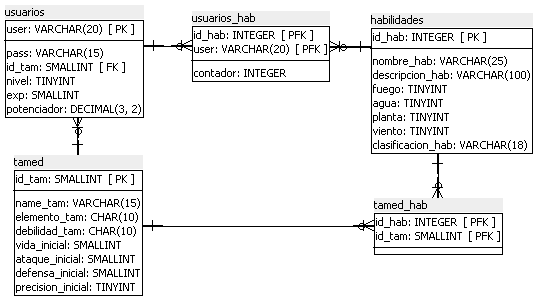
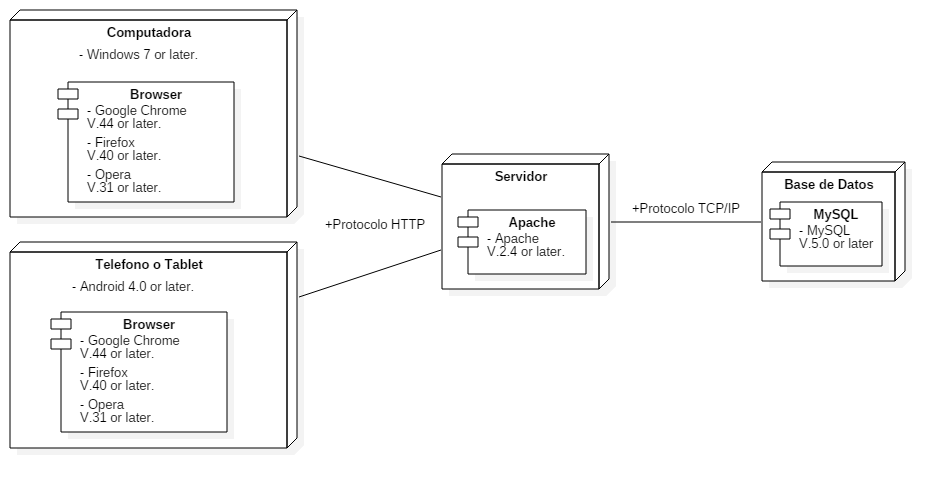
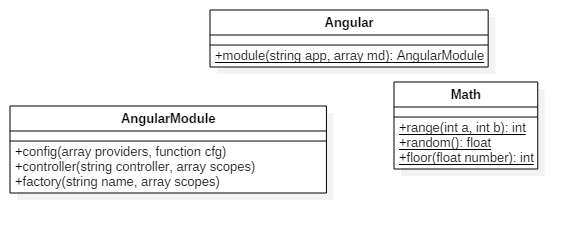
**Diagrama entidad relación:**



