OFICINA ONLINE

BASE DE DADOS ONLINE PARA OFICINAS

Curso(s):	Computação Móvel
Unidade(s) Curricular(es):	Base de Dados Avançada
Ano Letivo:	2016/2017
Docente:	Dr. Carlos Fonseca
Coordenador(a) da área disciplinar:	Dr. Carlos Fonseca
Data:	03 de Fevereiro de 2017

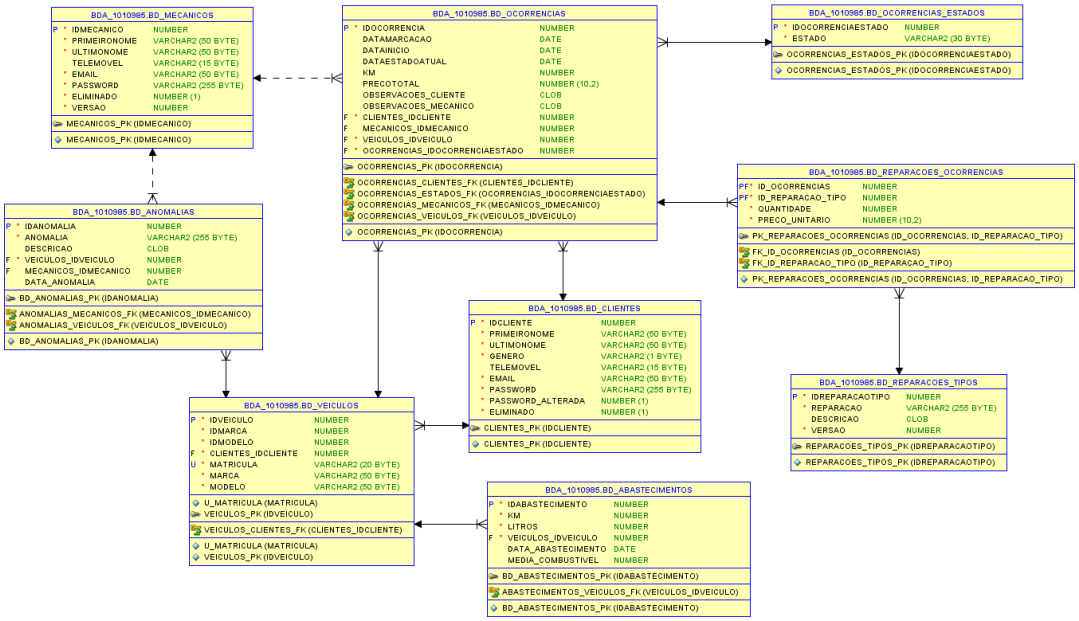
ÍNDICE

INTRODUÇÃO

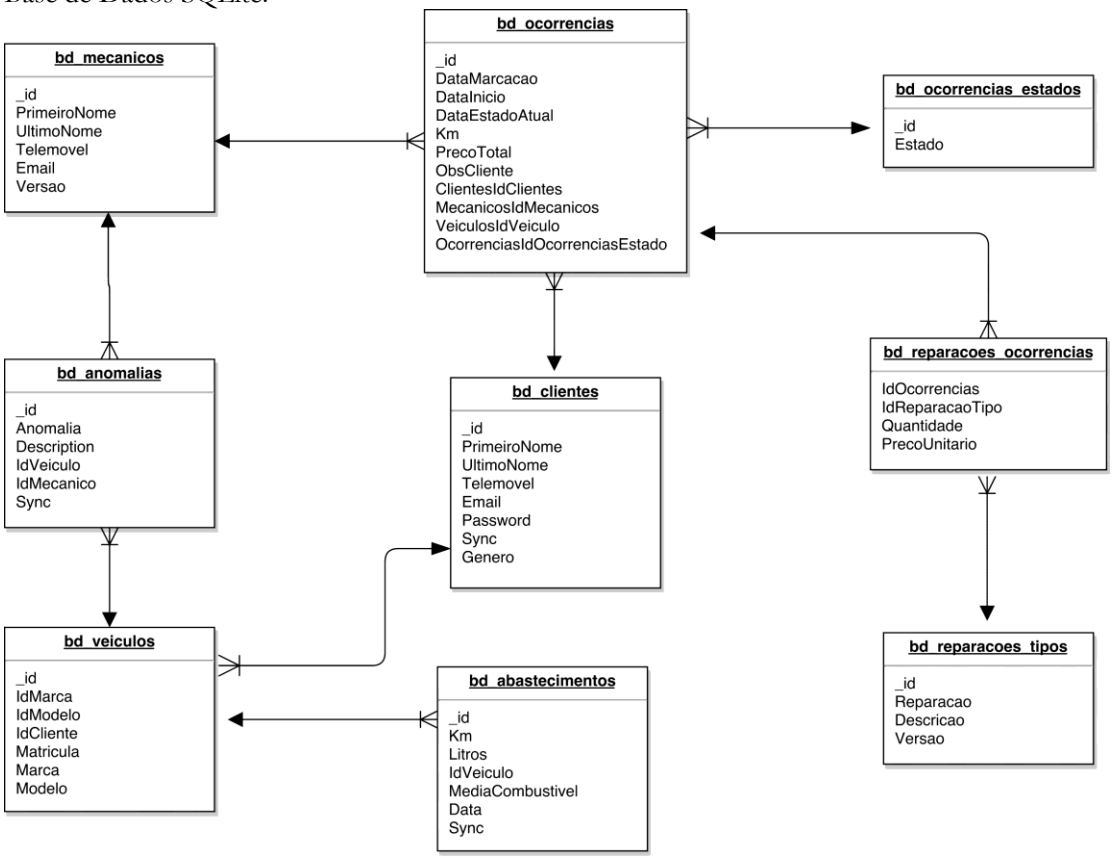
Este projeto tem como objetivo desenvolver uma aplicação que gere as manutenções e reparações de veículos dos clientes de uma oficina de automóveis. Assim, a oficina consegue manter um historial das intervenções efetuadas nos veículos e assim fornecer aos seus clientes um serviço de maior qualidade ao mesmo tempo que fidelizar o cliente. A aplicação pode ser utilizada pelos clientes para fazer marcações na oficina, registar e gerir novos veículos e registar anomalias dos veículos.

MODELO ER

Base de Dados Oracle:



Base de Dados SQLite:



DICIONÁRIO DE DADOS

BD_ABASTECIMENTOS

Tabela onde o utilizador vai registar os abastecimentos que efetua no(s) veículo(s)

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
IDABASTECIMENTO	NUMBER	(PK) ID do abastecimento	Not Null
KM	NUMBER	Km do veículo quando o abastecimento foi efetuado	Not Null
LITROS	NUMBER	Quantidade (L) abastecidos	Not Null
VEICULOS_IDVEICULO	NUMBER	(FK) ID do veículo. BD_VEICULOS.IDVEICULO	Not Null
DATA	DATE	Data do abastecimento	Not Null
MEDIA_COMBUSTIVEL	NUMBER	Média de um veículo	Null

Media_CombustíveL- CAMPO DESNORMALIZADO, DERIVADO DOS CAMPOS KM E LITROS, DE MODO A FACILITAR A CONSULTA DA MEDIA TOTAL DE UM VEICULO.

SQLite

Nome do campo	Tipo de dados	Descrição	Restrições
_id	INTEGER	(PK) ID do abastecimento	Not Null
Km	INTEGER	Km do veículo quando o abastecimento foi efetuado	Not Null
Litros	INTEGER	Quantidade (L) abastecidos	Not Null
IdVeiculo	INTEGER	(FK) ID do veículo. BD_VEICULOS.IDVEICULO	Not Null
DataAbastecimento	TEXT	Data do abastecimento	Not Null
Sync	INTEGER	Indica se o registo foi enviado p/ o servidor (0:não enviado | 1:enviado)	Not Null

BD_ANOMALIAS

Tabela onde são registadas as anomalias do veículo (registos efetuados pelo cliente e mecânico. Ex: Ruido da direção ao virar p/ a esquerda)

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
IDANOMALIA	NUMBER	(PK) ID da anomalia	Not Null
ANOMALIA	VARCHAR2(255)	Nome da anomalia	Not Null
DESCRICAO	CLOB	Descricao da anomalia	
VEICULOS_IDVEICULO	NUMBER	(FK) ID do veículo. BD_VEICULOS.IDVEICULO	Not Null
MECANICOS_IDMECANICO	NUMBER	(FK) ID do mecânico. Pode ser Null porque a anomalia pode ser adicionada por um cliente.	
DATA_ANOMALIA	DATE	Data da anomalia	Not Null

SQLite

Nome do campo	Tipo de dados	Descrição	Restrições
_id	INTEGER	(PK) ID da anomalia	Not Null
Anomalia	TEXT	Nome da anomalia	Not Null
Desc	TEXT	Descricao da anomalia	
Idveiculo	INTEGER	(FK) ID do veiculo. bd_veiculos.IdVeiculo	Not Null
IdMecanico	INTEGER	(FK) ID do mecânico. Pode ser Null porque a anomalia pode ser adicionada por um cliente. bd_mecanicos.IdMecanico	
Sync	INTEGER	Indica se o registo foi enviado p/ o servidor (0:não enviado | 1:enviado)	Not Null

BD_CLIENTES

Tabela onde são registados os clientes da oficina

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
IDCLIENTE	NUMBER	(PK) ID do cliente	Not Null
PRIMIRONOME	VARCHAR2(50)	Primeiro nome	Not Null
ULTIMONOME	VARCHAR2(50)	Último nome	Not Null
TELEMOVEL	VARCHAR2(15)	Telemóvel	
EMAIL	VARCHAR2(50)	Email	Not Null
PASSWORD	VARCHAR2(255)	Palavra-passe encriptada	Not Null
PASSWORD_ALTERADA	NUMBER(1)	Indica se o utilizardor já alterou a password	
ELIMINADO	NUMBER(1)	Indica se o utilizador foi eliminado (default: 0)	Not Null
GENERO	VARCHAR2(1)	Género do cliente (m/f)	Not Null, Check

SQLite

Nome do campo	Tipo de dados	Descrição	Restrições
_id	INTEGER	(PK) ID do cliente	Not Null
PrimeiroNome	TEXT	Primeiro nome	Not Null
UltimoNome	TEXT	Último nome	Not Null
Telemovel	TEXT	Telemóvel	
Email	TEXT	Email	Not Null
Password	TEXT	Palavra-passe encriptada	Not Null
Sync	INTEGER	Indica se o registo foi enviado p/ o servidor (0:não enviado | 1:enviado)	Not Null
Genero	TEXT	Género do cliente (m/f)	

BD_VEICULOS

Tabela onde são registados os veículos dos clientes

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
IDVEICULO	NUMBER	(PK) ID do veículo	Not Null
IDMARCA	NUMBER	(FK) ID da marca (web service)	Not Null
IDMODELO	NUMBER	(FK) ID do modelo (web service)	Not Null
CLIENTES_IDCLIENTE	NUMBER	(FK) ID do proprietário. BD_CLIENTES.IDCLIENTE	Not Null
MATRICULA	VARCHAR2(20)	Matrícula do veículo	Not Null, Unique
MARCA	VARCHAR2(50)	Marca do veículo	Not Null
MODELO	VARCHAR2(255)	Modelo do veículo	Not Null

SQLITE

Nome do campo	Tipo de dados	Descrição	Restrições
_id	INTEGER	(PK) ID do veículo	Not Null
IdMarca	INTEGER	(FK) ID da marca (web service)	Not Null
IdModelo	INTEGER	(FK) ID do modelo (web service)	Not Null
IdCliente	INTEGER	(FK) ID do proprietário. BD_CLIENTES.IDCLIENTE	Not Null
Matricula	TEXT	Matrícula do veículo	Not Null
Marca	TEXT	Marca do veículo	Not Null
Modelo	TEXT	Modelo do veículo	Not Null

BD_MECANICOS

Tabela onde são registados os mecânicos da oficina

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
IDMECANICO	NUMBER	(PK) ID do mecânico	Not Null
PRIMIRONOME	VARCHAR2(50)	Primeiro nome	Not Null
ULTIMONOME	VARCHAR2(50)	Último nome	Not Null
TELEMOVEL	VARCHAR2(15)	Telemóvel	
EMAIL	VARCHAR2(50)	Email	Not Null
PASSWORD	VARCHAR2(255)	Palavra-passe encriptada	Not Null
ELIMINADO	NUMBER(1)	Indica se o mecânico foi eliminado (default: 0)	Not Null
VERSAO	NUMBER	Indica a versão do registo. É incrementado quando existe uma alteração (default: 1)	Not Null

SQLite

Nome do campo	Tipo de dados	Descrição	Restrições
_id	INTEGER	(PK) ID do cliente	Not Null
PrimeiroNome	TEXT	Primeiro nome	Not Null
UltimoNome	TEXT	Último nome	Not Null
Telemovel	TEXT	Telemóvel	
Email	TEXT	Email	Not Null
Versao	INTEGER	Indica a versão do registo. Se for diferente do servidor, o registo deve ser atualizado	

BD_REPARACOES_TIPOS

Tabela onde são registados os tipos de reparações. Estes dados ficam disponíveis para seleção. Ex: mudança de óleo, troca de pneus, etc.

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
IDREPARACAOTIPO	NUMBER	(PK) ID do tipo de reparação	Not Null
REPARACAO	VARCHAR2(255)	Identifica o tipo de reparação. Ex: Mudança de óleo	Not Null
DESCRICAO	CLOB	Descrição do tipo de reparação	
VERSAO	NUMBER	Indica a versão do registo. É incrementado quando existe uma alteração (default: 1)	Not Null

SQLite

Nome do campo	Tipo de dados	Descrição	Restrições
_id	NUMBER	(PK) ID do tipo de reparação	Not Null
Reparacao	TEXT	Identifica o tipo de reparação. Ex: Mudança de óleo	Not Null
Descricao	TEXT	Descrição do tipo de reparação	
Versao	INTEGER	Indica a versão do registo. Se for diferente do servidor, o registo deve ser atualizado	

BD_OCORRENCIAS

Tabela onde são registados todos os dados das ocorrências. Ex: Mudança de óleo, datas, etc.

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
IDOCORRENCIA	NUMBER	(PK) ID da ocorrência	Not Null
DATAMARCACAO	DATE	Data da marcação da ocorrência	
DATAINICIO	DATE	Data em que a ocorrência foi iniciada	
DATAESTADOATUAL	DATE	Data do estado atual	Not Null
KM	NUMBER	Km do do veiculo quando foi iniciada a reparação	
PRECOTOTAL	NUMBER(10,2)	Preço total da ocorrência	
OBSERVACOES_CLIENTE	CLOB	Observações colocadas pelo mecânico	
OBSERVACOES_MECANICO	CLOB	Observações colocadas pelo cliente	
CLIENTES_IDCLIENTE	NUMBER	(FK) ID do cliente. BD_CLIENTES.IDCLIENTE	Not Null
MECANICOS_IDMECANICO	NUMBER	(FK) ID do mecânico. BD_MECANICOS.IDMECANICO	
VEICULOS_IDVEICULO	NUMBER	(FK) ID do veiculo. BD_VEICULOS.IDVEICULO	Not Null
OCORRENCIAS_IDOCORRENCIAESTADO	NUMBER	(FK) ID da ocorrências_estados. BD_OCORRENCIAS_ESTADOS. IDOCORRENCIAESTADO	Not Null

PREÇOTOTAL-> CAMPO DESNORMALIZADO, DERIVADO DA SOMA DOS OUTROS PREÇOS, DE MODO A FACILITAR A CONSULTA DO PREÇO TOTAL DE UMA ENCOMENDA.

