**FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS**

**FATEC PROFESSOR JESSEN VIDAL**

**THIAGO LUIS SILVA FORTUNATO**

**IMPLANTAÇÃO DA PLATAFORMA**

**DE MAPEAMENTO DE COMPETÊNCIAS**

São José dos Campos

2017

**THIAGO LUIS SILVA FORTUNATO**

**IMPLANTAÇÃO DA PLATAFORMA**

**DE MAPEAMENTO DE COMPETÊNCIAS**

Trabalho de Graduação apresentado à Faculdade de Tecnologia São José dos Campos, como parte dos requisitos necessários para a obtenção da graduação de Tecnólogo em Banco de Dados.

**Orientador: Me. Eduardo Sakaue**

São José dos Campos

2017

**Dados Internacionais de Catalogação-na-Publicação (CIP)**

**Divisão de Informação e Documentação**

FORTUNATO, Thiago Luis Silva Fortunato

Implantação do Plataforma para Mapeamento de Competências.

São José dos Campos, 2017.

34fs.

Trabalho de Graduação – Curso de Tecnologia em Banco de Dados,

FATEC de São José dos Campos: Professor Jessen Vidal, 2017.

Orientador: Me. Eduardo Sakaue.

1. Banco de dados, Sistemas de informação. I. Faculdade de Tecnologia. FATEC de São José dos Campos: Professor Jessen Vidal. Divisão de Informação e Documentação. II. Plataforma para Mapeamento de Competências

**REFERÊNCIA BIBLIOGRÁFICA –**

FORTUNATO, Thiago Luis Silva. **Implantação da Plataforma para Mapeamento de Competências.**

2017. 48f. Trabalho de Graduação - FATEC de São José dos Campos: Professor Jessen Vidal.

**CESSÃO DE DIREITOS –**

NOME DO AUTOR: Thiago Luis Silva Fortunato

TÍTULO DO TRABALHO: Implantação Da Software Avaliador de Competências.

TIPO DO TRABALHO/ANO: Trabalho de Graduação / 2017.

É concedida à FATEC de São José dos Campos: Professor Jessen Vidal permissão para reproduzir cópias deste Trabalho e para emprestar ou vender cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte deste Trabalho pode ser reproduzida sem a autorização do autor.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Thiago Luis Silva Fortunato

**Thiago Luis Silva Fortunato**

**IMPLANTAÇÃO DA PLATAFORMA**

**DE MAPEAMENTO DE COMPETÊNCIAS**

Trabalho de Graduação apresentado à Faculdade de Tecnologia São José dos Campos, como parte dos requisitos necessários para a obtenção da graduação de Tecnólogo em Banco de Dados.

**Composição da Banca**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Fabiano Sabha Walczak, Mestre, FATEC Professor Jessen Vidal**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Jean Carlos Lourenço Costa, Pós-Graduado, FATEC Professor Jessen Vidal**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Eduardo Sakaue, Mestre, FATEC Professor Jessen Vidal**

**\_\_\_\_\_/\_\_\_\_\_/\_\_\_\_\_**

**DATA DA APROVAÇÃO**

**DEDICATÓRIA**

Dedico este trabalho à Deus e minha família, em especial a meus pais Paulo Fortunato e Rosângela Fortunato pelo apoio e compreensão demonstrados nesses anos de faculdade.

**AGRADECIMENTOS**

Agradeço em primeiro lugar ao Autor da Existência, Aquele que permite que todas as coisas se concretizem, nosso Deus.

Este trabalho de conclusão de curso não seria possível sem a colaboração de várias pessoas. Em especial, gostaria de agradecer:

Ao orientador Prof. Me. Eduardo Sakaue, por todo apoio e incentivo durante a elaboração deste trabalho e a todo período da faculdade.

A todos professores da FATEC, que são responsáveis pela minha formação técnica.

A todos meus colegas e amigos que formei durante os anos de estudo, principalmente Eduardo Di Nizo, Marcelo Inácio, Daniel Barbosa e William Penna que fizeram maior parte desta jornada.

**RESUMO**

O projeto Permanência e Desenvolvimento de Talentos Profissionais foi idealizado e mantido pelo Centro Paula Souza, onde visa reduzir a evasão dos alunos nas ETECs e FATECs. O projeto é formado por etapas, que vão desde a matrícula até a contratação dos alunos pelo mercado. Nessas etapas informatizar o processo de mapeamento de perfil do aluno traz grandes benefícios. É de interesse da gestão do Centro Paula Souza que todos alunos ingressantes tenham acesso a plataforma, desta forma, é fundamental que a plataforma esteja disponibilizada de forma centralizada, onde todas instituições (FATECs e ETECs) possam acessa-las. Implantar uma arquitetura que garante a disponibilidade com fácil manutenção é fundamental para O Centro Paula Souza possui uma conta na Plataforma Azure da Microsoft, porém existe uma dificuldade no gerenciamento e suporte das aplicações já disponíveis, por falta de pessoal. Ao centralizar a implantação, garante-se o alcance a todos alunos além de facilitar o gerenciamento dos serviços necessários para mantê-la sempre disponível.

