



# Projeto de Bloco de Arquitetura e Infraestrutura de Aplicações

***PROF.º : Carlos Nilton***

***ALUNO: MARCELO CHAPELEM***

***TURMA : GRADUAÇÃO EM REDES***

***DE COMPUTADORES/MANHÃ***

***Matricula :037458347-12***

# Sumário

➤ Introdução.....	1
➤ Aplicação a ser implantada em uma infraestrutura de nuvem.....	2
➤ Diagrama e arquitetura da aplicação e seu banco de dados.....	3
➤ Implantação da aplicação distribuída virtualizada via SSH e seu Cronograma.....	5 e 6
➤ Captura das telas do processo de Implantação e configuração.....	7
➤ Implantação da aplicação e seus módulos em sua estrutura corporativa...	8
➤ Requisições de desenvolvimento e correção de bugs por parte dos usuários e Stakeholders.....	9
➤ Motivos para usar o Odoo na plataforma cloud compute AWS AWS.....	10
➤ Armazenamento do conteúdo do trabalho em um repositório GitHub, com o link contido no corpo do trabalho.....	11
➤ Print das telas com a falha na construção do Playbook.....	12
➤ Conclusões .....	13

# Introdução

## “ Odoo OpenERP Vs10 ”

O Sistema Odoo OpenERP, sigla (ERP) em inglês para *Enterprise Resource Planning*, é um software corporativo que, por meio de módulos, integra dados de uma empresa para garantir um fluxo de informações entre todos os departamentos. Para integrar departamentos, o ERP é estruturado por meio de módulos que, quando conectados, sincronizam as informações de várias áreas em um só sistema. Isso gera um fluxo de dados valioso, pois diminui erros evitando o retrabalho, otimiza processos e gera informações que garantem um planejamento eficaz. O objetivo é controlar e armazenar os dados de todos os processos e, com isso, aumentar a produtividade da equipe e dar uma base confiável de informações para possibilitar uma gestão estratégica com tomada de decisões mais assertivas . Para esse Assessment de bloco usaremos uma implantação em nuvem da aplicação Odoo OpenERP v10 na plataforma [Amazon AWS](#) em uma única instância t2-micro [ubuntu server 16.4 LTS](#) , com uma perspectiva Horizontal , Formada por módulos básicos e mais abrangentes para empresas que não exigem processos específicos. Esses módulos são comuns e necessários para os mais diferentes tipos de companhias. Por exemplo, o módulo contábil, financeiro, entre outros , para uma empresa pequena fictícia , vamos dar um MEI ( MEI – micro empreendedor Individual ) como exemplo que será enquadrado no Simples Nacional e ficará isento dos tributos federais (Imposto de Renda, PIS, Cofins, IPI e CSLL).O servidor de aplicação OpenERP, que contém toda a lógica empresarial e garante que OpenERP funcione melhor, e o servidor web, um aplicativo separado chamado o objeto aberto no cliente web, que permite que você se conecte a OpenERP de navegadores web padrão. As possibilidades são enormes. Hoje no Odoo você pode instalar o site corporativo e o e-commerce, caso trabalhe com ele online. Pode criar até mesmo uma rede social interna com blogs, grupos, chat de comunicação entre os colaboradores, chat de atendimento externo, forum, etc.

# **1- Aplicação a ser implantada em uma infraestrutura de nuvem.**

**Odoo** : Anteriormente chamado de OpenERP, como pode ser visto em seu próprio site, ali mesmo descrito como um Customer Relationship Management (CRM), ou seja, um sistema de Gestão de Relacionamento com o Cliente, integrado com um Enterprise Resource Planning (ERP), que quer dizer Sistema Integrado de Gestão OpenERP. Como funciona ? Para integrar departamentos, o ERP é estruturado por meio de módulos que, quando conectados, sincronizam as informações de várias áreas em um só sistema. Isso gera um fluxo de dados valioso, pois diminui erros evitando o retrabalho, otimiza processos e gera informações que garantem um planejamento eficaz. O ERP pode ser estruturado sob duas perspectivas: a “horizontal e a vertical” , vamos ver !

**PERSPECTIVA HORIZONTAL** : Formada por módulos básicos e mais abrangentes para empresas que não exigem processos específicos. Esses módulos são comuns e necessários para os mais diferentes tipos de companhias. Por exemplo, o módulo contábil, financeiro, entre outros.

**PERSPECTIVA VERTICAL** : Trabalha com módulos específicos para cada segmento de mercado, como exemplo temos a gestão de grandes fortunas, gestão hospitalar, de acervos, gestão ambiental, entre outros. Nesses casos, os sistemas contam com processos próprios de cada mercado, particularidades necessárias para a gestão de determinado tipo de negócio.

**Licença** : O Servidor e o cliente GTK+ do OpenERP é publicado sob a licençaGPL versão 3.0. O cliente Web está disponível através da "OpenERP Public Licence". que é uma licença derivada da "Mozilla Public License", gratuito para uso e modificações. A maior restrição importante é manter os logotipos originais da Tiny, ERP Open e Axelor nas páginas web visível para o usuário.