SQLite

Nome do campo	Tipo de dados	Descrição	Restrições
_id	INTEGER	(PK) ID da ocorrência	Not Null
DataMarcacao	TEXT	Data da marcação da ocorrência	
DataInicio	TEXT	Data em que a ocorrência foi iniciada	
DataEstadoAtual	TEXT	Data do estado atual	Not Null
KM	INTEGER	Km do do veiculo quando foi iniciada a reparação	
PrecoTotal	TEXT	Preço total da ocorrência	
ObsMecanico	TEXT	Observações colocadas pelo mecânico	
ObsCliente	TEXT	Observações colocadas pelo cliente	
ClientesIdClientes	INTEGER	(FK) ID do cliente. bd_clientes.IdCliente	Not Null
MecanicosIdMecanicos	INTEGER	(FK) ID do mecânico. bd_mecanicos.IdMecanico	
VeículosIdVeiculo	INTEGER	(FK) ID do veiculo	Not Null
OcorrenciasIdOcorrenciasEstado	INTEGER	(FK) ID da bd_ocorrencias_estados. bd_ocorrencias_estados.IdOcorrenciaEstado	Not Null

BD_OCORRENCIAS_ESTADOS

Tabela onde são registados os estados das ocorrências. Ex: marcada.

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
IDOCORRENCIAESTADO	NUMBER	(PK) ID da rel. ocorrências - estados	Not Null
ESTADO	VARCHAR2(30)	Nome do estado (marcada, efetuada, pendente, cancelada)	Not Null

SQLite

Nome do campo	Tipo de dados	Descrição	Restrições
_id	INTEGER	(PK) ID da ocorrência	Not Null
Estado	TEXT	Data da marcação da ocorrência	Not Null

BD_REPARACOES_OCORRENCIAS

Tabela onde são registadas as várias reparações de cada ocorrência.

ORACLE

Nome do campo	Tipo de dados	Descrição	Restrições
ID_OCORRENCIAS	NUMBER	(PK) ID da rel. reparações - ocorrências	Not Null
ID_REPARACAO_TIPO	NUMBER	(FK) ID do tipo de reparação. BD_REPARACOES_TIPOS. IDREPARACAOTIPO	Not Null
QUANTIDADE	NUMBER	Quantidade	Not Null
PRECO_UNITARIO	NUMBER(10,2)	Preço unitário	Not Null

SQLite

Nome do campo	Tipo de dados	Descrição	Restrições
IdOcorrencias	INTEGER	(PK) ID da ocorrência	Not Null
IdReparacaoTipo	INTEGER	Data da marcação da ocorrência	Not Null
Quantidade	INTEGER	Quantidade	Not Null
PrecoUnitario	TEXT	Preço unitário	Not Null

VIEWS

VIEW SOMALITROS

Esta View devolve o consumo total de litros de cada veículo.

```
create or replace view SOMALITROS as
SELECT v.IDVEICULO, sum(a.LITROS) litros
from BD_VEICULOS v full JOIN  BD_ABASTECIMENTOS a on v.IDVEICULO =
a.VEICULOS_IDVEICULO
group by v.IDVEICULO with READ ONLY ;
```

VIEW VEICULOSPORCLIENTES

Esta View devolve a marca, modelo, matricula, consumo total de litros e o cliente de cada veículo. O consumo de litros é devolvido pela View SOMALITROS.

```
create or replace view VEICULOSPORCLIENTES AS
select v.MARCA,v.MODELO,v.MATRICULA,v.CLIENTES_IDCLIENTE,s.litros
from BD_VEICULOS v join SomaLitros s on v.IDVEICULO = s.IDVEICULO
with READ ONLY ;
```

VIEW LOGINCLIENTES

Esta View devolve os dados dos clientes para efeitos de login. A View foi feita par aumentar a segurança da aplicação devido ao atributo "READ ONLY".

```
create OR REPLACE view LOGINCLIENTES AS
  select PRIMEIRONOME,ULTIMONOME,EMAIL,PASSWORD,IDCLIENTE from
BD_CLIENTES WITH READ ONLY;
```
[1]

Estas sequências foram criadas com o propósito de incrementar as chaves primárias das tabelas automaticamente.

```
CREATE SEQUENCE seq_idCliente
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;

CREATE SEQUENCE seq_idAbastecimento
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;

CREATE SEQUENCE seq_idAnomalia
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;

CREATE SEQUENCE seq_idMecanico
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;

CREATE SEQUENCE seq_idOcorrencia
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;

CREATE SEQUENCE seq_idOcorrenciaEstado
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;

CREATE SEQUENCE seq_idReparacaoTipo
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;

CREATE SEQUENCE seq_idVeiculo
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;
```

PROCEDIMENTOS E PACKAGE

Nesta secção, apresentamos os procedimentos e o package criados na nossa base de dados.

O seguinte package foi criado para organizar os nossos procedimentos de web service. Contém os procedimentos WS_VERVEICULOS, WS_INSEREOCORRENCIA, WS_REGISTAVEICULO e WS_LOGIN.

```
create or replace package WS as
procedure WS_VerVeiculos(id_cliente number,v_email
varchar2,v_password varchar2);
procedure WS_InsereOcorrencia (v_dMarcacao date,
    v_dInicio DATE,
    v_km NUMBER,
    v_obsC CLOB,
    v_obsM CLOB,
    v_idcliente NUMBER,
    v_idmecanico NUMBER,
    v_idveiculo number,
    v_estado number,
    v_email varchar2,
    v_password varchar2);
procedure WS_RegistaVeiculo (v_idmarca number, v_idmodelo number,
  v_marca varchar2, v_modelo varchar2, v_matricula varchar2,
v_idcliente number, v_email varchar2, v_password varchar2) ;
procedure WS_LOGIN (vemail varchar2, vpassword varchar2) ;

end WS;
```

Nesta secção, apresentamos os procedimentos de web service, criados dentro do package anterior. Todos os procedimentos verificam a autenticação do utilizador.

WS_INSEREOCORRENCIA

Este webservice insere uma ocorrência na base de dados. Esta ocorrência pode ser um serviço efetuado pelo mecânico ou uma marcação efetuada pelo cliente.

```
create or replace procedure WS_InsereOcorrencia
  (v_dMarcacao date,
    v_dInicio DATE,
    v_km NUMBER,
    v_obsC CLOB,
    v_obsM CLOB,
    v_idcliente NUMBER,
    v_idmecanico NUMBER,
    v_idveiculo number,
    v_estado number,
    v_email varchar2,
    v_password varchar2) is

    id_cliente number;
    resultado varchar2(10000);
  BEGIN

    select idCliente into id_cliente from bd_clientes where
email=v_email and password=v_password;
  INSERT INTO BD_OCORRENCIAS

(DATAMARCACAO,DATAINICIO,KM,PRECOTOTAL,OBSERVACOES_CLIENTE,OBSERVACOE
S_MECANICO,

CLIENTES_IDCLIENTE,MECANICOS_IDMECANICO,VEICULOS_IDVEICULO,OCORRENCIA
S_IDOCORRENCIAESTADO)
  VALUES

(v_dMarcacao,v_dInicio,v_km,0,v_obsC,v_obsM,v_idcliente,v_idmecanico,
v_idveiculo,v_estado);
    resultado:=resultado || '{"sucesso": "'||1||'"}';
    htp.print(resultado);
    EXCEPTION WHEN NO_DATA_FOUND  THEN
        htp.p('Acesso Negado');
    WHEN OTHERS THEN
          resultado:=resultado || '{"Erro": "'||SQLERRM||'"}';
          htp.print(resultado);

end;
```

WS_REGISTAVEICULO

Este web service regista um veículo de um cliente na base de dados.

```
create or replace procedure WS_RegistaVeiculo (v_idmarca number,
v_idmodelo number,
  v_marca varchar2, v_modelo varchar2, v_matricula varchar2,
v_idcliente number, v_email varchar2, v_password varchar2) is

  id_cliente number;
  resultado varchar2(10000);
  BEGIN
    select idCliente into id_cliente from bd_clientes where
email=v_email and password=v_password;

    INSERT INTO BD_VEICULOS
(IDMARCA,IDMODELO,MARCA,MODELO,MATRICULA,CLIENTES_IDCLIENTE)
      VALUES
(v_idmarca,v_idmodelo,v_marca,v_modelo,v_matricula,v_idcliente);

    resultado:='';
    resultado:=resultado || '{"Resposta": "Inserido com sucesso."}';
    htp.print(resultado);
 EXCEPTION
 WHEN NO_DATA_FOUND  THEN
        htp.p('Acesso Negado');
     WHEN OTHERS
      THEN
    resultado:= '{"Resposta": "Ocorreu um erro."}';
    htp.print(resultado);

end;
```

WS_VERVEICULOS

Este web service devolve todos os veículos de um utilizador.

```
  create or replace procedure WS_VerVeiculos(id_cliente
number,v_email varchar2,v_password varchar2) is

   CURSOR cursor_veiculo IS
     select MARCA,MODELO,MATRICULA,LITROS
     from VeiculosPorClientes
       where CLIENTES_IDCLIENTE = id_cliente;

  c_marca VARCHAR2(255);
  c_modelo VARCHAR2(255);
  c_matricula VARCHAR2(255);
  c_litros VARCHAR2(255);
  v_idcliente number;
  resultado CLOB;
  BEGIN
    select idCliente into v_idcliente from bd_clientes where
email=v_email and password=v_password;
    resultado:='[';
    OPEN cursor_veiculo;
    LOOP
      exit when cursor_veiculo%notfound;
      FETCH cursor_veiculo INTO c_marca, c_modelo,
c_matricula,c_litros;
      resultado:=resultado || '{"Marca": "'||c_marca||'", "Modelo":
"'
                ||c_modelo||'", "Matricula":"'||c_matricula||
                '", "Litros":"'||c_litros||'"},';
    END LOOP;
    CLOSE cursor_veiculo;
    resultado:=substr(resultado,0,length(resultado)-1)||']';
    htp.print(resultado);
    EXCEPTION WHEN NO_DATA_FOUND  THEN
        htp.p('Acesso Negado');
    WHEN OTHERS THEN
          resultado:=resultado || '{"Erro": "'||SQLERRM||'"}';
          htp.print(resultado);
end;
```

WS_LOGIN

Este web service devolve a informação de um cliente, se o username e password estiverem corretos.

```
create or replace procedure WS_LOGIN (vemail varchar2, vpassword
varchar2) is

      c_idcliente VARCHAR2(255):=NULL;
      c_pnome VARCHAR2(255):=NULL;
      c_unome VARCHAR2(255):=NULL;
      resultado varchar2(10000);
      BEGIN
        select IDCLIENTE,PRIMEIRONOME,ULTIMONOME
        into c_idcliente,c_pnome,c_unome
        from LOGINCLIENTES
        where email=vemail
              and password=vpassword;

        resultado:='{"idCliente": "'||c_idcliente
                   ||'", "primeiroNome": "'||c_pnome
                   ||'", "ultimoNome": "'||c_unome
                   ||'", "email": "'||vemail
                   ||'", "password": "'||vpassword
                   ||'"}';
        htp.print(resultado);
     EXCEPTION
         WHEN no_data_found
          THEN
           c_idcliente := 0;
           c_pnome:='0';
           c_unome:='0';
              resultado:='{"idCliente": "'||c_idcliente
                   ||'", "primeiroNome": "'||c_pnome
                   ||'", "ultimoNome": "'||c_unome
                   ||'"}';
         htp.print(resultado);

    end; [2]
```

WS_BUSCA_MARCAS

Este procedimento busca a uma API externa e devolve todas as marcas de carros.

```
create or replace procedure ws_busca_marcas is
v_request_handle UTL_HTTP.REQ;
v_response_handle UTL_HTTP.RESP;
lv_response_line varchar(32767);
BEGIN
v_request_handle :=
UTL_HTTP.BEGIN_REQUEST('fipeapi.appspot.com/api/1/carros/marcas.json'
);
UTL_HTTP.SET_HEADER(v_request_handle, 'User-Agent', 'Mozilla/4.0');
v_response_handle := UTL_HTTP.GET_RESPONSE(v_request_handle);
UTL_HTTP.READ_LINE(v_response_handle, lv_response_line, TRUE);
htp.p(lv_response_line);
EXCEPTION
WHEN UTL_HTTP.END_OF_BODY THEN
UTL_HTTP.END_RESPONSE(v_response_handle);
END;
```

WS_BUSCA_MODELOS

Este procedimento vai buscar todos os modelos de uma marca de carros específica.

```
create or replace procedure ws_busca_modelos(id_modelo number) is
v_request_handle UTL_HTTP.REQ;
v_response_handle UTL_HTTP.RESP;
lv_response_line VARCHAR2(32767);
BEGIN
v_request_handle :=
UTL_HTTP.BEGIN_REQUEST('http://fipeapi.appspot.com/api/1/carros/veicu
los/'||id_modelo||'.json');
UTL_HTTP.SET_HEADER(v_request_handle, 'User-Agent', 'Mozilla/4.0');
UTL_HTTP.set_header (v_request_handle, 'Transfer-Encoding', 'chunked'
);
v_response_handle := UTL_HTTP.GET_RESPONSE(v_request_handle);
LOOP
UTL_HTTP.READ_TEXT(v_response_handle, lv_response_line, 32000);
htp.p(lv_response_line);
END LOOP;
EXCEPTION
WHEN UTL_HTTP.END_OF_BODY THEN
UTL_HTTP.END_RESPONSE(v_response_handle);
END;
```

Criamos estes sinónimos para simplificar nomes de tabelas extensas:

```
CREATE SYNONYM REP_OCUR
FOR BD_REPARACOES_OCORRENCIAS;

CREATE SYNONYM OCUR_ESTADO
For BD_OCORRENCIAS_ESTADOS;
```

FN_CALC_MEDIA_COMBUSTIVEL

Esta função foi criada para calcular a média de combustível que gasta um veículo:

```
create or replace function fn_calc_media_combustivel (idveiculo
number)
return NUMBER
is mediaCombustivel number(10);

maxKM number;
sumLitros number;

BEGIN

select max(KM), sum(LITROS)
into maxKM, sumLitros
from BD_ABASTECIMENTOS, BD_VEICULOS
where VEICULOS_IDVEICULO = idveiculo;

mediaCombustivel := maxKM/sumLitros;
return mediaCombustivel;

END;
```

MY_CRYPTO_SHA2

Esta função encripta e retorna, em SHA-256, um valor passado em parâmetro:

```
create or replace FUNCTION MY_CRYPTO_SHA2
(VALUE IN VARCHAR2) RETURN VARCHAR2 AS
l_hash raw(20000);
BEGIN
l_hash := dbms_crypto.hash ( UTL_I18N.STRING_TO_RAW (value,
'AL32UTF8'), dbms_crypto.hash_sh256);
RETURN to_char(l_hash);
END MY_CRYPTO_SHA2;
```

Esta função retorna o número total de veículos que existem.

```
create or replace function fn_totalVeiculos
return NUMBER
is numeroVeiculos number(10);

BEGIN

Select count(idveiculo)
into numeroVeiculos
from bd_veiculos;


return numeroVeiculos;

END;
```

