UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS E INGENIERÍA

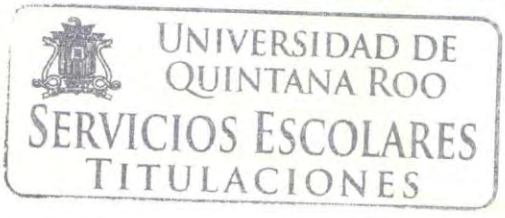
REVISIÓN AUTOMÁTICA DE TAREAS DE PROGRAMACIÓN

TRABAJO DE TESIS
PARA OBTENER EL GRADO DE
INGENIERO EN REDES

PRESENTA
JAFET OSORIO HAU

DIRECTOR DE TESIS
DR. JAIME SILVERIO ORTEGÓN AGUILAR

ASESORES
MTI. MELISSA BLANQUETO ESTRADA
MTI. VLADIMIR VENIAMIN CABAÑAS VICTORIA
LIC. SERGIO ALBERTO SOLIS SOSA
DR. JAVIER VÁZQUEZ CASTILLO



CHETUMAL QUINTANA ROO, MÉXICO, JULIO DE 2017

UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS E INGENIERÍA

TRABAJO DE TESIS ELABORADO BAJO SUPERVISIÓN DEL COMITÉ DE ASESORÍA Y APROBADO COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE:

INGENIERO EN REDES

COMITÉ DE TRABAJO DE TESIS

DIRECTOR:

DR. JAIME SILVERIO ORTEGÓN AGUILAR

ASESORA:

MTI.MELISSA BLANQUETO ESTRADA

ASESOR:

MTI. VLADIMIR VENIAMIN CABAÑAS VICTORIA

Universidad de Quintana Roo
DCI DIVISIÓN DE CIENCIAS E INGENIERÍA

UNIVERSIDAD DE QUINTANA ROO
SERVICIOS ESCOLARES
TITULACIONES

CHETUMAL QUINTANA ROO, MÉXICO, JULIO DE 2017

AGRADECIMIENTOS

A:

Mis profesores, que gracias a su guía he podido completar mi carrera como ingeniero de manera satisfactoria, no habría obtenido mejor formación que con ellos, ya que todos son profesionales muy capaces en sus ramas de estudio. También quiero agradecer a todos mis compañeros de clases, sin duda alguna hicieron mis días en la universidad más placenteros y divertidos, a todos aquellos con los que tuve la oportunidad de trabajar como equipo y que me ayudaron a seguir adelante en las materias.

También quiero agradecer al gobierno federal por brindarme el apoyo con una beca durante la mayoría de mi estancia en la Universidad, gracias a ese apoyo pude salir adelante con la mayoría de los gastos que requerí para mis clases y mi colegiatura.

Por último, me gustaría agradecer a mis compañeros de trabajo, que no me dieron la espalda cuando ingresé a la Universidad y me ayudaron a ajustar mis horarios a lo que yo necesitara para poder seguir asistiendo a mis clases y terminar la carrera de manera satisfactoria.

DEDICATORIA

A:

Mi madre, que siempre ha estado a mi lado durante todos mis años como estudiante, brindándome su apoyo y siempre alentándome a dar lo mejor de mí para salir adelante y cumplir con mis metas y aspiraciones en la vida, ella siempre ha celebrado mis triunfos y perdonado mis errores. Estoy muy agradecido con ella de que a pesar de ser madre soltera nunca se ha rendido para tratar de darme lo mejor, es por eso que le hago una dedicatoria especial a ella, ya que ella es el motivo de todos mis esfuerzos.

También la dedico este trabajo a todas aquellas personas que ha creído en mí y me han apoyado durante toda mi carrera, a mis hermanos y amigos que nunca me negaron su ayuda cuando la he necesitado y a todas las personas que han estado a mi lado todo este tiempo.

RESUMEN

A lo largo de este documento se explora el desarrollo y prueba de una aplicación web.

El primer capítulo se centra en presentar los objetivos, la justificación y el alcance del proyecto.

El segundo capítulo de este documento contiene toda la teoría sobre la que se basa el desarrollo de la aplicación, aquí se describen las distintas tecnologías que se utilizaron para completar el proyecto, así mismo se hace una breve descripción de los conceptos necesarios para la integración de dichas tecnologías y las librerías de desarrollo web utilizadas.

En el tercer capítulo se aborda a profundidad el desarrollo teórico de la aplicación web, se explica el uso que se le da a cada una de las tecnologías que la integran y los framework de desarrollo que se utilizaron, del mismo modo se describen los métodos y funciones utilizados.

El capítulo 4 está dedicado a las pruebas de funcionamiento de la aplicación, en este capítulo se explica la interfaz de usuario con todos sus componentes.

Por último en el capítulo 5 se encuentran las conclusiones a las que se llegó tras probar la aplicación.

Contenido

Capítulo 1

INTRODUCCIÓN

Para un ingeniero en las áreas de TI, identificar y aplicar el proceso de diseño, codificación, depuración y mantenimiento del código fuente de un programa (programación), es una de las bases fundamentales de su formación. La programación como ciencia brinda a los estudiantes la capacidad de tener enfoques distintos para resolver problemas, sin importar que rama de la ingeniería se encuentre estudiando, es por lo que la mayoría si no todas las carreras de ingeniería o de cualquier área informática incluyen en su plan de estudios una materia dedicada a la programación.

Una de las técnicas más recientes para la enseñanza de la programación, es el uso de plataformas web que permiten la compilación de código en línea, esto facilita a los estudiantes un compilador sin la necesidad de contar con un programa instalado en su computadora personal (pc) o incluso no contar con ésta, ya que el acceso a estas plataformas puede hacerse desde cualquier sitio con un dispositivo que cuenta con conexión a internet, como por ejemplo en un cibercafé. Estas plataformas especializadas son más comunes de encontrar en sitios web extraoficiales dedicados a dar cursos de programación.

La Universidad de Quintana Roo ofrece carreras de ingeniera como “ingeniería en redes” e “ingeniería en sistemas de energía” entre otras. Especialmente la carrera de ingeniera en redes se caracteriza por ofrecer dentro de su plan de estudios materias relacionadas con la programación, de ahí la idea de que los profesores cuenten con una plataforma que les permita brindar a los estudiantes los beneficios de poder compilar su código en línea para poder entregar trabajos o incluso presentar exámenes. A lo largo de este documento se explora el desarrollo y prueba de una herramienta que facilita la compilación de código de programación C++ en línea.

Justificación

El presente trabajo propone la implementación de una herramienta que permita evaluar los códigos de programación de los estudiantes de la Ingeniera en Redes de la Universidad de Quintana Roo.

El trabajo busca brindar una interfaz web para que los alumnos de programación sean capaces de acceder a la herramienta desde casa y que además sirva para impulsar el aprendizaje y la innovación de tecnologías en la Universidad de Quintana Roo.

Objetivo general

Desarrollar una herramienta en línea que permita validar y evaluar el código de estudiantes de programación.

Objetivos Particulares

- Brindar un ambiente controlado para la validación de código.
- Compilar el código y comparar a nivel de resultado con uno proporcionado por el profesor.
- Generar un stock de ejercicios para probar la funcionalidad.
- La herramienta debe brindar ayuda para buscar errores.

Alcance

Para cada programa el maestro generará un código que la herramienta comparará con el resultado de código proporcionado por el estudiante.

Todos los programas son compilados en un servidor privado, controlado para minimizar incidentes de seguridad informática.

La interfaz de la herramienta será vía web y constará de un editor de texto, un área que enlazará con el servidor de compilación para mostrar el resultado en pantalla y opcionalmente mensajes de ayuda en caso de que el resultado del estudiante no sea el esperado. La aplicación web compilará código en lenguaje C++.

Todos los ejercicios disponibles serán programas que no interactúan directamente con un usuario al ser ejecutados (programas de impresión de resultado únicamente).

Capítulo 2

MARCO TEÓRICO

La aplicación descrita en este documento hace uso de varias tecnologías de desarrollo web y la correcta interacción entre estas. A continuación, se describen cada una de estas tecnologías.

HTML

HTML, es el elemento de construcción más básico de una página web y se usa para crear y representar visualmente una página web. Determina el contenido de la página web, pero no su funcionalidad [1]. Es el estándar que ha sido adoptado por todos los navegadores de hoy en día para visualizar contenido web.

El lenguaje HTML utiliza elementos de marcado para representar imágenes, textos y demás contenido para mostrar en el navegador web. Comúnmente estos elementos son llamados “etiquetas”

Toda página web está constituida por 3 etiquetas fundamentales y de las cuales se derivan el resto de etiquetas. Las etiquetas fundamentales son:

- <html>
- <head>
- <body>

Se podría decir que una página web no es posible sin la presencia de estas etiquetas.

CSS

Hojas de Estilo en Cascada (Cascading Style Sheets) es el lenguaje utilizado para describir la presentación de documentos HTML o XML, esto incluye varios lenguajes basados en XML como son XHTML o SVG. CSS describe como debe ser renderizado el elemento estructurado en pantalla, en papel, hablado o en otros medios. [2].

CSS es uno de los lenguajes base de la Open Web y posee una especificación estandarizada por parte del W3C. Desarrollado en niveles, CSS1 es ahora obsoleto, CSS2.1 es una recomendación y CSS3, ahora dividido en módulos más pequeños, está progresando en camino al estándar. [2].

En diseño web, CSS es una herramienta fundamental para agregar lo que se denomina “reglas de estilos” a las páginas web. CSS permite redefinir tamaños, formas, posiciones, colores, transiciones,

etc. Las reglas de estilo son definidas para un objeto del navegador mediante un identificador único (ID) o para varios objetos mediante el uso de una clase, para hacer referencia a cada una de estas clases o identificadores existe lo que se denomina “selectores” los cuales consisten precisamente en apuntar a una clase o un ID. Para un objeto pueden definirse una o más reglas de estilos las cuales pueden estar definidas en una o más hojas de estilo. La ilustración 1 ejemplifica el cómo se presentan las reglas de estilo para un objeto en una hoja CSS.

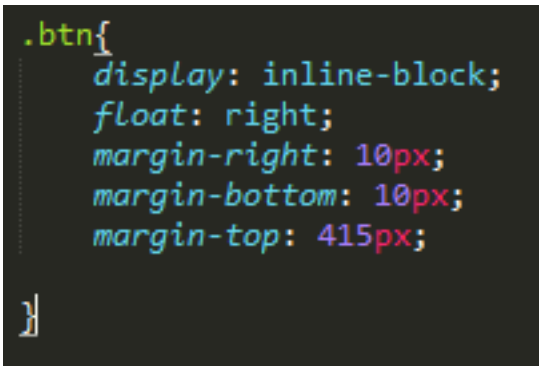


Ilustración 1 Ejemplo reglas de estilos para un elemento HTML

Una de las características fundamentales de CSS es el de poder utilizar una misma hoja de estilo para uno o más documentos HTML, es decir que para un sitio web una o más paginas pueden compartir las mismas reglas de estilo, esto permite que el aspecto visual de un sitio sea más consistente y mucho más fácil de manipular.

Bootstrap

Bootstrap es un framework de desarrollo web muy popular en la actualidad, por el hecho de facilitar el diseño y la construcción de páginas web gracias a su modelo de cuadricula (Grid), el cual permite tener un mejor control del posicionamiento de los elementos dentro de la página además de hacerla responsiva.

El Grid de Bootstrap consta de dividir toda la página en filas y columnas mediante el uso de etiquetas con clases particulares, las cuales están ligadas a reglas de estilo en un documento CSS donde están definidos todos sus parámetros (medidas, posicionamiento, ect.). Se puede insertar tantas filas (row) como sea necesario. cada fila consta de 12 columnas (col) las cuales pueden agruparse para formar una columna más grande. La ilustración 2 ejemplifica mejor el Grid de Bootstrap.

