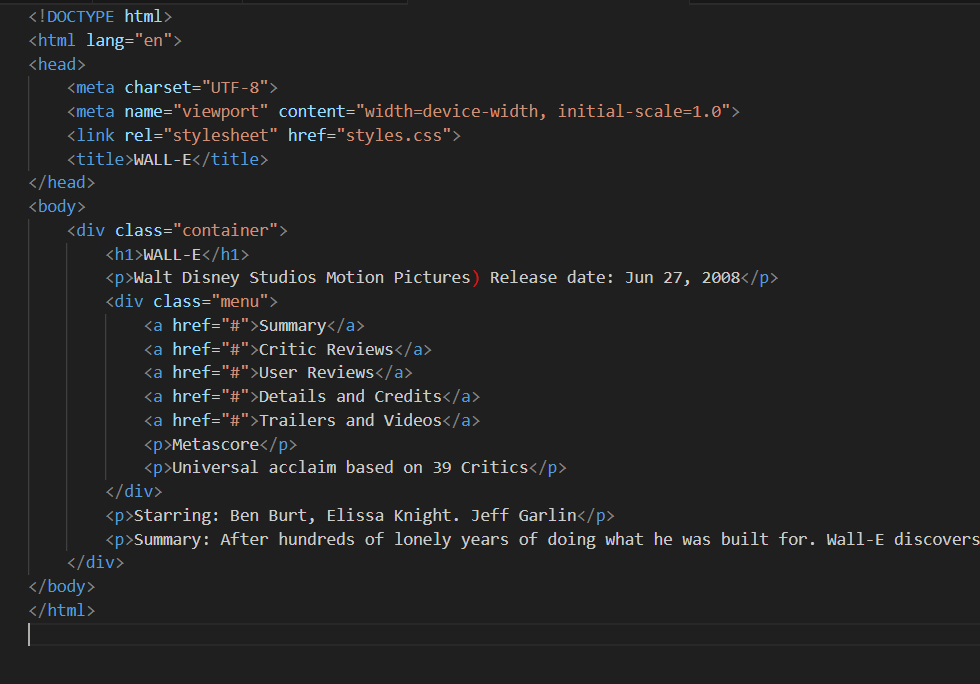
QUESTION 1

Create an HTML file with the following structure:



Create a CSS file named **styles.css** and apply the following styles:

body {

background-color: gray;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.container {

background-color: white;

width: 80%;

padding: 20px;

margin: 20px;

text-align: center;

}

h1 {

font-size: 30pt;

}

.menu {

background-color: black;

color: white;

padding: 20px;

display: flex;

justify-content: space-between;

align-items: center;

}

.menu a {

color: white;

text-decoration: none;

padding: 0 20px;

}

.menu a:hover {

text-decoration: underline;

}

p {

margin: 10px 0;

}

.green-background {

background-color: green;

display: inline-block;

padding: 5px;

margin: 5px;

font-size: 30pt;

color: white;

