

Henrique Ferreira Cardoso ----- RM 552325

Lucas de Lavor Andrade ----- RM 552426

Matheus da Silva Ferreira ----- RM552431

VICTOR FUSCO MARTINS ----- RM552399

Grupo – DEDSEC

Porto Chamados

FIAP

2023

SUMÁRIO

OBJETIVO E ESCOPO DO PROJETO.....	3
BREVE DESCRIÇÃO DAS FUNCIONALIDADES.....	4
PROTÓTIPO.....	5
PROCEDIMENTOS PARA RODAR A APLICAÇÃO.....	8
MODELO DO BANDO DE DADOS.....	9

OBJETIVO E ESCOPO DO PROJETO:

O objetivo é desenvolver uma aplicação que consiga de forma autônoma escolher o melhor guincho para a situação apresentada pelo usuário, trazendo a resposta mais assertiva e funcional para o caso. Nossa aplicação pegará dados do veículo e da situação em que se encontra e calculará, utilizando estruturas de decisões e validando diversas variáveis, como peso, altura, largura e variáveis de ambiente (está funcionando, está tombado, está em um subsolo etc.), tendo feito essa decisão, consultará qual guincho se enquadra na situação e apresentará ao usuário qual o guincho correto para a extração do veículo quebrado, sendo assertivo e trazendo a resposta o mais rápido possível. Trazendo essa aplicação completa até o final do Sprint 4.

BREVE DESCRIÇÃO DAS FUNCIONALIDADES:

Coletar dados: Através de um formulário que será preenchido pelo funcionário, conseguirá coletar dados da situação e do veículo em questão.

Estrutura de decisão: Através das informações fornecidas pelo funcionário, ativará a nossa estrutura de decisão que definirá o guincho, após calcular todos os dados e interpretar qual o melhor guincho.

PROTÓTIPO:

```
public class Teste {
    public static void main(String[] args) {
        //Usuario
        Chamado abreChamado = new Chamado(123321);
        Usuario usuario = new Usuario("Rick", "54915362884", 321123, "Rickzin1200", "RickOnFiap@123");
        Localizacao localizacao = new Localizacao("Avenida Lins", "São Paulo", "IDK", "proximo a fiap, onde vende churros");
        Apolice apolice = new Apolice(12321);
        Veiculo veiculo = new Veiculo("FIAT", "Uno", (short) 1994, "EAME-213", "idk");

        //Funcionario
        Solicitudacao abreSolicitudacao = new Solicitudacao(132231);
        Funcionario func = new Funcionario("Henrique213", "opsfugi", "Henrique", 5135);
        Guinchos guincho = new Guinchos("JOWD-912", "Guincho de Reboque", 15152);
        Mecanico mecanico = new Mecanico("seu Jorge", 6146);
        Localizacao localizacaoMecanico = new Localizacao("Aqui", "perto", "de", "voce");
    }
}
```

Aqui é onde entregará os dados fornecidos para o sistema, através de construtores.

Demonstração 1 – Construtores pré setados para uma fácil visualização.

```
// Método rastrearLocalizacao utilizado abaixo
try {
    //Digite o ID do mecânico para o sistema funcionar da maneira correta. Caso incorreta, irá entrar no else ou nas exceptions.
    int id;
    id = Integer.parseInt(JOptionPane.showInputDialog("Coloque aqui o id desejado para o rastreo."));
    if (id == mecanico.getId()) {
        System.out.println(localizacaoMecanico.rastrearLocalizacao(id));
    } else {
        System.err.println("ID não encontrado!");
    }
} catch (NumberFormatException e) {
    System.err.println("ID inválido!");
    System.err.println(e.getMessage());
} catch (Exception e) {
    System.err.println("Index inválido!");
    System.err.println(e.getMessage());
}

// Método cadastrarFuncionario para manter uma lista dos logins e senhas.
func.cadastrarFuncionario("Henrique213", "opsfugi");

// Método solicitarMecanico, a localização do usuário é enviada para ele e retorna uma localização desejada.
System.out.println(func.solicitarMecanico(localizacao));

// Método loginFuncionario utilizado abaixo
String login;
String senha;
login = JOptionPane.showInputDialog("Coloque aqui o seu login: ");
senha = JOptionPane.showInputDialog("Coloque aqui a sua senha: ");
JOptionPane.showMessageDialog(null, func.loginFuncionario(login, senha));
```

Aqui será realizado o método "rastrearLocalizacao", onde você entrega o ID para o sistema, ele verifica se o ID é existente e retorna a localização, caso inexistente, retornará que não foi encontrado. Em casos de digitar algo inválido, entrará para as "Exceptions".

Aqui será realizado o método "cadastrarFuncionario", onde o Funcionario coloca seu login desejado e senha, sendo inserida em uma lista para os próximos métodos

Demonstração 2 – Métodos rastrearLocalizacao e cadastrarFuncionario.

