



STORIAN

Julián García Castillo
Proyecto final DAW

1. Manual del Usuario

1.1 Descripción de la Aplicación

Storian es una aplicación web que elabora historias en base a la solicitud del usuario. Por una parte recoge los datos requeridos para crear la historia: Por un lado nombre y tipo de personajes, y por otro lado el lugar donde desea que se desarrolle la aventura.

La aplicación preparará el entorno con ilustraciones de los personajes creados por el usuario, con música y el fondo con el lugar seleccionado. De este modo, cuando se muestre la historia, el lector ya se encuentra inmerso en la aventura que él mismo ha confeccionado.

Nada en Storian está predefinido, de hecho, si el usuario vuelve a crear un cuento con las mismas características, le va a ser mostrado otro completamente diferente.

Además, en Storian el lector también puede convertirse en escritor y mandar sus propias historias. Con ayuda de un asistente y sin necesidad de saber programar, el usuario, una vez registrado, podrá transformar sus propias historias en formato Storian para que puedan ser disfrutadas por todos y recreadas por distintos personajes y localizaciones.

¿Necesitas inspiración o simplemente relajarte?

La página principal de Storian está creada con la intención relajarte o servir de inspiración, atenúa la luz de tu habitación y déjate llevar con el sonido de la lluvia y las gotas cayendo por el cristal de tu pantalla, con truenos que iluminarán tu habitación y un fondo que cambiará eventualmente.

En serio, pruébalo.

1.2. Funcionalidades y Características

A modo de listado, se citan todas las funcionalidades de la aplicación:

- **Absoluta libre confección de personajes:** Tanto el tipo como el nombre del personaje serán creados por el usuario.
- **Detección de género:** Para que el proceso sea sencillo y natural, el usuario no deberá seleccionar el género del personaje, la aplicación lo detectará de acuerdo al nombre y tipo.
- **Combinación infinita de historias:** El límite lo pone el usuario con su selección.
- **Ilustraciones e imágenes personalizadas** en función de las elecciones del usuario.
- **Bandas sonoras ambientales:** Todas las bandas sonoras están libres de derechos de autor y pueden ser usadas incluso de forma comercial.
- **Ilustraciones hechas a mano:** Realizadas con aplicaciones de diseño en un ipad con pluma estilográfica y sensor de presión.
- **Creación de cuenta de usuario:** La cuenta de usuario sólo es obligatoria para enviar historias, no para leer, en caso de poseer cuenta, podrás llevar un control del numero de historias que has leído y escrito.
- **Manager Asistente** para el envío de historias.
- *Pronto:* Valoraciones de historias y reporte de error.
- *Pronto:* Muchas más ilustraciones e historias base.
- *Pronto:* Más lugares para elegir.

1.3 Uso y requisitos mínimos

El uso de Storian es bastante sencillo e intuitivo, sin embargo el área de envío de historias requiere algo de esfuerzo por parte del usuario.

Vamos a dividir las instrucciones de uso:

1.3.1 Leer

El usuario no tiene más que entrar en la web, tocar la puerta del centro y clickar LEER. El proceso se ha simplificado para que pueda ser realizado por niños.

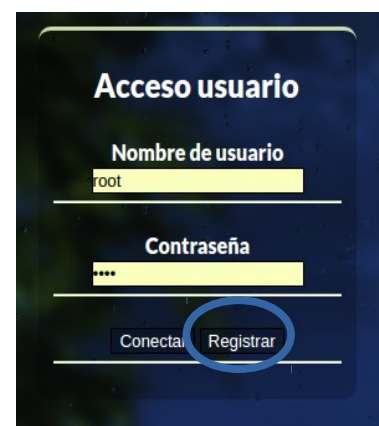
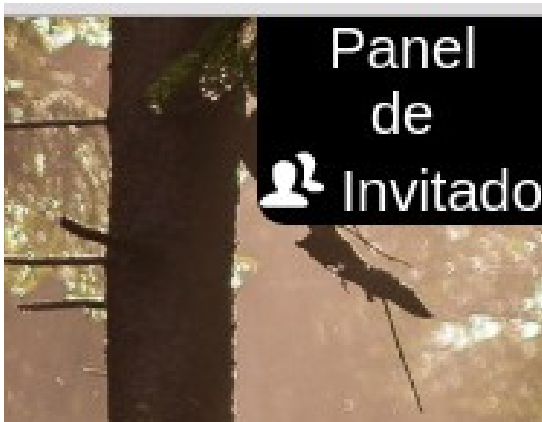


Una vez dentro del modo LEER, el usuario deberá ir completando el formulario y avanzar o bien, pulsando la cuerda o INTRO. Si no introduce un campo requerido, saltará una ventana de alerta.