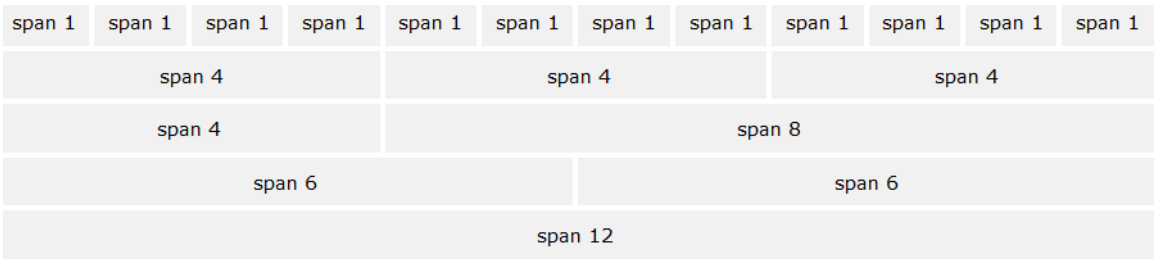


Ilustración 2 Representación del Grid de Bootstrap [3]

Otro de los motivos por el cual Bootstrap es tan popular, es por lo fácil que es personalizar un elemento de la página, dado que las reglas de estilo para sus clases ya están definidas en un documento CSS y bastas con incluir a los elementos deseados dichas clases. Bootstrap utiliza al máximo el potencial de CCS 3 permitiendo a los desarrolladores mayores ventajas a la hora de diseñar un sitio web.

Bootstrap es un framework que incluye funcionalidades tanto para HTML, CSS y JavaScript, en este apartado solo se ha mencionado las ventajas principales para CSS de este framework, dado que su uso para este proyecto no va más allá, para más información se puede consultar el sitio web oficial de Bootstrap.

JavaScript

JavaScript es el lenguaje interpretado orientado a objetos desarrollado por Netscape que se utiliza en millones de páginas web y aplicaciones de servidor en todo el mundo. JavaScript de Netscape es un superconjunto del lenguaje de scripts estándar de la edición de ECMA-262 3 (ECMAScript) que presenta sólo leves diferencias respecto a la norma publicada. [4].

Ilustración 3 Logo representativo de JavaScript

Diferencias respecto a otros lenguajes

A diferencia de Java, C, C++ u otros, JavaScript no es un leguaje compilado por lo que depende de un programa secundario para ser interpretado (navegador

web), esta es la mayor diferencia que existe con respecto a otros lenguajes de programación, dado que la sintaxis para JavaScript es muy parecida a la usada en C y Java.

JavaScript puede funcionar como lenguaje procedimental y como lenguaje orientado a objetos. Los objetos se crean programáticamente añadiendo métodos y propiedades a lo que de otra forma serían objetos vacíos en tiempo de ejecución, en contraposición a las definiciones sintácticas de clases comunes en los lenguajes compilados como C++ y Java. Una vez se ha construido un objeto, puede usarse como modelo (o prototipo) para crear objetos similares. [4].

Implementación

Para poder implementar JavaScript en una página web es necesario que este se encuentre embebido en el documento HTML, para ello se utiliza la etiqueta <script>. Existen dos maneras de realizar el embebido de JavaScript, una es escribir el código dentro del mismo documento HTML (dentro de la etiqueta <script>) y la otra es hacer referencia a un archivo especifico que contiene el código escrito, esto depende mucho del programador y de cómo se contemple la estructura de la página, ya que en algunos casos puede ser más sencillo escribir código dentro del documento HTML. La ilustración 4 ejemplifica las dos formas de realizar el embebido

```
<script type="text/javascript">
    var miTitulo = document.querySelector('h1');
    miTitulo.innerHTML = 'Hello world!';
</script>

<script src="js/main.js" ></script>
```

Ilustración 4 Ejemplos de embebido de JavaScript

Eventos

Cada elemento o etiqueta de una página web tiene un conjunto de eventos que se le pueden asignar, cada uno de estos eventos pueden estar definidos para varias etiquetas HTML y una etiqueta puede estar asociado a eventos diferentes. Estos denominados eventos son los que ocurren cuando se hace click a un botón o a un hipervínculo, cuando se hace click derecho al ratón o al mover el cursor sobre un elemento, entre otros. La tabla 1 resume los eventos más importantes definidos por JavaScript.

Tabla 1 Relación de eventos [5]

Evento	Descripción	Elementos para los que está definido
onblur	Un elemento pierde el foco	<button>, <input>, <label>, <select>, <textarea>, <body>
onchange	Un elemento ha sido modificado	<input>, <select>, <textarea>
onclick	Pulsar y soltar el ratón	Todos los elementos
ondblclick	Pulsar dos veces seguidas con el ratón	Todos los elementos
onfocus	Un elemento obtiene el foco	<button>, <input>, <label>, <select>, <textarea>, <body>
onkeydown	Pulsar una Tecla y no soltarla	Elementos de formulario y <body>
onkeypress	Pulsar una tecla	Todos los elementos
onmousemove	Mover el ratón	Todos los elementos
onmouseout	El ratón “sale” del elemento	Todos los elementos
onmouseover	El ratón “entra” en elemento	Todos los elementos
onmouseup	Soltar el botón del ratón	Todos los elementos
onreset	Inicializar el formulario	<form>
onresize	Se ha modificado el tamaño de la ventana del navegador	<body>
onselect	Seleccionar un texto	<input>, <textarea>
onsubmit	Enviar un formulario	<form>
onunload	Se abandona la página	<body>

Un evento por si solo carece de utilidad, por lo que es necesario ser asociados a funciones o código JavaScript. De esta manera se puede aprovechar la producción de un evento para ejecutar un segmento de código que realice una acción sobre la página. Todos los eventos siguen una estructura de programación parecida, la estructura es muy sencilla: el usuario interactúa con un elemento, se asocia el evento en cuestión y se ejecuta una función para posteriormente realizar el cambio en la página. En la ilustración 5 se puede apreciar el ciclo de un evento.

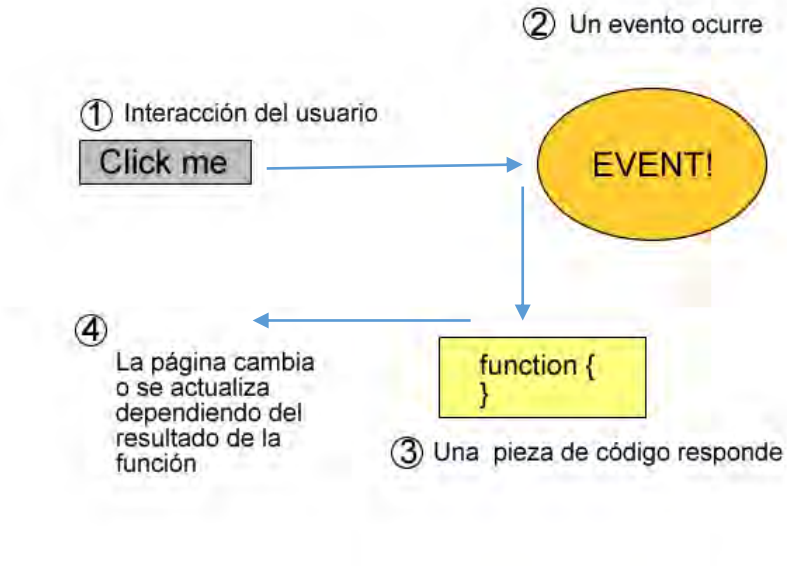


Ilustración 5 Ciclo de un evento

DOM

Document Object Model o DOM (Modelo de Objetos del Documento) es una interfaz de programa para documentos HTML y XML. Proporciona una representación estructurada del documento y define una forma en que se puede acceder a la estructura desde los programas, para que puedan cambiar la estructura, el estilo y el contenido del documento. El DOM proporciona una representación del documento como un grupo estructurado de nodos y objetos que tienen propiedades y métodos. Esencialmente, conecta páginas web a scripts o lenguajes de programación. [6].

En un principio toda página web es un documento que se visualiza mediante la ventana de un navegador, por lo tanto, este documento puede ser representado mediante DOM, esta representación de objetos sigue un orden jerárquico el cual es utilizado por lenguajes como JavaScript para acceder a dichos objetos dado que estos lenguajes no tienen noción alguna de todos los objetos existentes de la página.

JavaScript accede al DOM mediante el uso de métodos que apuntan a atributos como el identificador único de cada objeto o a una clase. Por ejemplo:

```
document.getElementById("Hello").innerHTML = "Hello World!";
```

En el ejemplo anterior se accede a un elemento con el identificador “Hello” y se modifica su contenido por la cadena “Hello World!”.

JQuery

JQuery es una biblioteca de JavaScript rápida, pequeña y característica. Hace cosas como el desplazamiento y manipulación de documentos HTML, manejo de eventos, animación y Ajax mucho más simple con una API fácil de usar que funciona a través de una multitud de navegadores. Con una combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma en que millones de personas escriben JavaScript. [7].

JQuery permite simplificar muchas de las instrucciones al escribir código JavaScript, haciéndolo mucho más fácil de escribir e interpretar; además de reducir el número de líneas de código, esto lo convierte en su principal ventaja dado que muchas veces lo que se necesita al trabajar en proyectos de diseño web es ahorrar tiempo y hacer los códigos lo más reducido posibles para facilitar la de depuración. Obsérvese el siguiente ejemplo:

jQuery

```
$(‘body’).css(‘background’, ‘#ccc’);
```

JavaScript

```
Function changeBachground(color) {
    Document.body.style.background = color;
}
Onload=”changeBackground(‘red’);”
```

Una sencilla línea de código consigue lo que normalmente llevaría 4 líneas de código para conseguirlo en JavaScript.

JQuery ha sido optimizado para trabajar con una gran variedad de navegadores lo cual simplifica mucho el problema de la compatibilidad que podría existir al trabajar con código JavaScript tradicional, además de que esta librería está diseñada específicamente para hacer el código JavaScript más simple y eficiente.

AJAX

JavaScript Asíncrono y XML (AJAX) no es una tecnología por sí misma, es un término que describe un nuevo modo de utilizar conjuntamente varias tecnologías existentes. Esto incluye: HTML o XHTML, CSS, JavaScript, DOM, XML, XSLT, y el objeto XMLHttpRequest. Cuando estas tecnologías se combinan en un modelo AJAX, es posible lograr aplicaciones web capaces de actualizarse continuamente sin tener que volver a cargar la página completa. Esto crea aplicaciones más rápidas y con mejor respuesta a las acciones del usuario. [8].

Con AJAX es posible trabajar con PHP. JavaScript y CSS para armar aplicaciones manteniendo el servidor, la interfaz de usuario y el diseño gráfico como componentes independientes en una arquitectura de cliente servidor.

De acuerdo con la forma de programar las aplicaciones AJAX las podemos dividir en dos: clientes livianos y clientes pesados. Los clientes livianos se llaman así porque tienen relativamente poco código del lado del cliente. Generalmente una llamada AJAX que al responder ejecuta una función que reemplaza el contenido de un DIV por el texto que devolvió el servidor. Es lo primero que la gente hace para implementar el "look" ajax en un programa. Y es lo más parecido a seguir generando HTML desde el lenguaje de servidor (PHP, ASP) pero sin hacer refresh entre páginas. [9].

Actualmente el uso de AJAX está presente en casi todas las aplicaciones web y en sitios que buscan hacer su página más dinámica para los usuarios, todo esto derivado del módulo AJAX que contiene la librería JQuery.

PHP

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. [10].

Ilustración 6 Logo PHP

Lo que distingue a PHP de la tecnología del lado del cliente como JavaScript es que el código es ejecutado en el servidor, genera un HTML y se envía al cliente. El cliente recibirá el resultado de ejecutar el script, aunque no se sabrá cuál es el código subyacente. El

servidor web puede ser configurado incluso para que procese todos los ficheros HTML con PHP, por lo que no hay manera de que los usuarios puedan saber qué se tiene debajo de la manga. [10].

PHP está enfocado principalmente a la programación de scripts del lado del servidor, por lo que se puede hacer cualquier cosa que pueda hacer otro programa CGI, como recopilar datos de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies. Aunque PHP puede hacer mucho más. [11].