**Palavras Chave:** *Arquitetura, Disponibilidade, Integração Contínua, DevOps.*

**ABSTRACT**

**LISTA DE FIGURAS**

[Figura 1- Arquitetura da Aplicação Mapskills. 14](file:///C:\Users\Thiago%20Fortunato\Documents\TG-Mapskills\Docs\Planejamento%20e%20Implantação%20do%20MapsSkills.docx#_Toc500357874)

[Figura 2 - Arquitetura Mapskills com base nos Requisitos 15](#_Toc500357875)

[Figura 3 - Processos realizados na Integração Contínua 17](file:///C:\Users\Thiago%20Fortunato\Documents\TG-Mapskills\Docs\Planejamento%20e%20Implantação%20do%20MapsSkills.docx#_Toc500357876)

[Figura 4 - Comparação entre a Arquitetura de Virtualização e o Docker 20](file:///C:\Users\Thiago%20Fortunato\Documents\TG-Mapskills\Docs\Planejamento%20e%20Implantação%20do%20MapsSkills.docx#_Toc500357877)

[Figura 5 - Solução proposta com base na Arquitetura Docker 22](#_Toc500357878)

[Figura 6 - Home Page HAProxy 25](file:///C:\Users\Thiago%20Fortunato\Documents\TG-Mapskills\Docs\Planejamento%20e%20Implantação%20do%20MapsSkills.docx#_Toc500357879)

[Figura 7 - Comunicação entre os Containers. 26](file:///C:\Users\Thiago%20Fortunato\Documents\TG-Mapskills\Docs\Planejamento%20e%20Implantação%20do%20MapsSkills.docx#_Toc500357880)

[Figura 8 - Home Page Jenkins 28](file:///C:\Users\Thiago%20Fortunato\Documents\TG-Mapskills\Docs\Planejamento%20e%20Implantação%20do%20MapsSkills.docx#_Toc500357881)

**LISTA DE TABELAS**

[Tabela 1 - Cronograma Fase 1 13](#_Toc500357923)

[Tabela 2 - Cronograma Fase 2 13](#_Toc500357924)

[Tabela 3 - Tecnologias Utilizadas 18](#_Toc500357925)

**SUMÁRIO**

[1. **INTRODUÇÃO** 12](#_Toc501898025)

[1.1 Objetivo 12](#_Toc501898026)

[1.2 Cronograma 13](#_Toc501898027)

[2. **LEVANTAMENTO DE REQUISITOS** 14](#_Toc501898028)

[2.1 Metodologia de levantamento de requisitos 14](#_Toc501898029)

[2.2 Requisitos do Projeto 15](#_Toc501898030)

[2.2.1 PaaS – Plataforma como Serviço 15](#_Toc501898031)

[2.2.2 Servidor Web Front-End 16](#_Toc501898032)

[2.2.3 Servidor Web Back-End 16](#_Toc501898033)

[2.2.4 Servidor Banco de Dados 16](#_Toc501898034)

[2.2.5 Balanceador de Carga 16](#_Toc501898035)

[2.2.6 Integração Contínua 17](#_Toc501898036)

[2.2.7 Interface de Monitoramento 17](#_Toc501898037)

[2.2.8 Quantidade e Escalabilidade 17](#_Toc501898038)

[2.3 Tecnologias Utilizadas 18](#_Toc501898039)

[2.3.1 DevOps 19](#_Toc501898040)

[2.3.2 Docker 19](#_Toc501898041)

[2.3.3 Tomcat 20](#_Toc501898042)

[2.3.4 Haproxy 20](#_Toc501898043)

[2.3.5 Jenkins 21](#_Toc501898044)

[2.3.6 Mysql 21](#_Toc501898045)

[3. **DESENVOLVIMENTO** 22](#_Toc501898046)

[3.1 Mapskills-Cadivisor 23](#_Toc501898047)

[3.2 Tomcat-Back 23](#_Toc501898048)

[3.3 Tomcat-Front 23](#_Toc501898049)

[3.4 Mapskills-Mysql 24](#_Toc501898050)

[3.5 Mapskills-Haproxy 24](#_Toc501898051)

[3.6 Docker Compose 25](#_Toc501898052)

[3.6.1 Links 26](#_Toc501898053)

[3.6.2 Volumes 26](#_Toc501898054)

[3.7 Mapskills-Jenkins 27](#_Toc501898055)

[3.7.1 Build-Mapksills-App 28](#_Toc501898056)

[3.7.2 Build Mapskills-Web 28](#_Toc501898057)

[3.7.3 Copy Artifact Mapskills App 29](#_Toc501898058)

[3.7.4 Copy Artifact Mapskills Front 29](#_Toc501898059)

[3.7.5 Deploy Mapskills 29](#_Toc501898060)

[4. **RESULTADOS** 30](#_Toc501898061)

[4.1 Experimento 1 30](#_Toc501898062)

[4.2 Experimento 2 30](#_Toc501898063)

[5. **TRABALHOS FUTUROS** 31](#_Toc501898064)

[**REFERÊNCIAS** 32](#_Toc501898065)

[**ANEXOS** 33](#_Toc501898066)

# **INTRODUÇÃO**

Com o intuito aumentar a permanência dos alunos nas instituições de ensino para mitigar a evasão dos estudantes, de forma a garantir a conclusão no prazo previsto, o Projeto Permanência e Desenvolvimento de Talentos Profissionais do Centro Paula Souza deseja reduzir em 50% o índice de evasão nos cursos das FATECs e ETECs selecionadas, visando desenvolver metodologia, ferramentas, processos, parâmetros, indicadores e recursos (PROJETO PERMANÊNCIA, 2017).

