

# Facultad de Estudios Superiores Aragón

Ingeniería en Computación.

Organización y Administración de  
Centros de Computo  
Grupo 2809 (2022-II)

Profesor: AARON VELASCO AGUSTIN

Alumnos: Badillo Mendoza Jaime Yair

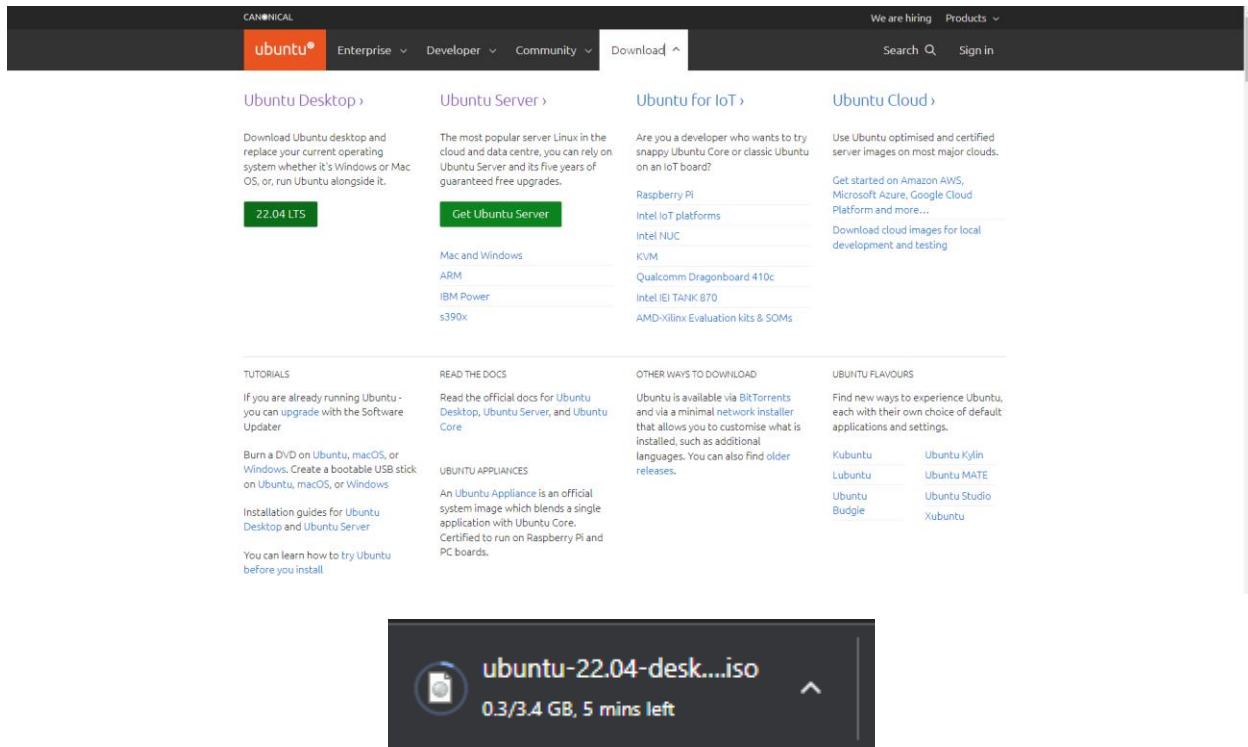
Martínez Bautista Luis Ángel

Rosales Lázaro José Eduardo

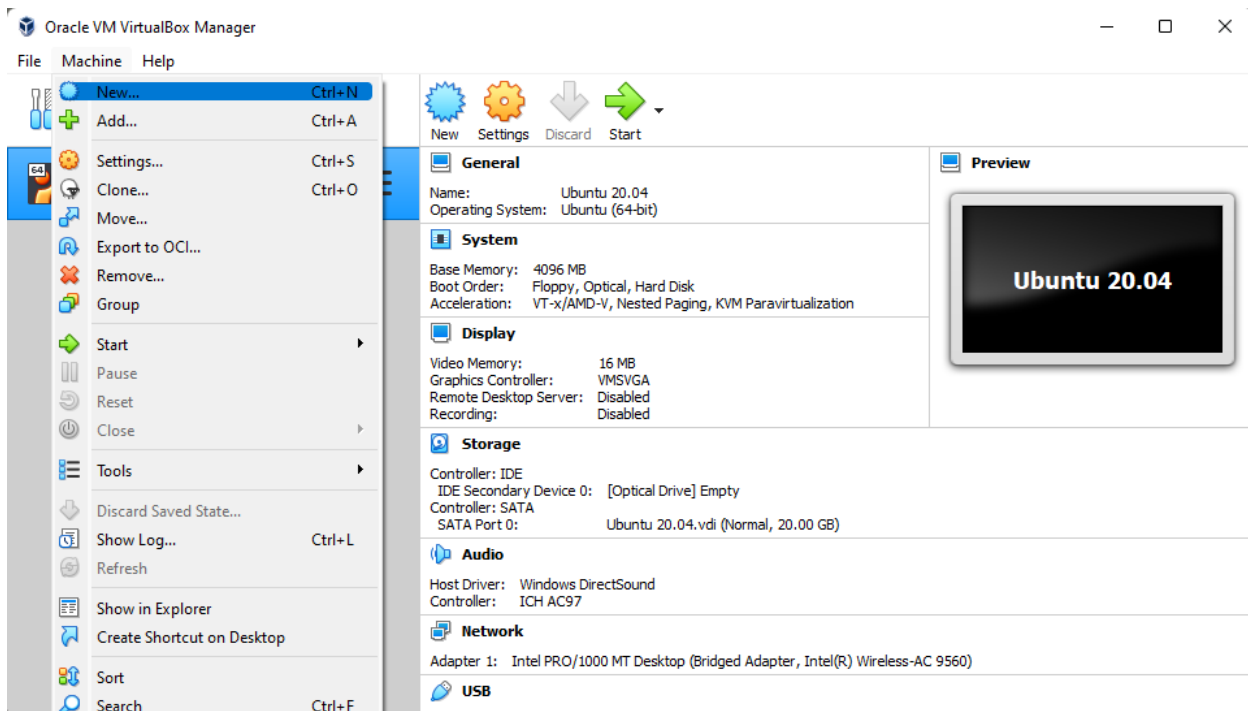
Documentación del Proyecto

## Descargando Ubuntu 22.04 para el servidor local

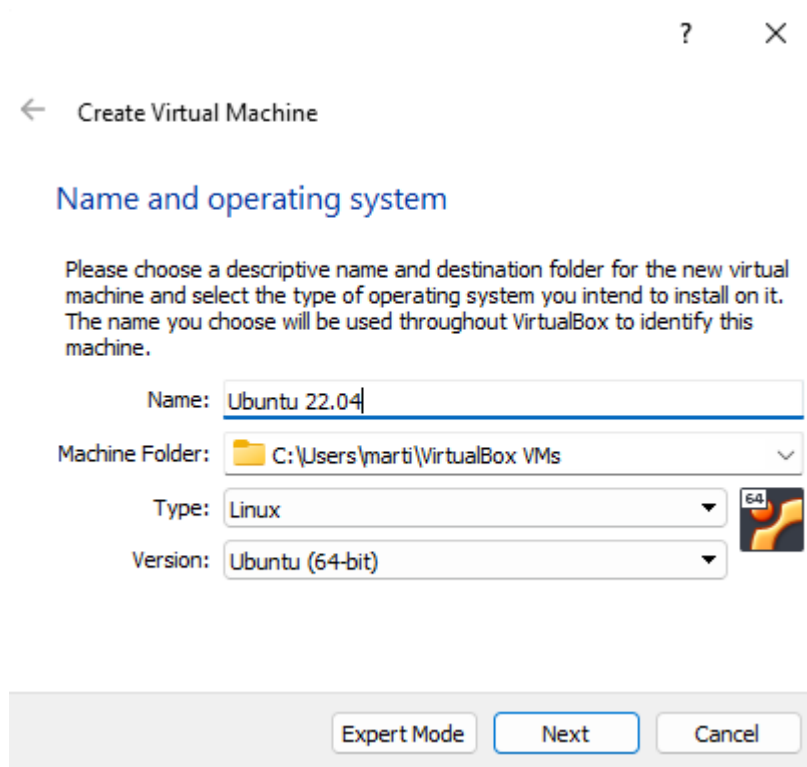
Utilizamos la versión de escritorio ya que para un servidor tan sencillo no hay mucha ventaja en utilizar la versión server y el ambiente gráfico ayuda para manipular archivos



## Se crea una nueva máquina virtual en VirtualBox



Se le pone un nombre y sistema operativo que se emulará



?

