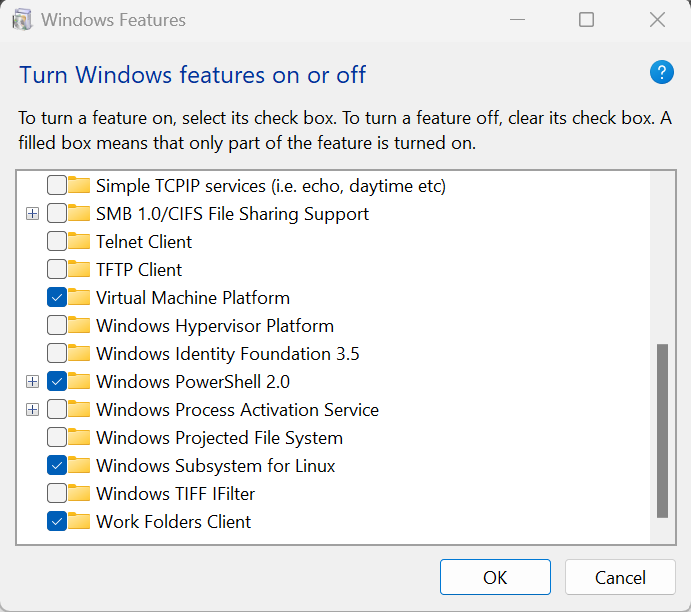
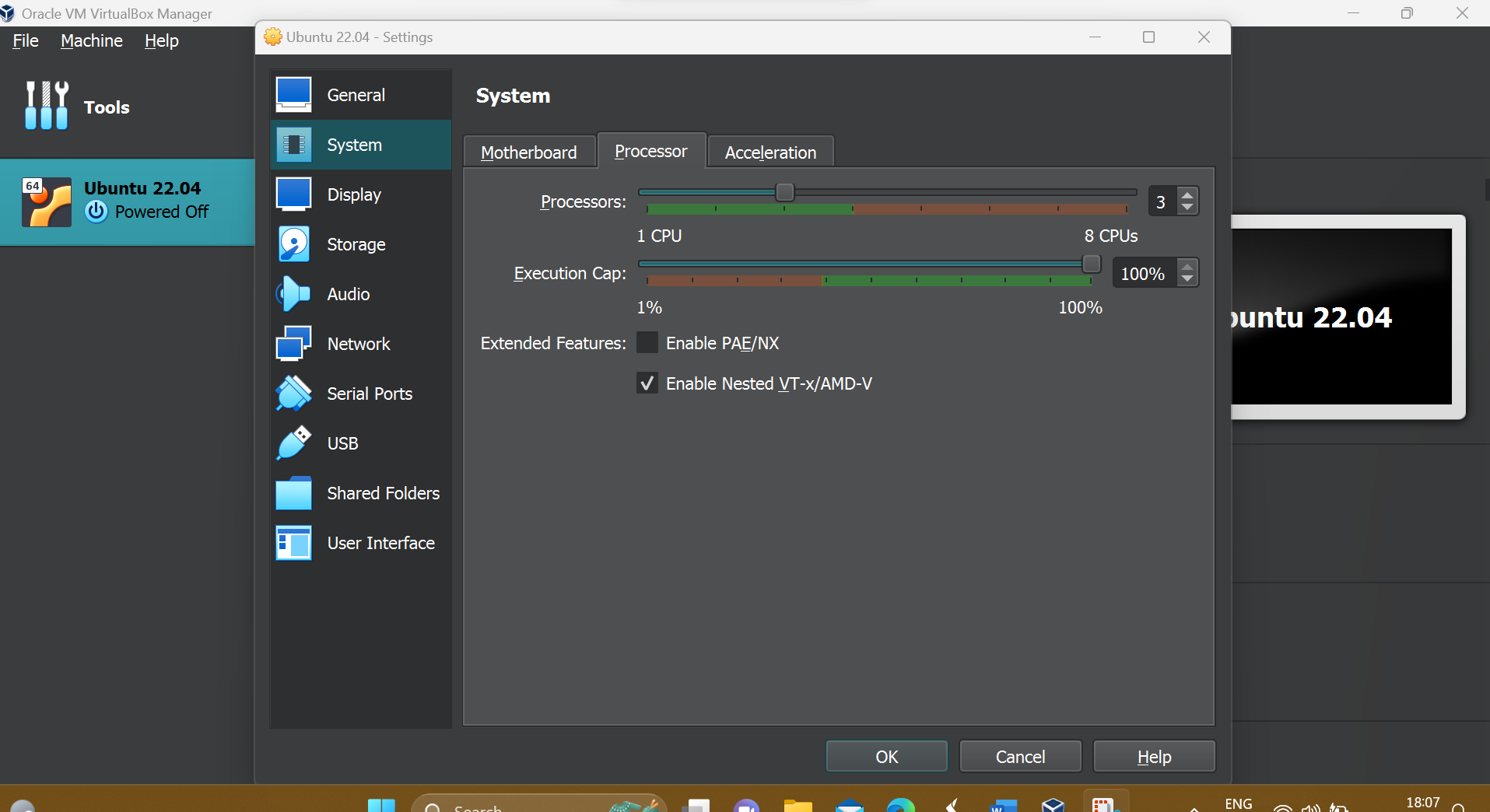
1. **Enable Window Subsystem for Linux (WSL)**

