UNIVERSIDADE FEDERAL DE CAMPINA GRANDE – UFCG

CENTRO ENGENHARIA ELÉTRICA E INFORMÁTICA– CEEI

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DISCIPLINA: LABORATÓRIO DE PROGRAMAÇÃO 2

PROFESSOR: MATHEUS GAUDENCIO DO RÊGO

GRADUANDO: DOUGLAS PEREIRA DE LIMA

MARCELLA MEDEIROS SIQUEIRA COUTINHO DE ALMEIDA

GABRIEL FELIPE CARDOSO GOMES

MARTA LAIS DE MACEDO DANTAS

RELATÓRIO - PROJETO QUEM ME AJUDA

CAMPINA GRANDE, 15 DE MARÇO DE 2018.

Link GitHub > Projeto_p2

Como parte da avaliação da disciplina Laboratório de Programação 2, tivemos de desenvolver o back-end de um sistema de tutorias, o qual permite o cadastro de aluno, a definição de alunos como tutores, a criação de ajuda, entre outras funcionalidades. Para tal desenvolvimento, aplicando os conhecimentos adquiridos nas cadeiras de Programação 2 e Laboratório de Programação 2, optamos por utilizar uma Facade que se liga a dois controllers, responsáveis por duas lógicas distintas do sistema: controllerAluno, que fica responsável por tudo que envolve um aluno, seja ele um tutor ou não, e controllerAjuda, responsável por tudo que envolve as ajudas, presenciais ou online.

US1

A US1 traz como especificação a possibilidade de cadastrar alunos e registrá-los através de um nome, uma matrícula, o código do curso, seu telefone e seu email, bem como ser possível recuperar esse aluno registrado através de sua matrícula. Para realizar tal tarefa, decidimos por criar uma classe Aluno e uma classe controllerAluno, responsável por realizar as lógicas de cadastrar um aluno, recuperá-lo, listá-lo e obter informações deste. (Marta e Gabriel)

US2

Este quesito solicita a definição de papel de um aluno para que este possa se tornar tutor, funcionalidade essa realizada através da matrícula do aluno, da disciplina a qual será tutor e de sua proficiência na mesma disciplina. Para isso, criamos uma classe Tutoria, bem como uma classe Disciplina. Tutoria é uma classe que é um atributo de Aluno e em Tutoria temos um Mapa que tem como valores objetos do tipo Disciplina. Realizamos essas escolhas de design ao perceber que um tutor é uma função que um aluno pode exercer ou não e criamos a coleção que guarda as disciplinas em Tutoria, pois um tutor pode ser tutor de mais de uma disciplina. Um Aluno é um tutor se o mapa em

tutoria não estiver vazio. Funcionalidades envolvendo certa lógica, como tornar tutor, recuperar tutor e listar tutores, foram realizadas em controllerAluno.(Douglas e Marcella)

US3

A US3 pede que sejam disponibilizados horários e locais para atendimento dos tutores. Para isso, encapsulamos todas essas informações em uma classe chamada Disponibilidade, que é armazenada em Tutoria, pois cada tutor tem a sua disponibilidade. Em Disponibilidade existe um mapa, onde as chaves são os dias da semana e os valores são sets de String, que contém os horários, e também existe um set de locais. (Marcella)

US4

Nessa US é solicitado para que seja possível cadastrar ajudas online e presencial, e com isso também ser possível recuperar informações sobre essa ajuda. Para isso, usamos herança da seguinte forma: criamos uma classe abstrata Ajuda, e criamos duas classes filhas, AjudaOnline e AjudaPresencial. Criamos também um ControllerAjuda que coordena toda a lógica que envolve as ajudas. Nesse controller existe um Arraylist de objetos do tipo Ajuda, onde podemos guardar os dois tipos de ajuda. Na classe abstrata Ajuda temos um método, também abstrato, getInfoAjuda, sendo responsabilidade das classes filhas o modo como esse método vai funcionar. (Douglas).

US5

O requisito desta parte é o de que, no sistema, possamos avaliar um tutor, após uma ajuda e dependendo da avaliação atual do tutor, ele terá um nível específico. Para isso, colocamos os métodos e atributos que envolvem essa lógica na classe Tutoria, respeitando o padrão Expert. Os controllers de ajuda e de aluno se comunicam para identificar qual tutor será avaliado. (Marcella)

US6

A US6 trazia como solicitação que fosse possível fazer doações aos tutores após realizarem alguma ajuda. Foram adicionados alguns métodos em e atributos em tutoria.

Pelo mesmo motivo da lógica da us5, isso foi feito pra respeitar o padrão Expert. Toda a lógica é realizada pelos dois controllers, para que a doação seja feita ao tutor de alguma ajuda específica. (Gabriel)

US7

A especificação da US7 pedia para que fosse possível mudar o parâmetro de ordenação da lista de tutores, onde podia se escolher entre: nome (padrão), email ou matrícula. Para isso, usamos o padrão strategy, tendo em vista que fizemos uso de objetos para determinar a estratégia de ordenação dos tutores. Da seguinte forma: criemos três classes que implementam o Comparator (NomeComparador, EmailComparador e MatriculaComparador) e adicionamos um atributo do tipo Comparator ao ControllerAluno, pois é ele que tem um mapa com valores do tipo aluno. Ao modificar o parâmetro de ordenação com o método configurarOrdem, criamos um objeto com um dos três tipos de comparadores e atribuímos esse objeto ao atributo do tipo Comparator no ControllerALuno, esse atributo é usado como parâmetro no método Collections.sort, que vai ordenar uma coleção de alunos que são tutores.(Douglas)

US8

A especificação da US8 pedia que todo o sistema desenvolvido até o momento pudesse ser persistido. Essa persistência contava com três métodos simples: gravar, limpar e carregar. A primeira dependência para a conclusão dessa tarefa foi a implementação da interface *Serializable* em todas as classes que foram definidas até o momento, isso se deve ao fato de que um objeto precisa ser serializável para que possa ser escrita em um arquivo. A serialização ocorre na Facade já que ela possui o atributo “sistema” que é o objeto o qual inclui todas as funcionalidades desenvolvidas até então Esses três métodos irão trabalhar escrevendo, deletando e lendo as informações de um arquivo de texto onde todo o sistema estará persistindo em forma de objeto.(Gabriel)

Considerações Finais

No geral, cada pessoa que ficou responsável por alguma parte, também ficou responsável por lançar exceções, testar e documentar essa parte. Caso alguém visualizasse algum erro, ou tinha uma sugestão a fazer em alguma parte que não era sua responsabilidade, essa questão era discutida em grupo.