|  |
| --- |
| **SPRINT 2: Repositorio de Código - Diseño Base de Datos** |

|  |  |
| --- | --- |
| **Identificación Proyecto** | |
| Nombre Proyecto: | Aplicación web bajo entorno de código libre, para la recaudación de las cuotas de administración de conjuntos residenciales |
| Número Equipo: | 1 |
| **Integrantes del equipo** | |
| Rol  (Líder-Desarrollador – Cliente) | Nombre |
| Desarrollador | Oscar Niño Gutiérrez |
| Desarrollador | Oscar Durango González |
|  |  |

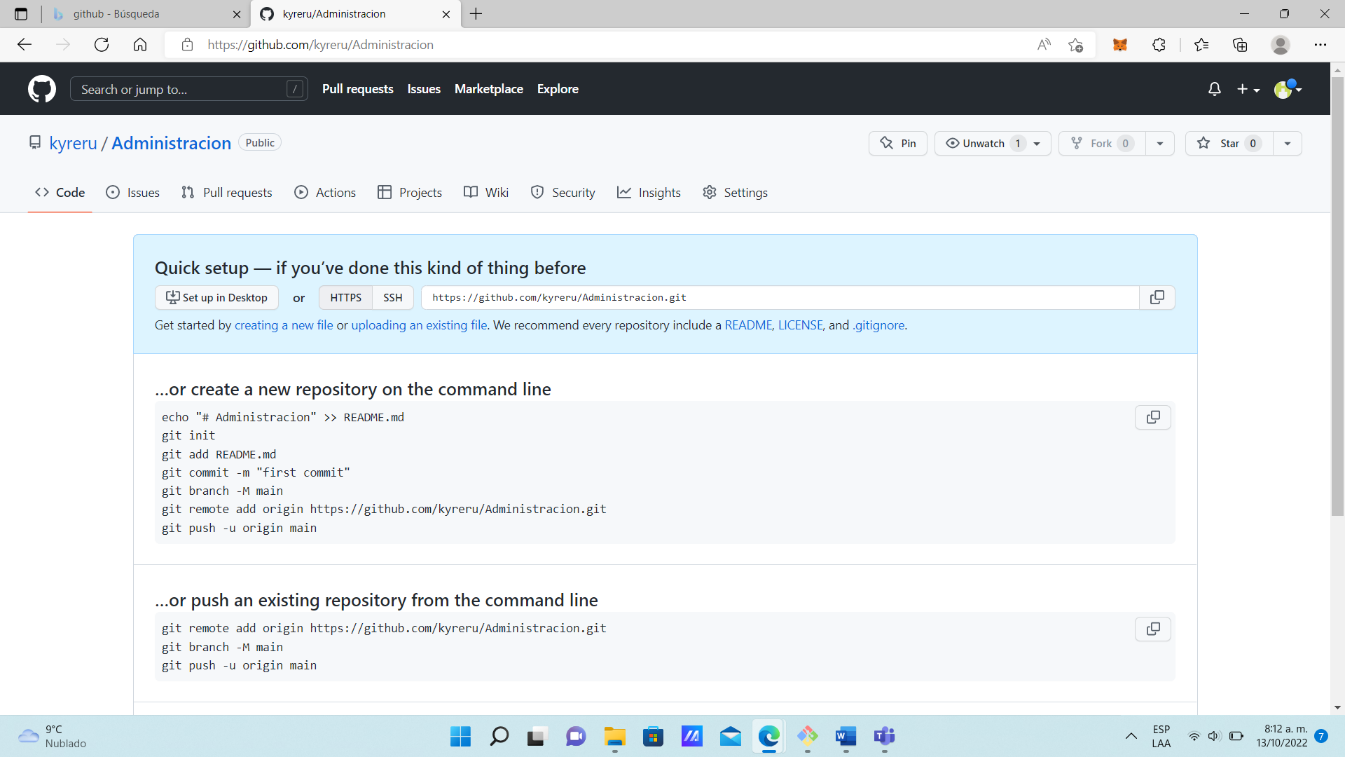
|  |
| --- |
| **Repositorio de Código GitLab o GitHub** |

Como evidencia del repositorio de código, creado con GitLab o GitHub, además de la URL del repositorio, se debe presentar capturas de pantalla donde se visualicen aspectos:

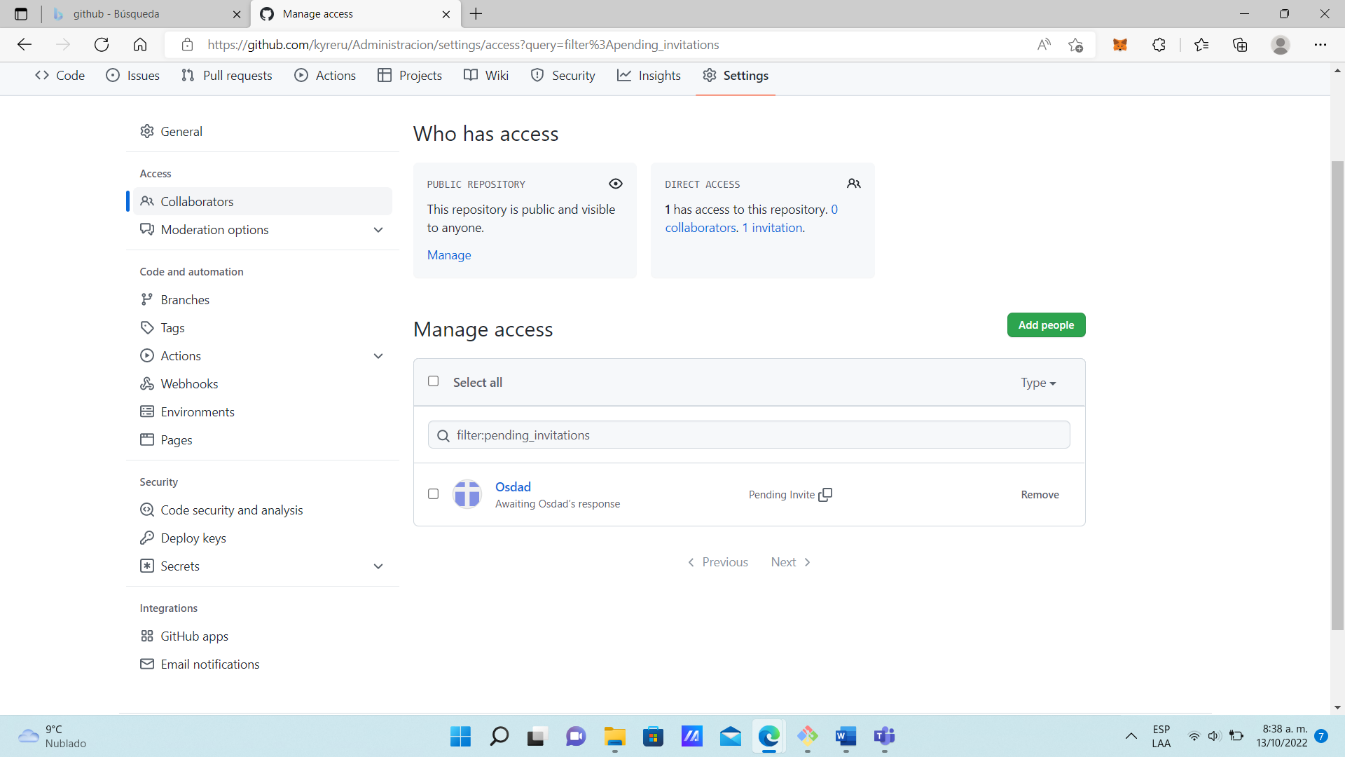
* Creación del proyecto del repositorio.

