HuntingBytes

Eduardo Alexandre de Amorim
Gabriel André Camargo Pimentel de Barros
João Pedro Freire de Andrade
João Pedro Oliveira da Silva
Lucas Lins Pereira da Silva
Moésio Wenceslau da Silva Filho

The Cooks' Books

Repositório GitHub: https://github.com/HuntingBytes/The-Cooks-Books

Diagrama de classes:

Abaixo existem 2 versões do diagrama de classes, uma representação mais simples que abstrai os métodos e atributos das classes focando principalmente nas associações, e uma representação mais detalhada que apresenta os atributos das classes e seus métodos principais (Não incluindo todos os getters/setters).

As imagens na resolução original podem ser encontradas no repositório do projeto no ***Github***. Uma descrição detalhada das classes e suas associações pode ser encontrada ao fim do documento, tratando algumas classes separadamente.

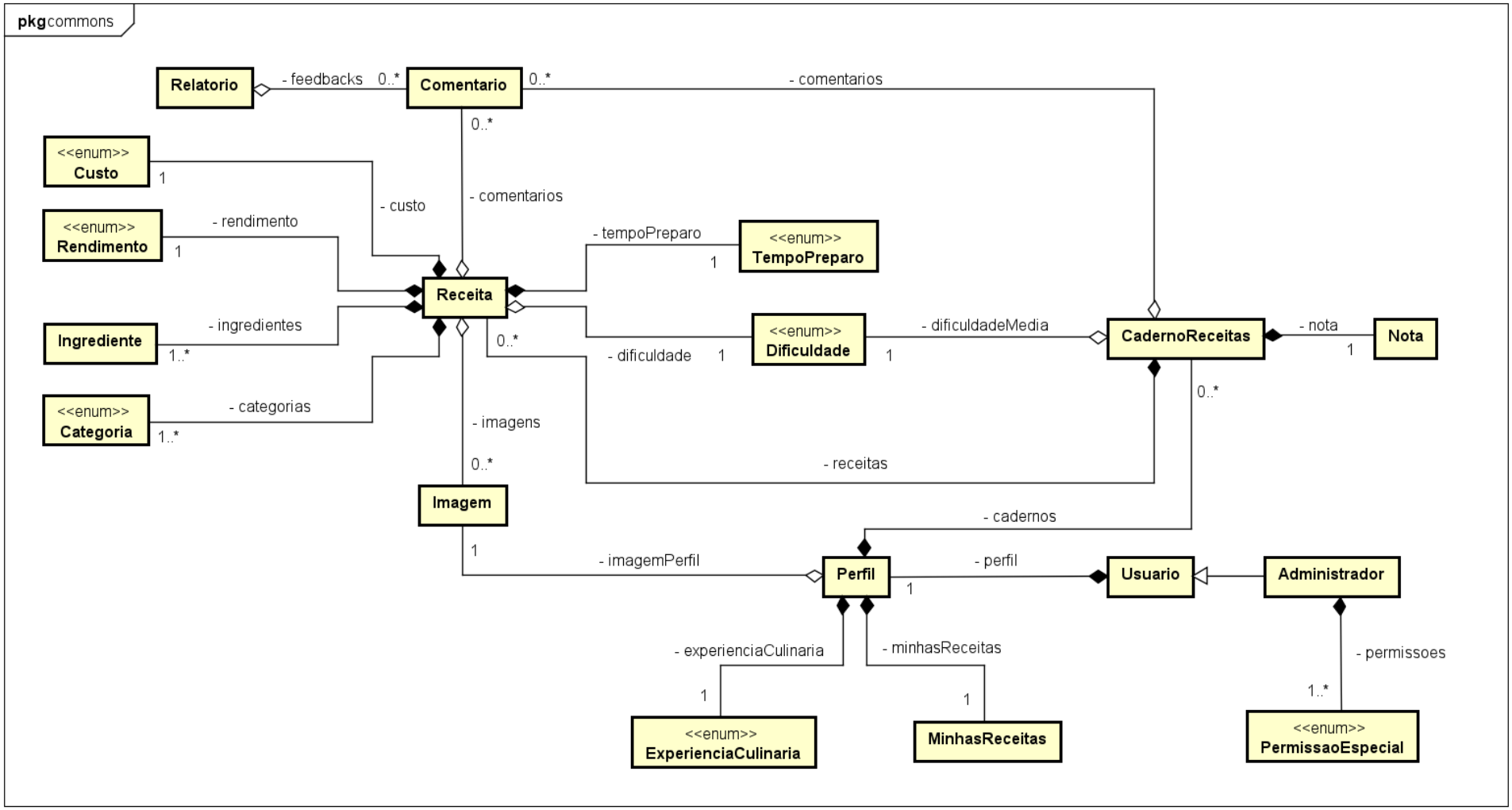


Diagrama de classes simples. Principais associações entre as classes de domínio.

