UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS

FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMATICA

Escuela Profesional de Ingeniería de Software



MANUAL TÉCNICO

Curso:

Gestión de la Configuración de Software

Docente:

Lenis Wong Portillo

Integrantes:

* Marcos Valdez, Alexander Junior
* Caicedo Porras, José Ricardo
* Navarro Ortiz, Eduardo Gustavo
* Ñontol Lozano, Rafael Santiago
* Noriega Vera, Brandom Marcelo
* Ortega Chavez, Eddy Manuel
* Valentin Ricadi, David Frank

2020

ÍNDICE

[INTRODUCCION 2](#_Toc62542913)

[OBJETIVO DE ESTE MANUAL 2](#_Toc62542914)

[DIRIGIDO 2](#_Toc62542915)

[CONOCIMIENTOS PREVIOS 2](#_Toc62542916)

[PASOS PARA INICIAR UNA PAGINA WEB 3](#_Toc62542917)

[ESPECIFICACIONES TECNINCAS 3](#_Toc62542918)

[HARDWARE 4](#_Toc62542919)

[SOFTWARE 4](#_Toc62542920)

[DESCRIPCION DEL SISTEMA SOE 4](#_Toc62542921)

[DESCRIPCION DE CONTENIDO DEL SISTEMA SOE 5](#_Toc62542922)

[MYSQL 5](#_Toc62542923)

[FUNCIONES PRINCIPALES DEL SISTEMA SOE 6](#_Toc62542924)

[CREACION DEL SISTEMA EN VISUAL STUDIO CODE 8](#_Toc62542925)

[CONEXIÓN DE BASE DE DATOS 9](#_Toc62542926)

[CREACION DEL PATRON MVC 10](#_Toc62542927)

[CREACION DE CONTROLADORES 11](#_Toc62542928)

[INGRESO A CLASES 13](#_Toc62542929)

[DEFINICION 13](#_Toc62542930)

[CODIGO MODEL USER 14](#_Toc62542931)

[INGRESO A VISTAS 14](#_Toc62542932)

[DEFINICION 14](#_Toc62542933)

[EJEMPLO DE VISTA EN EL SISTEMA SOE 15](#_Toc62542934)

[CODIGO DE APP.JS 15](#_Toc62542935)

[CODIGO DE INDEX.JS 16](#_Toc62542936)

[DISTRIBUCION DE ROLES DE USUARIO 16](#_Toc62542937)

[Administrador: 16](#_Toc62542938)

[Usuario: 17](#_Toc62542939)

[Cliente: 17](#_Toc62542940)

# INTRODUCCION

El sistema SOE está compuesto de forma que permita que la información pueda ser agregada, editada y/o modificada por la persona en cargado del departamento de desarrollo.

Para alcanzar estos propósitos se ha hecho de JavaScript que es un lenguaje que se adecua a las nuevas necesidades de las aplicaciones web actuales, sin la necesidad de actualizaciones, a través de esta tecnología la administración y seguridad de la información se dará de forma centralizada y segura de daos, al mismo tiempo facilita la actualización eficiente de dicha información.

En cuando ese manual se ha considerado incluso todos los aspectos técnicos necesarios para el manejo y control del sitio web para Amezaga Code.

# OBJETIVO DE ESTE MANUAL

El objetivo principal de este Manual es ayudar y guiar al técnico a informarse y utilizar herramientas para que el Sistema SOE, implementado por el equipo de Amezaga Code, entre en producción (ejecución), para de esa manera poder hacer uso de la información deseada para poder despejar las dudas existentes y para poder comprender lo siguiente:

* Guía para gestión de herramientas para poner en funcionamiento el sistema SOE.
* Conocer cómo utilizar el sistema, mediante una descripción detallada e ilustrada de las opciones.
* Conocer el alcance de toda la información por medio de una explicación detallada e ilustrada de cada una de las páginas que conforman el manual técnico.

# DIRIGIDO

Este manual está orientado a los técnicos u otros tipos de personal encargado del departamento de desarrollo de Amezaga Code. Solamente dichas personas están autorizadas a realizar modificaciones en el sistema.

