

# YOGA GO

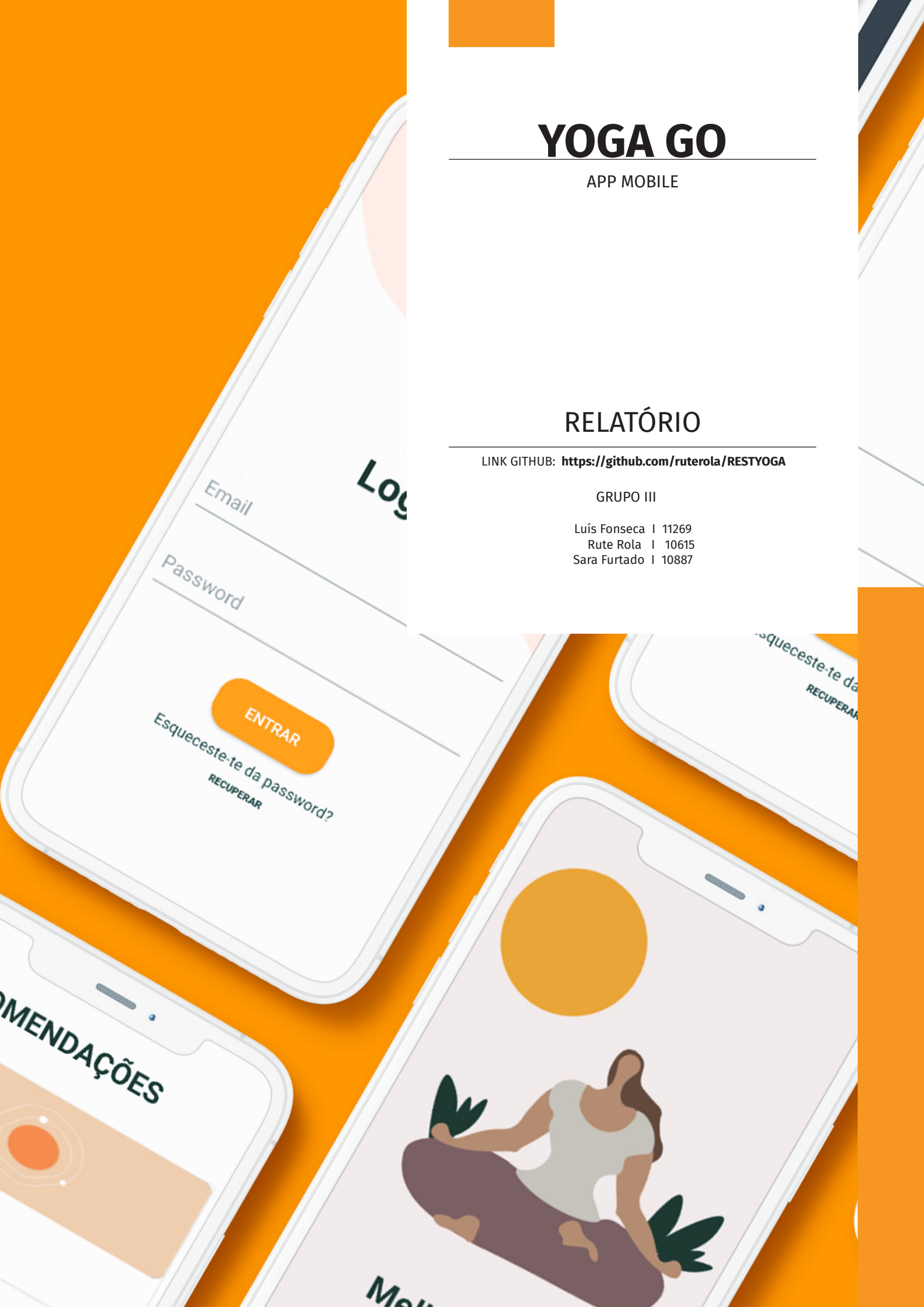
APP MOBILE

## RELATÓRIO

LINK GITHUB: <https://github.com/ruterola/RESTYOGA>

GRUPO III

Luís Fonseca | 11269  
Rute Rola | 10615  
Sara Furtado | 10887



# Índice

**03** Introdução

---

**04** Definição de ideia

---

**05** Planos existentes na aplicação

---

**07** Definição do Design

---

**08** Mapa de Navegação

---

**10** Modelação da Base de Dados

---

**12** Opções tomadas ao longo da implementação

---

**13** Excertos de código e a sua explicações

---

**15** Dificuldades no processo

---

**16** Conclusão

---

**17** Bibliografia

## Introdução

O presente projeto insere-se no segundo ano da licenciatura de Multimédia, no Instituto Superior Miguel Torga em Coimbra, na Unidade Curricular, Programação III, tendo como tema, a gestão de uma aplicação de planos de yoga.

A escolha do tema da aplicação possibilita a quem procura um estilo de vida saudável e / ou pretende praticar alguma atividade física.

No que respeita ao conteúdo apresentado, o seguinte projeto divide-se em vários tópicos, nomeadamente: todas as definições do conceito YogaGo e de toda a sua essência, as fases fundamentais para a execução do projeto, todas as suas vantagens ao praticar esta “modalidade” e por fim , uma conclusão sucinta e reflexão pessoal sobre o trabalho produzido.



**Yoga é como a música, o ritmo do corpo, a melodia da mente e a harmonia da alma, criando a sinfonia de vida.**

B. K. S . Lyengar

# Definição de ideia

*A escolha do tema foi a gestão de planos de yoga, pois para além de possibilitar uma vida tranquila e feliz, ter benefícios físicos emocionais e mentais, também possibilita o utilizador de fazer esta prática no conforto da sua casa ou então em qualquer outro local que o faça sentir livre.*

O desenvolvimento da aplicação baseia-se em outras aplicações já existentes, mas com objetividade nos planos e na sua execução. A aplicação YogaGo é vocacionada sobretudo para planos bem combinados com diversas posturas dentro dos mesmos, que são estipulados pelo utilizador de acordo com os seus objetivos.

