Introduction  
This exercise entails the implementation of RFC 2616 in C++11 utilizing sockets. The web-server executes asynchronous tasks using C++ selectors.

Testing  
Postman was employed for conducting tests, with each request accompanied by a corresponding test within the collection provided alongside this submission.

Code  
The primary objective was to eliminate strings. Consequently, upon receiving each message, it is promptly parsed into a self-implemented *HttpRequest* object.  
Subsequently, the request is processed, generating an *HttpResponse* object. Prior to transmission via the socket, the *HttpResponse* object undergoes parsing into a valid HTTP response string.

The object-oriented perspective of this implementation involves several key classes designed to facilitate its functionality:

1. **Uri**: This class defines a URI, featuring a single string property named path.
2. **SingletonHtmlPlaceholder**: Implementing the Singleton design pattern, this class ensures the creation of a single instance of a string.
3. **HttpMessage**:

• Enumerations for *HttpVersion*, *HttpMethod*, *HttpStatusCode*, *QLanguage* (representing the "lang" query parameter), and utility functions for converting strings to enums/class instances and vice versa.

• **QueryParams**: A container for a map<string, string> of query parameters.

• **HttpMessageInterface**: Comprising class members such as content, headers and version. The interface automatically updates the *Content-Length* header when content is set, initially set to 0.

• **HttpRequest**: Implementing the *HttpMessageInterface*, this class also includes class members such as method, params, and URI.

• **HttpResponse**: Also implementing the *HttpMessageInterface*, this class adds the status code class member.

1. **HttpServer**:

* Defined types for a generic HTTP request handler: *HttpRequestHandler\_t = std::function<HttpResponse(const HttpRequest&)>*.
* A collection is established to store all server request handlers: *map<Uri, map<HttpMethod, HttpRequestHandler\_t>>*, facilitating the *HandleHttpRequest* function's determination of when to return *NotFound* or *MethodNotAllowed*

1. **SocketService**: This class encompasses all functions relating to the sockets utilized by the *HttpServer*.

API  
The web-server provides the following endpoints:

1. **GET /health:**Query parameters: None  
   Response: HTTP Response with a 200 OK status code and text/plain content, containing the string "OK"
2. **HEAD /health:**Query parameters: None  
   Response: Similar to **GET /health**, but without content
3. **TRACE /health:**Query parameters: None  
   Response: HTTP Response with a 200 OK status code and message/http content, reflecting the sent request
4. **OPTIONS /health:**Query parameters: None  
   Response: HTTP Response with a 204 No Content status code. The *Allow* header lists the available REST methods for the **/health** route, which are *OPTIONS*, *TRACE*, *HEAD*, and *GET*
5. **GET /index.html:**  
   Query parameters: An **optional** parameter "lang", which can contain one of the strings "he", "en", or "fr"  
   Response: HTTP Response with a 200 OK status code. If no query parameters are provided or an invalid string other than "en", "he", or "fr" is supplied, the server defaults to returning the English version of the HTML. Otherwise, it returns the HTML corresponding to the specified language
6. **HEAD /index.html:**  
   Query parameters: An **optional** parameter "lang", which can contain one of the strings "he", "en", or "fr"  
   Response: Similar to **GET /index.html**, but without content
7. **TRACE /index.html:**  
   Query parameters: None  
   Response: HTTP Response with a 200 OK status code and message/http content, reflecting the sent request
8. **POST /index.html:**  
   Query parameters: None  
   Payload: a plain text  
   Response: HTTP Response with a 200 OK status code
9. **PUT /index.html:**  
   Query parameters: None  
   Payload: a plain text  
   Response: HTTP Response with a 200 OK status code and text/plain content, containing the string "OK".
10. **DELTE /index.html:**  
    Query parameters: None   
    Response: HTTP Response with a 200 OK status code and text/plain content, containing the string "OK".

Wireshark screenshots  
A consolidated .pcap file comprising all the requests has been included in the MAMA submission as well.  
  
  
