



Departamento de Engenharia de Electrónica
e Telecomunicações e de Computadores
Lic. Engenharia Informática e Computadores

Sistemas de Informação I

Relatório Trabalho Prático
Parte 2 – Fase 2

Semestre Inverno
2009/2010

Elaborado por:

Grupo 1

30896 – Ricardo Canto

31401 – Nuno Cancelo

33595 – Nuno Sousa

A/C: Eng. Nuno Datia
ISEL, 19 de Fevereiro de 2010

Índice

Introdução.....	3
Etapa 5 – Modelo Físico: Vistas.....	4
Etapa 6 – Aplicação Java	6
Listagem de Pacientes.....	6
Ficha de diagnóstico de Paciente	7
Adicionar Paciente	8
Adicionar Terapeuta	9
Registo de Terapia	9
Nova Consulta.....	10
Remoção de Habilitação de Terapeuta.....	11
Suspensão de Terapeuta.....	12
Conclusão	13
Anexos	14
Listagem de rotinas de acesso a BD	14
Abertura da base de dados	14
Fecho de base de dados.....	14
Inserção de Paciente	14
Inserção de Terapeuta	15
Inserção de Consulta.....	16
Remoção de Paciente	16
Listagem de Terapeutas	17
Listagem de Pacientes.....	17
Apresentação de ficha de diagnóstico.....	17
Listagem de consultas.....	18
Função de Suporte setQuery.....	18
Suspensão de Terapeuta.....	19
Instruções de Execução.....	20

Introdução

A 2ª Parte, Fase II do trabalho tem como objectivo a criação de vistas de forma a permitirem implementar algumas funcionalidades. Em paralelo é necessário construir uma aplicação em Java, utilizando JDBC, via ODBC.

Etapa 5 – Modelo Físico: Vistas

O objectivo desta etapa é a criação de vistas que facilitem a concretização de algumas funcionalidades da aplicação, descritas na Etapa 6.

```
/*a) Listar todos os pacientes do centro, contemplando
a informação respeitante apenas aos dados pessoais;*/
CREATE VIEW LISTA_PACIENTES
    (Nome, BI, NIF, Morada, Telefone, Idade,
     "Data Nascimento", Profissao, "Estado Civil", "Decl.
Responsabilidade",
     Email, "Data Registo")
AS SELECT PESSOA.nomePessoa, PESSOA.BI, PESSOA.NIF,
    PESSOA.morada, PESSOA.telefone, PESSOA.idade
    , PESSOA.dataNasc, PACIENTE.profissao,
    PACIENTE.estadoCivil, PACIENTE.declRespon, PACIENTE.email,
    PACIENTE.dataRegisto
FROM PESSOA INNER JOIN PACIENTE ON (PESSOA.BI = PACIENTE.BI)

/*b) Apresentar a ficha de diagnóstico diferencial para um determinado
paciente;*/
CREATE VIEW DIAGNOSTICO_PACIENTES
    ("Nome Paciente", "BI Paciente", "NIF Paciente", Morada, Telefone,
Idade,
    "Data Nascimento", Profissao, "Estado Civil", "Decl.
Responsabilidade",
    Email, "Data Registo", "Nome Sintoma", "Desc. Sintoma", "Data
Sintoma",
    Notas, "Nome Padrao", "Desc. Padrao", "Nome Terapeuta" )
AS SELECT P.nomePessoa, P.BI, P.NIF,
    P.morada, P.telefone, P.idade
    , P.dataNasc, PACIENTE.profissao,
    PACIENTE.estadoCivil, PACIENTE.declRespon, PACIENTE.email,
    PACIENTE.dataRegisto, SINTOMA.nomeSintoma, SINTOMA.descSintoma,
    SINTOMA_PACIENTE.dataSintoma, SINTOMA_PACIENTE.notas,
    PADRAO.nomePadrao, PADRAO.descPadrao, T.nomePessoa
FROM PESSOA as P INNER JOIN PACIENTE ON (P.BI = PACIENTE.BI)
INNER JOIN SINTOMA_PACIENTE ON (PACIENTE.BI =
SINTOMA_PACIENTE.BIPaciente)
INNER JOIN SINTOMA ON (SINTOMA.numSintoma =
SINTOMA_PACIENTE.numSintoma)
INNER JOIN PACIENTE_SINTOMA_TERAPEUTA ON
(PACIENTE_SINTOMA_TERAPEUTA.BIPaciente = PACIENTE.BI)
INNER JOIN PADRAO ON (PADRAO.numPadrao =
PACIENTE_SINTOMA_TERAPEUTA.numPadrao)
INNER JOIN TERAPEUTA ON (TERAPEUTA.BI =
PACIENTE_SINTOMA_TERAPEUTA.BITerapeuta)
INNER JOIN PESSOA as T ON (T.BI = TERAPEUTA.BI )

CREATE VIEW LISTA_CONSULTAS
    ("BI Paciente", "Nome Paciente", "BI Terapeuta", "Nome Terapeuta",
    "Num Consulta", "Data Consulta", "Relatorio")
AS SELECT P.BI, P.nomePessoa, T.BI, T.nomePessoa,
    CONSULTA.numConsulta, CONSULTA.dataConsulta, CONSULTA.relatorio
FROM PESSOA as P INNER JOIN PACIENTE ON (P.BI = PACIENTE.BI)
INNER JOIN CONSULTA ON (P.BI = CONSULTA.BIPaciente)
INNER JOIN PESSOA as T ON (T.BI = CONSULTA.BITerapeuta)
INNER JOIN TERAPEUTA ON (T.BI = TERAPEUTA.BI)
```

