



Manual Técnico Sistema de notas Colegio Salarrué

AUTORES

Mónica Lissette Rivera Mendoza

Christopher Amílcar Cruz Reyes

Carlos Fernando Campos Monterroza

José Carlos Aguilar Linares

Contenido

Introducción	5
1. Descripción general del SNCS	6
2. Base de datos	8
3. Arquitectura de Software	11
4. AdminController	13
4.1 Función inicio	13
4.2 Función gestionUsuarios y función gestionGrados	13
4.3 Función gestionAño	13
4.4 Función creacionAdmin	13
4.5 Función storeAdmin	13
4.6 Función index	13
4.7 Función show	14
4.8 Función showPerfil	14
4.9 Función getAdmin	14
4.10 Función delete	14
4.11 Función update	14
4.12 Función getAdmin	14
4.13 Función restore	14
4.14 Función updateFoto	15
4.15 Función cambiarContraseña	15
4.16 Función getUser	15
4.17 Función cambiarContraseñaAdmin	15
5. AñoController	16
5.1 Función store	16
5.2 Función getPeriodoActivo	16
5.3 Función getAñoActivo	16
5.4 Función terminarPeriodo	16
5.5 Función terminarAño	16
5.6 Función historialAños	16
6. EstudianteController	17
6.1 Función crearEstudiante	17
6.2 Función storeEstudiante	17
6.3 Función index	17

6.4	Función show.....	17
6.5	Función getStudent.....	17
6.6	Función update	17
6.7	Función updateFoto	18
6.8	Función delete.....	18
6.9	Función restore	18
6.10	Función inicio	18
6.11	Función showNotas.....	18
6.12	Función showPerfil.....	18
6.13	Función cambiarContraseñaEstudent.....	18
6.14	Función cambiarContraseñaEstudiante	19
7.	GradoController.....	20
7.1	Función show.....	20
7.2	Función getDetalleGM	20
7.3	Función eliminarDetalleGM.....	20
7.4	Función agregarDetalleGM	20
8.	GrupoController.....	21
8.1	Función show.....	21
8.2	Función agregarNotas.....	21
8.3	Función insertarNotas	21
8.4	Función mostrarNotas.....	21
8.5	Función updateNotas	21
8.6	Función getNota.....	21
9.	LoginController.....	22
9.1	Función welcome	22
9.2	Función guardarPrimerAdmin	22
9.3	Función login.....	22
9.4	Función logout.....	22
9.5	Función passForgot	22
9.6	Función recuperarContraseña	22
9.7	Función recuperacion.....	22
9.8	Función cambioContraseña	23
9.9	Función redirect	23
9.10	Función callback	23

10.	<i>MateriaController</i>	24
10.1	<i>Función index</i>	24
10.2	<i>Función store</i>	24
10.3	<i>Función getMateria</i>	24
10.4	<i>Función update</i>	24
10.5	<i>Función delete</i>	24
10.6	<i>Función restore</i>	24
11.	<i>ProfesorController</i>	25
11.1	<i>Función crearProfesor</i>	25
11.2	<i>Función storeProfesor</i>	25
11.3	<i>Función index</i>	25
11.4	<i>Función show</i>	25
11.5	<i>Función getProfesor</i>	25
11.6	<i>Función getProfesorMateria</i>	25
11.7	<i>Función agregarDetallePM</i>	25
11.8	<i>Función deleteMateria</i>	26
11.9	<i>Función update</i>	26
11.10	<i>Función updateFoto</i>	26
11.11	<i>Función delete</i>	26
11.12	<i>Función restore</i>	26
11.13	<i>Función inicio</i>	26
11.14	<i>Función cambiarContraseñaProfe</i>	26
11.15	<i>Función cambiarContraseñaProfesor</i>	26
11.16	<i>Función showPerfil</i>	27
12.	<i>SeccionController</i>	28
12.1	<i>Función index</i>	28
12.2	<i>Función store</i>	28
12.3	<i>Función show</i>	28
12.4	<i>Función addEstudiantes</i>	28
12.5	<i>Función getProfesoresPorMateria</i>	28
12.6	<i>Función asignarProfesor</i>	28
12.7	<i>Función miSeccion</i>	28
12.8	<i>Función showCuadroNotas</i>	29

Introducción

En el presente manual se definen los aspectos técnicos más relevantes para el adecuado funcionamiento del sitio web Sistema de Notas Colegio Salarrué, además se describen las herramientas y software utilizados para su desarrollo.

