Web Terminology

1.. Introduction

A web application is an application accessible from the web. A web application is composed of web components like Servlet, JSP, Filter, etc. and other elements such as HTML, CSS, and JavaScript. The web components typically execute in Web Server and respond to the HTTP request.

2.. Website

Website is a collection of related web pages that may contain text, images, audio and video. The first page of a website is called home page. Each website has specific internet address (URL) that you need to enter in your browser to access a website. Website is hosted on one or more servers and can be accessed by visiting its homepage using a computer network. A website is managed by its owner that can be an individual, company or an organization.

A Website can be of two types:

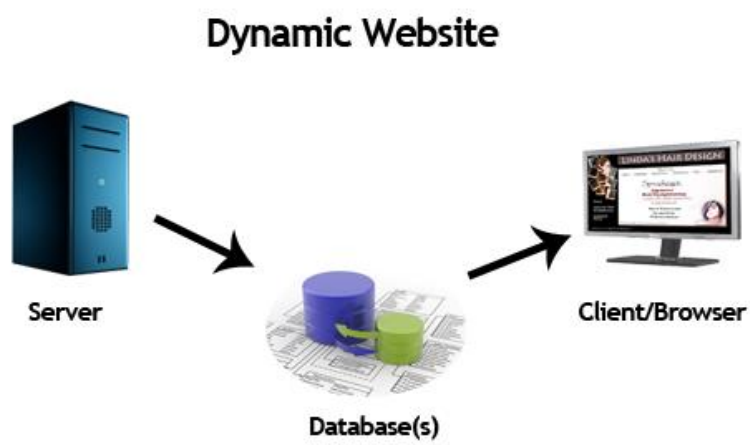
- Static Website
- Dynamic Website

2.1 Static Website

Static website is the basic type of website that is easy to create. You don't need the knowledge of web programming and database design to create a static website. Its web pages are coded in HTML. The codes are fixed for each page so the information contained in the page does not change and it looks like a printed page.

2.2 Dynamic Website

Dynamic website is a collection of dynamic web pages whose content changes dynamically. It accesses content from a database or Content Management System (CMS). Therefore, when you alter or update the content of the database, the content of the website is also altered or updated. Dynamic website uses client-side scripting or server-side scripting, or both to generate dynamic content. Client-side scripting generates content at the client computer on the basis of user input. The web browser downloads the web page from the server and processes the code within the page to render information to the user. In server-side scripting, the software runs on the server and processing is completed in the server then plain pages are sent to the user.



Static Website	Dynamic Website
Prebuilt content is same every time the page is loaded.	Content is generated quickly and changes regularly.
It uses the **HTML** code for developing a website.	It uses the server side languages such as **PHP,SERVLET, JSP, and ASP.NET** etc. for developing a website.
It sends exactly the same response for every request.	It may generate different HTML for each of the request.
The content is only changed when someone publishes and updates the file (sends it to the web server).	The page contains "server-side" code which allows the server to generate the unique content when the page is loaded.
Flexibility is the main advantage of static website.	Content Management System (CMS) is the main advantage of dynamic website.

3.. Rest API

REST stands for REpresentational State Transfer. REST is web standards-based architecture and uses HTTP Protocol. It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods. In REST architecture, a REST Server simply provides access to resources and REST client accesses and modifies the resources. Here each resource is identified by URIs/ global IDs. REST uses various representation to represent a resource like text, JSON, XML. JSON is the most popular one.

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.

Web services based on REST Architecture are known as RESTful web services. These webservices uses HTTP protocol and its methods to implement the concept of REST architecture.

The ways of Writing API Endpoints

POST - /api/v1/users

GET - /api/v1/users/{userId}

PUT - /api/v1/users/{userId}

PATCH - /api/v1/users/{userId}

DELETE - /api/v1/users/{userId}

Suppose I want to retrieve all the Posts done by a Specific User, then Valid End-point will be:

