



# ATENEA

---

Proyecto de Plataforma de Formación online

*Una aplicación SPA desarrollada en Javascript y PHP*

MÁSTER ESPECIALIZADO EN DISEÑO Y DESARROLLO DE  
APLICACIONES WEB.

MDI

CICE – 2019

**Vicente Alejandro Garcerán Blanco**

## ÍNDICE

Introducción .....	3
Planteamiento .....	4
Estructura .....	6
Estructura general de la aplicación .....	6
Estructura de la base de datos .....	9
usuarios. ....	9
medios. ....	9
tests. ....	9
union_tests_medios. ....	9
imagenes_opciones_tests. ....	10
baterias_tests. ....	10
union_tests_baterias_tests. ....	10
union_alumnos_baterias_tests. ....	10
Arquitectura y código .....	10
Funcionamiento de ATENEA. ....	16
Página del administrador .....	17
Gestión de usuarios .....	18
Gestión de medios .....	19
Gestión de test .....	24
Gestión de Baterías de test .....	28
Gestión de Planes de Formación .....	30
Página del alumno .....	31
Mejoras y desarrollos futuros. ....	33
Conclusiones. ....	34

## Introducción

El objetivo de todo proyecto fin de Máster es recoger y presentar una propuesta en la que se aúnen los conocimientos tratados durante el mismo y que permita demostrar unas habilidades desarrolladas para emplear dichos conocimientos de una forma coordinada.

Eso es precisamente lo que se ha tratado de desarrollar en este proyecto.

Por ello es importante destacar las tecnologías que se han tratado durante el curso las cuales constituyen el núcleo en el que se basa el desarrollo de sitios y aplicaciones web.

Estas aplicaciones se caracterizan porque se ejecutan en un navegador bajo el paradigma cliente-servidor. La información se encuentra alojada en un servidor y el cliente (el navegador del usuario) realiza peticiones de información al servidor. Las peticiones y la interacción con el usuario se componen de:

- Estructura, configurada mediante el lenguaje de marcado HTML 5.
- Aspecto, configurado mediante la tecnología de hojas de estilo en cascada CSS3.
- Interacción entre el usuario y la aplicación mediante un lenguaje del lado del cliente, Javascript.

El servidor procesa las peticiones provenientes del usuario mediante un lenguaje del lado del servidor, PHP y, tras realizar las consultas necesarias a una base de datos, genera y envía un documento al cliente directamente interpretable por el navegador del mismo.

El rápido desarrollo tecnológico que tiene este negocio, y la extensión y complejidad creciente de las aplicaciones y sitios web demandados por clientes y empresas produce una evolución sistemática de las tecnologías involucradas que se manifiesta en el desarrollo de herramientas de mejora de la productividad tales como frameworks, librerías, etc., algunos de los cuales se han transformado de hecho en imprescindibles en el trabajo diario. En este curso se han cubierto algunos de ellos tales como:

- JQuery, librería de desarrollo de Javascript.
- Bootstrap, framework de desarrollo de estilos CSS3.
- Sass, preprocesador de CSS3.

Pese a la capacidad que proporcionan las herramientas cubiertas por el curso, el desarrollo de aplicaciones web actual exige, debido a la enorme complejidad de los proyectos, del uso profesional de frameworks avanzados tanto del lado del cliente (Angular, Vue, React,...), como del lado del servidor en sus diferentes lenguajes (Laravel, CodeIgniter, Symphony, nodeJs, Django, Ruby on Rails,...). Todos estos frameworks se construyen sobre la base de las tecnologías fundamentales cubiertas en el curso y exceden el ámbito del mismo.

El objetivo, por tanto, es desarrollar una propuesta a modo de sitio web o aplicación haciendo uso solamente de las tecnologías básicas y del empleo de programación pura en Javascript y PHP, sin el uso de frameworks profesionales.

## Planteamiento

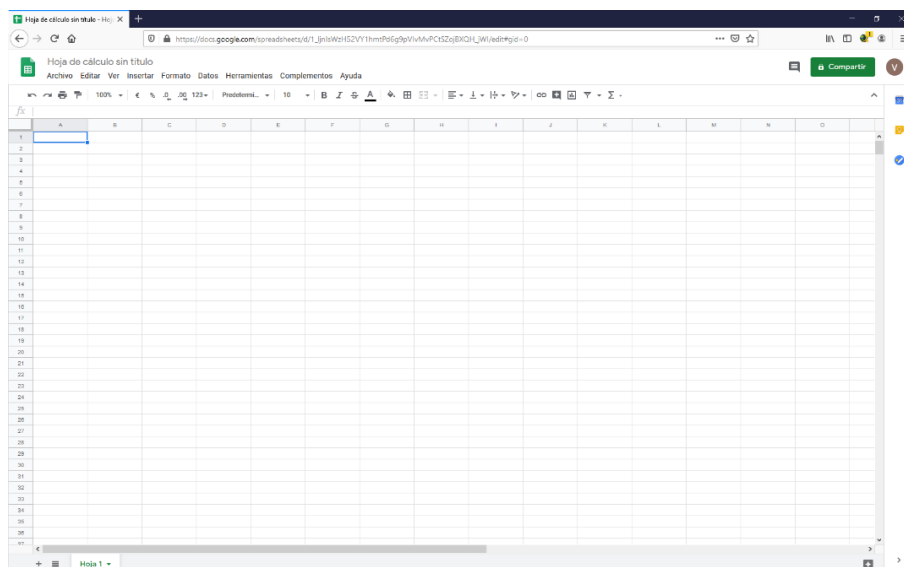
Tal como se ha explicado, el objetivo es desarrollar una solución que haga uso solamente de las tecnologías básicas tratadas en el curso.

En términos generales se pueden considerar dos caminos a tomar a la hora de realizar este desarrollo:

- Implementación de un sitio web dinámico constituido por varias páginas basadas en un tema concreto como pueden ser blogs, tiendas, etc. El usuario navega por el sitio de página en página.
- Desarrollo de una aplicación web. Esta solución implica la programación de una aplicación que corre en el navegador y con la que el usuario interactúa de forma semejante a como lo hace con una aplicación de escritorio o de un dispositivo móvil.

Para este proyecto se ha elegido la segunda opción. El motivo ha sido que permite un ámbito de desarrollo más variado y que, según mi punto de ver, ofrece un reto de programación y diseño mucho mayor que la opción de desarrollo de una colección de páginas separadas que se presenten al usuario.

El paradigma actual de desarrollo de aplicaciones web se basa en las SPA o Single Page Applications (Aplicaciones de una sola página). Estas aplicaciones de cara al usuario no se estructuran como una sucesión de páginas, sino que el flujo de trabajo se ejecuta dentro de la misma página sin solución de continuidad. La información se va solicitando y cargando desde el servidor en tiempo real en función de la interacción con el usuario. Esto proporciona una experiencia de uso semejante a la que se tiene cuando se utiliza una aplicación de escritorio habitual. Un ejemplo claro de esta filosofía de funcionamiento lo constituyen aplicaciones como Google® Docs, por ejemplo. En esta aplicación, podemos hacer uso de una hoja de cálculo semejante a Excel®, pero insertada en el navegador, incluso con un aspecto muy parecido a la aplicación de Microsoft®. Aplicaciones de bancos, agencias de viajes, y un largo etcétera, son soluciones de este tipo.



Hay que tener en cuenta que el desarrollo de una aplicación web necesita un control sobre la misma que se asemeje lo más posible al flujo de trabajo que se sigue en una aplicación usual de escritorio. Es precisamente esa complejidad la que hace que los desarrollos actuales de estas aplicaciones se realicen de forma profesional mediante el uso de frameworks, que se encargan de gestionar la gran complejidad que reside en su diseño, configuración y posterior mantenimiento.

No es, de ninguna manera, por tanto, tarea sencilla lograr desarrollar una aplicación web sólo con el uso de lenguajes de programación “planos” sin uso de ayuda alguna de frameworks. Precisamente por ello, el desarrollo se ha limitado en muchos aspectos, pues abordar la implementación completa de una aplicación web moderna en su totalidad por una sola persona sin el uso de herramientas de desarrollo avanzadas, es una tarea inabordable para el ámbito de un proyecto de este nivel.

La aplicación que he desarrollado se llama **ATENEA** y es una SPA del ámbito de la educación a modo de plataforma de formación.

La idea inicial consistía en el desarrollo de una plataforma de formación completa con capacidad para el desarrollo de elementos de formación a modo de cursos, lecciones y test evaluativos. Estaba prevista la capacidad de diseñar lecciones tanto de contenido teórico como de evaluación. No obstante, la complejidad acumulativa de las tareas a desarrollar para su completo funcionamiento aconsejó limitar el alcance de la plataforma sólo al desarrollo de test evaluativos. Este desarrollo, sin embargo, es lo suficientemente ambicioso como para necesitar un importante esfuerzo de diseño de arquitectura y de programación.

Por ello, **ATENEA**, se configura como una plataforma de desarrollo de test de carácter general que, con las adecuadas adaptaciones y mejoras, podría emplearse perfectamente en empresas del ámbito de la formación, tales como academias de preparación de oposiciones, o autoescuelas, por ejemplo.

Está programada siguiendo el paradigma de SPA o Aplicación de una sola Página, en la que los usuarios navegan por la misma tal como si utilizaran una aplicación de escritorio, pero corriendo en el navegador.

Pese a que no se cubre en el ámbito del Máster cursado, he intentado seguir el patrón **MVC** (Modelo-Vista-Controlador) en la medida de mis posibilidades. Es conveniente resaltar, de nuevo, que los principales frameworks de desarrollo profesional actual se basan, precisamente, en este patrón de diseño y se encuentran, por tanto, perfectamente preparados y estructurados para evitar al programador de la necesidad de “reinventar la rueda” y tener que desarrollarlo. En mi caso, he tratado de separar las vistas del usuario (vistas), del modelo de gestión de la información con la base de datos (modelo) y de la interacción entre el modelo y el usuario (controladores).

Las tecnologías empleadas en su desarrollo han sido:

- HTML 5, para la estructura de las vistas.
- CSS3 para el diseño y capacidades responsive, con Sass como preprocesador.
- Bootstrap fundamentalmente para el grid, las cards y estilos de botones.
- Javascript como lenguaje extensivo del cliente con uso intensivo de la librería JQuery.

- AJAX (Asynchronous Javascript And XML) para las consultas a la base de datos.
- PHP como lenguaje del servidor para el modelo de consulta a la base de datos.

La programación se ha realizado bajo Visual Studio Code como entorno de desarrollo, con plug-in para Sass y para depuración on-line del PHP.

## Estructura

### Estructura general de la aplicación

**ATENEA** es una muestra de plataforma de formación para la generación y ejecución de test multipropósito. Permite la interacción de dos tipos de usuarios:

- *Administradores*: son usuarios con todos los poderes, capacitados para crear contenido (elementos multimedia, preguntas de test y baterías de test), así como para gestionar los usuarios del sistema y asignar formación a los alumnos.
- *Alumnos*: usuarios que ejecutan los planes formativos que les han sido asignados por el usuario administrador.

**ATENEA** permite desarrollar las siguientes operativas:

- Gestionar los usuarios del sistema. Dar de alta a alumnos y administradores, así como alterar sus datos.
- Gestionar los medios para la configuración de contenido. Constituyen los elementos básicos con los que configurar contenido a modo de test. Pueden ser textos, imágenes, vídeos y audios.
- Gestionar las preguntas de test. Permite la configuración de preguntas de test mediante una agregación de elementos básicos antes descritos.
- Gestionar las baterías de preguntas de test. Son colecciones de preguntas antes creadas para la configuración de baterías.
- Gestionar los planes formativos de los alumnos. Asignación de baterías de test ya configuradas a los alumnos dados de alta en el sistema.
- Presentación y ejecución de los planes formativos por parte de los alumnos según las asignaciones realizadas.

Para la gestión de toda esta información **ATENEA** dispone de CRUDs (Create, Read, Update and Delete), para cada grupo de elementos. La información se almacena en una base de datos (*ateneadb*).

