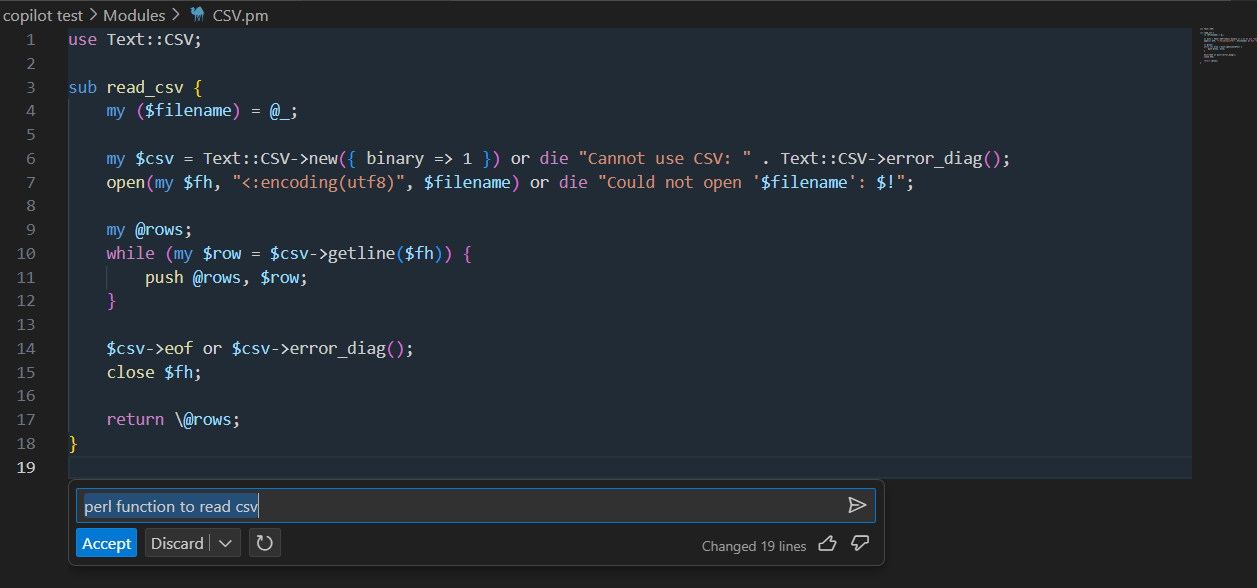
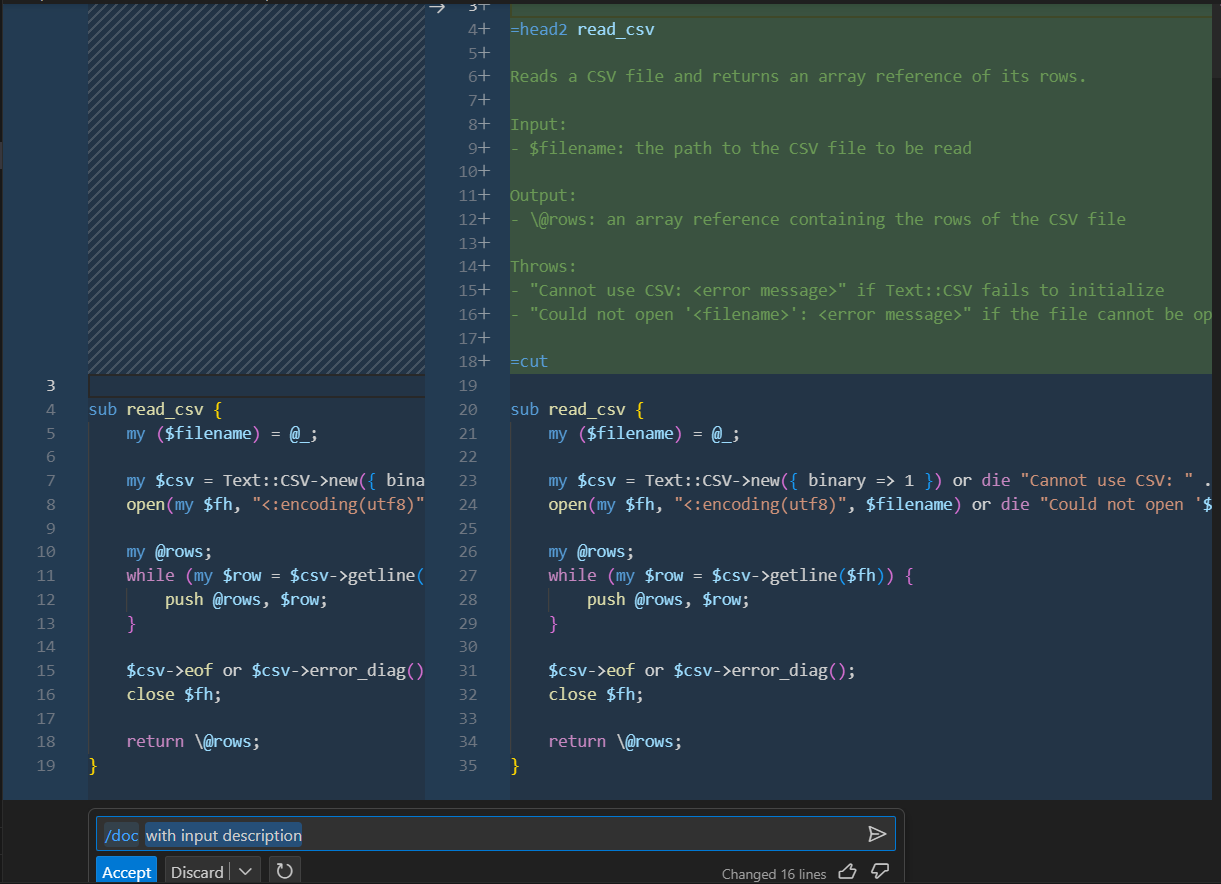
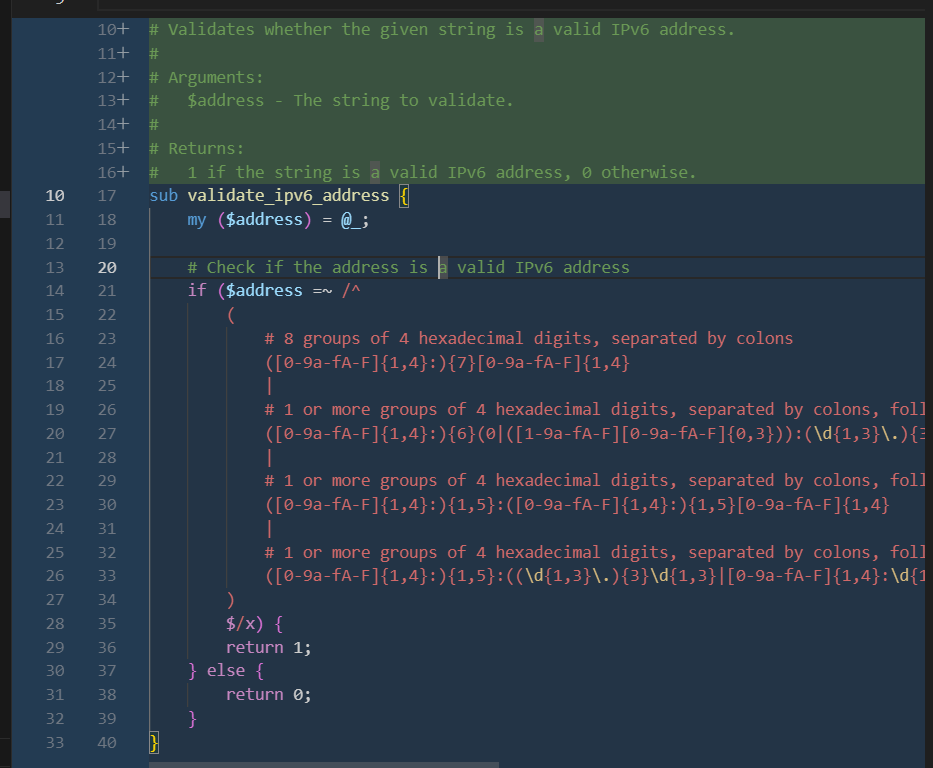
1. Co-pilot generated : **perl function to read csv**