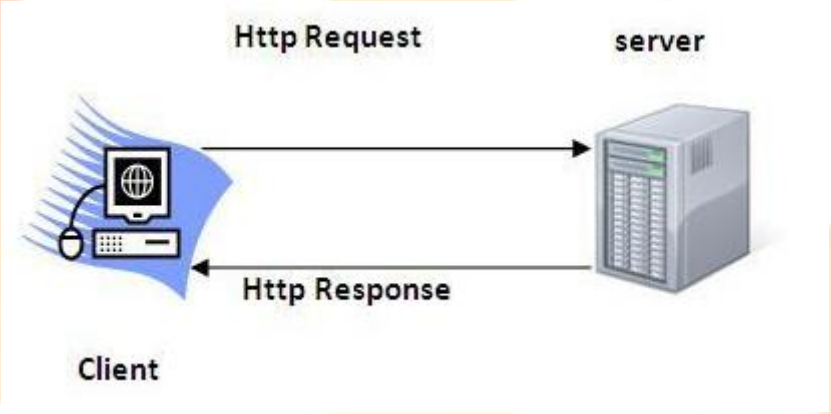
GET - /api/v1/users/{userId}/posts

Suppose I want to retrieve all the Posts then valid end-point will be:

GET - /api/v1/posts

4.. HTTP

The Hypertext Transfer Protocol (HTTP) is application-level protocol for collaborative, distributed, hypermedia information systems. It is the data communication protocol used to establish communication between client and server. HTTP is TCP/IP based communication protocol, which is used to deliver the data like image files, query results, HTML files etc on the World Wide Web (WWW) with the default port is TCP 80. It provides the standardized way for computers to communicate with each other.



The Basic Characteristics of HTTP (Hyper Text Transfer Protocol):

- It is the protocol that allows web servers and browsers to exchange data over the web.
- It is a request response protocol.
- It uses the reliable TCP connections by default on TCP port 80.
- It is stateless means each request is considered as the new request. In other words, server doesn't recognize the user by default.

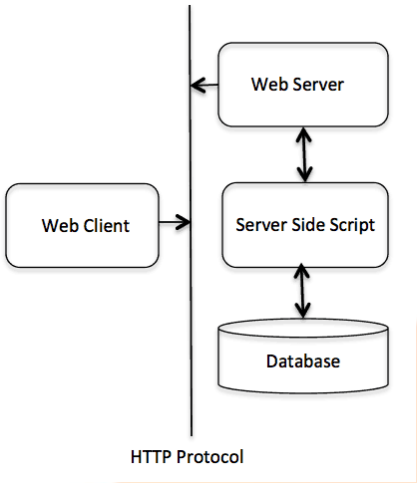
The Basic Features of HTTP (Hyper Text Transfer Protocol):

There are three fundamental features that make the HTTP a simple and powerful protocol used for communication:

- **HTTP is media independent**: It specifies that any type of media content can be sent by HTTP as long as both the server and the client can handle the data content.
- **HTTP is connectionless**: It is a connectionless approach in which HTTP client i.e., a browser initiates the HTTP request and after the request is sent the client disconnects from server and waits for the response.
- **HTTP is stateless**: The client and server are aware of each other during a current request only. Afterwards, both of them forget each other. Due to the stateless nature of protocol, neither the client nor the server can retain the information about different request across the web pages.

The Basic Architecture of HTTP (Hyper Text Transfer Protocol):

The below diagram represents the basic architecture of web application and depicts where HTTP stands:



HTTP is request/response protocol which is based on client/server-based architecture. In this protocol, web browser, search engines, etc. behave as HTTP clients and the Web server like Servlet behaves as a server.

5.. HTTP Requests

The request sent by the client to a web server, contains all sorts of potentially interesting information; it is known as HTTP requests. The HTTP client sends the request to the server in the form of request message which includes following information:

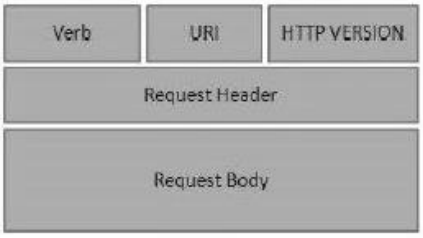
- The Request-line
- The analysis of source IP address, proxy and port
- The analysis of destination IP address, protocol, port and host
- The Requested URI (Uniform Resource Identifier)
- The Request method and Content
- The User-Agent header
- The Connection control header
- The Cache control header

The HTTP request method indicates the method to be performed on the resource identified by the Requested URI (Uniform Resource Identifier). This method is case-sensitive and should be used in uppercase.

