ESCUELA TECNICA SUPERIOR DE
INGENIERIA DE TELECOMUNICACION
UNIVERSIDAD POLITECNICA DE CARTAGENA

Trabajo Fin de Grado

Plataforma web para administración y análisis de un juego RPG

Web platform for administration and analysis of an RPG game.

AUTOR: Manuel Hernández Bastida
DIRECTOR: Juan Ángel Pastor Franco
Octubre de 2018

Autor	Manuel Hernández Bastida
E-mail del Autor	manu.hernandez.bastida@gmail.com
Director	Juan Ángel Pastor Franco
E-mail del director	juanangel.pastor@upct.es
Título del TFG	Plataforma web para administración y análisis de un juego RPG
Descriptores	Sitio web, REST, Shiny

Resumen

El siguiente proyecto fin de estudios busca proporcionar un soporte web a un juego RPG para la administración de cuentas, noticias y estadísticas de análisis del juego.

El proyecto se desarrolla sobre un juego RPG ya definido (juego base) y consta de una serie de tareas que vinculan los campos del diseño web, del Big data y de la programación avanzada.

Las tareas principales del proyecto son (1) el desarrollo de un sitio web asociado al juego, donde se ofrezca información relacionada con el juego, permitiendo a un usuario registrarse y descargar el juego, (2) la recopilación de información del usuario en el juego para ofrecerle así estadísticas relacionadas con sus partidas y (3) garantizar la seguridad de los datos.

Titulación	Grado en Ingeniería Telemática
Departamento	Tecnologías de la Información y las Comunicaciones
Fecha de presentación	Octubre 2018

Agradecimientos especiales para
Daniel Pérez Berenguer,
Mathieu Kessler
y Juan Ángel Pastor
por su ayuda a largo de esta Odisea.

A todos mis amigos y familiares
por el apoyo que siempre me habéis dado.

Manuel Hernández Bastida

Resumen

El siguiente proyecto fin de estudios busca proporcionar un soporte web a un juego RPG para la administración de cuentas, noticias y estadísticas de análisis del juego.

El proyecto se desarrolla sobre un juego RPG ya definido (juego base) y consta de una serie de tareas que vinculan los campos del diseño web, del Big data y de la programación avanzada.

Las tareas principales del proyecto son (1) el desarrollo de un sitio web asociado al juego, donde se ofrezca información relacionada con el juego, permitiendo a un usuario registrarse y descargar el juego, (2) la recopilación de información del usuario en el juego para ofrecerle así estadísticas relacionadas con sus partidas y (3) garantizar la seguridad de los datos.

Abstract

The following end of studies project aims to provide a web support to an RPG game for the administration of accounts, news and game analysis statistics.

The project is developed on a RPG game already defined (base game) and consists of a series of tasks that link the fields of web design, Big Data and advanced programming.

The main tasks of the project are (1) the development of a website associated with the game, where information related to the game is offered, allowing a user to register and download the game, (2) the collection of user information in the game to offer you statistics related to your games and (3) to guarantee the security of the data.

Índice

Índice de ilustraciones

Índice de tablas

Capítulo 1 Introducción

Introducción

En este trabajo se presenta y desarrolla un sitio web para dar soporte a un juego RPG multiplataforma, el cual busca ofrecer a sus usuarios/jugadores un portal donde poder consultar su progreso y evolución, además de información acerca del juego.

Utilizaremos tecnologías y aplicaciones web que nos permitirán crear un sitio web donde un usuario pueda obtener información acerca del juego y sus condiciones, realizar acciones de registro e inicio de sesión, modificación de datos personales de la cuenta, consulta de estadísticas y donaciones, todo ello con diseños interactivos que permitan al usuario una agradable experiencia.

Para ello, analizaremos los recursos disponibles e intentaremos realizar un diseño básico que cumpla los requisitos y, además, permita la escalabilidad y ampliación de las funcionalidades del sitio web. Posteriormente realizaremos las correspondientes pruebas para obtener los resultados deseados.

1.1 Antecedentes

Para entender que se va a desarrollar es necesaria una breve explicación de lo que es un sitio web, un juego RPG y cuál es la relación entre ambos en este proyecto.

- RPG: del inglés *role-playing game*, es un género de videojuegos que incluye una amplia variedad de sistemas y estilos de juego. Una de las características asociadas a los juegos de rol son el desarrollo estadístico de personajes permitiendo al usuario progresar a lo largo de sus partidas.
- Sitio web: es el conjunto de archivos y páginas web que ofrecen al usuario contenido sobre algún tema, y que le permiten ciertas funcionalidades para poder interactuar con él.

Actualmente, el tratamiento de los datos, Big Data[1], permite ofrecer una gran variedad de servicios al usuario que los genera. En los juegos RPG, donde existen una gran cantidad de usuarios generando datos surge la oportunidad de ofrecer un servicio. Los desarrolladores del juego quieren aprovechar esa oportunidad, por un lado, para que el usuario que juegue también pueda disfrutar de un sitio web donde visualice los progresos y, por otro, con vistas al futuro usar esos datos para mejorar el juego.

El material necesario para el diseño de la web será proporcionado por los compañeros encargados del desarrollo del juego[2], donde encontraremos los siguientes contenidos multimedia:

[1] Big Data: Concepto que hace referencia al tratamiento de los datos masivos.
[2] Desarrollo del juego: Desarrollo de un juego RPG para prueba de algoritmos de IA con plataforma Unity e integración con la nube. Salvador Murcia Martínez

- Videos: que proporcionan al usuario una visualización previa de los contenidos del juego tales como los personajes, sus ataques y una serie de escenas de una parte del juego.
- Imágenes: para el diseño de la interfaz ofreciendo un diseño ambientado en el juego y sus personajes.

Además de los contenidos multimedia proporcionados, se requiere de una base de datos compartida entre el sitio web y el juego, para que sea capaz de recoger los datos que el juego genera y posteriormente utilizarlos.

1.2 Objetivos

Para dar un soporte adecuado al juego se determinan los siguientes objetivos para el sitio web:

- Proporcionar un diseño intuitivo, interactivo y responsivo: capaz de adaptarse a todos los usuarios y medios fácilmente, proporcionándoles una interfaz ambientada en el juego.

- Proporcionar medidas de seguridad: en las funcionalidades del sitio web se utilizan datos sensibles que deben ser protegidos y tratados de la forma correcta a través de un servicio REST que atienda nuestras peticiones, instalación de certificados y utilización de métodos de cifrado.

- Desarrollar un servicio de análisis de datos: encargado de analizar los datos del usuario para ofrecer estadísticas, publicidad personalizada y sugerencias a través de una aplicación web.

1.3 Estructura de la memoria del proyecto

La memoria del proyecto está organizada por capítulos de diferentes dimensiones y contenido, y para facilitar la lectura, haremos una breve explicaciones de estos:

- Capítulo 1: Introducción

  En este capítulo se nos presenta al proyecto, dándonos una visión general de este para permitirnos entender mejor lo que estamos desarrollando.

- Capítulo 2: Tecnologías y requisitos previos

  En el segundo capítulo veremos las tecnologías actuales tanto para el diseño web como la programación necesaria para las aplicaciones del sitio web, incluyendo desde lenguajes

clásicos a la utilización de los frameworks [3]que nos facilitan el trabajo. Se hablará del software necesario, pero las instalaciones de estos estarán referenciadas a los anexos correspondientes del capítulo 5.

- Capítulo 3: Desarrollo del proyecto

  El capítulo 3 es el más extenso, y muestra el desarrollo del proyecto, divido en fases. En el apartado 1.5 del capítulo 1 se explicarán brevemente esas fases y, en este, se desarrollarán.

- Capítulo 4: Conclusiones y líneas futuras

  En este capítulo se sacan conclusiones y posibles ideas para el futuro del sitio web.

- Capítulo 5: Anexos

  En el capítulo 5 encontraremos información relacionada con la instalación, configuración y puesta en marcha del software, así como alguno ejemplo de los primeros pasos a seguir a la hora de empezar a desarrollar.

- Capítulo 6: Bibliografía

  La bibliografía contiene las fuentes de información necesarias y utilizadas para la realización del proyecto.

1.4 Estructura de los subproyectos

Este proyecto está formado a su vez por varios proyectos que interactúan entre sí. Para que pueda entender con claridad la distribución de archivos que los forman y como se relacionan entre sí, se hará continuación una breve explicación de cada uno ellos.

Podemos distinguir entre tres proyectos diferentes: el sitio web, la aplicación web API REST y la aplicación web Shiny/R. Cada uno de ellos hace uso de tecnologías diferentes para poder ofrecer al usuario final el mejor resultado posible.

1.4.1 Estructura del sitio web

El sitio web se estructura en carpetas cuyos nombres hacen referencia su contenido, agrupando archivos que utilizan la misma tecnología.

[3] Framework: es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular.

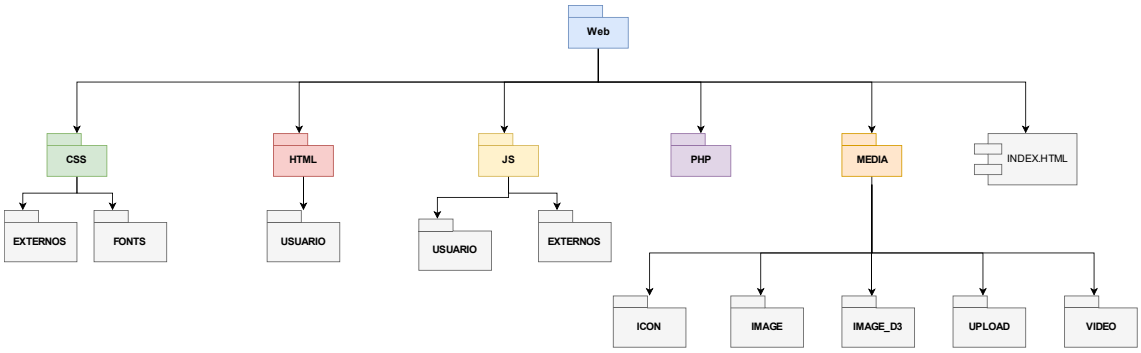

Ilustración 1: Estructura de carpetas del sitio web

La carpeta "CSS" incluye las subcarpetas "Externos" y "Fonts", las cuales albergan hojas de estilo creadas por otros usuarios, como por ejemplo las que ofrece Bootstrap. Dentro también encontraremos las hojas de estilo creadas por el autor del TFG: style.css, generalStyle.css, indexStyle.css, general_style.css, login_informacion_usuario_style.css y usuario_style.cs, las cuales modelan en su mayoría las páginas del sitio web disponibles en la carpeta "HTML".

La carpeta "HTML" incluye la subcarpeta "Usuario", la cual contiene los archivos .html que corresponde a la parte de la web que esta modelada con el framework de AngularJs, y los archivos login.html e información.html que se explicarán más adelante. Los documentos HTML hacen usos de los scripts de JavaScript disponibles en la carpeta JS para ofrecer características y funcionalidades al usuario que se utilice la web.

La carpeta "JS" compuesta por las subcarpetas "Externos", que al igual que la carpeta "Externos" del directorio "CSS" contiene scripts que son de otros desarrolladores, y "Bubble-chart" que contiene los scripts necesarios para la animación de D3 de JavaScript. De esta última nos interesan sobre todos los archivos que están dentro de la subcarpeta "Test", test_bubble.js y datos_heros.json, que veremos más adelante. La carpeta "JS" también está formada por scripts que añaden funcionalidad a los documentos HTML de la carpeta "HTML".

La carpeta "PHP" contiene tres scripts: upload.php, callRest.php y actívate.php, que realizaran la comunicación entre el servidor y la aplicación web API REST, y subir imágenes al servidor.

La carpeta "MEDIA", es la carpeta con el contenido multimedia proporcionado por los desarrolladores del juego. Contiene videos e imágenes para el diseño web.

El archivo index.html, es la página principal de nuestra web, es decir, la primera página que vera el usuario cuando se introduzca en nuestro sitio web.

1.4.2 Estructura de la aplicación web API REST

La estructura de la aplicación web API REST que se presenta a continuación hace referencia a las partes desarrolladas por el autor del TFG. El proyecto en sí está formado por más paquetes pero que se crean por defecto y que no se han modificado por ello aquí no se mencionan. En los anexos, podrá ver el proceso de creación del proyecto y su estructura completa.

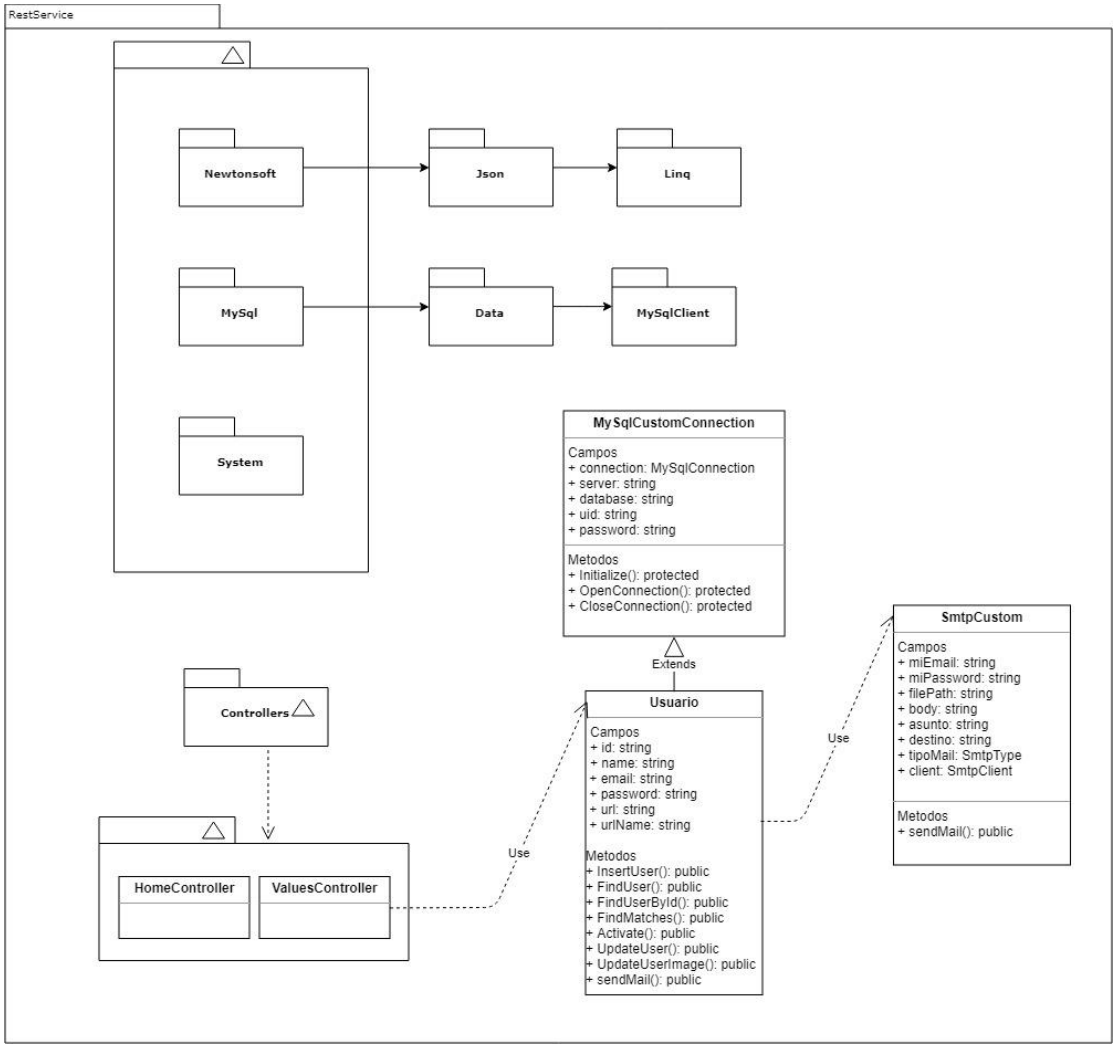

Ilustración 2: Estructura del proyecto aplicación web API REST

Principalmente nuestro desarrollo se centrará en la clase "ValuesController.cs" dentro del paquete "Controllers" y las clases "MySqlCustomConnection", "Usuario" y "SmtpCustom".

