

Sistema de agendamento para serviços de limpeza

Projeto de extensão - Java



1. *Partes interessadas*

- Cliente Valdinete Cabral de Oliveira
- Contexto
- Necessidade



2. Problemática e objetivos

- Modelo
- Objetivo (CRUD)
- Faturamento
- Histórico

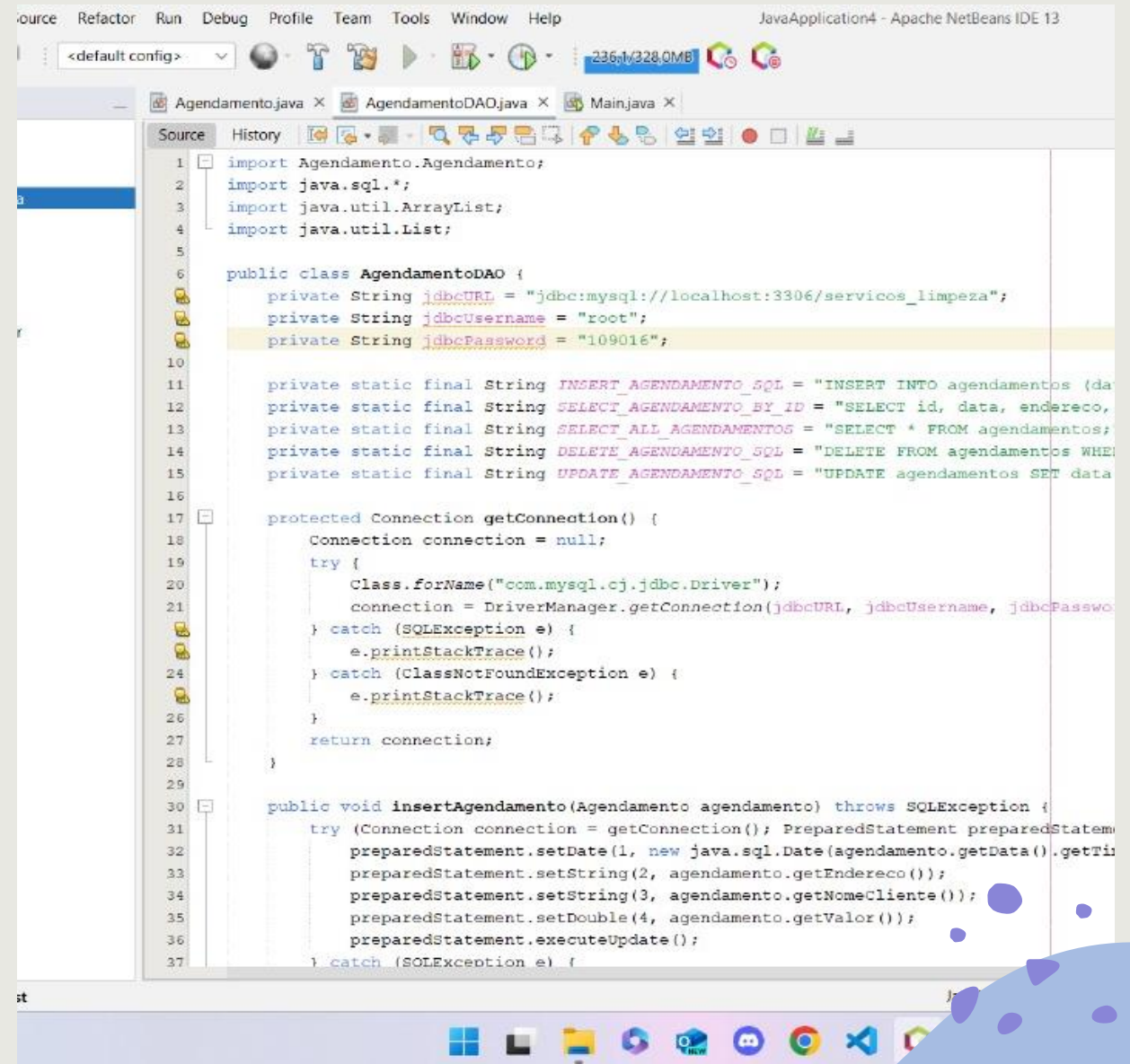
Minimundo - Sistema de agendamento de diárias

