Component-wise Visual Diagram:

We'll represent the following components:

1. **App Component**
2. **GameBoard Component**
3. **Player Component**
4. **Log Component**

We'll also highlight the relationships, how data flows between components, and how the state is managed.

---

1. App Component:

- **Purpose**: This is the main container. It manages the game state (`gameTurns`), determines the active player, and passes necessary data to child components.
- **State**: `gameTurns` (array of moves).
- **Logic**: Calculates active player using `deriveActivePlayer()` .
- **Passes props**:
  - To `GameBoard` : `gameTurns` , `handleSelectSquare` .
  - To `Player` : `activePlayer` , player symbols (X or O).
  - To `Log` : `gameTurns` (move history).

---

2. GameBoard Component:

- **Purpose**: Displays the game board with clickable squares.
- **Receives Props**: `gameTurns` , `handleSelectSquare` (to handle user interaction).
- **Interaction**: When a square is clicked, `handleSelectSquare` is triggered, which updates `gameTurns` .

---

3. Player Component:

- **Purpose**: Displays the current player's information and highlights the active player.
- **Receives Props**: `symbol` , `isActive` (whether the player is active or not).

4. Log Component:

- **Purpose**: Displays the history of the moves made so far in the game.
- **Receives Props**: `gameTurns` (history of moves).

Component-wise Visual Diagram Representation:

plaintext Copy Edit

```
+------------------------------+ | App Component | |------------------------------| | -
Holds state: `gameTurns` | | - Determines active player | | - Passes `gameTurns` and | |
`activePlayer` to children | | - Passes `handleSelectSquare` | +--------------+----------
------+ | v +---------------------------+ | GameBoard Component | |--------------------
-------| | - Receives `gameTurns` | | - Receives `handleSelect` | | - Displays game board
| | with clickable squares | +------------+---------------+ | (square click triggers) v
+---------------------------+ | handleSelectSquare | |---------------------------| | -
Updates `gameTurns` | | - Calls `deriveActivePlayer`| +------------+---------------+ |
(triggers state update) v +---------------------------+ | setGameTurns | |-------------
--------------| | - Adds the new move to | | `gameTurns` | +------------+---------------+
| (triggers re-render) v +---------------------------+ | Log Component | |-------------
--------------| | - Receives `gameTurns` | | - Displays move history | +----------------
-----------+ | v +---------------------------+ | Player Component | |------------------
---------| | - Receives `symbol` | | - Receives `isActive` | | - Highlights active player
| +---------------------------+
```

Explanation of Flow:

1. **App Component**:
   - Holds the **game state** (`gameTurns`).
   - **Determines the active player** using `deriveActivePlayer(gameTurns)` .
   - Passes down relevant props to the `GameBoard` , `Player` , and `Log` components.
2. **GameBoard Component**:
   - **Displays the game board** using the `gameTurns` state (which tracks all moves).
   - Each square is clickable, and when clicked, it triggers `handleSelectSquare(row, col)` to register the move.
3. **handleSelectSquare**:
   - This function is called when a square is clicked.

- It updates the `gameTurns` state with the new move (square and player).
- It also determines the **next active player** using `deriveActivePlayer`.

4. **setGameTurns**:
   - This function updates the `gameTurns` state with the new move.
   - React triggers a re-render of the components: **App**, **GameBoard**, and **Log**.
5. **Log Component**:
   - Displays the **move history** from the `gameTurns` state.
   - It shows all the moves made so far (which player played which square).
6. **Player Component**:
   - Receives the **player's symbol** (X or O) and the **active player status**.
   - Highlights the currently active player by showing the symbol and marking the player as "active".

Key Data Flow:

- **App** → **GameBoard**: Passes `gameTurns` and `activePlayer`.
- **GameBoard** → **handleSelectSquare**: When a square is clicked, this function is triggered to update `gameTurns`.
- **handleSelectSquare** → **setGameTurns**: Updates the game state with the new move.
- **setGameTurns** → **App, GameBoard, Log**: Triggers re-rendering to reflect the updated state.
- **App** → **Log**: Passes `gameTurns` to display the move history.