

Sistema de criação de chamado

```

31  src/sistemaChamados/Main.java
...
...  @@ -0,0 +1,31 @@
1  + package sistemaChamados;
2  +
3  + public class Main {
4  +
5  +     public static void main(String[] args) {
6  +         armazenaChamado armazenaChamado = new armazenaChamado();
7  +
8  +         Chamado chamado1 = new Chamado (1, "Problema no servidor");
9  +         Chamado chamado2 = new Chamado(2, "Erro de conexao");
10 +         Chamado chamado3 = new Chamado(3, "Aplicativo travado");
11 +
12 +         armazenaChamado.adicionarChamado(chamado1);
13 +         armazenaChamado.adicionarChamado(chamado2);
14 +         armazenaChamado.adicionarChamado(chamado3);
15 +
16 +         System.out.println("Lista de Chamados:");
17 +         for (Chamado chamado : armazenaChamado.getChamados()) {
18 +             System.out.println("Codigo: " + chamado.getCodigo());
19 +             System.out.println("Codigo: " + chamado.getCodigo());
20 +             System.out.println("Descricao: " + chamado.getDescricao());
21 +             System.out.println("-----");
22 +         }
23 +
24 +         int codigoBusca = 2;
25 +         Chamado chamadoBusca = armazenaChamado.getChamado(codigoBusca);
26 +         if (chamadoBusca != null) {
27 +             System.out.println("Chamado encontrado (Codigo " + codigoBusca + "): " + chamadoBusca.getDescricao());
28 +         } else {
29 +             System.out.println("Chamado nao encontrado (Codigo " + codigoBusca + ")");
30 +         }
31 +     }

```

▼ 29 ■■■■■ src/sistemaChamados/armazenaChamado.java

... -0,0 +1,29 @@

```
1 + package sistemaChamados;
2 +
3 + import java.util.ArrayList;
4 + import java.util.List;
5 +
6 + public class armazenaChamado {
7 +     private List<Chamado> chamados;
8 +
9 +     public armazenaChamado() {
10 +         chamados = new ArrayList<>();
11 +     }
12 +
13 +     public void adicionarChamado(Chamado chamado) {
14 +         chamados.add(chamado);
15 +     }
16 +
17 +     public Chamado getChamado(int codigo) {
18 +         for (Chamado chamado : chamados) {
19 +             if (chamado.getCodigo() == codigo) {
20 +                 return chamado;
21 +             }
22 +         }
23 +         return null;
24 +     }
25 +
26 +     public List<Chamado> getChamados() {
27 +         return chamados;
28 +     }
29 + }
```

src/sistemaChamados/Chamado.java

@@ -1,20 +1,19 @@

1120

12

package sistemaChamados;

34

34

- public class Chamado {

53

53

- private int codigo;

34

4+

+ public class Chamado {

44

4+

+ private String tipoChamado;

65

5

private String descricao;

76

6

8-

9-

public Chamado(int codigo, String descricao) {

9-

10-

this.codigo = codigo;

7+

8+

public Chamado(String tipoChamado, String descricao) {

8+

9

this.tipoChamado = tipoChamado;

109

10

this.descricao = descricao;

1110

11

}

1211

12

13-

14-

public int getCodigo() {

14-

15-

return codigo;

14-

15-

return codigo;

12+

13+

public String getTipoChamado() {

13+

14

return tipoChamado;

1514

16

}

1615

17

1716

18

public String getDescricao() {

1817

19

return descricao;

1918

20

}

20-

21-

}

19+

20+

}

```
src/sistemaChamados/Main.java
... @@ -1,31 +1,12 @@
1 package sistemaChamados;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         armazenarChamado armazenarChamado = new armazenarChamado();
7         criarChamado criarChamado = new criarChamado(armazenarChamado);
8
9         Chamado chamado1 = new Chamado (1, "Problema no servidor");
10        Chamado chamado2 = new Chamado(2, "Erro de conexao");
11        Chamado chamado3 = new Chamado(3, "Aplicativo travado");
12
13        armazenarChamado.adicionarChamado(chamado1);
14        armazenarChamado.adicionarChamado(chamado2);
15        armazenarChamado.adicionarChamado(chamado3);
16
17        System.out.println("Lista de Chamados:");
18
19        for (Chamado chamado : armazenarChamado.getChamados()) {
20            System.out.println("Codigo: " + chamado.getCodigo());
21            System.out.println("Descricao: " + chamado.getDescricao());
22            System.out.println("-----");
23        }
24
25        int codigoBusca = 2;
26        Chamado chamadoBusca = armazenarChamado.getChamado(codigoBusca);
27        if (chamadoBusca != null) {
28            System.out.println("Chamado encontrado (Codigo " + codigoBusca + "): " + chamadoBusca.getDescricao());
29        } else {
30            System.out.println("Chamado nao encontrado (Codigo " + codigoBusca + ")");
31        }
32
33        criarChamado.criarChamado();
34        armazenarChamado.listarChamados();
35    }
36 }
```

src/sistemaChamados/armazenaChamado.java

@@ -14,16 +14,15 @@ public void adicionarChamado(Chamado chamado) {

14 chamados.add(chamado);

15 }

16

17 - public Chamado getChamado(int codigo) {

18 - for (Chamado chamado : chamados) {

19 - if (chamado.getCodigo() == codigo) {

20 - return chamado;

21 - }

22 - }

23 - return null;

24 - }

25 -

26 17 public List<Chamado> getChamados() {

27 18 return chamados;

28 19 }

29 - }

20 +

21 + public void listarChamados() {

22 + for (Chamado chamado : chamados) {

23 + System.out.println("Tipo: " + chamado.getTipoChamado());

24 + System.out.println("Descrição: " + chamado.getDescricao());

25 + System.out.println("-----");

