



# MedStock Sprint 4

## Integrantes:

Felipe Rivetti Mizher

Pedro Hosken Fernandes Guimarães

Rafael Rehfeld Martins de Oliveira

Vitor Augusto Viana Azevedo


# Requisitos funcionais e não funcionais:

Os requisitos iniciais foram implementados corretamente, incluindo ajustes feitos ao longo do projeto. No entanto, algumas ideias apresentadas anteriormente, como o **sistema de notificações**, não foram incluídas, mas poderão ser consideradas em futuras atualizações.

# Projeto de Banco de Dados:

```
public boolean conectar() {  
    String driverName = "org.postgresql.Driver";  
    String serverName = "medstocker.postgres.database.azure.com";  
    String mydatabase = "postgres";  
    int porta = 5432;  
    String url = "jdbc:postgresql://" + serverName + ":" + porta + "/" + mydatabase;  
    String username = "vitor";  
    String password = "Medstocker@";  
    boolean status = false;  
  
    try {  
        Class.forName(driverName);  
        conexao = DriverManager.getConnection(url, username, password);  
        status = (conexao == null);  
    } catch (ClassNotFoundException e) {  
        System.err.println("Conexão NÃO efetuada com o postgres -- Driver não encontrado -- " + e.getMessage());  
    } catch (SQLException e) {  
        System.err.println("Conexão NÃO efetuada com o postgres -- " + e.getMessage());  
    }  
  
    return status;  
}
```

### Registro



Redes Sociais

Facebook Twitter Instagram

Contato de Suporte

Política de Privacidade

© 2024 Red Social

Desenvolvido por: [Logo] | [Logo]

	crm	nome	sobrenome	email	data_nascimento	senha
	integer	character varying	character varying	character varying	date	character varying
1	1	Dr. Pedro	Hosken	pg87066@gmail.com	2004-01-29	senha_padrao
2	1234	Dr. Carlos	Silva	carlos@exemplo.com	1980-05-12	senha_padrao
3	4312	Felipe	Rivetti	felipe@exemplo.com	1234-05-12	senha_padrao
4	123456	Marcilio	Guimaraes	marcilio.guimaraes@gmail.com	1967-06-16	itmar67
5	12345	julia	hosken	123@gmail.com	1200-12-12	123
6	98	Vitorvitor	Azevedo	teste1234@gmail.com	2005-04-13	12345
7	123	video	video	video@gmail.com	4567-03-12	video



	crm	nome	sobrenome	email	data_nascimento	senha
	integer	character varying	character varying	character varying	date	character varying
1	1	Dr. Pedro	Hosken	pg87066@gmail.com	2004-01-29	senha_padrao
2	1234	Dr. Carlos	Silva	carlos@exemplo.com	1980-05-12	senha_padrao
3	4312	Felipe	Rivetti	felipe@exemplo.com	1234-05-12	senha_padrao
4	123456	Marcilio	Guimaraes	marcilio.guimaraes@gmail.com	1967-06-16	itmar67
5	12345	julia	hosken	123@gmail.com	1200-12-12	123
6	98	Vitorvitor	Azevedo	teste1234@gmail.com	2005-04-13	12345
7	123	video	video	video@gmail.com	4567-03-12	video
8	222	apresentacao	apresentacao	apresentacao@gmail.com	2022-02-22	12345

# Tela de Listagem e CRUD's



```

public Statement createStatement() throws SQLException {
    stmt = conexao.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);
    return stmt;
}

public PreparedStatement prepareStatement(String sql) throws SQLException {
    createStatement();
    return conexao.prepareStatement(sql);
}

public int executeUpdate(PreparedStatement preparedStatement) throws SQLException {
    // Func para CREATE, INSERT E DELETE
    return preparedStatement.executeUpdate();
}

public ResultSet executeQuery(String sql) throws SQLException {
    // Func apenas para READ
    createStatement();
    return stmt.executeQuery(sql);
}