HTTP Request Specification

An HTTP Request has five major parts –

- **HTTP Method** – HTTP methods can be GET, POST, DELETE, PUT, etc.
- **URI** – Uniform Resource Identifier (URI) to identify the resource on the server.
- **Request Header** – Contains metadata for the HTTP Request message as key-value pairs. For example, client (or browser) type, format supported by the client, format of the message body, cache settings, etc.
- **Request Body** – Message content or Resource representation. It should be represented in JSON.



HTTP Request

5.1 URI

Each of the REST API is identified by a unique URI(Uniform Resource Interface) or URL(Uniform Resource Locator). URI is also called endpoints. We can use Different HTTP methods on different URI.

Example of URI:

http://www.abc.com/api/v1/users?username=vikash

Different parts of URI:

- http://www.abc.com: This is a Domain Name of a Rest API
- /api/v1/users: This is known as the path parameter
- ?username=vikash: This is known as a Query Parameter. The text after ? is known as Query Parameter.

5.2 HTTP methods

Following four HTTP methods are commonly used in REST based architecture.

- **GET** – Provides a read only access to a resource.
- **POST** – Used to create a new resource.
- **DELETE** – Used to remove a resource.
- **PUT** – Used to update an existing resource.
- **PATCH** – Used to partially-update an existing Resource
- **HEAD** – Asks for only the header part of whatever a GET would return. Just like GET but with no body.
- **TRACE** – Asks for the loopback of the request message, for testing or troubleshooting.
- **OPTIONS** - Asks for a list of the HTTP methods to which the thing at the request URL can respond.

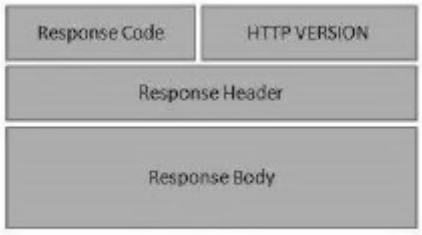
5.3 GET vs POST

GET	POST
1) In case of Get request, only **limited amount of data** can be sent because data is sent in header.	In case of post request, **large amount of data** can be sent because data is sent in body.
2) Get request is **not secured** because data is exposed in URL bar.	Post request is **secured** because data is not exposed in URL bar.
3) Get request **can be bookmarked.**	Post request **cannot be bookmarked.**
4) Get request is **idempotent** . It means second request will be ignored until response of first request is delivered	Post request is **non-idempotent.**
5) Get request is **more efficient** and used more than Post.	Post request is **less efficient** and used less than get.

6.. HTTP Response

An HTTP Response has four major parts –

- **Status/Response Code** – Indicates the Server status for the requested resource. For example, 404 means resource not found and 200 means response is ok.
- **Response Header** – Contains metadata for the HTTP Response message as key-value pairs. For example, content length, content type, response date, server type, etc.
- **Response Body** – Response message content or Resource representation.



HTTP Response

6.1 HTTP Response Codes

REST APIs use the Status-Line part of an HTTP response message to inform clients of their request's overarching result. RFC 2616 defines the Status-Line syntax as shown below:

Status-Line = HTTP-Version SP Status-Code SP Reason-Phrase CRLF

HTTP defines these standard status codes that can be used to convey the results of a client's request. The status codes are divided into five categories.

- **1xx**: **Informational** – Communicates transfer protocol-level information.

Status Code	Description
100 Continue	An interim response. Indicates to the client that the initial part of the request has been received and has not yet been rejected by the server. The client SHOULD continue by sending the remainder of the request or, if the request has already been completed, ignore this response. The server MUST send a final response after the request has been completed.
101 Switching Protocol	Sent in response to an Upgrade request header from the client, and indicates the protocol the server is switching to.
102 Processing (WebDAV)	Indicates that the server has received and is processing the request, but no response is available yet.
103 Early Hints	Primarily intended to be used with the Link header. It suggests the user agent start preloading the resources while the server prepares a final response.

- **2xx: Success** – Indicates that the client's request was accepted successfully.

