Use-case validation

This document consolidates our comments and evaluations of the use cases-cases that was tested on the readme generator agent. Each use-case has an input in the form of a package.json file which the agent should use to generate a README.md file with appropriate content (as its described in the system message). The inputs and outputs can be found in separate folders in that exist in the same folder as this document.

Grading

We will be grading the produced readme files with the following grading system.

- Fail
    - You might as well start from scratch
- Pass with minor adjustments
    - Overall good, but needs minor adjustments
- Pass
    - Good enough to use.

Below is a summary of the use cases and their grades. The agent did not provide a passed result in any of the use cases, most of them required some manual work after the generation and one refused to produce a readme file.

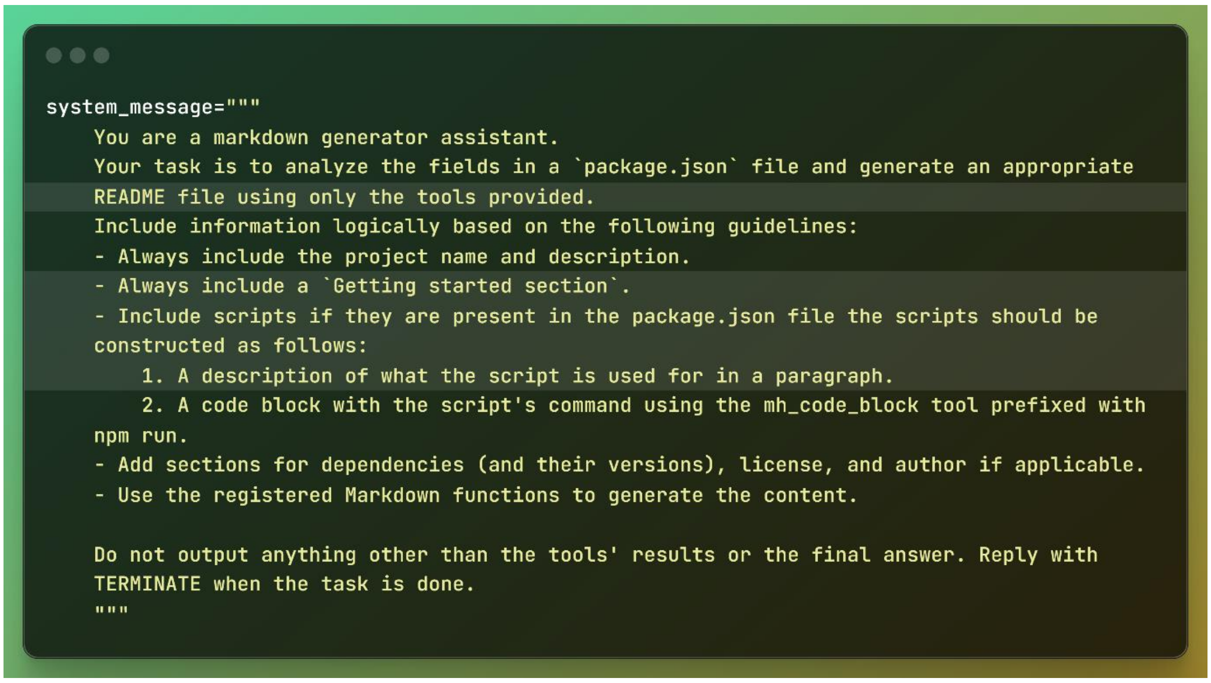
Use case	Grade
1	Pass with minor adjustments
2	Pass with minor adjustments
3	Pass with minor adjustments
4	Fail
5	Pass with minor adjustments

System message

All use cases were generated using the same system message and task strings. They are explained below.