**Arquitetura** : O sistema possui três componentes principais: o servidor de banco de dados PostgreSQL,, O componente cliente web pode ser pensado como um servidor ou um cliente dependendo do seu ponto de vista. Ele atua como um servidor web para um usuário final a conexão a partir de um navegador web, mas também age como um cliente para o servidor de aplicativos OpenERP. Para você abrir as páginas web, e usar um navegador, portanto o navegador é o cliente e o site é o servidor, pois o primeiro acessa informações disponibilizadas pelo segundo. Desta forma, as redes que utilizam servidores que são chamados do tipo Cliente-Servidor. Assim, neste contexto o seu livro irá determinar se o componente cliente Web é referido como um servidor ou cliente.

**Banco de dados:** é o “**PostgreSQL**” um SGBDs (Sistema Gerenciador de Bancos de Dados) que contém todas as bases de dados cada um dos quais contém todos os dados e a maioria dos elementos da configuração do sistema ; de código aberto escrito em Python de alto desempenho ; as funcionalidades de negócio do Odoo são organizadas em "módulos". Um módulo é uma pasta com um script predefinido contendo código Python e arquivos XML. Um módulo define a estrutura e contém todas as bases de dados e a maioria dos elementos da configuração do sistema, OpenERP , formulários, relatórios, procedimentos, workflows , etc. Ele funciona em todos os tipos de sistema operacional, de Unix / Linux para as várias versões do Windows, via Mac OS X, Solaris, SunOS e BSD.

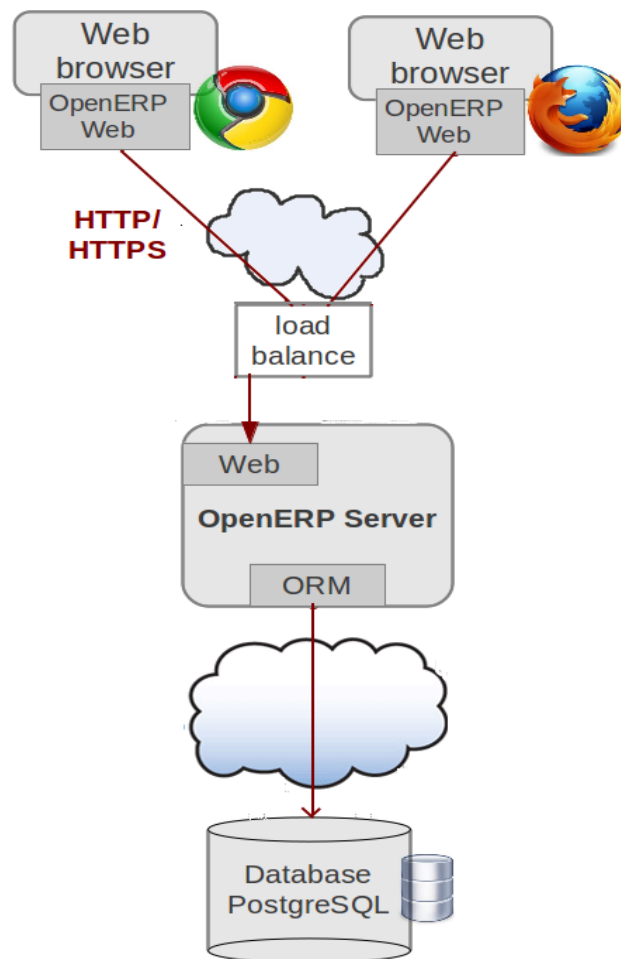
**O servidor de aplicação OpenERP**, que contém toda a lógica empresarial e garante que OpenERP funcione da melhor maneira, o servidor de aplicação OpenERP, que contém toda a lógica empresarial e garante que OpenERP corre da melhor maneira,

**O servidor web**, um aplicativo separado chamado o objeto aberto no cliente web, que permite que você se conecte a OpenERP de navegadores web padrão.

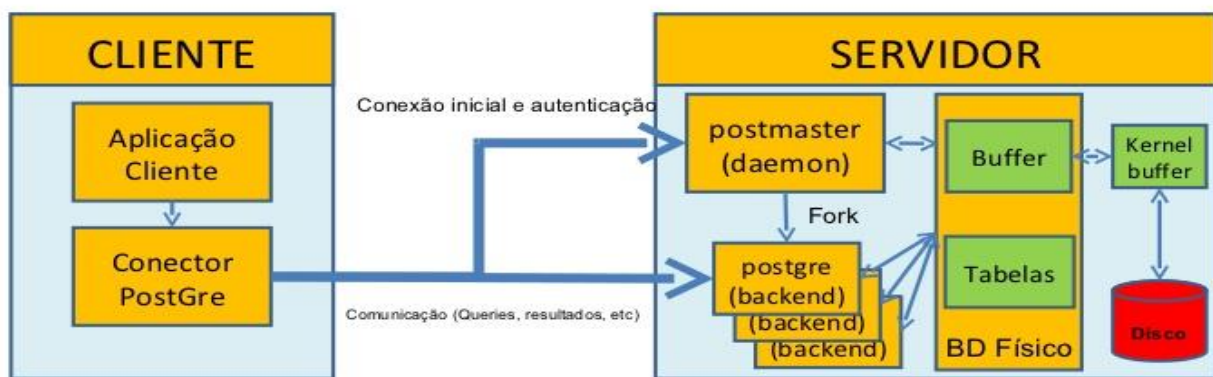
Estes três componentes podem ser instalados no mesmo servidor ou pode ser distribuído para servidores de computador separados, se considerações de desempenho necessário.