[3]

FN_PRECO_TOTAL_REPARACAO

Esta função retorna o preço total de uma ocorrência.

```
CREATE OR REPLACE FUNCTION fn_preco_total_reparacao (p_id_ocorrencia
IN number)
RETURN NUMBER AS preco_total number(10,2);

BEGIN
  SELECT
    SUM(quantidade * preco_unitario) INTO preco_total
    FROM BD_REPARACOES_OCORRENCIAS
    WHERE id_ocorrencias = p_id_ocorrencia;
RETURN preco_total;
END;
```

[3]

Estes são os triggers criados para inserir as chaves primárias de cada tabela:

```
create or replace TRIGGER T_IDCLIENTE
  BEFORE INSERT ON BD_CLIENTES
  FOR EACH ROW
BEGIN
  SELECT seq_idCliente.nextval
    INTO :new.IDCLIENTE
    FROM dual;
END;

create or replace TRIGGER T_IDABASTECIMENTO
  BEFORE INSERT ON BD_ABASTECIMENTOS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDABASTECIMENTO.nextval
    INTO :new.IDABASTECIMENTO
    FROM dual;
END;

create or replace TRIGGER T_IDANOMALIA
  BEFORE INSERT ON BD_ANOMALIAS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDANOMALIA.nextval
    INTO :new.IDANOMALIA
    FROM dual;
END;

create or replace TRIGGER T_IDMECANICO
  BEFORE INSERT ON BD_MECANICOS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDMECANICO.nextval
    INTO :new.IDMECANICO
    FROM dual;
END;

create or replace TRIGGER T_IDOCORRENCIA
  BEFORE INSERT ON BD_OCORRENCIAS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDOCORRENCIA.nextval
    INTO :new.IDOCORRENCIA
    FROM dual;
END;

create or replace TRIGGER T_IDOCORRENCIAESTADO
  BEFORE INSERT ON BD_OCORRENCIAS_ESTADOS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDOCORRENCIAESTADO.nextval
    INTO :new.IDOCORRENCIAESTADO
    FROM dual;
END;

create or replace TRIGGER T_IDREPARACAOTIPO
  BEFORE INSERT ON BD_REPARACOES_TIPOS
  FOR EACH ROW
```

```
BEGIN
  SELECT SEQ_IDREPARACAOTIPO.nextval
    INTO :new.IDREPARACAOTIPO
    FROM dual;
END;

create or replace TRIGGER T_IDVEICULO
  BEFORE INSERT ON BD_VEICULOS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDVEICULO.nextval
    INTO :new.IDVEICULO
    FROM dual;
END;
```

SEGURANÇA

Nós decidimos criar três utilizadores únicos que têm acesso à base de dados: Administrador, Mecânicos e Clientes. O Administrador será utilizado para criar a base de dados e será desativado posteriormente. Os privilégios de cada utilizador encontram-se na seguinte tabela CRUD:

Tipo de utilizador da aplicação	Administrador	Mecânicos	Clientes
	oracle_admin	oracle_mecanico	oracle_cliente
BD_OCORRENCIAS	CRUD	CRUD	CRD
BD_OCORRENCIAS_ESTADOS	CRUD	R	R
BD_REPARACOES_OCORRENCIAS	CRUD	CRUD	CRUD
BD_REPARACOES_TIPOS	CRUD	CRUD	R
BD_VEICULOS	CRUD	CRD	CRD
BD_CLIENTES	CRUD	RU	RU
BD_MECANICOS	CRUD	RU	RU
BD_ANOMALIAS	CRUD	CRUD	CRUD
BD_ABASTECIMENTOS	CRUD	CRUD	CRUD

ROLES

Foram criados os seguintes roles, consoante os privilégios anteriormente sugeridos:

ROLE_CLIENTES

```
CREATE role role_clientes;
grant CREATE,SELECT,DELETE on BD_OCORRENCIAS to role_clientes;
grant SELECT on BD_OCORRENCIAS_ESTADOS to role_clientes;
grant CREATE,SELECT,ALTER,DELETE on BD_REPARACOES_OCORRENCIAS to
role_clientes;
grant SELECT on BD_REPARACOES_TIPOS to role_clientes;
grant CREATE,SELECT,DELETE on BD_VEICULOS to role_clientes;
grant SELECT,ALTER on BD_CLIENTES to role_clientes;
grant SELECT,ALTER on BD_MECANICOS to role_clientes;
grant CREATE,SELECT,ALTER,DELETE on BD_ANOMALIAS to role_clientes;
grant CREATE,SELECT,ALTER,DELETE on BD_ABASTECIMENTOS to
role_clientes;
```

ROLE_MECANICOS

```
CREATE role role_mecanicos;
grant CREATE,SELECT,ALTER,DELETE on BD_OCORRENCIAS to role_mecanicos;
grant SELECT on BD_OCORRENCIAS_ESTADOS to role_mecanicos;
grant CREATE,SELECT,ALTER,DELETE on BD_REPARACOES_OCORRENCIAS to
role_mecanicos;
grant CREATE,SELECT,ALTER,DELETE on BD_REPARACOES_TIPOS to
role_mecanicos
grant CREATE,SELECT,DELETE on BD_VEICULOS to role_mecanicos;
grant SELECT,ALTER on BD_CLIENTES to role_mecanicos;
grant SELECT,ALTER on BD_MECANICOS to role_mecanicos;
grant CREATE,SELECT,ALTER,DELETE on BD_ANOMALIAS to role_mecanicos;
grant CREATE,SELECT,ALTER,DELETE on BD_ABASTECIMENTOS to
role_mecanicos;
```

ROLE_ADMIN

```
CREATE role role_admin;
grant all on BD_OCORRENCIAS to role_admin;
grant all on BD_OCORRENCIAS_ESTADOS to role_admin;
grant all on BD_REPARACOES_OCORRENCIAS to role_admin;
grant all on BD_REPARACOES_TIPOS to role_admin
grant all on BD_VEICULOS to role_admin;
grant all on BD_CLIENTES to role_admin;
grant all on BD_MECANICOS to role_admin;
grant all on BD_ANOMALIAS to role_admin;
grant all on BD_ABASTECIMENTOS to role_admin;
```

[4]

TRANSAÇÕES

Nesta secção, apresentamos as transações criadas na nossa base de dados.

DELUSER

A seguinte transação delUser, apaga todos os dados de um utilizador de forma correta. É criado um savepoint no início da transação e, se ocorrer uma exceção, a base de dados faz um rollback até esse savepoint. Os deletes são feitos na seguinte ordem:

1. Bd_Reparacoes_Ocrrencias
2. Bd_Anomalias
3. Bd_Abastecimentos
4. Bd_Ocorrencias

```
CREATE or REPLACE PROCEDURE delUser(idUser IN NUMBER)IS

BEGIN

savepoint erro;

DELETE
FROM BD_REPARACOES_OCORRENCIAS
WHERE BD_REPARACOES_OCORRENCIAS.ID_OCORRENCIAS IN
(
SELECT IDOCORRENCIA FROM BD_OCORRENCIAS WHERE CLIENTES_IDCLIENTE =
idUser
);

DELETE FROM BD_ANOMALIAS
where BD_ANOMALIAS.VEICULOS_IDVEICULO IN
(
select idveiculo
from BD_VEICULOS where CLIENTES_IDCLIENTE = idUser
);

DELETE FROM BD_ABASTECIMENTOS
where BD_ABASTECIMENTOS.VEICULOS_IDVEICULO IN
( select idveiculo
from BD_VEICULOS where CLIENTES_IDCLIENTE = idUser
);

DELETE FROM BD_OCORRENCIAS
where CLIENTES_IDCLIENTE = idUser;

DELETE FROM bd_veiculos
where CLIENTES_IDCLIENTE = idUser;

DELETE FROM bd_clientes
where IDCLIENTE = idUser;

exception when others then rollback to erro;

END;
/
```

DELREPARACAOTIPO

A seguinte transação apaga uma reparação_tipo, juntamente com as suas ocorrências.

```
CREATE or REPLACE PROCEDURE delReparacaoTipo(idRepT IN NUMBER)IS

BEGIN

savepoint erro;

DELETE
FROM BD_REPARACOES_OCORRENCIAS
WHERE BD_REPARACOES_OCORRENCIAS.ID_REPARACAO_TIPO = idRepT;

DELETE
FROM BD_REPARACOES_TIPOS
where BD_REPARACOES_TIPOS.IDREPARACAOTIPO = idRepT;

exception when others then rollback to erro;

END;
/
```

Até este momento, foram implementadas, numa aplicação Android, as funcionalidades de login, registar um veículo e ver os veículos.

A aplicação deixa os clientes fazerem login com o seu email e password e, através do web service WS_LOGIN, a aplicação recebe os dados do cliente correspondente (Anexo A1).

Depois do login, o cliente pode escolher entre registar um veículo e ver os veículos(Anexo A2). Ao registar um veículo, o cliente é presenteado com os modelos e marcas de automóveis, que são devolvidas por uma API externa (http://fipeapi.appspot.com). Após selecionar uma marca, um modelo e introduzir a matrícula, o veículo é registado na base de dados back-end através do web service WS_REGISTAVEÍCULO (Anexo A3).

Ao selecionar a opção de ver os veículos, a aplicação vai buscar os veículos do cliente utilizando o web service WS_VERVEICULOS e devolve uma lista com todos os veículos do cliente (Anexo A4).

Foram feitas auditorias automáticas às transações mais críticas da base de dados. Optámos por não fazer auditoria aos SELECTs de forma a não sobrecarregar a base de dados e fizemos auditoria apenas às operações que modificam a base de dados:

```
audit INSERT on bd_abastecimentos by access WHENEVER successful;
audit INSERT on bd_anomalias by access WHENEVER successful;
audit INSERT on bd_clientes by access WHENEVER successful;
audit INSERT on bd_veiculos by access WHENEVER successful;
audit INSERT on bd_mecanicos by access WHENEVER successful;
audit INSERT on bd_repacacoes_tipos by access WHENEVER successful;
audit INSERT on bd_ocorrencias by access WHENEVER successful;
audit INSERT on bd_ocorrencias_estados by access WHENEVER successful;
audit INSERT on bd__reparacoes_ocorrencias by access WHENEVER
successful;

audit UPDATE on bd_abastecimentos by access WHENEVER successful;
audit UPDATE on bd_anomalias by access WHENEVER successful;
audit UPDATE on bd_clientes by access WHENEVER successful;
audit UPDATE on bd_veiculos by access WHENEVER successful;
audit UPDATE on bd_mecanicos by access WHENEVER successful;
audit UPDATE on bd_repacacoes_tipos by access WHENEVER successful;
audit UPDATE on bd_ocorrencias by access WHENEVER successful;
audit UPDATE on bd_ocorrencias_estados by access WHENEVER successful;
audit UPDATE on bd__reparacoes_ocorrencias by access WHENEVER
successful;

audit DELETE on bd_abastecimentos by access WHENEVER successful;
audit DELETE on bd_anomalias by access WHENEVER successful;
audit DELETE on bd_clientes by access WHENEVER successful;
audit DELETE on bd_veiculos by access WHENEVER successful;
audit DELETE on bd_mecanicos by access WHENEVER successful;
audit DELETE on bd_repacacoes_tipos by access WHENEVER successful;
audit DELETE on bd_ocorrencias by access WHENEVER successful;
audit DELETE on bd_ocorrencias_estados by access WHENEVER successful;
audit DELETE on bd__reparacoes_ocorrencias by access WHENEVER
successful;
```

CONCLUSÃO

Este projeto não só foi uma oportunidade para alargar os nossos conhecimentos em programação para PL/SQL com webservices, segurança na base de dados, permissões e acessos para utilizadores e auditoria de base de dados, mas também uma experiência educativa na implementação destes conceitos.

Apesar de só termos implementados uma pequena parte da aplicação, temos a certeza que aplicaremos estes conceitos no futuro, quer na interação entre a aplicação e a base de dados, quer na integridade da própria base de dados.

BIBLIOGRAFIA

[1] "Managing Views," Oracle, [Online]. Available: https://docs.oracle.com/cd/B28359_01/server.111/b28310/views001.htm#ADMIN11774. [Acedido em 01 02 2017].

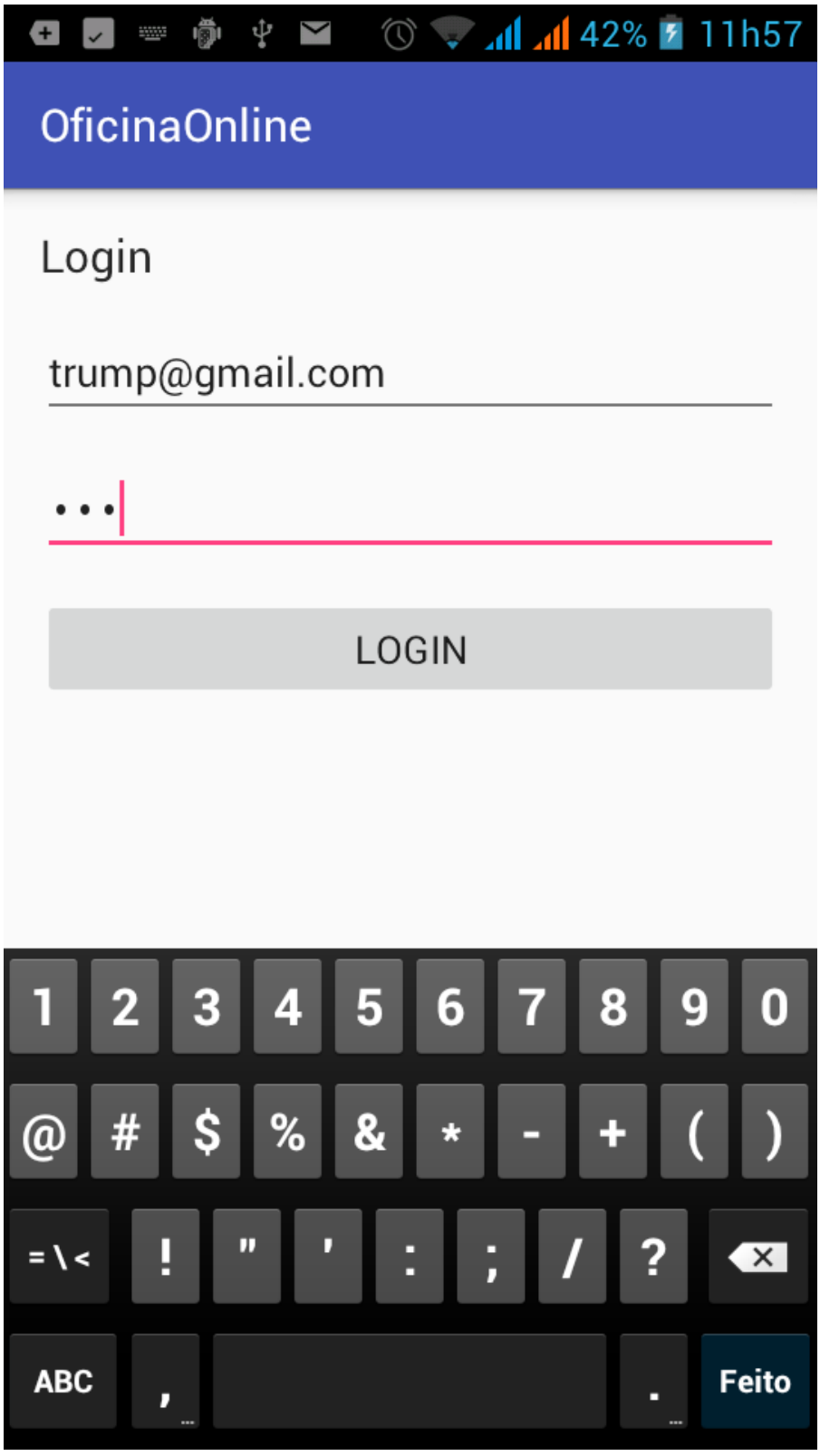
[2] J. C. Fonseca, "Encriptação," Guarda.