×

← Create Virtual Machine

### Name and operating system

Please choose a descriptive name and destination folder for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name:

Machine Folder:

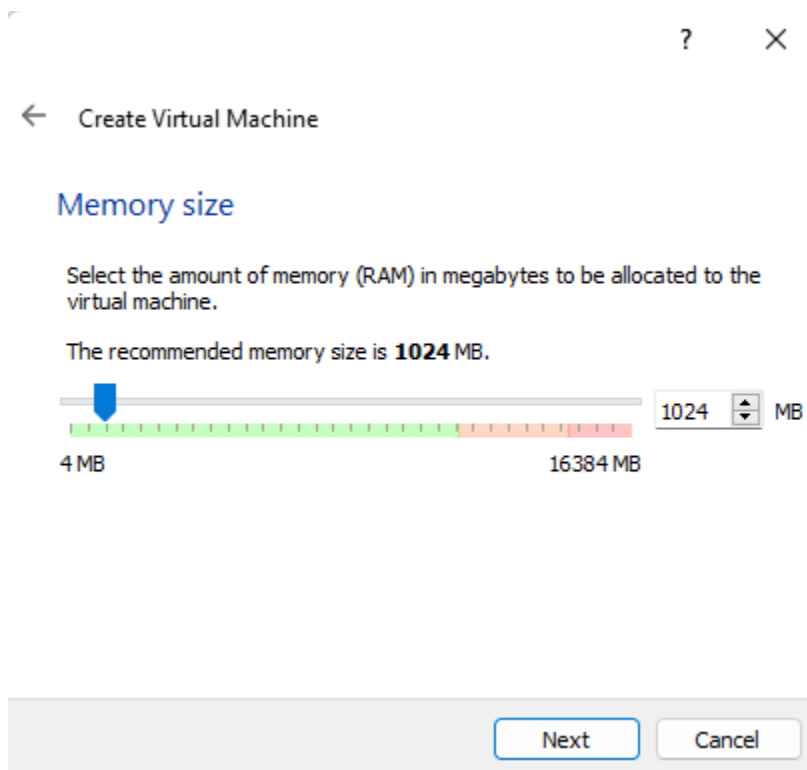
Type:

Version:

64

Expert Mode Next Cancel

Se le da 1GB de memoria RAM



?

×

← Create Virtual Machine

### Memory size

Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

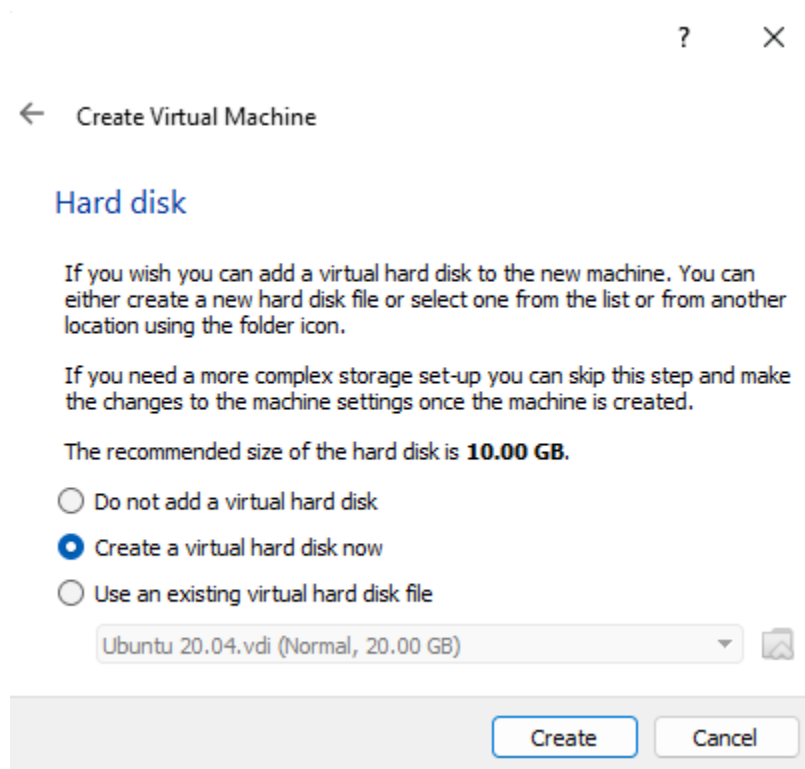
The recommended memory size is **1024 MB**.