```

```

public ResultSet executeQuery(String sql) throws SQLException {
    // Func apenas para READ
    createStatement();
    return stmt.executeQuery(sql);
}

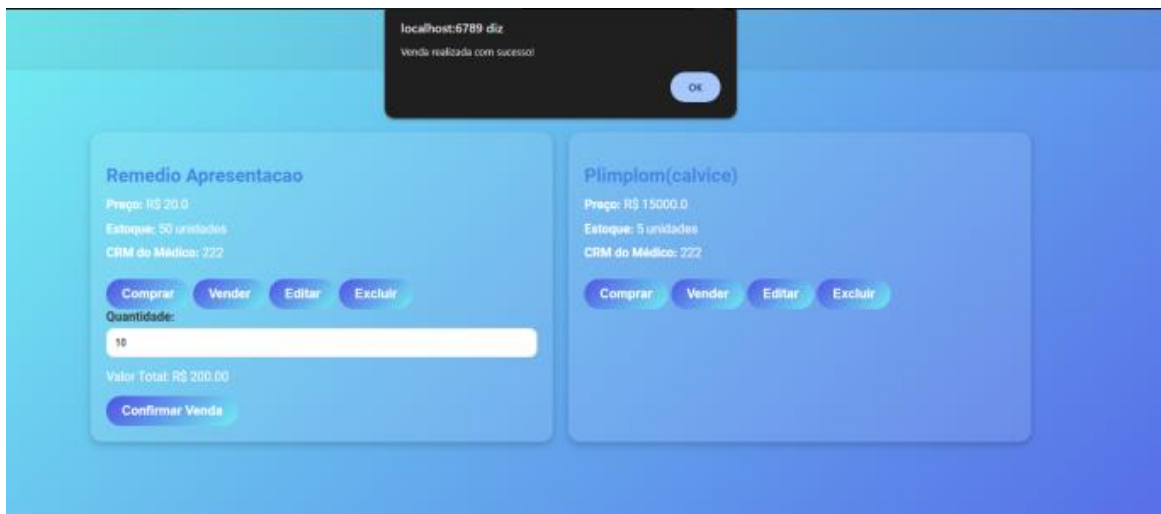
public boolean closeStatement() {
    boolean status = false;

    try {
        if (stmt != null) {
            stmt.close();
            status = true;
        }
    } catch (SQLException e) {
        System.err.println(e.getMessage());
    }

    return status;
}

```

# Exemplo Venda:



# Sistema Inteligente:

```
public static void main(String[] args) {

    String endpoint = System.getenv("VISION_ENDPOINT");
    String key = System.getenv("VISION_KEY");

    if (endpoint == null || key == null) {
        System.out.println("Missing environment variable 'VISION_ENDPOINT' or 'VISION_KEY'.");
        System.out.println("Set them before running this sample.");
        System.exit(1);
    }

    // Criando cliente de análise de imagem
    ImageAnalysisClient client = new ImageAnalysisClientBuilder()
        .endpoint(endpoint)
        .credential(new KeyCredential(key))
        .buildClient();

    // Análise síncrona da imagem a partir de uma URL
    ImageAnalysisResult result = client.analyzeFromUrl(
        "https://th.bing.com/th/id/OIP.2iBkGDTzNIuRJehKHn1r1QHADM?rs=1&pid=ImgDetMain",
        Arrays.asList(VisualFeatures.CAPTION, VisualFeatures.READ),
        new ImageAnalysisOptions().setGenderNeutralCaption(true));
}
```

```
// Imprimindo resultados da análise
System.out.println("Resultados da análise de imagem:");
System.out.println("Legenda:");
System.out.println("  \"\" + result.getCaption().getText() + "\", Confiança "
    + String.format("%.4f", result.getCaption().getConfidence()));

System.out.println("Texto detectado:");
List<String> datasEncontradas = new ArrayList<>(); // Lista para armazenar as datas encontradas
String regex = "\\b(\\d{1,2})/(\\d{1,2})/(\\d{2,4})\\b"; // Regex para datas no formato DD/MM/AAAA
Pattern pattern = Pattern.compile(regex);

for (DetectedTextLine line : result.getRead().getBlocks().get(0).getLines()) {
    String text = line.getText();
    // Procurando datas na linha
    Matcher matcher = pattern.matcher(text);
    while (matcher.find()) {
        String dataExpandida = expandirAno(matcher.group());
        if (dataExpandida != null) {
            datasEncontradas.add(dataExpandida); // Adiciona apenas datas válidas
        }
    }
}
```