[3] J. C. Fonseca, "Privilégios e Roles".

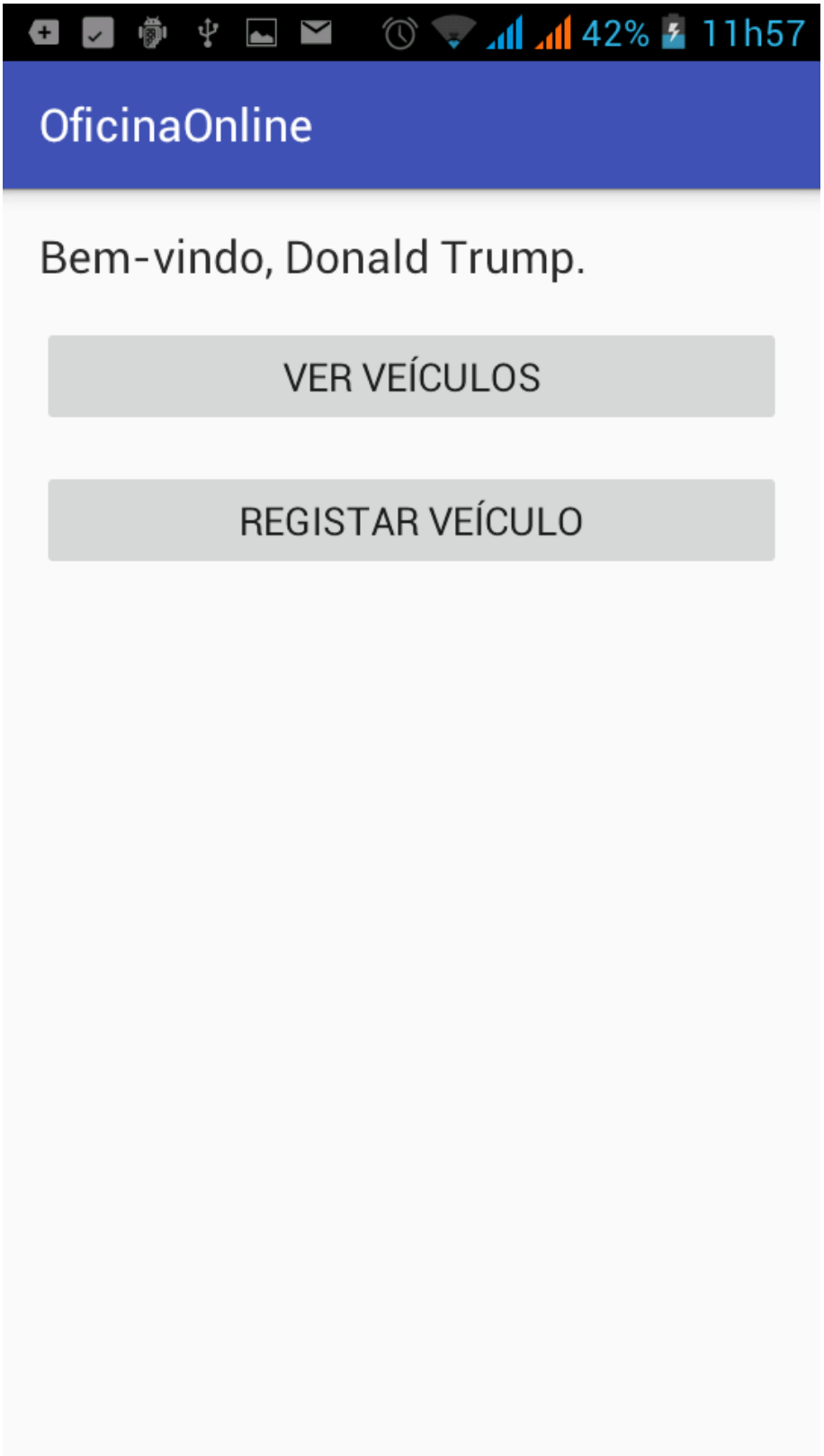
[4] J. C. Fonseca, "Oracle Access a Soap WebServices," Guarda.

ANEXO A

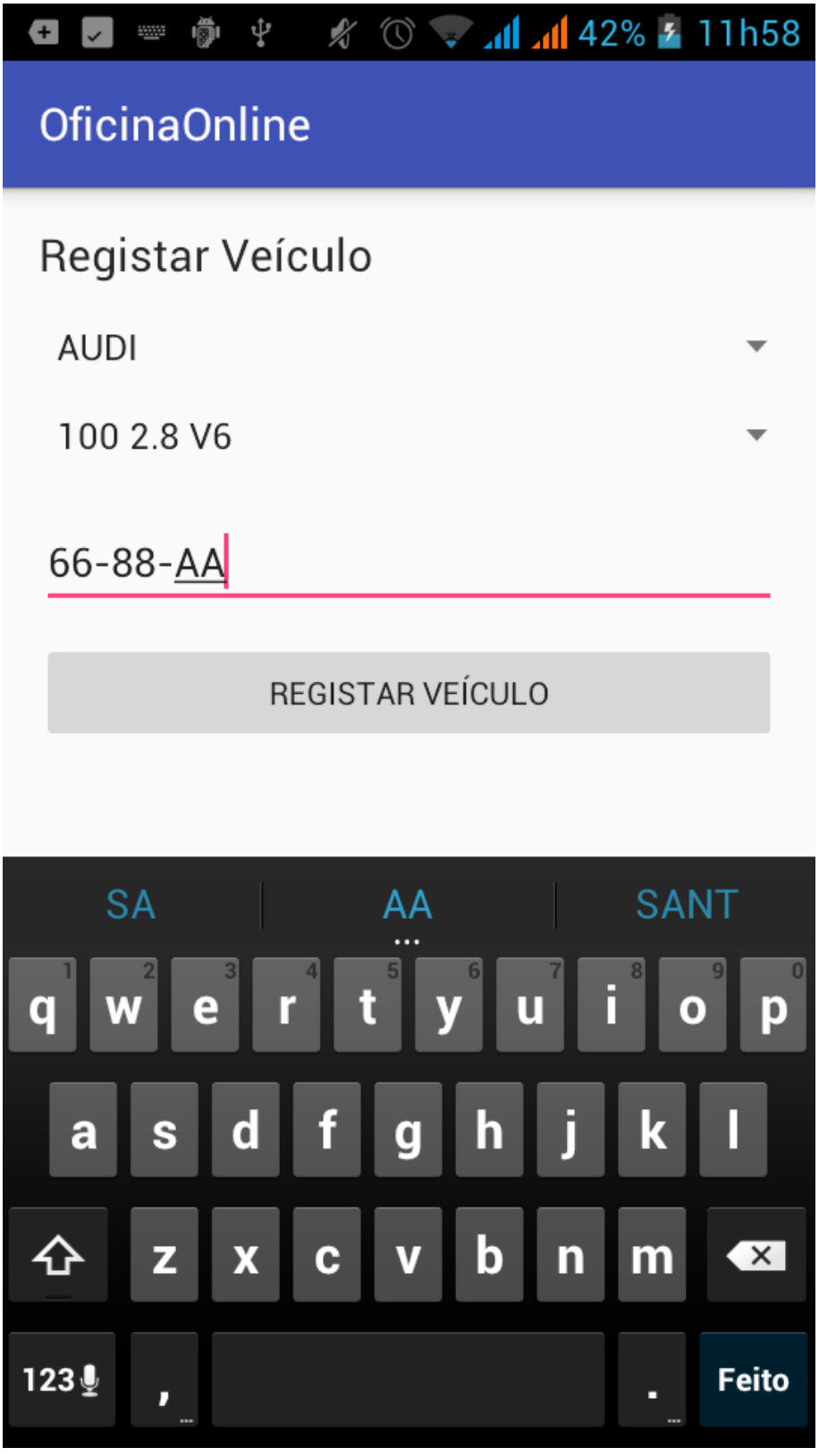
1)


42%
11h57
OficinaOnline
Login
trump@gmail.com
LOGIN
1 2 3 4 5 6 7 8 9 0
@ # $ % & * - + ( )
=\< ! " ' : ; / ?
ABC
Feito

2)


42%
11h57
OficinaOnline
Bem-vindo, Donald Trump.
VER VEÍCULOS
REGISTAR VEÍCULO

3)


42% 11h58
OficinaOnline
Registar Veículo
AUDI
100 2.8 V6
66-88-AA
REGISTAR VEÍCULO
SA
AA
SANT
Feito

4)


42%
11h57
OficinaOnline
Veiculos do Cliente
Marca: CITROEN.
Modelo: Xsara Picasso Exclus. 1.6/ 1.6 Flex 16V.
Matricula: aaavvv111.
Consumo total: .
Marca: Ferrari.
Modelo: 360 Modena.
Matricula: 11-22-AA.
Consumo total: 45.
Marca: Lamborguini.
Modelo: Aventadoro.
Matricula: 22-33-BB.
Consumo total: 60.
Marca: Bentley.
Modelo: Continental.
Matricula: 33-44-CC.
Consumo total: 30.
Marca: BMW.
Modelo: 430i Cab. Sport Limited Ed.2.0 TB 2p.