El Sistema de Notas Colegio Salarrué, en adelante referido como SNCS, es una aplicación web que puede ser ejecutada en cualquier navegador web, debido a esto, es necesaria la conexión a internet para utilizar el sistema.

Para el desarrollo del SNCS, se utilizó el framework de PHP, Laravel, auxiliándose de diferentes herramientas para complementar el diseño y el funcionamiento de éste, tales como Bootstrap, Javascript, AJAX, JQuery, MySQL a través de phpMyAdmin, Data-Tables, Flatpickr, imask, entre otras.

1. Descripción general del SNCS

SNCS es una aplicación web desarrollada para el Colegio Salarrué, que permite a los administradores del sistema, profesores y estudiantes, la gestión de notas académicas. Incluye tres sitios principales, dependiendo del tipo de usuario registrado:

- Sitio del administrador: en este sitio, los administradores asignados por la institución realizarán la gestión de usuarios, permitiendo agregar usuarios de tipo administrador, profesor y estudiante, así como de la modificación, eliminación y reactivación de éstos. Además, el administrador podrá iniciar y finalizar el año escolar, de igual forma podrá realizar estos eventos para los cuatro periodos que conforman el año escolar. Para cada año, el administrador podrá crear las secciones de los diferentes grados ofertados por el colegio, seleccionando los estudiantes que las conforman, así como los profesores encargados de impartir las materias. El administrador tendrá acceso al historial académico de las secciones de cualquier año registrado. Finalmente, el administrador podrá cambiar las materias que el colegio imparte en los diferentes grados.
- Sitio del profesor: en este sitio, los profesores se encargarán de ingresar las notas de las cinco evaluaciones realizadas por periodo de todos los grupos de clases que se le han asignado. Además, si el profesor es encargado de una sección, tendrá acceso a los registros académicos de sus estudiantes.
- Sitio del estudiante: en este sitio, los estudiantes podrán consultar las notas de todas las materias de los diferentes años cursados en el colegio.

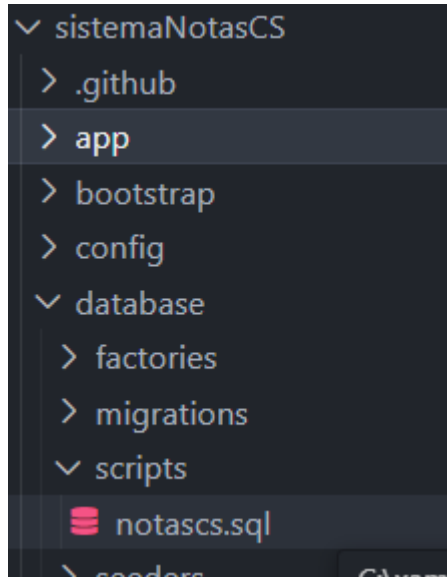
SNCS está diseñado para ser utilizado en cualquier versión de los navegadores actuales, pero se recomienda su uso en Mozilla Firefox. Para entrar al sistema, el colegio será el encargado de implementarlo en el servicio de hosting de su elección, y después compartirán el enlace de la aplicación con sus usuarios.

Para el desarrollo de SNCS éstas fueron las herramientas utilizadas:

- Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.
- XAMPP es un paquete de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl. El nombre es en realidad un acrónimo: X, Apache, MariaDB/MySQL, PHP, Perl.
- MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual: Licencia pública general/Licencia comercial por Oracle Corporation y está considerada como la base de datos de código abierto más popular del mundo.
- phpMyAdmin es una herramienta escrita en PHP con la intención de manejar la administración de MySQL y MariaDB a través de páginas web, utilizando un navegador web.