## Funcionalidades

A aplicação tem uma apresentação simples e intuitiva ao público em geral, contudo só é possível visualizar a maior parte da informação se o utilizador estiver registado e com a sua sessão iniciada. O conteúdo em aberto contém umas pequenas descrições sobre o que yoga pode trazer à sua vida, o registo, o login e ainda a formulário caso seja necessário a recuperação da password.

Após estar registado e com a sessão iniciada (login feito), a aplicação mostra diversos planos de yoga constituídos por várias posturas. Além disso, também existe a possibilidade de aprender ainda mais sobre os diferentes tipos de yoga.

O foco da aplicação são os planos, é permitido ao utilizador regista-los com a data de inicio e a data final. Esta função ajuda ao utilizador a gestão da sua atividade física e a ter noção do tempo que demora a efetua-los.

A aplicação é facilmente comparada a algumas aplicações que existentes no mercado atual, mas com as suas peculiaridades.





## PLANOS EXISTENTES NA APLICAÇÃO

### Saudação ao sol

A saudação ao sol clássica é composta por doze posturas que se conectam com as quatro fases da respiração: inspirar, reter ou suspender, expirar e manter vazios os pulmões.

Esta pode ser feita ao amanhecer como também ao entardecer.

### Noite Calma

Este plano ajuda quem tem algumas dificuldades em adormecer, consequentemente em organizar os seus pensamentos em momentos decisivos. É um plano com posturas relaxantes e que eliminam o stress.

### Manhã feliz

Este plano é indicado para as pessoas que querem começar o dia com a mente tranquila. O plano “noite calma” vem completar o plano “manhã feliz” pois é necessário uma noite de sonhos esplêndida para um acordar alegre.

### Queimar Gordura

Queimar gordura é um plano para quem quer queimar a gordura indesejada e melhorar a resistência e o funcionamento do metabolismo. Ajuda o utilizador a ter o abdómen tonificado.

### Iniciante

O plano para iniciantes prova que não é preciso ser experiente para praticar yoga e melhorar o corpo e a mente. É constituído por cinco posturas com um nível de dificuldade baixo e que prometem fortalecer os músculos, melhorar o equilíbrio e relaxar a mente.

### Intermédio

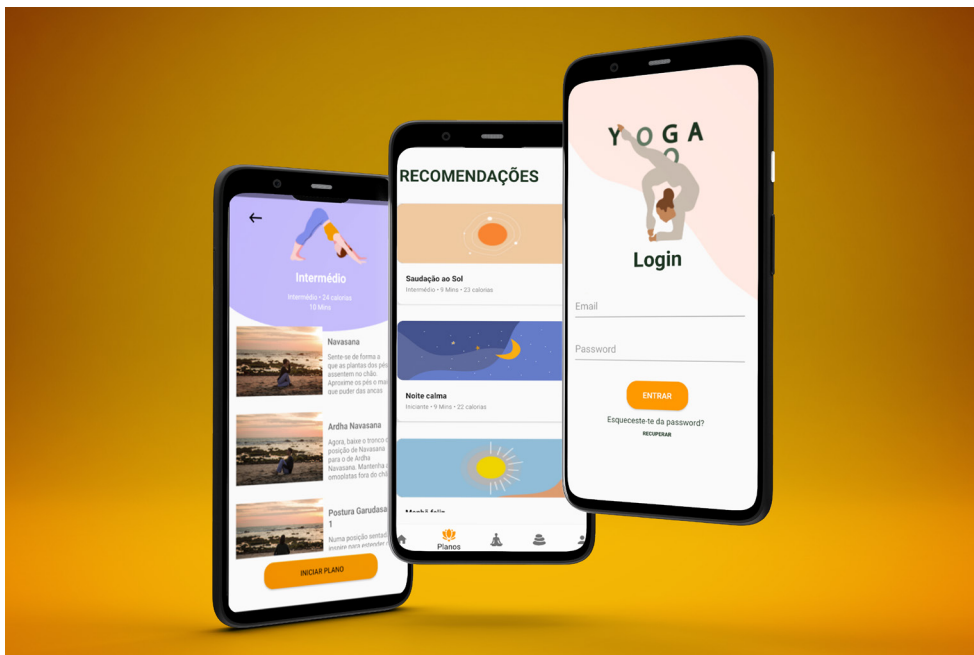
O plano intermédio já promete algum grau de dificuldade perante o plano para iniciantes. Este já se adequa a pessoas que tenham algum conhecimento e/ou praticaram esta modalidade. É um excelente plano para quem queira uma preparação mais pormenorizada.

### Avançado

Este plano com grau avançado contém posturas com movimentos mais rápidos e desafiadores para mantê-lo em movimento. Esses movimentos são essenciais e podem aumentar a frequência cardíaca a um nível equivalente a caminhar durante algumas partes da rotina.

# Definição do Design

A ferramenta utilizada para o design do layout, foi o Android Studio. A maior parte do conteúdo extra ao Android Studio foi produzido por nós, desde ilustrações a fotografias.



O design da aplicação tem por base uma paleta de cores quentes e relaxantes e também cores mais frias para que todas elas tenham um bom contraste na legibilidade entre o texto e o fundo.

