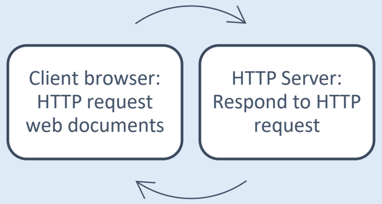
Client server model

Client server model

- **Server:** runs programs to serve applications to other computers
- **Client:** a computer that makes use of a service
- A client will make a request to a server. The server will run processes that are continuously listening for communications on a specific port. A server will serve many clients.
- When the server receives the request it then responds to the request
- Typical servers: file server, email server, FTP server, web server

Web server

- A web server is a computer on which are stored all the elements of a website including text images and other multimedia content as well as the HTML and CSS files.
- The web server will be able to understand HTTP requests from clients and respond to those requests.
- The web server will continuously be listening out for requests from clients.
- A browser requests a file from a web server using HTTP. When the request reaches the web server the file sent back to the browser, via HTTP.


Client browser: HTTP request web documents
HTTP Server: Respond to HTTP request

Websocket

- A websocket is an API that define the protocols between a client web browser and a server.
- A websocket protocol allows a persistent and dedicated full duplex (simultaneous two-way communication) connection between the client web browser and the server.
- Allows continuous transmission of data
- By comparison HTTP is half duplex and has greater overheads so is less efficient.
- Much faster because packets are smaller and contain less information so bandwidth requirement is reduced
- Useful for applications that require continuous real time data transfer ways between the client and server such as online gaming video conferencing, live video streaming anything that requires the constant transfer of data especially both

CRUD

There are four processes needed in a database with full functionality: Create, Retrieve, Update, Delete (CRUD).

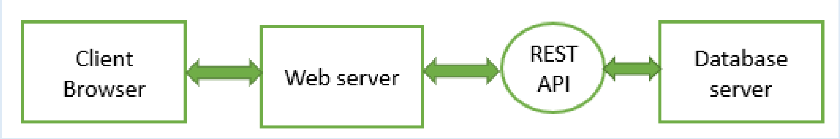
REST (Representational state transfer)

- REST is an API (Application Program Interface (API) that allows programs to work together. The functionality of one program can be accessed from another program
- REST runs on a server and allows clients to communicate with the server
- The database is connected to a client browser using REST API
- The HTTP request methods are mapped to SQL using the REST API following the principles of CRUD
- Javascript which runs on the client can communicate with the server through HTTP and can make calls to the REST API

CRUD, HTTP and SQL mapping

CRUD	HTTP	SQL
Create	POST	INSERT
Retrieve	GET	SELECT
Update	PUT	UPDATE
Delete	DELETE	DELETE

Web database architecture



JSON and XML

JSON (Javascript Object Notation) and XML (Extensible Markup Language) are standard methods of transferring data between a server and a client.

JSON Example

```
{“students":[
    { "firstName":“Thomas", "lastName":“Brown“,
“dateOfBirth":27/3/2001},
    { "firstName":“James", "lastName":“Frank“,
“dateOfBirth":13/4/2002}
   ]}
```

XML Example

```
<students>
 <student>
  <firstName>Thomas</firstName>
  <lastName>Brown</lastName>
  <dateOfBirth> 27/3/2001 </dateOfBirth>
 </ student >
 < student >
  <firstName>James</firstName>
  <lastName>Frank</lastName>
  <dateOfBirth> 13/4/2002 </dateOfBirth>
 </student>
</ students>
```

JSON versus XML

JSON	XML
Very easy to read	Contains tags so is not so easy to read
More compact less code	Lots of tags needed so is less compact
Only set data types can be used	Greater flexibility of data types
Syntax is very simple so easy to create	More complex syntax
Quick to parse	Slow to parse because it contains lots of tags

Thin versus thick client computing

Thin client

- Relies on a server to do much of the processing.
- The server needs to be extremely powerful to in order to be able to process all the requests from all clients on a network.
- A thin client computer can be low specification and does not need much hard disk storage or processing power.
- Much of the application software will be installed on the server.
- Client is essentially a terminal

Thick client

- All the applications are installed on the local machine.
- The processing is performed on the client.
- Very little reliance on the server

	Advantages	Disadvantages
Thin Client	Cheap low spec and old machines can be used for the clients Easy to maintain and manage software updates on a server Data are stored in one central location on the server so can be more secure Make it harder to pirate software	If server goes down the whole network is affected Need expensive very high performing server Need a good quality network to transfer data and requests between clients and server Can be a security risk as data are transferred over the network
Thick client	Do not need such a robust network and there is a lot less network traffic Do not need such high spec servers Different software can be installed on different machines	Need to have high specification clients Software needs to be on all client machines, thereby making maintenance and updating software more difficult