Captura de pantalla de computadora

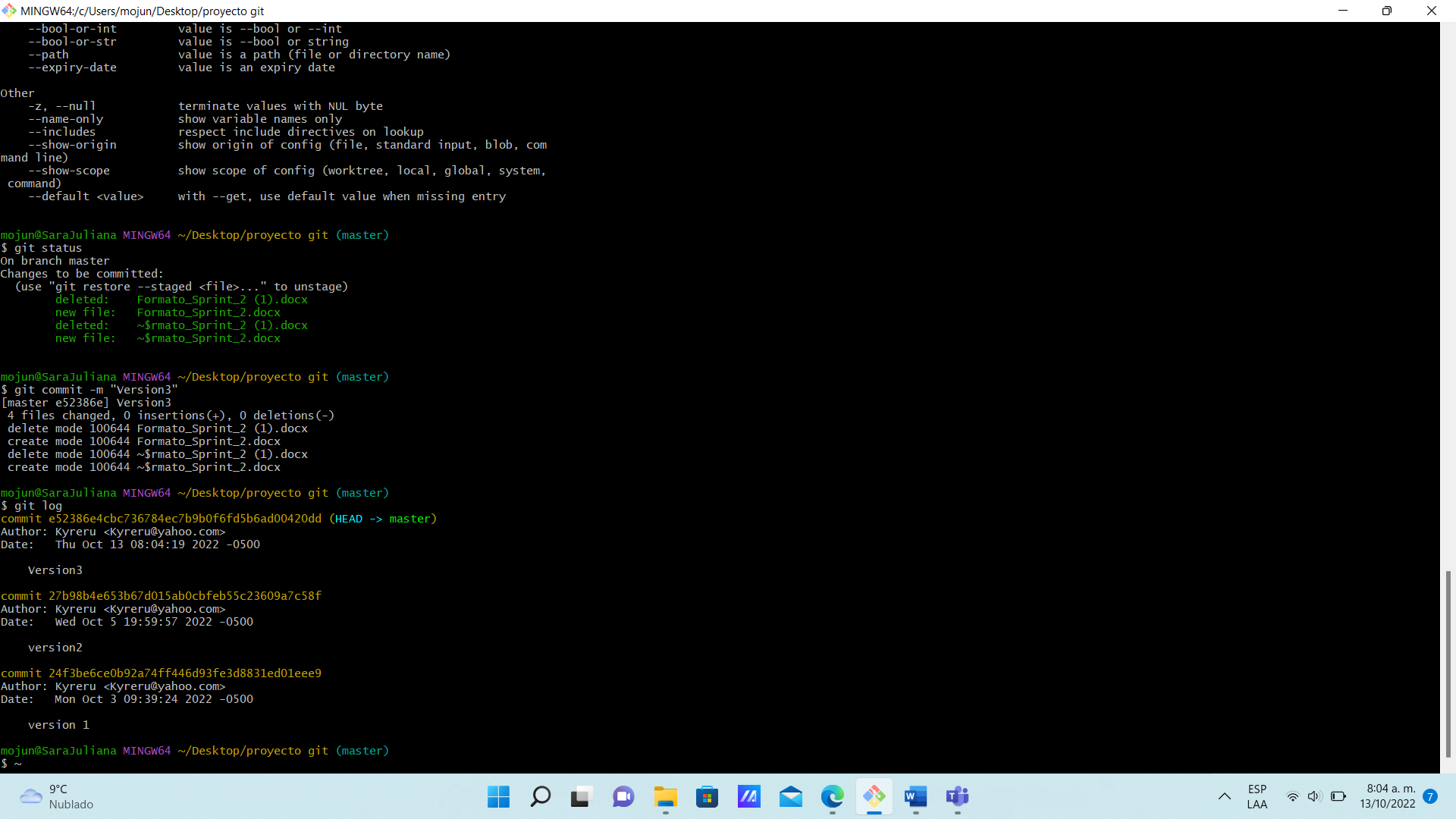
Descripción generada automáticamente