## Paleta de cores



#f8971d

R: 248

G: 151

B: 29

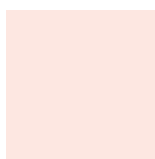


#916052

R: 145

G: 96

B: 82



#fee7e0

R: 254

G: 231

B: 224



#c6baae

R: 198

G: 186

B: 174



#dbdfc6

R: 219

G: 223

B: 198



#233a26

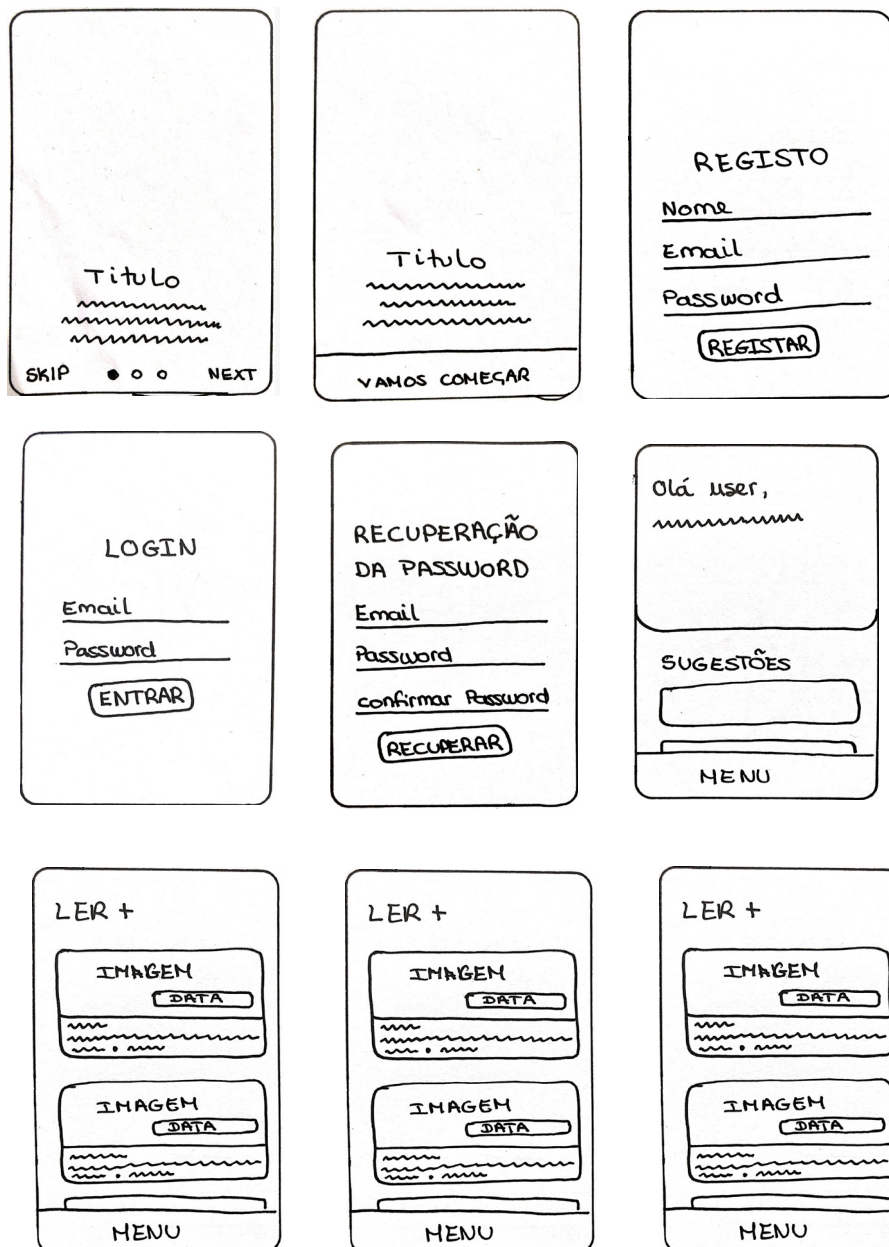
R: 35

G: 58

B: 38

# Esboços do layout da aplicação

Inicialmente, começou-se por esboçar como estaria organizada toda a informação, que se pretendia colocar na aplicação. As inspirações, ajudaram bastante nesta tarefa que requer sempre algum tempo, mas que acaba por ser um dos elementos mais importantes. Um dos objetivos, foi a utilização da informação necessária, não exagerar em textos porque estes, acabam sempre por “aborrecer” o utilizador. Estas são algumas das fases do processo criativo:



# Mapa de Navegação

*Toda esta aplicação foi pensada em fornecer ao público o conteúdo mais importante para a prática de yoga. Por isso mesmo, esta é de fácil uso para que o utilizador possa navegar sem qualquer dificuldade.*

*A aplicação Yoga Go dispõem de diferente conteúdo antes e após efetuar o login. Antes do utilizador abrir a sessão, é lhe permitido visualizar alguma informação que o motive a se registar. Além disso pode aceder ao formulário de registo, recuperação da password e por fim o tão esperado login.*

*Após ser feito o login, é possível aceder a um menu de navegação, é este que faz a ligação entre todas as páginas, que dispõem de 5 opções: a página home, planos, cursos, ler+ e por último, o perfil.*

## Páginas pós login

**1**

### Home

Esta é a primeira página a que o utilizador acesso após efetuar o login. Aqui, é possível a visualização de uma mensagem a dar as boas vinda, como algumas sugestões de planos que poderá fazer ao longo do dia.

**2**

### Planos

É nesta página que é apresentada algumas recomendações de planos diferentes, ou seja, planos que o utilizador poderá fazer consoante os seus objetivos, ou mesmo para a fase do dia em que se encontra. Ao clicar sobre a imagem deste plano, é permitido visualizar a lista de posturas que o plano contém. Além disso, dá a oportunidade de iniciar o mesmo através do botão “Iniciar plano”. Após começar o plano, é disponibilizado um botão “Terminar plano” para que o utilizador o possa concluir.

**3**

### Cursos

O separador cursos é bastante idêntico à página descrita anteriormente. Aqui, todos os planos são definidos por um grau de dificuldade, desde o iniciante ao avançado, para que todas as pessoas possam praticar yoga.

**4**

### Ler+

A página Ler+ dispõe dos vários tipos de yoga existentes. Esta página é composta por cards clicáveis que exibem uma imagem que caracterize o mesmo, o seu título, e uma pré visualização da sua descrição. Além disso, estes cards funcionam como publicações, ou seja, mostram a data e há quanto tempo o artigo foi publicado. Este conceito de publicação não está em devido funcionamento de momento, no entanto foi pensado assim, para um possível melhoramento. Quando clicado, é visível uma pequena informação sobre o tipo de yoga selecionado.