UFRPE

Introdução à Programação II - Projeto
Entrega 02 – Diagrama de classes em UML

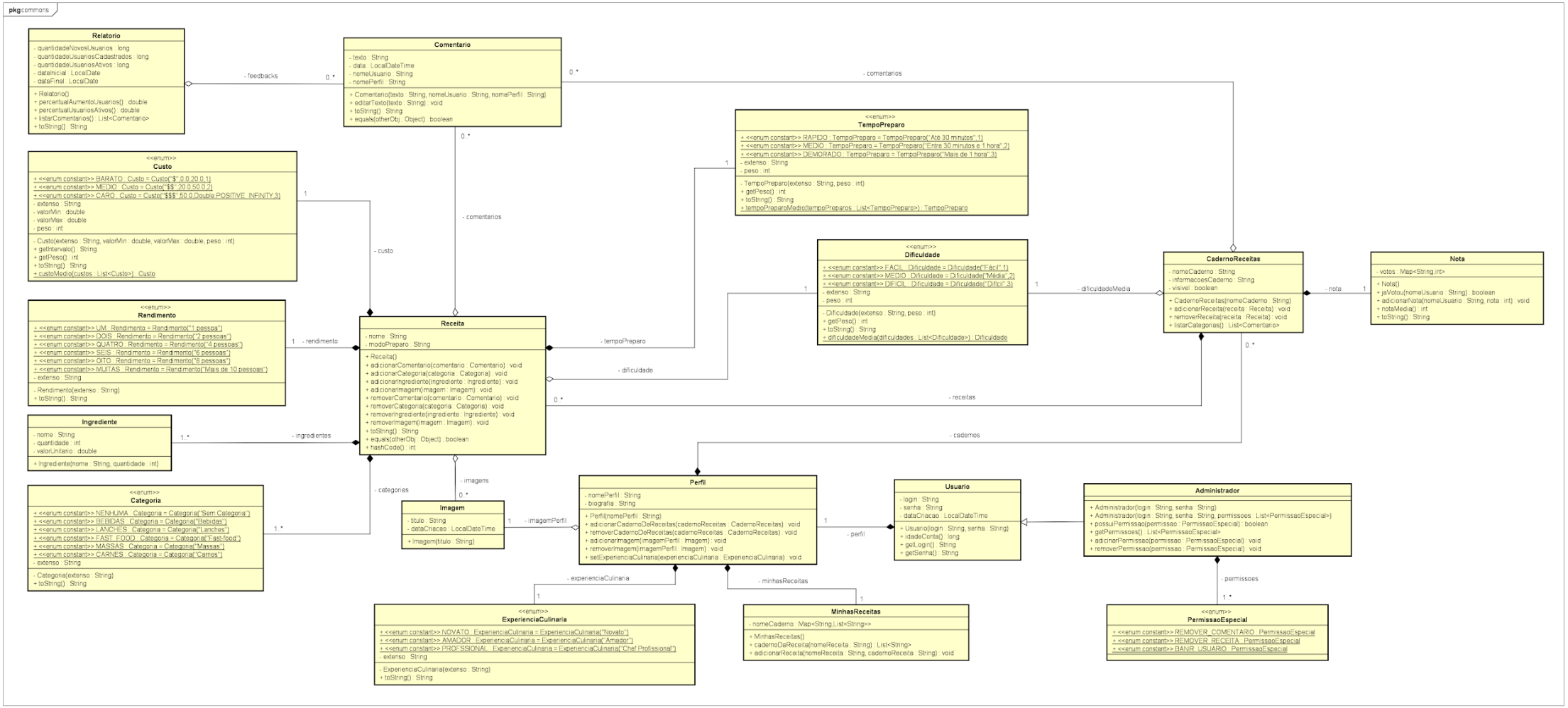
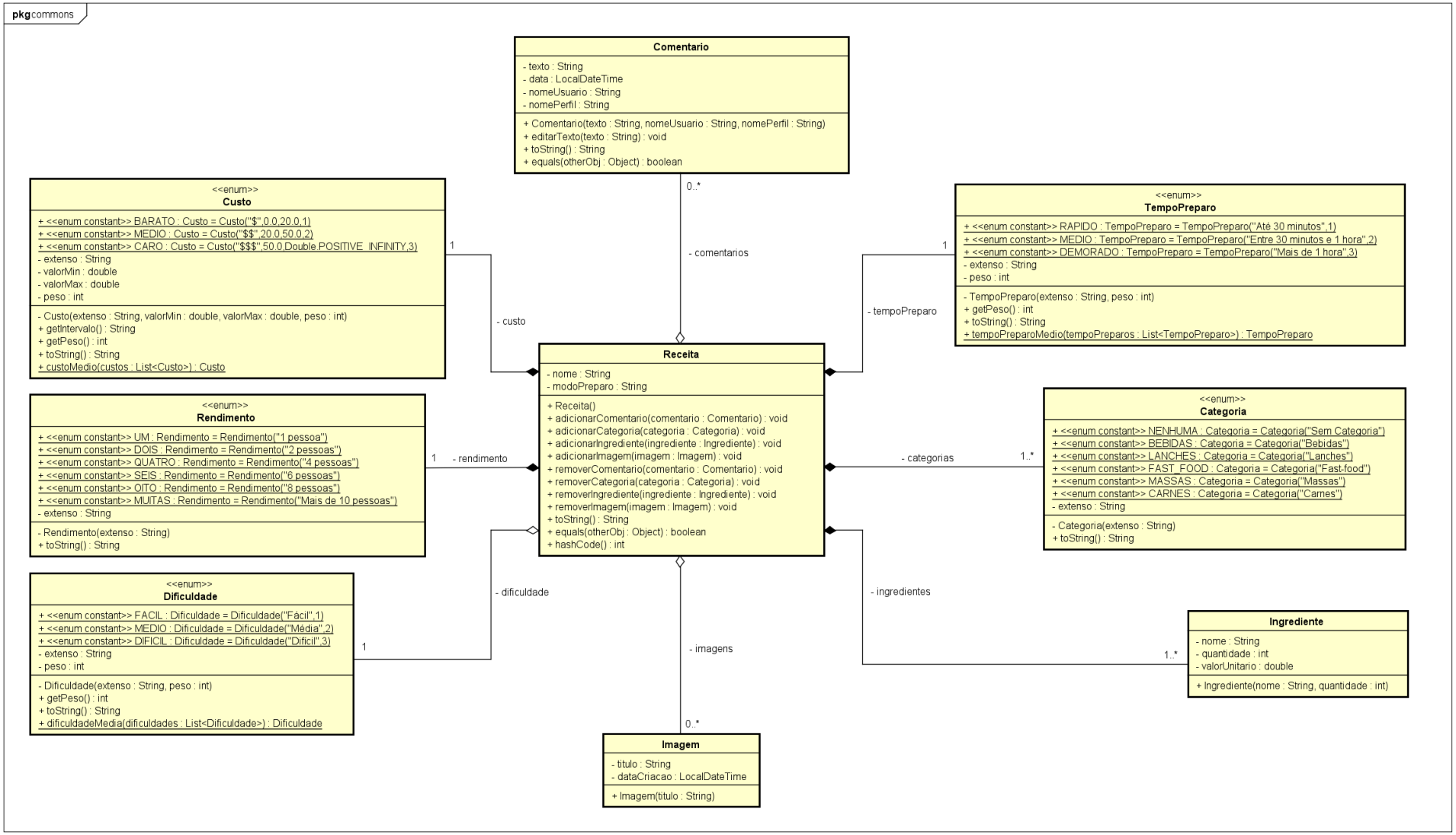