**Ambiente de desenvolvimento** : Não existe um ambiente de desenvolvimento integrado. O código Python deve ser editado em um editor externo. A lógica da aplicação (ou seja, os fluxos de trabalho e estrutura de dados) pode ser alterada através da interface do cliente. O Odoo hoje em dia é utilizado no mundo inteiro. Assim, surge a necessidade de adaptar suas funcionalidades à exigências locais. Por isso, desenvolve-se uma localização apropriada por usuários desenvolvedores de um país específico. No Brasil, temos a Localização Brasileira. Comunidade que conversa com os responsáveis pelo Odoo e disponibiliza módulos que adaptam o Odoo à funcionalidades requeridas no Brasil como a NF-e por exemplo.

## Diagrama da arquitetura da aplicação:



## Arquitetura do PostGres



# **Passo a passo de como será feita a implantação da aplicação distribuída virtualizada via SSH.**

## **PASSO 1 - Atualizar lista de fontes do Apt - sudo apt-get update**

## **PASSO 2 - Instalar atualizações**

```
sudo apt-get upgrade
```

## **ETAPA 3 - Instale Dependências do Python para Odoo**

```
sudo apt-get install python-dateutil python-docutils python-feedparser
python-jinja2 python-ldap python-libxslt1 python-lxml python-mako
python-mock python-openid python-psycopg2 python-psutil python-
pybabel python-pychart python-pydot python-pyparsing python-
reportlab python-simplejson python-tz python-unittest2 python-
vatnumber python-vobject python-webdav python-werkzeug python-xlwt
python-yaml python-zsi poppler-utils python-pip python-pypdf python-
passlib python-decorator gcc python-dev mc bzr python-setuptools
python-markupsafe python-reportlab-accel python-zsi python-yaml
python-argparse python-openssl python-egenix-mxdatetime python-usb
python-serial lptools make python-pydot python-psutil python-paramiko
poppler-utils python-pdftools antiword python-requests python-xlswriter
python-suds python-psycogreen python-ofxparse python-gevent
```

## **PASSO 4 - Odoo Web Dependencies**

```
sudo apt-get install -y npm
```

```
sudo ln -s /usr/bin/nodejs /usr/bin/node
```

```
sudo npm install -g less less-plugin-clean-css
```

## **PASSO 5 - Instale o PostgreSQL**

```
sudo apt-get install python-software-properties
```

```
sudo vim /etc/apt/sources.list.d/pgdg.list
```

Adicione uma linha para o repositório deb  
<http://apt.postgresql.org/pub/repos/apt/> xenial-pgdg main

```
wget --quiet -O -
https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key
add -
```

```
sudo apt-get update
```

```
sudo apt-get install postgresql-9.6
```

## **PASSO 6 - Criar usuário do banco de dados para Odoo**

```
sudo su postgres
```

```
cd
```

```
createuser -s odoo
```

```
createuser -s ubuntu_user_name
```

```
exit
```

## **PASSO 7 - Criar usuário e grupo Odoo**

```
sudo adduser --system --home=/opt/odoo --group odoo
```

## **PASSO 8 - Instale o Gdata**

```
cd /opt/odoo
```

Vá para o link "<https://pypi.python.org/pypi/gdata>" e baixe o gdata-2.0.18 e transfira o arquivo para o servidor.

Caso contrário, através do terminal, siga o passo abaixo:

```
sudo wget
https://pypi.python.org/packages/a8/70/bd554151443fe9e89d9a934a7891a
affc63b9cb5c7d608972919a002c03c/gdata-2.0.18.tar.gz
```

```
sudo tar zxvf gdata-2.0.18.tar.gz
```

```
sudo chown -R odoo: gdata-2.0.18
```

```
sudo -s
```

```
cd gdata-2.0.18/
```

```
python setup.py install
```

```
exit
```

**PASSO 9 - Odoo 10 Download de GitHub**Obtenha o último Odoo 10 do repositório github. Baixe o arquivo Zip do URL: "https://github.com/odoo/odoo/tree/10.0". Transfira o mesmo arquivo para o diretório / opt / odoo no servidor através do ftp.