- HTML, acrónimo en inglés de HyperText Markup Language, hace referencia al lenguaje de marcado utilizado en la creación de páginas web. Este estándar sirve de referencia del software que interactúa con la elaboración de páginas web en sus diferentes versiones.
- PHP es un lenguaje de programación interpretado del lado del servidor y de uso general que se adapta especialmente al desarrollo web.
- Laravel es un framework de código abierto para desarrollar aplicaciones y servicios web con PHP.
- Bootstrap es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web.
- JavaScript es un lenguaje de programación interpretado. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.
- jQuery es una biblioteca multiplataforma de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.
- AJAX, acrónimo de Asynchronous JavaScript And XML, es una técnica de desarrollo web para crear aplicaciones web asíncronas. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano.
- Adicionalmente, se han utilizado herramientas de acceso libre para brindarle al sistema mejor funcionalidad. Se ocupó Data-Tables para brindar todas las características necesarias a las diferentes tablas del sitio, se utilizó flatpickr en todos los inputs de tipo fecha, se utilizó imask js para establecer las máscaras en los inputs donde fuera necesario validar la entrada de datos, se han utilizado diferentes íconos de los ofrecidos por Font Awesome, así como SweetAlert2 para la comunicación necesaria entre el sistema y el usuario.

2. Base de datos



Para la correcta implementación del SNCS, el administrador tendrá que configurar la base de datos según el script “notascs.sql” incluido en los archivos del sistema por los desarrolladores. Dicho script se encuentra en la subcarpeta scriptps dentro de la carpeta database de la carpeta principal llamada sitemaNotasCS.

El script deberá ejecutarse en su totalidad en el gestor de base de datos del servicio de hosting elegido por la institución. Será responsabilidad del administrador asegurarse de tener incluidas en la base del sistema, las 17 tablas creadas por los desarrolladores.

Tabla	Acción						
<input type="checkbox"/> administrador	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> añosescolar	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> detallegradoestudiante	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> detallegradomateria	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> detalleprofesormateria	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> detalleseccionestudiante	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> detalleseccionmateria	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> estudiante	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> etapa	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> evaluacion	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> grado	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> materia	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> nota	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> periodos	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> profesor	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> secciones	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
<input type="checkbox"/> usuarios	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar
17 tablas	Número de filas						

Además, el administrador deberá verificar que las siguientes tablas contengan los registros establecidos en el script de la base de datos.

Tabla grado:

			idGrado	nombreGrado	idEtapa
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	1 Séptimo Grado	1
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	2 Octavo Grado	1
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	3 Noveno Grado	1
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	4 Primer Año	2
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	5 Segundo Año	2

Tabla etapa:







			idEtapa	nombreEtapa
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	1 Tercer Ciclo
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	2 Bachillerato General

Tabla periodos:














			idPeriodo	nombrePeriodo	estado
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	1 Periodo I	0
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	2 Periodo II	0
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	3 Periodo III	0
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	4 Periodo IV	0

Tabla evaluación:

			idEvaluacion	nombreEvaluacion	porcentaje
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	1 Actividad 1	15
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	2 Actividad 2	15
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	3 Actividad 3	20
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	4 Laboratorio	20
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	5 Examen	30

Tabla materias:

<div>← T →</div>				<div>▼ idMateria</div>	nombreMateria	estadoeliminacion
<input type="checkbox"/>		Editar	 Copiar	 Borrar	1 Lenguaje y Literatura	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	2 Matemática	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	3 Ciencia, Salud y Medio Ambiente	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	4 Estudios Sociales y Cívica	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	5 Inglés	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	6 Informática	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	7 Educación Física	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	8 Educación en Valores	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	9 Ciencias Físicas	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	10 Ciencias Biológicas	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	11 Orientación para la Vida	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	12 Seminario	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	13 Emprendedurismo	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	14 Laboratorio de Creatividad	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	15 Dibujo Técnico	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	16 Proyecto de Graduación	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	17 Mercadeo	0
<input type="checkbox"/>		Editar	 Copiar	 Borrar	18 Matemática Pre Universitaria	0

Las tablas grado, etapa, periodo, y evaluación no podrán modificarse durante la utilización del sistema, si el colegio cambiara alguno de estos valores, tendrá que realizarse directo a la base de datos. Por el contrario, la tabla materia si podrá actualizarse desde el sistema. Todo esto correspondiendo a los requerimientos establecidos por la institución.

