**PLANO DE GERENCIAMENTO DE CONFIGURAÇÃO**

**SISTEMA DE PATRIMÔNIO**

SUMÁRIO

[1. INTRODUÇÃO 3](#_Toc504519626)

[1.1. Objetivos 3](#_Toc504519627)

[1.2. Evolução 3](#_Toc504519628)

[2. Gerência de Configuração de Software 4](#_Toc504519629)

[2.1. Ferramentas 4](#_Toc504519630)

[2.2. Ambientes](#_Toc504519631) 5

[3. o programa de gerenciamento de configuração 5](#_Toc504519632)

[3.1. Estrutura do Repositório 5](#_Toc504519633)

[4. Implantação 5](#_Toc504519634)

[4.1. Implantação…………………………………………………………………………. 5](#_Toc504519635)

4.2. Criação de novo projeto 6

[4.3. Dicas Uteis](#_Toc504519636) 8

# INTRODUÇÃO

O Plano de Gerenciamento de Configuração descreve todas as atividades do Gerenciamento de Controle de Configuração e Mudança que serão executadas durante o ciclo de vida do produto. Suas atividades envolvem identificar a configuração do software, manter sua integridade durante o projeto e controlar sistematicamente as mudanças.

## Objetivos

O objetivo deste documento é criar um padrão a ser seguido por todos os membros da equipe com o intuito de garantir o maior controle do produto no decorrer do projeto. Para que isso aconteça serão detalhados os recursos necessários (equipes, ferramentas e ambiente), as responsabilidades atribuídas e o cronograma de atividades.

## Evolução

O Plano de Gerenciamento de Configuração deve ser mantido atualizado para refletir o planejamento corrente. Dessa forma, as seguintes situações representam gatilhos para atualização do plano e nova aprovação deste documento:

* Mudança nos itens de configuração;
* Mudança na identificação dos arquivos;
* Mudança na identificação das *Tags*/*Branches*;
* Mudança no padrão de versionamento;

# Gerência de Configuração de Software

### Ferramentas

|  |  |  |
| --- | --- | --- |
| Termo | Versão | Descrição |
| *GitHub* | 0.0.0 | Ferramenta de administração dos repositórios, serviço disponibilizado no endereço: |
| Apache | 2.5 | Servidor Web, o qual funciona sob o protocolo HTTP, difundido em escala mundial. Instalação por meio do comando:  **# apt-get install apache2**  Para Inicializar a configuração do servidor e módulos adicionais:  **# apt-get install libapache2-mod-php7.0**  **# apt-get install php7.0-mysql**  **# apt-get install php7.0-curl** |
| MySql | 5.7 | Sistema de Gerenciamento de Banco de Dados, disponivel atraves do comando:  **# apt-get install mysql** |
| Laravel Framework | 5.4.32 | Framework de Desenvolvimento, serviço disponibilizado no endereço: [https://laravel.com](https://laravel.com/) ou  **# composer create-project --prefer-dist laravel/laravel nome-do-projeto** |
| Composer com Laravel | 1.5.1 | Gerenciador de Dependencias, serviço disponibilizado no endereço: <https://getcomposer.org/download/> ou pelos comandos  **# php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"**  **# php -r "if (hash\_file('SHA384', 'composer-setup.php') === '669656bab3166a7aff8a7506b8cb2d1c292f042046c5a994c43155c0be6190fa0355160742ab2e1c88d40d5be660b410') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP\_EOL;"**  **# php composer-setup.php**  **# php -r "unlink('composer-setup.php');"**  **# mv composer.phar /usr/local/bin/composer**  **# composer global require "laravel/installer"**  **# “~/.config/composer/vendor/bin/” ao PATH do terminal**  **#export PATH=$PATH:~/.config/composer/vendor/bin/** |
| *PHP* | 7.0 | Linguagem de Programação Utilizada, serviço disponibilizado no endereço: [http://php.net](http://php.net/) ou pelo comando:  **# apt-get install php** |
| Netbeans | 8.2.1 | IDE de desenvolvimento, download disponibilizado no endereço: <http://download.netbeans.org/netbeans/8.2/final/bundles/netbeans-8.2-windows.exe> ou  **$ sudo su -**  **#wget**  <http://download.netbeans.org/netbeans/8.2/final/bundles/netbeans-8.2-linux.sh>  **# chmod + x netbeans-8.0.2-php-linux.sh**  **# sh netbeans-8.0.2-php-linux.sh** |
| *Git* | 2.14.1 | Ferramenta de controle de versão, download disponibilizado no endereço: <https://git-scm.com/download/win> ou  **$ apt-get install git** |
| *MySqlWorkbench* | 6.3.1 | Ferramenta de Modelagem do Banco, download disponibilizado no endereço: <https://dev.mysql.com/downloads/workbench/> |
| *XAMMP* | 5.6.30 | Ferramenta Pacote Com Servidor Apache, PHP e SGBD Mysql para Desenvolvimento no Windows, download disponibilizado no endereço: [https://www.apachefriends.org](https://www.apachefriends.org/) |

