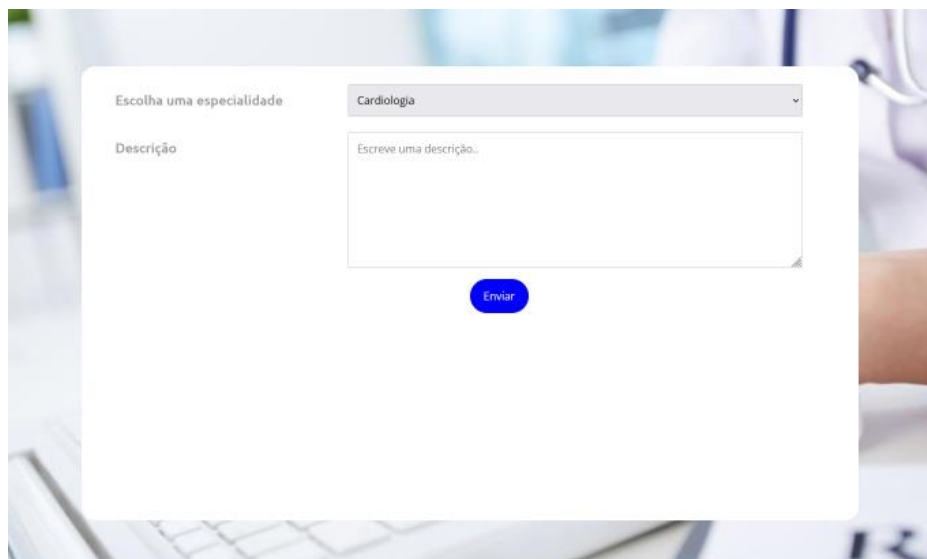
Password

LOGIN

Figura 12 – Login

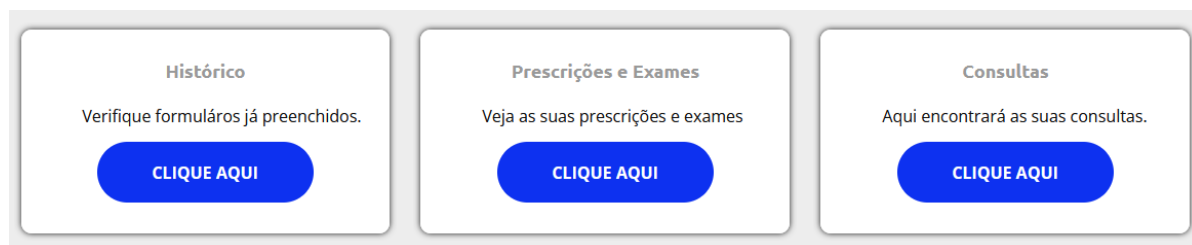
2.5.2. Página principal do utente

Após o login, o doente dá de caras com a página principal dele, onde o formulário é a figura em destaque (Figura 13). Mais em baixo, o doente poderá então consultar formulários antigos que tenha preenchido, exames e prescrições que tenha e consultas futuras (Figura 14).



O formulário apresenta um campo de seleção para 'Escolha uma especialidade' com 'Cardiologia' selecionado. Abaixo, há um campo de texto rotulado 'Descrição' com o placeholder 'Escreve uma descrição...'. Um botão azul 'Enviar' está posicionado na base do formulário.

Figura 13 - Formulário do doente



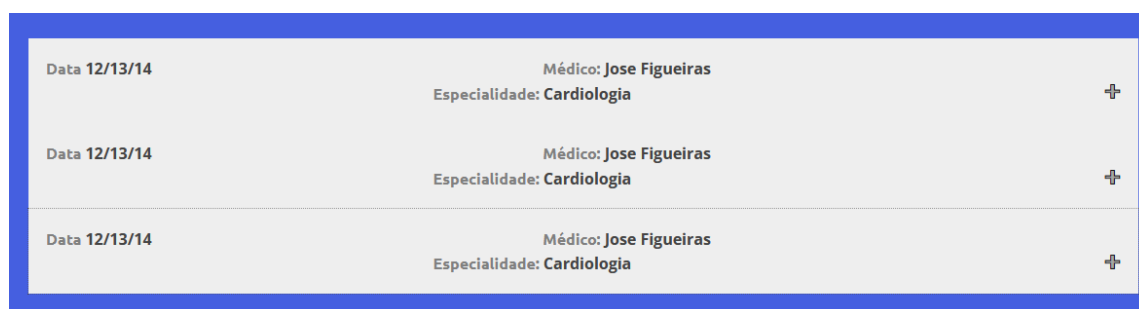
Esta seção contém três cartões de acesso:

- Histórico**: Verifique formulários já preenchidos. Botão: CLIQUE AQUI
- Prescrições e Exames**: Veja as suas prescrições e exames. Botão: CLIQUE AQUI
- Consultas**: Aqui encontrará as suas consultas. Botão: CLIQUE AQUI

Figura 14 - Consultas possíveis para o doente

- **Histórico:**

Como referenciado, o histórico vai apresentar os formulários já preenchidos pelo utente (Figura 15), com as informações de cada (Figura 16):



Data 12/13/14	Médico: Jose Figueiras Especialidade: Cardiologia	✚
Data 12/13/14	Médico: Jose Figueiras Especialidade: Cardiologia	✚
Data 12/13/14	Médico: Jose Figueiras Especialidade: Cardiologia	✚

Figura 15 – Ver as consultas do doente

Data: 12/13/14	Médico: Jose Figueiras
Especialidade: Cardiologia	

Médico: José Figueiras

Especialidade: Cardiologia

Descrição: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos, e vem sendo utilizado desde o século XVI, quando um impressor desconhecido pegou uma bandeja de tipos e os embaralhou para fazer um livro de modelos de tipos. Lorem Ipsum sobreviveu não só a cinco séculos, como também ao salto para a editoração eletrônica, permanecendo essencialmente inalterado. Se popularizou na década de 60, quando a Letraset lançou decalques contendo passagens de Lorem Ipsum, e mais recentemente quando passou a ser integrado a softwares de editoração eletrônica como Aldus PageMaker.

Observações: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos, e vem sendo utilizado desde o século XVI, quando um impressor desconhecido pegou uma bandeja de tipos e os embaralhou para fazer um livro de modelos de tipos. Lorem Ipsum sobreviveu não só a cinco séculos, como também ao salto para a editoração eletrônica, permanecendo essencialmente inalterado. Se popularizou na década de 60, quando a Letraset lançou decalques contendo passagens de Lorem Ipsum, e mais recentemente quando passou a ser integrado a softwares de editoração eletrônica como Aldus PageMaker.

Figura 16 - Informações abertas do historico

- Prescrições e exames:

Para mostrar as informações dos exames e prescrições, aparecerá na mesma como no histórico (Figura 15), mas ao abrir aparece as informações dos exames e prescrições (Figura 17):

Data: 12/13/14	Consulta: Jose Figueiras
Especialidade: Cardiologia	

Médico: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos, e vem sendo utilizado desde o século XVI, quando um impressor desconhecido pegou uma bandeja de tipos e os embaralhou para fazer um livro de modelos de tipos. Lorem Ipsum sobreviveu não só a cinco séculos, como também ao salto para a editoração eletrônica, permanecendo essencialmente inalterado. Se popularizou na década de 60, quando a Letraset lançou decalques contendo passagens de Lorem Ipsum, e mais recentemente quando passou a ser integrado a softwares de editoração eletrônica como Aldus PageMaker.

Prescrição: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos, e vem sendo utilizado desde o século XVI, quando um impressor desconhecido pegou uma bandeja de tipos e os embaralhou para fazer um livro de modelos de tipos. Lorem Ipsum sobreviveu não só a cinco séculos, como também ao salto para a editoração eletrônica, permanecendo essencialmente inalterado. Se popularizou na década de 60, quando a Letraset lançou decalques contendo passagens de Lorem Ipsum, e mais recentemente quando passou a ser integrado a softwares de editoração eletrônica como Aldus PageMaker.

Exame: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos, e vem sendo utilizado desde o século XVI, quando um impressor desconhecido pegou uma bandeja de tipos e os embaralhou para fazer um livro de modelos de tipos. Lorem Ipsum sobreviveu não só a cinco séculos, como também ao salto para a editoração eletrônica, permanecendo essencialmente inalterado. Se popularizou na década de 60, quando a Letraset lançou decalques contendo passagens de Lorem Ipsum, e mais recentemente quando passou a ser integrado a softwares de editoração eletrônica como Aldus PageMaker.