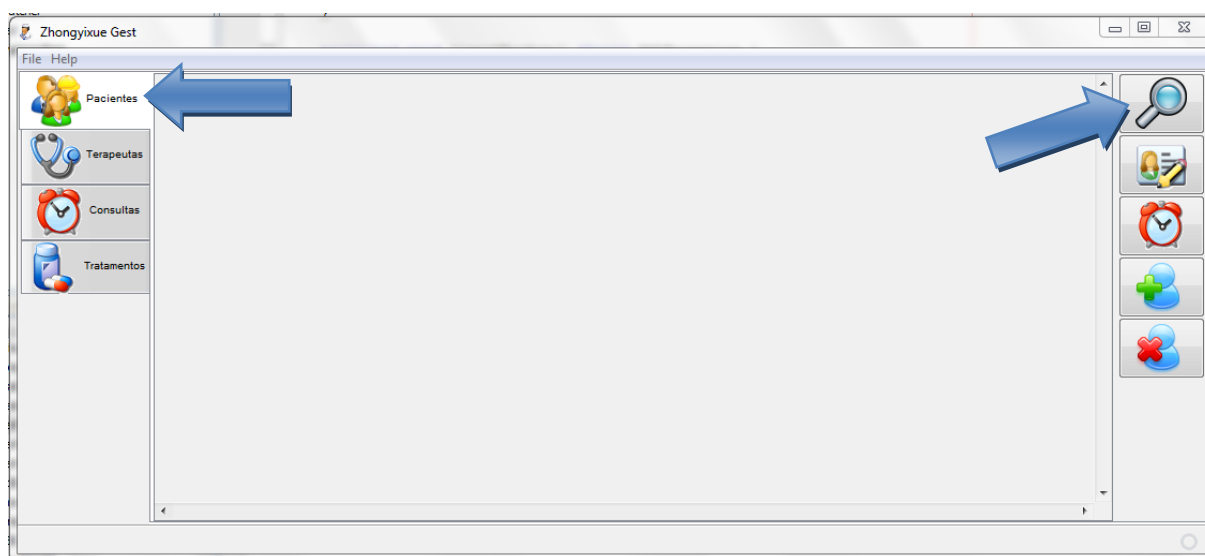
```
CREATE VIEW LISTA_TERAPEUTAS
  (Nome, BI, NIF, Morada, Telefone, Idade,
   "Data Nascimento", "Data Conclusao", "Em Funcoes")
AS SELECT PESSOA.nomePessoa, PESSOA.BI, PESSOA.NIF,
  PESSOA.morada, PESSOA.telefone, PESSOA.idade
  , PESSOA.dataNasc, TERAPEUTA.dataConclusao,
  TERAPEUTA.emFuncoes
FROM PESSOA INNER JOIN TERAPEUTA ON (PESSOA.BI = TERAPEUTA.BI)
```

Etapa 6 – Aplicação Java

Nesta etapa pretende-se que seja implementada uma aplicação que utilize o sistema de informação construído. A aplicação será realizada na linguagem Java, utilizando JDBC (*Java DataBase Connectivity*), via ODBC, para ligação, acesso e comunicação com o servidor de base de dados.

Listagem de Pacientes

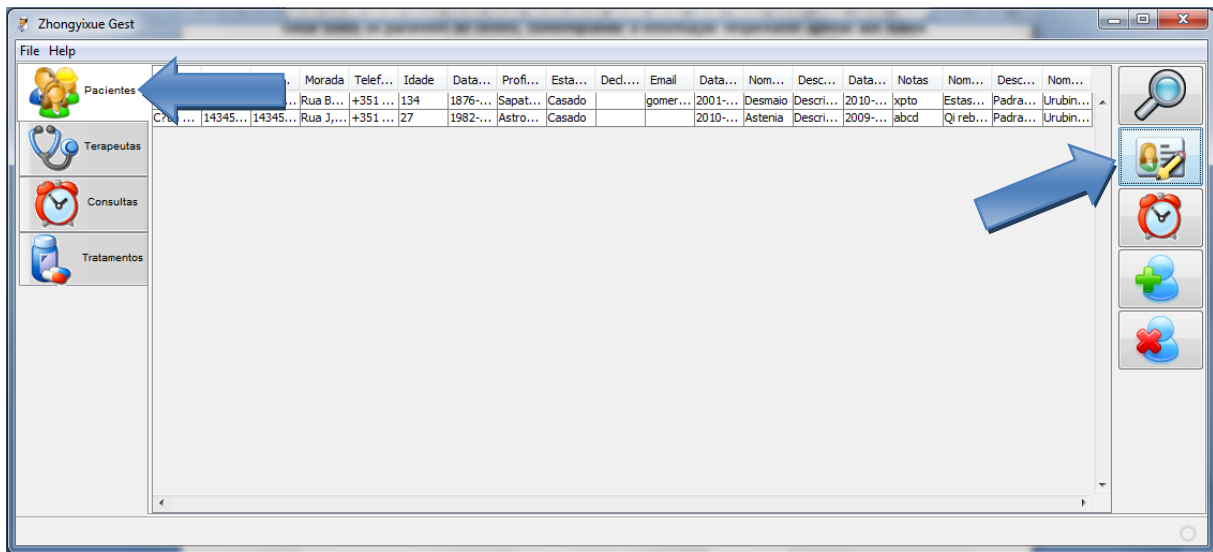
Listar todos os pacientes do centro, contemplando a informação respeitante apenas aos dados pessoais.