ANEXO B

```
-- v.6.1.4 (07.02.2017)
-- INSERT do utilizador para autênticação (Donald Trump) (pass em
plain text) + views

-- tables
drop table bd_abastecimentos cascade constraints;
drop table bd_anomalias cascade constraints;
drop table bd_reparacoes_ocorrencias cascade constraints;
drop table bd_ocorrencias cascade constraints;
drop table bd_reparacoes_tipos cascade constraints;
drop table bd_mecanicos cascade constraints;
drop table bd_veiculos cascade constraints;
drop table bd_clientes cascade constraints;
drop table bd_ocorrencias_estados cascade constraints;

-- sequences
drop sequence SEQ_IDCLIENTE;
drop sequence SEQ_IDABASTECIMENTO;
drop sequence SEQ_IDANOMALIA;
drop sequence SEQ_IDMECANICO;
drop sequence SEQ_IDOCORRENCIA;
drop sequence SEQ_IDOCORRENCIAESTADO;
drop sequence SEQ_IDREPARACAOTIPO;
drop sequence SEQ_IDVEICULO;

-- funções
drop function MY_CRYPTO_SHA2;
create or replace FUNCTION MY_CRYPTO_SHA2
    (VALUE IN VARCHAR2) RETURN VARCHAR2 AS
    l_hash raw(20000);
    BEGIN
    l_hash := dbms_crypto.hash ( UTL_I18N.STRING_TO_RAW (value,
    'AL32UTF8'), dbms_crypto.hash_sh256);
    RETURN to_char(l_hash);
END MY_CRYPTO_SHA2;
/
drop function fn_calc_media_combustivel;

create or replace function fn_calc_media_combustivel (idveiculo
number, km number, litros number)
return NUMBER
is mediaCombustivel number(10);

BEGIN

mediaCombustivel := km/litros;

return mediaCombustivel;

END;
/

drop function fn_totalVeiculos;
create or replace function fn_totalVeiculos
return NUMBER
is numeroVeiculos number(10);

BEGIN
```

```
Select count(idveiculo)
into numeroVeiculos
from bd_veiculos;


return numeroVeiculos;

END;
/

drop function fn_preco_total_reparacao;
CREATE OR REPLACE FUNCTION fn_preco_total_reparacao (p_id_ocorrencia
IN number)
RETURN NUMBER AS preco_total number(10,2);

BEGIN
  SELECT
    SUM(quantidade * preco_unitario) INTO preco_total
    FROM BD_REPARACOES_OCORRENCIAS
    WHERE id_ocorrencias = p_id_ocorrencia;
RETURN preco_total;
END;
/


drop SYNONYM REP_OCUR;
drop SYNONYM OCUR_ESTADO;

-- triggers
-- já são eliminados pelo cascade
/*
drop trigger T_IDCLIENTE;
drop trigger T_IDABASTECIMENTO;
drop trigger T_IDANOMALIA;
drop trigger T_IDMECANICO;
drop trigger T_IDOCORRENCIA;
drop trigger T_IDOCORRENCIAESTADO;
drop trigger T_IDREPARACAOTIPO;
drop trigger T_IDVEICULO;
*/
-- drop trigger T_INSERE_DATA_ABASTECIMENTO;
-- drop trigger T_INSERE_DATA_ANOMALIA;

CREATE TABLE BD_ABASTECIMENTOS
  (
    IDABASTECIMENTO    NUMBER NOT NULL ,
    KM                 NUMBER NOT NULL ,
    LITROS             NUMBER NOT NULL ,
    VEICULOS_IDVEICULO NUMBER NOT NULL ,
    DATA_ABASTECIMENTO DATE default sysdate,
    MEDIA_COMBUSTIVEL NUMBER
  );

CREATE UNIQUE INDEX BD_ABASTECIMENTOS_PK ON BD_ABASTECIMENTOS
  (
    IDABASTECIMENTO ASC
  );

ALTER TABLE BD_ABASTECIMENTOS ADD CONSTRAINT BD_ABASTECIMENTOS_PK
PRIMARY KEY ( IDABASTECIMENTO ) USING INDEX BD_ABASTECIMENTOS_PK ;
```

```
CREATE TABLE BD_ANOMALIAS
  (
    IDANOMALIA NUMBER NOT NULL ,
    ANOMALIA   VARCHAR2 (255 BYTE) NOT NULL ,
    DESCRICAO CLOB ,
    VEICULOS_IDVEICULO   NUMBER NOT NULL ,
    MECANICOS_IDMECANICO NUMBER ,
    DATA_ANOMALIA DATE default sysdate
  );

CREATE UNIQUE INDEX BD_ANOMALIAS_PK ON BD_ANOMALIAS
  (
    IDANOMALIA ASC
  );
ALTER TABLE BD_ANOMALIAS ADD CONSTRAINT BD_ANOMALIAS_PK PRIMARY KEY (
IDANOMALIA ) USING INDEX BD_ANOMALIAS_PK ;


CREATE TABLE BD_CLIENTES
  (
    IDCLIENTE   NUMBER NOT NULL ,
    PRIMEIRONOME VARCHAR2 (50 BYTE) NOT NULL ,
    ULTIMONOME  VARCHAR2 (50 BYTE) NOT NULL ,
    GENERO      VARCHAR2 (1 BYTE) NOT NULL,
    TELEMOVEL   VARCHAR2 (15 BYTE) ,
    EMAIL       VARCHAR2 (50 BYTE) NOT NULL,
    PASSWORD    VARCHAR2 (255 BYTE) NOT NULL,
    PASSWORD_ALTERADA    NUMBER(1) default 0 not null,
    ELIMINADO    NUMBER(1) default 0 not null
  );
CREATE UNIQUE INDEX CLIENTES_PK ON BD_CLIENTES
  (
    IDCLIENTE ASC
  );
ALTER TABLE BD_CLIENTES ADD CONSTRAINT CLIENTES_PK PRIMARY KEY (
IDCLIENTE ) USING INDEX CLIENTES_PK ;
ALTER TABLE BD_CLIENTES ADD CONSTRAINT CHECK_GENERO CHECK (GENERO
IN('M', 'F'));

CREATE TABLE BD_MECANICOS
  (
    IDMECANICO   NUMBER NOT NULL ,
    PRIMEIRONOME VARCHAR2 (50 BYTE) NOT NULL ,
    ULTIMONOME  VARCHAR2 (50 BYTE) NOT NULL ,
    TELEMOVEL   VARCHAR2 (15 BYTE) ,
    EMAIL       VARCHAR2 (50 BYTE) NOT NULL,
    PASSWORD    VARCHAR2 (255 BYTE) NOT NULL,
    ELIMINADO    NUMBER(1) default 0 not null,
    VERSAO    NUMBER not null
  );
CREATE UNIQUE INDEX MECANICOS_PK ON BD_MECANICOS
  (
    IDMECANICO ASC
  );
ALTER TABLE BD_MECANICOS ADD CONSTRAINT MECANICOS_PK PRIMARY KEY (
IDMECANICO ) USING INDEX MECANICOS_PK ;


CREATE TABLE BD_OCORRENCIAS
  (
```

```
    IDOCORRENCIA     NUMBER NOT NULL ,
    DATAMARCACAO     DATE ,
    DATAINICIO       DATE default sysdate,
    DATAESTADOATUAL DATE , --NULL AGORA, depois NOT NULL, porque
depois e necessario por isto automatico
    KM               NUMBER ,
    PRECOTOTAL       NUMBER (10,2) ,
    OBSERVACOES_CLIENTE CLOB ,
    OBSERVACOES_MECANICO CLOB ,
    CLIENTES_IDCLIENTE              NUMBER NOT NULL ,
    MECANICOS_IDMECANICO            NUMBER ,
    VEICULOS_IDVEICULO              NUMBER NOT NULL ,
    OCORRENCIAS_IDOCORRENCIAESTADO NUMBER NOT NULL
  );

CREATE UNIQUE INDEX OCORRENCIAS_PK ON BD_OCORRENCIAS
  (
    IDOCORRENCIA ASC
  );
ALTER TABLE BD_OCORRENCIAS ADD CONSTRAINT OCORRENCIAS_PK PRIMARY KEY
( IDOCORRENCIA ) USING INDEX OCORRENCIAS_PK ;


CREATE TABLE BD_OCORRENCIAS_ESTADOS
  (
    IDOCORRENCIAESTADO NUMBER NOT NULL ,
    ESTADO             VARCHAR2 (30 BYTE) NOT NULL
  );
CREATE UNIQUE INDEX OCORRENCIAS_ESTADOS_PK ON BD_OCORRENCIAS_ESTADOS
  (
    IDOCORRENCIAESTADO ASC
  );
ALTER TABLE BD_OCORRENCIAS_ESTADOS ADD CONSTRAINT
OCORRENCIAS_ESTADOS_PK PRIMARY KEY ( IDOCORRENCIAESTADO ) USING INDEX
OCORRENCIAS_ESTADOS_PK ;



CREATE TABLE BD_REPARACOES_OCORRENCIAS
  (
    ID_OCORRENCIAS      NUMBER NOT NULL,
    ID_REPARACAO_TIPO   NUMBER NOT NULL,
    QUANTIDADE          NUMBER NOT NULL ,
    PRECO_UNITARIO      NUMBER (10,2) NOT NULL
  );
ALTER TABLE BD_REPARACOES_OCORRENCIAS ADD CONSTRAINT
PK_REPARACOES_OCORRENCIAS PRIMARY KEY ( ID_OCORRENCIAS,
ID_REPARACAO_TIPO ) ;



CREATE TABLE BD_REPARACOES_TIPOS
  (
    IDREPARACAOTIPO NUMBER NOT NULL ,
    REPARACAO       VARCHAR2 (255 BYTE) NOT NULL ,
    DESCRICAO CLOB ,
    VERSAO    NUMBER not null
  );
CREATE UNIQUE INDEX REPARACOES_TIPOS_PK ON BD_REPARACOES_TIPOS
  (
```

```
    IDREPARACAOTIPO ASC
  );
ALTER TABLE BD_REPARACOES_TIPOS ADD CONSTRAINT REPARACOES_TIPOS_PK
PRIMARY KEY ( IDREPARACAOTIPO ) USING INDEX REPARACOES_TIPOS_PK ;


CREATE TABLE BD_VEICULOS
  (
    IDVEICULO          NUMBER NOT NULL ,
    IDMARCA            NUMBER NOT NULL ,
    IDMODELO           NUMBER NOT NULL ,
    CLIENTES_IDCLIENTE NUMBER NOT NULL ,
    MATRICULA          VARCHAR2 (20 BYTE) NOT NULL  ,
    MARCA              VARCHAR2 (50 BYTE) NOT NULL ,
    MODELO             VARCHAR2 (50 BYTE) NOT NULL
  );
CREATE UNIQUE INDEX VEICULOS_PK ON BD_VEICULOS
  (
    IDVEICULO ASC
  );
ALTER TABLE BD_VEICULOS ADD CONSTRAINT VEICULOS_PK PRIMARY KEY (
IDVEICULO ) USING INDEX VEICULOS_PK ;
ALTER TABLE BD_VEICULOS ADD (CONSTRAINT U_MATRICULA
UNIQUE(MATRICULA));


ALTER TABLE BD_ABASTECIMENTOS ADD CONSTRAINT
ABASTECIMENTOS_VEICULOS_FK FOREIGN KEY ( VEICULOS_IDVEICULO )
REFERENCES BD_VEICULOS ( IDVEICULO ) NOT DEFERRABLE ;


ALTER TABLE BD_REPARACOES_OCORRENCIAS ADD CONSTRAINT
FK_ID_OCORRENCIAS FOREIGN KEY ( ID_OCORRENCIAS ) REFERENCES
BD_OCORRENCIAS ( IDOCORRENCIA ) NOT DEFERRABLE ;
ALTER TABLE BD_REPARACOES_OCORRENCIAS ADD CONSTRAINT
FK_ID_REPARACAO_TIPO FOREIGN KEY ( ID_REPARACAO_TIPO ) REFERENCES
BD_REPARACOES_TIPOS ( IDREPARACAOTIPO ) NOT DEFERRABLE ;


ALTER TABLE BD_ANOMALIAS ADD CONSTRAINT ANOMALIAS_MECANICOS_FK
FOREIGN KEY ( MECANICOS_IDMECANICO ) REFERENCES BD_MECANICOS (
IDMECANICO ) NOT DEFERRABLE ;

ALTER TABLE BD_ANOMALIAS ADD CONSTRAINT ANOMALIAS_VEICULOS_FK FOREIGN
KEY ( VEICULOS_IDVEICULO ) REFERENCES BD_VEICULOS ( IDVEICULO ) NOT
DEFERRABLE ;

ALTER TABLE BD_OCORRENCIAS ADD CONSTRAINT OCORRENCIAS_CLIENTES_FK
FOREIGN KEY ( CLIENTES_IDCLIENTE ) REFERENCES BD_CLIENTES ( IDCLIENTE
) NOT DEFERRABLE ;

ALTER TABLE BD_OCORRENCIAS ADD CONSTRAINT OCORRENCIAS_ESTADOS_FK
FOREIGN KEY ( OCORRENCIAS_IDOCORRENCIAESTADO ) REFERENCES
BD_OCORRENCIAS_ESTADOS ( IDOCORRENCIAESTADO ) NOT DEFERRABLE ;

ALTER TABLE BD_OCORRENCIAS ADD CONSTRAINT OCORRENCIAS_MECANICOS_FK
FOREIGN KEY ( MECANICOS_IDMECANICO ) REFERENCES BD_MECANICOS (
IDMECANICO ) NOT DEFERRABLE ;
```

```
ALTER TABLE BD_OCORRENCIAS ADD CONSTRAINT OCORRENCIAS_VEICULOS_FK
FOREIGN KEY ( VEICULOS_IDVEICULO ) REFERENCES BD_VEICULOS ( IDVEICULO
) NOT DEFERRABLE ;

ALTER TABLE BD_VEICULOS ADD CONSTRAINT VEICULOS_CLIENTES_FK FOREIGN
KEY ( CLIENTES_IDCLIENTE ) REFERENCES BD_CLIENTES ( IDCLIENTE ) NOT
DEFERRABLE ;

-- Sinonimos

  CREATE SYNONYM REP_OCUR
  FOR BD_REPARACOES_OCORRENCIAS;

  CREATE SYNONYM OCUR_ESTADO
  For BD_OCORRENCIAS_ESTADOS;

-- Sequences

  CREATE SEQUENCE seq_idCliente
  START WITH 1
  INCREMENT BY 1
  NOCACHE
  NOCYCLE;

  CREATE SEQUENCE seq_idAbastecimento
  START WITH 1
  INCREMENT BY 1
  NOCACHE
  NOCYCLE;

  CREATE SEQUENCE seq_idAnomalia
  START WITH 1
  INCREMENT BY 1
  NOCACHE
  NOCYCLE;

  CREATE SEQUENCE seq_idMecanico
  START WITH 1
  INCREMENT BY 1
  NOCACHE
  NOCYCLE;

  CREATE SEQUENCE seq_idOcorrencia
  START WITH 1
  INCREMENT BY 1
  NOCACHE
  NOCYCLE;

  CREATE SEQUENCE seq_idOcorrenciaEstado
  START WITH 1
  INCREMENT BY 1
  NOCACHE
  NOCYCLE;

  CREATE SEQUENCE seq_idReparacaoTipo
  START WITH 1
  INCREMENT BY 1
  NOCACHE
  NOCYCLE;

  CREATE SEQUENCE seq_idVeiculo
```

```
  START WITH 1
  INCREMENT BY 1
  NOCACHE
  NOCYCLE;

-- Triggers

  create or replace TRIGGER T_IDCLIENTE
  BEFORE INSERT ON BD_CLIENTES
  FOR EACH ROW
BEGIN
  SELECT seq_idCliente.nextval
    INTO :new.IDCLIENTE
    FROM dual;
END;
/

create or replace TRIGGER T_IDABASTECIMENTO
  BEFORE INSERT ON BD_ABASTECIMENTOS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDABASTECIMENTO.nextval
    INTO :new.IDABASTECIMENTO
    FROM dual;
END;
/

create or replace TRIGGER T_IDANOMALIA
  BEFORE INSERT ON BD_ANOMALIAS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDANOMALIA.nextval
    INTO :new.IDANOMALIA
    FROM dual;
END;
/

create or replace TRIGGER T_IDMECANICO
  BEFORE INSERT ON BD_MECANICOS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDMECANICO.nextval
    INTO :new.IDMECANICO
    FROM dual;
END;
/

create or replace TRIGGER T_IDOCORRENCIA
  BEFORE INSERT ON BD_OCORRENCIAS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDOCORRENCIA.nextval
    INTO :new.IDOCORRENCIA
    FROM dual;
END;
/

create or replace TRIGGER T_IDOCORRENCIAESTADO
  BEFORE INSERT ON BD_OCORRENCIAS_ESTADOS
  FOR EACH ROW
BEGIN
```

```
  SELECT SEQ_IDOCORRENCIAESTADO.nextval
    INTO :new.IDOCORRENCIAESTADO
    FROM dual;
END;
/

create or replace TRIGGER T_IDREPARACAOTIPO
  BEFORE INSERT ON BD_REPARACOES_TIPOS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDREPARACAOTIPO.nextval
    INTO :new.IDREPARACAOTIPO
    FROM dual;
END;
/

create or replace TRIGGER T_IDVEICULO
  BEFORE INSERT ON BD_VEICULOS
  FOR EACH ROW
BEGIN
  SELECT SEQ_IDVEICULO.nextval
    INTO :new.IDVEICULO
    FROM dual;
END;
/

CREATE or replace TRIGGER T_MEDIA_COMBUSTIVEL
  BEFORE INSERT
  ON BD_ABASTECIMENTOS
  FOR EACH ROW

DECLARE

BEGIN

:new.MEDIA_COMBUSTIVEL :=
FN_CALC_MEDIA_COMBUSTIVEL(:new.VEICULOS_IDVEICULO, :new.km,
:new.litros);

END;
/


CREATE or REPLACE PROCEDURE delUser
(idUser IN NUMBER)

IS

BEGIN

savepoint erro;

DELETE
FROM BD_REPARACOES_OCORRENCIAS
WHERE BD_REPARACOES_OCORRENCIAS.ID_OCORRENCIAS IN
(
SELECT IDOCORRENCIA FROM BD_OCORRENCIAS WHERE CLIENTES_IDCLIENTE =
idUser
);

-- IN porque pode devolver mais que um registo
```

```
DELETE FROM BD_ANOMALIAS
where BD_ANOMALIAS.VEICULOS_IDVEICULO IN
(
select idveiculo
from BD_VEICULOS where CLIENTES_IDCLIENTE = idUser
);


DELETE FROM BD_ABASTECIMENTOS
where BD_ABASTECIMENTOS.VEICULOS_IDVEICULO IN
( select idveiculo
from BD_VEICULOS where CLIENTES_IDCLIENTE = idUser
);

DELETE FROM BD_OCORRENCIAS
where CLIENTES_IDCLIENTE = idUser;

DELETE FROM bd_veiculos
where CLIENTES_IDCLIENTE = idUser;

DELETE FROM bd_clientes
where IDCLIENTE = idUser;

exception   when others then rollback to erro;

END;
/

CREATE or REPLACE PROCEDURE delReparacaoTipo(idRepT IN NUMBER)IS

BEGIN

savepoint erro;

DELETE
FROM BD_REPARACOES_OCORRENCIAS
WHERE BD_REPARACOES_OCORRENCIAS.ID_REPARACAO_TIPO = idRepT;

DELETE
FROM BD_REPARACOES_TIPOS
where BD_REPARACOES_TIPOS.IDREPARACAOTIPO = idRepT;

exception when others then rollback to erro;

END;
/


-- ****************** dummy data ******************

INSERT INTO BD_CLIENTES
(PRIMEIRONOME, ULTIMONOME, GENERO, TELEMOVEL, EMAIL, PASSWORD)
VALUES
('Donald', 'Trump', 'M', 961236819, 'trump@gmail.com', '123');

update bd_clientes SET PASSWORD=MY_CRYPTO_SHA2(PASSWORD);

INSERT INTO BD_VEICULOS
(IDMARCA, IDMODELO, CLIENTES_IDCLIENTE, MATRICULA, MARCA, MODELO)
VALUES
(1, 1, 1, '11-22-AA', 'Ferrari', '360 Modena');
```

```
INSERT INTO BD_VEICULOS
(IDMARCA, IDMODELO, CLIENTES_IDCLIENTE, MATRICULA, MARCA, MODELO)
VALUES
(2, 2, 1, '22-33-BB', 'Lamborguini', 'Aventadoro');

INSERT INTO BD_VEICULOS
(IDMARCA, IDMODELO, CLIENTES_IDCLIENTE, MATRICULA, MARCA, MODELO)
VALUES
(3, 3, 1, '33-44-CC', 'Bentley', 'Continental');


INSERT INTO BD_ABASTECIMENTOS
(KM, LITROS, VEICULOS_IDVEICULO)
VALUES
(123809, 45, 1);

INSERT INTO BD_ABASTECIMENTOS
(KM, LITROS, VEICULOS_IDVEICULO)
VALUES
(98345, 60, 2);

INSERT INTO BD_ABASTECIMENTOS
(KM, LITROS, VEICULOS_IDVEICULO)
VALUES
(68123, 30, 3);


INSERT INTO BD_MECANICOS
(PRIMEIRONOME, ULTIMONOME, TELEMOVEL, EMAIL, PASSWORD,VERSAO)
VALUES
('Ricardo', 'Reis', 961678354, 'reisTeste@oficina.pt','blabla', 1);

INSERT INTO BD_MECANICOS
(PRIMEIRONOME, ULTIMONOME, TELEMOVEL, EMAIL,PASSWORD, VERSAO)
VALUES
('Alberto', 'Caeiro', 961687654, 'caeiroTeste@oficina.pt','blabla',
1);

INSERT INTO BD_MECANICOS
(PRIMEIRONOME, ULTIMONOME, TELEMOVEL, EMAIL,PASSWORD, VERSAO)
VALUES
('Alvaro', 'Campos', '961678354', 'camposTeste@oficina.pt','blabla',
1);


INSERT INTO BD_ANOMALIAS
(ANOMALIA, DESCRICAO, VEICULOS_IDVEICULO, MECANICOS_IDMECANICO)
VALUES
('Barulho', 'Barulho na direcao do lado direito', 1,2);

INSERT INTO BD_ANOMALIAS
(ANOMALIA, DESCRICAO, VEICULOS_IDVEICULO, MECANICOS_IDMECANICO)
VALUES
('Pintura danificada', 'Tejadilho do veiculo a perder cor', 3,1);


INSERT INTO BD_OCORRENCIAS_ESTADOS
(ESTADO)
VALUES
('EM APROVAÇÂO');
```

```
INSERT INTO BD_OCORRENCIAS_ESTADOS
(ESTADO)
VALUES
('PRONTO');

INSERT INTO BD_OCORRENCIAS_ESTADOS
(ESTADO)
VALUES
('EM ARRANJO');


INSERT INTO BD_REPARACOES_TIPOS
(REPARACAO, DESCRICAO, VERSAO)
VALUES
('Mudança Óleo', 'Mudança do oleo do motor do veículo', 1);

INSERT INTO BD_REPARACOES_TIPOS
(REPARACAO, DESCRICAO, VERSAO)
VALUES
('Mudança Pneus', 'Mudança de pneus', 1);

INSERT INTO BD_REPARACOES_TIPOS
(REPARACAO, DESCRICAO, VERSAO)
VALUES
('Revisao de inspeção', 'Cumprir X checkList- TODO', 1);


INSERT INTO BD_OCORRENCIAS
(KM, OBSERVACOES_CLIENTE, OBSERVACOES_MECANICO,
CLIENTES_IDCLIENTE,MECANICOS_IDMECANICO,VEICULOS_IDVEICULO,OCORRENCIA
S_IDOCORRENCIAESTADO)
VALUES
(152352, 'teste observacao cliente', 'observacoes mecanicos',1, 2, 1,
1);

INSERT INTO BD_REPARACOES_OCORRENCIAS
(ID_OCORRENCIAS, ID_REPARACAO_TIPO, QUANTIDADE, PRECO_UNITARIO)
VALUES
(1,1,3,5.5);

-- views
create or replace view SOMALITROS as
SELECT v.IDVEICULO, sum(a.LITROS) litros
from BD_VEICULOS v full JOIN  BD_ABASTECIMENTOS a on v.IDVEICULO =
a.VEICULOS_IDVEICULO
group by v.IDVEICULO with READ ONLY ;

create or replace view VEICULOSPORCLIENTES AS
select v.MARCA,v.MODELO,v.MATRICULA,v.CLIENTES_IDCLIENTE,s.litros
from BD_VEICULOS v join SomaLitros s on v.IDVEICULO = s.IDVEICULO
with READ ONLY ;

create OR REPLACE view LOGINCLIENTES AS
  select PRIMEIRONOME,ULTIMONOME,EMAIL,PASSWORD,IDCLIENTE from
BD_CLIENTES WITH READ ONLY;

grant select on SOMALITROS to scott;
grant select on VEICULOSPORCLIENTES to scott;
grant select on LOGINCLIENTES to scott;
```

```
-- CREATE SEQUENCE reparacoes_ocorrencias_reparac START WITH 1
NOCACHE ORDER ;

-- CREATE OR REPLACE TRIGGER REPARACOES_OCORRENCIAS_REPARAC BEFORE
--  INSERT ON BD_REPARACOES_OCORRENCIAS FOR EACH ROW WHEN
(NEW.ID_REP_OCORRENCIAS IS NULL) BEGIN :NEW.ID_REP_OCORRENCIAS :=
reparacoes_ocorrencias_reparac.NEXTVAL;
-- END;
```

