**Shape, rectangle

Description automatically generated**



**AI Companion – POC**

**Oct 2025**

**Version 1.0**

Table of Contents

[1.1 Objective 3](#_Toc211355619)

[1.2 Team & Timeline 3](#_Toc211355620)

[1.3 AI Companions leveraged 3](#_Toc211355621)

[1.4 Environment Setup 3](#_Toc211355622)

[1.5 Existing Code Structure and Setup 4](#_Toc211355623)

[1.6 Scenarios & Outcomes 5](#_Toc211355624)

[1.7 Devin Session Summary 13](#_Toc211355625)

[1.8 Framework Enhancement Scenario 14](#_Toc211355626)

[1.9 AI Augmented Software Development 19](#_Toc211355627)

[1.9.1 Team Structure 19](#_Toc211355628)

[1.9.2 Training the Workforce 19](#_Toc211355629)

[1.9.3 Artifacts Maintained and Leveraged 19](#_Toc211355630)

[1.9.4 Prompt Engineering Aligned to Agile Process 19](#_Toc211355631)

[1.9.5 AI Companion Aligned to Agile Process 20](#_Toc211355632)

[1.9.6 Prompt Template 21](#_Toc211355633)

[1.10 Conclusion 22](#_Toc211355634)

**‘**

## Objective

To explore the capability of various AI Companions tools to handle complex enhancements on large, highly customizable bespoke applications and demonstrate the developer productivity gained through their adoption.

## Team & Timeline

|  |  |
| --- | --- |
| **The Team** | |
| **Name** | **Role** |
| Subhash Bhaskaran | AI Engineer |
| Murugan Sundaram | Data Hub SME |
| Viswanatha R | Data Hub SME |

**Start Date: 1st Oct 2025**

**End Date: 10th Oct 2025**

## AI Companions leveraged

1. Devin
2. Github Co Pilot

## Environment Setup

1. Developer Laptop
   * Github
   * Any IDE like IntelliJ or VS Code with Co Pilot plugin enabled
   * Devin web subscription
2. Code
   * Python based ingestion framework
   * Any data, excel reconciliation code from public domain, cloned locally

## Existing Code Structure and Setup

**GitHub Location**

https://github.com/ltibfspoc/devinpoc

**Summary**

The code is data processing ETL application built using PySpark for distributed data transformation and Luigi for workflow orchestration. The application supports transformation like filter, sort, merge, etc. The application is configurable using json template to define data pipeline.

Devin was then asked to understand the code structure and build its knowledge. Devin using this knowledge then was asked to work on different scenarios.

**Structure**

A screenshot of a computer

AI-generated content may be incorrect.

**Code Size**

|  |  |
| --- | --- |
| **Folders** | 17 under src |
| **Total Lines of Code** | 1285 python code |

## Scenarios & Outcomes

|  |  |
| --- | --- |
| **Scenario 1** | Requirement to create a new ingestion job.  Tool Plan Evaluation:   * able to identify all the necessary changes to the job and configuration files * able to identify the transformation logic changes in configuration files * ability to use existing ingestion job(s), that can be used as base for this new requirement |
| **Prompt** | Using the Test1 branch, perform the below new requirement  1) Create a new pipeline by name "test51.json" to read the csv file "factbook.csv" from the "data" folder.  2) After reading operation, filter the records with condition "Area < 300"  3) Put the output in folder "factbook300" in the folder "outputfile".  4) You can use “test1.json” as an example to understand the pipeline setup  5) Run the ingestion job by changing the run\_pipeline.py |
| **AI Companions Used** | Devin |
| **Outcome** | Devin was able to understand the configuration required to develop the new ingestion job and ensure only the relevant code change was performed. |
| **Evidence** |  |
| **Learnings** | * Providing the context by selecting the repo and the branch reduces the time spent on searching and hence ACU * Breaking up the prompt into individual steps instead one big paragraph helps to refer them later, to provide any feedback * Giving examples of code to study, reduces the time spent searching and hence ACU |

|  |  |
| --- | --- |
| **Scenario 2** | Requirement to create new test cases based on the requirement and code generated for scenario 1  In the Prompt provide requirements of the new ingestion job  Tool Plan Evaluation:   * able to identify all the necessary scenarios for which testing is required (Happy Path, Edge Case, Boundary Condition) * Ability to create plain text Test Case, with the steps to be performed to execute and validate the Test Case. |
| **Prompt** | Using the Test1 branch performs the below new requirement  1) Write new python code to test the pipeline “test51.json” ingestion job  2) create the new code in a new folder called “testautomation” under “src”  3) The new code should take the source file and output file path and filename as parameter  4) Test Case scenario 1: The new code should verify the row count match between source and target file based on the filter condition  5) Test Case scenario 2: The new code should verify if all records in output file have Area < 300  6) Test Case scenario 3: The new code should verify the sum of CurrentAccountBalance match between source and target file based on the filter condition  7) The new code should create an excel file called “result51” in a new folder called result" under "src" containing the results of this test. Sheet 1 is summary, Sheet 2 should indicate each column tested for count or sum depending on the data type. |
| **AI Companions Used** | Devin |
| **Outcome** | Devin was able to create the test automation script and create the result in the format specified. |
| **Evidence** |  |
| **Learnings** | * While Devin was able to create the output excel based on the specification, it can also be provided with excel template and asked to fill the same. |

