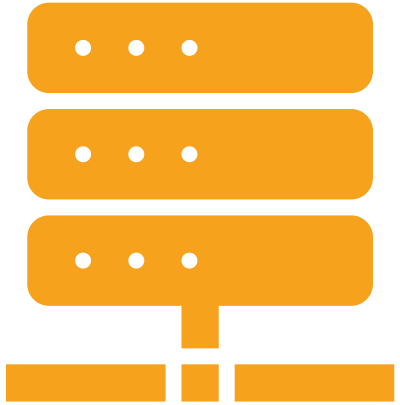


NODEJS

Feb 19, 2025

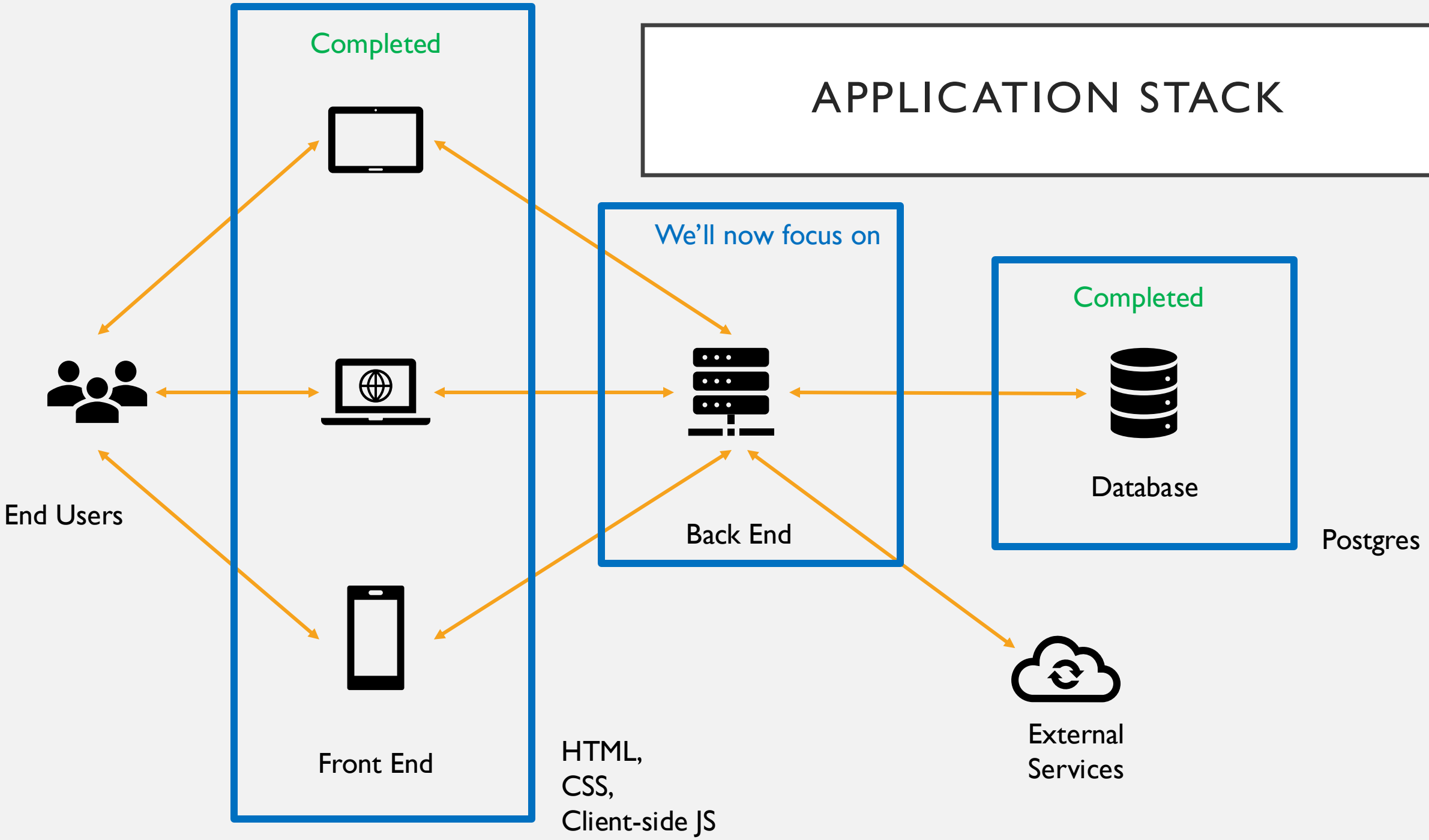
ANNOUNCEMENTS

- Watch Videos linked on Canvas
- Project Ideas survey
- Next Wednesday

AGENDA

- Where are we in the tech stack?
- Communication across the stack
- Introduction to NodeJS
- NodeJS Basics
- HTML Forms
- NodeJS Concepts
- Express Framework
- Routing
- Initializing a node project