### Ambientes

O ambiente que será entregue a equipe de desenvolvimento, deverá ser mantido pela equipe de arquitetura, através de *Virtual* *Machines* que seguiram os padrões dos ambientes mantidos pela equipe de infraestrutura. As ferramentas de desenvolvimento “*IDEs*” serão de livre escolha do desenvolvedor, desde que a mesma seja uma ferramenta de Software Livre, tais como *Atom*, *Eclipse*, mas de preferência Netbeans...

# o programa de gerenciamento de configuração

### Estrutura do Repositório

O repositório do projeto está armazenado no GitHub, podendo ser acessado pelo link: https://github.com/fgpacheco/Projeto\_APS.

# Implantação

## Implantação a partir do repositório existente

* 1. **Clone o projeto**

git clone https://github.com/FuturoDaUAG/ProjetoPW.git

* 1. **Acesse o projeto**

cd seuprojeto

* 1. **Instale as dependências e o framework**

composer update

composer install --no-scripts

* 1. **Copie o arquivo .env.example**

cp .env.example .env

* 1. **Crie uma nova chave para a aplicação**

php artisan key:generate

* 1. **Em seguida você deve configurar o arquivo .env e rodar as migrations com:**

php artisan migrate –seed

## Criação de novo projeto

Após a instalação das ferramentas da sessão 2.1, executar os comandos na sequencia abaixo para a criação de um novo projeto:

**a) Ambiente Linux**

# cd~

# mkdir nomedoprojeto

# cd nomedoprojeto

# laravel new exemplo

Edite o arquivo /etc/hosts, para incluir o endereço de seu projeto. Para o nosso exemplo, iremos inserir a chamada no *host* local, conforme segue:

127.0.0.1 exemplo.dev

Acrescentar ao arquivo de configurações do Apache o novo projeto criado com o seu endereço. No diretório /etc/apache2/apache.conf, inserir:

<VirtualHost \*:80>

ServerName exemplo.dev

DocumentRoot **/home/usuario/nomedoprojeto/exemplo/public**

<Directory "**/home/usuario/nomedoprojeto/exemplo/public**">

Options Indexes FollowSymLinks MultiViews

AllowOverride All

Order allow,deny

allow from all

Require all granted

</Directory>

</VirtualHost>

Dar permissão aos usuários para escrita nos diretórios **storage** e **bootstrap/cache**, acessando o respectivos diretório e executando o comando:

**$chmod 777 storage**

**$chmod 777 bootstrap/cache**

Observação: Para garantias de segurança, as permissões de escrita devem se restringir aos diretórios informados

Ainda acerca de segurança, deve ser gerada uma chave de string aleatória, a qual possuirá 32 caracteres, com o comando:

**$ php artisan key:generate**

Para a criação do Controller, que irá realizar as comunicações entres as camadas da aplicação deverá ser executado o comando:

**$ php artisan make:controller nomedocontroller**

Para a criação do modelo do banco de dados, deve ser executado o comando abaixo:

**$ php artisan make:model -m nomedomodelo**

Ao inserir o comando “-m” são gerados também os migrations, os quais servirão como inteface com a base de dados. Executados os comandos acima, por meio do comando “migrate”, são migrados os modelos gerados para a criação das tabelas no banco de dados.

**$ php artisan migrate**

## Dicas Uteis

Ao apagar Migrations e Models de modo manual e tentar cria-los novamente:

* 1. Delete manualmente migration e model
  2. Resetar arquivos autoload do composer com o comando: composer dump-autoload