Convert Website Design to Angular Project

Procedure:

1. Download Free Website Template (e.g., https://www.free-css.com)
2. Create Angular Project with Routing Option
   - ng new demo_14072022
3. Install and Configure Bootstrap
   - npm install bootstrap -save
   - Copy Reference Paths of bootstrap.min.css and bootstrap.min.js files in angular.json

     "styles": [
       "src/styles.css",
       "node_modules/bootstrap/dist/css/bootstrap.min.css"
     ],
     "scripts": [
       "node_modules/bootstrap/dist/js/bootstrap.min.js"
     ]
4. Copy Header Content in index.html
5. Copy Body Content in app.component.html
6. Copy CSS files into assets/css folder
7. Copy images into assets/images folder
8. Change Image Path wherever required in app.component.html
9. Run Angular Project

Creating Single Page Application (SPA)

Procedure:

1. Create Custom Components: home, category, product, deal and contact

   ng g c [component name]

2. Create Links in app.component.html and Configure Route Path in app-routing.module.ts
3. Add <router-outlet></router-outlet> tag in app.component.html
4. Cut and copy relevant content from app.component.html to corresponding components
5. Create Dynamic Content in product.component.html

   (Modify first product box and remove all other 7 boxes)

```
<div class="box" *ngFor = "let item of products">
            <span class="discount">{{item.discount}}</span>
            <div class="icons">
                <a href="#" class="fas fa-heart"></a>
                <a href="#" class="fas fa-share"></a>
                <!-- <a href="#" class="fas fa-eye"></a> -->
            </div>
            <img src="assets/images/{{item.image}}" alt="">
            <h3>{{item.name}}</h3>
            <div class="stars">
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star-half-alt"></i>
            </div>
            <div class="price"> {{item.discount_price}} <span>
{{item.original_price}} </span> </div>
            <div class="quantity">
                <span>quantity : </span>
                <input type="number" min="1" max="1000" value="1">
                <span> /kg </span>
            </div>
            <a href="#" class="btn">add to cart</a>
        </div>
```

6. Add Declaration with Data in product.component.ts

```
products = [
    {"discount": "-33%", "image" : "product-1.png", "name" : "organic
banana", "discount_price" : "$10.50", "original_price" : "$13.20"},
    {"discount": "-45%", "image" : "product-2.png", "name" : "organic
tomato", "discount_price" : "$10.50", "original_price" : "$13.20"},
    {"discount": "-33%", "image" : "product-3.png", "name" : "organic
banana", "discount_price" : "$10.50", "original_price" : "$13.20
  ]
```

Create and Build Grocery Store Database (MySQL)

Procedure:

1. Download, Install and Open MySQL Workbench
2. Connect MySQL workbench

   UserId: root

   Password: mysql
3. Create New Schema (Database): Grocery_Store
4. Create Table: Products

   Fields: ProductNo, Name, OriginalPrice, DiscountPrice, ImagePath

```
CREATE TABLE `products` (
  `ProductId` int NOT NULL,
  `Name` varchar(50) DEFAULT NULL,
  `OriginalPrice` float DEFAULT NULL,
  `DiscountPrice` float DEFAULT NULL,
  `ImageName` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`ProductId`)
)
```

5. Insert Record (Product Data)

```
INSERT INTO `store`.`products`
(`ProductId`,
`Name`,
`OriginalPrice`,
`DiscountPrice`,
`ImageName`)
VALUES
(1, 'Banana', 50, 40, '1.png');
```

Create Node.js Express Application

Procedure:

1. Create Node Project Folder.
2. Go to Project Folder and type code . command to open project in VS Code
3. Generate package.json:

   npm init -y (to create node workspace/ to setup node project)

4. Installing Express.js and its Dependencies:

   npm install express – Web Framework

5. Create index.js file and Write "Hello World" JavaScript Code

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

// Default Route
app.get('/', function (req, res) {
   res.send('<h1>Hello World</h1>');
})

// set port, listen for requests
const PORT = process.env.PORT || 8080;

app.listen(PORT, () => {
  console.log(`Server is running on port ${PORT}.`);
});
```

6. Run Project: node index.js
7. Open Browser and type NodeServer URL:

localhost:8080

8. Create GetProducts Method and Connect MySQL Database

- Install MySQL Library

  npm install mysql

- Import MySQL Library

```
var mysql = require('mysql')
```

- Create Connection

```
var connection = mysql.createConnection({
  host:'localhost',
  user:'root',
  password:'mysql',
  database:'store'
});
```

- Create Method

```
app.get("/products", function(req , res){
connection.query("SELECT * FROM store.products", function (err,
data) {
      if (err) return next(new AppError(err, 500));
          res.status(200).json({
          status: "success",
          length: data?.length,
          data: data,
      });
  });
});
```

9. Open Browser. Type localhost:8080/products

Add Http Service Class to Existing Angular Application

Procedure:

1. Create Service Component

   ng g s product

2. Import HttpClientModule in app.module.ts

```
import { HttpClientModule } from '@angular/common/http';
```

3. Add **GetProducts** method in product.service.ts

```
import { Injectable } from '@angular/core';

import { Observable, throwError } from 'rxjs';
import { HttpClient } from '@angular/common/http';

@Injectable({
  providedIn: 'root'
})
export class ProductService {

  constructor(private http: HttpClient) { }

  url:string = "http://localhost:8080/products";

  GetProducts(): Observable<any> {
    return this.http.get<any>(this.url)
  }

}
```

4. Call **GetProducts** method from product.component.ts