Status Code	Description
200 OK	Indicates that the request has succeeded.
201 Created	Indicates that the request has succeeded and a new resource has been created as a result.
202 Accepted	Indicates that the request has been received but not completed yet. It is typically used in log running requests and batch processing.
203 Non-Authoritative Information	Indicates that the returned metainformation in the entity-header is not the definitive set as available from the origin server, but is gathered from a local or a third-party copy. The set presented MAY be a subset or superset of the original version.
204 No Content	The server has fulfilled the request but does not need to return a response body. The server may return the updated meta information.
205 Reset Content	Indicates the client to reset the document which sent this request.
206 Partial Content	It is used when the `Range` header is sent from the client to request only part of a resource.
207 Multi-Status (WebDAV)	An indicator to a client that multiple operations happened, and that the status for each operation can be found in the body of the response.
208 Already Reported (WebDAV)	Allows a client to tell the server that the same resource (with the same binding) was mentioned earlier. It never appears as a true HTTP response code in the status line, and only appears in bodies.
226 IM Used	The server has fulfilled a GET request for the resource, and the response is a representation of the result of one or more instance-manipulations applied to the current instance.

- **3xx: Redirection** – Indicates that the client must take some additional action in order to complete their request.

Status Code	Description
300 Multiple Choices	The request has more than one possible response. The user-agent or user should choose one of them.
301 Moved Permanently	The URL of the requested resource has been changed permanently. The new URL is given by the `Location` header field in the response. This response is cacheable unless indicated otherwise.
302 Found	The URL of the requested resource has been changed temporarily. The new URL is given by the `Location` field in the response. This response is only cacheable if indicated by a `Cache-Control` or `Expires` header field.
303 See Other	The response can be found under a different URI and SHOULD be retrieved using a GET method on that resource.
304 Not Modified	Indicates the client that the response has not been modified, so the client can continue to use the same cached version of the response.
305 Use Proxy (Deprecated)	Indicates that a requested response must be accessed by a proxy.
306 (Unused)	It is a reserved status code and is not used anymore.
307 Temporary Redirect	Indicates the client to get the requested resource at another URI with same method that was used in the prior request. It is similar to `302 Found` with one exception that the same HTTP method will be used that was used in the prior request.
308 Permanent Redirect (experimental)	Indicates that the resource is now permanently located at another URI, specified by the `Location` header. It is similar to `301 Moved Permanently` with one exception that the same HTTP method will be used that was used in the prior request.

- **4xx: Client Error** – This category of error status codes points the finger at clients.

Status Code	Description
400 Bad Request	The request could not be understood by the server due to incorrect syntax. The client SHOULD NOT repeat the request without modifications.
401 Unauthorized	Indicates that the request requires user authentication information. The client MAY repeat the request with a suitable Authorization header field
402 Payment Required (Experimental)	Reserved for future use. It is aimed for using in the digital payment systems.
403 Forbidden	Unauthorized request. The client does not have access rights to the content. Unlike 401, the client's identity is known to the server.
404 Not Found	The server can not find the requested resource.
405 Method Not Allowed	The request HTTP method is known by the server but has been disabled and cannot be used for that resource.
406 Not Acceptable	The server doesn't find any content that conforms to the criteria given by the user agent in the `Accept` header sent in the request.
407 Proxy Authentication Required	Indicates that the client must first authenticate itself with the proxy.
408 Request Timeout	Indicates that the server did not receive a complete request from the client within the server's allotted timeout period.
409 Conflict	The request could not be completed due to a conflict with the current state of the resource.
410 Gone	The requested resource is no longer available at the server.
411 Length Required	The server refuses to accept the request without a defined Content- Length. The client MAY repeat the request if it adds a valid `Content-Length` header field.
412 Precondition Failed	The client has indicated preconditions in its headers which the server does not meet.
413 Request Entity Too Large	Request entity is larger than limits defined by server.
414 Request-URI Too Long	The URI requested by the client is longer than the server can interpret.
415 Unsupported Media Type	The media-type in `Content-type` of the request is not supported by the server.
416 Requested Range Not Satisfiable	The range specified by the `Range` header field in the request can't be fulfilled.
417 Expectation Failed	The expectation indicated by the `Expect` request header field can't be met by the server.
418 I'm a teapot (RFC 2324)	It was defined as April's fool joke and is not expected to be implemented by actual HTTP servers. (RFC 2324)
420 Enhance Your Calm (Twitter)	Returned by the Twitter Search and Trends API when the client is being rate limited.
422 Unprocessable Entity (WebDAV)	The server understands the content type and syntax of the request entity, but still server is unable to process the request for some reason.
423 Locked (WebDAV)	The resource that is being accessed is locked.
424 Failed Dependency (WebDAV)	The request failed due to failure of a previous request.