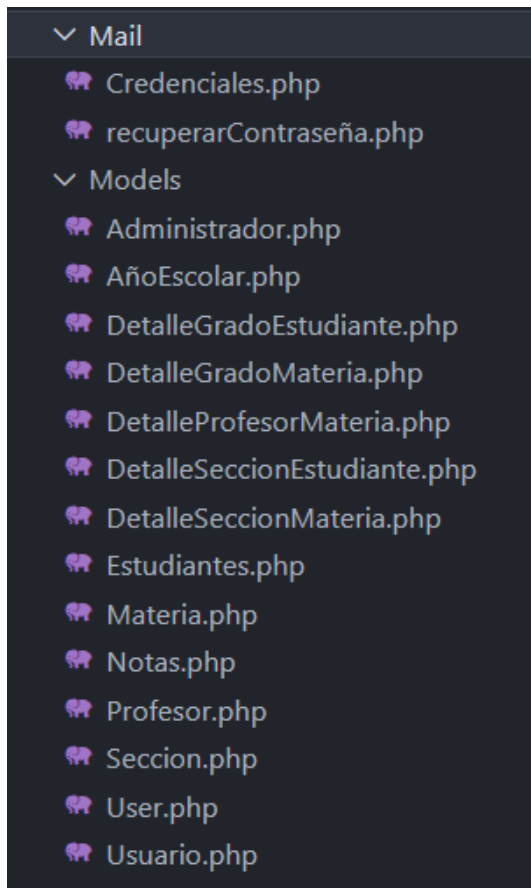
3. Arquitectura de Software

Para el desarrollo del SNCS se ha utilizado el patrón de arquitectura de software Modelo-Vista-Controlador (MVC). MVC es un patrón en el diseño de software comúnmente utilizado para implementar interfaces de usuario, datos y lógica de control. Realiza una separación entre la lógica de negocios y su visualización, proporcionando una mejor división del trabajo y una mejora de mantenimiento.

Las tres partes del patrón de diseño de software MVC se pueden describir de la siguiente manera:

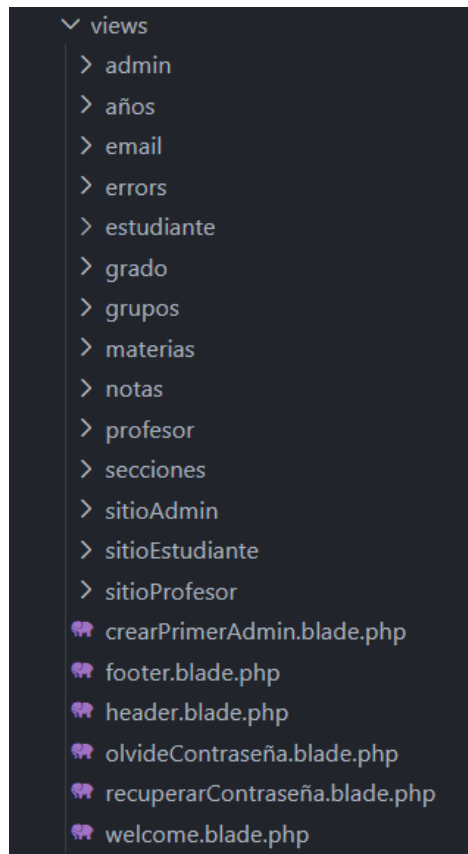
- Modelo: Maneja datos y lógica de negocios.
- Vista: Se encarga del diseño y presentación.
- Controlador: Enruta comandos a los modelos y vistas.