1.3.2 Escribir

Para escribir, el usuario debe haberse registrado previamente, para hacerlo, pulsará en icono de la esquina superior derecha e ira al panel de acceso, una vez ahí pulsar, registrarse y seguir lo pasos.



Cuando haya finalizado el registro, accederá con sus nuevos datos y hará click en ESCRIBIR. Esto le llevará al asistente, en primer lugar serás requerido para introducir al menos, un personaje ¡Una historia necesita al menos uno!
Tras completarlo, verás la siguiente pantalla:

The screenshot shows a story creation interface with a dark, rainy background. At the top, there are two tables for selecting characters and a location. Below these is a title input field and two large text boxes for writing the story. A button 'ENVIAR HISTORIA' is at the bottom right.

PERSONAJE PRINCIPAL							PERSONAJE SECUNDARIO							LUGAR													
Nombre	Tipo	Genero	El	La	el	ella	un	una	lo	la	Nombre	Tipo	Genero	El	La	el	ella	un	una	lo	la	El	la	Tipo	un	una	Genero
David	perro	o	El	el	él		un		lo		Maria	ratita	a	La	la	ella		una		la		El	el	Castillo	un		o

Panel de lolopo

Area de trabajo

Título: GUARDAR TITULO

Título:

Esta es una historia de ejemplo sobre artDetP1 tipoP11 nombP11, que vivía en artDetLu nomLui encantadoaLu de Storian.

Esta es una historia de ejemplo sobre el perro David, que vivía en el Castillo encantado de Storian.

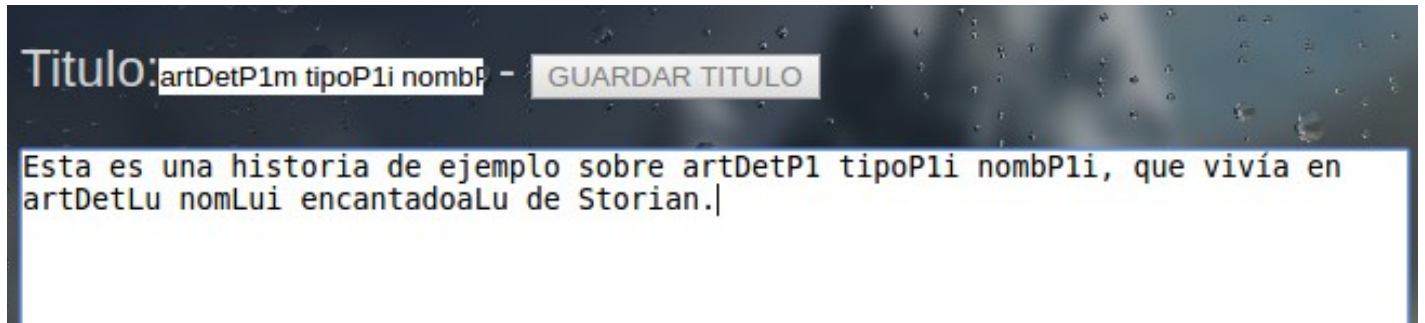
ENVIAR HISTORIA

Panel Superior: Contiene los valores de la historia que son sensibles al género, aquí está la clave de Storian, es importante que se usen los botones en lugar de escribir, por ejemplo, el nombre del personaje. Lo mejor es que pulses en ellos y ves como quedaría en la ventana de resultados.

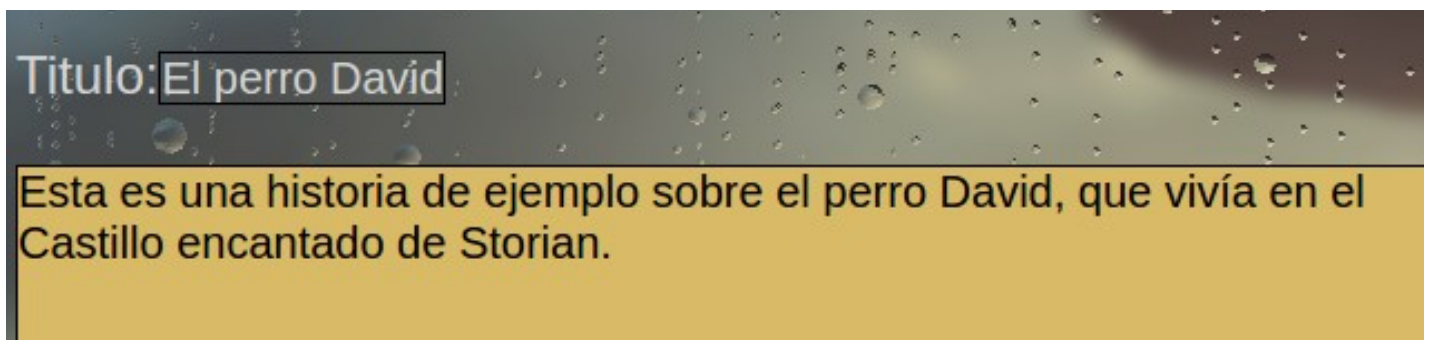
This is a close-up of the 'PERSONAJE PRINCIPAL' table from the previous screenshot. It shows the headers and the first row of data.

