MANUAL TECNICO

Estudiantes

Cristina Jaqueline Herrera Rivas

Andrea Hernández Cervantes

Alberto Pablo Torres López

Cesar Daniel Cruz Correa

Rafael Sánchez Quirarte

Universidad Autónoma de la Laguna

Ingeniería en Sistemas Computacionales

Torreón, Coahuila

2022

CONTROL DE VERSIONES

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Descripción | Autor |
| 28/04/2022 | 1.0 | Primera versión manual técnico de software | Andrea Hernández Cervantes  Estudiante Ingeniería en Sistemas Computacionales  Universidad Autónoma de la Laguna |

DESCRIPCIÓN DEL DOCUMENTO

En el presente documento se muestra las técnicas que fueron aplicadas durante la ejecución del proyecto, con el fin de administrar un minado de criptomonedas mediante un bot.

Tener en cuenta que el fin de este manual es documentar las técnicas usadas durante el proyecto, y no como un curso de aprendizaje.

Contenido

[INTRODUCCION 4](#_Toc101867486)

[PRESENTACIÓN 5](#_Toc101867487)

[RESUMEN 5](#_Toc101867488)

[OBJETIVO 5](#_Toc101867489)

[FINALIDAD DEL MANUAL 5](#_Toc101867490)

[1. ASPECTOS TECNICOS 5](#_Toc101867491)

[1.1 HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO 5](#_Toc101867492)

[1.1.1 GITHUB 5](#_Toc101867493)

[1.1.2 VISUAL STUDIO CODE 5](#_Toc101867494)

[1.1.3 MONGO DB 5](#_Toc101867495)

[1.1.4 NPM 5](#_Toc101867496)

[1.1.5 NODE JS 5](#_Toc101867497)

[1.1.6 NODEMON 6](#_Toc101867498)

[1.1.7 TAILWIND 6](#_Toc101867499)

[1.1.8 EXPRESS 6](#_Toc101867500)

[1.1.9 EXPRESS-SESSION 6](#_Toc101867501)

[1.1.10 BCRYPT 6](#_Toc101867502)

[1.1.11 HBS 6](#_Toc101867503)

[1.1.12 PASSPORT 6](#_Toc101867504)

[1.1.13 CHART JS 6](#_Toc101867505)

[2. DIAGRAMAS DE MODELAMIENTO 6](#_Toc101867506)

[2.1 REGISTRAR UN NUEVO USUARIO 7](#_Toc101867507)

[2.1.1 PARAMETRO PARA LA SOLICITUD 7](#_Toc101867508)

[2.1.2 CODIGO 7](#_Toc101867509)

[2.1.3 DIAGRAMA DE FLUJO 7](#_Toc101867510)

[1.1 LOGIN DE ADMINISTRADOR Y USUARIO 7](#_Toc101867511)

[1.1.1 PARAMETRO PARA LA SOLICITUD 7](#_Toc101867512)

[1.1.2 CODIGO 7](#_Toc101867513)

[1.1.3 DIAGRAMA DE FLUJO 7](#_Toc101867514)

[1.2 VER PAGINA DE INICIO POR PARTE DEL ADMINISTRADOR 7](#_Toc101867515)

[1.2.1 PARAMETRO PARA LA SOLICITUD 7](#_Toc101867516)

[1.2.2 CODIGO 7](#_Toc101867517)

[1.2.3 DIAGRAMA DE FLUJO 7](#_Toc101867518)

[1.3 VER PAGINA DE INICIO POR PARTE DEL USUARIO 8](#_Toc101867519)

[1.3.1 PARAMETRO PARA LA SOLICITUD 8](#_Toc101867520)

[1.3.2 CODIGO 8](#_Toc101867521)

[1.3.3 DIAGRAMA DE FLUJO 8](#_Toc101867522)

[2. ASPECTO TECNICO DEL DESARROLLO DEL SISTEMA 8](#_Toc101867523)

[2.1 MODIFICACIÓN LOCAL 8](#_Toc101867524)

[3. REQUERIMIENTOS DEL SOFTWARE 8](#_Toc101867525)

[3.1 REQUISITOS MINIMOS 8](#_Toc101867526)

[4. REFERENCIAS 8](#_Toc101867527)

Contenido de tablas

[Tabla 1 Parámetros para la solicitud- Registro 6](#_Toc101867157)

Contenido de diagramas

[Diagrama 1 Registro 8](#_Toc101867008)

# INTRODUCCION

El manual se realiza con el fin de detallar el “NOMBRE DEL PROYECTO” en términos técnicos para que la persona que vaya a administrar, editar o configurar el software lo haga de una manera apropiada. El documento se encuentra dividido en las siguientes secciones.

Aspectos técnicos: Se darán a conocer conceptos, definiciones de los componentes del sistema desde el punto de vista teórico para mayor entendimiento por parte del lector sobre el funcionamiento del sistema de información y herramientas.

Aspecto técnico del desarrollo del sistema: Corresponde a la instrucción para el lector sobre los componentes del sistema desde una perspectiva técnica en los aspectos de almacenamiento de datos, estructura del desarrollo y recomendaciones del uso debido del sistema.

