DCC 802 – Projeto e Implementação de Sistemas

Discente: Jean Bertrand, João Pedro, Victor Rocha, João Paulo, Larissa Santos, Allan Cordeiro, Francisco Pires Junior, Hermino Junior, Gabriel Carvalho, Talles Bezerra de Assunção, Ibukun Chife Didier, Felipe Derkian

### Subdivisão do Modelo Cascata

1. **Análise**

Será elaborado um questionário com 15 perguntas, para serem feitas na entrevista com o cliente afim de colher os requisitos necessários para que possamos elaborar um sistema que atenda a suas especificações da forma mais abrangente possível.

1. **Projeto**

Elaborar um mapeamentos dos processos e realizar modelagem dos dados elaborando um protótipo inicial das telas, apresentar o protótipo ao cliente, com o objetivo de estudar e implementar o sistema de acordo com as regras pré-estabelecidas. Reformulação do protótipo assim deixando de forma clara para os programadores o que deverá ser realizado, através de um diagrama UML.

1. **Desenvolvimento**

Divisão do time em grupos focados no front-end e no back-end. Esse vai criar as rotas da aplicação e a conexão com o banco de dados e aquele vai digitalizar as telas criadas pelo time de design.

1. **Teste**

Analisar o aplicativo desenvolvido para que possíveis bugs sejam corrigidos enquanto estão sendo desenvolvidos, com o objetivo de corrigir as falhas antes que o produto final fique pronto.

1. **Manutenção**

Realizar a correção de bugs e/ou erros especificados pelos testadores e usuário.

#### Artefatos que serão produzidos no projeto

#### • Questionário de requisitos;

• “Manual do Usuário” onde terá um tutorial contendo o passo a passo de como utilizar o sistema;

• Protótipos do APP (somente as telas);

• Relatório de Requisitos do Sistema;

• Relatório de Requisitos Gráficos;

• Relatório de erros e bugs encontrados no sistema;

• Diagrama ER;

• Diagrama Casos de uso;

• Diagrama de Classes;

• Diagrama de Atividades;

• Diagrama de Sequências;

• Descrição do APP;

• Código Aberto do Sistema;

• Documentação do Código.

#### Ferramentas e padronizações:

Será utilizado o google docs para realizar todos os relatórios e documentos textuais para o referente projeto;

* Ferramentas e Padronizações Construção dos diagramas UML:

Os diferentes diagramas que compõem a UML podem ser agrupados em categorias, levando em

conta para isto o contexto em que cada uma dessas representações pode vir a ser empregada:

* Diagramas Estruturais: priorizam a descrição estática de estruturas de um sistema, como classes, atributos e operações destas últimas, além de prováveis relacionamentos entre tais construções. A Tabela 1 lista os diversos diagramas que pertencem a esta classificação;
* Diagramas Comportamentais: detalha o funcionamento (comportamento) de partes de um sistema ou processos de negócio relacionados a tal aplicação. Na Tabela 2 são apresentados os diversos diagramas que se enquadram nesta categoria;

Os diagramas utilizados nesse projeto, seriam os seguintes diagramas:

**Tabela 1:** Diagramas Estruturais da UML

|  |  |
| --- | --- |
| **Diagrama** | **Função** |
| Diagrama de  Casos de Uso | Voltado à apresentação de funcionalidades e características de um sistema,  assim como de que forma tais elementos se relacionam com usuários e  entidades externas envolvidas num determinado processo. |
| Diagrama de  Atividades | Contempla as diversas tarefas desempenhadas na execução de uma atividade,  sendo utilizado geralmente na representação de processos dentro de uma  empresa/organização. |

**Tabela 2:** Diagramas Comportamentais da UML

|  |  |
| --- | --- |
| **Diagrama** | **Função** |
| Diagrama de  Sequência | Demonstra as interações entre diferentes objetos na execução de uma  operação, destacando ainda a ordem em que tais ações acontecem num  intervalo de tempo. A sequência em que as diversas operações são executadas  ocorre na vertical, de cima para baixo. |

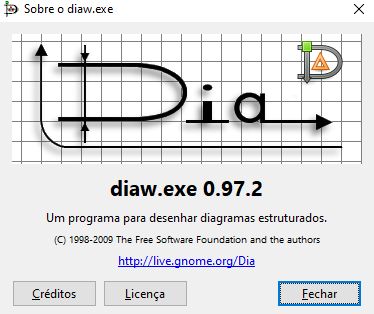
Um outro diagrama que será ser utilizado é o diagrama de de classes. Pois este representa

um conjunto de classes e suas relações.

|  |  |
| --- | --- |
| **Diagrama** | **Função** |
| Diagrama de Classes | Permite a visualização de um conjunto de classes, detalhando atributos e  operações (métodos) presentes nesta última, assim como prováveis  relacionamentos entre essas estruturas. Este tipo de representação pode incluir  ainda definições de interfaces. |

O software que usaremos para construção de diagramas baseados nas notações da UML será:

* Dia Diagram Editor



* Desenvolvimento do projeto de interface;

• balsamiq - Ferramenta utilizada para primeira prototipação da interface, com um certo nível de navegação. (Versão 4.1.4)

• marvelapp - Ferramenta onde será feito a reformulação da interface, de forma mais completa com ícones e cores, e uma navegação mais elaborada. (Versão 2.3.8)

* Time de programação:

A IDE será o Visual Studio Code (Versão 1.49.2). A linguagem será Javascript no framework React (Versão 16.13.1) e React Native (Versão 0.63). O banco de dados vai ser o MongoDB, que é NOSQL. A padronização do código vai ser feita com as ferramentas ESLint (Versão 7.10.0), para definir as regras, e o Prettier (Versão 2.1.2), que formata o código baseado nessas regras. Utilizaremos para controle de versão e gerenciamento de tarefas o Github.

* Testes de Documentação, para verificar os gráficos de navegação, layouts de tela e outros requisitos no design; Testes Funcionais, para garantir que o aplicativo esteja funcionando de acordo com os requisitos definidos; Testes de Usabilidade, para verificar três critérios básicos: satisfação, eficiência e eficácia; Teste de Interface do Usuário, para garantir a conformidade com os padrões e capacidade de resposta do aplicativo em diferentes dispositivos; Testes de Desempenho, para determinar a operabilidade, estabilidade e consumo de recursos; e Teste Beta, para identificar o número máximo de erros para sua posterior eliminação antes do lançamento oficial do aplicativo. Será utilizado a norma ISO/IEC 9126 que é norma para qualidade de produto de software que visa tanto a qualidade interna quanto externa do software quanto a Funcionalidade, Confiabilidade, Usabilidade, Eficiência, Manutenibilidade.

**Referências**

Catho. **Analista de Sistemas**. Disponível em: <https://www.catho.com.br/profissoes/analista-de-sistemas/>. Acesso em 26 de Setembro de 2020.

EasyQA. **Como testar aplicativo móvel**. Disponível em: <https://geteasyqa.com/pt/qa/mobile-apps-testing/>. Acesso em 28 de Setembro de 2020.

RIBEIRO, Silas G. **Padronização de código com EsLint, Prettier e VSCode**. Disponível em: <https://medium.com/@guimaraessilas/padroniza%C3%A7%C3%A3o-de-c%C3%B3digo-com-eslint-prettier-e-vscode-e5f7f9cd3bb>. Acesso em 26 de Setembro de 2020.

DevMedia. **Modelagem de sistemas através de UML: Uma visão geral**. Dísponivel em: <https://www.devmedia.com.br/modelagem-de-sistemas-atraves-de-uml-uma-visao-geral/27913>. Acesso em 30 de Setembro de 2020.