26 + }

27 + }

28 + }

```
81 src/sistemaChamados/criarChamado.java
... @@ -0,0 +1,81 @@
1 + package sistemaChamados;
2 +
3 + import java.util.Scanner;
4 +
5 + public class criarChamado {
6 +
7 +     private armazenaChamado armazenaChamado;
8 +     private Scanner scanner;
9 +     private String tipoChamadoSelecionado;
10 +
11 +     public criarChamado(armazenaChamado armazenaChamado) {
12 +         this.armazenaChamado = armazenaChamado;
13 +         this.scanner = new Scanner(System.in);
14 +         this.tipoChamadoSelecionado = "Desconhecido";
15 +     }
16 +
17 +     public void criarChamado() {
18 +         System.out.println("Criar um chamado");
19 +         System.out.println("-----");
20 +
21 +         System.out.println("Selecione o tipo de chamado:");
22 +         System.out.println("1. Sistema Operacional");
23 +         System.out.println("2. Hardware");
24 +         System.out.println("3. Software");
25 +         System.out.println("4. Telefonia Fixa");
26 +         System.out.println("5. File Server");
27 +         System.out.println("6. Internet");
28 +         System.out.println("7. E-mail");
29 +         System.out.println("8. Telefonia Movel");
30 +         System.out.print("Opcao: ");
31 +
32 +         int opcao = scanner.nextInt();
33 +         scanner.nextLine(); // Consumir a quebra de linha
34 +
35 +         switch (opcao) {
36 +             case 1:
37 +                 tipoChamadoSelecionado = "Sistema Operacional";
38 +                 break;
39 +             case 2:
40 +                 tipoChamadoSelecionado = "Hardware";
41 +                 break;
42 +             case 3:
```

```

43 +         tipoChamadoSelecionado = "Software";
44 +         break;
45 +     case 4:
46 +         tipoChamadoSelecionado = "Telefonia Fixa";
47 +         break;
48 +     case 5:
49 +         tipoChamadoSelecionado = "File Server";
50 +         break;
51 +     case 6:
52 +         tipoChamadoSelecionado = "Internet";
53 +         break;
54 +     case 7:
55 +         tipoChamadoSelecionado = "E-mail";
56 +         break;
57 +     case 8:
58 +         tipoChamadoSelecionado = "Telefonia Móvel";
59 +         break;
60 +     default:
61 +         System.out.println("Opção inválida. O chamado será criado com
        tipo desconhecido.");
62 +         tipoChamadoSelecionado = "Desconhecido";
63 +         break;
64 +     }
65 +
66 +     System.out.print("Digite a descricao do chamado: ");
67 +     String descricao = scanner.nextLine();
68 +
69 +     Chamado chamado = new Chamado(tipoChamadoSelecionado, descricao);
70 +     armazenaChamado.adicionarChamado(chamado);
71 +
72 +     System.out.println("Chamado criado com sucesso!");
73 +     System.out.println("Tipo: " + chamado.getTipoChamado());
74 +     System.out.println("Descricao: " + chamado.getDescricao());
75 +     System.out.println();
76 + }
77 +
78 + public String getTipoChamadoSelecionado() {
79 +     return tipoChamadoSelecionado;
80 + }
81 + }

```

Interface + CSV

Nessa release foi implantada a interface gráfica do sistema além do arquivo CSV onde fica armazenado os chamados para melhor administração.

```
7 chamados.csv
... @@ -0,0 +1,7 @@
1 +
2 + Hardware,teste
3 + Software,teste
4 + Telefonia Movei,teste
5 + Telefonia Fixa,teste
6 + Internet,teste
7 + File Server,teste



55 src/sistemaChamados/ArmazenaChamado.java
... @@ -0,0 +1,55 @@
1 + package sistemaChamados;
2 +
3 + import java.io.BufferedReader;
4 + import java.io.FileReader;
5 + import java.io.FileWriter;
6 + import java.io.IOException;
7 + import java.io.PrintWriter;
8 + import java.util.ArrayList;
9 + import java.util.List;
10 +
11 + public class ArmazenaChamado {
12 +     private static final String FILENAME = "chamados.csv";
13 +
14 +     public static void salvarChamado(Chamado chamado) {
15 +         try (PrintWriter writer = new PrintWriter(new FileWriter(FILENAME,
16 +             true))) {
17 +             String linha = chamado.getTipo() + "," + chamado.getDescricao();
18 +             writer.println(linha);
19 +         } catch (IOException e) {
20 +             e.printStackTrace();
21 +         }
22 +     }
23 +
24 +     public static List<Chamado> carregarChamados() {
25 +         List<Chamado> chamados = new ArrayList<>();
26 +         try (BufferedReader reader = new BufferedReader(new
27             FileReader(FILENAME))) {
```



```

27 +         String linha;
28 +
29 +         while ((linha = reader.readLine()) != null) {
30 +             String[] campos = linha.split(",");
31 +             if (campos.length == 2) {
32 +                 String tipo = campos[0];
33 +                 String descricao = campos[1];
34 +                 Chamado chamado = new Chamado(tipo, descricao);
35 +                 chamados.add(chamado);
36 +             }
37 +         }
38 +     } catch (IOException e) {
39 +         e.printStackTrace();
40 +     }
41 +
42 +     return chamados;
43 + }
44 +
45 + public static void salvarChamados(List<Chamado> chamados) {
46 +     try (PrintWriter writer = new PrintWriter(new FileWriter(FILENAME))) {
47 +         for (Chamado chamado : chamados) {
48 +             String linha = chamado.getTipo() + "," + chamado.getDescricao();
49 +             writer.println(linha);
50 +         }
51 +     } catch (IOException e) {
52 +         e.printStackTrace();
53 +     }
54 + }
55 + }

```

✓ 12  src/sistemaChamados/Chamado.java 

... @@ -1,19 +1,19 @@

1	1	package sistemaChamados;
2	2	
3	3	public class Chamado {
4	-	private String tipoChamado;
4	+	private String tipo;
5	5	private String descricao;
6	6	
7	-	public Chamado(String tipoChamado, String descricao) {
8	-	this.tipoChamado = tipoChamado;
7	+	public Chamado(String tipo, String descricao) {
8	+	this.tipo = tipo;
9	9	this.descricao = descricao;
10	10	}
11	11	
12	-	public String getTipoChamado() {
13	-	return tipoChamado;
12	+	public String getTipo() {
13	+	return tipo;

```
13 + return tipo;
14 14 }
15 15
16 16 public String getDescricao() {
17 17 return descricao;
18 18 }
19 - }
19 + }
```