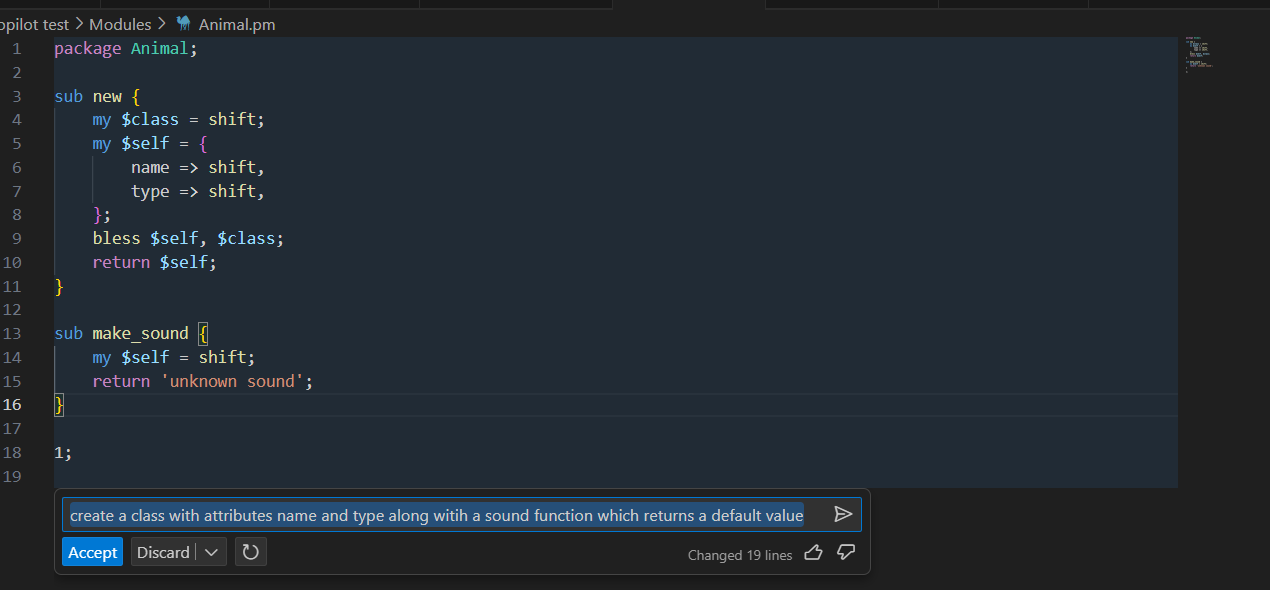
1. Documentation of the function



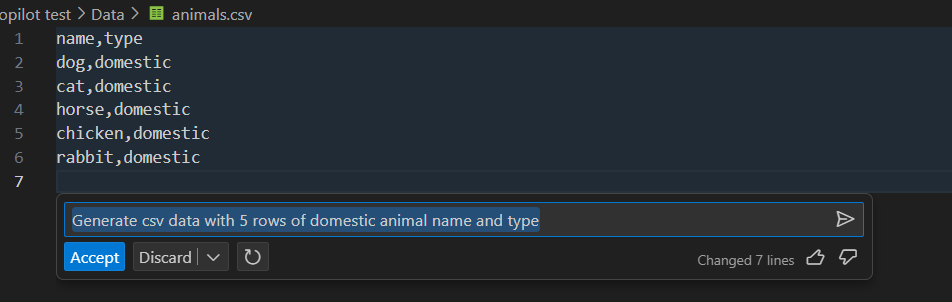
Some times generates exactly the way we usually do in perl like the below example



