UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS DE CRATEÚS

ENGENHARIA DE SOFTWARE

TRABALHO DIRIGIDO 05

VICTOR JONES MESQUITA DE SOUSA – 511418

SUMÁRIO:

INTRODUÇÃO:

1) **uma descrição do tema do seu projeto:**
   R: O projeto que eu escolhi apresentar para participar da competição é um chamado de R-View ele é um editor de pdf onde o usuário pode realizar a visualização, edição e o salvamento do arquivo editado. Cada usuário poderá escolher várias formas de edição, podendo ser pincel livre, imagens ou formas geométricas. Também pode ser feita a escolha da cor dos objetos, tamanhos e pode ser realizada translações, rotação e escalonamento dos objetos. O salvamento do arquivo é feito em pdf, e é um projeto de livre acesso, ou seja sem taxa de uso.

VERIFICAÇÃO E VALIDAÇÃO:

2) **uma descrição completa sobre como você pretende verificar e validar o código do seu software;**
   R: A verificação e validação seria um trabalho para os QA da equipe, então ele iria fazer todos os testes disponíveis que temos no mercado, começando por um code Review junto com o techlead do time, e a análise do código para ver se está nos padrões SOLID, se está indentado corretamente, se o padrão de definições de nome

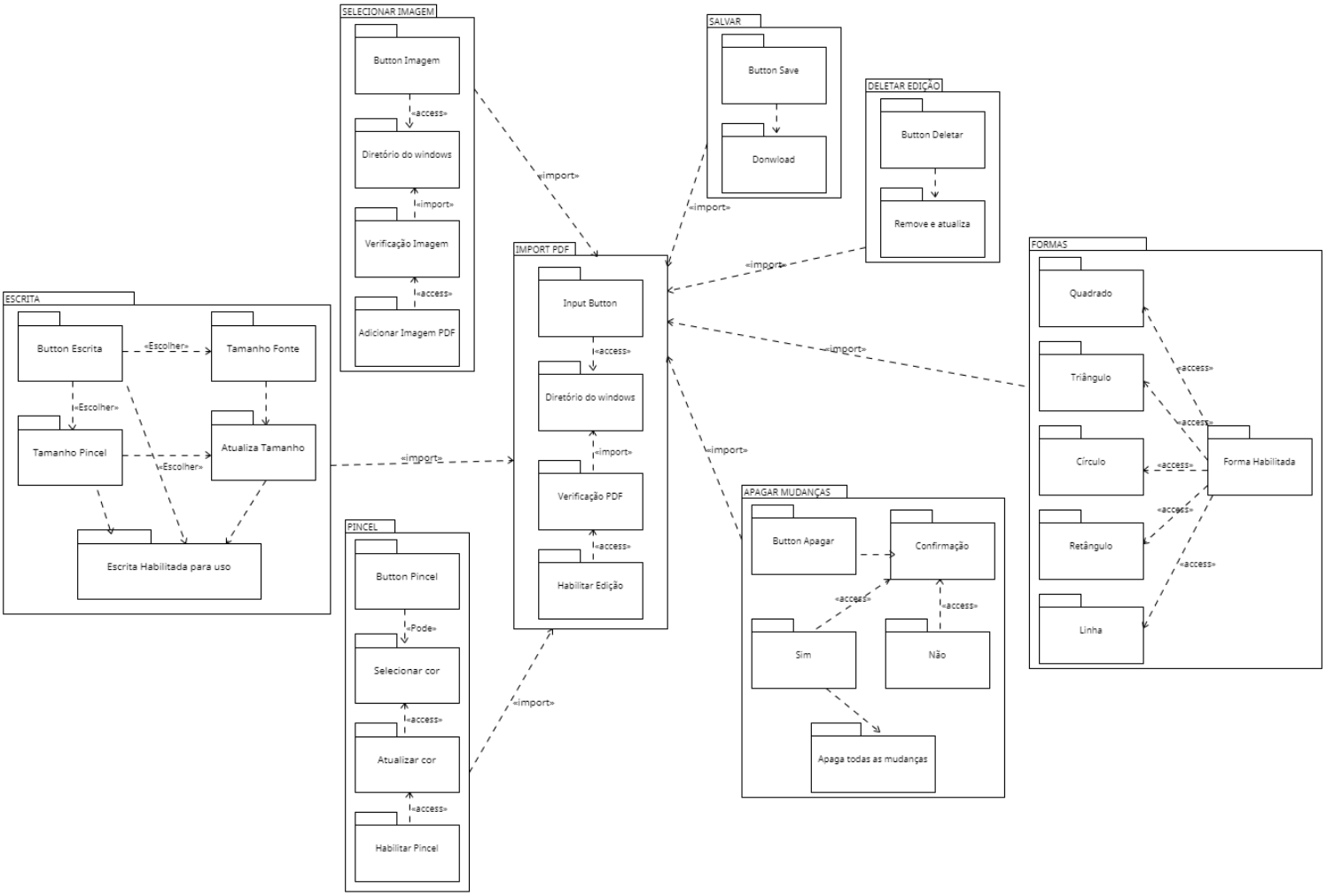
está correto, como por exemplo o uso de camel e pascal case, então todas essas validações são iniciais. Após isso poderíamos fazer uma verificação “automática” onde seriam feitos testes unitários da aplicação, essa é uma etapa onde é feito a validação de componentes separados, individualmente, inclusive geralmente é feito usando um mock base, que é um banco de dados “fake”. Após isso poderíamos fazer os testes de integração que ao invés de testar individualmente cada função/método fazemos a validação de um modulo inteiro, ou de um fluxo inteiro, ou seja ele testa a integração dos testes unitários um com o outro. Poderíamos fazer também testes com os próprios usuários do sistema, então poderia ser feita uma versão do software onde a gente iria disponibilizar para um grupo de possíveis cliente e então esse grupo retornaria um feedback do software, essa é uma abordagem muito utilizadas em jogos, mas que poderia ser utilizada aqui também. E por fim poderíamos disponibilizar todo o sistema para uma empresa que realiza auditorias de software, ou seja uma empresa que é feita para realizar esses testes e encontrar problemas. **Portanto,** acredito que com essas validações e verificações o nosso software seria bem visto como um projeto solido e íntegro.