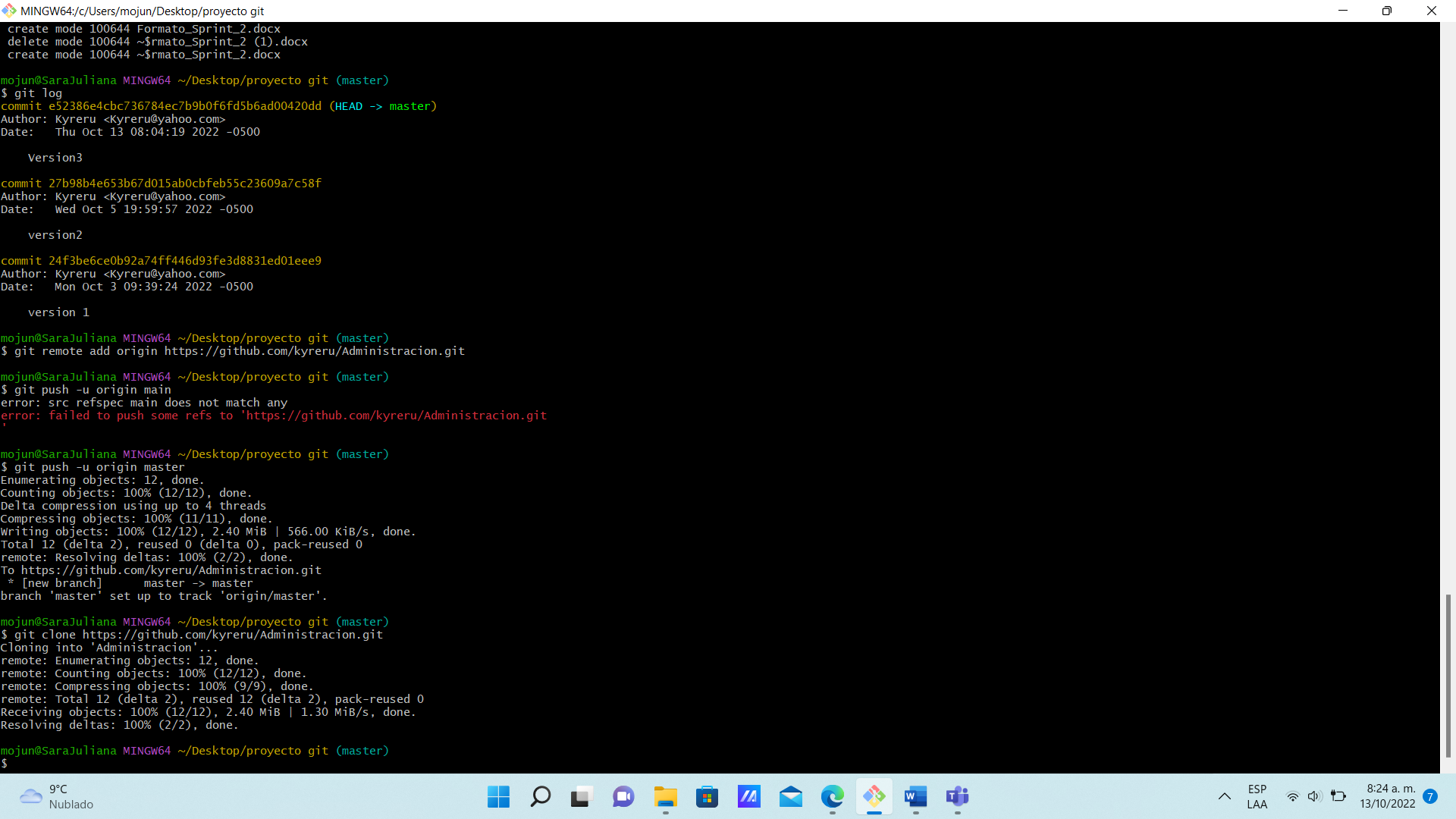


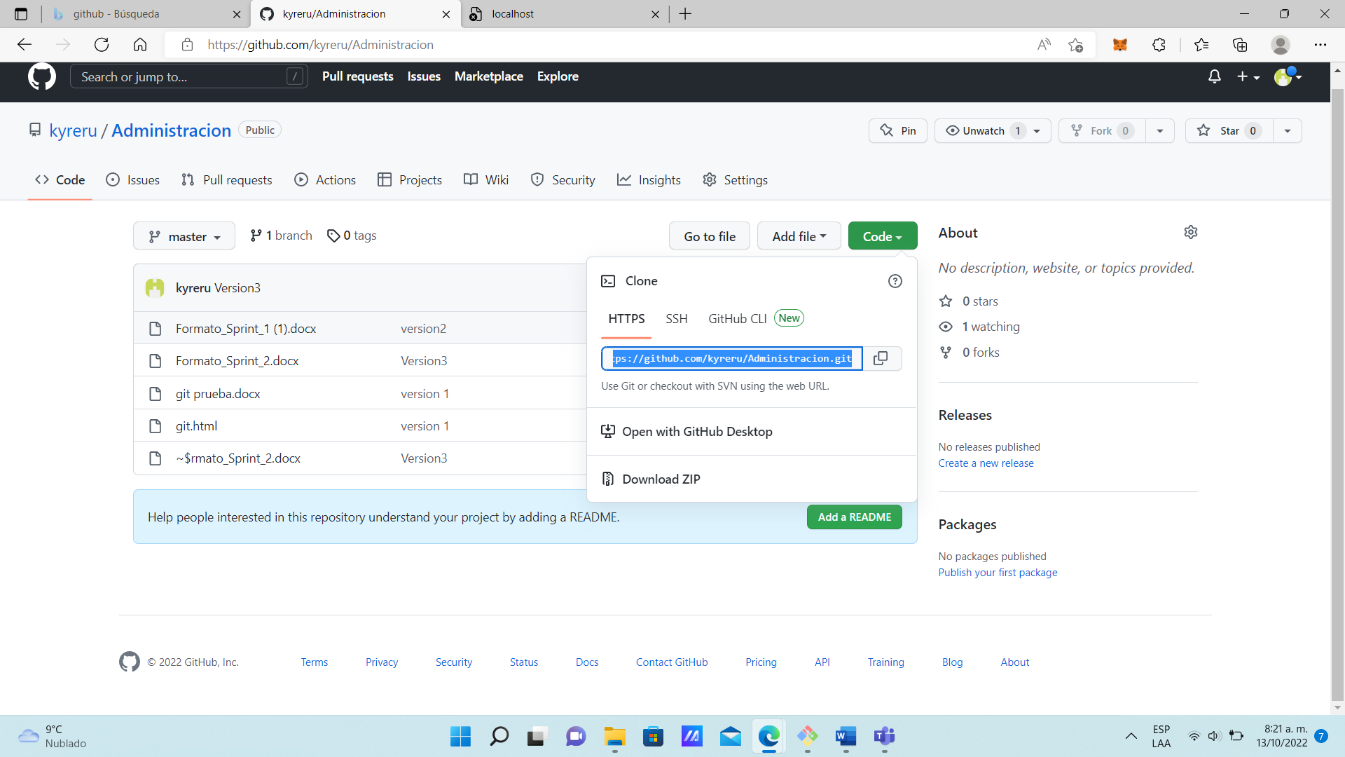
* Integrantes del equipo invitados.