La explicación de las funcionalidades de las clases se hará más adelante en el capítulo 3.

1.4.3 Estructura de la aplicación web Shiny/R

La estructura de la aplicación de web Shiny parte de un directorio ya creado "aplicación_web_shiny" donde crearemos el proyecto "partidas_ganadas". Este proyecto hará uso de los scripts app.R, get_data.js y style.css para ofrecer al usuario agradables y vistosas graficas con sus progresos en el juego.

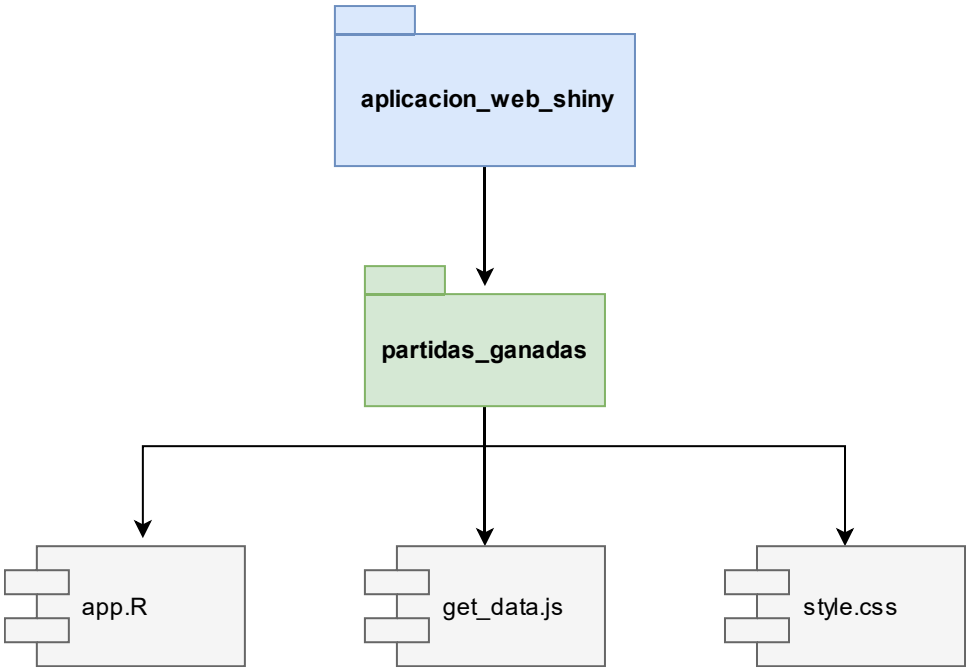

Ilustración 3: Estructura del proyecto aplicación web Shiny/R

1.5 Fases del desarrollo del proyecto

Dadas las dimensiones del proyecto, resulta más cómodo y eficiente dividirlo en una serie de fases que permitan un desarrollo fluido.

A continuación, se explicarán brevemente las mismas.

1.5.1 Modelar base de datos

En esta fase, haremos un estudio sobre la estructura que debe seguir nuestra base de datos. Crearemos un modelo entidad-relación con el que hacer las tablas que utilizaremos en el sitio web y, además, se realizaremos las consultas apropiadas para poder utilizarlas más tarde.

1.5.2 Desarrollo servicio web REST API

Durante la fase del desarrollo del servicio REST API, explicaremos como crear una solución en Visual Studio 2017 capaz de responder a peticiones HTTP[4], generalmente del tipo POST[5], ajustándose al modelo de objetos diseñado para nuestro sitio web. Se presentarán las clases necesarias, así como, el intercambio de mensajes entre el sitio web y nuestra solución.

[4] HTTP: es el protocolo de comunicación que permite las transferencias de información en la World Wide Web.
[5] POST: Método de petición del protocolo HTTP donde los datos se incluyen en el cuerpo del mensaje.

1.5.3 Desarrollo aplicación web Shiny/R

En la fase de desarrollo de una aplicación web con Shiny, nos enseñara los diferentes paquetes de R en RStudio que utilizaremos para realizar graficas a partir de un conjunto de datos guardados en un objeto JSON. Con detalle se verán los procesos de manipulación de datos y como realizar una aplicación web con Shiny.

1.5.4 Desarrollo Front-end:

En la fase de diseño y desarrollo del Front-end nos dedicaremos a la interfaz web y las funcionalidades para que el usuario pueda interactuar con ella.

Desarrollaremos casi todo lo que vemos en la pantalla cuando accedes a una web. Front-end, la estructuración de los apartados, tamaños, márgenes entre estructuras, tipos de letra, colores, adaptación para distintas pantallas, los efectos de ratón, teclado, movimientos, desplazamientos, efectos visuales...

Se intentará explicar cada uno de los componentes y como han sido diseñados utilizando lenguajes como HTML, CSS y JAVASCRIPT. Intentaremos que el usuario disfrute de la inmersión y usabilidad de nuestro sitio web. El uso de frameworks, como Bootstrap o AngularJs, también estará presente, ya que hoy en día han aumentado sus funcionalidades permitiendo grandes mejoras para el diseño web.

1.5.5 Desarrollo Back-end

En esta fase tenemos el Backend, enfocado en hacer que todo lo que está detrás de un sitio web funcione correctamente. Veremos cómo utilizando PHP podremos subir imágenes, enviar peticiones al servicio REST, o comunicarnos con la aplicación web de Shiny/R.

1.5.6 Pruebas

Llegados a esta fase, veremos las posibles variantes del sitio web ante las acciones del usuario. Se mostrarán datos sobre comunicación entre el servicio REST y el usuario.

1.6 Metodología de trabajo

La metodología seguida para desarrollar cada subproyecto sigue los siguientes pasos:

- Paso 1: Elegir subproyecto con el que empezar a trabajar.
- Paso 2: Elegir las tecnologías apropiadas para ese subproyecto elegido.
- Paso 3: Desarrollar las funciones, documentos, scripts y/o clases que necesite el subproyecto elegido.
- Paso 4: Comprobar funcionamiento.

Y la metodología seguida para desarrollar el proyecto en su totalidad se muestra en la siguiente ilustración:

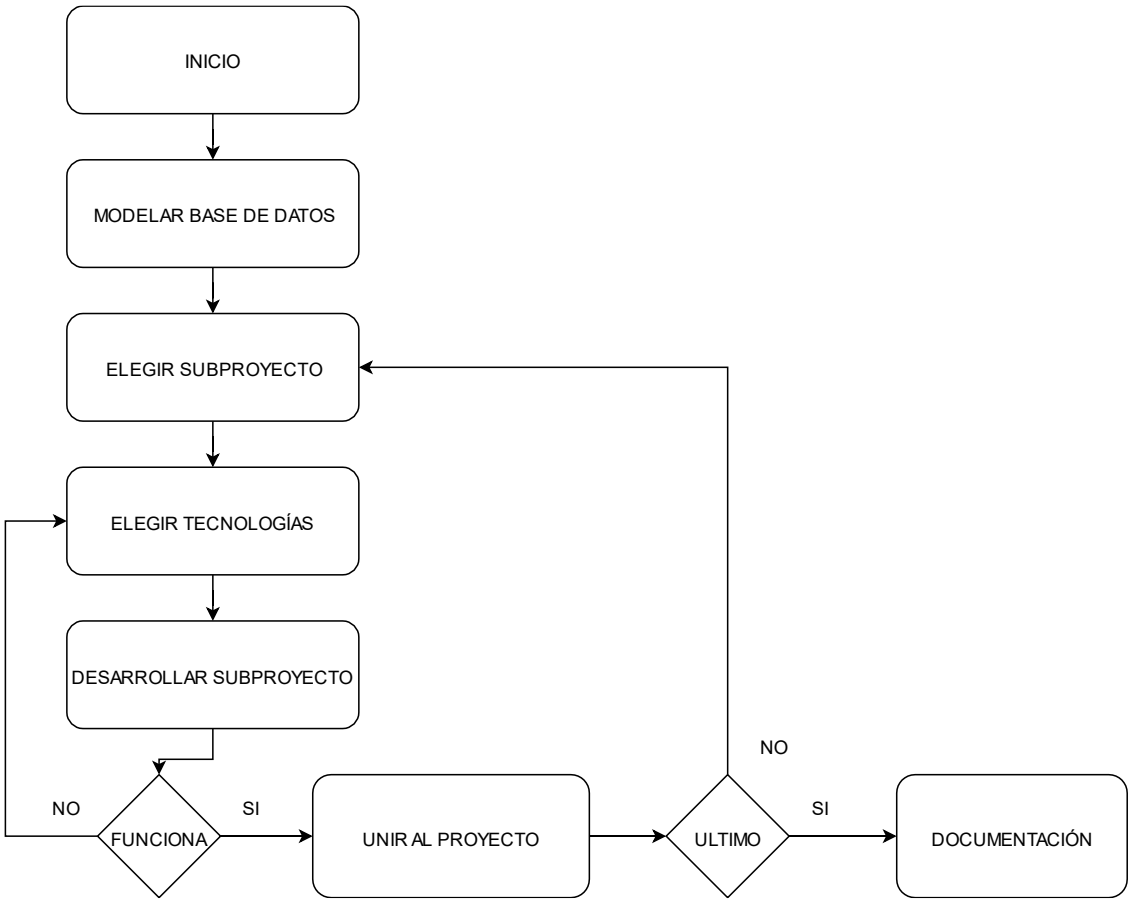

Ilustración 4: Diagrama de la metodología de trabajo seguida

Capítulo 2 Tecnologías y requisitos previos

Tecnologías y requisitos previos

Antes de empezar a desarrollar el proyecto, debemos saber que herramientas y tecnologías utilizar. Dada la cantidad disponible recursos, haremos una selección de los que nos serán más útiles. Hoy en día, la mayoría de información la podemos encontrar en las paginas oficiales de estas tecnologías y software, pero también hemos utilizado algunos libros con contenido practico que muestran ejemplos de utilización. Durante la memoria, se harán alguna referencia a estos libros, pero para lo demás consultar la pagina web oficial de la tecnología o software en cuestión disponibles también en el capítulo 6, Bibliografía.

Ilustración 5: Dispositivos multimedia que utilizan la web

Por un lado, veremos las posibles tecnologías que nos acompañaran en nuestro proyecto, y, por otro lado, el software necesario para poder utilizarlas correctamente.

Ilustración 6: Logotipos de las tecnologías web más utilizadas

2.1 Tecnologías

A continuación, se describen las diferentes tecnologías disponibles para nuestro proyecto. Se realizará una descripción breve pero técnica intentando explicar el funcionamiento y los resultados que buscan ofrecer cada una de ellas.

2.1.1 HTML

HTML hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código para la definición de contenido de una página web como texto, imágenes, videos y juegos entre otros.

Al ser un estándar[6], HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma por cualquier navegador web actualizado.

2.1.2 CSS

CSS es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritas en HTML.

CSS está diseñado principalmente para marcar la separación del contenido del documento y la forma de presentación de este buscando mejorar la accesibilidad del documento, proveer más flexibilidad y control en la especificación de características visuales de los elementos HTML.

La especificación CSS describe un esquema prioritario para determinar qué reglas de estilo se aplican si más de una regla coincide para un elemento en particular con un sistema llamado de cascada.

Para más información véase Gómez, M. R. (2013).

2.1.3 Bootstrap

Bootstrap es un framework para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales.

Soporta diseños web responsivos. Esto significa que el diseño gráfico de la página se ajusta dinámicamente, tomando en cuenta las características del medio usado.

[6] Estándar: Que sirve como tipo, modelo, norma, patrón o referencia por ser corriente, de serie

2.1.4 PHP

PHP es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico, aunque ha evolucionado aumentando sus posibilidades, siendo capaz de hacer casi cualquier cosa.

2.1.5 JavaScript

JavaScript es un lenguaje de programación interpretado principalmente usado en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

- AngularJS

  AngularJS es un framework de JavaScript de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Facilita la creación de sitios web. Posee etiquetas personalizadas que realizan funciones sobre los elementos que las contienen dependiendo de los valores de sus atributos.

  Para ver más ejemplo y explicaciones más detalladas Ollivier, S., & Pierre-Alexandre, G. U. R. Y. (2016).

- D3

  D3 es una librería de JavaScript para producir, a partir de datos, infogramas dinámicos e interactivos en navegadores web. Hace uso de tecnologías bien sustentadas como SVG[7], HTML5, y CSS. Permite tener control completo sobre el resultado visual final dando fluidez e interacción a la web.

2.1.6 R

R es un entorno y lenguaje de programación con un enfoque al análisis estadístico.

Se trata de uno de los lenguajes más utilizados en investigación por la comunidad estadística. A esto contribuye la posibilidad de cargar diferentes bibliotecas o paquetes con funcionalidades de cálculo y gráficas.

2.1.7 JSON

JSON es un formato de texto ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript, aunque hoy, debido a su amplia adopción como alternativa a XML[8], se considera un formato de lenguaje independiente.

[7] SVG: formato de gráficos vectoriales bidimensionales
[8] XML: lenguaje para el intercambio de información estructurada entre diferentes plataformas

2.1.8 C#

C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET. C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común.

2.1.9 SQL

SQL es un lenguaje específico del dominio utilizado en programación; y diseñado para administrar sistemas de gestión de bases de datos relacionales.

Una de sus principales características es el manejo del álgebra y el cálculo relacional para efectuar consultas con el fin de recuperar, de forma sencilla, información de bases de datos, así como realizar cambios en ellas.

El alcance de SQL incluye la inserción de datos, consultas, actualizaciones y borrado, la creación y modificación de esquemas y el control de acceso a los datos.

2.2 Requisitos previos

Previamente a desarrollar nuestro sitio web necesitaremos las herramientas necesarias para nuestros objetivos.

2.2.1 Servidor HTTP, interprete PHP y base de datos

Ya que se trata de un sitio web, necesitaremos un servidor web HTTP, un intérprete de PHP y una base de datos que nos permitan crear nuestro sitio. Cada uno de estos componentes necesarios se pueden instalar de forma individual o podemos utilizar algún paquete que los contenga a todos para facilitar la instalación.

Ya que el sistema operativo que vamos a utilizar será Windows 10, Xampp es nuestra opción.

- XAMPP

XAMPP es una distribución de Apache[9] completamente gratuita y fácil de instalar que contiene MariaDB, PHP y Perl. El paquete de instalación de XAMPP ha sido diseñado para ser increíblemente fácil de instalar y usar.

Con su mera descarga e instalación, tendríamos disponible un sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl.

Su interfaz intuitiva nos permitirá iniciar cada uno de los componentes que necesitemos con un simple clic.

[9] Apache: es un servidor web HTTP de código abierto

Ilustración 7: Logotipo Xampp y sus componentes

2.2.2 Editor de texto

Para empezar a programar, necesitaremos un editor de texto que nos ayude con la programación y diseño web. Existen gran cantidad de editores de texto que ofrecen funcionalidades que facilitan la programación al usuario final como Visual Code o Notepad++. Sin embargo, Brackets, ofrece herramientas visuales en el editor que lo distinguen:

- Editores en línea

  En lugar de pasar de una pestaña a otra, Brackets te permite editar el CSS asociado a un elemento en un documento HTML directamente con una combinación de teclas Ctrl+E encima del mismo.

- Vista previa en vivo

  Conecta en tiempo real con nuestro navegador. Al realizar cambios en CSS y HTML instantáneamente se pueden ver esos cambios en la pantalla. También se puede observar dónde se está aplicando el selector de CSS en el navegador simplemente colocando el cursor sobre él. Es la combinación de un editor de código con la comodidad de las herramientas de desarrollo del navegador.

- Extensiones

  Brackets permite añadir extensiones que aumentan sus funcionalidades como atajos de teclado y funcionalidades JavaScript entre otras.