La aplicación dispone de un módulo de gestión de login que permite que los usuarios registrados puedan acceder al sistema en función de sus perfiles. En caso de ser usuarios administradores, éstos acceden a la parte de gestión de la aplicación que permite realizar todas las tareas de creación, modificación, asignación y mantenimiento. En el caso de los alumnos, éstos acceden a su propia página en la que se muestran las baterías de test asignadas, listas para su ejecución.

A diferencia de las aplicaciones profesionales habituales, el proceso de registro no lo desarrolla el propio usuario, sino que se considera que se ha de hacer por parte de los administradores en la propia academia. Por ello, inicialmente se proporciona al nuevo usuario

una contraseña maestra (*ATENEA*), que permitirá identificarlo como usuario activo, pero que no permite un acceso al sistema. Para lograr el acceso, el nuevo usuario ha de insertar la contraseña maestra (de carácter público). El sistema lo reconocerá como usuario registrado y comprobará que no ha accedido aún al mismo. En ese momento se le insta a que personalice su clave privada que sustituirá a la maestra en su registro. A partir de ese momento el usuario puede utilizar su clave privada para acceder. Dicha clave solo será conocida por él, de forma que ni siquiera los usuarios administradores la pueden acceder pues se almacena en la base de datos encriptada. En cualquier momento, según su deseo o bien por olvido, el usuario puede solicitar el cambio de su clave privada insertando la clave maestra. En ese caso le será solicitada la dirección de correo de contacto que debió proporcionar durante su proceso de alta y se habilitará un procedimiento de cambio de clave.

**ATENEA** almacena en su base de datos todos los contenidos multimedia que se pueden emplear para crear test que, posteriormente, se asignarán a baterías que podrán ejecutar los alumnos. Estos elementos multimedia básicos reutilizables se denominan medios. Los medios son de cuatro tipos:

- *Textos*: son párrafos específicos que se almacenan como tales y que pueden ser después empleados en la configuración de los enunciados de las preguntas de test. Estos textos son reutilizables por lo que sólo es necesario crearlos la primera vez.
- *Imágenes*: son fotografías o imágenes de un tamaño máximo de 2Mb que se almacenan en formatos jpeg, png, bmp, que se pueden utilizar al igual que los textos en los enunciados de los test.
- *Videos*: también se almacenan con un tamaño máximo de 500Mb y se pueden insertar en los test.
- *Audios*: clips de audio que se utilizan para apoyar los contenidos de los enunciados.

Tal como se puede comprobar, se dispone almacenados en la base de datos de unos repositorios de elementos multimedia reutilizables con los que se pueden configurar enunciados de test complejos, que integren texto, imágenes, vídeos y audios con carácter general. Esto proporciona la flexibilidad completa para crear prácticamente cualquier tipo de enunciado de test por complejo que sea. Es útil, por ejemplo, para crear test de autoescuela en los que se presenten imágenes o vídeos de una situación de tráfico real sobre la que se realice después una pregunta.

Dado que en muchas ocasiones es posible que el texto del enunciado de una pregunta de test sea de un uso único y no reutilizable, (simplemente porque lo que se pregunta no tiene sentido que se vaya a usar en otro test), **ATENEA** permite escribir el texto del enunciado directamente en el test como texto propio y no hacer uso de ningún medio de los explicados con anterioridad. Dicho texto se almacena junto con el propio test y sólo se utiliza en él. Digamos que el texto del enunciado de un test se puede escribir directamente en el propio test o bien incorporarse como uno de los textos almacenados en el repositorio de medios. En el primer caso ese texto sólo existe para ese test y en el segundo se puede reutilizar en otros test. Depende de las necesidades del usuario administrador que lo configura.

Para cada pregunta de test, **ATENEA** dispone de cuatro opciones de respuesta posibles, de las que sólo una será la válida. Para dotar de mayor versatilidad a la configuración, además del texto propio de cada opción, es posible adjuntar una imagen adicional. Esto permite, por

ejemplo, la configuración de preguntas de psicotécnicos, en las que las respuestas sean imágenes y no textos, tales como las típicas preguntas de fichas de dominó o secuencias gráficas. Estas imágenes, al ser bastante específicas de cada test se almacenan por separado de las imágenes del repositorio de medios.

Las preguntas de test se almacenan en un repositorio propio en la base de datos y quedan disponibles para ser insertadas en baterías de test. De esta forma se constituyen todas las baterías que se deseen combinando los test existentes, lo que permite su actualización a lo largo de las sesiones.

Finalmente, las baterías se asignan a los alumnos, lo cual constituye sus planes de formación.

Los alumnos, cuando acceden, pueden comprobar las baterías asignadas, así como su estado de ejecución y su nota. **ATENEA** realiza una calificación automática de los test respondidos pues el usuario administrador consignó las respuestas correctas durante la creación de los test.

Se puede comprobar cómo los contenidos dentro de **ATENEA** se configuran en niveles de agregación progresivos, donde con los medios se configuran test, con éstos se configuran baterías y éstas se asignan a los alumnos. Evidentemente, cualquier eliminación de un elemento en un nivel de forma indiscriminada puede provocar la inconsistencia en la información. Por todo ello, **ATENEA** supervisa el uso de los distintos elementos y no permite operaciones que puedan conducir a la inconsistencia. No me refiero aquí a inconsistencias en la estructura de la base de datos pues eso se ha garantizado en el diseño de la misma haciendo uso de las restricciones sobre las claves foráneas. Las inconsistencias a las que me refiero son:

- No se puede eliminar un medio del repositorio de medios si éste está siendo usado en algún test, pues no se podría localizar dicha información cuando el test se presente.
- No se puede eliminar un test que se haya incluido en una batería por el mismo motivo.
- No se puede eliminar una batería de test que haya sido asignada a un alumno.
- No se puede eliminar la asignación de una batería a un alumno si éste ya ha comenzado su ejecución.

**ATENEA** supervisa estas operaciones para que no se produzcan las inconsistencias anteriores.

En la ejecución de los test, el alumno puede, en todo momento resetear el estado de ejecución del mismo con objeto de recomenzar. Se ha optado por este comportamiento sencillo y en la parte final de desarrollos de mejora se describen opciones más completas a considerar.



## Estructura de la base de datos

**ATENEA** guarda toda la información en una pequeña base de datos MYSQL. En realidad, en la actualidad el motor es MariaDB, que es el resultado de un fork de MYSQL desarrollado para continuar con la filosofía de código libre propio de este motor una vez que fue adquirido por Oracle®. A todos los efectos es completamente transparente al usuario, por lo que su desarrollo y API es idéntica a la original de MYSQL.

A continuación, presento una breve explicación de las diferentes tablas de la base de datos:

### usuarios.

Esta tabla almacena los datos de todos los usuarios del sistema tales como sus nombres, apellidos, nombre de usuario, clave (encriptada), perfil (ADMINISTRADOR, ALUMNO), dirección, código postal, teléfono, fecha de nacimiento, fecha de alta, dirección de correo electrónico y finalmente, una fotografía del mismo.

### medios.

Esta tabla almacena todos los elementos multimedia que se utilizan posteriormente para crear los enunciados (preguntas) de los test. Son de cuatro tipos diferentes: textos, imágenes, vídeos y audios. Se guardan datos como su tipo, descripción, denominación, valor y ruta del archivo.

Habitualmente es necesario utilizar una base de datos para almacenar información relacionada con archivos. En este caso tenemos archivos de imágenes, vídeos y audios. En estos casos, lo habitual es almacenar en las tablas de la base de datos solamente las rutas de acceso a dichos archivos. Los archivos propiamente dichos no se almacenan dentro de la base de datos pues ocupan mucho espacio y la gestión de los mismos no es óptima. Se localizan en unas carpetas del servidor fuera de la base de datos y en las tablas de la misma se almacenan las rutas donde se encuentran dichos archivos. **ATENEA** sigue este procedimiento y por ello se usan rutas para los datos que son archivos. En el caso de la tabla de medios, los textos, sin embargo, sí se almacenan en la propia base de datos bajo el campo valor que es de tipo texto. Para el resto de medios sólo se almacena la ruta. Se ha dispuesto una estructura de carpetas en el servidor para almacenar las imágenes, fotos de usuarios, vídeos, audios e imágenes de las opciones de las respuestas de los test.

### tests.

Tabla para almacenar los datos propios de la configuración de cada pregunta de test tales como el nombre y descripción del mismo, el texto propio del enunciado (optativo), la opción correspondiente a la respuesta correcta y los textos de las respuestas opcionales, así como las referencias a las imágenes de las mismas si se utilizan.

### union\_tests\_medios.

Tabla de unión para la correspondencia muchos a muchos entre la tabla de tests y la de medios. Recoge los campos clave de dichas dos tablas y permite asignar medios a los test. Es necesaria esta tabla pues un test puede contener múltiples medios (textos, imágenes, vídeos, etc.) con los que configura su enunciado. Asimismo, múltiples test pueden utilizar los mismos medios. También guarda el orden de cada medio en la estructura del enunciado del test.

### imagenes\_opciones\_tests.

Tal como se ha indicado previamente, existe la posibilidad de incorporar imágenes como opciones de respuesta de los test. Dichas imágenes se almacenan en una carpeta propia del servidor. Esta tabla recoge las denominaciones y rutas de dichas imágenes. Su campo clave se usa como referencia dentro de la tabla de test para la imagen de cada opción.

### baterias\_tests.

Tabla para almacenar las distintas baterías de test. Simplemente incorpora el campo clave que identifica a cada batería y su nombre y descripción. Los test que componen la batería se recogen en una tabla de unión.

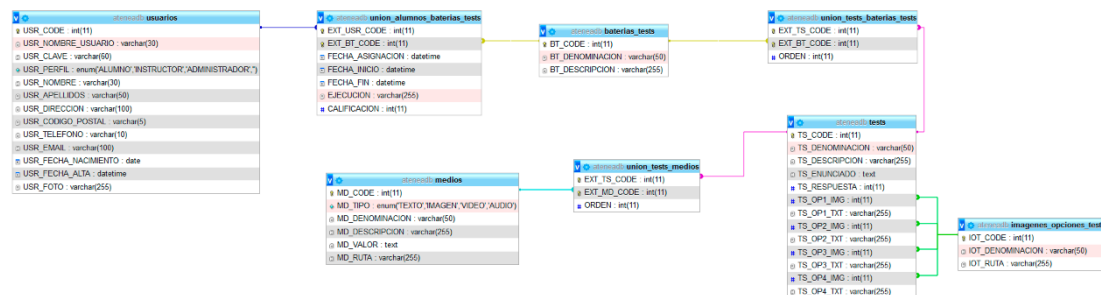
### union\_tests\_baterias\_tests.

En esta tabla se almacena la estructura de cada batería de test. Es una tabla de unión muchos a muchos donde se incluyen los campos clave de las tablas de baterías de test y de test, de tal forma que relaciona todos los test que componen cada batería. Además, incluye el orden de cada test en cada batería.

### union\_alumnos\_baterias\_tests.

Tabla que almacena las asignaciones de baterías a los alumnos. Es una tabla de unión muchos a muchos que incorpora los campos clave de las tablas de baterías de test y alumnos. De esa forma se determinan las baterías asignadas a cada alumno. Además, se incluyen datos propios de cada asignación tales como las fechas de asignación, inicio y finalización de las ejecuciones, la calificación y el estado de ejecución actual.

Como se ve, la base de datos es bastante sencilla y funcional. A continuación, se presenta un esquema de la misma obtenido a partir de la herramienta PHPPMyAdmin.



## Arquitectura y código

**ATENEA** se ha diseñado como una aplicación de única página SPA. Esto significa que, durante la interacción con la misma, no se producen cambios ni recargas de páginas excepto en los casos imprescindibles tales como:

- Procedimiento de login.
- Acceso a la página del alumno.
- Acceso a la página del administrador.