4 MB 16384 MB

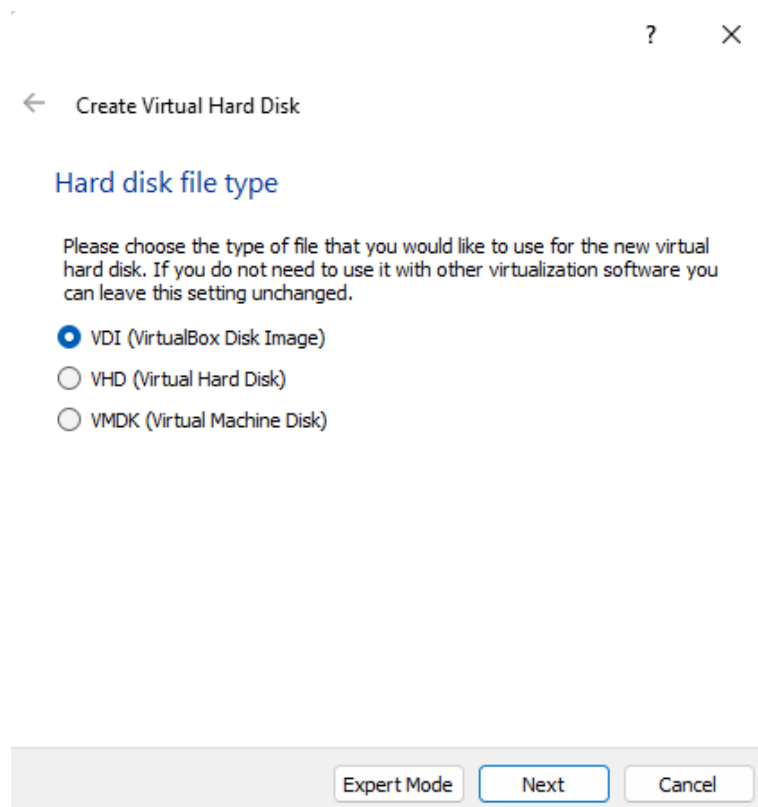
1024 MB

Next Cancel

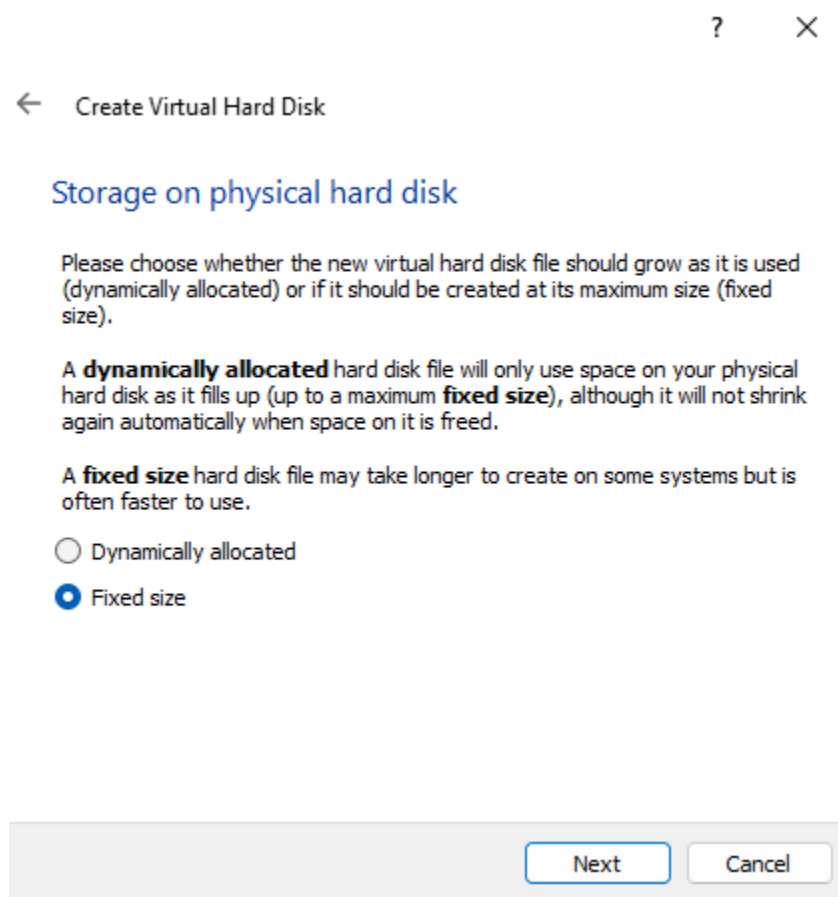
Se crea un disco duro virtual



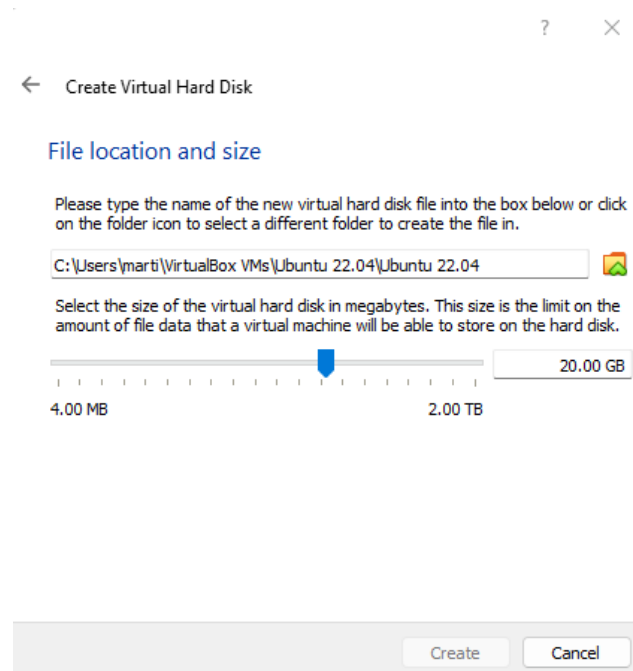
Se elige la opción por defecto



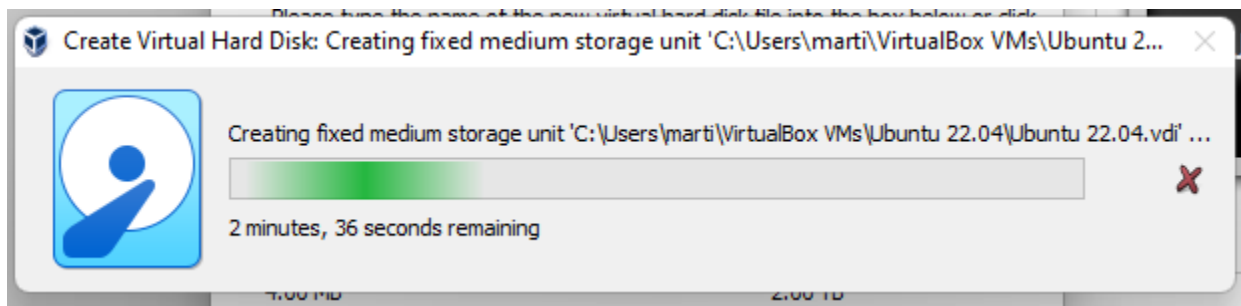
Se elige un tamaño de disco fijo



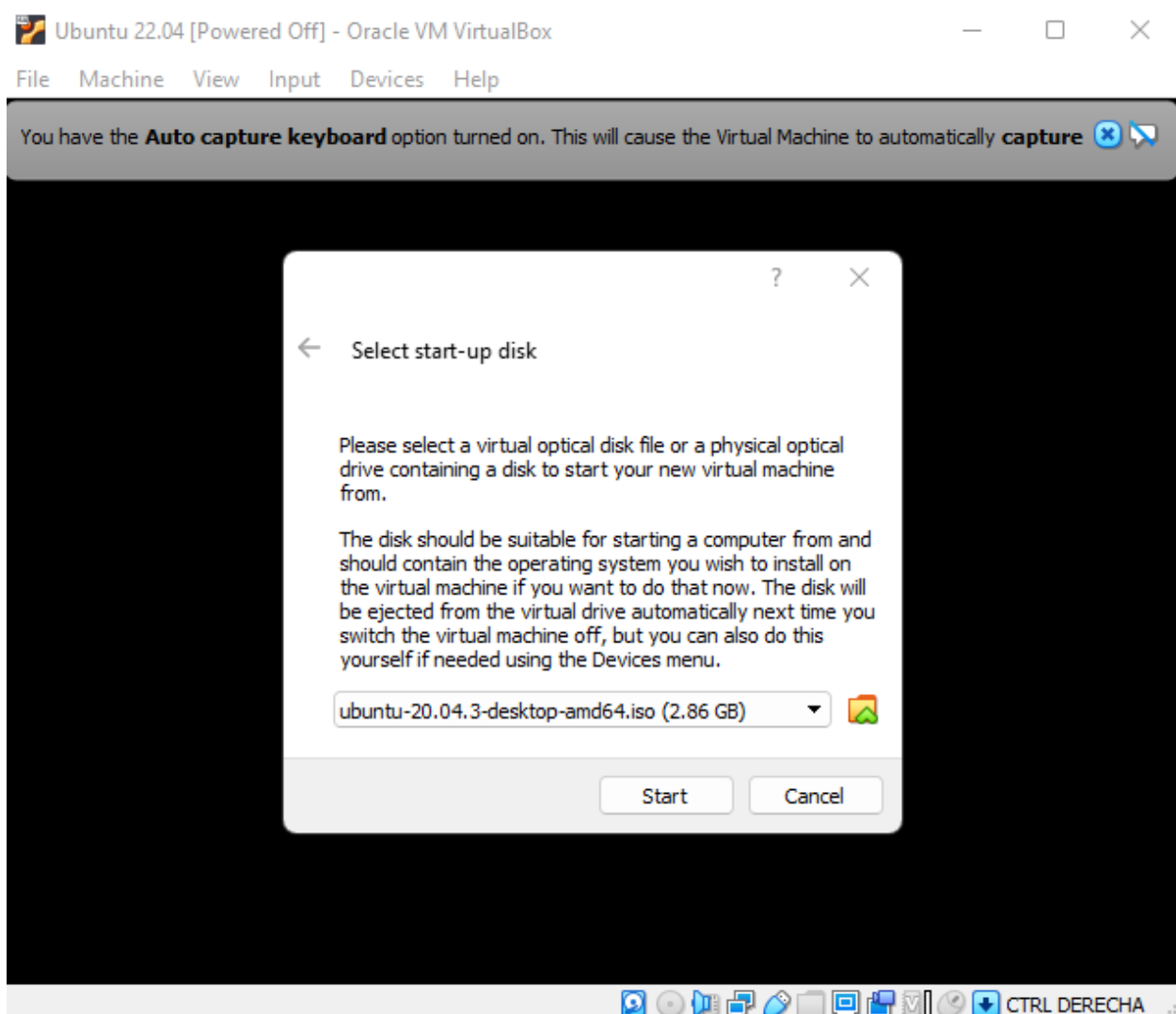
Se le asignan 20GB a la máquina virtual

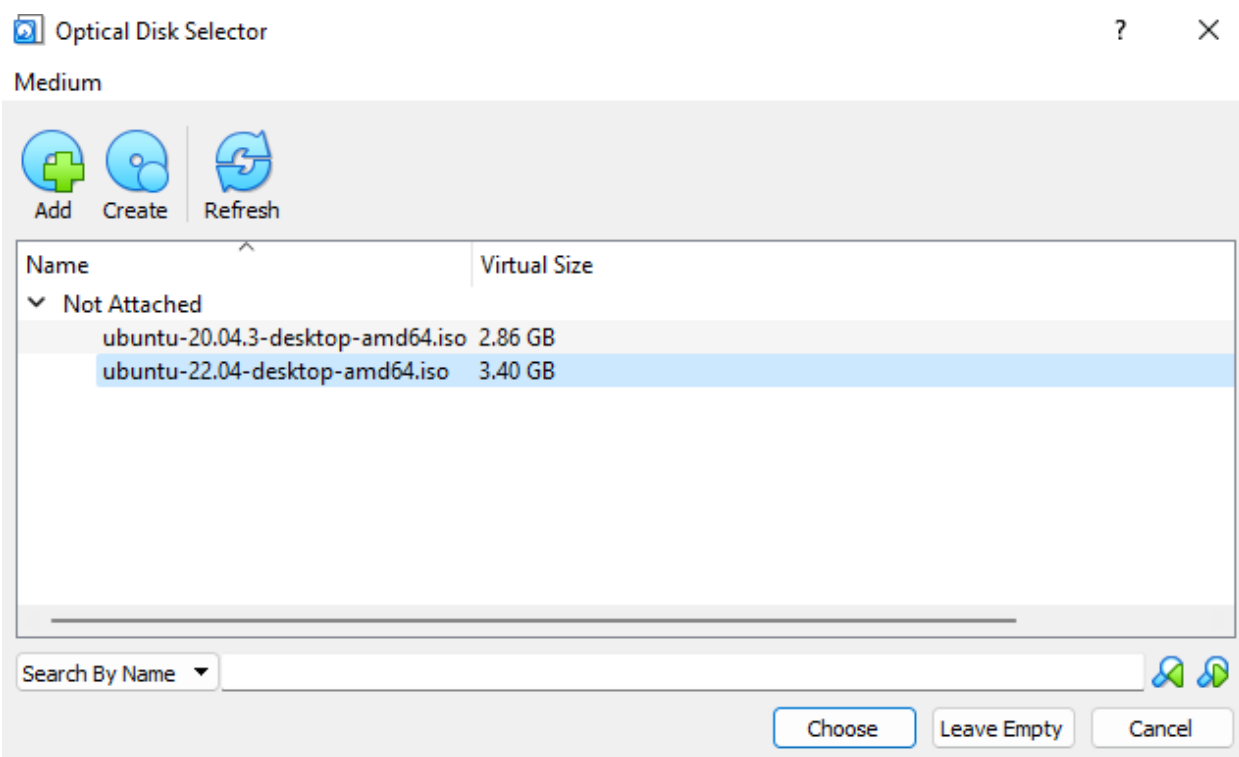
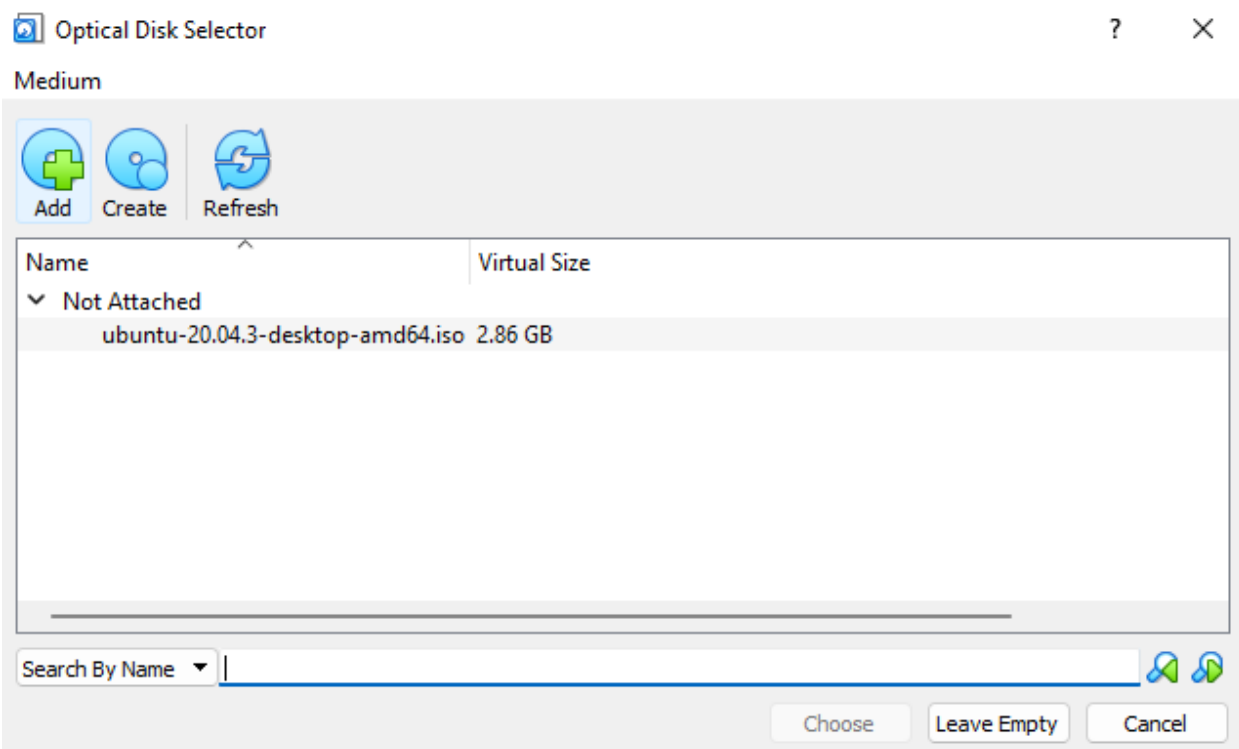


Se espera a que se cree el disco duro virtual

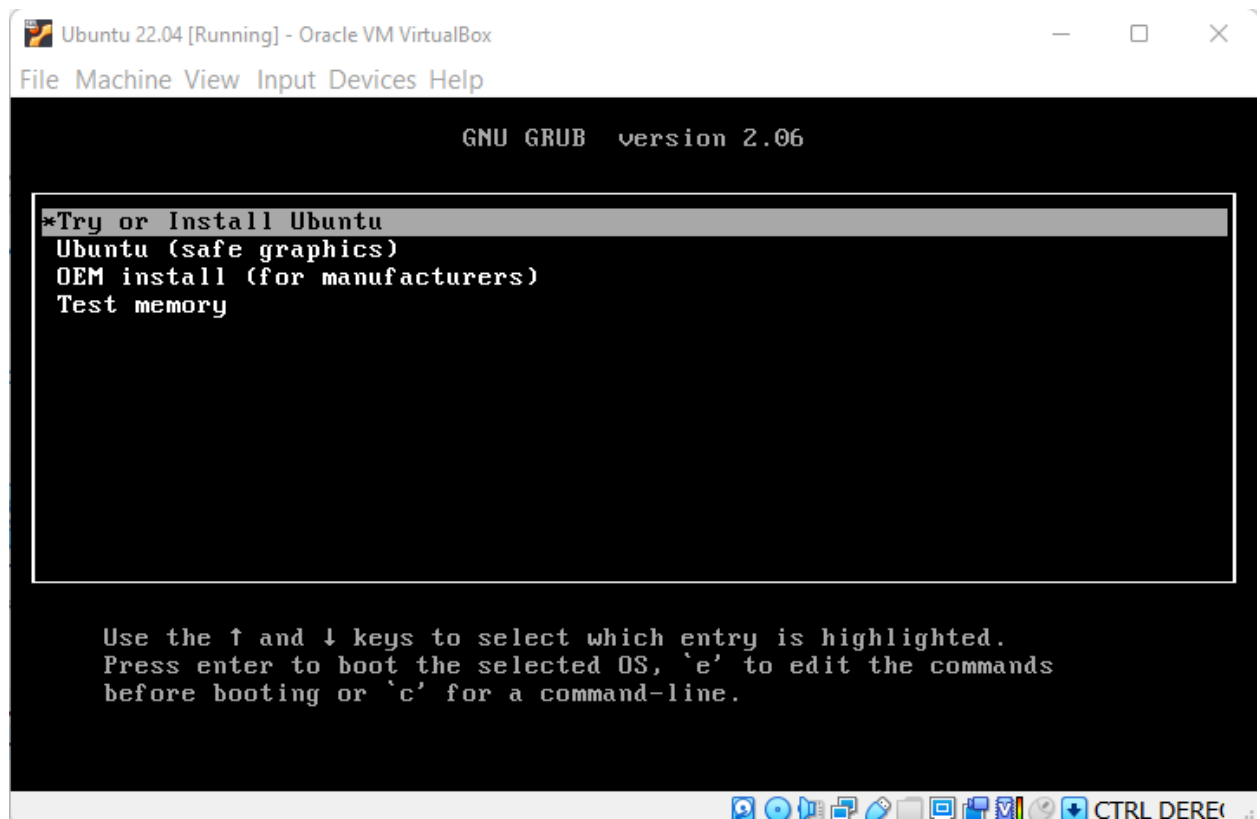


Al iniciarse la máquina virtual se debe de asignar la imagen para arrancar la misma





Se procede con la instalación del sistema operativo





Ubuntu 22.04 [Running] - Oracle VM VirtualBox

FileMachineViewInputDevicesHelp

Jun 9 18:13

Install

# Welcome

Asturianu

Bahasa Indonesia

Bosanski

Català

Čeština

Cymraeg

Dansk

Deutsch

Eesti

English

Español

Esperanto

Euskara


Français

Gaeilge


Galego

Hrvatski

Íslenska



Try Ubuntu




Install Ubuntu

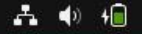
You can try Ubuntu without making any changes to your computer, directly from this CD.

Or if you're ready, you can install Ubuntu alongside (or instead of) your current operating system. This shouldn't take too long.

You may wish to read the [release notes](#).