Para que esta informação seja apresentada, seleccione o separador “Pacientes” e carregue no botão com a lupa.

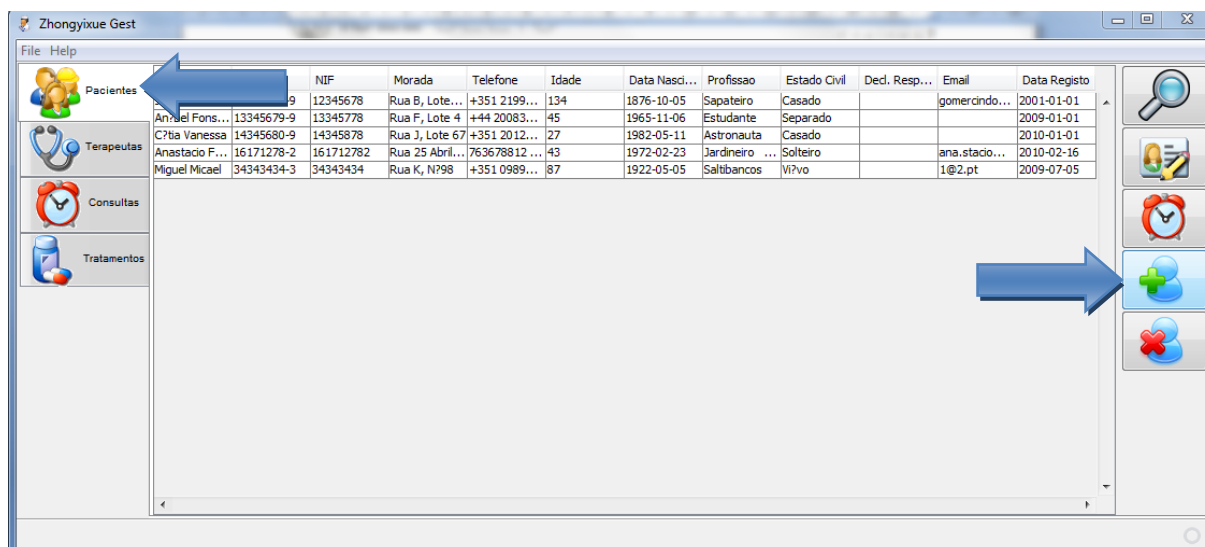
Ficha de diagnóstico de Paciente

Apresentar a ficha de diagnóstico diferencial para um determinado paciente;



Para que esta informação seja apresentada, seleccione o separador “Pacientes” e carregue no botão com a ficha de cliente. Nota: Por motivos de tempo não foi possível apresentar a informação em específico para um paciente.

Adicionar Paciente



Para adicionar um paciente carregue no botão Adicionar. Serão colocadas várias questões para preenchimento de dados.

Input

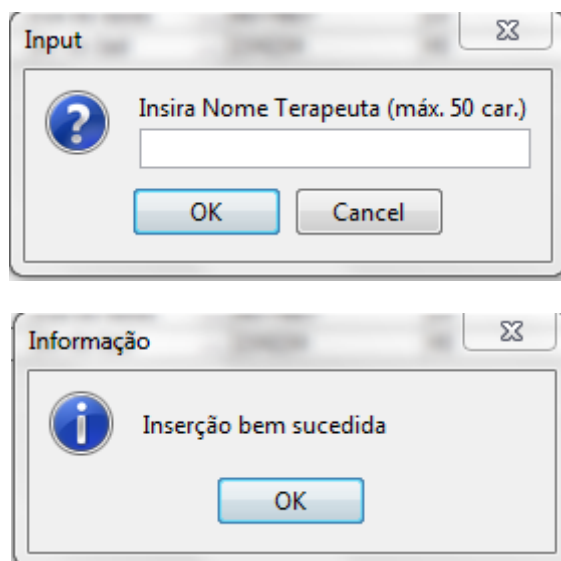
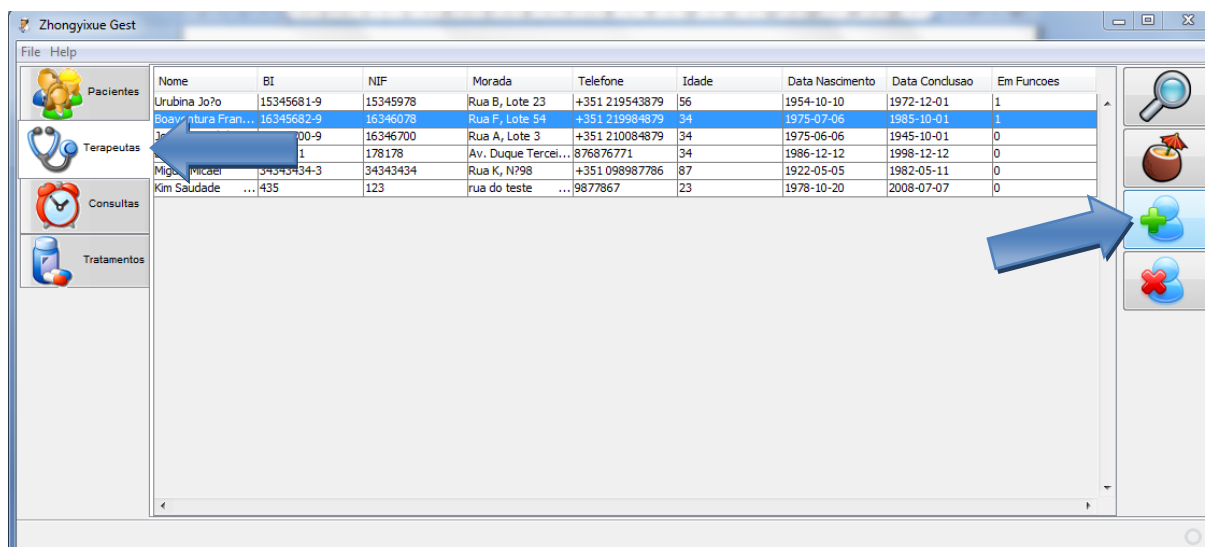
Insira número BI (máx. 10 car.)