Una vez finalizado el proyecto, Amezaga Code está encargado de definir políticas, normas, etc. para la administración del sistema SOE. También a través de este manual el personal podrá estar en la capacidad de supervisar el cumplimiento de políticas, normas, etc. que permitan el correcto funcionamiento de los Sistemas. Definir conjuntamente con los departamentos pertinentes, los contenidos o cambios para el o los sistemas para que también de igual forma puedan ser capacitados en herramientas necesarias para el mantenimiento y ejecución.

# CONOCIMIENTOS PREVIOS

Los conocimientos mínimos que debe tener las personas que operarán las páginas y deberán utilizar este manual son:

Conocimientos básicos acerca Programas Utilitarios

Conocimientos básicos en manejo de plataformas web.

Conocimiento básico de Internet

Conocimiento básico de Windows.

## PASOS PARA INICIAR UNA PAGINA WEB

Deberá seleccionar un navegador que e adecue mejor a sus preferencias, el más común de todos es Google Chrome el cuales es muy usado en todas partes del mundo, pero podemos obtener otras opciones si se desea.



***Imagen 1 Navegadores Web Disponibles***

Espere mientras el Sistema operativo cargue completamente el navegador web, la página de inicio varía de acuerdo al tipo de navegador que se esté usando.

# ESPECIFICACIONES TÉCNINCAS

Para la implementación del Sistema SOE para la preservación y validar la información obtenida en la implementación de la FISI necesitamos lo siguiente.

## HARDWARE

**Cliente Requerido**

El software soporta Microsoft Edge, Chrome, Mozilla, Visual Studio Code, MySQL con sus versiones más recientes.

**Servidor Windows**

* El sistema requiere básicamente todo lo que una instalación de MySQL requiere.
* Se recomienda que se utilicen los requerimientos expuestos anteriormente para la mejor funcionalidad del Sistema SOE.

## SOFTWARE

Base de datos requerida para el sistema es MySQL

El software del servidor es NodeJs, el sistema se puede administrar a través de la Web.

# DESCRIPCION DEL SISTEMA SOE

El sistema constará de toda la información preservada que se maneja en el Departamento de Desarrollo de Amezaga Code, tales como: Relación de alumnos matriculados, clases y tareas ingresadas al sistema, etc.

Contendrá una interfaz atractiva respetando los parámetros de diseño para los usuarios, la cual será aprobada por el Departamento de Gestión de Amezaga Code.

El menú superior constará de las siguientes opciones:

* + Inicio de Sesion
  + Registro para Usuario

En el DashBoard principal constará de las siguientes opciones:

* + Cursos del alumno
  + Tareas del alumno

En el menú lateral tenemos un acceso directo a los enlaces hacia las redes sociales de AmezagaCode y del equipo de desarrollo.

* + Email
  + Facebook
  + Twitter

# DESCRIPCION DE CONTENIDO DEL SISTEMA SOE

Para la realización de este Sistema SOE se basó fundamentalmente en las necesidades que tenía el Departamento de Gestión, aquí pueden preservar toda la información relacionada con su funcionamiento y descripción general.

Uno de los requerimientos del Sistema SOE es que sea fácil de navegar para el usuario en la página web, por este motivo el siguiente sitio tiene las características.

Plataforma Visual Studio Code

* + Imágenes
  + Scripts BD
  + Modulos
  + Plugins
  + MySQL

A continuación, se tiene la información fundamental de cada uno de los aspectos.

## VISUAL STUDIO CODE

Visual Studio Code o simplemente VSC es un potente editor de código fuente para Windows, Linux y macOS, desarrollado por la marca Microsoft. Este editor de código -además de ser multiplataforma- ofrece otras peculiaridades realmente interesantes. Algunas de las más destacadas es que siempre tiene plugins nuevos y actualizados a disposición de los programadores y desarrolladores. Cuenta con la peculiaridad de que su editor de texto es “intellisense”, por lo que se adelanta de manera intuitiva a el texto que se va escribiendo en el editor.

