

Universidad de Costa Rica

Escuela de Ciencias de la Computación e
Informática

Desarrollo de Aplicaciones para Internet - CI2413

Tarea programada 1: Servidor web miniatura

Profesor: Braulio Solano Rojas.

Estudiantes:

Esteban Ortega Acuña. B35033.

Jose Pablo Vargas Cordero. B37275.

2019

Introducción	3
Descripción del Problema	3
Descripción de la Metodología	3
Análisis del Problema	4
Diseño de Clases (UML)	5
Casos de Prueba y Resultados	5
Análisis de los Resultados de las Pruebas	9

Introducción

Descripción del Problema

El problema consiste en la realización de un servidor web con el fin de entender el funcionamiento del protocolo HTTP. Este servidor deberá ser capaz de soportar los siguientes métodos:

- GET
- HEAD
- POST

Además, el servidor deberá apoyar los siguientes encabezados:

- Accept
- Content-type
- Content-length
- Date
- Host
- Referer
- Server

También el servidor debe ser capaz de responder con los códigos de retorno: 200, 404, 406 501.

El servidor debe ser programado con sockets y no debe utilizarse ninguna biblioteca HTTP. El servidor debe ser programado de manera concurrente con la intención de mejorar su eficiencia. Además, el servidor debe manejar una lista de mime types para dar el manejo correcto a los diferentes tipos de archivos que puedan ser solicitados.

Descripción de la Metodología

Para resolver el problema fue necesario hacer pequeñas investigaciones sobre diferentes aspectos referentes a servidores web. Temas como la forma en que se procesan los archivos de diferente formato en un servidor para entender el uso de mime types o la utilidad de los diferentes atributos de un encabezado HTTP. También fue necesario investigar sobre los diferentes tipos de métodos HTTP soportados por el servidor.

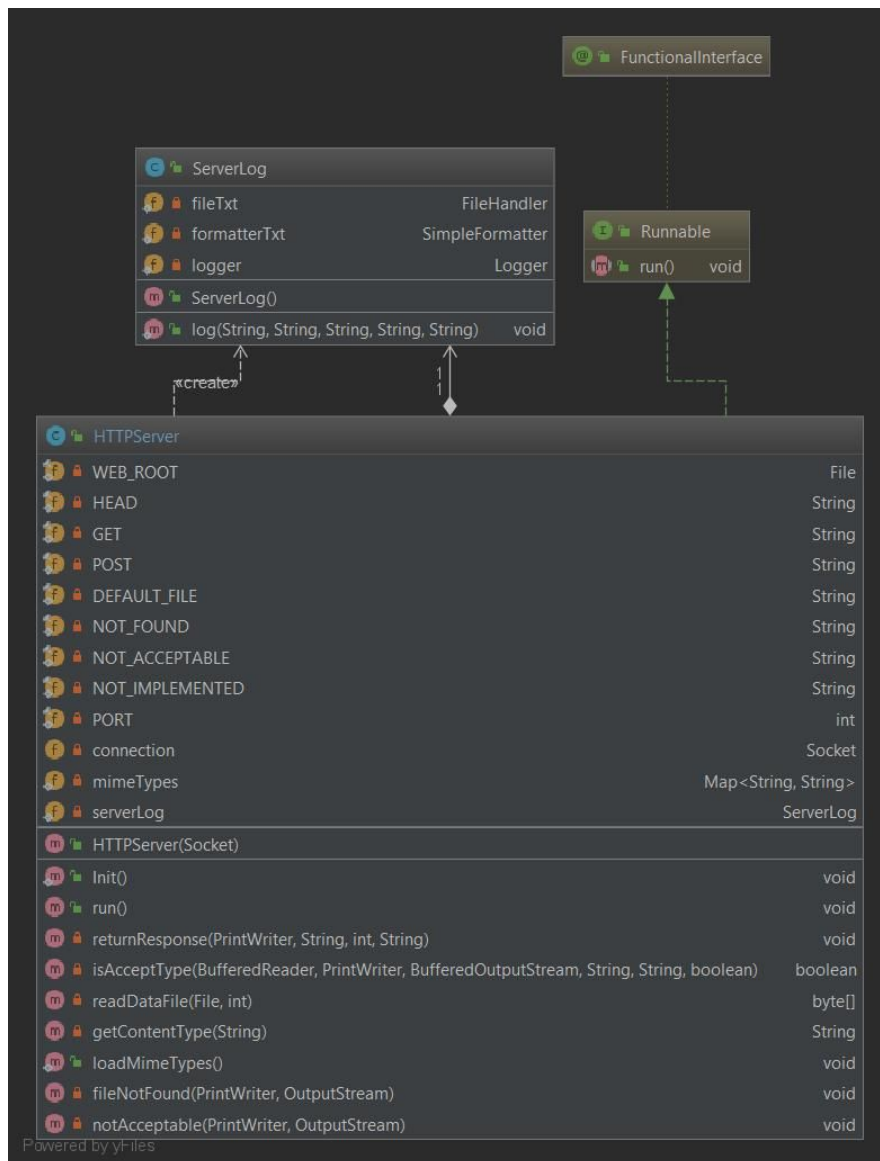
Una vez encontrada la información necesaria el siguiente paso fue seleccionar el lenguaje de programación para la resolución del problema. En este caso se seleccionó Java como la herramienta de programación, ya que es un lenguaje que ofrece independencia del sistema operativo, además de ser un lenguaje con el que los