#### Caso contrário, siga a seguir o Step

```
cd /opt/odoo
```

```
sudo wget https://github.com/odoo/odoo/archive/10.0.zip
```

```
sudo unzip 10.0.zip
```

```
sudo chown -R odoo: odoo-10.0
```

Or Git Clone Odoo

```
git clone --depth=1 --branch=10.0 https://github.com/odoo/odoo.git /opt/odoo/odoo
```

```
sudo mv odoo/ odoo-10.0/
```

```
sudo chown -R odoo: odoo-10.0
```

#### PASSO 10 - Criar arquivo de registro Odoo

```
sudo mkdir /var/log/odoo
```

```
sudo chown -R odoo:root /var/log/odoo
```

#### PASSO 11 - Edite o arquivo de configuração de Odoo

```
sudo cp /opt/odoo/odoo-10.0/debian/odoo.conf /etc/odoo.conf
```

```
sudo chown odoo: /etc/odoo.conf
```

```
sudo vim /etc/odoo.conf
```

#Copy and paste below content in config file , write correct addons paths

[options]

; This is the password that allows database operations:

; admin\_passwd = PASSWORD

db\_host = False

db\_port = False

db\_user = odoo

db\_password = False

addons\_path = /opt/odoo/odoo-10.0/addons

;Log Settings

logfile = /var/log/odoo/odoo.log

log\_level = error

Depois do fim do passo 11, fizemos a sequência a seguir:

```
cd
```

```
wget https://downloads.wkhtmltopdf.org/0.12/0.12.4/wkhtmltox-0.12.4_linux-generic-amd64.tar.xz
```

```
xz -d wkhtmltox-0.12.4_linux-generic-amd64.tar.xz
```

```
tar xvf wkhtmltox-0.12.4_linux-generic-amd64.tar
```

```
cd wkhtmltox/
```

```
sudo cp -Rav * /usr/local/
```

4. Criamos um script de inicialização com a sequência a seguir:

```
sudo vi /etc/systemd/system/odoo10.service
```

Coloque este conteudo no arquivo:

[Unit]

Description=Odoo

Documentation=http://www.odoo.com/

[Service]

# Ubuntu/Debian convention:

Type=simple

User=odoo

ExecStart=/opt/odoo/odoo-10.0/odoo-bin -c /etc/odoo.conf

[Install]

WantedBy=default.target

E depois faça:

```
sudo systemctl daemon-reload
```

```
sudo mkdir /var/lib/odoo
```

```
sudo chown odoo:root /var/lib/odoo -R
```

```
service odoo10 start
```

Pronto! Pode fazer o teste acessando o sua VM AWS assim: http://IP.DA.INST.ANCIA:8069

#### PASSO 12

**WKHTMLTOPDF para Odoo**Siga as etapas mencionadas no link abaixo e volte para a próxima etapa.

<https://www.getopenerp.com/wkhtmltopdf-for-odoo8/>

#### PASSO 13 - \*Agora, comece o servidor Odoo

```
cd /opt/odoo/odoo-10.0
```

```
./odoo-bin
```

#### PASSO 14

Acesse o navegador da Web para acessar o Odoo 10

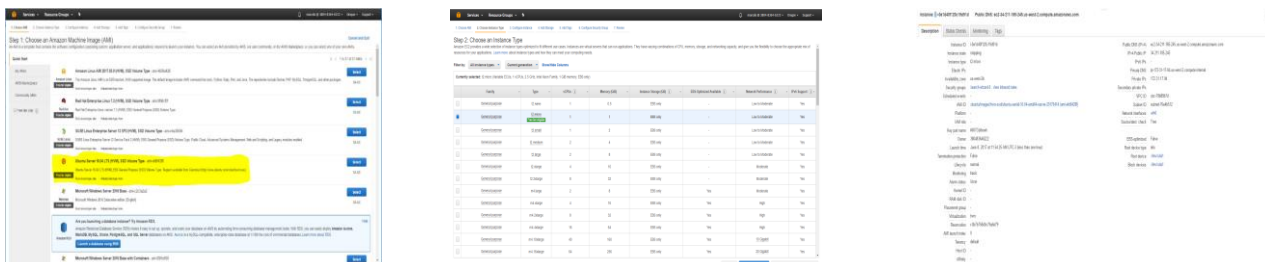
<http://localhost:8069>

**Um cronograma estimado com o prazo para execução da atividade.**  
2 horas e 55 minutos.

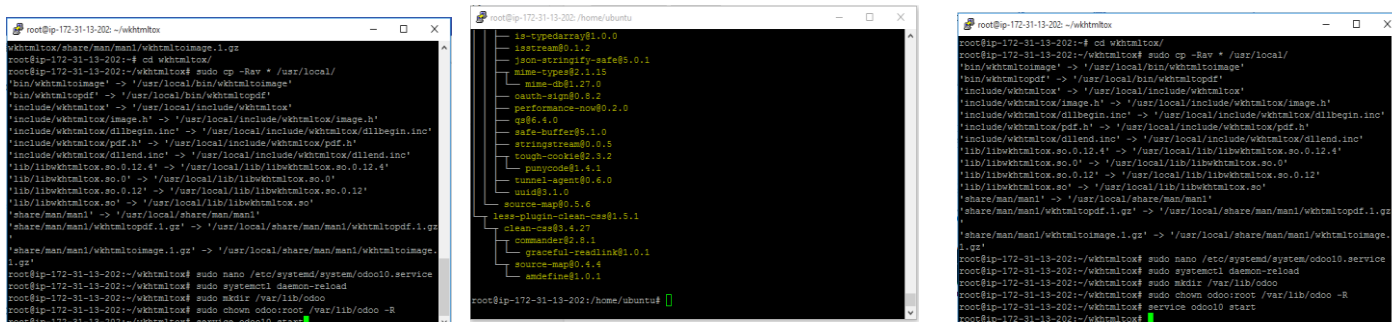
# Captura das telas do processo de Implantação e configuração

Primeiramente iremos criar uma instância na plataforma de computação em nuvem amazona AWS , após ter aberto uma conta “Estudante” fornecida gratuitamente pela Amazon .