PERSONAJE PRINCIPAL										
Nombre	Tipo	Genero	El	La	el	ella	un	una	lo	la
David	perro	o	El	el	él		un		lo	

Para entender como funciona Storian, nada mejor que un ejemplo gráfico. Aquí va lo que tú escribes, como ves, aparecen codigos como *artDetP1*, no te preocupes, aparecen sólo cuando pulsas el botón asignado a esa palabra.



El cuadro marrón es donde se muestra el resultado de como quedaría una vez personalizado. Pasa el tiempo que necesites en esta zona para familiarizarte con el contenido.



Una vez que estes de acuerdo con el resultado, pulsa en ENVIAR HISTORIA ¡Y eso es todo! La historia llegará al servidor de Storian y una vez que sea aprobada por el guardián de los cuentos (o administrador), estará disponible para los usuarios.

Nota: La historia estará disponible pero no podrá ser elegida específicamente, los algoritmos aleatorios decidirán cuando aparece.

1.3.3 Requisitos mínimos

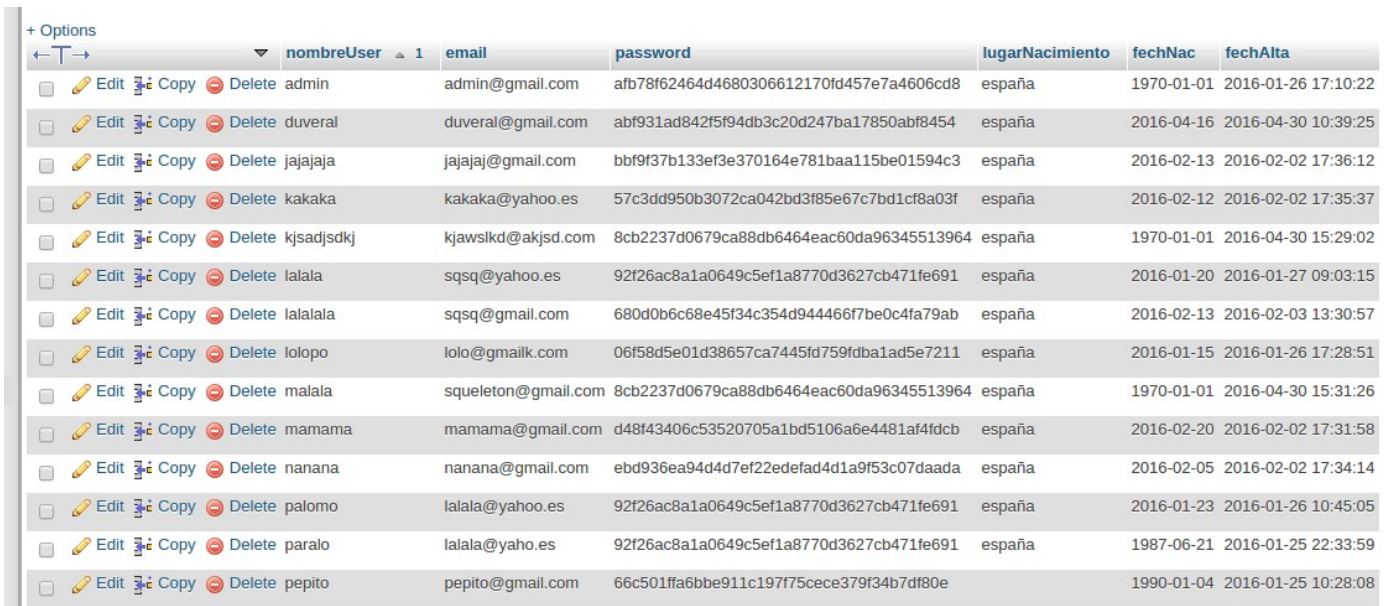
Como cualquier aplicación web, los requisitos mínimos son los básicos para correr cualquier web, la tecnología que recrea la lluvia se basa en canvas, que puede tener cierto peso en el rendimiento si permanece en la misma página, en este caso la página se refrescará de forma automática para evitar saturación en la memoria.