Diagrama de classes detalhado. Principais associações, atributos e métodos

A Classe Receita e suas associações



A classe **Receita** é uma das principais classes do sistema. Ela é composta pelas classes **Custo**, **Rendimento**, **TempoPreparo**, **Categoria** e **Ingrediente**. Ela também possui **Imagem** e **Comentario**. Essa classe é responsável por abstrair a noção de uma receita real, dessa forma possui uma lista de *ingredientes* (List<**Ingrediente**>), um *rendimento* (**Rendimento**), um *tempo de preparo* (**TempoPreparo**), um *custo* (**Custo**), uma lista de *categorias* (List<**Categoria**>), uma *dificuldade* (**Dificuldade**), uma lista de *comentários* (List<**Comentario**>), uma lista de *imagens* (List<**Imagem**>), um *nome* (String) e um *modo de preparo* (String).

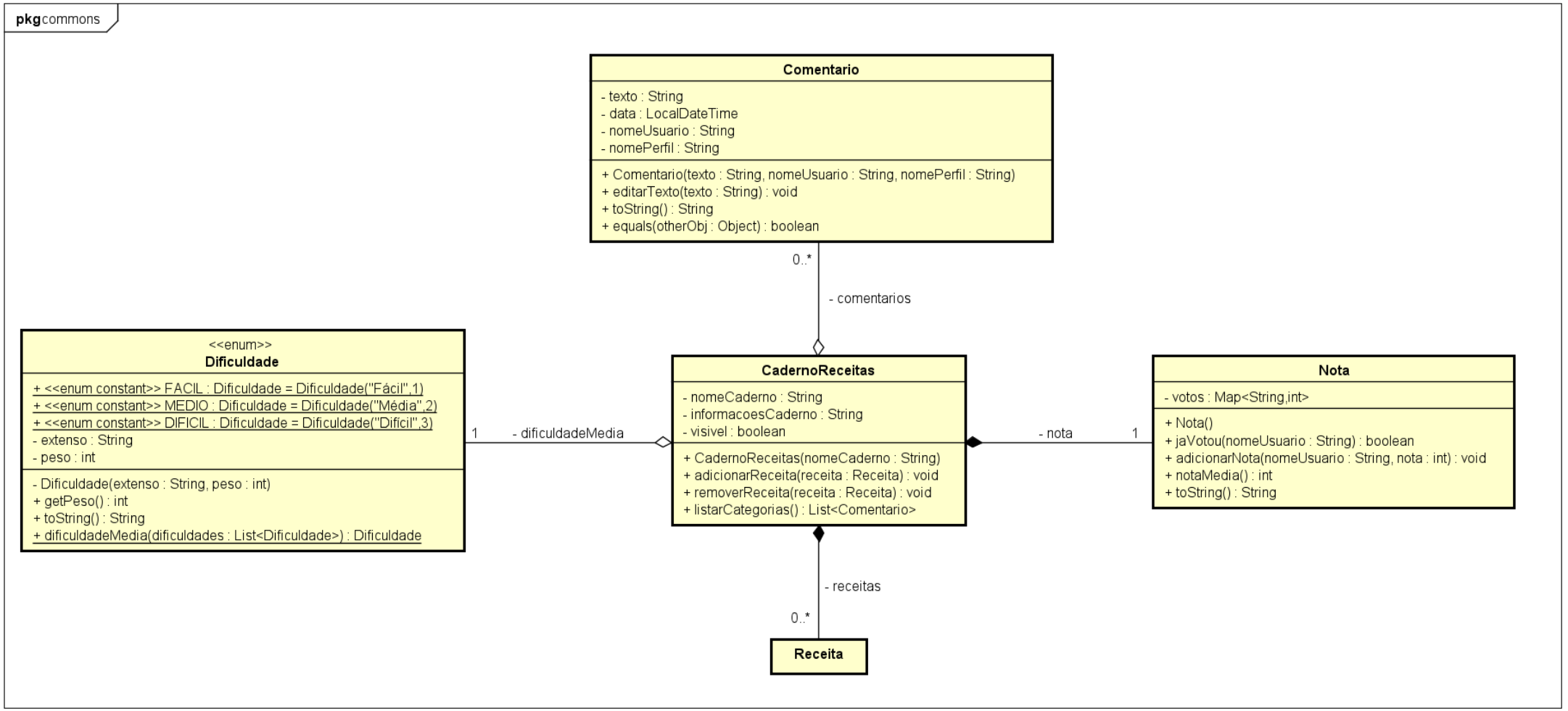
As classes **Rendimento**, **Dificuldade**, **Custo**, **TempoPreparo** e **Categoria** são classes enumeradas (pelo fato delas possuírem instâncias discretas e específicas) responsáveis por abstrair alguns conceitos usados no dia a dia. Elas apresentam métodos que permitem a fácil manipulação (Cálculo do valor médio, representação como String) durante as operações realizadas pelo sistema. Essas classes possuem uma relação de composição com **Receita** (Elas compõem **Receita**).