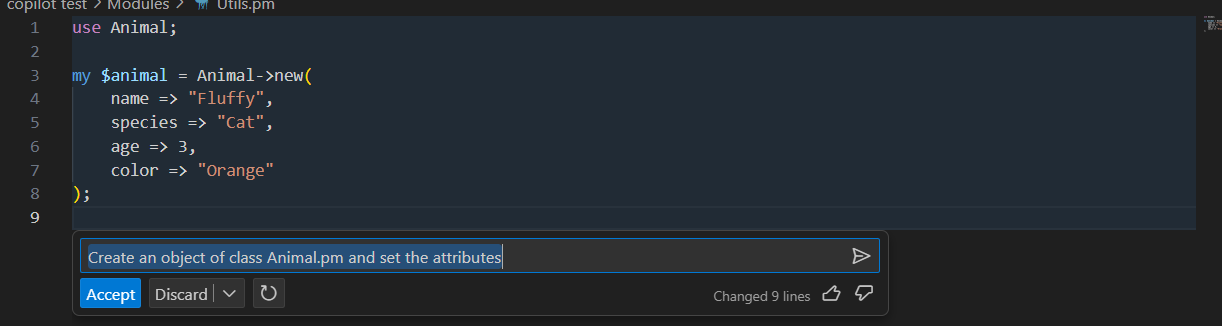
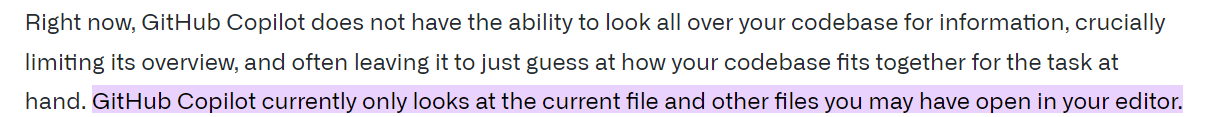
1. Class creation



1. CSV data generation

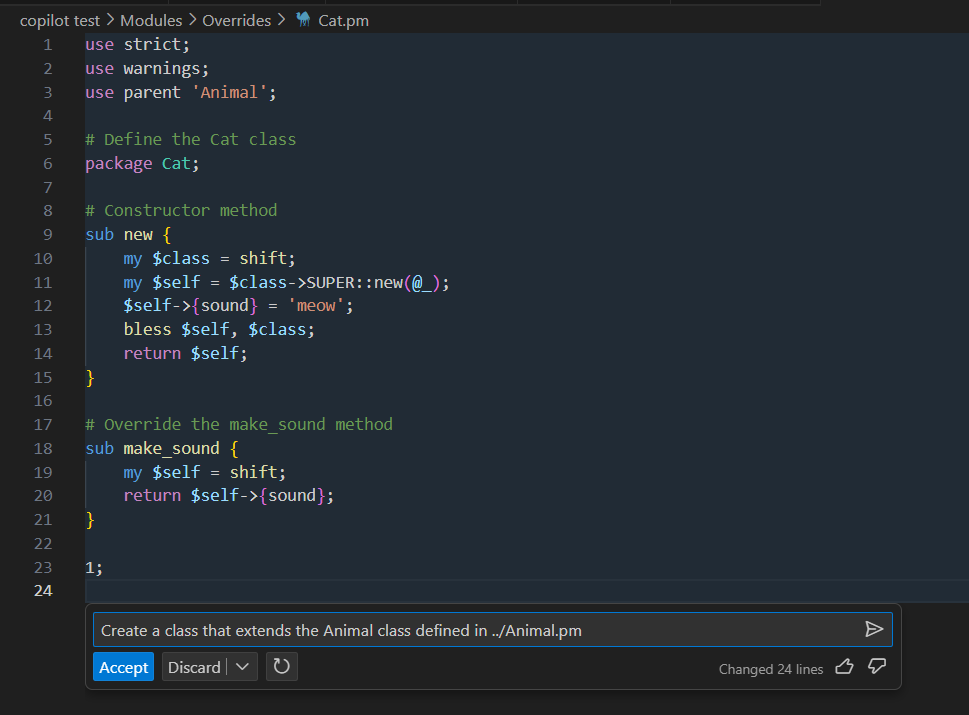
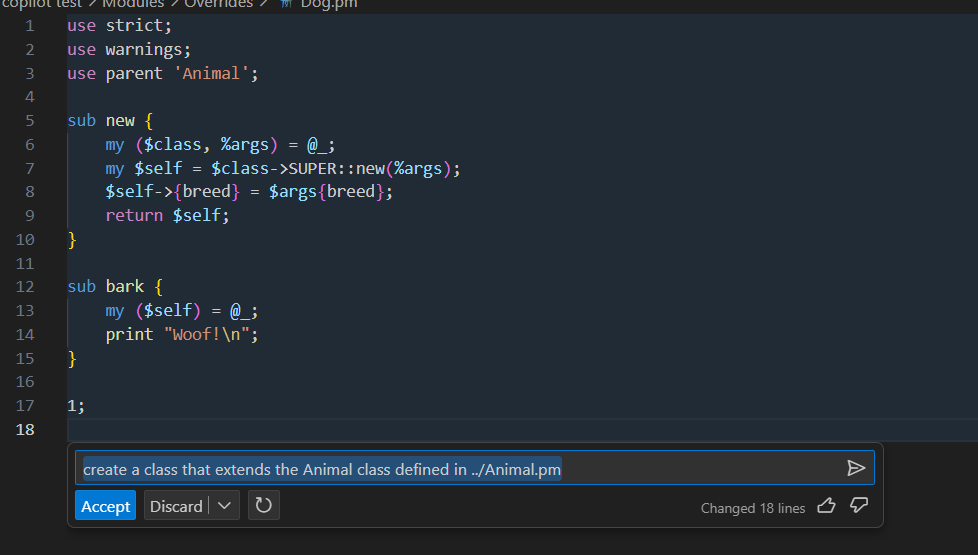