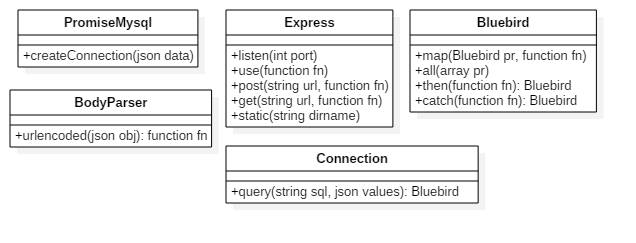
**Diagrama de despliegue:**

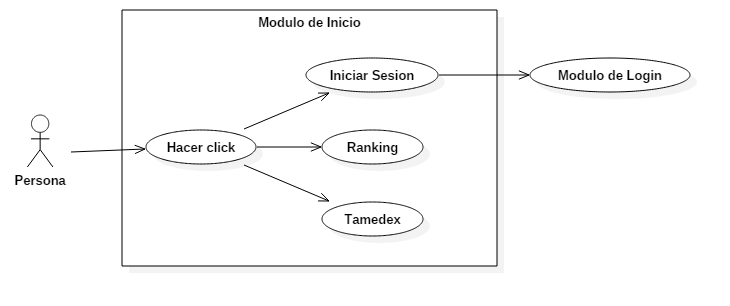
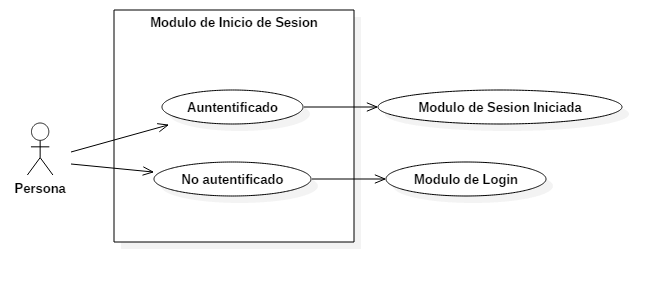
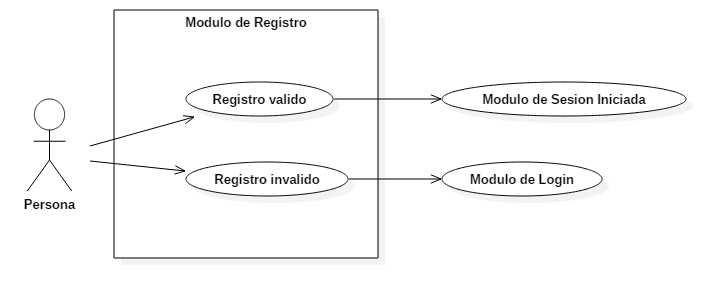
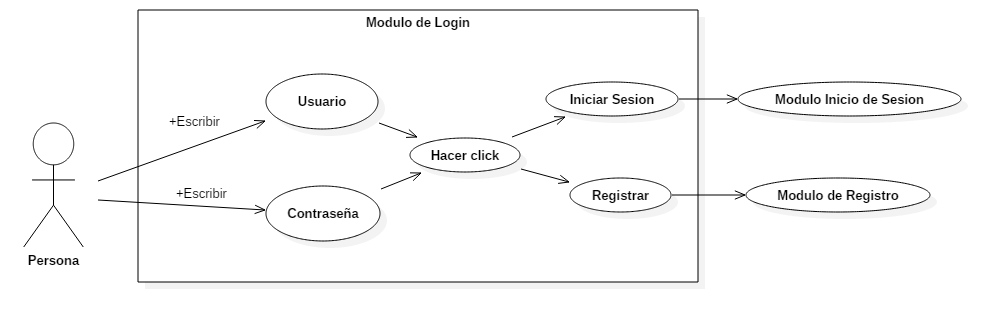


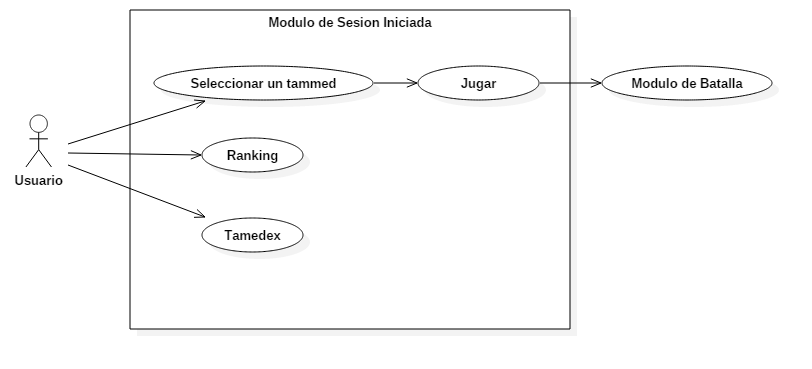
**Diagramas de clases:**

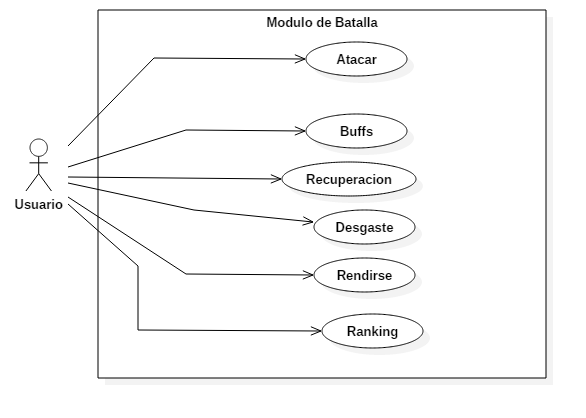
**Cliente: **

**Servidor:**



**Diagramas de casos de uso:** ****

****



**Listado de métodos del lado del cliente:**

**Archivo: main.js**

**1. var** app = angular.module('elementif', [...]);

El método module() busca y obtiene el módulo elementif indicado en el código HTML, además de inyectarle los módulos indicados entre los corchetes “[ ]”.