Requerimientos del software: Detalla los requerimientos básicos necesarios para el funcionamiento del software.

# PRESENTACIÓN

# RESUMEN

# OBJETIVO

# FINALIDAD DEL MANUAL

# ASPECTOS TECNICOS

El “NOMBRE DEL PROYECTO” tiene la finalidad de

## HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO

### GITHUB

### VISUAL STUDIO CODE

### MONGO DB

MongoDB es un **sistema de base de datos NoSQL orientado a documentos de código abierto y escrito en C++**, que en lugar de guardar los datos en tablas lo hace en estructuras de datos BSON (similar a JSON) con un esquema dinámico. Al ser un proyecto de código abierto, sus binarios están disponibles para los sistemas operativos Windows, GNU/Linux, OS X y Solaris.

Esto representa una de las diferencias más importantes con respecto a las bases de datos relacionales. Y resulta que no es que no es necesario seguir un esquema. Los documentos de una misma colección - concepto similar a una tabla de una base de datos relacional -, pueden tener esquemas diferentes.

### NPM

### NODE JS

Node.js es un entorno controlado por eventos, diseñado para crear aplicaciones escalables, permitiéndote establecer y gestionar múltiples conexiones al mismo tiempo.

Las APIs de la biblioteca de Node.js son asíncronas, sin bloqueo. Un servidor basado en Node.js no espera que una API devuelva datos. El servidor pasa a la siguiente API después de llamarla, y un mecanismo de notificación de eventos ayuda al servidor a obtener una respuesta de la llamada a la API anterior.

Un subproceso escalable. Node.js utiliza un modelo de un solo subproceso con bucle de eventos. Gracias al mecanismo de eventos, el servidor responde sin bloqueos, esto hace que el servidor sea altamente escalable comparando con los servidores tradicionales como el Servidor HTTP de Apache.

### NODEMON

### TAILWIND

### EXPRESS

### EXPRESS-SESSION

### BCRYPT

### HBS

### PASSPORT

### CHART JS

# DIAGRAMAS DE MODELAMIENTO

El direccionamiento hace referencia a la determinación de cómo responde una aplicación a una solicitud de cliente en un determinado punto final, que es un URI (o una vía de acceso) y un método de solicitud HTTP específico (GET, POST, etc.).

Cada ruta puede tener una o varias funciones de manejador, que se excluyen cuando se correlaciona la ruta.

La definición de ruta tiene la siguiente estructura:

app. METHOD (PATH, HANDLER)

Donde:

app es una instancia de express.

METHOD es un método de solicitud HTTP.

PATH es una vía de acceso en el servidor.

HANDLER es la función que se ejecuta cuando se correlaciona la ruta.

## REGISTRAR UN NUEVO USUARIO

Crear un nuevo usuario por parte del Administrador.

Método: POST

### PARAMETRO PARA LA SOLICITUD

|  |  |
| --- | --- |
| **nombre**  Localización: body  Nombre del usuario | Requerido  String |
| **apellido**  Localización: body  Apellido del usuario | Requerido  String |
| **email**  Localización: body  Email del usuario | Requerido  String |
| **password**  Localización: body  Contraseña del usuario | Requerido  String |

Tabla 1 Parámetros para la solicitud- Registro

### CODIGO

var express = require('express');

var router = express.Router();

var {client, dbName} = require('../db/mongo');

var User = require('../models/models');

const bcrypt = require('bcryptjs')

/\* GET users listing. \*/

router.get('/', function(req, res, next) {

  res.render('signupAdmin');

});

router.post("/registro", function (req,res,next){

    regUser(req.body)

    .then(()=>{

      res.render('usersAdmin')

    })

    .catch((err)=>{

     console.log(err);

    })

    .finally(()=>{

      client.close;

    })

  });

async function regUser(datos){

    await client.connect();

    const db = client.db(dbName);

    const collection = db.collection('Usuarios');

    const hashUserPassword = await bcrypt.hash(datos.password, 10)

    await collection.insertOne(

      {

        nombre: datos.user,

        apellido: datos.name,

        email: datos.email,

        password: hashUserPassword

      }

    )

   }

module.exports = router;

En la ruta de registro se tiene dos métodos la primera ruta con el método GET para renderizar la vista donde se hará el registro de usuario.

La segunda ruta con el método POST es para obtener los datos de registro, mandados desde la función async *regUser,* donde se hace la conexión a la base de datos y a su colección Usuarios y con el método *insertOne* se insertar los datos que están dentro del objeto {nombre, apellido, email, password}, la contraseña esta encriptada mediante la librería bcrypy de npm.