FIAP

2023

```
// Método rastrearLocalizacao utilizado abaixo
try {
    // Digite o ID do mecânico para o sistema funcionar de maneira correta. Caso incorreta, irá entrar no else ou nas exceptions.
    int id;
    id = Integer.parseInt(JOptionPane.showInputDialog("Coloque aqui o id desejado para o rastreio."));
    if (id == mecanico.getId()) {
        System.out.println(localizacaoMecanico.rastrearLocalizacao(id));
    } else {
        System.err.println("ID não encontrado!");
    }
} catch (NumberFormatException e) {
    System.err.println("ID inválido!");
    System.err.println(e.getMessage());
} catch (Exception e) {
    System.err.println("Index inválido!");
    System.err.println(e.getMessage());
}

// Método cadastrarFuncionario para manter uma lista dos logins e senhas.
func.cadastrarFuncionario("Henrique213", "opsfugi");

// Método solicitarMecanico, a localização do usuário é enviada para ele e retorna uma confirmação.
System.out.println(func.solicitarMecanico(localizacao));

// Método loginFuncionario utilizado abaixo
String login;
String senha;
login = JOptionPane.showInputDialog("Coloque aqui o seu login: ");
senha = JOptionPane.showInputDialog("Coloque aqui a sua senha: ");
JOptionPane.showMessageDialog(null, func.loginFuncionario(login, senha));
```

Aqui será realizado o método "solicitarMecanico", onde o funcionário entrega a localização fornecida pelo usuário, e retorna uma confirmação de chamado realizado

Demonstração 3 - Método solicitarMecanico pré setado para fácil visualização.

```
// Método rastrearLocalizacao utilizado abaixo
try {
    // Digite o ID do mecânico para o sistema funcionar de maneira correta. Caso incorreta, irá entrar no else ou nas exceptions.
    int id;
    id = Integer.parseInt(JOptionPane.showInputDialog("Coloque aqui o id desejado para o rastreio."));
    if (id == mecanico.getId()) {
        System.out.println(localizacaoMecanico.rastrearLocalizacao(id));
    } else {
        System.err.println("ID não encontrado!");
    }
} catch (NumberFormatException e) {
    System.err.println("ID inválido!");
    System.err.println(e.getMessage());
} catch (Exception e) {
    System.err.println("Index inválido!");
    System.err.println(e.getMessage());
}

// Método cadastrarFuncionario para manter uma lista dos logins e senhas.
func.cadastrarFuncionario("Henrique213", "opsfugi");

// Método solicitarMecanico, a localização do usuário é enviada para ele e retorna uma confirmação.
System.out.println(func.solicitarMecanico(localizacao));

// Método loginFuncionario utilizado abaixo
String login;
String senha;
login = JOptionPane.showInputDialog("Coloque aqui o seu login: ");
senha = JOptionPane.showInputDialog("Coloque aqui a sua senha: ");
JOptionPane.showMessageDialog(null, func.loginFuncionario(login, senha));
```

Aqui será realizado o método "loginFuncionario", onde aparecerá na tela um input de login, e outro para colocar a senha. O sistema irá conferir se ambos estão corretos, realizando o seu login, ou retornando inválidos.

Demonstração 4 – Método loginFuncionario para verificar Login e Senha e login ou não

```
// Método preencherFormulario utilizado abaixo
try{
    func.preencherFormulario(JOptionPane.showInputDialog("Informe aqui o nome para o formulário : "), JOptionPane.showInputDialog("Informe aqui o data para o formulário : "), JOptionPane.showInputDialog("Informe aqui o peso para o formulário : "), JOptionPane.showInputDialog("Informe aqui a altura para o formulário : "), JOptionPane.showInputDialog("Informe aqui a largura para o formulário : "), JOptionPane.showInputDialog("Informe aqui o endereço para o formulário : "));
} catch (NumberFormatException e) {
    System.err.println("Valores informados inválidos!");
    System.err.println(e.getMessage());
} catch (Exception e) {
    System.err.println("Index inválidos!");
    System.err.println(e.getMessage());
}
```

Método preencherFormulario em execução, o funcionário preencherá o formulário com as devidas informações, sendo nome, data, peso, altura, largura e informações adicionais. Caso algum dado inválido, entrará para as Exceptions.

Demonstração 5 – Método preencherFormulario para informar dados relevantes.

PROCEDIMENTOS PARA RODAR A APLICAÇÃO:

Assim que abrir o projeto, haverá uma classe chamada “Teste”, com construtores já pré setados para facilitar. O primeiro método na ordem, será rastrearLocalizacao, para acompanhar se o mecânico já está perto. Para isso, você deverá digitar “6146” para conseguir rastreá-lo, pois foi o id definido nos construtores.

Segundo e terceiro métodos, cadastrarFuncionario e solicitarMecanico serão feitos de forma automática, retornarão apenas uma mensagem confirmando que deu certo.

Quarto método loginFuncionario, você colocará o login e senha que foram informados no “cadastrarFuncionario” que no caso foram Login:”Henrique213” e senha:”opsfugi”, retornando que foi logado, ou se não forem essas, retornará incorretos.

Quinto método preencherFormulario você preencherá de forma que quiser as tabelas, único requisito é onde está pedindo números, digitar números, pois caso não, entrará nas Exceptions.