Como parte do Projeto de Desenvolvimento o Escritório de Carreiras da Fatec de São José dos Campos foi moldado para ser um mecanismo direcionado a ajudar na preparação dos alunos para o mercado de trabalho.

Uma das etapas do projeto Escritório de Carreiras é o mapeamento de competências. Nesta etapa foi desenvolvida uma plataforma que tem o objetivo de hospedar jogos do tipo perguntas e respostas, que possibilite gerar relatórios dos alunos que finalizaram o jogo a partir das respostas fornecidas (Silva, 2017).

Para suportar esta plataforma é necessário prover toda uma infraestrutura adequada ao formato que a plataforma foi desenvolvida. O Centro Paula Souza detém uma conta na Plataforma *Cloud* Azure da Microsoft, que tem a finalidade de centralizar o gerenciamento e controle de todos serviços necessários para que a plataforma esteja disponível, mas não existe uma quantidade de pessoas disponíveis para dar suporte à todas aplicações já implantadas no Azure. Neste contexto, é necessário prover uma arquitetura em que todos serviços sejam gerenciados de forma contínua, prática, de fácil manutenção e centralizada.

## Objetivo

Prover uma arquitetura para dar suporte a plataforma, fornecendo os recursos necessários ao acesso em larga escala da aplicação, garantindo a agilidade, qualidade e estabilidade com escalabilidade, além de integrar de forma contínua.

Disponibilizar uma arquitetura para alta demanda de requisições em ambiente com pouco recurso computacional e pessoal.

## Cronograma

Para este projeto foi definido um planejamento dividido em duas fases. A primeira fase elaborada no período do segundo semestre de 2016. Foi buscado ter todo conhecimento dos requisitos a serem atendidos e iniciado desenvolvimento.

Tabela 1 - Cronograma Fase 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Etapa / 2016 | Setembro | Outubro | Novembro | Dezembro |
| Levantamento de requisitos |  |  |  |  |
| Desenvolvimento |  |  |  |  |

A segunda fase contemplada no primeiro semestre de 2017. Foi continuado o desenvolvimento da aplicação com previsão de uma primeira versão para a primeira semana de maio. Com o objetivo de testar a primeira versão da aplicação, entre os meses de maio e julho de 2017, se teve o lançamento da publicação parcial e em simultâneo a realização de testes com alunos e realização de correções de problemas encontrados na utilização. Por fim previsão da publicação final no mês de julho, mas devido a problemas de hospedagem só pode ser possível a publicação no mês seguinte de agosto de 2017.

Tabela 2 - Cronograma Fase 2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Etapa / 2017 | Janeiro | Fevereiro | Março | Abril | Maio | Junho | Julho | Agosto |
| Desenvolvimento |  |  |  |  |  |  |  |  |
| Testes com alunos |  |  |  |  |  |  |  |  |
| Publicação parcial |  |  |  |  |  |  |  |  |
| Correções |  |  |  |  |  |  |  |  |
| Publicação da versão final |  |  |  |  |  |  |  |  |

# **LEVANTAMENTO DE REQUISITOS**

Neste Capítulo serão apresentadas as metodologias de levantamento de requisitos não funcionais e casos de uso.

## Metodologia de levantamento de requisitos

Para este projeto foi adotado a clássica metodologia de entrevista, pois se trata de uma metodologia simples e que produz bons resultados na fase inicial do projeto (PRESSMAN, 2016).

A partir desta técnica foi possível refinar os requisitos e reduzir o tempo de desenvolvimento da aplicação, trabalhando em cima das necessidades que à aplicação deve atender.

A organização da aplicação é distribuída conforma a Figura 1. Com base na nesta Figura, pode-se obter uma visão geral de como e com quem os serviços se relacionam e além de exibir demonstrar suas dependências.

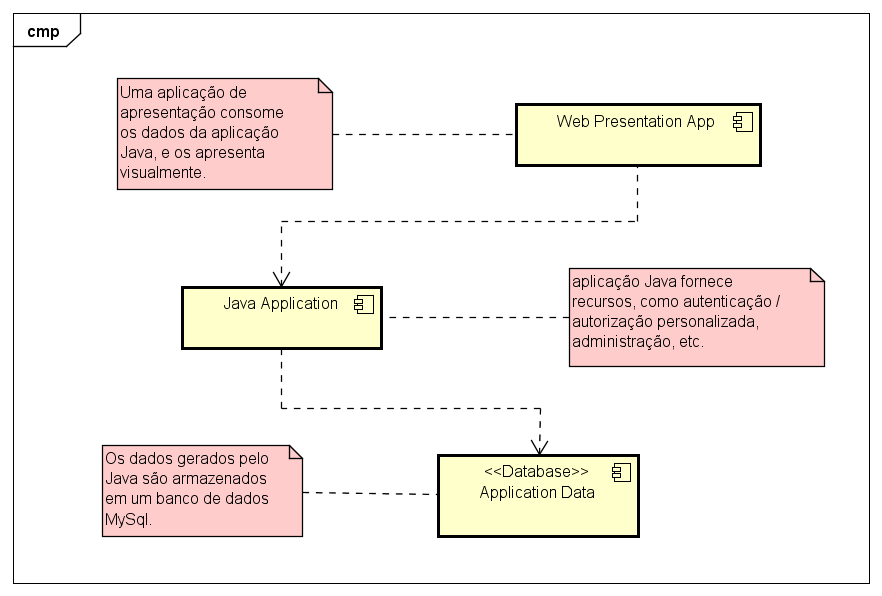


Figura 1- Arquitetura da Aplicação Mapskills.

