Manual De Configuración Base de Datos

Easy House Rent

Versión: 2.3

**HISTORIAL DE REVISIÓN**

| **VERSIÓN** | **ELABORACIÓN** | | **REVISIÓN** | | **APROBACIÓN** | |
| --- | --- | --- | --- | --- | --- | --- |
| **Fecha** | **Responsable** | **Fecha** | **Responsable** | **Fecha** | **Responsable** |
| 2.0 | 09/10/2021 | Jhon Maicol |  | Sebastian |  | Willington |
| 2.2 | 3/3/2022 | Jhon Maicol |  | Sebastian |  |  |
| 2.3 | 07/06/2022 | Julian David Amaya Gomez | 08/06/2022 |  |  |  |
|  |  |  |  |  |  |  |
| 2.4 | 28/06/2022 | Julian David Amaya Gomez | Hugo Hernán Henao Hernández | | Documento aprobado | |

**CAMBIOS RESPECTO A LA VERSIÓN ANTERIOR**

| **VERSIÓN** | **MODIFICACIÓN RESPECTO VERSIÓN ANTERIOR** |
| --- | --- |
| 1.0 | Documento inicial |
| 2.0 | Se cambio todo el documento |
| 2.1 | modificación en entidad usuarios y creación de entidad municipios |
| 2.2 | Cambio del alcance |
| 2.3 | Se cambio la justificación, se actualizo introducción y alcance |
| 2.4 | Aprobación del documento. |

**Tabla de contenido**

[1. Introducción](#_30j0zll) **4**

[2. Alcance](#_1fob9te) **4**

[4. Responsables e involucrados](#_3znysh7) **4**

[**5. Modelo Entidad Relación (MER)**](#_aq6d48fdutqt) **5**

[**6. Diccionario de Datos.**](#_nzjzyrb3ljr7) **6**

[**tabla Usuarios:**](#_zguxuqip7vix) **6**

[**Tabla anuncios**](#_63y1rgc1yuuu) **7**

[**Tabla Oferta**](#_1dstx53hdgo5) **10**

[**Tabla departamentos**](#_gyzj84jvo4ip) **11**

[**Tabla municipios**](#_nrt981u6nnbl) **11**

[**7. Modelo Relacional.**](#_g35romzbuhay) **12**

[**8. Justificación Motor Seleccionado**](#_as59snvo2szq) **13**

[**9. Requisitos de Configuración**](#_ndng4xdoa1lf) **14**

[10. Scripts](#_2s8eyo1) **15**

[11. Configuración y Ejecución de la Base de Datos](#_17dp8vu) **19**

[12. Otras Consideraciones](#_3rdcrjn) **20**

# 1. Introducción

El proyecto a trabajar denominado Easy House Rent está enfocado en la venta, renta y compra de casas en línea, para la elaboración de la base de datos se utiliza el Sistema de gestión de bases de datos relacionales MySQL.

# 2. Alcance

La aplicación web EHR (Easy House Rent) cuenta con funcionalidades y herramientas orientadas a facilitar la búsqueda de vivienda a los usuarios/clientes del sistema, para ello se tiene una interfaz principal en la cual se visualizan los diferentes anuncios creados por la comunidad.

En su primera versión estará disponible solo para la ciudad de Armenia-Quindío, EHR no permite realizar pagos a través de ella, ni se hará responsable por los acuerdos realizados entre los compradores y clientes.

El presente documento tiene como finalidad poner a disposición la configuración de la base de datos de la aplicación, con el fin de dar a conocer su estructura y los datos que la componen.