**8.** Math.range = **function**( a, b );

El método range(a, b) genera un numero aleatorio entre el rango ingresado [a, b].

**18.** app.config([..., **function**(...) {...}]);

El método config() es utilizado para configurar el enrutamiento de la página, en este método se indica según qué dirección url se escriba se mostrara cual contenido.

**39.** app.controller('sesionController', [...,  
 **function**(...){...}]);

El método controller() es utilizado para indicar la programación referente al inicio, registro y cerrado de sesión en el controlador sesionController.

**42.** $scope.signup = **function**(){...};

El método signup() es el encargado de obtener los datos ingresados, verificarlos y en tal caso de ser todo correcto crear un nuevo usuario en la base de datos, o indicar que hubo un fallo en cualquier otro caso..

**64.** $scope.login = **function**(){...};

El método login() se encarga de obtener la información suministrada, verificarla e iniciar sesión si todo es correcto, o indicar que hubo un fallo en cualquier otro caso.

**85.** $scope.logout = **function**(){...};

El método logout() se encarga de cerrar la sesión del usuario.

**93.** app.controller('battleController', [...,  
 **function**(...){...}]);

El método controller() es utilizado para indicar la programación y lógica de las batallas que serán utilizados en el controlador battleController.

**126.** $scope.ataque = **function**( habilidad ){...};

El método ataque( habilidad ) es el método que se ejecuta cuando el usuario hace click a algunas de sus habilidades, la habilidad es enviado como argumento, y se procede a ejecutar el código correspondiente a su uso, y la respuesta de la inteligencia artificial.

**208. var** iaAtaque = **function**(){...};

El método iaAtaque() es el encargado de hacer las verificaciones correspondientes a la IA, y de procesar la habilidad escogida.

**297. var** evaluarVidas = **function**(...){...};

El método evaluarVidas() se encarga de evaluar la vida de ambos tamed y verificar si alguno de los dos murió para acabar la batalla.

**306. var** hurt = **function**(...){...};

El método hurt() es el encargado de restarle la vida al tamed que fue dañado.

**311. var** recoverLife = **function**(...){...};

El método recoverLife() es el encargado de recuperar la vida del usuario cuando utilice alguna habilidad de recuperación

**316. var** calculateDmg = **function**(...){...};

El método calculateDmg() es el encargado de calcular el daño que se le hará al enemigo en base a nuestro ataque, y poder de la habilidad y defensa del enemigo.

**321. var** buffHab = **function**(...){...};

El método buffHab() es el encargado de utilizar las habilidades de tipo buff en nosotros, y verificar cuando estás acabaron.

**357. var** debuffHab = **function**(...){...};

El método debuffHab() es el encargado de utilizar las habilidades de tipo debuff en el enemigo, y verificar cuando estás acabaron.

**400. var** ArtificialInteligent =

**function**(...){...};

El método ArtificialInteligent() es el encargado de hacer los cálculos correspondientes y en base a como se encuentre la batalla hacer una elección razonable de la habilidad a utilizar en el turno actual.

**721. var** powerfulHab = **function**(...){...};

El método powerfulHab() es el encargado de obtener la habilidad más fuerte del tipo que le indiquemos, o buscar la habilidad más fuerte del tipo que le indiquemos y que no sea “x” habilidad.

**745.** app.controller('myProfileController', [...,  
 **function**(...){...}]);

El método controller() es utilizado para indicar la programación que harán que se muestren las características de nuestro tamed en la página en el controlador myProfileController.

**779.** $scope.selectTamed = **function**(...){...};

El método selectTamed() es el encargado de asignar el tamed que seleccionamos en la página como nuestro.

**800. var** generateRandomIA = **function**(...){...};

El método generateRandomIA() es el encargado de elegir aleatoriamente un tamed entre todos, contra quien luchara el usuario.

**809. var** potenciar = **function**(...){...};

El método potenciar() es el encargado de aumentar las características del tamed que le pasemos como argumento según el nivel, y su potenciador.

**816. var** porcentajeVida = **function**(...){...};

El método porcentajeVida() es el encargado de calcular el porcentaje de vida actual del tamed que le pasemos como argumento.

**Archivo: services.js**

**316. var** calculateDmg = **function**(...){...};

**316. var** calculateDmg = **function**(...){...};

**Listado de métodos del lado del servidor:**

**Archivo: server.js**

**316. var** calculateDmg = **function**(...){...};