BASICS OF NODE JS

LAB-10

OCTOBER 18, 2023- BATCH A

Node.js

- Node.js is an open source runtime environment for server-side and networking applications and is single threaded.
- It uses Google Chrome's JavaScript Engine (V8 Engine) to execute code.
- It support cross-platform application deployment
- Provides an event-driven architecture and non-blocking I/O

Features of Node.js

- **Asynchronous and Event Driven −** All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data.

- **Very Fast** − Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.

- **Single Threaded but Highly Scalable** − Node.js uses a single threaded model with event looping.

- **Event Driven -** Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests.

Node.js

- The official Node.js website has installation instructions for Node.js: https://nodejs.org
- Create a js file named **main.js** on the Visual Studio Code or any other editor which having the following code.

```
console.log("Hello, World!")
```

Now execute main.js file using terminal to see the result –

```
node main.js
Hello, World!
```

First Application

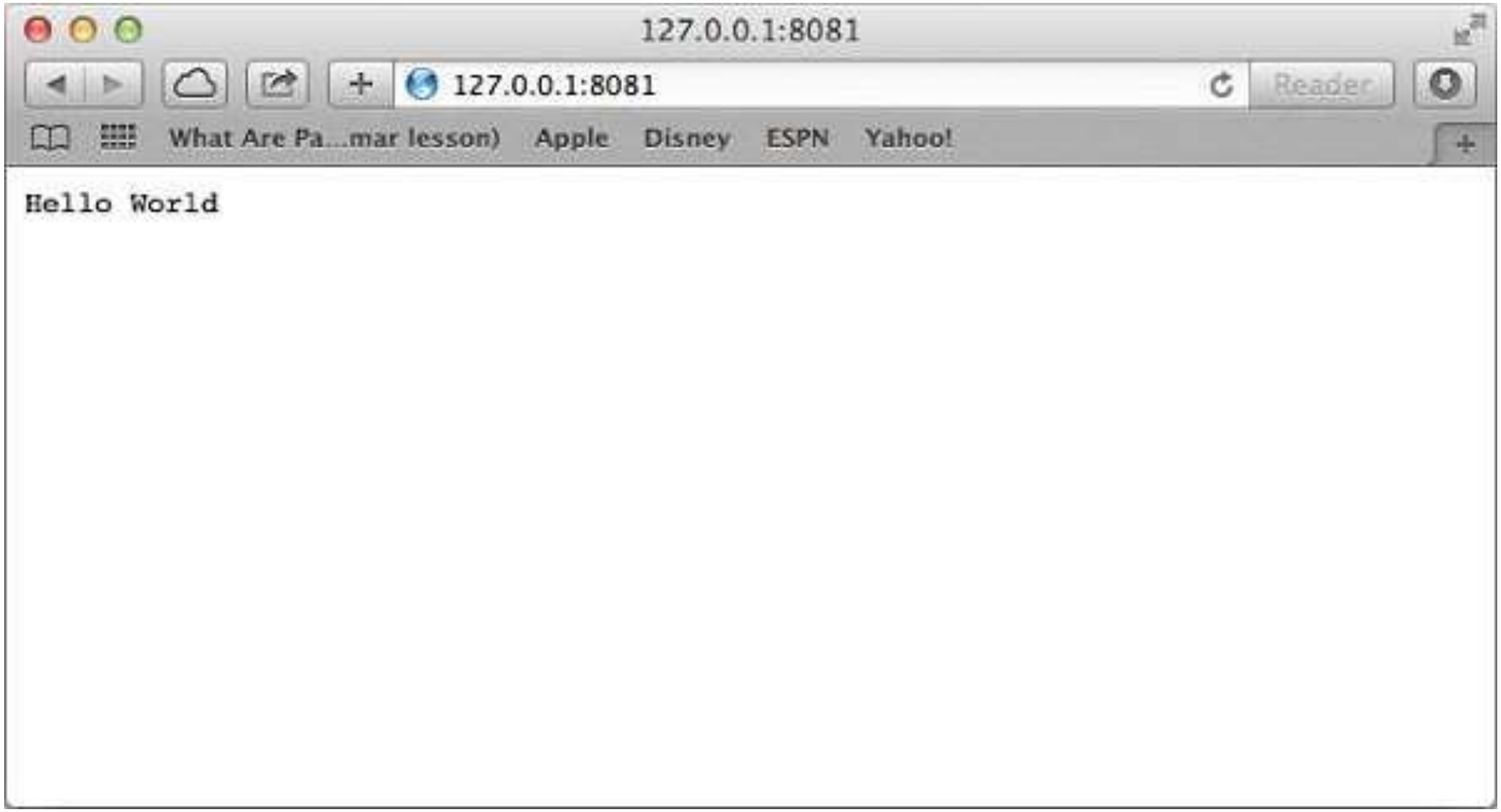
- Create a directory and make a .js file (e.g., main.js)

```
var http = require("http");
http.createServer(function (request, response) {
        response.writeHead(200, {'Content-Type': 'text/plain'});
    response.end('Hello World\n');
}).listen(8081);
console.log('Server running at http://127.0.0.1:8081/');
```