1. Object creation for the previously defined class

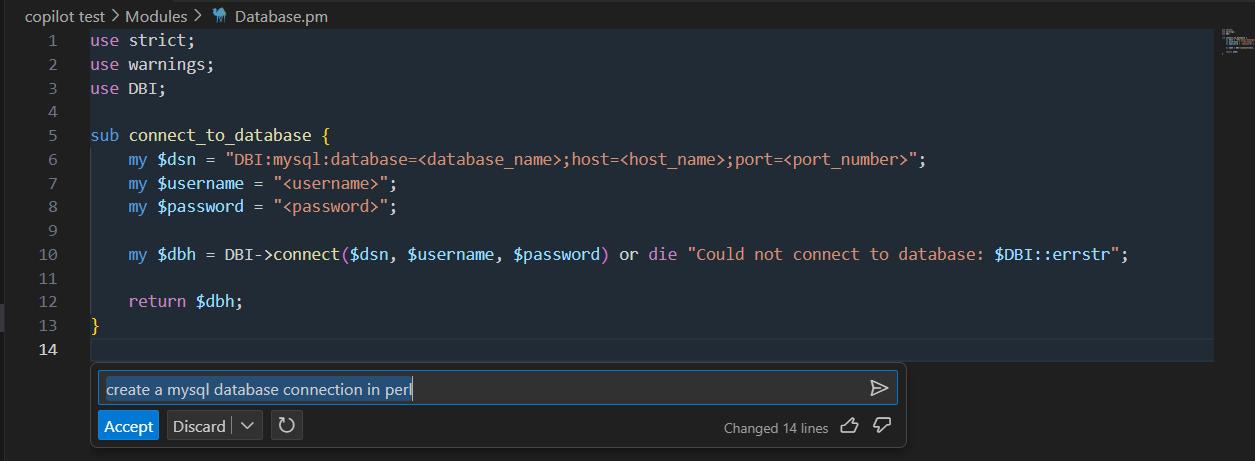
  
  


<https://githubnext.com/projects/copilot-view/>

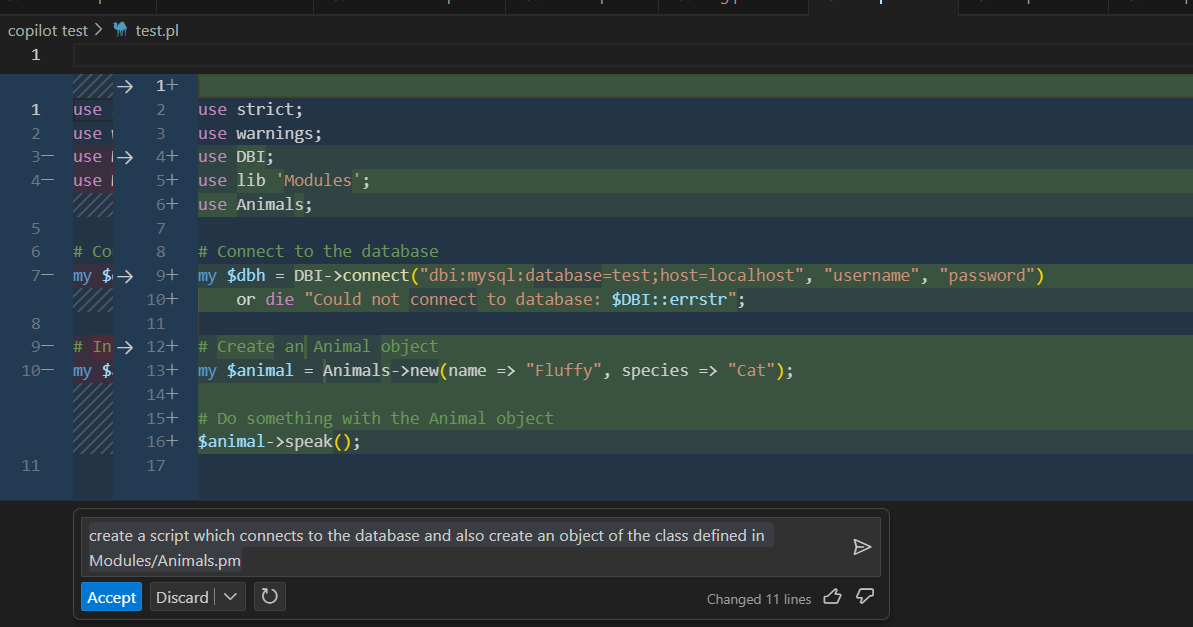
1. Overriding the Animal class

  
  
  
  
