# Data Plan

The project requires several categories of data to function effectively. Primarily, the system will rely on user-provided input in the form of natural language commands, which will be entered through a command-line interface. This data forms the basis for interpreting the user's intent and translating it into actionable development environment setup instructions. Additionally, system-level metadata such as the underlying operating system type, installed software packages, configuration files, and current system state will be programmatically retrieved using Python's standard libraries (such as platform, os, and subprocess) or native shell commands. This metadata is essential for ensuring that the commands generated are compatible with the user's specific machine configuration.

To support the core logic of the application, the system will use a large language model (LLM) accessed via the OpenAI API. This LLM will process user input along with system metadata to generate shell commands, file modifications, or installation procedures. The output from the LLM serves as a critical intermediary data source that guides the agent in automating the environment setup process. Furthermore, as part of MCP (Model Context Protocol) integration, simulated or actual project metadata such as dependency lists and project-specific tooling configurations will be utilized. These may be acquired from structured configuration files or stubbed API responses and are used to customize and optimize the environment for specific use cases.

For evaluation purposes, we will generate a curated set of test commands and their expected outputs, which will act as the benchmark for validating system performance and correctness. This evaluation data will be manually crafted and versioned to ensure repeatable testing.

In summary, the following data types will be required and used:

* User input: Natural language commands provided via CLI.
* System metadata: OS info, installed packages, current configurations (retrieved locally).
* LLM output: AI-generated shell instructions based on user/system context.
* Project metadata: MCP-defined configurations, dependency lists (from files or stubs).
* Evaluation data: Manually created test inputs and expected outputs for validation.

All data will either be interactively provided, locally inspected, fetched via APIs, or manually constructed. These data sources collectively support the development, functionality, customization, and evaluation of the AI agent as defined by the project scope.