OK Cancel

Input

Insira Nome Paciente (máx. 50 car.)

OK Cancel

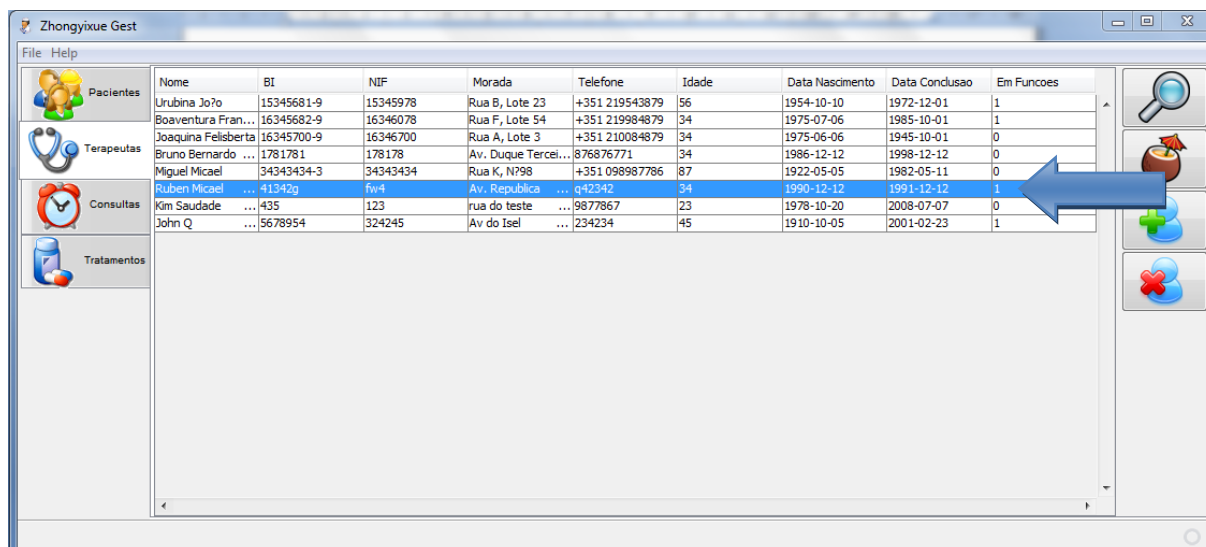


Funcionalidade não implementada.

