Relatório da Atividade: Refatoração Código Fonte

Integrantes:

- Raul Melo Farias (GU3046923)
- Rodrigo Quadrante Freitas (GU3042073)

---

1. Preparação e Estratégia Inicial

Antes de alterar qualquer linha de código, o primeiro passo foi estabelecer uma "rede de segurança" para garantir que a refatoração não quebraria as regras de negócio existentes.

Utilizando o Maven no VS Code, limpamos os arquivos desnecessários do projeto (como os do Gradle). Em seguida, executamos a classe TexttestFixture.java para gerar a saída original do sistema. Essa saída foi salva em um arquivo chamado golden-master.txt, que se tornou nossa "verdade absoluta" para validar o comportamento do código.

2. Análise e Solução de Refatoração

Com a rede de segurança pronta, o processo de refatoração de fato se iniciou, com uma análise do método updateQuality() da classe GildedRose. Esse método percorria a lista de itens e aplicava uma quantidade cômica de declarações if para verificar o nome de um item e, então, aplicar alguma lógica de atualização de qualidade. A descrição preliminar das regras de atualização de qualidade se provou extremamente útil.

Dentre as soluções pensadas, a que pareceu mais sensata foi uma que incorporasse o polimorfismo. Para isso, foram criadas subclasses da classe Item (AgedBrie, BackstagePasses, Sulfuras, e a nova feature Conjured), que sobrescreveriam um novo método updateIndividualItemQuality().

Para limitar a qualidade entre 0 e 50, foram também criados métodos de verificação na classe Item base. Com isso, o método updateQuality() foi reescrito para apenas percorrer a lista de itens e chamar o método updateIndividualItemQuality() para cada item.

3. Dificuldades: Integração e Depuração

O trabalho de refatoração foi feito em uma *branch* separada do Git (refatoracao01). Ao integrar esse novo código com a main, encontramos duas dificuldades principais:

1. **Erro de Visibilidade:** O TexttestFixture.java não conseguia "enxergar" as novas classes (AgedBrie, Sulfuras, BackstagePasses). Isso ocorria porque as classes foram declaradas sem o modificador public, tornando-as "package-private" e inacessíveis para a classe de teste. A correção foi simplesmente adicionar public à declaração de cada classe.

2. **Erro de Instanciação:** Após corrigir a visibilidade, a lógica ainda não funcionava. O TexttestFixture.java continuava criando os objetos como new Item("Aged Brie", ...) em vez de new AgedBrie("Aged Brie", ...). O polimorfismo não era ativado. Foi necessário corrigir a inicialização dos itens para usar os construtores das novas subclasses.

4. Validação e Lições Aprendidas

Após a correção dos bugs de integração, rodamos o TexttestFixture.java novamente e comparamos sua saída com o golden-master.txt.

O resultado foi um sucesso: a saída para todos os itens originais foi **idêntica** ao *golden master*. A **única** diferença foi no item "Conjured Mana Cake", que agora degradava a qualidade duas vezes mais rápido (de 6 para 4, ao invés de 6 para 5), exatamente como a nova regra pedia.

Isso provou que a refatoração foi bem-sucedida, produzindo um código muito mais fácil de ser compreendido e, principalmente, de ser mantido e estendido, sem quebrar as funcionalidades existentes.