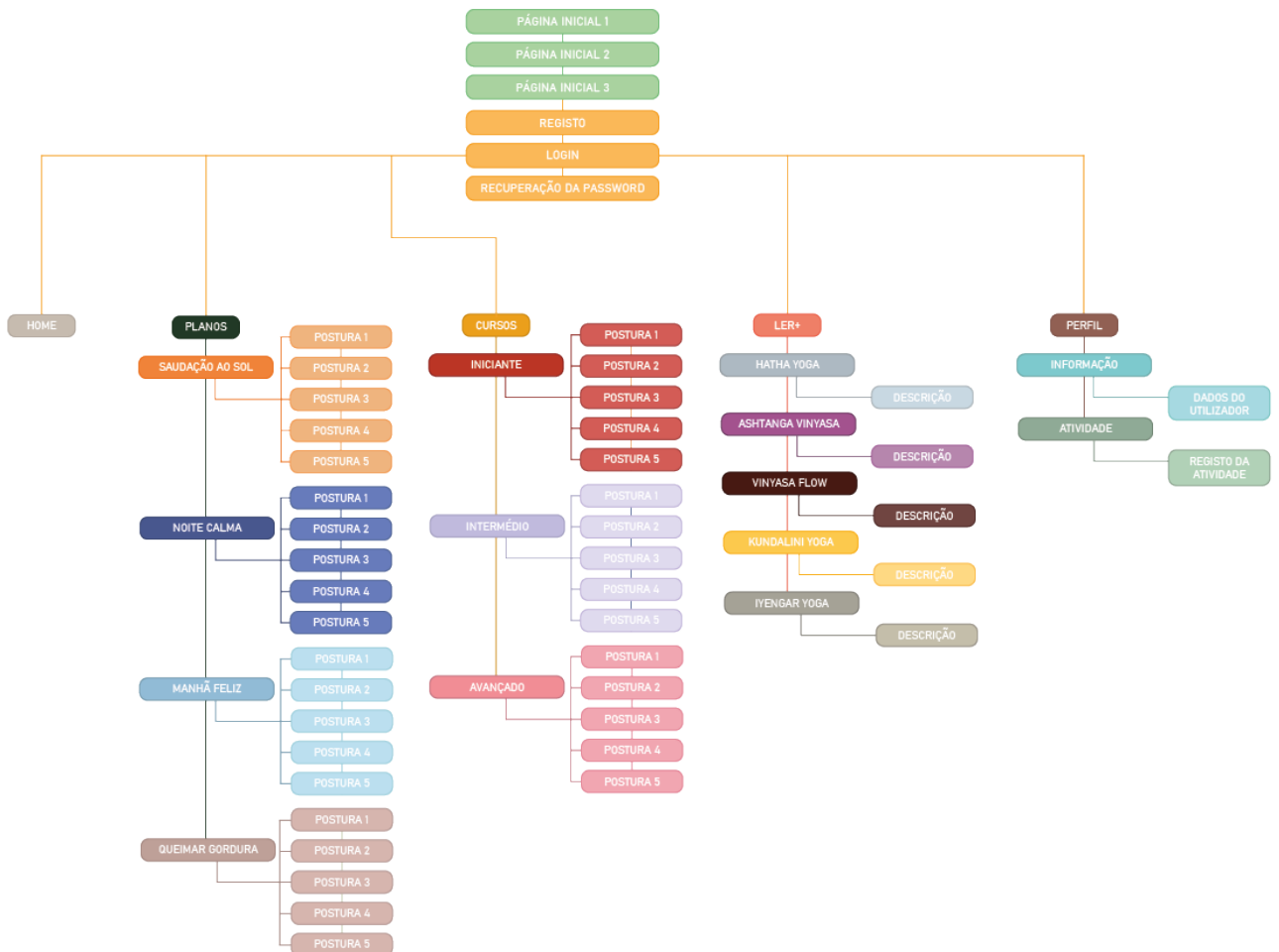
**5**

### Perfil

No separador “perfil” o utilizador tem acesso a algumas informações, derivadas à sua atividade na aplicação. É constituído por duas “abas”, uma delas com o nome “Informação” e outra “Atividade”. O separador “Informação”, contém referências ao gasto de calorias, à quantidade água ingerida até ao momento de atualização, ao peso atual do utilizador e à quantidade de passos que deu, caso o utilizador realize uma caminhada.

Neste momento, este separador é apenas





uma imagem ilustrada, que mais tarde prevemos implementar para tornar a aplicação ainda mais interessante e dinâmica para o utilizador. Já o separador “Atividade”, regista todos os planos que o utilizador realize, isto é, a aplicação dá oportunidade de visualizar o plano que iniciou e a data e hora a que este foi iniciado e terminado.

## Páginas pré login

### Informações iniciais

Na aplicação existem 3 páginas que motivam o utilizador a fazer o próprio registo para se “fidelizar” na nossa app. São páginas meras informativas.

### Registo

Como o nome indica, este separador possibilita o registo do utilizador. Existe um formulário em que

pede algumas informações ( nome, email e por fim a password), contém também um botão na qual regista automaticamente essa informação na a base de dados e é redirecionado para o login.

### Login

Após o registo do utilizador, este terá de preencher um formulário ( email e password) para ter acesso a toda a informação oferecida pela nossa app. Neste separador também é possível ver uma hiperligação com o nome “ Recuperação da password”.

### Recuperação da password

Esta página possibilita a recuperação da password caso o utilizador se tenha esquecido da mesma. Apresenta um formulário que contém ( email, password e a confirmação da password).

# MODELAÇÃO DA BASE DE DADOS

No que diz respeito à modelação da base de dados, as dificuldades foram quase nulas devido a esta já ter sido abordada na Unidade Curricular de Programação II, então só foi necessário fazer pequenas alterações.

O maior desafio que nos foi colocado e superado, foi o de imprimir o conteúdo da base de dados no Android Studio.