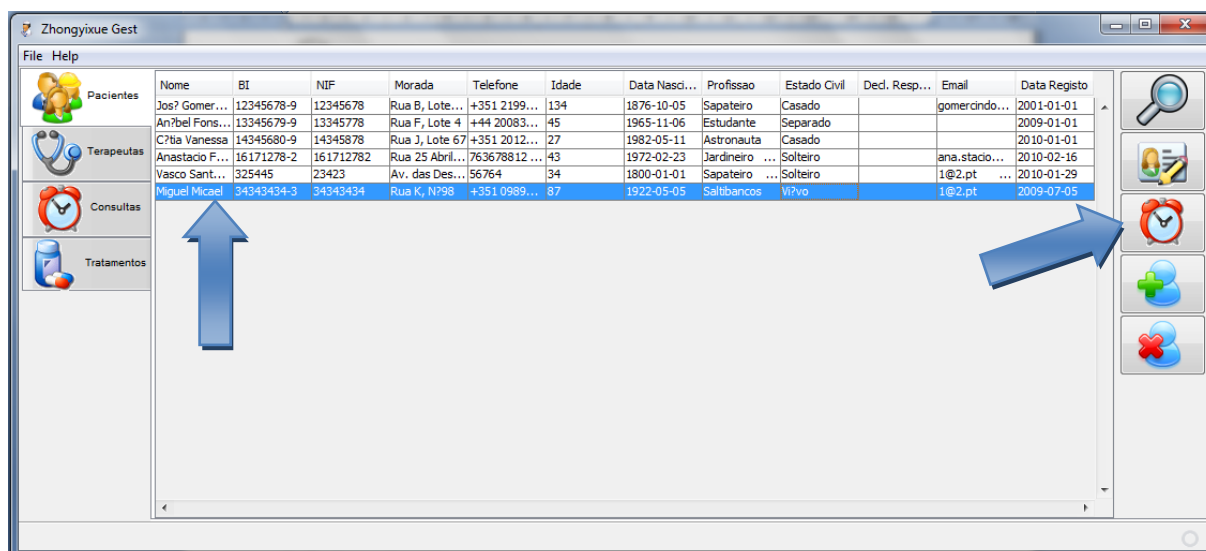
Nova Consulta

Registrar uma nova consulta para um paciente.

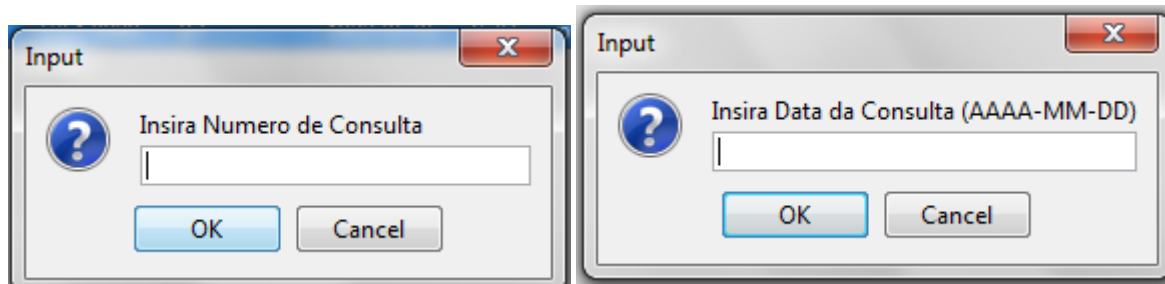
Primeiro será necessário seleccionar o terapeuta que irá realizar a consulta.



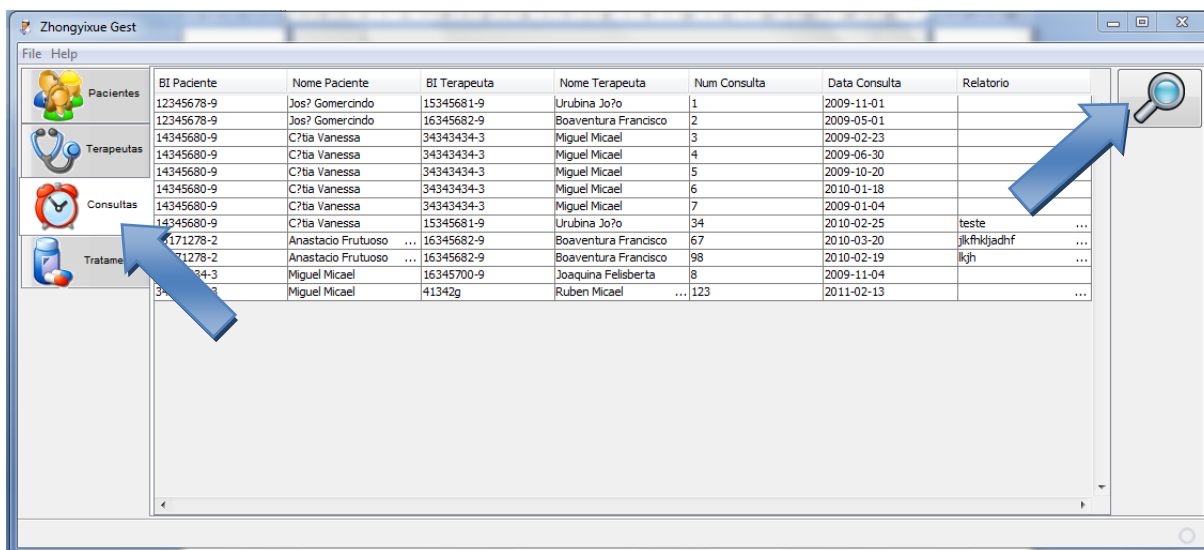
De seguida seleccione o paciente para qual deverá ser marcada a consulta e carregue no botão de agendamento.



Serão colocadas algumas questões:



Podemos depois verificar no separador de Consultas o agendamento.