PHP también nos proporciona las siguientes posibilidades [12]:

- Soporte para múltiples sistemas operativos; Unix (entre otras, Linux, HP UX, Solaris y OpenBSD), Microsoft Windows, Mac Os X, RISC OS. Actualmente está en preparación para las plataformas IBM OS/390 y AS/400.
- Soporte para múltiples servidores Web: Apache, Microsoft Internet Information Server,Personal Web Server, Netscape e iPlanet, Oreilly Website Pro Server, Caudium, Xitami, OmniHTTPd y muchos otros.
- Soporte para ODBC y extensiones DBX.
- Soporte para comunicarse con otros servicios usando protocolos tales como: LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros.
- Puede utilizar objetos Java de forma transparente, como objetos PHP.
- La extensión de CORBA puede ser utilizada para acceder a objetos remotos.
- PHP soporta WDDX para intercambio de datos entre lenguajes de programación Web.
- Generación de resultados en múltiples formatos como XHTML, XML ficheros de imágenes, ficheros PDF y películas Flash.
- Manejo de expresiones regulares POSIX Extended o Perl.
- Funciones de comercio electrónico, como Cybercash, Cybermut Verisign Payflow Pro y CCVS para pasarelas de pago.
- Otras extensiones muy interesantes so las funciones del motor de búsquedas mnoGosearch, funciones para pasarelas de IRC, utilidades de compresión (gzip, bz2), convención de calendarios y traducciones.

PHP también se puede usar para ejecutar programas externos. En muchas ocasiones puede ser necesario que la aplicación web recurra a un programa alojado en el servidor para realizar acciones

que no serían posibles desde la aplicación. La librería principal de PHP contiene un conjunto de funciones que permiten ejecutar un programa externo, estas funciones devuelven el resultado de la ejecución del programa. Para ejecutar un programa externo estas funciones reciben como parámetro una instrucción de línea de comando como un string y posteriormente es ejecutado en el servidor. Cada función tiene un uso distinto y depende mucho de lo que se necesite, la función a utilizar. Las funciones mencionadas son:

- **escapeshellarg** — Escapar una cadena a ser usada como argumento del intérprete de comandos. [13]
- **escapeshellcmd** — Escapar meta-caracteres del intérprete de comandos. [13]
- **exec** — Ejecutar un programa externo. [13]
- **passthru** — Ejecuta un programa externo y muestra la salida en bruto. [13]
- **proc_close** — Cierra un proceso abierto mediante proc_open y devuelve el códido de salida de tal proceso [13]
- **proc_get_status** — Obtiene información sobre un proceso abierto por proc_open. [13]
- **proc_nice** — Modificar la prioridad del proceso actual. [13]
- **proc_open** — Ejecuta un comando y abre un puntero de fichero para entrada/salida. [13]
- **proc_terminate** — Mata un proceso abierto mediante proc_open. [13]
- **shell_exec** — Ejecutar un comando mediante el intérprete de comandos y devolver la salida completa como una cadena. [13]
- **system** — Ejecutar un programa externo y mostrar su salida. [13]

Ace Editor

Ace es un editor de código incrustado escrito en JavaScript. Combina las características y el rendimiento de los editores nativos como Sublime, Vim y TextMate. Se puede incrustar fácilmente en cualquier aplicación de página web y JavaScript. Ace se mantiene como editor principal de Cloud9 IDE y es el sucesor del proyecto Mozilla Skywriter (Bespin). [14]

Ace es posiblemente el editor de código más popular de la web, la mayoría de los sitios web en donde se pueda tomar cursos de programación ya sea web o de cualquier leguaje en específico incluye este editor para brindar la posibilidad de editar los códigos, el motivo principal es la variedad de características que ofrece Ace, las cuales son las siguientes [14]:

- Resaltado de sintaxis para más de 110 idiomas (se pueden importar archivos TextMate / Sublime Text.tmlanguage).
- Más de 20 temas (se pueden importar archivos TextMate / Sublime Text.tmtheme)
- Recorte automático y extractor.
- Una línea de comandos opcional.
- Maneja documentos enormes (¡cuatro millones de líneas parecen ser el límite!).
- Complementos de teclas totalmente personalizables, incluidos los modos vim y Emacs.
- Buscar y reemplazar con expresiones regulares
- Resaltar paréntesis coincidentes.
- Alternar entre pestañas suaves y pestañas reales.
- Muestra caracteres ocultos.
- Arrastre y suelte texto con el ratón.
- Envoltura de líneas.
- Plegado de código.
- Múltiples cursores y selecciones.
- Comprobador de sintaxis en vivo (actualmente JavaScript / CoffeeScript / CSS / XQuery).
- Funcionalidad de cortar, copiar y pegar.

Ace se encuentra alojado en los repositorios de GitHub y está disponible para cualquiera que desee descargarlo e incrustarlo en su sitio web.

Ace se utiliza en toda la web en todo tipo de aplicaciones de producción. Algunos de los proyectos donde utilizan Ace son: Khan Academy, Codecademy, Cluoud 9 IDE, Rstudio, Zed, Tedit, Wikipedia, Chrome Dev Editor, Weecod, entre muchos más. [15]

Shell Scripting

Es una técnica muy popular al trabajar en sistemas basados en UNIX, la cual consiste en generar un archivo (script) que contenga un conjunto de comandos, para después ejecutarlos en orden secuencial desde el primero hasta el último, con el fin de automatizar las tareas en la shell. Los comandos que contengan los scripts varían en función de la shell con la que se esté trabajando. Las shell más populares son **bash y sh.**

Al generar un script para shell se puede trabajar con las herramientas típicas de los lenguajes de programación como lo son los ciclos, las variables, las operaciones aritméticas, los controles de flujo, etc. Lo cual añade una ventaja considerable al realizar tareas en la shell.

Para el caso particular de este proyecto, se trabajó con una shell **bash** y en el script generado se incluyeron comandos de procesamiento básico de texto, como lo son:

- `head y tail.` Los comandos `head` y `tail` se usan para examinar las partes superiores (cabecera) o inferiores (pie) de los archivos.
- `cut.` El comando `cut` se utiliza para 'recortar' secciones basadas en columnas de un archivo o de datos redirigidos al mismo desde `stdin.`
- `diff.` El comando `diff` compara dos archivos, informando las deferencias que hay entre ellos.
- `wc.` El comando `wc` (wordcount) cuenta el número de líneas, palabras (separadas por espacio blanco), caracteres en filas especificadas o de `stdin.`
- `cat`. El comando `cat` es uno de los comandos más básicos. Se usa para crear, adjuntar, mostrar y concatenar archivos.

Existen muchos más comandos para procesamiento de texto en shell, si se desea obtener más información de estos comandos y de los mencionados en este documento, se puede visitar el artículo de IBM "*Introducción a la manipulación de texto en sistemas basados en UNIX*", el cual contiene información más detallada. Se puede acceder al artículo desde la siguiente URL: https://www.ibm.com/developerworks/ssa/aix/library/au-unixtext/.

Seguridad en aplicaciones web

La seguridad siempre ha sido un tema de suma importancia en el desarrollo web, especialmente cuando se desarrollan aplicaciones que de algún modo tienen más acceso al servidor, ya sea para ejecutar algún programa o acceder a archivos, las medidas de seguridad varían dependiendo las acciones de la aplicación en el servidor. Para el caso de las aplicaciones que ejecuten algún programa externo en el servidor, existe la posibilidad de aislar procesos, hay muchas técnicas de cómo lograr esto, una de las más conocidas es lo que se denomina **Sandbox-**

Un **Sandbox** puede verse de manera simple como un sistema de pruebas, es un entorno que permite la ejecución de códigos nuevos o de dudosa procedencia, a menudo estos entornos se

encuentran separados del servidor principal. El aislamiento permite controlar mejor los recursos al que el cliente puede acceder. Normalmente se restringe de igual forma el acceso a las redes desde el **Sandbox.** Aislar las ejecuciones de código en un servidor permite tener mejor control de la seguridad para las aplicaciones web, ya que no se compromete el servidor principal, de esta manera todos los procesos que se ejecuten no impactaran de manera directa a la red.

Otra alternativa para aislar procesos de una aplicación web o una página es restringir el acceso de **Apache** en el servidor. Los sistemas basados en UNIX cuentan con una herramienta llamada **chroot** la cual invoca un proceso y cambia para este y sus subprocesos el directorio raíz del sistema, en términos simples, se aíslan los procesos en un directorio específico dentro del sistema. Esto funciona perfectamente pera establecer limitantes a **Apache** en su acceso al servidor y de esta manera tener un mejor control de los procesos que se puedan generar desde el lado del cliente en la web. La ilustración 7 describe el aislamiento de apache con chroot.

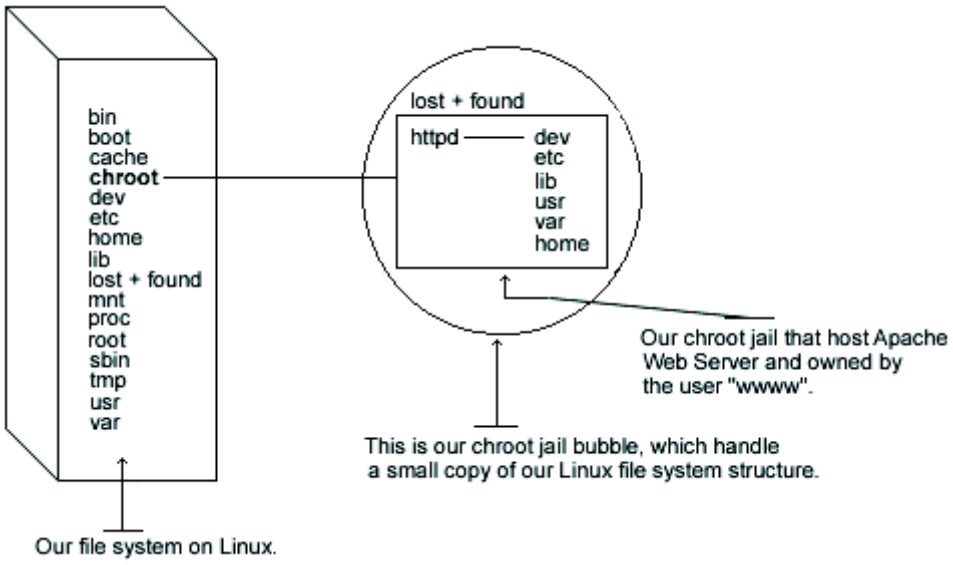


Ilustración 7 Apache in a chroot jail [16]

Cuando esta alternativa es utilizada se tiene que tener en cuenta las siguientes consideraciones [16]:

- Si Apache está comprometido, el atacante no tendrá acceso a todo el sistema de archivos.
- Las secuencias de comandos CGI mal escritas que pueden permitir que alguien acceda a su servidor no funcionarán.

- Hay bibliotecas adicionales que necesitarás tener en la cárcel de chroot para que Apache funcione.
- Si utiliza cualquier característica de Perl / CGI con Apache, necesitará copiar los archivos binarios, bibliotecas Perl y archivos necesarios en el lugar apropiado dentro del espacio chroot. Lo mismo se aplica a SSL, PHP, LDAP, PostgresSQL y otros programas de terceros.

Capítulo 3

DESAROLLO

Este capítulo está dedicado a la descripción del proceso de análisis, los requisitos, el diseño y la configuración que se realizaron durante el desarrollo de la aplicación web.

Requisitos

Los siguientes fueron los requisitos sobre los cuales se basa el diseño de la interfaz.

- Contar con un espacio de trabajo para que los alumnos puedan escribir código.
- Contar con un área para mostrar errores y resultados.
- Tener la posibilidad de visualizar más de un ejercicio.
- Contar con un área de retroalimentación.
- Incluir la opción para guardar el progreso en cada ejercicio.
- Tener la posibilidad de descargar el código generado.
- Reiniciar el ejercicio cunado el alumno lo solicite.
- Una opción para probar el trabajo antes de enviar a calificar.

