INSTITUTO FEDERAL DE SÃO PAULO

CÂMPUS PIRACICABA

ABNER CARDOSO DE SOUZA

CLAYTON JOSÉ DOS SANTOS JUNIOR

GABRIEL JORGE DE OLIVEIRA

DESENVOLVIMENTO EM JAVA DO JOGO

O HOSPÍCIO

PIRACICABA – SP

2018

1 INTRODUÇÃO

Atualmente, existem diversas aplicações que fazem parte da vida da maioria das pessoas, não só dentro das empresas, como também nas instituições de ensino e até mesmo dentro das casas de quase todos. Apesar dos casos de pessoas que aparentemente não conseguem ficar longe de tecnologia se tornando algo prejudicial à saúde, é indiscutível a importância da tecnologia e suas aplicações no mundo contemporâneo.

Até mesmo na área de jogos, as aplicações ganham espaço de modo exponencial com suas funcionalidades e a maneira como se apresentam para os usuários finais do sistema. E é exatamente essa a área escolhida para o desenvolvimento desse projeto envolvendo interface gráfica, algumas funcionalidades essenciais e banco de dados. A aplicação faz uso da linguagem de programação Java e do Sistema Gerenciador de Banco de Dados MySQL.

O intuito desse projeto é melhorar a programação dos integrantes do grupo na linguagem Java, adquirir experiência para a criação de novos sistemas, se envolver com um jogo que poderá ser adaptado futuramente, obter o grau de satisfação dos usuários que fizerem uso do sistema e melhorar os conceitos referentes à programação em geral. A temática d jogo é enigma e suspense sendo inspirado no jogo de videogame Until Dawn e por isso, o efeito do jogo também se baseia no Efeito Borboleta ou Efeito Dominó.

É interessante mencionar que a versão do jogo pertencente ao projeto é simplificada se comparado ao jogo Until Dawn, até pelo fato de que o Until Dawn é um jogo de console que foi lapidado por um bom tempo. No entanto, a lógica usada em Until Dawn é interessante o suficiente para agradar determinados usuários do jogo do projeto, mesmo que em interface gráfica mais simples. Ainda assim, o jogo tenta ser bem intuitivo.

2 INFORMAÇÕES GERAIS SOBRE O JOGO

Como mencionado, o jogo se baseia em *Until Dawn*, porém, de maneira mais simplificada e bem amigável para atender jogadores de muita experiência em jogos, mas também os mais leigos que querem ter a experiência de jogá-lo.

2.1 Informações sobre o local da história

A história tem como local escolhido um hospício, lugar que estudantes de medicina realizam uma visita técnica para ter contato direto com pessoas que precisam de ajuda. Vale mencionar que o enredo do começo ao fim se passa no hospício.

2.2 Informações sobre os personagens e decisões

A ideia principal do jogo se resume a uma serie de decisões de personagens que poderão afetar ou não, o rumo da história. Tais decisões podem influenciar também nos atributos dos personagens de modo que os personagens apresentem comportamentos diferenciados de acordo com o grau do atributo, por exemplo, se um personagem decidir se esconder, o grau de um atributo específico desse personagem poderá ser elevado ou não.

Inicialmente, alguns atributos que estariam presentes no jogo e que poderiam ser influenciados pelas decisões eram: carisma, empatia e coragem. Porém, mais tarde foi planejado que tivesse os atributos: sanidade, emocional, carisma e coragem deixando de lado o atributo empatia. Outro atributo que está presente no jogo como informação do personagem, mas que não altera no enredo é o nome do personagem. Vale mencionar que a ideia de inserir essas informações também foi inspirada no jogo de console Until Dawn.

3 REQUISITOS DO SISTEMA

O sistema apresenta requisitos que se dividem em requisitos funcionais e não funcionais. Tais requisitos fazem total diferença na qualidade de software e em sua popularidade. Abaixo estão descritos os requisitos referentes ao jogo “O Hospício”.

3.1 Requisitos Funcionais

- Realizar cadastro do usuário para que o mesmo possa ter acesso ao jogo:
    - Ele deve criar um *login*, senha e apelido.
- O sistema deve fazer comunicação com o banco de dados:
    - Os dados do usuário devem ficar no banco de dados para fazer a checagem durante o *login* e o *save*.