In Turn windows features on or off check the box of window subsystem for Linux and restart the machine.

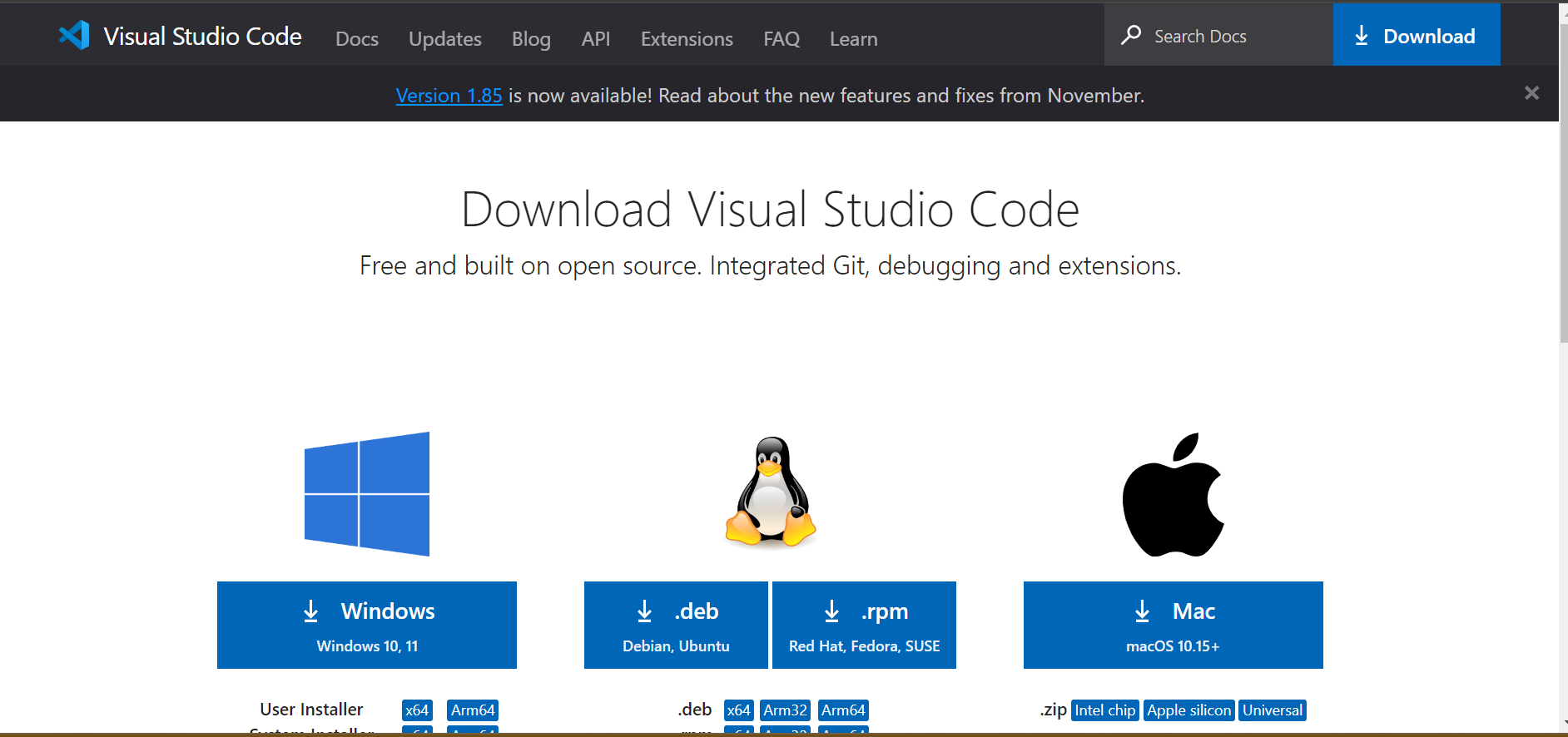


1. **Install VirtualBox 7.0 latest version**
2. **Install ubuntu 22.04 iso file and open in virtual machine**

Enable Nested VT-x/AMD-V in ubuntu 22.04 –system—processor—Extended Features.



