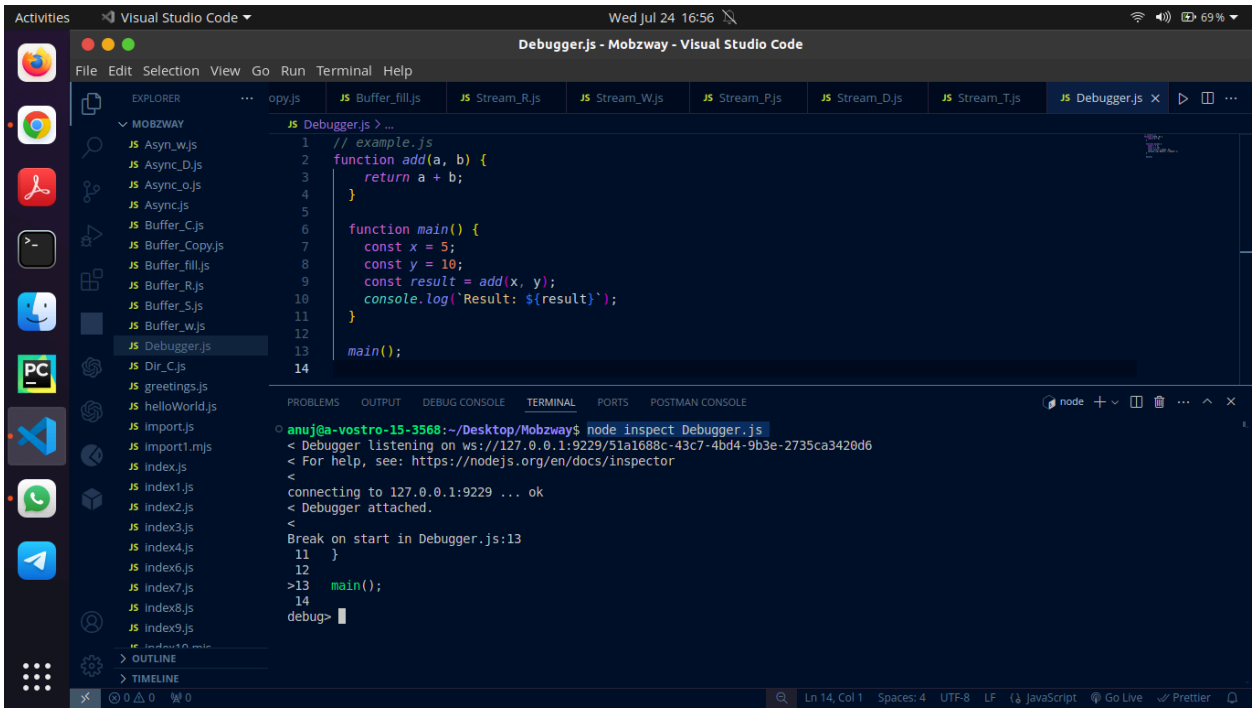
Node.js Documentation

Module 8: Debugging Node JS Application

- **Core Node JS Debugger**

  The core Node.js debugger allows you to debug your Node.js applications directly from the command line. This is useful for inspecting your code, setting breakpoints, and stepping through your code to understand its behavior. The debugger can be accessed using the built-in `inspect` command.



1. Start the Debugger: Run your script with the `inspect` flag.

```
node inspect example.js
```

2. **Running and Stepping Through Code**:

- `cont` or `c`: Continue execution until the next breakpoint.
- `next` or `n`: Step to the next line of code.
- `step` or `s`: Step into a function call.
- `out` or `o`: Step out of the current function.
- `repl`: Open a REPL session to inspect variables and execute code.

3. Setting Breakpoints:

- `setBreakpoint()` or `sb()`: Set a breakpoint at a specific line.
- `clearBreakpoint()` or `cb()`: Clear a breakpoint.

- ## Callbacks: Error-first

Error-first callbacks are a common pattern in Node.js and JavaScript programming, especially when dealing with asynchronous operations. This pattern ensures that the first argument of the callback function is reserved for an error object (if any error occurs), and subsequent arguments are used for successful results. This approach helps to consistently handle errors and makes the code more predictable and easier to maintain.

Structure of an Error-first Callback