2.2.3 Entornos de desarrollo integrado (IDE)

Ya que se van a desarrollar aplicaciones web que ampliarán la funcionalidad del sitio web, necesitaremos los entornos de desarrollo integrado que nos proporcionen servicios integrales asociados a las funcionalidades de dichas aplicaciones.

- RStudio

RStudio es un entorno de desarrollo integrado (IDE) para R donde crearemos nuestra aplicación de análisis estadísticos de datos. Ofrece un gran potencial para el manejo masivo de datos y permite crear aplicaciones web con gran facilidad.

Ilustración 8: Logotipo de R y RStudio

- Visual Studio Enterprise

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE) para sistemas operativos Windows. Soporta múltiples lenguajes de programación, como C#, al igual que entornos de desarrollo web, como ASP.NET, a lo cual hay que sumarle las nuevas capacidades online bajo Windows Azure.

Ilustración 9: Logotipo Visual Studio Enterprise 2017

Como guía orientativa podemos ver Guérin, B. A. (2016).

2.2.4 Navegador

Utilizaremos Google Chrome como navegador, gracias a sus herramientas de desarrollado, como el inspector, facilitan el trabajo al programador web.

Ilustración 10: Logotipo Google Chrome

Capítulo 3 Desarrollo del proyecto

Desarrollo del proyecto

Una vez está preparado nuestro entorno de trabajo ya podemos ponernos a desarrollar el proyecto.

Como comentamos en el capítulo 1, estará divido en una serie de fases que englobaran partes del proyecto relacionadas entre sí.

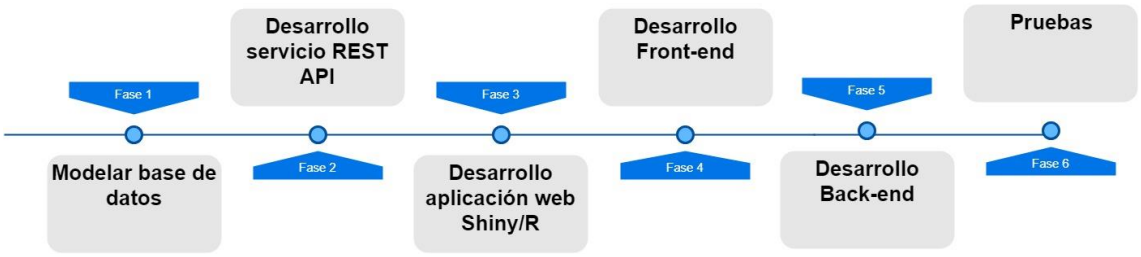

Ilustración 11: Diagrama de las fases del proyecto

3.1 Modelar la base de datos

Mencionado anteriormente, la base de datos tiene un diseño destinado a los usos propuestos por los desarrolladores del juego, donde se necesita saber los usuarios que existen y sus partidas ganadas.

Por ello, el diseño incluye dos tablas:

- **Users**: la cual contiene la información del usuario. Los campos disponibles son la id, única para cada usuario, email, también único para cada usuario, name, para el nombre del usuario, password, para la contraseña, el campo active que determina si una cuenta esta verificada o no, y image, que contiene la dirección de la imagen de usuario.
- **Partidas**: que albergará la información recopilada de las partidas del juego. Los campos que contiene esta tabla son la id, única para cada partida, id_user, id del usuario que realizó esa partida, score, puntuación, fecha, de cuando se realizó la partida y personaje con el que se jugó.

3.1.1 Modelo entidad-relación

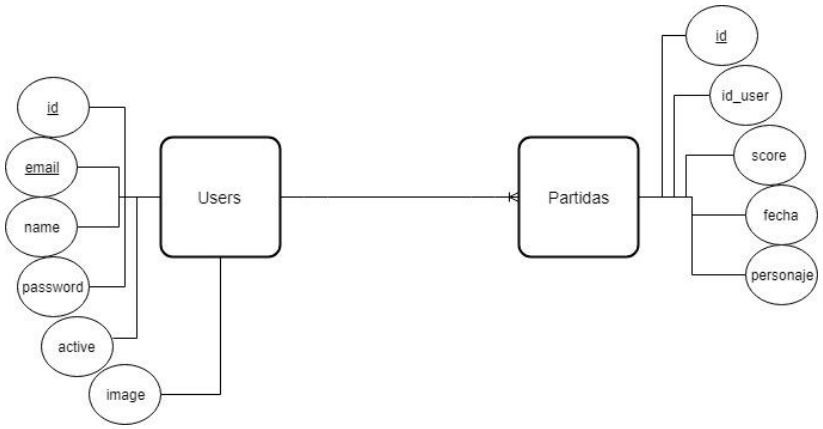

Ilustración 12: Modelo entidad relación de la base de datos

3.1.2 Tablas

Columna	Id	Name	Password	Email	Image	Active
Tipo	Int(255)	Varchar(255)	Varchar(255)	Varchar(255)	Varchar(255)	Enum{activada,desactivada}
Ejemplo	1	Manuel	99ae3c28363efaa7ad7453398ad3e2c4	manu@manu.es	Images/1.jpg	activada

Tabla 1: Estructura tabla Users de la base de datos

Columna	Id	Id_user	Score	Personaje	Fecha
Tipo	Int	1nt	Int	Varchar	Date
Ejemplo	1	1	20	Skeleton	2018-07-01

Tabla 2: Estructura tabla Partidas de la base de datos

3.1.3 Consultas

Las consultas que utilizaremos en nuestro sitio web:

INSERT INTO user(name, password,email,activada,url) VALUES (?name,?password,?email,?activada,?url)
SELECT id,activada FROM user WHERE password=@param_password AND (name=@param_name OR email=@param_email)

SELECT * FROM user WHERE id=@param_id
SELECT * FROM partidas WHERE id_user=@param_iduser order by fecha asc
UPDATE user SET activada=@activada WHERE email =@email
UPDATE user SET name=@name,password=@password WHERE id =@id
UPDATE user SET image=@image WHERE id =@id

Tabla 3: Consultas base de datos

3.2 Desarrollo servicio web API REST

En la actualidad no existe proyecto o aplicación que no disponga de una API REST para la creación de servicios profesionales a partir de ese software. Buscando una definición sencilla, REST es cualquier interfaz entre sistemas que use HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles, como XML y JSON.

Para nuestro sitio web vamos a crear un servicio REST encargado de manipular los datos para proporcionar seguridad y control sobre estos.

Trabajaremos en Visual Studio 2017 Enterprise, el cual nos permite crear una aplicación web API REST de manera sencilla. Este recibirá peticiones HTTP, realizará consultas a la base de datos y responderá la información correspondiente.

3.2.1 Clases

- **ValuesController**

  Esta clase será la que reciba las peticiones HTTP. Las peticiones que se recibirán serán del método POST y el formato será JSON. La estructura del mensaje es siempre la misma, cambiando los valores de los campos y su importancia. La clase se encargará de determinar de qué tipo de servicio se trata y actuar en consecuencia. Una vez determinado el servicio, se cogen el resto de los parámetros necesarios del objeto JSON, se crea el objeto Usuario con las propiedades y funciones necesarias según el servicio elegido para generar una respuesta adecuada. Los servicios disponibles se verán más adelante.

- **MySqlCustomConnection**

  Es la clase encargada de establecer, abrir y cerrar la conexión con la base de datos. Dispone de tres métodos que permitirán a la clase Usuario establecer conexiones con la base de datos cuando lo requiera:

  -Initialize(): Creara la conexión con los valores propuestos hacia la base de datos.

  -OpenConnection(): Abrirá la conexión con la base de datos

  -CloseConnection(): Cerrará la conexión con la base de datos

- **Usuario**

  Clase que modela nuestro servicio con la base de datos, contiene una serie de propiedades o campos asociados con los datos necesarios para realizar peticiones. Extiende de MySqlCustomConnection, por lo que puede utilizar sus métodos y establecer una conexión con la base de datos para realizar sus peticiones. Según los resultados de esas peticiones, creará un mensaje de vuelta al sitio web en formato JSON que permitirá realizar unas funciones u otras.

- **SmtpCustom**

  Esta clase se utiliza para enviar un correo SMTP con un enlace donde el usuario que se ha registrado deberá hacer clic para cambiar el estado de su cuenta a activada, verificando así su identidad. Usuario hará uso del objeto SmtpCustom cuando se trate del servicio INSERT para confirmar que el correo que se ha introducido es válido. Mientras está permanezca desactivada el usuario, aunque este registrado no podrá acceder al sitio web.

Tanto la programación de la clase Usuario como la de SmtpCustom siguen la corriente Fluent API, tratándose está de una forma de programación orientada a objetos en la que se busca la sencillez y fluidez a la hora de programar. De manera sencilla, se basa en devolver el mismo objeto continuamente, aplicándole los cambios de cada método.

3.2.2 Servicios

Los servicios o peticiones disponibles, que la aplicación es capaz de atender son los siguientes:

Mensaje	Servicio
INSERT	Añadir usuario
FIND	Buscar usuario
PROFILE	Buscar datos de usuario
ESTADISTICAS	Seleccionar partidas de usuario
UPDATE_PROFILE	Actualizar datos de usuario

UPDATE_IMAGE	Actualizar imagen de usuario
ACTIVATE	Activar cuenta

Tabla 4: Servicios disponibles en la aplicación web API REST

3.2.3 Mensajes

Los servicios se solicitan mediante el intercambio de mensajes entre el sitio web, donde se encuentra el usuario, y el servidor REST a través de las peticiones HTTP.

La estructura del mensaje para los servicios es siempre la misma, aunque varía el contenido de los campos. Mencionado anteriormente, el formato del mensaje, tanto de la solicitud como de la respuesta es JSON. La respuesta puede variar en función de los resultados conseguidos:

Estructura mensaje sitio web	**Estructura respuesta REST**	
----------	Correcto	Incorrecto

{
"type": "INSERT",
"id": "",
"name": "name",
"password": "password",
"email": "email",
"image": ""
}
	{
	"STATUS": "200",
	"CONTENT":
	"Something"
	}
		{
		"STATUS": "400",
		"CONTENT":
		“Something”
		}

Tabla 5: Estructura de los mensajes

3.2.4 Publicación

A partir de ahora, ya disponemos de un servicio REST que nos respalde frente al servidor. Visual Studio ofrece la posibilidad de subirlo a sus servidores de Azure permitiéndote acceder desde la nube a tu servicio REST.

3.3 Desarrollo aplicación web Shiny/R

En esta parte, vamos a desarrollar una aplicación web encargada de analizar las partidas ganadas de cada personaje por usuario en función del tiempo, la cual estará presente en nuestro sitio web. Para ello haremos uso de Shiny, un paquete de R para crear fácilmente aplicaciones web interactivas que permiten al usuario utilizar sus datos sin cambiar código.

Nuestra aplicación Shiny está conformada por un archivo app.R, el cual contiene tanto los elementos de la interfaz(UI) como del servidor(SERVER).

3.3.1 Paquetes

Además del paquete Shiny utilizaremos los siguientes paquetes:

Paquete	**Funcionalidades**
Jsonlite	El paquete ofrece herramientas flexibles, robustas y de alto rendimiento para trabajar con JSON en R y es particularmente potente

	para construir tuberías e interactuar con una API web.
Dygraphs	El paquete de los dygraphs es una interfaz R para la biblioteca de gráficos de JavaScript de los dygraphs. Proporciona servicios completos para representar datos de series de tiempo en R.
Lubridate	Manipular fechas e intervalos de tiempo
Xts	Proporcionar un manejo uniforme de las diferentes clases de datos basados en el tiempo de R al extender zoo, maximizando la preservación de la información de formato nativo y permitiendo la personalización del nivel de usuario y extensión, al tiempo que simplifica la interoperabilidad entre clases.

Tabla 6: Paquetes utilizados en RStudio

3.3.2 UI

La interfaz o UI, es donde se deben incluir los demás scripts que vayamos a necesitar para que nuestra aplicación web funcione correctamente, y los elementos que queramos que aparezcan en nuestra aplicación.

En este caso, como necesitamos recuperar los datos enviados desde nuestro sitio web utilizaremos un script JS que se encargue de detectar la llegada de los datos y guardarlos en una variable que Shiny recuperara a través de un elemento reactivo.

Incluso podemos mejorar el diseño de nuestra aplicación incluyendo una hoja de estilos CSS. Ya que queremos mostrar esos datos que hemos recuperado en una gráfica del paquete dygraphs, debemos añadir un elemento de salida de la clase dygraph.

3.3.3 SERVER

Una vez recibidos los datos, el servidor de nuestra aplicación web de Shiny se encargará de manipular los datos para poder utilizarlos en una gráfica del paquete dygraphs.

Observaremos detalladamente la manipulación de los datos

Desde nuestro sitio web los datos se envían en un objeto JSON llamado “partidas” que contiene el recuento de partidas ganadas de cada personaje por cada día del año por un usuario.

siguiendo esta estructura:

```
{
  "partidas": [
    {
```

```
    "Brutus": 0,
    "Skeleton": 10,
    "Razinguer": 0,
    "Amporio": 0,
    "Goliad": 0,
    "fecha": "2018-7-1"
  },
  ...
 ]
}
```

Tabla 7: Estructura objeto JSON "partidas"

Una vez el servidor de la aplicación ha detectado un cambio en los datos, recupera el objeto JSON "partidas" del elemento reactivo que hemos declarado utilizando el paquete jsonlite y se crea el dataframe[10].

	Brutus	Skeleton	fecha
1	11	10	2014-7-1
2	0	5	2015-7-2
3	15	0	2016-7-5
4	20	0	2017-7-6
5	130	0	2018-7-12
6	100	0	2018-7-13

Ilustración 13: Estructura del dataframe "partidas"

```
partidas          6 obs. of 3 variables
  Brutus : int 11 0 15 20 130 100
  Skeleton: int 10 5 0 0 0 0
  fecha : chr "2014-7-1" "2015-7-2" "2016-7-5" "2017-7-6" ...
```

Ilustración 14: Formato y contenido de los elementos del dataframe

Como podemos observar, el campo fecha está en formato de carácter y necesitamos que este en formato date por lo que necesitamos del paquete lubridate para poder realizar este cambio.

[10] Dataframe: son una estructura de datos en R que generaliza a las matrices

```
partidas          6 obs. of 3 variables
  Brutus : int 11 0 15 20 130 100
  Skeleton: int 10 5 0 0 0 0
  fecha : Date, format: "2014-07-01" "2015-07-02" "2016-07-05…
```

Ilustración 15: Formato y contenido de los elementos del dataframe tras modificar la columna fecha con el paquete lubridate

Ahora, para poder introducir un conjunto de datos validos a la gráfica producida por dygraph debemos sacar y eliminar del conjunto de datos partidas, la columna correspondiente a la fecha, y transformarlo a un objeto xts, ya que para representarlo con dygraphs es necesario este formato.

	Brutus	Skeleton
2014-07-01	11	10
2015-07-02	0	5
2016-07-05	15	0
2017-07-06	20	0
2018-07-12	130	0
2018-07-13	100	0

Ilustración 16: Estructura del objeto XTS

```
partidas          An 'xts' object on 2014-07-01/2018-07-13 …
  Data: int [1:6, 1:2] 11 0 15 20 130 100 10 5 0 0 ...
  attr(*, "dimnames")=List of 2
  ..$ : NULL
  ..$ : chr [1:2] "Brutus" "Skeleton"
  Indexed by objects of class: [Date] TZ: UTC
  xts Attributes:
  ULL
```

Ilustración 17: Formato y contenido de los elementos del objeto XTS

Los datos ya están listos para poder ser utilizados por dygraph

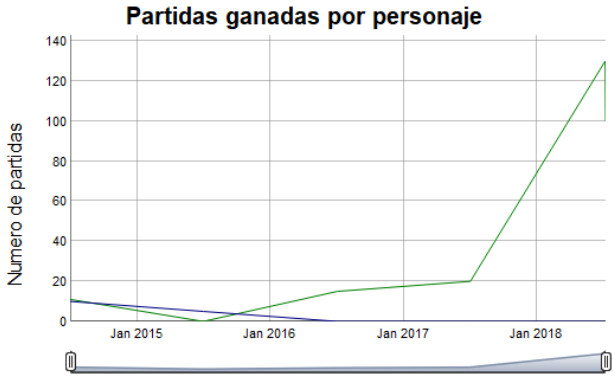

Ilustración 18: Grafica dygraph resultante de nuestros datos

Aunque no se puede apreciar en la imagen, la librería permite interactuar con la gráfica de la siguiente forma:

- Aumentando o disminuyendo el rango de datos que se muestran en la gráfica respecto al tiempo.
- Mostrando una leyenda con los datos en la marca temporal sobre la que se sitúa el cursor.