Los requisitos para la parte funcional del programa son los siguientes.

- Compilar el código en un servidor, de preferencia Unix o tipo Unix dado que el sistema Moodle de la universidad está trabajando sobre un servidor Solaris.
- El guardado de progreso de cada ejercicio debe ser en el host del cliente y no el servidor.
- El resultado de los ejercicios se da mediante la comparación del resultado de la compilación del código generado por el alumno y uno generado por el profesor.
- Contar con una medida para controlar el resultado de la comparación de los programas (el del alumno y el del profesor) para que sin importar el procedimiento del alumno la respuesta (en caso de ser correcta) sea igual a la esperada.
- La aplicación debe brindar tres tipos de respuestas para cuando se prueben los códigos:
    1. “Error de compilación”.
    2. “Bien”.
    3. “No hay coincidencia”.

    Para el caso de errores de compilación se debe mostrar en pantalla los errores que genera el compilador en el servidor.

Si la respuesta es correcta se imprime el resultado en el espacio designado para este.

Para cuando el código del alumno se compile sin errores, pero no coincida con el resultado esperado, se debe informar en que línea del resultado está el error y mostrar la respuesta esperada para esa línea.

- La comparación de resultados debe realizase en el servidor mediante un script en Shell bash.
- Debe considerarse una medida de seguridad para contener la ejecución de los códigos en el servidor.

Análisis

Diseño de componentes

En el desarrollo de la aplicación una de las partes más importantes es determinar los componentes de la interfaz gráfica y el cómo estarán distribuidos los elementos en el espacio visible para el usuario.

Ilustración 8 Bosquejo de componentes de la interfaz

La ilustración 8 nos muestra el bosquejo de lo que será la interfaz del programa, se puede apreciar que está constituida por cuatro espacios distintos. En la parte superior se encuentra la herramienta que nos permita la edición de texto, dado que ese será el espacio de trabajo para los alumnos, en esta área se escribirá todo el código que se tenga que generar, para posteriormente mandarlo a compilar.

Seguido nos encontramos con un área de botones, los cuales están considerados para realizar las funciones requeridas para la aplicación, también podemos apreciar un botón con la etiqueta “Ejercicios”, cuya función nos permita seleccionar de entre los ejercicios disponibles y dado que se trabajaráa con HTML la mejor opción es usar una etiqueta “<SELECT>” que se genera de manera dinámica mediante JavaScript, la cantidad de ejercicios que se listen en esta dependerá de la cantidad de archivos que se encuentren en el directorio de ejercicios..

Más abajo se encuentra el “Área de debug”, aquí se presentarán los resultados tras la compilación que incluye: errores, disparidades, y por supuesto respuestas correctas. Esta área está directamente relacionada con el botón “Probar” que será el que mande la instrucción de compilación.

Por último, encontramos el “Área de retroalimentación”, este espacio está pensado para que el profesor coloque toda la información que considere le será de utilidad al alumno para poder resolver el ejercicio. La información que aparezca aquí estará ligada al ejercicio en cuestión.

Arquitectura

El funcionamiento del programa está constituido de tres partes fundamentales: la interfaz de usuario, el servidor y el enlace entre estos. En la ilustración 9 se puede apreciar de forma gráfica la arquitectura de la aplicación.

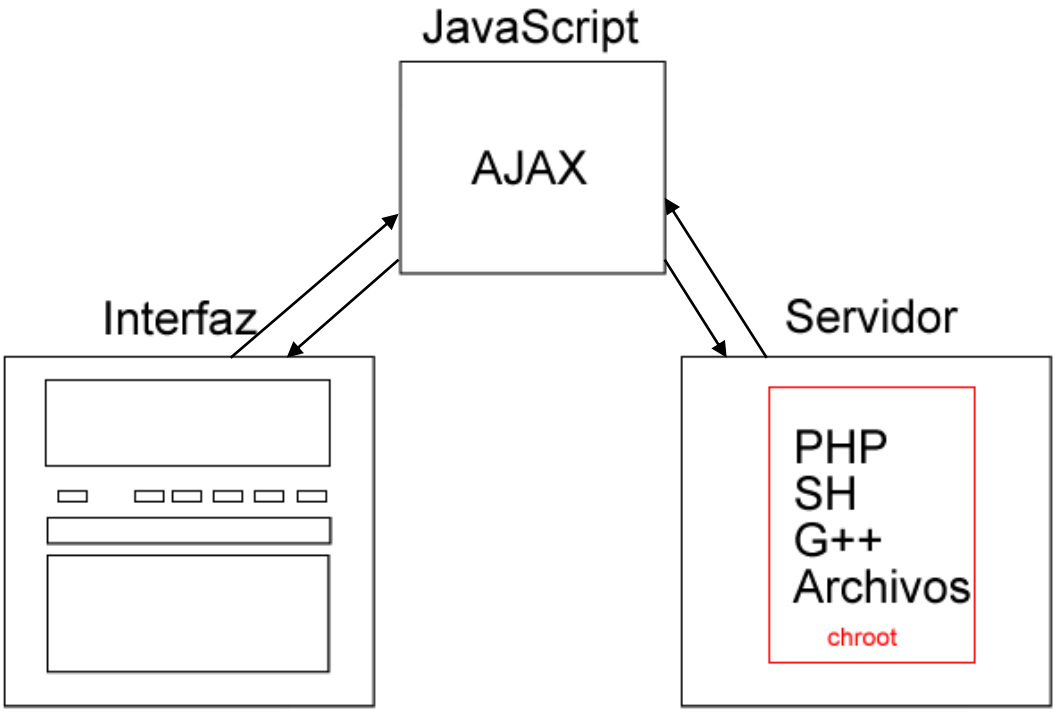


Ilustración 9 Arquitectura de conexión con el servidor

La interfaz es la parte del programa con el que el usuario tendrá interacción directa, es decir: todos los botones, el editor, el selector de ejercicios y las áreas de debug y de retroalimentación. Por otra parte, el servidor contiene los elementos necesarios para el correcto funcionamiento de todos los componentes de la aplicación, en él se encuentran instalados los servicios de PHP y el compilador

para C++; así como todos los archivos que se crearan y cargaran a la interfaz y el Script de comparación de resultados hecho en bash. Adicionalmente se está considerando lo que se denomina *Apache jail,* que son las limitaciones de apache dentro del servidor. Es una medida para prevenir huecos de seguridad evitando la ejecución de procesos fuera del directorio indicado para Apache. La ilustración 10 muestra la estructura de la integración de los lenguajes utilizados para la construcción de la aplicación.

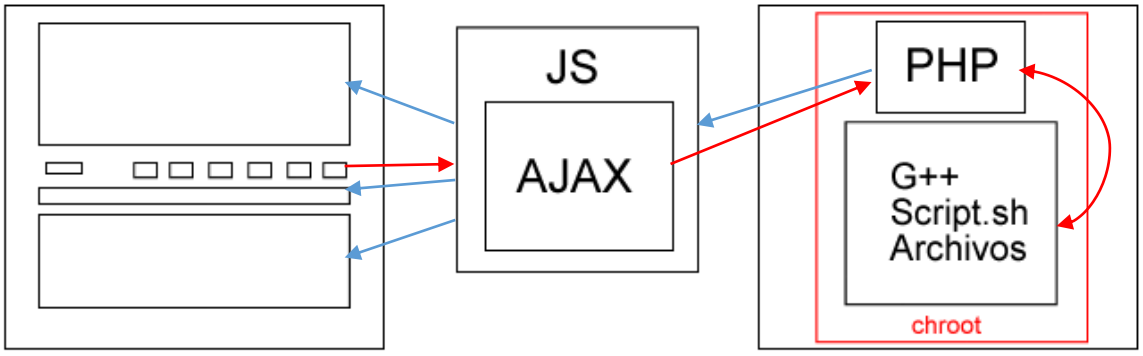


Ilustración 10 Estructura de interacción de lenguajes

El enlace entre la interfaz de usuario y el servidor está controlado mediante JavaScript haciendo uso del framework *JQuery* y el módulo de *Ajax* que este contiene. Con *JavaScript* se controlan todas las acciones del usuario en la aplicación y los cambios visibles en esta, mientras que al mismo tiempo y mediante *Ajax* se hacen peticiones al módulo PHP que se encarga de todas las actividades de manipulación y acceso a archivos y directorios; mediante PHP también se ejecuta el compilador y el script Shell bash encargado de la comparación de resultados.

Configuración

El código del programa está repartido en 4 módulos con tecnologías distintas, cada módulo se encarga de un aspecto fundamental del programa. La configuración de cada uno se explica a continuación.

HTML y CSS. La interfaz gráfica de usuario (GUI) está pensada para mostrarse en un entorno web, por lo tanto, su estructura básica se encuentra escrito en *HTML y CSS*. Para completar el diseño de la interfaz el documento principal contiene embebido el framework para desarrollo web *Bootstrap* y

la librería para JavaScript *JQuery* para realizar un trabajo más completo. También contiene embebido el editor "*ACE Editor*" y aunque este está diseñado con tecnología JavaScript, la incrustación se hace en el documento *HTML*. La ilustración 11 muestra el código con el cual se inicia el editor y sus configuraciones.

```
<script>
    // trigger extension
    ace.require("ace/ext/language_tools");
    var editor = ace.edit("editor"); //asociasion de la variable con el
    espacio en la pagina
    editor.session.setMode("ace/mode/c_cpp") //sintaxis del editor
    editor.setTheme("ace/theme/monokai"); //thema del editor
    // enable autocompletion and snippets
    editor.setOptions({
        //opciones para configuracion del editor
        enableBasicAutocompletion: true,
        enableSnippets: true,
        enableLiveAutocompletion: false
    });

</script>
```

Ilustración 11 Código de inserción de Ace Editor

PHP. Es posiblemente el módulo principal para el funcionamiento de todo el programa, dado que es el encargado de acceder al servidor para lectura y escritura de archivos, la ejecución de programas dentro de este y el acceso a los directorios. A continuación, se describen los métodos fundamentales con los que se trabajó.

$_POST. Este método es el que recibe la información desde el archivo *Script.js* para posteriormente asignarlo una variable y ser tratado dentro del código. Comúnmente es utilizado para trabajar con formularios en páginas web y dado que el editor de texto de la página es básicamente un formulario e de aquí su uso. El uso de este método es fundamental ya que sin el no sería posible trabajar el código PHP en un archivo separado al HTML.

is_writable. Utilizado para comprobar si un archivo es escribible en el servidor, su uso dentro del código está en conjunto con la sentencia *"if"* y con los métodos *fopen y fwrite*.

