PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

PUC Minas Virtual

Pós-graduação *Lato Sensu* em Arquitetura de *Software* Distribuído

Projeto Integrado

Relatório Técnico

Sistema de Gestão de Ocorrência - SGO

Thiago Gutenberg Carvalho da Costa

Belo Horizonte
Fevereiro 2022

Projeto Integrado – Arquitetura de Software Distribuído

Sumário

1. Introdução

Segundo o Dicionário Escolar (2015, p. 284) a infraestrutura nada mais é que um conjunto de serviços básicos (de saneamento, de transporte, de energia e de telecomunicação) tornando-se uma área fundamental para o desenvolvimento socioeconômico de uma região. Nessa perspectiva, à medida em que centros urbanos se expandem, percebe-se a necessidade de manutenção dessas estruturas que compõem uma infraestrutura urbana, se tornando um ponto vital e de extrema importância para a não depreciação das mesmas e também para o bem estar da comunidade.

Neste contexto, o processo de solicitação de manutenção em estruturas dos serviços públicos (poste de luz, canos de distribuição de água, asfalto, etc.) é feito de forma geral unicamente via canal telefônico, onde além da restrição de horário de atendimento sendo somente em horário comercial, limita-se o registro de uma ocorrência distinta por vez para cada intervenção corretiva, mesmo sendo em uma mesma localização. Portanto, mediante este problema, indaga-se: diante de enorme incidência de reclamações em uma região, o processo de registro de ocorrência está sendo feito de forma desburocratizada e eficiente?

Então, afim de simplificar o processo de registro de incidentes pela população e também com o intuito de modernizar, ampliar e prover mais eficiência e transparência, há a necessidade de implementação de um Sistema de Gestão de Ocorrência onde seja possível a abertura de um chamado para resolução de vários problemas relacionados a um tipo de serviço em um determinado endereço.

Sendo assim, o objetivo geral deste trabalho é apresentar a descrição do projeto arquitetural de uma aplicação de gerenciamento de ocorrências de manutenção em estruturas públicas para uma administração regional.

Onde os objetivos específicos são:

- Implementar serviço de ocorrências, parte responsável por expor APIs Web em conformidade com o padrão arquitetural RESTful provendo interoperabilidade para realização do processamento negocial e de persistência de uma ocorrência;

- Implementar serviço de localização, que através da comunicação com um sistema externo será responsável por obter informações sobre um endereço a partir de um CEP;
- Implementar serviço de notificações, cuja responsabilidade será o processamento e disparo assíncronos de eventos de notificação relacionados a situação de uma ocorrência;
- Implementar serviço web, responsável por permitir que uma pessoa possa interagir com o sistema, a princípio através de um navegador web com uma interface gráfica responsiva;
- Implementar serviço de atendimento, que será responsável por classificar/priorizar uma ocorrência e também por alterar a situação da ocorrência que foi registrada no sistema, aprovando ou devolvendo a mesma;
- Implementar serviço de relatórios, responsável por disponibilizar relatórios com resumo e/ou estatísticas de atividades realizadas aos interessados.
- Implementar serviço de ordem de serviço, módulo responsável por se integrar com o sistema externo que gerencia ordem de serviços. Quando uma ocorrência for aprovada, o sistema de O.S deverá ser acionado para que inicie os tramites para executar o(s) serviço(s) de manutenção descrito(s) na ocorrência.

2. Cronograma do Trabalho

A seguir é apresentado o cronograma proposto para as etapas deste trabalho.

