FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS
FATEC PROFESSOR JESSEN VIDAL

THIAGO LUIS SILVA FORTUNATO

PLANEJAMENTO E IMPLANTAÇÃO DA PLATAFORMA DE MAPEAMENTO PARA COMPETÊNCIAS

São José dos Campos
2017

THIAGO LUIS SILVA FORTUNATO

PLANEJAMENTO E IMPLANTAÇÃO DA PLATAFORMA DE MAPEAMENTO PARA 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
Planejamento e Implantação do Plataforma para Mapeamento de Competências.
São José dos Campos, 2017.
33fs.

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. **Planejamento e 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: Planejamento e Implantação do Software Avaliador de Competências.
TIPO DO TRABALHO/ANO: Trabalho de Graduação / 2017.

_____
Thiago Luis Silva Fortunato

Thiago Luis Silva Fortunato

PLANEJAMENTO E IMPLANTAÇÃO DA PLATAFORMA DE MAPEAMENTO PARA 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.

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

The Permanence and Professional Talent Development project was conceived and maintained by the Paula Souza Center, with the target to reduce student evasion in ETECs and FATECs. The project consists of steps, ranging from enrollment to hiring students through the market. In these steps computerizing the process of mapping the profile of the student brings great benefits. It is in the interest of the management of the Paula Souza Center that all incoming students have access to the platform, so it is fundamental that the platform is centrally available, where all institutions (FATECs and ETECs) can access them.

The Paula Souza Center has an account in the Azure Platform of Microsoft, but there is a difficulty in the management and support of the applications already available, due to lack of personnel. By centralizing the implementation, it guarantees the reach to all students besides facilitating the management of the necessary services to keep it always available.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. 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 (Inácio, 2017).

Para suportar esta plataforma é necessário prover toda uma infraestrutura adequada ao formato que a plataforma foi desenvolvida. O Centro Paula Souza detém de 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.

1.1 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.

1.2 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 março. Com o objetivo de testar a primeira versão da aplicação, entre os meses de março e abril 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 maio, mas devido a problemas de hospedagem só pode ser possível a publicação no mês de julho de 2017.

Tabela 2 - Cronograma Fase 2

Etapa / 2017	Janeiro	Fevereiro	Março	Abril	Maio
Desenvolvimento					
Testes com alunos					
Publicação parcial					
Correções					
Publicação da versão final					

2. LEVANTAMENTO DE REQUISITOS

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

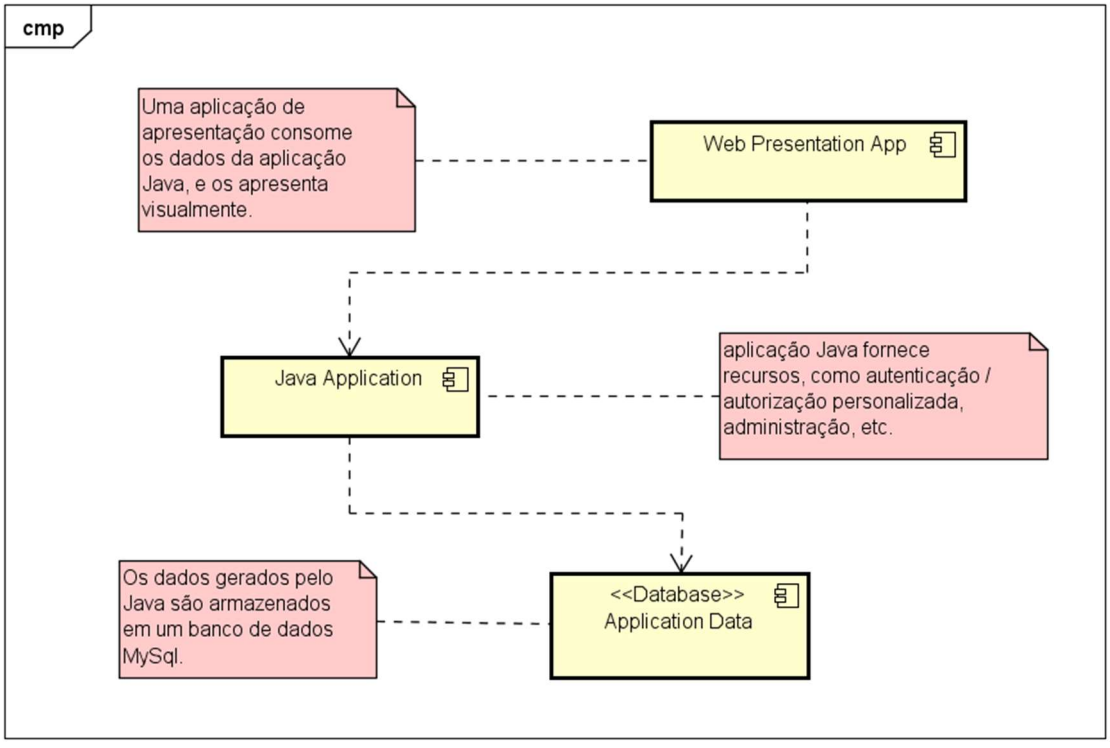
2.1 Metodologia de levantamento de requisitos