SOLID:

3) **como você empregaria os padrões SOLID no seu projeto;**
   R: Eu iria seguir todos os conceitos que compõem o SOLID, então eu iria fazer com que cada classe tivesse uma responsabilidade única e iria separar as regras de negócio da lógica de apresentação, para isso que server o fron-end e bank-end, justamente para fazer essa separação e manter um certo padrão. Com isso temos o uso do **Single Responsibility Principle**. Assim como uma classe iria ter responsabilidade única, ela também seria aberta para extensão, porem fechada para modificação, e também iria utilizar os conceitos de POO, então iria fazer uso de herança, polimorfismo etc. Com isso temos o uso do **Open/Close principle.** Tambem vamos fazer com que as classes derivadas, ou herdadas que elas possam ser usadas no lugar de suas classes bases sem afetar o comportamento do programa. Vamos tentar evitar a sobrecarga de métodos também, pois quanto mais sobrecarga mais diferente do método base fica, e isso não é recomendado nos padrões SOLID. Com isso temos o **Liskov Principle.** Além desses cuidados que vamos ter quando usar as classes, também temos que ter com as interfaces, então vamos fazer com que uma classe só implemente a interface caso ela faça seu uso. E fazer uma modularização maior dessas interfaces, então ao invés de ter uma interface “mãe”, vamos ter várias menores. Com isso garantimos o **Interface segregation Principle.** Quando for usar módulos devemos nos atentar se o modulo de mais alto nível não tem uma dependência de um modulo de mais baixo nível. Pois ambos têm devem depender apenas de abstrações, caso seja necessário pode se fazer o uso de injeção de dependência. Com isso garantimos o **Dependency Inversion Principle.**
   Portanto, temos o uso do padrão SOLID no nosso projeto.

ARQUITETURA E DIAGRAMAS DE PACOTES:

4) **um diagrama de pacotes para a sua aplicação;**
   R: Ficou meio pequeno para ver por aqui então eu upei a imagem google drive só clicar no link ou copiar e colar.
   Link:
   https://drive.google.com/file/d/1h1ImWR7RFoFFw81mwXSG0FyszcroqGau/view?usp=sharing



IMPLANTAÇÃO:

5) **quais serão suas estratégias de implantação do software descrevendo também a fundamentação teórica das suas escolhas.**
   R: Bom, na parte de desenvolvimento eu iria começar fazendo uma pesquisa para determinar as melhores tecnologias que se encaixa melhor ao projeto, essa pesquisa seria bem ampla, então eu iria analisar bibliotecas, frameworks, funções, padrões de projeto, scripts etc. Iria fazer também a analise se é um produto escalável, responsivo e métodos de otimização. Após isso teríamos um grande documento bem estruturado com tudo que devemos e vamos utilizar no projeto, isso é bem útil pois economiza possíveis problemas no futuro e em caso de uma equipe grande, deixaria todos os integrantes da equipe em uma mesma página para começar o

desenvolvimento. A partir disso poderíamos passar para o desenvolvimento em si, onde como por exemplo teríamos uma equipe de 4 pessoas, onde seriam 1 TechLead do projeto que é aquele responsável por passar as demandas para os outros 3 integrantes, iria fazer o code Review de todo o time, então para isso ele teria que ser o desenvolvedor mais experiente da equipe, além disso ele seria o responsável por determinar o tempo de desenvolvimento de cada task e sua complexidade. Teríamos mais 2 desenvolvedores sendo 1 front-end e outro back-end, o front iria fazer toda a parte visual do projeto, e para isso ele teria que ter o domínio das tecnologias que está sendo usada para o front-end, no nosso caso vamos usar o HTML e Css como ferramentas de front-end. O desenvolvedor back-end iria fazer todas as funções/métodos/classes do projeto, além do banco de dados, para isso ele teria que ter o domínio das tecnologias de back-end que como exemplo usaríamos JavaScript, Node e Mysql para o banco de dados. E por último teríamos um QA na equipe sendo o responsável por todo o teste do projeto, sendo feitos testes unitários e de integração, além de auxiliar o techlead quando for subir a aplicação para produção fazendo o papel de sustentação. Acredito que com essa equipe de 4 pessoas, iriamos conseguir fazer a implementação do projeto e apresentar um produto viável e utilizável para o público consumidor.

CONSIDERAÇÕES FINAIS:

Portanto, concluímos o projeto de software que eu apresentaria, onde fazemos o uso de boas práticas de programação, seguimos os padrões SOLID que são muito importantes para manutenções futuras, flexibilidade e extensões. Fazemos o uso das atuais tecnologias do mercado, também usaríamos os métodos de teste mais completos possíveis para ter um software íntegro e solido para seus usuários. O projeto seria desenvolvido em uma equipe relativamente pequena de 4 pessoas, temos também o diagrama de pacotes para se ter uma noção mais ampla do fluxo da aplicação.