integrantes del grupo están familiarizados y que ofrece herramientas para trabajar con sockets.

Análisis del Problema

Para la resolución del problema se decidió hacer una división del mismo en temas o sub problemas con la intención de resolver cada uno de estos de manera individual y de esta forma avanzar hacia la solución completa. Se decidió hacer la siguiente división para la solución: implementación de los métodos HTTP (GET, HEAD, POST), procesamiento de los atributos de los encabezados HTTP, manejo de los diferentes códigos de retorno y procesamiento de los mime types para el manejo de los diferentes formatos de archivos solicitados al servidor.

Diseño de Clases (UML)



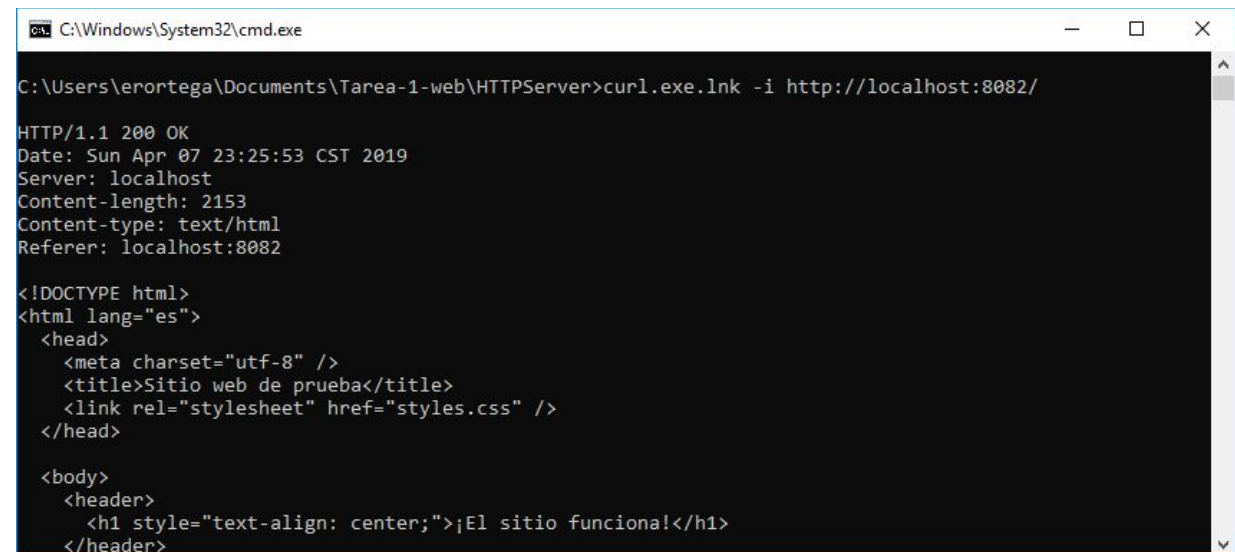
Casos de Prueba y Resultados

Para verificar el funcionamiento correcto del servidor se llevaron a cabo las siguientes pruebas:

Método	Servidor	Refiere	Url	Datos
--------	----------	---------	-----	-------

GET	localhost:8082	localhost:8082	http://localhost:8082/	
HEAD	localhost:8082	localhost:8082	http://localhost:8082/index.html	
HEAD	localhost:8082	localhost:8082	http://localhost:8082/index.html	
POST	localhost:8082	localhost:8082	http://localhost:8082/index.html	mensaje=Hola +Mundo
GET	localhost:8082	localhost:8082	http://localhost:8082/nofile.html	
PUT	localhost:8082	localhost:8082	http://localhost:8082/index.html	mensaje=Hola +Mundo

```
curl -i http://localhost:8082/
```



```

C:\Windows\System32\cmd.exe

C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>curl.exe -i http://localhost:8082/

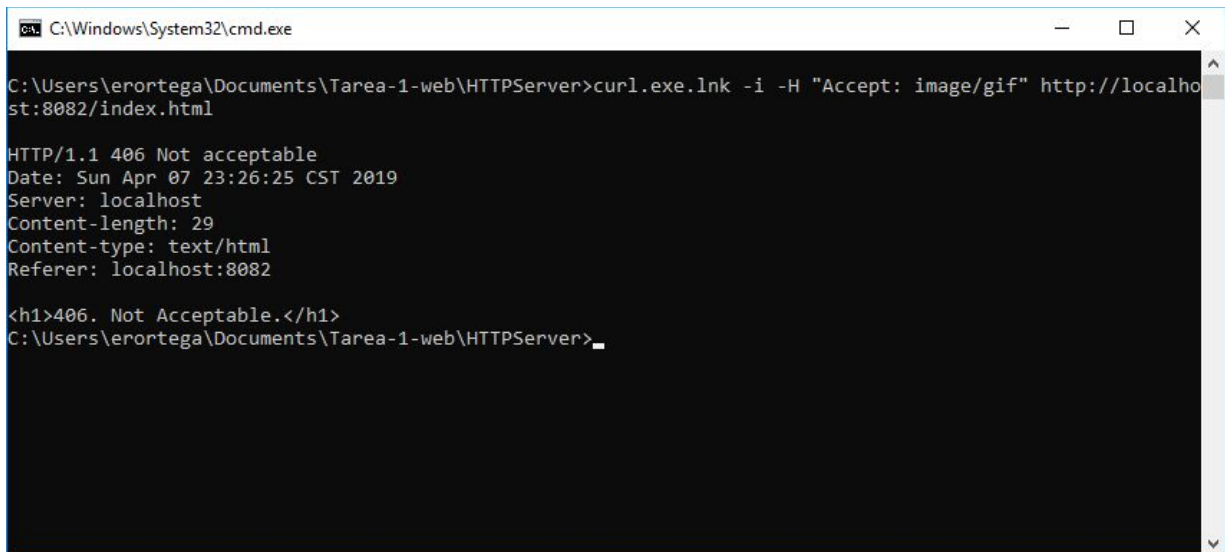
HTTP/1.1 200 OK
Date: Sun Apr 07 23:25:53 CST 2019
Server: localhost
Content-length: 2153
Content-type: text/html
Referer: localhost:8082

<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="utf-8" />
    <title>Sitio web de prueba</title>
    <link rel="stylesheet" href="styles.css" />
  </head>

  <body>
    <header>
      <h1 style="text-align: center;">¡El sitio funciona!</h1>
    </header>
  </body>
</html>

```

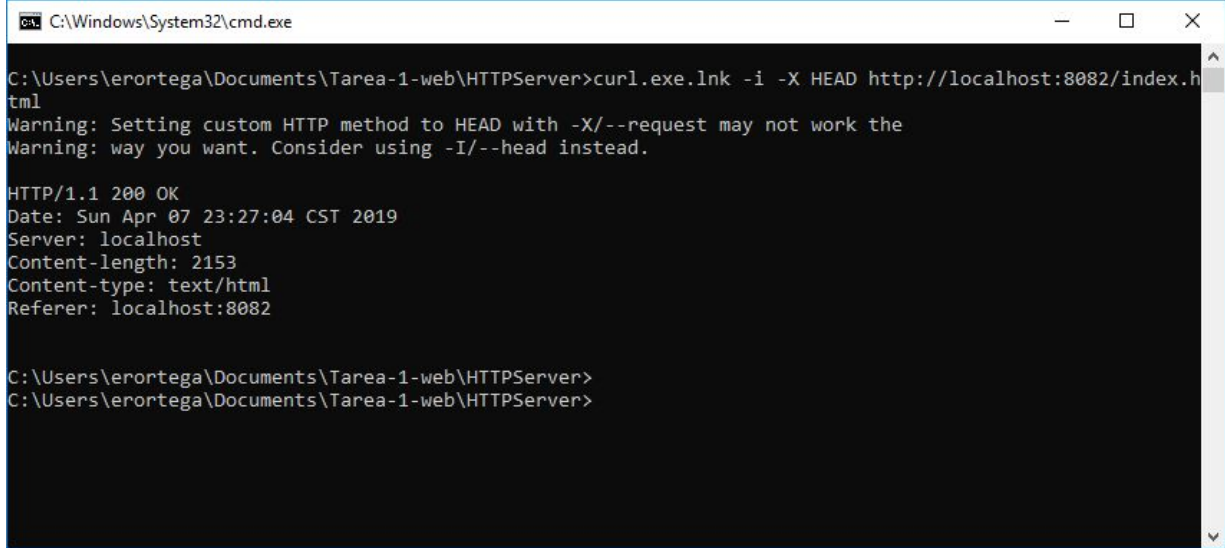
```
curl.exe.lnk -i -H "Accept: image/gif"  
http://localhost:8082/index.html
```