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.

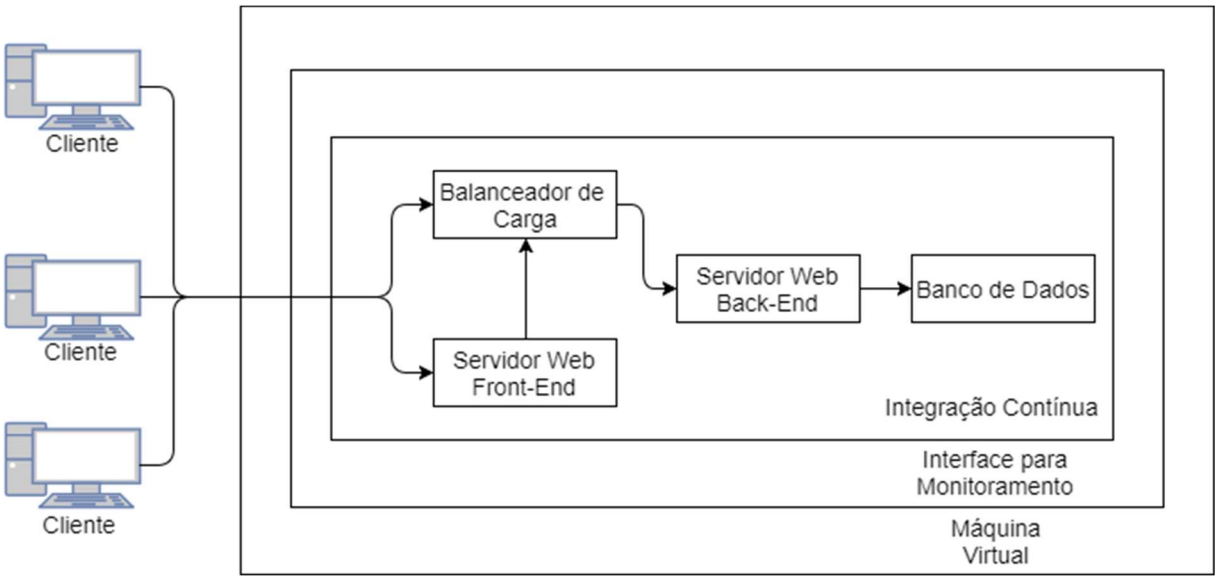


2.2 Requisitos do Projeto

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 está descrita na Figura 2.

Figura 2 - Arquitetura Mapskills com base nos Requisitos



2.2.1 Máquina Virtual

Ter um computador disponível na Plataforma Cloud Azure com Sistema Operacional instalado, fornecendo todos recursos necessários para que a plataforma esteja em produção, escalável, com acesso remoto e customizavel. É fundamental que tenha acesso única e exclusivamente por usuário e senha.

2.2.2 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.

2.2.3 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).

2.2.4 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.