O programa consiste em um sistema de gerenciamento para facilitar e mapear os agendamentos referentes às diárias dos serviços de limpeza e cuidados de casa contratados. Nesse sentido, o projeto deverá abordar os seguintes tópicos:

- O programa será feito em modelo de calendário com as datas ocupadas por agendamento e datas livres sinalizadas;
- Objetivo principal do programa é fornecer a opção de gerenciar agendamentos, portanto o cliente deverá ser capaz de adicionar, remover e editar registros;
- Além disso, o cliente deverá ser capaz de visualizar dados dos agendamentos, como nome do contratante, endereço do serviço, telefone para contato, tipo de serviço contratado e valor total da diária;
- Por fim, o sistema deverá fornecer uma estimativa de ganho mensal conforme serviços contratados por meio de um contador associado aos agendamentos.

3. Desenvolvimento do sistema

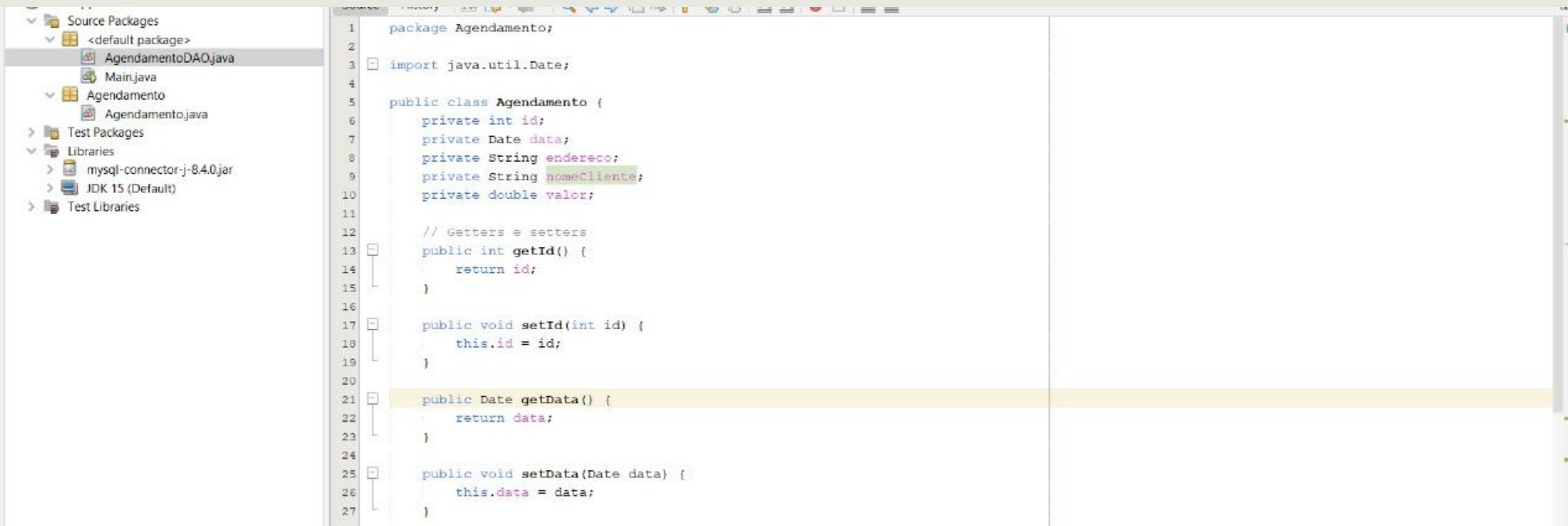
- Classe AgendamentoDAO



```
1 import Agendamento.Agendamento;
2 import java.sql.*;
3 import java.util.ArrayList;
4 import java.util.List;
5
6 public class AgendamentoDAO {
7     private String jdbcURL = "jdbc:mysql://localhost:3306/servicos_limpeza";
8     private String jdbcUsername = "root";
9     private String jdbcPassword = "109016";
10
11     private static final String INSERT_AGENDAMENTO_SQL = "INSERT INTO agendamentos (data, endereco, nome_cliente, valor) VALUES (?, ?, ?, ?)";
12     private static final String SELECT_AGENDAMENTO_BY_ID = "SELECT id, data, endereco, nome_cliente, valor FROM agendamentos WHERE id = ?";