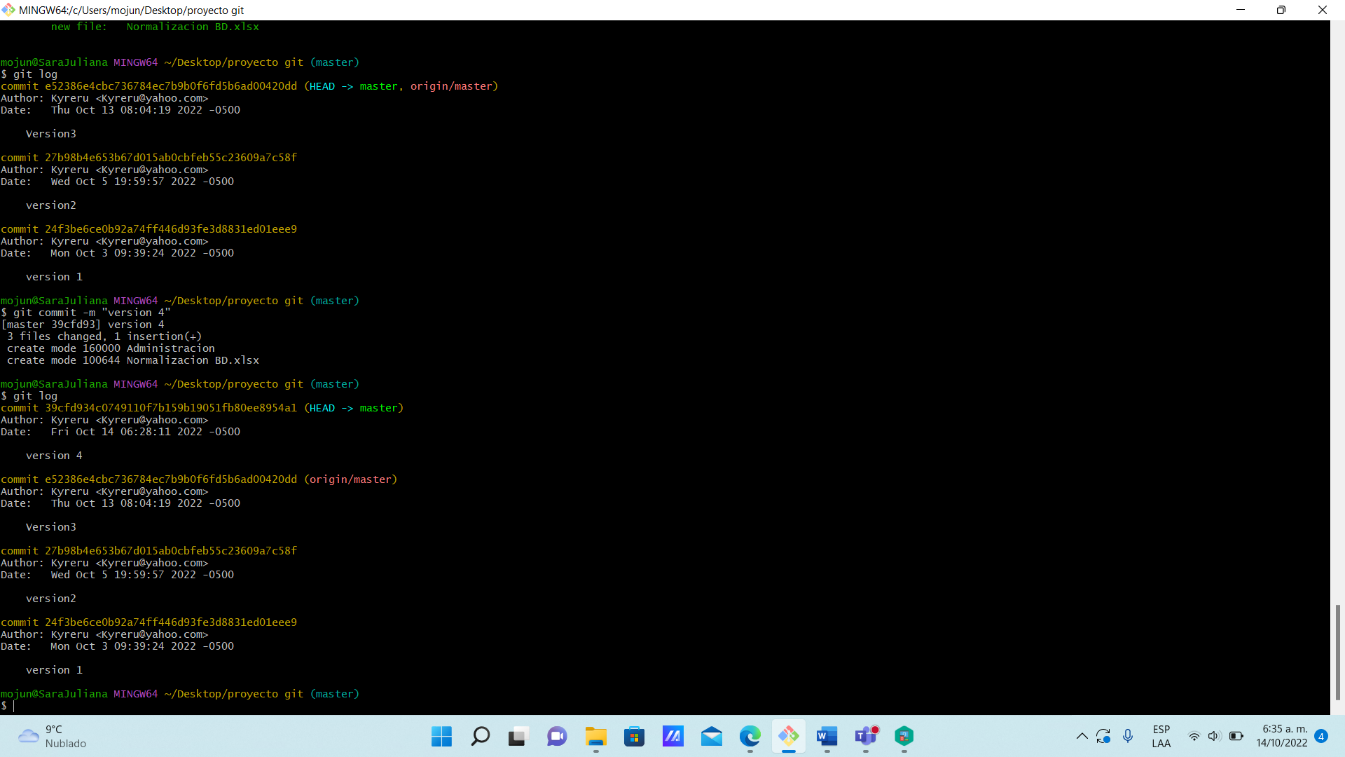


* Evidencia de la realización de alguna actualización (commit), donde se visualice la actualización y el historial de actualizaciones (Versiones)



