React Bootcamp – 7 Days Plan

Day	Topic / Concept	Project(s)	Key Points / Concepts Covered	Notes / Extra Focus / Tips
1	HTML, CSS, JS Basics → React Intro	Number Guessing Game & Word Scramble	JSX, useState, Events, Conditional Rendering, Basic Styling	Emphasize how React updates UI without full page reload; modularity, input handling
2	Component Structure & Props	Flashcard App	Components, Props, State, Mapping Arrays, Event Handling, Simple Styling	Show how props flow between components; modularization
3	Forms & Lists	To-Do List	State, Controlled Inputs, Adding/Deleting Items, Mapping Lists, Basic CSS	Explain controlled vs uncontrolled inputs; key usage for mapping lists
4	API Fetch & Async	Thirukkural App & Pokedex	Fetch API, Async/Await, Display API Data, Conditional Rendering, Styling	Handle loading/error states; show JSON data parsing; discuss combining multiple APIs
5	Games & Logic + Git Intro	Tic Tac Toe	State, Conditional Rendering, Event Handling, Game Logic, CSS Grid / Flex	Git basics: init, add, commit, push; branches, workflow; small live demo
6	State Management & Calculations	Expense Tracker	Adding Expenses, Categorizing, Calculations, Conditional Rendering, Array Mapping	Show component interaction, data flow, calculations, and filtering/sorting
7	Open Choice / Mini-Project + Hosting	Student Choice (from pre-selected options)	Consolidate concepts: State, Props, Events, API, Styling, Custom Features, Deploying Project	Deploy via Netlify/Vercel/GitHub Pages; folder structure, naming, comments

---

Project Structure Tips

Basic recommended folder structure:

```
my-project/
│
├─ public/          # HTML, favicon, static assets
│
├─ src/
│   ├─ components/  # All React components (cards, buttons, inputs)
│   ├─ pages/       # Page components (if using multiple pages)
│   ├─ hooks/       # Custom hooks (e.g., useRandomNumber)
│   ├─ services/    # API fetch functions
│   ├─ assets/      # Images, icons, JSON files
│   ├─ App.jsx
│   └─ index.jsx
│
├─ package.json
└─ README.md
```

Tips to tell students:

- Keep **one component per file**.
- Use **meaningful variable and function names**.
- Keep **state minimal and scoped** to where it's needed.

- Use **modular CSS** (or CSS modules) for reusable styles.
- Comment **only critical logic**, avoid over-commenting.
- Always check **console for errors/warnings**.
- Test each feature before moving to next component.
- Deploy **final project online** to see it live.

Naming Conventions

Type	Example	Notes / Usage
Camel Case	userName, handleClick	Variables, function names
Pascal Case	NumberGuessGame, FlashcardApp	Component names
Kebab Case	my-component.js, user-input.css	File names (JS/JSX/CSS files)
UPPER CASE	API_KEY, MAX_NUMBER	Constants
Folder names	components, hooks, assets	Lowercase preferred
State variables	const [count, setCount] = useState(0)	[value, setValue] pattern

---

Extra Tips for Students

- Keep projects **small & functional first**, styling later.
- Encourage **customization**: let students change colors, words, or layouts.
- Use **React DevTools** to inspect component state.
- Emphasize **JSX + HTML difference** and why React re-renders efficiently.
- Teach them **how to debug** (console.log, browser dev tools).
- Deploy **at least one project** to see the result live.