13     private static final String SELECT_ALL_AGENDAMENTOS = "SELECT * FROM agendamentos";
14     private static final String DELETE_AGENDAMENTO_SQL = "DELETE FROM agendamentos WHERE id = ?";
15     private static final String UPDATE_AGENDAMENTO_SQL = "UPDATE agendamentos SET data = ?, endereco = ?, nome_cliente = ?, valor = ? WHERE id = ?";
16
17     protected Connection getConnection() {
18         Connection connection = null;
19         try {
20             Class.forName("com.mysql.cj.jdbc.Driver");
21             connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);
22         } catch (SQLException e) {
23             e.printStackTrace();
24         } catch (ClassNotFoundException e) {
25             e.printStackTrace();
26         }
27         return connection;
28     }
29
30     public void insertAgendamento(Agendamento agendamento) throws SQLException {
31         try (Connection connection = getConnection(); PreparedStatement preparedStatement = connection.prepareStatement(INSERT_AGENDAMENTO_SQL)) {
32             preparedStatement.setDate(1, new java.sql.Date(agendamento.getData().getTime()));
33             preparedStatement.setString(2, agendamento.getEndereco());
34             preparedStatement.setString(3, agendamento.getNomeCliente());
35             preparedStatement.setDouble(4, agendamento.getValor());
36             preparedStatement.executeUpdate();
37         } catch (SQLException e) {
38             e.printStackTrace();
39         }
40     }
41 }
```