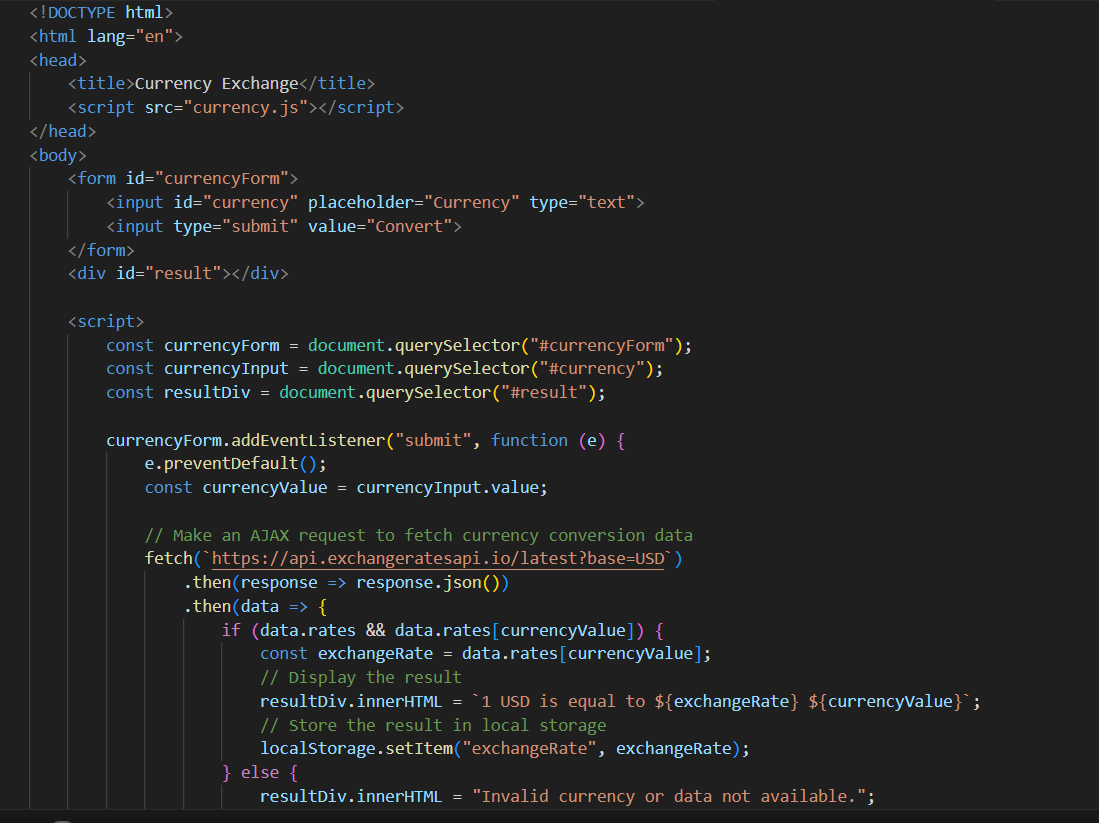
}

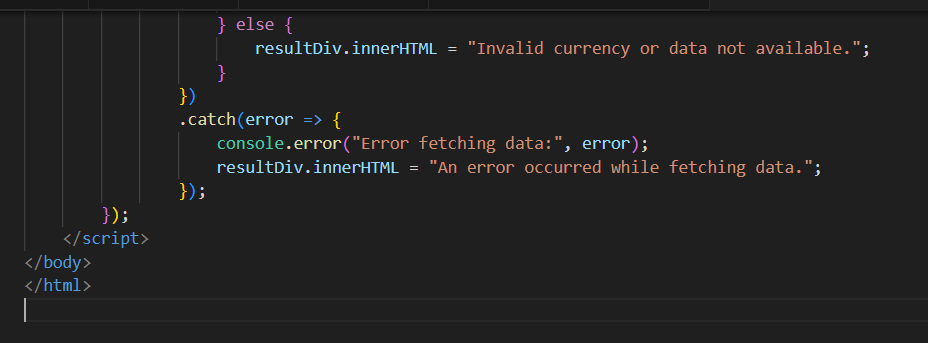
QUESTION 2



This JavaScript code selects the buttons, adds click event listeners, and changes the text color of the **<h1>** element based on the data-color attribute of the button.

b. For currency conversion and storing the result in local storage, you can use JavaScript and AJAX to fetch data from the provided API. Here's an example:





**QUESTION THREE**

a. Communication between browsers and web servers using the HTTP protocol involves various components being transferred in both directions. Here's a breakdown of the components, their directions, and the HTTP methods/verbs used:

i. Angular HTML templates:

- Direction: Web server to browser (initial load)

- HTTP Method/Verb: Typically, a GET request is used to retrieve HTML templates from the web server and send them to the browser. However, subsequent interactions may involve browser-to-server requests (e.g., to fetch updated templates).

ii. CSS style sheets:

- Direction: Browser to web server (rarely)

- HTTP Method/Verb: Typically, CSS style sheets are requested by the browser using GET requests when loading a web page. CSS files are not commonly sent from the browser to the web server.

iii. JavaScript libraries:

- Direction: Browser to web server (rarely)

- HTTP Method/Verb: JavaScript libraries are typically fetched by the browser via GET requests when loading a web page. They are not commonly sent from the browser to the web server.

iv. Model data:

- Direction: Browser to web server (when sending data for processing or updates) and web server to browser (when fetching data for rendering).