ANEXO C

```
-- SQLite | oficina.db v.4.2.2 (13.02.2017)
-- Campo desnormalizado, Media_Combustivel -> bd_abastecimentos.

DROP TABLE IF EXISTS "bd_abastecimentos";
CREATE TABLE "bd_abastecimentos" (
    _id INTEGER CONSTRAINT pk_IdAbastecimento PRIMARY KEY,
    DSVDKm INTEGER NOT NULL,
    Litros INTEGER NOT NULL ,
    IdVeiculo INTEGER NOT NULL ,
    Data TEXT NOT NULL,
    Media_Combustivel REAL NULL
    Sync INTEGER NOT NULL DEFAULT 0
);

DROP TABLE IF EXISTS "bd_anomalias";
CREATE TABLE "bd_anomalias" (
    _id INTEGER CONSTRAINT pk_IdAnomalia PRIMARY KEY,
    Anomalia TEXT NOT NULL,
    Description TEXT ,
    IdVeiculo INTEGER NOT NULL ,
    IdMecanico INTEGER NOT NULL,
    Sync INTEGER NOT NULL DEFAULT 0
)

DROP TABLE IF EXISTS "bd_clientes";
CREATE TABLE "bd_clientes" (
    _id INTEGER CONSTRAINT pk_IdCliente PRIMARY KEY,
    PrimeiroNome TEXT NOT NULL,
    UltimoNome TEXT NOT NULL,
    Telemovel TEXT NOT NULL,
    Email TEXT NOT NULL,
    Password TEXT NOT NULL,
    Sync INTEGER NOT NULL DEFAULT 0,
    Genero TEXT
);

DROP TABLE IF EXISTS "bd_veiculos";
CREATE TABLE "bd_veiculos" (
    _id INTEGER CONSTRAINT pk_IdVeiculo PRIMARY KEY,
    IdMarca INTEGER NOT NULL,
    IdModelo INTEGER NOT NULL,
    IdCliente INTEGER NOT NULL,
    Matricula TEXT NOT NULL,
    Marca TEXT NOT NULL,
    Modelo TEXT NOT NULL
);

DROP TABLE IF EXISTS "bd_mecanicos";
CREATE TABLE "bd_mecanicos" (
    _id INTEGER CONSTRAINT pk_IdMecanico PRIMARY KEY,
    PrimeiroNome TEXT NOT NULL,
    UltimoNome TEXT NOT NULL,
    Telemovel TEXT NOT NULL,
    Email TEXT NOT NULL,
    Versao INTEGER
);

DROP TABLE IF EXISTS "bd_reparacoes_tipos";
```

```
CREATE TABLE "bd_reparacoes_tipos" (
    _id INTEGER CONSTRAINT pk_IdReparacaoTipo PRIMARY KEY,
    Reparacao TEXT NOT NULL,
    Descricao TEXT NOT NULL,
    Versao INTEGER
);

DROP TABLE IF EXISTS "bd_ocorrencias";
CREATE TABLE "bd_ocorrencias" (
    _id INTEGER CONSTRAINT pk_IdOcorrecia PRIMARY KEY,
    DataMarcacao TEXT ,
    DataInicio TEXT ,
    DataEstadoAtual TEXT NOT NULL,
    KM INTEGER ,
    PrecoTotal TEXT ,
    ObsCliente TEXT,
    ClientesIdClientes INTEGER NOT NULL ,
    MecanicosIdMecanicos INTEGER ,
    VeículosIdVeiculo INTEGER NOT NULL,V
    OcorrenciasIdOcorrenciasEstado INTEGER NOT NULL
);

DROP TABLE IF EXISTS "bd_ocorrencias_estados";
CREATE TABLE "bd_ocorrencias_estados" (
    _id INTEGER CONSTRAINT pk_IdOcorrenciaEstado PRIMARY KEY,
    Estado TEXT NOT NULL
);

DROP TABLE IF EXISTS "bd_reparacoes_ocorrencias";
CREATE TABLE "bd_reparacoes_ocorrencias" (
    IdOcorrencias INTEGER CONSTRAINT pk_IdOcorrencias PRIMARY KEY,
    IdReparacaoTipo INTEGER NOT NULL ,
    Quantidade INTEGER NOT NULL ,
    PrecoUnitario TEXT NOT NULL
);
```

ANEXO D

```
package ipg.pt.oficinaonline;

import android.content.Context;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

import org.json.JSONException;
import org.json.JSONObject;

import java.io.UnsupportedEncodingException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.concurrent.ExecutionException;

import static java.sql.Types.NULL;

public class MainActivity extends AppCompatActivity {
    public static Cliente cliente;
    String[] s = new String[2];
    EditText mEmail,mPassword;
    String resul;
    Context context;
    CharSequence text;
    int duration;
    Toast toast;
    JSONObject oneObject;
    public final static String EXTRA_PNOME=
"ipg.pt.testescodigo.PNOME";
    public final static String EXTRA_IDCLIENTE =
"ipg.pt.testescodigo.IDCLIENTE";
    public final static String EXTRA_UNOME =
"ipg.pt.testescodigo.UNOME";
    public final static String EXTRA_EMAIL =
"ipg.pt.testescodigo.EMAIL";
    public final static String EXTRA_PASSWORD =
"ipg.pt.testescodigo.PASSWORD";
    public  static String pnome,unome,email,pass = null;
    public  static int id = 0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mEmail = (EditText) findViewById(R.id.editTextEmail);
        mPassword = (EditText) findViewById(R.id.editTextPassword);
        context = getApplicationContext();
        duration = Toast.LENGTH_SHORT;

    }

    public void BotaoLogin(View view){
        s[0] = mEmail.getText().toString();
        s[1] = mPassword.getText().toString();
        try {
            // Hash SHA256
            MessageDigest md = MessageDigest.getInstance("SHA-256");
```

```
            md.update(s[1].getBytes("UTF-8")); // Change this to
"UTF-16" if needed
            byte[] digest = md.digest();
            String hash = String.format("%064x", new
java.math.BigInteger(1, digest));
            s[1] = hash.toUpperCase();

            //TextView textViewHash = (TextView)
findViewById(R.id.textViewHash);
            //textViewHash.setText(hash);

            resul = new WebServiceLogin().execute(s).get();
            oneObject = new JSONObject(resul);
            pnome = oneObject.getString("primeiroNome");
            unome = oneObject.getString("ultimoNome");
            email = oneObject.getString("email");
            pass = oneObject.getString("password");
            id = oneObject.getInt("idCliente");
            cliente = new Cliente(id, pnome, unome, "", email, pass,
"", NULL);

            if(!pnome.equals("0")){
                if(id==1){
                    Intent intent = new Intent(this,
MenuCliente.class);
                    intent.putExtra(EXTRA_PNOME, pnome);
                    intent.putExtra(EXTRA_UNOME, unome);
                    intent.putExtra(EXTRA_IDCLIENTE, id);
                    intent.putExtra(EXTRA_EMAIL, email);
                    intent.putExtra(EXTRA_PASSWORD, pass);
                    startActivity(intent);
                }
            }else{
                text = "Login ou password errada.";
                toast = Toast.makeText(context, text, duration);
                toast.show();
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (ExecutionException e) {
            e.printStackTrace();
        }catch (JSONException e) {
            e.printStackTrace();
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        }
    }

}
```

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="ipg.pt.oficinaonline.MainActivity">


    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/loginTitulo"
        android:id="@+id/textViewLogin"
        android:textSize="20sp"
        android:textColor="@color/colorHeader"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/emailTitulo"
        android:id="@+id/editTextEmail"
        android:inputType="textEmailAddress"
        android:layout_below="@+id/textViewLogin"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="15dp" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/passwordTitulo"
        android:id="@+id/editTextPassword"
        android:inputType="textPassword"
        android:layout_below="@+id/editTextEmail"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="15dp" />
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editTextPassword"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="15dp"
        android:textSize="17sp"
        android:id="@+id/buttonLogin"
        android:text="@string/loginTitulo"
        android:onClick="BotaoLogin"/>

</RelativeLayout>
```

```
package ipg.pt.oficinaonline;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

public class MenuCliente extends AppCompatActivity {
    String pnome,unome;
    int id = 0;
    TextView campoUtilizador;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_menu_cliente);

        Intent intent = getIntent();
        pnome = intent.getStringExtra(MainActivity.EXTRA_PNOME);
        unome = intent.getStringExtra(MainActivity.EXTRA_UNOME);
        id = intent.getIntExtra(MainActivity.EXTRA_IDCLIENTE,0);

        campoUtilizador = (TextView)
findViewById(R.id.textViewMenuCliente);
        campoUtilizador.setText("Bem-vindo, "+pnome+" "+unome+".");

    }

    public void BotaoVerVeiculos(View view){
        Intent intent = new Intent(this, ClienteVerVeiculos.class);
        intent.putExtra(MainActivity.EXTRA_IDCLIENTE,id);
        startActivity(intent);
    }

    public void BotaoRegistarVeiculo(View view){
        Intent intent = new Intent(this, RegistarVeiculo.class);
        intent.putExtra(MainActivity.EXTRA_IDCLIENTE,id);
        startActivity(intent);
    }
}
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_menu_cliente"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="ipg.pt.oficinaonline.MenuCliente">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text=""
        android:id="@+id/textViewMenuCliente"
        android:textSize="20sp"
        android:textColor="@color/colorHeader"/>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textViewMenuCliente"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="15dp"
        android:textSize="17sp"
        android:id="@+id/buttonVerVeiculos"
        android:text="@string/ver_veiculos"
        android:onClick="BotaoVerVeiculos"/>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/buttonVerVeiculos"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="15dp"
        android:textSize="17sp"
        android:id="@+id/buttonRegistarVeiculo"
        android:text="@string/registar_veiculo"
        android:onClick="BotaoRegistarVeiculo"/>