Una vez accedido a la página propia del perfil del usuario (administrador o alumno), no se produce un cambio de página, sino que toda la información se muestra formando parte de un único archivo php en el que se van creando/destruyendo-mostrando/ocultando secciones del mismo. Todos los accesos a información de la base de datos se realizan utilizando la tecnología asíncrona AJAX, que permite evitar las recargas de páginas al evitar el redireccionado a otro archivo php de forma consecutiva. Esto logra la experiencia de uso de una aplicación de escritorio formada por menús, pestañas, controles, etc.

Precisamente, uno de los aspectos más problemáticos del desarrollo ha derivado, precisamente, del empleo de AJAX. Dado que implica una ejecución asíncrona, el flujo de ejecución en una llamada AJAX se bifurca lo que da problemas cuando es necesario esperar a la ejecución de la llamada para completar la tarea correspondiente. Esto obliga a un uso intensivo de funciones callback y a la ejecución anidada de llamadas en las tareas más complejas tal como se puede comprobar tras inspeccionar el código que se suministra. Precisamente la anidación de callbacks puede conducir a lo que se llama “callback hell” o infierno de callbacks cuando se anidan de forma desproporcionada y provoca un grave problema de mantenimiento. En este caso no se ha llegado a ese extremo pues los anidamientos no han sido excesivos. Los frameworks actuales solventan con creces estas circunstancias gracias a su diseño.

He intentado seguir el patrón **MVC** (Modelo Vista Controlador) en lo que se refiere a la estructura de **ATENEA**. Probablemente no sea una implementación completa de dicho patrón, pero sí que se trató de separar la lógica de acceso a los datos en el modelo, el proceso de datos en los controladores y la presentación e interacción con el usuario en las vistas. Los frameworks más utilizados están basados profusamente en este patrón y su implementación en los mismos es completamente nativa con gestión de vistas y controladores por routing, por ejemplo. Implementar este patrón es complejo sobre todo cuanto más compleja es la aplicación a soportar.

La estructura SPA descrita provoca que el grueso de la programación de las vistas se produzca solamente en dos archivos javascript: [codigo\\_AdministradorView.js](#) y [codigo\\_AlumnoView.js](#). Estos archivos contienen todas las funciones necesarias para la ejecución de las distintas tareas. Fundamentalmente se agrupan en funciones de carácter general y manejadoras de eventos.

Dentro de las funciones de carácter general quiero destacar la función Factoría que es un wrapper para aligerar las tareas de creación de elementos del DOM. Permite evitar la escritura recurrente de las órdenes de código javascript plano para el manejo del DOM como crear nodos e insertarlos. De una forma intuitiva se emplea esta función que es capaz de crear un elemento nuevo con toda una colección de atributos evitando engorrosas líneas de código. En **ATENEA** se hace un uso intensivo de dicha Factoría pues tal como he indicado, se modifica continuamente el DOM. Se puede encontrar dicha función al comienzo de los archivos.

Las funciones manejadoras de eventos son declaradas directamente por el evento si su creación es estática o a través de una función propia en el caso de creación dinámica. En algunos de estos casos ha sido necesario tratar con un asunto bastante problemático dentro de la programación en javascript como es el objeto this. Este objeto es una referencia que varía en función del contexto de llamada. Ha habido ocasiones en las que ha sido necesario enviar a un manejador tanto el propio evento como el objeto this en función del contexto.

Aunque durante el curso no se menciona con intensidad el concepto de programación orientada a objetos, he intentado usar objetos en varias circunstancias. Fundamentalmente en

el modelo en el acceso a la base de datos, también he hecho uso de objetos javascript (JSON) en el intercambio por AJAX y de los objetos Array y Map como contenedores.

Respecto al modelo de acceso a la base de datos se ha programado en PHP dentro del archivo [conexionbd.php](#). En él he implementado una clase ConectorBD que se encarga de generar las conexiones con la base de datos y ofrecer todo un set de funciones de interacción con la misma para todas las diferentes operaciones de consulta y modificación que es necesario hacer desde **ATENEA**.

Es cierto que una inspección sobre dicho archivo, así como sobre los controladores [MediaController.php](#) con los que se relaciona, revela que su estructura no es muy eficiente. Se presentan muchas funciones para desarrollar prácticamente tareas similares. Explicaré el motivo de haber tomado esta solución.

Uno de los problemas graves que tiene la interacción de los usuarios con las bases de datos en aplicaciones de interfaz de usuario, reside en la capacidad potencial que pueden tener ciertos usuarios avanzados en el dominio del lenguaje SQL para inyectar sentencias manipuladas que alteren el resultado previsto por el programador. Esta operativa se denomina “inyección SQL”. Consiste en aprovechar la literalidad en la composición de las sentencias por parte del programador despreocupado, para escribir en los campos de formulario proporcionados al usuario bloques de código SQL, con los que se consigue generar una sentencia que proporciona datos no previstos al quedar alterada su estructura. De esta forma podría ser posible acceder a todos los usuarios y datos de éstos e incluso, con sentencias más elaboradas, acceder inclusive a datos maestros del servidor.

Para evitar estos riesgos potenciales, se han desarrollado técnicas propias de creación de las sentencias SQL que no utilizan la generación literal de las mismas. Dichas consultas llamadas consultas preparadas están implementadas en PHP 7 mediante una serie de funciones especiales tales como *prepare*, *bind\_param*, *execute*, *bind\_result*. La utilización de dichas funciones exige la creación de las sentencias de una forma particular y, en concreto, la función *bind\_param* que se encarga de vincular los parámetros pasados como argumento a la sentencia me ha obligado a tener que mantener una función específica para cada consulta pues no he conseguido poder generalizar la construcción de las sentencias a partir de parámetros diferentes. Basta con que cambie la sentencia que se modifica la vinculación y ya no sirve la misma función.

Una tarea de mejora sería investigar una forma de realizar todas estas llamadas de manera más genérica usando un grupo reducido de funciones en lugar de tantas.

Otro asunto que deseo destacar y que me ha provocado no pocos quebraderos de cabeza durante el diseño es el de la gestión del historial del navegador. En una aplicación de escritorio nativa el programador tiene el control sobre la misma en todo momento. No me refiero a que controle el flujo de ejecución del programa puesto que éste depende en cada momento de las actuaciones del usuario. Me refiero a que el ámbito de ejecución siempre está controlado. Por ejemplo, en una aplicación como Photoshop®, el usuario en cualquier momento puede hacer cualquier acción que el programa le proporcione y, desde ese punto de vista, el programador no sabe lo que puede terminar haciendo el usuario. No obstante, sí que tiene control sobre todo lo que se puede hacer en cada momento y tener control sobre ello.

En una aplicación que se ejecuta en un navegador no sucede lo mismo. El programador no controla al navegador (o al menos no se me ocurre cómo), de tal forma que la manipulación

por parte del usuario del historial (adelante, atrás) del navegador puede dar al traste con el flujo de las operaciones diseñadas por el programador. Al pulsar hacia adelante o hacia atrás, se provoca la recarga de otras páginas previas o posteriores. En un sitio web o un blog, no supone demasiado problema. Simplemente sería como si el usuario revisara una revista y decidiera leer páginas hacia adelante o hacia atrás. No obstante, en una aplicación web que persigue comportarse como una aplicación de escritorio, la manipulación del historial puede provocar que el usuario abandone una tarea y se cambie a otra página sin sentido.

Evitar esto es una de los motivos por los que se me apareció la filosofía SPA como indispensable pues, al estar toda la funcionalidad incluida en una única página, evitaba el problema de la manipulación del historial pues no existe una página anterior (excepto el login) o posterior.

El login precisamente se gestiona con el empleo de sesiones.

La estructura de carpetas de **ATENEA** es:

- **Controllers:** contiene los archivos php que se comunican con la base de datos.
  - [\*LoginController.php\*](#). Archivo del controlador del login. Se encarga de ordenar al modelo las acciones para la gestión del acceso.
  - [\*UsuariosController.php\*](#). Archivo del controlador para la gestión de los datos de los usuarios.
  - [\*MediosController.php\*](#). Archivo del controlador general de **ATENEA**. Contiene todas las interacciones no sólo para los medios sino para todas las operaciones con la base de datos.
- **Models:** contiene el archivo [\*conexionbd.php\*](#) del objeto conexión con la base de datos.
- **Views:** contiene los php maestros de la aplicación para las páginas del administrador y del alumno. Contiene todos los elementos de estructura estáticos.
  - [\*AdministradorView.php\*](#).
  - [\*AlumnoView.php\*](#).
  - [\*Favicon.ico\*](#). Icono de la aplicación.
- **js:** contiene el código javascript de la aplicación.
  - [\*codigo\\_AdministradorView.js\*](#). Archivo de javascript para la página del administrador.
  - [\*codigo\\_AlumnoView.js\*](#). Archivo de javascript para la página del alumno.
  - [\*codigo\\_login.js\*](#). Archivo de javascript para la página de login.
  - [\*DlgUsuario.js\*](#). Archivo de javascript para el diálogo de datos del usuario.
- **css:** archivos css de estilos. Son creados por procesamiento de los correspondientes archivos de sass.
- **scss:** archivos sass para el preprocesador de estilos. He usado un plugin de Visual Studio Code® en lugar de Koala pues está integrado en el Visual.
  - [\*\\_globales.scss\*](#). Archivo no compilable de sass con las definiciones y elementos globales que se incluyen en el resto de archivos. Define colores, fuentes, puntos de corte del responsive y un mixin para presentaciones de las tablas.
  - [\*estilos.scss\*](#). Archivo de gestión de los estilos de la página del administrador.
  - [\*estilos\\_alumno.scss\*](#). Archivo de gestión de los estilos de la página del alumno.

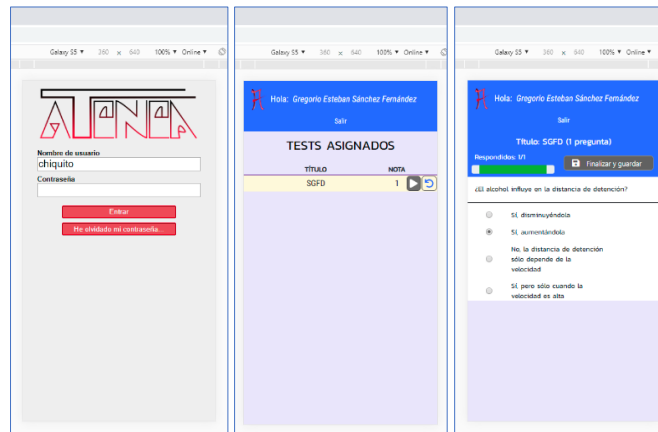
- [\*estilos\\_barra\\_nav.scss\*](#). Archivo de gestión de los estilos de las barras de navegación de ATENEA.
- [\*estilos\\_formulario\\_usuario.scss\*](#). Archivo de gestión de los estilos del diálogo de datos de los usuarios.
- [\*estilos\\_login.scss\*](#). Archivo de gestión de los estilos de la página de login.
- [\*nav-reset.scss\*](#). Archivo de reseteo de algunos parámetros de estilos del navegador.
- **php**: contiene los archivos php adicionales para tareas accesorias.
  - [\*abrirSesion.php\*](#). Archivo para abrir la sesión tras login.
  - [\*datos\\_conexion.php\*](#). Archivo que contiene las variables genéricas de conexión como el nombre del servidor, base de datos, usuario y clave de acceso al servidor.
  - [\*DlgUsuario.php\*](#). Archivo de estructura del diálogo de datos del usuario.
  - [\*eliminarFoto.php\*](#). Archivo para eliminar una foto del usuario del servidor.
  - [\*eliminarImagen.php\*](#). Archivo para eliminar una imagen de la carpeta del servidor.
  - [\*eliminarImagenOpcion.php\*](#). Archivo para eliminar una imagen de opción de respuesta de test de la carpeta del servidor.
  - [\*logout.php\*](#). Archivo para realizar el logout o salida de **ATENEA** y cerrar la sesión.
  - [\*subirAudio.php\*](#), [\*subirFoto.php\*](#), [\*subirImagen.php\*](#), [\*subirImagenOpcion.php\*](#), [\*subirVideo.php\*](#). Archivos para ordenar la subida de los distintos medios a sus carpetas respectivas en el servidor.
- **media**: carpetas de almacenamiento en el servidor de los medios usados como elementos básicos. Además, tiene los archivos de las imágenes de los iconos y logo de **ATENEA**. Tiene las carpetas:
  - *Audio*: almacén para los archivos de audio.
  - *Imágenes*: almacén para las imágenes.
  - *Videos*: almacén para los archivos de vídeo.
  - *Tests*: almacén para las imágenes particulares de los test. Como he dicho antes, dentro de los test existe la posibilidad de que las respuestas optativas contengan imágenes. Estas imágenes he considerado que son muy específicas y que no tiene sentido que se almacenen en el repositorio general de imágenes pues difícilmente serán reutilizables. Por eso, estas imágenes se almacenan en una carpeta diferente separada de las imágenes que efectivamente puedan ser de uso más general.
  - *Usuarios*: almacén para las fotos de los usuarios.
- **fontawesome**: carpeta de la librería gratuita de recursos fontawesome. Se ha usado para algunos iconos.