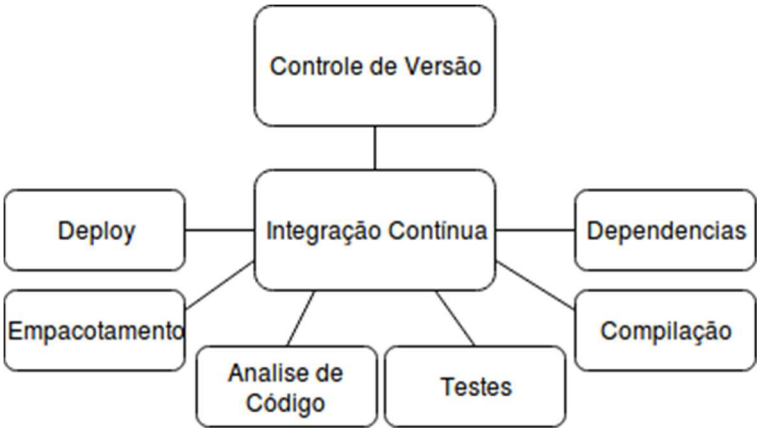
2.2.5 Balanceador de Carga

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.

2.2.6 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



2.2.7 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.

2.2.8 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.

2.3 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.

2.3.1 DevOps

O termo DevOps surgiu num evento organizado por [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.

2.3.2 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 neste link.

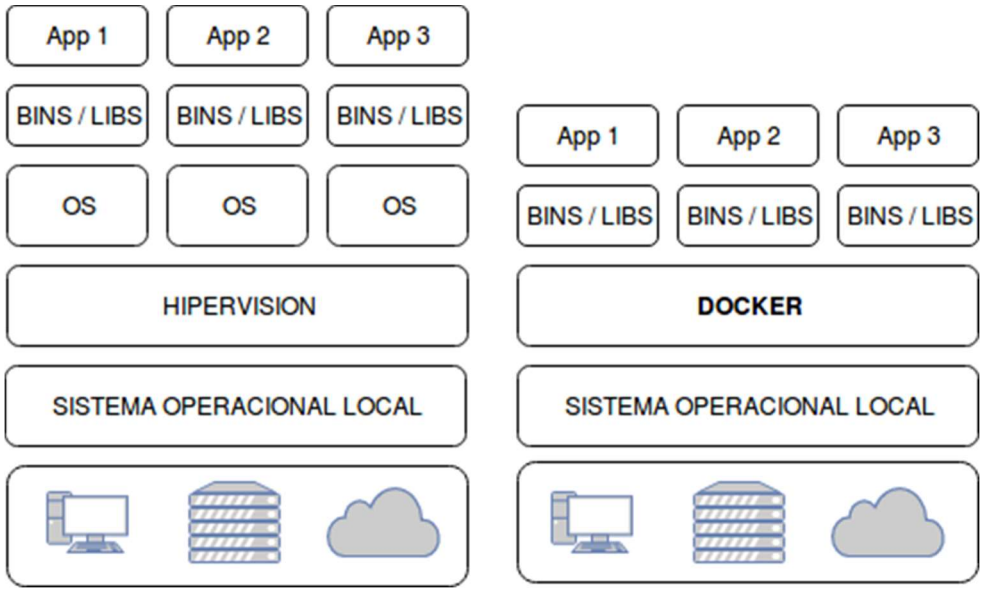
Com o Docker realiza o empacotamento do ambiente inteiro dentro de um contêiner, compartilhando com outras aplicações o que irão utilizarA. Foi utilizado esta aplicação pois diminuirá drasticamente o tempo de necessário para disponibilizar a aplicação, pois não será necessário configurar o ambiente toda vez que for instalado.

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

Cada container criado no Docker é isolado a virtualização a nível do sistema operacional, um método de virtualização onde o Kernel do sistema operacional permite que múltiplos processos sejam executados isoladamente no mesmo host.

[1]Andrew Shaffer e John Allspaw – Idealizadores da 1 º Evento a tratar do assunto DevOps.

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



2.3.3 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*.

2.3.4 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.

2.3.5 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.

2.3.6 Mysql

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.

