Node JS - Week-3 Assignment

npm commands→

→ npm init - initializes a node project. creates packages.json to track required modules/packages as well as node modules folder to track imported packages.

→ npm install - installs any requirements for the local node package based on the contents of package.json.

→ npm install <package-name> - install a package from npm's own repository as well as any requirements specified in that package's package.json file.

→ Web Service - A type of API that support HTTP requests, returning data such as JSON or plain text.

→ npm - The node package manager, used to initializes package.json files & install Node project dependencies.

→ Express - A module for simplifying the http server core modules in Node to implement API.

→ Module - A standalone package which can be used to extend the functionality of a Node project.

→ Server - A publicly accessible machine which exchanges information with one or more client at a time.

→ client - A private or public machine which requests information from a server.

Useful core Modules→

→ fs - The 'file system' module with various function to process data in file system.

→ path - Provides functions to process path string.

→ util - Provides various "utility" functions, such as util.promisify.

#Express Route Functions-

① Syntax- app.get("path", middlewareFn(s));

Defines a server endpoint which accepts a valid GET request.

```
=> app.get("/", (req, res) => {
       - -
   });
=> app.get("/:city", (req, res) => {
       let city = req.params.city;
   });
=> app.get("/citydata", (req, res) => {
       let city = req.query.city;

   });
=> app.get("/", validateInput, (req, res) => {
       - - -
   }, handleError);
```

② Syntax- app.post("path", middlewareFn(s));

Define a server endpoint which accepts a valid POST request.

Request Object Properties/Function-

=> req.params => captures a dictionary of desired path parameters. Keys are placeholder names & values are URL itself.

=> req.query => Captures a dictionary of query parameters, specified via ?key1=value1&key2=value2&..pattern.

=> req.body => Holds a dictionary of POST parameters as key/value pairs,

=> req.cookies => Retrieves all cookies sent in request. Requires cookie-parser module.

Response Object properties | Functions -

① syntax- res.set(HeadersName, value);

→ res.set("Content-Type", "text | plain");
→ res.set("Content-Type", "application/json");

Used to set different response header- commonly the "content-Type".

② Syntax- → res.type("text");
→ res.type("json");

Shorthand for setting the "content-Type" header.

③ Syntax- res.send (data);

→ res.send ("Hello");
→ res.send ({ "msg": "hello" });

Sends the data back to client, signaling an end to the response (doesnot terminate your JS program).

④ Syntax- res.end();

Ends the request/response cycle without any data. (doesnot terminate your JS program).

⑤ ~~Syn~~ Syntax- res.json (data);

Shorthand for setting the content type to JSON & sending JSON.

⑥ Syntax- res. status (status Code)

→ res.status (400). send ("client-side error message");
→ res. status (500). send ("server side error message");

Specifies the HTTP status code of response to communicate success/ failure to a client.