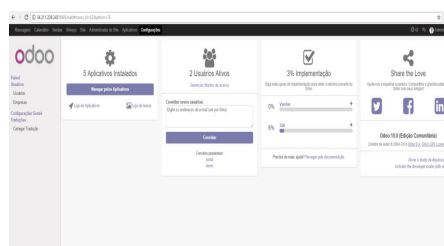
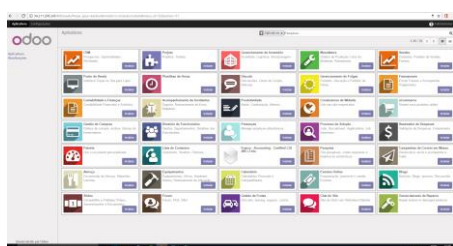
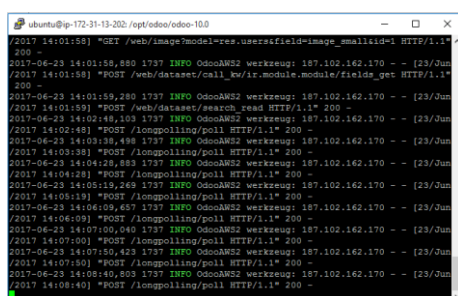
## Telas de criação da instância *ubuntu server 16.4 LTS /T2-micro via ssh.*



## Implantação da aplicação distribuída na instância ubuntu server 16.4 LTS /T2-micro via ssh na AWS



## Acesso a aplicação Odoo em funcionamento na instância Ubuntu na AWS





# Oficina de Serralheria

**Problema a ser tratado a partir de uma aplicação distribuída rodando sobre uma infraestrutura com virtualização.**

- Organização financeira - / Módulo CRM.
- Controle no pedido de Vendas – / Módulo contábil.
- Controle no pedido de compras – / Módulo Gestão compras .
- Controle de estoque de materiais - / módulo de gestão de estoque .
- Controle de aquisições de refeições e controle de horário de funcionários .

## **Implantação da aplicação em sua estrutura corporativa.**

- Nesta pequena empresa de caráter fictício iremos implantar cinco módulos do Odoo



**Módulo de Vendas e CRM** – A OpenERP CRM permite que você identifique seus melhores prospectos e oportunidades. Você pode personalizar o seu ciclo de vendas, estatísticas de controles e previsões e implementar a automação de campanhas de marketing para melhorar o seu desempenho de vendas



**Módulo de Contabilidade e Finanças** – Registre suas operações em poucos cliques e gerencie todas as suas atividades financeiras em um só local. Operações Financeiras nunca foram tão fáceis de realizar.



**Módulo de Compras** – Crie e localize facilmente seus pedidos de compras, administre as informações de seus fornecedores, controle o processo de recebimento de seus produtos e confira as faturas de fornecedores.

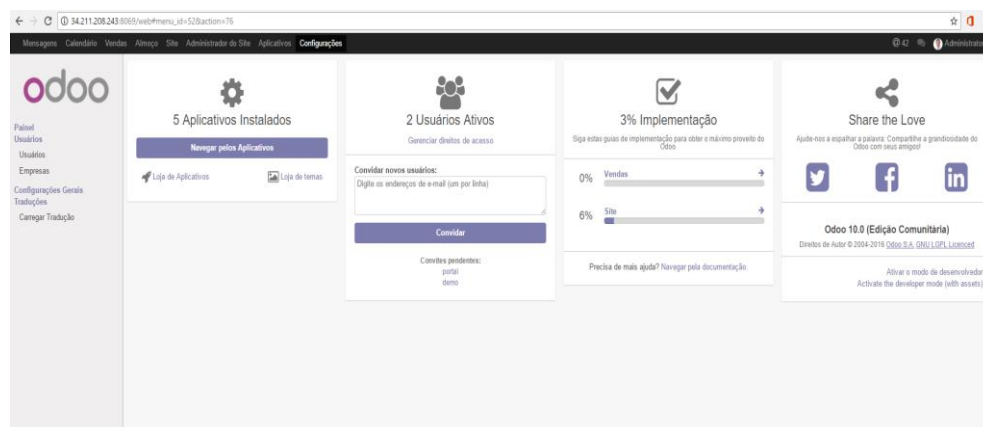


**Módulo de Gestão de Estoque** – A OpenERP inventou o sistema de gestão de estoque de entrada dupla, que possibilita a gestão de necessidades complexas de um modo bem fácil: identificação de estoques de fornecedores/clientes, rastreabilidade total, links de contabilidade, etc.



**Módulo almoço** - Muitas empresas pedem quentinhas, pizzas e outras refeições , de fornecedores habituais, para os seus empregados. O módulo “Almoço Order” foi desenvolvido para tornar a gestão de fornecedores mais fácil, e ágil , além de uma refeição completa , este módulo oferece a possibilidade de exibir aviso e fornece seleção rápida com base nas preferências do empregado. Se você quiser economizar tempo dos seus funcionários e controlar melhor as refeições e horários .