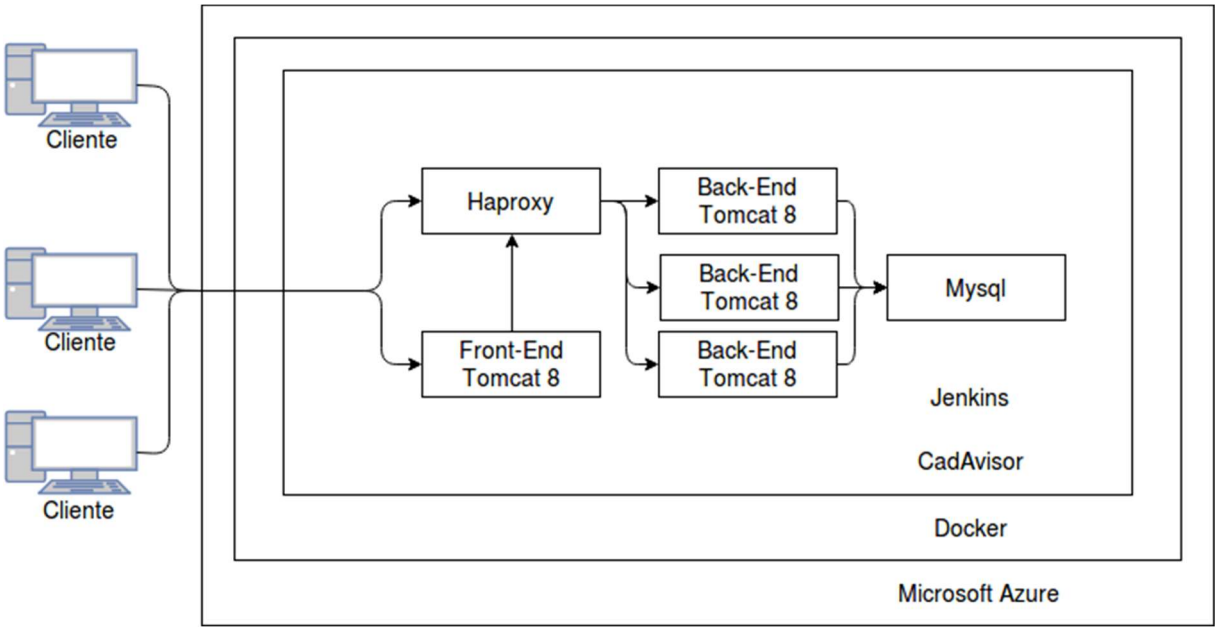
3. DESENVOLVIMENTO

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 (link): 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



3.1 Mapskills-Cadivisor

Container que 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.

3.2 Tomcat-Back

Container 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*

3.3 Tomcat-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.

Para se encaminhar uma requisição ao front-end é necessário acessar aplicação através da url: http://ip_do_host:80/mapskills-web

3.4 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.

3.5 Mapskills-Haproxy

O Haproxy tem a finalidade gerenciar todas requisições HTTP destinadas ao Mapskills-app, realizando o balanceamento de carga entre os containers do *back-end*.

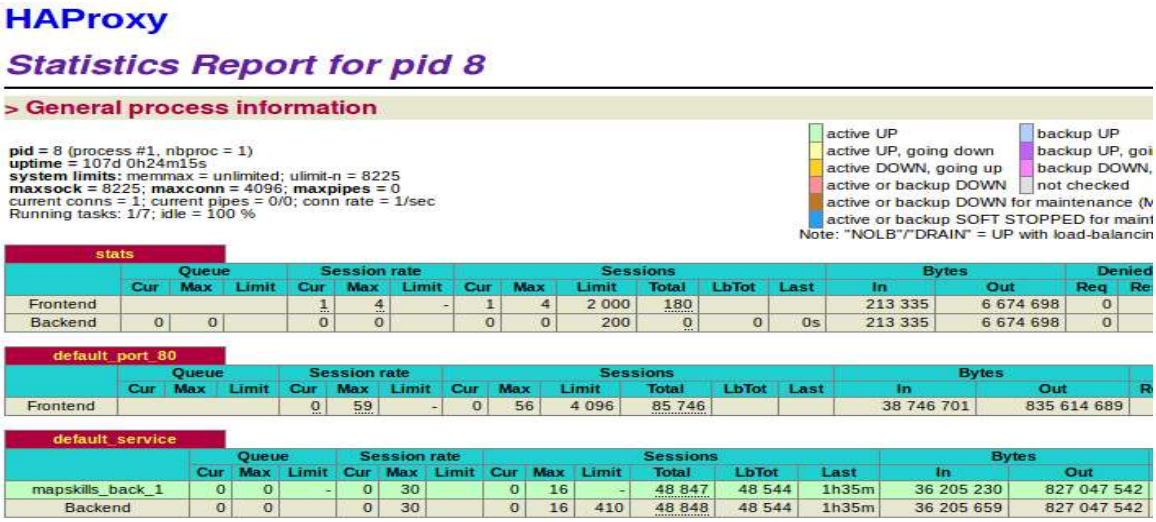
Para isto, o Haproxy utiliza do algoritmo Round Robin, este algoritmo tratar os servers como iguais, independente do número de conexões solicitadas, sempre redirecionando a próxima requisição ao server seguinte, desta forma, todos servers terão o mesmo número de conexões.