- Permitir que o usuário possa escolher as decisões
    - O usuário é quem decide o que é mais interessante de se fazer.
- Permitir que o usuário salve a parte do jogo que ele queira jogar mais tarde:
    - O jogo não possui *checkpoints*, mas, por outro lado, o usuário tem a liberdade para escolher quando deseja parar e salvar uma determinada fase.
- O jogo deve transmitir uma mensagem ao usuário quando o mesmo terminar o jogo:
    - Quando o jogo for zerado, uma mensagem indicará o fim do jogo.

3.2 Requisitos Não Funcionais

- Apresenta bom desempenho:
    - Tudo deve fluir como desejado, com boa velocidade de execução.
- Apresenta interface gráfica intuitiva:
    - O modo como os botões foram inseridos e a posição de cada componente presente no jogo são muito amigáveis a fim de atender também jogadores menos experientes.

4 DIAGRAMAS

Os diagramas também compõem as ferramentas de extrema importância para diversas áreas inclusive a área de computação. Eles ajudam a facilitar o entendimento de algo por meio de um esquema visual.

4.1 Diagrama de Casos de Uso e sua Descrição Resumida

Escopo: sistema que é um jogo.

Ator: usuário.

Interessados: pessoas em geral que queiram ter a experiência de jogar o jogo e saber como ele funciona podendo ser de maneira mais superficial ou de modo mais aprofundado.

Garantia de sucesso: fases salvas quando desejado, fácil de jogar.

Cenário Principal:

1. Usuário realiza cadastro.
2. Usuário realiza o *login*.
3. Usuário tem acesso as fases.
4. Usuário decide entre duas decisões ou avança, podendo salvar a fase que poderá ser jogada mais tarde.
5. O sistema varia o status do personagem de acordo com as decisões.
6. O sistema transmite mensagem informando quando se chega ao fim.

Cenário Alternativo:

1. O sistema informa que o cadastro não pode ser efetuado por já ter um cadastro com os mesmos dados. Sistema envia mensagem de erro e não aceita entrada.
2. O sistema informa que o *login* não pode ser realizado por alguma inconsistência de dados.

