

MANUAL TÉCNICO

Morant

**APLICACIÓN WEB PARA DETERMINAR EL NIVEL DE SEVERIDAD DE LA
ENFERMEDAD ANTRACNOSIS EN EL CULTIVO DE MORA**

JORGE ALONSO TORO HOYOS

**UNIVERSIDAD TECNOLÓGICA DE PEREIRA
CIDT CENTRO DE INNOVACIÓN Y DESARROLLO TECNOLÓGICO
PEREIRA, MAYO 2017**

CONTENIDO

INTRODUCCIÓN.....	4
1.1 Términos y Abreviaciones.....	4
2. Requerimientos Técnicos.....	5
3. Configuraciones del sistema.....	6
3.1 Actualización del sistema operativo.....	6
3.2 Crear un usuario ‘morant’.....	6
4. Servidor Web.....	7
4.1 Instalación Nginx.....	7
5. Despliegue de la App.....	8
5.1.1 Ambiente.....	8
5.1.2 Morant App.....	8
5.1.3 uWSGI.....	9
6. Despliegue de la Aplicación.....	10
6.1.1 Configuración Nginx.....	10
6.1.2 Habilitar el Site.....	11
6.2.1 Configurar uWSGI.....	11
7. Daemon.....	12
8. Puesta en Marcha.....	13
9. Servidor Smarthost.....	14
9.1 Instalación MTA.....	14
9.2 Configurar el Smarthost.....	14
9.2 Configurar la cuenta.....	17
9.3 Restart.....	18

Título del Documento:	Manual Técnico Aplicación Web Morant
Revisión:	1.0.0
Fecha:	18/05/2017
Elaborado por:	Jorge Alonso Toro <jolthgs@gmail.com>

INTRODUCCIÓN

Morant es una aplicación que permite determinar el nivel de severidad de la enfermedad antracnosis en el cultivo de mora. Morant cuenta con una **aplicación web** que le permite a los usuarios enviar una imagen y determinar el nivel de la enfermedad antracnosis en el cultivo.

En este manual se describe los pasos necesarios para la puesta en marcha de la aplicación web.

1.1 Términos y Abreviaciones

Términos	Descripción
App o APP	Aplicación Informática
Python	Lenguaje de programación
Nginx	Servidor web/proxy inverso
WSGI	Web Server Gateway Interface
Daemon	Proceso informático no interactivo

2. Requerimientos Técnicos

HARDWARE

Intel(R) Xeon(R) CPU E5-2683 v3 @ 2.00GHz

1GB RAM

40GB HD

OPERATING SYSTEM

Debian GNU/Linux o sus derivados

SOFTWARE

Python

Nginx

NETWORK CONNECTION

Es necesario que el servidor Web esté expuesto a Internet por medio de una IP pública.

3. Configuraciones del sistema

3.1 Actualización del sistema operativo

Realizar actualización de la base de datos y los paquetes del sistema operativo:

```
# apt update && apt upgrade --yes && apt full-upgrade --yes
```

3.2 Crear un usuario 'morant'

Para un funcionamiento más seguro de la aplicación se creará el usuario *morant* propietario de los ficheros de la aplicación. Creamos y asignamos un password al usuario:

```
# useradd --shell /bin/bash --home-dir /var/local/morant morant
```

```
# passwd morant
```

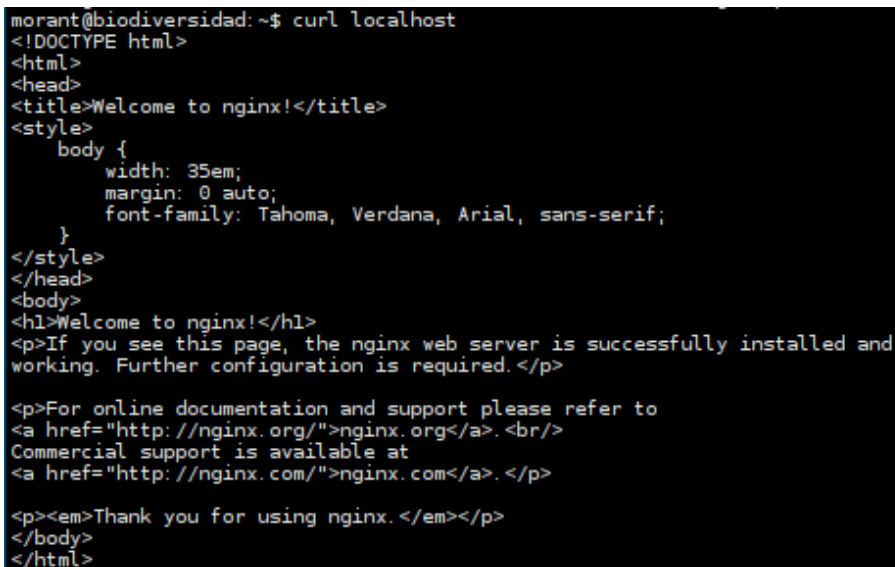
4. Servidor Web

4.1 Instalación Nginx

Para instalar Nginx en el servidor sólo es necesario correr los comandos:

```
$ sudo apt-get update  
$ sudo apt-get install nginx
```