Same prompt doesn’t always generate the same type of code

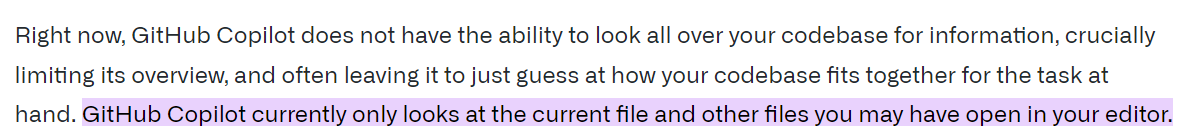
1. Database connection



1. Script : Level 1

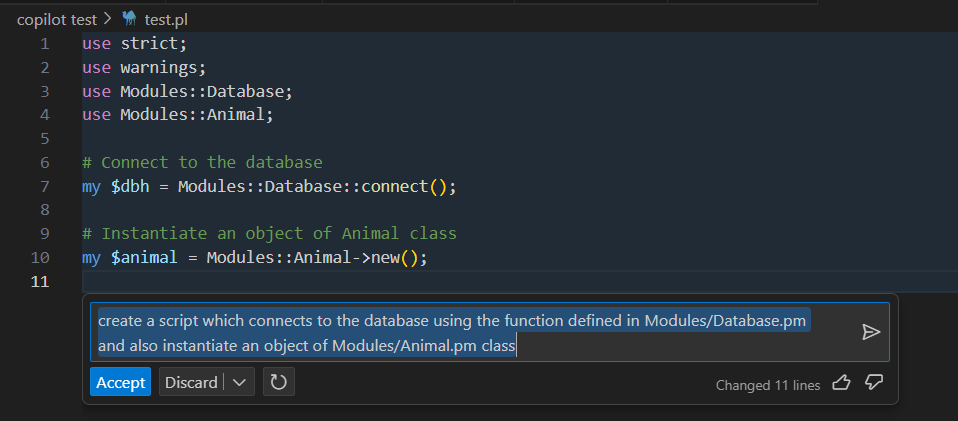


Creates the database connection code from scratch and also assigns the wrong attributes to the Animal Object



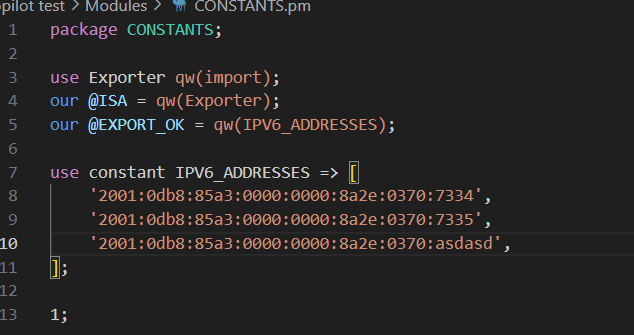
<https://githubnext.com/projects/copilot-view/>

1. Script : Level 2

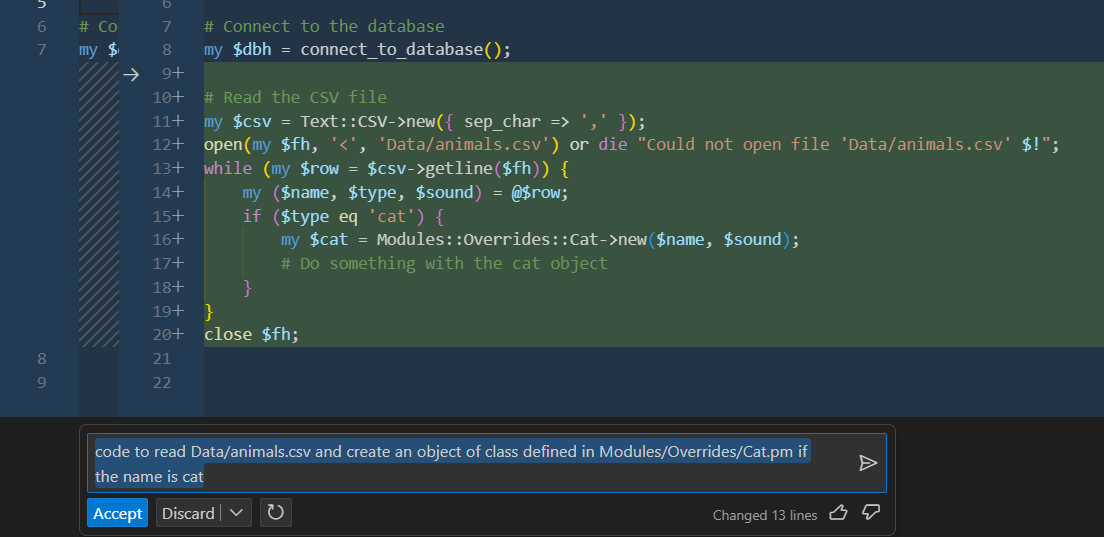


Even after explicitly mentioning the Database connection file in prompt. Still, it doesn’t pick up the existing function even though it imports the module.