</RelativeLayout>
```

```
package ipg.pt.oficinaonline;

import android.content.Context;
import android.content.Intent;
import android.database.sqlite.SQLiteDatabase;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.concurrent.ExecutionException;

import static java.sql.Types.NULL;

public class RegistarVeiculo extends AppCompatActivity implements
AdapterView.OnItemSelectedListener {
    TextView tv;
    String s, resul, resposta;
    EditText matricula;
    int id;
    Spinner spinner, spinner2;
    String[] param = new String[8];
    String[] nomes;
    String[] valores;
    String[] nomesMarcas;
    String[] valoresMarcas;
    CharSequence text;
    int duration;
    Toast toast;
    Context context;
    JSONObject oneObject;
    ArrayAdapter<String> adapter;

    private boolean isNetworkAvailable() {
        ConnectivityManager connectivityManager
                = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo activeNetworkInfo =
connectivityManager.getActiveNetworkInfo();
        return activeNetworkInfo != null &&
activeNetworkInfo.isConnected();
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_registar_veiculo);

        try {
            s = new WebServiceBuscarMarca().execute().get();
```

```
            JSONArray jArray = new JSONArray(s);
            nomes = new String[jArray.length()];
            valores = new String[jArray.length()];
            for (int i = 0; i < jArray.length(); i++) {
                oneObject = jArray.getJSONObject(i);
                nomes[i] = oneObject.getString("name");
                valores[i] = oneObject.getString("id");
            }
            //ViewGroup layout = (ViewGroup)
findViewById(R.id.activity_registar_veiculo);
            spinner = (Spinner)
findViewById(R.id.spinnerInserirMarca);
            // Create an ArrayAdapter using the string array and a
default spinner layout
            adapter = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, nomes);
            // Specify the layout to use when the list of choices
appears

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropd
own_item);
            // Apply the listAdapter to the spinner
            spinner.setAdapter(adapter);
            spinner.setOnItemSelectedListener(this);
            matricula = (EditText)
findViewById(R.id.editTextMatricula);
            id =
getIntent().getIntExtra(MainActivity.EXTRA_IDCLIENTE, 0);
            context = getApplicationContext();
            duration = Toast.LENGTH_LONG;
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (ExecutionException e) {
            e.printStackTrace();
        } catch (JSONException e) {
            e.printStackTrace();
        }


    }

    @Override
    public void onItemSelected(AdapterView<?> parent, View view,
                               int pos, long id) {
        // An item was selected. You can retrieve the selected item
using
        // parent.getItemAtPosition(pos)
        int p = Integer.parseInt(valores[pos]);

        try {
            s = new WebServiceBuscarModelo().execute(p).get();
            JSONArray jArray = new JSONArray(s);
            nomesMarcas = new String[jArray.length()];
            valoresMarcas = new String[jArray.length()];
            for (int i = 0; i < jArray.length(); i++) {
                oneObject = jArray.getJSONObject(i);
                nomesMarcas[i] = oneObject.getString("fipe_name");
                valoresMarcas[i] = oneObject.getString("id");
            }
            spinner2 = (Spinner)
findViewById(R.id.spinnerInserirModelo);
```

```
            // Create an ArrayAdapter using the string array and a
default spinner layout
            adapter = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, nomesMarcas);
            // Specify the layout to use when the list of choices
appears

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropd
own_item);
            // Apply the listAdapter to the spinner
            spinner2.setAdapter(adapter);

        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (ExecutionException e) {
            e.printStackTrace();
        } catch (JSONException e) {
            e.printStackTrace();
        }

    }

    @Override
    public void onNothingSelected(AdapterView<?> adapterView) {

    }

    public void BotaoRegistarVeiculo(View view) {

        // Inserir vaículo no Web Service
        param[0] = valores[(int) spinner.getSelectedItemId()]; //Id
Marca
        param[1] = valoresMarcas[(int) spinner2.getSelectedItemId()];
//Id Modelo
        param[2] = spinner.getSelectedItem() + "";//Marca
        param[3] = spinner2.getSelectedItem() + "";//Modelo
        param[4] = matricula.getText().toString();//Matricula
        param[5] = id + "";//Id_cliente
        param[6] = MainActivity.email;//email
        param[7] = MainActivity.pass;//password

        try {
            if (isNetworkAvailable()) {
                resul = new
WebServiceRegistarVeiculo().execute(param).get();
                oneObject = new JSONObject(resul);
                resposta = oneObject.getString("Resposta");
                text = resposta;
                toast = Toast.makeText(context, text + " (back-end)",
duration);
                toast.show();
            } else {
                // Inserir veículo no SQLite
                Veiculo veiculo = new Veiculo(NULL,
Long.parseLong(param[0]), Long.parseLong(param[1]),
MainActivity.cliente.getId(), param[4], param[2], param[3]);
                SQLiteDatabase bd = new
DatabaseHelper(this).getWritableDatabase();
                VeiculoRepository veiculo_repository = new
VeiculoRepository(bd);
                veiculo_repository.insert(veiculo);
```

```
                toast = Toast.makeText(context, "Veículo inserido na
BD do dispositivo.", duration);
                toast.show();
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

}
```

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_registar_veiculo"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="ipg.pt.oficinaonline.RegistarVeiculo">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/registar_veiculo"
        android:textSize="20dp"
        android:textColor="@color/colorHeader"
        android:id="@+id/textViewRegistarVeiculo" />
    <Spinner
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/spinnerInserirMarca"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_below="@+id/textViewRegistarVeiculo"
        android:layout_marginTop="15dp"
        >

    </Spinner>
    <Spinner
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/spinnerInserirModelo"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_below="@+id/spinnerInserirMarca"
        android:layout_marginTop="15dp"
        >
    </Spinner>
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/editTextMatricula"
        android:hint="@string/matricula"
        android:layout_below="@+id/spinnerInserirModelo"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="21dp"
        android:inputType="text"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editTextMatricula"
        android:text="@string/registar_veiculo"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="11dp"
        android:onClick="BotaoRegistarVeiculo"/>
</RelativeLayout>
```

```
package ipg.pt.oficinaonline;

import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;
import android.widget.Spinner;
import android.widget.Toast;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.ArrayList;
import java.util.concurrent.ExecutionException;

public class ClienteVerVeiculos extends AppCompatActivity {
    private static final String TAG =
ClienteRepository.class.getName();
    Cursor cursorVeiculos;
    ListView listView;
    ArrayAdapter<String> adapter;
    ArrayList<String> listItems = new ArrayList<String>();
    String resul, s;
    String[] param = new String[3];
    JSONArray jArray;
    JSONObject oneObject;

    private boolean isNetworkAvailable() {
        ConnectivityManager connectivityManager
                = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo activeNetworkInfo =
connectivityManager.getActiveNetworkInfo();
        return activeNetworkInfo != null &&
activeNetworkInfo.isConnected();
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_cliente_ver_veiculos);
        try {

            param[0] = MainActivity.id + "";
            param[1] = MainActivity.email;
            param[2] = MainActivity.pass;

            if (isNetworkAvailable()) {
                // Online - ler dados do Web Service
                resul = new
WebServiceBuscarVeiculos().execute(param).get();
                jArray = new JSONArray(resul);
```

```
                for (int i = 0; i < jArray.length(); i++) {
                    oneObject = jArray.getJSONObject(i);
                    s = "Marca: " + oneObject.getString("Marca") +
".\nModelo: " + oneObject.getString("Modelo")
                            + ".\nMatricula: " +
oneObject.getString("Matricula") + ".\nConsumo total: " +
oneObject.getString("Litros") + ".";
                    listItems.add(s);
                }
            } else {
                // Offline - ler dados do SQLite
                DatabaseHelper dbHelper = new DatabaseHelper(this);
                SQLiteDatabase db = dbHelper.getReadableDatabase();
                final VeiculoRepository veiculoRepository = new
VeiculoRepository(db);
                cursorVeiculos =
veiculoRepository.getAllItemsFiltered(Long.toString(MainActivity.clie
nte.getId()));
                while (cursorVeiculos.moveToNext()) {
                    s = "Marca: " +
cursorVeiculos.getString(cursorVeiculos.getColumnIndex("Marca")) +
".\n"
                            + "Modelo: " +
cursorVeiculos.getString(cursorVeiculos.getColumnIndex("Modelo")) +
".\n"
                            + "Matricula: " +
cursorVeiculos.getString(cursorVeiculos.getColumnIndex("Matricula"));
                    listItems.add(s);
                }
                cursorVeiculos.close();
                db.close();
            }

            // Mostrar na ListView
            listView = (ListView) findViewById(R.id.list);
            adapter = new ArrayAdapter<String>(this,
                    android.R.layout.simple_list_item_1,
                    listItems);
            listView.setAdapter(adapter);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_cliente_ver_veiculos"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="ipg.pt.oficinaonline.ClienteVerVeiculos">

    <TextView style="?android:textAppearanceMedium"
        android:padding="16dp"
        android:lineSpacingMultiplier="1.2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textViewVeiculos"
        android:textColor="@color/colorHeader"
        android:textSize="22sp"
        android:text="@string/veiculos_cliente" />

    <ListView
        android:id="@+id/list"
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:layout_below="@+id/textViewVeiculos"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="15dp">
    </ListView>

</RelativeLayout>
```

```
package ipg.pt.oficinaonline;

import android.os.AsyncTask;
import android.widget.TextView;

import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Scanner;

/**
 * Created by vasco on 11/01/2017.
 */

public class WebServiceBuscarMarca extends AsyncTask<Void,
Void,String> {


    @Override
    protected String doInBackground(Void... voids) {
        try {
            URL url = new
URL("http://bd.ipg.pt:8080/alunos/bda_1010985.WS_BUSCA_MARCAS");
            HttpURLConnection con = (HttpURLConnection)
url.openConnection();
            InputStream in = con.getInputStream();
            Scanner scanner = new Scanner(in);
            scanner.useDelimiter("\\A"); // read all the file
            if (scanner.hasNext()) {
                return scanner.next();
            }

        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }


}
```

```
package ipg.pt.oficinaonline;

import android.os.AsyncTask;

import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Scanner;

/**
 * Created by vasco on 12/01/2017.
 */

public class WebServiceBuscarModelo extends AsyncTask<Integer,
Void,String> {


    @Override
    protected String doInBackground(Integer... integers) {
        try {
            URL url = new
URL("http://bd.ipg.pt:8080/alunos/bda_1010985.WS_BUSCA_MODELOS?id_mod
elo="+integers[0]);
            HttpURLConnection con = (HttpURLConnection)
url.openConnection();
            InputStream in = con.getInputStream();
            Scanner scanner = new Scanner(in);
            scanner.useDelimiter("\\A"); // read all the file
            if (scanner.hasNext()) {
                return scanner.next();
            }

        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

```
package ipg.pt.oficinaonline;

import android.os.AsyncTask;

import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLEncoder;
import java.util.Scanner;

/**
 * Created by vasco on 01/02/2017.
 */

public class WebServiceBuscarVeiculos extends AsyncTask<String[],
Void,String> {
    @Override
    protected String doInBackground(String[]... strings) {
        String[] s = strings[0];
        HttpURLConnection con = null;
        try {
            URL url = new
URL("http://bd.ipg.pt:8080/alunos/bda_1010985.WS.WS_VERVEICULOS" +
                    "?id_cliente=" + s[0] + "&v_email=" + s[1]
+"&v_password=" +s[2]);
            con = (HttpURLConnection) url.openConnection();
            InputStream in = con.getInputStream();
            Scanner scanner = new Scanner(in);
            scanner.useDelimiter("\\A"); // read all the file
            if (scanner.hasNext()) {
                return scanner.next();

            }

        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            con.disconnect();
        }
        return null;
    }
}
```

```
package ipg.pt.oficinaonline;

import android.os.AsyncTask;

import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Scanner;

/**
 * Created by vasco on 27/01/2017.
 */

public class WebServiceLogin  extends AsyncTask<String[],
Void,String> {
    @Override
    protected String doInBackground(String[]... strings) {
        String[] s = strings[0];
        HttpURLConnection con = null;
        try {
            URL url = new
URL("http://bd.ipg.pt:8080/alunos/bda_1010985.WS.ws_login?vemail=" +
s[0] + "&vpassword=" + s[1]);
            con = (HttpURLConnection) url.openConnection();
            InputStream in = con.getInputStream();
            Scanner scanner = new Scanner(in);
            scanner.useDelimiter("\\A"); // read all the file
            if (scanner.hasNext()) {
                return scanner.next();

            }

        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            con.disconnect();
        }
        return null;
    }
}
```

```
package ipg.pt.oficinaonline;

import android.os.AsyncTask;

import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLEncoder;
import java.util.Scanner;

/**
 * Created by vasco on 30/01/2017.
 */

public class WebServiceRegistarVeiculo extends AsyncTask<String[],
Void,String> {
    @Override
    protected String doInBackground(String[]... strings) {
        String[] s = strings[0];
        HttpURLConnection con = null;
        try {
            URL url = new
URL("http://bd.ipg.pt:8080/alunos/bda_1010985.WS.WS_RegistaVeiculo" +
                    "?v_idmarca=" + s[0] + "&v_idmodelo=" + s[1]+
                    "&v_marca=" + URLEncoder.encode(s[2],"utf-8") +
"&v_modelo=" + URLEncoder.encode(s[3],"utf-8")+
                    "&v_matricula=" + s[4] + "&v_idcliente=" + s[5] +
                    "&v_email=" + s[6] + "&v_password=" + s[7]);
            con = (HttpURLConnection) url.openConnection();
            InputStream in = con.getInputStream();
            Scanner scanner = new Scanner(in);
            scanner.useDelimiter("\\A"); // read all the file
            if (scanner.hasNext()) {
                return scanner.next();

            }

        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            con.disconnect();
        }
        return null;
    }
}
```

```
package ipg.pt.oficinaonline;

/**
 * Created by Sandro on 02/02/2017.
 * Cliente - Entity - Data model - POJO model
 */

public class Cliente {

    private long Id;
    private String PrimeiroNome;
    private String UltimoNome;
    private String Telemovel;
    private String Email;
    private String Password;
    private String Genero;
    private int Sync;

    public Cliente(long Id, String PrimeiroNome, String UltimoNome,
String Telemovel, String Email, String Password, String Genero, int
Sync) {
        this.Id = Id;
        this.PrimeiroNome = PrimeiroNome;
        this.UltimoNome = UltimoNome;
        this.Telemovel = Telemovel;
        this.Email = Email;
        this.Password = Password;
        this.Genero = Genero;
        this.Sync = Sync;
    }

    public long getId() {
        return Id;
    }
    public String getPrimeiroNome() {
        return PrimeiroNome;
    }
    public String getUltimoNome() {
        return UltimoNome;
    }
    public String getTelemovel() {
        return Telemovel;
    }
    public String getEmail() {
        return Email;
    }
    public String getPassword() {
        return Password;
    }
    public String getGenero() {
        return Genero;
    }
    public int getSync() {
        return Sync;
    }

}
```

```
package ipg.pt.oficinaonline;

import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.provider.BaseColumns;
import android.util.Log;

import java.sql.SQLException;