Datas		**Atividade / Tarefa**	**Produto / Resultado**
De	**Até**		
01/01/2022	07/01/2022	1. Analisar ideias e selecionar uma para ser abordada e servir como a base para elaboração do relatório técnico.	Tema central do projeto integrado.
08/01/2022	15/01/2022	2. Apresentar informações gerais, referentes ao escopo do projeto, mais especificamente quanto ao contexto, a problemática e a motivação do mesmo.	Introdução, objetivo geral e objetivos específicos do projeto.
16/01/2022	22/01/2022	3. Definir os requisitos arquiteturais da solução.	Descrição dos requisitos que podem impactar diretamente ou indiretamente na macro arquitetura da solução.
23/01/2022	01/02/2022	4. Definir os requisitos funcionais da solução.	Descrição do fornecimento de serviços do sistema e como o mesmo deverá se comportar a partir de entradas específicas.
02/02/2022	06/02/2022	5. Definir os requisitos não-funcionais da solução.	Especificar restrições às funcionalidades do sistema.
07/02/2022	07/02/2022	6. Definição dos mecanismos arquiteturais.	Descrição de conceitos técnicos para padronizar aspectos do sistema.
13/02/2022	14/02/2022	7. Criar o diagrama de contexto.	Diagrama de alto nível que mostra de uma forma geral a macroarquitetura da solução no modelo C4.
13/02/2022	14/02/22	8. Vídeo de apresentação inicial.	Apresentação inicial do projeto, ideia completa sobre o projeto.
16/02/2022	19/02/2022	9. Criar o diagrama de container.	Diagrama que detalha o sistema, mostra os bancos de dados, serviços e descreve as tecnologias usadas.
20/02/2022	28/02/2022	10. Criar o diagrama de componentes.	Diagrama com detalhes de um serviço, mostra camadas de integração de componentes na base de código.
16/02/2022	16/05/2022	11. Iniciar o desenvolvimento do projeto.	Protótipo funcional contendo o em seu escopo os requisitos prioritários.
01/03/2022	08/03/2022	12. Criar o diagrama de código.	Diagrama de detalha a composição de classes definidas em um componente, refletindo diretamente o código.
16/05/2022	31/05/2022	13. Avaliar a arquitetura através do modelo ATAM (*Architecture Tradeoff Analysis Method*).	Análise da abordagem da arquitetura com os cenários e apresentação dos resultados.

01/06/2022	08/06/2022	14. Conclusão.	Lições aprendidas e pontos de melhorias na arquitetura.
09/06/2022	11/06/2022	15. Vídeo de apresentação final.	Apresentação completa do projeto realizado, focando a arquitetura.

3. Especificação Arquitetural da solução

3.1 Restrições Arquiteturais

R1: Seguir o padrão arquitetural de microsserviços;

R2: As APIs devem estar no padrão RESTful preferivelmente seguindo o princípio HATEOAS (Hypermedia as the Engine of Application State) para prover melhor interoperabilidade;
R3: O back end deve ser implementado utilizando o conjunto de ferramentas para padrões comuns em sistemas distribuídos providos pelo projeto Spring Cloud;
R4: Os componentes que compõem a interface gráfica do sistema devem ter comportamento responsivo para se adequar melhor nas dimensões de tela dos dispositivos dos usuários;
R5: O sistema deve ser complacente à LGPD (Lei Geral de Proteção de Dados).

3.2 Requisitos Funcionais

ID	Descrição Resumida	Dificuldade (B/M/A)*	Prioridade (B/M/A)*
RF01	O sistema deve permitir que um usuário externo se registre através de um formulário de cadastro indicando nome de usuário, e-mail, primeiro nome, último nome e senha.	M	A
RF02	O sistema deve permitir que o usuário se autentique através de um formulário de acesso (login) onde deverá ser informado um nome de usuário e senha previamente cadastrada.	A	A
RF03	O sistema deve permitir o registro de uma ocorrência por um usuário.	A	A
RF04	O sistema deve permitir a alteração de detalhes de uma ocorrência registrada, caso a mesma tenha sido devolvida pelo atendente para ajustes.	M	A
RF05	O sistema deve permitir que uma ocorrência registrada possa ser cancelada pelo usuário que a criou caso a mesma não esteja com situação já aprovada.	B	B
RF06	O sistema deve permitir que sejam vinculados vários incidentes de um mesmo tipo de serviço em uma ocorrência durante o processo de registro.	M	M
RF07	O sistema deve permitir anexar foto para cada incidente vinculado a uma ocorrência durante o processo de registro.	A	B
RF08	O sistema deve permitir o acompanhamento de uma ocorrência através de uma pesquisa a partir de um número do protocolo.	B	A