- In terminal, go to working directory and run the page with

```
node main.js
```

First Application



```
Hello World
```

NPM

Node Package Manager (NPM) provides two main functionalities –

1. Online repositories for node.js packages/modules which are searchable on search.nodejs.org
2. Command line utility to install Node.js packages, do version management and dependency management of Node.js packages.

Version of npm could be check with

```
$ npm --version
2.7.1
```

Installing modules using npm

There is a simple syntax to install any Node.js module –

```
$ npm install <Module Name>
```

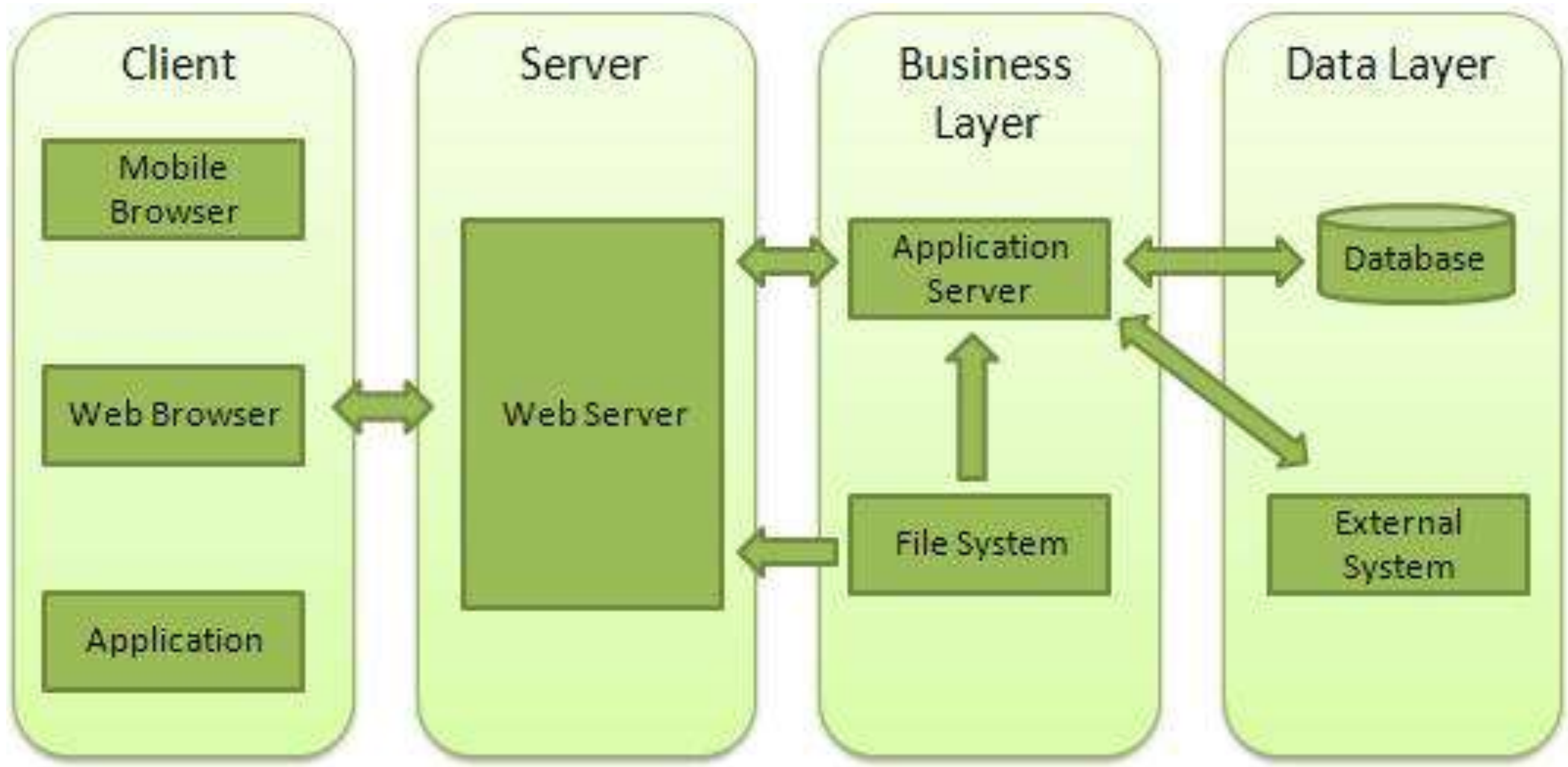
For example, following is the command to install a famous Node.js web framework module called express –

```
$ npm install express
```