92 src/sistemaChamados/CriarChamado.java

```
... @@ -0,0 +1,92 @@
1 + package sistemaChamados;
2 +
3 + import javax.swing.*;
4 + import java.awt.*;
5 + import java.awt.event.ActionEvent;
6 + import java.awt.event.ActionListener;
7 + import java.util.Arrays;
8 + import java.util.List;
9 +
9 +
10 + public class CriarChamado {
11 +     private JFrame frame;
12 +     private JComboBox<String> tipoComboBox;
13 +     private JTextArea descricaoTextArea;
14 +
15 +     public CriarChamado() {
16 +         initialize();
17 +     }
18 +
19 +     private void initialize() {
20 +         frame = new JFrame();
21 +         frame.setBounds(100, 100, 450, 300);
22 +         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
23 +         frame.getContentPane().setLayout(new BorderLayout(0, 0));
24 +
25 +         JPanel panel = new JPanel();
26 +         frame.getContentPane().add(panel, BorderLayout.CENTER);
27 +         panel.setLayout(new GridBagLayout());
28 +
29 +         JLabel lblNewLabel = new JLabel("Tipo de Chamado:");
30 +         GridBagConstraints gbc_lblNewLabel = new GridBagConstraints();
```

```

31 +         gbc_lblNewLabel.anchor = GridBagConstraints.WEST;
32 +         gbc_lblNewLabel.insets = new Insets(0, 0, 5, 5);
33 +         gbc_lblNewLabel.gridx = 0;
34 +         gbc_lblNewLabel.gridy = 0;
35 +         panel.add(lblNewLabel, gbc_lblNewLabel);
36 +
37 +         tipoComboBox = new JComboBox<>();
38 +         GridBagConstraints gbc_tipoComboBox = new GridBagConstraints();
39 +         gbc_tipoComboBox.fill = GridBagConstraints.HORIZONTAL;
40 +         gbc_tipoComboBox.insets = new Insets(0, 0, 5, 0);
41 +         gbc_tipoComboBox.gridx = 1;
42 +         gbc_tipoComboBox.gridy = 0;
43 +         panel.add(tipoComboBox, gbc_tipoComboBox);
44 +
45 +         JLabel lblNewLabel_1 = new JLabel("Descrição:");
46 +         GridBagConstraints gbc_lblNewLabel_1 = new GridBagConstraints();
47 +         gbc_lblNewLabel_1.anchor = GridBagConstraints.WEST;
48 +         gbc_lblNewLabel_1.insets = new Insets(0, 0, 5, 5);
49 +         gbc_lblNewLabel_1.gridx = 0;
50 +         gbc_lblNewLabel_1.gridy = 1;
51 +         panel.add(lblNewLabel_1, gbc_lblNewLabel_1);
52 +
53 +         descricaoTextArea = new JTextArea();
54 +         descricaoTextArea.setRows(5); // Define o número de linhas exibidas
55 +         descricaoTextArea.setLineWrap(true); // Habilita a quebra de linha
56 +         JScrollPane scrollPane = new JScrollPane(descricaoTextArea); // Adiciona
57 +         uma barra de rolagem
58 +         GridBagConstraints gbc_descricaoTextArea = new GridBagConstraints();
59 +         gbc_descricaoTextArea.insets = new Insets(0, 0, 5, 0);
60 +         gbc_descricaoTextArea.fill = GridBagConstraints.BOTH;
61 +         gbc_descricaoTextArea.gridx = 1;
62 +         gbc_descricaoTextArea.gridy = 1;
63 +         panel.add(scrollPane, gbc_descricaoTextArea);
64 +
65 +         JButton enviarButton = new JButton("Enviar");
66 +         enviarButton.addActionListener(new ActionListener() {
67 +             public void actionPerformed(ActionEvent e) {
68 +                 String tipo = (String) tipoComboBox.getSelectedItem();
69 +                 String descricao = descricaoTextArea.getText();
70 +
71 +                 Chamado chamado = new Chamado(tipo, descricao);

```

```

71 +             ArmazenaChamado.salvarChamado(chamado);
72 +
73 +             JOptionPane.showMessageDialog(frame, "Chamado aberto com
              sucesso!");
74 +             descricaoTextArea.setText("");
75 +         }
76 +     });
77 +     GridBagConstraints gbc_enviarButton = new GridBagConstraints();
78 +     gbc_enviarButton.gridx = 1;
79 +     gbc_enviarButton.gridy = 2;
80 +     panel.add(enviarButton, gbc_enviarButton);
81 +
82 +     List<String> tiposChamado = Arrays.asList("Hardware", "Software",
"Telefonia Movei",
83 +     "Telefonia Fixa", "Internet", "File Server");
84 +     for (String tipo : tiposChamado) {
85 +         tipoComboBox.addItem(tipo);
86 +     }
87 + }
88 +
89 + public void show() {
90 +     frame.setVisible(true);
91 + }
92 + }

```

▼ 14 ■■■ src/sistemaChamados/Main.java

```

...   ...   @@ -1,12 +1,16 @@
1     1     package sistemaChamados;
2     2
3     3     - public class Main {
4     4
5     5     + import java.util.List;
6     6
7     7     + public class Main {
8     8         public static void main(String[] args) {
9     9         -     armazenChamado armazenChamado = new armazenChamado();
10    10        -     criarChamado criarChamado = new criarChamado(armazenChamado);
11    11        +     // Carregar chamados existentes
12    12        +     List<Chamado> chamados = ArmazenaChamado.carregarChamados();
13    13
14    14        +     ArmazenaChamado.salvarChamados(chamados);
15    15
16    16        -     criarChamado.criarChamado();
17    17        -     armazenChamado.listarChamados();
18    18
19    19        +     // Iniciar a interface de abertura de chamados
20    20        +     CriarChamado criarChamado = new CriarChamado();
21    21        +     criarChamado.show();
22    22
23    23        }
24    24    }

