Slide 1: Capa

* Apresentar-se
* Apresentar o tema muito por alto

***[Iurie Marcinschi]***

***Boa tarde,***

***Antes de mais agradecemos aos Srs. Engenheiros em aceitar o nosso do projeto como projeto final de curso, e de nos dar esta oportunidade de o apresentar.***

***Os participantes no progeto são o Iurie Marcinschi, … sou eu…, e Pavel Egorov o meu colega!***

***O projeto a apresentar tem como enfase a integração continua no desenvolvimento de applicações, e possui o nome de “Plataforma de Integração Continua” .***

Slide 2: Agenda

* Falar no plano da apresentaçao

***[Iurie Marcinschi]***

***O plano da apresentação terá o:***

***Enquadramento - em que vamos falar no âmbito de integração continua em que o projeto visa se posicionar.***

***Motivção que nos levou a optar por um tal projeto e os problemas com que actualmente se deparam uma grande parte de equipas de desenvolvimento.***

***Objetivos que foram definidos por alcançar em conformidade com as necessidades no ambito de integração continua.***

***Solução - onde falaremos em feramentas ao nosso dispor que permitiram atingir o objetivo, nos componnentes da aplicação desenvolvida, e interação sobre os mesmos e o resultado a saida.***

***No capitulo Arquitetura vamos aboradar os componentes mais em pormenor, nas suas propriedades e proposito dos mesmos.***

***Por fim vamos efetuar uma demonstração video da utilização então do produto desenvolvido e concluir com identificação das limitações do mesmo e possiveis desenvolvimentos futuros.***

Slide 3: Enquadramento

* Falar acerca da Integração continua e o seu papel no que toca o desenvolvimento em equipa
* Falar da importância dos testes e construção do código frequentes

***[Iurie Marcinschi]***

***No contexto do desenvolvimento de software a tarefa de cada membro da equipa traduz-se diariamente na realização de integrações de código, onde cada integração é verificada por meio de testes unitários. Assim sendo, a integração contínua é uma prática comum no dia-a-dia de um programador.***

***A prática mostra que integrar e testar com mais frequência diminui a relevância dos conflitos e erros na base de código comum. Na década de 90 a compilação diária do código tornou-se uma prática frequente. No início dos anos 2000 esta prática foi levada ao extremo empregando a integração contínua e a validação das integrações com testes unitários.***

Slide 4: Motivação/Problema

* Falar em desvantagens praticas em efetuar os testes e compilações de código frequentes em termos de recurços humanos e tempo despendido
* Falar em custos materiais e tempo em montar um ambiente com propriedades diferentes necessarios

***[Iurie Marcinschi]***

***Trazendo estes benificios a integração continua também implica custos tanto a nivel de recurços humanos como a nivel de ambientes para os testes e tempo para sua montagem.***

***Ou seja, numa situação ideal, seria bom existir uma pessoa que a medida que é efectuada uma submição de alterações de código, execute os testes para se certificar que as alterações não afetaram o correto funcionamento do produto como um todo.***

***Alem disto os custos também se aplicam a preparação de ambientes para executar o código, isto é, montar ambiente, configurar, migrar o código, e muitas vezes, dependendo de projeto para projeto, é necessario testar o produto em ambientes com propriedades diferentes, como por exemplo abientes com vesões de java diferentes.***

***Ou seja, estes problemas deduzem-se em tempo e dinheiro.***

Slide 5: Objetivo

* Falar em ter necessidade da existência de um serviço que automatiza o processo de testes
* Falar em ter necessidade em ter um sistema que disponibiliza ambientes facilmente configuraveis
* Falar da necessidade de ter um feedback o mais rapido possivel no impacto das alterações no projeto

***[Iurie Marcinschi]***

***Nestas condições surje a necessidade da existência de um serviço que consiga resolver este trabalho de forma autómatica.***

***A partir destas necessidades nasceu a ideia de criar um serviço na internet que saiba montar ambientes a medida e executar codigo de forma autómatica.***

***Quando falamos em ambientes a medida, o que se pretende é que o sistema, com alguma configuração prévia do utilizador, saiba montar e desponibilizar ambientes isolados de execução de forma rapida na sua criação e ao mesmo tempo facilmente descartaveis no fim de cada execução.***

***Tendo um ambiente montado o sistema deve saber importar o código a partir do sistema de verções GitHub, e executar os comandos sobre o mesmo dentro do ambiente montado. Estes comandos deveram ser indicados pelo utilizador na configuração prévia do projeto.***

Slide 6: Solução

* Falar em feramentas/tecnologias que existem ao nosso dispor que permitiram atingir o objetivo
* Explicar o fluxo de comunicação entre componentes e o resultado produzido

Slide 7: Arquitetura

* Falar nas caracteristicas especificas da arquitetura implementada (fracamente acoplada, auto-suficiente e escalavel)

Slide 8 e 9: Componente Web Server

* Falar na arquitetura conceptual do mesmo, componentes com que interage
* Falar no fluxo de ações por parte do Web server na criação de um trabalho e interação com o cliente

Slide 10: Componente Worker

* Falar na arquitetura conceptual do mesmo, componentes com que interage
* Falar no fluxo de ações do mesmo na criação de um trabalho e criação de artifato

Slide 11: Componente Hub

* O proposito do mesmo, e a interação com os componentes

Slide 12: Demonstração