# 4. Responsables e involucrados

| **Nombre** | **Tipo (Responsable/ Involucrado)** | **Rol** |
| --- | --- | --- |
| *Juan Pablo* | *Responsable* | *diseñador,desarrollador, redactor* |
| *Sebastian Ruiz* | *Responsable* | *diseñador,desarrollador, redactor* |
| *Karen Agudelo* | *Responsable* | *diseñador,desarrollador, redactor* |
| *Julian Amaya* | *Responsable* | *diseñador,desarrollador, redactor* |
| *Jhon Montoya* | *Responsable* | *diseñador,desarrollador, redactor* |

# 5. Modelo Entidad Relación (MER)

# 

# 

# 

# 6. Diccionario de Datos.

# tabla Usuarios:

| usuarios | | | | | |
| --- | --- | --- | --- | --- | --- |
| atributo | descripción | características | tipo de dato | dominio de valores | índices y relaciones |
| idUsuario | número de identificación del usuario | campo obligatorio no permite valores nulos | INT |  | llave principal  auto incremental |
| nombre | nombre del usuario | campo obligatorio no permite valores nulos | CHAR(80) |  |  |
| Apellidos | apellidos del usuario | campo obligatorio no permite valores nulos | CHAR(80) |  |  |
| edad | La edad del usuario | campo obligatorio no permite valores nulos | INT |  |  |
| teléfono | número de teléfono del usuario |  | VARCHAR(10) |  |  |
| email | correo electrónico del usuario | campo obligatorio no permite valores nulos | VARCHAR(120) |  | índice único |
| contraseña | contraseña del usuario encriptada a 64 caracteres | campo obligatorio no permite valores nulos | VARCHAR(200) |  |  |
| estado | estado del usuario en el sistema | campo obligatorio no permite valores nulos | CHAR(1) | I = inactivo  A = activo  E = en espera |  |
| departamento | departamento donde vive el usuario |  | INT |  | llave foránea departamento (iddepartamento) |
| municipio | municipio donde vive el usuario |  | INT |  | llave foránea  municipios (idmunicipio) |
| foto | url de la foto del usuario | campo obligatorio no permite valores nulos | VARCHAR(300) |  |  |

# 

# Tabla anuncios

| anuncios | | | | | |
| --- | --- | --- | --- | --- | --- |
| atributo | descripción | características | tipo de dato | dominio de valores | índices y relaciones |
| idanuncio | número de identificación del anuncio | campo obligatorio no permite valores nulos | INT |  | llave principal  auto incremental |
| idusuario | número de identificación del usuario | campo obligatorio no permite valores nulos | INT |  | llave foránea  usuarios(idusuario) |
| título | titulo del anuncio | campo obligatorio no permite valores nulos | CHAR(80) |  |  |
| descripción | descripción del anuncio | campo obligatorio no permite valores nulos | VARCHAR(3000) |  |  |
| modalidad | determinar si es arrendo, venta o permuta | campo obligatorio no permite valores nulos | CHAR(30) |  |  |
| zona | la zona en la que se encuentra el inmueble | campo obligatorio no permite valores nulos | CHAR(40) |  |  |
| garaje | si el inmueble cuenta con un garaje | campo obligatorio no permite valores nulos | CHAR(2) | SI= cuenta con garaje  NO=cuenta con garaje |  |
| habitaciones | cantidad de habitaciones con las que cuenta el inmueble | campo obligatorio no permite valores nulos | INT |  |  |
| edificacion |  | campo obligatorio no permite valores nulos | VARCHAR(50) |  |  |
| calificación | puntuación del anuncio dada por los usuarios |  | TINYINT |  |  |
| dirección | direccion del anuncio publicado | campo obligatorio no permite valores nulos | VARCHAR(150) |  |  |
| estado | estado del producto anunciado | campo obligatorio no permite valores nulos | CHAR(1) | E = excelente  R = regular  P = pésimo |  |
| precio | precio del inmueble publicado en el anuncio | campo obligatorio no permite valores nulos | FLOAT |  |  |
| fecha | fecha de publicación | campo obligatorio no permite valores nulos | DATE |  |  |
| certificado | certificado de propiedad |  | VARCHAR(500) |  |  |
| url1 | primer url de las fotos del inmueble | campo obligatorio no permite  valores nulos | VARCHAR(250) |  |  |
| url2 | segunda url de las fotos del inmueble | campo obligatorio no permite  valores nulos | VARCHAR(250) |  |  |
| url3 | tercera url de las fotos del inmueble | campo obligatorio no permite  valores nulos | VARCHAR(250) |  |  |
| url4 | cuarta url de las fotos del inmueble | campo obligatorio no permite  valores nulos | VARCHAR(250) |  |  |
| Ciudad | La ciudad donde se encuentra el inmueble |  | CHAR(50) |  |  |