Figura 17 - Informações abertas dos exames e prescrições

- Consultas:

Para ver as futuras consultas, o mostrado será parecido com os anteriores, no entanto, não terá informações adicionais (Figura 18)

Data: 12/13/14	Médico: Jose Figueiras Especialidade: Cardiologia
Data 12/13/14	Médico: Jose Figueiras Especialidade: Cardiologia
Data: 12/13/14	Médico: Jose Figueiras Especialidade: Cardiologia

Figura 18 - Consultas futuras do doente

2.5.3. Página principal do medico

Na página do médico, aparece a caixa de entrada, com formulários por responder da sua especialidade (Figura 19) que, abrindo, terá a exposição da descrição do doente e onde poderá escrever as suas observações e marcar uma consulta caso tenha considerado necessário, caso contrário poderá ignorar(Figura 20). Também terá o histórico de formulários já respondidos, mostrando e as consultas dele(Figura 21). Na parte dos formulários, aparecerá as observações do médico e o conteúdo do formulário (Figura 22) e, na parte das consultas, apenas aparecerá a data e o doente respetivo (Figura 23).

Data 12/13/14	Utente: Tiago Pereira	+
Data 12/13/14	Utente: Tiago Pereira	+
Data 12/13/14	Utente: Tiago Pereira	+

Figura 19 - Formulários por responder

Data 12/13/14	Utente: Tiago Pereira	—
---------------	-----------------------	---

Utente: Tiago Pereira

Descrição: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos, e vem sendo utilizado desde o século XVI, quando um impressor desconhecido pegou uma bandeja de tipos e os embaralhou para fazer um livro de modelos de tipos. Lorem Ipsum sobreviveu não só a cinco séculos, como também ao salto para a editoração eletrônica, permanecendo essencialmente inalterado. Se popularizou na década de 60, quando a Letraset lançou decalques contendo passagens de Lorem Ipsum, e mais recentemente quando passou a ser integrado a softwares de editoração eletrônica como Aldus PageMaker.

Observações:

Data: Hora:

[Marcar Consulta](#) [Ignorar](#)

Figura 20 - Formulário não respondido aberto

<p>Histórico</p> <p>Verifique formuláros já respondidos.</p> <p>CLIQUE AQUI</p>	<p>Consultas</p> <p>Veja as suas consultas.</p> <p>CLIQUE AQUI</p>
--	---

Figura 21 - Ver o histórico de formulários já preenchidos e consultas já dadas do médico

Data: 12/13/14	Utente: Jose Figueiras
<p>Descrição: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos, e vem sendo utilizado desde o século XVI, quando um impressor desconhecido pegou uma bandeja de tipos e os embaralhou para fazer um livro de modelos de tipos. Lorem Ipsum sobreviveu não só a cinco séculos, como também ao salto para a editoração eletrônica, permanecendo essencialmente inalterado. Se popularizou na década de 60, quando a Letraset lançou decalques contendo passagens de Lorem Ipsum, e mais recentemente quando passou a ser integrado a softwares de editoração eletrônica como Aldus PageMaker.</p> <p>Observações: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos, e vem sendo utilizado desde o século XVI, quando um impressor desconhecido pegou uma bandeja de tipos e os embaralhou para fazer um livro de modelos de tipos. Lorem Ipsum sobreviveu não só a cinco séculos, como também ao salto para a editoração eletrônica, permanecendo essencialmente inalterado. Se popularizou na década de 60, quando a Letraset lançou decalques contendo passagens de Lorem Ipsum, e mais recentemente quando passou a ser integrado a softwares de editoração eletrônica como Aldus PageMaker.</p>	

Figura 22 - Exemplo do histórico de formulários do médico

Data: 12/13/14	Utente: Jose Figueiras
Data: 12/13/14	Utente: Jose Figueiras
Data: 12/13/14	Utente: Jose Figueiras

Figura 23 - Consultas futuras do médico

3. Desenvolvimento da base de dados

O primeiro passo na construção da aplicação foi a implementação da base de dados, seguindo o modelo de entidade relação. De modo a haver mais eficiência, foram adicionadas umas tabelas adicionais, que são as seguintes e os seus atributos:

Distrito: id_distrito (*serial*), nom_dist (*varchar(60)*)

Concelho: id_concelho(*serial*), nom_conc(*varchar(60)*)

3.1 Tabelas e relações

De modo a ligar as tabelas entre si, foram utilizadas as chaves estrangeiras, primeiramente conforme o teoricamente correto e depois alterado caso necessário:

Para as relações 1 para 1 sem obrigatoriedade, tem de ser criada uma terceira tabela, a tabela relação, de modo a evitar elementos nulos.