RF09	O sistema deve listar os protocolos de ocorrências registradas para o usuário logado.	B	M
RF10	O sistema deve permitir que um atendente analise e classifique priorizando uma ocorrência em uma escala de urgência de: Baixo, Moderado ou Alto.	B	A
RF11	O sistema deve permitir que um atendente altere a situação de uma ocorrência para: Aprovada, Cancelada, Devolvida.	B	A
RF12	O sistema deve permitir a buscar automaticamente um endereço a partir de um CEP informado pelo usuário durante cadastro de uma ocorrência.	M	A
RF13	O sistema deve permitir que o usuário selecione uma localidade a partir de um mapa durante cadastro de uma ocorrência.	A	B
RF14	O sistema deve permitir que o usuário possa desligar o recebimento de notificações via e-mail sobre a situação de uma ocorrência registrada.	M	A
RF15	O sistema deve enviar notificações via componente de interface (badge) sobre a situação da ocorrência para o usuário.	M	M
RF16	O sistema deve enviar notificações via e-mail para o usuário sobre a situação da ocorrência caso o usuário opte por receber esse tipo de notificação.	M	M
RF17	O sistema deve permitir gerar relatórios referente as localizações que mais tiveram ocorrências registradas.	M	A
RF18	O sistema deve permitir gerar relatórios sobre a quantidade de horas de participação de um atendente nas ocorrências - TMA (Tempo Médio de Atendimento)	M	M
RF19	O sistema deve permitir gerar relatórios sobre a quantidade de horas em que uma ocorrência ficou em aberto até ser atendida - TME (Tempo Médio de Espera)	M	M
RF20	O sistema deve notificar o sistema externo de ordem de serviço para gerar uma O.S (Ordem de Serviço) quando uma ocorrência mudar para situação aprovada.	M	A

*B=Baixa, M=Média, A=Alta.
Obs.: acrescente quantas linhas forem necessárias.

3.3 Requisitos Não-funcionais

ID	Descrição	Prioridade B/M/A
RNF01	Desempenho ─ A operação de salvar uma ocorrência tem que ser rápida, não podendo exceder mais do que 3 segundos em média para ser executada.	A
RNF02	Usabilidade ─ A interface gráfica deve ser responsiva aplicando um design adaptativo para melhorar a disposição do conteúdo na tela do dispositivo melhorando a experiência do usuário.	M
RNF03	Compatibilidade ─ O sistema deverá funcionar corretamente nos navegadores web modernos mais usados (Google Chrome, Edge e Firefox).	B
RNF04	Segurança ─ O acesso a APIs privadas precisa ser seguro, exigindo a devida autenticação e autorização do usuário, caso contrário resultará respectivamente nos erros HTTP 401 - (*Unauthorized*) ou HTTP 403 (*Forbidden*).	A
RNF05	Padrão ─ O software deverá ser orientado a camadas possuindo baixo acoplamento e alta coesão seguindo o princípio de Responsabilidade Única para prover melhor manutenibilidade e evolução.	A
RNF06	Confiabilidade ─ Rastreamento distribuído para coleta e pesquisa de dados de tempo de requisições para detectar possíveis problemas de latência na arquitetura de serviços deverá ter um percentual de amostragem de 100%	A

*B=Baixa, M=Média, A=Alta.
Obs.: acrescente quantas linhas forem necessárias.

3.4 Mecanismos Arquiteturais

Esta seção descreve os mecanismos que compõem a arquitetura do software fazendo uma ligação entre a definição conceitual da solução (análise) e sua implementação, passando pela visão de desenho.

Análise	*Design*	Implementação
Segurança - Gestão de Identidade e Acesso	Método de acesso - Single-Sign On (SSO)	Keycloak
Segurança - Autenticação e Autorização	Protocolo de autenticação interoperável	OpenID Connect – OIDC
Persistência – Abstração	Spring Data JPA (Java Persistence API)	Hibernate

Persistência – Relacional	Sistema de gerenciamento de banco de dados (SGBD)	MySQL
Persistência – Não-relacional	Banco de dados distribuído e baseado em documentos	MongoDB
Versionamento – Ferramenta	Sistema de controle de versões distribuído	Git
Versionamento – Sistema online	Sistema online de controle para desenvolvimento de software	GitHub
Front end – Estratégia de implementação web	Single Page Application	Angular
Front end – Componentes	Biblioteca de Componentes de Interface do Usuário	Angular Material
Back end - Framework	Framework de aplicação	Spring
Back end - Ferramentas	Padrões comuns para sistemas distribuídos (Serviço de registro e descoberta, rastreamento distribuído)	Spring Cloud
Integração – Assíncrona	Broker de mensageria	RabbitMQ
Integração – Síncrona	Web Service - REST API	Comunicação de requisição/resposta via protocolo *Hypertext Transfer Protocol* (HTTP)
Integração - Rastreio de eventos distribuídos	Coletar dados de tempo necessários para solucionar problemas de latência em arquiteturas de serviço	Zipkin
Log do sistema	Simple Logging Facade for Java (SLF4J) API	Logback
Teste de Software	Framework para teste unitário	JUnit 5
Teste de Software – Mock	Framework de mocking para teste unitário	Mockito
Deploy – Distribuição	Unidade padrão de software que empacota o código e todas as suas dependências de forma autocontida.	Docker Container
Deploy – Orquestração	Orquestrar ambientes distribuídos baseados em containers	Kubernetes