Para a criação da base de dados foi dado o nome de “yoga” e contém as seguintes tabelas:

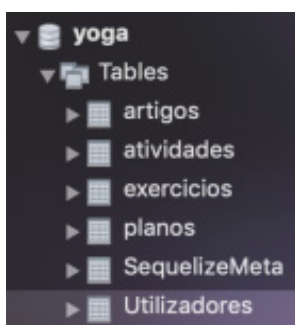


Figura 1 - base de dados

A figura 1 representa a estrutura da base de dados que terá todo o conteúdo da aplicação YogaGo. Seguidamente será abordado isoladamente o que contém cada tabela:

## Tabela “utilizadores”

A tabela “utilizadores” é constituída por um id (primário), nome (varchar255), email (varchar255), password (varchar255), createdAt (datetime) e updatedAt (datetime).

Esta tabela é a principal, pois gere todos os utilizadores tanto no registo na aplicação, no login como no registo de atividade da mesma.

Column	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	G	Default / Expression
id	INT	✓	✓					✓		
name	VARCHAR(255)									NULL
email	VARCHAR(255)									NULL
password	VARCHAR(255)									NULL
createdAt	DATETIME			✓						
updatedAt	DATETIME			✓						

Figura 2 - tabela utilizadores

É a partir do id do utilizador que tudo se realiza, até mesmo as relações que mencionaremos mais à frente. Quando o utilizador se regista na aplicação, a base de dados é mostrada como está representada na figura 3.

id	name	email	password	createdAt	updatedAt
1	Rute	rutarute@gmail.com	1234	2020-06-09 01:36:09	2020-06-09 01:36:09
2	António	antonio@gmail.com	12345	2020-06-10 20:08:05	2020-06-11 01:14:08
3	Rui	rui@gmail.com	2020	2020-06-11 12:05:54	2020-06-11 12:05:54
4	Rui	rui@gmail.com	2020	2020-06-11 12:06:45	2020-06-11 12:06:45
6	maria	m@soumemail.pt	ola	2020-06-13 02:10:27	2020-06-13 02:10:27
7	maria	m@soumemail.pt	ola	2020-06-13 02:10:59	2020-06-13 02:10:59
8	Rita	rita@gmail.com	ola1	2020-06-13 04:32:19	2020-06-13 04:32:19
9	Sara	sara@gmail.com	col	2020-06-13 04:37:48	2020-06-13 04:37:48
10	ana	ana@gmail.com	123	2020-06-13 20:43:07	2020-06-13 20:43:07
11	luis	luis@gmail.com	12345	2020-06-13 20:45:45	2020-06-14 00:41:30
12	luis	luis@gmail.com	12345	2020-06-13 20:48:37	2020-06-14 00:41:30
13	a	a@gmail.com	123	2020-06-14 00:43:34	2020-06-14 00:43:34
14	Filipe	filipe@gmail.com	123	2020-06-14 01:30:23	2020-06-14 01:30:23
15	ola	ola@gmail.com	111111111...	2020-06-14 01:32:35	2020-06-14 01:32:35

figura 3 - registo de utilizadores na base de dados

## Tabela “exercicios”

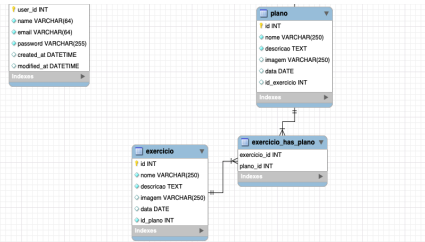
Esta tabela é composta por id (primário), nome (varchar255), descricao(vvarchar255), imagem (varchar255), createdAt (datetime), updatedAt (datetime) e também tem um “planold” associado. Aqui encontram-se todos os exercícios que irão estar visíveis na página “planos” e “cursos”.

Column	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	G	Default / Expression
id	INT	✓	✓					✓		
nome	VARCHAR(255)									NULL
imagem	VARCHAR(255)									NULL
descricao	VARCHAR(100)									NULL
createdAt	DATETIME			✓						
updatedAt	DATETIME			✓						
planold	INT									NULL

figura 4 -tabela exercicios

id	nome	imagem	descricao	createdAt
3	Uttarasana	uploads/exercicios/Sutuhandhasananga.jpg	Expire e dobre-se para a frente, deixando que o...	2020-05-28 03:10:51
4	Shivasana	uploads/exercicios/Vikasana2.jpg	Expire, levando um pé atrás de cada vez, até fic...	2020-05-28 03:10:51
5	Urdhva Mukha Shivasana	uploads/exercicios/Vikasana1.jpg	Expire e dobre os dedos, pondo os pés da tr...	2020-05-28 03:10:51
6	Navasana	uploads/exercicios/PosturaadebracoGom...	Sente-se de forma a que as plantas dos pés at...	2020-05-28 03:10:51
7	Ardra Navasana	uploads/exercicios3.jpg	Agora, balce o tronco da posição de Navasana...	2020-05-28 03:10:51
8	Postura Garudasana 1	uploads/exercicios2.jpg	Numa posição simétrica, inspira para estender o...	2020-05-28 03:10:51
9	Postura Garudasana 2	uploads/exercicios1.jpg	Inspire e levante os braços.	2020-05-28 03:10:51
10	Postura Garudasana 3	uploads/exercicios2Ardrhanavasana.jpg	Expire e mova o corpo para a frente. Inspire e v...	2020-05-28 03:10:51
11	Postura de Inga Gomukha	uploads/exercicios/navasana.jpg	Coloque o tempo direito para cima e logo a brin...	2020-05-28 03:10:51
12	Sethu Bandha Sarvangasana	uploads/exercicios/urdhvashvanasana.jpg	Deite-se com os pés apoiados no chão, à distâ...	2020-05-28 03:10:51
13	Uttarasana	uploads/exercicios/shivasana.jpg	Comече agachado. Pouse as palmas das mãos...	2020-05-28 03:10:51
14	Postura Virasana 1	uploads/exercicios/dhanasana.jpg	Ponte-se de pé. Mude o peso para o pé da...	2020-05-28 03:10:51
15	Postura Virasana 2	uploads/exercicios/tadasana.jpg	Erga os braços e junte as palmas da mão mov...	2020-05-28 03:10:51
16	Bakasana	uploads/exercicios/hasthasana.jpg	Respire fundo!	2020-05-28 03:10:51
17	Padmasana	uploads/exercicios/ML_2217.jpg	Posicione-se de pé. Junte os dedos grandes do...	2020-05-28 03:10:48