## **Módulos da aplicação instalados**

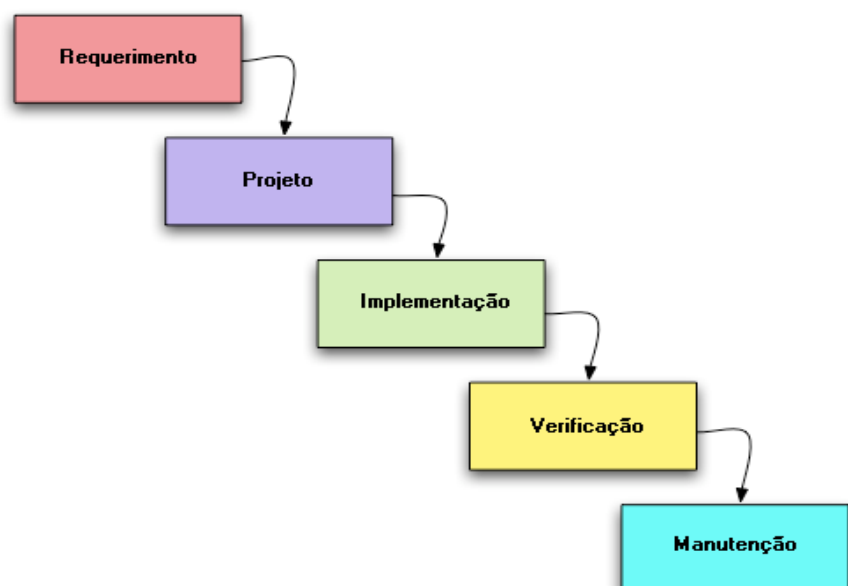


## Requisições de desenvolvimento e correção de bugs por parte dos usuários e stakeholders.

Durante o desenvolvimento de um software, Desenvolvedores buscam garantir a entrega de um produto com uma baixa quantidade de bugs no menor tempo possíveis , os teste do software é a investigação do software a fim de fornecer informações sobre sua qualidade em relação ao contexto em que ele deve operar. Isso inclui o processo de utilizar o produto para encontrar seus defeitos. Mesmo que o modelo de desenvolvimento de software a ser seguido seja o sequencial (como uma cascata ou Waterfall Mode, em inglês) no qual o desenvolvimento é visto com uma fluidez constante para frente ; Diante desse cenário, é fundamental que certas rotinas sejam tomadas, e algumas análises e correções sejam executadas no projeto com **“feedback”** de usuários ; sendo ainda que os modelos de criação de software mais tradicionais sejam focados no desenvolvimento de um software seguro. O Modelo em cascata estático. O andamento do processo flui de cima para baixo, como uma cascata. Nesse cenário em que softwares serão responsáveis pela manipulação de um número cada vez maior de informações, garantir que esses dados estejam seguros e o sistema funcional é fundamental para as empresas e para os seus usuários. Ao adotar boas práticas que levam ao desenvolvimento de software seguro e estável .

Suas etapas são:

- verificação dos requerimentos de software e sistema;
- análise e criação da estrutura do software
- desenvolvimento do código do programa;
- teste, depuração e busca por erros;
- instalação, suporte e manutenção de todos os sistemas interligados ao software.



## **Motivos para usar o Odoo:**

- 1) É realmente abrangente e utilizável em todos os ramos de atividade;
- 2) Tem uma localização Brasileira forte e muito bem estruturada aos padrões fiscais nacional, com NF-e, Impostos, NCM, etc;
- 3) Muito fácil de instalar suas funcionalidades básicas, roda em Windows (podendo ser instalado com poucos cliques) ou Linux;
- 4) É de graça;
- 5) Permite integração com varios sistemas;
- 6) usa tecnologia de ponta