As classes **Imagem** e **Comentario** são responsáveis por agregar mais informações a uma **Receita**. As imagens permitem uma visualização de algum processo da produção da receita (seja o produto final ou uma etapa intermediária), já os comentários são notas extras, adicionadas pelo autor, que não se encaixam em nenhum outro atributo de uma receita. Essas classes possuem uma relação de agregação com **Receita** (Elas se relacionam, mas não compõem somente **Receita**).

UFRPE

Introdução à Programação II - Projeto
Entrega 02 – Diagrama de classes em UML

A Classe CadernoReceitas e suas associações



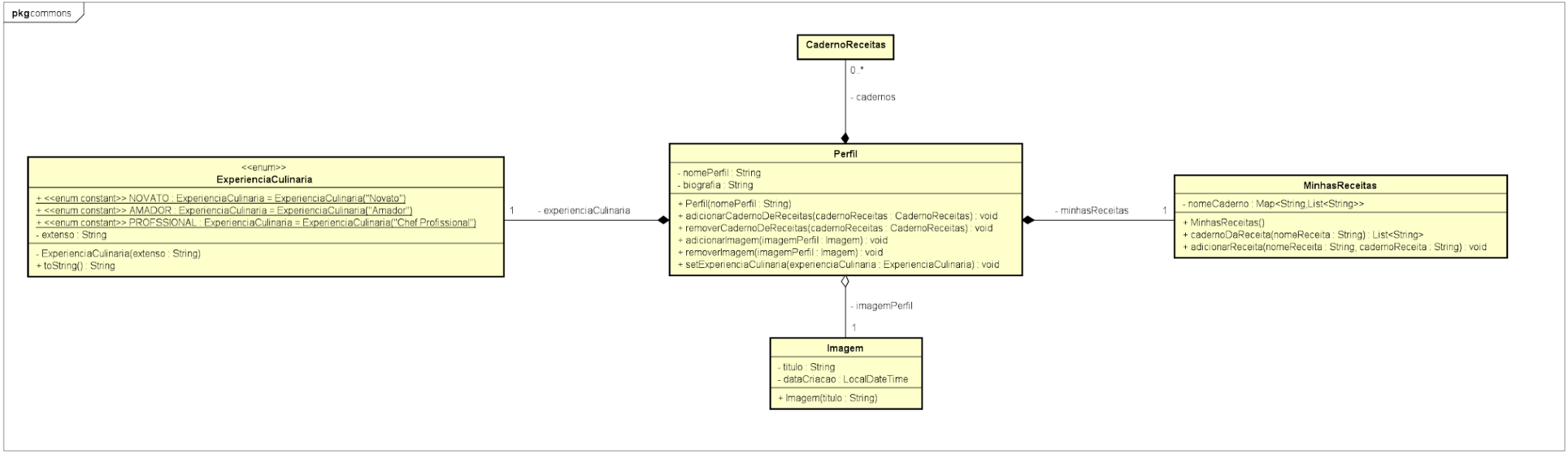
A classe **CadernoReceitas** também é uma das principais classes do sistema. Ela é composta pelas classes **Dificuldade**, **Comentario** e **Nota.** Ela também se relaciona com a classe **Receita,** através de uma relação de Composição, onde **Receita** é uma parte de **CadernoReceitas**. Essa classe é responsável por agrupar uma certa quantidade de receitas criadas pelo usuário, além de possuir qualidades próprias como os comentários, as notas (o usuário poderá avaliar o caderno de receita como um todo e dar sua opinião através de uma nota) e a dificuldade média do caderno (média aritmética da dificuldade das receitas agrupadas nele).

Cada **CadernoReceitas** possui métodos para adicionar ou remover receitas (*adicionarReceita, removerReceita,* respectivamente). Além disso, possui também um método para listar todas as categorias relacionadas com as receitas inseridas em cada caderno (*listarCategorias),* retornando uma lista de **Categoria**.

A classe **Dificuldade** é uma classe enumerada (pelo fato dela possuir instâncias discretas e específicas) responsável por representar a média geral das dificuldades das receitas contidas no caderno. Ela apresenta métodos que permitem a fácil manipulação (Cálculo do valor médio, Representação por String) durante as operações realizadas pelo sistema. Essa classe possui uma relação de agregação com **CadernoReceitas**.

As classes **Nota** e **Comentario** são responsáveis por agregar mais informações ao **CadernoReceitas**. As notas permitem representar a avaliação média dada pelos usuários, já os comentários são um artifício disponível para que outros usuários possam avaliar o caderno do autor e expressar suas opiniões de maneira mais extensa, funcionando como uma extensão de Nota. **Nota** possui uma relação de composição com **CadernoReceitas**, enquanto **Comentario** possui uma relação de agregação com a classe principal em questão.