4. Modelagem Arquitetural

Esta seção apresenta a modelagem arquitetural através de diagramas hierárquicos de forma a permitir completo entendimento em diferentes níveis de abstração da solução proposta visando à implementação da prova de conceito (seção 5).

4.1 Diagrama de Contexto

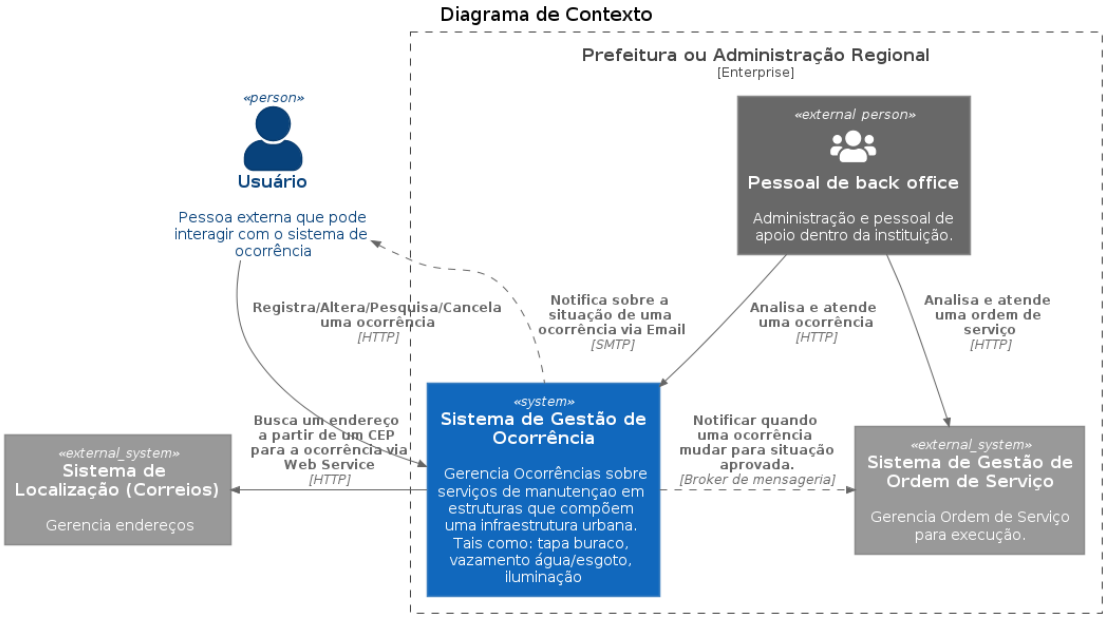

Figura 1 - Visão Geral da Solução. Fonte: https://www.infoq.com/br/articles/C4-architecture-model/

A figura 1 descreve a macroarquitetura do Sistema de Gestão de Ocorrência através do diagrama de contexto, mostrando como o sistema se encaixa no mundo em termos das pessoas que o utilizam e dos outros sistemas de software com os quais ele interage. Em azul está o Sistema de Gestão de Ocorrência que será construído, usuários externos e internos podem interagir com ele afim de registrar ou atender uma ocorrência. Em cinza, são os sistemas de software externos que já existem no qual o Sistema de Gestão de Ocorrência irá interagir com eles para buscar uma localização ou para notificar sobre a situação de uma ocorrência para que seja gerada uma ordem de serviço.

4.2 Diagrama de Container

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 2. Apresente o Diagrama de *Container* da aplicação, indicando como os componentes (aplicativos, armazenamentos de dados, microservices, etc.) que compõem esse sistema de software estão distribuídos e organizados. Lembre-se que as decisões de tecnologia que você tomou devem ser contempladas nesse diagrama.