1. **Install vscode for ubuntu**

****

Download .deb file for installation of vscode

Then run following cmd in terminal

$sudo apt install ./<file>.deb

1. **Install docker desktop for ubuntu 22.04 in VirtualBox**

To install Docker Desktop successfully, your Linux host must meet the following general requirements:

64-bit kernel and CPU support for virtualization.

KVM virtualization support. Follow the KVM virtualization support instructions to check if the KVM kernel modules are enabled and how to provide access to the KVM device.check official documentation of docker ubuntu.

QEMU must be version 5.2 or later.

Gnome, KDE, or MATE Desktop environment

At least 4 GB of RAM



Download DEB package then in terminal type

Cmd:

$ cd Downloads

$sudo apt-get update

$sudo apt-get install ./docker-desktop-<version>-<arch>.deb

After you’ve successfully installed Docker Desktop, you can check the versions of these binaries by running the following commands:

$ docker compose version

Docker Compose version v2.17.3

$ docker --version

Docker version 23.0.5, build bc4487a

$ docker version

Client: Docker Engine - Community

Cloud integration: v1.0.31

Version: 23.0.5

API version: 1.42

<...>

1. **In terminal :** Cmd

$ cd Documents

$git clone https://github.com/frappe/frappe\_docker.git

$cd frappe\_docker

$cp -R devcontainer-example .devcontainer

$cp -R development/vscode-example development/.vscode

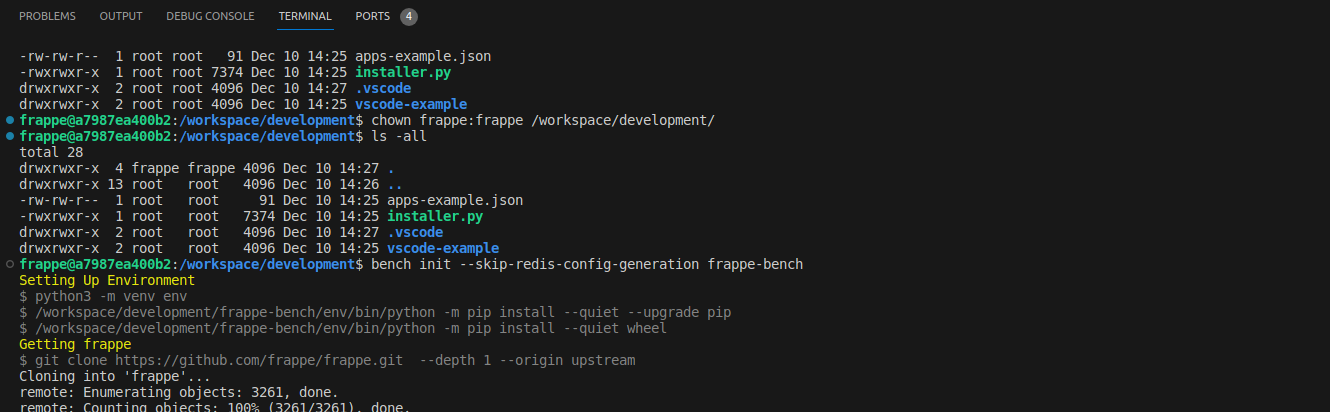
$code --install-extension ms-vscode-remote.remote-containers

$code .

**7. Run the following commands in the terminal inside the container. You might need to create a new terminal in VSCode.**

NOTE: Prior to doing the following, make sure the user is frappe.

