

Session 1 Instructor Guide: Setting Up Your Trivia Game

Learning Outcomes

By the end of Session 1, students will be able to:

1. **Embrace the developer role** at Wizcamp Enterprises and understand the course scope including the 12-session exploration and hands-on approach
2. **Explain what they are building** throughout the course including the trivia game features and development journey
3. **Define modern web development and fullstack** including component-based architecture, API integration, and frontend-focused fullstack approach
4. **Identify React's role** as a JavaScript library for building user interfaces with reusable components
5. **Compare development approaches** by building the same counter functionality using both Vanilla JavaScript and React
6. **Distinguish React's component-based architecture** from traditional HTML/CSS/JS approaches
7. **Launch a professional development environment** using GitHub Codespaces and VS Code
8. **Navigate project structure** and identify key folders and files
9. **Start a React development server** and understand the basic development workflow
10. **Modify React components** and experience Hot Module Replacement
11. **Identify development tools** and customize basic project settings
12. **Trace the React startup flow** and describe how a React app is bootstrapped

Instruction

Instructor introduces key concepts students need to succeed:

1. **Welcome to Wizcamp Enterprises** - Student scenario, developer role-play, and what students will experience as Junior Fullstack Developers
 2. **What We're Building** - Walkthrough of the finished trivia game to show the end goal
 3. **Modern Web Development & The Fullstack Approach** - Define modern web development and explain fullstack in context, including backend/frontend distinction
 4. **Meet React** - The 5 W's of React: what it is, who created/uses it, when it was released, where it's used, and why it matters
 5. **Old School vs. New School** - Counter comparison experience with both Vanilla and React sandboxes using StackBlitz
 6. **React vs. Vanilla JS** - Direct comparison based on the counter experiences: "You just built the same thing two different ways - under which tech stack would adding another counter be more straightforward?"
 7. **Your Development Setup** - Overview: "Codespaces gives you a complete coding environment in your browser - no downloads needed"
 8. **Anatomy of a React Project** - Project structure, folders, what are all these files, focus on understanding the big picture
 9. **The Magic of Live Updates** - Preview Hot Module Replacement: "When you change code, you'll see results instantly without refreshing"
 10. **Let's Code!** - Overview of today's mission and reference to SESSION-01 handout
 11. **Explore Your Toolkit** - Tools, architecture, development workflow, package.json, Vite, npm basics for future reference
 12. **Behind the Scenes** - How React apps bootstrap from HTML to components using carnival analogy
-

Slide Deck Outline

Slide 1: Welcome to Fullstack Explorer: Trivia Game Edition! 🎉

- **Title:** "Session 1: Welcome to React Development — Setting Up Your Trivia Game"
- **Student Scenario:**
 - **You've just been hired** as a Junior Fullstack Developer at Wizcamp Enterprises
 - **Creative tech company** known for bold ideas and playful spirit
 - **Your first assignment:** Join the Gaming Division to build Wizcamp's newest open-world trivia adventure

- **Role-playing experience:** Step into the shoes of a real developer
- **What You'll Experience:**
 - **Modern developer workflow** with GitHub Codespaces, VS Code, Vite, npm
 - **Real coding challenges** using React, JavaScript, and professional tools
 - **Problem-solving mindset** that defines modern web development
 - **AI-powered assistance** with tools like GitHub Copilot
- **Visual:** Wiscamp Enterprises logo with game screenshot and developer workspace
- **Hook:** "Experience what it's like to think and work like a developer while building something amazing"

Slide 2: What We're Building 🎮

- **Live Demo:** 2-minute walkthrough of completed trivia game
- **Key Points:**
 - **Interactive trivia game** across multiple themed zones
 - **Professional UI** that looks like apps you actually use
 - **Real trivia questions** pulled live from the internet
 - **Instant feedback** - know if you're getting it right or need to level up
- **Hook:** "You'll build this entire game over 12 sessions - step by step"