# Tabla Oferta

| oferta | | | | | |
| --- | --- | --- | --- | --- | --- |
| atributo | descripción | características | tipo de dato | dominio de valores | índices y relaciones |
| oferta | número de identificación de la oferta | campo obligatorio no permite valores nulos | INT |  | llave primaria |
| idanuncio | número de identificación del anuncio | campo obligatorio no permite valores nulos | INT |  | llave foránea  anuncios (idanuncio) |
| idusuario | número de identificación del usuario | campo obligatorio no permite valores nulos | INT |  | llave foránea usuarios (idusuario) |

# 

# 

# Tabla departamentos

| departamentos | | | | | |
| --- | --- | --- | --- | --- | --- |
| atributo | descripción | características | tipo de dato | dominio de valores | índices y relaciones |
| iddepartamento | número de identificación del departamento | campo obligatorio no permite valores nulos | INT |  | llave primaria |
| nombre | nombre del departamento | campo obligatorio no permite valores nulos | CHAR(80) |  |  |

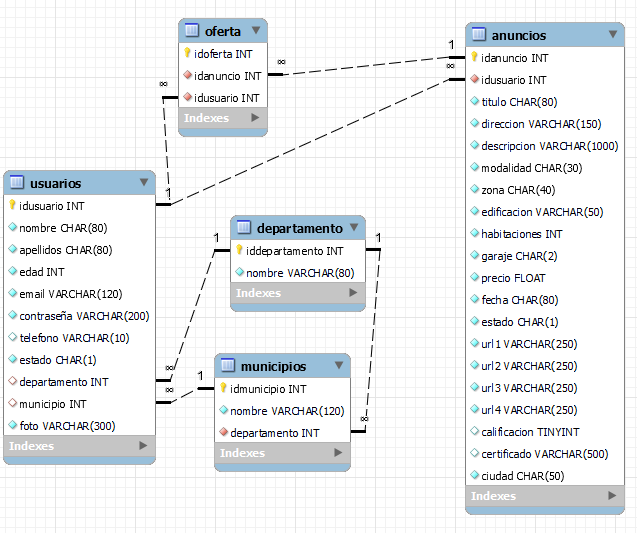
# Tabla municipios

| municipios | | | | | |
| --- | --- | --- | --- | --- | --- |
| atributo | descripción | características | tipo de dato | dominio de valores | índices y relaciones |
| idmunipio | número de identificación del municipio | campo obligatorio no permite valores nulos | INT |  | llave primaria |
| iddepartamento | número de identificación del departamento | campo obligatorio no permite valores nulos | INT |  | llave foranea  departamento(iddepartamento) |
| nombre | nombre del departamento | campo obligatorio no permite valores nulos | VARCHAR(120) |  |  |

# 

# 

# 7. Modelo Relacional.



*.*

# 

# 8. Justificación Motor Seleccionado

Se trabaja con el Sistema de gestión de bases de datos relacionales MySQL por múltiples motivos como lo son el hosting de múltiples plataformas para este sistema, es un Sistema de gestión de bases de datos relacionales muy robusto permitiendo escalamiento de manera sencilla a niveles más grandes además de una flexibilidad de alto grado.

