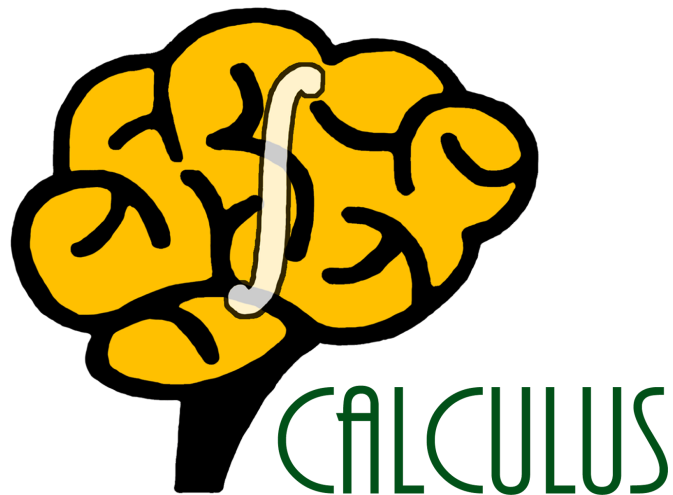


Proyecto Calculus
2016 - Grupo 06
Descripción de la arquitectura
Versión 2.1



Historia de revisiones

Fecha	Versión	Descripción	Autor
25/08/2016	1.0	Comienzo del informe	Héctor Almeida
26/08/2016	1.1	Verificación	Martín Calcagno
26/08/2016	1.2	Revisión SQA	Manuel Alzugaray
20/11/2016	2.0	Actualización información sobre backend	Héctor Almeida
20/11/2016	2.1	Revisión SQA	Manuel Alzugaray

Contenido

1. INTRODUCCIÓN	3
1.1. PROPÓSITO	3
1.2. ALCANCE	3
1.3. SIGLAS, DEFINICIONES Y ABREVIACIONES	
2. DISEÑO	3
2.1. DIAGRAMA DE COMPONENTES	4
2.1.1. <i>Servidor Web</i>	4
2.1.2. <i>WebApp Alumno</i>	5
2.1.3. <i>WebApp Profesor</i>	5
2.1.4. <i>Base de Datos</i>	5
2.2. MODELO DE DOMINIO	6
2.2.1. <i>Descripción</i>	6

1.Introducción

1.1 Propósito

En este documento se describirá el diseño de la arquitectura para el proyecto Calculus, el cual servirá como guía para los desarrolladores de la aplicación.

1.2 Alcance

Se utilizarán dos diagramas distintos para esta descripción, un Modelo de Dominio el cual es fundamental para obtener una visión general de cómo se modelarán las distintas entidades correspondientes a la lógica de la aplicación, y por otro lado un Diagrama de Componentes, el cual muestra cómo será dividido el sistema en componentes, y la interacción entre estos.

1.3 Siglas, definiciones y abreviaciones

MVC: Model View Controller, patrón de diseño que divide la arquitectura en 3 componentes: Modelo de datos, Vistas y Controladores.

ORM: Object/Relational Mapping, capa intermedia que abstrae al programador de la base de datos.