APPLICATION STACK



COMMUNICATION ACROSS THE STACK

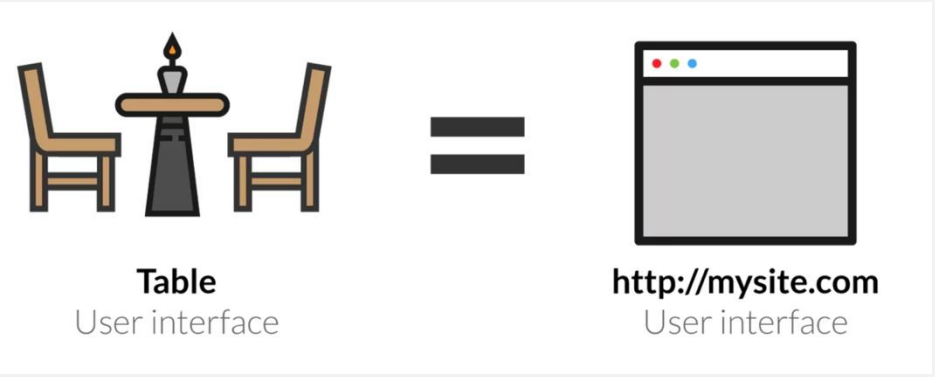
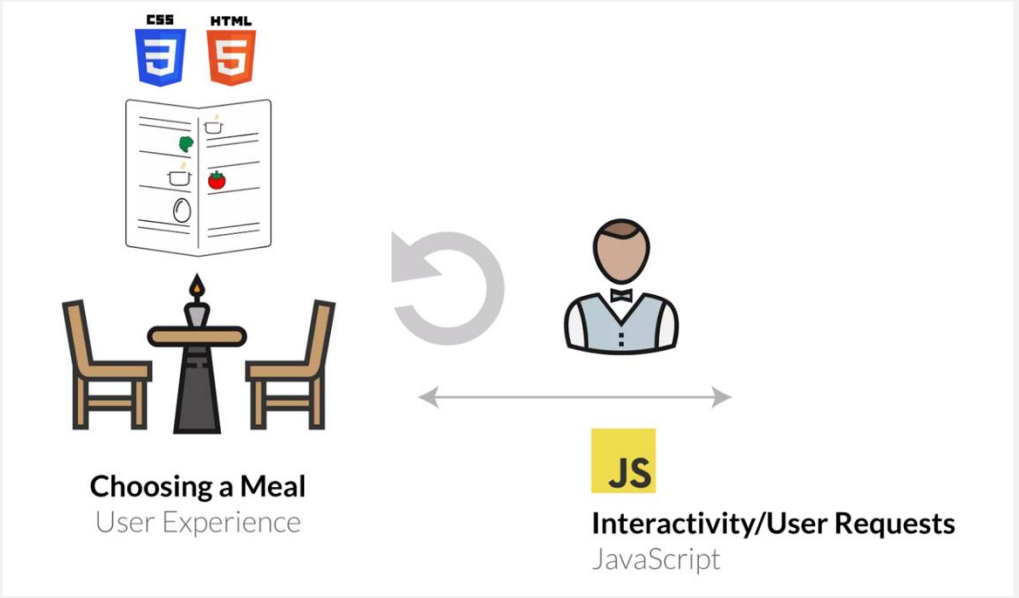
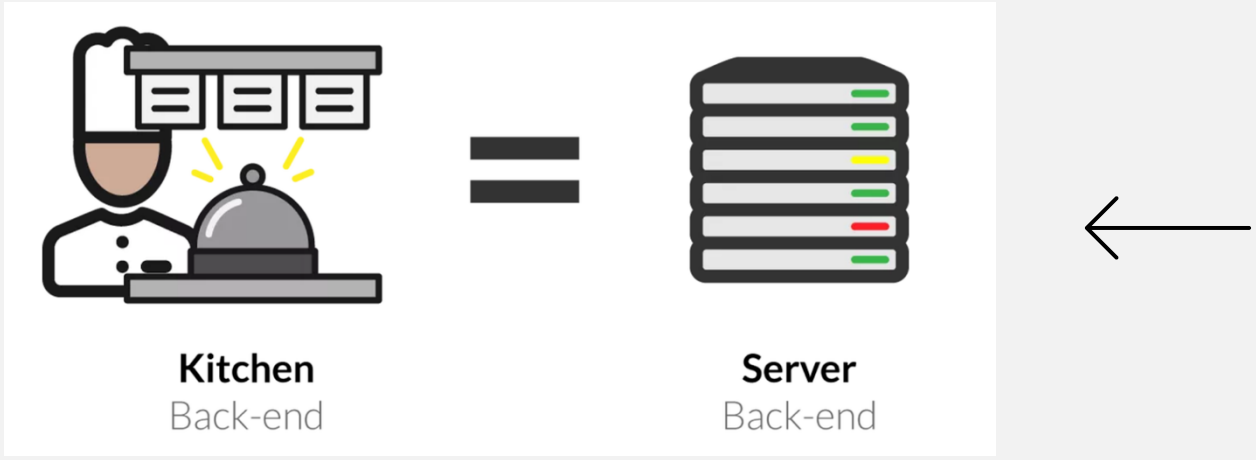
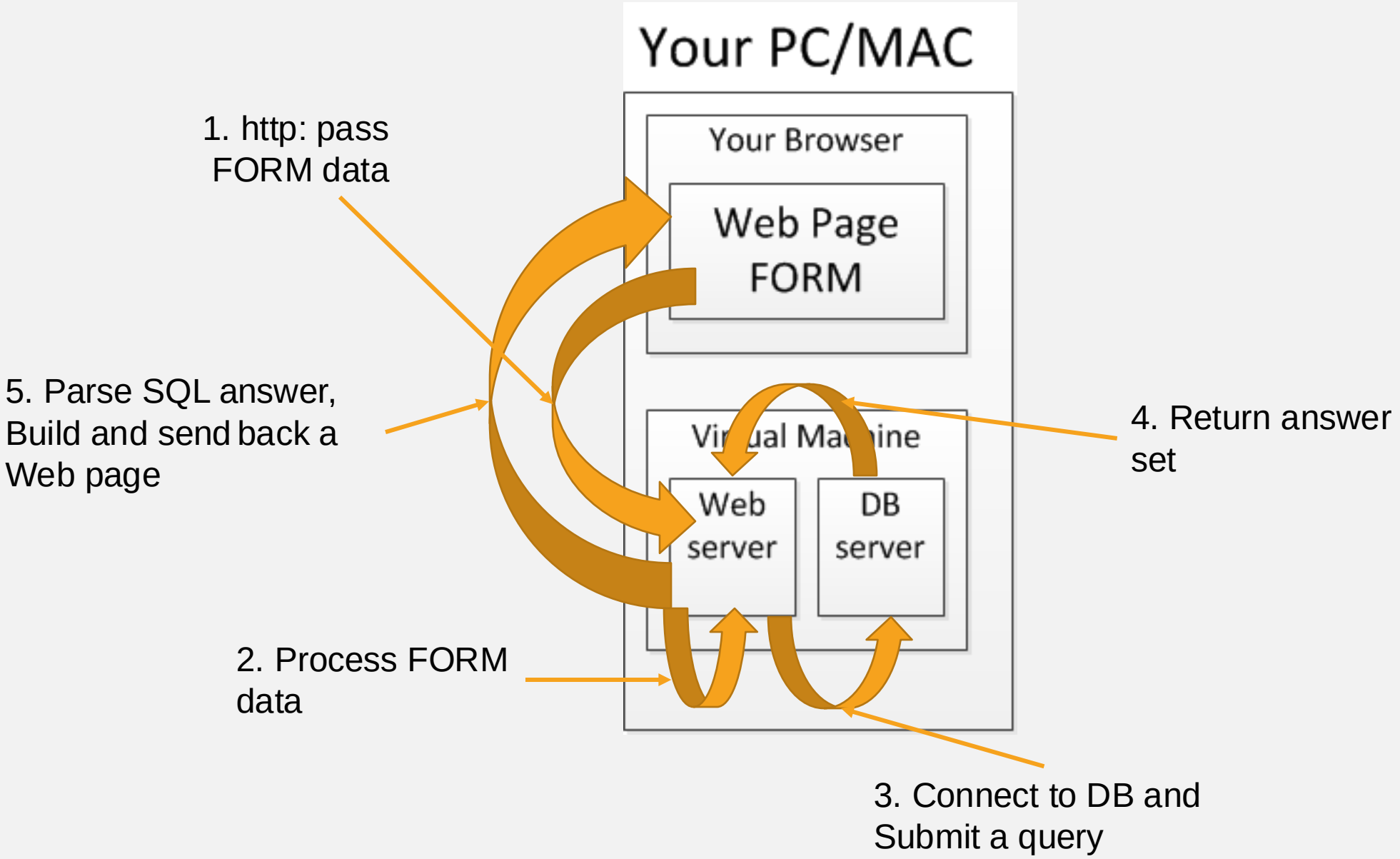