2. Manual Técnico

2.1 Modelo de Datos

La base de datos del proyecto se ha basado en la extensión PDO porque soporta hasta 12 tipos de drivers diferentes (Oracle, mySQL o incluso postgreSQL) mientras que MySQLi sólo soporta MySQL. Además es más sencillo realizar consultas, ya que usa un lenguaje algo más natural, o al menos lógico.

El sistema consta únicamente de dos tablas: USUARIO y CUENTOS.



	nombreUser	email	password	lugarNacimiento	fechNac	fechAlta
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	admin	admin@gmail.com	afb78f62464d4680306612170fd457e7a4606cd8	españa	1970-01-01	2016-01-26 17:10:22
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	duveral	duveral@gmail.com	abf931ad842f5f94db3c20d247ba17850abf8454	españa	2016-04-16	2016-04-30 10:39:25
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	jajajaja	jajajaj@gmail.com	bbf9f37b133ef3e370164e781baa115be01594c3	españa	2016-02-13	2016-02-02 17:36:12
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	kakaka	kakaka@yahoo.es	57c3dd950b3072ca042bd3f85e67c7bd1cf8a03f	españa	2016-02-12	2016-02-02 17:35:37
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	kjsadjsdkj	kjawsldk@akjsd.com	8cb2237d0679ca88db6464eac60da96345513964	españa	1970-01-01	2016-04-30 15:29:02
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	lalala	sqsq@yahoo.es	92f26ac8a1a0649c5ef1a8770d3627cb471fe691	españa	2016-01-20	2016-01-27 09:03:15
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	lalalala	sqsq@gmail.com	680d0b6c68e45f34c354d944466f7be0c4fa79ab	españa	2016-02-13	2016-02-03 13:30:57
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	lolopo	lolo@gmailk.com	06f58d5e01d38657ca7445fd759fdbad5e7211	españa	2016-01-15	2016-01-26 17:28:51
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	malala	skeleton@gmail.com	8cb2237d0679ca88db6464eac60da96345513964	españa	1970-01-01	2016-04-30 15:31:26
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	mamama	mamama@gmail.com	d48f43406c53520705a1bd5106a6e4481af4fdcb	españa	2016-02-20	2016-02-02 17:31:58
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	nanana	nanana@gmail.com	ebd936ea94d4d7ef22edefad4d1a9f53c07daada	españa	2016-02-05	2016-02-02 17:34:14
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	palomo	lalala@yahoo.es	92f26ac8a1a0649c5ef1a8770d3627cb471fe691	españa	2016-01-23	2016-01-26 10:45:05
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	paralo	lalala@yaho.es	92f26ac8a1a0649c5ef1a8770d3627cb471fe691	españa	1987-06-21	2016-01-25 22:33:59
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	pepito	pepito@gmail.com	66c501ffa6bbe911c197f75cece379f34b7df80e		1990-01-04	2016-01-25 10:28:08

Illustration 1: TABLA USUARIO

En la tabla USUARIO, nos encontramos con las siguientes columnas:

nombreUser: Nombre elegido por el usuario.

Email: Email del usuario.

Password: Encriptada desde el modelo por SHA1

lugarNacimiento: Campo no activo a la creación de este manual.

FechNac: Cuando nació el user, para determinar la edad.

FechAlta: Cuando se registró el usuario

numLeidos: Indica cuantas historias ha leído el usuario.

La PK es el campo del email, pero el nombre del usuario es un index único, para identificarlo de ambas formas. La cantidad de historias que ha escrito un usuario se realiza por consulta SQL.

En la tabla CUENTO, nos encontramos con las siguientes columnas:

	id	titulo	contenido	autor	fecha
e	18	artDetP2m tipoP2i nombP2i	Todas las mañanas, camino de artDetLu nomLui, artD...	pepito@gmail.com	2016-01-20 17:44:12
e	19	nombP1i, artDetP1 tipoP1i ricoaP1	nombP1i era detIndP1 tipoP1i, que soñaba con vola...	pepito@gmail.com	2016-01-21 11:20:45
e	27	artDetP1m tipoP1i pirata	Erase una vez detIndP1 tipoP1i pirata cuyo nombre ...	lolo@gmailk.com	2016-02-03 10:32:26
e	28	artDetP1m tipoP1i encantadoaP1	Erase un príncipe muy admirado en su reino. Todas ...	lolo@gmailk.com	2016-02-03 10:55:52
e	30	artDetP1m honradoaP1 tipoP1i del bosque	Había una vez detIndP1 pobre tipoP1i en el bosque ...	lolo@gmailk.com	2016-02-11 16:27:34
e	31	La bobina y artDetP1 tipoP1i perezosoaP1	Erase detIndP1 tipoP1i llamadoaP1 nombP1i que n...	lolo@gmailk.com	2016-02-11 16:39:16

id: Auto incrementable

titulo: Título de la historia.

contenido: La historia en sí.

autor: Usuario que ha enviado la historia.

fecha: Fecha de envío de la historia.

publicado: Todos los cuentos se crean sin publicacion, es un boolean que cambia el admin una vez aprobado.