Respecto al carácter responsive de **ATENEA** he de hacer la siguiente consideración. Las operaciones relacionadas con la gestión del sistema, propias de los administradores, considero que no se ajustan a ser desarrolladas en dispositivos móviles tales como tablets o smartphones. Son operaciones complejas en las que hay que manipular bastante información en pantalla y aunque se pueda configurar una respuesta responsive, no resultaría en una correcta usabilidad. Por ello he restringido la resolución de dicha página del administrador hasta los 1200px como mínimo pues por debajo, no operaría correctamente.

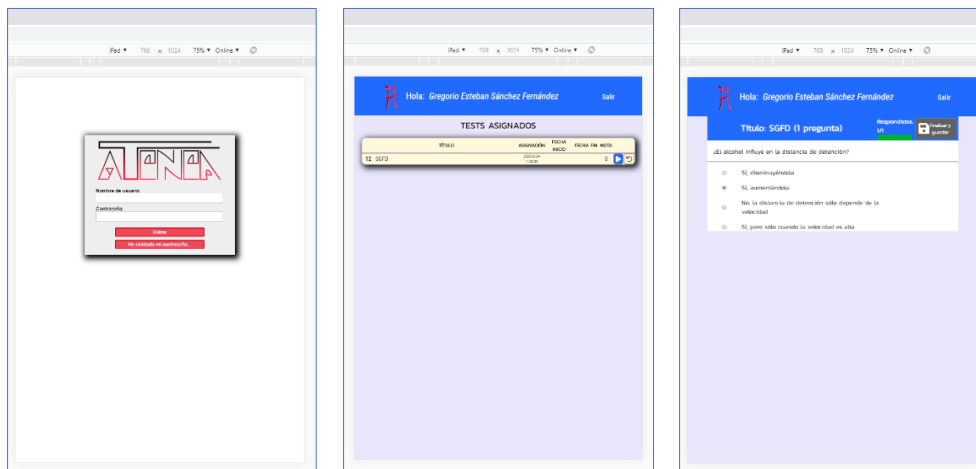
Otra cosa son las páginas de login y del alumno. Estas sí las he configurado responsive con tres puntos de corte o breakpoints:

- Estándar de smartphone: 360px.
- Estándar de tablet: 768px.
- Estándar laptop y superior (desktop): a partir de 1200px.

Tanto el login como la página del alumno sí es posible que se puedan ejecutar en dispositivos móviles por lo que he habilitado un diseño responsive para dichos puntos de corte.



Modelos responsive para smartphone



Modelos responsive para tablet

He de hacer una consideración con respecto al diseño responsive. Durante el curso se explica que mediante el uso de las media queries, dentro de CSS3, se consigue configurar diferentes diseños dependiendo del dispositivo de visualización. Esto se entiende muy bien en el caso de páginas estáticas o cuya estructura esté determinada. No obstante, cuando los elementos de la página se crean dinámicamente aparece una complicación añadida al realizar un diseño responsive porque hay ocasiones en las que dicho diseño no se refiere tanto a cambiar la estructura visual de los elementos, sino que puede ser que ciertos elementos se creen de otra forma o con otra estructura diferente dependiendo del dispositivo. Esto es lo que me ha sucedido en **ATENEA**. No es tanto que haya que cambiar la visualización de ciertas columnas de las tablas o modificar tamaños de letra o anchos de contenedores. Se trata más de que un

determinado diseño de contenedores debe ser distinto dependiendo de la resolución. Ello me ha obligado a hacer operaciones de diseño responsive desde el propio lenguaje javascript mediante el uso del objeto matchMedia. Aunque lo he utilizado, esta solución trasciende a lo tratado durante el curso en lo que se refiere al diseño responsive.

## Funcionamiento de ATENEA.

Tras acceder a **ATENEA** a través del archivo *login.php* se muestra la pantalla de acceso.

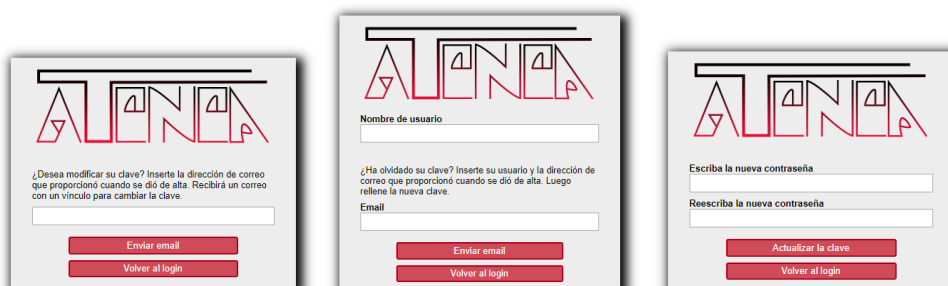
La imagen muestra la interfaz de usuario para el acceso a ATENEA. En la parte superior, hay un logo con el nombre 'ATENEA' en un estilo de letra geométrica. Debajo del logo, hay dos campos de entrada: 'Nombre de usuario' y 'Contraseña'. Debajo de estos campos, hay dos botones: 'Entrar' y 'He olvidado mi contraseña.'.

**ATENEA** no se ha diseñado con procedimiento de auto registro, por lo que todo usuario que pretenda acceder ha de haber sido dado de alta previamente por los usuarios administradores.

El procedimiento de login está vinculado a la contraseña maestra *ATENEA*. Dicha contraseña es de dominio público y no se puede acceder con ella al sistema. Se asigna a cada usuario cuando se le da de alta y permite reconocer si el usuario ha accedido alguna vez y si es un usuario registrado.

La primera vez que un usuario accede lo debe hacer con la contraseña maestra. El sistema lo detectará e informará de la necesidad de personalizar la misma para lograr un acceso privado. Una vez cambiada el usuario ya podrá hacer uso de la misma para acceder según su perfil.

Bien sea por conveniencia o por olvido, la clave siempre se puede cambiar. Para ello o bien se pulsa en el botón “*He olvidado mi contraseña...*” o bien se introduce la clave maestra como clave de acceso. El sistema lo detectará y, en cualquier caso, solicitará al usuario que inserte la dirección de correo electrónico que proporcionó cuando se dio de alta.

La imagen muestra tres pantallas de recuperación de contraseña. La primera pantalla pregunta si el usuario desea modificar su clave y solicita la dirección de correo electrónico que proporcionó al darse de alta. La segunda pantalla pregunta si el usuario ha olvidado su clave y solicita su usuario y la dirección de correo electrónico que proporcionó al darse de alta. La tercera pantalla solicita que el usuario escriba la nueva contraseña y la reescriba. Cada pantalla tiene botones para 'Enviar email' y 'Volver al login'.

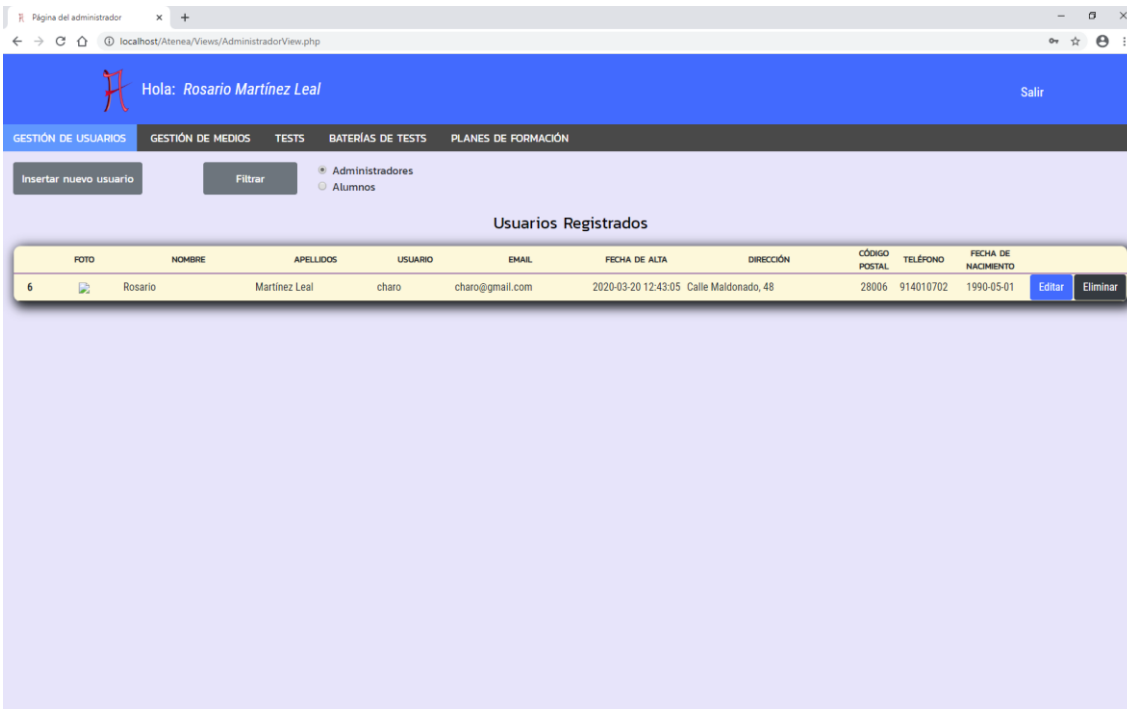


Tras comprobar que dicha cuenta está asociada al usuario, los sistemas habitualmente envían un correo a dicha cuenta con un link que permite el acceso para la modificación de la clave. En este caso no me ha resultado posible hacer eso porque para que funcione el link de vuelta es necesario hacer referencia desde el exterior al servidor. Esto es un dato proporcionado por el hosting, pero en este caso, el servidor Apache es local y no he visto cómo puedo acceder a él desde el exterior. Ante este inconveniente, he optado por comprobar la corrección de la dirección de correo introducida y en caso correcto, simplemente se ofrece al usuario los controles para que cambie la clave.

Una vez comprobados los datos de acceso, en función de su perfil, se accede a la página propia del mismo.

### Página del administrador

Una vez accedido al sistema con perfil de administrador aparece la página de administración de la aplicación. En esta página se pueden desarrollar todas las tareas de gestión tanto de usuarios como de elementos de formación.



The screenshot shows the ATENEA administrator interface. At the top, a blue header bar displays a greeting "Hola: Rosario Martínez Leal" and a "Salir" link. Below the header is a navigation menu with options: "GESTIÓN DE USUARIOS", "GESTIÓN DE MEDIOS", "TESTS", "BATERÍAS DE TESTS", and "PLANES DE FORMACIÓN". The "GESTIÓN DE USUARIOS" option is selected. Below the menu, there are buttons for "Insertar nuevo usuario" and "Filtrar", along with radio buttons for "Administradores" (selected) and "Alumnos". The main content area is titled "Usuarios Registrados" and contains a table with the following data:

FOTO	NOMBRE	APELLIDOS	USUARIO	EMAIL	FECHA DE ALTA	DIRECCIÓN	CÓDIGO POSTAL	TÉLEFONO	FECHA DE NACIMIENTO	
6	Rosario	Martínez Leal	charo	charo@gmail.com	2020-03-20 12:43:05	Calle Maldonado, 48	28006	914010702	1990-05-01	<a href="#">Editar</a> <a href="#">Eliminar</a>

En el margen superior de la página se muestra un saludo con el nombre del usuario que ha accedido. En el extremo derecho se muestra un link para salir de la aplicación cerrando la sesión.