- HTTP Method/Verb: Data sent from the browser to the web server can use either POST or PUT requests, depending on the operation. Data fetched from the web server to the browser often uses GET requests.

v. Angular controllers:

- Direction: Browser to web server (rarely)

- HTTP Method/Verb: Angular controllers are not typically sent from the browser to the web server. They are part of the client-side application logic.

b. Two different ways information, such as parameters, can be sent from the browser to a web server using an HTTP POST request are:

1. Form Data: You can send data from the browser to the web server by including it in the body of an HTTP POST request. This is commonly done when submitting web forms. The data is usually encoded in key-value pairs and sent as the request body. The `application/x-www-form-urlencoded` content type is often used.

2. JSON Payload: Another way to send data is by constructing a JSON payload and including it in the request body. This is common in API requests where complex data structures need to be sent to the server. The content type is typically set to `application/json`.

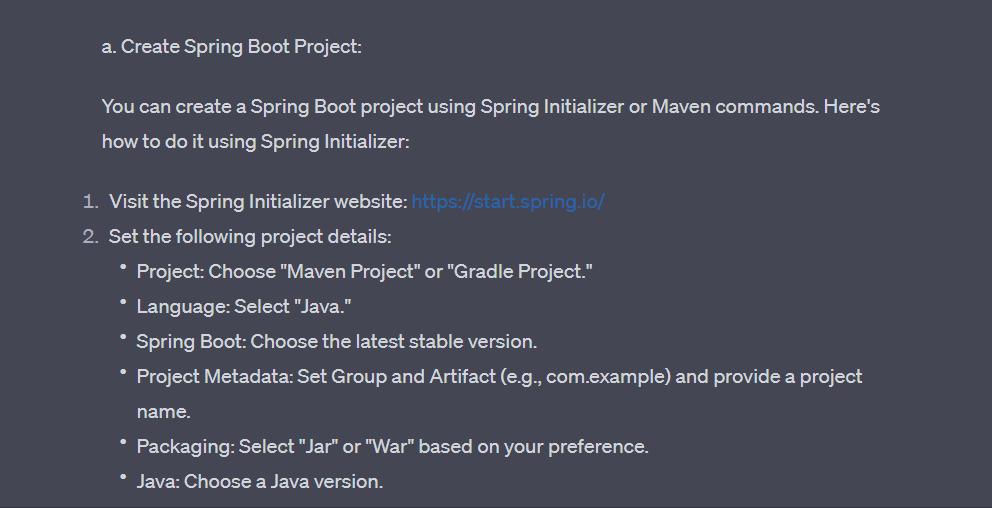
c. A browser determines how long it can cache a web page fetched with an HTTP GET request based on the information provided in the HTTP response headers. Specifically, the `Cache-Control` header and the `Expires` header influence caching:

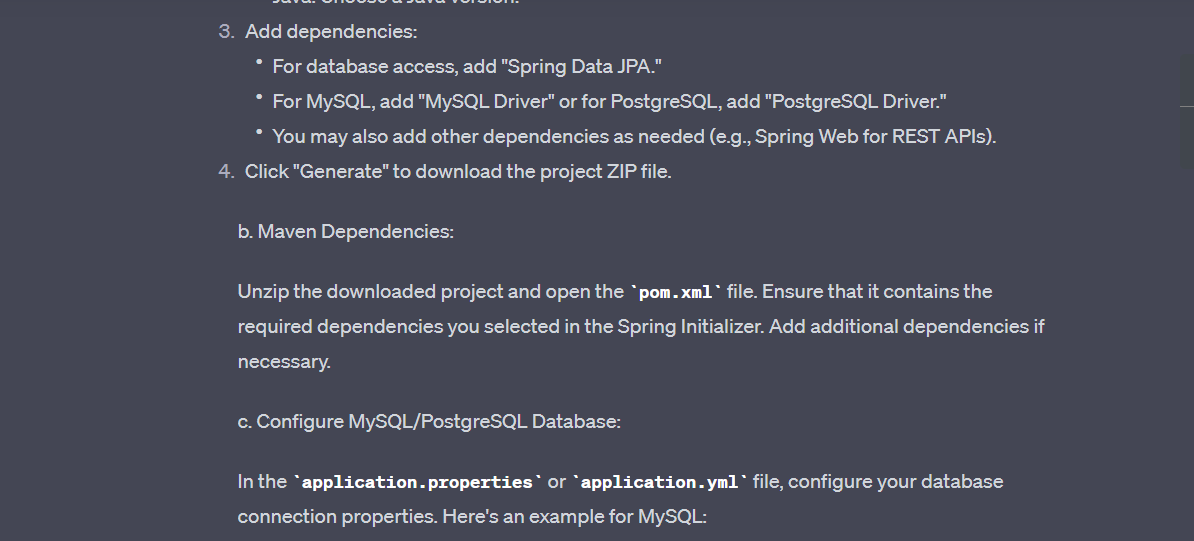
- `Cache-Control`: This header contains directives that specify caching behavior. For example, `Cache-Control: max-age=3600` indicates that the browser can cache the page for 3600 seconds (1 hour) before making a new request to the server. Other directives like `no-cache` and `no-store` can prevent caching.