figura 5 - conteúdo da tabela exercicios



## Tabelas “artigos” e “planos”

Seguidamente, na figura 4, estão representadas a estrutura das tabelas **artigos** e **planos**. Estas são constituídas, cada uma por um id (primário), nome (varchar255), descricao(vchar255), imagem (varchar255), createdAt (datetime) e updatedAt (datetime). As duas tabelas seguem a mesma lógica.

Column	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	G	Default / Expression
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
nome	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
imagem	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
descricao	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
createdAt	DATETIME	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
updatedAt	DATETIME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

figura 7 - estrutura das tabelas artigos e planos

## Relações entre tabelas

As relações são realizadas de N para N, isto é, vários utilizadores podem fazer diversos registos de planos assim como vários registos podem ser realizados por diversos utilizadores, como é possível visualizar na figura 6.

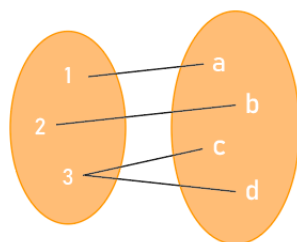


figura 8 - esquema de relação entre utilizadores e planos

# MODELAÇÃO DA BASE DE DADOS

Foi então formada uma tabela relação (atividade), onde foram inseridos o user\_id e o id\_plano. Além disso, foi adicionada a data\_inicio e a data\_fim para que seja possível o utilizador visualizar o tempo que demorou a realizar os planos. Esta relação faz com que a tabela dos utilizadores e a tabela planos se relacionem entre si.

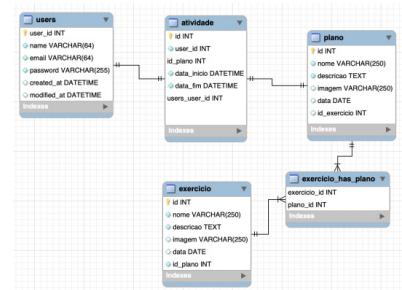


figura 9- relação entre tabelas

Na figura 9 estão representadas todas as relações do projeto, que determina alguma interatividade na aplicação.

# OPÇÕES TOMADAS AO LONGO DA IMPLEMENTAÇÃO

Ao longo do projeto, foram tomadas algumas decisões para que todo este trabalho fizesse sentido. Todas elas foram importantes na construção de toda a aplicação

## Menu

Foi produzido um menu, composto por fragments, sendo este de rápido acesso para todos os utilizadores registados.

Este é a porta de acesso para aceder a qualquer página pós login, são estas: home, planos, cursos , ler+, e por fim o perfil. Apenas quando o utilizador clica num dos ícones é que aparece o nome da página, de modo a que não tivesse demasiada informação num menu tão “pequeno”.

regista a data e a hora a que o plano foi terminado. Sempre que o utilizador pretender registar um plano, este processo repetir-se-á.



figura 8 - botão terminar plano

## Botão iniciar e terminar plano

Ao abrir um plano, é nos permitido visualizar uma lista de posturas que esse contém. Além disso, a mesma página inclui um botão “Iniciar plano”. Ao clicar neste, o plano selecionado é automaticamente registado na base de dados e consequentemente na página “perfil”, na aba “actividade”. A escolha desta opção veio dar interação à aplicação.



figura 8 - botão iniciar plano

Já clicado no botão, este muda de cor para vermelho com a informação “terminar plano”. Este faz exatamente o mesmo que o botão iniciar plano, mas de modo inverso, ou seja,



Estes são excertos de código mais importantes para a realização do presente trabalho.

O especificação da api do presente trabalho teve como base o ponto de partida para o desenvolvimento do mesmo, delineando a api rest como forma de orientação. Esta foi feita no Swagger.

BD ambiente de trabalho MySQL Wokbench  
(Local)

Estes servem para guardar a autenticação/  
permissão da sessão.

	POST	→ http://localhost:3000/api/atividadestart	Send
	Params	Authorization Headers	Body Pre-request Script Tests Settings
	Headers	⚙ Hide auto-generated headers	
KEY	VALUE	DESCRIPTION	BODY
<input checked="" type="checkbox"/> Cache-Control Ⓢ	no-cache		
<input checked="" type="checkbox"/> Postman-Token Ⓢ	<calculated when request is sent>		
<input checked="" type="checkbox"/> Content-Length Ⓢ	0		
<input checked="" type="checkbox"/> Host Ⓢ	<calculated when request is sent>		
<input checked="" type="checkbox"/> User-Agent Ⓢ	PostmanRuntime/7.25.0		
<input checked="" type="checkbox"/> Accept Ⓢ	*/*		
<input checked="" type="checkbox"/> Accept-Encoding Ⓢ	gzip, deflate, br		
<input checked="" type="checkbox"/> Connection Ⓢ	keep-alive		
<input checked="" type="checkbox"/> Authorization	y7hoXkAIQKvJlQLChBgGoJIU1tNj9_eYtsdWIO fEsimhdqIcMxTmOWOODISOwZWhwjowNTz MTESODKSfq_nvqBhg3n4NaG7BMcpQmc1_FEAl U4toIKdgIM2Vcs94	Description	
Response			