Justo bajo esta área de encabezado aparece el menú principal de **ATENEA**. En este menú se despliegan las opciones de las diferentes subpáginas que permiten realizar las tareas de:

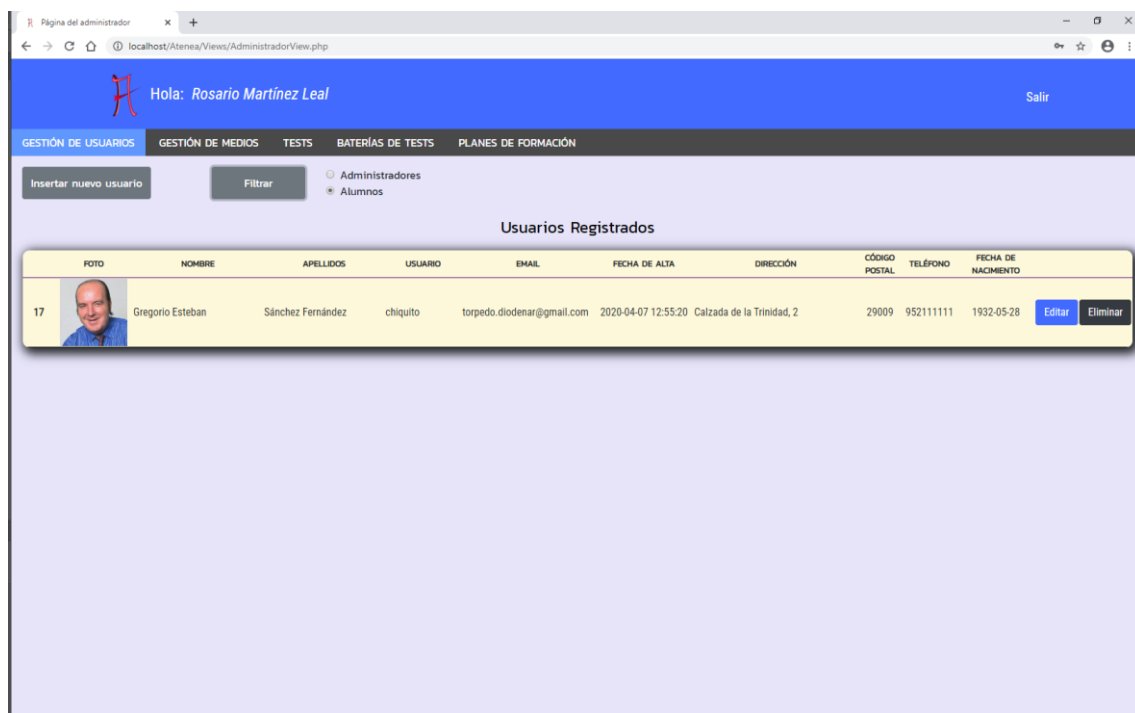
- Gestión de usuarios.
- Gestión de medios.

- Gestión de test.
- Gestión de baterías de test.
- Gestión de planes de formación.

### Gestión de usuarios

Una vez pulsada la entrada de menú de usuarios se muestran en una tabla todos los usuarios administradores del sistema.

En la parte superior se presenta un filtrado mediante dos check boxes para seleccionar la consulta sobre los usuarios administradores o alumnos.



La gestión de usuarios permite realizar todas las operaciones necesarias para realizar el mantenimiento de los usuarios del sistema. En concreto se pueden realizar todas las opciones propias de un CRUD:

- Crear y dar de alta nuevos usuarios.
- Editar los datos de los usuarios.
- Consultar los datos de los diferentes usuarios (excepto las claves que son privadas).
- Eliminar usuarios del sistema.

Tanto en la creación de nuevos usuarios como en la edición de los datos de uno existente, se muestra un diálogo modal en el que se pueden insertar y modificar los datos correspondientes excepto la clave que es privada del usuario. En el caso de creación de un nuevo usuario la clave inicial que se le asigna es la maestra **ATENEA** de uso público para que sea personalizada durante el primer acceso.

Insersión - edición de usuario

localhost/Atenea/php/DlgUsuario.php

Nombre de usuario:

Administrador  
Alumno

Email:

Nombre:

Apellidos:

Dirección:

Fecha de nacimiento: dd/mm/aaaa

Código Postal:

Teléfono:

Insertar Cancelar

La fotografía del alumno se puede cargar haciendo clic sobre el botón correspondiente y seleccionando el archivo de la misma. Será almacenada en una carpeta del servidor y su nombre y ruta en un campo de la base de datos. Si no se dispone de fotografía, se mostrará una silueta.

La eliminación de un usuario, al igual que todos los procesos destructivos e irreversibles en **ATENEA** solicita una confirmación al usuario mediante un mensaje.

### Gestión de medios

La opción de gestión de medios contiene otro submenú para el tratamiento individualizado de cada uno de los tipos de medios: textos, imágenes, vídeos y audios.



Las funcionalidades vuelven de nuevo a ser las propias de un CRUD solo que adaptado a la naturaleza de cada tipo de medio.

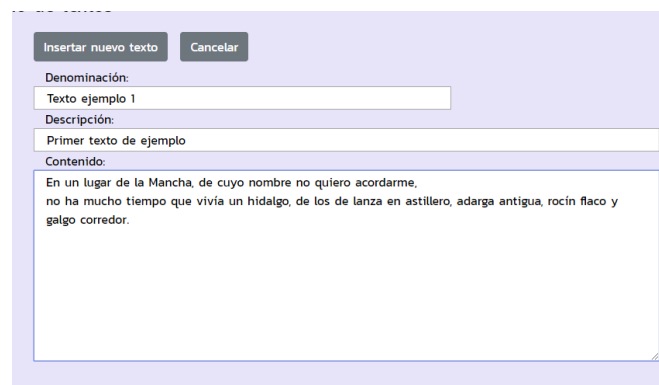
### Textos

Los textos que se tratan en este apartado son párrafos que se desean guardar para su reutilización posterior. Se almacenan directamente en un campo de texto en la tabla de medios a diferencia del resto de medios que se guardan en carpetas del servidor.



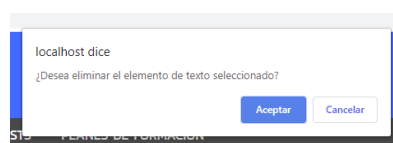
La estructura de la sección consta de una tabla en el lado izquierdo con la relación de todos los textos existentes y un área de detalle en el lado derecho en la que para cada texto seleccionado en la tabla de la izquierda se muestran el nombre, descripción y el propio contenido del texto correspondiente. Estos elementos no son editables en el modo de revisión.

Para insertar un nuevo texto se ha de pulsar el botón dispuesto para ello. Los controles se habilitan para permitir escribir.



La tabla incorpora unos botones para realizar la edición del texto correspondiente. Los controles se vuelven editables.

Asimismo, se puede solicitar la eliminación de un texto. Será posible eliminarlo siempre que no se encuentre en uso en ningún test. En caso de no estarlo se solicita la confirmación para su eliminación.



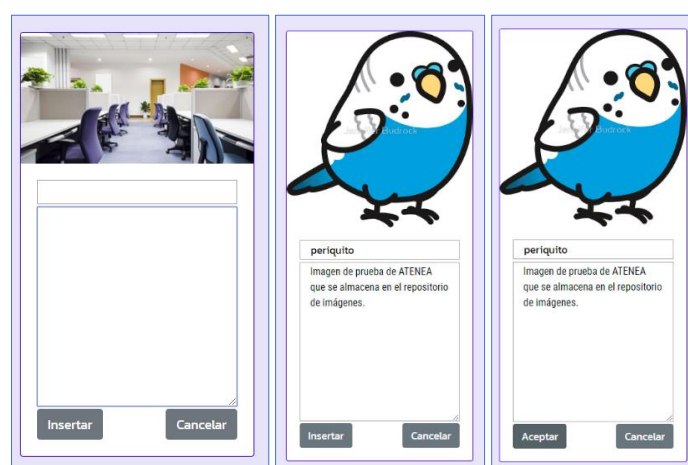
Con carácter general en toda la aplicación se ha preparado la capacidad de abortar los procesos en curso de creación y edición. En estos casos se desestima cualquier cambio o inserción y se retorna al estado de visualización normal.

### Imágenes

En este tipo de medio, la representación de las diferentes imágenes almacenadas se realiza mediante el uso de un control tipo *card* propio de Bootstrap. Estas son pequeñas áreas rectangulares en las que se presentan las imágenes junto con su nombre y descripción. Las imágenes se disponen en una matriz.



Para la inserción de una nueva imagen se pulsa en el botón “*Añadir nueva imagen*” y se crea una nueva carta vacía con los controles abiertos para la edición. Pulsando sobre el área de la foto se activa un browser para seleccionar el archivo de la imagen a insertar. Una vez seleccionada ésta se muestra. Una vez los datos son correctos se pulsa para aceptar la inserción. Se guardan los datos en la base de datos y se sube el archivo de imagen a la correspondiente carpeta del servidor.



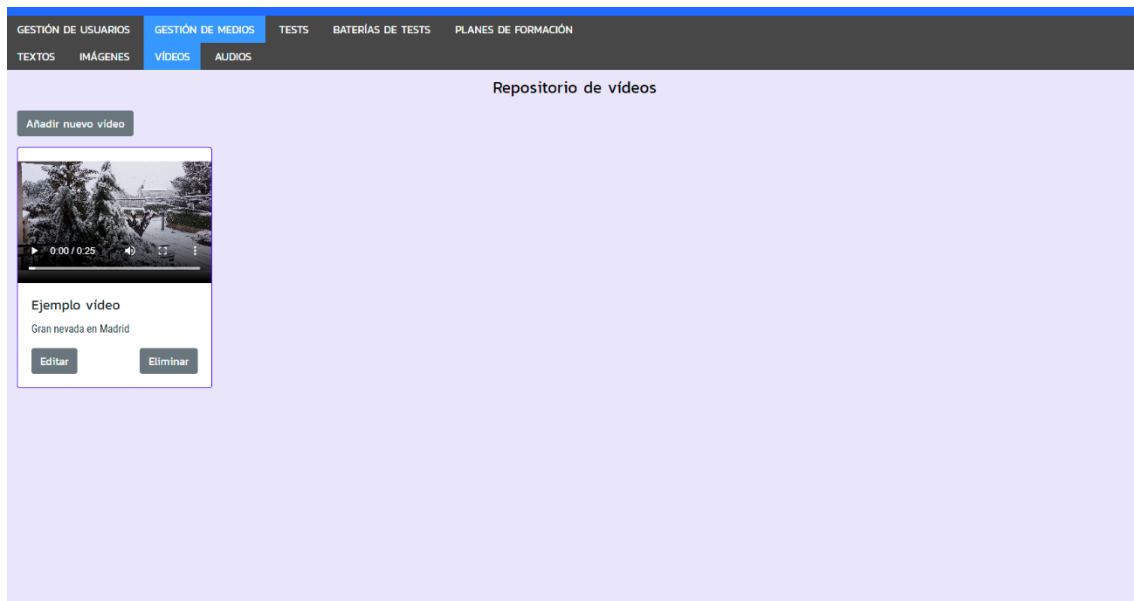
Para la edición de una imagen se pulsa el botón “*Editar*” y se abrirán los controles para su edición. Igualmente, pulsando sobre la imagen se lanzará el browser de selección de archivo.

En cualquier momento se puede abortar la operación volviendo a la situación inicial.