 CTRL DERECHA

Jun 9 13:19



## Install

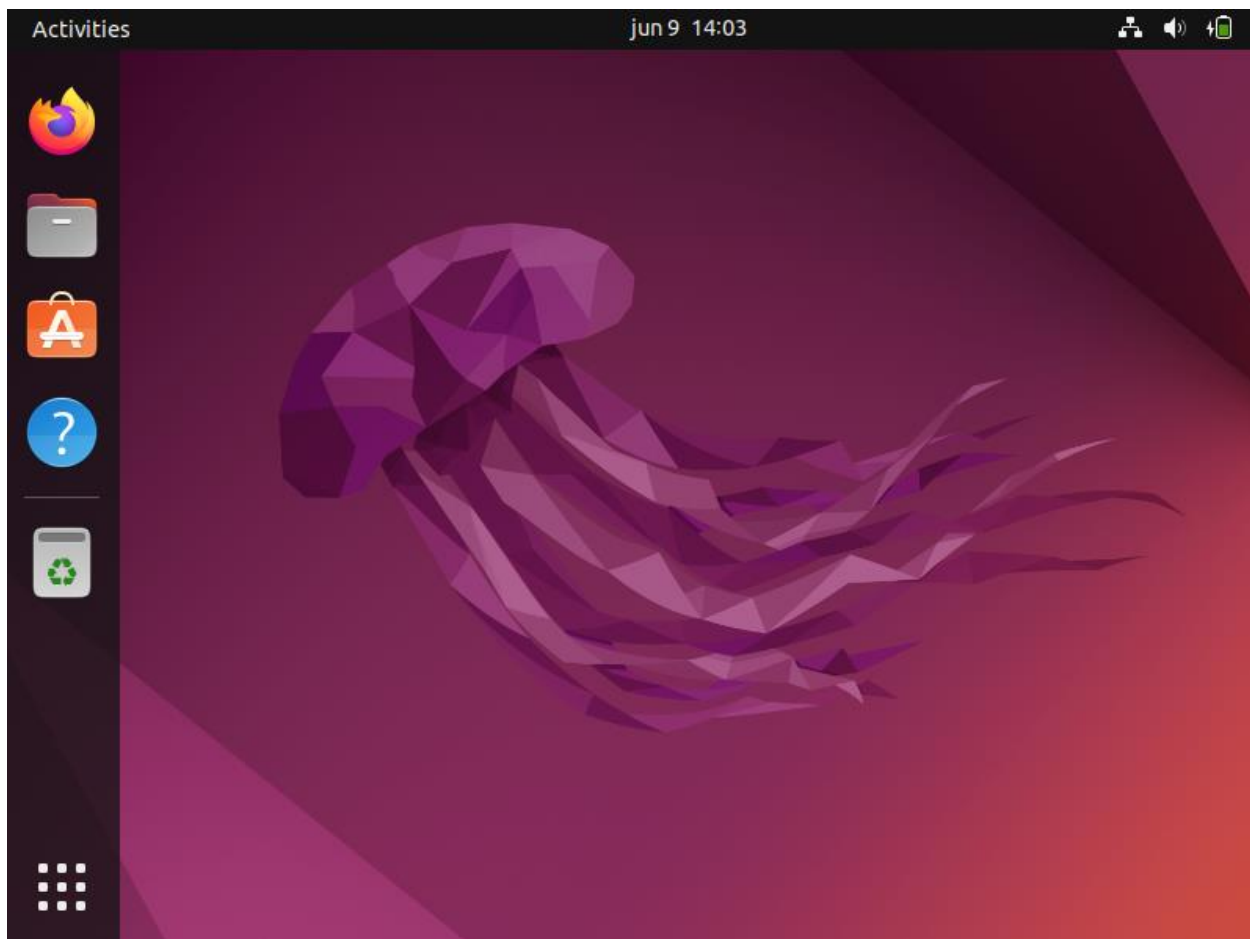
# Welcome to Ubuntu

Fast and full of new features, the latest version of Ubuntu makes computing easier than ever. Here are just a few cool new things to look out for...



> Copying files...

Skip



Una vez iniciado el sistema operativo se debe de ir a configuración de red, una vez ahí se introducen los siguientes datos para conseguir una IP estática dentro de la red

Cancel **Wired** Apply

Details Identity **IPv4** IPv6 Security

**IPv4 Method**

- ☐ Automatic (DHCP)
- ☒ Manual
- ☐ Shared to other computers
- ☐ Link-Local Only
- ☐ Disable

**Addresses**

Address	Netmask	Gateway	
192.168.1.211	/24	192.168.1.254	

**DNS** Automatic ☒

8.8.8.8

Separate IP addresses with commas

El server es detectado en el panel de configuración del modem

RE200	5G	06:21:92:0d:e d:79	192.168.1.211	Wifi	Active	2day 19 h 57min 17s	DHCP	0day 2 3h 21 min 55 s	0day 1
-------	----	-----------------------	---------------	------	--------	---------------------------	------	--------------------------------	--------

El server puede ser alcanzado por otras computadoras dentro de la red local

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.22000.675]
(c) Microsoft Corporation. All rights reserved.

C:\Users\marti>ping 192.168.1.211

Pinging 192.168.1.211 with 32 bytes of data:
Reply from 192.168.1.104: Destination host unreachable.
Reply from 192.168.1.104: Destination host unreachable.
Reply from 192.168.1.104: Destination host unreachable.
Reply from 192.168.1.104: Destination host unreachable.

Ping statistics for 192.168.1.211:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

C:\Users\marti>ping 192.168.1.211

Pinging 192.168.1.211 with 32 bytes of data:
Reply from 192.168.1.211: bytes=32 time=6ms TTL=63
Reply from 192.168.1.211: bytes=32 time=4ms TTL=63
Reply from 192.168.1.211: bytes=32 time=5ms TTL=63
Reply from 192.168.1.211: bytes=32 time=4ms TTL=63

Ping statistics for 192.168.1.211:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 4ms, Maximum = 6ms, Average = 4ms

C:\Users\marti>
```

Para instalar apache se abre la terminal y se accede a la misma como root

```
tgc@tgc-VirtualBox: ~
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

tgc@tgc-VirtualBox:~$ sudo -i
```

Ejecutar el comando “apt install apache2”

```
root@tgc-VirtualBox: ~  
root@tgc-VirtualBox:~# apt install apache2  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1  
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.3-0  
Suggested packages:  
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser  
The following NEW packages will be installed:  
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1  
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.3-0  
0 upgraded, 9 newly installed, 0 to remove and 143 not upgraded.  
Need to get 2 055 kB of archives.  
After this operation, 8 196 kB of additional disk space will be used.  
Do you want to continue? [Y/n] y
```

Para verificar que el servicio se haya instalado y esté funcionando ejecutar el comando “systemctl status apache2”

```
root@tgc-VirtualBox: ~  
root@tgc-VirtualBox:~# systemctl status apache2  
● apache2.service - The Apache HTTP Server  
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese  
   Active: active (running) since Thu 2022-06-09 14:17:00 CDT; 26s ago  
     Docs: https://httpd.apache.org/docs/2.4/  
   Main PID: 4898 (apache2)  
     Tasks: 55 (limit: 1084)  
    Memory: 7.3M  
       CPU: 47ms  
    CGroup: /system.slice/apache2.service  
            └─4898 /usr/sbin/apache2 -k start  
              └─4900 /usr/sbin/apache2 -k start  
                └─4901 /usr/sbin/apache2 -k start  
  
jun 09 14:16:59 tgc-VirtualBox systemd[1]: Starting The Apache HTTP Server...  
jun 09 14:17:00 tgc-VirtualBox apachectl[4896]: AH00558: apache2: Could not rel  
jun 09 14:17:00 tgc-VirtualBox systemd[1]: Started The Apache HTTP Server.  
lines 1-16/16 (END)
```

Configurar el archivo 000-default.conf para crear la página http

```
root@tgc-VirtualBox:~# cd /etc/apache2/sites-available/  
root@tgc-VirtualBox:/etc/apache2/sites-available# ls  
000-default.conf  default-ssl.conf  
root@tgc-VirtualBox:/etc/apache2/sites-available# nano 000-default.conf
```

El archivo debe de configurarse así, siendo ServerName y ServerAlias reemplazables por otros nombres.

```
GNU nano 6.2                                000-default.conf  
<VirtualHost *:80>  
    # The ServerName directive sets the request scheme, hostname and port that  
    # the server uses to identify itself. This is used when creating  
    # redirection URLs. In the context of virtual hosts, the ServerName  
    # specifies what hostname must appear in the request's Host: header to  
    # match this virtual host. For the default virtual host (this file) this  
    # value is not decisive as it is used as a last resort host regardless.  
    # However, you must set it for any further virtual host explicitly.  
    #ServerName www.example.com  
  
    ServerAdmin webmaster@localhost  
    ServerName www.tgc.com  
    ServerAlias www.tgc.com  
    DocumentRoot /var/www/html  
  
    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,  
    # error, crit, alert, emerg.  
    # It is also possible to configure the loglevel for particular  
    # modules, e.g.  
    #LogLevel info ssl:warn  
  
    ErrorLog ${APACHE_LOG_DIR}/error.log  
    CustomLog ${APACHE_LOG_DIR}/access.log combined  
  
    # For most configuration files from conf-available/, which are  
    # enabled or disabled at a global level, it is possible to  
    # include a line for only one particular virtual host. For example the  
    # following line enables the CGI configuration for this host only  
    # after it has been globally disabled with "a2disconf".  
    #Include conf-available/serve-cgi-bin.conf  
</VirtualHost>
```

El archivo index.html es el contenedor de la página de apache, este se debe de configurar para cambiar la página por defecto

```
root@tgc-VirtualBox:~# cd /var/www/html/
root@tgc-VirtualBox:/var/www/html# ls
index.html
root@tgc-VirtualBox:/var/www/html# nano index.html
```

Abrir el puerto 80 para permitir la salida del servicio de apache

```
root@tgc-VirtualBox:/var/www/html# ufw enable
Firewall is active and enabled on system startup
root@tgc-VirtualBox:/var/www/html# ufw allow 80
Skipping adding existing rule
Skipping adding existing rule (v6)
root@tgc-VirtualBox:/var/www/html# ufw status
Status: active

To Action From
--
80 ALLOW Anywhere
80 (v6) ALLOW Anywhere (v6)