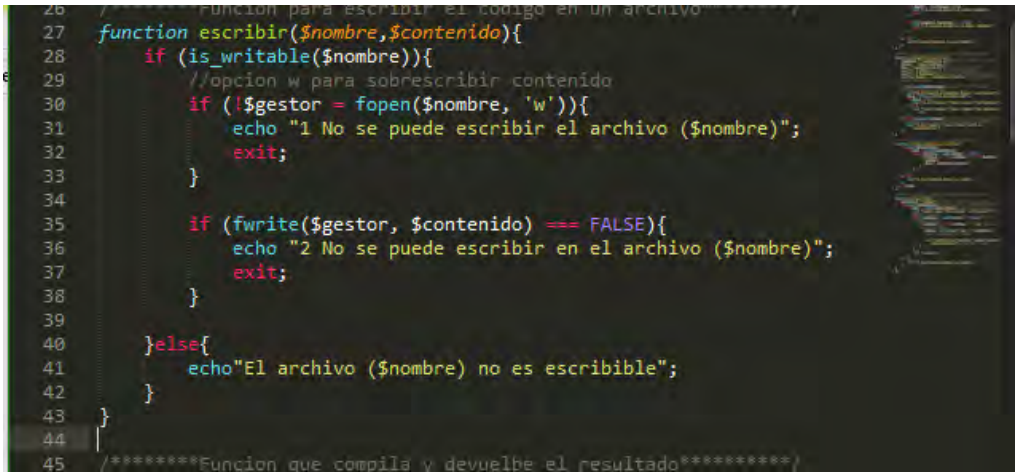


Ilustración 12 Función para modificar el contenido de un archivo

La ilustración 12 muestra la estructura de la función que se generó para escribir contenido en los archivos, donde primero se verifica que se pueda acceder al archivo especificado, si lo anterior es verdadero se prosigue a abrir el archivo en modo de sobre escritura en caso de que este tenga contenido, al mismo tiempo se realiza una verificación de que sea posible abrir el archivo en ese modo y se asigna al archivo abierto a una variable. Por último, se procede a escribir el contenido en el archivo y se verifica que esta acción se realice correctamente. En caso de que una de las verificaciones falle se devuelven mensajes de error que en posteriormente serian mostrados en el área de debug del programa.

fopen y fwrite. Métodos que nos permiten acceder a un archivo en el servidor para escribir contenido. El método *fopen* permite crear el archivo si es necesario. La ilustración 12 ejemplifica el uso de estos métodos.

exec. Permite ejecutar líneas de comando en el servidor desde el archivo PHP. Es un método muy importante dado que es el que se utiliza para ejecutar el compilador y también el que se usa para ejecutar el *script* hecho en *bash*, el cual comprueba los resultados de los ejercicios.

```
46  function compilar($nombre,$resp_c){
47      $compilado = "compilacion/a.out";//ruta del archivo de resultado
48      $result = "respuestas/$resp_c";
49      $log = "compilacion/log.txt";//ruta del archivo de log
50      exec("rm -f $log");//borrar el archivo log existente
51      exec("rm -f $compilado");//borrar archivo de resultado existente
52      exec("g++ $nombre -o $compilado 2> $log");//compilar el codigo y
        generar arechivo log
53
54      if (is_readable($compilado)){//comprobar que el archivo de
        resultado existe y es legible
55          exec("sh aux/compare2.sh $compilado $result > aux/resp.txt");
56          $resp = shell_exec("cat aux/compare.dat");
57          if ($resp == 1){
58              echo "<p id=\"cabecera\" class=\"success\">¡Correcto!</p>";
59          }else {
60              echo "<p id=\"cabecera\" class=\"nomatch\">¡No Coincide!</
                    p>";
61          }
62      }elseif (is_readable($log)){//comprobar que el archivo log existe
        y es legible en caso que el archivo de resultado no existiera
63          echo "<p id=\"cabecera\" class=\"error\">¡Error de
                Compilación!</p>";
64      }
65  }
```

Ilustración 13 Función para compilar en el servidor

La ilustración 13 muestra la función *compilar* y el uso del método *exec* dentro de esta, se puede apreciar que en la las líneas 50 a 52 se encuentra presente el método y los parámetros que contiene son instrucciones del intérprete de comandos de Shell y específicamente la línea 52 es la que contiene la instrucción para ejecutar el compilador *g++* con instrucciones para generar dos archivos posibles, uno para guardar el resultado de la compilación y el otro para contener todos los posibles errores de compilación que se puedan generar. Dependiendo que archivo es el que se genere hay dos posibles rutas que el código puede seguir, en una se ejecuta la siguiente instrucción en Shell para comprobar el resultado de la compilación y dependiendo lo que sea determinado se crean etiquetas HTML con las cabeceras que se colocaran en el área de debug del programa, mientras que la segunda ruta genera la cabecera de error de compilación. En la ilustración 13 también se puede apreciar la aparición del método *shell_exec,* que es una variante de *exec* y devuelve la salida estándar (stdout) de la instrucción que contiene como parámetro, en este caso un número que indica si la respuesta del ejercicio corresponde con el resultado esperado o no.

is_readable. Método que verifica si el archivo especificado es legible. Dentro del código este método es utilizado para verificar la existencia del archivo que se genera tras ejecutar el compilador y después decidir qué acciones debe realizar el programa. En la ilustración 13 se puede apreciar el uso de este método.

is_dir. Recibe un parámetro y verifica si éste es un directorio, este método es el que marca la pauta para acceder a un directorio y leer los archivos y/o directorios que contiene con los métodos de *opendir y readdir.*

opendir y readdir. Su función es muy similar a *fopen y fwrite* pero son utilizados para acceder a directorios. Estos dos métodos son usados para acceder al directorio de los ejercicios para posteriormente generar el conjunto de etiquetas que se insertaran en el documento HTML.

```
84  function generar_html(){
85      $directorio ="codigos/"; //declarar directorio de busqueda
86      if (is_dir($directorio)){ //verificar si es un directorio valido
87          if ($dir = opendir($directorio)){ //abrir el directorio
88              echo "<select class=\"form-control\" id=\"selector\">"; //
                imprime etiqueta
89              while (($archivo = readdir($dir)) != false){ //ciclo de
                busqueda de arcivos
90                  if ($archivo != '.' && $archivo != '..'){ //descartar
                    archivos de sistema
91                      $nombre = str_replace(".cpp", "", $archivo);//
                        quitar la extencion del archivo
92
93                      echo "<option value=\"$archivo\" class=\"selector\"
                            >$nombre</option>"; //imprimir etiqueta
94                  }
95              }
96              echo '</select>';
97          }
98      }else{
99          echo "no hay directorio valido o no existe";
100     }
101 }
```

Ilustración 14 Función generadora de etiquetas para lista de ejercicios

En la ilustración 14 se puede apreciar el uso de los métodos *is_dir, opendir y readdir.* El proceso es simple, primero verificamos que el parámetro que se recibe sea un directorio en el servidor, acto seguido se recorre el directorio al mismo tiempo que se va generando el

conjunto de etiquetas que componen una estructura “SELECT” para posteriormente ser insertadas en el documento HTML. Cuando las etiquetas son creadas son asociadas con los archivos que contiene el directorio.

Nota: un punto muy importante en la implementación de estos métodos dentro del código *PHP*, es tener en cuenta que el servidor sea correctamente configurado para dar acceso a *Apache*, es decir que *Apache* cuente con los permisos necesarios para trabajar en los directorios especificados. Otro punto de importancia es el *módulo de seguridad para kernel Linux* (SELinux), es muy importante que éste se encuentre deshabilitado o de lo contrario podría causar conflictos con la ejecución de los métodos implementados para esta aplicación. Todo esto tomando en cuenta que el servidor donde se realizaron las pruebas está en un sistema *Linux,* esta información será puede ver con más detalle en el capítulo 4.

JavaScript. Este módulo es el encargado de controlar todas las interacciones del usuario en la *GUI,* en este módulo es donde entra en acción la libreria *JQuery*, el cual permite que las configuraciones sean más sencillas de implementar y de entender, dado que la sintaxis de la mayoría del código esta reducida a instrucciones más cortas. En este módulo también se encuentran las funciones para enlazar con el servidor mediante PHP.

Modulo Ajax de JQuery. Una parte fundamental del código *JavaScript* son las funciones Ajax que están presentes en todo el documento con la finalidad de enlazar con el código *PHP* para realizar tareas que no son posibles desde *JavaScript*. Estos segmentos de código se pueden encontrar en los eventos de los botones *compilar y descargar,* así como en las instrucciones para construir las etiquetas *<select>,* también se puede encontrar código *Ajax* en la función utilizada para asignar el contenido al editor y al área de retroalimentación. En la ilustración 15 se puede apreciar la sintaxis de una función utilizando *Ajax.*

```
95      $('.btn-d').on('click', function(e){
96          var code = $("#codigo").val(editor.getValue());
97      var texto = code.val();
98      $.ajax({
99        url: "aplication.php",
100       type: "post",
101       data: {codigo: texto, flag: "modificar_archivo", dir: ""},
102       success: function(data){
103         var nombre = $("#selector").val();//recuperacion de nombre de ejer
104         $("#link").attr('download',nombre);//asignacion de nombre de archi
105         document.getElementById("link").click();//genera el click en la et
106       }
107     });
108      });
```

Ilustración 15 Ejemplo de llamada Ajax

Una sentencia Ajax está compuesta principalmente por 4 partes: *url, type, data y succes*. **url**: hace referencia al documento con el que se va a enlazar. **type**: es el tipo de enlace (*GET* o *POST* para *PHP*). **data**: son los parámetros que serán enviados. **succes:** apartado donde se escribe la función que se ejecutara con los datos del resultado devuelto por *PHP*.

El código de la ilustración 15 pertenece al evento *click* del botón “descargar”, donde se hace una petición para generar un archivo con el texto actual del editor para después modificar el atributo name de una etiqueta <*a*> insertada previamente en el documento HTML, con un nombre de archivo (nombre de ejercicio correspondiente) y simular el click en esta para poder dar la posibilidad al usuario de descargar su código escrito. La ilustración 16 muestra la ventana que despliega normalmente el navegador al descargar un archivo.

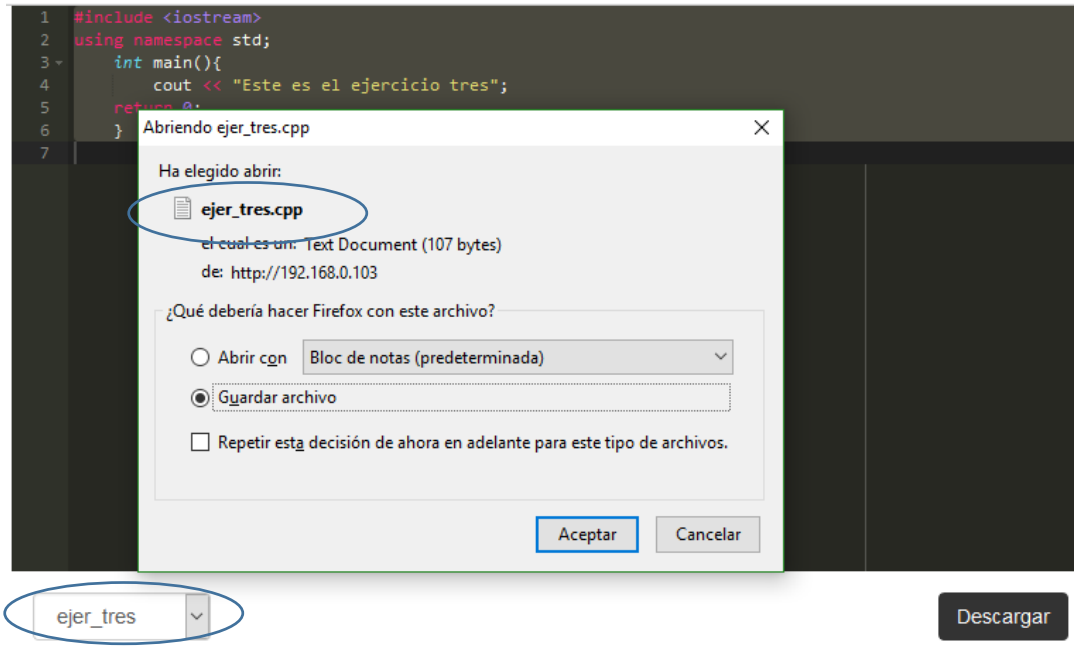


Ilustración 16 Descarga de código

Otra de las funciones que se le dio al módulo de *Ajax*, es la de hacer la petición para generar el conjunto de etiquetas que componen la lista de ejercicios. De la misma manera que en la ilustración # se realiza una peticon al archivo “aplication.php” y se hace la construcción de etiquetas que posteriormente son utilizadas como parámetro del método ***append*.** El tamaño de la lista está en función del número de archivos que se encuentren en el directorio de ejercicios. La ilustración 17 ejemplifica el cómo se ve la lista de ejercicios con 7 de estos.