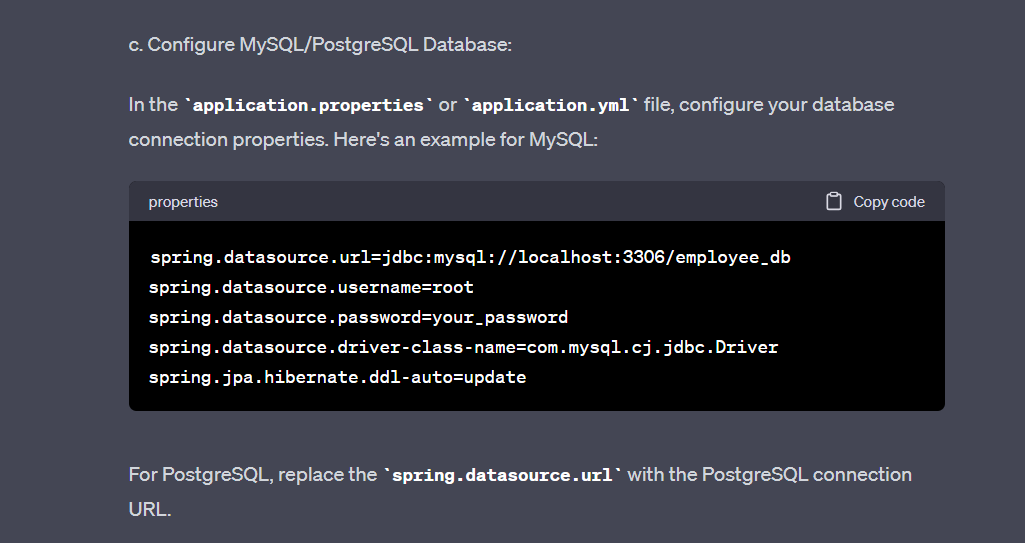
- `Expires`: This header provides an absolute date and time when the page will expire and should no longer be considered fresh. If the `Expires` header is set to a future date, the browser will cache the page until that date.

These headers allow the web server to instruct the browser on how long it should keep a cached copy of the page before checking for updates.