Now we can use this module in our js file as following –

```
var express = require('express');
```

Web Application Architecture



Creating a Web Server using Node

- Create a directory and make a .js file (e.g., index.js)

```
const http = require("http");

const myServer = http.createServer((req, res) => {
    console.log('New Req Rec.’);
    res.end("hello from Server");
 });
myServer.listen(8091, () => console.log("server started"));
```

- Go to terminal and write the command

```
npm init
```

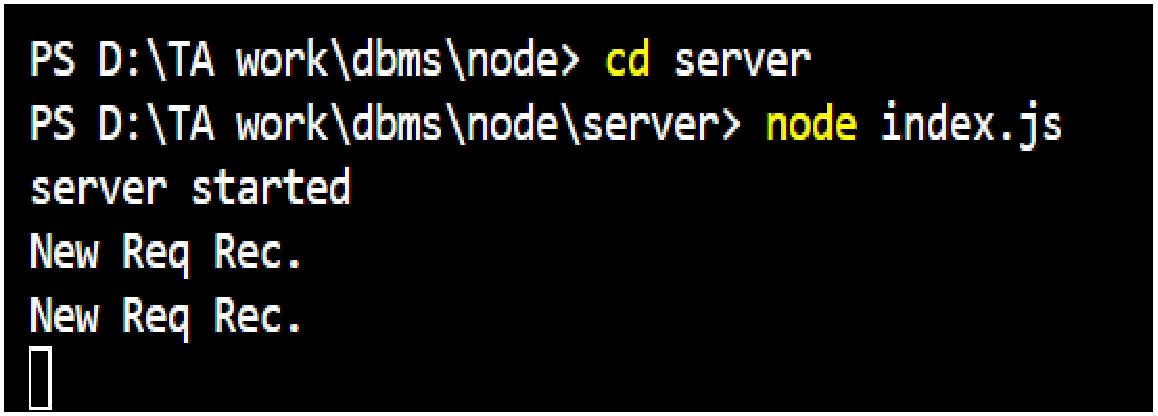
package.json file will create

```
{
 "name": "new",
 "version": "1.0.0",
 "description": "",
 "main": "new.js",
 "scripts": {
 "test": "echo \"Error: no test specified\" && exit 1“
},
 "author": "",
  "license": "ISC"
}
```

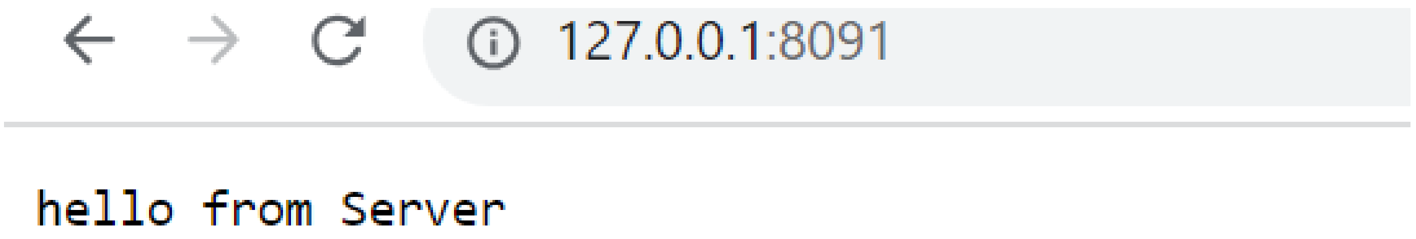
$\downarrow$

"start": "node index"

- Open the terminal, write the command *node index.js*



```
PS D:\TA work\dbms\node> cd server
PS D:\TA work\dbms\node\server> node index.js
server started
New Req Rec.
New Req Rec.
```



```
hello from Server
```

Node.js – Express

- Express js is a very popular web application framework built to create Node.js Web based applications.
- Following are some of the core features of Express framework –

  - ➢ Allows to set up middlewares to respond to HTTP Requests.
  - ➢ Defines a routing table which is used to perform different actions based on HTTP Method and URL.
  - ➢ Allows to dynamically render HTML Pages based on passing arguments to templates.

