



Técnicas de Programação

Trabalho Final 2021/1

Objetivo

O objetivo deste trabalho é explorar a aplicação dos 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 descrição abaixo. Neste sistema deverão evidenciar sua capacidade de desenvolvimento em equipe e de aplicação dos conceitos apresentados.

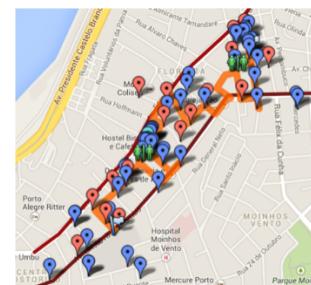
Serão avaliados tanto os artefatos entregues pelo Moodle como a apresentação do sistema feita através do software Zoom em horário a ser agendado com o professor. Os requisitos apresentados são mínimos e os grupos têm a liberdade de propor alterações ou novas funcionalidades. Alterações “drásticas”, entretanto, deverá ser apresentado e aprovado pelo professor até a data definida no cronograma.

Enunciado do problema:

“Encontrou um problema na sua cidade?

Avise a população e a Prefeitura sobre os problemas da cidade: buracos na rua, trânsito, terrenos com mato alto, saúde, educação, obras paradas e mais.

Este é o lugar para tornar sua reclamação pública. Você mostra o problema, juntos cobramos a Prefeitura.” (App Cidadera, 2015)



Requisitos funcionais

O sistema deve permitir os seguintes casos de uso por parte de usuários “registrados”:

- Abrir reclamação. Para abrir uma reclamação o usuário deve informar: um título, uma breve descrição do problema que está relatando, a data, bairro e endereço onde foi verificado o problema. O usuário também deve categorizar o tipo de problema encontrado, como por exemplo: iluminação pública, preservação de vias públicas, preservação de calçadas, etc. Opcionalmente o usuário pode adicionar uma imagem (um link para uma imagem ou realizar o upload da imagem).
- Atualizar as informações das suas próprias reclamações;
- Comentar reclamações postadas por outros usuários ou mesmo as suas. Ao comentar uma reclamação o usuário poderá adicionar novas imagens e, também, poderá marcar a reclamação como “Resolvida”. Ao serem cadastradas as reclamações têm sempre o status “Aberta”.



O sistema deve possuir um usuário administrador, que pode realizar as seguintes operações:

- Habilitar contas de “usuários oficiais” que serão disponibilizados para respostas oficiais por parte da prefeitura municipal
- Acessar um módulo de informações gerenciais que permite gerar:
 - o total de reclamações por categoria/bairro num determinado período
 - o número médio de comentários (filtradas por categoria, bairro ou período)
 - o percentual de informações com status “resolvido” ou “encerrado” por categorias, ou bairros
 - o percentual de reclamações respondidas (e encerradas) por órgãos oficiais
 - outros relatórios que eventualmente o grupo julgar de interesse da aplicação

Os “usuários oficiais” são contas de usuário criadas para que a prefeitura municipal possa se comunicar com os cidadãos e permite:

- Comentar reclamações de usuários e eventualmente mudar o status de uma reclamação para “Encerrada”. Uma reclamação encerrada por órgão oficial não pode mais ser comentada nem atualizada pelo seu criador.

Requisitos não funcionais

Deverá ser desenvolvido um sistema Web que atenda 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.
- O módulo cliente pode ser (bastante) “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 “backend” 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 “backend”.
- O módulo servidor deverá ser construído segundo os preceitos da arquitetura CLEAN.; 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.
- 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”, “builder” e injeção de dependências.
- 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.

- 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.

Cronograma de entregas

Os entregáveis que não correspondem a código fonte devem ser armazenados em uma pasta chamada “OutrosEntregaveis” a ser criada na raiz do projeto no “GitHub”. Para as apresentações intermediárias é necessária a presença de apenas um dos integrantes do grupo no momento síncrono. Na data da apresentação final todos os integrantes devem estar “presentes”. Caso algum grupo perca alguma entrega/apresentação intermediária deverá contatar o professor para agendar horário alternativo. Os atrasos terão impacto na avaliação final.

Data	Entrega
26/05	Organização dos grupos na área moodle e envio do link do projeto no GitHub. O projeto deve já estar corretamente configurado. O projeto deve ser criado privativo e o professor da disciplina convidado como colaborador (alexandre.agustini@pucrs.br ou aagustini).
01/06	Data limite para propor alterações nos requisitos do trabalho. Deve ser entregue um arquivo contendo a especificação e alterações de requisitos do sistema de será desenvolvido. A entrega deve conter um cronograma das sprits do trabalho, backlog de atividades e “mocks” das telas. A definição do trabalho deverá ser armazenada no projeto do GitHub, enviada por email e apresentada na aula do dia 09/07. A aprovação ou não das sugestões de alteração ficarão registradas na Wiki de acompanhamento dos trabalhos no Moodle.
08/06	Entrega do diagrama de classes e pacotes demonstrando a arquitetura do sistema. O diagrama deverá ser encaminhado na área moodle disponibilizada e apresentado na aula. A aprovação do diagrama ficará registrada na Wiki de acompanhamento dos trabalhos no Moodle.
15/06	Entrega das classes do nível de entidades com os respectivos drivers de teste. As classes deverão estar disponíveis no GitHub do projeto e deverão ser apresentadas em aula. Um email de notificação deve ser enviado tão logo as classes estejam disponíveis. O acompanhamento ficará registrado na Wiki de acompanhamento dos trabalhos no Moodle.
17/06	Entrega de documento explicitando as tecnologias a serem usadas na aplicação cliente e na aplicação servidora. O documento deverá ser encaminhado na área moodle disponibilizada e apresentado em aula para o professor. A aprovação das tecnologias ficará registrada na Wiki de acompanhamento dos trabalhos no Moodle.
01/07	Entrega final do trabalho e apresentação do projeto durante a aula.
06/07	Apresentação do projeto para aqueles em que não houve tempo hábil na aula do dia 02