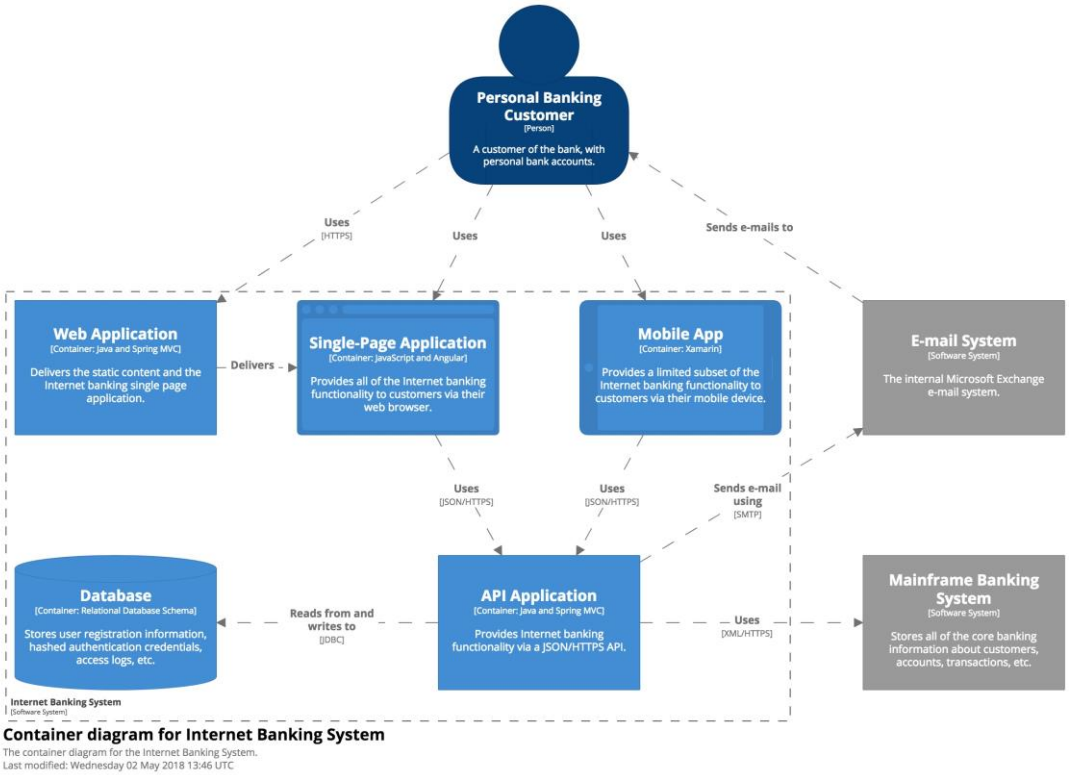

Figura 2 – Diagrama de container. Fonte: https://www.infoq.com/br/articles/C4-architecture-model/

Obs: esta é uma figura da literatura. Substitua-a por outra elaborada por você, que seja adequada ao seu projeto. Lembre-se que cada arquitetura é única.

A figura 2 apresenta os *containers* da aplicação...

4.3 Diagrama de Componentes

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 2. Apresente o Diagrama de Componentes da aplicação (baseado na UML), indicando os elementos da arquitetura e as interfaces entre eles. Liste os estilos/padrões arquiteturais utilizados e faça uma descrição sucinta dos componentes indicando o papel de cada um deles dentro da arquitetura/estilo/padrão arquitetural. Indique também quais componentes serão reutilizados (navegadores, SGBDs, middlewares, etc), quais componentes serão adquiridos por serem proprietários e quais componentes precisam ser desenvolvidos.