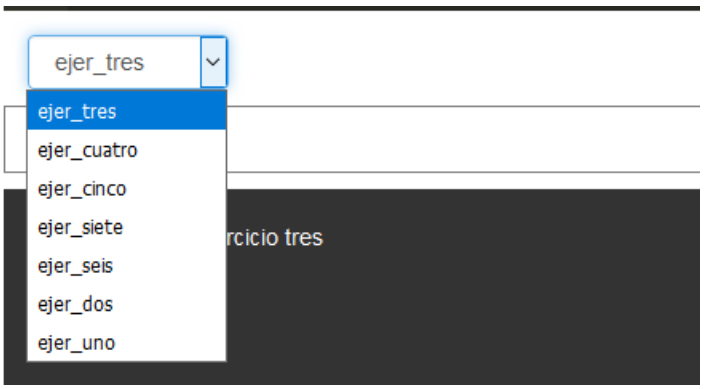


Ilustración 17 Lista de ejercicios

La lista de los ejercicios se genera cada vez que se recarga la página, y son mostrados cronológicamente por fecha de modificación de los archivos, es decir, los últimos ejercicios agregados aparecerán al principio de la lista.

Cada elemento de la lista está ligado a dos archivos: el archivo de ejercicio que contiene el código que se mostrara en el editor y el archivo de retroalimentación que se colocara en el área con el mismo nombre. La relación antes mencionada es posible mediante la función generada “asignar_contenido”, la cual se encarga de realizar una petición al archivo PHP para localizar los archivos correspondientes para después acceder a su contenido mediante el uso otro método de Ajax ($.get) e insertarlos en el área correspondiente. La ilustración ejemplifica el uso del método $.get.

```
$.get(data, function(contenido){
    if (dir == "codigos/"){
        editor.setValue(contenido);
    }else{
        $("#informacion").val(contenido);
    }
});
```

Ilustración 18 Función get de Ajax

La instrucción para compilar está controlada con el evento que se produce al hacer click en el botón *“probar”* que del mismo modo que con la lista de ejercicios y el botón *“descargar”* hace uso del módulo *Ajax* para completar esta acción.
En este caso el código PHP se encarga de compilar el código que se le es enviado en la instrucción *Ajax,* la respuesta a esta petición es una de tres posibles etiquetas que hacen la función de encabezado para mostrar respuesta al usuario y dependiendo de cuál sea la etiqueta recibida es la acción que tomara la función.

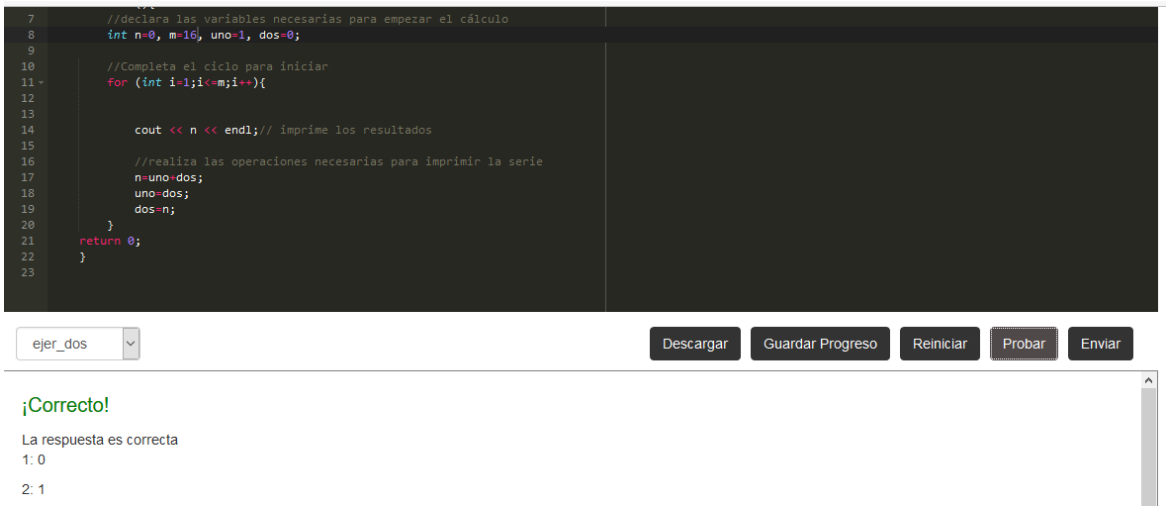


Ilustración 19 Respuesta del programa ante un resultado correcto

El primer encabezado es el que corresponde a una respuesta correcta por parte del usuario. Las ilustraciones 19, 20 y 21 muestran la información que se despliega con cada envesado generado.

Ilustración 20 Respuesta del programa ante un resultado erróneo

El segundo encabezado corresponde a un error de coincidencia con el resultado.

Ilustración 21 Respuesta del programa ante un error de compilación

El tercer encabezado es el correspondiente a errores de compilación.

La información que aparece junto con los encabezados es la que genera el *script en Shell bash,* esto para los primeros dos encabezados, pera el caso de los errores de compilación la información es tomada del archivo “log” que se genera al mandar la instrucción para compilar, esta información se obtiene con el método *“$.get”*.

sessionStorage. Uno de los requisitos para el programa, es que sea posible para el usuario guardar su progreso actual en cada ejercicio (en el navegador del cliente), al principio se pensaba en el uso de *cookies*, pero esto significaba un problema dado que una *cookie* no se activa hasta recargar la página. Por otro lado, *sessionSorage* funciona del mismo modo que una *cookie,* pero sin esta limitación haciendo posible guardar variables en el navegador y acceder a estas en la misma sesión de navegación.

En la ilustración 15 se puede apreciar el segmento de código que se usa para guardar las variables en el navegador. Básicamente lo que se hace es programar un evento que responda al botón “guardar progreso” y dentro de su función se crea una variable con el contenido del

editor en ese instante, posteriormente se crea otra variable con el nombre del ejercicio actual (extrayendo el valor actual de la etiqueta <select>), estas dos variables son guardadas en el navegador mediante la instrucción "*sessionStorage.setItem(item,codigo)"*. Donde el parámetro "item" representa el nombre del ejercicio y el nombre con el que se guardara la variable (se guarda con el nombre del ejercicio), mientras que el parámetro "código" representa el texto del editor y el valor de la variable, de esta manera se asegura que cada variable guardad en el navegador corresponda a un ejercicio diferente. Para verificar que la variable se ha guardado con éxito, se recupera la misma mediante le instrucción "*sessionStorage.getItem(item),* acto seguido, se comprueba que el valor de la variable corresponda con el valor de la variable que se envió.

Dentro del código también se podrá encontrar la instrucción "*sessionStorage.removeItem()",* esta instrucción se encuentra el segmento de código correspondiente al evento del botón "reiniciar" y cuya finalidad es remover la variable existente para el ejercicio en cuestión.

Shell script bash. Una parte fundamental para el funcionamiento del programa es el cómo se realiza la verificación del resultado para cada ejercicio que el alumno resuelva con lo que el profesor brinde como respuesta, para ello se hace uso de las propiedades del *Shell* en el servidor mediante un script escrito en este, dicho scrip*t* contiene instrucciones para comparar ambos resultados y brindar una respuesta que pueda mostrarse en la *GUI*.

El script empieza recibiendo dos parámetros (respuesta del alumno y respuesta del profesor) los cuales son dos archivos ejecutables creados por el compilador, posteriormente estos archivos son ejecutados con un tiempo límite de ejecución como medida para prevenir bucles infinitos y así reducir la carga de funcionamiento en el servidor, cuando estos archivos son ejecutados, todo su contenido es transferido a dos archivos de texto. La comparación se realiza accediendo a los dos archivos de texto al mismo tiempo y comparando línea por línea los caracteres que contengan. Lo que se pretende al realizar la comparación línea a línea es brindar al alumno la oportunidad de ver en qué punto su respuesta es incorrecta y que además pueda visualizar la diferencia y sin revelarle toda la respuesta, esto suponiendo que las respuestas de los ejercicios estén conformadas por más de una línea. El script al detectar una inconsistencia imprime todas las líneas correctas antes de la

que no coincide y se cierra, y cuando todas las líneas son correctas simplemente imprime el archivo tal cual.

Las instrucciones básicas utilizadas en la construcción del Shell script bash son respectivamente:

- `if`
- `while`
- `cp`
- `echo`
- `head y tail`
- `cut.`
- `diff`
- `wc`
- `cat`

Capítulo 4

RESULTADOS EXPERIMENTALES

Para verificar los resultados se realizaron varias pruebas con la interfaz, cada una con la intención de comprobar y validar el correcto funcionamiento de todas las funciones previstas para el programa.

Todas las pruebas se realizaron en un servidor virtualizado con las siguientes características:

- Sistema operativo CentOS versión 7 con instalación mínima de componentes.
- Servicio para web Apache versión 2.4.6.
- Servicio para web PHP versión 5.4.16
- Interfaz de virtualización VirtualBox versión 5.1.18.
- La máquina virtual dispone de 1GB de memoria RAM y 20GB de memoria en disco duro.

La primera prueba está basada en comprobar el funcionamiento básico de la aplicación, la cual es que la misma sea capaz de conectar con el servidor para compilar los códigos escritos en el editor de texto, esta función es activada por el botón “Probar”. La comprobación se hace buscando en el servidor al archivo que resulta de la compilación del código en la ruta especificada para este archivo, en este caso el archivo se encuentra con el nombre “a.out” tal y como se muestra en la ilustración 22.

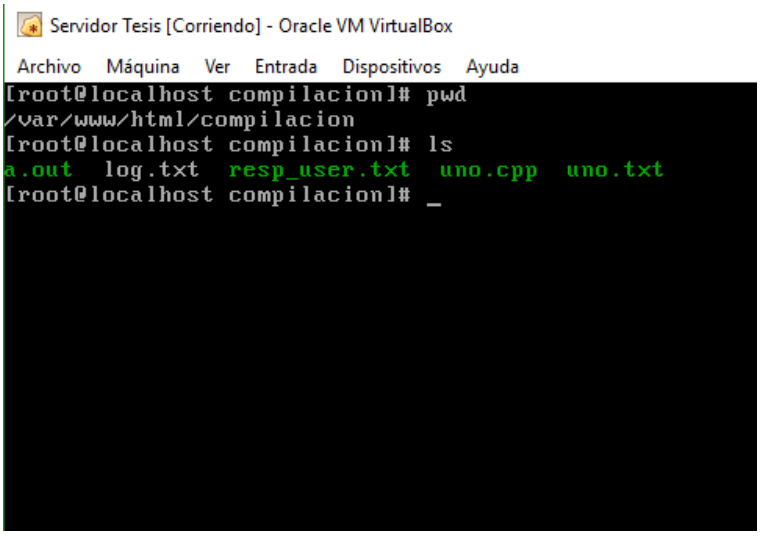


Ilustración 22 Directorio de compilación

Este es el archivo que posteriormente se envía para la comprobación de resultados.

También se tiene que comprobar que el archivo generado sea el resultado correspondiente al código, en la ilustración 22 se puede apreciar la existencia del archivo llamado “uno.cpp” el cual contiene el código que se encuentra en el editor al momento de mandar la instrucción para compilar. Todos los códigos de los ejercicios se guardarán en este mismo archivo por lo cual su contenido es variable y por lo tanto siempre dependerá del ejercicio en cuestión.

Anteriormente se mencionó que la lista de ejercicios se agrega a la interfaz de manera dinámica, esto se puede comprobar listando los ejercicios directamente en su directorio correspondiente, la cantidad de archivos que se encuentra en el directorio coincide a la perfección con los ejercicios listados en la interfaz, como se aprecia en las ilustraciones 23 y 24.