Modelos



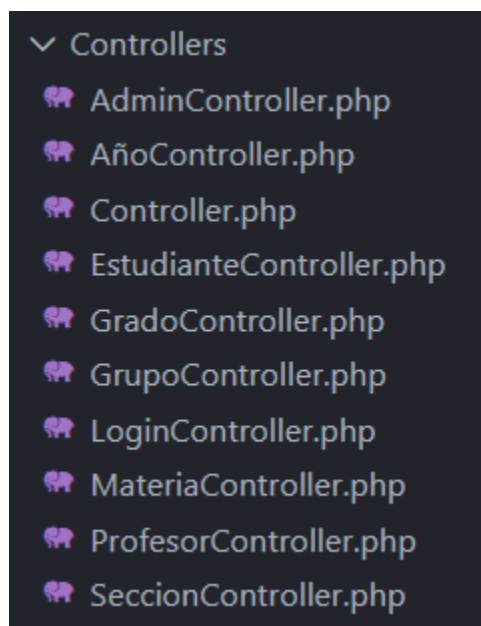
El SNCS cuenta con 14 diferentes modelos utilizados para el manejo de datos y la lógica de negocios, todos están directamente asociados a una de las tablas incluidas en la base de datos. A través de estos modelos, el sistema es capaz de ingresar y consultar datos de la base de datos, entre otras operaciones. Además, se incluyen dos modelos para el envío de correos electrónicos.

Vistas



En total, el SNCS contiene 52 vistas en sus recursos. De estas 52 vistas, seis son generales para todo tipo de usuarios, el resto se encuentra repartidas en 14 diferentes carpetas identificadas según su uso. Si en algún dado caso la institución decide eliminar algún dato mostrado en las vistas tendrá que buscar la vista y realizar modificaciones según lo deseado.

Controladores



El SNCS ocupa 10 diferentes controladores todos nombrados según las funciones que realiza. En cada controlador se encuentran las diferentes funciones diseñadas para resolver las necesidades del sistema, accediendo a y recuperando datos a través de los modelos, para transformar la data y convertirla en información que será mostrada en las vistas.

Además, en el archivo *web.php* de la carpeta routes se encuentran definidas todas las rutas que ocupa el sitio, con una pequeña descripción para mejor identificación de la ruta, en caso se necesite modificar una de las rutas.

Los controladores serán mejor definidos en las siguientes secciones.

4. AdminController

4.1 Función inicio

Función encargada de mostrar la vista de inicio del sitio administrador, devolviendo los datos del administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.2 Función *gestionUsuarios* y función *gestionGrados*

Funciones encargadas de mostrar las vistas con las opciones para la gestión de usuarios y grados respectivamente, verificando que en la sesión exista un usuario de tipo administrador.

4.3 Función *gestionAño*

Función encargada de mostrar la vista con las opciones para la gestión del año escolar, devolviendo variables del año escolar y periodo activo, verificando que en la sesión exista un usuario de tipo administrador.

4.4 Función *creacionAdmin*

Función encargada de mostrar la vista con el formulario para crear un nuevo usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.5 Función *storeAdmin*

Función encargada de insertar un nuevo registro en la tabla administrador, realiza validaciones del formulario y si el ingreso es exitoso, envía un correo al usuario con las credenciales de ingreso al sistema, verificando que en la sesión exista un usuario de tipo administrador.

4.6 Función *index*

Función encargada de mostrar la vista con la tabla que incluye todos los usuarios de tipo administrador activo y eliminados, verificando que en la sesión exista un usuario de tipo administrador.

4.7 Función show

Función encargada de mostrar la vista con la información específica de un usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.8 Función showPerfil

Función encargada de mostrar la vista con la información del usuario de tipo administrador que haya iniciado sesión, verificando que en la sesión exista un usuario de tipo administrador.

4.9 Función getAdmin

Función encargada de retornar un objeto con la información del usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.10 Función delete

Función encargada de establecer el estado de eliminación a 0 de un usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.11 Función update

Función encargada de actualizar los datos de un usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.12 Función getAdmin

Función encargada de retornar un objeto con la información del usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.13 Función restore

Función encargada de establecer el estado de eliminación de un usuario de tipo administrador a 1, verificando que en la sesión exista un usuario de tipo administrador.

4.14 Función updateFoto

Función encargada de la foto de perfil de un usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.15 Función cambiarContraseña

Función encargada de mostrar el formulario para cambiar contraseña, verificando que en la sesión exista un usuario de tipo administrador.