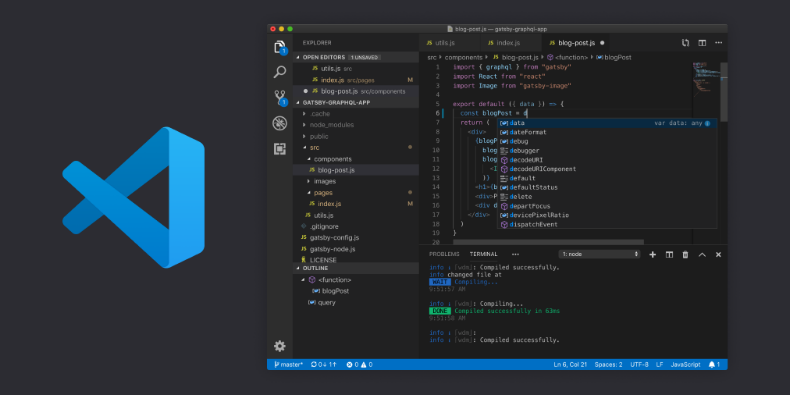
## MYSQL

MySQL es una de las base de datos más importantes actualmente en el mercado. Podemos definir a MySQL, como un gestor de base de datos relacional, multiusario y multihilo. Usada para mover grandes cantidades de información, muchos proyectos en internet como Facebook o Twitter nacieron con este SGBD.

La andadura de esta base de datos comienza en los años 80 con Michael Widenius (Monty), que busca un nuevo sistema de archivos ya que los existentes no le convencían. Posteriormente, con la colaboración de David Axmark, incluye el uso de SQL y en 1995 nace MySQL AB. Actualmente, tras la compra de Oracle a Sun (antiguos propietarios), ha surgido gran polémica acerca de su futuro.

# FUNCIONES PRINCIPALES DEL SISTEMA SOE

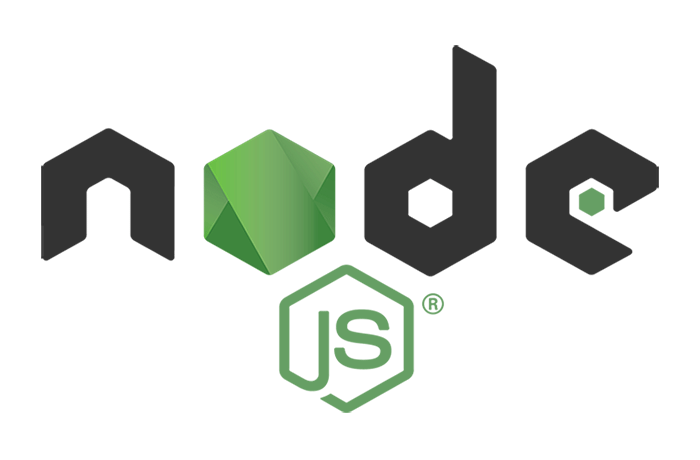
Se requiere instalar los dos programas bases para la ejecución del sistema SOE, entre ellos tenemos a VisualStudioCode, Node Js y MySQL.



***Imagen 2 Visual Studio Code***

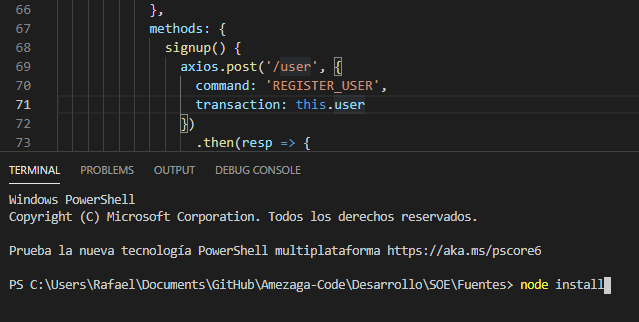


***Imagen 3 MySQL***



***Imagen 4 NodeJs***

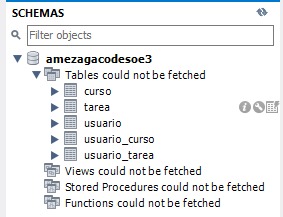
# CREACION DEL SISTEMA EN VISUAL STUDIO CODE