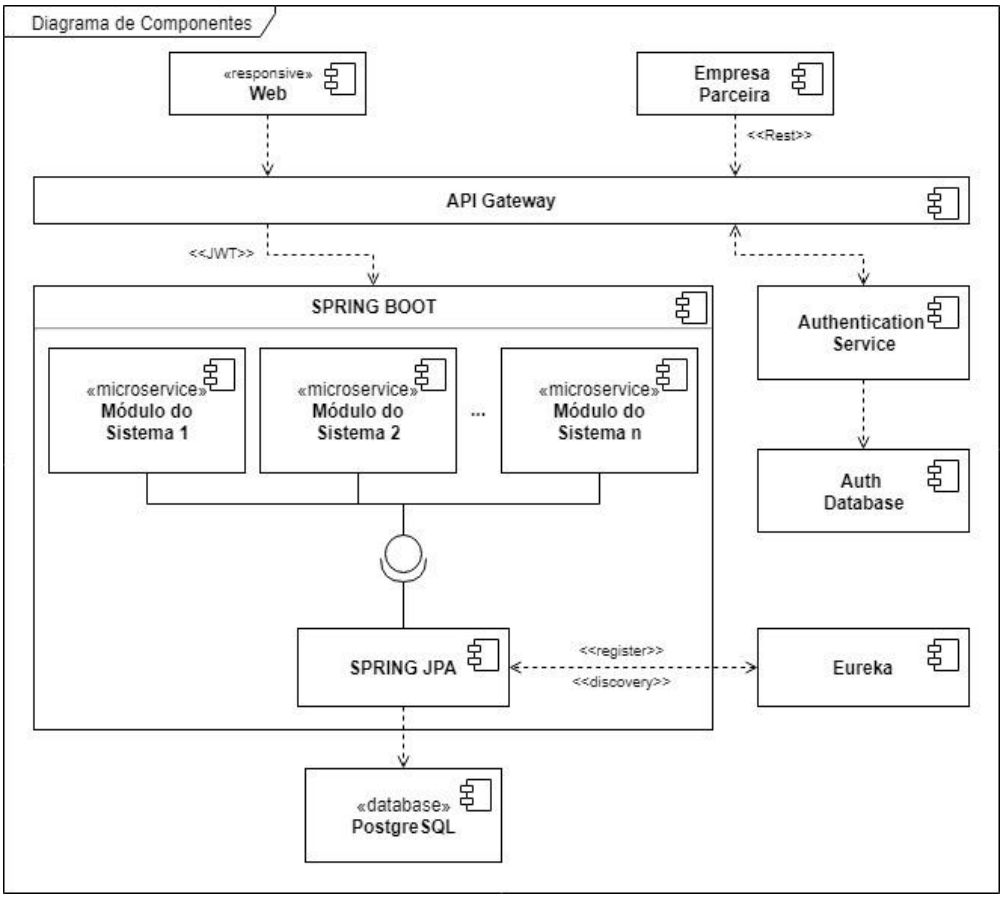

Figura 2 – Diagrama de Componentes (exemplo). Fonte: imagens Google.

Obs: esta é uma figura da internet. Substitua-a por outra elabora por você, que seja adequada ao seu projeto. Lembre-se que cada arquitetura é única.

Apresente uma descrição detalhada dos artefatos que constituem o Diagrama de Componentes. Lembre-se de apresentar no diagrama todos os componentes da aplicação.
Ex: conforme diagrama apresentado na Figura 3, as entidades participantes da solução são:

- *Componente 1 - Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras nunc magna, accumsan eget porta a, tincidunt sed mauris. Suspendisse orci nulla, sagittis a lorem laoreet, tincidunt imperdiet ipsum. Morbi malesuada pretium suscipit.*

- *Componente 2 - Praesent nec nisi hendrerit, ullamcorper tortor non, rutrum sem. In non lectus tortor. Nulla vel tincidunt eros.*

5. Prova de Conceito (PoC)

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 2. Nesta seção deve ser detalhada a prova de conceito arquitetural produzida. Não é necessário desenvolver todo o escopo da aplicação proposta, mas deve-se gerar um protótipo que permita avaliar as funcionalidades relativamente aos requisitos arquiteturais definidos, segundo o modelo ATAM (seção 6).

Deve-se produzir também um **vídeo de apresentação do protótipo** criado, disponibilizando-o a os professores envolvidos no processo de avaliação do trabalho.

5.1 Integrações entre Componentes

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 2. Desenvolva um protótipo navegável e interativo do sistema, que apresente as interfaces (pode-se utilizar *mock objects*) e as integrações (protocolos, *middlewares*, padrão de troca de dados, etc) entre **os três requisitos prioritários selecionados**. A forma de comunicação entre os componentes por meio dessas integrações deve ser completamente detalhada. Pode-se utilizar alguma ferramenta para a apresentação dessas integrações, além de simulações dos mecanismos de comunicação em ambiente distribuído.

