Módulo de bitácoras grupales

Descripción

De una manera simple, se resume que el módulo bitácoras grupales registrará las actividades vistas en clases. Los registros serán ingresados en colaboración sin la existencia de un tutor o profesor, más bien será autoadministrado entre pares.

Los grupos serán creados por cualquier usuario y será él quien asumirá el rol de administrador del grupo y podrá invitar, a otros usuarios registrados, eliminar, y actualizar cualquier información relacionada.

La bitacoras dividirán su contenido en temas los cuales se abordarán de manera similar a un foro. Por Cada tema visto en clase se creará una nueva sección donde los integrantes del grupo agregaran preguntas que se podrán responder y complementar mediante citas y comentarios. Cabe señalar que cualquier miembro del grupo puede crear el tema, preguntar y responder las veces que sea necesario, pero sólo podrá editar o eliminar el contenido de su autoría.

Tareas realizadas

1. **conexión API**
   Se implementó la API mediante la librería GuzzleHTTP (cliente HTTP para PHP). Para ello se estableció la conexión cliente mediante lla url de la API (HTTPs://noestudiosolo.inf.uct.cl) y la verificación de certificados de seguridad en false debido que laravel denegaba la conexión por temas de seguridad. Esto se puede ver en app\Providers\AppServiceProvider.PHP.
   Además se crearon dos clases PHP la primera GuzzleHTTPRequest en app\Repositories\GuzzleHTTPRequest.PHP que estructura la consulta HTTP y las clase ConexionAPI en app\Repositories\ConexionAPI.PHP la cual contiene las funciones del CRUD que se pueden ver más adelante:

   ```
   public function add($url,$data){ ... }
   public function all($url){ ... }
   public function find($url,$id){ ... }
   public function delete($url,$id){ ... }
   public function update($url,$id, $data){ ... }
   ```

   Variables:
   - $url contiene una cadena de caracteres de la ruta sin considerar base (HTTPs://noestudiosolo.inf.uct.cl). El cuadro siguiente se puede apreciar las rutas y su respectivo método
   - $data es un variable de tipo JSON
   - $id es un identificador de registro de la API

Funciones:

- Las función add agrega un nuevo archivo JSON a la API según la URL que se agregue.
- All: Muestra todos los archivos JSON.
- Find: Muestra el contenido de un solo archivo especificado por la $id.
- Delete: Elimina un archivo JSON especificado por la $id.
- Update: Actualiza los registros a través de una estructura JSON, el cual recibe por parámetros los datos a actualizar.

Rutas API

	get(todo)	**post(agregar)**	**get(uno solo)**	**put(actualizar)**	**delete(borrar)**
Usuario	/usuario	/usuario	/usuario/:_id	/usuario/:_id	/usuario/:_id
Anuncio	/anuncio	/anuncio	/anuncio/:_id	/anuncio/:_id	/anuncio/:_id
Grupo	/grupo	/grupo	/grupo/:_id	/grupo/:_id	/grupo/:_id
Tecnica	/tecnica	/tecnica	/tecnica/:_id	/tecnica/:_id	/tecnica/:_id
Sesion_de_estudio	/sesion_de_estudio	/sesion_de_estudio	/sesion_de_estudio/:_id	/sesion_de_estudio/:_id	/sesion_de_estudio/:_id
Foro	/foro	/foro	/foro/:_id	/foro/:_id	/foro/:_id

Se implementó la base como usar las clases anteriormente mencionado de una manera básica en la cual cuando se guardaba una pregunta, respuesta o comentario siempre se creaba un nuevo archivo JSON.

2. **Diseño**
   Se creó un diseño básico funcional para ordenar el contenido del módulo mediante el uso de plantillas blade, bootstrap, CSS y HTML.

3. **Formularios**
   Se crearon formularios para añadir contenido tales como: preguntas, respuestas, comentarios, citas y grupos con su respectivo controlador y vista.

Tareas pendientes:

1. **Diseño**

   En el apartado de diseño se encontrara un estilo simple y no estandarizado encontrando en cada módulo un estilo diferente.

   1.1. Estandarizar diseño

   Para un uso más intuitivo con una mayor fluidez es de vital importancia la existencia de un único estilo dentro de la web aplicado este a todos los módulos existentes y futuros a implementar. El estilo y base del maquetado general de la página se debe implementar con el uso de blade un motor de plantillas de laravel con la implementación de layouts para reducir el código duplicado.

   1.2. Organización e Interactividad

   La carencia de este factor dentro del módulo de bitácoras grupales genera una mala experiencia en el usuario final no existiendo diferencia de un foro.
   Lo que se busca es generar un punto de inflexión donde los usuarios puedan interactuar en grupos dar opiniones, respuestas, reflexiones sin perder el orden dentro de este.
   Los temas se deben agrupar por semanas y a su vez por unidades, facilitando a la hora de obtener información y distribuir en el diseño del módulo.
   La interactividad se debe implementar para perder la monotonía de la página sin perder la seriedad, invitando a los usuarios a permanecer en ella.
   En el area de preguntas y respuestas se debe reorganizar el contenido para facilitar la lecturas manteniendo un mejor orden.