```

Sistema de login

Essa atualização dá ao sistema a possibilidade de logar no sistema antes de enviar o chamado.

```
95 src/sistemaChamados/Main.java
... @@ -1,16 +1,97 @@
1 1 package sistemaChamados;
2 2
3 - import java.util.List;
4 + import javax.swing.*;
5 + import java.awt.*;
6 + import java.awt.event.ActionEvent;
7 + import java.awt.event.ActionListener;
8 7
9 8 public class Main {
10 - public static void main(String[] args) {
11 - // Carregar chamados existentes
12 - List<Chamado> chamados = ArmazenaChamado.carregarChamados();
13 + private JFrame frame;
14 + private JTextField usernameTextField;
15 + private JPasswordField passwordField;
16 +
17 + public Main() {
18 + initialize();
19 + }
20 +
21 + private void initialize() {
22 + frame = new JFrame();
23 + frame.setBounds(100, 100, 300, 200);
24 + frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
25 + frame.getContentPane().setLayout(new BorderLayout(0, 0));
26 +
27 + JPanel panel = new JPanel();
28 + frame.getContentPane().add(panel, BorderLayout.CENTER);
29 + panel.setLayout(new GridBagLayout());
30 +
31 + JLabel lblNewLabel = new JLabel("Usuário:");
32 + GridBagConstraints gbc_lblNewLabel = new GridBagConstraints();
33 + gbc_lblNewLabel.anchor = GridBagConstraints.WEST;
34 + gbc_lblNewLabel.insets = new Insets(0, 0, 5, 5);
35 + gbc_lblNewLabel.gridx = 0;
36 + gbc_lblNewLabel.gridy = 0;
37 + panel.add(lblNewLabel, gbc_lblNewLabel);
38 +
39 + usernameTextField = new JTextField();
```

```

36 + GridBagConstraints gbc_usernameTextField = new GridBagConstraints();
37 + gbc_usernameTextField.insets = new Insets(0, 0, 5, 0);
38 + gbc_usernameTextField.fill = GridBagConstraints.HORIZONTAL;
39 + gbc_usernameTextField.gridx = 1;
40 + gbc_usernameTextField.gridy = 0;
41 + panel.add(usernameTextField, gbc_usernameTextField);
42 + usernameTextField.setColumns(10);
43 +
44 + JLabel lblNewLabel_1 = new JLabel("Senha:");
45 + GridBagConstraints gbc_lblNewLabel_1 = new GridBagConstraints();
46 + gbc_lblNewLabel_1.anchor = GridBagConstraints.WEST;
47 + gbc_lblNewLabel_1.insets = new Insets(0, 0, 0, 5);
48 + gbc_lblNewLabel_1.gridx = 0;
49 + gbc_lblNewLabel_1.gridy = 1;
50 + panel.add(lblNewLabel_1, gbc_lblNewLabel_1);
51 +
52 + passwordField = new JPasswordField();
53 + GridBagConstraints gbc_passwordField = new GridBagConstraints();
54 + gbc_passwordField.fill = GridBagConstraints.HORIZONTAL;
55 + gbc_passwordField.gridx = 1;
56 + gbc_passwordField.gridy = 1;
57 + panel.add(passwordField, gbc_passwordField);
9 58
10 - ArmazenaChamado.salvarChamados(chamados);
59 + JButton loginButton = new JButton("Entrar");
60 + loginButton.addActionListener(new ActionListener() {
61 +     public void actionPerformed(ActionEvent e) {
62 +         String username = usernameTextField.getText();
63 +         char[] password = passwordField.getPassword();
11 64
12 - // Iniciar a interface de abertura de chamados
65 +         if (username.equals("admin") && new
String(password).equals("admin")) {
66 +             JOptionPane.showMessageDialog(frame, "Admin logado com
sucesso!");
67 +             frame.dispose();
68 +             openTicketCreation();
69 +         } else if (username.equals("user") && new
String(password).equals("user")) {
70 +             JOptionPane.showMessageDialog(frame, "User logado com
sucesso!");
71 +             frame.dispose();
72 +             openTicketCreation();
73 +         } else {

```

```

74 +             JOptionPane.showMessageDialog(frame, "Usuário ou senha
      incorretos");
75 +         }
76 +
77 +         // Clear the password field
78 +         passwordField.setText("");
79 +     }
80 + });
81 + frame.getContentPane().add(loginButton, BorderLayout.SOUTH);
82 + }
83 +
84 + public void show() {
85 +     frame.setVisible(true);
86 + }
87 +
88 + private void openTicketCreation() {
13 89     CriarChamado criarChamado = new CriarChamado();
14 90     criarChamado.show();
15 91 }
16 - }
92 +
93 + public static void main(String[] args) {
94 +     Main login = new Main();
95 +     login.show();
96 + }
97 + }