Como, por exemplo, a relação entre as tabelas formulário e consulta tem a tabela relação formulário_consulta.

Para as relações 1 para muitos, estará a chave estrangeira da tabela que tem apenas uma unidade na tabela que terá múltiplas.

A relação entre as tabelas doente e formulário é uma relação onde um doente pode preencher vários formulários, mas um formulário apenas é preenchido por um doente. Como tal, a chave primária do doente aparece na tabela de formulário como chave estrangeira.

Já na relação médico consulta, que contém a mesma relação, foi pensado de forma diferente, tendo sido criado uma tabela relação, visto que em um dos primeiros protótipos, a relação teria sido de muitos para muitos. No entanto, o diagrama foi mudado após a implementação das tabelas, tendo se mantido com a tabela relação medico_consulta.

Para a relação 1 para 1 com obrigatoriedade, o teoricamente correto seria a junção das duas entidades em uma única tabela. No entanto, para permitir uma maior segurança, a chave primária da tabela com mais informação passou para a tabela com acesso à informação mais confidencial.

Por exemplo, as entidades login medico e médico. A chave primária de médico encontra-se na tabela do login medico como chave estrangeira. Assim, ao fazer as *queries* à base de dados, mesmo que se pedisse para mostrar todos os dados do médico, não terá acesso à sua palavra-passe.

3.2 CRUD

O próximo passo tomado foi a criação do CRUD para as tabelas, com recurso a procedimentos que depois serão chamados.

3.2.1. Create

O create adiciona um novo elemento à tabela. Nas tabelas mais simples, o procedimento é mais simples, como o create de um médico novo, da seguinte forma:

```
create or replace procedure inserir_medico(nom varchar(60),especi int, email
varchar(60), out id_med int)
as $$
declare conta int;
begin
  if(nom is null)
  then
    raise notice 'nome nulo';
    return;
  end if;

  insert into medico(nom_med, especi_med, email) values (nom, especi, email)
returning id_medico into id_med;
  raise notice 'bem sucedido';

end; $$ Language PLPGSQL
```

Já o signup do médico, ou seja, na criação de uma conta nova, tem um passo a mais, chamando aquele procedimento de inserção de médico:

```
create or replace procedure signup_medico(emai varchar(60), passwor
varchar(60), nom varchar(60), especialidade int)
as $$
declare sign int;
declare id_med int;
begin
  --verifica se medico já tem signup
  select count(*) into sign
  from login_med lm
  where lm.email = emai;
  if (sign > 0)
  then
    raise notice 'email já utilizado';
    return;
  end if;
  --inserir os dados
  call inserir_medico(nom ,especialidade, emai, id_med);
  insert into login_med (id_medico, email, pass) values (id_med, emai,
passwor);
end; $$ Language PLPGSQL
```

Para criar a consulta pelo formulário, é um procedimento mais complexo. Começando por verificar se o formulário já originou uma consulta. Após, se o médico
 Ana Carvalho
 Ana Pinto
 Diana Dinis

já terá alguma consulta marcada para o mesmo horário e acaba por inserir todos os dados necessários e modificando a tabela do formulário para guardar o médico que avaliou o formulário:

```
create or replace procedure criar_consulta_form(id_med int, horari timestamp,
id_form int)
as $$
declare
    form_cons int;
    horar int;
    id_cons int;
begin
    --verificar se existe consulta vindo do formulario
    if (id_form is not null)
    then
        select count(*) into form_cons
        from formulario_consulta fc
        where fc.id_formulario = id_form;
        if (form_cons > 0)
        then
            raise notice 'Formulario com consulta já marcada';
            return;
        end if;
    end if;
    --verificar se existe consulta com o medico no mesmo horario
    call count_cons_med(id_med, horari, horar);
    if (horar > 0)
    then
        raise notice 'medico não disponivel no horario';
        return;
    end if;
    --inserir
    insert into consulta (horario) values (horari) returning id_consulta into
id_cons;
    --inserir nas tabelas relacao
    insert into formulario_consulta (id_formulario, id_consulta) values
(id_form, id_cons);
    insert into medico_consulta (id_medico, id_consulta) values (id_med,
id_cons);
    --guardar o medico que viu o formulario
    update formulario set id_medico = id_med where id_formulario = id_form ;
end; $$ Language PLPGSQL
```