Para poner un mayor contexto usaremos un cuadro comparativo:

# 

|  | **MySQL** | **MongoDB** | **PostgreSQL** |
| --- | --- | --- | --- |
| Modelo de base de datos | RDBMS es un sistema de gestión de bases de datos relacionales basado en el modelo relacional de datos. | Almacenamiento de documentos | ORDBMS es un sistema de gestión de bases de datos relacionales orientado a objetos basado en el modelo de base de datos relacional y orientado a objetos. |
| Año de lanzamiento | 1995 | 2009 | 1989 |
| Soporte para XML | si | no | si |
| Scripts del lado del servidor | JavaScript | JavaScript | funciones definidas por el usuario |
| Uso de “foreign Keys” | Si | No | Si |
| Licencia | Open source | Open source | Open source |
| Sistemas operativos compatibles | * FreeBSD * Linux * OS X * Solaris * Windows | * Linux * OS X * Solaris * Windows | * FreeBSD * HP-UX * Linux * NetBSD * OpenBSD * OS X * Solaris * Unix * Windows |
| Esquema de datos | Si | No | Si |
| API y otros métodos de acceso | * ADO.NET * JDBC * ODBC * API nativa patentada | protocolo propietario usando JSON | * ADO.NET * JDBC * biblioteca C nativa API de transmisión * ODBC * para objetos grandes |

* Comparison between MongoDB vs MySQL vs PostgreSQL. (2021, octubre 18). Linearloop - We Are Innovative Software Development Company; Linearloop Private Limited. <https://www.linearloop.io/blog/comparison-between-mongodb-vs-mysql-vs-postgresql/>

En conclusión, debido a la complejidad del proyecto y a la cantidad de datos por la cual vamos a manejar (la cual no será muy alta en sus primeras versiones) se usará MYSQL por su fácil instalación y despliegue, una alta estabilidad y un mantenimiento casi nula,

# 9. Requisitos de Configuración

En este momento estamos realizando el trabajo de forma local, por lo tanto hemos instalado el software XAMPP que nos brinda la posibilidad de instalar una instancia del motor de base de datos MYSQL (versión 8.0). lo hemos instalado en Windows y estan son las especificaciones del equipo para XAMPP:

Memoria RAM 256 MB

disco duro 85 MB

instalamos también el gestor de base de datos MYSQL Workbench, mediante el cual podemos realizar todas las operaciones sobre la base de datos, en este momento de forma local y sus especificaciones mínimas son:

512 Mb de memoria Ram.

1024 Mb máquina virtual.

1 GB de espacio de disco duro.

Y para el despliegue se necesita:

un hosting que acepte SQL

1 GB de memoria RAM

2 GB de almacenamiento

# 10. Scripts

CREATE SCHEMA IF NOT EXISTS `ehr`;

CREATE TABLE IF NOT EXISTS `ehr`.`departamento` (

`iddepartamento` INT NOT NULL,

`nombre` VARCHAR(80) NOT NULL,

PRIMARY KEY (`iddepartamento`));

CREATE TABLE IF NOT EXISTS `ehr`.`municipios` (

`idmunicipio` INT NOT NULL AUTO\_INCREMENT,

`nombre` VARCHAR(120) NOT NULL,

`departamento` INT NOT NULL,

PRIMARY KEY (`idmunicipio`),

INDEX `municipio\_departamento` (`departamento` ASC),

CONSTRAINT `municipio\_departamento`

FOREIGN KEY (`departamento`)

REFERENCES `ehr`.`departamento` (`iddepartamento`));