## Requisitos do Projeto

Para garantir a disponibilidade e qualidade da plataforma é necessário que contenha serviços específicos para que a plataforma consiga realizar seu objetivo proposto.

A arquitetura definida como requisito é descrita na Figura 2.

Figura 2 - Arquitetura Mapskills com base nos Requisitos



### **PaaS – Plataforma como Serviço**

A Microsoft Azure é uma plataforma de execução de serviços e aplicações baseada em um conceito de *Clouding Computing*. Este termo está associado a utilização da rede mundial de computadores, a internet, para alocação de um ambiente computacional.

Um dos conceitos vinculados à *Cloud* a Plataforma como um Serviç*o* (PaaS), é a disponibilização de plataformas de desenvolvimento que facilitam a implantação de aplicações assim como o gerenciamento do hardware e das camadas de software. Fornece todas facilidades necessárias para suportar todo o ciclo de vida das aplicações Web, sem a necessidade de download ou instalação de aplicativos para desenvolvedores ou usuários finais. (Nogueira, 2010).

Ter um computador disponível na Plataforma Cloud Azure com Sistema Operacional Ubuntu 14.10 instalado, fornecendo todos recursos necessários para que a plataforma *Mapskills* esteja em produção, sendo escalável, com acesso remoto.

### **Servidor Web Front-End**

Servidor Web com a finalidade de disponibilizar a aplicação que realizará será a interface de comunicação com o usuário. A aplicação contida neste servidor trabalha no modelo Cliente-Servidor, onde o HTTP é o protocolo de comunicação entre as aplicações de *front-end* (interface gráfica) e *back-end* (regra de negócio) que seja responsável por disponibilizar a aplicação *front-end* compilada no formato Recurso de Aplicação Web (war) com Java 8.

### **Servidor Web Back-End**

Servidor responsável por disponibilizar a aplicação *back-end* da plataforma. A aplicação contida neste servidor trabalha no modelo Cliente-Servidor, realizando conexão com o *front-end* por meio do protocolo HTTP, esta aplicação também foi desenvolvida com Java 8 e compilada no formato Recurso de Aplicação Web (war).

### **Servidor** **Banco de Dados**

É necessário armazenar todas informações pertinentes ao jogo, como características do jogo a ser aplicado, imagens, perguntas e alternativas, bem como informações das instituições, usuários, afim de gerar relatórios para que os responsáveis possam visualizar os resultados do jogo.

Mysql é um Gerenciador de Banco de Dados gratuito, que utiliza da linguagem SQL (Linguagem de Consulta Estruturada), como interface para manipulação e gerenciamento dos dados.

### **Balanceador de Carga**

É fundamental controlar a quantidade de acesso ao Servidor de *back-end*, para que isso ocorra é necessário que este serviço seja escalável. Este serviço será responsável por direcionar as requisições para o Servido *back-end* com menos carga.

### **Integração Contínua**

É necessário garantir que um novo código esteja apto a ser disponibilizado frequentemente. Controlando as construções, versionamento, validações e testes. Para que o processo de frequentes alterações parciais enteja disponível de maneira automática e com garantia de que a plataforma funcione como esperado. A Figura 3 demonstra os processos realizados na Integração Contínua.

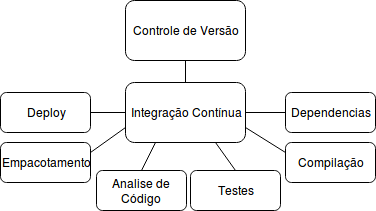


Figura 3 - Processos realizados na Integração Contínua

### **Interface de Monitoramento**

Serviço de monitoramento em tempo real para que informe ao Técnico responsável pela disponibilização da aplicação o estado da Máquina Virtual, demonstrando graficamente quanto é consumido de recurso de memória, processador, entrada e saída dados, bem como métricas referente a consumo de internet do host, além do estado dos containers.

### **Quantidade e Escalabilidade**

É primordial que a plataforma seja escalável, garantindo que o software estará sempre disponível sendo disponibilizado aos usuários com qualidade.

Este software será utilizado por 200 ETECs, 70 FATECs, cada uma oferecendo seus cursos, sendo estes, com em média 40 alunos matriculados.

Pode-se ter como exemplo a FATEC de São José dos Campos, que contém 7 cursos, cada um com 40 alunos ingressantes. Na primeira quinzena do semestre haverá um acesso de 360 alunos simultâneos, onde os mesmos devem acessar a qualquer momento e de qualquer lugar.

## Tecnologias Utilizadas

Neste capítulo está descrito cada tecnologia utilizada na arquitetura e o porquê ela foi escolhida.