### DIAGRAMA DE FLUJO

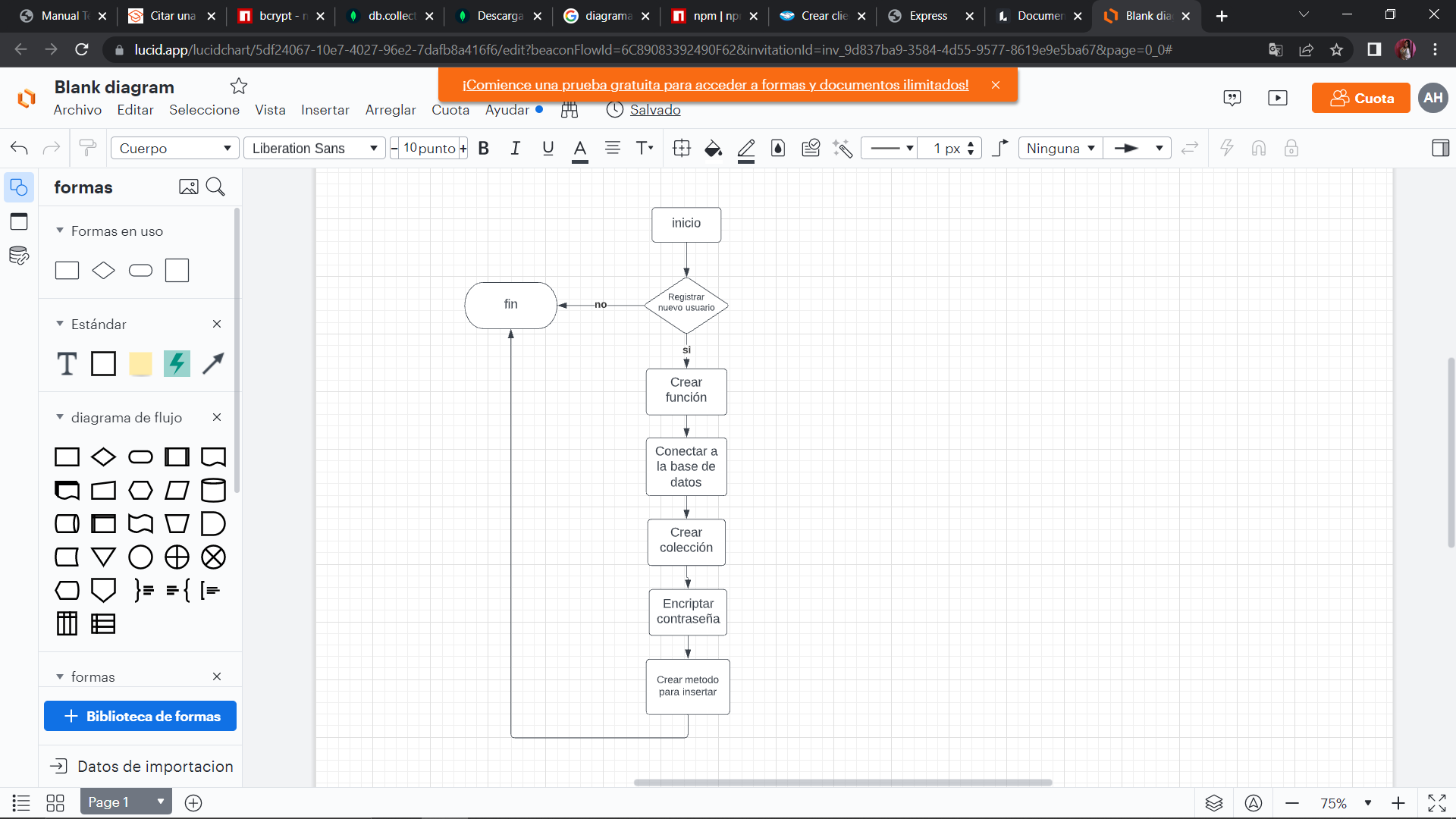


Diagrama 1 Registro

## LOGIN DE ADMINISTRADOR Y USUARIO

### PARAMETRO PARA LA SOLICITUD

### CODIGO

### DIAGRAMA DE FLUJO

Usuario

Interfaz Crypto

Sistema

1. Iniciar Sesión
2. Enviar datos

4.[Usuario correcto]

1. [Bienvenido]
2. Validar usuario []

Diagrama 2 Inicio Sesión

## VER PAGINA DE INICIO POR PARTE DEL ADMINISTRADOR

### PARAMETRO PARA LA SOLICITUD

### CODIGO

### DIAGRAMA DE FLUJO

## VER PAGINA DE INICIO POR PARTE DEL USUARIO

### PARAMETRO PARA LA SOLICITUD

### CODIGO

### DIAGRAMA DE FLUJO

# ASPECTO TECNICO DEL DESARROLLO DEL SISTEMA

## MODIFICACIÓN LOCAL

# REQUERIMIENTOS DEL SOFTWARE

## REQUISITOS MINIMOS

Sistema Operativo: Windows 7

Procesador:

Memoria RAM:

Disco Duro:

Resolución de Pantalla:

Periféricos:

# REFERENCIAS

*What is GitHub*. (s. f.). W3 Schools. Recuperado 25 de abril de 2022, de https://www.w3schools.com/whatis/whatis\_github.asp

MongoDB. (s. f.). *¿Qué Es MongoDB?* Recuperado 25 de abril de 2022, de https://www.mongodb.com/es/what-is-mongodb