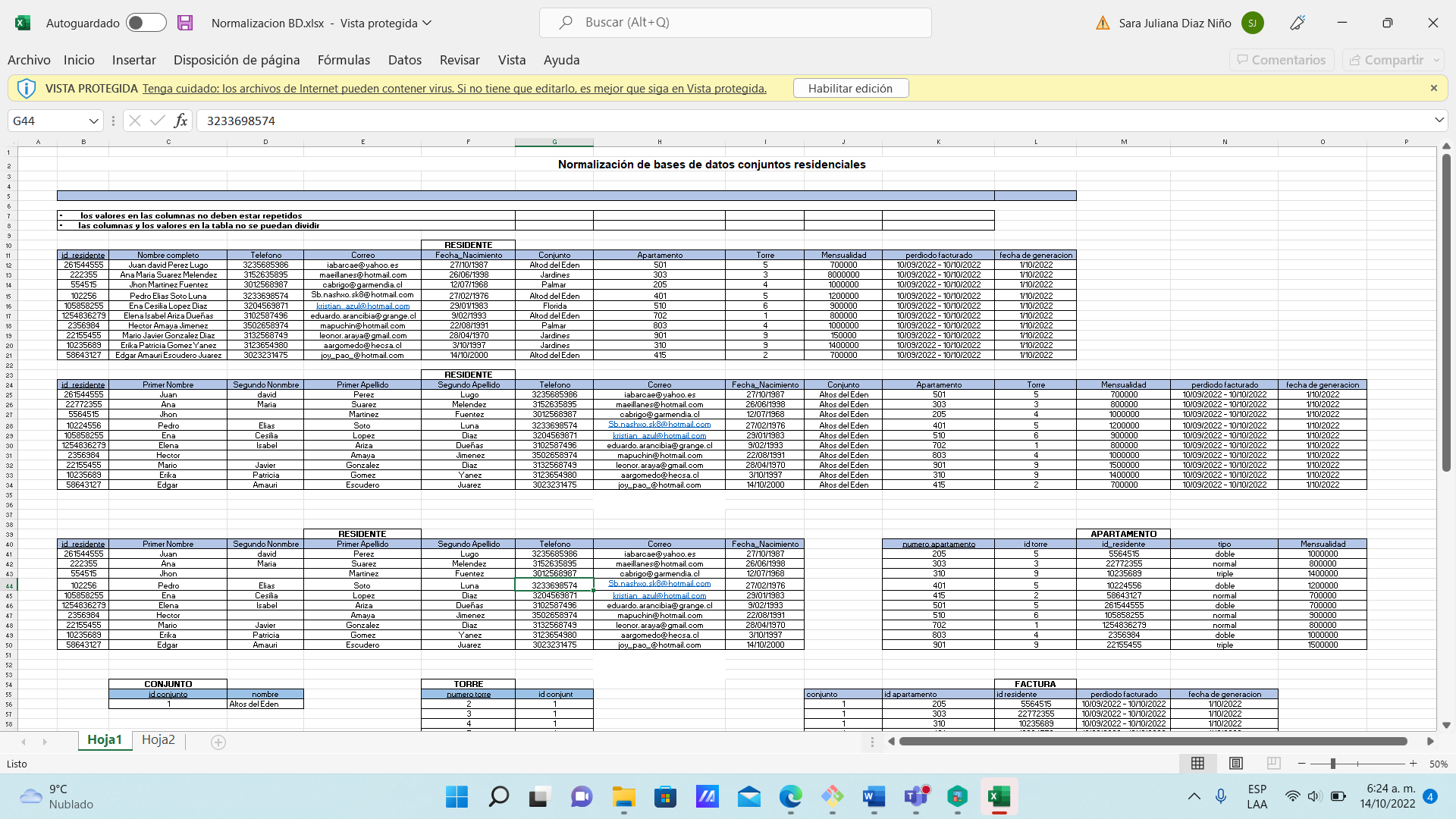


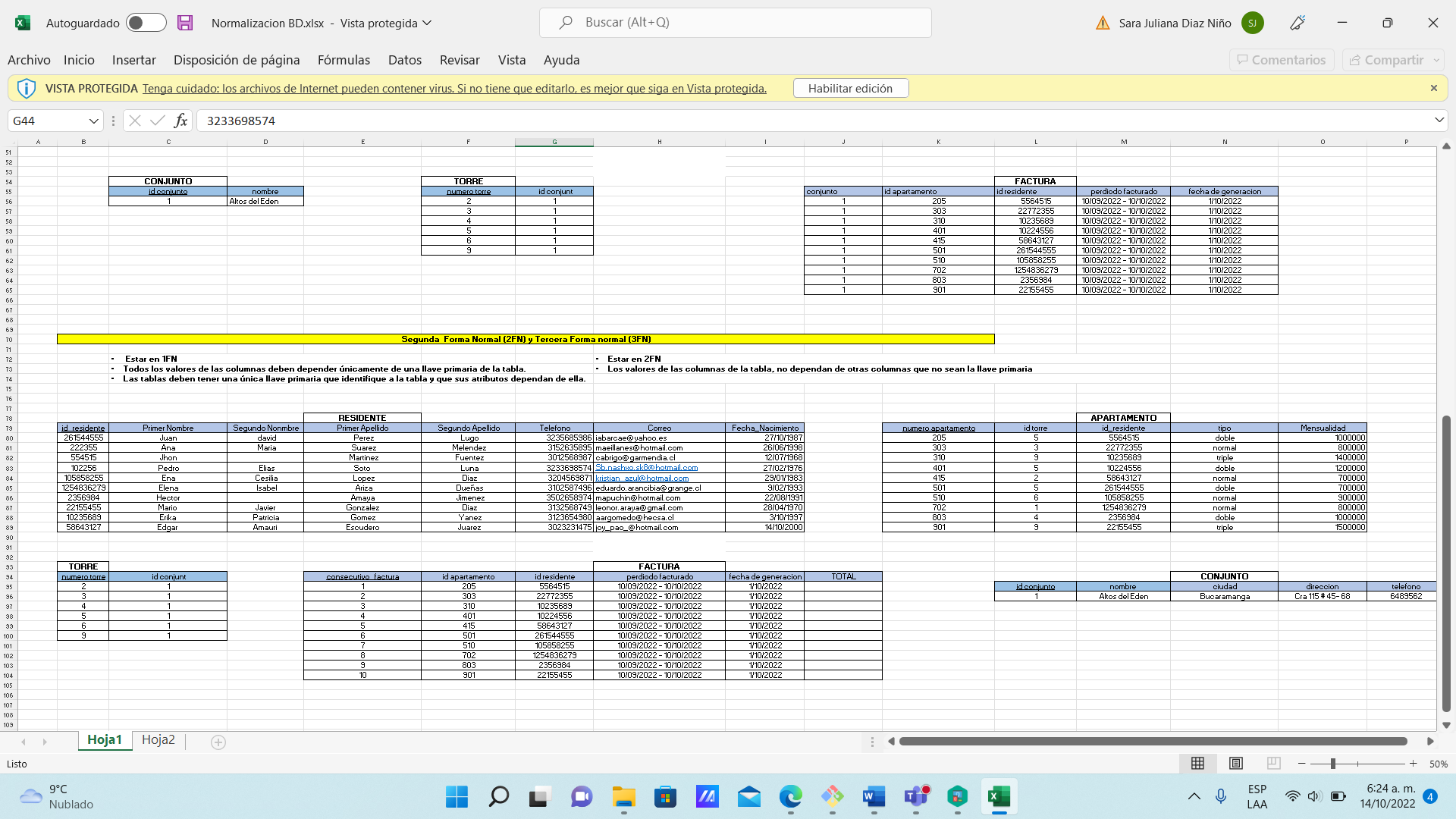




|  |
| --- |
| **Diseño de la Base de Datos (Proceso de normalización)** |

Como evidencia del Diseño de la Base de Datos, se debe presentar el proceso de normalización efectuado (Formas Normales).





|  |
| --- |
| **Esquema de la Base de Datos (Código SQL)** |

Como evidencia del Esquema de la Base de Datos, se debe presentar el código SQL de creación de la base de datos.

-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

-- -----------------------------------------------------

-- Schema conjunto

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema conjunto

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `conjunto` DEFAULT CHARACTER SET utf8 ;

USE `conjunto` ;

-- -----------------------------------------------------

-- Table `conjunto`.`conjunto`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `conjunto`.`conjunto` (

`id\_conjunto` INT NOT NULL AUTO\_INCREMENT,

`nombre` VARCHAR(45) NOT NULL,

`direccion` VARCHAR(45) NOT NULL,

`telefono` VARCHAR(45) NOT NULL,

PRIMARY KEY (`id\_conjunto`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `conjunto`.`torre`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `conjunto`.`torre` (

`id\_torre` INT NOT NULL AUTO\_INCREMENT,

`conjunto\_id\_conjunto` INT NOT NULL,

PRIMARY KEY (`id\_torre`),

CONSTRAINT `fk\_torre\_conjunto1`

FOREIGN KEY (`conjunto\_id\_conjunto`)

REFERENCES `conjunto`.`conjunto` (`id\_conjunto`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `conjunto`.`residente`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `conjunto`.`residente` (

`id\_residente` INT NOT NULL,

`primer\_nombre` VARCHAR(45) NOT NULL,

`segundo\_nombre` VARCHAR(45) NULL,

`primer\_apellido` VARCHAR(45) NOT NULL,

`segundo\_apellido` VARCHAR(45) NOT NULL,

`telefono` VARCHAR(45) NOT NULL,

`correo` VARCHAR(45) NOT NULL,

`fecha\_nacimiento` DATE NOT NULL,

PRIMARY KEY (`id\_residente`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `conjunto`.`apartamento`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `conjunto`.`apartamento` (

`id\_apartamento` INT NOT NULL AUTO\_INCREMENT,

`tipo` VARCHAR(45) NOT NULL,

`mensualidad` INT NOT NULL,

`residente\_id\_residente` INT NOT NULL,

`torre\_conjunto\_id\_conjunto` INT NOT NULL,

PRIMARY KEY (`id\_apartamento`, `residente\_id\_residente`, `torre\_conjunto\_id\_conjunto`),

