Relatório de CES-29
CASD-Vest – Processos Ágeis

18 de junho de 2018

Disciplina: CES-33

Estudante: Felipe Guimarães, Lucs Jorge

Turma 19

Instituto Tecnológico de Aeronáutica

I. Introdução

Nesse documento, estarão relatadas as práticas relacionadas aos processos ágeis usados no projeto da disciplina de CES-29 onde trabalhamos no site da entrevista de renda do vestibulinho do CASD Vest. A equipe é composta por Felipe Uchida, Felipe Guimarães, Dennys Rocha, Lucas Jorge, Daniel Rocha ,Talize Facó e Lucas Nogueira.

II. Divisão da Equipe

A equipe foi dividida em duas frentes: o back end e o front end. O front end do projeto foi responsabilidade dos alunos Felipe Guimarães e Lucas Jorge. O aluno Daniel Rocha foi o Scrum Master do projeto. Os demais alunos foram responsáveis pelo back end, junto com o Daniel Rocha.

Durante as sprints, a equipe de front end foi responsável por realizar as tarefas e estórias relacionadas a UX do site e seu design, trabalhando com HTML, CSS e javascript.

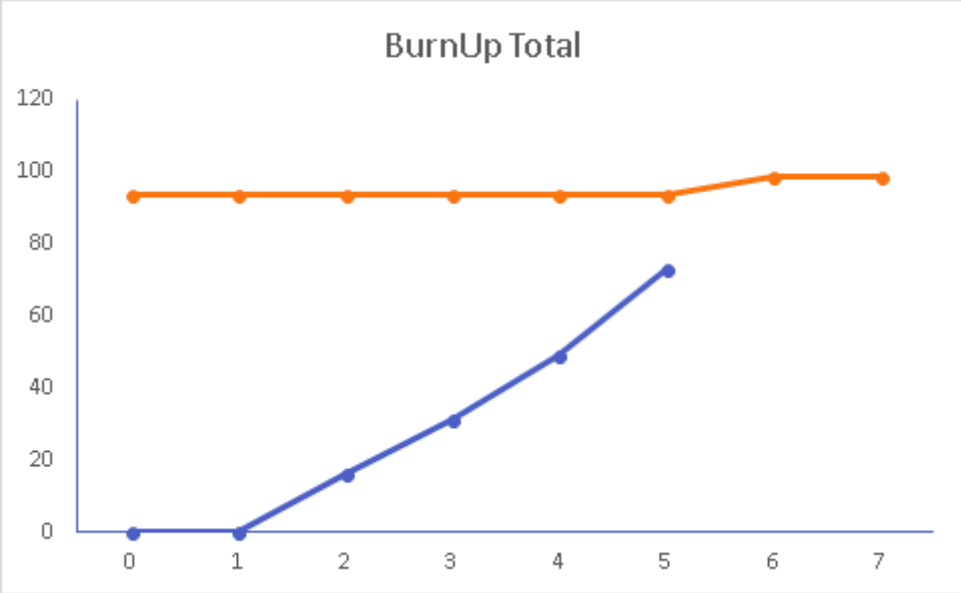
III. User Stories e Métricas Adotadas

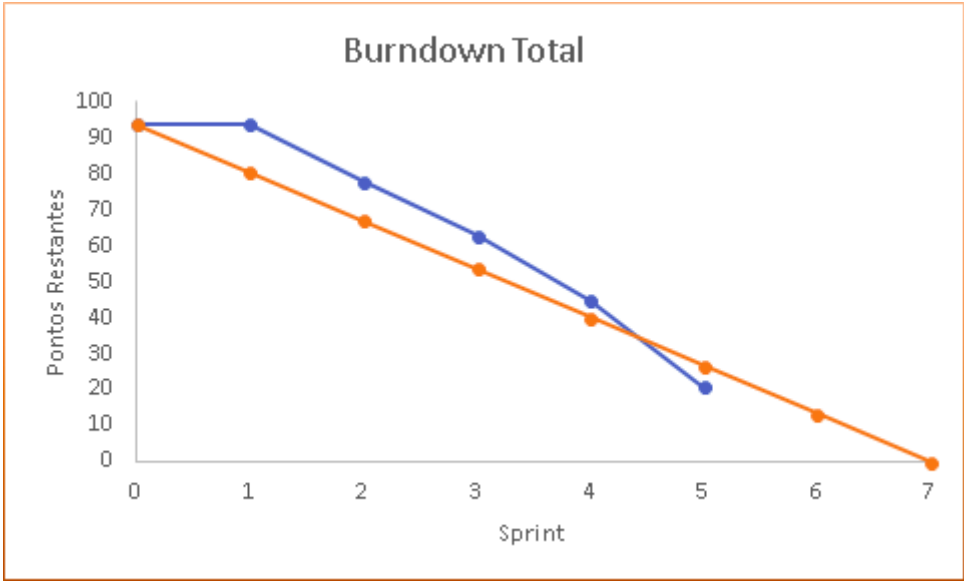
Na organização do projeto, usadmos o Trello para organizar e monitorar as user stories do projeto. Nos cartões de cada user story, está também definida a sua respectiva pontuação que mede a quantidade de tempo necessária para a atividade em específico e para construir os gráficos de burn up e burn down.

Para cada sprint, temos a tabela no trello de suas stories e suas pontuações, assim como a equipe responsável por elas. Na imagem abaixo estão as sotories concluídas até então com suas pontuações e responsáveis indicados pela cor do label da story.

IV. Planejamento e Execução das Sprints

Os gráficos de burn-up e burn-down foram construídos a partir das pontuações das user stories vistas anteriormente. Eles estão apresentados abaixo.





V. Ferramentas de Testes

A ferramenta adotada para cobertura de código foi o PHPUnit. Os testes unitários nesse framework funcionam de forma análoga ao JUnit, já trabalhado anteriormente na disciplina de Introdução à Engenharia de Software (CES-28).