A screenshot of a Windows command prompt window titled "C:\Windows\System32\cmd.exe". The window shows the execution of a curl command from the directory "C:\Users\erortega\Documents\Tarea-1-web\HTTPServer". The command is "curl.exe.lnk -i -H 'Accept: image/gif' http://localhost:8082/index.html". The output shows an HTTP 406 Not Acceptable response with headers: Date: Sun Apr 07 23:26:25 CST 2019, Server: localhost, Content-length: 29, Content-type: text/html, and Referer: localhost:8082. The body of the response is "<h1>406. Not Acceptable.</h1>".

```
C:\Windows\System32\cmd.exe  
C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>curl.exe.lnk -i -H "Accept: image/gif" http://localhost:8082/index.html  
HTTP/1.1 406 Not acceptable  
Date: Sun Apr 07 23:26:25 CST 2019  
Server: localhost  
Content-length: 29  
Content-type: text/html  
Referer: localhost:8082  
  
<h1>406. Not Acceptable.</h1>  
C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>
```

```
curl.exe.lnk -i -X HEAD http://localhost:8082/index.html
```

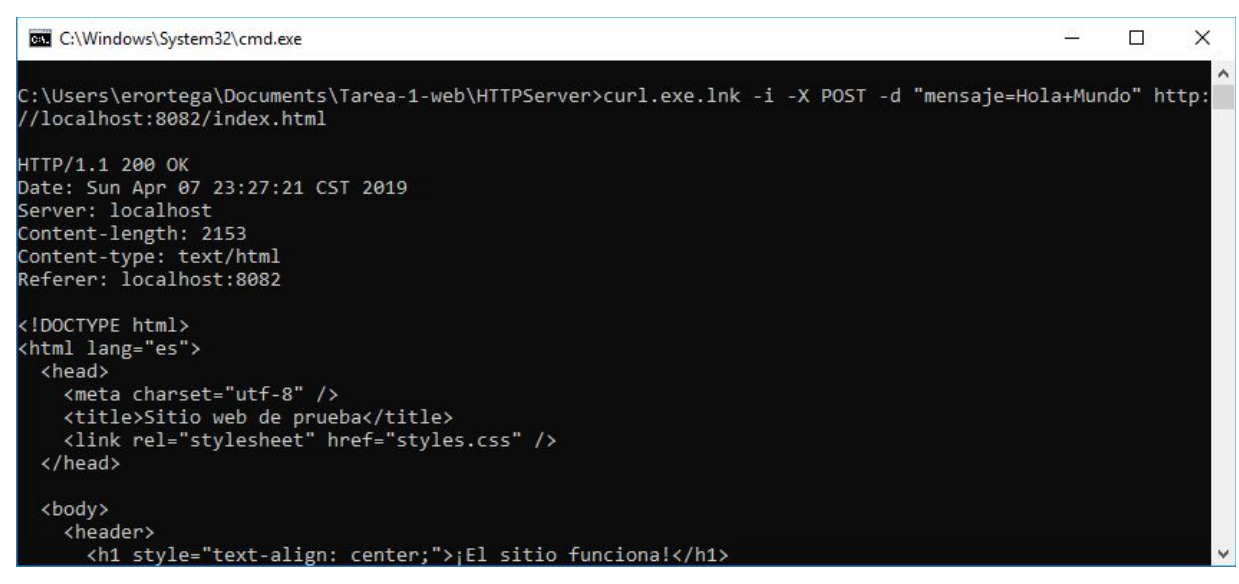


A screenshot of a Windows command prompt window titled "C:\Windows\System32\cmd.exe". The window shows the execution of a curl command from the directory "C:\Users\erortega\Documents\Tarea-1-web\HTTPServer". The command is "curl.exe.lnk -i -X HEAD http://localhost:8082/index.html". The output shows a warning about the custom HTTP method, followed by an HTTP 200 OK response with headers: Date: Sun Apr 07 23:27:04 CST 2019, Server: localhost, Content-length: 2153, Content-type: text/html, and Referer: localhost:8082.