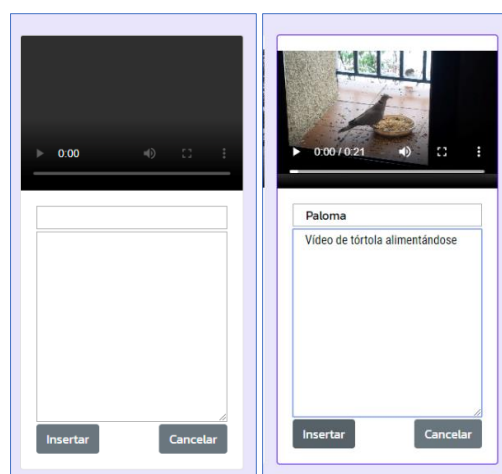
De igual forma que en el caso anterior se puede eliminar una imagen tras confirmación y siempre que no se encuentre en uso.

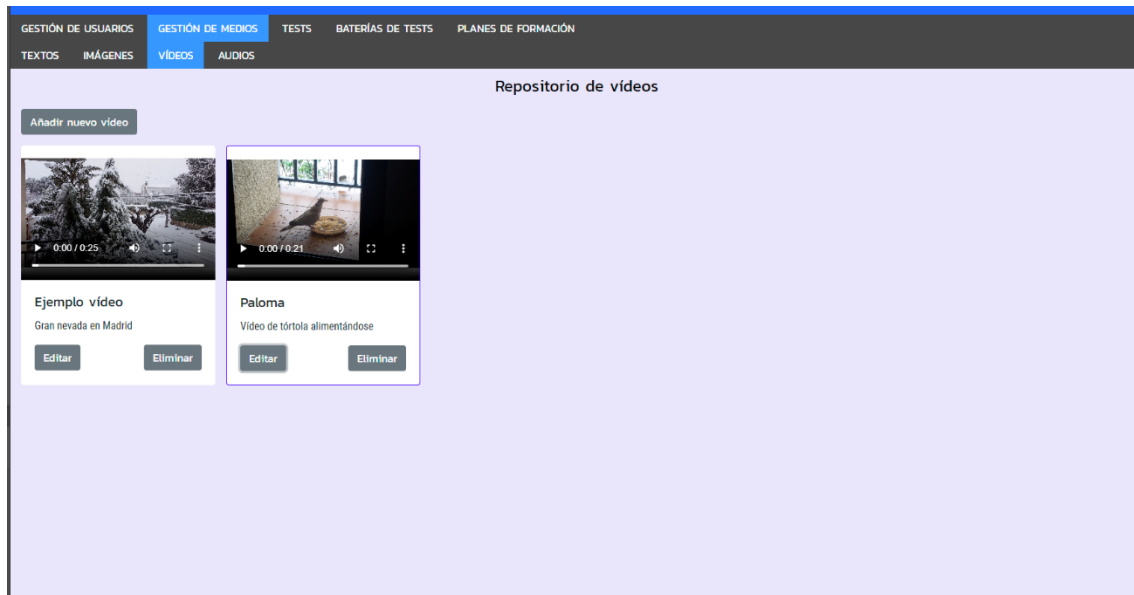
### Vídeos

La gestión de los vídeos mantiene un paralelismo importante con la de las imágenes. Del mismo modo se presentan usando cards de Bootstrap. En este caso, el área de representación mostrará un reproductor de vídeo. Pulsando sobre él se puede controlar la reproducción.



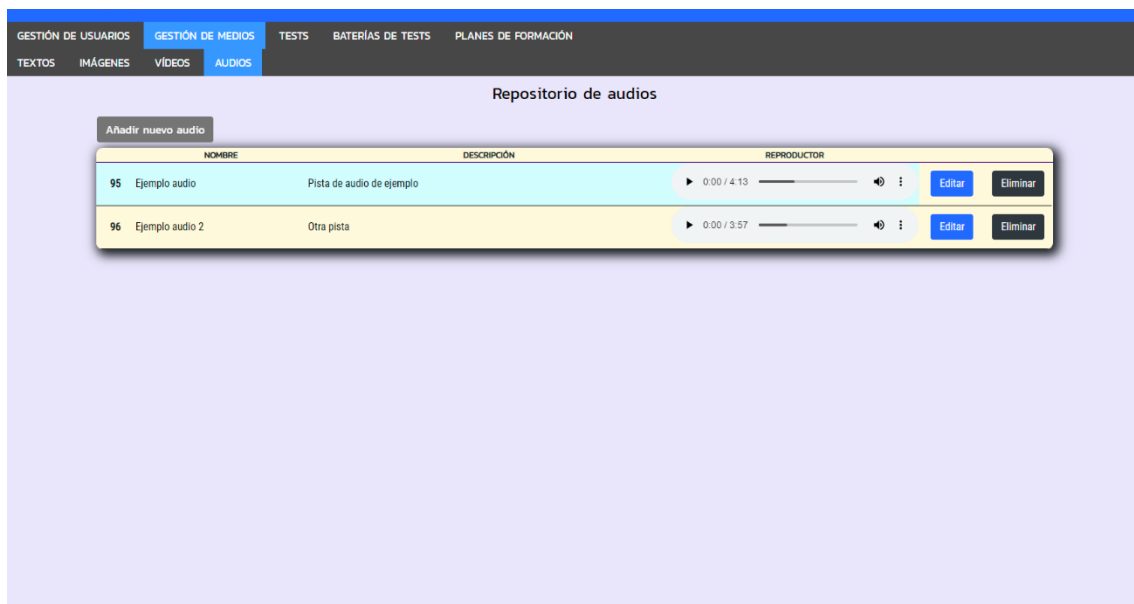
He de resaltar que sólo es posible la reproducción del vídeo correspondiente a la carta en curso, que se encuentra seleccionada tras hacer clic sobre ella. Caso de cambiar la selección, se detiene la reproducción del vídeo anterior. En todo lo demás el comportamiento es como el de la gestión de las imágenes, con lanzamiento de un browser de selección, etc.





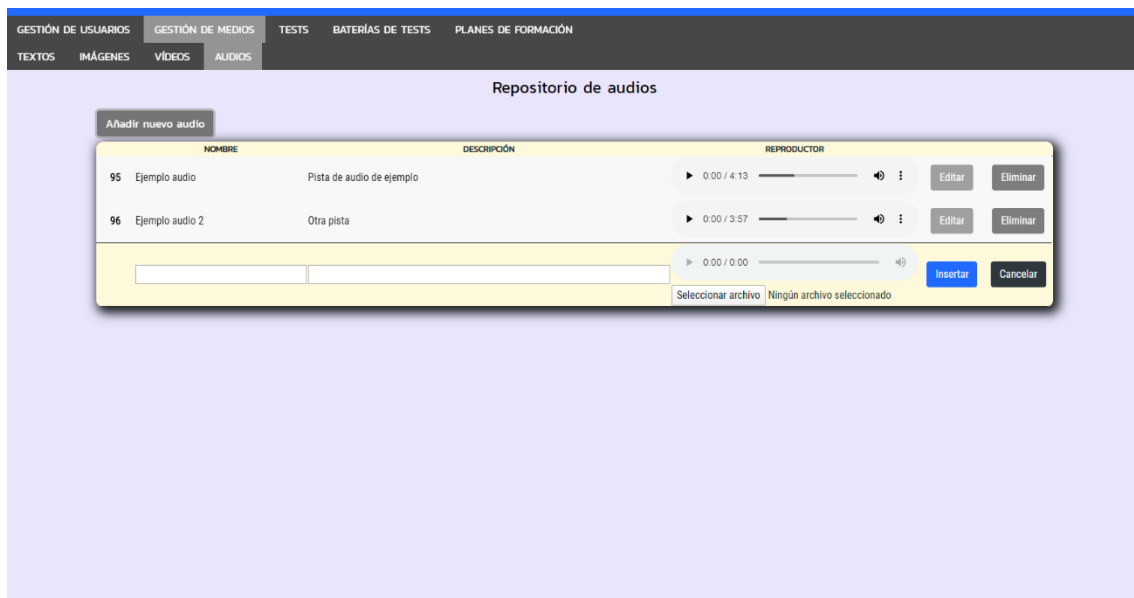
### Audios

Los audios se procesan de forma parecida a los textos, mostrándose apilados en una tabla.



En este caso, el contenido se presenta mediante un reproductor de audio y, de igual forma, sólo se puede reproducir la pista correspondiente a la fila de la tabla en curso. Caso de cambiar la selección se detiene la reproducción. No me ha resultado elemental resolver el problema de la cancelación de las reproducciones al modificar la selección de forma automática pues entre los métodos ofrecidos por el control del reproductor no hay ninguno que distinga entre reproducción basada en pulsación del botón de play o mediante invocación a la función `play()`. Esta distinción era necesaria para controlar el cambio de selección de fila tras pulsación sobre el propio reproductor.

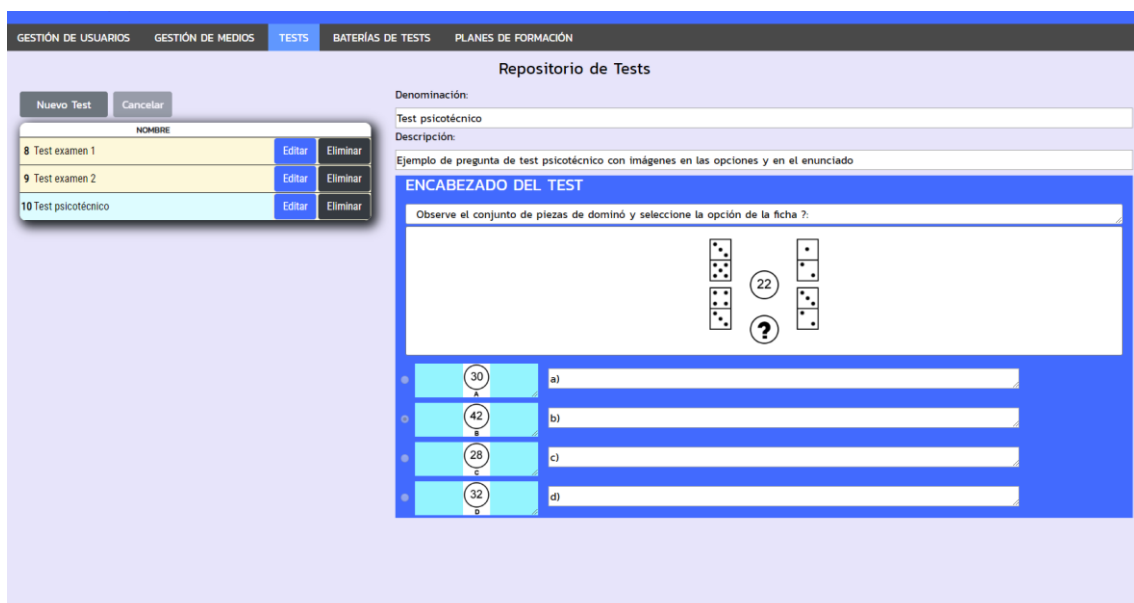
Para añadir una pista de audio al repositorio se pulsa sobre el botón “*Añadir nuevo audio*” y se creará una nueva fila editable en la tabla al final de la misma. Se proporciona un botón para lanzar el browser con el que seleccionar la pista de audio. Una vez todo relleno y correcto, se pulsa el botón “*Insertar*” para subir los datos.



### Gestión de test

La mayoría del trabajo se desarrollará en esta pestaña en la que se crean todos los test que posteriormente se agruparán en baterías que serán asignadas a los alumnos.

La estructura de la página consta de una tabla lateral izquierda en la que se presentan todos los test existentes y un área de datos a la derecha en la que se muestra el contenido del test actualmente seleccionado en la tabla o test en curso.





Esta área de datos consta del nombre y descripción dados al test y un área de encabezado y otra de opciones. El área de encabezado es el enunciado o pregunta propio del test. En él se presentan los elementos que permitan realizar la pregunta y puede ser cualquier combinación de medios (textos, imágenes, vídeos o audios), dispuestos uno debajo de otro.

