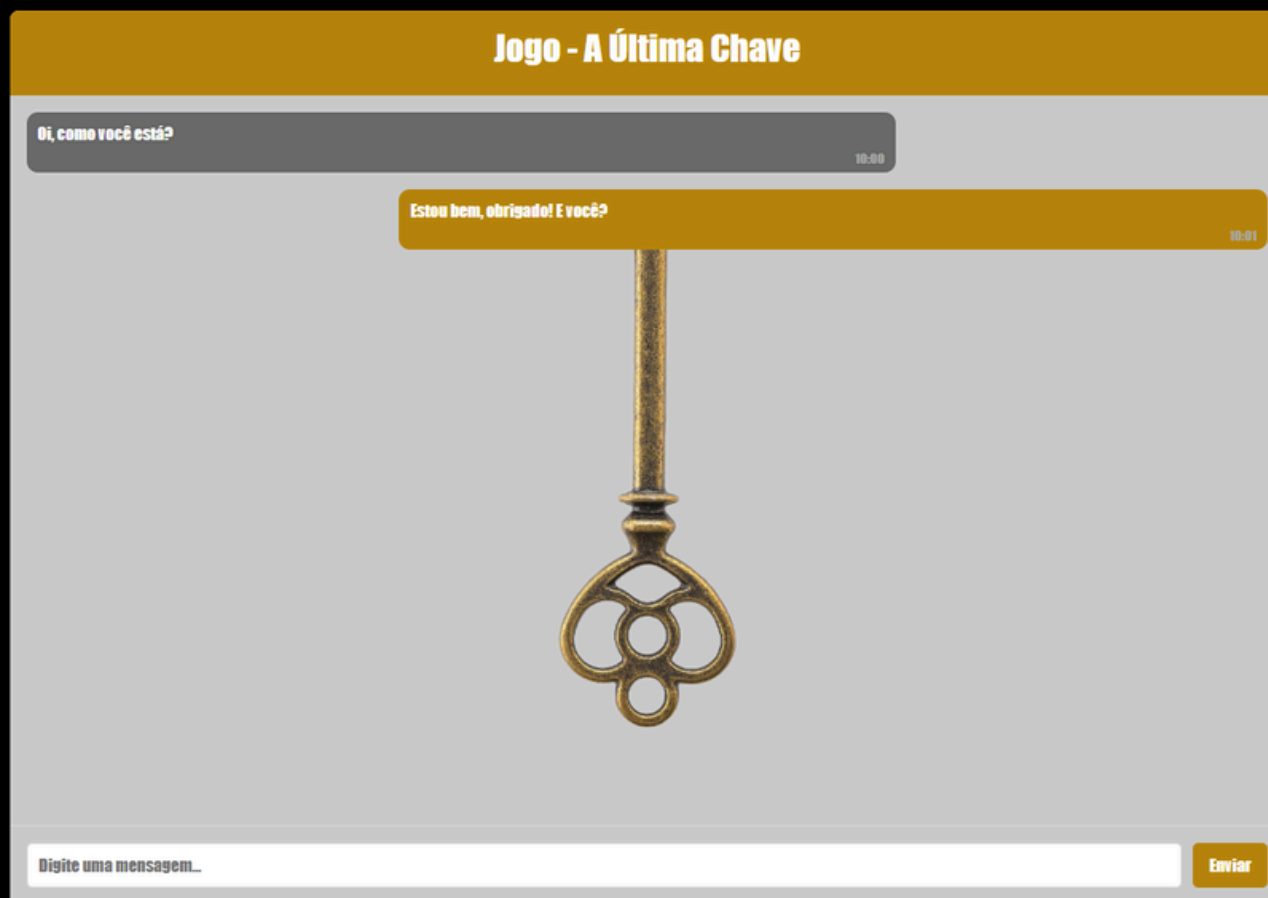




"A Última Chave"

X- Não conseguimos front-end

← Voltar






Comandos


Inventory(INVENTÁRIO),
Help(AJUDA), save(SALVAR),
Reset(RESETAR) e SAIR


```
case "USAR CHAVE NA PORTA":  
    if (room.useKey()) {  
        nextRoom();  
    }  
    break;  
case "INVENTÁRIO":  
    showInventory(); // Mostrar inventário  
    break;  
case "AJUDA":  
    showHelp(); // Exibir comandos de ajuda  
    break;  
case "SALVAR JOGO":  
    saveGame();  
    break;  
case "RESETAR PROGRESSO":  
    resetProgress();  
    break;  
case "SAIR":  
    gameRunning = false;  
    break;  
default:  
    System.out.println("Comando não reconhecido.");  
    showCommands();  
    break;
```



Pacotes Game, Room e Player (POO e Pacotes)

>  game

>  player

>  room

```
import database.JogoMySQL;  
import player.Player;  
import room.Room;  
  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.Scanner;
```