425 Too Early (WebDAV)	Indicates that the server is unwilling to risk processing a request that might be replayed.
426 Upgrade Required	The server refuses to perform the request. The server will process the request after the client upgrades to a different protocol.
428 Precondition Required	The origin server requires the request to be conditional.
429 Too Many Requests	The user has sent too many requests in a given amount of time ("rate limiting").
431 Request Header Fields Too Large	The server is unwilling to process the request because its header fields are too large.
444 No Response (Nginx)	The Nginx server returns no information to the client and closes the connection.
449 Retry With (Microsoft)	The request should be retried after performing the appropriate action.
450 Blocked by Windows Parental Controls (Microsoft)	Windows Parental Controls are turned on and are blocking access to the given webpage.
451 Unavailable For Legal Reasons	The user-agent requested a resource that cannot legally be provided.
499 Client Closed Request (Nginx)	The connection is closed by the client while HTTP server is processing its request, making the server unable to send the HTTP header back.

- **5xx: Server Error** – The server takes responsibility for these error status codes.

Status Code	Description
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request.
501 Not Implemented	The HTTP method is not supported by the server and cannot be handled.
502 Bad Gateway	The server got an invalid response while working as a gateway to get the response needed to handle the request.
503 Service Unavailable	The server is not ready to handle the request.
504 Gateway Timeout	The server is acting as a gateway and cannot get a response in time for a request.
505 HTTP Version Not Supported (Experimental)	The HTTP version used in the request is not supported by the server.
506 Variant Also Negotiates (Experimental)	Indicates that the server has an internal configuration error: the chosen variant resource is configured to engage in transparent content negotiation itself, and is therefore not a proper endpoint in the negotiation process.
507 Insufficient Storage (WebDAV)	The method could not be performed on the resource because the server is unable to store the representation needed to successfully complete the request.
508 Loop Detected (WebDAV)	The server detected an infinite loop while processing the request.
510 Not Extended	Further extensions to the request are required for the server to fulfill it.
511 Network Authentication Required	Indicates that the client needs to authenticate to gain network access.

7.. Server: Application vs Web

Server is a device or a computer program that accepts and responds to the request made by another program, known as client. It is used to manage the network resources and for running the program or software that provides services. There are two types of servers:

- Web Server
- Application Server

7.1 Web Server

Web server contains only web or servlet container. It can be used for servlet, jsp, struts, jsf etc. It can't be used for EJB. It is a computer where the web content can be stored. In general web server can be used to host the web sites but there also used some other web servers also such as FTP, email, storage, gaming etc.

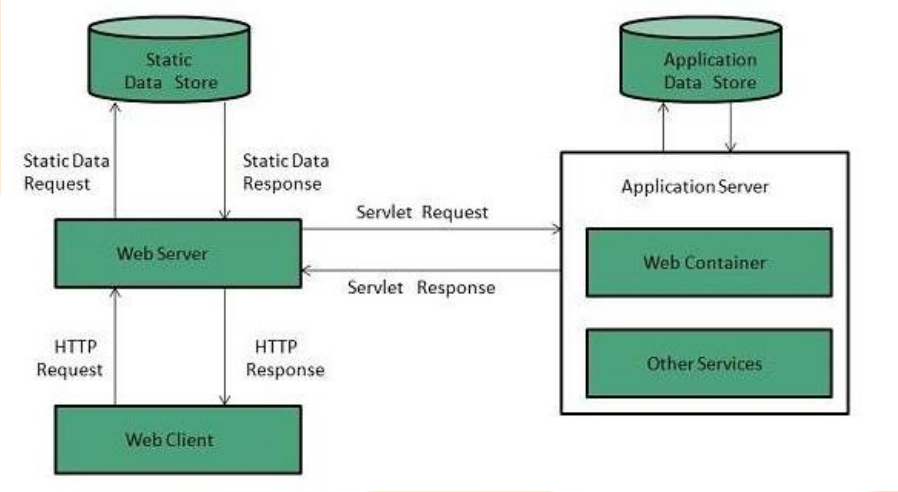
Examples of Web Servers are: **Apache Tomcat** and **Resin**.