3.3.4 Publicación

Ahora ya disponemos de una aplicación web que permite el análisis y representación de estadísticas del juego. Sin embargo, todavía nos falta un último paso para poder utilizarla en nuestro sitio web. Debemos publicarla en un servidor que permita ejecutar aplicaciones Shiny/R[11].

Una vez publicada en un servidor ya está disponible para su uso.

3.4 Desarrollo Front-end

Dada la diversidad de tecnologías que existen para realizar la interfaz de usuario, decidimos modelar el Front-end de nuestro sitio web con diferentes tecnologías para más tarde analizarlas y sacar nuestras conclusiones.

[11] De forma gratuita podemos publicar nuestras aplicaciones web en https://www.shinyapps.io/

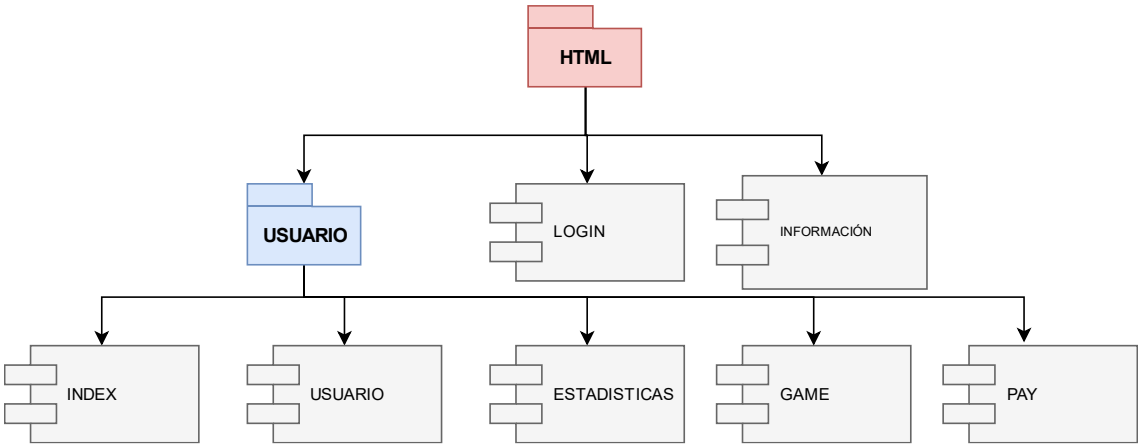

Ilustración 19: Estructura carpeta HTML

En la ilustración podemos ver de manera visual como están organizados los documentos HTML. Estos documentos ampliarán sus funcionalidades apoyándose en los scripts JS que se pueden ver a continuación:

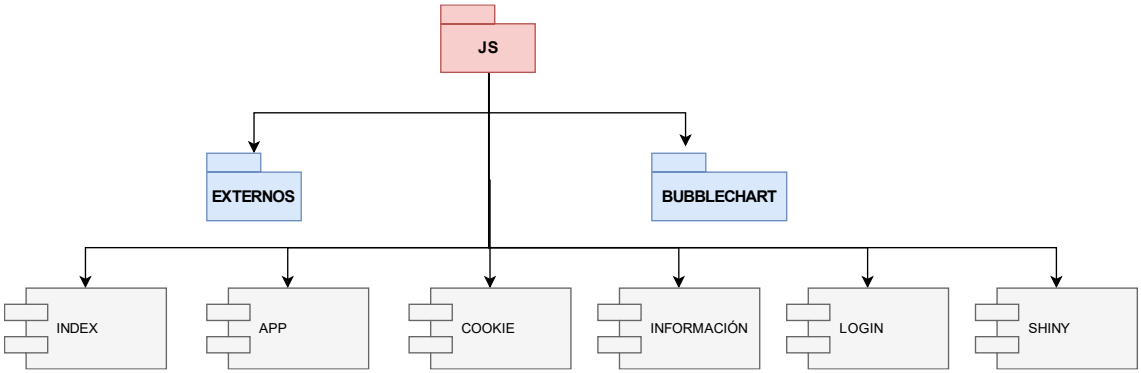

Ilustración 20: Estructura carpeta JS

Y en los scripts de JS que encuentran dentro de la carpeta Bubble-chart dentro de la carpeta JS. Allí encontraremos dentro del directorio test, los archivos test_bubble.js, encargado de la animación D3 con los datos disponibles en el archivo datos_heroes.json.

Las partes en las que se pueden dividir son las correspondientes a las páginas que el usuario podrá visualizar. A continuación, se explicarán con detalle cómo están organizadas, que tecnologías utilizan y que funcionalidad desempeñan las siguientes páginas o agrupaciones de páginas:

3.4.1 Index.html

La función que desempeña esta página en el sitio web es meramente informativa para nuevos jugadores. Dado que es la página principal de nuestro sitio, se buscará que sea llamativa para conseguir la atención del usuario. Está formada exclusivamente por el archivo index.html y se complementa con los scripts CSS generalStyle.css e indexStyle.css para los estilos, y JS, index.js y cookie.js.

La página está programada en HTML5, apoyándose en la librería de D3 "Bubble-chart" y Bootstrap para su diseño.

- **Header y Aside**

En el header podemos encontrar un video de fondo en el que los personajes del juego nos saludan. Utilizando una animación de CSS hacemos aparecer tras unos segundos el logo del juego con un botón que nos permitirá acceder a la página de registro/inicio de sesión y continuar con la experiencia del sitio web.

En el aside, tendremos los iconos de las redes sociales que el juego utiliza para distribuir información sobre el juego y expandirse. Permanecerá estático en la página principal.

Ilustración 21: Header y aside de la página principal del sitio web

- **Body**

La primera sección que encontramos después del header, nos proporciona información sobre los modos de juego y nos presenta el juego mediante un video promocional.

Ilustración 22: Sección 1 del Body de la página principal

En la siguiente sección de la página principal podemos ver las características de los personajes del juego.

Ilustración 23: Sección 2 del Body de la página principal

Realizada con una animación D3 Bubblechart a la que le hemos añadido dos plugins, uno para mostrar el nombre y otro para situar en el centro el nodo seleccionado, y JavaScript, el usuario al hacer clic encima de algún personaje se producirá una redistribución de los nodos que componen la animación. Además, aparecerá un cuadro de características donde se mostrarán las aptitudes y los ataques del personaje permitiendo incluso una visualización de previa de estos. La aparición del cuadro de características implica una animación en CSS en la que se produce un "shake" del cuadro.

La animación D3 obtiene los datos de un fichero JSON con un formato especifico y diseñado para este uso. El fichero JSON se lee al completar la carga del body de la página principal.

El archivo JSON con la información, está formado por un array de objetos JSON que contienen los datos de cada uno de los personajes. De esta manera, cada vez que se hace clic en un nodo, se recupera el objeto JSON con los datos del personaje correspondiente y mediante JavaScript se actualiza el valor de los campos del cuadro de características.

- **Footer**

En el footer encontramos iconos e imágenes relacionadas con el ámbito del videojuego proporcionando información adicional, enlaces a la página de políticas del sitio web y un botón que nos invita a entrar al sitio web.

Ilustración 24: Footer página principal

3.4.2 Informacion.html

La página web correspondiente al apartado de políticas, está compuesta por único archivo, información.html, el cual modifica el contenido al hacer clic encima de los elementos A.

Ilustración 25: Página de políticas, cookies y términos de uso

Al hacer clic, el cuadro de información aparece con una animación de CSS y simultáneamente se hace una llamada a una función JavaScript, donde se realizan cambios en los elementos HTML del DOM[12] para mostrar la información deseada. La función del botón close, es cerrar el cuadro de información y el elemento A Back, es el encargado de volver a la página principal.

Las funciones JavaScript que se utilizan se encuentra en el script información.js.

3.4.3 Login.html

En esta página web, con el diseño y comportamiento semejante, encontramos los paneles de inicio de sesión y de registro. Compuesta por un solo archivo, identificación.html, juega con JavaScript con el script login.js y modifica sus estilos con la hoja de CSS login_informacion_usuario.css, para mostrar un contenido u otro, y ocultar el contrario.

Además, cuenta con las nuevas posibilidades que ofrece HTML5 para poder establecer tipos de input, patrones a seguir y formatos, agilizando el proceso de comprobación de que los datos están correctamente.

- **Identificación**

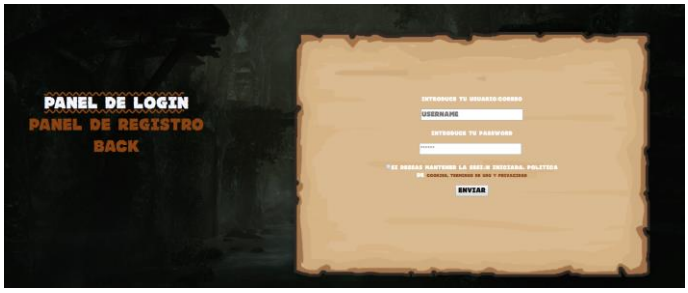

Ilustración 26: Panel de login

[12] DOM: es esencialmente una interfaz de plataforma que proporciona un conjunto estándar de objetos para representar documentos HTML, XHTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.

En el panel de login o inicio de sesión, el usuario podrá poner su nombre o correo y su contraseña para poder acceder a la web. Podrá decidir si mantener la sesión iniciada o no, dependiendo de su elección se creara una cookie con más o menos tiempo de duración. Esta cookie será comprobada cada vez que el usuario acceda al panel de login por una función JavaScript, y en el caso de estar creada accederá directamente a su página de usuario.

Tras introducir los datos y pulsar enviar, se enviará una petición HTTP del tipo POST mediante AJAX[13]. Dependiendo del resultado, la página nos redireccionara a nuestra página de usuario si todo ha ido bien o nos solicitara que revisemos los datos o activemos cuenta mediante un alert de JavaScript.

- **Registro**

Ilustración 27: Panel de registro

En el panel de registro, un usuario podrá registrarse en nuestro sitio web y a la vez en el juego, permitiéndole así poder acceder al contenido disponible y jugar tras su descarga e instalación.

El usuario no verá el botón enviar a menos que seleccione el checkbox que indique que acepta las condiciones del sitio web.

En el caso de que las acepte y lo seleccione, se llamará a una función JavaScript que hará aparecer el botón enviar. Cuando el usuario lo pulse, otra función JavaScript comprobara que el contenido de las contraseñas es el mismo para evitar errores de seguridad.

Si todo es correcto debería llegarnos un correo de los autores del juego con un enlace que active nuestra cuenta y verificando así nuestra identidad.

La función del elemento A Back es la de volver a la página principal.

[13] AJAX: es una técnica de desarrollo web para crear aplicaciones interactivas

3.4.4 Carpeta usuario

Una vez el usuario se ha registrado correctamente, ha activado su cuenta y ha iniciado sesión, accederá al corazón de nuestro sitio web. Una página web diseñada a partir de AngularJs, formada por los archivos.html (de la carpeta html/usuario/somename.html) y los archivos app.js y usuario_style.css, basando su diseño en una sola página en la que se cambian las vistas proporcionando al usuario otras funcionalidades.

El usuario dispondrá de un selector formado por elementos A, al igual que anteriormente, pero esta vez, se encargará de manera dinámica a través de AngularJs de cambiar el contenido.

Las diferentes vistas disponibles son las siguientes:

- **Usuario.html**

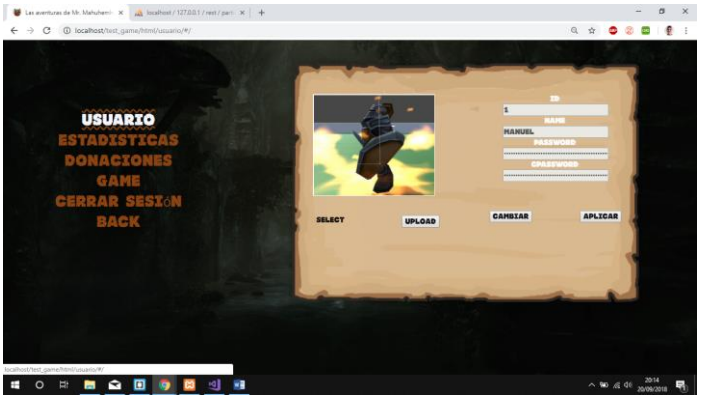

Ilustración 28: Vista Usuario

En esta vista el usuario dispondrá de las funcionalidades propias de un perfil de usuario, donde podrá cambiar su imagen, nombre o contraseña.

Los cambios no se podrán efectuar a menos que se pulse el botón modificar, que cambiara el estado de los elementos HTML a través de la etiqueta ng-disabled de AngularJS.

Una vez modificados, el usuario aplicara los cambios realizados pulsando el botón aceptar. Las comunicaciones con el servidor se realizan utilizando los métodos AJAX que proporciona AngularJS.

- **Estadísticas.html**

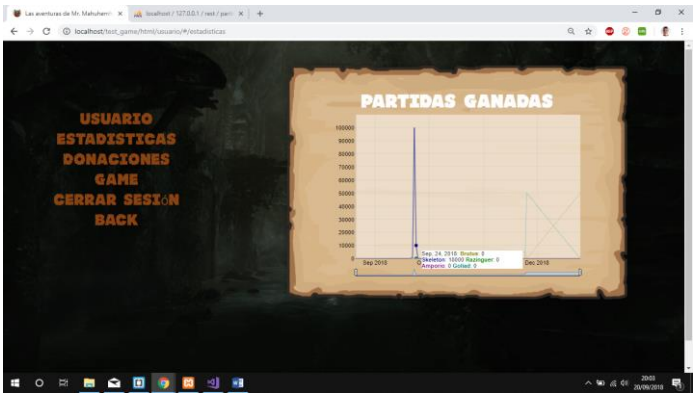

Ilustración 29: Vista estadísticas

La vista Estadísticas ofrece una gráfica proporcionada por la aplicación web Shiny creada anteriormente. Para poder introducir esta aplicación en nuestro sitio web haremos uso del elemento IFRAME[14], haciendo referencia al enlace donde este publicada. El script JS que acompaña a este documento HTML es el script shiny.js que obtendrá los datos de la aplicación web API REST para enviárselos al IFRAME que contiene la aplicación web Shiny para que está trate los datos y los represente.

- **Pay.html**

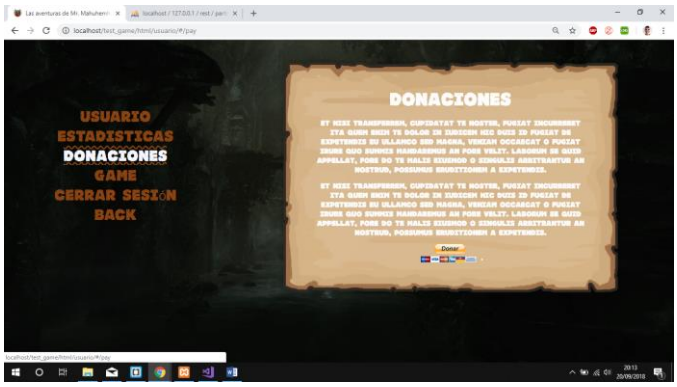

Ilustración 30: Vista donaciones

En la vista donaciones, encontraremos un botón que redireccionara a una cuenta PayPal donde poder realizar donaciones para que los desarrolladores del juego puedan seguir ampliando sus fronteras.

- **Game.html**

[14] IFRAME: es un elemento HTML que permite insertar o incrustar un documento HTML dentro de un documento HTML principal.

