PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS

ESCOLA POLITÉCNICA

Lucas Valério Berti - 22007440

PARADIGMA E PROGRAMAÇÃO ORIENTADA A OBJETOS

Projeto Palavra Secreta

Campinas

1. INTRODUÇÃO E VISÃO DO PRODUTO

O presente projeto consiste no desenvolvimento do software **WordQuest**, um jogo digital do tipo *"Palavra Secreta"* implementado em linguagem Java. O sistema foi concebido integralmente sob os princípios da Programação Orientada a Objetos (POO), atendendo aos requisitos técnicos estabelecidos e adotando uma perspectiva empreendedora, considerando o jogo como um potencial produto digital.

1.1. Proposta de Valor e Mercado

O **WordQuest** posiciona-se como uma solução híbrida entre **Gamificação Educacional** e **Entretenimento Cognitivo**. Diferentemente de jogos tradicionais da força ou palavras cruzadas, o produto foi projetado para oferecer uma experiência dinâmica, adaptável e facilmente expansível.

Público-Alvo Educacional:
• Escolas, professores e instituições que buscam ferramentas lúdicas para apoio ao ensino de ortografia, ampliação de vocabulário e fixação de conteúdos do ensino fundamental.
• Aplicações em reforço escolar, atividades complementares e dinâmicas de sala de aula.

Público-Alvo Casual:
• Jovens e adultos interessados em jogos de raciocínio rápido (*brain training*) para exercícios cognitivos no dia a dia.
• Usuários que buscam entretenimento leve e desafiador em ambientes digitais.

1.2. Inovação e Diferenciais

O principal diferencial do **WordQuest** é sua **adaptabilidade algorítmica**. O jogo distingue automaticamente a complexidade entre uma *Palavra Única* e uma *Frase Completa*, ajustando parâmetros como:

- tempo máximo permitido;
- número de tentativas;
- regras de pontuação.

Essa capacidade garante desafios equilibrados, mantendo o engajamento do usuário.

Outro diferencial relevante é o uso de **arquivos externos (.txt)** para armazenamento das palavras, dicas e metadados. Essa abordagem permite atualizar o conteúdo do jogo sem modificar o código-fonte, o que aumenta sua longevidade como produto e reduz custos de manutenção. Esse atributo contribui para a viabilidade comercial ao facilitar a criação de novos pacotes temáticos e expansões.

2. ARQUITETURA E ESPECIFICAÇÃO TÉCNICA

A aplicação foi construída com foco em modularidade, segurança e escalabilidade, explorando plenamente os conceitos fundamentais da Programação Orientada a Objetos.

2.1. Estrutura de Classes e Herança

A arquitetura adota uma hierarquia clara de classes que separa a lógica de negócio da interação com o usuário:

• Challenge (Classe Abstrata)
Representa a generalização de um desafio. Define atributos comuns — como *challengeText*, *hint* e *level* — e estabelece métodos abstratos relacionados às regras de dificuldade, garantindo consistência entre os diferentes tipos de desafio.

• SecretWord (Classe Concreta)
Especializada em palavras únicas. Implementa regras de pontuação mais rígidas e um tempo de resolução reduzido.

• SecretPhrase (Classe Concreta)
Responsável por desafios compostos por frases inteiras. Possui maior tolerância a erros e um tempo ampliado, adequando-se naturalmente à maior complexidade.

Essa estrutura evita redundância, facilita a manutenção e possibilita expansão futura com novos tipos de desafios.

2.2. Aplicação de Polimorfismo

O Polimorfismo é aplicado principalmente nos métodos de controle de dificuldade:

- `getMaxAttempts()`
- `getMaxTimeInSeconds()`

O controlador principal (`GameManager`) interage exclusivamente com objetos do tipo **Challenge**, sem precisar conhecer suas subclasses. Durante a execução, a JVM identifica automaticamente o tipo real do objeto (palavra ou frase) e aplica a regra apropriada.

Essa abordagem permite adicionar futuros modos de jogo — como “charadas”, “palavras embaralhadas” ou “expressões idiomáticas” — sem modificar o código já existente do controlador.

2.3. Persistência e Padrão Factory

A classe **DataAccess** centraliza toda a lógica de entrada e saída de dados.

Ao ler o arquivo *palavras.txt*, ela identifica o tipo de desafio descrito e instância automaticamente a classe correspondente, funcionando como um **Factory simplificado**. Essa abordagem garante baixo acoplamento e simplifica a inclusão de novos tipos de desafios no arquivo.

---

3. FUNCIONALIDADES E REGRAS DO JOGO

O **WordQuest** implementa um ciclo completo de jogo com as seguintes funcionalidades:

Carregamento de Dados:
O sistema lê um arquivo `.txt` contendo desafios organizados por nível de dificuldade. Cada entrada possui texto, dica e parâmetro para cálculo de dificuldade.

Configuração da Partida:
O usuário seleciona o nível desejado (Fácil, Médio ou Difícil). O sistema então escolhe aleatoriamente um desafio compatível.

Mecânica de Jogo:

- O jogador propõe uma letra por vez.
- **Acerto:** a letra é revelada em todas suas posições e o jogador recebe pontuação.
- **Erro:** o jogador perde pontos e uma tentativa.
- Tentativas duplicadas são prevenidas automaticamente pelo sistema.
- O jogador pode solicitar uma *dica* quando necessário.

Condições de Término:
A partida finaliza quando:

- o tempo máximo é atingido;

- o número de tentativas chega a zero;
- ou a palavra/frase é completada.

O jogo monitora esses limites em tempo real, garantindo fluidez e clareza para o usuário.

---

4. CONCLUSÃO

O projeto **WordQuest** atendeu plenamente aos requisitos propostos, demonstrando a eficácia da Programação Orientada a Objetos na construção de um software robusto e bem estruturado. O uso de **Herança** reduziu redundâncias, o **Polimorfismo** proporcionou flexibilidade para criar diferentes modalidades de desafio e o **Encapsulamento** assegurou a integridade dos dados internos.

Sob a perspectiva de produto, o WordQuest demonstra potencial real como ferramenta educacional e de entretenimento cognitivo, caracterizando-se como uma solução escalável, atualizável e alinhada às tendências contemporâneas de gamificação e aprendizagem interativa.