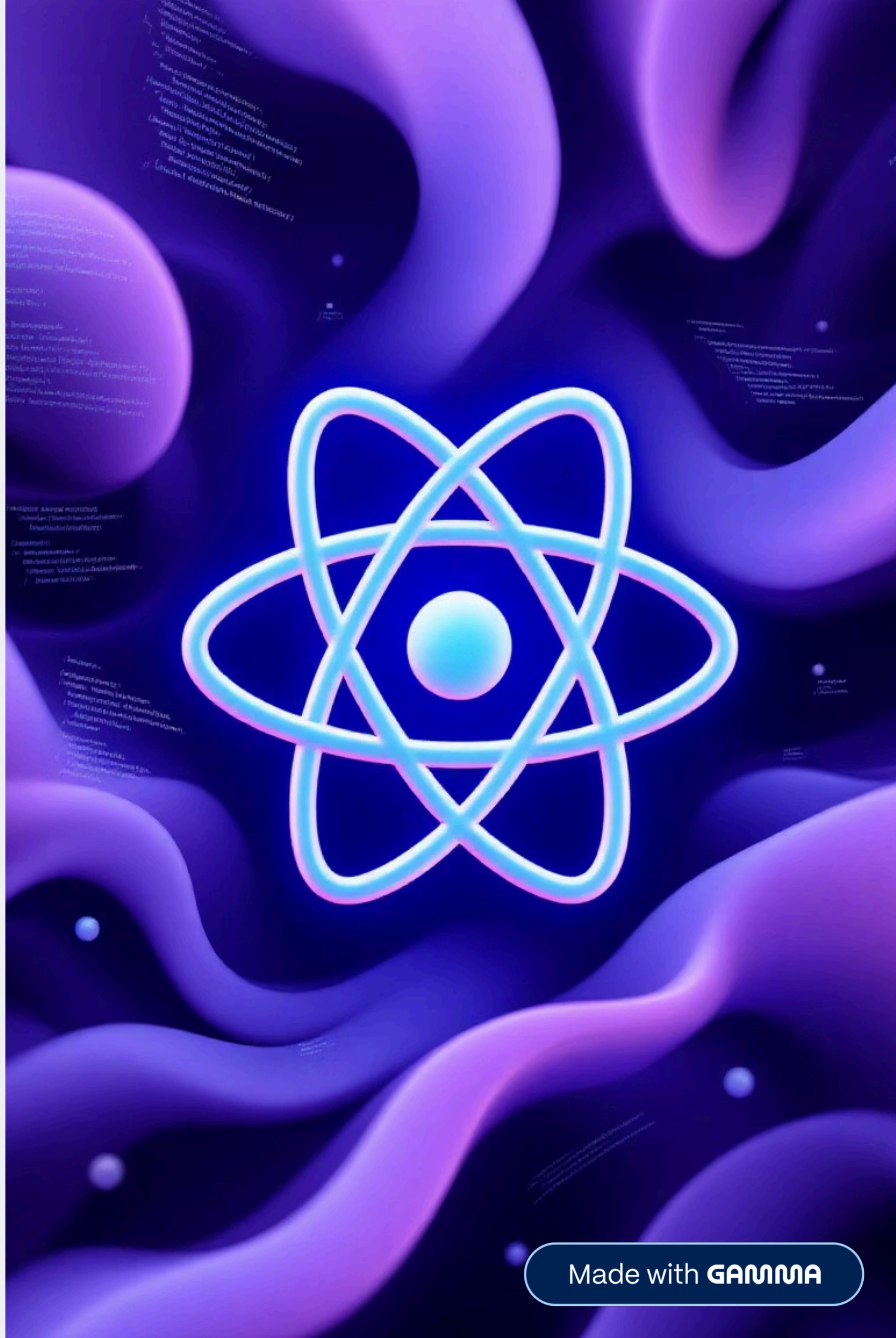
React: The Modern UI Library for Web Development

React is a powerful JavaScript library developed by Meta (Facebook) in 2013 that revolutionised how developers build user interfaces. As an open-source tool, it has become the framework of choice for creating dynamic, responsive web applications through its component-based architecture.



How React DOM Works: Virtual DOM

The Document Object Model (DOM) is the browser's representation of webpage elements as a tree structure. React introduces a clever optimisation with its Virtual DOM:

1 **Change Detection**

When data changes, React builds a new Virtual DOM tree

2 **Diffing**

React compares old and new Virtual DOM trees

3 **Reconciliation**

Only necessary changes are applied to the real DOM

This approach minimises costly DOM manipulations, resulting in significant performance improvements for complex applications.