La tabla de CUENTO contiene las historias codificadas que se usan en el programa de Storian, listas para interpretar y reconvertirse con las elecciones del usuario.

Relación: La relación de dependencia entre ambas tablas es de 1 a n. Un usuario puede tener muchos cuentos, pero un cuento pertenece sólo a un usuario.

Abajo, una captura de las clases definidas en la carpeta /storian/Model

```
10 class Usuario {
11
12     // Atributos de instancia
13     private $nombreUser;
14     private $email;
15     private $password;
16     private $lugarNac;
17     private $fechaNac;
18     private $fechaAlt;
19     //private $numLeidos;
20
21     // Constructor
22     function __construct($nombreUser, $email, $pa
23         $this->nombreUser = $nombreUser;
24         $this->email = $email;
25         $this->password = sha1($password);
26         $this->lugarNac = $lugarNac;
27         $this->fechaNac = $fechaNac;
28         $this->fechaAlt = $this->fechAltaVer();
29 }
```

```
class Historia {
    // Atributos de instancia
    private $id;
    private $titulo;
    private $contenido;
    private $autor;
    private $fecha;

    // Constructor
    function __construct($id, $titulo, $conteni
        $this->id = $id;
        $this->titulo = $titulo;
        $this->contenido = $contenido;
        $this->autor = $autor;
        $this->fecha = $fecha;
    }
```

2.1 Funcionamiento

El proceso mediante el cual una historia acaba mostrada en la pantalla es el siguiente.

1. El usuario completa los inputs requeridos para la creación de personajes y selección de lugar, entonces tira de la cuerda y comienza el trabajo.
2. Ante de la consulta AJAX se hace una **comprobación si existe una ilustración del tipo de personaje** elegido en el array de personajes(llamado animales).

Si es así, se elige esa imagen, sino, se escoge una imagen de ambiente.

</storian/Model/Animatios-menu.js>

```
Dado un array de raiz de nombres de personajes, si coincide con la entrada del tipo, se usa esa imagen
function imagenPersonajes(tipoP1, tipoP2) {
  var animales = ["pajar", "elefant", "os", "cocodril", "perr", "rat", "gat", "jiraf", "princes", "pat";
  var ambiente = ["arbol", "casa", "varita", "arbusto", "libro"];
  var elegP1 = false;
  var elegP2 = false;
  var rnd = 0;
  var rnd2 = 0;

  $.each(animales, function (key, value) {
    var personaje_array = new RegExp("^" + value + ".*", "gi");
    if (tipoP1.match(personaje_array)) {
      $('#imgP1').css({"background-image": "url(/storian/View/Assets/img/" + value + ".png)"});
      elegP1 = true;
      console.log("encuentra p1");
    }
    if (tipoP2.match(personaje_array)) {
      $('#imgP2').css({"background-image": "url(/storian/View/Assets/img/" + value + ".png)"});
      elegP2 = true;
      console.log("encuentra p2");
    }
  });
  // Si llega al final y no se ha encontrado similitud, elegir img random,
  //FIX : PERO DE ARRAY DE AMBIENTE
  if ((key == (ambiente.length) - 1) && (!elegP1)) {
    console.log("no ha encontrado similitud");
    console.log("longitud array:" + ambiente.length);
    rnd = Math.floor(Math.random() * (ambiente.length) + 0);
    rnd2 = rnd;
    $('#imgP1').css({"background-image": "url(/storian/View/Assets/img/" + ambiente[rnd] + ".png)"});
  }
  if ((key == (ambiente.length) - 1) && (!elegP2)) {
    console.log("no ha encontrado similitud con personaje 2");
  }
}
```


La comprobación se hace mediante una expresión regular de tipo **new RegExp**, que mediante una búsqueda dentro del array con **match**, busca coincidencias aunque sea parcialmente, por ello si el usuario escribe perro, como si escribe perrito, el resultado será **TRUE**.