***Imagen 5 Instalación de dependencias NodeJs***

***Imagen 6 Inicialización del Sistema SOE***

# CONEXIÓN DE BASE DE DATOS



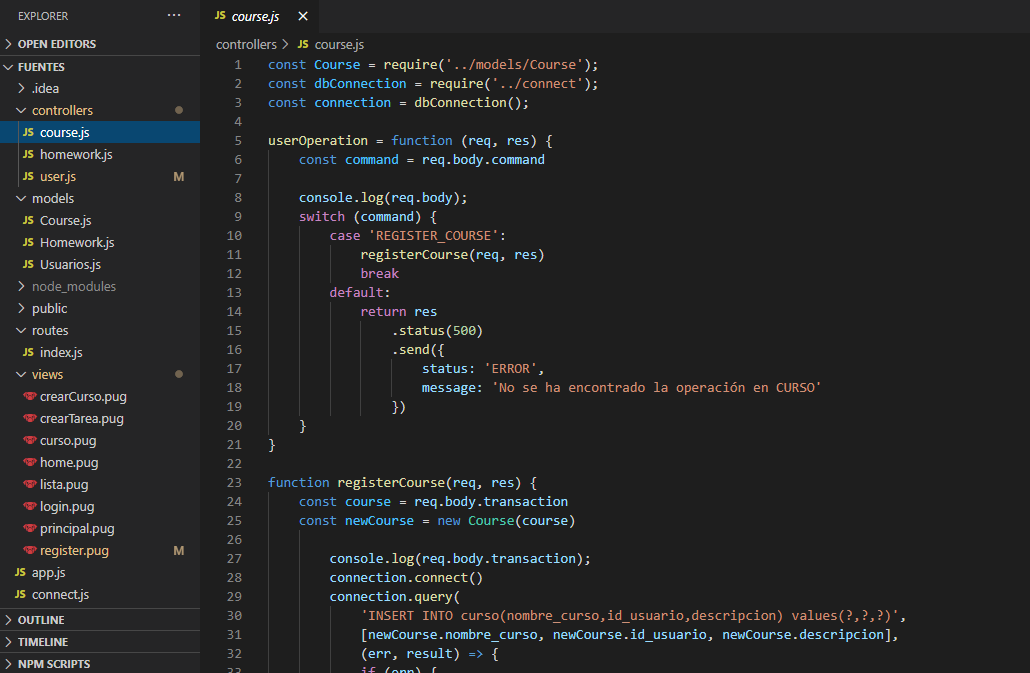
***Imagen 7 Conexión con la base de datos utilizada***

# CREACION DEL PATRON MVC

El Sistema SOE está estructurado con un patrón de diseño de software utilizado para implementar sistemas donde se requiere el uso de interfaces de usuario. Se potencia la facilidad de mantenimiento, reutilización del código y la separación de conceptos.

Su fundamento es la separación del código en tres capas diferentes, acotadas por su responsabilidad:

* + Modelo
  + Vista
  + Controlador



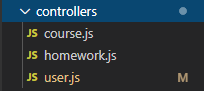
***Imagen 8 Estructura del MVC***

# CREACION DE CONTROLADORES

Es el intermediario entre la Vista y el Modelo, se encarga de controlar las interacciones del usuario en la vista, contiene el código necesario para responder a las acciones que se solicitan en la aplicación.

Si se solicita desarrollar un sistema de administración de usuarios con un CRUD(Create, Read, Update, Delete).