Table
User interface
http://mysite.com
User interface


Menu
Website Content
CSS
HTML
HTML + CSS
Code for Static Content


CSS
HTML
Choosing a Meal
User Experience
JS
Interactivity/User Requests
JavaScript


Kitchen
Back-end
Server
Back-end


Your PC/MAC
Your Browser
Web Page
FORM
Virtual Machine
Web
server
DB
server
1. http: pass
FORM data
2. Process FORM
data
3. Connect to DB and
Submit a query
4. Return answer
set
5. Parse SQL answer,
Build and send back a
Web page

INTRODUCTION TO NODEJS

NodeJS is a multi-purpose server-side processing engine.

- It is Open-Source (GPL – Gnu Public License)
- It is FREE
- It runs anywhere (Windows, Linux, Unix, Mac OS X)
- It uses the Java Script programming language – the "default" language for most web-based applications.
- It looks good on your resume.

NODEJS BASICS

NodeJS can process HTTP requests from your browser:

- Node.js code can generate dynamic page content – it *creates* the HTML on the fly
- Node.js code can create, open, read, write, delete, and close files on the server
- Node.js code can collect and process form data from an HTML page
- Node.js code can read, add, change, delete data in your database

HTML FORMS

- How are they used?
  - Use the browser's window as a data entry screen
  - Collect information from the user
  - Pass it to the web server via http
  - Invoke a server-side script
  - Passes **form data** as input to the script
  - Script on server parses out the form data

HTML FORMS

- `<form>` tag has several attributes – two are required
- ACTION
  - `<form action="http://URL">` name of a program on the web server
    - URL specifies the location of the executable file on the web server
  - `<form action="mailto:mailrecipient">` sends an email
- METHOD
  - `<form method="POST">` or `<form method="GET">`
    - **POST** when you have large amount of data being sent, encryption available, a two-step process
    - **GET** for small amounts, no security – all in one step

NODEJS CONCEPTS

- NodeJS programs handle HTTP: calls from the client
  - Two types of calls:
    - **req** – the HTTP: request coming from the client (properties for the request query string, parameters, body, HTTP headers)
    - **res** – the HTTP: response being sent back to the client