El área de opciones representa las cuatro respuestas optativas que se presentan al alumno. Pueden estar formadas sólo por textos o acompañados de una pequeña imagen, tal como sucede con los test psicotécnicos. Sólo una de dichas opciones es la respuesta correcta.

El proceso de creación de un nuevo test comienza pulsando el botón “*Nuevo Test*”.

Los controles del área de la derecha se abren y permiten la escritura de un nombre y una descripción para el nuevo test.

The screenshot shows a form titled "ENCABEZADO DEL TEST" with a "+ Elemento" button in the top right. Below the title are two input fields: "Denominación:" and "Descripción:". Below these is a section with a blue header "ENCABEZADO DEL TEST" containing a text input field and four radio button options, each with a small image icon and a text input field.

En el área de encabezado del test existe un editable que permite escribir directamente el texto de la pregunta si así se desea. De esta forma no se recurre a inserción de ningún texto del repositorio si se considera que la pregunta no tiene sentido en su reutilización.

The screenshot shows the "ENCABEZADO DEL TEST" form with the "+ Elemento" button. Below the title is a large text input field with the placeholder text "Texto propio del enunciado del test. Este texto se almacena junto con el propio test." and a small icon in the top right corner.

En la esquina superior derecha del área de encabezado existe un botón etiquetado “+ *Elemento*”. Dicho botón permite añadir un medio al encabezado. Se genera un área rectangular en el encabezado vacía. En la esquina superior derecha de dicha área hay un grupo de controles

que se visualizan atenuados. Al situar el puntero del ratón sobre ellos se muestran y permiten añadir un medio del repositorio o bien eliminarlo del grupo.



Para añadir un medio éste ha de estar previamente seleccionado en la correspondiente pestaña de medios. Una vez pulsado el “+”, se puede uno mover a la pestaña del medio que considere (textos, imágenes, vídeos, audios). En ellas aparece un botón para adjuntar el medio seleccionado al test.

Adjuntar texto seleccionado a test en curso

Pulsando dicho botón y volviendo a la pestaña de test se comprobará que el medio ha sido insertado en el área vacía.

ENCABEZADO DEL TEST

+ Elemento

Texto propio del enunciado del test. Este texto se almacena junto con el propio test.

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo, de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

70-90

Así se puede ir componiendo la combinación que se desee de medios para crear la pregunta buscada.

Para configurar el área de opciones se escribe el texto de las mismas en cada editable.

<input type="radio"/>	<input type="text"/>	OPCIÓN 1
<input type="radio"/>	<input type="text"/>	OPCIÓN 2
<input type="radio"/>	<input type="text"/>	OPCIÓN 3
<input type="radio"/>	<input type="text"/>	OPCIÓN 4

Para añadir una imagen a la opción se debe hacer uso de los botones atenuados que se muestran en la esquina superior derecha de cada área de imagen.



Una vez pulsado el botón del lápiz, se muestran todas las imágenes disponibles en el repositorio de imágenes de opciones. He de recordar que todas estas imágenes se almacenan en un lugar diferente a las de los medios y no tienen nada que ver con ellas.

Se puede seleccionar una imagen y pulsar en el botón “*Aceptar*” para que se adjunte en su posición en la opción. Para salir se pulsa en “*Cancelar*”. Se puede repetir el proceso en el resto de opciones.

Los cinco botones que aparecen tienen las siguientes funciones:

- *Importar imagen*: permite el lanzamiento de un browser para seleccionar una nueva imagen externa que añadir al repositorio y poder ser empleada en las opciones.
- *Aceptar*: añade la imagen del repositorio seleccionada (recuadrada en azul), a la opción actual.
- *Cancelar*: abandona el selector.
- *Editar*: abre los controles de la imagen seleccionada para modificarla tanto en su nombre como la propia imagen.
- *Eliminar*: elimina la imagen seleccionada de la base de datos.

Una vez terminadas las selecciones sólo resta elegir la opción correcta mediante el radio button y el nuevo test queda configurado.

42 B	OPCIÓN 1
30 A	OPCIÓN 2
28 C	OPCIÓN 3
32 D	OPCIÓN 4

Tras pulsar el botón “Insertar Test” el test será subido a la base de datos y estará disponible.

### Gestión de Baterías de test

En esta pestaña se gestionan las colecciones de test que se agrupan bajo un mismo nombre y que se pueden asignar a los alumnos para su ejecución.

Las baterías se presentan mediante tablas. Una tabla a la izquierda con todas las baterías existentes acompañada de una tabla a la derecha en la que se muestran todos los test que componen la batería seleccionada en la tabla anterior o batería en curso. Al seleccionar una batería de la tabla de la izquierda cambia la presentación de la tabla de la derecha consecuentemente.

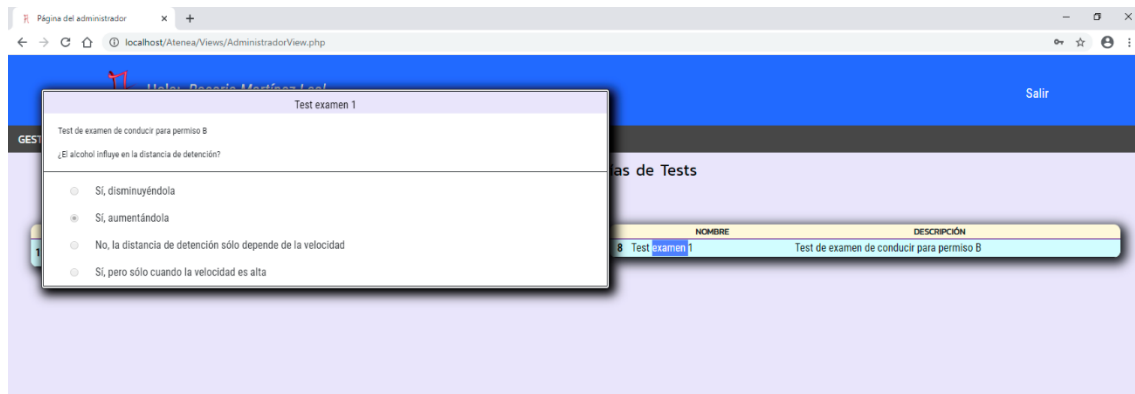
Repositorio de Baterías de Tests

NOMBRE	DESCRIPCIÓN	
12 SGFD	SFGS	<a href="#">Editar</a> <a href="#">Eliminar</a>

NOMBRE	DESCRIPCIÓN
8 Test examen 1	Test de examen de conducir para permiso B

Dado que el contenido de un test puede ser muy extenso, obviamente no se puede mostrar dentro de la tabla de la derecha, por lo que solamente se muestra el nombre y la descripción del mismo. No obstante, esto no es problema para poder visualizar el contenido de cada test. He dispuesto un sistema rápido de visualización de los test que permite rápidamente comprobar su contenido sin tener que memorizar su código y acudir a la pestaña de test. Efectivamente, basta con hacer doble clic sobre el nombre del test y se desplegará una ventana superpuesta en la que se presentará el contenido del test con todos sus elementos. Dicha

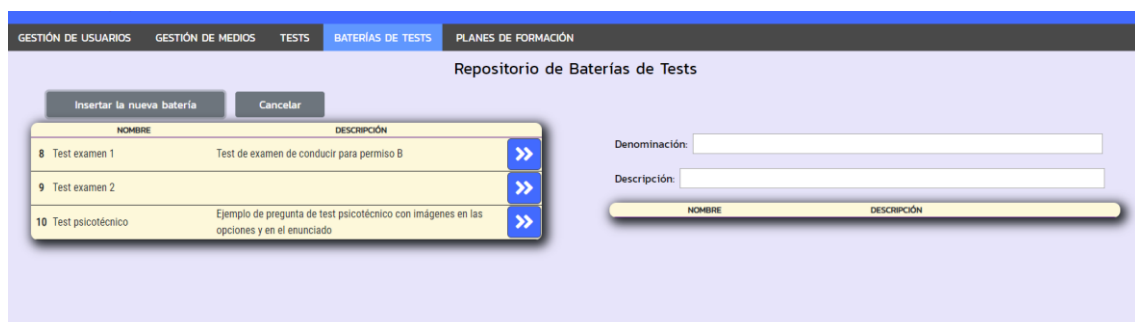
ventana permanecerá desplegada mientras el cursor del ratón se sitúe sobre ella. En caso de salir de la misma, a los 2 segundos, la ventana automáticamente se destruirá.



La interfaz presenta un botón para configurar nuevas baterías a partir de los test almacenados en el sistema y otro habitual para cancelar la operación.

Pulsando sobre el botón “Nueva Batería de Tests”, se sustituye la tabla de baterías por otra que contiene todos los test existentes en el sistema. También la tabla de la derecha es sustituida por otra que representa la nueva batería a crear. Esta última inicialmente se encontrará vacía a la espera de insertarle test. También se incluyen dos editables en los que escribir el nombre y la descripción de la nueva batería.

En el extremo derecho de cada test de la tabla de la izquierda se muestra un botón con un icono de doble flecha a la derecha. Este botón permite añadir a la nueva batería el test correspondiente, que será apilado al final de la tabla de la derecha.



El orden de los test en la batería se puede alterar, adaptándolo a las preferencias del administrador. Para ello se hace uso de los botones del extremo derecho de la tabla de la nueva batería. Con ellos se puede subir o bajar el correspondiente test dentro de la batería para situarlo en la posición deseada. También hay otro botón con una doble flecha hacia la izquierda que permite retirar el correspondiente test de la batería.

Repositorio de Baterías de Tests

Insertar la nueva batería Cancelar

NOMBRE	DESCRIPCIÓN	
8 Test examen 1	Test de examen de conducir para permiso B	>>
9 Test examen 2		>>
10 Test psicotécnico	Ejemplo de pregunta de test psicotécnico con imágenes en las opciones y en el enunciado	>>

Denominación: Ejemplo de nueva batería

Descripción: Descripción optativa larga de la batería

NOMBRE	DESCRIPCIÓN	
8 Test examen 1	Test de examen de conducir para permiso B	<< < > >>
9 Test examen 2		<< < > >>

Estos mismos controles aparecen en caso de pulsar el botón “*Editar*” de la tabla de baterías existentes. De esta forma se puede alterar la estructura de la misma.


Al igual que en los casos anteriores, se dispone, para cada batería, de un botón “*Eliminar*” que provoca el borrado de la batería correspondiente de la base de datos.

### Gestión de Planes de Formación

Finalmente, una vez que todos los elementos formativos han sido creados, se puede asignar las baterías de test a los alumnos para diseñar sus planes de formación.

En esta última pestaña se presentan todos los alumnos en una tabla. A su derecha se muestran todas las baterías de test asignadas al alumno. En dicha tabla se presentan las fechas de asignación, de inicio de ejecución, de finalización y la correspondiente calificación.

Repositorio de asignaciones

NOMBRE	APELLIDOS	
17 Gregorio Esteban	Sánchez Fernández	 Editar

NOMBRE	DESCRIPCIÓN	ASIGNACIÓN	INICIO	FIN	NOTA
12 SGFD	SFGS	2020-03-24 11:03:23	2020-04-07 09:41:25	2020-04-07 09:41:31	1

Para alterar las asignaciones de los alumnos se utiliza el botón “*Editar*” que activa el modo de edición y permite asignar y retirar baterías al alumno. Para ello, la interfaz cambia de manera que la tabla de los alumnos es sustituida por otra con la colección completa de baterías de test existentes.

Repositorio de asignaciones

Actualizar asignaciones Cancelar

NOMBRE	DESCRIPCIÓN	
12 SGFD	SFGS	>>

NOMBRE	DESCRIPCIÓN	ASIGNACIÓN	INICIO	FIN	NOTA
12 SGFD	SFGS	2020-03-24 11:03:23	2020-04-07 09:41:25	2020-04-07 09:41:31	1

En dicha tabla se presenta un botón en el extremo derecho de cada fila con una doble flecha para asignar la correspondiente batería al alumno. Una batería podrá ser asignada siempre que no se encuentre ya formando parte del plan de formación del alumno. En ese caso el sistema avisará y no se podrá duplicar. Caso de que no haya inconveniente, la batería será asignada y aparecerá apilada al final de todas las asignaciones existentes para el alumno.

