El sistema LSAT será un SaaS (Software as a Service) lo cual nos limita a que la única interacción del usuario va a ser a través del navegador web de su elección.

Reconocemos que las preferencias de los usuarios finales de este sistema en cuanto navegadores web son diversos y es deseable que en un futuro sean soportados todos los navegadores web pero para el alcance de este proyecto solo es necesario que el sistema trabaje en Firefox 36

LSAT deberá ser programado con php para el backend y con html5 y javascript para el frontend.

Para facilitar el entendimiento y la organización del código el sistema está distribuido en 3 capas.

La de vistas – front end, páginas web que ve el usuario

Los controladores – Mantienen la lógica del sistema, se comunican entre sí a través de os modelos y clases auxiliares. No son accesibles o visibles por el usuario. La única forma en que se comunican con el front end es a través de httprequests preferentemente de tipo POST por parte de las vistas.

La información recibida en estas peticiones deberá ser validada para comprobar su auteticidad y prevenir ataques de seguridad.

Las respuestas de los controladores a estas requests deberá ser siempre una string en formato de JSON, manteniendo siempre una propiedad obligada llamada “message” la cual contendrá un string de “success” o “error” según sea el caso y debe ir acompañada de la información correspondiente.

Los modelos – Clases en php que representan entidades de la base de datos, trabajan con los controladores y con las operaciones CRUD correspondientes a su entidad.

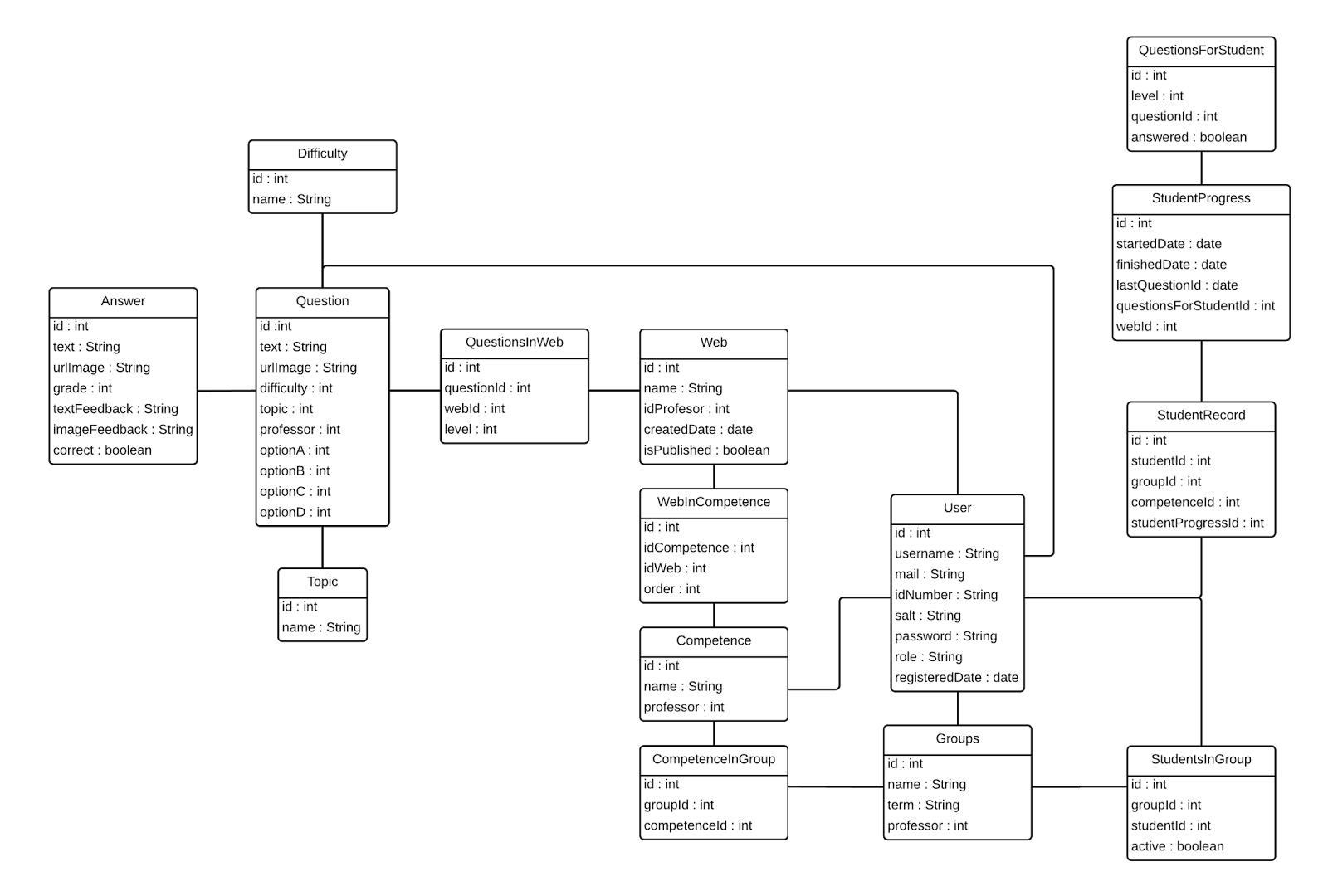
No todas las entidades de la base de datos tienen un modelo y algunos modelos pueden realizar operaciones para varias tablas. Es decir, algunas tablas pequeñas en la base de datos que son usadas para un mismo propósito general pueden ser agrupadas y trabajadas desde el mismo modelo. Tal es el caso de las tablas “competence” “websincompetence” y “competenceingroup” las cuales son trabajadas por el mismo modelo dentro del archivo “Competence.php”