2. **Funcionalidades**

   2.1. Slider

   En la página principal de bitácoras grupales se implementó un slider el cual debiera mostrar las últimas actualizaciones o nuevo contenido dentro de la sección de preguntas o respuestas.

   2.2. Implementar sesiones

   Se debe implementar sesiones para que solo usuarios registrados tengan acceso al contenido.

   2.3. Privilegios

   Las funcionalidades de eliminar y editar solo pueden ser usadas por el dueño del contenido, para los demás usuarios estarán ocultas.

   2.4. Semanas

   Se debe implementar “crear semana” para organizar los contenidos.

2.5. Unidades

Se debe implementar unidades de contenido para que los usuarios puedan acceder de mas fácilmente al contenido a repasar.

2.6. Material de apoyo

Se debe crear una sección de material de apoyo donde se podrá subir contenido en documentos con posibilidades de descarga, este contenido podría ser agregado en las unidades de materia al cual correspondan.

2.7. Eliminar participantes

El administrador de del grupo de la bitácora tiene las capacidad de eliminar participantes.

2.8. Mostrar archivos JSON

Se debe mostrar el contenido de los JSON correspondiente solo al módulo de bitácoras grupales en las vistas.

2.9. Implementar de manera correcta crud API

Actualmente se probó el CRUD de foro pero su contenido no estaba estructurado, como se ha mencionado anteriormente se guardado preguntas, respuestas, comentarios y citas en foros separados sin tener correlación alguna entre sí, por lo cualse debe crear un foro por cada tema que vaya a crear y agregar el contenido correspondiente como preguntas sus respuestas, comentarios, citas posteriormente de su creación guise por app\HTTP\Controllers\EjForoAPIController.PHP donde muetra la estructura del foro. Además implementar vista para la edición de preguntas, respuestas, comentarios, citas y grupos.

2.10. Ver citas y comentarios

Cambiar el diseño de ver citas y comentarios las cuales deben mostrarse en la sección de preguntas, respuestas y comentarios no en la vista de /semanas.

2.11. Crear cita y comentario

Dentro de la vista de preguntas y respuestas debe estar la opcion de crear cita cada vez que creemos una nueva pregunta, respuesta o comentario.

2.12. Ordenar código

Se recomienda ordenar el código crear un controlador para cada contenido por ejemplo un controlador para preguntas otro para respuestas, etc. por ejemplo GrupoController y GrupoEditController unirlo en un solo y ordenar sus funciones.