O procedimento `count_cons_med` serve para verificar mais facilmente se um médico está disponível ou não num determinado horário, da seguinte forma:


```
create or replace procedure count_cons_med(med int, horari timestamp, out
conta int)
as $$
begin
    select count(*) into conta
    from consulta c inner join medico_consulta mc using (id_consulta)
    where c.horario = horari and mc.id_medico = med;
end; $$ Language PLPGSQL
```

3.2.2. Read

O read é o mais simples, sendo apenas os selects para mostrar toda a informação apenas de uma tabela, que serão implementados na API, sendo assim:

```
select *
from consulta
```

3.2.3. Update

O update atualiza informações já existentes. O procedimento de atualizar as informações do doente vai verificar se altera o email, de modo a poder alterar o login, modificando de seguida na tabela doente as informações. É pressuposto que as informações não alteradas serão enviadas na mesma para o procedimento, definindo os valores mesmo que não tenham sido alterados.

```
create or replace procedure atualiza_doente (id_doe int, nom varchar(60),
nmrnif int, mail varchar(60), nmr int, dist int, conc int, dat_nas date)
as $$
declare count_e int;
begin
    --verifica se alterou o email
    select count(*) into count_e
    from login_doe
    where email = mail;
    --se alterou o email
    if(count_e = 0)
    then
        --remover o constraint de chave estrangeira do email
        alter table login_doe drop constraint emaildoe_fk;
        --atualiza o login
        update login_doe
        set email = mail
        where id_doente = id_doe;
    end if;
    --atualizar a tabela
    update doente
    set nom_doen = nom,
    data_nasc = dat_nas,
```

```

    nif = nmrnif,
    email = mail,
    nmr_tele = nmr,
    distrito = dist,
    concelho = conc
where id_doente = id_doe;

--se alterou o email, para voltar o constraint
if(count_e = 0)
then
    --retornar o constraint
    alter table login_doe add constraint emaildoe_fk foreign key (email)
references doente(email);
end if;
end; $$ Language PLPGSQL;

```

Já o de atualizar a consulta verifica se o médico está disponível e depois se ele muda, mudando também a ligação na tabela relação medico_consulta. Por fim, atualiza a tabela:

```

create or replace procedure atualiza_consulta (id_cons int, horari timestamp,
id_med int, obser varchar[500])
as $$
declare horar int;
        medi int;
begin
    --verificar se medico está disponível
    call count_cons_med(id_med, horari, horar);
    if (horar > 0)
    then
        raise notice 'medico não disponível no horario';
    end if;
    --verificar se muda o medico
    select count(*) into medi
    from medico_consulta
    where id_consulta = id_cons and id_medico = id_med;
    if (medi > 0)
    then
        update medico_consulta
        set id_medico = id_med
        where id_consulta = id_cons;
    end if;
    --atualiza a tabela
    update consulta
    set horario = horari and observacoes = obser
    where id_consulta = id_cons;
end; $$ Language PLPGSQL;

```

Um dos mais simples será o update de exames, da seguinte forma:

```
create or replace procedure atualiza_exame (id_exa int, nom varchar(60),
id_cons int)
as $$
declare exa int;
begin
    --verificar se exame já foi receitado na mesma consulta
    select count(*) into exa
    from exame e
    where e.id_consulta = id_cons and e.nome = nom;
    --verificar se já foi prescrito
    if (exa > 0)
    then
        raise notice 'o exame já foi prescrito';
        return;
    end if;
    --atualizar
    update exame
    set nome = nom
    where id_exame = id_exa;
end; $$ Language PLPGSQL;
```