Cmd: $bench init --skip-redis-config-generation --frappe-branch version- 14 frappe-bench



**Setup hosts**

We need to tell bench to use the right containers instead of localhost. Run the following commands inside the container:

bench set-config -g db\_host mariadb

bench set-config -g redis\_cache redis://redis-cache:6379

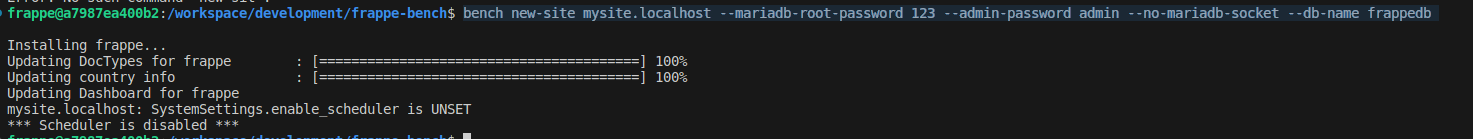
bench set-config -g redis\_queue redis://redis-queue:6379

bench set-config -g redis\_socketio redis://redis-queue:6379

sed -i '/redis/d' ./Procfile

**Create a new site with bench**

$bench new-site mysite.localhost --mariadb-root-password 123 --admin-password admin --no-mariadb-socket --db-name frappedb



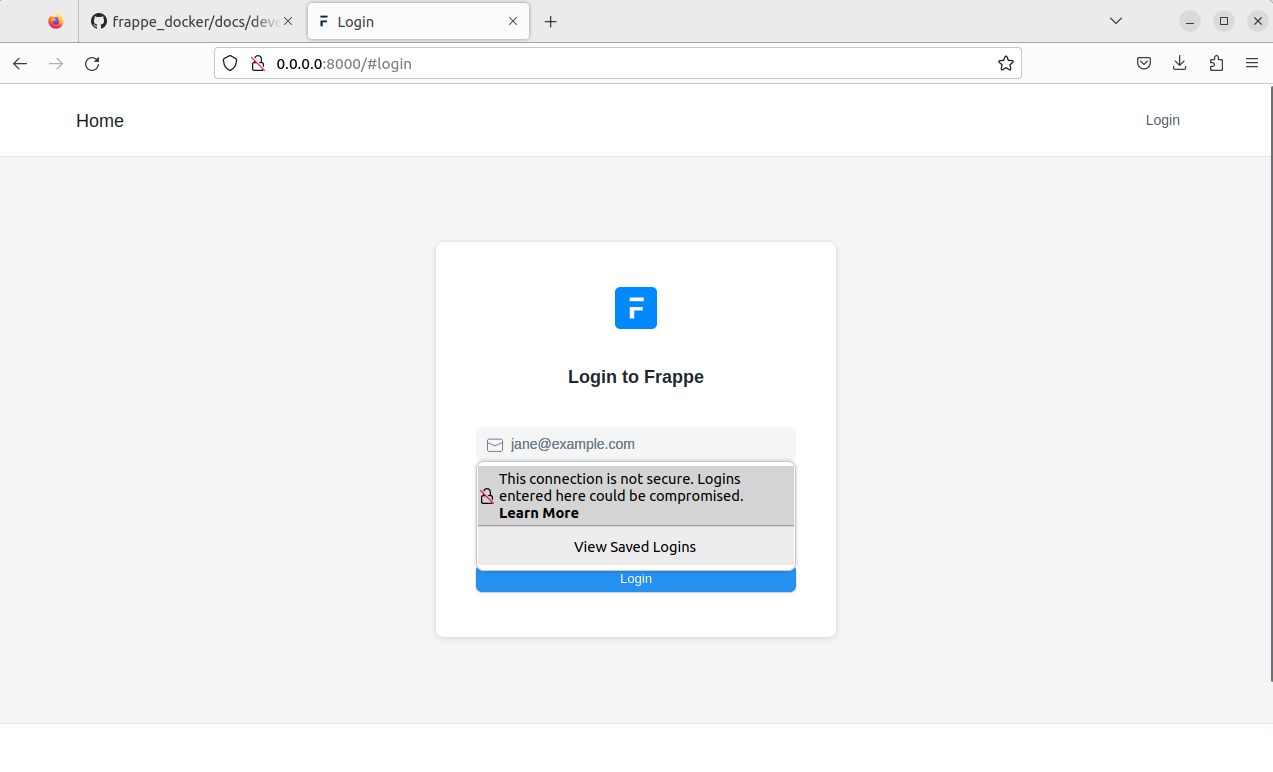
**Set bench developer mode on the new site**

To develop a new app, the last step will be setting the site into developer mode. Documentation is available at this link.

$bench --site mysite.localhost set-config developer\_mode 1

$bench --site mysite.localhost clear-cache

$bench start



**User-id :** administrator

**Password** : mahadik