/**
 * Created by CirculesX on 06/02/2017.
 */

public class ClienteRepository implements BaseColumns {

    private static final String TAG =
ClienteRepository.class.getName();
    private static final String TABLE_NAME = "bd_clientes";
    private static final String FIELD_PRIMEIRO_NOME = "PrimeiroNome";
    private static final String FIELD_ULTIMO_NOME = "UltimoNome";
    private static final String FIELD_TELEMOVEL = "Telemovel";
    private static final String FIELD_EMAIL = "Email";
    private static final String FIELD_PASSWORD = "Password";
    private static final String FIELD_GENERO = "Genero";
    private static final String FIELD_SYNC = "Sync";
    private SQLiteDatabase db;

    public ClienteRepository(SQLiteDatabase db) {
        this.db = db;
    }

    public void create() throws SQLException {
        db.execSQL("CREATE TABLE " + TABLE_NAME + "(" +
                _ID + " INTEGER PRIMARY KEY AUTOINCREMENT," +
                FIELD_PRIMEIRO_NOME + " TEXT NOT NULL," +
                FIELD_ULTIMO_NOME + " TEXT NOT NULL," +
                FIELD_TELEMOVEL + " TEXT NOT NULL," +
                FIELD_EMAIL + " TEXT NOT NULL," +
                FIELD_PASSWORD + " TEXT NOT NULL," +
                FIELD_GENERO + " TEXT," +
                FIELD_SYNC + " INTEGER" +
                ");"
        );

        if (Globals.DUMMY_DATA) {
            // gravar 5 registros na tabela
            String PrimeiroNome, UltimoNome, Telemovel, Email,
Password, Genero, Sync;
            for (int i=1; i<=5; i++) {
                PrimeiroNome = "Mecânico";
                UltimoNome = "n. " + i;
                Telemovel = "Telemovel " + i;
                Email = "mecanico" + i + "@mecman.pt";
                Password = "segredo";
                Genero = "m";
                Sync = "0";
                db.execSQL("INSERT INTO "
                        + TABLE_NAME
                        + " ("
                        + FIELD_PRIMEIRO_NOME + ", "
```

```
                        + FIELD_ULTIMO_NOME + ", "
                        + FIELD_TELEMOVEL + ", "
                        + FIELD_EMAIL + ", "
                        + FIELD_PASSWORD + ", "
                        + FIELD_GENERO + ", "
                        + FIELD_SYNC
                        + ") VALUES ("
                        + "'" + PrimeiroNome + "', "
                        + "'" + UltimoNome + "', "
                        + "'" + Telemovel + "', "
                        + "'" + Email + "', "
                        + "'" + Password + "', "
                        + "'" + Genero + "', "
                        + "'" + Sync + "'"
                        + ");"
                );
            }
        }

    }

    public long insert(Cliente cliente) {
        ContentValues contentValues = new ContentValues();
        contentValues.put(FIELD_PRIMEIRO_NOME,
cliente.getPrimeiroNome());
        contentValues.put(FIELD_ULTIMO_NOME,
cliente.getUltimoNome());
        contentValues.put(FIELD_TELEMOVEL, cliente.getTelemovel());
        contentValues.put(FIELD_EMAIL, cliente.getEmail());
        contentValues.put(FIELD_PASSWORD, cliente.getPassword());
        contentValues.put(FIELD_GENERO, cliente.getGenero());
        contentValues.put(FIELD_SYNC, cliente.getSync());
        return db.insertOrThrow(TABLE_NAME, null, contentValues);
    }

    public Cursor getAllItems() {
        String [] dbColumns = new String[] {
                _ID,
                FIELD_PRIMEIRO_NOME,
                FIELD_ULTIMO_NOME,
                FIELD_TELEMOVEL,
                FIELD_EMAIL,
                FIELD_PASSWORD,
                FIELD_GENERO,
                FIELD_SYNC
        };
        return db.query(TABLE_NAME, dbColumns, null, null, null,
null, FIELD_PRIMEIRO_NOME + " COLLATE NOCASE");
    }

    public Cursor getAllItemsFiltered(String filterString) {
        String sql = "SELECT "+_ID+", "+FIELD_PRIMEIRO_NOME+",
"+FIELD_ULTIMO_NOME
                + " FROM " + TABLE_NAME
                + " WHERE "+FIELD_PRIMEIRO_NOME+" LIKE ?";
        Log.d(TAG, "sql -> "+sql);
        return db.rawQuery(
                sql,
                new String[] {
                        '%'+filterString+'%'
                }
```

```
        );
    }

    public Cliente getItem(long Id) {
        String [] dbColumns = new String[] {
                _ID,
                FIELD_PRIMEIRO_NOME,
                FIELD_ULTIMO_NOME,
                FIELD_TELEMOVEL,
                FIELD_EMAIL,
                FIELD_PASSWORD,
                FIELD_GENERO,
                FIELD_SYNC
        };
        Cursor cursor = db.query(
                TABLE_NAME,
                dbColumns,
                _ID+"=?",
                new String[] {String.valueOf(Id)},
                null, null, null
        );
        if (!cursor.moveToFirst()) {
            return null;
        }
        String PrimeiroNome =
cursor.getString(cursor.getColumnIndex(FIELD_PRIMEIRO_NOME));
        String UltimoNome =
cursor.getString(cursor.getColumnIndex(FIELD_ULTIMO_NOME));
        String Telemovel =
cursor.getString(cursor.getColumnIndex(FIELD_TELEMOVEL));
        String Email =
cursor.getString(cursor.getColumnIndex(FIELD_EMAIL));
        String Password =
cursor.getString(cursor.getColumnIndex(FIELD_PASSWORD));
        String Genero =
cursor.getString(cursor.getColumnIndex(FIELD_GENERO));
        int Sync = cursor.getInt(cursor.getColumnIndex(FIELD_SYNC));
        return new Cliente(
                Id,
                PrimeiroNome,
                UltimoNome,
                Telemovel,
                Email,
                Password,
                Genero,
                Sync
        );
    }

    public void update(Cliente cliente) {
        ContentValues contentValues = new ContentValues();
        contentValues.put(FIELD_PRIMEIRO_NOME,
cliente.getPrimeiroNome());
        contentValues.put(FIELD_ULTIMO_NOME,
cliente.getUltimoNome());
        contentValues.put(FIELD_TELEMOVEL, cliente.getTelemovel());
        contentValues.put(FIELD_EMAIL, cliente.getEmail());
        contentValues.put(FIELD_PASSWORD, cliente.getPassword());
        contentValues.put(FIELD_GENERO, cliente.getGenero());
        contentValues.put(FIELD_SYNC, cliente.getSync());
```

```
        db.update(TABLE_NAME, contentValues, _ID+"=?", new String[]
{Long.toString(cliente.getId())});
    }

    public long delete(Cliente cliente) {
        return db.delete(TABLE_NAME, _ID+"=?", new String[]
{String.valueOf(cliente.getId())});
    }

    public void drop() {
        db.execSQL("DROP TABLE IF EXISTS "+TABLE_NAME+";");
    }

}
```

```
package ipg.pt.oficinaonline;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

import java.sql.SQLException;

/**
 * Created by Sandro on 01/02/2017.
 */

public class DatabaseHelper extends SQLiteOpenHelper {

    public static final String TAG = DatabaseHelper.class.getName();
    private static final String DB_NAME = "oficina.db";
    private static final int DB_VERSION = 1;

    public DatabaseHelper(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        ClienteRepository cliente_repository = new
ClienteRepository(db);
        VeiculoRepository veiculo_repository = new
VeiculoRepository(db);
        try {
            cliente_repository.create();
            veiculo_repository.create();
        } catch (SQLException e) {
            if (BuildConfig.DEBUG) {
                Log.e(TAG, e.getMessage());
            }
        }
    }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int
i1) {

    }
}
```

```
package ipg.pt.oficinaonline;

/**
 * Created by Sandro on 06/02/2017.
 * Vaiculo - Entity - Data model - POJO model
 */

public class Veiculo {

    private long Id;
    private long IdMarca;
    private long IdModelo;
    private long IdCliente;
    private String Matricula;
    private String Marca;
    private String Modelo;

    public Veiculo(long Id, long IdMarca, long IdModelo, long
IdCliente, String Matricula, String Marca, String Modelo) {
        this.Id = Id;
        this.IdMarca = IdMarca;
        this.IdModelo = IdModelo;
        this.IdCliente = IdCliente;
        this.Matricula = Matricula;
        this.Marca = Marca;
        this.Modelo = Modelo;
    }

    public long getId() {
        return Id;
    }

    public long getIdMarca() {
        return IdMarca;
    }

    public long getIdModelo() {
        return IdModelo;
    }

    public long getIdCliente() {
        return IdCliente;
    }

    public String getMatricula() {
        return Matricula;
    }

    public String getMarca() {
        return Marca;
    }

    public String getModelo() {
        return Modelo;
    }

}
```

```
package ipg.pt.oficinaonline;

import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.provider.BaseColumns;
import android.util.Log;

import java.sql.SQLException;

/**
 * Created by CirculesX on 06/02/2017.
 */

public class VeiculoRepository implements BaseColumns {

    private static final String TAG =
VeiculoRepository.class.getName();
    private static final String TABLE_NAME = "bd_veiculos";
    private static final String FIELD_ID_MARCA = "IdMarca";
    private static final String FIELD_ID_MODELO = "IdModelo";
    private static final String FIELD_ID_CLIENTE = "IdCliente";
    private static final String FIELD_MATRICULA = "Matricula";
    private static final String FIELD_MARCA = "Marca";
    private static final String FIELD_MODELO = "Modelo";
    private SQLiteDatabase db;

    public VeiculoRepository(SQLiteDatabase db) {
        this.db = db;
    }

    public void create() throws SQLException {
        db.execSQL("CREATE TABLE " + TABLE_NAME + "(" +
                _ID + " INTEGER PRIMARY KEY AUTOINCREMENT," +
                FIELD_ID_MARCA + " INTEGER NOT NULL," +
                FIELD_ID_MODELO + " INTEGER NOT NULL," +
                FIELD_ID_CLIENTE + " INTEGER NOT NULL," +
                FIELD_MATRICULA + " TEXT NOT NULL," +
                FIELD_MARCA + " TEXT NOT NULL," +
                FIELD_MODELO + " TEXT NOT NULL" +
                ");"
        );

        if (Globals.DUMMY_DATA) {
            // gravar 5 registros na tabela
            String IdMarca, IdModelo, IdCliente, Matricula, Marca,
Modelo;
            for (int i=1; i<=5; i++) {
                IdMarca = "" + i;
                IdModelo = "" + i;
                IdCliente = "" + i;
                Matricula = "AA-BB-" + i;
                Marca = "Marca " + i;
                Modelo = "Modelo " + i;
                db.execSQL("INSERT INTO "
                        + TABLE_NAME
                        + " ("
                        + FIELD_ID_MARCA + ", "
                        + FIELD_ID_MODELO + ", "
                        + FIELD_ID_CLIENTE + ", "
                        + FIELD_MATRICULA + ", "
```

```
                        + FIELD_MARCA + ", "
                        + FIELD_MODELO
                        + ") VALUES ("
                        + "'" + IdMarca + "', "
                        + "'" + IdModelo + "', "
                        + "'" + IdCliente + "', "
                        + "'" + Matricula + "', "
                        + "'" + Marca + "', "
                        + "'" + Modelo + "'"
                        + ");"
                );
            }
        }
    }

    public long insert(Veiculo veiculo) {
        ContentValues contentValues = new ContentValues();
        contentValues.put(FIELD_ID_MARCA, veiculo.getIdMarca());
        contentValues.put(FIELD_ID_MODELO, veiculo.getIdModelo());
        contentValues.put(FIELD_ID_CLIENTE, veiculo.getIdCliente());
        contentValues.put(FIELD_MATRICULA, veiculo.getMatricula());
        contentValues.put(FIELD_MARCA, veiculo.getMarca());
        contentValues.put(FIELD_MODELO, veiculo.getModelo());
        return db.insertOrThrow(TABLE_NAME, null, contentValues);
    }

    public Cursor getAllItems() {
        String [] dbColumns = new String[] {
                _ID,
                FIELD_ID_MARCA,
                FIELD_ID_MODELO,
                FIELD_ID_CLIENTE,
                FIELD_MATRICULA,
                FIELD_MARCA,
                FIELD_MODELO
        };
        return db.query(TABLE_NAME, dbColumns, null, null, null,
null, FIELD_MATRICULA + " COLLATE NOCASE");
    }

    /**
     * Veículos de um determinado cliente
     * @param filterString ID do cliente
     * @return Cursor Lista de Veículos de um cliente
     */
    public Cursor getAllItemsFiltered(String filterString) {
        String sql = "SELECT "
                + _ID + ", "
                + FIELD_MARCA + ", "
                + FIELD_MODELO + ", "
                + FIELD_MATRICULA + " "
                + "FROM " + TABLE_NAME + " "
                + "WHERE " + FIELD_ID_CLIENTE + " LIKE ?";
        Log.d(TAG, "sql -> "+sql);
        return db.rawQuery(
                sql,
                new String[] {
                        '%'+filterString+'%'
                }
        );
    }
```

```
    public Veiculo getItem(long Id) {
        String [] dbColumns = new String[] {
                _ID,
                FIELD_ID_MARCA,
                FIELD_ID_MODELO,
                FIELD_ID_CLIENTE,
                FIELD_MATRICULA,
                FIELD_MARCA,
                FIELD_MODELO
        };
        Cursor cursor = db.query(
                TABLE_NAME,
                dbColumns,
                _ID+"=?",
                new String[] {String.valueOf(Id)},
                null, null, null
        );
        if (!cursor.moveToFirst()) {
            return null;
        }
        int IdMarca =
cursor.getInt(cursor.getColumnIndex(FIELD_ID_MARCA));
        int IdModelo =
cursor.getInt(cursor.getColumnIndex(FIELD_ID_MODELO));
        int IdCliente =
cursor.getInt(cursor.getColumnIndex(FIELD_ID_CLIENTE));
        String Matricula =
cursor.getString(cursor.getColumnIndex(FIELD_MATRICULA));
        String Marca =
cursor.getString(cursor.getColumnIndex(FIELD_MARCA));
        String Modelo =
cursor.getString(cursor.getColumnIndex(FIELD_MODELO));
        return new Veiculo(
                Id,
                IdMarca,
                IdModelo,
                IdCliente,
                Matricula,
                Marca,
                Modelo
        );
    }
    public void update(Veiculo veiculo) {
        ContentValues contentValues = new ContentValues();
        contentValues.put(FIELD_ID_MARCA, veiculo.getIdMarca());
        contentValues.put(FIELD_ID_MODELO, veiculo.getIdModelo());
        contentValues.put(FIELD_ID_CLIENTE, veiculo.getIdCliente());
        contentValues.put(FIELD_MATRICULA, veiculo.getMatricula());
        contentValues.put(FIELD_MARCA, veiculo.getMarca());
        contentValues.put(FIELD_MODELO, veiculo.getModelo());
        db.update(TABLE_NAME, contentValues, _ID+"=?", new String[]
{Long.toString(veiculo.getId())});
    }
    public long delete(Veiculo veiculo) {
        return db.delete(TABLE_NAME, _ID+"=?", new String[]
{String.valueOf(veiculo.getId())});
    }
    public void drop() {
        db.execSQL("DROP TABLE IF EXISTS "+TABLE_NAME+";");
    }
}
```

```
package ipg.pt.oficinaonline;

/**
 * Created by Sandro on 01/02/2017.
 */

public final class Globals {

    private Globals() {
        throw new AssertionError();
    }

    public static final boolean DUMMY_DATA = true;

}
```

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="ipg.pt.oficinaonline">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".InserirUtilizador" />
        <activity android:name=".RegistarVeiculo" />
        <activity android:name=".InserirOcorrencia" />
        <activity android:name=".VerVeiculos" />
        <activity android:name=".MenuCliente" />
        <activity android:name=".ClienteVerVeiculos"></activity>
    </application>

</manifest>
```