3. Desenvolvimento do sistema

- Classe AgendamentoDAO

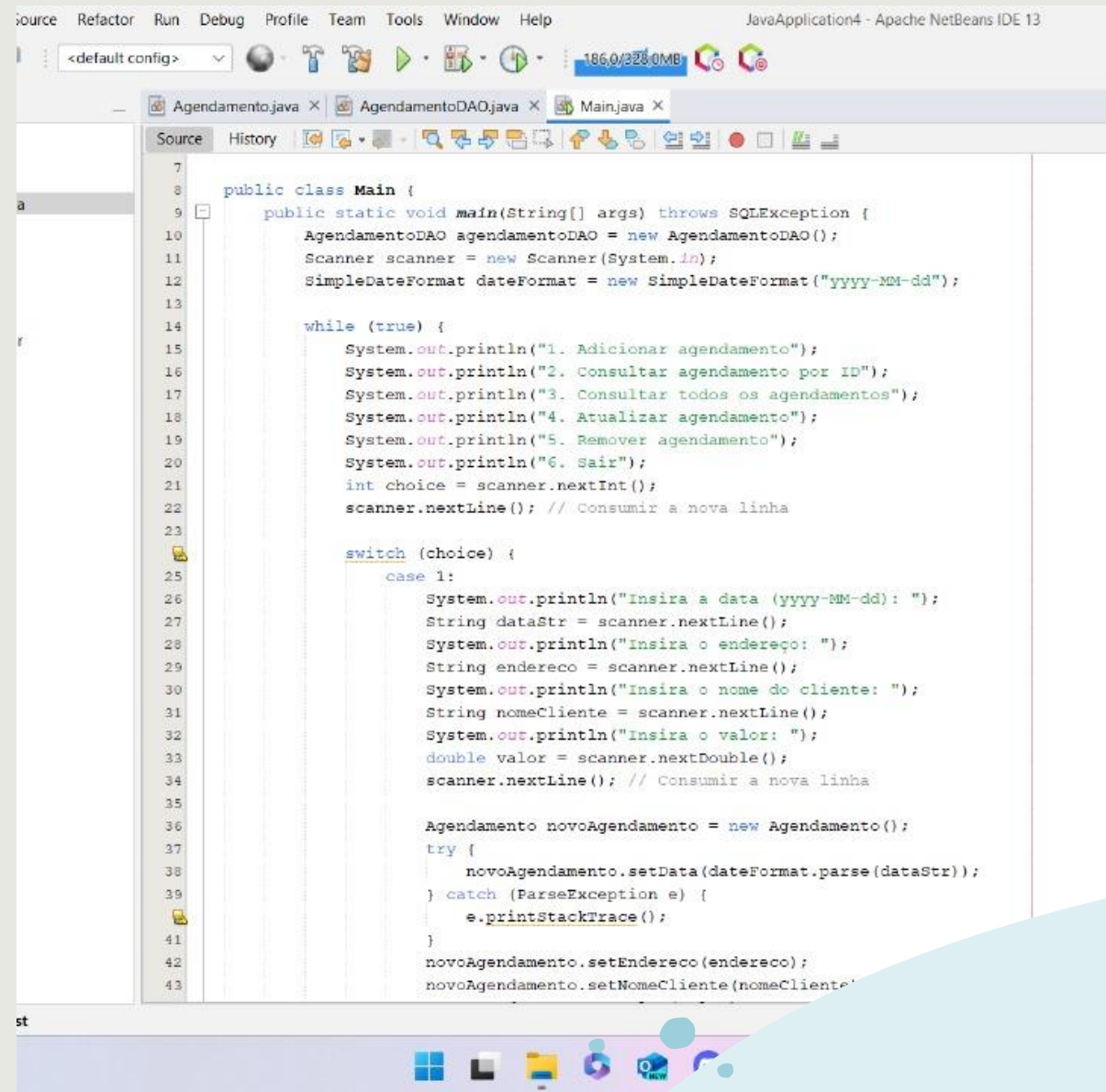


The screenshot displays an IDE interface. On the left, the 'Source Packages' view shows a project structure with 'AgendamentoDAO.java' selected under the 'Agendamento' package. The main editor window shows the code for 'AgendamentoDAO.java'. The code includes package declarations, imports, and the implementation of the 'Agendamento' class with its attributes and methods.

```
1 package Agendamento;
2
3 import java.util.Date;
4
5 public class Agendamento {
6     private int id;
7     private Date data;
8     private String endereco;
9     private String nomeCliente;
10    private double valor;
11
12    // Getters e setters
13    public int getId() {
14        return id;
15    }
16
17    public void setId(int id) {
18        this.id = id;
19    }
20
21    public Date getData() {
22        return data;
23    }
24
25    public void setData(Date data) {
26        this.data = data;
27    }
28 }
```