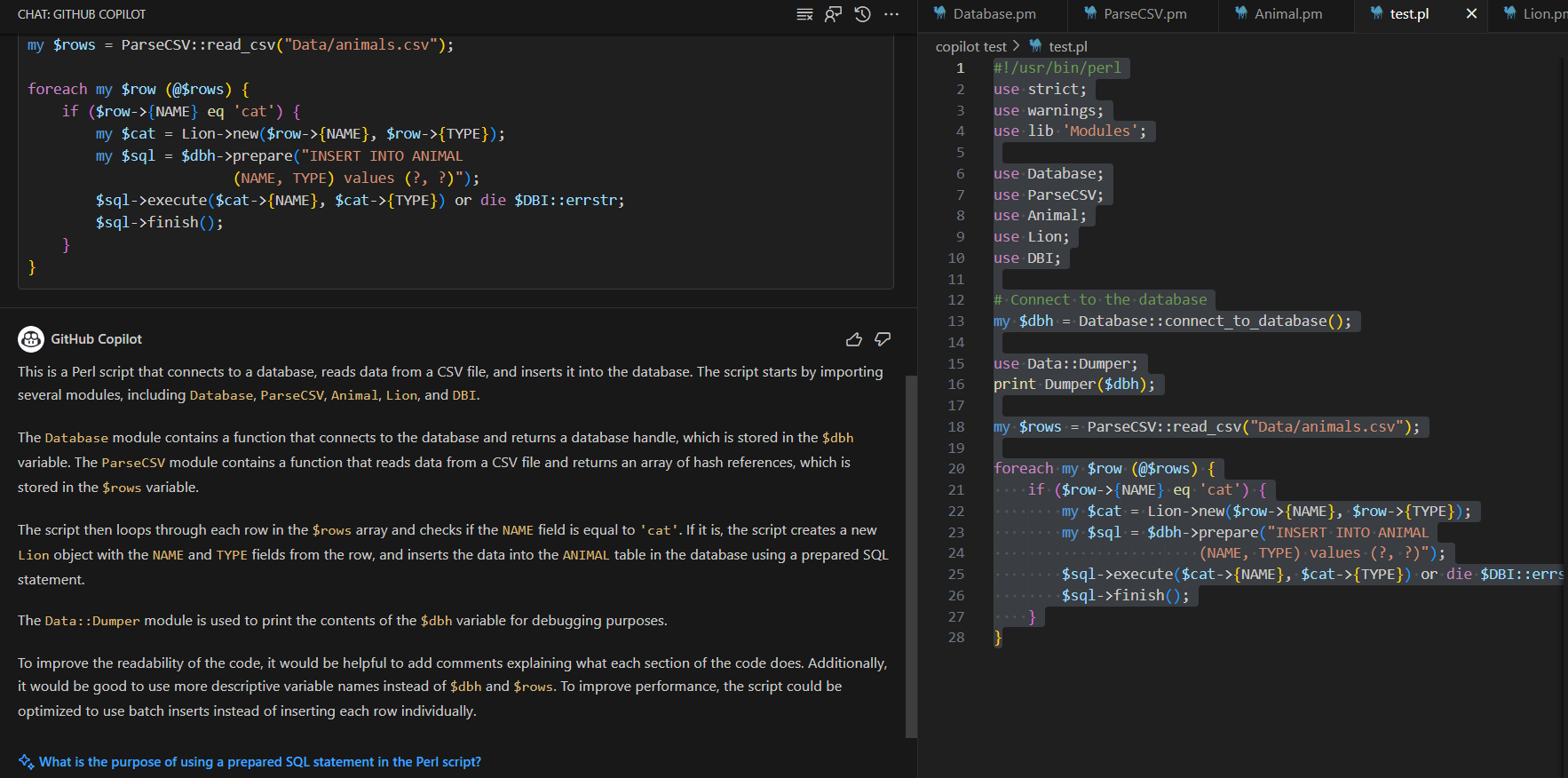
1. Constant export creation : Created the constant. But missed the package

