Angular 17 Development Environment Setup Exercise

1. Installing Node.js and npm (or Yarn)

a. Download and Install Node.js

- **Download:** Head over to nodejs.org and download the latest LTS version for your operating system.
- **Install:** Run the installer and follow the prompts.

b. Verify Installation

Open your terminal (or command prompt) and run:

```
node -v
npm -v
```

You should see version numbers for both Node.js and npm.

c. (Optional) Install Yarn

If you prefer Yarn over npm:

```
npm install --global yarn
```

Verify with:

```
yarn --version
```

2. Installing Angular CLI (Including NPX Usage)

The Angular CLI simplifies project creation and management. You have two choices: install it globally or use NPX to run it without a global installation.

Option A: Global Installation

Install the CLI for Angular 17 globally:

```
npm install -g @angular/cli@17
```

Verify by running:

```
ng version
```

Option B: Using NPX

If you'd rather not install Angular CLI globally, you can run it directly using NPX. This command uses NPX to run the latest Angular CLI to create your project without installing it globally. For Angular 17:

```
npx @angular/cli@17 new my-angular-app
```

3. Creating a New Angular 17 Project

Angular 17 brings some enhancements—like refined support for standalone components, improved build speeds, and stricter type checking when using the `--strict` flag. Use the Angular CLI to scaffold your project.

a. Basic Project Creation if Angular Installed

```
ng new my-angular-app
```

During the process you will have a few prompts:

- **Stylesheet Format:** Choose your preferred style (CSS, SCSS, etc.). For simplicity, you can choose CSS.

b. Using Additional CLI Flags

Here are some useful flags:

- **`--strict`**
  Enable strict mode to improve type safety and catch errors early

  ```
  ng new my-angular-app --strict
  ```

- **`--skip-tests`**
  Skip generating test files if you prefer not to have them

  ```
  ng new my-angular-app --skip-tests
  ```

c. Differences in Angular 17

Angular 17 builds on previous versions with several key improvements:

- **Standalone Components Enhanced:** While Angular 16 introduced standalone components, Angular 17 further refines their usability, reducing boilerplate code and enabling more flexible module-less development.
- **Improved Build Performance:** Optimizations in the Angular CLI now yield faster build times and a more responsive development server.

After project creation, navigate into your project directory:

```
cd my-angular-app
```

4. Exploring the Angular Project Structure and Files

Understanding your project structure is key to effective development. Here's a breakdown:

- **`src/` Directory:**
  This directory contains your application's source code.
  - **`app/`**: Contains Angular components, modules, services, etc.
  - **`assets/`**: Static assets like images and fonts.
  - **`environments/`**: Environment-specific configuration files (e.g., development, production).

- **Configuration Files:**
  - **`angular.json`**: Workspace configuration file. It controls build options, development server settings, and more.
  - **`package.json`**: Lists project dependencies, development scripts, and metadata.
  - **`tsconfig.json`**: Configures TypeScript compilation settings.

- **Development Server & Build Process:**
  - **Development:**
    Run the following command to launch a live-reloading server:

    ```
    ng serve
    ```
  -

Open your browser and visit http://localhost:4200 to see your app in action.

- **Production Build:**
  Build your project for production with:

  ```
  ng build --prod
  ```

  The output will be in the `dist/` folder.

5. Detailed CLI Commands and NPX Usage

For quick reference, here are some common CLI commands and NPX examples:

- **Using NPX for one-off commands (without global install):**

  ```
  npx @angular/cli@17 new my-angular-app
  ```

- **Serving the Application:**

  ```
  ng serve
  ```

  ```
  npx @angular/cli@17 serve
  ```

- **Generating Components, Services, etc.:** For example, to generate a new component:

  ```
  ng generate component my-new-component
  ```

  ```
  npx @angular/cli@17 generate component my-new-component
  ```

Or the shorthand:

```
ng g c my-new-component
```

```
npx @angular/cli@17 g c my-new-component
```

- **Running Tests:**

  ```
  ng test
  ```

- **Linting:**

  ```
  ng lint
  ```