Si queremos saber si nuestro servidor web está instalado correctamente y funcional solo hay que realizar una petición http al mismo, como se muestra en la imagen.



```
morant@biodiversidad:~$ curl localhost  
<!DOCTYPE html>  
<html>  
<head>  
<title>Welcome to nginx!</title>  
<style>  
  body {  
    width: 35em;  
    margin: 0 auto;  
    font-family: Tahoma, Verdana, Arial, sans-serif;  
  }  
</style>  
</head>  
<body>  
<h1>Welcome to nginx!</h1>  
<p>If you see this page, the nginx web server is successfully installed and  
working. Further configuration is required.</p>  
  
<p>For online documentation and support please refer to  
<a href="http://nginx.org/">nginx.org</a>.<br/>  
Commercial support is available at  
<a href="http://nginx.com/">nginx.com</a>.</p>  
  
<p><em>Thank you for using nginx.</em></p>  
</body>  
</html>
```

Ilustración 1: Prueba servidor web

5. Despliegue de la App

5.1.1 Ambiente

Ya que la aplicación fue desarrollada en *Python* es necesario instalar las dependencias necesarias para su funcionamiento.

Por defecto todas las distro *GNU/Linux* cuentan ya con Python instalado, podemos verificar que se encuentre en nuestro sistema con el comando:

```
morant@biodiversidad:~$ python --version  
Python 2.7.12
```

Ilustración 2: Versión de Python

Ahora procederemos a instalar los paquetes necesarios para nuestro ambiente:

```
$ sudo apt install python-dev python-pip python-virtualenv build-essential
```

```
$ pip install web.py
```

5.1.2 Morant App

Para realizar la instalación de la aplicación se debe contar con las fuentes *morant.tar.gz* donde se encuentran los ficheros necesarios para realizar la instalación y configuración.

```
$ tar -xvzf morant.tar.gz
```

```
$ cd morant
```


5.1.3 uWSGI

Para la comunicación WSGI con el Servidor Web se va a usar uWSGI con soporte para python.

```
$ sudo pip install uwsgi
```

6. Despliegue de la Aplicación

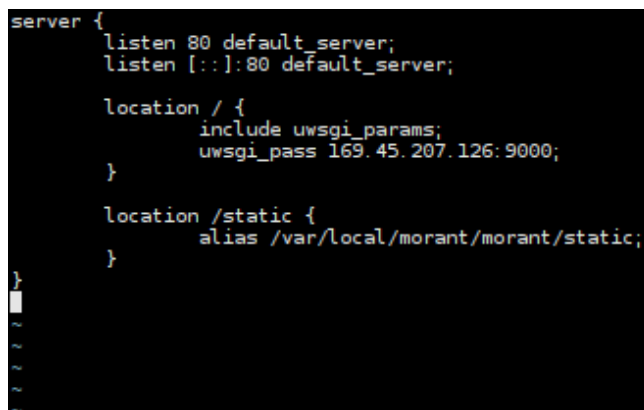
6.1.1 Configuración Nginx

Para el despliegue de la aplicación se realizará la configuración del Nginx como un proxy.

Se debe crear un site válido para Nginx en sus ficheros de configuración en `/etc/nginx/`. Nuestro site lo vamos a llamar *morantapp* y se creará de la siguiente manera:

```
$ sudo vim /etc/nginx/sites-available/morantapp
```

La configuración sería la siguiente:

A screenshot of a terminal window with a black background and white text. It displays the Nginx configuration for the 'morantapp' site. The configuration includes a 'server' block with two 'listen' directives (80 and [::]:80), a 'location /' block that includes 'uwsgi_params' and passes requests to '169.45.207.126:9000', and a 'location /static' block that aliases the path to '/var/local/morant/morant/static'. The terminal shows the end of the file with '~' characters and a cursor at the bottom.

```
server {  
    listen 80 default_server;  
    listen [::]:80 default_server;  
  
    location / {  
        include uwsgi_params;  
        uwsgi_pass 169.45.207.126:9000;  
    }  
  
    location /static {  
        alias /var/local/morant/morant/static;  
    }  
}  
~  
~  
~  
~
```