```

Acessos e cadastros

Permite a criação de acessos e visualização de chamados abertos anteriormente.

```
src/sistemaChamados/CriarChamado.java

@@ -4,12 +4,14 @@
4 4 import java.awt.*;
5 5 import java.awt.event.ActionEvent;
6 6 import java.awt.event.ActionListener;
7 - import java.util.Arrays;
8 - import java.util.List;
9 + import java.io.FileWriter;
10 + import java.io.IOException;
11 + import java.io.PrintWriter;
12 + import java.time.LocalDateTime;

11
12 public class CriarChamado {
13     private JFrame frame;
14 - private JComboBox<String> tipoComboBox;
15 + private JTextField assuntoTextField;
16     private JTextArea descricaoTextArea;
17     public CriarChamado() {

@@ -18,72 +20,68 @@ public CriarChamado() {
18 20
19 21     private void initialize() {
20 22         frame = new JFrame();
21 - frame.setBounds(100, 100, 450, 300);
22 - frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
23 + frame.setBounds(100, 100, 400, 300);
24 + frame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
25     frame.getContentPane().setLayout(new BorderLayout(0, 0));
26
27     JPanel panel = new JPanel();
28     frame.getContentPane().add(panel, BorderLayout.CENTER);
29     panel.setLayout(new GridBagLayout());
30
31 - JLabel lblNewLabel = new JLabel("Tipo de Chamado:");
32 + JLabel lblNewLabel = new JLabel("Assunto:");
33     GridBagConstraints gbc_lblNewLabel = new GridBagConstraints();
34     gbc_lblNewLabel.anchor = GridBagConstraints.WEST;
35     gbc_lblNewLabel.insets = new Insets(0, 0, 5, 5);
36     gbc_lblNewLabel.gridx = 0;
```



```

34 36      gbc_lblNewLabel.gridy = 0;
35 37      panel.add(lblNewLabel, gbc_lblNewLabel);
36 38
37      -      tipoComboBox = new JComboBox<>();
38      -      GridBagConstraints gbc_tipoComboBox = new GridBagConstraints();
39      -      gbc_tipoComboBox.fill = GridBagConstraints.HORIZONTAL;
40      -      gbc_tipoComboBox.insets = new Insets(0, 0, 5, 0);
41      -      gbc_tipoComboBox.gridx = 1;
42      -      gbc_tipoComboBox.gridy = 0;
43      -      panel.add(tipoComboBox, gbc_tipoComboBox);
44 47
45 48      JLabel lblNewLabel_1 = new JLabel("Descrição:");
46 49      GridBagConstraints gbc_lblNewLabel_1 = new GridBagConstraints();
47      -      gbc_lblNewLabel_1.anchor = GridBagConstraints.WEST;
48      -      gbc_lblNewLabel_1.insets = new Insets(0, 0, 5, 5);
49 52
50 53      gbc_lblNewLabel_1.anchor = GridBagConstraints.NORTHWEST;
51 54      gbc_lblNewLabel_1.insets = new Insets(0, 0, 0, 5);
52 55      gbc_lblNewLabel_1.gridx = 0;
53 56      gbc_lblNewLabel_1.gridy = 1;
54 57      panel.add(lblNewLabel_1, gbc_lblNewLabel_1);
55 58
56 59      descricaoTextArea = new JTextArea();
57 60      -      descricaoTextArea.setRows(5); // Define o número de linhas exibidas
58 61      -      descricaoTextArea.setLineWrap(true); // Habilita a quebra de linha
59 62      automática
60 63      -      JScrollPane scrollPane = new JScrollPane(descricaoTextArea); // Adiciona
61 64      uma barra de rolagem
62 65      GridBagConstraints gbc_descricaoTextArea = new GridBagConstraints();
63 66      -      gbc_descricaoTextArea.insets = new Insets(0, 0, 5, 0);
64 67      gbc_descricaoTextArea.fill = GridBagConstraints.BOTH;
65 68      gbc_descricaoTextArea.gridx = 1;
66 69      gbc_descricaoTextArea.gridy = 1;
67 70      -      panel.add(scrollPane, gbc_descricaoTextArea);
68 71      +      panel.add(new JScrollPane(descricaoTextArea), gbc_descricaoTextArea);

```

```


63 62
64 - JButton enviarButton = new JButton("Enviar");
65 - enviarButton.addActionListener(new ActionListener() {
63 + JButton criarButton = new JButton("Criar");
64 + criarButton.addActionListener(new ActionListener() {
66 65 public void actionPerformed(ActionEvent e) {
67 - String tipo = (String) tipoComboBox.getSelectedItem();
66 + String assunto = assuntoTextField.getText();
68 67 String descricao = descricaoTextArea.getText();
68 + String dataHora = LocalDateTime.now().toString();
69 69
70 - Chamado chamado = new Chamado(tipo, descricao);
71 - ArmazenaChamado.salvarChamado(chamado);
70 + // Salva as informações em "chamados.csv"
71 + try (PrintWriter writer = new PrintWriter(new
FileWriter("chamados.csv", true))) {
72 + writer.println("Assunto: " + assunto);
73 + writer.println("Descrição: " + descricao);
74 + writer.println("Data/Hora: " + dataHora);
75 + writer.println();
76 + } catch (IOException ex) {
77 + ex.printStackTrace();
78 + }
72 79
73 - JOptionPane.showMessageDialog(frame, "Chamado aberto com
sucesso!");
74 - descricaoTextArea.setText("");
80 + JOptionPane.showMessageDialog(frame, "Chamado criado com
sucesso!");
81 + frame.dispose();
75 82 }
76 83 });
77 - GridBagConstraints gbc_enviarButton = new GridBagConstraints();
78 - gbc_enviarButton.gridx = 1;
79 - gbc_enviarButton.gridy = 2;
80 - panel.add(enviarButton, gbc_enviarButton);
81 -
82 - List<String> tiposChamado = Arrays.asList("Hardware", "Software",
"Telefonia Movei",
83 - "Telefonia Fixa", "Internet", "File Server");
84 - for (String tipo : tiposChamado) {
85 - tipoComboBox.addItem(tipo);
86 - }

```

```

84 + frame.getContentPane().add(criarButton, BorderLayout.SOUTH);
87 85 }
88 86
89 87 public void show() {
-----
↓
src/sistemaChamados/Main.java
@@ -4,6 +4,9 @@
4 4 import java.awt.*;
5 5 import java.awt.event.ActionEvent;
6 6 import java.awt.event.ActionListener;
7 + import java.io.*;
8 + import java.util.ArrayList;
9 + import java.util.List;
7 10
8 11 public class Main {
9 12 private JFrame frame;
-----
@@ -56,7 +59,7 @@ private void initialize() {
56 59 gbc_passwordField.gridy = 1;
57 60 panel.add(passwordField, gbc_passwordField);
58 61
59 - JButton loginButton = new JButton("Entrar");
62 + JButton loginButton = new JButton("Enviar");
60 63 loginButton.addActionListener(new ActionListener() {
61 64     public void actionPerformed(ActionEvent e) {
62 65         String username = usernameTextField.getText();
-----
@@ -65,16 +68,16 @@ public void actionPerformed(ActionEvent e) {
65 68         if (username.equals("admin") && new
String(password).equals("admin")) {
66 69             JOptionPane.showMessageDialog(frame, "Admin logado com
sucesso!");
67 70             frame.dispose();
68 -             openTicketCreation();
71 +             openAdminOptions();
69 72         } else if (username.equals("user") && new
String(password).equals("user")) {
70 73             JOptionPane.showMessageDialog(frame, "User logado com
sucesso!");
71 74             frame.dispose();