|  |  |
| --- | --- |
| **Scenario 3** | Requirement to create new test cases based on the requirement and code generated for scenario 1  In the Prompt provide requirements of the new ingestion job  Tool Plan Evaluation:   * able to identify all the necessary scenarios for which testing is required (Happy Path, Edge Case, Boundary Condition) * Ability to create plain text Test Case, with the steps to be performed to execute and validate the Test Case. |
| **Prompt** | try harder following task using Test1 branch  1 create test cases as per the source code for test51.json provided in the repo  2 write the automation code in python for the test cases created in step 1  3 execute the automation code and provide the output of test case in excel format and put in the result folder |
| **AI Companions Used** | Devin |
| **Outcome** | Devin identified 13 new test case and updated the existing result excel |
| **Evidence** |  |
| **Learning** |  |

|  |  |
| --- | --- |
| **Scenario 3** | Generating Technical Documentation  In the Prompt provide the location of the Github and the describe the format of the Documentation  Tool Plan Evaluation:   * able to incorporate all critical points from codebase * able to summarize in easy-to-understand language for developers * Documentation should follow the format that can be copy paste as readme file. |
| **Prompt** | try harder to access the "https://github.com/ltibfspoc" and repo name is "devinpoc" and provide me the documentation and steps to run this python code |
| **AI Companions Used** | ChatGPT enabled as Agent mode and provided access to Github repository |
| **Outcome** | Read the GitHub repository and complied content in concise format with tables and diagram. |
| **Evidence** |  |
| **Learnings** | * Using Wiki generated by Devin is comprehensive but not easy to read * Using Copilot or ChatGPT produce easy to read and more concise document |

## Devin Session Summary

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Summary** | **Session Size** | **ACU** |
| 1 | Create pipeline for filtering | Small | 2.4 |
| 2 | Python test pipeline ingestion | Small | 1.4 |
| 3 | Create architectural diagram Test1 branch | Extra Small | 1.2 |
| 4 | Suggest new transformations | Extra Small | 0.4 |
| 5 | Create and automate test cases | Small | 2.4 |
|  | **Total Consumed** |  | **7.91** |

## Framework Enhancement Scenario

**Discussion with CoPilot on opportunities to enhance the framework.**

A screenshot of a computer

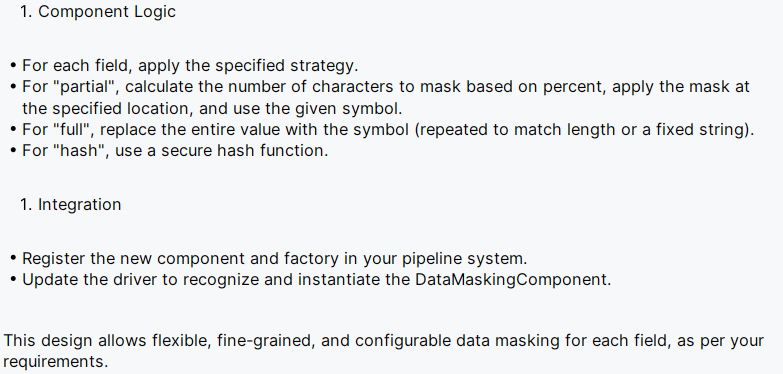
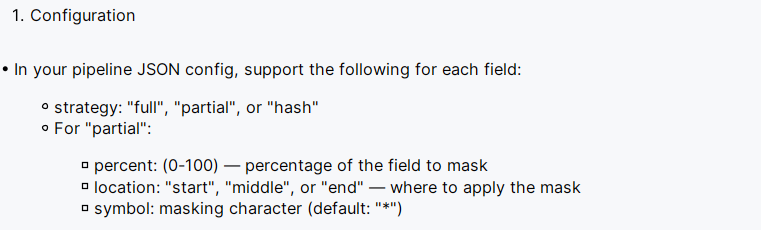
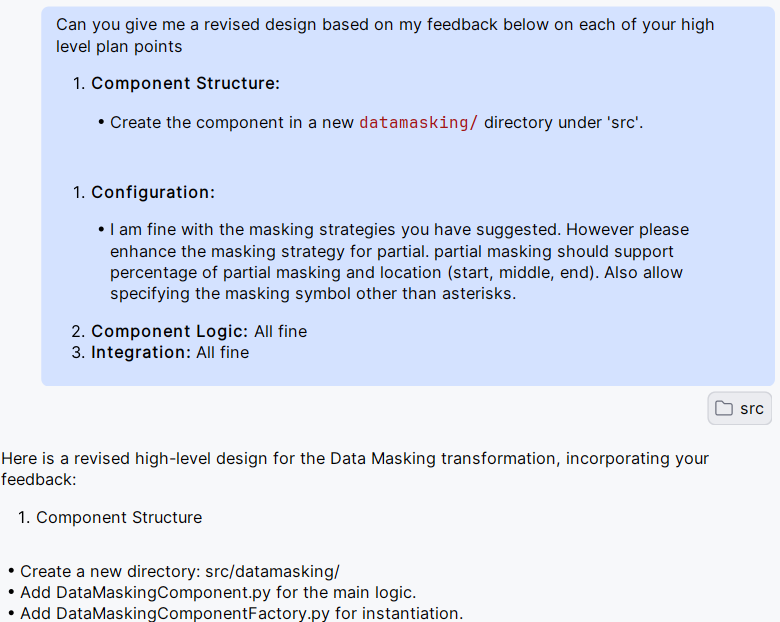
AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