BI Paciente	Nome Paciente	BI Terapeuta	Nome Terapeuta	Num Consulta	Data Consulta	Relatorio
12345678-9	Jos? Gomerindo	15345681-9	Urubina Jo?o	1	2009-11-01	
12345678-9	Jos? Gomerindo	16345682-9	Boaventura Francisco	2	2009-05-01	
14345680-9	C?tia Vanessa	34343434-3	Miguel Micael	3	2009-02-23	
14345680-9	C?tia Vanessa	34343434-3	Miguel Micael	4	2009-06-30	
14345680-9	C?tia Vanessa	34343434-3	Miguel Micael	5	2009-10-20	
14345680-9	C?tia Vanessa	34343434-3	Miguel Micael	6	2010-01-18	
14345680-9	C?tia Vanessa	34343434-3	Miguel Micael	7	2009-01-04	
14345680-9	C?tia Vanessa	15345681-9	Urubina Jo?o	34	2010-02-25	teste ...
171278-2	Anastacio Frutuoso ...	16345682-9	Boaventura Francisco	67	2010-03-20	jkfthdjadhf ...
171278-2	Anastacio Frutuoso ...	16345682-9	Boaventura Francisco	98	2010-02-19	lkgh ...
343434-3	Miguel Micael	16345700-9	Joaquina Felisberta	8	2009-11-04	
343434-3	Miguel Micael	41342g	Ruben Micael	123	2011-02-13	

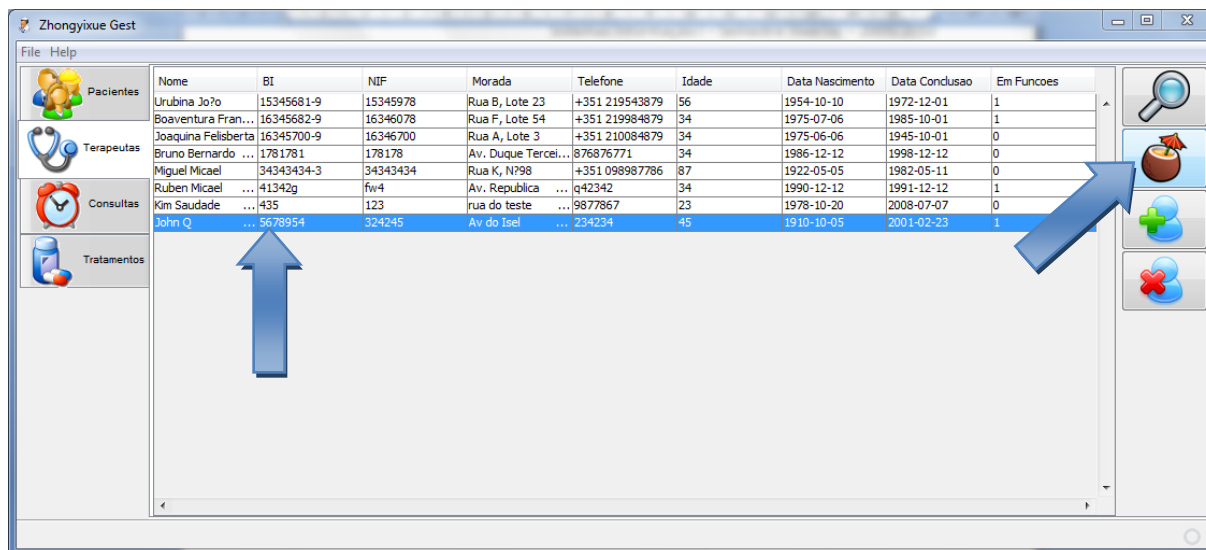
Remoção de Habilitação de Terapeuta

Remover a habilitação de um terapeuta para a realização de um determinado tratamento.

Funcionalidade não implementada.

Suspensão de Terapeuta

Registar a suspensão de funções de um terapeuta.



Para registar a suspender as funções deverá seleccionar o terapeuta que pretende suspender e carregar no botão de Suspensão. O campo “Em Funções” passará a 0.

Conclusão

Foram criadas diversas vistas de forma a garantir as funcionalidades solicitadas, para além disso procurámos que as operações de acesso à base de dados fossem concretizadas de forma atómica através da utilização de transacções.

O cumprimento de todas as restrições apresenta-se como o maior desafio nesta fase de implementação, aliada à necessidade de apresentação dos dados em interface gráfico. Apesar da tentativa de separarmos a parte de interface e a parte de acesso aos dados (um esboço de padrão de design MVC) foi muito complicado de o concretizar devido ao curto espaço de tempo na implementação desta aplicação.

Anexos

Listagem de rotinas de acesso a BD

Abertura da base de dados

```
private void loadDatabase() {
    try {
        Class.forName(SQL_DRIVER);
        String url = ODBC_SOURCE ;

        con = DriverManager.getConnection(url);
        System.out.println("Connection Established Successfully");
    } catch (ClassNotFoundException e) {
        Frame errorFrame = new Frame();
        JOptionPane.showMessageDialog(errorFrame, "SQL Driver not
found", "Error", JOptionPane.ERROR_MESSAGE);
        System.exit(0);

    } catch (SQLException e) {
        Frame errorFrame = new Frame();
        JOptionPane.showMessageDialog(errorFrame, "ODBC Source not
found", "Error", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    }
}
```

Fecho de base de dados

```
protected void closeDatabase() {
    try {
        if (con != null)
            con.close();
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(new Frame(), "Database Closing
Error", "Erro", JOptionPane.ERROR_MESSAGE);
    }
}
```

