API standard:json,..

http methods: get, post, put, delete, patch

GET:retrieve data from a specified resource

POST: submit data to be processed to a specified resource

PUT: update a specified resource

DELETE: Delete a specified resource

HEAD: same as get but does not return a body

OPTIONS: Returns the supported HTTP methods

PATCH: Update partial resources

-REST = representation state tranfer, standard design principals,

{

1.Uniform interface

-identification of resources

-manipulation through representation

Self-descriptive messages

-HATEOAS

2.Cliente-server

-Information hiding; encapsulation

-use programming language features like pivate variables or an explicit exporting policy

3-Statless

-No session, no history

-If client application needs to be a stateful application for end user, request from client should contain all the information necessary to servive

4.Cacheable(Never have to generate a response twice, brings performance improvements)

-first class citizen

-resources must declare themselves cachable

-the benefit of doing this is that we gainn speed and reduce server load

5.Layered system

-The componentes in the API should do one process or action, and do it well

-Each component should only know what the next layer is

6.Executable Code

-optional

-servers can temporarily extend or customize the functionality of a client by the transfer of executable code (basically an scrip tag in html page)

}

**CONVENTION#1: RESOURCES**

-Collection & instances

-Nouns, not verbs

-Coarse-grained

To have it all in folder/albums -> /albums

/albums/113

/artist

/artist/23412

**CONVENTION#2: BEHAVIOR**

**-**Http verbs do not have 1:1 relationship with crud

(C:post only, R:get, U:put only, D:delete)wrong, you should leave the desition to the person using the API

**CONVENTION#3: DOCUMENTION**

-Easy to find, publicity accessible

-Show examples of complete request/response cycles

-Keep user updated of changes

**CONVENTION#4: VERSIONING**

-Want to try ensure backwards compability

Head or url

**MORE CONVENTIONS**

-Result filtering, ordering and searching

-allow users to limit fields returned by api

-pagination

-auto loading related resource representation

-rate liming

-error handling

**EXAMPLES**

ENDPOINTS

The URI/URL where api/service can be accessed by a client applicaction

GET <https://mysite.com/api/users>

GET <https://mysite.com/api/users/1> OR <https://mysite.com/api/users/details/1>

POST https://mysite.com/api/users

PUT <https://mysite.com/api/users/1> OR <https://mysite.com/api/users/update/1>

DELETE <https://mysite.com/api/users/1> OR <https://mysite.com/api/users/delete/1>