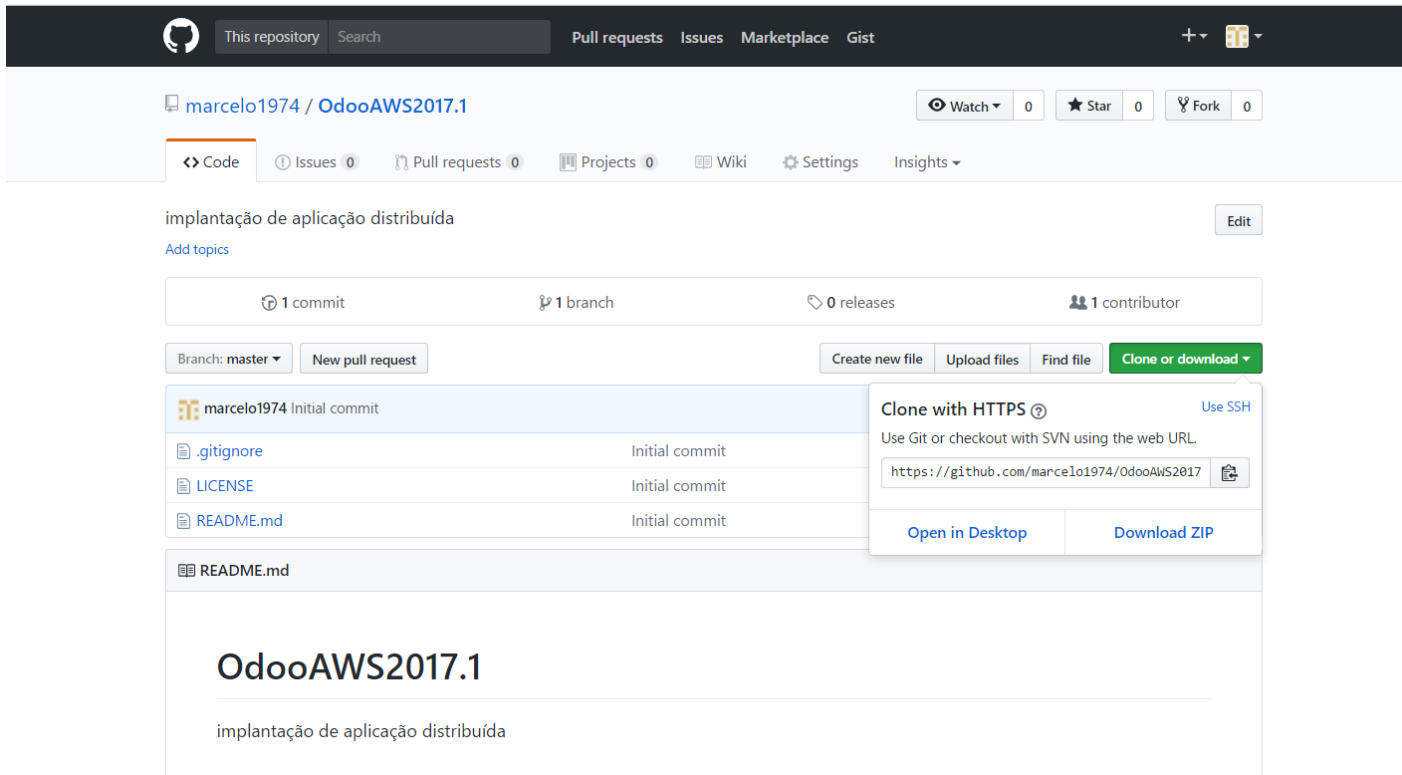
Para se ter uma ideia, tenho hoje um exemplo de empresa que possui o Odoo instalado online no AWS da Amazon, o mesmo utiliza uma loja virtual do proprio Odoo, o acessa de qualquer lugar que se tenha internet, seus funcionários andam pela loja com um tablet na mão, onde já atende o cliente andando pela loja e já emite o pedido para o caixa. Muito bom não? Um empresario Optante pelo MEI, dispensado de emitir NF-e, instala o Odoo em minutos na sua máquina e já pode gerenciar, seus clientes, seus fornecedores, seus produtos, suas vendas e suas compras, suas contas a pagar e a receber. Claro que a geração de NF-e e funcionalidades mais específicas demandam conhecimento em sua instalação e manutenção, sendo indispensável ter conhecimento de programação ou contratar um profissional que os tenha! Assim, considero o Odoo o sistema mais completo e acessível do mundo.

## **Motivos para usar o Odoo na AWS :**

**Alcançando ainda mais tolerância a falhas para seus aplicativos :** Quando não se consegue mais gerenciar todas as requisições e constantes atualizações de aplicações o que gera a indisponibilidade momentânea de aplicativos podemos usar os recursos de verificação de saúde e de failover de DNS do Amazon Route 53 para aprimorar a disponibilidade de aplicativos em execução por trás de Elastic Load Balancers , podemos criar aplicações tolerantes a falhas colocando instâncias do Amazon EC2 em várias zonas de disponibilidade. Para obter ainda mais tolerância a falhas com menos intervenção manual, podendo usar o Elastic Load Balancing. Podemos alcançar maior tolerância a falhas, colocando suas instâncias computacionais por trás de um Elastic Load Balancer, pois ele pode automaticamente equilibrar o tráfego entre várias instâncias e várias Zonas de disponibilidade e se certificar de que apenas as instâncias íntegras do Amazon EC2 recebam tráfego, podemos configurar um Elastic Load Balancer para equilibrar a carga de tráfego de entrada dos aplicativos entre instâncias do Amazon EC2 em uma única Zona de disponibilidade ou em várias Zonas de disponibilidade. O Elastic Load Balancing pode detectar a saúde das instâncias do Amazon EC2. Quando ele detecta instâncias com problemas de integridade do Amazon EC2, ele desvia o tráfego dessas instâncias. Em vez disso, ele distribui a carga entre as instâncias íntegras restantes do Amazon EC2.