4.16 Función getUser

Función encargada de retornar un objeto con la información del usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

4.17 Función cambiarContraseñaAdmin

Función encargada de cambiar la contraseña de un usuario de tipo administrador, verificando que en la sesión exista un usuario de tipo administrador.

5. AñoController

5.1 Función store

Función encargada de iniciar y crear un nuevo año escolar, iniciando el primer periodo automáticamente, verificando que en la sesión exista un usuario de tipo administrador.

5.2 Función getPeriodoActivo

Función encargada de devolver información del periodo activo, verificando que en la sesión exista un usuario de tipo administrador.

5.3 Función getAñoActivo

Función encargada de devolver información del año activo, verificando que en la sesión exista un usuario de tipo administrador.

5.4 Función terminarPeriodo

Función encargada de terminar los diferentes periodos del año escolar, si se termina el primer, segundo o tercer periodo, inicia el siguiente, verificando que en la sesión exista un usuario de tipo administrador.

5.5 Función terminarAño

Función encargada de finalizar el año escolar, cambiando el grado a todos los estudiantes que cumplieron el requisito para aprobar el año, si el estudiante aprueba el último grado, el sistema establece su estado de finalización a 1, verificando que en la sesión exista un usuario de tipo administrador.

5.6 Función historialAños

Función encargada de mostrar la vista con la tabla para ver el historial de años escolares registrados en el sistema, verificando que en la sesión exista un usuario de tipo administrador.

6. *EstudianteController*

6.1 *Función crearEstudiante*

Función encargada de mostrar la vista con el formulario para crear un nuevo usuario de tipo estudiante, verificando que en la sesión exista un usuario de tipo administrador.

6.2 *Función storeEstudiante*

Función encargada de almacenar un nuevo estudiante en la base de datos, validando los datos ingresados, si el ingreso es correcto, envía un correo al usuario con las credenciales de ingreso al sistema, verificando que en la sesión exista un usuario de tipo administrador.

6.3 *Función index*

Función encargada de mostrar la vista con la tabla con todos los usuarios de tipo estudiantes creados, activos y eliminados, verificando que en la sesión exista un usuario de tipo administrador.

6.4 *Función show*

Función encargada de mostrar la vista con información de un usuario específico de tipo estudiante, verificando que en la sesión exista un usuario de tipo administrador.

6.5 *Función getStudent*

Función encargada de devolver un arreglo con la información de un usuario de tipo estudiante, verificando que en la sesión exista un usuario de tipo administrador.

6.6 *Función update*

Función encargada de actualizar la información de un usuario de tipo estudiante, verificando que en la sesión exista un usuario de tipo administrador.

6.7 Función updateFoto

Función encargada de actualizar la foto de perfil de un usuario de tipo estudiante, verificando que en la sesión exista un usuario de tipo administrador.

6.8 Función delete

Función encargada de establecer el estado de eliminación de un usuario de tipo estudiante a 0, verificando que en la sesión exista un usuario de tipo administrador.

6.9 Función restore

Función encargada de establecer el estado de eliminación de un usuario de tipo estudiante a 1, verificando que en la sesión exista un usuario de tipo administrador.

6.10 Función inicio

Función encargada de mostrar la vista de inicio del sitio estudiante, estableciendo en la sesión las secciones a las que el estudiante a pertenecido en su historial académico, verificando que en la sesión exista un usuario de tipo estudiante.

6.11 Función showNotas

Función encargada de mostrar las notas obtenidas por el estudiante en cada evaluación de cada periodo de cada materia de la sección elegida, verificando que en la sesión exista un usuario de tipo estudiante.

6.12 Función showPerfil

Función encargada de mostrar la vista con la información del estudiante que ha iniciado sesión, verificando que en la sesión exista un usuario de tipo estudiante.

6.13 Función cambiarContraseñaEstudent

Función encargada de mostrar la vista con el formulario para cambiar contraseña de un usuario de tipo estudiante, verificando que en la sesión exista un usuario de tipo estudiante.