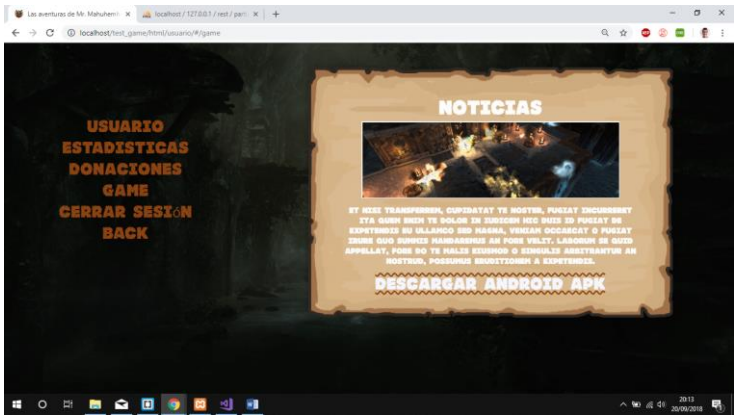

Ilustración 31: Vista game

La última vista disponible sería Game, donde se ofrecen noticias relacionadas con el juego y enlaces para las últimas versiones del juego en sus diferentes plataformas.

Las opciones del selector Cerrar sesión y Back, ofrecen la función de cerrar sesión como se puede intuir, eliminando las cookies correspondientes al usuario, y la función de ir a la página principal, respectivamente.

3.5 Desarrollo Back-end

En esta fase del proyecto, debemos aclarar que el servicio REST también debería ser considerado parte del Back-end, sin embargo, la complejidad de su desarrollado exigía una fase propia.

Nos centraremos en los scripts PHP que apoyan la comunicación con el servicio REST.

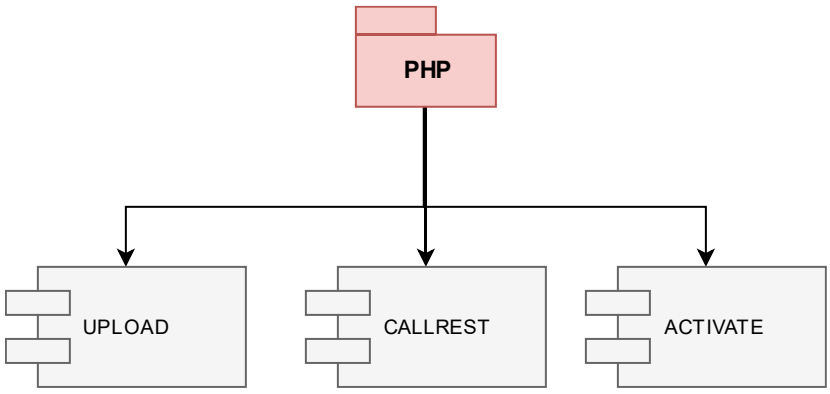

Ilustración 32: Estructura carpeta PHP

3.5.1 CallRest.php

El script callRest.php será el encargado de realizar las peticiones desde el servidor de nuestra página web a la aplicación web API REST creada anteriormente. Recogerá los datos que son enviados

desde las funcionalidades de las páginas del sitio web, les dará un formato y realizará las peticiones al servicio mediante la tecnología cURL[15].

Ya que la aplicación REST está preparada para recibir los datos en formato JSON, se deberá especificar en las cabeceras del mensaje y transformar los datos a formato JSON.

```
<?php

$json=json_decode(file_get_contents('php://input'), true);

$array =["type" => $json['type'],
        "id" => $json['id'],
        "name" => $json['name'],
        "password" => $json['password'],
        "email" => $json['email'],
        "image" => $json['image']
       ];


$ch = curl_init();

$headers = ["Content-Type: application/json"];

curl_setopt($ch, CURLOPT_URL,"http://localhost:55690/api/values");
curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS,json_encode($array));
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);

$server_output = curl_exec ($ch);

curl_close ($ch);

echo $server_output;

?>
```

Tabla 8: Script callRest.php

La configuración del mensaje debe también especificar que se utilizará el método POST y la URL[16] de nuestra aplicación API REST.

3.5.2 Upload.php

El archivo upload.php es el encargado de subir una imagen seleccionada por el usuario. La imagen será guardada con el nombre de fichero igual a la ID del usuario que la subió. Además, utilizar la

[15] cURL: software consistente en una biblioteca y un intérprete de comandos generalmente orientado a la transferencia de archivos y datos.
[16] URL: es la dirección específica que se asigna a cada uno de los recursos disponibles en la red con la finalidad de que estos puedan ser localizados o identificados.

función informaREST() que de igual forma que el script callRest.php realizara la petición para actualizar el valor de la columna image de la fila correspondiente al usuario en la tabla Users de la base de datos.

En cualquiera de los casos, se hará una redirección a la página del usuario.

```
$target_dir = "../media/upload/";
$id = str_replace("","",$_COOKIE["id_user"]);


$uploadOk = 1;
$imageFileType =
strtolower(pathinfo(basename($_FILES["fileToUpload"]["name"]),PATHINFO_EXTENSION
));
$target_file = $target_dir.$id.".".$imageFileType ;

    if (move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], $target_file)) {

        informaRest($id.".".$imageFileType);
        header ("Location: ../html/usuario");


    } else {

    header ("Location: ../html/usuario");

    }
```

Tabla 9: Script upload.php

3.5.3 Activate.php

Este script se llama desde el email que es enviado al usuario para confirmar su dirección de correo electrónico. Al hacer clic, el usuario está realizando una llamada GET en este script de PHP, donde se recupera el correo del usuario y como se hacía en los dos scripts anteriores se envía una petición al servicio REST para cambiar el estado de la cuenta a activada.

3.6 Pruebas

Una vez llegados a este punto se procederá a realizar un recorrido a lo largo de nuestro sitio web para comprobar que el funcionamiento es el esperado ante las distintas funcionalidades que ofrece. Se han realizado pruebas sobre todas las funcionalidades, sin embargo, a continuación, solo se mostrarán las más importantes.

Antes de empezar con el recorrido se muestra el siguiente diagrama que explica la comunicación entre nuestros distintos proyectos para cubrir esas funcionalidades viendo como desde la web se realizan peticiones al servicio REST o a la aplicación Shiny, conectando con la base de datos y otros elementos.

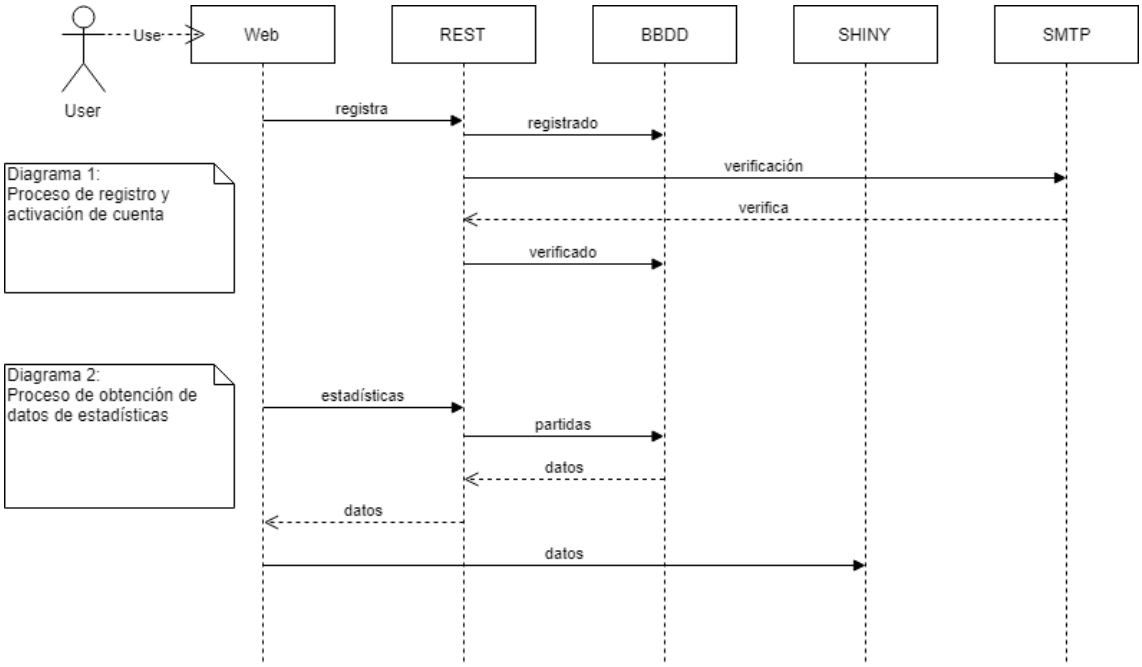

Ilustración 33: Diagrama de intercambio de mensajes entre el sitio web y las diferentes aplicaciones

El usuario accederá a la web desde nuestra página principal donde encontrará la animación D3. Antes de realizar ningún clic sobre la animación, el panel de información de personaje no será visible y en su lugar se verá el logotipo del juego.

Ilustración 34: Animación D3 1

Cuando el usuario haga clic encima en el nodo central, se mostrará el panel de información de personaje con sus características principales, una breve explicación de su historia en el juego y los ataques. Al hacer clic sobre alguna de las imágenes que corresponde a los ataques, en el apartado VIEW del panel de información aparecerá un video demostrando como es el ataque.

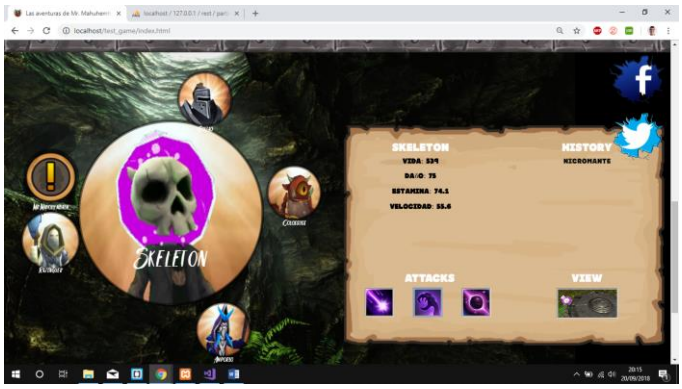

Ilustración 35: Animación D3 2

Si el usuario hace clic en cualquier nodo que no sea el central, el nodo seleccionado se posicionará en el centro y el resto se distribuirá a su alrededor de forma aleatoria. Además, el panel de información se actualizará con la información correspondiente al personaje del nodo seleccionado acompañado de una animación CSS en la que el panel se agita.

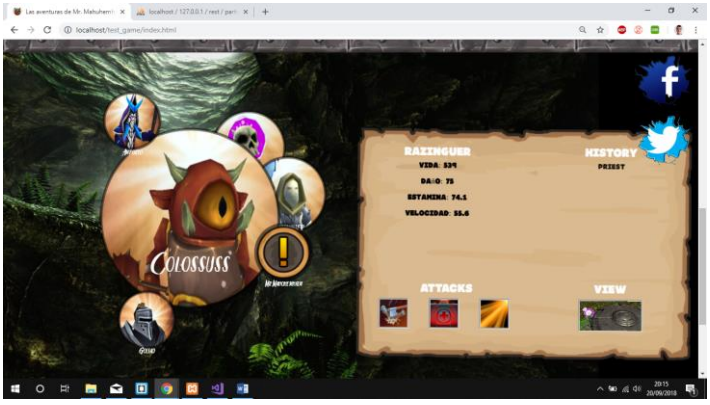

Ilustración 36: Animación D3 3

Al hacer clic en cualquiera de los botones **PLAY** de la página principal el usuario accederá a la página que contiene los paneles de login y registro.

En el panel de login, el usuario deberá introducir su nombre de usuario o su correo electrónico, y la contraseña asociada.

Si la contraseña no tiene el formato correcto o si alguno de los campos no está completo saldrá un panel de información advirtiéndonos de ello.

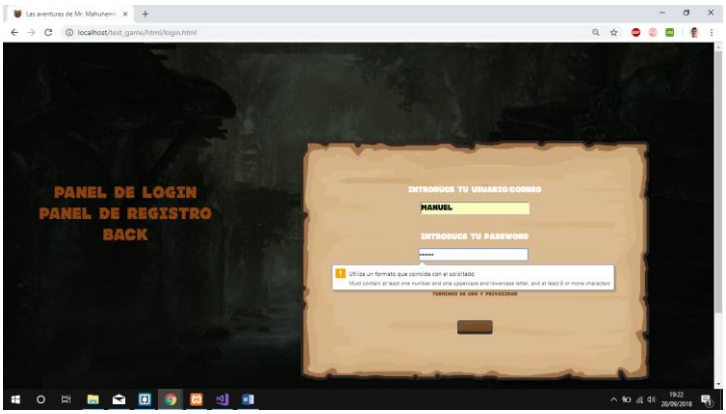

Ilustración 37: Panel de login con dialogo advirtiendo error en el formato de la contraseña

Suponiendo que los campos se han rellanado correctamente, el usuario puede no haber activado su cuenta por lo que aparecería un ALERT de JavaScript informándonos de ello.

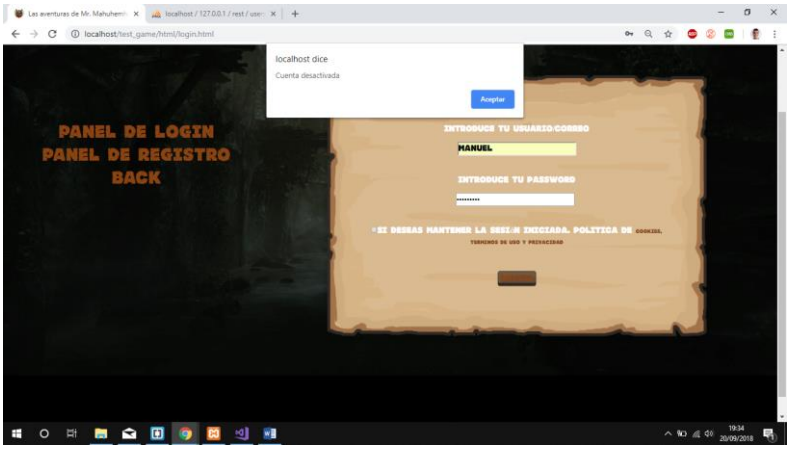

Ilustración 38: Panel de login con ALERT informando de cuenta desactivada

El usuario puede no haberse registrado, por lo que accederá al panel de registro. Deberá rellenar correctamente cada uno de los campos que aparecen además de aceptar las condiciones para que aparezca el botón Enviar.

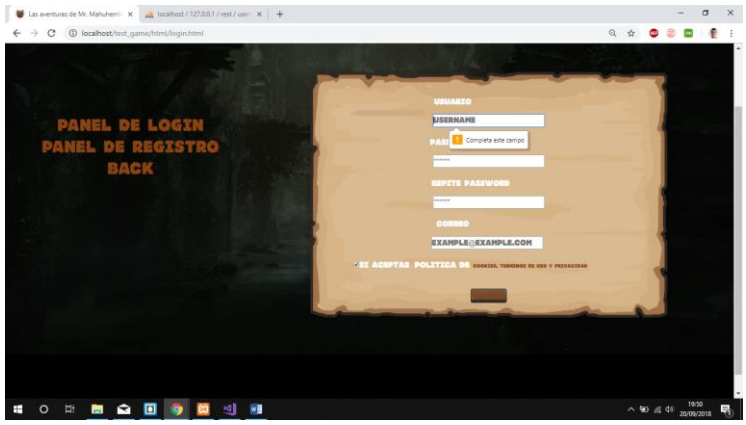

Ilustración 39: Panel de registro con campo sin completar

Una vez rellenados los campos, el usuario aceptará las condiciones del sitio haciendo clic en checkbox disponible en el formulario y como consecuencia aparecerá el botón que se encargará de enviar los datos.