3. Desenvolvimento do sistema

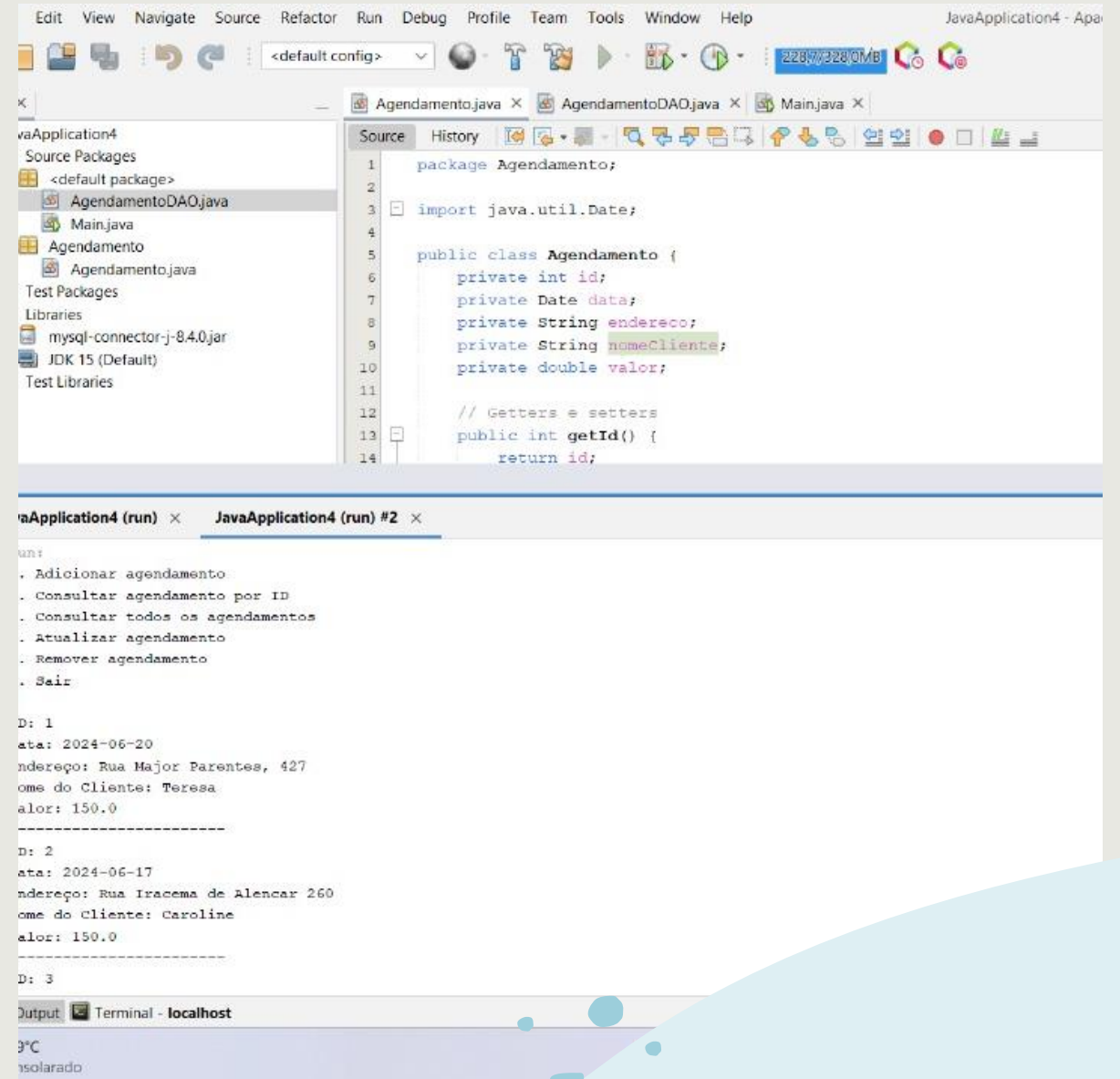
- Classe Main



```
7
8 public class Main {
9     public static void main(String[] args) throws SQLException {
10         AgendamentoDAO agendamentoDAO = new AgendamentoDAO();
11         Scanner scanner = new Scanner(System.in);
12         SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
13
14         while (true) {
15             System.out.println("1. Adicionar agendamento");
16             System.out.println("2. Consultar agendamento por ID");
17             System.out.println("3. Consultar todos os agendamentos");
18             System.out.println("4. Atualizar agendamento");
19             System.out.println("5. Remover agendamento");
20             System.out.println("6. Sair");
21             int choice = scanner.nextInt();
22             scanner.nextLine(); // Consumir a nova linha
23
24             switch (choice) {
25                 case 1:
26                     System.out.println("Insira a data (yyyy-MM-dd): ");
27                     String dataStr = scanner.nextLine();
28                     System.out.println("Insira o endereço: ");
29                     String endereco = scanner.nextLine();
30                     System.out.println("Insira o nome do cliente: ");
31                     String nomeCliente = scanner.nextLine();
32                     System.out.println("Insira o valor: ");
33                     double valor = scanner.nextDouble();
34                     scanner.nextLine(); // Consumir a nova linha
35
36                     Agendamento novoAgendamento = new Agendamento();
37                     try {
38                         novoAgendamento.setData(dateFormat.parse(dataStr));
39                     } catch (ParseException e) {
40                         e.printStackTrace();
41                     }
42                     novoAgendamento.setEndereco(endereco);
43                     novoAgendamento.setNomeCliente(nomeCliente);
```