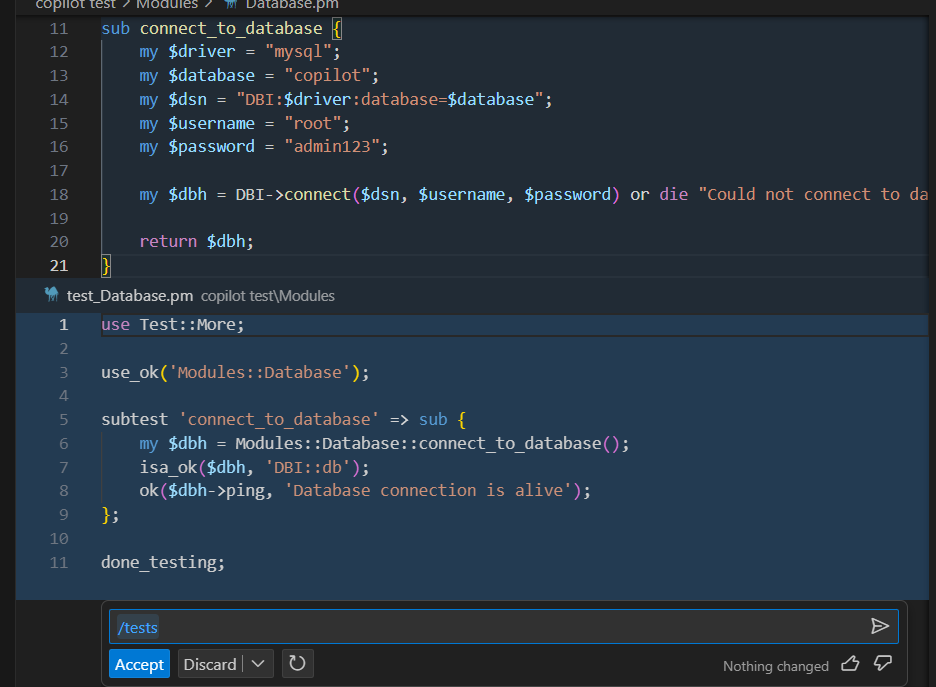


1. Read CSV from script

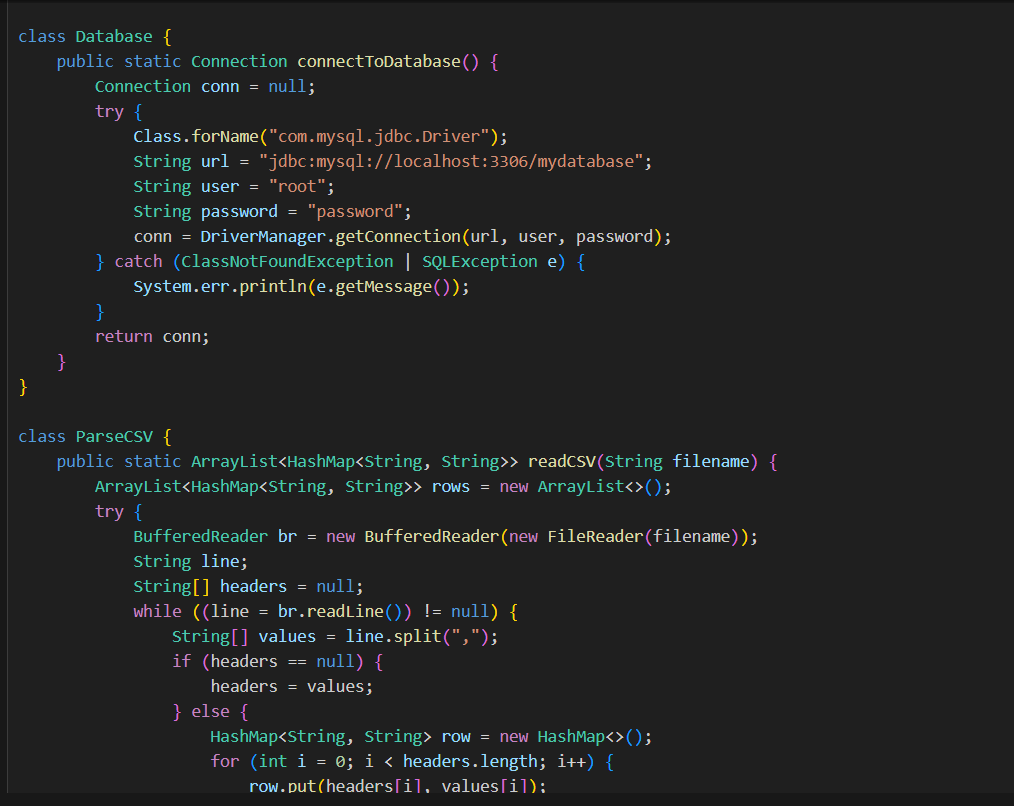


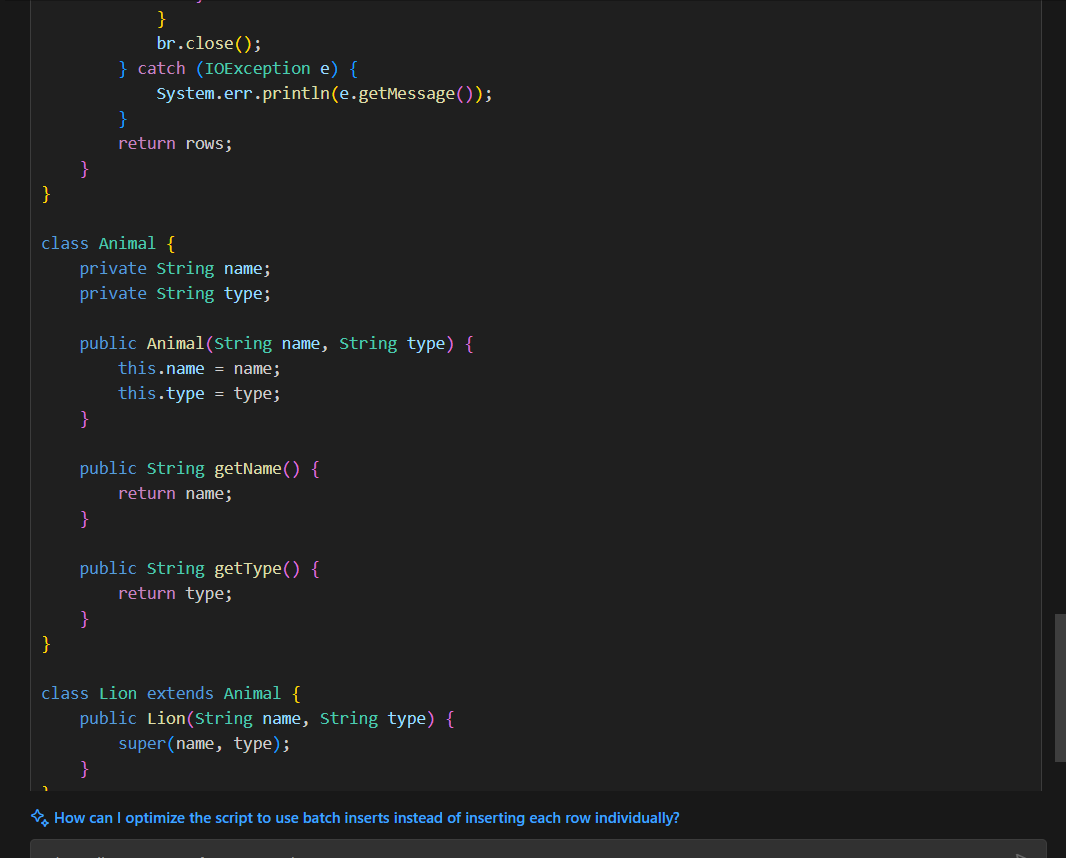
Doesn’t invoke the existing csv function. Instead writes the whole csv read code from scratch.