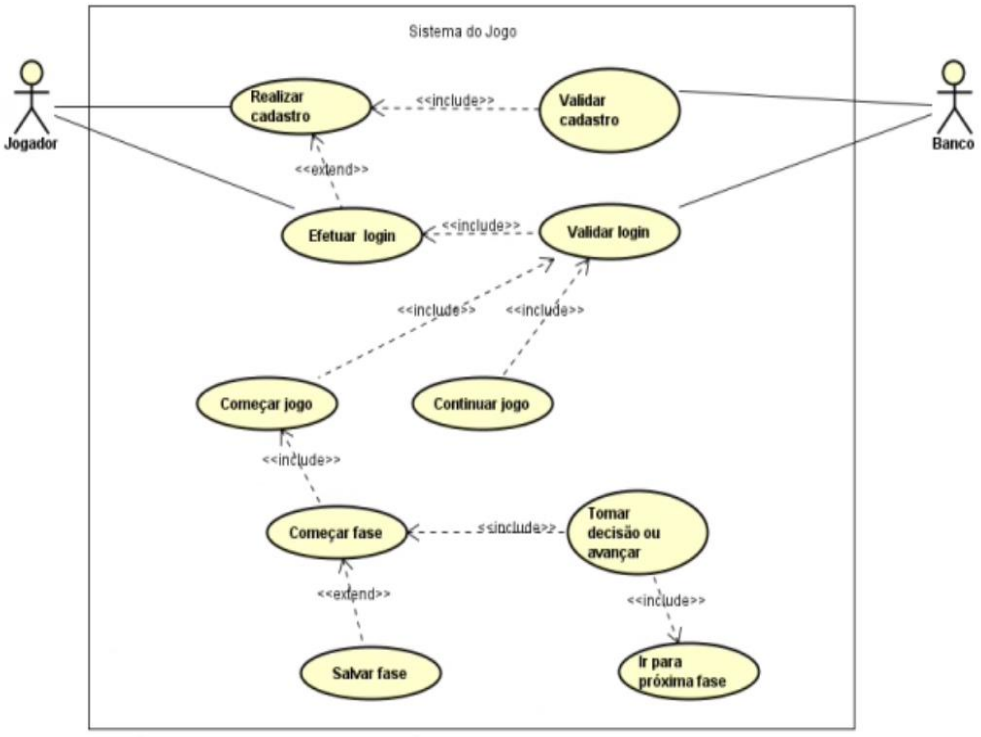

Figura 1 – Diagrama de Casos de Uso

O diagrama acima mostra basicamente que o jogador que é o usuário realiza o cadastro e efetua o *login*. O banco de dados é quem valida tanto o *login* como o cadastro. Se o *login* for feito com sucesso, então o jogador poderá começar o jogo ou continuar o jogo da onde parou por meio dos *saves*. E por fim, cada fase começa depois de uma decisão ou depois que o usuário clica em avançar.