Inserção de Paciente

```
protected void insertPatient() throws SQLException {
    con.setAutoCommit(false);
    String cmdInsPessoa = "insert into "+

"dbo.PESSOA (BI,NIF,nomePessoa,morada,telefone,idade,dataNasc) " +
    " values (?,?,?,?,?,?,?) ";
    String cmdInsPaciente = "insert into " +

"dbo.PACIENTE (BI,profissao,estadoCivil,declRespon,email,dataRegisto) " +
    "values (?,?,?,?,null,?,?);" ;
```

```
PreparedStatement pstmt = con.prepareStatement(cmdInsPessoa ) ;
String biString = JOptionPane.showInputDialog(null, "Insira número
BI (máx. 10 car.)");
pstmt.setString(1, biString);
pstmt.setString(2, JOptionPane.showInputDialog(null, "Insira NIF
(máx. 10 car.)"));
pstmt.setString(3, JOptionPane.showInputDialog(null, "Insira Nome
Paciente (máx. 50 car.)"));
pstmt.setString(4, JOptionPane.showInputDialog(null, "Insira Morada
(máx. 30 car.)"));
pstmt.setString(5, JOptionPane.showInputDialog(null, "Insira
Telefone (máx. 15 car.)"));
pstmt.setByte(6, Byte.valueOf(JOptionPane.showInputDialog(null,
"Insira Idade")) );
pstmt.setString(7, JOptionPane.showInputDialog(null, "Insira Data
Nascimento (AAAA-MM-DD)"));
pstmt.executeUpdate() ;

pstmt = con.prepareStatement(cmdInsPaciente) ;
pstmt.setString(1, biString);
pstmt.setString(2, JOptionPane.showInputDialog(null, "Insira
Profissão (máx. 20 car.)"));
pstmt.setString(3, JOptionPane.showInputDialog(null, "Insira Estado
Civil (máx. 15 car.)"));
pstmt.setString(4, JOptionPane.showInputDialog(null, "Insira Email
(máx. 20 car.)"));
pstmt.setString(5, JOptionPane.showInputDialog(null, "Insira Data de
Registo (AAAA-MM-DD)"));

pstmt.executeUpdate() ;
con.commit() ;
con.setAutoCommit(true);

}
```

Inserção de Terapeuta

```
protected void insertDoctor() throws SQLException {
    con.setAutoCommit(false) ;
    String cmdInsPessoa = "insert into "+
    "dbo.PESSOA (BI,NIF,nomePessoa,morada,telefone,idade,dataNasc)" +
    " values (?,?,?,?,?,?,?) ;" ;
    String cmdInsTerapeuta = "insert into " +
    "dbo.TERAPEUTA (BI,dataConclusao,emFuncoes) " +
    "values (?,?,?);" ;
    PreparedStatement pstmt = con.prepareStatement(cmdInsPessoa ) ;
    String biString = JOptionPane.showInputDialog(null, "Insira número
BI (máx. 10 car.)");
    pstmt.setString(1, biString);
    pstmt.setString(2, JOptionPane.showInputDialog(null, "Insira NIF
(máx. 10 car.)"));
    pstmt.setString(3, JOptionPane.showInputDialog(null, "Insira Nome
Terapeuta (máx. 50 car.)"));
    pstmt.setString(4, JOptionPane.showInputDialog(null, "Insira Morada
(máx. 30 car.)"));
    pstmt.setString(5, JOptionPane.showInputDialog(null, "Insira
Telefone (máx. 15 car.)"));
    pstmt.setByte(6, Byte.valueOf(JOptionPane.showInputDialog(null,
```

```
"Insira Idade")) ;
    pstmt.setString(7, JOptionPane.showInputDialog(null, "Insira Data
Nascimento (AAAA-MM-DD)"));
    pstmt.executeUpdate() ;

    pstmt = con.prepareStatement(cmdInsTerapeuta) ;
    pstmt.setString(1, biString);
    pstmt.setString(2, JOptionPane.showInputDialog(null, "Insira Data
Conclusão Especialização (AAAA-MM-DD)"));
    pstmt.setBoolean(3,
Boolean.valueOf(JOptionPane.showInputDialog(null, "Está em funções? 'true'
ou 'false'", true)));
    pstmt.executeUpdate() ;

    con.commit() ;
    con.setAutoCommit(true);
}
```

Inserção de Consulta

```
protected void insertSchedule( String biPaciente, String biTerapeuta )
throws SQLException {
    con.setAutoCommit(false) ;
    String cmdInsSchedule = "insert into "+
        "dbo.CONSULTA(BIPaciente,BITerapeuta, numConsulta,
dataConsulta, relatorio)" +
        " values (?, ?, ?, ?, ?) ;" ;
    PreparedStatement pstmt = con.prepareStatement(cmdInsSchedule) ;
    pstmt.setString(1, biPaciente);
    pstmt.setString(2, biTerapeuta);
    pstmt.setInt(3, (Integer.parseInt(JOptionPane.showInputDialog(null,
"Insira Numero de Consulta"))));
    pstmt.setString(4, JOptionPane.showInputDialog(null, "Insira Data
da Consulta (AAAA-MM-DD)"));
    pstmt.setString(5, JOptionPane.showInputDialog(null, "Insira
relatorio (máx. 500 car.)"));
    pstmt.executeUpdate() ;

    con.commit() ;
    con.setAutoCommit(true);
}
```