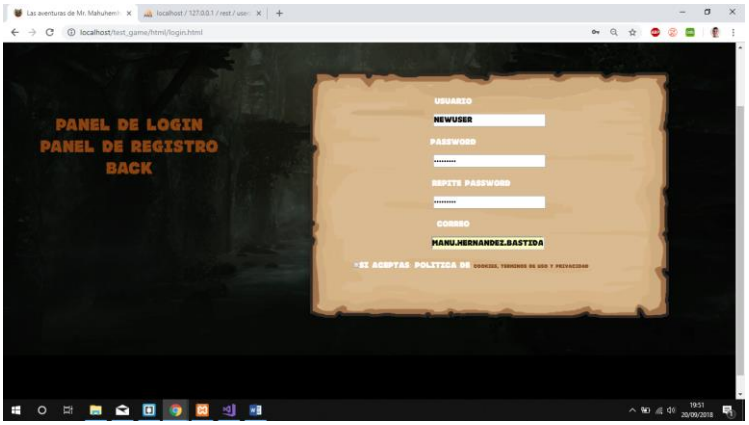

Ilustración 40: Panel de registro sin aceptar condiciones

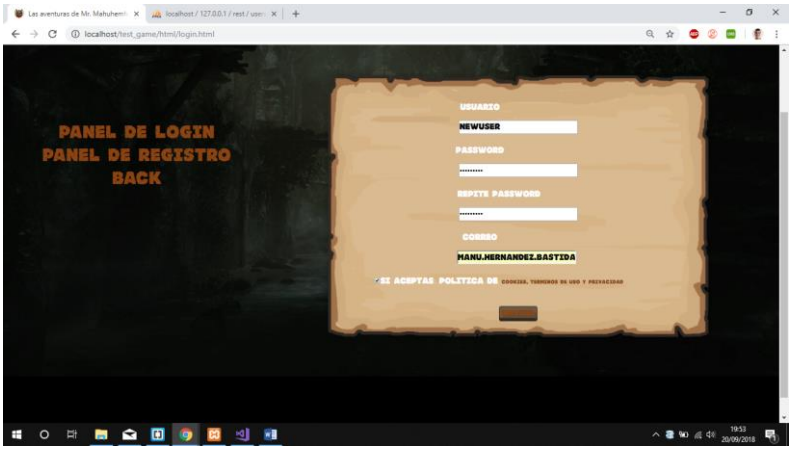

Ilustración 41: Panel de registro aceptando condiciones

En el caso de que las contraseñas sean diferentes un ALERT informará de ello.

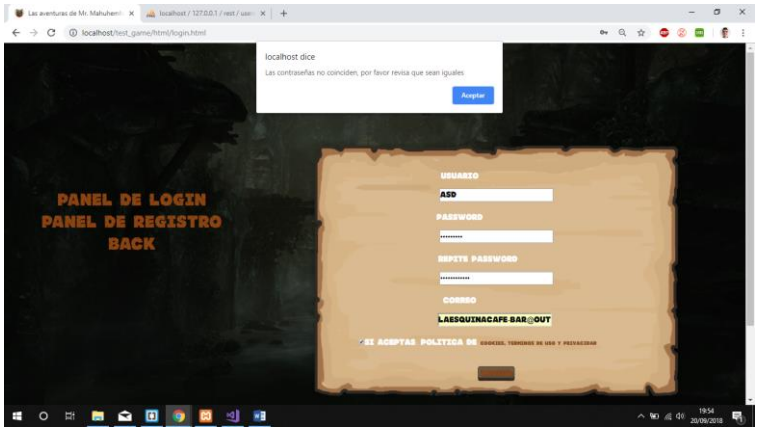

Ilustración 42: Panel de registro con ALERT de contraseñas diferentes

En el caso de que el email introducido exista ya en la base de datos, aparecerá un ALERT informando al usuario que su email esta repetido y no podrá registrarse con él.

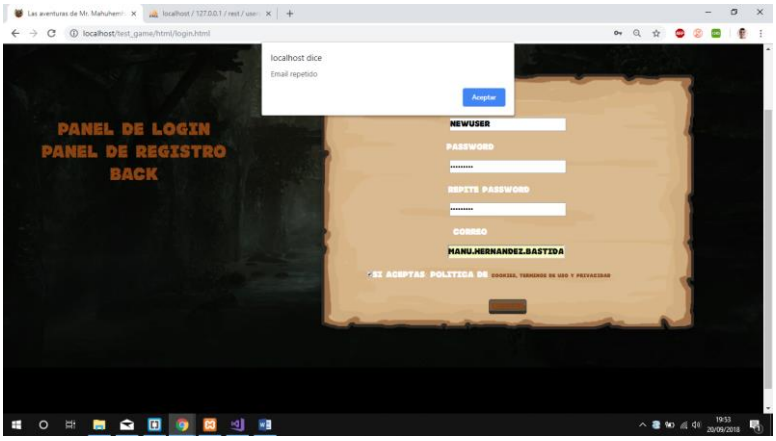

Ilustración 43: Panel de registro con ALERT de email repetido

Si todo va correctamente, aparecerá otro ALERT informando al usuario y le llegará un correo que le permitirá activar su cuenta y acceder desde el panel de login.

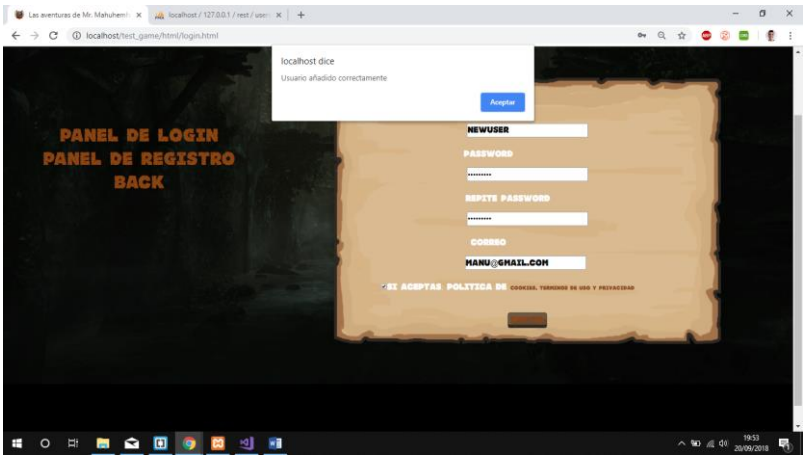

Ilustración 44: Panel de registro con ALERT de usuario registrado

Suponiendo que el usuario ya está registrado accederá desde el panel de login a su panel de usuario. Aquí podrá modificar su perfil de usuario. Para cambiar su imagen de perfil hará clic en el botón Select con el que aparecerá una ventana de selección de archivos.

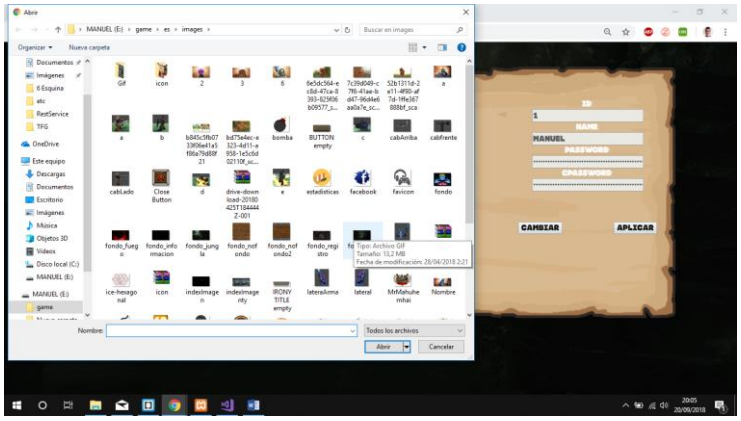

Ilustración 45: Panel de usuario con ventana de selección de archivos

Una vez seleccionada la imagen, está se subirá al servidor y se actualizará en el perfil haciendo clic en el botón upload.

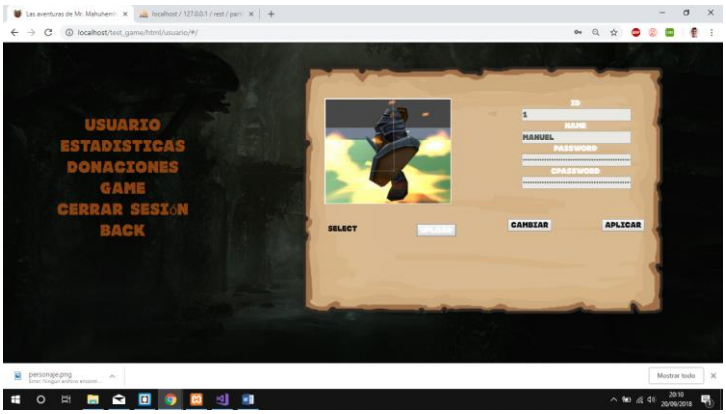

Ilustración 46: Panel de usuario haciendo clic en upload

También podrá actualizar su nombre de usuario y sus contraseñas activando la modificación pulsando el botón cambiar y tras el cambio actualizar el perfil haciendo clic en el botón aplicar.

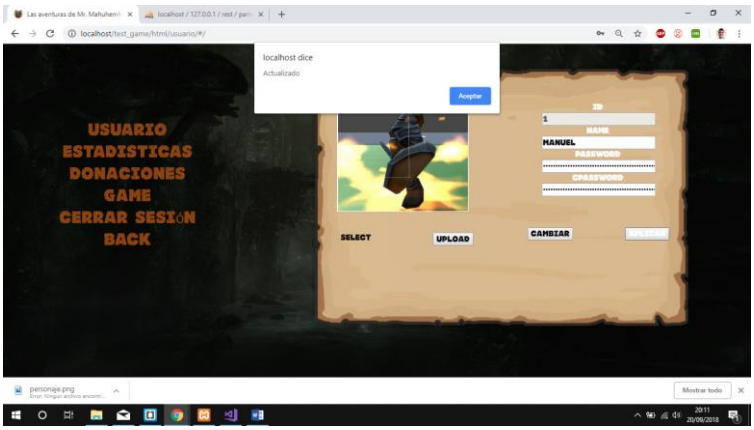

Ilustración 47: Panel de usuario con ALERT de perfil actualizado

Si es su primer contacto con el juego, su panel de estadísticas estará vacío y no se verá nada.

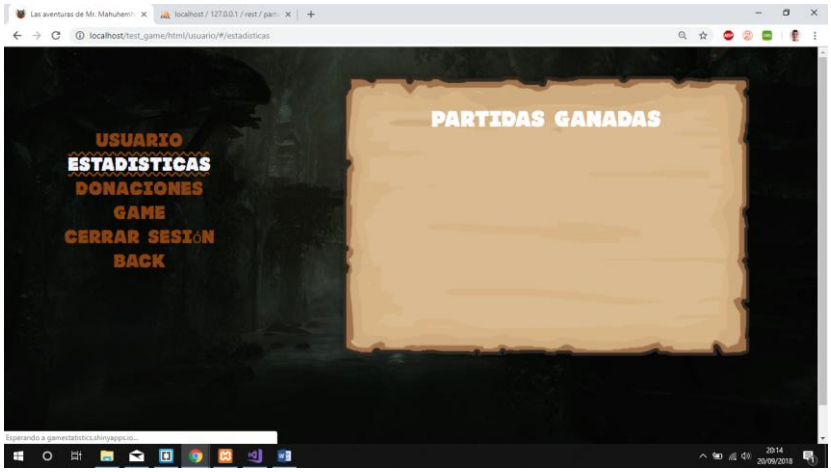

Ilustración 48: Panel de estadísticas vacío

Si por el contrario ya es jugador podrá comprobar sus progresos a través de una gráfica.

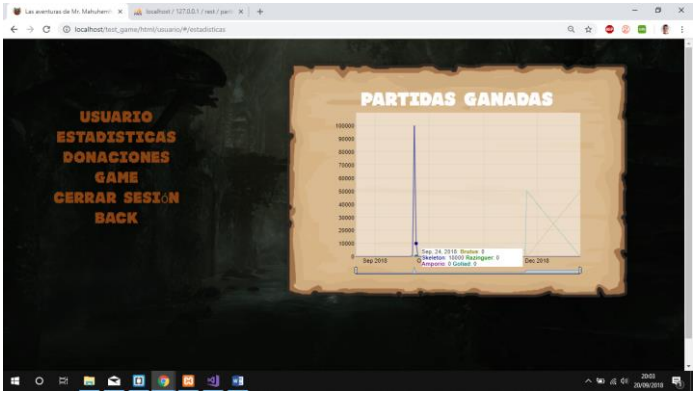

Ilustración 49: Panel de estadísticas con grafica

Si todavía no ha jugado, el usuario tendrá disponible en el panel de game la información necesaria para poder jugar y el instalador de juego con su última versión. Podrá descargarlo haciendo clic en el elemento A que está situado en el panel.

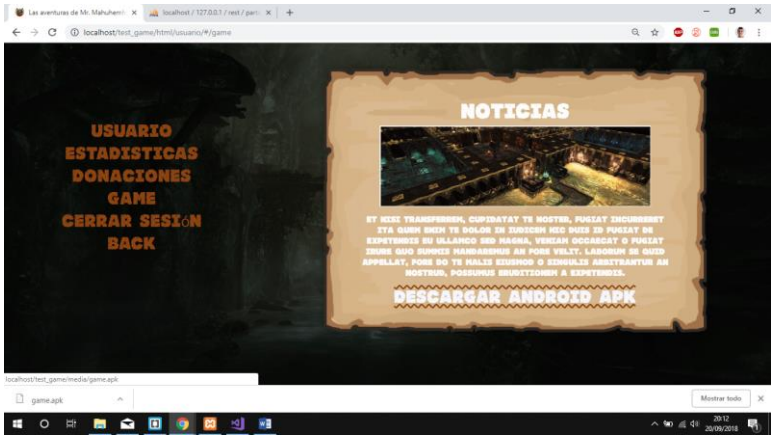

Ilustración 50: Panel de game descargando apk del juego

Capítulo 4 Conclusiones y líneas futuras

Conclusiones y líneas futuras

Una vez desarrollado el proyecto, llega la hora de sacar nuestras conclusiones y estudiar las posibles líneas de expansión para un futuro.

Tras un largo periodo de trabajo estamos preparados para analizar las tecnologías utilizadas y el resultado obtenido, permitiéndonos abrir nuestro proyecto hacia nuevas oportunidades de expansión.

4.1 Conclusiones

Durante el desarrollo del proyecto nos hemos enfrentado a una serie de dificultades que nos han hecho elegir entre un tipo de tecnología u otra para solventarlas. En el caso de la web nos encontrábamos ante la posibilidad de utilizar HTML, CSS y JS clásico, donde la evolución de este ha permitido implementar en nuestro sitio web grandes funcionalidades de manera sencilla. Sin embargo, sus funcionalidades están limitadas por su estado del arte, dejando un lugar durante el desarrollo a la utilización de los frameworks elegidos entre todos los disponibles, BootStrap y AngularJS, los cuales nos facilitan un diseño responsivo y fluido, donde la actualización de los contenidos se puede hacer de manera dinámica y transparente al usuario. Podemos decir, que para un diseño completo de la web es necesaria una combinación de todas las tecnologías que permita adaptarse a los objetivos que se buscan.

Siguiendo el desarrollo de la aplicación web Shiny, nos damos cuenta de lo importante que puede ser el adecuado tratamiento de los datos y su correspondiente aplicación para los servicios que se requieran. Para ofrecer el mejor servicio posible se ha elegido R como lenguaje de análisis de datos estadísticos, ya que no solo es el lenguaje más utilizado, sino que además contamos con RStudio que ha facilitado el desarrollo de la aplicación ya que cuenta con librerías y paquetes que se acercan al diseño web permitiendo su integración de manera intuitiva.

En el desarrollo de la aplicación web API REST, vemos como Visual Studio 2017 ofrece facilidades a la hora crear un proyecto de esas características y como a través de este podemos ampliar las funcionalidades, permitiendo así un tratamiento de los datos basado de un modelo objetos relacionado con la base de datos.

Con una vista general del proyecto, podemos observar que se trata de un proyecto donde la complejidad no surge en los subproyectos que lo componen sino en la vinculación de estos para dar lugar al proyecto final, con la finalidad de cumplir los objetivos propuestos. Se destaca la utilización del formato de objetos JSON para la comunicación que se establece entre las diferentes aplicaciones y el sitio web, y como estás utilizan este formato para manipular los datos.