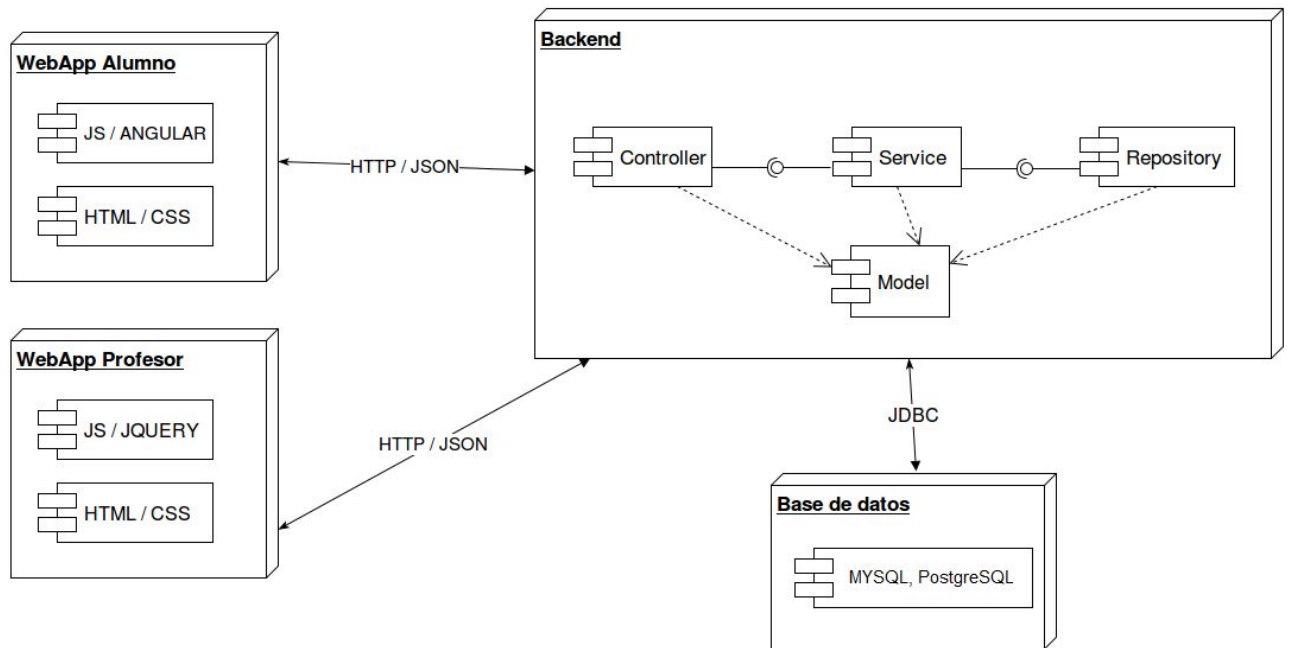
JSON: JavaScript Object Notation, formato de texto ligero para intercambio de información.

AJAX: Asynchronous JavaScript and XML, intercambiar información con el servidor de manera asíncrona.

2. Diseño

A continuación se presentará la arquitectura general del proyecto de manera abstracta, por medio del Diagrama de Componentes y Modelo de Dominio. También se hará referencia a la tecnología elegida para el desarrollo.

2.1 Diagrama de componentes



2.1.1 Servidor Web

El diseño del Servidor Web está basado en el patrón MVC y Service/Repository. Se utilizará el framework Hibernate, el cual es un ORM, para que la implementación del Modelo y su persistencia no dependa de ninguna base de datos en particular. El framework Spring MVC servirá para mapear los objetos a estructuras de datos que serán utilizadas en las vistas.

Los Controladores se encargarán de obtener información de Model así como de su actualización. Proveerán las interfaces con los métodos necesarios para los componentes que implementan las interfaces gráficas. Se encargan del Request mapping.

Los Servicios son los que realmente implementan la lógica, interactuando con las clases de Model. Obtienen y persisten los objetos utilizando las clases de Repository, las cuales son las únicas que trabajan directamente con la base de datos.

2.1.2 WebApp Alumno

Será una aplicación Web responsiva, lo cual tiene la ventaja de que será multiplataforma y podrá adaptarse a distintos tamaños de pantalla. Estará orientada a pantallas para dispositivos móviles.