INDEX `fk\_apartamento\_residente1\_idx` (`residente\_id\_residente` ASC) VISIBLE,

INDEX `fk\_apartamento\_torre1\_idx` (`torre\_conjunto\_id\_conjunto` ASC) VISIBLE,

CONSTRAINT `fk\_apartamento\_residente1`

FOREIGN KEY (`residente\_id\_residente`)

REFERENCES `conjunto`.`residente` (`id\_residente`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_apartamento\_torre1`

FOREIGN KEY (`torre\_conjunto\_id\_conjunto`)

REFERENCES `conjunto`.`torre` (`conjunto\_id\_conjunto`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `conjunto`.`factura`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `conjunto`.`factura` (

`consecutivo` INT NOT NULL AUTO\_INCREMENT,

`periodo\_facturado` DATE NOT NULL,

`fecha\_generacion` DATE NOT NULL,

`total` INT NOT NULL,

`apartamento\_id\_apartamento` INT NOT NULL,

`apartamento\_residente\_id\_residente` INT NOT NULL,

`apartamento\_torre\_conjunto\_id\_conjunto` INT NOT NULL,

PRIMARY KEY (`consecutivo`, `apartamento\_id\_apartamento`, `apartamento\_residente\_id\_residente`, `apartamento\_torre\_conjunto\_id\_conjunto`),

INDEX `fk\_factura\_apartamento1\_idx` (`apartamento\_id\_apartamento` ASC, `apartamento\_residente\_id\_residente` ASC, `apartamento\_torre\_conjunto\_id\_conjunto` ASC) VISIBLE,

CONSTRAINT `fk\_factura\_apartamento1`

FOREIGN KEY (`apartamento\_id\_apartamento` , `apartamento\_residente\_id\_residente` , `apartamento\_torre\_conjunto\_id\_conjunto`)

REFERENCES `conjunto`.`apartamento` (`id\_apartamento` , `residente\_id\_residente` , `torre\_conjunto\_id\_conjunto`)

ON DELETE NO ACTION

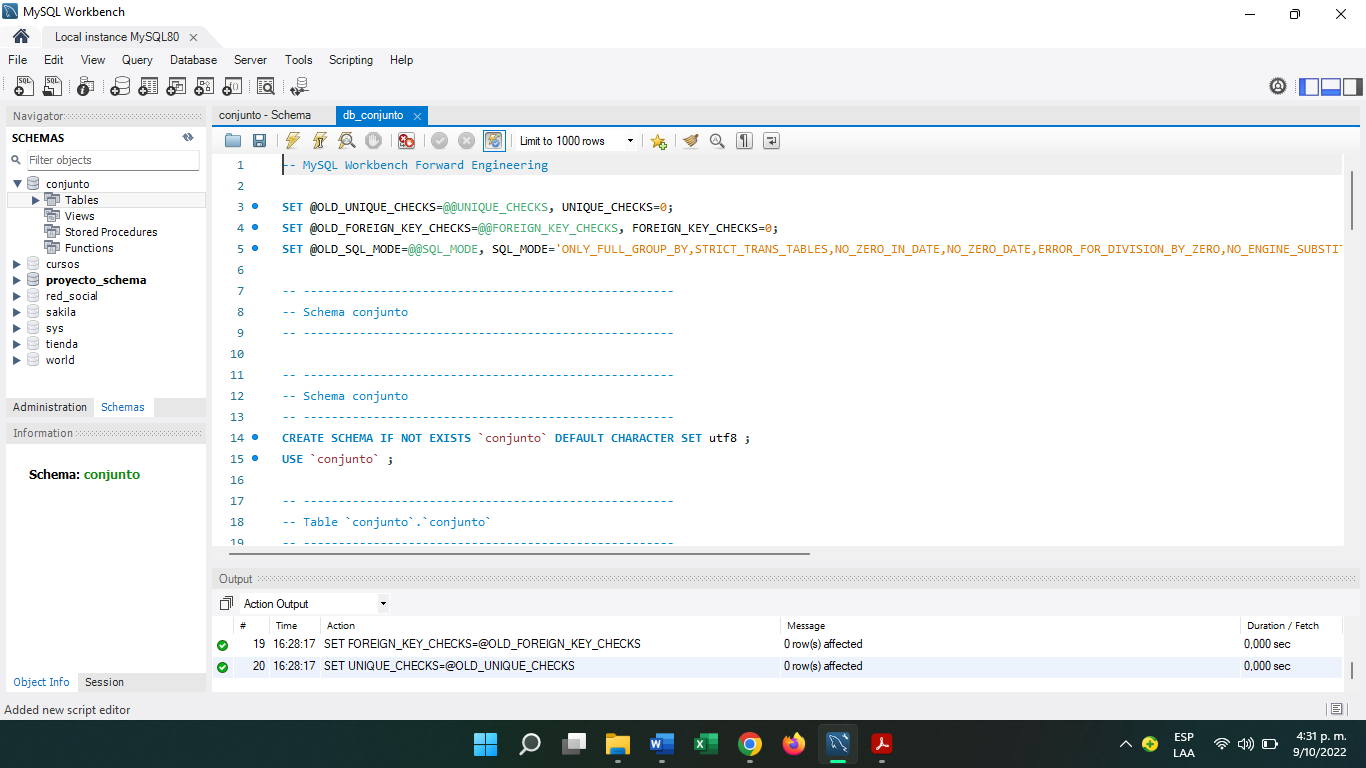
ON UPDATE NO ACTION)