Tabela 3 - Tecnologias Utilizadas

|  |  |  |
| --- | --- | --- |
| Nome | Versão | Funcionalidade |
| Java | 8 | Desenvolvimento *back-end*, regras de negócio e domínio. |
| Tomcat | 8.5 | Servidor Web para hospedagem da aplicação *back-end* e *front-end.* |
| MySQL | 5.7 | Persistência dos dados da aplicação. |
| Jenkins | 2.60.2 | Plataforma para Gerenciamento da Integração Contínua. |
| Cadivisor | 0.25.0 | Aplicação para Monitoramento. |
| Docker | 17.06-0 ce | Plataforma para Gerenciamento dos Containers. |
| Docker Compose | 1.14.0 | Plataforma para Orquestração dos Containers. |
| GitHub |  | Repositório para código Java. |

### **DevOps**

O termo DevOps surgiu num evento organizado por [[1]](#footnote-1)Andrew Shaffer e o engenheiro de sistemas John Allspaw, para discutir especificamente os desafios da integração das áreas de desenvolvimento e operações existentes nas empresas.

Modelo utilizado quando se trata de metodologia ágeis, afim de realizar entregas rápidas com qualidade. Tem finalidade de integrar os setores de desenvolvimento e operações, diminuindo a dificuldade que encontravam quando se lançava uma nova funcionalidade do software, pois os setores operacionais criam um ambiente propício para execução de determinadas ferramentas pré-definidas no escopo do projeto e caso algo seja alterado, pode-se perder pontos no quesito qualidade e disponibilidade da aplicação. (DevOps, 2017)

### **Docker**

Plataforma instalada na Máquina Virtual responsável por administrar e controlar todos serviços necessários para que a aplicação esteja em operação.

O tutorial seguido para instalação do Docker pode ser encontrado na url: *https://docs.docker.com/engine/installation/*.

Foi utilizado esta aplicação pois diminuirá drasticamente o tempo necessário para disponibilizar a aplicação, pois não é necessário configurar o ambiente toda vez que for instalado.

A escolha do Docker também foi baseada na redução de custo do projeto, pois o Docker é uma plataforma Open Source e foi disponibilizada apenas uma máquina virtual, hospeda no Azure, e todos recursos que serão necessários são instalados dentro dele, desta forma, não é necessário ter computadores robustos ou vários serviços específicos para execução de tarefas diferentes.

A Figura 4 demonstra a comparação entre a arquitetura de virtualização e a arquitetura Docker. Como pode-se observar, o Docker realiza a virtualização de cada container a nível de sistema operacional, excluindo a necessidade do Hipervision. Este método de virtualização faz com que o Kernel do sistema operacional permita que múltiplos processos sejam executados isoladamente no mesmo host. (Docker, 2107).

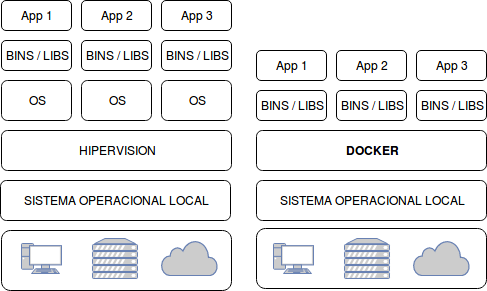


Figura 4 - Comparação entre a Arquitetura de Virtualização e o Docker

### **Tomcat**

Desenvolvido pela Apache, e distribuído como software livre. É basicamente um container de servidores. Um container em Java é descrito como um objeto pode-se dizer que é uma aplicação que contém mais de um servidor Web. Foi escrito em Java, linguagem esta que também foi utilizada no desenvolvimento do Software *Mapskills*.

### **Haproxy**

O HAProxy é um serviço Linux que garante um balanceamento e alta disponibilidade num conjunto de servidores, como também serviço de proxy, ao não expor diretamente estes mesmos servidores na rede externa.

### **Jenkins**

Ferramenta de Integração Contínua, automatizada. A atividade de construir um projeto é composta por várias etapas, incluindo a compilação do código fonte, execução de testes, empacotamento, além de métricas referente a qualidade do código. O Jenkins trabalha para monitorar todas alterações realizadas no código e integrar com as atividades de construção, afim de dirimir as falhas no desenvolvimento.

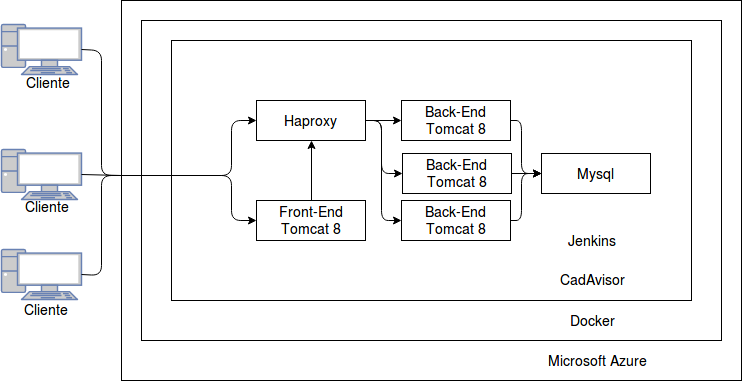
Para que todas etapas da Integração Contínua sejam concluídas é necessário a criação e configuração de *Jobs* onde cada um, tem a finalidade de executar uma tarefa específica.

# **DESENVOLVIMENTO**

Como pode ser observado na base na arquitetura descrita na Figura 2, são necessários diversos serviços para que a plataforma esteja em produção. Como solução para este problema foi utilizado o Docker para orquestrar e prover que todos serviços estejam sempre em funcionamento.

Todas imagens Docker utilizadas podem ser encontradas no repositório oficial do Docker Hub: *https://hub.docker.com/u/thiagolsfortunato/[nome\_da\_imagem]:[versão]*.