```
C:\Windows\System32\cmd.exe  
C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>curl.exe.lnk -i -X HEAD http://localhost:8082/index.html  
Warning: Setting custom HTTP method to HEAD with -X/--request may not work the way you want. Consider using -I/--head instead.  
HTTP/1.1 200 OK  
Date: Sun Apr 07 23:27:04 CST 2019  
Server: localhost  
Content-length: 2153  
Content-type: text/html  
Referer: localhost:8082  
  
C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>  
C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>
```

```
curl.exe.lnk -i -X POST -d "mensaje=Hola+Mundo"
```

http://localhost:8082/index.html



```
C:\Windows\System32\cmd.exe

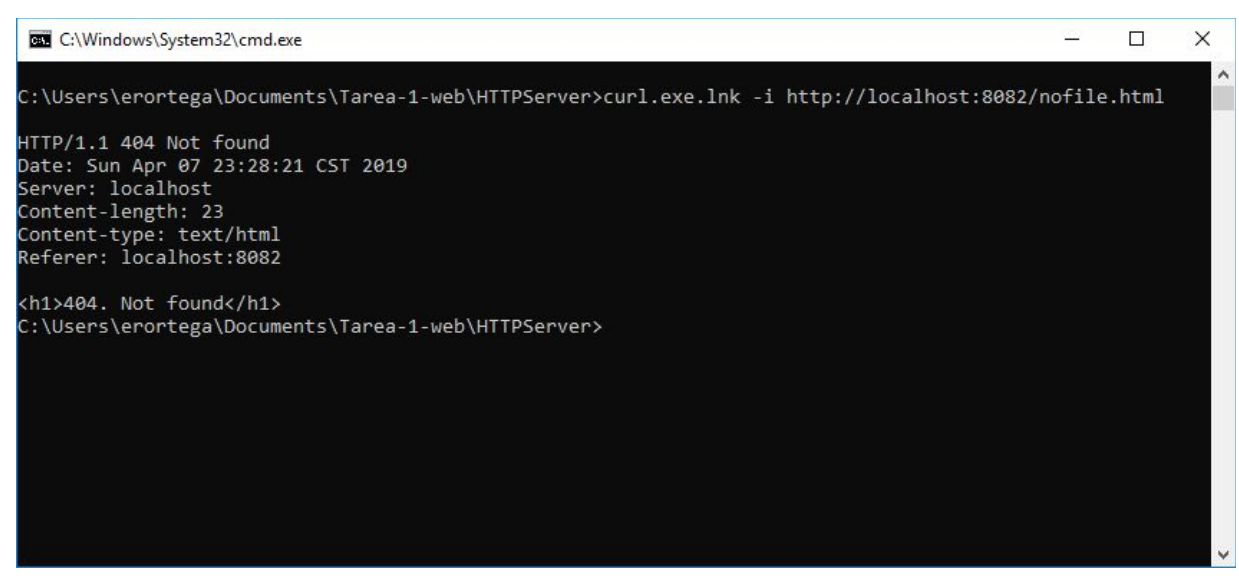
C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>curl.exe.lnk -i -X POST -d "mensaje=Hola+Mundo" http://localhost:8082/index.html

HTTP/1.1 200 OK
Date: Sun Apr 07 23:27:21 CST 2019
Server: localhost
Content-length: 2153
Content-type: text/html
Referer: localhost:8082

<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="utf-8" />
    <title>Sitio web de prueba</title>
    <link rel="stylesheet" href="styles.css" />
  </head>

  <body>
    <header>
      <h1 style="text-align: center;">¡El sitio funciona!</h1>
```