React CLI & NPM Commands

1

Project Creation

```
npx create-react-app my-app
```

Creates a new React application with a predefined directory structure and configuration

2

Development

```
npm start
```

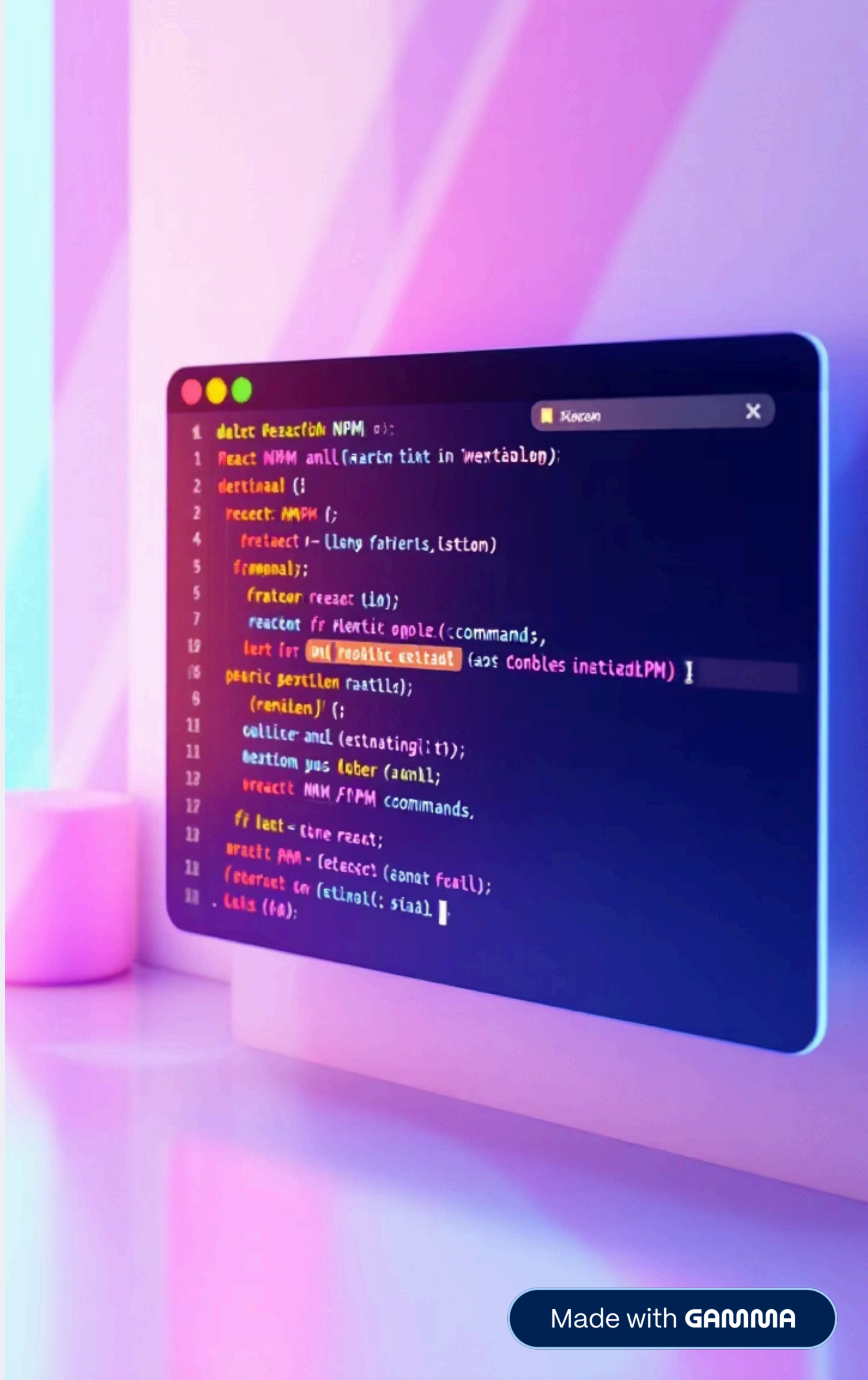
Launches the development server with hot reloading at localhost:3000

3

Building & Testing

```
npm run build
npm test
```

Creates optimised production files in build/ folder and runs test suites



React Components: The Building Blocks

1

Component Types

React offers two component types:

- Functional components (modern approach using hooks)
- Class components (traditional approach with lifecycle methods)

2

Component Properties

- **Props:** Read-only inputs passed from parent components
- **State:** Mutable data that affects component rendering
- **Context:** Way to share data across component tree

3

Component Lifecycle

- Mounting: Component appears in DOM
- Updating: Component re-renders due to state/prop changes
- Unmounting: Component removed from DOM

This component-based architecture promotes code reuse, maintainability, and testability in large applications.

JSX: JavaScript XML Syntax

JSX is a syntax extension for JavaScript that looks similar to HTML but provides the full power of JavaScript. It's transpiled to regular JavaScript before running in browsers.

- **HTML-like Syntax**

  Write declarative UI code that resembles HTML, making it intuitive for web developers

- **JavaScript Expressions**

  Use curly braces {} to embed JavaScript expressions directly within JSX

- **Attributes Become Props**

  HTML attributes transform into component props (e.g., className instead of class)

```
// JSX Example
function Greeting({ name }) {
  return (
```

Hello, {name}!

{name.length > 5 &&

You have a long name!

}

); }

JSX makes UI code more readable and maintainable while providing the full expressiveness of JavaScript.

Made with GAMMA

Project: Developer Profile Card

Implementation Steps

1. Create a new React project using create-react-app
2. Design the ProfileCard component with props for name, photo, skills, and social links
3. Style the component using CSS modules or styled-components
4. Add interactivity with React hooks (e.g., displaying more details on click)

```
function ProfileCard({ name, title, avatar, skills }) {
  return (
```

{name}

{title}

{skills.map(skill => {skill})}

); }