A Classe Perfil e suas associações

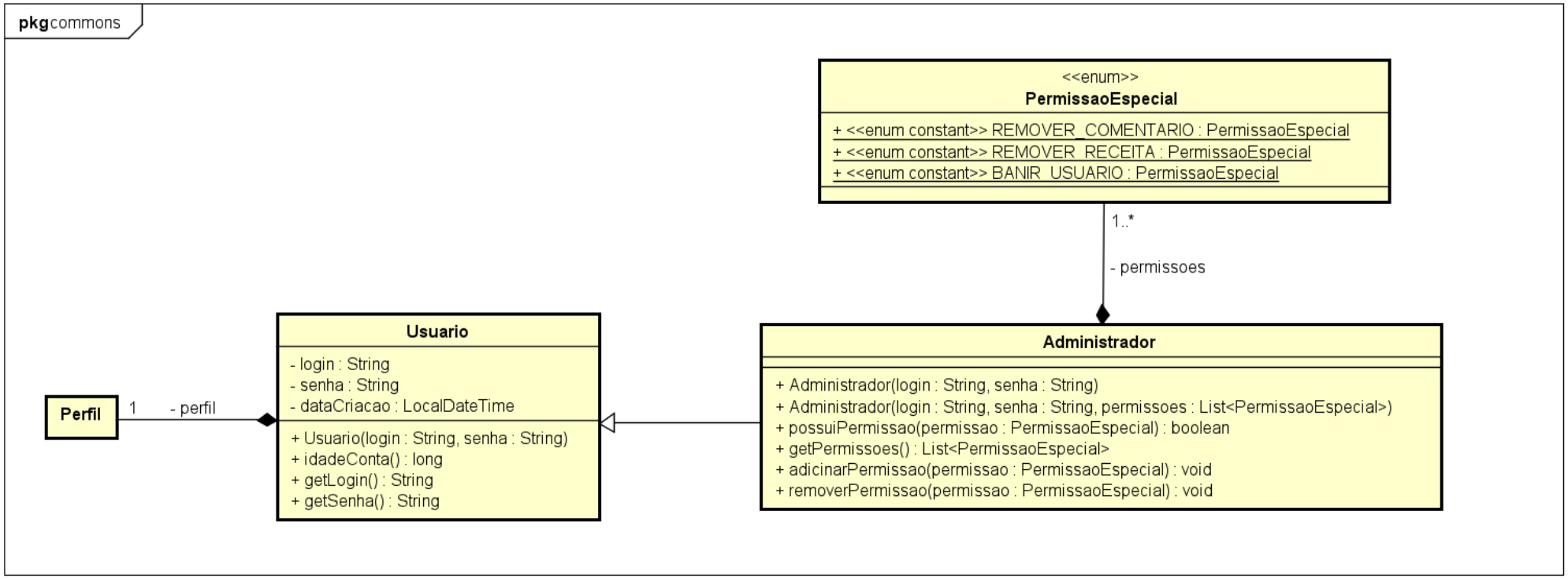


Perfil é a classe que define o usuário padrão, que cria receitas, compartilha cadernos, etc. Nela há os campos: *nomePerfil*, que é o nome que será exibido aos outros usuários; *biografia*, que serve para contar brevemente sobre si; *experienciaCulinaria*, que guarda o nível de habilidade da pessoa (dentre as possibilidades contidas no enum); *imagemPerfil*, da classe Imagem, para guardar a foto de perfil do usuárioe também guarda um título para a foto e a data em que foi adicionada (*dataCriacao*).

O campo *minhasReceitas* é uma composição com a classe **MinhasReceitas**, obrigatoriamente presente para todos os usuários e que não pode ser removida ou alterada diretamente pelo usuário. Nela são guardadas

todas as receitas presentes em todos os cadernos, exibindo o nome da receita e em qual caderno está, ou em quais cadernos estão, caso se trate de uma mesma receita presente em mais de um caderno. Cada nova receita, seja criada, seja copiada de um caderno que não seja do usuário, será adicionada automaticamente ao MinhasReceitas e só irá ser removida caso o próprio usuário remova a receita de seu (ou seus, em caso de multiplicidade da receita) respectivo(s) caderno(s).