El resultado obtenido es un proyecto sencillo pero completo, con bases que ofrecen un gran abanico de posibles extensiones en los diferentes caminos que está tomando el desarrollo web y que permitirá en un futuro abrir las puertas a nuevas funcionalidades que ofrecer al usuario que las utilice.

4.2 Líneas futuras

El proyecto ofrece la base para dos caminos de desarrollo que merecen destacar su importancia: Big data y Cloud-computing[17].

Por un lado, la utilización de R y RStudio en el proyecto es de manera sencilla y didáctica sin embargo la potencia que tienen el lenguaje y el entorno, permitirían el desarrollo de aplicaciones web más complejas cuyas finalidades sirvan al desarrollador del juego como fuente de información para la mejora del videojuego con vistas al usuario y con vistas a las oportunidades de éxito y económicas que puedan surgir, sirviéndose como fuente de datos el juego y los usuarios que lo utilizan.

Desde el plano del sitio web, se podría incluir funcionalidades y vistas que solo puedan ver los administradores del juego con graficas que muestren los usuarios que se unen a la comunidad, que se descargan el juego, el número de partidas que se juegan, posibles ingresos entre otras muchas características que se podrían analizar.

Tanto la aplicación web Shiny como la aplicación web API REST, encuentran su hueco en la computación en la nube permitiéndoles una escalabilidad de sus servicios y una ampliación de funciones.

Como se vio durante el desarrollo del proyecto, es necesario alojar en un servidor nuestra aplicación web Shiny. En este caso se trata de un servidor gratuito y limitado, pero podríamos optar por opción de pago donde aumentemos los recursos disponibles permitiendo el acceso a mas usuarios y mejorando el rendimiento del procesado de los datos.

Al igual que la aplicación web Shiny, la aplicación web API REST es un proyecto básico que puede servir de base para el desarrollo de un proyecto de mayor envergadura donde los servicios que se soliciten requieran de funciones más complejas, diferentes modelos de objetos y se sirvan de bases de datos mayores que permitan ampliar esas funcionalidades.

[17] Cloud-computing: es una tecnología que permite acceso remoto a softwares, almacenamiento de archivos y procesamiento de datos por medio de Internet, siendo así, una alternativa a la ejecución en una computadora personal o servidor local.

Aprovechando que utilizamos Visual Studio podríamos hacer uso de los servicios de pago ofrecidos por Azure, donde podríamos contratar servicios IaaS[18] y PaaS[19] para el alojamiento de nuestra base de datos permitiendo reducir o escalar verticalmente los recursos con rapidez para ajustarlos a la demanda e implementar nuestras aplicaciones, evitando el gasto y la complejidad que suponen tener servidores físicos propios.

18 IaaS: Infraestructura como servicio, infraestructura informática inmediata que se aprovisiona y administra a través de lnternet
19 PaaS: Plataforma como servicio, entorno de desarrollo de implementación completo en la nube

Capítulo 5 Anexos

Anexos

En este capítulo, veremos la instalación del software necesario, creación de proyectos, como añadir dependencias y extensiones de funcionalidad.

5.1 Instalación Xampp

La instalación de Xampp es bastante sencilla, únicamente deberemos descargar el ejecutable disponible en su página web y seguir los pasos que se muestran a continuación:

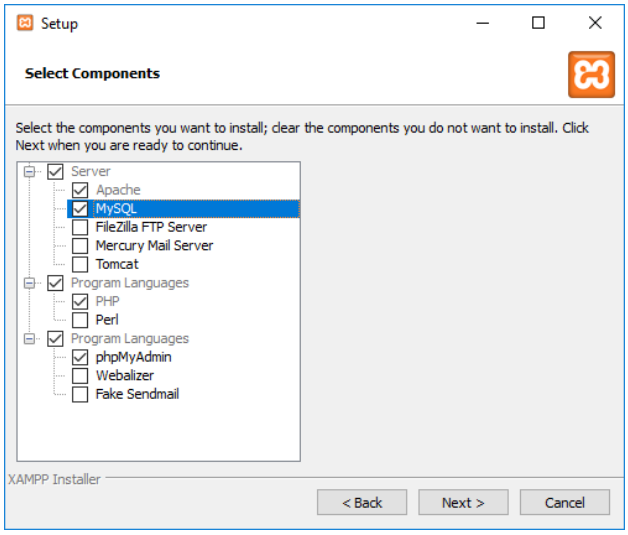

Ilustración 51: Panel de instalación Xampp 1

Seleccionar MySql y phpMyAdmin sino lo están por defecto.

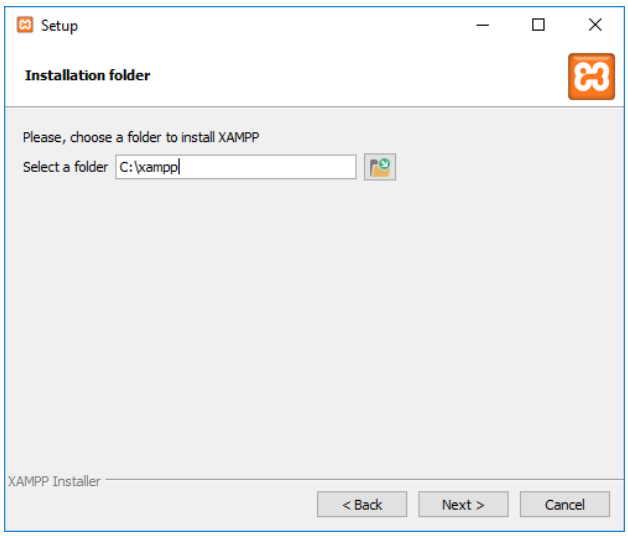

Ilustración 52: Panel de instalación Xampp 2

Elegir la carpeta de instalación e instalar

Para encender o apagar el servidor Apache y MySql basta con hacer clic encima del boton start/stop de la columna actions.

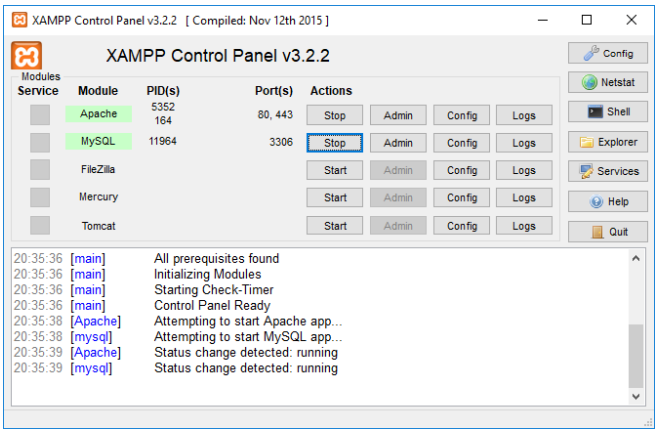

Ilustración 53: Panel de control Xampp

Con esto ya tendremos acceso a nuestro servidor web. Para facilitar nuestro trabajo, podremos crear un acceso directo a la carpeta htdocs disponible en la carpeta de instalación, elegida durante el proceso de instalación, en la cual se alojarán nuestras páginas web.

5.2 Instalación Visual Studio

Para la instalación de Visual Studio nos descargaremos el ejecutable disponible en la web vinculada con la universidad para poder utilizar la licencia de la versión Visual Studio Enterprise.

Al hacer doble clic sobre este, nos saldrá el wizard donde debemos seleccionar las cargas de trabajo o características de las que dispondrá nuestro Visual Studio. Disponemos de una gran variedad, pero elegimos las que nos interesan, ASP.NET y .NET, que nos permitirán crear de manera sencilla nuestro Web API REST. Una vez elegidas, clic Install.

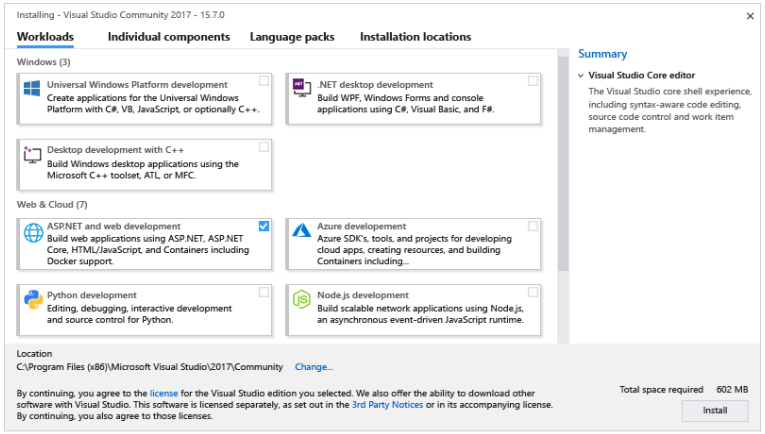

Ilustración 54: Panel de instalación de Visual Studio

5.3 Instalación RStudio

Para poder utilizar RStudio primero debemos instalar R en nuestro sistema. Para ello vamos a la página oficial de R y descargamos la versión de Windows, hacemos doble clic sobre el ejecutable y seguimos los pasos del wizard. La instalación de R no tiene complicación.

Una vez instalado R, podemos descargar el ejecutable de RStudio de Windows versión de escritorio gratuita. Los pasos que debemos seguir son los mismos que con R.

5.4 Proyecto Visual Studio Web API

Para crear un nuevo proyecto en Visual Studio haremos en clic en archivo -> nuevo -> proyecto y buscaremos entre las opciones disponibles la opción Web y elegiremos Aplicación web ASP.NET (.NET Framework) la cual nos ofrece soluciones sencillas para aplicaciones web

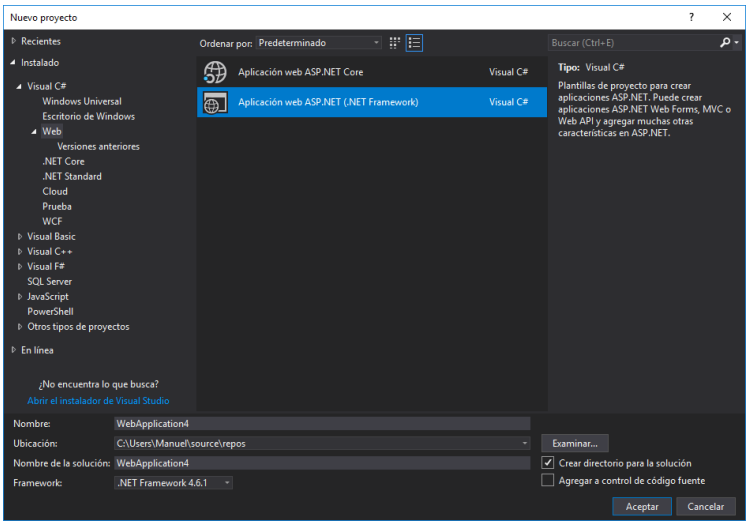

Ilustración 55: Panel de creación de proyecto de Visual Studio

Nuestro objetivo es crear un servicio API REST por lo que dentro de las soluciones que nos ofrece encontramos una que se llama Web API que satisface nuestras necesidades.

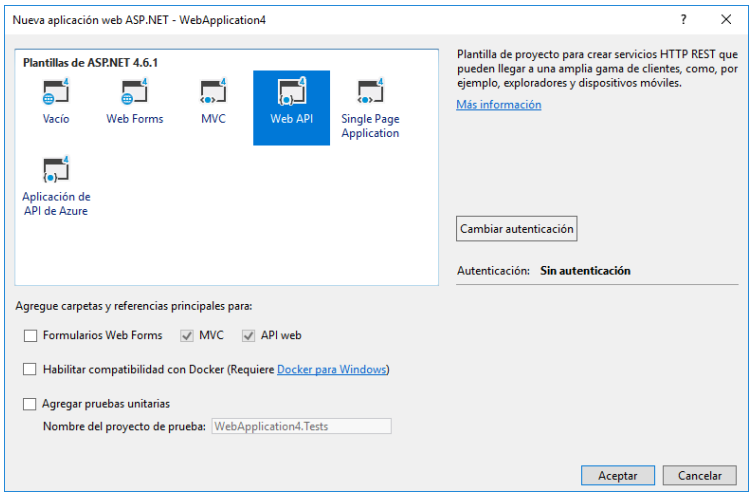

Ilustración 56: Panel de selección de plantilla de trabajo ASP.NET

Al hacer clic en aceptar, nos creara un proyecto con la siguiente estructura:

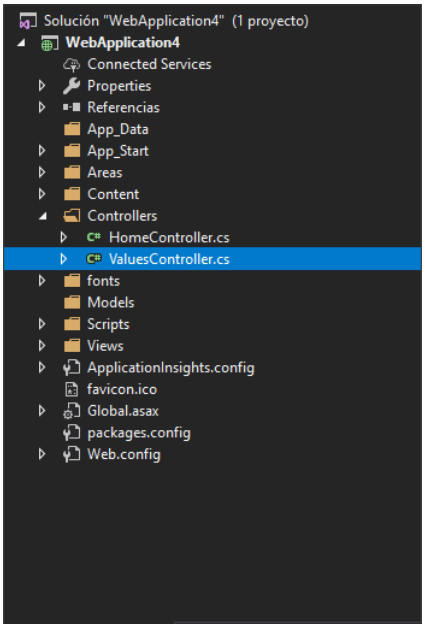

Ilustración 57: Estructura del proyecto en Visual Studio

Donde encontramos la clase ValuesController utilizada en nuestro proyecto.

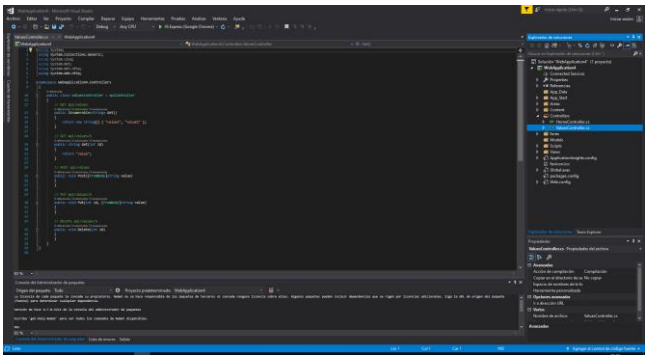

Ilustración 58: Clase ValueController.cs

5.5 Proyecto RStudio Shiny

Para crear un proyecto en RStudio, previamente crearemos un directorio donde se creará dicho proyecto.

Empezamos haciendo clic en File->New Project. Nos saldrá la siguiente ventana y elegiremos la opción existing directory.

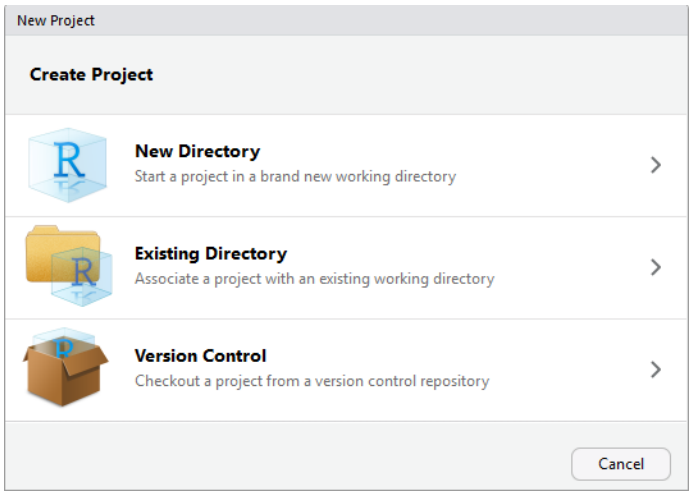

Ilustración 59: Panel de creación de proyecto en RStudio

Acto seguido elegimos la carpeta que hemos creado previamente para el proyecto y clic en créate project

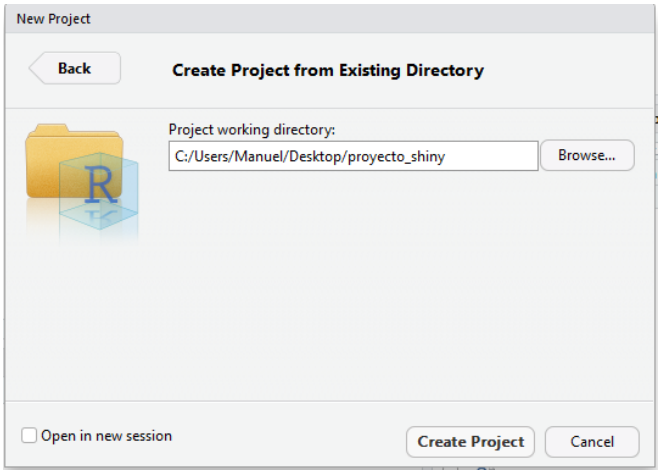

Ilustración 60: Panel de selección de directorio para el proyecto

Una vez creado el proyecto, crearemos nuestra aplicación Shiny haciendo clic en File-> New File-> Shiny Web App..