5.2 Código da Aplicação

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 2. Nesta seção você deve indicar, segundo o **padrão arquitetural C4**, a estrutura de código da sua aplicação. Exemplo:

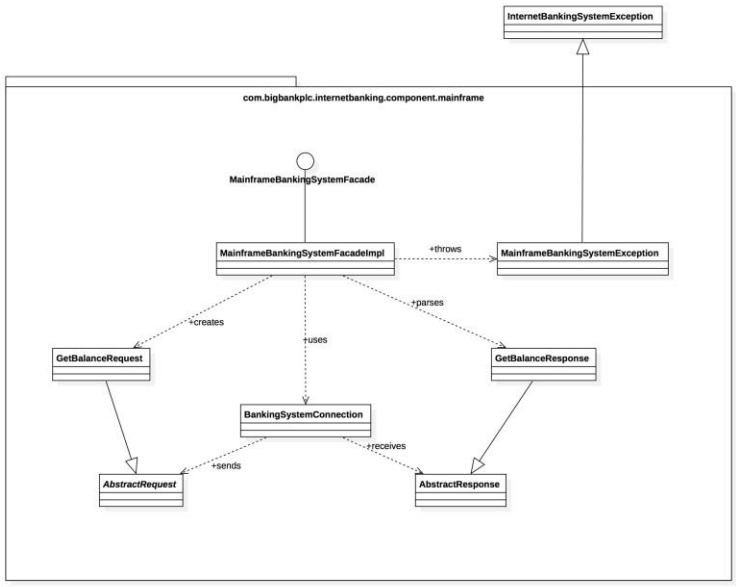

Figura 4 – Estrutura de código da aplicação (exemplo). Fonte: https://www.infoq.com/br/articles/C4-architecture-model/

Obs: esta é uma figura da internet. Substitua-a por outra elabora por você, que seja adequada ao seu projeto. Lembre-se que cada arquitetura é única.

A estrutura da aplicação mostrada na Figura 4 apresenta os componentes de código e suas funções no software implementado:

- xxxxxxxxxxxxxx
- xxxxxxxxxxxxxx
- etc...

Indique o ***link* do vídeo e do repositório** (como o GitHub, Bitbucket, etc) onde seu protótipo funcional está disponível. Deve ser disponibilizado um *link* que forneça acessa à página principal da sua aplicação (menu principal), bem como usuário e senha de acesso para eventuais testes.

Como indicado no início desta seção 5, grave um **vídeo de apresentação do seu projeto**. Espera-se a produção de um vídeo sintético, de **no máximo**, **5 minutos**, no formato **MP4** ou outro de ampla utilização, apresentando o projeto e a solução desenvolvida. Pede-se que o aluno coloque o foco da apresentação nas questões arquiteturais mais relevantes, que envolvem o atendimento aos requisitos arquiteturais definidos na seção 3.3.

6. *Avaliação da Arquitetura (ATAM)*

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 3. A avaliação da arquitetura desenvolvida neste trabalho é abordada nesta seção visando avaliar se ela atende ao que foi solicitado pelo cliente, segundo o método ATAM.

6.1. *Análise das abordagens arquiteturais*

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 3. Apresente aqui um breve resumo das principais características da proposta arquitetural. Para isto, utilize o método Architecture Tradeoff Analysis Method (ATAM), no qual são utilizados cenários para fazer essa análise.

Exemplo:

Atributos de Quailidade	Cenários	Importância	Complexidade
Interoperabilidade	Cenário 1: O sistema deve se comunicar com sistemas de outras tecnologias.	A	M
Usabilidade	Cenário 2: O sistema deve prover boa usabilidade.	M	B
Manutenibilidade	Cenário 3: O sistema deve ter a manutenção facilitada.	M	M

6.2. *Cenários*

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 3. Mostre os cenários utilizados na realização dos testes da sua aplicação. Escolha cenários de testes que demonstrem os requisitos não funcionais (atributos de qualidade) sendo satisfeitos. Priorize os cenários para a avaliação segundo critérios quantitativos ou qualitativos.

Exemplos de cenários:

Cenário 1 - Interoperabilidade: Ao acessar a URL do serviço de informações gerenciais via HTTP GET, o mesmo deve retornar as informações no formato JSON.

Cenário 2 - Usabilidade: Ao navegar na tela, o sistema deve apresentar boa usabilidade. A navegação deve apresentar facilidade e o acesso as funcionalidades deve ser bem objetivo para a função que precisar ser realizada, o usuário deve ser capaz de efetuar uma compra em no máximo 5 minutos, assim garantindo a agilidade e a usabilidade para ficar de acordo com um dos requisitos não funcionais.