Remoção de Paciente

```
/*Não implementado para várias tabelas referenciadas*/
protected void removePatient(String bi) throws SQLException {
    con.setAutoCommit(false) ;
    String cmdDelPessoa = "DELETE FROM dbo.PESSOA WHERE BI=?" ;
    String cmdDelPaciente = "DELETE FROM dbo.PACIENTE WHERE BI=?" ;

    PreparedStatement pstmt = con.prepareStatement(cmdDelPaciente) ;
    pstmt.setString(1, bi);
    pstmt.executeUpdate() ;

    pstmt = con.prepareStatement(cmdDelPessoa) ;
    pstmt.setString(1, bi);
}
```




```
pstmt.executeUpdate() ;  
con.commit() ;  
con.setAutoCommit(true) ;  
}
```

Listagem de Terapeutas

```
buttonListDoctors.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        try {  
            queryDoctor.setQuery("SELECT Nome, BI, NIF, Morada,  
Telefone," + " Idade, \"Data Nascimento\", \"Data Conclusao\", "  
+ "\"Em Funcoes\" " +  
"FROM LISTA_TERAPEUTAS");  
        } catch (Exception ex) {  
            ex.printStackTrace() ;  
            JOptionPane.showMessageDialog(new Frame(), "Erro na  
procura de Terapeuta",  
"Erro", JOptionPane.ERROR_MESSAGE);  
        }  
    }  
});
```

Listagem de Pacientes

```
buttonListPatients.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        try {  
            queryPatient.setQuery("SELECT Nome, BI, NIF, Morada,  
Telefone," + " Idade, \"Data Nascimento\", Profissao,  
\"Estado Civil\", " + "\"Decl. Responsabilidade\", Email,  
\"Data Registo\" " +  
"FROM LISTA_PACIENTES");  
        } catch (Exception ex) {  
            JOptionPane.showMessageDialog(new Frame(), "Erro na  
procura de pacientes",  
"Erro", JOptionPane.ERROR_MESSAGE);  
        }  
    }  
});
```

Apresentação de ficha de diagnóstico

```
buttonDiagnosticCard.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        try {  
            queryPatient.setQuery("SELECT \"Nome Paciente\", \"BI  
Paciente\", " +  
"\"NIF Paciente\", Morada, Telefone,  
Idade, \"Data Nascimento\", " +  
" Profissao, \"Estado Civil\", \"Decl.  
Responsabilidade\", " +  
" Email, \"Data Registo\", \"Nome Sintoma\",  
\"Desc. Sintoma\", " +  
" \"Data Sintoma\", Notas, \"Nome Padrao\",  
\"Desc. Padrao\", \"Nome Terapeuta\" " +
```

```
        "FROM DIAGNOSTICO_PACIENTES");
    }catch (Exception ex) {
        ex.printStackTrace() ;
        JOptionPane.showMessageDialog(new Frame(), "Erro na
ficha de diagnostico",
        "Erro", JOptionPane.ERROR_MESSAGE);
    }
}
});
```

Listagem de consultas

```
buttonListSchedule.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
            querySchedule.setQuery("SELECT \"BI Paciente\", \"Nome
Paciente\", \"+
            \"\"BI Terapeuta\", \"Nome Terapeuta\", \"Num
Consulta\", \"+
            \"\"Data Consulta\", \"Relatorio\" \" \" +
            \"FROM LISTA_CONSULTAS\");
        }catch (Exception ex) {
            ex.printStackTrace() ;
            JOptionPane.showMessageDialog(new Frame(), "Erro na
procura de Consulta",
            "Erro", JOptionPane.ERROR_MESSAGE);
        }
    }
});
```

Função de Suporte setQuery

```
public void setQuery(String q) throws SQLException {
    cache = new Vector();

    // Execute the query and store the result set and its metadata
    statement = db.createStatement() ;
    rs = statement.executeQuery(q);

    ResultSetMetaData meta = rs.getMetaData();
    colCount = meta.getColumnCount();

    // Now we must rebuild the headers array with the new column names
    headers = new String[colCount];
    for (int h = 1; h <= colCount; h++) {
        headers[h - 1] = meta.getColumnName(h);
    }
    while (rs.next()) {
        String[] record = new String[colCount];
        for (int i = 0; i < colCount; i++) {
            record[i] = rs.getString(i + 1);
        }
        cache.addElement(record);
    }
}
```

```
    }  
    fireTableChanged(null); // notify everyone that we have a new  
table.  
    cleanQuery() ;  
}
```

Suspensão de Terapeuta

```
protected void retireDoctor(String bi) throws SQLException {  
    con.setAutoCommit(false) ;  
  
    String cmdUpdtDoctor = "UPDATE dbo.TERAPEUTA " +  
                           "SET emFuncoes = 0" +  
                           "WHERE BI=?" ;  
    PreparedStatement pstmt = con.prepareStatement(cmdUpdtDoctor) ;  
    pstmt.setString(1, bi);  
    pstmt.executeUpdate() ;  
  
    con.commit() ;  
    con.setAutoCommit(true) ;  
}
```

Instruções de Execução

- 1) Deverá ser criada uma Data Source ODBC com driver SQL com o nome Zhongyixue.
- 2) Correr os serviços “sqlbrowser” e “mssqlserver”.
- 3) Aceder à pasta Zhongyixue e executar “RunZhongyixue.bat”.