3.2.4. Delete

Por fim, o delete para eliminar os dados das tabelas. Um dos mais simples será o de eliminar concelho, onde vai eliminar diretamente. Caso não seja encontrado, envia um aviso a dizer que não foi encontrado. Se não, avisa que foi bem sucedido:

```
create or replace procedure eliminar_concelho (_id_concelho int)
as $$
begin
    delete from concelho
    where id_concelho = _id_concelho;
    --caso nao exista
    if not found
    then
        raise notice 'Concelho nao existente';
    else
        raise notice 'Eliminacao bem sucedida';
    end if;
end; $$ Language PLPGSQL
```

Já um dos mais complexos será a eliminação de uma consulta proveniente de um formulário. Primeiro, será chamado o procedimento de eliminar a relação do médico à consulta:

```
create or replace procedure eliminar_medico_consulta(_id_consulta int)
as $$
begin
    delete from medico_consulta
    where id_consulta = _id_consulta;
end; $$ Language PLPGSQL
```

Após, elimina da tabela relação formulário consulta, chamando, por fim, o procedimento para eliminar uma consulta:

```
create or replace procedure eliminar_consulta(_id_consulta int)
as $$
begin
    delete from consulta
    where id_consulta = _id_consulta;
end; $$ Language PLPGSQL
```

Caso não tenha sido encontrado no decorrer do delete de formulário consulta, avisa que a consulta não existe, caso contrário diz que a eliminação foi bem sucedida. O procedimento encontra-se todo a seguir:

```
create or replace procedure eliminar_consulta_form (_id_consulta int)
as $$
begin
    --elimina relacao medico consulta
    call eliminar_medico_consulta(_id_consulta);
    --elimina de formulario consulta
    delete from formulario_consulta
    where id_consulta = _id_consulta;
    --elimina da consulta
    call eliminar_consulta(_id_consulta);
    --se não existir
    if not found
    then
        raise notice 'Consulta nao existente';
    else
        raise notice 'Eliminacao bem sucedida';
    end if;
end; $$ Language PLPGSQL
```

3.3. Views

Por fim, as views permitem criar uma tabela virtual que une várias outras tabelas definidas por queries. Permitem fazer queries mais facilmente à base de dados, unindo todas as informações relevantes juntas.

No desenvolvimento do trabalho, foram selecionados três views principais:

- View das consultas:

```
create or replace view consultas
as
  select c.id_consulta consulta, d.id_doente,d.nom_doen doente, c.horario,
  f.id_medico medico_formulario, c.observacoes,
         e.nome exame, p.medicamento, m.id_medico, f.id_formulario,
  m.especi_med, m.nom_med, es.nom_especi, es.id_especi, f.descricao
  from consulta c
    inner join medico_consulta mc on c.id_consulta = mc.id_consulta
    inner join medico m on m.id_medico = mc.id_medico
    inner join especialidade es on es.id_especi = m.especi_med
    left join formulario_consulta fc on fc.id_consulta = c.id_consulta
    left join consulta_consulta cc on cc.id_consulta = c.id_consulta
    inner join formulario f on fc.id_formulario = f.id_formulario
    inner join doente d on f.id_doen = d.id_doente
    left join exame e on e.id_consulta = c.id_consulta
    left join prescricao p on p.id_consulta = c.id_consulta;
```

O left join permite mostrar as instâncias mesmo que não haja elementos em comum. Caso tivesse sido escolhido um inner join, várias consultas não seriam mostradas, mesmo que existam.

- View dos formulários:

```
create or replace view ver_formulario
as
  select f.id_formulario, d.id_doente, d.nom_doen, d.data_nasc, d.email,
  f.descricao , f.id_medico medico_formulario, f.especialidade, mc.id_medico
  medico_consulta
  from formulario f
    inner join doente d on d.id_doente = f.id_doen
    left join formulario_consulta fc on f.id_formulario = fc.id_formulario
    left join medico m on m.id_medico = f.id_medico --quem viu o formulario
    left join consulta c on fc.id_consulta = c.id_consulta
    left join medico_consulta mc on mc.id_consulta = c.id_consulta;
```

- Especialidades e medicos

```
create or replace view medicos_especialidades
as
  select m.nom_med, m.email, e.nom_especi, e.id_especi
  from medico m
    inner join especialidade e on m.especi_med = id_especi
```

4. Desenvolvimento da página web e API

Para o desenvolvimento da página web, foi utilizado HTML, CSS e JS base como referenciado no diagrama da arquitetura técnica do sistema (Figura 3), pois foi o lecionado em aula. Utilizaram-se as bibliotecas Bootstrap e JQuery no auxílio da construção das páginas web para a aplicação.