3. Antes de mostrar nada, se carga la función encargada de **cerrar las cortinas**, son dos capas separadas, que se cierran dependiendo del ancho y alto de la pantalla, al mismo tiempo el background tiende a negro con una animación del opacity.

```
/* Crea y cierra cortinas */
function cortinas() {
    // Crea las cortinas
    var cortinaIzq = "<div class='cortina' id='cortinIzq' ></div>";
    var cortinaDcha = "<div class='cortina' id='cortinDcha' ></div>";
    $('body').append(cortinaIzq, cortinaDcha);

    // Cierra las cortinas
    $('#cortinIzq').animate({left: "-15%"}, 3000);
    $('#cortinDcha').animate({left: "50%"}, 3000);
    $('#fondoCortina').fadeIn(2000);
    // Abre las cortinas
    setTimeout(function () {
        $('#cortinIzq').animate({left: "-80%"}, 3000);
        $('#cortinDcha').animate({left: "105%"}, 3000);
        $('#fondoCortina').fadeOut(2000);
        $('#contenedorHistoria').fadeIn(1000);
    }, 6000);
}
```

4. Entonces, se **carga la función Cargando**, que muestra engranajes de carga, durante 7 segundos y una frase de espera que es aleatoria.

```
// Pone logo cargando y texto rand
function cargando() {
    $('#loading').show();
    var rnd = Math.floor((Math.random() * 7) + 0);

    var arrayText = [
        'Los personajes se estan preparando, el escenario esta casi listo',
        '¡Preparate para la aventura!',
        'Mezclando los personajes y el escenario',
        'La aventura esta a punto de comenzar',
        'Aguarda un instante...',
        'Creando la magia',
        'Espera a que todo este listo',
        'Los personajes estan siendo instruidos.'];

    var elemento = "<div class='centered'>" + arrayText[rnd] + "</div>";
    $('#loading').append(elemento);
    console.log(elemento);
}
```

5. Entonces, comienza la cola de ajax. **Se carga la funcion ambiente**, que cambia el fondo css del background y la musica:

```
// Cambia el fondo y la musica
function cambioAmbiente() {

    // Crear y reproducir random music
    var rnd = Math.floor((Math.random() * 23) + 1);
    var audio = "<audio loop autoplay id='music' src='/storian/View/Assets/music/" + rnd + ".mp3' ></audio>";
    $('body').append(audio);
    document.getElementById('music').volume = 0.1;

    // Cambiar fondo
    var lugarSelect = $('#lugar').val();
    $('body').css('background-image', 'url(/storian/View/Assets/img/lugares/' + lugarSelect + '.jpg)');
}
```

El código es sencillo, se usa un valor random entre el numero de pistas que hay, y se crea un audio. Para el fondo, se escoge el valor elegido por el user en el formulario y se cambia el css con jquery.

6. La **estructura de la cola** es la siguiente, se van retrasando las funciones con varios timeout:

```
$.ajax({
    type: "POST",
    url: "modos/leer/cuento",
    dataType: 'json',
    data: {nombreP1: nombreP1, tipoP1: tipoP1, nombreP2: nombreP2},
})
.done(function (data) {
    console.log(data['contenido']);
    setTimeout(function () {
        simulEscribe(data);
    }, 7000);
    // Cambia musica y fondo css
    setTimeout(function () {
        cambioAmbiente();
    }, 7000);
    // Muestra el logo cargando y un texto random
    setTimeout(function () {
        cargando();
    }, 3000);
    setTimeout(function () {
        $('#loading').hide();
    }, 5500);
})
.fail(function () {
    alert("Algo ha superpetado");
});
```


7. Por último, la data que devuelve la función procesar.php (mostrada más adelante), la recibe la función simulEscribe(), que se encarga de **mostrarla con efecto de escritura**, usando un plugin, una vez que el resto de elementos se ha ocultado.

```
/* Escribe la el cuento con efecto poco a poco */  
function simulEscribe(data) {  
    var titulo = data['titulo'];  
    var contenido = data['contenido'];  
    $("#contenedorHistoria").find('#tituloHistoria').typed({  
        strings: [titulo],  
        typeSpeed: 0  
    });  
    $("#contenedorHistoria").find('#contenidoHistoria').typed({  
        strings: [contenido],  
        typeSpeed: 0  
    });  
}
```

2.2 Algoritmo Storian

El núcleo de Storian lo define la principal característica que permite la confección de los cuentos, su algoritmo.

PHP ofrece muchas ventajas para alteración y búsqueda de strings, eso, más la ayuda de expresiones regulares, permiten el desarrollo de esta app.