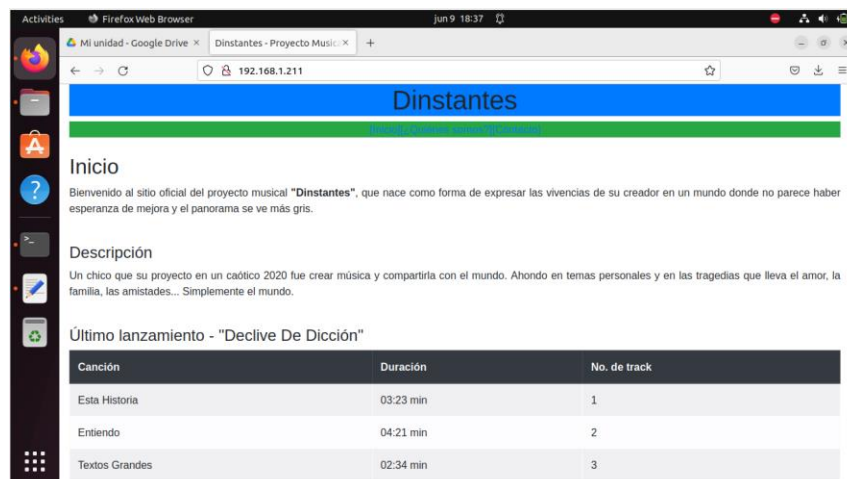
root@tgc-VirtualBox:/var/www/html#
```

Reiniciar el proceso de apache con el comando “systemctl restart apache2”

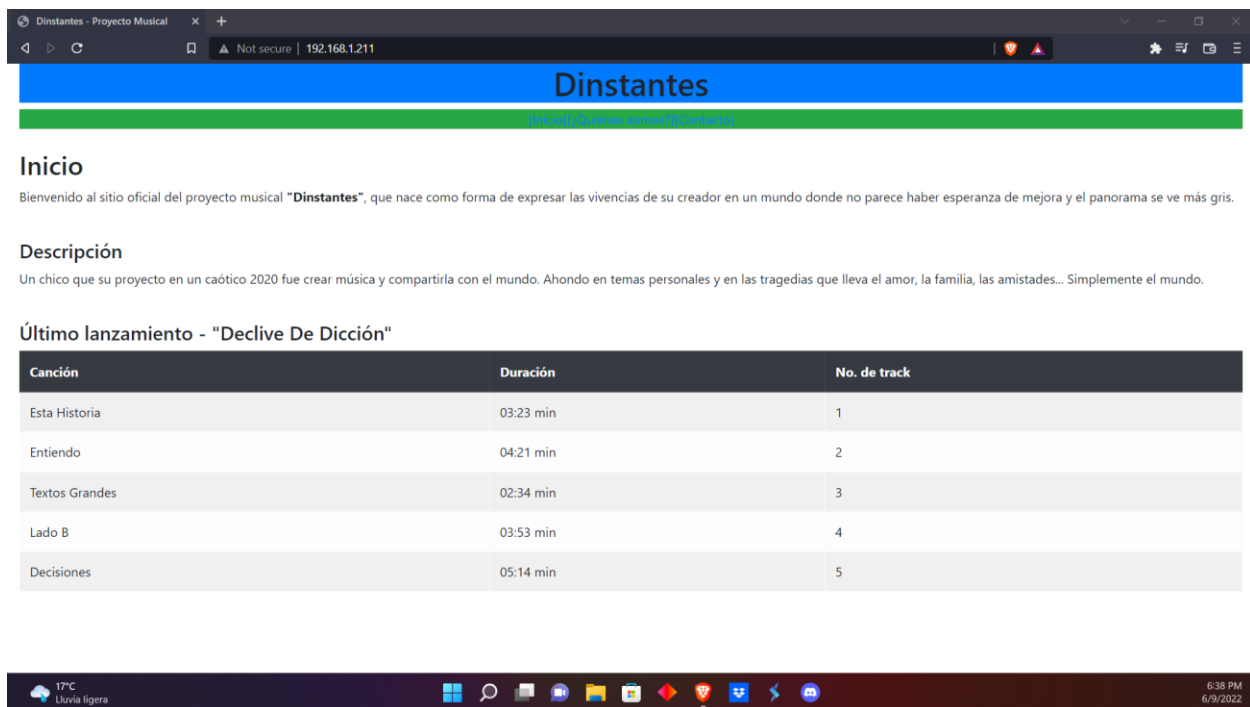
```
root@tgc-VirtualBox:/var/www/html# systemctl restart apache2
root@tgc-VirtualBox:/var/www/html# S
```

Finalmente se puede probar la página, en un navegador se debe de escribir la dirección IP de la página

Página vista desde la máquina virtual



## Página vista desde otra máquina dentro de la misma red local



Con respecto a las peticiones para la elaboración del proyecto se presenta a continuación la documentación del mismo, en este documento se vera el proceso de la implementación de una base de datos, así como el código y la sintaxis necesaria que se utilizó.

### Base de Dtos

Para comenzar a implementar nuestra base de datos nos dirigiremos a nuestro editor de código y anexamos el encabezado:

```
1 SET NAMES 'utf8';
2 DROP DATABASE IF EXISTS base;
3 CREATE DATABASE IF NOT EXISTS base DEFAULT CHARACTER SET utf8;
4 USE base;
5 CREATE TABLE clientes(
6     id_clientes          INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,
7     nombre_clientes      VARCHAR(25) NOT NULL,
8     apellido_clientes    VARCHAR(25) NOT NULL
9 )DEFAULT CHARACTER SET utf8;
```

Como podemos observar empezamos generando una declaración de todas las tablas, a su vez creamos la tabla y a su vez generamos las variables que se ocuparan a lo largo del código.



```
INSERT INTO clientes(nombre_clientes,apellido_clientes) VALUES('Cliente1','Apellido1');
INSERT INTO clientes(nombre_clientes,apellido_clientes) VALUES('Cliente2','Apellido2');
INSERT INTO clientes(nombre_clientes,apellido_clientes) VALUES('Cliente3','Apellido3');
```

Agregamos INSERT INTO para agregar registros a cualquier tabla individual de una base de datos relacional, en este caso agregamos solo 3 quienes serán apartados exclusivos para los clientes.

A continuacion creamos un apartado, en este caso un desencadenador el cual lo especificamos con clientes\_mayus, en el cual realizara un upper a los datos insertados en cliente, es decir, las letras de los datos introducidos seran mayusculas.

```
DELIMITER $$
create trigger clientes_mayus before insert on clientes for each row
begin
    set new.nombre_clientes=upper(new.nombre_clientes);
    set new.apellido_clientes=upper(new.apellido_clientes);
end $$
DELIMITER ;
```

Realizamos nuevamente el paso anterior pero ahora para después de actualizar los datos

```
DELIMITER $$
create trigger clientes_mayus_update before update on clientes for each row
begin
    set new.nombre_clientes=upper(new.nombre_clientes);
    set new.apellido_clientes=upper(new.apellido_clientes);
end $$
DELIMITER ;
```

Este apartado se encargará de descargar los clientes que se anexaron en la tabla de un principio.

Se genera nuevamente otro desencadenador en donde especificamos la importación de nuestros datos de la tabla. Al igual que se generara la inserción de clientes a la tabla

```
DELIMITER $$
CREATE DEFINER='root'@'localhost' PROCEDURE `insertarClientes`(IN nombre VARCHAR(25), IN apellido VARCHAR(25))
BEGIN
    INSERT INTO clientes(nombre_clientes, apellido_clientes) VALUES(nombre, apellido);
END $$
```

Posteriormente se crea otro desencadenador este mismo se utilizara para remover datos de la tabla, como clientes correspondiente de sus nombres y apellidos como se ha mostrado a lo largo de este proceso.

```
DELIMITER $$
CREATE PROCEDURE eliminarCliente(IN id INT)
BEGIN
    DELETE FROM clientes WHERE id_clientes = id;
END $$
```

Nuevamente se presenta este apartado donde se mostraran los datos de la tabla en este caso los clientes que se agregaron.

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `todosClientes`()
BEGIN
    SELECT * FROM todosClientes;
END $$
```

Con este apartado se realizará las modificaciones de los datos.

```
DELIMITER $$
CREATE PROCEDURE modificarClientes(IN id INT, IN nombre VARCHAR(25), IN apellido VARCHAR(25))
BEGIN
    UPDATE clientes SET nombre_clientes = nombre, apellido_clientes = apellido WHERE id_clientes = id;
END $$
```

Las ultimas líneas de nuestro código nos ayudaran a la funcionalidad de nuestra base de datos

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `base`.`todosclientes` AS
    SELECT
        `base`.`clientes`.`id_clientes` AS `id_clientes`,
        `base`.`clientes`.`nombre_clientes` AS `nombre_clientes`,
        `base`.`clientes`.`apellido_clientes` AS `apellido_clientes`
    FROM
        `base`.`clientes`;
```

Una vez concluido la primera parte de la implementacion de nuestra tabla y nuestra base de datos a costninuacion se mostrara la siguiente parte del codigo que sera fundamental para el correcto funcionamiento de nuestra base de datos

Empezaremos creando un nuevo archivo .java, el cual sera el controlador del cliente, se utilizara e importara las siguientes librerias:

```
1  package fes.aragon.controlador;
2
3  import java.io.IOException;
4  import java.net.URL;
5  import java.util.ResourceBundle;
6
7  import de.jensd.fx.glyphs.fontawesome.FontAwesomeIcon;
8  import de.jensd.fx.glyphs.fontawesome.FontAwesomeIconView;
9  import fes.aragon.modelo.Clientes;
10 import fes.aragon.mysql.Conexion;
11 import javafx.fxml.FXML;
12 import javafx.fxml.FXMLLoader;
13 import javafx.fxml.Initializable;
14 import javafx.geometry.Insets;|
15 import javafx.scene.Parent;
16 import javafx.scene.Scene;
17 import javafx.scene.control.Alert;
18 import javafx.scene.control.TableCell;
19 import javafx.scene.control.TableColumn;
20 import javafx.scene.control.TableView;
21 import javafx.scene.control.cell.PropertyValueFactory;
22 import javafx.scene.input.MouseEvent;
23 import javafx.scene.layout.HBox;
24 import javafx.stage.Stage;
```

Y comenzaremos a crear los eventos: en este caso el de nuevoCliente

```
void nuevoCliente(MouseEvent event) {
    try {
        Parent parent = FXMLLoader.load(getClass().getResource("/fes/aragon/vista/NuevoUsuario.fxml"));
        Scene escena = new Scene(parent);
        Stage escenario = new Stage();
        escenario.setScene(escena);
        escenario.initStyle(StageStyle.UTILITY);
        escenario.show();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

Uno más para refrescar:

```

@FXML
void refrescar(MouseEvent event) {
    this.traerDatos();
}

```

Posteriormente se inicializa el apartado donde se otorga la petición de los datos correspondientes, en este caso id, nombre y apellido son los parámetros que se solicitan y se enlazan a la tabla.

```

@Override
public void initialize(URL arg0, ResourceBundle arg1) {
    // TODO Auto-generated method stub
    this.clienteID.setCellValueFactory(new PropertyValueFactory<>("id"));
    this.clienteNombre.setCellValueFactory(new PropertyValueFactory<>("nombre"));
    this.clienteApellidoPaterno.setCellValueFactory(new PropertyValueFactory<>("apellidoPaterno"));

    Callback<TableColumn<Clientes, String>, TableCell<Clientes, String>>
    celda = (TableColumn<Clientes, String> parametros)-> {
        final TableCell<Clientes, String> cel = new TableCell<Clientes, String>(){

```

Y a continuación se implementan los otros eventos que utilizaremos posteriormente:

```

@Override
protected void updateItem(String arg0, boolean arg1) {
    // TODO Auto-generated method stub
    super.updateItem(arg0, arg1);
    if(arg1) {
        setGraphic(null);
        setText(null);
    } else {
        FontAwesomeIconView borrarIcono = new FontAwesomeIconView(FontAwesomeIcon.TRASH);
        FontAwesomeIconView modificarIcono = new FontAwesomeIconView(FontAwesomeIcon.PENCIL_SQUARE);
        borrarIcono.setGlyphStyle("-fx-fill:RED;-fx-size: 18px;-fx-cursor: hand;");
        modificarIcono.setGlyphStyle("-fx-cursor:hand;" + "-fx-size:18px;" + "-fx-fill:#0a1ce8;");
        borrarIcono.setOnMouseClicked((MouseEvent evento)-> {
            Clientes cliente = tblTablaCliente.getSelectionModel().getSelectedItem();
            borrarCliente(cliente.getId());
        });
        modificarIcono.setOnMouseClicked((MouseEvent evento)-> {
            Clientes cliente = tblTablaCliente.getSelectionModel().getSelectedItem();
            modificarCliente(cliente);
        });
        HBox hbox = new HBox(borrarIcono, modificarIcono);
        hbox.setStyle("-fx-alignment:center");
        HBox.setMargin(borrarIcono, new Insets(2, 2, 0, 3));
        HBox.setMargin(modificarIcono, new Insets(2, 3, 0, 2));
        setGraphic(hbox);
        setText(null);
    }
}

};
return cel;

```

Implementamos el apartado en donde se van a traerDatos:

```
private void traerDatos() {
    try {
        Conexion cnn = new Conexion();
        this.tblTablaCliente.getItems().clear();
        this.tblTablaCliente.setItems(cnn.todosClientes());
    } catch (Exception e) {
        // TODO Auto-generated catch block
        Alert alerta = new Alert(Alert.AlertType.WARNING);
        alerta.setTitle("Problema en B.D");
        alerta.setHeaderText("Error en la aplicacion");
        alerta.setContentText("Consulta al fabricante, por favor");
        alerta.showAndWait();
        e.printStackTrace();
    }
}
```

Otro específico para borrarCliente:

```
private void borrarCliente(int id) {
    try {
        Conexion cnn = new Conexion();
        cnn.eliminarClientes(id);
        this.traerDatos();
    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
        Alert alerta = new Alert(Alert.AlertType.WARNING);
        alerta.setTitle("Problema en B.D");
        alerta.setHeaderText("Error en la aplicacion");
        alerta.setContentText("Consulta al fabricante, por favor");
        alerta.showAndWait();
    }
}
```

Y el ultimo para modificarCliente:

```
private void modificarCliente(Clientes cliente) {
    try {
        FXMLLoader alta = new FXMLLoader(getClass().getResource("/fes/aragon/vista/NuevoUsuario.fxml"));
        Parent parent = (Parent)alta.load();
        ((NuevoClienteController)alta.getController()).modificarCliente(cliente);
        Scene escena = new Scene(parent);
        Stage escenario = new Stage();
        escenario.setScene(escena);
        escenario.initStyle(StageStyle.UTILITY);
        escenario.show();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

Una vez finalizado el apartado nos dirigiremos a crear otro archivo .java este será en particular para los nuevos clientes, por lo tanto comenzaremos importando los paquetes con sus librerías:

```
package fes.aragon.controlador;

import fes.aragon.modelo.Clientes;
import fes.aragon.mysql.Conexion;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.TextField;
```

Creamos Public Class que será el nuevo cliente controlador a su vez se creara un evento y otros mas que se verán a continuación:

```
public class NuevoClienteController {

    @FXML
    private TextField txtApellidoPaterno;

    @FXML
    private TextField txtNombre;

    @FXML
    void accionLimpiar(ActionEvent event) {
        this.limpiar();
    }
    private Clientes cliente = null;

    @FXML
    void guardarAccion(ActionEvent event) {
        if(cliente == null) {
            cliente = new Clientes();
        }
    }
}
```

Se creara una decisión que lo que hará es validar los datos de las peticiones correspondientes:

```
if(validar()) {
    Alert alerta = new Alert(Alert.AlertType.INFORMATION);
    if(cliente.getId() == null) {
        almacenar();
        alerta.setContentText("Se ha almacenado el cliente!");
        limpiar();
    } else {
        modificar();
        alerta.setContentText("Se ha modificado el cliente!");
    }
    alerta.setHeaderText(null);
    alerta.showAndWait();
}
```

Posteriormente el otro apartado que almacenara los datos:

```
private void almacenar() {  
    try {  
        Conexion cnn = new Conexion();  
        cliente.setNombre(txtNombre.getText());  
        cliente.setApellidoPaterno(txtApellidoPaterno.getText());  
        cnn.almacenarClientes(cliente);  
    } catch (Exception e) {  
        // TODO Auto-generated catch block  
        Alert alerta = new Alert(Alert.AlertType.WARNING);  
        alerta.setHeaderText(null);  
        alerta.setContentText("Lo sentimos, se ha presentado un problema");  
        alerta.showAndWait();  
        e.printStackTrace();  
    }  
}
```

De la misma manera otro apartado para modificar, en este caso modificar los clientes que se nos estan otorgando.

```
private void modificar() {  
    try {  
        Conexion cnn = new Conexion();  
        cliente.setNombre(txtNombre.getText());  
        cliente.setApellidoPaterno(txtApellidoPaterno.getText());  
        cnn.modificarClientes(cliente);  
    } catch (Exception e) {  
        // TODO Auto-generated catch block  
        Alert alerta = new Alert(Alert.AlertType.WARNING);  
        alerta.setHeaderText(null);  
        alerta.setContentText("Lo sentimos, se ha presentado un problema");  
        alerta.showAndWait();  
        e.printStackTrace();  
    }  
}
```

Y por ultimo se crean los metodos que son en particular para los apartados que acabamos de crear que nos ayudaran en su funcionalidad.