6.14 Función cambiarContraseñaEstudiante

Función encargada de cambiar la contraseña de un usuario de tipo estudiante, verificando que en la sesión exista un usuario de tipo estudiante.

7. GradoController

7.1 Función show

Función encargada de mostrar la vista con información del grado especificado, verificando que en la sesión exista un usuario de tipo administrador.

7.2 Función getDetalleGM

Función encargada de devolver información de materia agregada a un grado, verificando que en la sesión exista un usuario de tipo administrador.

7.3 Función eliminarDetalleGM

Función encargada de eliminar materia de un grado, verificando que en la sesión exista un usuario de tipo administrador.

7.4 Función agregarDetalleGM

Función encargada de agregar una materia a un grado, verificando que en la sesión exista un usuario de tipo administrador.

8. GrupoController

8.1 Función show

Función encargada de mostrar la vista con información del grupo especificado, verificando que en la sesión exista un usuario de tipo profesor, y, además, que el profesor esté asignado a dicho grupo.

8.2 Función agregarNotas

Función encargada de mostrar la vista para agregar notas de una evaluación específica, verificando que en la sesión exista un usuario de tipo profesor, y, además, que el profesor esté asignado a dicho grupo.

8.3 Función insertarNotas

Función encargada de insertar notas de una evaluación específica, verificando que en la sesión exista un usuario de tipo profesor.

8.4 Función mostrarNotas

Función encargada de mostrar la vista con las notas de una actividad específica, verificando que en la sesión exista un usuario de tipo profesor, y, además, que el profesor esté asignado a dicho grupo.

8.5 Función updateNotas

Función encargada de actualizar una nota específica, verificando que en la sesión exista un usuario de tipo profesor.

8.6 Función getNota

Función encargada de devolver la información de una nota, verificando que en la sesión exista un usuario de tipo profesor.

9. LoginController

9.1 Función welcome

Función encargada de redirigir al login, si ya existe un usuario de tipo administrador, o en caso contrario, redirigir al formulario para crear el primer administrador.

9.2 Función guardarPrimerAdmin

Función encargada de almacenar la información del primer administrador en la base de datos.

9.3 Función login

Función encargada de dar acceso al usuario, o en su defecto, denegarlo. Dependiendo del tipo de usuario ingresado, la función se encarga de mandar a los diferentes sitios que existen en el sistema. Además, guarda en la sesión información relacionada al usuario.

9.4 Función logout

Función encargada de eliminar de la sesión todas las variables almacenadas durante el uso del sistema, y luego regresa a la vista de login.

9.5 Función passForgot

Función encargada de mostrar la vista con el formulario para crear la solicitud para restablecer la contraseña.

9.6 Función recuperarContraseña

Función encargada de mandar un correo con el enlace para restablecer la contraseña.

9.7 Función recuperacion

Función encargada de mostrar la vista con el formulario para restablecer contraseña.

9.8 Función cambioContraseña

Función encargada de restablecer la contraseña.

9.9 Función redirect

Función encargada de mostrar la vista para seleccionar la cuenta de Google para realizar ingreso al sitio.

9.10 Función callback

Función encargada de realizar ingreso al sitio a través de Google. Realiza las mismas tareas que la función login, brindando acceso dependiendo del usuario autenticado por Google.

10. *MateriaController*

10.1 *Función index*

Función encargada de mostrar la vista con una tabla con todas las materias que la institución oferta, verificando que en la sesión exista un usuario de tipo administrador.

10.2 *Función store*

Función encargada de agregar una materia al sistema, verificando que en la sesión exista un usuario de tipo administrador.

10.3 *Función getMateria*

Función encargada de mostrar información de una materia específica, verificando que en la sesión exista un usuario de tipo administrador.

10.4 *Función update*

Función encargada de actualizar información de una materia específica, verificando que en la sesión exista un usuario de tipo administrador.

10.5 *Función delete*

Función encargada de establecer el estado de eliminación de una materia a 1, verificando que en la sesión exista un usuario de tipo administrador.

10.6 *Función restore*

Función encargada de establecer el estado de eliminación de una materia a 0, verificando que en la sesión exista un usuario de tipo administrador.