Web Server Working

It can respond to the client request in either of the following two possible ways:

- Generating response by using the script and communicating with database.
- Sending file to the client associated with the requested URL.

The block diagram representation of Web Server is shown below:



Important points

- If the requested web page at the client side is not found, then web server will sends the HTTP response: Error 404 Not found.
- When the web server searching the requested page if requested page is found then it will send to the client with an HTTP response.
- If the client requests some other resources then web server will contact to application server and data is store for constructing the HTTP response.

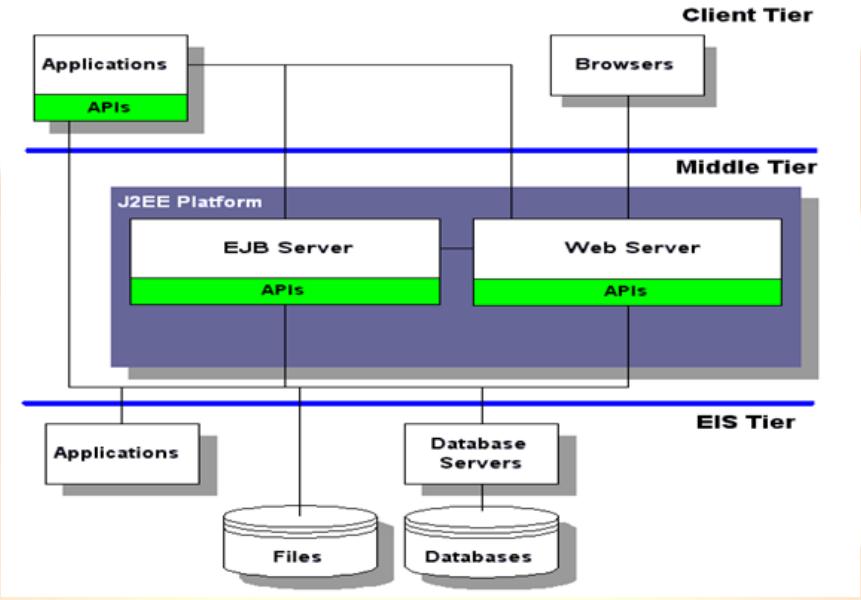
7.2 Application Server

An application server provides an environment to run Enterprise Application. Application servers can provide support for any technologies from Java J2EE which includes all web related Technologies and in addition to that EJBs, JMS components etc. Every application server contains an inbuilt web server to provide support for Web based technologies. If any

programmer doesn’t want an inbuilt web server then in that case he can go for any other web servers.

Application server contains Web and EJB containers. It can be used for servlet, jsp, struts, jsf, ejb etc. It is a component-based product that lies in the middle-tier of a server centric architecture. It provides the middleware services for state maintenance and security, along with persistence and data access. It is a type of server designed to install, operate and host associated services and applications for the IT services, end users and organizations.

The block diagram representation of Application Server is shown below:



The Example of Application Servers are:

- **JBoss**: Open-source server from JBoss community.
- **Glassfish**: Provided by Sun Microsystem. Now acquired by Oracle.
- **Weblogic**: Provided by Oracle. It more secured.
- **Websphere**: Provided by IBM.

8.. Content Type

Content Type is also known as MIME (Multipurpose internet Mail Extension)Type. It is a HTTP header that provides the description about what are you sending to the browser. MIME is an internet standard that is used for extending the limited capabilities of email by allowing the insertion of sounds, images and text in a message.

The features provided by MIME to the email services are as given below:

- It supports the non-ASCII characters
- It supports the multiple attachments in a single message
- It supports the attachment which contains executable audio, images and video files etc.
- It supports the unlimited message length.

8.1 List of Content Types

There are many content types. The commonly used content types are given below:

- text/html
- text/plain
- application/msword
- application/vnd.ms-excel
- application/jar
- application/pdf
- application/octet-stream
- application/x-zip
- images/jpeg
- images/png
- images/gif
- audio/mp3
- video/mp4
- video/quicktime etc.