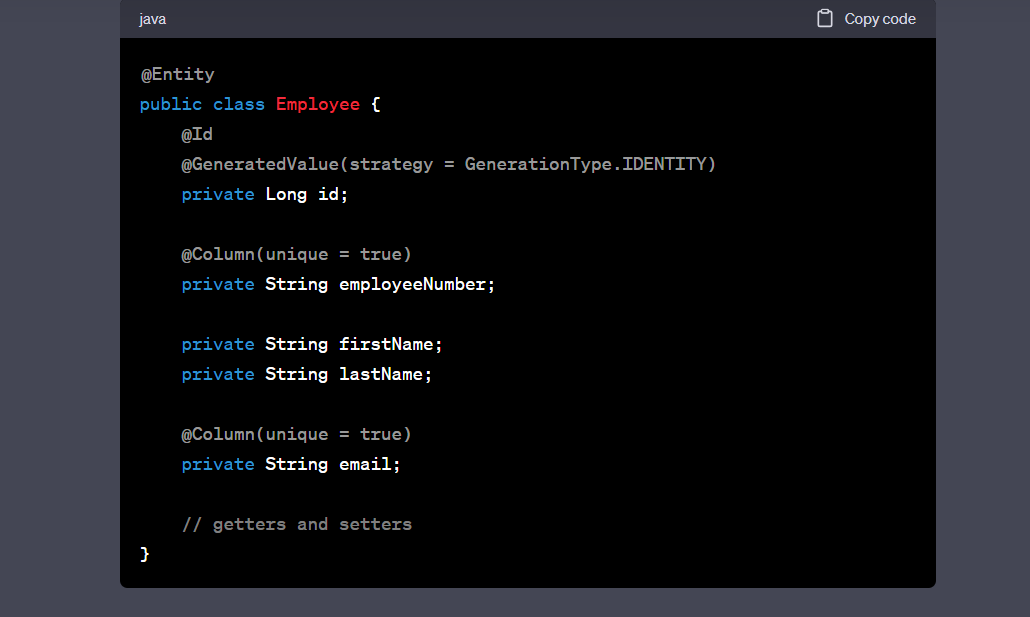
QUESTION 4

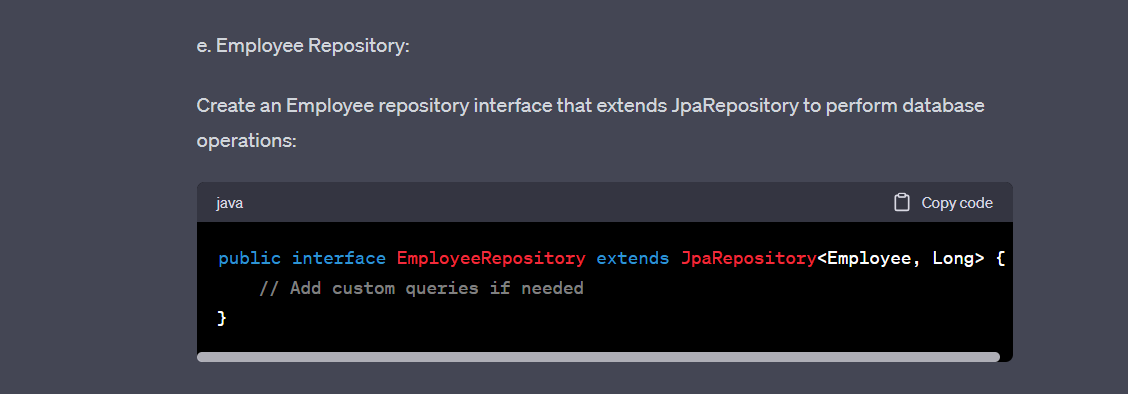


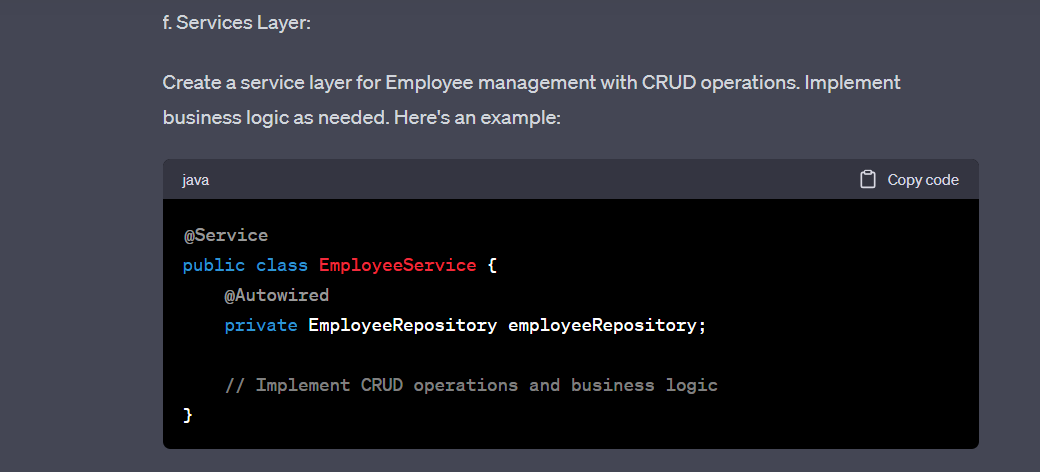


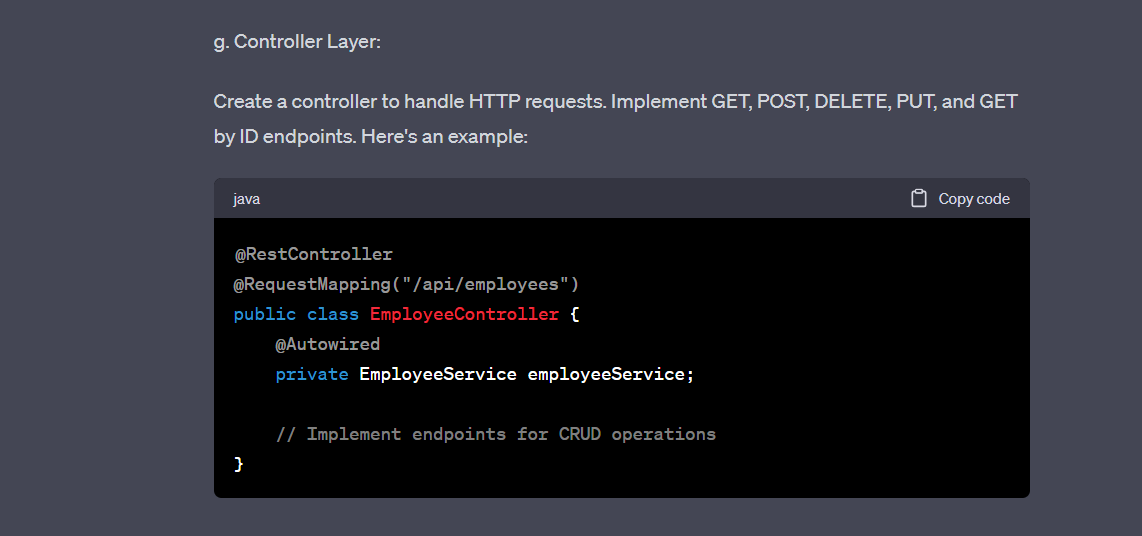


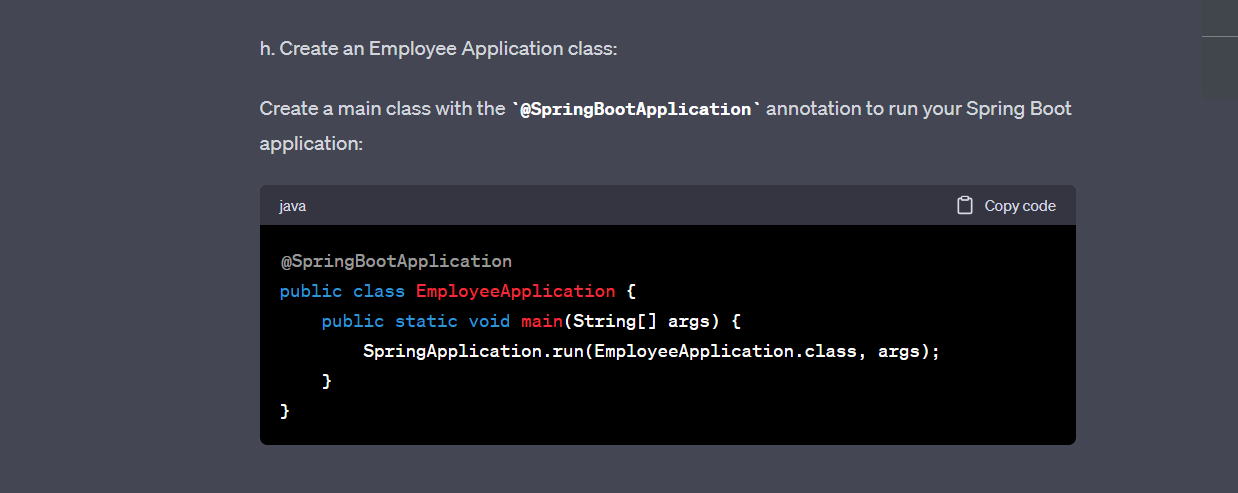
CREATE MODELS











i. Test Your API:

You can test your API using API clients like Postman or tools like **curl**. Use the endpoints defined in your controller to perform CRUD operations on employee data.

For example, to create a new employee:

* Send a POST request to **/api/employees** with JSON data containing employee details.

To retrieve all employees:

* Send a GET request to **/api/employees**.

To retrieve an employee by ID:

* Send a GET request to **/api/employees/{id}** where **{id}** is the employee's ID.

To update an employee:

* Send a PUT request to **/api/employees/{id}** with JSON data containing updated employee details.

To delete an employee:

* Send a DELETE request to **/api/employees/{id}** where **{id}** is the employee's ID.

Ensure that your controller handles these requests and responses properly, following RESTful conventions.

Remember to add proper error handling, validation, and security features as needed for a production-ready application.