* Pide los datos al modelo y los devuelve de nuevo a la vista para que ésta los muestre al usuario.
* Es decir las llamadas a clases y métodos, y los datos recibidos de formularios.
* La conexión entre la parte gráfica y los datos y los eventos que se producen cuando manejamos la aplicación



***Imagen 9 Estructura del Controlador***

## CÓDIGO CONTROLADOR USER

const Usuario = require("../models/Usuarios");

const bcrypt = require("bcrypt");

const dbConnection = require("../connect");

const connection = dbConnection();

userOperation = function(req, res) {

const command = req.body.command;

switch (command) {

case "REGISTER\_USER":

registerUser(req, res);

break;

default:

return res.status(500).send({

status: "ERROR",

message: "No se ha encontrado la operación",

});

}

};

function registerUser(req, res) {

const user = req.body.transaction;

const saltRounds = 10;

const newUsuario = new Usuario(user);

bcrypt.hash(newUsuario.password, saltRounds).then((hash) => {

connection.connect();

connection.query(

"INSERT INTO usuario values(?,?,?,?)", [

newUsuario.nombre,

newUsuario.email,

newUsuario.password,

newUsuario.createAt,

],

(err, result) => {

if (err) {

return res.status(200).send({

status: "FAILED",

message: err,

});

} else {

return res.status(200).send({

status: "SUCCESS",

message: "Usuario registrado correctamente",

});

}

}

);

connection.end();

/\*.then((res) => {

return res

.status(200)

.send({

status: 'SUCCESS',

message: 'Usuario registrado correctamente'

})

})

.catch((err) => {

return res

.status(200)

.send({

status: 'FAILED',

message: err

})

})\*/

});

}

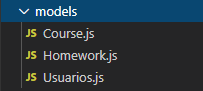
module.exports = userOperation;

# INGRESO A CLASES

## DEFINICIÓN

Contiene todos los atributos de la entidad que vamos a implementar, usar, editar o eliminar. Los procesos de creación están en descritas en el controlador.

Se cuentan con todas las clases que se utilizan en el proceso del proyecto.



***Imagen 10 Estructura del Model***

## CÓDIGO MODEL USER

class Usuario {

constructor(user){

this.nombre = user.nombre,

this.email = user.email,

this.password = user.password,

this.createAt = user.createAt || new Date()

}

}

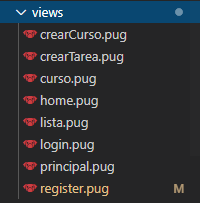
module.exports = Usuario

# INGRESO A VISTAS

## DEFINICION

Programaremos lo relativo a la interfaz gráfica, muestra la información al usuario e interactúa con él. Las vistas requerirán los datos a los modelos y ellas se generarán las salidas, tal como nuestra aplicación requiera

## EJEMPLO DE VISTA EN EL SISTEMA SOE



***Imagen 11 Estructura de las vistas***

## CÓDIGO DE APP.JS

"use strict";

const express = require("express");

const bodyParser = require("body-parser");

const methodOverride = require("method-override");

const cors = require("cors");

const route = require("./routes");

const morgan = require("morgan");

const app = express();

app.use(bodyParser.urlencoded({ extended: true }));

app.use(bodyParser.json({ limit: "50mb" }));

app.set("view engine", "pug");

app.use(morgan("dev"));

app.use(methodOverride());

app.use(cors());

app.use(route);

app.use("/static", express.static(\_\_dirname + "/public"));

app.use(function(req, res, next) {

res.status(404).send("Pagina 404");

});

module.exports = app;

## CÓDIGO DE INDEX.JS

const app = require('./app')

const GlobalEnv = require('./GlobalEnv')

app.listen(GlobalEnv.port, () => {

console.log(`Servidor web escuchando en ${GlobalEnv.host}:${GlobalEnv.port}`)

})

# DISTRIBUCIÓN DE ROLES DE USUARIO

En el sistema encontramos 3 tipos de usuarios con son: administrador, usuario y cliente.

ADMINISTRADOR:

Este usuario realiza tareas de preservar los archivos del sistema, registrar un nuevo usuario, editar información del usuario, asignar acciones y roles al usuario, visualizar a todos los usuarios registrados, acceder al sistema, administrar los archivos del sistema

USUARIO:

Este usuario realiza tareas de acceder al sistema, administrar los cursos que dicta en el sistema, manipula información de los documentos.

CLIENTE:

Este usuario realiza tareas de acceder al sistema y visualiza las tareas asignadas en los cursos que se ha matriculado.