Os ficheiros relativos à página web em si encontram-se na pasta www. Dentro desta, temos todo o html e o css, js e api separados em pastas respetivas.

Todos estes ficheiros em conjunto formam as páginas da aplicação, tendo o resultado sido bastante fiel aos mockups, da Figura 5 à Figura 23.

Foram utilizados alguns elementos que não foram abordados em aula, que foram os seguintes:

4.1. Modal

Um modal é um elemento gráfico de controlo. Desabilita a interação do utilizador com a página principal, mantendo-a visível, mas não podem voltar à página principal sem interagir com o modal, que aparece como um popup.

Na página principal, ao clicar no botão “ver médicos”, vai ser aberto um modal, como na Figura 8 do mockup. O html referente ao botão é o seguinte, no ficheiro PaginaPrincipal.html

```
<button class="myBtn u-align-center u-custom-font u-font-ubuntu u-text u-text-default u-text-grey-40 u-text-1" id="myBtn">Ver médicos</button>
```

A classe myBtn vai ser responsável por desplotar o modal, cujo html é o seguinte:

```
<!--MODAL DERMATOLOGIA-->
<div id="myModalDermatologia" class="modal">
  <!-- CONTEÚDO DO MODAL -->
  <div class="modal-content">
    <div class="modal-header">
      <h3 class="u-align-center u-custom-font u-font-ubuntu u-text u-text-default u-text-3">Dermatologia</h3>
      <span class="close">&times;</span>
    </div>
    <div class="modal-body">
      <p>Especialidade médico-cirúrgica dedicada ao estudo, diagnóstico e tratamento das doenças de pele, cabelo, unhas e mucosas.</p>
      <hr>
    </div>
  </div>
</div>
```

```
<h6 class="u-align-left u-custom-font u-font-ubuntu u-text u-text-  
palette-1-base u-text-8 u-color">Equipa Médica:</h6>  
<div data-id="equipaMedicaDermatologia"></div>  
</div>  
</div>  
</div>
```

No ficheiro modal.js está presente o js de todos os modals. A variável btn vai conter os elementos que contenham a classe myBtn, a variável modal os elementos de classe modal e o span os elementos com classe close:

```
var modal = document.getElementsByClassName('modal');  
var btn = document.getElementsByClassName("myBtn");  
var span = document.getElementsByClassName("close");
```

Quando o botão é clicado, a função de clique correspondente é executada, alterando o estilo de “display” do elemento modal para "block", ou seja, este fica visível:

```
//primeiro no html  
btn[0].onclick = function() {  
    modal[0].style.display = "block";  
}  
//segundo no html  
btn[1].onclick = function() {  
    modal[1].style.display = "block";  
}
```

(...)

Após, para fechar as respetivas caixas modais é alterado o estilo de “display” do elemento modal associado para "none", ou seja, este fica oculto, quando o x é clicado.

```
//fechar modal no x  
span[0].onclick = function() {  
    modal[0].style.display = "none";  
}  
  
span[1].onclick = function() {  
    modal[1].style.display = "none";  
}
```

(...)

Por fim, a seguinte função funciona de maneira que se o elemento clicado (*event.target*) for fora da caixa modal, a função é executada e altera o estilo de “display” da caixa modal para "none", ocultando-a.

```
// quando o utilizador carrega fora do modal, para fechá-lo
window.onclick = function(event) {
  if (event.target == modal) {
    modal.style.display = "none";
  }
}
```

4.3 Acordeão

Um acordeão é um elemento que permite organizar melhor e navegar por diferentes secções num único container. O histórico do médico, na Figura 19 e na Figura 22 foi organizado com um acordeão.

