**Fundamentos de Desenvolvimento de Software, turma 168**

**Prof. Bernardo Copstein**

# Trabalho Final

## Objetivo

O objetivo deste trabalho é avaliar a capacidade dos alunos em aplicar os conceitos apresentados ao longo do semestre a saber: sistemas de gerência de configuração, sistemas de automação da compilação, desenvolvimento de software em equipe, teste unitário, padrões de projeto e arquitetura de software (em especial arquitetura CLEAN). Para tanto os estudantes, organizados em grupos, deverão desenvolver um sistema conforme com as características descritas neste documento. Neste sistema deverão evidenciar sua capacidade de desenvolvimento em equipe e de aplicação dos conceitos apresentados. Serão avaliados tanto os artefatos entregues pelo Moodle como a apresentação do sistema feita através do software Zoom em horário a ser agendado com o professor. O tema do trabalho é livre desde que atenda aos requisitos não funcionais apresentados a seguir. O tema do trabalho, entretanto, deverá ser apresentado e aprovado pelo professor até a data definida no cronograma.

## Requisitos

Deverá ser desenvolvido um sistema Web de tema livre. Entretanto o sistema deverá atender aos seguintes requisitos não funcionais:

* Ser composto por dois módulos: um módulo cliente (front-end) e um módulo servidor (back-end). Para efeitos de demonstração os dois módulos podem executar em um único equipamento (localhost) porém será valorizado caso o sistema seja instalado em um servidor remoto e puder ser acessado pela WEB.

Esse requisito foi parcialmente atendido, há back-end e front-end, mas estes rodam somente em localhost.

* O módulo cliente deve ser “magro”, isto é, deve-se restringir aos seguintes aspectos: apresentação da interface do usuário, tratamento dos eventos de usuário e coleta de informações nos componentes de interface, solicitação de serviços para o back-end enviando os parâmetros informados pelo usuário, exibição de resultados e gestão do fluxo de telas. Todas as operações relativas à lógica da aplicação deverão ser executadas no back-end.

Os requisitos atendidos no parágrafo imediatamente anterior estão destacados em azul, e os não-atendidos em vermelho.

* O módulo servidor deverá ser construído segundo os preceitos da arquitetura CLEAN. O nível de “Entidades” deverá conter pelo menos 3 entidades distintas; o nível de casos de uso deverá explorar o padrão fachada e conter pelo menos duas políticas distintas; o nível de interface deverá abrigar a classe responsável por expor os “endpoints” e as responsáveis por abrigar os mecanismos de persistência se houverem.

O nível de entidades contém somente uma entidade, denominada ‘Receita’; o nível de casos de uso não possui políticas distintas e é composto por duas classes: RepositorioReceitas.java e ServicoReceitas.java. O nível de interfaces abriga a classe ClienteFachadaRemota.java, responsável por expor os endpoints. Os mecanismos de persistência do sistema utilizam tecnologias JPA e banco de dados MySQL.

* Padrões de projeto deverão ser explorados de maneira a garantir os princípios SOLID. É necessário justificar de que maneira cada um dos princípios foi garantido. Independente disso é obrigatório explorar, pelo menos, os seguintes padrões de projeto: “fachada”, “strategy”, “builder” e injeção de dependências.

A classe ServicoReceitas.java explora o padrão de projeto ‘fachada’, através da anotação @Service; os padrões ‘strategy’ e ‘builder’ não foram explorados; já a injeção de dependências foi explorada, através da anotação @Autowired em duas classes distintas, ClienteFachadaRemota.java e ServicoReceitas.java.

* Devem ser gerados drivers de teste para todas as classes dos níveis de entidade e casos de uso. As exceções são as classes do tipo “DTO” para as quais não é necessário gerar driver de teste. Os drivers de teste devem garantir pelo menos 100% de cobertura de linhas e de cobertura de condição de cada uma das classes. Tal condição deve poder ser comprovada pelos relatórios do Code Coverage ou equivalente (o uso de um framework de teste unitário e de uma ferramenta de análise de cobertura de código são obrigatórios). Pelo menos um driver de teste deve explorar testes parametrizados.

Por motivos de força maior, não foi possível implementar testes.

* O sistema deve ser desenvolvido em grupo com auxílio dos softwares Git e GitHub. Deverá ser possível demonstrar, durante a apresentação, que todos os integrantes do grupo foram capazes de criar seus próprios “branches” em submeter atualizações no “branch master”. Será avaliado se todos os integrantes do grupo tiveram as tarefas igualmente distribuídas.

O sistema foi desenvolvido por um único aluno, sem esforços colaborativos, baseado nas aulas dadas, materiais disponibilizados pelo Professor e após intensa pesquisa na Web. O sistema também foi desenvolvido com auxílio do GitHub, repositório individual de projetos gerenciados pelo Git, sendo o repositório oficial do trabalho localizado no endereço Web <https://github.com/lucaspiat/Recipes>. Não foram criadas branches adicionais, sendo a branch padrão a ‘master’.