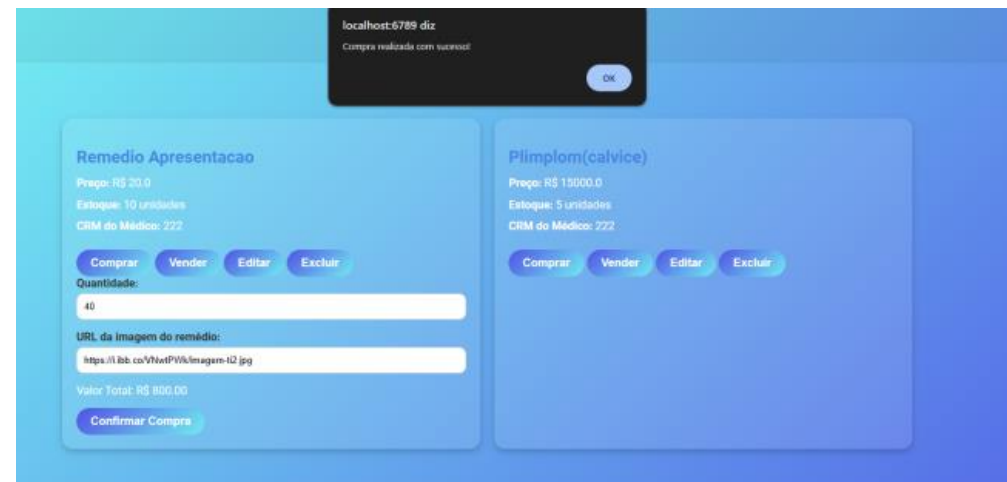
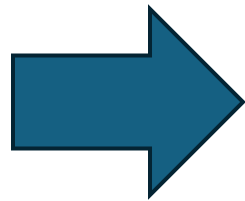


# Sistema Inteligente:

```
// Verificando a maior data
String dataValidade = null;
if (!datasEncontradas.isEmpty()) {
    System.out.println("\nDatas encontradas na imagem:");
    for (String data : datasEncontradas) {
        System.out.println(" - " + data);
    }
    dataValidade = datasEncontradas.get(0); // Assume a primeira data como validade inicialmente
    for (String data : datasEncontradas) {
        if (compareDatas(data, dataValidade) > 0) {
            dataValidade = data; // Atualiza para a maior data encontrada
        }
    }
    System.out.println("\nData de validade identificada: " + dataValidade);
}
```

```
boolean dentroDaValidade = isDataDentroValidade(dataValidade);
if (dentroDaValidade) {
    System.out.println("O remédio está dentro do prazo de validade.");
} else {
    System.out.println("O remédio está fora do prazo de validade.");
}

} else {
    System.out.println("Nenhuma data foi encontrada na imagem.");
}
}
```





ysis

Resultados da análise de imagem:

Legenda:

"a group of yellow pills next to a bottle", Confiança 0.7617

Texto detectado:

Datas encontradas na imagem:

- 09/11/2016
- 09/11/2018

Data de validade identificada: 09/11/2018

O remédio está fora do prazo de validade.

# Serviços Online da IA:

**Serviços:**

**URL:**

O Image Analysis 4.0, parte da Azure Cognitive Services, é uma poderosa API baseada em inteligência artificial que permite a análise e interpretação de imagens. Ela oferece funcionalidades avançadas para identificar objetos, texto, e outros detalhes visuais em imagens.

Serviço implementado da IA

# O Sistema está hospedado na nuvem?

- URL: Banco de Dados
- Usuário: vitor
- Senha: Medstocker@

Sim, pois o sistema tem um banco de dados no Azure, que é uma plataforma de serviço em nuvem.



---

O seu sistema é  
seguro em relação  
a armazenamento  
de senhas?

Sim, o sistema tem  
um sistema de  
criptografia para as  
senhas.

# O seu sistema é seguro em relação a ataques por meio SQL Injection?

Sim, o Azure SQL Database oferece várias camadas de proteção para mitigar ataques como SQL Injection. No entanto, a segurança total depende de como as consultas são implementadas no código.



# URL do Git:

<https://github.com/ICEI-PUC-Minas-CC-TI/plmg-cc-ti2-2024-2-22-medstocker/tree/master>



# URL do Video:

<https://github.com/ICEI-PUC-Minas-CC-TI/plmg-cc-ti2-2024-2-22-medstocker/tree/master/Divulgacao/Video>