```
private void limpiar() {
    this.txtNombre.setText("");
    this.txtApellidoPaterno.setText("");
}
private boolean validar() {
    boolean validos = true;
    if(this.txtNombre.getText().isEmpty() || this.txtNombre.getText().regionMatches(0, " ", 0, 1)) {
        validos = false;
    }
    if(this.txtApellidoPaterno.getText().isEmpty() || this.txtApellidoPaterno.getText().regionMatches(0, " ", 0, 1)) {
        validos = false;
    }
    return validos;
}
public void modificarCliente(Clientes cliente) {
    this.cliente = cliente;
    this.txtNombre.setText(cliente.getNombre());
    this.txtApellidoPaterno.setText(cliente.getApellidoPaterno());
}
```

Una vez finalizado vamos a crear otros 2 archivos `ams.java` los cuales seran la vista del controlador y el contenido que son mas apartados visuales que se puedan mostrar a la hora de interactuar con el programa:

```
package fes.aragon.controlador;

import javafx.fxml.FXML;
import javafx.scene.layout.BorderPane;
import javafx.event.ActionEvent;

public class VistaController {

    @FXML
    private BorderPane idPrincipal;

    // Event Listener on Button.onAction
    @FXML
    public void accionCliente(ActionEvent event) {
        Contenido contenido = new Contenido("/fes/aragon/vista/Cliente.fxml");
        idPrincipal.setCenter(contenido);
    }
}
```

Este seria el de la vista del controlador el siguiente trata en particular del contenido:



```
image.pngpackage fes.aragon.controlador;

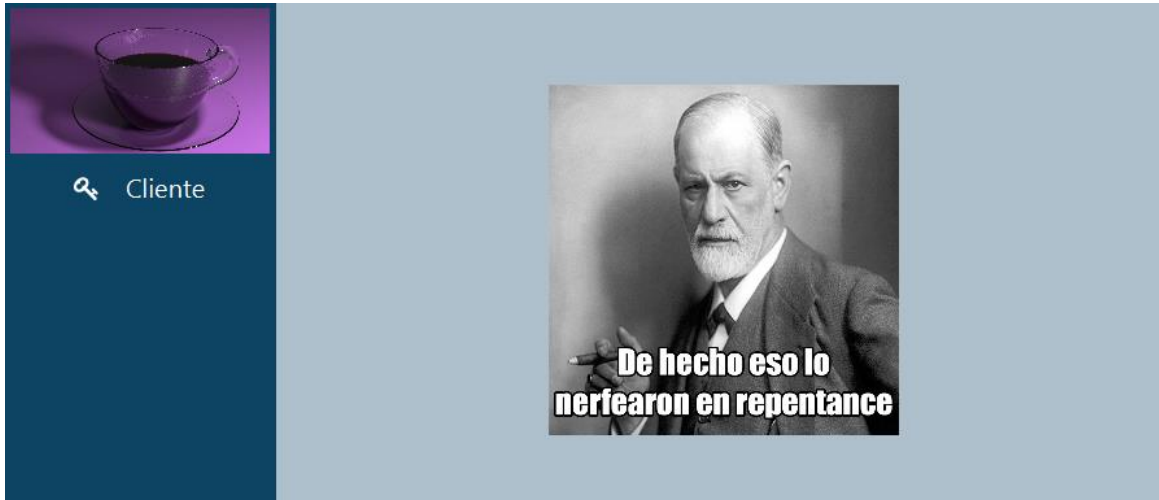
import java.io.IOException;

import javafx.fxml.FXMLLoader;
import javafx.scene.layout.Pane;

public class Contenido extends Pane{
    public Contenido(String ruta) {
        FXMLLoader fxml = new FXMLLoader(getClass().getResource(ruta));
        fxml.setRoot(this);
        try {
            fxml.load();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

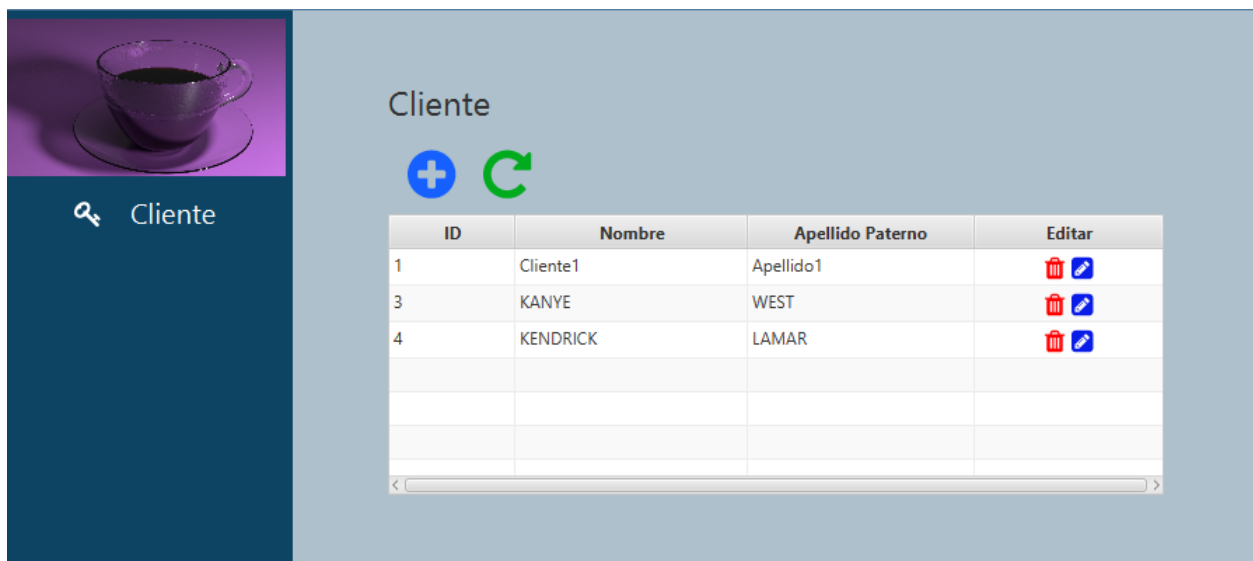
## CRUD

Teniendo los elementos de la base de datos, procederemos a mostrar las acciones que realiza. Al iniciar el proyecto se visualizar una pestaña, la cual daremos clic, en el botón de nombre cliente, el cual nos abrirá otra



### Create(Crear)

Dentro de esta pestaña, arriba de la tabla del lado izquierdo, abra dos iconos, el primero, de forma de un más, nos brinda la opción de poder crear un numero usuario



Se nos abrirá una nueva venta, la cual será para agregar al usuario, solo daremos al botón de guardar



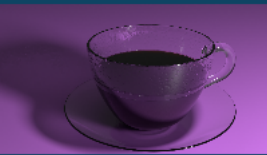
## Nuevo Cliente

Nombre:

Apellido Paterno:



Read(Leer)







En la tabla, se nos visualizara los usuario o clientes que hay, pero el siguiente icono, que esta en la parte de arriba, nos actualizara los usuarios, si es que hubo un cambio o actualización de datos



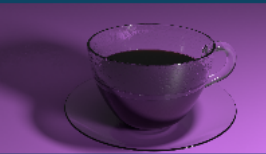
🔍 Cliente


### Cliente





ID	Nombre	Apellido Paterno	Editar
1	Cliente1	Apellido1	 
3	KANYE	WEST	 
4	KENDRICK	LAMAR	 














Ciente

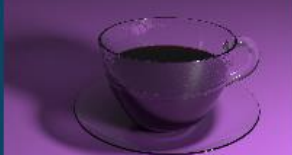
### Ciente





ID	Nombre	Apellido Paterno	Editar
1	Ciente1	Apellido1	 
3	KANYE	WEST	 
4	KENDRICK	LAMAR	 
6	ASAP	ROCKY	 



Update(Actualizar)









Dentro de la ultima columna, de cada cliente, habrá dos iconos, del cual el que tiene forma de lapicero, será para actualizar o hacer cambios a los datos ingresados

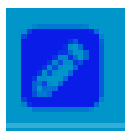



Ciente

### Ciente

ID	Nombre	Apellido Paterno	Editar
1	Ciente1	Apellido1	 
3	KANYE	WEST	 
4	KENDRICK	LAMAR	 
6	ASAP	ROCKY	 



De igual forma al dar clic se nos presentara una nuevamente una ventana, en la cual se ingresar los datos, que deseamos actualizar

## Nuevo Cliente

Nombre:

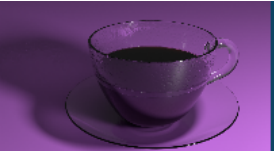
Apellido Paterno:

## Nuevo Cliente

Nombre:



Apellido Paterno:









Actualizamos



🔍 Cliente

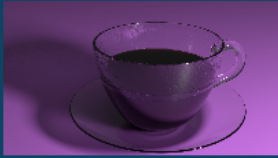
### Cliente



ID	Nombre	Apellido Paterno	Editar
1	DRAKE	GRAHAM	 
3	KANYE	WEST	 
4	KENDRICK	LAMAR	 
6	ASAP	ROCKY	 



Delete(Eliminar)









El siguiente icono, que se muestra, en forma de un bote de basura, borrara al cliente en cuestión.



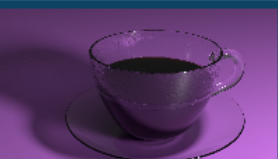
🔍 Cliente

### Cliente





ID	Nombre	Apellido Paterno	Editar
1	DRAKE	GRAHAM	 
3	KANYE	WEST	 
4	KENDRICK	LAMAR	 
6	ASAP	ROCKY	 









🔍 Cliente

### Cliente



ID	Nombre	Apellido Paterno	Editar
1	DRAKE	GRAHAM	 
3	KANYE	WEST	 
4	KENDRICK	LAMAR	