Ilustración 3: Configuración morantapp

6.1.2 Habilitar el Site

Al tener listo el site solo falta habilitar esta configuración, así:

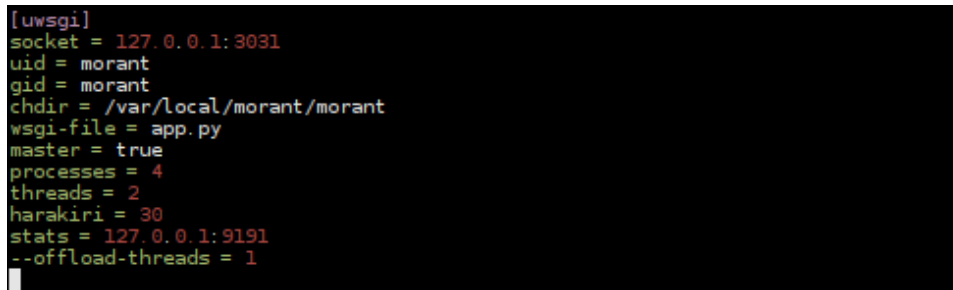
```
$ sudo rm /etc/nginx/sites-enabled/default  
$ sudo ln -s /etc/nginx/sites-available/morantapp /etc/nginx/sites-enabled
```

6.2.1 Configurar uWSGI

Ahora es necesario crear un fichero de configuración para la comunicación con el servidor web, pero antes se debe crear un directorio dónde alojar dichas configuraciones:

```
$ sudo mkdir -p /etc/uwsgi/sites
```

Seguido se debe crear la configuración de uWSGI para la App en un fichero que llamaremos *uwsgi.ini*:



```
[uwsgi]  
socket = 127.0.0.1:3031  
uid = morant  
gid = morant  
chdir = /var/local/morant/morant  
wsgi-file = app.py  
master = true  
processes = 4  
threads = 2  
harakiri = 30  
stats = 127.0.0.1:9191  
--offload-threads = 1
```

Ilustración 4: uwsgi.ini

y creamos un enlace a la configuración:

```
$ sudo ln -s /var/local/morant/morant/uwsgi.ini /etc/uwsgi/sites/
```

7. Daemon

Para el despliegue automático de la aplicación se debe crear un *daemon* que ejecute las configuraciones necesarias. Este daemon se realizará por medio de una unidad que llamaremos *morant.service*, así:

```
$ sudo vim /etc/systemd/system/morant.service
```

Con la siguiente configuración:

```
[Unit]
Description=uWSGI Emperor service - Morant App

[Service]
ExecStartPre=/bin/bash -c 'mkdir -p /run/uwsgi; chown morant:www-data /run/uwsgi'
ExecStart=/usr/local/bin/uwsgi --emperor /etc/uwsgi/sites
Restart=always
KillSignal=SIGQUIT
Type=notify
NotifyAccess=all

[Install]
WantedBy=multi-user.target
~
~
~
~
```

Ilustración 5: unidad *morant.service*

8. Puesta en Marcha

Ahora que tenemos todo configurado sólo sería necesario iniciar el servidor y la aplicación:

Iniciar Nginx:

```
$ sudo systemctl restart nginx.service
```

Habilitar e iniciar nuestra unidad:

```
$ sudo systemctl enable morant.service
```

```
$ sudo systemctl start morant.service
```

9. Servidor Smarthost

Este tipo de transferencia de mensajes le permite a un SMTP enrutar mensajes a un servidor de correo intermediario en lugar de hacerlo directamente al servidor de destino.