curl.exe.lnk -i http://localhost:8082/nofile.html



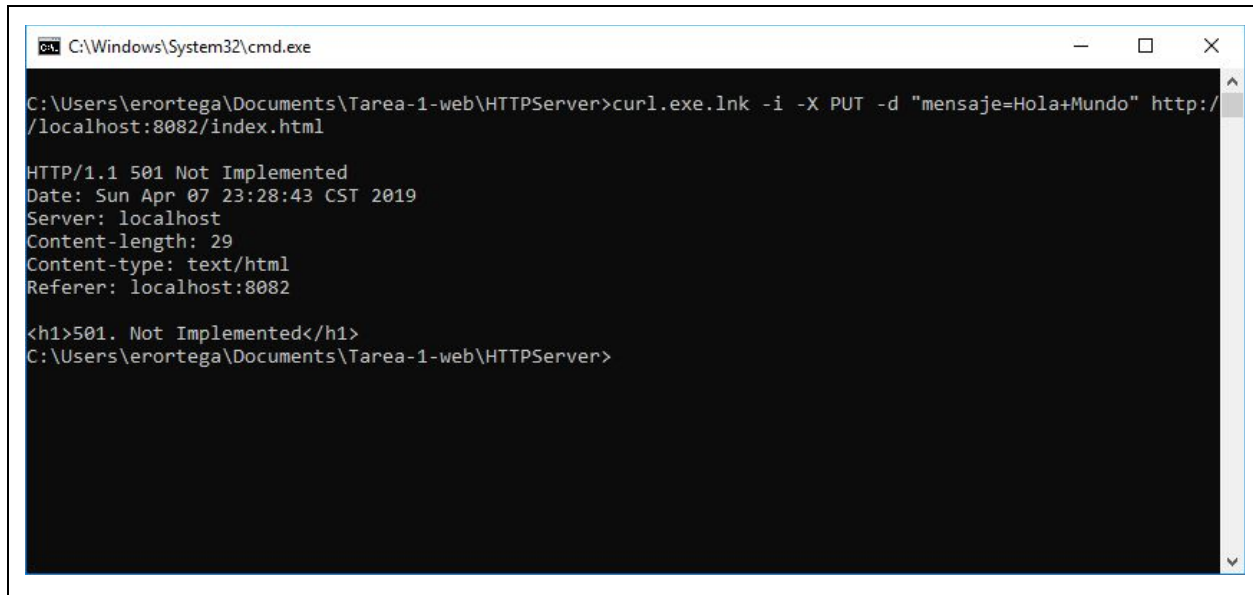
```
C:\Windows\System32\cmd.exe

C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>curl.exe.lnk -i http://localhost:8082/nofile.html

HTTP/1.1 404 Not found
Date: Sun Apr 07 23:28:21 CST 2019
Server: localhost
Content-length: 23
Content-type: text/html
Referer: localhost:8082

<h1>404. Not found</h1>
C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>
```

curl.exe.lnk -i -X PUT -d "mensaje=Hola+Mundo"
http://localhost:8082/index.html



```
C:\Windows\System32\cmd.exe

C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>curl.exe,lnk -i -X PUT -d "mensaje=Hola+Mundo" http://
localhost:8082/index.html

HTTP/1.1 501 Not Implemented
Date: Sun Apr 07 23:28:43 CST 2019
Server: localhost
Content-length: 29
Content-type: text/html
Referer: localhost:8082

<h1>501. Not Implemented</h1>
C:\Users\erortega\Documents\Tarea-1-web\HTTPServer>
```

Análisis de los Resultados de las Pruebas

Los casos de prueba mostrados anteriormente permiten verificar el funcionamiento del servidor. En los casos de prueba hechos con cURL es posible confirmar que el servidor responde correctamente a solicitudes a través de los métodos GET, HEAD y POST y que si se realiza una solicitud mediante otro método el servidor retorna el error 501, que indica que el método solicitado no ha sido implementado en ese servidor. Así como se manejan los errores de métodos no implementados en los casos de prueba también se observa que el servidor es capaz de retornar los demás códigos HTTP; 200 cuando el recurso existe y el tipo de media es soportado, 404 cuando el recurso no existe y 406 cuando aunque el recurso exista se solicita un tipo de media no soportado.

Los casos de prueba también fueron útiles durante el proceso de resolución del problema, ya que con estas pruebas fue posible hacer revisiones del programa y al recibir resultados incorrectos se lograron corregir a tiempo, esto debido a que los casos de prueba fueron pensados para probar el programa de diferentes maneras con el fin de encontrar distintos errores.