Node.js – Express

- Express js is a very popular web application framework built to create Node.js Web based applications.
- Following are some of the core features of Express framework −

  - ➢ Allows to set up middlewares to respond to HTTP Requests.
  - ➢ Defines a routing table which is used to perform different actions based on HTTP Method and URL.
  - ➢ Allows to dynamically render HTML Pages based on passing arguments to templates.

Installing Express

```
$ npm install express
```

- The above command saves the installation locally in the node_modules directory and creates a directory express inside node_modules.
- We need to install the following important modules along with express –
  - body-parser – This is a node.js middleware for handling JSON, Raw, Text and URL encoded form data.
  - cookie-parser – Parse Cookie header and populate req.cookies with an object keyed by the cookie names.
  - multer – This is a node.js middleware for handling multipart/form-data.

Installing Express

```
$ npm install body-parser
$ npm install cookie-parser
$ npm install multer
```

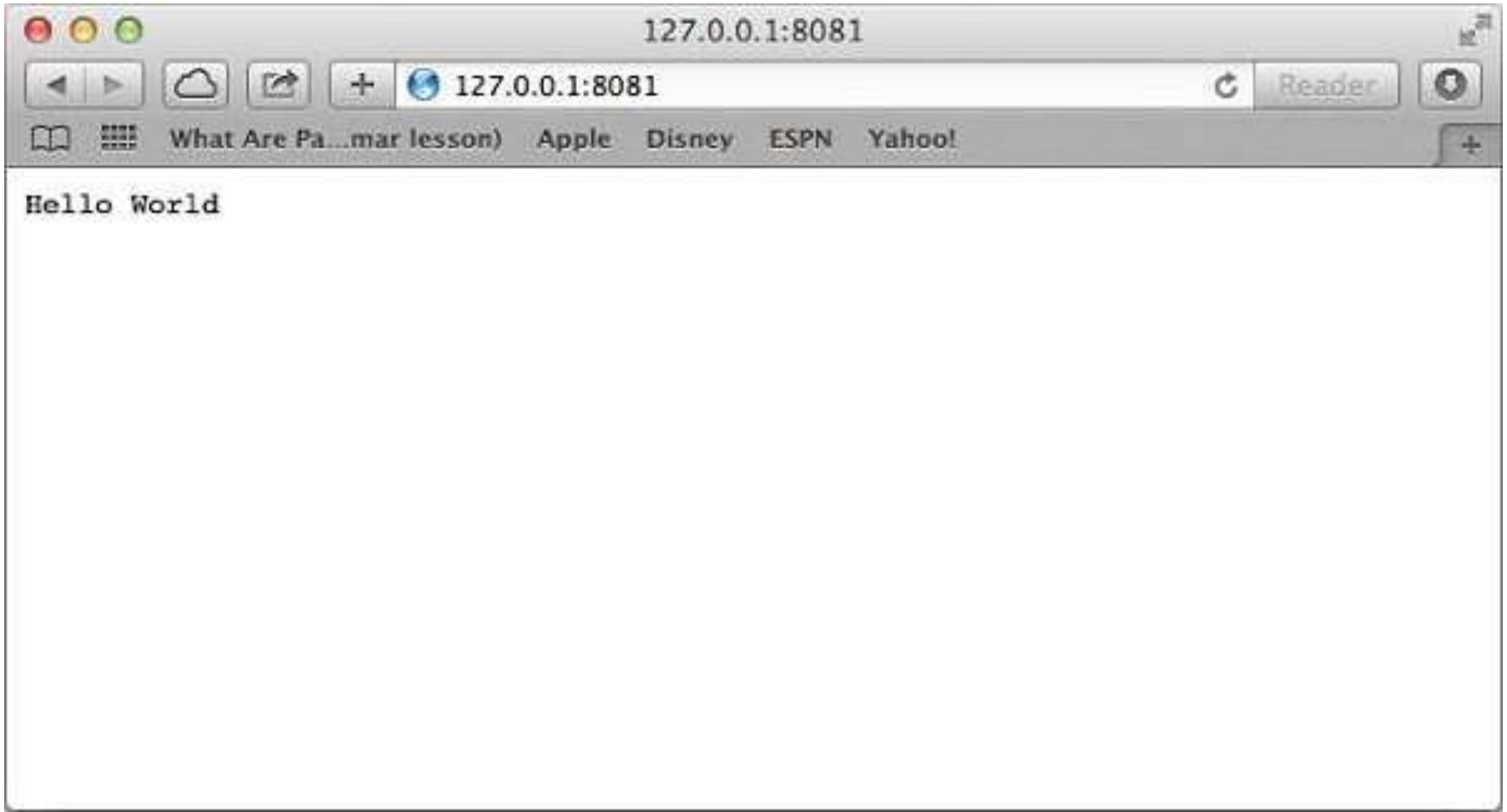
Hello world Example

```
var express = require('express');
var app = express();

app.get('/', function (req, res) {
   res.send('Hello World');
})

var server = app.listen(8081, function () {
   var host = server.address().address
   var port = server.address().port

   console.log("Example app listening at http://%s:%s", host, port)
})
```