```

72	-	<code>openTicketCreation();</code>
75	+	<code>openUserOptions();</code>
73	76	<code>} else {</code>
74	77	<code>JOptionPane.showMessageDialog(frame, "Usuário ou senha</code>
75	78	<code>incorretos");</code>
76	79	<code>}</code>
77	-	<code>// Clear the password field</code>
80	+	<code>// Limpa o textbox de senha.</code>
78	81	<code>passwordField.setText("");</code>
79	82	<code>}</code>
80	83	<code>});</code>
		<code>@@ -85,13 +88,170 @@ public void show() {</code>
85	88	<code>frame.setVisible(true);</code>
86	89	<code>}</code>
87	90	
91	+	<code>private void openAdminOptions() {</code>
92	+	<code>JFrame adminFrame = new JFrame();</code>
93	+	<code>adminFrame.setBounds(100, 100, 400, 300);</code>
94	+	<code>adminFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);</code>
95	+	<code>adminFrame.getContentPane().setLayout(new BorderLayout(0, 0));</code>
96	+	
97	+	<code>JPanel panel = new JPanel();</code>
98	+	<code>adminFrame.getContentPane().add(panel, BorderLayout.CENTER);</code>
99	+	<code>panel.setLayout(new GridBagLayout());</code>
100	+	
101	+	<code>JButton verificarChamadosButton = new JButton("Verificar chamados</code>
		<code>abertos");</code>
102	+	<code>verificarChamadosButton.addActionListener(new ActionListener() {</code>
103	+	<code>public void actionPerformed(ActionEvent e) {</code>
104	+	<code>List<String> chamadosAbertos = readChamadosCSV();</code>
105	+	<code>String chamadosString = String.join("\n", chamadosAbertos);</code>
106	+	<code>JOptionPane.showMessageDialog(adminFrame, "Chamados abertos:\n" +</code>
		<code>chamadosString);</code>
107	+	<code>}</code>
108	+	<code>});</code>
109	+	<code>GridBagConstraints gbc_verificarChamadosButton = new</code>
		<code>GridBagConstraints();</code>
110	+	<code>gbc_verificarChamadosButton.insets = new Insets(0, 0, 5, 0);</code>
111	+	<code>gbc_verificarChamadosButton.gridx = 0;</code>
112	+	<code>gbc_verificarChamadosButton.gridy = 0;</code>
113	+	<code>panel.add(verificarChamadosButton, gbc_verificarChamadosButton);</code>

```

114 +
115 +         JButton cadastrarMembrosButton = new JButton("Cadastrar membros de
           equipe");
116 +         cadastrarMembrosButton.addActionListener(new ActionListener() {
117 +             public void actionPerformed(ActionEvent e) {
118 +                 String nomeCompleto = JOptionPane.showInputDialog(adminFrame,
           "Digite o nome completo:");
119 +                 String re = JOptionPane.showInputDialog(adminFrame, "Digite o RE
           do funcionário:");
120 +                 String nomeUsuario = JOptionPane.showInputDialog(adminFrame,
           "Digite o nome de usuário:");
121 +                 String senhaAcesso = JOptionPane.showInputDialog(adminFrame,
           "Digite a senha de acesso:");
122 +
123 +                 // Salvar as informações em "usuarios.csv"
124 +
125 +                 JOptionPane.showMessageDialog(adminFrame, "Membro de equipe
           cadastrado com sucesso!");
126 +             }
127 +         });
128 +
129 +         GridBagConstraints gbc_cadastrarMembrosButton = new GridBagConstraints();
130 +         gbc_cadastrarMembrosButton.insets = new Insets(0, 0, 5, 0);
131 +         gbc_cadastrarMembrosButton.gridx = 0;
132 +         gbc_cadastrarMembrosButton.gridy = 1;
133 +         panel.add(cadastrarMembrosButton, gbc_cadastrarMembrosButton);
134 +
135 +         JButton cadastrarClientesButton = new JButton("Cadastrar clientes");
136 +         cadastrarClientesButton.addActionListener(new ActionListener() {
137 +             public void actionPerformed(ActionEvent e) {
138 +                 String nomeCliente = JOptionPane.showInputDialog(adminFrame,
           "Digite o nome do cliente:");
139 +                 String cnpj = JOptionPane.showInputDialog(adminFrame, "Digite o
           CNPJ do cliente:");
140 +                 String nomeUsuario = JOptionPane.showInputDialog(adminFrame,
           "Digite o nome de usuário:");
141 +                 String senhaAcesso = JOptionPane.showInputDialog(adminFrame,
           "Digite a senha de acesso:");
142 +
143 +                 // Salvar as informações em "clientes.csv"
144 +
145 +                 JOptionPane.showMessageDialog(adminFrame, "Cliente cadastrado com
           sucesso!");

