Questão 1 (2,0 pontos) Em um sistema de software para uma rede de "food-truck", há diversas variedades de sanduíches. O produto mais simples é o sanduíche básico, feito com apenas pão e carne. Um sanduíche básico pode ser "incrementado" com porções extras. Cada porção possui um nome e um valor. O cliente da rede pode escolher comprar um sanduíche básico, ou um sanduíche incrementado. De todo modo, tanto um sanduíche básico quanto um incrementado possuem um preço. Nesse sistema de software, o preço de um sanduíche deve ser calculado como o preço do sanduíche básico mais o preço de cada porção adicionada a ele. Para organizar essa solução, um desenvolvedor iniciou a criação de uma solução usando o padrão de projeto denominado `Decorator` (do catálogo GoF). Essa solução inicial e incompleta corresponde às classes (em notação da UML) apresentadas na Figura 1.

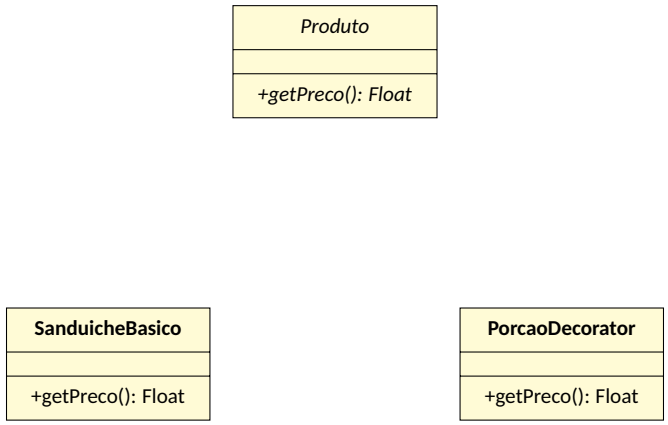

Figura 1: Solução inicial e incompleta

(a) Por meio da notação UML, complete o diagrama de classes fornecido para descrever uma solução de software para o problema apresentado acima, com a restrição de continuar o uso do padrão `Decorator`. Seja o mais detalhado possível do ponto de vista de decisões de projeto (*design*), no sentido de definir outros eventuais elementos do diagrama (classes, atributos, operações, associações, multiplicidades e navegabilidades) adequados e consistentes com a situação-problema apresentada. Apresente também uma descrição textual dos elementos contidos no diagrama final que produzir.

(b) Apresente (em linguagem Java) a implementação dos métodos `getPreco` definidos nas classes denominadas `SanduicheBasico` e `PorcaoDecorator` apresentadas no diagrama.

Questão 2 (2,0 pontos) Durante a revisão manual de determinada classe alterada no contexto de uma *pull request*, o revisor afirmou que a classe revisada era uma *God Class*. Ele também reportou a incidência de um método longo nesta classe (*Long Method*), composto por uma longa lista de parâmetros (*Long Parameter List*). As três anomalias de código (*code smells*) foram reportadas para o autor do código. Como resposta, o autor respondeu afirmando que havia submetido a classe a uma ferramenta de detecção de anomalias de código popular baseada em métricas. Entretanto, a ferramenta havia detectado apenas uma das anomalias apontadas na revisão (*Long Method*). Além disto, a ferramenta também encontrou outro método da referida classe com *Feature Envy*. Considerando o cenário descrito acima, responda os itens a seguir.

(a) Disserte sobre os possíveis motivos para a revisão manual ter identificado um conjunto de anomalias diferente do que foi detectado pela ferramenta.

(b) Apresente possíveis ações de refatoração tipicamente aplicáveis para combater cada um dos quatro tipos de anomalias apresentadas no enunciado. Não é necessário referenciar nomes de refatoração formalmente catalogados.

Questão 3 (2,0 pontos) As atividades de Verificação, Validação e Testes de Software são parte integrante de um processo de desenvolvimento de software baseado em princípios de engenharia.

(a) Caracterize o conceito de **inspeção de software**.

(b) Caracterize **testes de unidade** e **testes de sistema**.

Questão 4 (2,0 pontos) Agilidade em desenvolvimento de software é uma característica desejada. Os métodos ágeis introduziram a agilidade em desenvolvimento de software sendo suas práticas amplamente conhecidas e utilizadas na indústria de desenvolvimento de software. Nesse contexto de métodos e práticas ágeis, considere os itens a seguir.

(a) Descreva a metodologia de gestão de projetos ágeis denominada Scrum. Nessa descrição, inclua as etapas e atividades envolvidas em ordem cronológica de acontecimento em um projeto.