4.2 Diagrama de Casos de Uso e sua Descrição Resumida

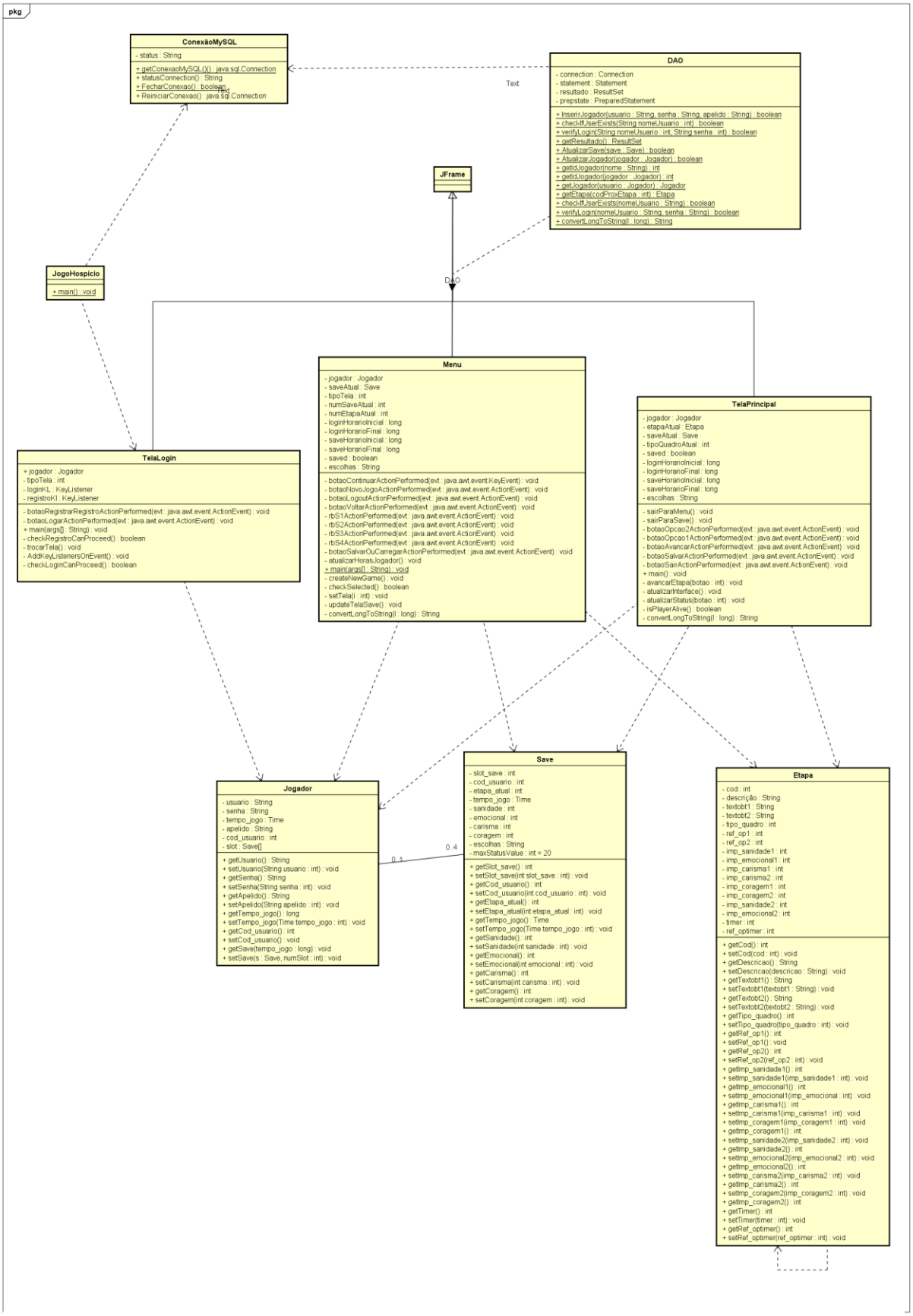

Figura 2 – Diagrama de Classe

O diagrama acima apresenta as classes utilizadas no nosso jogo de LP3. Todos os seus atributos são *private*, para melhorar a segurança e confiabilidade do jogo. A classe *main* cria uma instância de *TelaLogin*, para que o usuário possa logar. Caso o usuário ainda não tenha um cadastro, ele pode optar por se cadastrar. Depois de feito o *login*, o programa irá exibir o menu principal para o jogador, onde ele pode escolher começar um Novo Jogo, Continuar um jogo salvo (caso o mesmo possua um jogo salvo), e caso nenhuma dessas opções lhe convenha, ele pode escolher fazer *logout* para trocar de usuário ou simplesmente para sair do programa. Caso ele escolha Novo Jogo, será apresentado uma série de “etapas” (fases) contendo parte da história, e as vezes uma decisão. Independente de ter ou não uma “decisão”, o usuário pode ter seus status alterados durante a *gameplay* por conta do que aconteceu na fase atual (por exemplo, durante uma situação tensa, ou uma escolha ruim, o usuário pode receber incrementos negativos em seus status). Depois de cada seleção, o programa aplica as conseqüências da decisão selecionada no jogador, verifica se o mesmo ainda está apto a continuar (se ele ainda está *vivo),* então solicita do banco de dados a próxima etapa (caso ela não exista, significa que o jogador terminou o jogo), e então atualiza a tela com as informações da próxima etapa, junto das novas barras de status do jogador. O jogador pode salvar seu progresso a qualquer momento, clicando no botão Salvar.