Este botão vai ser responsável por fechar e abrir a secção do acordeão:

```
<button class="accordion">
  <span class="u-custom-font u-font-ubuntu u-text u-text-2 u-
accordion-title">Data</span>
  <span>12/13/14</span>
  <span class="u-custom-font u-font-ubuntu u-text u-text-2 u-
accordion-title" style="margin-left: 40vh;">Utente:</span>
  <span>Jose Figueiras</span>
</button>
```

Tornando visível o conteúdo:

```
<div class="panel">
  <div class="u-div-accordion-content">
    <p class="u-accordion-content">
      <span class="u-custom-font u-font-ubuntu u-text u-text-2 u-
accordion-title">Descrição:</span>
      <span>(texto)</span>
    </p>
  </div>
  <div class="u-div-accordion-content">
    <p class="u-accordion-content">
      <span class="u-custom-font u-font-ubuntu u-text u-text-2 u-
accordion-title">Observações:</span>
      <span>(texto).</span>
    </p>
  </div>
</div>
```

A parte de js do acordeão está presente no ficheiro accordion.js. A variável acc vai buscar todos os elementos que contenham a classe accordion:

```
var acc = document.getElementsByClassName("accordion");
```


O ciclo `for` está a ouvir o `click`. Quando este acontece, verifica se está aberto ou não, vendo a altura máxima (`panel.style.maxHeight`). Caso tenha algum valor, significa que está aberto e, portanto, fecha aquela secção. Caso não contenha, este é adicionado de modo a abrir a secção

```
for (i = 0; i < acc.length; i++) {  
  acc[i].addEventListener("click", function() {  
    this.classList.toggle("active");  
    var panel = this.nextElementSibling;  
    if (panel.style.maxHeight) {  
      panel.style.maxHeight = null;  
    } else {  
      panel.style.maxHeight = panel.scrollHeight + "px";  
    }  
  });  
}
```

4.3 API

O `api` permitiu ligar a base de dados à aplicação web, permitindo utilizar os dados guardados na base de dados e utilizar a aplicação na sua totalidade.

Os ficheiros referentes à `api` são, dentro do `www`, o `api.js`, as `routes` e os serviços estão dentro das suas próprias pastas, permitindo haver a conexão direta à base de dados, atualizando e mostrando os dados de forma dinâmica.

Após a implementação da `API`, a parte dos acordeões deixou de funcionar, tendo sido então escolhido a abordagem de apenas apresentar os dados todos completos.

5. Conclusão

A construção desde o início de uma aplicação web não é fácil. Planear, desenhar e desenvolver são tarefas enormes. O lecionado em aula não foi de todo suficiente para conseguir desenvolver a aplicação, tendo de recorrer muito de páginas web, pesquisa e tutoriais e também a pessoas já da área conhecidas. De certa forma, este trabalho permitiu desenvolver além de competências de pesquisa, também competências em saber como e quando pedir ajuda.

Todo este trabalho foi muito desafiante, especialmente a parte da API e a integração desta na página web, visto que nunca tínhamos tido contacto com nada parecido.

6. Webgrafia

- Desenhos dos diagramas e mockups:

<https://www.figma.com>

<https://app.diagrams.net>

<https://www.visual-paradigm.com>

- Pesquisa

<https://blog.logrocket.com/crud-rest-api-node-js-express-postgresql/>

<https://stackoverflow.com/questions/71152025/making-api-request-with-javascript>

<https://developer.mozilla.org/pt-BR/docs/Mozilla/Add-ons/WebExtensions>

<https://www.w3schools.com/js/default.asp>

<https://swagger.io/solutions/api-testing/>

<https://css-tricks.com/video-screencasts/208-a-css-grid-layout-with-pictures-down-one-side-matched-up-with-paragraphs-on-the-other/>

https://eloquentjavascript.net/20_node.html

<https://m3o.com/ai/api>

<https://getbootstrap.com/docs/4.0/components/modal/>

<https://mdbootstrap.com/docs/standard/components/modal/>

<https://stackoverflow.com/questions/13183630/how-to-open-a-bootstrap-modal-window-using-jquery>

https://www.w3schools.com/howto/howto_js_accordion.asp

<https://codepen.io/raubaca/pen/PZzpVe>

<https://code-boxx.com/simple-responsive-accordion-pure-html-css/>

<https://nodejs.org/api/>

<https://www.simplilearn.com/tutorials/nodejs-tutorial/nodejs-mysql>