You, 5 hours ago | 2 authors (Marcos Vinicius Bartoli Senko and one other)


```
public class Game {  
  
    private Player player;  
    private Room room;  
    private Connection connection;  
    private int currentRoom;
```




CONEXÃO BANCO SQL

PARA EFETUAR A CONEXÃO COM
O BANCO DE DADOS SQL
UTILIZAMOS OS SEGUINTE
CODIGOS:

```
private static final String URL = "jdbc:mysql://localhost:3306/text_adventure_game"; 1 usa
private static final String USER = "root"; 1usage
private static final String PASSWORD = ""; 1usage
```



localhost:3306 = é a porta do banco
text_adventure_game = banco de dados criado no
DBeaver





CLASS saveGame

Para realizar o “SALVAR JOGO”, é necessário este código (com SQL) com reconhecimento se jogo foi salvo ou não

```
private void saveGame() { 2 usages  ⤴ Marcos Vinicius Bartoli Senko
    try {
        String query = "INSERT INTO game_state (player_name, room_description, has_key, door_open) VALUES (?, ?, ?, ?) " +
            "ON DUPLICATE KEY UPDATE room_description = VALUES(room_description), has_key = VALUES(has_key), door_open = VALUES(door_open)";
        PreparedStatement stmt = connection.prepareStatement(query);
        stmt.setString( parameterIndex: 1, player.getName());
        stmt.setString( parameterIndex: 2, room.getDescription());
        stmt.setBoolean( parameterIndex: 3, room.hasKey());
        stmt.setBoolean( parameterIndex: 4, room.isDoorOpen());
        stmt.executeUpdate();
        System.out.println("Progresso salvo com sucesso!");
    } catch (SQLException e) {
        System.out.println("Erro ao salvar o jogo: " + e.getMessage());
    }
}
```



Array/List

```
public Room(Connection connection, int roomNumber) { 4 usages  👤 koda012 +1
    this.connection = connection;
    this.roomNumber = roomNumber;
    this.hasKey = false;
    this.hasHammer = false;
    this.hasCrowbar = false;
    this.doorOpen = false;
    this.inventory = new ArrayList<>();
}
```



ROOMWRAPPER

Criamos a class
RoomWrapper, conforme a
imagem abaixo:

```
3 import room.Room;
4
5 public class RoomWrapper { no usages 1 Marcos Vinicius Bartoli Senko
6     private Room room; 9 usages
7
8     public RoomWrapper(Room room) { no usages 1 Marcos Vinicius Bartoli Senko
9         this.room = room;
10    }
11
12
13    public void handleCommand(String command) { no usages 1 Marcos Vinicius Bartoli Senko
14        switch (command) {
15            case "OLHAR AO REDOR":
16                room.lookAround();
17                break;
18            case "EXAMINAR CAMA":
19                room.examineBed();
20                break;
21            case "EXAMINAR PEDRA":
22                room.examineStone();
23                break;
24            case "PEGAR CHAVE":
25                room.pickKey();
26                break;
27            case "EXAMINAR PORTA":
28                room.examineDoor();
29                break;
30            case "USAR CHAVE NA PORTA":
31                room.useKey();
32                break;
33            default:
34                System.out.println("Comando não reconhecido.");
35                break;
36        }
37    }
38
39
40    public void showInventory() { no usages 1 Marcos Vinicius Bartoli Senko
41        System.out.println("Inventário: " + room.getInventory());
42    }
```



Laços de rep. (WHILE)

```
while (gameRunning) {  
    System.out.print("Jogador digita: ");  
    String input = scanner.nextLine().toUpperCase();  
  
    switch (input) {  
        case "OLHAR AO REDOR":  
            room.lookAround();  
            break;  
        case "EXAMINAR CAMA":  
            room.examineBed();  
            break;  
        case "EXAMINAR ARMÁRIO":  
            room.examineCabinet();  
            break;  
        case "PEGAR CHAVE":  
            room.pickKey();  
            break;  
        case "ABRIR ARMÁRIO":  
            room.openCabinet();  
            break;  
        case "EXAMINAR PEDRA":  
            room.examineStone();  
            break;  
        case "EXAMINAR PORTA":  
            room.examineDoor();  
            break;  
        case "USAR CHAVE NA PORTA":  
            if (room.useKeyOnStoneDoor()) {  
                nextRoom();  
            }  
            break;  
        case "USAR MARRETA NA PORTA":  

```


BANCO – TABLES – PK/FK

```
CREATE TABLE IF NOT EXISTS game_state (  
    player_name VARCHAR(50) PRIMARY KEY,  
    room_description TEXT,  
    has_key BOOLEAN,  
    door_open BOOLEAN  
);  
  
CREATE TABLE IF NOT EXISTS game_phrases (  
    phrase_key VARCHAR(50) PRIMARY KEY,  
    phrase TEXT  
);  
  
CREATE TABLE IF NOT EXISTS inventory (  
    player_name VARCHAR(50),  
    item_name VARCHAR(50),  
    PRIMARY KEY (player_name, item_name),  
    FOREIGN KEY (player_name) REFERENCES game_state(player_name)  
);
```