3. Desenvolvimento do sistema

- Teste de funcionamento do programa



The screenshot displays an IDE window titled 'JavaApplication4 - Apa'. The top menu bar includes 'Edit', 'View', 'Navigate', 'Source', 'Refactor', 'Run', 'Debug', 'Profile', 'Team', 'Tools', 'Window', and 'Help'. The toolbar shows various icons for file operations and execution. The left sidebar shows the project structure with 'Source Packages' containing '<default package>', 'AgendamentoDAO.java', 'Main.java', and 'Agendamento.java'. The 'Libraries' section lists 'mysql-connector-j-8.4.0.jar' and 'JDK 15 (Default)'. The 'Test Packages' and 'Test Libraries' sections are empty. The main editor shows the source code of 'Agendamento.java' with the following content:

```
1 package Agendamento;
2
3 import java.util.Date;
4
5 public class Agendamento {
6     private int id;
7     private Date data;
8     private String endereco;
9     private String nomeCliente;
10    private double valor;
11
12    // Getters e setters
13    public int getId() {
14        return id;
15    }
16 }
```

Below the editor, the 'Run' button is visible. The bottom section shows the output of the program, titled 'JavaApplication4 (run) #2'. The output displays a menu of options for managing appointments, followed by three test cases (D: 1, D: 2, D: 3) where user input is shown for date, address, client name, and value.

```
run:
. Adicionar agendamento
. Consultar agendamento por ID
. Consultar todos os agendamentos
. Atualizar agendamento
. Remover agendamento
. Sair

D: 1
ata: 2024-06-20
ndereço: Rua Major Parentes, 427
ome do Cliente: Teresa
alor: 150.0
-----
D: 2
ata: 2024-06-17
ndereço: Rua Iracema de Alencar 260
ome do Cliente: Caroline
alor: 150.0
-----
D: 3
```

The bottom status bar shows 'Output Terminal - localhost' with a temperature of '3°C' and the word 'isolado'.

3. Desenvolvimento do sistema

- Demonstração do Banco de Dados configurado

The screenshot displays the MySQL Workbench interface. On the left, the 'Navigator' pane shows the 'Schemas' tree with 'servicos_limpeza' expanded, revealing the 'agendamentos' table. The 'Columns' section for 'agendamentos' lists: id, data, endereco, nome_cliente, and valor. Below this, the 'Table: agendamentos' section shows the column definitions: id (int AI PK), data (date), endereco (varchar(255)), nome_cliente (varchar(255)), and valor (decimal(10,2)).

The main query editor shows the query: `SELECT * FROM servicos_limpeza.agendamentos;`. The 'Result Grid' pane displays the query results in a table format:

	id	data	endereco	nome_cliente	valor
1	1	2024-06-20	Rua Major Parentes, 427	Teresa	150.00
2	2	2024-06-19	Rua Iracema de Alencar 260	Caroline	160.00
3	3	2024-06-17	Rua Temístocles Sávio	Maria Lúcia	150.00
4	4	2024-06-17	Av das Américas, 13600	Luiz Carlos	250.00
5	5	2024-06-11	Rua São Francisco Xavier 1615	Bruno	160.00
6	6	2024-06-17	Av das Américas, 14572	Helena	220.00
7	7	2024-06-13	Rua Princesa Leopoldina, 106	Cristiane	160.00
*		NULL	NULL	NULL	NULL

The 'Output' pane at the bottom shows the execution details: a green checkmark, the time 12:28:51, and the query: `SELECT * FROM servicos_limpeza.agendamentos LIMIT 0, 1000`. The status bar at the bottom indicates a temperature of 29°C and 'Ensolarado'.

4. *Considerações finais*

- Conclusão
- Espaço para retirada de dúvidas

Integrantes do grupo:

- Bernardo Andrade da Costa
- Felippe Pereira Maciel
- Matheus Rocha de Oliveira