```

```

145 +         }
146 +     });
147 +     GridBagConstraints gbc_cadastrarClientesButton = new
    GridBagConstraints();
148 +     gbc_cadastrarClientesButton.insets = new Insets(0, 0, 5, 0);
149 +     gbc_cadastrarClientesButton.gridx = 0;
150 +     gbc_cadastrarClientesButton.gridy = 2;
151 +     panel.add(cadastrarClientesButton, gbc_cadastrarClientesButton);
152 +
153 +     JButton abrirChamadoButton = new JButton("Abrir chamado");
154 +     abrirChamadoButton.addActionListener(new ActionListener() {
155 +         public void actionPerformed(ActionEvent e) {
156 +             openTicketCreation();
157 +         }
158 +     });
159 +     GridBagConstraints gbc_abrirChamadoButton = new GridBagConstraints();
160 +     gbc_abrirChamadoButton.insets = new Insets(0, 0, 5, 0);
161 +     gbc_abrirChamadoButton.gridx = 0;
162 +     gbc_abrirChamadoButton.gridy = 3;
163 +     panel.add(abrirChamadoButton, gbc_abrirChamadoButton);
164 +
165 +     JButton desconectarButton = new JButton("Desconectar");
166 +     desconectarButton.addActionListener(new ActionListener() {
167 +         public void actionPerformed(ActionEvent e) {
168 +             adminFrame.dispose();
169 +             Main login = new Main();
170 +             login.show();
171 +         }
172 +     });
173 +     GridBagConstraints gbc_desconectarButton = new GridBagConstraints();
174 +     gbc_desconectarButton.gridx = 0;
175 +     gbc_desconectarButton.gridy = 4;
176 +     panel.add(desconectarButton, gbc_desconectarButton);
177 +
178 +     adminFrame.setVisible(true);
179 + }
180 +
181 + private void openUserOptions() {
182 +     JFrame userFrame = new JFrame();
183 +     userFrame.setBounds(100, 100, 400, 200);
184 +     userFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

```

```

185 +         userFrame.getContentPane().setLayout(new BorderLayout(0, 0));
186 +
187 +         JPanel panel = new JPanel();
188 +         userFrame.getContentPane().add(panel, BorderLayout.CENTER);
189 +         panel.setLayout(new GridBagLayout());
190 +
191 +         JButton abrirChamadoButton = new JButton("Abrir novo chamado");
192 +         abrirChamadoButton.addActionListener(new ActionListener() {
193 +             public void actionPerformed(ActionEvent e) {
194 +                 openTicketCreation();
195 +             }
196 +         });
197 +         GridBagConstraints gbc_abrirChamadoButton = new GridBagConstraints();
198 +         gbc_abrirChamadoButton.insets = new Insets(0, 0, 5, 0);
199 +         gbc_abrirChamadoButton.gridx = 0;
200 +         gbc_abrirChamadoButton.gridy = 0;
201 +         panel.add(abrirChamadoButton, gbc_abrirChamadoButton);
202 +
203 +         JButton verificarStatusButton = new JButton("Verificar status de
chamado");
204 +         verificarStatusButton.addActionListener(new ActionListener() {
205 +             public void actionPerformed(ActionEvent e) {
206 +                 List<String> chamadosUsuario = readChamadosCSV();
207 +                 String ultimoChamado = chamadosUsuario.get(chamadosUsuario.size()
- 1);
208 +                 JOptionPane.showMessageDialog(userFrame, "Último chamado
aberto:\n" + ultimoChamado);
209 +             }
210 +         });
211 +         GridBagConstraints gbc_verificarStatusButton = new GridBagConstraints();
212 +         gbc_verificarStatusButton.insets = new Insets(0, 0, 5, 0);
213 +         gbc_verificarStatusButton.gridx = 0;
214 +         gbc_verificarStatusButton.gridy = 1;
215 +         panel.add(verificarStatusButton, gbc_verificarStatusButton);
216 +
217 +         JButton desconectarButton = new JButton("Desconectar");
218 +         desconectarButton.addActionListener(new ActionListener() {
219 +             public void actionPerformed(ActionEvent e) {
220 +                 userFrame.dispose();
221 +                 Main login = new Main();
222 +                 login.show();

```

```

223 +         }
224 +     });
225 +     GridBagConstraints gbc_desconectarButton = new GridBagConstraints();
226 +     gbc_desconectarButton.gridx = 0;
227 +     gbc_desconectarButton.gridy = 2;
228 +     panel.add(desconectarButton, gbc_desconectarButton);
229 +
230 +     userFrame.setVisible(true);
231 + }
232 +
88 233     private void openTicketCreation() {
89 234         CriarChamado criarChamado = new CriarChamado();
90 235         criarChamado.show();
91 236     }
92 237
238 +     private List<String> readChamadosCSV() {
239 +         List<String> chamados = new ArrayList<>();
240 +
241 +         try (BufferedReader reader = new BufferedReader(new
242 +             FileReader("chamados.csv"))) {
243 +             String line;
244 +             while ((line = reader.readLine()) != null) {
245 +                 chamados.add(line);
246 +             }
247 +             } catch (IOException e) {
248 +                 e.printStackTrace();
249 +             }
250 +             return chamados;
251 +         }
252 +
93 253     public static void main(String[] args) {
94 254         Main login = new Main();
95 255         login.show();
96 256     }
97 - }
    257 + }

```


Testes do sistema

Teste classe “Chamado”

The screenshot displays an IDE window titled "ChamadoNGTest.java" with the following Java code:

```
1 package sistemaChamados;
2
3 import com.opencsv.exceptions.CsvValidationException;
4 import org.testng.Assert;
5 import org.testng.annotations.BeforeClass;
6 import org.testng.annotations.Test;
7
8 import java.util.ArrayList;
9 import java.util.List;
10
11 public class ChamadoNGTest {
12     private List<Chamado> chamados;
13
14     @BeforeClass
15     public void setUp() {
16         chamados = new ArrayList<>();
17         chamados.add(new Chamado(id: 1, assunto: "Assunto 1", descricao: "Descrição 1", status: "Aberto", responsavel: "Responsável 1"));
18         chamados.add(new Chamado(id: 2, assunto: "Assunto 2", descricao: "Descrição 2", status: "Fechado", responsavel: "Responsável 2"));
19     }
20
21     @Test
22     public void testLerChamados() throws CsvValidationException {
23         List<Chamado> chamadosLidos = Chamado.lerChamados();
24         Assert.assertEquals(actual: chamadosLidos.size(), expected: chamados.size(), message: "O número de chamados lidos está incorreto.");
25
26         for (int i = 0; i < chamados.size(); i++) {
27             Chamado chamadoLido = chamadosLidos.get(index: i);
28             Chamado chamadoEsperado = chamados.get(index: i);
29
30             Assert.assertEquals(actual: chamadoLido.getId(), expected: chamadoEsperado.getId(), message: "O ID do chamado lido está incorreto.");
31             Assert.assertEquals(actual: chamadoLido.getAssunto(), expected: chamadoEsperado.getAssunto(), message: "O assunto do chamado lido está incorreto.");
32         }
33     }
34 }
```

Below the code editor, the "Test Results" tab is active, showing the following output:

```
com.mycompany:SistemadeChamadosA3:jar:1.0-SNAPSHOT (Unit) X
>> Tests passed: 100.00 %
>> Both tests passed. (0,352 s)
```

The test results indicate that all tests passed successfully.