4.3 Diagrama de Sequência para Troca de Tela

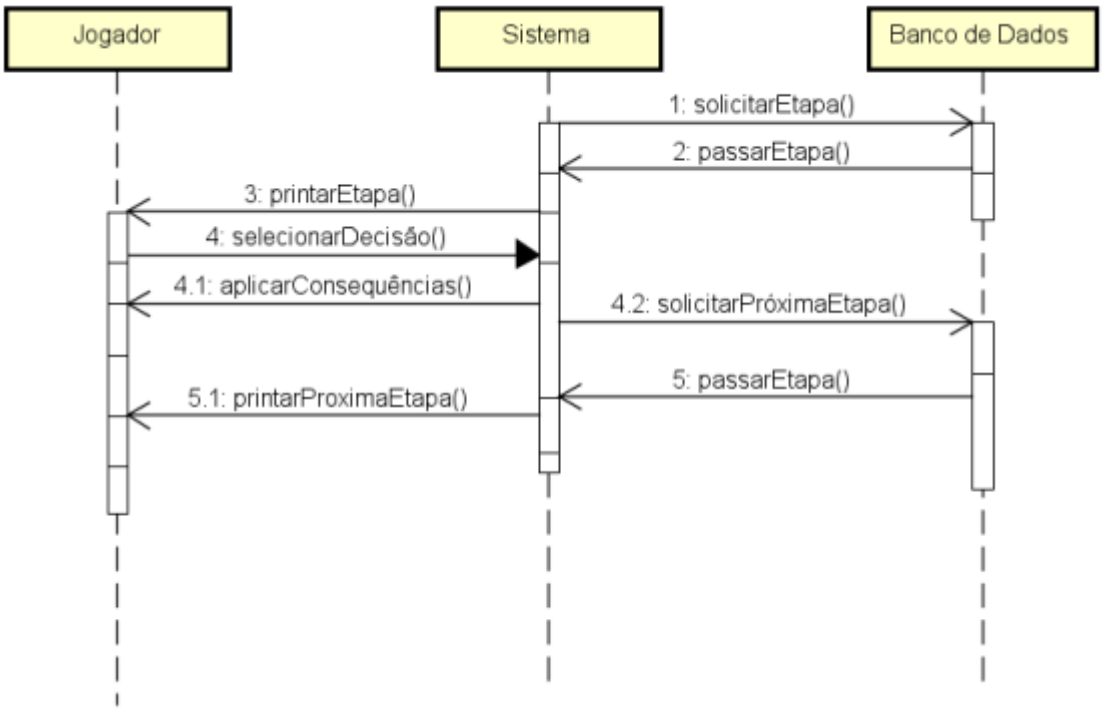

Figura 3 – Diagrama de Sequência para troca de tela

O diagrama acima mostra basicamente que o sistema solicita a etapa ao banco de dados que concede a etapa. Então o sistema mostra na tela a etapa permitindo que o usuário ou o jogador escolha sua decisão. Em seguida, alguma(s) conseqüência(s) podem ocorrer, ou seja, os status podem ser afetados. O sistema, então, solicita a etapa seguinte que é passada para o sistema por meio do banco de dados e o ciclo recomeça, mas com novas informações na tela.

4.4 Diagrama de Estados para o Funcionamento da Execução do Programa

Figura 4 - Diagrama de Estados para o Funcionamento da Execução do Programa

O diagrama acima mostra basicamente que o sistema está ocioso até o momento que alguma atividade seja executada mudando o estado do mesmo.

