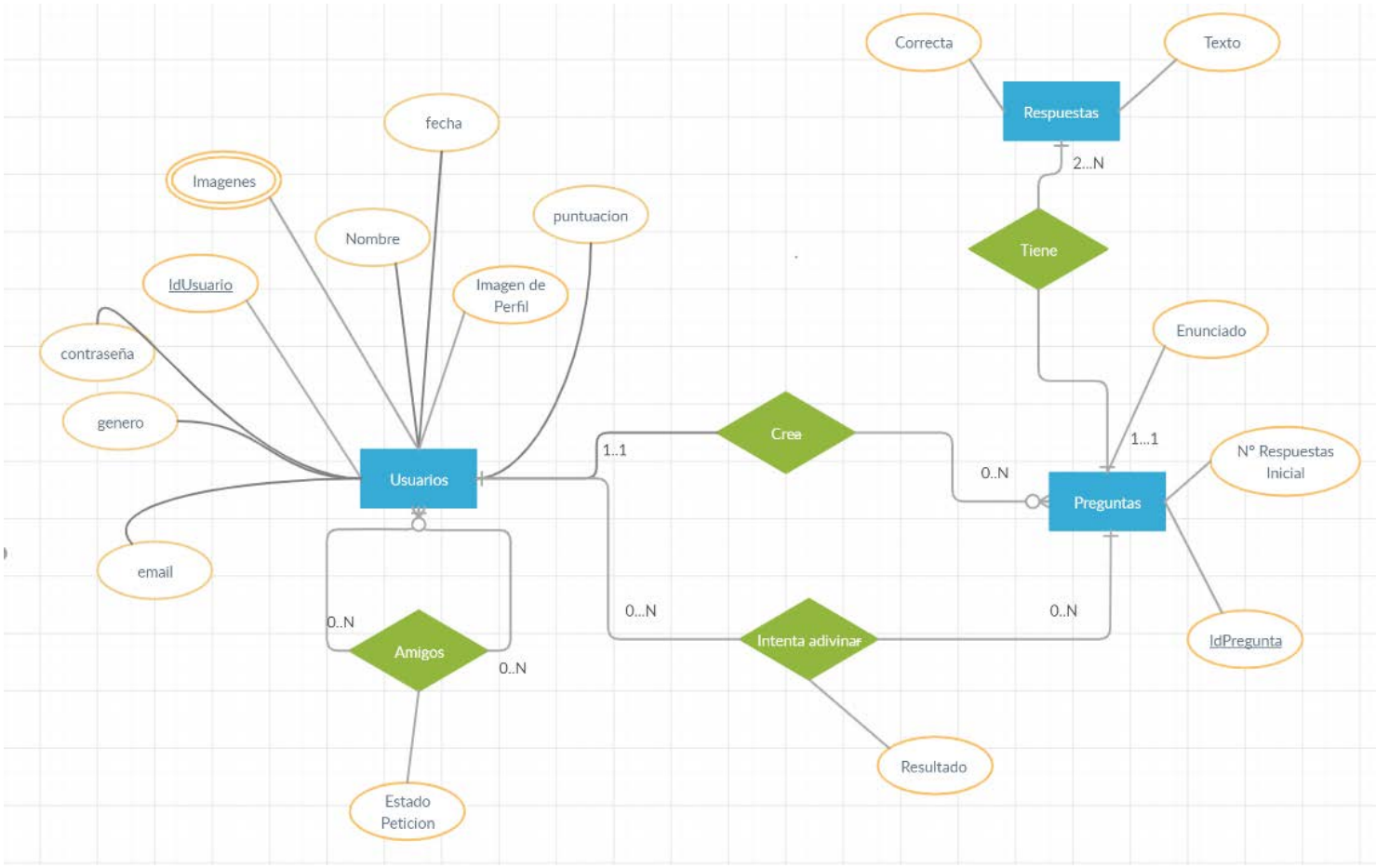
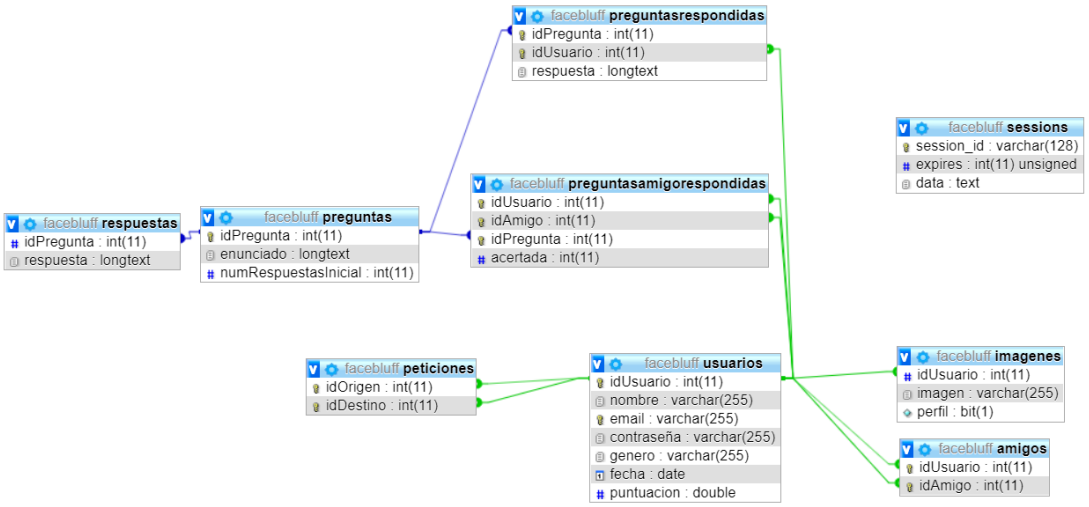
Presentación FaceBluff Grupo9