Slide 3: Modern Web Development & The Fullstack Approach 🌐

- **Title:** "What Makes Development 'Modern' and 'Fullstack'?"
- **Modern Web Development:**
 - **Component-based architecture** - Reusable, maintainable code
 - **Real-time interactivity** - Dynamic user experiences
 - **API-driven design** - Data from multiple sources
 - **Professional tooling** - Automated workflows and deployment
- **Fullstack in Context:**
 - **Frontend** - User interface and experience (our focus)
 - **Backend** - Servers, databases, APIs (we'll consume existing ones)
 - **DevOps** - Deployment, hosting, CI/CD (GitHub Pages, Actions)
 - **The "Full" Picture** - Understanding how all pieces connect
- **Our Approach:** "Frontend-focused fullstack - master the client side while understanding the complete ecosystem"
- **Visual:** Diagram showing frontend, backend, and deployment layers with emphasis on frontend

Slide 4: Meet React - The Heart of Modern Web Development ⚙️

- **Title:** "React: The 5 W's"
- **What:** JavaScript library for building user interfaces with reusable components
- **Who:** Created by Facebook (Meta), used by Netflix, Airbnb, Instagram, WhatsApp
- **When:** Released 2013, now the most popular frontend framework
- **Where:** Powers millions of websites, mobile apps (React Native), desktop apps
- **Why:** Makes complex UIs manageable, reusable, and maintainable at scale
- **Visual:** React logo with logos of major companies using React
- **Key Insight:** "React turns UI development from manual DOM manipulation into declarative component composition"

Slide 5: Old School vs. New School 🔥

- **StackBlitz:** "Instant development environment in your browser - no setup required!"
- **Lab Experience:** "Build the same counter using two different tech stacks"

- **Part 1 - Vanilla Counter - Sandbox:**
 - **Link:** <https://stackblitz.com/edit/wizcamp-vanilla-counter?file=README.md>
 - **Instructions:** "Open link, click 'Open Preview to the Side', complete README steps"
 - **Watch Closely:** "Observe the terminal and splash screen during spin-up - what's happening?"
- **Part 2 - React Counter - Sandbox:**
 - **Link:** <https://stackblitz.com/edit/wizcamp-react-counter?file=README.md>
 - **Instructions:** "Open link, click 'Open Preview to the Side', complete README steps"
 - **Watch Closely:** "Notice the different startup process - what tools are running?"
- **Key Questions:** "Under which tech stack would adding another counter be more straightforward? What did you notice about the project setup process?"
- **README Tip:** "Use 'Open Preview to the Side' to see formatted instructions while coding"

Slide 6: React vs. Vanilla JS

- **Reference Both Experiences:** "You just built the same thing two ways - let's compare"
- **Split Screen Comparison:**
 - **Left: Vanilla approach** (manual DOM updates, gets messy with scale)
 - **Right: React approach** (automatic updates, reusable components)
- **Key Insight:** "Adding a second counter: Vanilla = copy/paste mess, React = just `<Counter />`"
- **Analogy:** "Like **building with LEGO blocks** vs. **crafting everything from scratch**"

Slide 7: Your Cloud Development Setup

- **Visual:** GitHub Codespaces interface screenshot
- **Benefits:**
 - **Zero setup** - no downloads, no installs, no headaches
 - **Everyone gets the same** pro-level coding environment
 - **VS Code in your browser** - the editor real developers use
- **Reassurance:** "**Break stuff without fear** - just restart and you're back!"

Slide 8: Anatomy of a React Project

- **Visual:** Clean project tree diagram with file explanations
- **Key Focus:** "**What are all these files?** Let's decode your project structure"
- **Essential Folders:** `src/` (your code), `public/` (static files), `session-guides/` (your roadmap)
- **Key Files:** `package.json` (project info), `vite.config.js` (build tool), `index.html` (entry point)
- **Empowerment:** "**Focus on understanding the big picture** - you'll naturally learn where things are as you code"

Slide 9: The Magic of Live Updates

- **Demo:** Show HMR in action (change code → instant browser update)
- **Key Point:** "**Change code, see results instantly** - no refresh button needed"
- **Student Expectation:** "**You'll experience this** in the next 5 minutes"
- **Preview:** "It's pretty cool when you see it in action"

Slide 10: Launch Your Development Journey!

- **Your Mission Today:**
 1. **Launch** your cloud coding setup
 2. **Fire up** the development server
 3. **Swap out** a React component (StartHere → SplashScreen)
 4. **Experience** Hot Module Replacement in action

- 5. **Customize** your page title
- 6. **Chat with AI** for bonus points
- **Important Note:** “Follow the provided **SESSION-01 handout** - you'll find session-guides inside your Codespace once it launches”
- **Connection:** “You'll see the same instant updates you experienced in both sandboxes - but now in a real project!”