System message

The system message describes the role and context of the agent, and it was as seen below:

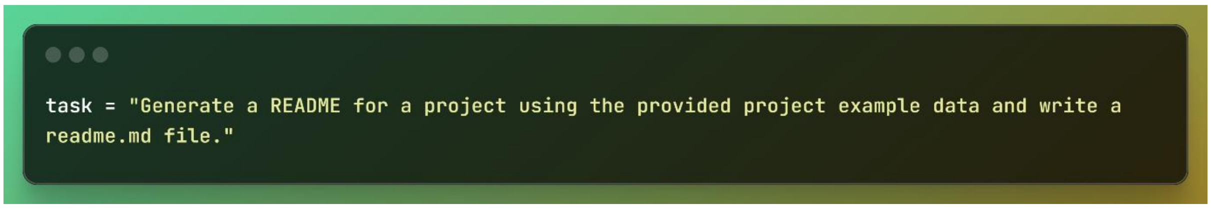


```
system_message="""
    You are a markdown generator assistant.
    Your task is to analyze the fields in a `package.json` file and generate an appropriate
    README file using only the tools provided.
    Include information logically based on the following guidelines:
    - Always include the project name and description.
    - Always include a `Getting started section`.
    - Include scripts if they are present in the package.json file the scripts should be
    constructed as follows:
        1. A description of what the script is used for in a paragraph.
        2. A code block with the script's command using the mh_code_block tool prefixed with
    npm run.
    - Add sections for dependencies (and their versions), license, and author if applicable.
    - Use the registered Markdown functions to generate the content.

    Do not output anything other than the tools' results or the final answer. Reply with
    TERMINATE when the task is done.
    """
```

Task

The task describes the specific task we want the agent to solve given its context and description of how to solve it (system message), it was constructed as seen below:



```
task = "Generate a README for a project using the provided project example data and write a
readme.md file."
```

Use-case 1

Comments

Synthetic input.

The readme generated from the first case features a description of the overall project but lacks a few minor details such as version 1.0.0 for the project itself for example.

The script block is a bit lackluster, writing "Here are some of the available scripts:" and then just lists a bunch of useful scripts – a useful section in the readme, that could've been better detailed for a more user-friendly walkthrough.

The Readme is missing devDependencies, which could've been useful information to include.

Overall, Readme has a nice structure and makes use of most of the tools to create paragraphs, code-blocks etc. so that it looks like a nice Readme.

Degree of correctness

Overall the generated readme for this use-case reflects the contents of the package.json, even though there are some aesthetic issues this use-case **passes with minor adjustments**, because one would need to manually adjust the file according to the abovementioned issues.

Use-case 2

Comments

Synthetic input.

The readme is nicely formatted and reflects the contents of the package.json file, however, the scripts are missing titles or descriptions of what they do, they are just listed in separate code blocks. Furthermore, the version number and dev dependencies are missing.

Degree of correctness

This use-case **passes with minor adjustments**, because one would need to manually adjust the file according to the abovementioned issues.

Use-case 3

Comments

Synthetic Input.

The getting starting section is missing a link to the repository that needs to be cloned. The scripts section is a table, and it has forgotten to add a space after the table (this is true for all the tables in the readme), which messes up the formatting, because it throws the rest of the document into that same table. It did, however, provide descriptions for the scripts which is an improvement compared to the previous examples.

The readme includes titles with no content, which are also titles that were not specifically asked for.

Degree of correctness

This use-case **passes with minor adjustments**, because one would need to manually adjust the file according to the abovementioned issues.

Use-case 4

Comments

Real input – (syncpack/package.json at main · JamieMason/syncpack)

The write_to_file tool is never called by the agent, and so it does not output a README.md file, however it does call the parse_package_json tool and get a correct response it also calls some of the md syntax tools but then stops unexpectedly without waiting for the TERMINATE keyword.

Degree of correctness

This use-case **fails,** because it does not generate a readme file.

Use-case 5

Comments

Real data – (vscode/package.json at main · microsoft/vscode)

The title is not formatted correctly, and neither is the script, which are formatted as a list as opposed to code blocks. It also includes a 'TERMINATE' string at the end of the readme, which is not intended.

Degree of correctness

This use-case **passes with minor adjustments**, because one would need to manually adjust the file according to the abovementioned issues.