EXPRESS FRAMEWORK

- Node web framework that supports several other popular Node frameworks
- It is minimalistic
- Handles requests with different HTTP verbs
- Integrates rendering engines to generate responses
- Sets common web application settings like the port for connecting, the location of the resources for rendering

https://expressjs.com/en/resources/middleware.html

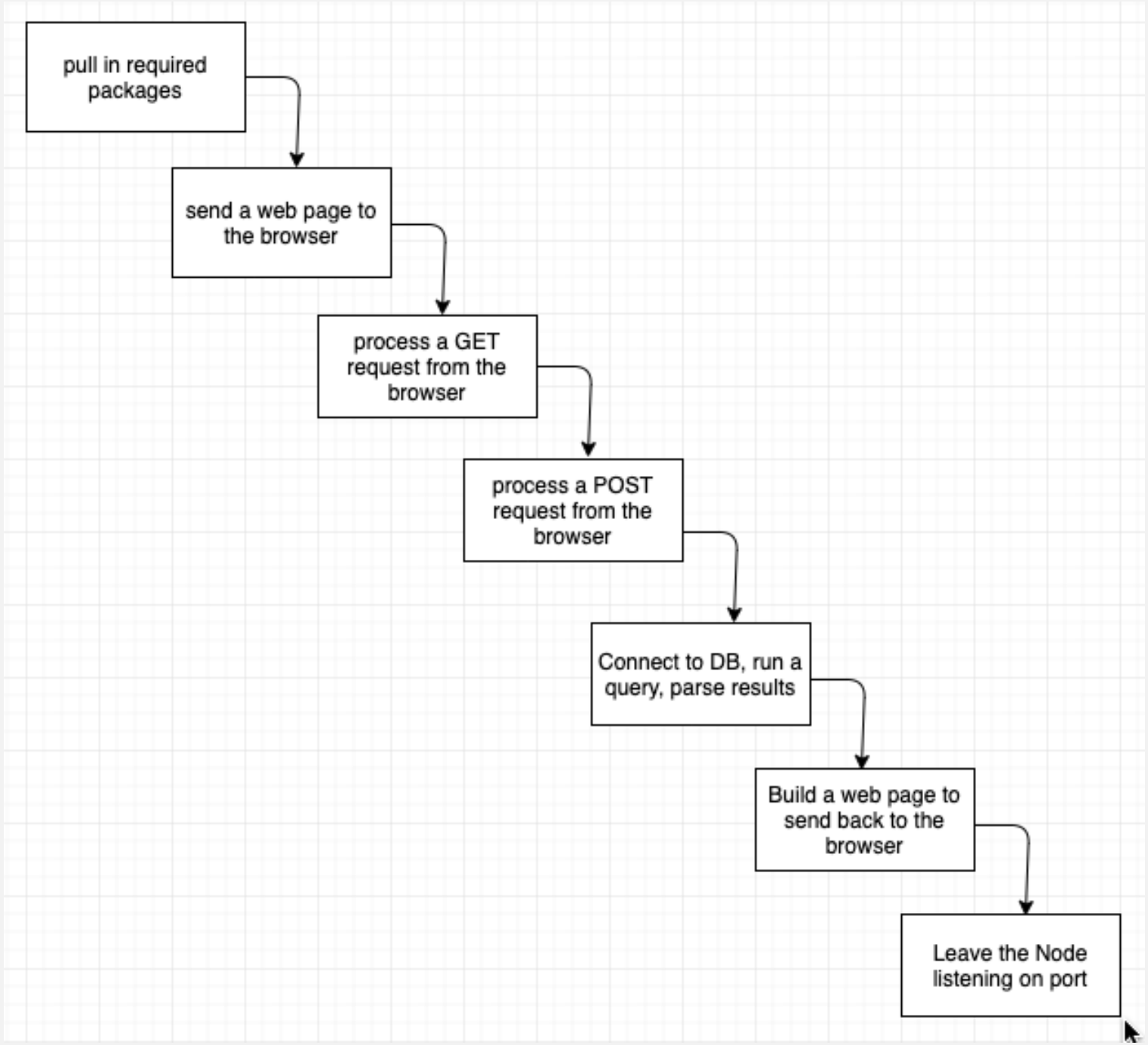
ROUTING

- Routing determines the way in which the NodeJS application responds to a client request to an endpoint.
- Defined by blocks/sections of code within your NodeJS program (will illustrate with code demos)
- Each section handles a URL and a method (GET or POST)
- uses the Express "app" object