11. *ProfesorController*

11.1 *Función crearProfesor*

Función encargada de mostrar la vista con el formulario para crear un nuevo usuario de tipo profesor, verificando que en la sesión exista un usuario de tipo administrador.

11.2 *Función storeProfesor*

Función encargada de almacenar nuevo usuario de tipo profesor en la base de datos, verificando que en la sesión exista un usuario de tipo administrador.

11.3 *Función index*

Función encargada de mostrar la vista con una tabla con todos los usuarios de tipo profesor activos y eliminados, verificando que en la sesión exista un usuario de tipo administrador.

11.4 *Función show*

Función encargada de mostrar la vista con la información de un usuario de tipo profesor específico, verificando que en la sesión exista un usuario de tipo administrador.

11.5 *Función getProfesor*

Función encargada de mostrar la información de un usuario de tipo profesor específico, verificando que en la sesión exista un usuario de tipo administrador.

11.6 *Función getProfesorMateria*

Función encargada de mostrar las materias que el profesor puede impartir, verificando que en la sesión exista un usuario de tipo administrador.

11.7 *Función agregarDetallePM*

Función encargada de agregar una nueva materia para que el docente pueda impartir, verificando que en la sesión exista un usuario de tipo administrador.

11.8 Función deleteMateria

Función encargada de eliminar una materia que el docente imparte, verificando que en la sesión exista un usuario de tipo administrador.

11.9 Función update

Función encargada de actualizar la información de un usuario de tipo profesor, verificando que en la sesión exista un usuario de tipo administrador.

11.10 Función updateFoto

Función encargada de actualizar la foto de perfil de un usuario de tipo profesor, verificando que en la sesión exista un usuario de tipo administrador.

11.11 Función delete

Función encargada de establecer el estado de eliminación de un usuario de tipo profesor a 0, verificando que en la sesión exista un usuario de tipo administrador.

11.12 Función restore

Función encargada de establecer el estado de eliminación de un usuario de tipo profesor a 1, verificando que en la sesión exista un usuario de tipo administrador.

11.13 Función inicio

Función encargada de mostrar la vista de inicio del sitio profesor, estableciendo en la sesión las secciones a las que el profesor ha sido asignado, así como la sección de la que es coordinador, verificando que en la sesión exista un usuario de tipo profesor.

11.14 Función cambiarContraseñaProfe

Función encargada de mostrar la vista con el formulario para cambiar contraseña de un usuario de tipo profesor, verificando que en la sesión exista un usuario de tipo profesor.

11.15 Función cambiarContraseñaProfesor

Función encargada de cambiar la contraseña de un usuario de tipo profesor, verificando que en la sesión exista un usuario de tipo profesor.

11.16 Función *showPerfil*

Función encargada de mostrar el perfil de usuario de tipo profesor, verificando que en la sesión exista un usuario de tipo profesor.

12. SeccionController

12.1 Función index

Función encargada de mostrar la vista con una tabla con todas las secciones creadas por año, verificando que en la sesión exista un usuario de tipo administrador.

12.2 Función store

Función encargada de almacenar una nueva sección al año escolar, verificando que en la sesión exista un usuario de tipo administrador.

12.3 Función show

Función encargada de mostrar la vista con la información de una sección específica, verificando que en la sesión exista un usuario de tipo administrador.

12.4 Función addEstudiantes

Función encargada de agregar estudiantes a una sección, verificando que en la sesión exista un usuario de tipo administrador.

12.5 Función getProfesoresPorMateria

Función encargada de obtener los profesores asignados a las materias de una sección, verificando que en la sesión exista un usuario de tipo administrador.

12.6 Función asignarProfesor

Función encargada de asignar un profesor a una materia de una sección, verificando que en la sesión exista un usuario de tipo administrador.

12.7 Función miSeccion

Función encargada de mostrar la vista con la información académica de una sección, verificando que en la sesión exista un usuario de tipo profesor y que éste sea el coordinador de la sección.

12.8 Función *showCuadroNotas*

Función encargada de mostrar la vista con la información académica de una sección, verificando que en la sesión exista un usuario de tipo administrador.