O Haproxy trabalha como um Proxy Reverso, 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 e 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



3.6 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.

3.6.1 Links

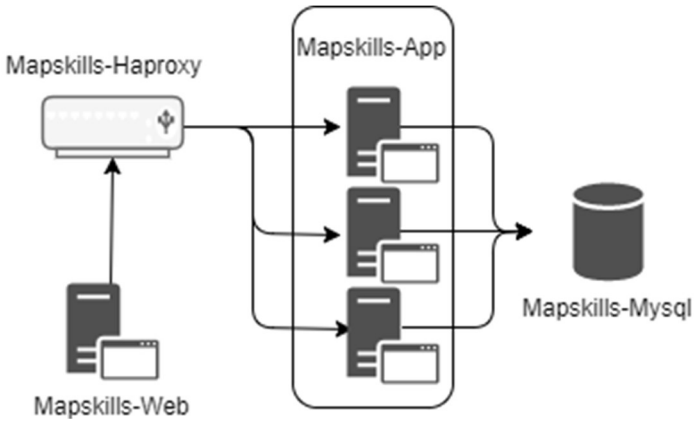
Além de inicializar os serviços, o arquivo docker-compose.yml é responsável por configurar como e com quem cada um dos containers irão se comunicar, bem como os volumes necessários para cada container.

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.



3.6.2 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.

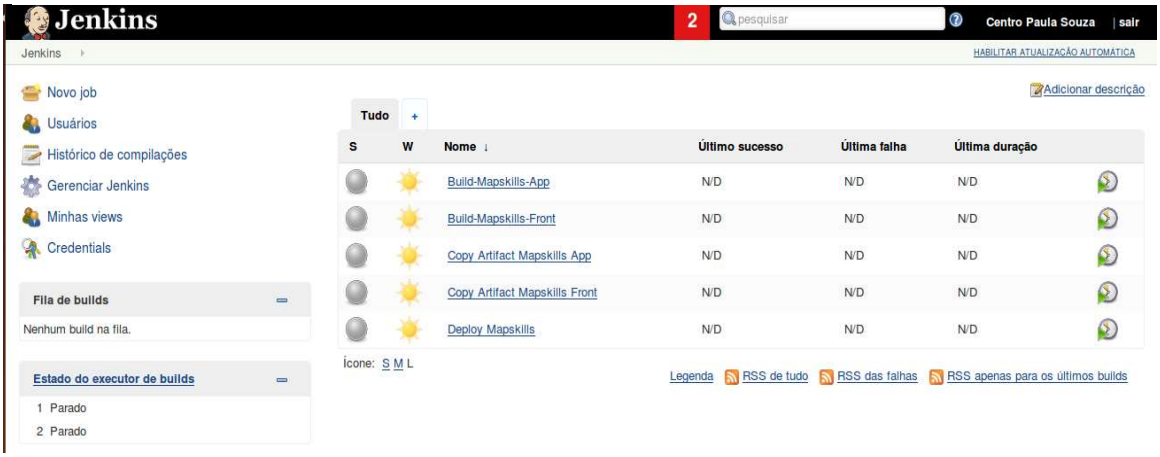
3.7 Mapskills-Jenkins

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



3.7.1 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
```

3.7.2 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
```

3.7.3 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
```

3.7.4 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 container* responsável por conter a aplicação de *front-end*.

3.7.5 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
```

4. 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.

4.1 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.

4.2 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.

5. 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

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

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

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

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

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

ANEXOS

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