CREATE TABLE IF NOT EXISTS `ehr`.`usuarios` (

`idusuario` INT NOT NULL AUTO\_INCREMENT,

`nombre` CHAR(80) NOT NULL,

`apellidos` CHAR(80) NOT NULL,

`edad` INT NOT NULL,

`email` VARCHAR(120) NOT NULL,

`contraseña` VARCHAR(200) NOT NULL,

`telefono` VARCHAR(10) NULL DEFAULT NULL,

`estado` CHAR(1) NOT NULL,

`departamento` INT NULL DEFAULT NULL,

`municipio` INT NULL DEFAULT NULL,

`foto` VARCHAR(300) NOT NULL,

PRIMARY KEY (`idusuario`),

UNIQUE INDEX `email\_UNIQUE` (`email` ASC),

INDEX `usuarios\_municipio\_idx` (`municipio` ASC),

INDEX `usuarios\_departamento\_idx` (`departamento` ASC),

CONSTRAINT `usuarios\_departamento`

FOREIGN KEY (`departamento`)

REFERENCES `ehr`.`departamento` (`iddepartamento`),

CONSTRAINT `usuarios\_municipio`

FOREIGN KEY (`municipio`)

REFERENCES `ehr`.`municipios` (`idmunicipio`))

;

CREATE TABLE IF NOT EXISTS `ehr`.`anuncios` (

`idanuncio` INT NOT NULL AUTO\_INCREMENT,

`idusuario` INT NOT NULL,

`titulo` CHAR(80) NOT NULL,

`direccion` VARCHAR(150) NOT NULL,

`descripcion` VARCHAR(1000) NOT NULL,

`modalidad` CHAR(30) NOT NULL,

`zona` CHAR(40) NOT NULL,

`edificacion` VARCHAR(50) NOT NULL,

`habitaciones` INT NOT NULL,

`garaje` CHAR(2) NOT NULL,

`precio` FLOAT NOT NULL,

`fecha` CHAR(80) NOT NULL,

`estado` CHAR(1) NOT NULL,

`url1` VARCHAR(250) NOT NULL,

`url2` VARCHAR(250) NOT NULL,

`url3` VARCHAR(250) NOT NULL,

`url4` VARCHAR(250) NOT NULL,

`calificacion` TINYINT NULL DEFAULT NULL,

`certificado` VARCHAR(500) NULL DEFAULT NULL,

`ciudad` CHAR(50) NOT NULL,

PRIMARY KEY (`idanuncio`),

INDEX `usuarios\_idx` (`idusuario` ASC),

CONSTRAINT `anuncio - usuarios`

FOREIGN KEY (`idusuario`)

REFERENCES `ehr`.`usuarios` (`idusuario`))

;

CREATE TABLE IF NOT EXISTS `ehr`.`oferta` (

`idoferta` INT NOT NULL,

`idanuncio` INT NOT NULL,

`idusuario` INT NOT NULL,

PRIMARY KEY (`idoferta`),

INDEX `anuncios\_idx` (`idanuncio` ASC),

INDEX `usuarios\_idx` (`idusuario` ASC),

CONSTRAINT `anuncios`

FOREIGN KEY (`idanuncio`)

REFERENCES `ehr`.`anuncios` (`idanuncio`),

CONSTRAINT `usuarios`

FOREIGN KEY (`idusuario`)

REFERENCES `ehr`.`usuarios` (`idusuario`))

;

# 11. Configuración y Ejecución de la Base de Datos

El proceso para la configuración de la base de datos es, primero se crea la base de datos con sus respectivas tablas, también se deben relacionar entre ellas para un buen funcionamiento, la base de datos se conectara y funcionara con el lenguaje de programación C# ASP.NET Core en su versión 5, esta base de datos se utiliza en el gestor de bases de datos MySQL Workbench de manera local y el despliegue de este proyecto es en AWS.

Para instalar el motor de Base de Datos MySQL Workbench se debe hacer el siguiente paso a paso que está en el siguiente link:

https://support.academicsoftware.eu/hc/es/articles/360007014958-C%C3%B3mo-instalar-MySQL-Workbench

# 12. Otras Consideraciones

Actualmente no hay consideraciones en la documentación del Manual de Configuración de la Base de Datos. A medida que se vaya avanzando en el proyecto y se vaya teniendo una mejor documentación sobre el proyecto se socializa con el grupo de trabajo posibles aspectos faltantes.