No es un algoritmo muy complicado, pero es necesario que esté bien estructurado porque es muy largo, y para ello, el uso de las clases facilita mucho ya que los objetos vienen ya con su género detectado y da un aspecto más limpio al código.

2.2.1 Detección género personajes:

La función se encuentra en la clase del modelo y se determina el género en función de las últimas letras del tipo y el nombre (subsrty), usando ciertas excepciones.

[/storian/Model/Usuario](#)

Es decir, **cuando se crea el personaje, automáticamente se determina el sexo.**

```

// Devuelve un sexo Personaje
public function sexo(){
    $ultLetraNom = substr($this->getNombre(), -1) ;
    $ultLetraTipo = substr($this->getTipo(), -1) ;
    $exceptNomFem = array ("Rocio","Vero","Amparo");
    $exceptNomMasc = array("Rafa","Seba");
    $exceptTipoMasc = array("aguila","pez");
    $exceptTipoFem = array("serpiente","anguila");

    if (( $ultLetraTipo=="a" ) || (( $ultLetraTipo=="e" ) && ( $ultLetraNom=="a" ))){
        $this->sexo = "femenino";
    }else{
        $this->sexo = "masculino";
    }

    // Excepciones Nombre Fem
    foreach($exceptNomFem as $value){
        if ($this->getNombre() == $value){
            $this->sexo = "femenino";
        }
    }

    // Excepciones Nombre Masc
    foreach($exceptNomMasc as $value){
        if ($this->getNombre() == $value){
            $this->sexo = "masculino";
        }
    }

    // Excepciones Tipo Masc
    foreach($exceptTipoMasc as $value){
        if ($this->getTipo() == $value){
            $this->sexo = "masculino";
        }
    }
}

```

2.2.2 Codificación de la historia:

Una vez que los personajes ya tienen asignados un género, es hora de buscar las variables sensibles a cambio de sexo, como artículos, pronombres, adverbios... Para ello, las historias están guardadas usando claves únicas en dichas variables, será el php el que buscará esas claves y las cambiará dependiendo del género.

Además de los personajes, el lugar también tiene un género, no es lo mismo UN Castillo, que UNA playa. Una vez que tenemos todo creado, sucede lo siguiente:

</storian/Controller/procesarHistoria.php>

```

1 // crear personajes y lugar
2 $personaje1 = new Personaje($nombreP1, $tipoP1);
3 $personaje2 = new Personaje($nombreP2, $tipoP2);
4 $lugar = new Lugar($nomLui);
5
6 // Articulos
7 // Personaje 1
8 if ($personaje1->getSexo() == "masculino"){
9     $artDetP1m = "El"; $artDetP1 = "el"; $oaP1 = "o"; $pronP1 = "él"; $detIndP1 = "un"; $neutP1 = "lo";
10 }else{
11     $artDetP1m = "La"; $artDetP1 = "la"; $oaP1 = "a"; $pronP1 = "ella"; $detIndP1 = "una"; $neutP1 = "la";
12 }
13 // Personaje 2
14 if ($personaje2->getSexo() == "masculino"){
15     $artDetP2m = "El"; $artDetP2 = "el"; $oaP2 = "o"; $pronP2 = "él"; $detIndP2 = "un"; $neutP2 = "lo";
16 }else{
17     $artDetP2m = "La"; $artDetP2 = "la"; $oaP2 = "a"; $pronP2 = "ella"; $detIndP2 = "una"; $neutP2 = "la";
18 }
19 // Lugar
20 if ($lugar->getSexo() == "masculino"){
21     $artDetLum = "El"; $artDetLu = "el"; $oaLu = "o"; $detIndLu = "un";
22 }else{
23     $artDetLum = "La"; $artDetLu = "la"; $oaLu = "a"; $detIndLu = "una";
24 }
25 // Selecciona historia random de DB
26 $historiaRnd = Historia::random();
27 $historia = new Historia($historiaRnd->id, $historiaRnd->titulo, $historiaRnd->contenido, $historiaRnd->autor, $his
28
29 // Aumentar numero de Cuentos leidos por ese user
30 // $usuario = new Usuario($usuarioSes, "", "", "", "", "");
31 // $usuario->addNumLeidos();
32
33 // Extrae titulo y contenido y autor
34 $titulo = $historia->getTitulo();
35 $contenido = $historia->getContenido();
36 $sautor = $historia->getAutor();

```

En este caso se declaran las variables en todos los géneros, se busca una historia random usando la propia clase, y con esos datos se crea la historia, aún queda descodificarla, se realiza con el str_replace del tipo

```
$titulo = str_replace("artDetP1m",$artDetP1m,$titulo);
```

Es decir, SI encuentras, y sólo SI encuentras equivalencia, reemplaza. La razón de usar str_replace es porque de ese modo no resulta error en caso de no encontrar alguno de los códigos.