A arquitetura foi divida em serviços independentes, onde cada um exerce uma funcionalidade e tem uma relação com outros serviços como pode ser observado na Figura 5.

Figura 5 - Solução proposta com base na Arquitetura Docker

## Mapskills-Cadivisor

Container com a finalidade de solucionar o requisito descrito no item 2.2.7, auxilia no gerenciamento dos recursos consumidos pelo host. Como são necessários vários serviços que formam a arquitetura da aplicação *Mapskills*, fica difícil monitora-los individualmente, para isto o Cadvisor tem a finalidade de monitorar todos processos no host e containers, sabendo em tempo real o quanto esta sendo consumido de recurso.

A aplicação Cadivisor pode ser acessada pela url: [*http://ip\_do\_host:8888*](http://ip_do_host:8888/).

## Mapskills-Front

Container responsável por conter o projeto de Interface Front-Web, mapskills-web.war. Neste container também está instalado o Java na versão 1.8 e o Tomcat na versão 8.5, para criação deste container foi utilizada uma Imagem *Alpine*, pois reduziu significativamente o tamanho da imagem, rodando apenas um processo Java, atendendo desta forma os itens descritos no 2.2.3.

Para se encaminhar uma requisição ao front-end é necessário acessar aplicação através da url: *http://ip\_do\_host:80/mapskills-web*

## Mapskills-Back

Container que atende os requisitos descritos no item 2.2.4, responsável por conter o projeto Java de *back-end*, *mapskills.war*. Neste container está instalado o Java na versão 1.8 e o Tomcat na versão 8.5, para criação deste container foi utilizada uma Imagem Alpine, pois reduziu significativamente o tamanho da imagem, rodando apenas um processo Java.

Para se encaminhar uma requisição ao *back-end* é necessário acessar aplicação através da url: *http://ip\_do\_host:8080/mapskills.*

## Mapskills-Mysql

Container responsável por armazenar todas informações referente a plataforma.

Foi configurado para permitir acesso remoto ao banco de dados, pois a aplicação pode não estar no mesmo servidor que a plataforma. Para isto, foi alterado o valor do parâmetro *bind-address* para *0.0.0.0* no arquivo *my.cnf* permitindo acesso remoto.

Um usuário com permissões para consultar, inserir, alterar ou deletar informações na Base de Dados Mapskills foi criado. Este é o único usuário que tem acesso remoto as informações do Banco de Dados.

Somente o Container Tomcat-Back-End pode se comunicar com a Base de Dados, desta forma, foi isolado o acesso aos dados.

Este container tem a finalidade de atender os requisitos descritos no item 2.2.4.

## Mapskills-Haproxy

O Haproxy tem a finalidade de solucionar os requisitos descritos no item 2.2.5. Gerencia todas requisições HTTP destinadas ao Mapskills-app, realizando o balanceamento de carga entre os containers do *back-end*.

Para isto, o Haproxy utiliza do algoritmo Round Robin, este algoritmo tratar os servers como iguais, independentemente do número de conexões solicitadas, sempre redirecionando a próxima requisição ao server seguinte, desta forma, todos servers terão o mesmo número de conexões.

O Haproxy trabalha recebendo todas requisições através da porta 80 e redirecionando internamente a porta 8080 destinada ao Tomcat que contém o mapskills-war. O balanceamento de carga utilizado é de camada 4 (camada de transporte) da Tabela OSI, encaminhando o tráfego do usuário com base no alcance e na porta do IP, no caso definida como 80.

Além de controlar as requisições o Haproxy exibe também fornece um Dashboard para visualização dos servers, números de requisições com sucesso e falha, bem como saber em tempo real a quantidade de Kbps que foi trafegada pela rede, conforme figura abaixo.

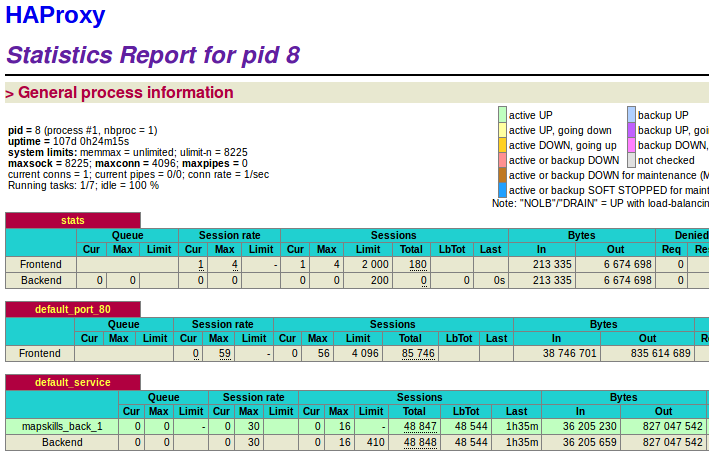
 Para acesso a esse Dashboard é necessário acessar a porta 1936.

Figura 6 - Home Page HAProxy

Figura - Home Page HAProxy

## Docker Compose

Este arquivo de configuração tem a finalidade de automatizar a implantação de todo ambiente de produção necessário para que o Mapskills funcione. O arquivo docker-compose.yml criara e iniciara todos serviços definidos.

Para separar o software Mapskills dos aplicativos que realizam o gerenciamento da aplicação, foi necessário criar dois arquivos. Em um arquivo docker-compose será responsável por gerenciar as aplicações Jenkins, Cavisor e o Banco de Dados Mysql.

