ChatGPT Prompt Engineering To-Do List App

1. I prompted ChatGPT to create a simple python to-do list application. The initial functionality of the app includes a display menu that offers the user 4 choices.
   * 1. View tasks
   * 2. Adding a new task
   * 3. Removing a task by the number it correlates to
   * 4. Exiting the app
2. The first prompt engineering I applied to ChatGPT is Enhanced Input Validation.
   * I specifically prompted ChatGPT to enhance the input validation in the remove\_task function. After carefully reviewing the code, I saw that the review\_task function had exception handling applied since it prompted the user to remove a task by deleting the task using the number the user inputted from the list. The reason why the first prompt engineering I applied was the enhanced input validation, was to help prevent any input errors from the user and the application crashing due to it. This prompt engineering strategy also handled an error occurring due to due to a user attempting to remove a task when the task list is already empty. This improves the app by improving user experience. The improved exception handling would provide the user with concise outputs on why certain entries are not permitted such as using letters when an integer is required. This is due to separating the exceptions to provide the user with a corrective action that reduces repeated errors rather than grouping them together with same error messages.

**Before**A screenshot of a computer program

Description automatically generated

**After**

A screenshot of a computer program

Description automatically generated

1. The second prompt engineering strategy I applied to ChatGPT was to improve user interface.
   * The reason why I decided to utilize this prompt engineering strategy was to improve interactivity with user by displaying more information. I asked ChatGPT to improve the user interface of the application by adding more confirmation messages and formatting the task display in a cleaner fashion. I the view\_task function was improved by including a header and a total task count. The add\_task and remove\_task functions now print success messages after the user completes an action. The main reason I chose to improve the user interface was to make the application more user-friendly and positively influence a user’s overall experience with the app. The functionality of the application was already top-notch and robust, so improving the UI would increase a user’s engagement, satisfaction and ultimately serve as an error prevention as well.

**Before**

A screenshot of a computer program

Description automatically generated

**After**

A screenshot of a computer program

Description automatically generated