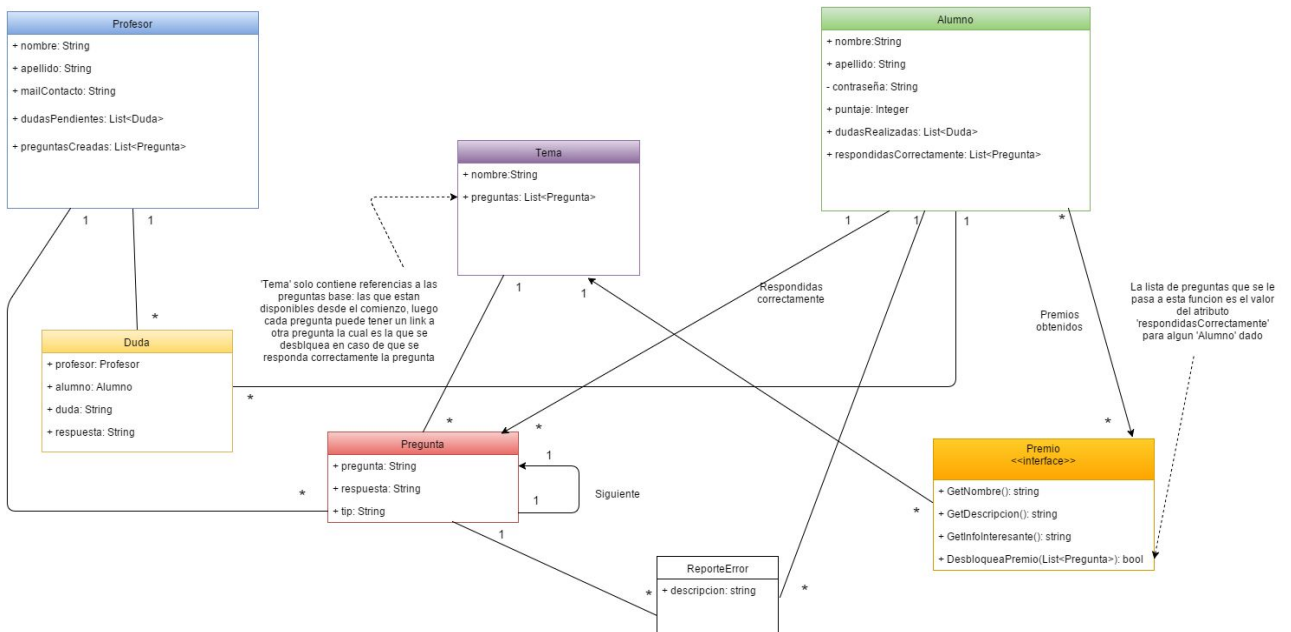
2.1.3 WebApp Profesor

También será una aplicación web pero en este caso estará orientada a pantallas de computadoras, puesto que es más útil y cómodo para los profesores editar y administrar las preguntas, así como acceder a los datos sobre el avance de los estudiantes.

2.1.4 Base de datos

La base de datos estará alojada en un servidor de Azure. Como ya se ha mencionado, se usará el framework Hibernate el cual nos permite trabajar con los objetos sin preocuparnos de las particularidades de una base de datos en específica.

2.2 Modelo de Dominio



2.2.1 Descripción

Profesor

Contiene información básica del profesor. Contiene una lista de referencias a las preguntas que ha creado así como a las dudas pendientes que debe responder.

Alumno

Posee una lista con referencias a todas las preguntas respondidas correctamente, lo cual es necesario para evaluar el avance que ha logrado.

También contiene la lista de Premios que ha recibido.

Premio

Los premios indican que el Alumno ha logrado superar un determinado desafío, como por ejemplo: "Responder correctamente 10 preguntas de la categoría Series". Se espera que los Premios puedan ser totalmente arbitrarios, por eso se ha modelado a Premio como una interfaz. Cada vez que surja un nuevo tipo de premio, basta con agregar una nueva implementación de esta interfaz, la cual debe proveer un método que recibirá una lista de preguntas (la cual es la lista de preguntas respondidas correctamente por parte de algún alumno) y retornará verdadero/falso, indicando si la lista de preguntas cumple los requisitos para desbloquear dicho premio.

Tema

Un tema agrupa un conjunto de preguntas relacionadas. Contiene referencias solamente a las preguntas "base", las cuales son a las que el jugador accede directamente. A medida que vaya contestando preguntas correctamente irá desbloqueando nuevas preguntas.

Pregunta

Esta entidad contiene la pregunta en sí, la respuesta correcta y un tip, el cual se le mostrará al alumno en caso de que responda incorrectamente. Referencia a la siguiente pregunta que será desbloqueada en caso de que sea respondido correctamente la actual pregunta.

Duda

En caso de que el tip proporcionado por la pregunta no sea suficiente, el alumno podrá también generar una consulta al profesor, el cual será notificado.

ReporteError

En caso de que el alumno reciba un resultado incorrecto pero esté seguro de haber respondido correctamente, podrá generar un reporte de error. El profesor que ha creado la pregunta será notificado.