[HANDS-ON WORK HAPPENS HERE]

Slide 11: Explore Your Toolkit 🛠️

- **Visual:** Complex development architecture diagram (Codespace infrastructure)
- **Purpose:** “I'm showing you this **detailed diagram** so you can **refer back to it later** when you're curious about how everything connects”
- **Key Tools:** Vite (build tool), npm (package manager), VS Code (editor)
- **Essential Commands:** `npm run dev` (start server), `npm install` (get dependencies)
- **Student Approach:** “**Don't try to memorize this** - just know this diagram exists for when you want to understand the full picture”

Slide 12: Behind the Scenes: From HTML to React 🎪

- **Title:** “Curious how React brings apps to life? This diagram walks you through the bootstrapping process—from loading a barebones HTML file to rendering a fully interactive app.”
- **Visual:** React bootstrap flowchart diagram + Carnival analogy
- **Purpose:** “Once initialized, React takes over and manages the DOM for your entire app”

The Journey in a Nutshell (with walkthrough):

1. **HTML Entry Point** - A single browser DOM node—an empty container—is defined in `index.html`. Think of it as a blank canvas.
2. **React Root Injection** - In `main.jsx`, a script creates a React root element (the top-level UI manager) and injects it into that empty container.
3. **Component Tree Rendering** - The App component—wrapped in a `GameProvider` (shared memory for game state)—along with its child components (buttons, text, images, etc.), is rendered into the root. This is where your UI and user interactions come to life.
4. **Boom! Your Game Appears** - Your interactive trivia game is now visible on the page. From this moment on, React handles all updates, interactions, and DOM changes seamlessly.

🎪 Think of it Like Setting Up a Carnival:

- `<div id='root'>` in `index.html` = Empty field
- `main.jsx` = Carnival trucks arrive
- Bootstrapping = Rides go up, booths get built, lights get wired
- `GameProvider` = Main power grid powering all attractions
- App renders = Gates open—carnival is live and buzzing
- **Student Approach:** “**Focus on the carnival analogy today** - the technical diagram is there when you're ready to dive deeper”

```

---
config:
  layout: elk
  look: neo
---
flowchart TB
  subgraph Bootstrap["⚡ React Bootstrap Process"]
    FindContainer@{"🔍 Locate empty container<br>document.getElementById('root')"}
    PrepareReact@{"🔧 Initialize React renderer<br>ReactDOM.createRoot()"}
    RenderWrappedApp@{"📄 Render app with context<br>GameProvider → App.jsx<br>render()"}
  end
  end
  Browser["🌐 Browser loads page"] --> IndexHTML["📄 index.html<br>Includes empty container + script reference"]
  IndexHTML -- Waiting for React to initialize --> EmptyDiv@{"👉 <div id='root'><br>React app is live"}
  EmptyDiv --> MainJSX --> Bootstrap
  Bootstrap --> FindContainer
  FindContainer --> PrepareReact
  PrepareReact --> RenderWrappedApp
  RenderWrappedApp -- Final result appears in --> EmptyDiv
  EmptyDiv --> MainJSX
  classDef files fill:#e3f2fd,stroke:#2196f3,stroke-width:3px
  classDef container fill:#fff3e0,stroke:#ff9800,stroke-width:2px
  classDef bootstrap fill:#e8f5e9,stroke:#4caf50,stroke-width:2px
  classDef files
  classDef container
  classDef bootstrap

```

Slide 13: What's Next - Building Your Own Components 🧩

- **Title:** "Preview of Session 2"
- **Today's Foundation:** "You experienced React's component system by swapping existing components"
- **Next Challenge:** "Create your own reusable components from scratch"
- **Concepts Coming:**
 - **Component creation** - Build custom GameButton component
 - **Props** - Pass data and behavior between components
 - **JSX mastery** - Write dynamic markup with curly braces
 - **Styling variants** - Create flexible, reusable UI elements
- **Motivation:** "Your buttons will be truly yours - custom-built and infinitely reusable!"
- **Visual:** Preview of GameButton component with different variants