O multer serviu para o armazenamento no backend de imagens. Abaixo é apresentado uma imagem onde representa o método utilizado

```

113 };
114
115 //GET ARTIGO IMAGE BY:ID
116 controllers.getImage = async (req, res) => {
117   const { id } = req.params;
118   const pathImage = await Artigo.findOne({ where: { id: id } })
119     .then(function (artigos) {
120       if (artigos.length === 0) {
121         return 'Nenhum artigo com o identificador encontrado.';
122       }
123       console.log(art any);
124       return artigos.dataValues.imagem;
125     })
126     .catch((error) => {
127       res.status(404).send({
128         message: error.message || "Nenhum artigo com o identificador encontrado."
129       });
130     });
131   res.sendFile(path.join(__dirname, '..', '/', pathImage));
132 };
133
134 module.exports = controllers;

```

Este ajudou no processo de criação da base de dados com Mysql e nas relações entre tabelas (abordadas em aula)

```

1  'use strict';
2
3  module.exports = (sequelize, DataTypes) => {
4    const Utilizador = sequelize.define('Utilizador', {
5      name: DataTypes.STRING,
6      email: DataTypes.STRING,
7      password: DataTypes.STRING
8    }, {
9      timestamps: true,
10     tableName: 'utilizadores'
11   });
12   Utilizador.associate = function(models) {
13     Utilizador.hasMany(models.Atividade, {foreignKey: 'userId'});
14   };
15   return Utilizador;
16 };

```

## Repository (Android)

Esta é a “ligação” relativa à area atividades, relativas ao que o utilizador fez na aplicação (get).

```
//CALL ATIVIDADES
public AtividadesResponse atividades (String url, String token, int id) throws JSONException {
    String result = "";
    try {
        URL apiEnd = new URL(url + "utilizador/" + id);
        int responseCode;
        HttpURLConnection connection;
        InputStream is;
        connection = (HttpURLConnection) apiEnd.openConnection();
        connection.setRequestMethod("GET"); //PERFIL
        connection.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
        connection.setRequestProperty("Authorization", token);
        //HashMap<String, String> body = new HashMap<>();
        //body.put("email", email);
        //body.put("password", password);
        connection.setReadTimeout(15000);
        connection.setConnectTimeout(15000);
        connection.connect();
        responseCode = connection.getResponseCode();
        if (responseCode < HttpURLConnection.HTTP_BAD_REQUEST) {
            is = connection.getInputStream();
        } else {
            is = connection.getErrorStream();
        }
    }
}
```

## JSON

Este é utilizado para a conversão das strings que vêm do backend e são transformadas em objetos por este conversor.

## Shared Preferences

O próximo método guarda todas as informações do utilizador.

```
import com.google.gson.Gson;

import pt.iset.yogago.model.LoginResponse;

public class SharedPreferencesManager {
    private static SharedPreferencesManager INSTANCE;
    private Application application;
    private static String USERPREFS_KEY = "USERPREFS_KEY";
    private static String USERPREFS = "USERPREFS";

    //Criar uma única instância do objecto (singleton)
    public SharedPreferencesManager(Application application) {
        this.application = application;
    }

    public static SharedPreferencesManager getInstance(Application application) {
        if (INSTANCE == null) {
            synchronized (SharedPreferencesManager.class) {
                if (INSTANCE == null) {
                    INSTANCE = new SharedPreferencesManager(application);
                }
            }
        }
        return INSTANCE;
    }
}
```

## Adapters

É um templates views para as fragments com as ações a despertar em cada evento, por exemplo titleHY = itemView.findViewById(R.id.titleHY) estes item views estão definidos no layout XML do fragment (Res) que despertam ações.

```
69         });
70     }
71     viewHolder.bindView(articleList.get(i));
72 }
73
74 @Override
75 public int getItemCount() {
76     return articleList.size();
77 }
78
79 class TemplateViewHolder extends RecyclerView.ViewHolder {
80     //**
81     // * Installs a new ViewHolder and binds the view with ButterKnife.
82     // * @param itemView The layout of each row.
83     //**
84     private TextView titleHY;
85     private TextView descHY;
86     private ImageView imageHY;
87     public View view;
88
89     TemplateViewHolder(@NonNull View itemView) {
90         super(itemView);
91         this.view = itemView;
92         titleHY = itemView.findViewById(R.id.titleHY);
93         descHY = itemView.findViewById(R.id.descHY);
94         imageHY = itemView.findViewById(R.id.imageHY);
95     }
96 }
97
```

## Path imagens (Android)

Este código transforma as imagens que vêm em node em “streams” para serem vistas no android. Neste também é possível ver o caminho path que vem do backend para impressão da mesma

```
//PREENCHE
void bindView(Planos planos) {
    titleCursos.setText(planos.getNome());
    descricaoCursos.setText(planos.getDescricao());
    new Thread(new Runnable() {
        @Override
        public void run() {
            Bitmap bitmap = getBitmapFromURL(BASE_URL + "imagem/plano/" + planos.getId());
            activity.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    buttonCursos.setImageBitmap(bitmap);
                }
            });
        }
    }).start();
}

public Bitmap getBitmapFromURL(String src) {
    try {
        java.net.URL url = new java.net.URL(src);
        HttpURLConnection connection = (HttpURLConnection) url
            .openConnection();
        connection.setDoInput(true);
        connection.connect();
        InputStream input = connection.getInputStream();
        Bitmap myBitmap = BitmapFactory.decodeStream(input);
        return myBitmap;
    } catch (IOException e) {
        e.printStackTrace();
        return null;
    }
}
```

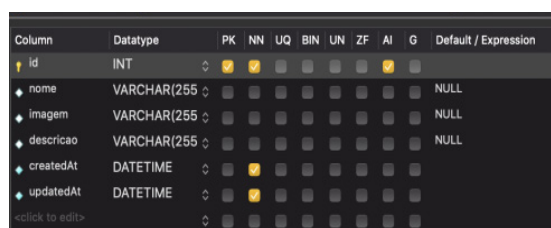
# Dificuldades no processo

O trabalho desenvolvido foi sem dúvida um grande desafio e de constante aprendizagem. Além do conteúdo lecionado nas aulas, foi necessário uma grande procura de informação para complementar toda a matéria. Nem todo o material encontrado na Internet é funcional, ou seja é necessário saber selecionar o que realmente interessa e aplicar no projeto. Contudo surgiram algumas dificuldades que atrasaram o processo do mesmo. Em baixo, estarão listadas algumas das dificuldades que surgiram ao longo do desenvolvimento da aplicação YogaGo.