Hello world Example

- Save the above code in a file named server.js and run it with the following command.

  node server.js



```
Hello World
```

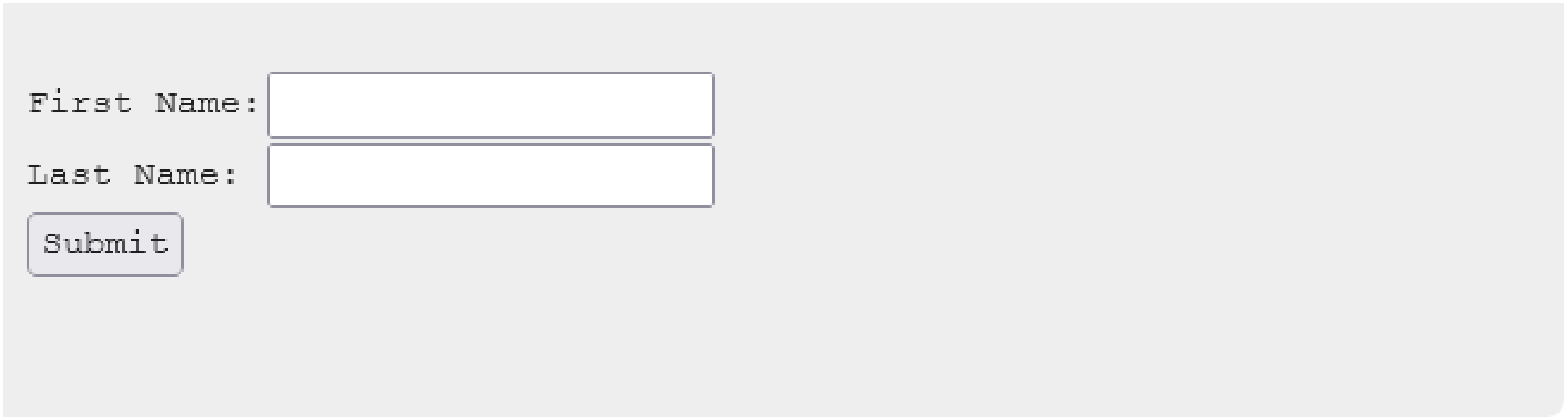
Form Data Example

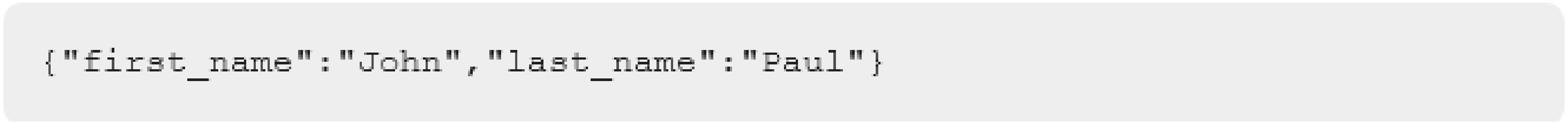
```
<html>
  <body>

    <form action = "http://127.0.0.1:8081/process_post" method =
   "POST">
      First Name: <input type = "text" name = "first_name"> <br>
      Last Name: <input type = "text" name = "last_name">
      <input type = "submit" value = "Submit">
    </form>

  </body>
</html>
```

Form Data Example

```
var express = require('express');
var app = express();
var bodyParser = require('body-parser');
var urlencodedParser = bodyParser.urlencoded({ extended: false })
app.use(express.static('public'));
app.get('/index.htm', function (req, res) {
  res.sendFile( __dirname + "/" + "index.htm" );
})
app.post('/process_post', urlencodedParser, function (req, res) {
response = {
    first_name:req.body.first_name,
    last_name:req.body.last_name
  };
  console.log(response);
  res.end(JSON.stringify(response));
})

var server = app.listen(8081, function () {
  var host = server.address().address
  var port = server.address().port

  console.log("Example app listening at http://%s:%s", host, port)
})
```

Output





```
{"first_name":"John","last_name":"Paul"}
```