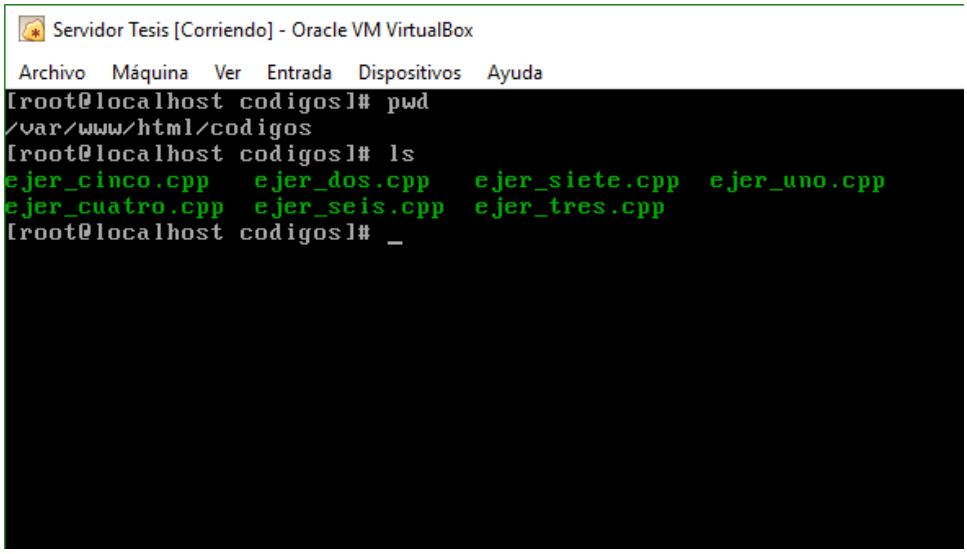


Ilustración 23 Directorio de ejercicios

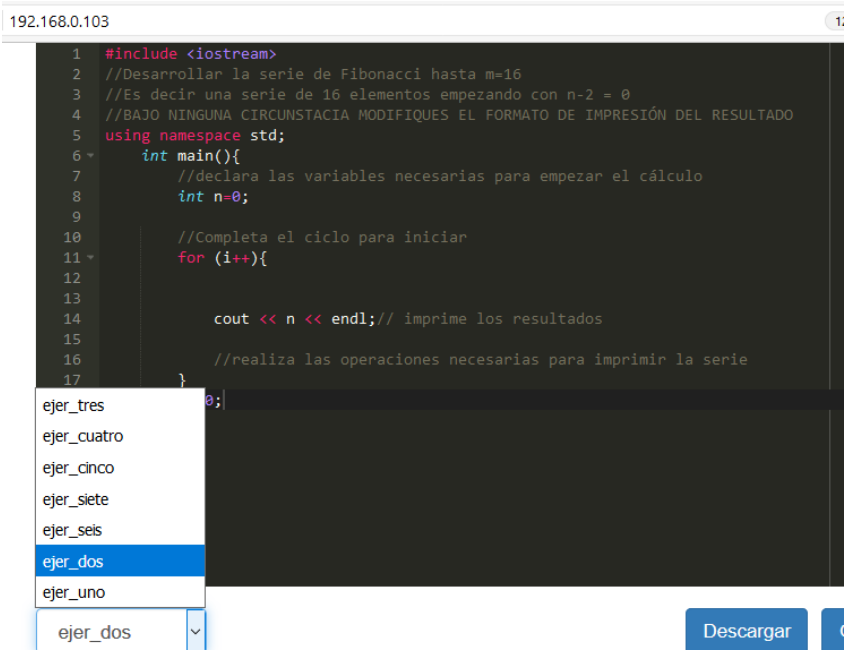


Ilustración 24 Lista de ejercicios

Si se procediera a revisar el contenido de todos los archivos que se encuentran en el directorio de los ejercicios, se corroboraría que dicho contenido corresponde al código que se carga con cada ejercicio en el editor de texto.

Cada ejercicio cuenta con una retroalimentación, la cual es desplegada en el área correspondiente dentro de la interfaz y al igual que el código que se carga al editor, la retroalimentación para cada ejercicio es creada en un archivo de texto en el directorio correspondiente. El programa está considerado para que cada ejercicio esté ligado a una retroalimentación.

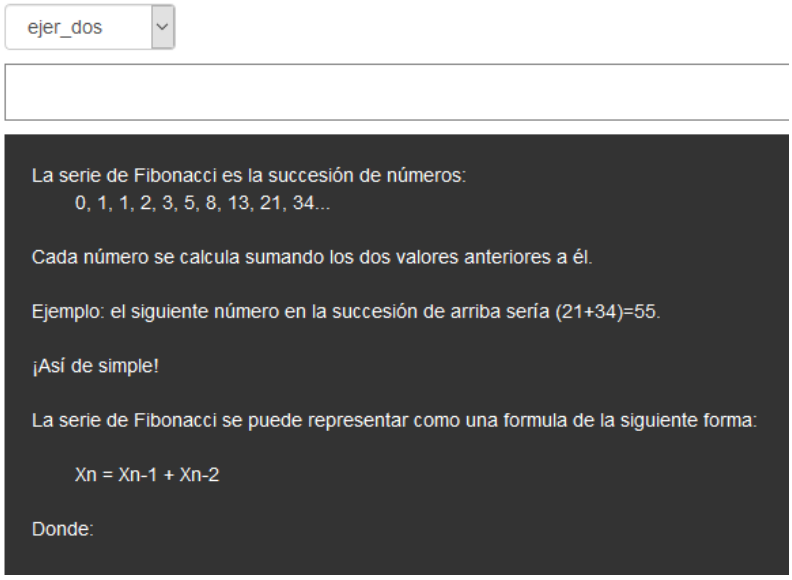


Ilustración 25 Retroalimentación del ejercicio No. 2

Todo lo explicado antes es transparente para el usuario, pero el resto de funciones del programa son completamente visibles cuando está resolviendo un problema. En las ilustraciones 24 y 25 se aprecia el ejercicio 2 con su correspondiente plantilla de código y su retroalimentación, cuando el usuario procede a resolver el ejercicio tiene que entrar en interacción con los botones de acción que se encuentran del lado derecho de bajo del editor de texto.

Para que el usuario guarde todo el progreso que lleve con el ejercicio, es necesario que accione esta función con el botón “Guardar Progreso”, esta acción solo guarda el progreso para el ejercicio que se encuentre editando y solo se guardará para esa sesión de navegación, es decir que para cuando el usuario abandone la página todo su progreso se guardará automáticamente. La función de guardar progreso solo está considerada para conservar los cambios en los ejercicios cuando el usuario decide resolver más de un ejercicio a la vez, es decir que el usuario podrá modificar libremente cualquier ejercicio y acceder a estas modificaciones mientras se encuentre dentro de la página, siempre y cuando guarde todo el progreso con cada modificación.

Cuando el usuario activa la opción de guardar progreso, un mensaje es desplegado en la pantalla informando de la acción, tal como se aprecia en la ilustración 26.

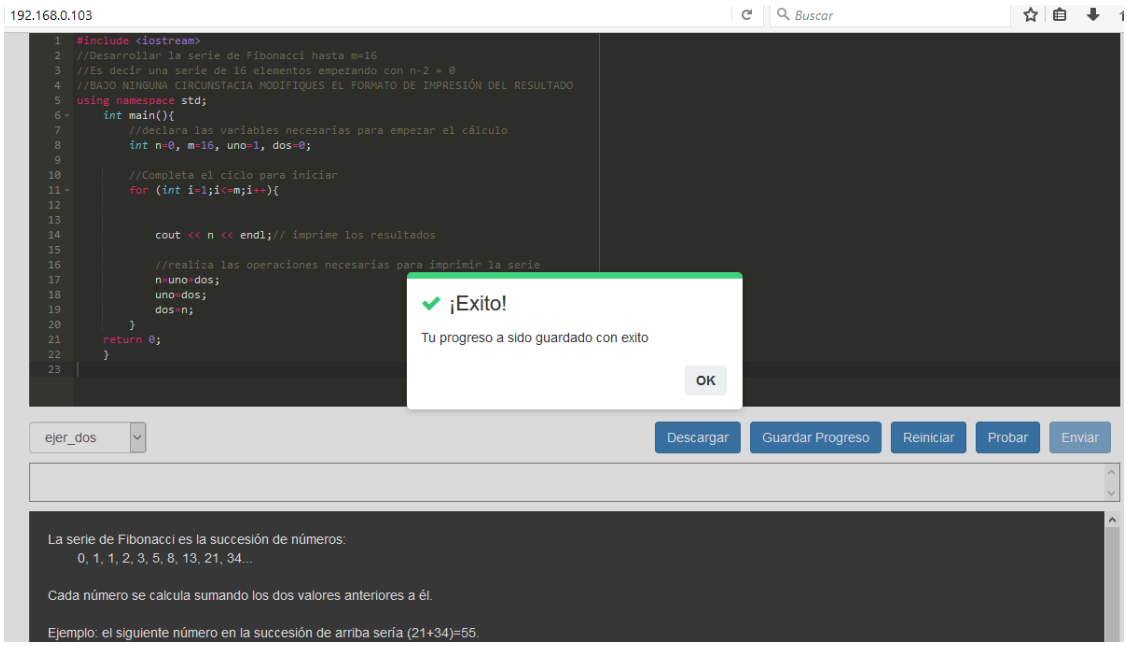


Ilustración 26 Mensaje de guardado exitoso

Otra de las funciones que el programa cumple, es el de brindar la posibilidad a los usuarios de descargar todo el código que tengan generado. Al activar la función de descarga, el programa pasa todo el código que se encuentre en el editor de texto a un archivo en el servidor y posteriormente desplegar la opción de descarga. El archivo se nombra con el ejercicio correspondiente de manera automática, un ejemplo es mostrado en la ilustración 27.

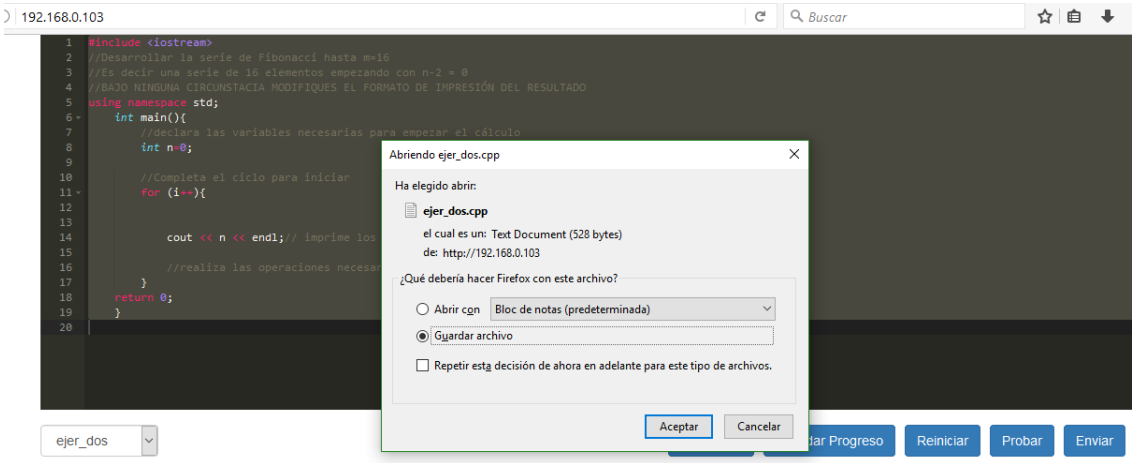


Ilustración 27 Descarga de código

Los usuarios también cuentan con la posibilidad de reiniciar los ejercicios al estado en el que se cargan inicialmente, esta opción la brinda el botón “Reiniciar”, realizar esta acción eliminará todo el

contenido modificado y borrara las variables del navegador existentes con progreso guardado para ese ejercicio, es decir que se empezará el ejercicio desde cero sin posibilidades de acceder a modificaciones pasadas, es por este motivo que antes de eliminar todos los datos se pide la autorización del usuario para continuar. La ilustración 27 muestra la alerta que se despliega al presionar sobre el botón reiniciar.