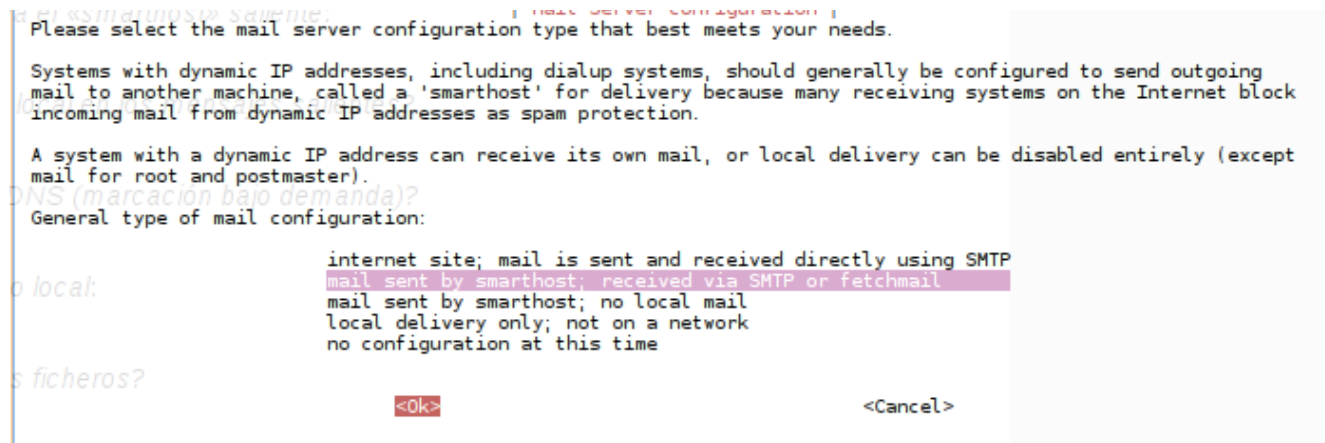
9.1 Instalación MTA

```
$ sudo apt install exim4
```

9.2 Configurar el Smarthost

```
$ sudo dpkg-reconfigure exim4-config
```

Seguido, se debe realizar la siguiente configuración deacurdo a las imagenes a continuación:



Mail Server configuration

The 'mail name' is the domain name used to 'qualify' mail addresses without a domain name. This name will also be used by other programs. It should be the single, fully qualified domain name (FQDN). Thus, if a mail address on the local host is foo@example.org, the correct value for this option would be example.org.

This name won't appear on From: lines of outgoing messages if rewriting is enabled.

System mail name:

biodiversidad

<Ok> <Cancel>

Mail Server configuration

Please enter a semicolon-separated list of IP addresses. The Exim SMTP listener daemon will listen on all IP addresses listed here. An empty value will cause Exim to listen for connections on all available network interfaces.

If this system only receives mail directly from local services (and not from other hosts), it is suggested to prohibit external connections to the local Exim daemon. Such services include e-mail programs (MUAs) which talk to localhost only as well as fetchmail. External connections are impossible when 127.0.0.1 is entered here, as this will disable listening on public network interfaces.

IP-addresses to listen on for incoming SMTP connections:

127.0.0.1 ; ::1

<Ok> <Cancel>

Mail Server configuration

Please enter a semicolon-separated list of recipient domains for which this machine should consider itself the final destination. (These domains are commonly called 'local domains'. The local hostname (biodiversidad.cidt.com) and 'localhost' are always added to the list given here.

By default all local domains will be treated identically. If both a.example and b.example are local domains, acc@a.example and acc@b.example will be delivered to the same final destination. If different domain names should be treated differently, it is necessary to edit the config files afterwards.

Other destinations for which mail is accepted:

añs ficheros?

<Ok> <Cancel>

Mail Server configuration

Please enter a semicolon-separated list of IP address ranges for which this system will unconditionally relay mail, functioning as a smarthost. You should use the standard address/prefix format (e.g. 194.222.242.0/24 or 5f03:1200:836f::/48). If this system should not be a smarthost for any other host, leave this list blank.

Machines to relay mail for:

añs ficheros?

<Ok> <Cancel>

Manual Técnico – Aplicación Web Morant

Mail Server configuration

Please enter the IP address or the host name of a mail server that this system should use as outgoing smarthost. If the smarthost only accepts your mail on a port different from TCP/25, append two colons and the port number (for example smarthost.example::587 or 192.168.254.254::2525). Colons in IPv6 addresses need to be doubled.

If the smarthost requires authentication, please refer to the Debian-specific README files in /usr/share/doc/exim4-base for notes about setting up SMTP authentication.

IP address or host name of the outgoing smarthost:

smtp.gmail.com::587

<Ok><Cancel>

Mail Server configuration

The headers of outgoing mail can be rewritten to make it appear to have been generated on a different system. If this option is chosen, 'biodiversidad', 'localhost' and '' in From, Reply-To, Sender and Return-Path are rewritten.

Hide local mail name in outgoing mail?

<Yes><No>

Mail Server configuration

In normal mode of operation Exim does DNS lookups at startup, and when receiving or delivering messages. This is for logging purposes and allows keeping down the number of hard-coded values in the configuration.

If this system does not have a DNS full service resolver available at all times (for example if its Internet access is a dial-up line using dial-on-demand), this might have unwanted consequences. For example, starting up Exim or running the queue (even with no messages waiting) might trigger a costly dial-up-event.