## Activity / Fragment

Inicialmente, quando realizado o menu, as páginas foram desenvolvidas em Activities. Mais tarde, descobrimos que a maneira mais correta de realizar estas páginas pertencentes ao menu era através de Fragments e não de Activities. No entanto, o processo de passagem do código XML e Java das Activities para os Fragments, foi bastante complicado, nomeadamente o código Java, pois o que funciona numa Activity não é o mesmo que funciona num Fragment.

Percebemos que o código java que possibilita estar numa página e passar para outra através de um botão (OnClick) é diferente, ou seja, se for de Activity para Activity é um código, de Fragment para Fragment outro, de Fragment para Activity outro, e de Activity para Fragment outro. São códigos parecidos mas não iguais, ou seja, têm de ser adaptada a cada situação específica



Column	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	G	Default / Expression
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
nome	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
imagem	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
descricao	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
createdAt	DATETIME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
updatedAt	DATETIME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
-click to edit>		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

figura 8 - exemplo da passagem de um Fragment para uma Activity através de um botão

## Conclusão

O projeto prático desenvolvido, permitiu-nos ganhar um novo conhecimento sobre o tema “yoga” pois foram feitas bastantes pesquisas para aproximar a aplicação da realidade. Além disso, YogaGo é uma aplicação atual e bastante útil para quem deseja praticar algum exercício físico ou relaxar a mente.

Para uma primeira aplicação, consideramos que podia ser uma possível aplicação concorrente a outras idênticas no mercado.

No que diz respeito às competências adquiridas, aprendemos a utilizar o node.js e a fazer a sua ligação com a base de dados para o backend, e a trabalhar com o Android Studio para o frontend .

Concluindo, consideramos que somos capazes de elaborar qualquer aplicação com este nível de complexidade.



## Bibliografia

Este trabalho foi realizado não apenas com ajuda dos power points disponibilizados pelo professor, mas também por alguns sites e tutorias do youtube. Seguem-se os link que consultamos para responder a algumas dúvidas que surgiram, e também alguns links que nos ajudaram na inspiração do layout da aplicação YogaGo.

- <https://www.udemy.com/course/curso-completo-do-desenvolvedor-nodejs/learn/lecture/5591026?start=255#content> **curso “Curso Completp do Desenvolvedor NodeJS e MongoDB”**
- <https://stackoverflow.com/questions/18210700/best-method-to-download-image-from-url-in-android>, **site para fazer uploud do url numa imagem no Android Studio;**
- <https://stackoverflow.com/questions/8740381/getting-a-jsonexception-end-of-input-at-character-0>, !
- <https://www.udemy.com/course/aprenda-android-do-basico-ao-profissional/>, **curso “Aprenda Android do básico ao profissional”;**
- <https://www.udemy.com/course/desenvolvimento-android-do-absoluto-zero-para-iniciantes/learn/lecture/17798754?start=0#overview>, **curso “Desenvolvimento Android de Absoluto ZERO para INICIANTES”;**
- <https://www.udemy.com/course/java-curso-completo/>, **curso “Java COMPLETO 2020 Programação Orientada a Objetos + Projetos!;**
- <https://www.youtube.com/watch?v=1HeTGijQMvI&start=15s>, **site utilizado construção do menu;**
- <https://www.youtube.com/watch?v=92HbUUU4wLc>, **site utilizado para construção do menu;**
- <https://www.youtube.com/watch?v=OVmLLet9aAQ>, **site utilizado para a construção do botão “Iniciar plano” e “Terminnar plano”;**
- <https://www.youtube.com/watch?v=UsXv6VRqZKs>, **site utilizado para a construção dos CardsViews;**
- <https://www.youtube.com/watch?v=ZTlIMdH2V5g>, **site utilizado para realizar o ScrollView;**
- <https://www.youtube.com/watch?v=xJHgipZiAyU>, **site utilizado para arredondar os botões;**

- <https://stackoverflow.com/questions/5658675/replacing-a-fragment-with-another-fragment-inside-activity-group>, **site utilizado para substituir um fragmento por outro fragmento dentro dum atividade;**
- <https://stackoverflow.com/questions/19112357/java-simpledateformatyyyy-mm-ddthhmmssz-gives-timezone-as-ist, !!!!;>
- <https://stackoverflow.com/questions/20020902/android-httpurlconnection-how-to-set-post-data-in-http-body, !!!!;>
- <https://www.youtube.com/watch?v=SuJg1yg4a2E>, **site utilizado para fazer a página Ler+;**
- <https://www.youtube.com/watch?v=g1LY9ZGSP1s&vl=pt>, **site utilizado para a extilização de botões;**
- <https://www.flaticon.com/home>, **site de onde foram retirados os icons para o menu;**
- [https://www.behance.net/gallery/81051045/Plants-Need-Water-Mobile-App-Concept?tracking\\_source=search\\_projects\\_recommended%7Capp](https://www.behance.net/gallery/81051045/Plants-Need-Water-Mobile-App-Concept?tracking_source=search_projects_recommended%7Capp), **site de inspiração para o layout da página “Home”;**
- [https://www.behance.net/gallery/94516665/Stay-Home-App?tracking\\_source=search\\_projects\\_recommended%7Capp](https://www.behance.net/gallery/94516665/Stay-Home-App?tracking_source=search_projects_recommended%7Capp), **site de inspiração para o layout da página “Home”;**
- [https://www.behance.net/gallery/96897059/City-Touch-App-Redesign?tracking\\_source=search\\_projects\\_recommended](https://www.behance.net/gallery/96897059/City-Touch-App-Redesign?tracking_source=search_projects_recommended), **site de inspiração para o layout do registo de atividade;**