4.4 Diagrama de Atividades para Troca de Etapas

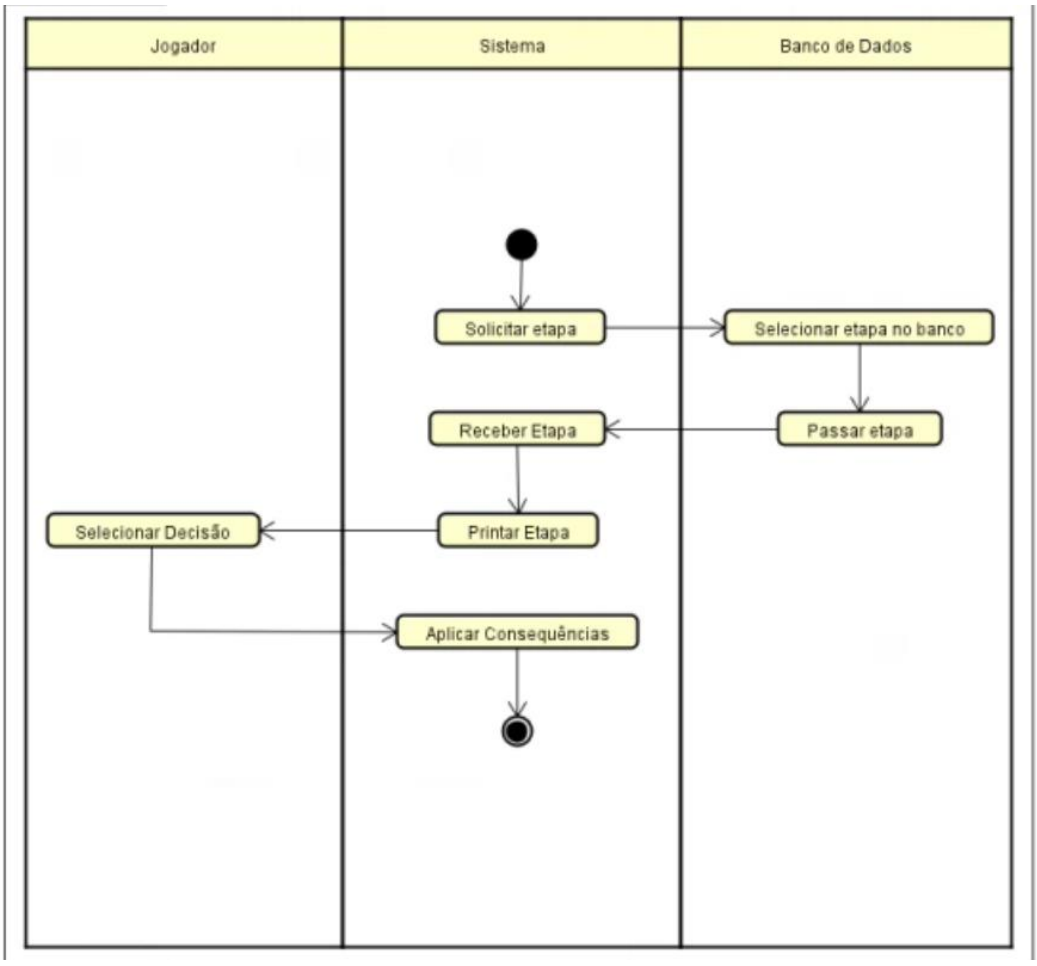

Figura 5 - Diagrama de Atividades para Troca de Etapas

O diagrama acima mostra basicamente que o sistema solicita uma etapa ao banco de dados que fornece a etapa que “printa” na tela a etapa permitindo que o jogador escolha uma decisão e que por causa da decisão, o sistema poderá alterar o status do personagem.

5 TABELAS EM SQL

A linguagem SQL é praticamente indispensável nos dias de hoje em meio a globalização e a evolução tecnológica, como é o caso dos bancos de dados. Para o armazenamento das fases do jogo, por exemplo, a linguagem SQL acabou sendo muito útil, permitindo que as fases fossem puxadas diretamente do banco de dados.

As tabelas usadas foram:

```
create table etapa(
    cod INT NOT NULL,
    descricao VARCHAR(2000) NOT NULL,
    textobt1 VARCHAR(70) NOT NULL,
    textobt2 VARCHAR(70) NOT NULL,
    tipo_quadro INT NOT NULL CHECK(tipo_quadro=0 OR tipo_quadro=1),
    ref_op1 INT NOT NULL,
    ref_op2 INT NOT NULL,
    imp_sanidade1 INT NOT NULL,
    imp_emocional1 INT NOT NULL,
    imp_carisma1 INT NOT NULL,
    imp_coragem1 INT NOT NULL,
    imp_sanidade2 INT NOT NULL,
    imp_emocional2 INT NOT NULL,
    imp_carisma2 INT NOT NULL,
    imp_coragem2 INT NOT NULL,
    timer INT NOT NULL,
    ref_optimer INT NOT NULL CHECK (timer=-1 AND ref_timer=-1 OR ref_timer>0),
    PRIMARY KEY(cod),
    FOREIGN KEY(ref_op1) REFERENCES etapa(cod),
    FOREIGN KEY(ref_op2) REFERENCES etapa(cod),
    FOREIGN KEY(ref_optimer) REFERENCES etapa(cod)
);

create table jogador(
    cod_usuario INT NOT NULL AUTO_INCREMENT,
    usuario VARCHAR(50) NOT NULL,
    senha VARCHAR(50) NOT NULL,
    apelido VARCHAR(50) NOT NULL,
    tempo_jogo LONG,
    PRIMARY KEY(cod_usuario),
    CONSTRAINT UNIQUE_jogador_usuario UNIQUE(usuario)
);
```

```sql
create table save(
    slot_save INT NOT NULL CHECK (slot_save>=1 AND slot_save<=4),
    cod_usuario INT NOT NULL,
    etapa_atual INT DEFAULT 0,
    tempo_jogo LONG,
    sanidade INT DEFAULT 0,
    emocional INT DEFAULT 0,
    carisma INT DEFAULT 0,
    coragem INT DEFAULT 0,
    escolhas VARCHAR(1000),
    CONSTRAINT PK_save_cod_usuario FOREIGN KEY(cod_usuario)
    REFERENCES jogador(cod_usuario)
);
```

CONCLUSÃO

Portanto, é possível dizer que mais uma vez a programação mostrou-se relevante possibilitando a realização de um jogo que envolveu interface gráfica com o usuário, persistência de dados a partir do uso de código em SQL e banco de dados com suas respectivas restrições. Felizmente, essa experiência não é válida apenas para esse jogo, mas sim, é uma experiência que inspira à ambição de novos projetos e futuras experiência iguais a essa.

Que todo o esforço e a busca pelo aprendizado sejam cada vez mais valorizados nas diversas áreas da sociedade, inclusive na área de desenvolvimento de sistemas que contribuem e ajudam no entretenimento de indivíduos que buscam sair de suas rotinas e que anseiam pelo mundo dos jogos e da imaginação.