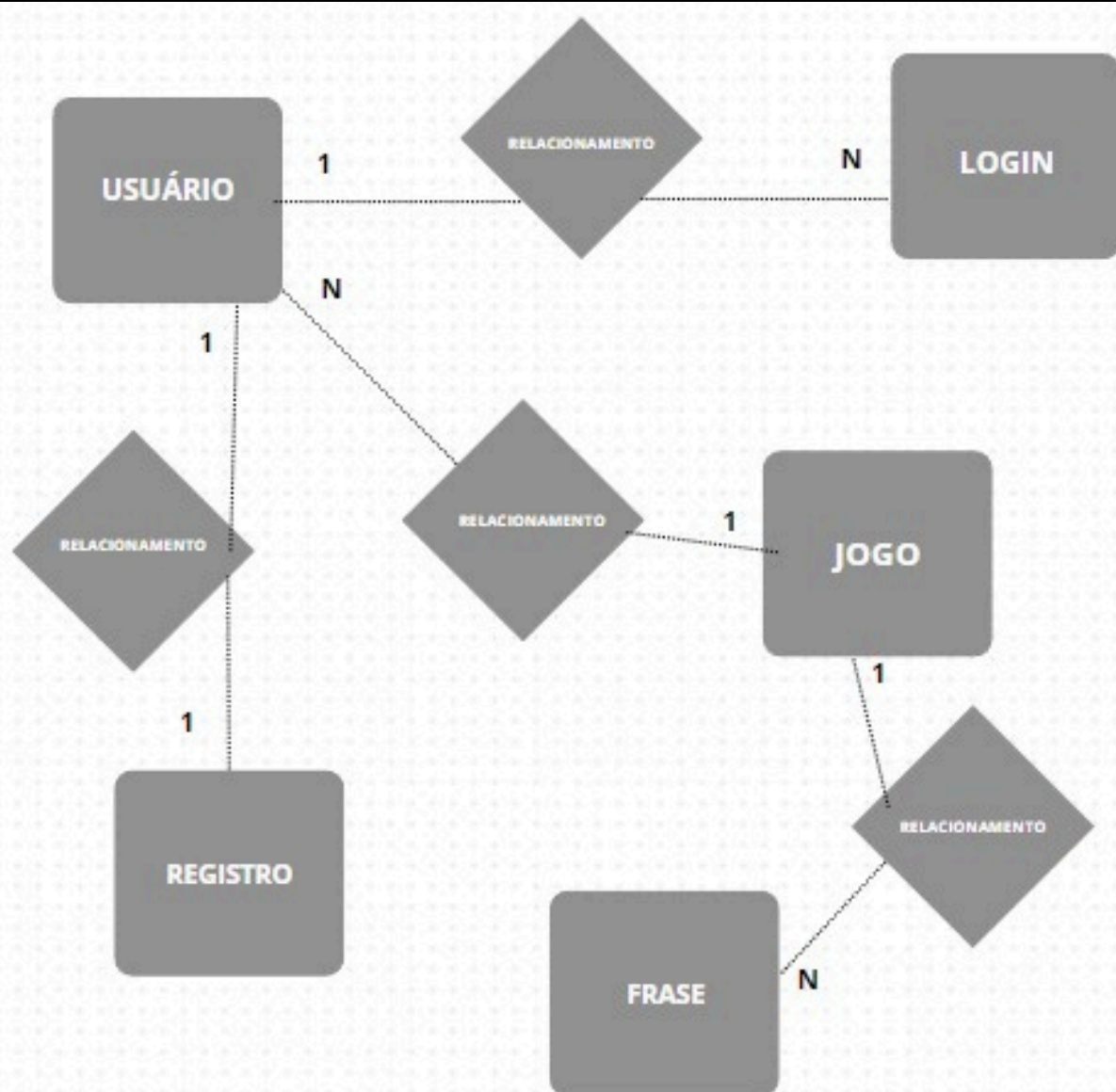


INSERT INTO

```
⚠ ● INSERT INTO game_frases (phrase_key, phrase) VALUES
    ('look_around', 'Você olha ao redor e vê uma cama e uma p
    ('examine_bed', 'A cama está velha e empoeirada, sem nada
    ('found_key', 'Você encontra uma chave debaixo da pedra.'
    ('already_has_key', 'Você já pegou a chave.'),
    ('pick_key', 'Você pegou a chave.'),
    ('already_picked_key', 'Você já pegou a chave.'),
    ('key_not_visible', 'Você não pode pegar a chave, ela não
    ('examine_door', 'A porta é de metal pesado e está tranca
    ('use_key_success', 'Você usou a chave para abrir a porta
    ('door_already_open', 'A porta já está aberta.'),
    ('key_not_found', 'Você não tem a chave para abrir a port
```



DER





LINGUAGENS

Linguagens HyperText/Programação
presentes no projeto “A Última
Chave”.