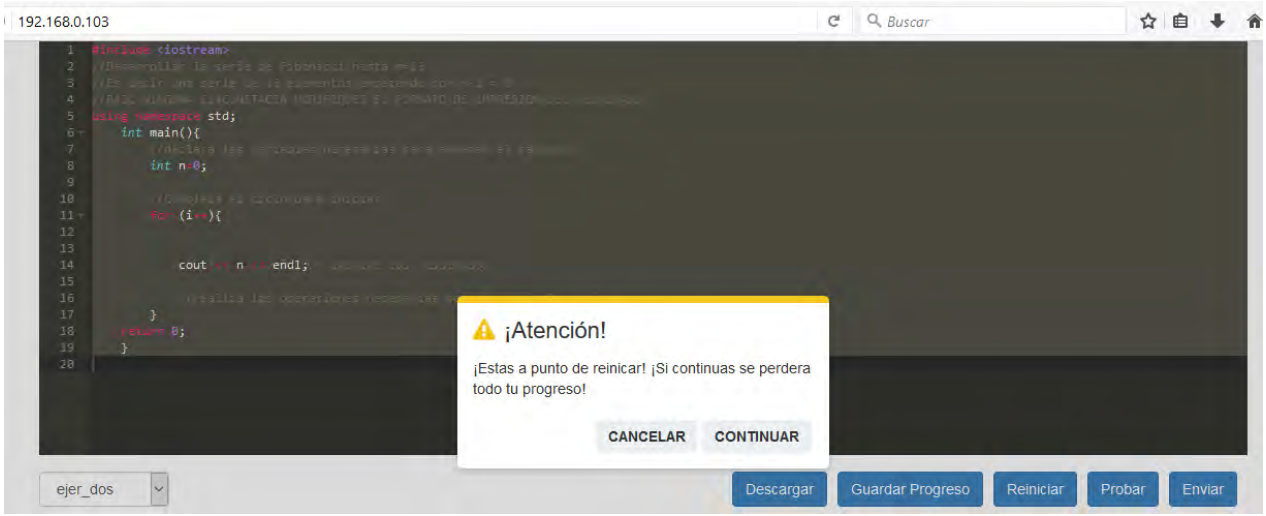


Ilustración 28 Alerta de autorización para reiniciar un ejercicio

Cuando se comprueba un ejercicio, el programa puede generar tres respuestas, tal y como se ha mencionado antes en este documento. La primera puede ser un **error en la compilación del código**, contempla errores comunes como errores de sintaxis o equivocaciones en los operadores y todos los errores que puedan ser detectados por el compilador, en este caso el programa informa del error y brinda al usuario la posibilidad de corregir dicho error mostrando el archivo generado por el compilador al ejecutarse.

Una segunda repuesta se da cuando la solución del ejercicio **corresponde a la respuesta esperada**, en este caso el programa simplemente imprime la respuesta en pantalla.

La tercera respuesta es más compleja que las anteriores, esta se trata de una **inconsistencia con el resultado**, es decir que las respuestas no coinciden y dado que el propósito del programa es orientar al estudiante a resolver los ejercicios de manera correcta sin brindarle la misma. La información que se despliega con esta respuesta tiene que ser solo la necesaria y dicha información

varia de cómo se presente el resultado del ejercicio. Hay dos posibles formas en las que se puede presentar el resultado de los ejercicios, en una la respuesta está dada en **una sola línea de impresión**, puede ser una cadena de caracteres o un número, esto depende enteramente de como el profesor construya el código, en la segunda forma la respuesta al ejercicio puede estar dada por **dos o más líneas de impresión**, por ejemplo cuando se trabaja con ciclos o vectores, de igual manera a la primera forma esto depende de cómo el profesor visualice el resultado. Cuando el resultado es presentado en la forma uno, la respuesta del programa a la comprobación consta de un mensaje informando que las respuestas no coinciden, el programa no imprime ningún resultado junto a esto para no brindar la respuesta al usuario. Cuando el resultado es presentado en la forma dos, el programa responde imprimiendo todas las líneas correctas que coinciden con el resultado y trunca la impresión donde encuentra el error, adicionalmente informa de la respuesta esperada para esa línea, pero sin dar información de las líneas siguientes.

La respuesta tres del programa (inconsistencia de resultado) también está preparada para considerar un error muy común cuando se está programando, esto es el generar ciclos infinitos. Detectar un ciclo infinito puede ser una tarea muy difícil para ser realizada de manera automática, es por eso por lo que el programa considera un tiempo de ejecución para los archivos generados por el compilador previniendo así el gasto de procesamiento en el servidor. Cuando esto sucede la respuesta que se brida, es parecida a la que se da con la segunda forma de presentar los resultados (dos o más líneas de impresión), imprimiendo todas las líneas de resultado hasta donde coincide con la respuesta correcta y truncando en el error, pero presenta una variante, esta se da cuando la solución al ejercicio está contenida entre las líneas que se generaron hasta llegar al límite de tiempo de ejecución, en este caso el programa imprime dos veces las líneas de la solución correcta del ejercicio e informa que la inconsistencia se encuentra en el excedente de líneas de resultado, pero que existe la posibilidad que la respuesta correcta al ejercicio se encuentre contenido entre estas líneas.

Capítulo 5

CONCLUSIONES

Durante el desarrollo de esta aplicación web pude vencer retos que no había tenido la oportunidad de enfrentar hasta ahora, un ejemplo de esto fue el trabajar con PHP, un lenguaje relativamente nuevo para mí, ya que este lenguaje se aborda de manera muy rápida y general en la materia de desarrollo web que se imparte como parte del plan de estudios de la carrera y para el lograr este proyecto tuve que aprender nuevos conceptos para poder adaptarme a cómo utilizarlo. En un principio no sabía si trabajar con PHP ya que no tenía muy clara la idea de cómo lograr el funcionamiento de la aplicación, pero gracias a la asistencia de mis profesores llegue a la decisión de que este lenguaje era la mejor opción para lograr las metas propuestas para la aplicación, de este modo y con un poco de esfuerzo fue descubriendo la potencia de este lenguaje de programación para diseñar aplicaciones de este tipo, en combinación con otras tecnologías.

La idea de que la interfaz de la aplicación se vía web, facilito un poco el trabajo debido a que con anterioridad había tenido la oportunidad de trabajar en el diseño de páginas web, no de manera profesional, pero si en la práctica. Esto no demerita el proyecto ya que durante el camino hubo tropiezos y dificultades para lograr la funcionalidad que se tenía planteada para el programa.

Desde un principio la aplicación contempló brindar a los alumnos la posibilidad de trabajar con ejercicios de programación desde su hogar y a la vez brindarles a los profesores la flexibilidad de calificar estos ejercicios de una manera muchas más rápida y eficiente. Al principio del documento se mencionó la existencia de otras aplicaciones web de este tipo, tal es el caso de la plataforma Moodle que cuenta con un plugin llamado **CodeRunner** el cual funciona casi de la misma manera que la aplicación aquí desarrollada. La principal diferencia entre ambos radica en la forma en la que se aborda el tema de la seguridad, en el caso de la aplicación desarrollada por un servidor utilicé lo que algunos desarrolladores denominan ***ApacheJail*** que básicamente trata de controlar los directorios en donde apache podrá trabajar y realizar modificaciones desde la página web como si se tratara de una prisión, todo esto con la ayuda de la llamada al sistema ***chroot***, esta configuración se hace desde el mismo servidor donde se encuentre alojada la aplicación web, mientras que **CodeRunner** utiliza ***Sandbox***, que realiza una función similar a **ApacheJail**, con la diferencia que este normalmente confina los procesos de la aplicación en un servidor separado que utiliza como entorno de pruebas. Es posible que esta sea una mejor opción para una aplicación de este tipo,

aunque en su momento no se consideró, pero la alternativa de ***chroot*** cumple perfectamente con lo deseado de igual manera.

Con el desarrollo de esta aplicación, considero que se cumplió el objetivo principal planteado para este proyecto, el cual era desarrollar una aplicación web capaz de validar los códigos de programación de los alumnos. Sin duda alguna si decidieran utilizar esta aplicación en un futuro, esta cumpliría con las expectativas, tanto de los alumnos para completar los ejercicios que el profesor brinde, como para los profesores para facilitar la validación de los ejercicios para calificar, ya que la interfaz y las funciones de la aplicación fueron diseñadas para cumplir con el resto de objetivos planteados y así facilitar el uso para los alumnos.

Del mismo modo quiero hacer especial énfasis en la justificación del proyecto y el resultado de este, ya que la aplicación aquí planteada demuestra el uso de tecnologías nuevas de desarrollo web como lo son JQuery y Bootstrap que actualmente es utilizado por la gran mayoría de desarrolladores en el diseño de sus sitios web, tanto por la flexibilidad como por la presentación que se logra con estas tecnologías. Sin duda alguna, espero que este trabajo le sea de ayuda a los alumnos que en un futuro deseen trabajar con aplicaciones web de este estilo o páginas web en general.

El trabajo futuro para este proyecto es el de poder vincular la aplicación con una plataforma que registre y lleve a cabo el seguimiento de las calificaciones de los ejercicios como lo es la plataforma Moodle, ya que el alcance de este proyecto no considera esta opción y por lo tanto el envío de ejercicios no está disponible actualmente; también sería factible desarrollar un entorno para la generación de ejercicios desde un ambiente gráfico.

.

REFERENCIAS BIBLIOGRÁFICAS

[1] Mozilla Corporation, «Mozilla Developer Network | Mozilla Docs HTML,» [En línea]. Available: https://developer.mozilla.org/es/docs/Web/HTML. [Último acceso: 25 Abril 2017].

[2] Mozilla Corporation, «Mozilla Developer Network | Mozilla Docs CSS,» [En línea]. Available: https://developer.mozilla.org/es/docs/Web/CSS. [Último acceso: 28 Abril 2017].

[3] W3School, «W3School | Bootstrap Grid,» [En línea]. Available: https://www.w3schools.com/bootstrap/bootstrap_grid_basic.asp. [Último acceso: 28 Abril 2017].

[4] Mozilla Corporation, «Mozilla Developer Network | Mozilla Docs JavaScript,» [En línea]. Available: https://developer.mozilla.org/es/docs/Web/JavaScript/Acerca_de_JavaScript. [Último acceso: 28 Abril 2017].

[5] L. Web, «Libros Web | Modelo básico de eventos,» [En línea]. Available: http://librosweb.es/libro/javascript/capitulo_6/modelo_basico_de_eventos_2.html. [Último acceso: 29 Abril 2017].

[6] Mozilla Corporation, «Mozilla Developer Network | Mozilla Docs DOM,» [En línea]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction. [Último acceso: 28 Abril 2017].

[7] JQuery.com, «JQuery.com | What´s JQuery?,» [En línea]. Available: https://jquery.com/. [Último acceso: 29 Abril 2017].

[8] Mozilla Corporation, «Mozilla Developer Network | Mozilla Docs AJAX,» [En línea]. Available: https://developer.mozilla.org/es/docs/AJAX. [Último acceso: 30 Abril 2017].

[9] C. R. Think, «Arquitectura de Cliente-Servidor con AJAX y JSON,» [En línea]. Available: http://thinkcoderepeat.blogspot.mx/2006/08/arquitectura-cliente-servidor-con-ajax.html. [Último acceso: 30 Abril 2017].

[10] PHP, «php.net,» [En línea]. Available: http://php.net/manual/es/intro-whatis.php. [Último acceso: 30 Abril 2017].

[11] PHP, «php.net,» [En línea]. Available: http://php.net/manual/es/intro-whatcando.php. [Último acceso: 1 Mayo 2017].

[12] U. F. Gavida, «UFG,» [En línea]. Available: http://ri.ufg.edu.sv/jspui/bitstream/11592/7147/3/005.74-A594d-Capitulo%20II.pdf. [Último acceso: 2017 Mayo 2017].

[13] PHP, «php.net | Funciones de ejecucución de programas,» [En línea]. Available: http://php.net/manual/es/ref.exec.php. [Último acceso: 1 Mayo 2017].

[14] Ace, «Ace | The high performance code editor for the web,» [En línea]. Available: https://ace.c9.io/. [Último acceso: 10 Mayo 2017].

[15] Ace, «Ace | Real World Users,» [En línea]. Available: https://ace.c9.io/#nav=production. [Último acceso: 10 Mayo 2017].

[16] I. F. Archives, «Securing and Optimizing Linux: RedHat Edition -A Hands on Guide | Chapter 29. Software -Network Server, web/Apache,» [En línea]. Available: http://www.faqs.org/docs/securing/chap29sec254.html. [Último acceso: 12 Mayo 2017].

Índice de Ilustraciones

Índice de Tablas