Cenário 3 - Manutenibilidade: Havendo a necessidade de alterar o gateway de pagamento somente será necessário fazer alteração no broker da funiconalidade de pagamento, facilitando a manutenção e os testes.

6.3. Evidências da Avaliação

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 3. Apresente as medidas registradas na coleta de dados. Para o que não for possível quantificar apresente uma justificativa baseada em evidências qualitativas que suportem o atendimento ao requisito não-funcional.

Atributo de Qualidade:	Interoperabilidade
Requisito de Qualidade:	O sistema deve se comunicar com outras tecnologias.
Preocupação:	
O sistema deve ter como resposta a uma requisição uma saída de fácil leitura por outro componente.	
Cenário(s):	
Cenário 1	
Ambiente:	
Sistema em operação normal	
Estímulo:	
O sistema de monitoramento envia uma requisição para o serviço REST do módulo de informações gerenciais.	
Mecanismo:	
Criar um serviço REST para atender às requisições do sistema de monitoramento	
Medida de resposta:	
Retornar os dados requisitados no formato JSON	
Considerações sobre a arquitetura:	
Riscos:	Alguma instabilidade na rede pode deixar a conexão lenta ou mesmo a perda de pacotes.

Pontos de Sensibilidade:	Não há
Tradeoff:	Não há

Acrescente imagens e descreva os testes realizados, de tal forma que se comprove a realização da avaliação.

Faça isto para todos os cenários apresentados no tópico 6.1.

6.4. Resultados Obtidos

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 3. Apresente os resultados da arquitetura produzida, indicando seus pontos fortes e suas limitações. A título de sugestão construa uma tabela apresentando esses resultados, como no exemplo que segue:

Requisitos Não Funcionais	Teste	Homologação
RNF01: O sistema deve ...	OK	OK
RNF02: O sistema deve ...	OK	N.A.
RNF03: ...	OK	N.A.

Obs: N.A.: não se aplica.

7. Avaliação Crítica dos Resultados

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 3. Apresente aqui, de forma resumida, os principais pontos positivos e negativos da arquitetura proposta. Adote uma postura crítica que permita entender as limitações arquiturais, incluindo os prós e contras das tecnologias. Você pode utilizar o formato textual ou produzir um quadro resumo.

Ex. de quadro resumo:

Ponto avaliado	**Descrição**
xxxxxxxxxxxxxxx	xxxxxxxxxxxxxxxxxxxx

8. Conclusão

Conforme o cronograma de trabalho, essa parte será feita futuramente na Etapa 3. Descreva, de forma sucinta, quais foram as lições aprendidas na execução do seu projeto arquitetural. Procure apresentá-las de tal forma que fiquem configurados os *trade-offs* da arquitetura produzida, como por exemplo, Segurança X Desempenho, Granularidade X Manutenibilidade, etc.

Aqui deve ser apresentado também tudo que se aprendeu com esse projeto, de modo a servir como ajuda para outros profissionais.

Também se faz necessário evidenciar as possibilidades de melhoria do projeto, caso se deseje dar continuidade a ele. Nesse sentido, indique possíveis ajustes ou melhorias arquiteturais, que possam vir a ser realizados futuramente.

Lições aprendidas (ex.):

1. xxxxxxxxxxxxxxxxx
2. xxxxxxxxxxxxxxxxx
3. xxxxxxxxxxxxxxxxx

Referências

BROWN, Simon. **O modelo C4 de documentação para Arquitetura de Software.** Tradução de: Marcelo Costa. InfoQ, São Paulo, 01, ago. de 2018. Arquitetura. Disponível em: https://www.infoq.com/br/articles/C4-architecture-model. Acesso em: 13, fev. 2022.

PLANTUML. **Drawing UML with PlantUML**: PlantUML Language Reference Guide. Brest: PlantUML, 2021 p. 390. Disponível em: https://plantuml.com/guide. Acessado em: 14, fev. 2022.

MIHALCEA, Vlad. **High-Performance Java Persistence.** Victoria: Leanpub, 2021.

APÊNDICE

URL da apresentação da Etapa 1: https://youtu.be/GZdCEQZyvSQ