Diseño de la base de datos

Modelo entidad-relación



Modelo relacional



Estructura de la aplicación

Controlador

El controlador de la aplicación se encuentra en el archivo app.js

Routers

La aplicación contiene dos routers para gestionar las peticiones relativas a usuarios y preguntas. Estos routers se encuentran en los archivos routerUsers.js y routerQuestions.js respectivamente

Modelos

Para gestionar el acceso a la BBDD hemos utilizado 4 DAOs diferentes. Estos DAOs se encuentran en los ficheros DAOAmigo, DAOImagenes, DAOPreguntas y DAOUsuario.

- DAOAmigo: se encarga de gestionar las peticiones y los amigos de la base de datos.
- DAOImagenes: se encarga de insertar y leer tanto la imagen de perfil como las que el usuario añade posteriormente.
- DAOPreguntas: se encarga de crear preguntas, leer preguntas, leer las respuestas de una pregunta con un límite en función del numero de respuestas iniciales asignadas en la creación de la pregunta. Actualiza la puntuación de un usuario si acierta una pregunta de un amigo. Inserta y lee todo lo relacionado con adivinar una pregunta de un amigo.
- DAOUsuario: se encarga de insertar, actualizar y leer usuarios.

Otros módulos

Para la configuración del pool de conexiones a la base de datos utilizamos el módulo config (archivo config.js). También empleamos un módulo utils para realizar algunas operaciones de parseo de datos.

Rutas gestionadas por el servidor

Usuarios: (comienzan por /users)

- /login
- /my_profile
- /my_profile/:idUsuario
- /update_user
- /upload_img
- /new_user
- /imagenNormal/:ruta
- /imagen/:idUsuario
- /search
- /friends
- /friends/request_friend/:idUsuario
- /friends/add_friend/:idUsuario

Preguntas: (comienzan por /question)

- /show
- /selected/:idPregunta
- /answer/:idPregunta
- /answerToOther/:idPregunta/:idAmigo
- /create

Implementación de la sesión para un usuario logueado

Al realizar una petición de tipo POST en la ruta /users/login, se comprueba que el email y la contraseña coinciden con los de algún usuario de la BBDD. En caso afirmativo, se utiliza el atributo request.session.currentUser para guardar el usuario logueado. Si no coincide, se redirige a la ruta /users/login con un mensaje de error.

Restricción de acceso a rutas para usuarios no logueados

En todas las rutas excepto /users/login y users/new_user se introduce un middleware de control de acceso que consiste en una función llamada accesControl. En caso de que el usuario de la sesión actual (request.session.currentUser) sea null o la contraseña introducida no coincida con la de la base de datos, se redirige a la ruta /users/login mostrando un mensaje informativo.

Gestión de errores 404 y 500

Para gestionar el error 404 y el error 500 hemos utilizado dos archivos ejs (error404.ejs y error500.ejs respectivamente). El error 404 se lanza cuando se recibe una petición que no es repsondida por ninguna función y el error 500 al tener algún fallo en la BBDD o al introducir campos erróneos en los formularios.