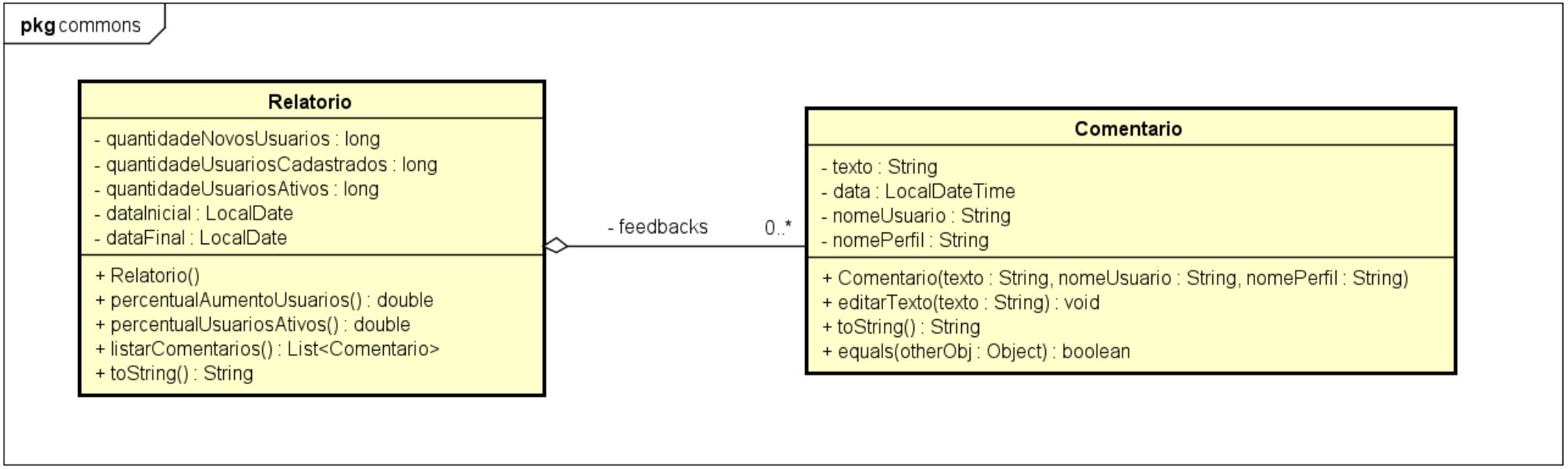
A Classe Usuario e suas associações



A classe **Usuario** é a classe mãe de Administrador. Ela possui 4 campos: *login*, *senha*, *dataCriacao* e *perfil*, caso seja uma conta de usuário padrão. Login e senha são usados para acessar a plataforma e *dataCriacao* guarda a data em que o usuário foi criado. Dos métodos da classe, o *idadeConta()* retorna há quanto tempo existe a conta.

Administrador é a classe filha de **Usuario**, portanto, herda os campos e métodos, além dos próprios métodos e o seu único campo, um enum de permissões, que dá a possibilidade ao administrador realizar ações no sistema que um usuário comum não pode, como remover comentários e remover uma receita. **Administrador** possui dois construtores, um caso já lhe seja passada uma List de permissões e um caso não seja passada, o que acarreta na necessidade de acessar o método *adicionarPermissao()* (que possui em contraparte, o *removerPermissao()*). Além desses métodos, há o *possuiPermissao()* para checar se o administrador possui a permissão necessária para efetuar uma ação (como as descritas acima).

A Classe Relatorio e suas associações



Vide o que foi colocado no *REQ6* da entrega 1 do projeto, ***a classe Relatorio existe para fornecer algumas informações cruciais acerca do sistema ao administrador em um determinado intervalo de tempo que ele escolher, quando solicitado***.

Ela possui 6 campos que são: *quantidadeNovosUsuarios*, *quantidadeUsusariosCadastrados*, *quantidadeUsuariosAtivos*, *dataInicial*, *dataFinal* e a lista dos comentários feito pelos usuários sobre o sistema.

Tem uma relação de agregação (ou seja, estamos enfatizando que se um relatório que possuía um certo comentario deixar de existir, esse comentário continua existindo) com a classe **Comentario**, que vai fornecer os feedbacks necessários para construir o relatório (detalhe é que como mostrado no UML acima, o relatório pode

também não conter nenhum feedback fornecido e ainda assim ser construído). Usuários administradores podem solicitar a criação de um **Relatorio** entre duas datas para o sistema (que será o responsável por criar um objeto **Relatorio**).

Possui um construtor e os seguintes métodos: *percentualAumentoUsuarios(), percentualUsuarioAtivos()*, *listarComentarios()* e o *toString()*.