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

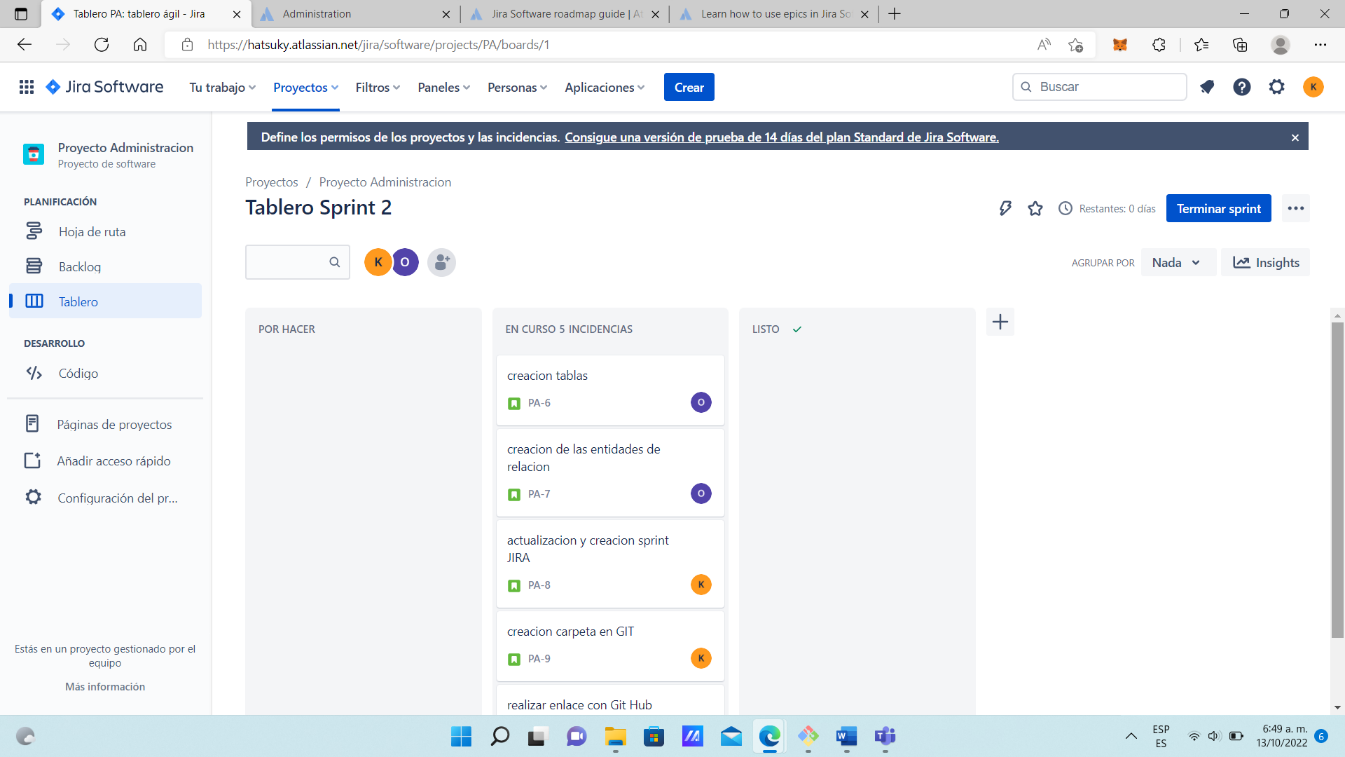
SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

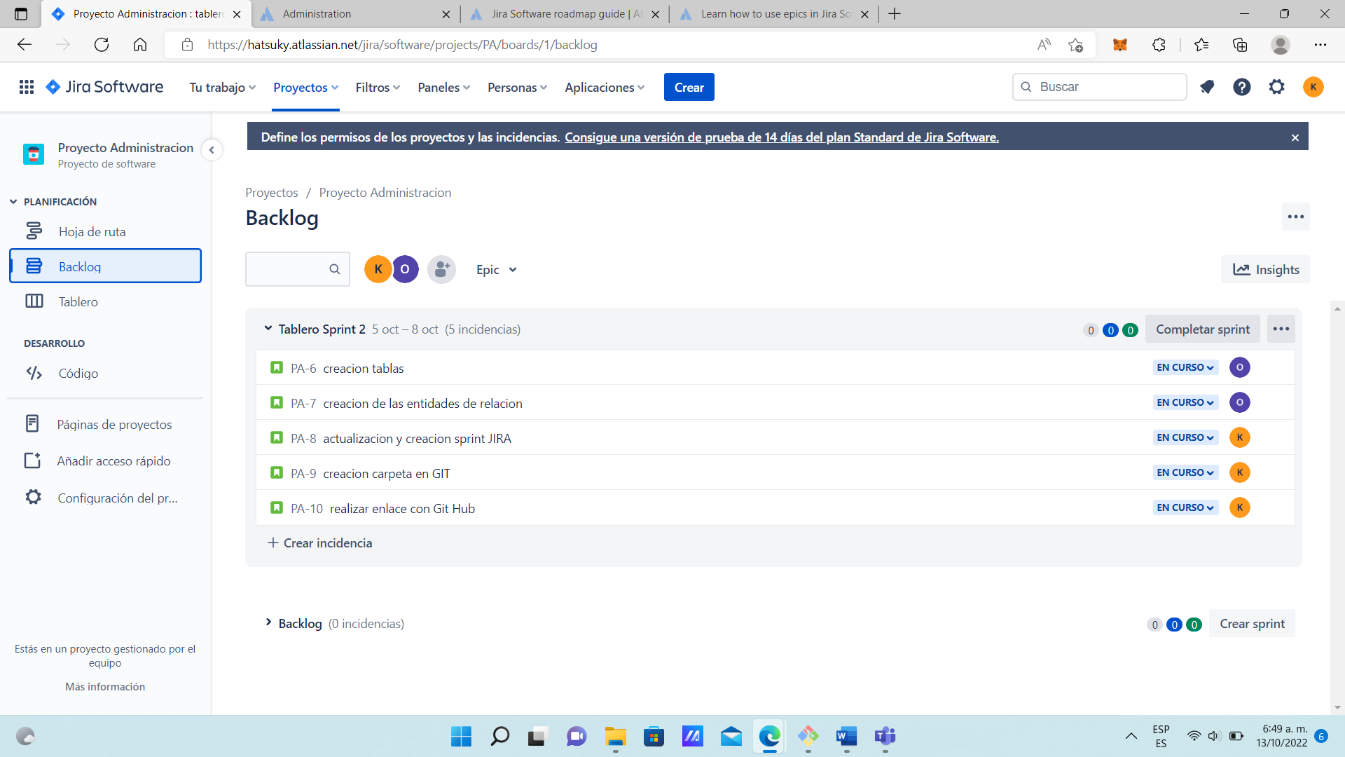
SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