Este arquivo localizado no diretório /opt/mapskills/manager, é inicializado primeiro, pois desta forma a segunda parte poderá cair, a qualquer momento que os dados pertinentes ao gerenciamento da aplicação, bem como a base de dados não serão perdidos.

Para que a segunda parte seja inicializada, é necessário que a primeira esteja com todos serviços ativos. Neste arquivo docker-compose, serão inicializados os containers Mapkills-app, Mapskills-web e Mapkills-Haproxy.

### **Links**

Além de inicializar os serviços, o arquivo docker-compose.yml é responsável por configurar como e com quem cada um dos containers irão se comunicar, bem como os volumes necessários para cada container.

Foi criado um link entre os containers Mapskills-app e o Mapskills-Mysql, pois desta forma o *back-end* da aplicação poderá persistir, consultar deletar ou alterar qualquer informação no banco de dados.

O link entre o container Mapskills-app e Mapskills-web para que os dados inseridos na interface web seja trafegada para o *back-end*, e assim os dados sejam manipulados.

Link entre o Haproxy e Mapskills-app para que todas requisições destinadas ao Mapskills-app sejam controladas pelo balanceador de carga.

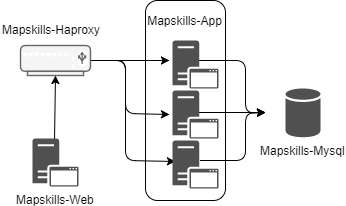
****

Figura 7 - Comunicação entre os Containers.

Figura - Comunicação entre os Containers.

### **Volumes**

Os Volumes Docker têm a finalidade de persistir os dados usados pelos Containers Docker. Cada container é responsável pelo seu volume e informação sensível ao serviço em que oferece.

O Mapskills-app e Mapskills-web tem os volumes mapeados para que o arquivo .war seja atualizado pelo Jenkins a qualquer momento, desta forma, é solucionado o problema de integração contínua, pois uma versão nova do sistema, atualizará automaticamente.

O volume Mapskills-Mysql tem a finalidade de armazenar todas informações salva na base de dados, podendo assim para a execução ou mesmo excluir o container Mapskills-Mysql que as informações não serão perdidas.

Os volumes utilizados no container Mapkills-Jenkins, tem funções importantes na solução para o requisito de Integração Contínua, pois eles formam uma comunicação entre os containers Mapkills-App e Mapskills-Web para que sejam atualizados a qualquer instante, além de compartilhar os arquivos de execução do Docker e Docker-Compose, para que possam ser executados comando de dentro do container.

No container Mapskills-Cadvisor são compartilhados com o host os volumes necessários para o monitoramento de dados referente ao Host e Containers Docker.

## Mapskills-Jenkins

Container Jenkins realiza o controle das implantações realizadas durante a implementação do software. Após configurado, tem o trabalho de realizar construções de forma instantânea, com testes sendo executados e falhas detectadas caso encontre-as.

Seu funcionamento baseia-se na criação de *Jobs* para execução de tarefas específicas, para que seja atendido todos requisitos propostos à Integração Contínua, foi necessário configurar quatro Trabalhos: Build-Mapksills-App, Build-Mapskills-Front, Copy Artifact Mapskills App, Copy Artifact Mapskills Front, Deploy Mapskills.

A Interface Web do Jenkins pode ser acessada através do <http://ip_do_host:8585>.



Figura 8 - Home Page Jenkins

Figura 6 - Home Page Jenkins

### **Build-Mapksills-App**

Este *Job* irá baixar o projeto Java do repositório GitHub e compilar o projeto no formato .war, para que seja disponibilizado posteriormente por outro *Job*.

O comando utilizado apaga a pasta Target, instala os pacotes nos respectivos repositórios e não rodar o script de criação do Banco de Dados, pois desta forma é garantido que os dados nunca serão apagados. Todos esses comandos são rodados pelo usuário Azure, configurado na aplicação:

mvn clean install -Dliquibase.should.run=false -Pazure

### **Build Mapskills-Web**

Este job irá baixar o projeto de Interface Web do repositório GitHub e compilar o projeto no formato .war, para que seja disponibilizado posteriormente.

mvn clean install

### **Copy Artifact Mapskills App**

Este job depende de que o Build-Mapskills-App (capítulo 3.7.2) tenha sido finalizado com sucesso, só assim, ele irá copiar o arquivo .war do diretório */var/jenkins\_home/workspace/Build-Mapskills-Back/target/mapskills.war* para o volume Docker */mapskills/back, pois desta forma o arquivo .war é compartilhado com o container* responsável por conter a aplicação de *back-end*.

sudo cp /var/jenkins\_home/workspace/Build-Mapskills-Back/target/mapskills.war /mapskills/back

### **Copy Artifact Mapskills Front**

Assim como o *Job* Copy Artifact Mapskills Back Este trabalho depende também de que o job Build-Mapskills-Web tenha sido finalizado com sucesso. Após isso ele copia o arquivo .war do diretório */var/jenkins\_home/workspace/Build-Mapskills-Front/target/mapskills-web.war* para o volume Docker */mapskills/front,* pois desta forma o arquivo .war é compartilhado com o containerresponsável por conter a aplicação de *front-end*.