Diagramas

Diagrama modular

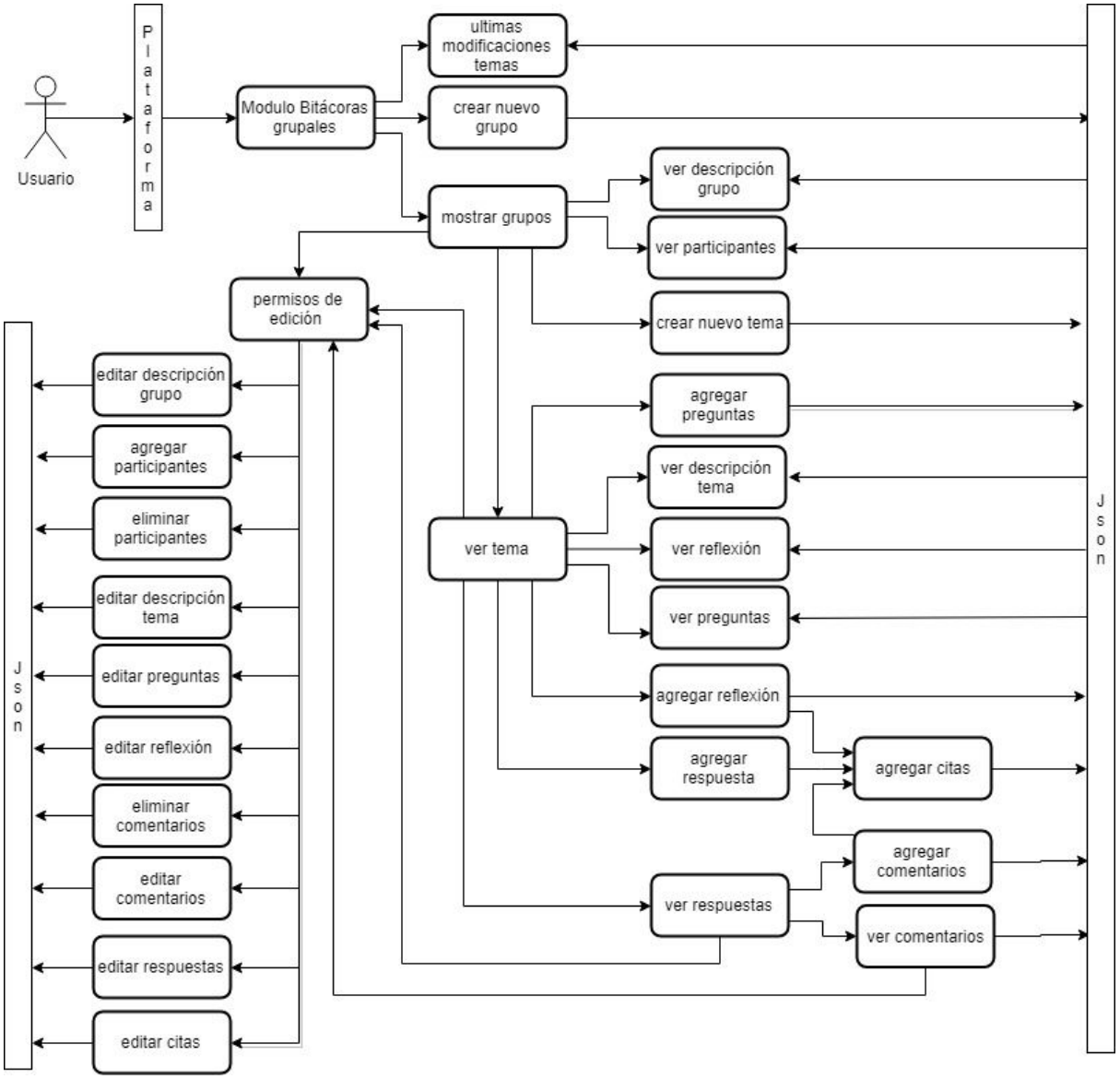

Diagrama de sistema

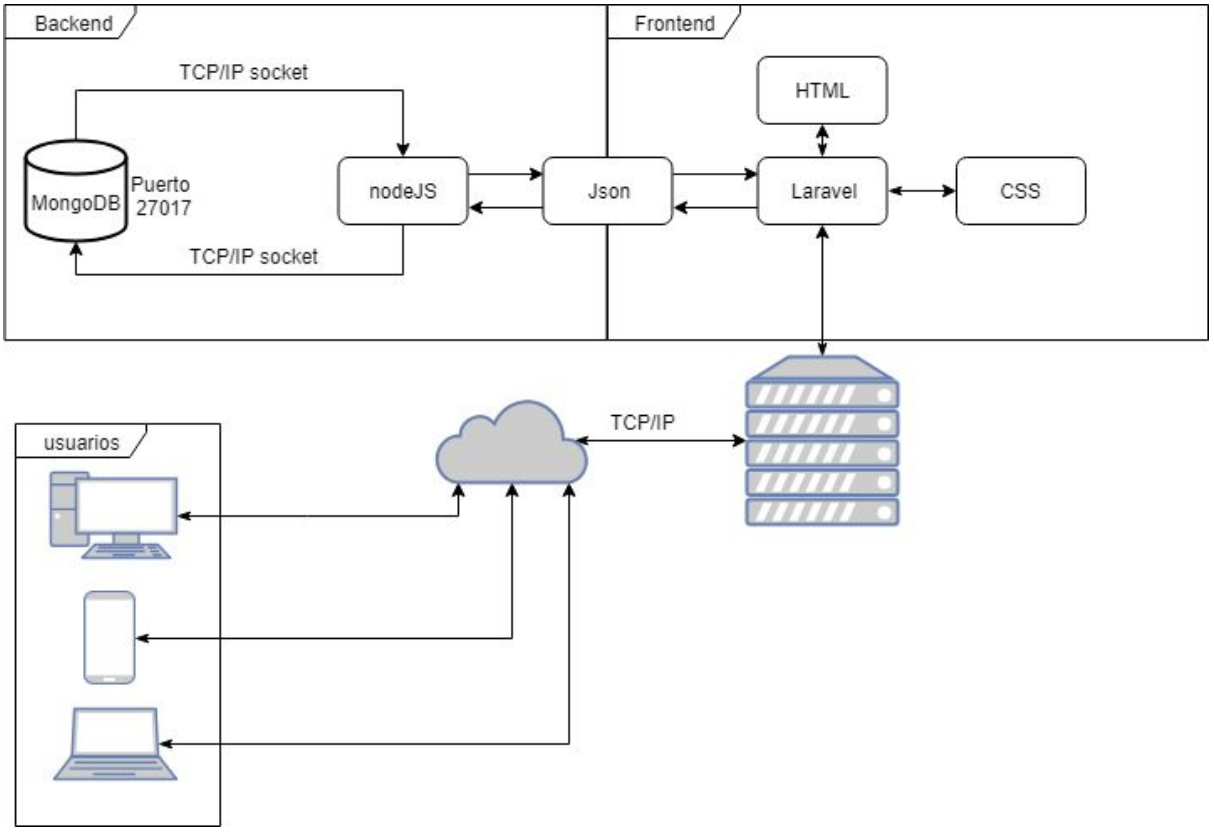

Diagrama de clases backend