ROUTING

- For example, a block of code may do
  - `app.get(/)` = process a get request for the "index" html file
  - `app.post(/)` = process a post request for the "index" html file
  - `app.get(/URL...)` = process a get request for the file "URL"
  - `app.post(/URL...)` = process a post request for the file "URL"

- Further reading: https://expressjs.com/en/guide/routing.html


pull in required packages
send a web page to the browser
process a GET request from the browser
process a POST request from the browser
Connect to DB, run a query, parse results
Build a web page to send back to the browser
Leave the Node listening on port

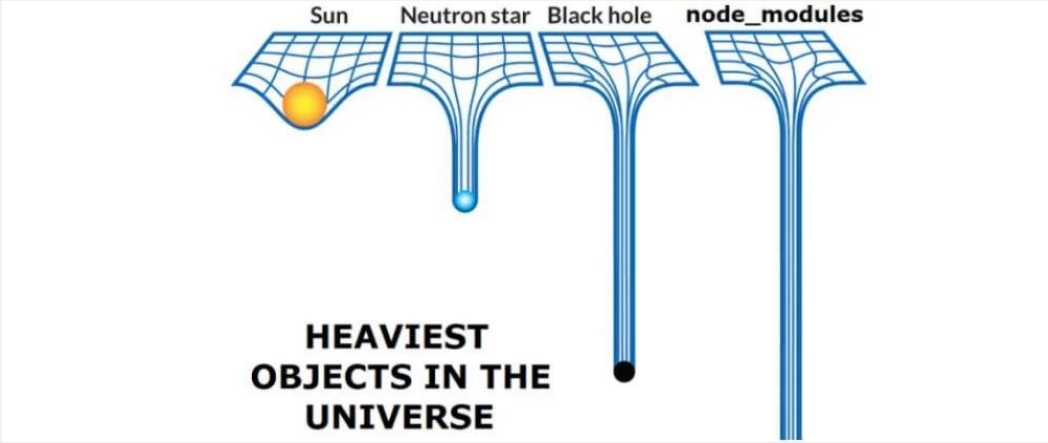
INITIALIZING A NODE PROJECT

`npm init` – Initializes a node project and generates a package.json file

`package.json` – Manifest that stores information about the application, modules, packages etc.

`node_modules` – Contains files for all dependencies and transitive dependencies

`.gitignore` – Make sure to include a .gitignore file in your node projects to avoid checking in generated files/folders.



https://nodesource.com/blog/the-basics-of-package-json

READING CLIENT INPUT

- **GET** request
- **Input parameter:** `recipeId`
- **Route on the server:**

```
app.get('/getRecipeById', (req, res) => {
  let id = req.query.recipeId;
});
```

- **Request URL**: `localhost:<PORT_NUMBER>/getRecipeById?recipeId=1`

READING CLIENT INPUT

- **GET** request
- **Input parameter:** `recipeId`
- **Route on the server:**

```
app.get('/getRecipeById/:recipeId', (req, res) => {
  let id = req.params.recipeId;
});
```

- **Request URL**: `localhost:<PORT_NUMBER>/getRecipeById/1`

READING CLIENT INPUT

- **POST** request
- **Input parameter(s):** `recipeName, author, …`
- **Route on the server:**

```
app.post('/addRecipe', (req, res) => {
  let rName = req.body.recipeName;
  let rName = req.body.author;
  });
```

- **Request URL:** `localhost:<PORT_NUMBER>/addRecipe`

Note: In a post request, you shouldn't be sending any data as part of the query URL

READING CLIENT INPUT

- **PUT** request
- **Input parameter(s):** `recipeName, author, …`
- **Route on the server:**

```
app.put('/updateRecipe', (req, res) => {
  let rName = req.body.recipeName;
  });
```

- **Request URL**: `localhost:<PORT_NUMBER>/addRecipe`

Note: You should consider the sensitivity of the data being sent to the server when choosing how to send the data

READING CLIENT INPUT

- **DELETE** request
- **Input parameter(s):** `recipeName`
- **Route on the server:**

```
app.delete('/deleteRecipe/:recipeName', (req, res) => {
  let rName = req.params.recipeName;
  });
```

- **Request URL**: `localhost:<PORT_NUMBER>/deleteRecipe/chicken`

Note: In a Delete request, you could send data in a query or body as well