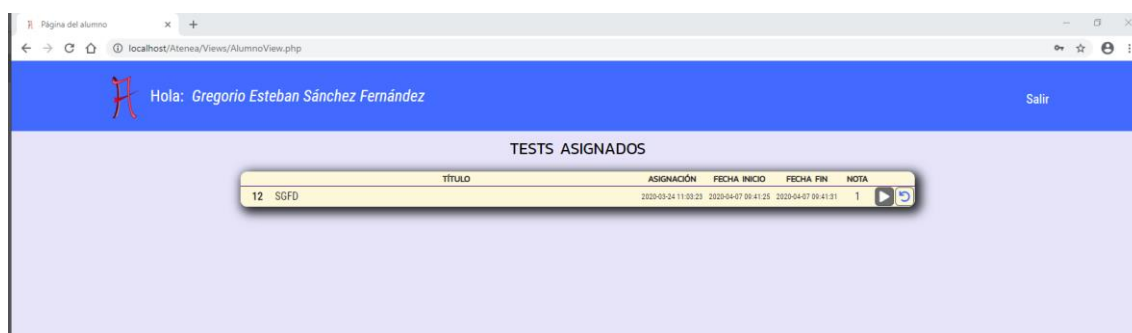
Un botón de doble flecha hacia la izquierda se muestra en el extremo derecho de las asignaciones y permite eliminarla siempre que no existan restricciones. En este caso no tienen sentido las flechas de subida y bajada pues en este caso la posición de una batería en la tabla no indica orden alguno. El orden viene determinado por la fecha en la que se realiza la asignación.

Para finalizar los cambios se ha de pulsar sobre el botón “*Actualizar asignaciones*”, o bien sobre el botón “*Cancelar*” para abortar la operativa.

## Página del alumno

Una vez accedido al sistema con perfil de alumno aparece la página de su plan de formación.

De igual modo que en el caso del administrador, se muestra el nombre del alumno en el encabezado junto al link para cerrar la sesión.

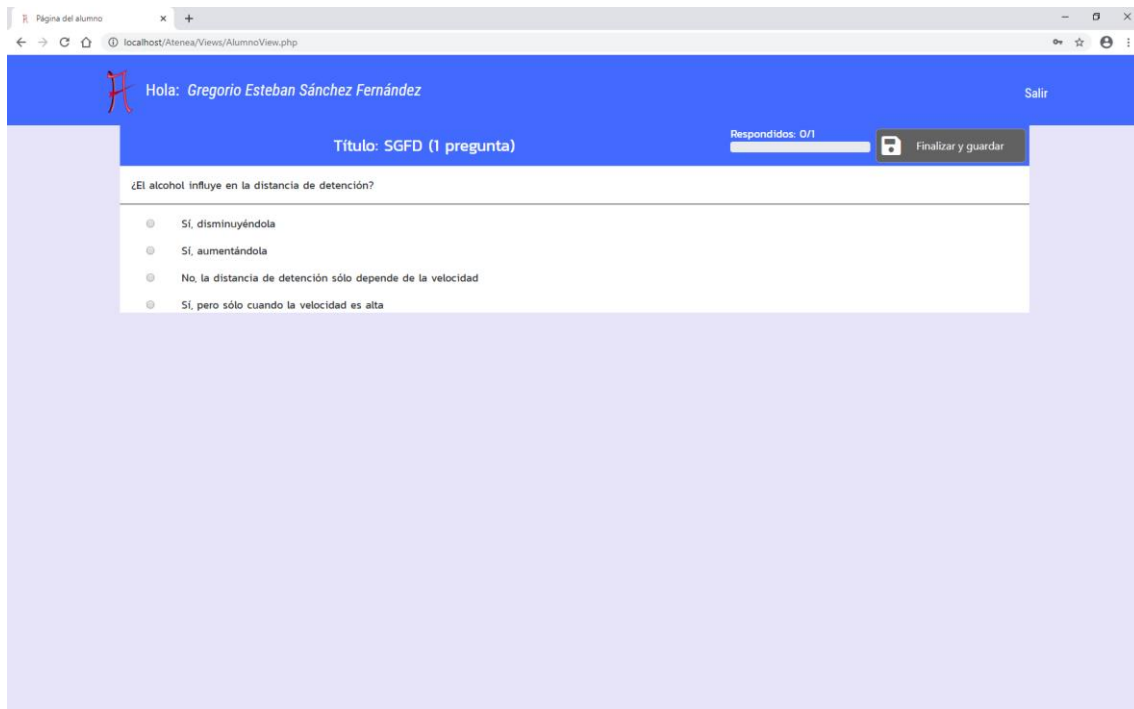


Se muestran todas las baterías de test asignadas al alumno junto con sus calificaciones y unas fechas:

- *Fecha de asignación*: fecha y hora en la que dicha batería fue asignada por un administrador al alumno.
- *Fecha de inicio*: fecha y hora en la que comenzó la primera ejecución de dicha batería. Esta fecha puede ser reseteada por el alumno si pulsa el botón reset.
- *Fecha de finalización*: fecha y hora de finalización de la batería. Igualmente, esta fecha se borrará si el alumno resetea la ejecución.

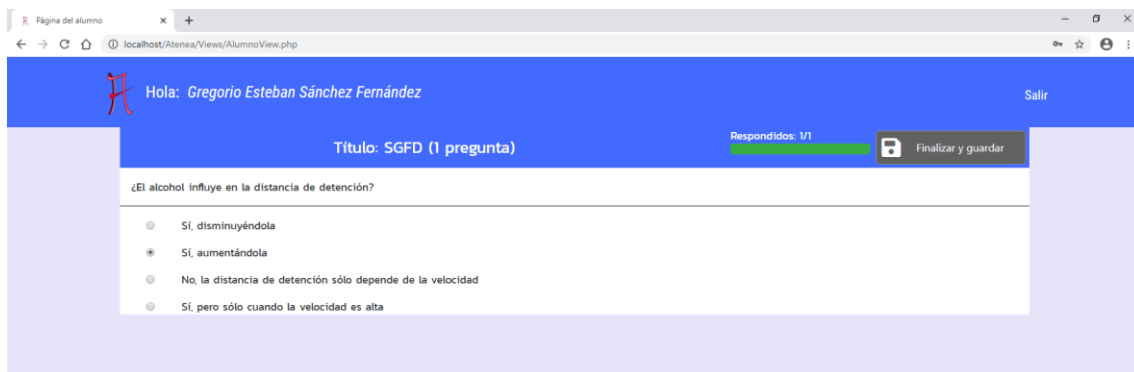
Cada batería tiene dos botones situados a la derecha que permiten entrar en la ejecución o bien resetearla. El reset implica restaurar la batería a su estado inicial borrando todas las respuestas y la calificación.

En la ejecución de la batería se presentan los test al alumno y éste los puede ir respondiendo. El sistema le va indicando el grado de avance en la ejecución.



The screenshot shows a web browser window with the URL `localhost/Atenea/Views/AlumnoView.php`. The page has a blue header with a logo on the left, the text "Hola: Gregorio Esteban Sánchez Fernández" in the center, and a "Salir" button on the right. Below the header, there is a section titled "Título: SGFD (1 pregunta)" with a progress indicator "Respondidos: 0/1" and a "Finalizar y guardar" button. The main content area contains a question: "¿El alcohol influye en la distancia de detención?". Below the question are four radio button options: "Sí, disminuyéndola", "Sí, aumentándola", "No, la distancia de detención sólo depende de la velocidad", and "Sí, pero sólo cuando la velocidad es alta".

Una vez que desee terminar y guardar el estado de ejecución, puede pulsar el botón “”.



This screenshot is similar to the previous one, but the progress indicator now shows "Respondidos: 1/1" with a green bar. The second radio button option, "Sí, aumentándola", is now selected, indicated by a filled circle.

El conjunto de sus respuestas se guardará y la nota se actualizará igualmente.



## Mejoras y desarrollos futuros.

Obviamente, tras inspeccionar **ATENEA** y su implementación se pueden descubrir muchos puntos de mejora y rediseño. Yo mismo cada vez que lo ejecuto encuentro elementos cuya disposición es cuando menos bastante mejorable, pero hay que mantener una perspectiva del tipo de trabajo que se ha pretendido desarrollar y la complejidad potencial del mismo.

A continuación, propongo una serie de mejoras y cambios.

- Respecto a la configuración de los test, sería muy conveniente sustituir el actual procedimiento de adición de medios por otro más intuitivo y que no necesitara el movimiento del usuario de pestaña en pestaña.
- Sería bueno que los tamaños de imágenes y vídeos pudieran ser configurables y definidos por el administrador.
- También resultan útiles tooltips sobre los controles de la interfaz de usuario como ayuda.
- La filosofía de utilizar medios en un repositorio se adapta mejor al desarrollo de lecciones teóricas en las que se presenten al alumno conceptos. Desde ese punto de vista **ATENEA** puede evolucionar hacia una plataforma de formación más completa en la que aunar clases teóricas y ejercicios de evaluación tales como los test que se han presentado. Los medios almacenados en un repositorio reutilizable tienen más sentido a la hora de configurar lecciones teóricas que pueden ir evolucionando y actualizándose. Además, en las circunstancias actuales de la crisis de la **COVID-19**, los sistemas de formación online pueden tener un notable incremento de demanda a todos los niveles educativos.
- Las páginas de los usuarios, tanto alumnos como administradores, se pueden enriquecer con acceso a modificación de datos personales, recomendaciones de cursos, planning, foro de usuarios, área de comunicación con el profesorado, estadísticas de rendimiento y calidad, etc., mediante el uso de módulos adicionales escalables.
- El proceso de ejecución de los test se ha desarrollado aquí en su forma más simple. El alumno simplemente responde y puede resetear la ejecución y repetirla. Una situación más realista y completa implica la definición de test evaluativos y no evaluativos. Los evaluativos sólo se pueden ejecutar en una misma sesión y sólo una vez. Adicionalmente pueden tener el tiempo de ejecución restringido según definición del administrador. Los no evaluativos pueden considerarse de entrenamiento y fijación de conceptos y pueden seguir el procedimiento de ejecución aquí implementado.
- Considero conveniente el estudio de una forma alternativa para gestionar la creación de las sentencias preparadas en los accesos a la base de datos que permita compactar el código y evitar la gran variedad de funciones similares.

## Conclusiones.

Finalmente, deseo poner en valor el importante trabajo desarrollado para elaborar esta aplicación. Pese a que sin duda es fácil encontrar errores o déficits en su desarrollo comparado con herramientas comerciales existentes, no debemos olvidar que dichas herramientas son desarrolladas no por una sola persona, sino por un equipo de programadores y diseñadores con amplia experiencia y con el uso de frameworks de desarrollo testados y desarrollados por empresas tales como Google (Angular).

He disfrutado durante el desarrollo de esta aplicación, en tanto en cuanto que me ha permitido estructurar los conocimientos adquiridos durante el Máster e incluso ampliarlos en muchos aspectos que han exigido la búsqueda de soluciones más específicas.

Considero que, aún con limitaciones, este curso me ha capacitado para desarrollar páginas web dinámicas de cierta complejidad y, lo que es muy importante, me ha ayudado a poner algo de orden y desarrollar una perspectiva completa del mundo del desarrollo web y sus tecnologías. Ese conocimiento te permite poder elegir el camino próximo por el que discurrir y que creo que debe ser el aprendizaje de algún framework de desarrollo profesional (Angular o nodeJs).

Finalmente quiero agradecer a Rosario Martínez Leal, mi profesora del Máster su ayuda y predisposición durante todo el desarrollo del curso y de este trabajo.

Agradezco al lector su atención al haber completado la lectura de esta memoria.

Gracias.

Madrid, 12 de Mayo de 2020



Vicente Alejandro Garcerán Blanco.