Con lo que unas líneas más abajo nos encontramos con la zona más tediosa del algoritmo pero la más fácil. El archivo procesarHistoria.php es posiblemente lo más parecido al cerebro de Storian. Dicha descodificación se debe realizar tanto en el titulo, como la historia, y por distinto a los dos personajes y lugar, por eso es tan larga.


```

55
56 // Personaliza historia
57     // Titulo
58     // Personaje 1
59     $titulo = str_replace("artDetP1m", $artDetP1m, $titulo);
60     $titulo = str_replace("artDetP1", $artDetP1, $titulo);
61     $titulo = str_replace("nombP1i", $nombreP1, $titulo);
62     $titulo = str_replace("tipoP1i", $tipoP1, $titulo);
63     $titulo = str_replace("oaP1", $oaP1, $titulo);
64     $titulo = str_replace("pronP1", $pronP1, $titulo);
65     $titulo = str_replace("detIndP1", $detIndP1, $titulo);
66     $titulo = str_replace("neutP1", $neutP1, $titulo);
67     // Personaje 2
68     $titulo = str_replace("artDetP2m", $artDetP2m, $titulo);
69     $titulo = str_replace("artDetP2", $artDetP2, $titulo);
70     $titulo = str_replace("nombP2i", $nombreP2, $titulo);
71     $titulo = str_replace("tipoP2i", $tipoP2, $titulo);
72     $titulo = str_replace("oaP2", $oaP2, $titulo);
73     $titulo = str_replace("pronP2", $pronP2, $titulo);
74     $titulo = str_replace("detIndP2", $detIndP2, $titulo);
75     $titulo = str_replace("neutP2", $neutP2, $titulo);
76     // Lugar
77     $titulo = str_replace("artDetLum", $artDetLum, $titulo);
78     $titulo = str_replace("artDetLu", $artDetLu, $titulo);
79     $titulo = str_replace("oaLu", $oaLu, $titulo);
80     $titulo = str_replace("detIndLu", $detIndLu, $titulo);
81     $titulo = str_replace("nomLui", $nomLui, $titulo);
82
83     // Historia
84     // Personaje 1
85     $contenido = str_replace("artDetP1m", $artDetP1m, $contenido);
86     $contenido = str_replace("artDetP1", $artDetP1, $contenido);
87     $contenido = str_replace("nombP1i", $nombreP1, $contenido);
88     $contenido = str_replace("tipoP1i", $tipoP1, $contenido);
89     $contenido = str_replace("oaP1", $oaP1, $contenido);

```

Controller/procesaHistoria.php 81:53

2.2.3 ¡ Envio Historia !

El envio a la llamda ajax es algo tan sencillo como agrupar la historia ya decodificada en un array y enviarla como json. De esa forma podemos colocar el titulo y el contenido en diferentes <div>.

```

$arrayHistoria = array(
    "titulo" => $titulo,
    "contenido" => $contenido,
    "autor" => $autor,
);

```

2.3 Tecnologías usadas

JQUERY

Se usa durante toda la app, pero donde más importancia cobra es en la pagina de LEER, es la encargada de manejar los eventos de clicks, animar y crear elementos.

Ademas cabe destacar el uso de JQUERY Validation, y JQUERY UI para las ventanas de dialogo y las comprobaciones de formularios.

AJAX

Todas las llamadas que se realizan en esta web se realizan a través de Ajax, permite una navegación más rápida y atractiva para una aplicación de este tipo. Los datos se pasan como JSON.

ANGULAR

En este caso, su uso se restringe al punto de información de la pagina principal, se usan controladores y un array para cargar las funcionalidades y así aligerar el código html.

CSS3

De nuevo, al igual que jquery, CSS es usado por toda la app, sin ningún tipo de framework, adaptándose a diferentes tamaños de pantalla por medio de media queries, se ha realizado un logo con inkscape, y todos los iconos están en formato svg, que permite animarlos y no tener que usar jquery ni css, sino sus propiedades internas.

CANVAS

En este caso, su uso se limita al script de la lluvia, mi trabajo en este aspecto ha consistido en adaptarlo, darle diferentes zIndex ya que el original no permitía pintar nada encima, y por otro lado, he añadido los truenos de forma aleatoria.

REDES

El sitio web está alojado en un servidor que corre apache y se puede tener acceso a él via FTP, además las rutas URL son amigables gracias al añadido de un archivo .htaccess.