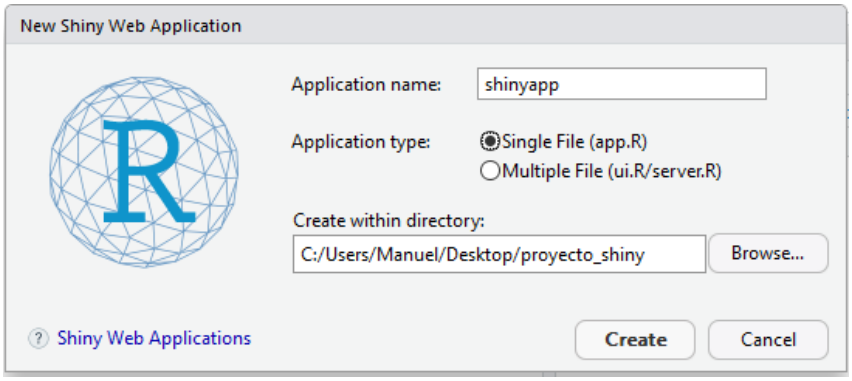

Ilustración 61: Panel de creación de aplicación web Shiny en RStudio

Seleccionamos la opción de single File y créate, creándonos una sencilla aplicación de ejemplo como se ve en la figura:

```
#
# Find out more about building applications with Shiny here:
#
#    http://shiny.rstudio.com/
#

library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

   # Application title
   titlePanel("Old Faithful Geyser Data"),

   # Sidebar with a slider input for number of bins
   sidebarLayout(
      sidebarPanel(
         sliderInput("bins",
                     "Number of bins:",
                     min = 1,
                     max = 50,
                     value = 30)
      ),

      # Show a plot of the generated distribution
      mainPanel(
         plotOutput("distPlot")
      )
   )
)

# Define server logic required to draw a histogram
server <- function(input, output) {

   output$distPlot <- renderPlot({
      # generate bins based on input$bins from ui.R
      x    <- faithful[, 2]
      bins <- seq(min(x), max(x), length.out = input$bins + 1)

      # draw the histogram with the specified number of bins
      hist(x, breaks = bins, col = 'darkgray', border = 'white')
   })
}

# Run the application
shinyApp(ui = ui, server = server)
```

Ilustración 62: Aplicación web Shiny por defecto

5.6 Instalación certificado SSL

Xampp está preparado para crear e instalar certificados en el servidor Apache. A continuación, vamos a crear uno que no será un certificado valido ya que no nos los acredita una entidad certificadora oficial pero que nos valdrá como ejemplo.

Nos situamos en la carpeta de apache en el directorio de instalación de Xampp y encontramos el script .bat makecert.

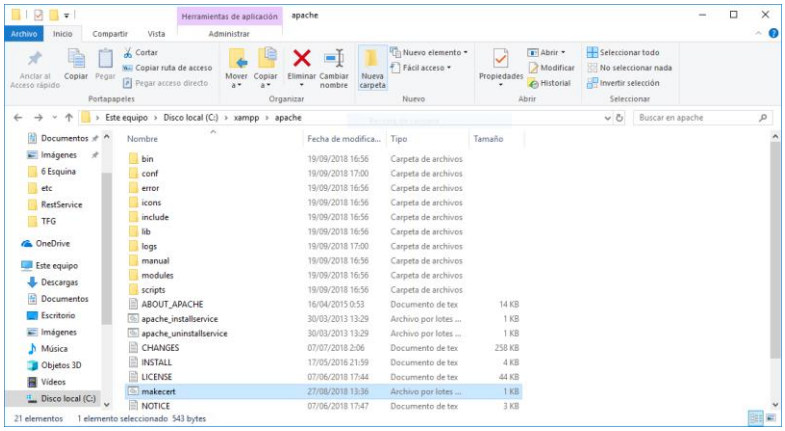

Ilustración 63: Directorio de apache con script makecert.bat

Al hacer clic, aparecer la consola con una serie de peticiones para que el usuario introduzca sus datos para crear el certificado asociado.

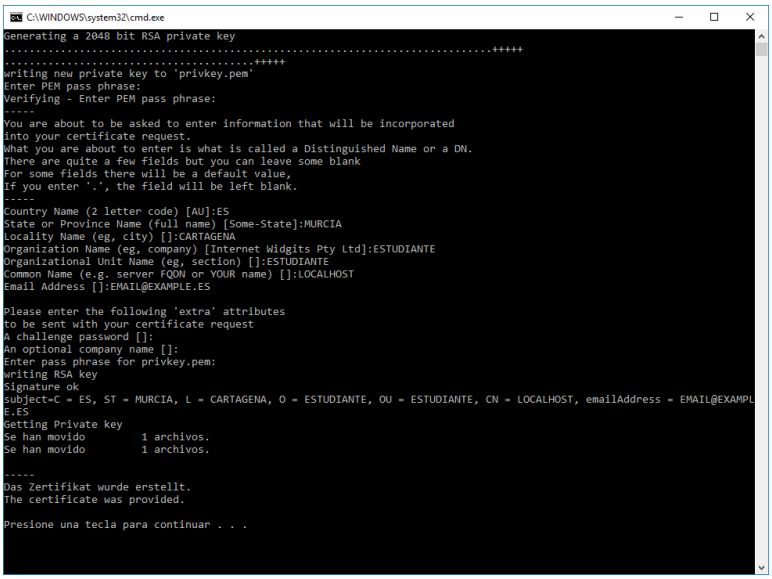

Ilustración 64: Consola de windows para creación de certificados

Una vez terminemos de completar la información requerida, se crearán en las carpetas conf → ssl.key y conf→ssl.crt nuestra key y nuestro certificado respectivamente.

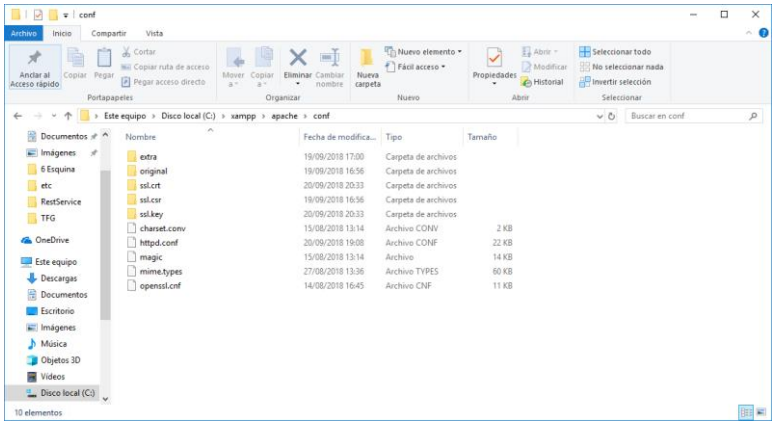

Ilustración 65: Directorios ssl.key y ssl.crt

Si queremos instalar el certificado solo tendremos que hacer clic en el archivo server.crt de la carpeta ssl.crt y seguir la guía de instalación.

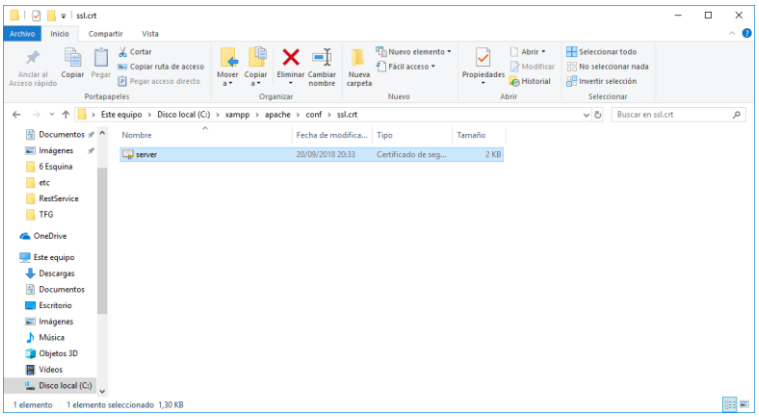

Ilustración 66: Archivo server.crt

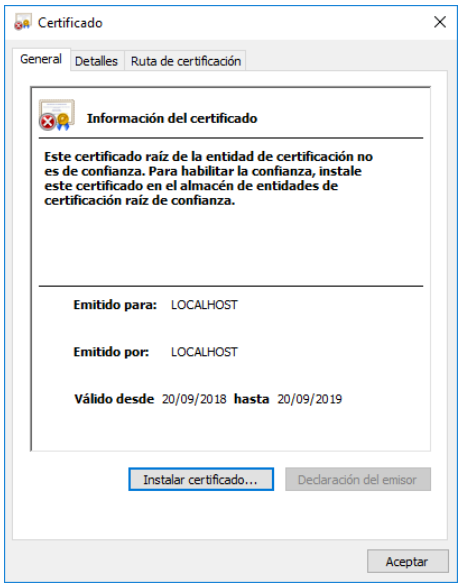

Ilustración 67: Panel de instalación de certificados

5.7 Añadir paquetes RStudio

Nos situamos en la ventana de abajo-derecha y seleccionamos la pestaña Packages.

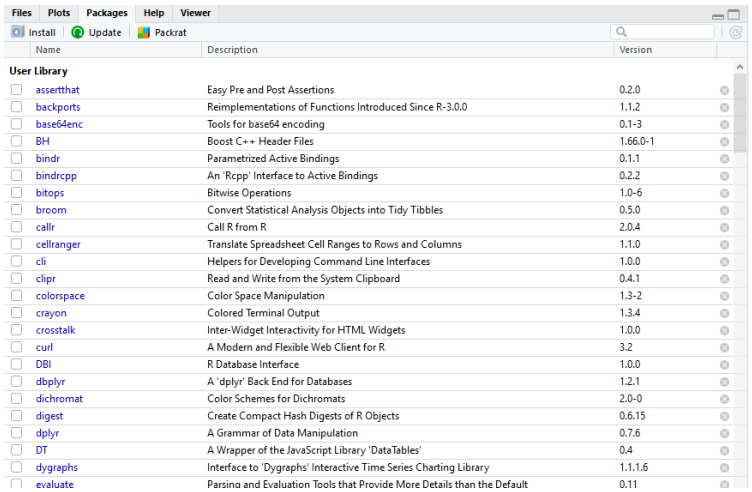

Ilustración 68: Panel de paquetes del proyecto en RStudio

En esta ventana podemos ver los paquetes disponibles en nuestro proyecto, si queremos añadir más, podemos hacer clic en Install y buscar el nombre de nuestro paquete. Podemos descargar mas de un paquete a la vez si ponemos los nombres separados por coma.

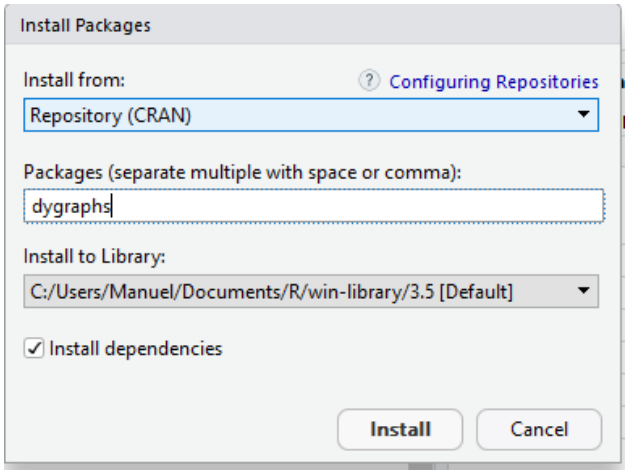

Ilustración 69: Panel de instalación de paquetes nuevos en el proyecto de RStudio

5.8 Añadir dependencias en Visual Studio

Para añadir una dependencia externa de Visual Studio, en nuestro proyecto la dependencia necesaria para utilizar la clase MySql, descargamos el paquete MySql Connector/NET de para .NET y mono.

Descomprimimos el paquete descargado, hacemos clic derecho encima de nuestro proyecto y seleccionamos Agregar -> Referencia-> Examinar. Buscamos la carpeta del archivo descomprimido y seleccionamos los archivos disponibles.

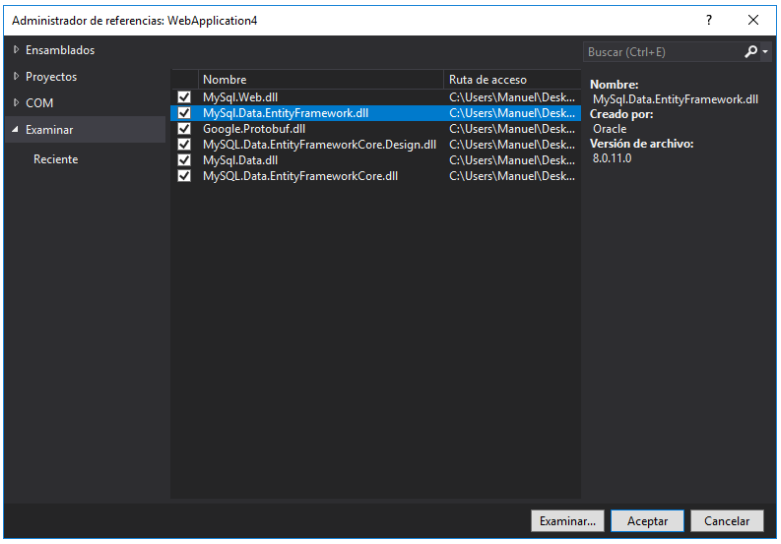

Ilustración 70: Panel de agregación de dependencias en Visual Studio

Una vez seleccionadas hacemos clic en aceptar y ya podemos utilizar nuestra librería.

Capítulo 6 Bibliografía

6.1 Bibliografía

- Gómez, M. R. (2013). HTML5, CSS3 y Javascript. Anaya Multimedia-Anaya Interactiva.
- Ollivier, S., & Pierre-Alexandre, G. U. R. Y. (2016). AngularJS: Desarrolle hoy las aplicaciones web de mañana. Ediciones ENI.
- Hernández, J. (2014). Análisis y desarrollo web. Jesús Hernández.
- Guérin, B. A. (2016). ASP. NET en C# con Visual Studio 2015: Diseño y desarrollo de aplicaciones Web. Ediciones ENI.
- Azaustre, C. (2014). Desarrollo web ágil con angularJS. Latinoamérica.
- W3Schools es un sitio web donde aprender, probar y entrenar ejemplos de programación web, «http://www.w3schools.com/»
- Easy web publishing from R, «https://rpubs.com//»
- Página oficial de Shiny donde encontramos todo lo necesario para desarrollar nuestra aplicación web Shiny/R, «https://shiny.rstudio.com/»
- Página oficial de Microsoft donde encontramos todo lo necesario para desarrollar nuestra solución API REST con ASP.NET, «https://docs.microsoft.com /»
- Página oficial PHP, «http://php.net//»
- Página oficial JavaScript, «https://www.javascript.com//»
- Página oficial D3, «https://d3js.org///»
- Página oficial AngularJS, «https://angularjs.org///»