# 个人作品集搭建笔记（React+Next.js+Tailwind 入门篇）

## 一、关键技术栈和工具

**React**：用于构建 UI 组件，现代前端必备框架；

**Next.js**：React 的服务器端渲染框架，支持路由、API 等，简化项目结构和开发流程；

**Tailwind CSS**：实用主义 CSS 框架，用类名快速写样式，省时高效；

**Node.js & npm**：项目环境管理与包依赖管理；

**Git & GitHub**：版本控制和远程代码托管，方便项目管理和展示；

## 二、项目搭建流程总结

**1.初始化项目目录与环境管理**

使用 nvm 切换 Node.js 版本，保证兼容现代开发工具；

使用 npm init 或 npx create-next-app 创建项目（本次为手动逐步补全依赖）；

**2.安装关键依赖**

安装 React、React DOM 和 Next.js，确保项目框架基础；

安装 Tailwind CSS、PostCSS 和 Autoprefixer，完成样式工具链搭建；

**3.配置 Tailwind**

运行 npx tailwindcss init -p 生成 tailwind.config.js 和 postcss.config.js；

在全局 CSS 文件（styles/globals.css）引入 Tailwind 指令 @tailwind base; @tailwind components; @tailwind utilities;；

**4.设置路径别名**

通过根目录下的 jsconfig.json 配置 @ 代表项目根路径，方便模块导入；

**5.编写入口页面**

在 pages/index.js 写 React 组件，使用 Tailwind 类名快速搭建 UI；

在 pages/\_app.js 中全局引入 CSS；

**6.启动项目验证**

运行 npm run dev 启动开发服务器；

浏览器打开 http://localhost:3000 查看效果，验证样式与组件渲染是否正确；

· **掌握完整的现代前端项目搭建流程**：从环境搭建、包管理、样式工具链配置到页面组件开发，具备从零搭建 React + Next.js 项目的能力。

· **熟悉 Next.js 约定目录结构和路由机制**，知道如何使用 \_app.js 统一引入样式和状态管理。

· **掌握 Tailwind CSS 的实用主义 CSS 思路**，能够快速用类名完成复杂样式，节省设计与调试时间。

· **理解模块路径别名配置，提升代码可维护性与导入效率**。

· **熟练使用 npm 和 nvm 管理依赖与 Node.js 版本，确保项目环境兼容性。**

· **拥有项目调试与错误排查能力**，解决了如 Tailwind 版本兼容、路径别名、npm 脚本缺失等问题。

· **具备 Git 版本管理基础，知道将项目同步到 GitHub 方便远程管理与展示。**