# Armazenamento do conteúdo do trabalho em um repositório GitHub, com o link contido no corpo do trabalho.

## Captura de tela com o link do repositório do GitHub



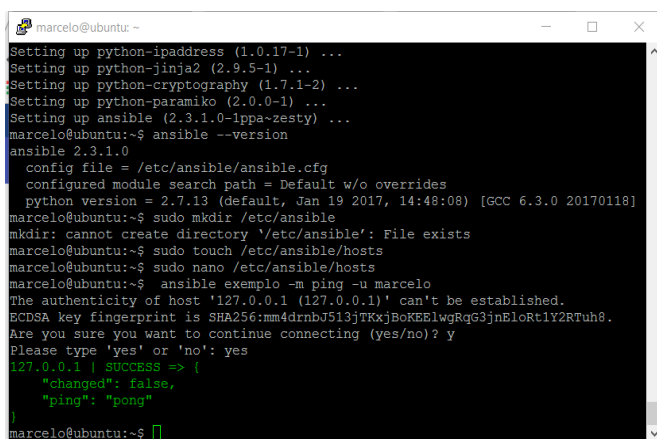
**<https://github.com/marcelo1974/OdooAWS2017.1.git>**

## PLAYBOOKS

São a base do gerenciamento de configuração e deployment de múltiplas máquinas.

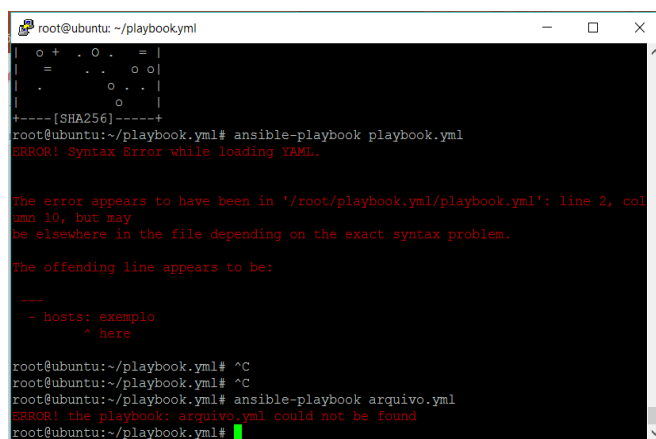
*Playbooks* contém declarações, mas também, podem orquestrar passos de deploy de aplicativos, podem ativar/desativar serviços e garantir que a configuração de um servidor está de um modo específico. O ideal é que os playbooks fique sob algum controle de versão - de preferência, o Git. Acostume-se a, sempre que alterar e testar um playbook, fazer o git commit, adicionando um comentário relevante sobre a mudança, para consultas futuras.

### Justificativa : falha na aplicação do playbook



```
marcelo@ubuntu: ~$ sudo apt-get install ansible
Setting up python-ipaddress (1.0.17-1) ...
Setting up python-jinja2 (2.9.5-1) ...
Setting up python-cryptography (1.7.1-2) ...
Setting up python-paramiko (2.0.0-1) ...
Setting up ansible (2.3.1.0-lppa-zesty) ...
marcelo@ubuntu:~$ ansible --version
ansible 2.3.1.0
  config file = /etc/ansible/ansible.cfg
  configured module search path = Default w/o overrides
  python version = 2.7.13 (default, Jan 19 2017, 14:48:08) [GCC 6.3.0 20170118]
marcelo@ubuntu:~$ sudo mkdir /etc/ansible
mkdir: cannot create directory '/etc/ansible': File exists
marcelo@ubuntu:~$ sudo touch /etc/ansible/hosts
marcelo@ubuntu:~$ sudo nano /etc/ansible/hosts
marcelo@ubuntu:~$ ansible exemplo -m ping -u marcelo
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:mm4drnbJ513jTKxjBoKEElwRqG3jnEloRt1Y2Rtuh8.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
127.0.0.1 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
marcelo@ubuntu:~$
```

Instalação do Ansible bem sucedida !



```
root@ubuntu: ~/playbook.yml
o + . 0 . = |
= . . 0 0 |
. . . |
o . . |
o |
+----[SHA256]-----+
root@ubuntu:~/playbook.yml# ansible-playbook playbook.yml
ERROR! Syntax Error while loading YAML.

The error appears to have been in '/root/playbook.yml/playbook.yml': line 2, col
umn 10, but may
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

---
- hosts: exemplo
  ^ here

root@ubuntu:~/playbook.yml# ^C
root@ubuntu:~/playbook.yml# ^C
root@ubuntu:~/playbook.yml# ansible-playbook arquivo.yml
ERROR! the playbook: arquivo.yml could not be found
root@ubuntu:~/playbook.yml#
```

Falha (error) na criação do Playbook !

# Conclusões

1. Se o prazo estabelecido no início do projeto foi adequado para a execução (não há problema se levou mais tempo que o planejado, mas se levou, explique as razões).

**Sim o prazo foi suficientemente satisfatório .**

2. Se os recursos planejados (quantidade de memória, disco, etc.) foram suficientes para colocar a solução em funcionamento. Se não foram, explique as razões e sugira uma nova configuração.

**Os recursos de Hardware foram o suficiente .**

3. Se as funcionalidades previstas na solução original funcionaram a contento. Se não foram, explique as razões.

**Sim funcionaram a contento .**

1. Quais seriam as melhorias futuras que poderiam ser feitas no projeto executado, adotando soluções adicionais, aperfeiçoando instalações ou tornando o cenário mais complexo. Melhorias futuras : **A configuração do Balanceador de carga (Load balance) de acordo com a demanda de usuários para instâncias em zonas de disponibilidade diferentes .**