## AI Augmented Software Development

This section describes the various facets of the new engagement model

### Team Structure

### Training the Workforce

Courses provided by LTIM Shoshin School (online self paced courses)

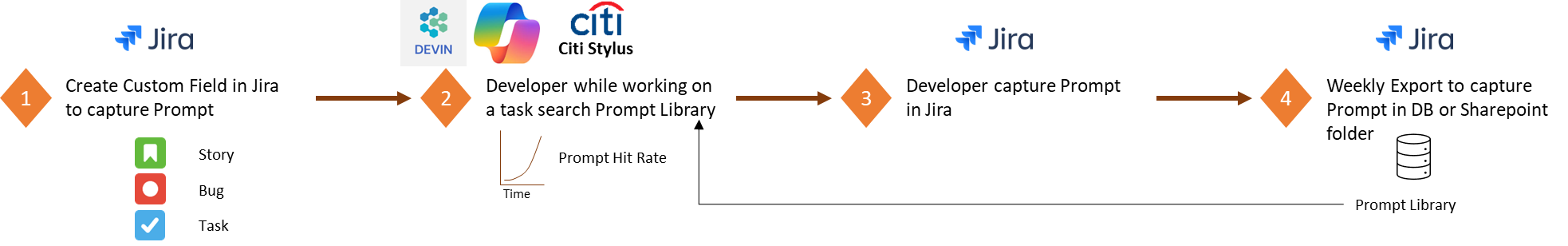
1. Generative AI: A Primer for Beginners (4 h, mandatory)
2. Getting Started with Prompt Engineering with Generative AI (49 min, mandatory)
3. LTIM – Gen AI Copilot Training (5 h, mandatory)
4. Generative AI- ChatGPT Prompt Engineering for Developers (1h 30min, mandatory)
5. Prompt Engineering: ChatGPT & AI Prompts For Work Success! (2h 22min, optional)
6. Devin AI Tutorial - 2025 | How to Use Devin AI? (Step-by-step Guide) (optional – Youtube)