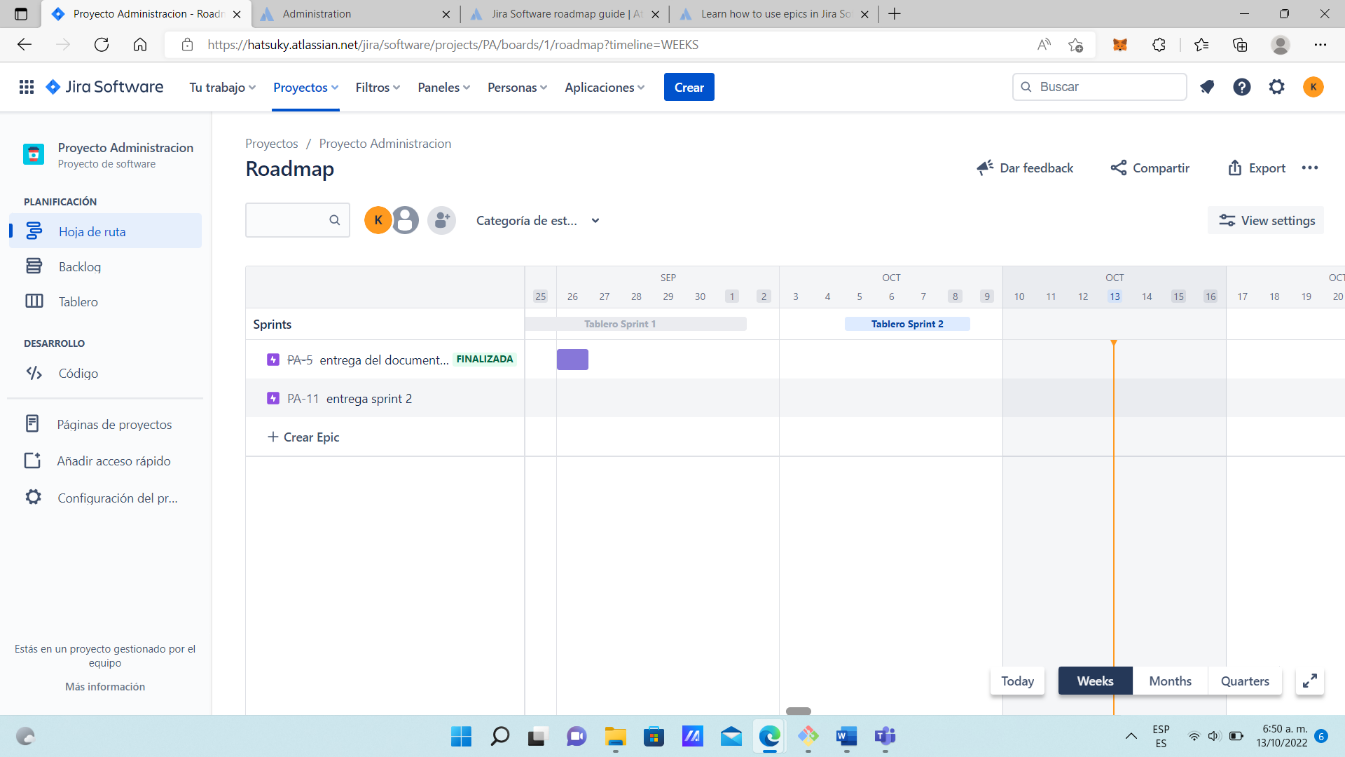


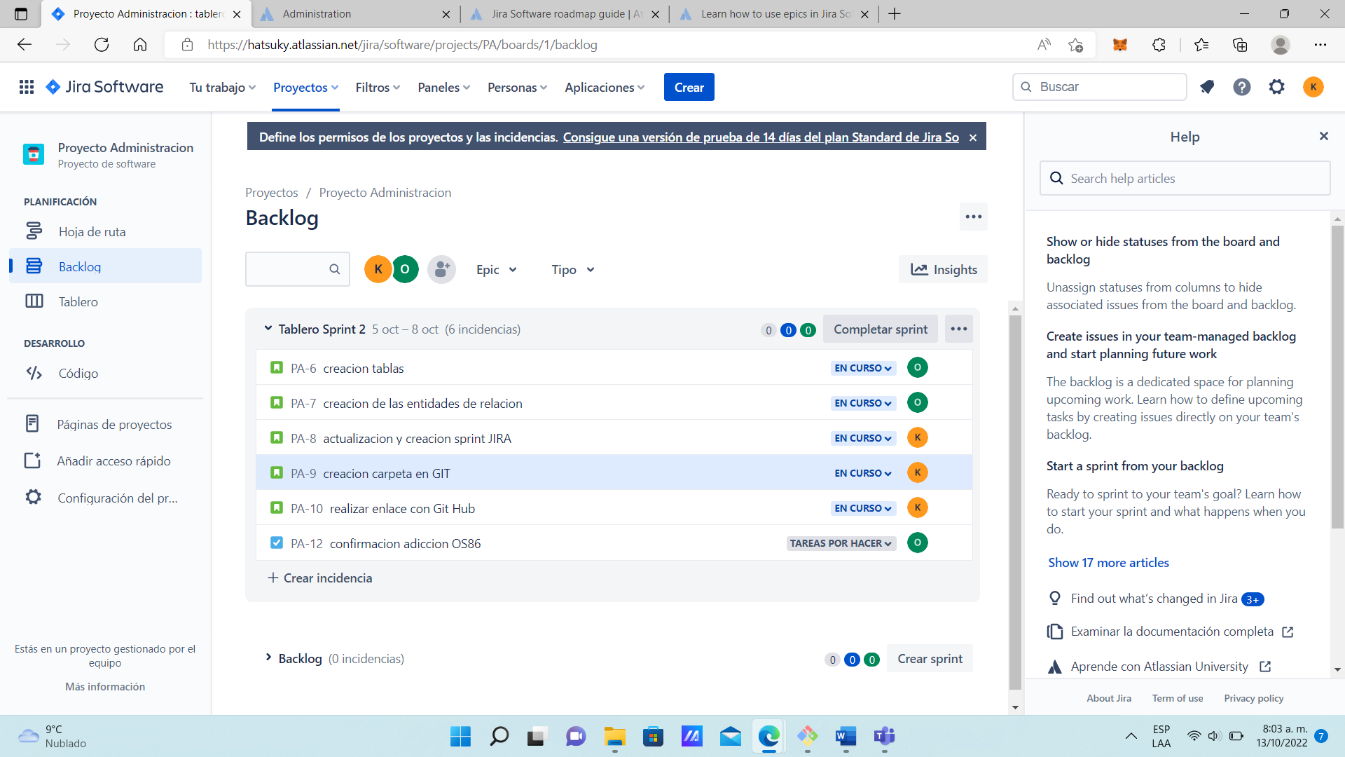
|  |
| --- |
| **Evidencia JIRA (Seguimiento del proyecto)** |

Como evidencia del seguimiento del proyecto con la metodología ágil SCRUM, utilizando el software JIRA, se debe presentar capturas de pantalla donde se visualice la ejecución de los Sprint con las historias de usuario relacionadas con el repositorio de código y el diseño de la base de datos.









|  |
| --- |
| **Evidencias de las Reuniones de Equipo** |

Como evidencia de las reuniones que efectúa el equipo del proyecto, presentar capturas de pantalla de las reuniones efectuadas y si lo consideran pertinente algunas actas de las reuniones.