Dentro de la tabla “Questions for students” hay muchos registros (MUCHOS) cada uno contiene información sobre el nivel, la pregunta y si fue contestada o no.

Va a haber un registro en student record por cada red que tenga la competencia. Despues se puede hacer un query y agrupar por id de competencia y tener una lista separada por comas con los ids de todas las webs diferentes.

Con esas webs ids podemos ir a studentprogress y saber cual es el avance por red. Podemos ver cual es la primer pregunta de la red, la ultima y como están guardadas por orden una simple resta nos dice cuanto lleva el alumno.

lastQuestionId nos dice en donde se quedo el alumno. Osea cuál es la última pregunta que contesto. Si la última pregunta es igual que la “lastQuestionId” quiere decir que termino esa web y lo podemos pasar a la siguiente red de esa competencia.



**Comenzar a contestar**

El alumno le da click a una competencia

Lo mandamos a answer.php

Si apenas va a comenzar a contestar - > llenamos las tres tablas

Despues:

//Un estudiante no puede dejar empezada una red dentro de una competencia, tiene que contestarla toda de un jalon.

//Probablememte tenga que cotestar la competencia completa tmabien.

Para saber cual es la siguiente pregunta a contestar

Question.php

getNextQuestion($studentId, $groupId, $competenceId){

//Traer los regidtros de studentrecord que cumplan con los tres ids

//Despues traer de studentProgress todos los que cumplan con studentProgressId

// Si todos los de student progress tienen seteado un finished date quiere decir que la competencia fue terminada

//Si la competencia no esta termianda tenemos que saber el orden de las redes, para saber cual sigue

//Obtenemos “websincompetence” con el competenceId

Select \* from websincompetence where competenceId = $competenceId ORDER BY order

//Cada element del arreglo resultante es una red, ordenadas de menor a mayor según el prden

//iteramos el arreglo vamos viendo los ids de las webs uno por uno

//si ese webid ya fue terminado, nos pasamos a la siguiente

// a la primera ocurrencia de web no terminada(sin finished date) sabemos que esa es la que tiene que seguir contestando

Vamos a last answeredQuestionId dentro de studentprogress

//Vamos a questionsforstudent y buscamos el nivel en el que se encuentra esa pregunta.

//Buscamos ahí también la primerpregunta con “answered”=false del siguiente nivel

//Si ya no hay otro nivel ¿

//Si ya no hay preguntas libres del nivel que sigue?