### Artifacts Maintained and Leveraged

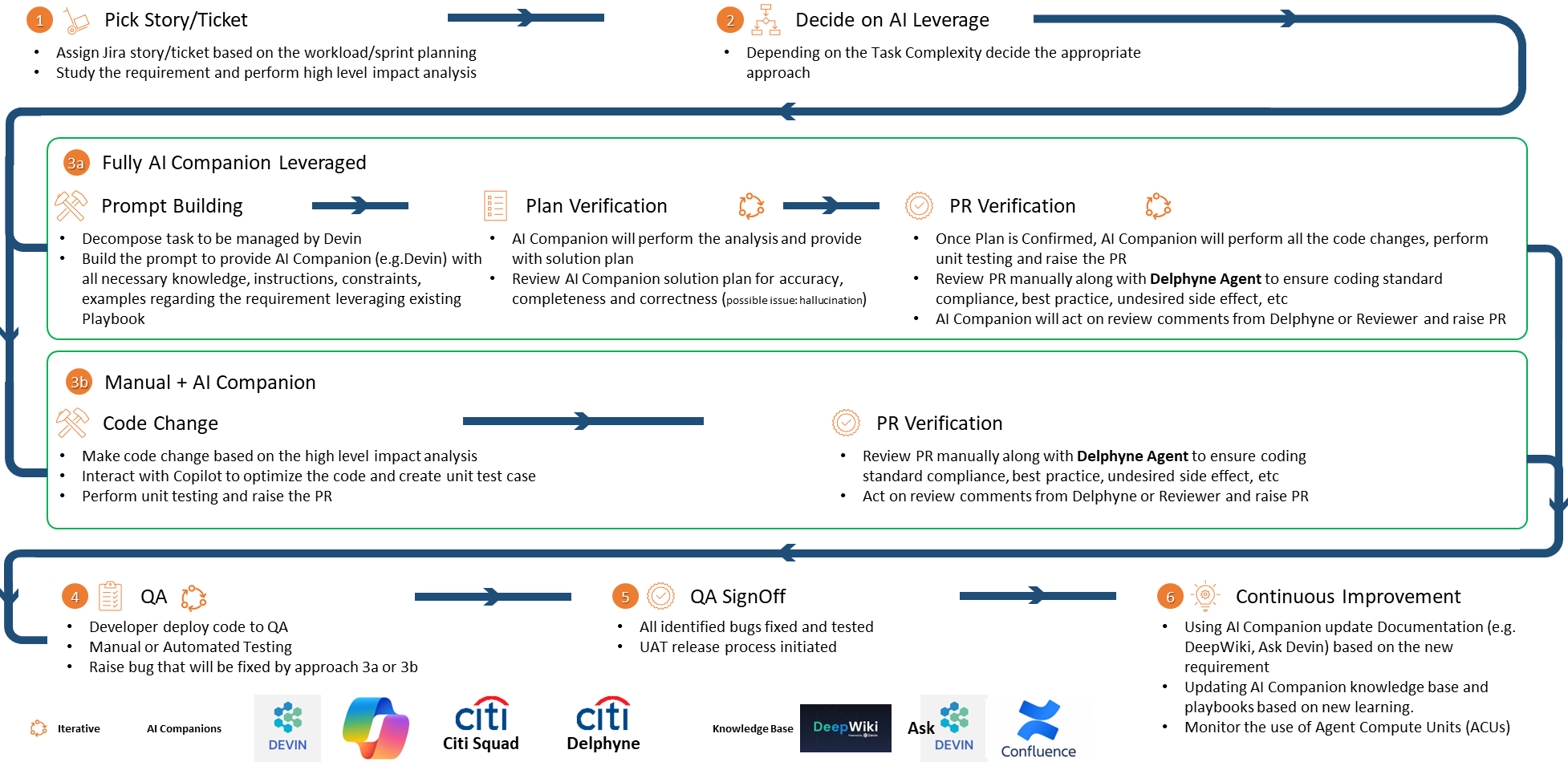
1. Knowledge Base – Confluence and Devin Wiki Page
2. Prompt Library – Playbook. Template for recurring tasks and detailed instructions on current project knowledge reference, example code snippet, compliance, naming standards reference, code quality metrics benchmark, etc.
3. Weekly meeting to review token usage and document new learnings and update Prompt Library as needed.

### Prompt Engineering Aligned to Agile Process

1. Developer will leverage Prompt Template based on the task instead of creating prompt from scratch
2. Every Story, Bug, Test Case in Jira will have a prompt field to be filled by developer



### AI Companion Aligned to Agile Process



### Prompt Template

**Role:**

Code generating scenario use the role is Expert <mention the language or area example Java , Oracle, etc> Developer,

Testing scenario mentioned role as Senior Software Tester,

High level design or exploring new ideas use the role as Software Architect

Code Review or Critique use role as Code Reviewer

Statement to start or add in Prompt

As an Expert Software Developer, write a function to …..

**Specifications**: Describe requirement in simple numbered steps. Avoid open-ended or ambiguous statements.

**Required from User**: Describe any input or information required from the user, example the repo or branch name or jira id

**Example**: Include tips to guide Devin’s learning curve, example code snippet or file name along with the scenario to use the example

**Forbidden Actions**: Include any action Devin should not take. Example merge PR, changes to any specific files, upgrade any library to latest version, skip execution and only perform analysis, skip execution of test cases, use of EOL library or module

**Advice**: Include tips to correct Devin’s priors. Example documentation link to coding standards, CI/CD, tech stack, versions

Please add the below text to all your code generating prompts

Please identify and add error handling

Please add detailed comments to all your code. Highlight any complex part that might require deeper explanation.

## Conclusion

* Devin is like instructing a junior developer or intern, hence more detailed instructions increase the probability of achieving the desired outcome. Reverting changes in PR done by Devin could be time consuming
* GitHub CoPilot appears to be good for more exploratory tasks like impact analysis, design approaches, etc. The same in Devin can consume more ACU and the detail might be too much to go over
* Do not combine multiple tasks or activities in a single session. This helps with better monitoring of ACU and recall
* Prompt Library is key so that developers do not spend time writing repeating instructions. Cross learning among team members is important. Hence this should be topic of discussion in retrospective ceremony