This option should be selected if this system is using Dial-on-Demand. If it has always-on Internet access, this option should be disabled.

Keep number of DNS-queries minimal (Dial-on-Demand)?

<Yes><No>

Mail Server configuration

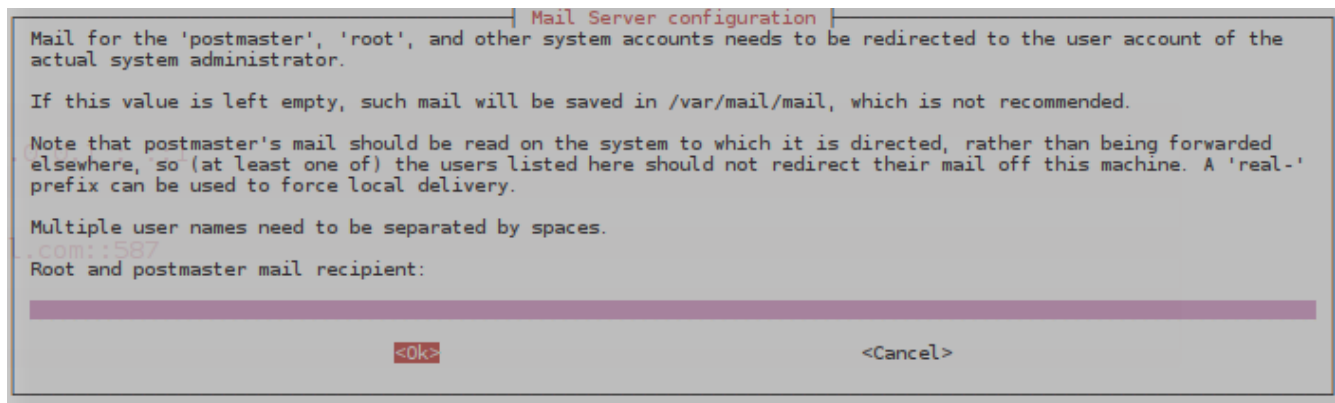
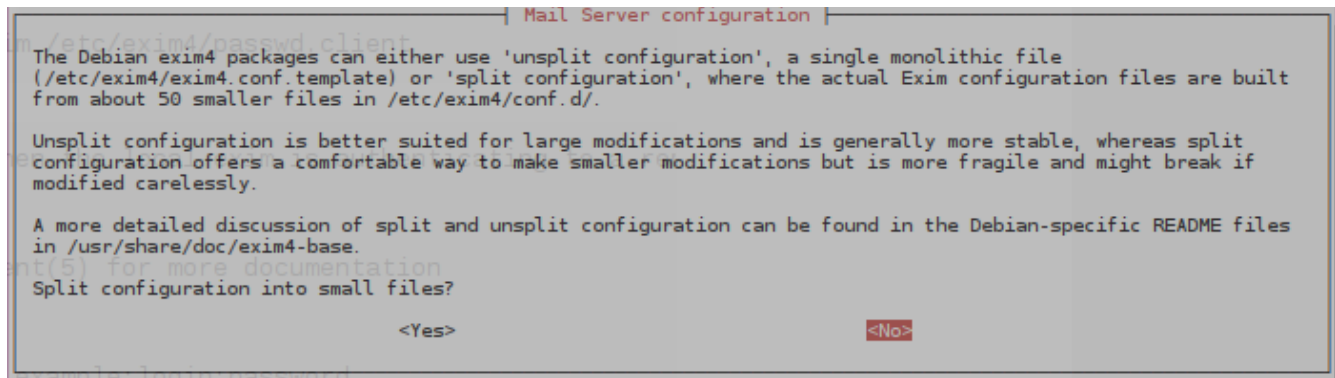
Exim is able to store locally delivered email in different formats. The most commonly used ones are mbox and Maildir. mbox uses a single file for the complete mail folder stored in /var/mail/. With Maildir format every single message is stored in a separate file in ~/Maildir/.

Please note that most mail tools in Debian expect the local delivery method to be mbox in their default.

Delivery method for local mail:

mbox format in /var/mail/
Maildir format in home directory

<Ok><Cancel>



9.2 Configurar la cuenta

Editamos la cuenta en el fichero `/etc/exim4/passwd.client` con el siguiente contenido:

```
$ sudo vim /etc/exim4/passwd.client
```

```
gmail-smtp.l.google.com:soportemorant@gmail.com:password_here  
*.google.com:soportemorant@gmail.com:password_here  
smtp.gmail.com:soportemorant@gmail.com:password_here
```

9.3 Restart

```
$ sudo systemctl restart exim4.service
```