(b) O Extreme Programming (XP) utiliza práticas ágeis específicas para desenvolvimento de software que vão além do que o Scrum realiza para gestão de projetos. Explique quais práticas ágeis específicas para desenvolvimento de software são utilizadas pelo XP.

Questão 5 (2,0 pontos) Considere o protótipo de interface gráfica apresentado na Figura 2. Esse é o protótipo de um formulário para a inclusão de dados em um sistema web. Apresente o código-fonte que implementa a interface gráfica apresentada e as partes do sistema, conforme descrito nos itens a seguir.

(a) Frontend utilizando Javascript e React. Deve permitir a entrada dos dados pelo usuário e realizar o envio de uma requisição HTTP para API REST do backend com o objetivo de incluir os dados no banco de dados. O botão de rótulo “Salvar” deve acionar o envio dos dados para o backend.

(b) Backend utilizando Node.js, Express e MongoDB. Deve disponibilizar um ponto de entrada por meio de uma API REST para o recebimento dos dados provenientes do frontend e fazer a inclusão destes em uma coleção de um banco de dados em MongoDB.

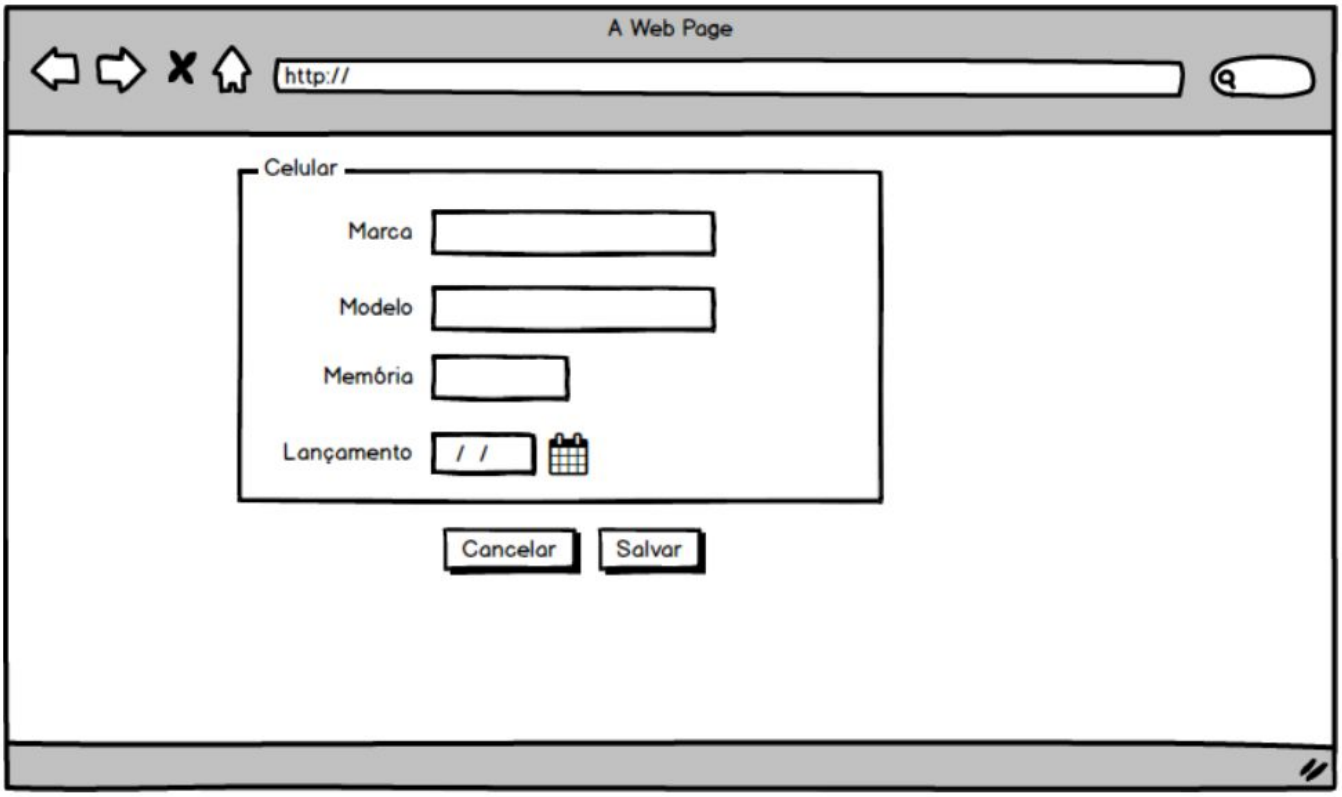

Figura 2: Protótipo de Interface