### **Deploy Mapskills**

Como produto final, o trabalho Deploy Mapskills é responsável por disponibilizar os projetos de *front* e *back-end* em produção, ou seja, esteja disponível a última versão estável do projeto. Para que seja esteja em produção, é executado o arquivo “docker-compose.yml”.

sudo docker-compose -f /mapskills/docker-compose.yml up -d

# **RESULTADOS**

Neste Capítulo serão apresentados os resultados obtidos com a implantação da aplicação. Ao total entre os meses de maio e atualmente aplicação foi testada por mais de 1400 alunos das instituições da FATEC São José dos Campos, FATEC Tatuapé, FATEC Araçatuba, FATEC Garça, FATEC Mogi Mirim, FATEC Pindamonhangaba e FATEC Jales.

## Experimento 1

- Servidor: Rede Interna.

- Data Prevista: 13 de março de 2017

- Horário: das 18h às 21h

- Quantidade de Alunos: 120 alunos

O primeiro experimento foi utilizado a rede interna da Fatec para observar como a aplicação iria se comportar. Neste primeiro evento, muitos alunos não sabiam seu e-mail ou, e por diversas vezes foi necessário acessar o Banco de Dados, também foi observado o consumo de Memória e Processador do computador usado como servidor.

Durante o experimento pode-se observar um alto consumo de Memória pois eram realizadas diversas tarefas concorrentemente, além de operações de acesso ao banco de dados.

Para resolver este requisito, foi adicionado na arquitetura do projeto um balanceador de carga, que tem a responsabilidade de direcionar as requisições ao *back-end* com menor número de tarefas em execução.

## Experimento 2

- Servidor: Rede Interna e Cloud Azure.

- Data Prevista: 01 de agosto de 2017

- Horário: das 18h às 21h

- Quantidade de Alunos: 150 alunos

O segundo experimento foi realizado na Fatec com o acesso de aproximadamente 150 alunos, com o objetivo de monitorar pelos containers Mapskills-Cadvisor e Mapskills-Haproxy o acesso a plataforma além dos recursos utilizados pelo servidor.

Dado momento, foi constatado um problema na aplicação de *back-end* e foi necessária atualiza-la já em produção. Ao atualizar a versão aplicação de *back-end,* foi interrompido o Servidor de Banco de Dados. Por conta da indisponibilidade de acesso ao Banco de Dados a arquitetura do projeto foi fracionada em duas partes. Uma contendo a parte de gerenciamento da aplicação com os serviços: Mapskills-Jenkins, Mapskills-Cadivisor, Mapskills-Mysql, pois estes containers armazenam ou gerenciam dados importantes e críticos na aplicação. E a outra parte com os serviços Mapskills-Haproxy, Mapskills-Front, Mapskills-Back, com a finalidade automatizar os serviços de acesso a plataforma, facilitando também a integração, e controle na entrega de uma nova versão do software.

# **TRABALHOS FUTUROS**

Finalizado a implantação da plataforma e atendendo aos requisitos levantados, foram encontradas melhorias na arquitetura para que torne ainda mais eficiente e robusta.

- Alterar o Servidor de Gerenciador de Repositório para privado, como por exemplo o Git Lab, que permite a realização do processo de Integração Contínua.

- Criar aplicação Android para que possa ser consultado o resultado do aluno.

- Excluir imagens não utilizadas das cenas quando são editadas, não gerando acumulo de arquivos não utilizados no servidor.

- Deixar flexível o arquivo *application.properties* da aplicação *back-end* para configuração do local onde ficarão as imagens das cenas dos jogos.

- Separar do Servidor de Banco de Dados em outro servidor para garantir a integridade e disponibilidade.

- Implementar uma arquitetura de clusters utilizando do Docker Swarm, onde vários computadores trabalham juntos afim de garantir desempenho e disponibilidade da plataforma.

# **REFERÊNCIAS**

Aulbach, Stefan; JACOBS, Dean; KEMPER, Alfons; et al. (2009) “**A Comparison of Flexible Schemas for Software as a Service**” 35th SIGMOD - International Conference on Management of Data, 2009.

Docker Documentation, **Docker.** Disponível em <https://docs.docker.com>. Acesso em: 28 de outubro de 2017.

DevOps Conceitos, **DevOps**. Disponível em <https://aws.amazon.com/pt/devops/what-is-devops>. Acesso em: 28 de outubro de 2017

Jenkins Documentation, **Jenkin**, Disponível em <https://jenkins.io/doc>. Acesso em: 28 de outubro de 2017.

Nogueira, Matheus Cadori1; PEZZI, Daniel da Cunha (2010) “**A Computação Agora é nas Nuvens**” Universidade de Cruz Alta (UNICRUZ) – Cruz Alta, RS – Brasil.

PROJETO PERMANÊNCIA. **Projeto Permanência e Desenvolvimento de Talentos Profissionais**. Disponível em: <http://www.projpermanencia.com.br>. Acesso em: 24 de junho de 2017.

PRESSMAN, Roger; MAXIM**, Bruce. Engenharia de Software-8ª Edição**. McGraw Hill Brasil, 2016.

SILVA, Marcelo Inácio da. **Plataforma para Jogos de Mapeamento de Competências, 2017.**

# **ANEXOS**

[1] Configuração Jenkins – Configuração dos Jobs no Jenkins.

1. Andrew Shaffer e John Allspaw – Idealizadores da 1 º Evento a tratar do assunto DevOps. [↑](#footnote-ref-1)