Optou-se pelo PHPUnit porque ele é amplamente utilizado por desenvolvedores globalmente, de forma que existe uma grande base de conhecimentos em forums, blogs e na documentação sobre o framework, facilitando o esclarecimento de dúvidas que surgissem durante o projeto.

Exemplos de testes realizados foram os testes de conexção com o DB. Nesse exemplo, criou-se uma classe mockada ‘Service’ que, quando tem seu método ‘openDb’ chamado, retorna o

Callback ‘openDbCallback’ por um stub. Então, verificou-se com o método assertEquals se a classe testada, chamando o método ‘opendDb’, retornava o mesmo resultado da classe mockada: $this->assertEquals(ServiceTest::openDB(), $mockedServiceClass::openDb());. Na verdade, devido à dificulade de mockar um método estático em PHP, criou-se um artifício para que a classe testada retornasse seu Db truncado durante os testes, e esse era comparado com o Db truncado retornado por stub na classe mockada.

No final, conseguiu-se testar um número razoável de funções para ter segurança sobre o bom funcionamento do back-end. Por garantia, testou-se desde as funções mais simples, como setters e getters e producers.

VI. Lições Aprendidas

Os métodos ágeis são uma ótima forma de se organizar um projeto de computação, pois divide exatamente as tarefas o todos do grupo sabiam exatamente o que fazer. Porém, como não estávamos trabalhando full time no projeto, algumas coisas acabam sendo prejudicadas devido à não sincronização da equipe como seria o caso do trabalho em uma empresa onde reuniões diárias e semanais sobre o projeto ocorreriam.

Algumas tarefas acabavam ficando para a véspera do sprint ou no máximo 5 dias anteriores por conta desses motivos: o time precisava estar sincronizado para trabalhar e tínhamos trabalhos de outras disciplinas para entregar e fazer durante o período do sprint.

Por fim, as sprints são muito eficientes quando usadas do jeito correto e no ambiente correto. Aprendemos a trabalhar em equipe em um projeto de engenharia e a resolver tarefas objetivas de maneira eficiente. Porém, o essa maneira de se gerenciar um projeto não é tão eficiente quando a equipe não trabalha diariamente nele e não está sincronizada, como acabou sendo o nosso caso.