Teste classe “Criar chamado”

The screenshot displays an IDE window with two main panes. The top pane shows the source code of a Java test class named `CriarChamadoNGTest.java`. The code is as follows:

```
1 package sistemaChamados;
2
3 import org.testng.annotations.Test;
4 import javax.swing.*;
5 import static org.testng.Assert.*;
6
7 public class CriarChamadoNGTest {
8
9     @Test
10     public void testCriarButtonActionPerformed() {
11         CriarChamado criarChamado = new CriarChamado();
12         criarChamado.show();
13
14         JTextField assuntoTextField = criarChamado.getAssuntoTextField();
15         JTextArea descricaoTextArea = criarChamado.getDescricaoTextArea();
16         JButton criarButton = criarChamado.getCriarButton();
17
18         SwingUtilities.invokeLater(() -> {
19             assuntoTextField.setText("Assunto do chamado");
20             descricaoTextArea.setText("Descrição do chamado");
21
22             criarButton.doClick();
23
24             JOptionPane pane = (JOptionPane) JOptionPane.getRootFrame().getComponent(0);
25             String message = (String) pane.getMessage();
26             assertEquals("Chamado criado com sucesso!", message);
27
28             criarChamado.getFrame().dispose();
29         });
30     }
31 }
```

The bottom pane shows the Test Results and Output. The Test Results tab indicates that all tests passed (100.00 %). The Output tab shows the message "The test passed. (0,6 s)".

Teste classe “Usuario”

The screenshot displays an IDE window titled 'UsuarioNGTest.java'. The code defines a test class 'UsuarioNGTest' with two test methods: 'lerUsuariosTest()' and 'gravarUsuariosTest()'. The first method tests the 'lerUsuarios()' method by comparing a list of expected users with the actual list returned. The second method tests the 'gravarUsuarios()' method. The test results panel at the bottom shows that both tests passed successfully.

```
1 package sistemaChamados;
2
3 import org.testng.annotations.Test;
4 import java.util.ArrayList;
5 import java.util.List;
6 import static org.testng.Assert.assertEquals;
7
8 public class UsuarioNGTest {
9
10     @Test
11     public void lerUsuariosTest() {
12         List<Usuario> usuariosEsperados = new ArrayList<>();
13         usuariosEsperados.add(new Usuario(nome: "João Silva", registro: "123", username: "joao.silva", senha: "senha123"));
14         usuariosEsperados.add(new Usuario(nome: "Maria Santos", registro: "456", username: "maria.santos", senha: "senha456"));
15
16         List<Usuario> usuariosLidos = Usuario.lerUsuarios();
17
18         assertEquals(actual: usuariosLidos.size(), expected: usuariosEsperados.size());
19         for (int i = 0; i < usuariosLidos.size(); i++) {
20             Usuario usuarioLido = usuariosLidos.get(i);
21             Usuario usuarioEsperado = usuariosEsperados.get(i);
22
23             assertEquals(actual: usuarioLido.getNome(), expected: usuarioEsperado.getNome());
24             assertEquals(actual: usuarioLido.getRegistro(), expected: usuarioEsperado.getRegistro());
25             assertEquals(actual: usuarioLido.getUsername(), expected: usuarioEsperado.getUsername());
26             assertEquals(actual: usuarioLido.getSenha(), expected: usuarioEsperado.getSenha());
27         }
28     }
29
30     @Test
31     public void gravarUsuariosTest() {
32         List<Usuario> usuarios = new ArrayList<>();
33     }
```

Test Results × Output

com.mycompany:SistemadeChamadosA3jan1.0-SNAPSHOT (Unit) ×

Tests passed: 100.00%

Both tests passed. (0.333 s)

Teste classe “Main”

The screenshot displays an IDE window with two tabs: 'MainNGTest.java' and 'Test Results'. The 'MainNGTest.java' tab is active, showing the following code:

```
1 package sistemaChamados;
2
3 import org.testng.Assert;
4 import org.testng.annotations.*;
5
6 import javax.swing.JButton;
7
8 public class MainNGTest {
9     private Main main;
10
11     @BeforeMethod
12     public void setUp() {
13         main = new Main();
14     }
15
16     @Test
17     public void testLoginWithAdminCredentials() {
18         // Arrange
19         main.usernameTextField.setText("admin");
20         main.passwordField.setText("admin");
21         JButton loginButton = (JButton) main.frame.getContentPane().getComponent(1);
22
23         // Act
24         loginButton.doClick();
25
26         // Assert
27         // Assert the admin frame is opened or any other expected behavior
28         Assert.assertEquals(actual: main.frame.isVisible(), expected: false);
29     }
30
31     @Test
32     public void testLoginWithUserCredentials() {
33         // Arrange
34         main.usernameTextField.setText("user");
35         main.passwordField.setText("user");
36         JButton loginButton = (JButton) main.frame.getContentPane().getComponent(1);
37
38         // Act
39         loginButton.doClick();
40
41         // Assert
42         // Assert the user frame is opened or any other expected behavior
43         Assert.assertEquals(actual: main.frame.isVisible(), expected: true);
44     }
45 }
```

The 'Test Results' tab is also active, showing the following output:

```
com.mycompany.SistemaChamadosA3jan1.0-SNAPSHOT (Unit) X
Tests passed: 66.67 %
2 tests passed, 1 test failed. (3.279 s)
  sistemaChamados.MainNGTest Failed
    testLoginWithInvalidCredentials Failed
```

The test results indicate that 2 tests passed and 1 test failed. The failed test is 'testLoginWithInvalidCredentials' in the 'sistemaChamados.MainNGTest' class.