1. Explain function for the perl code written  
     
   
2. Unit test

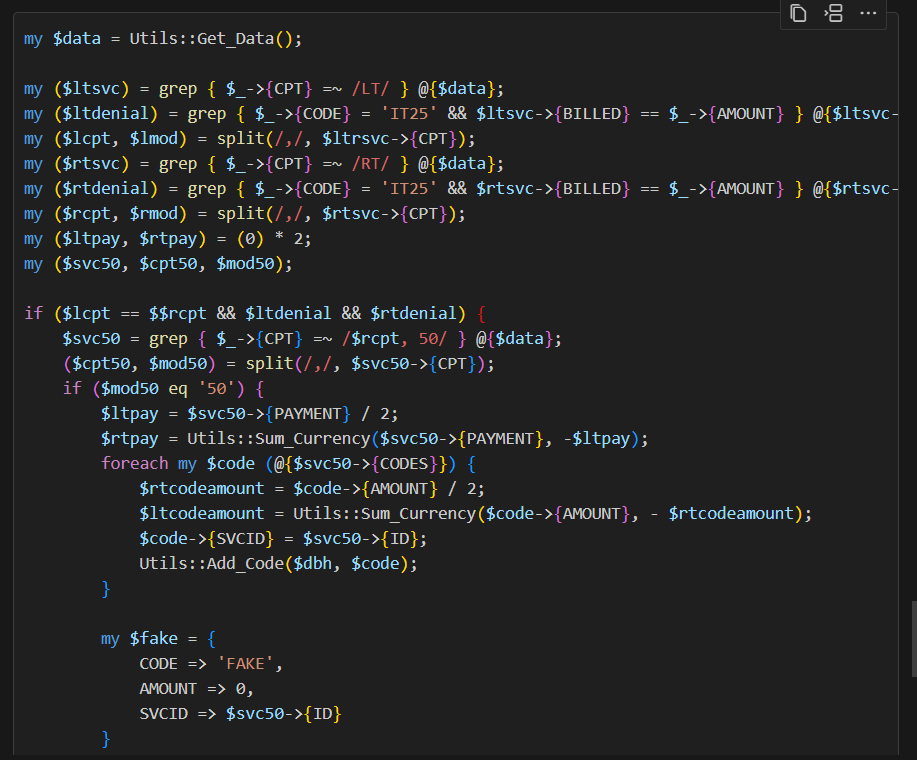


1. Converted java code

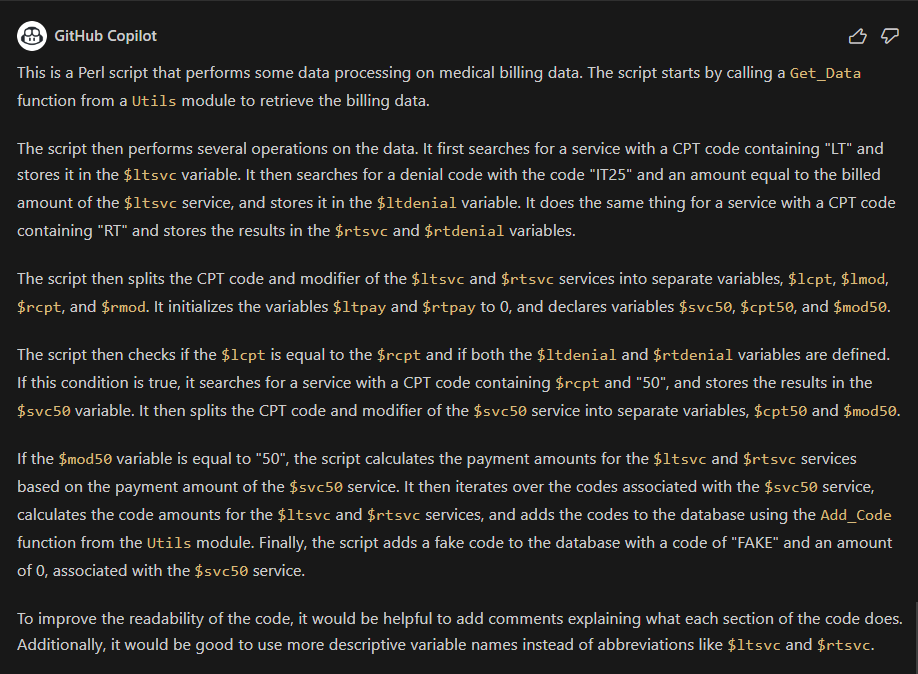
  
  


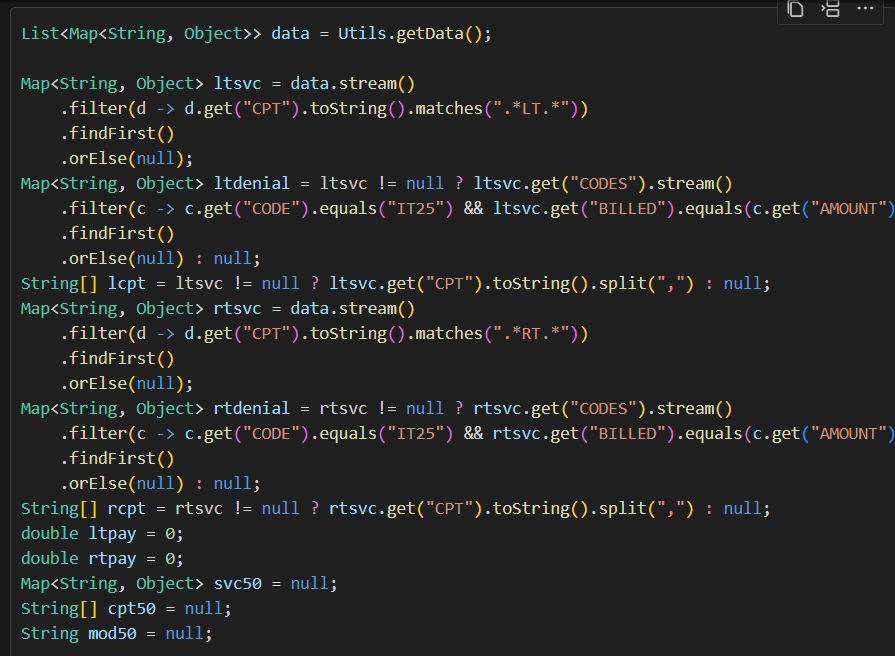
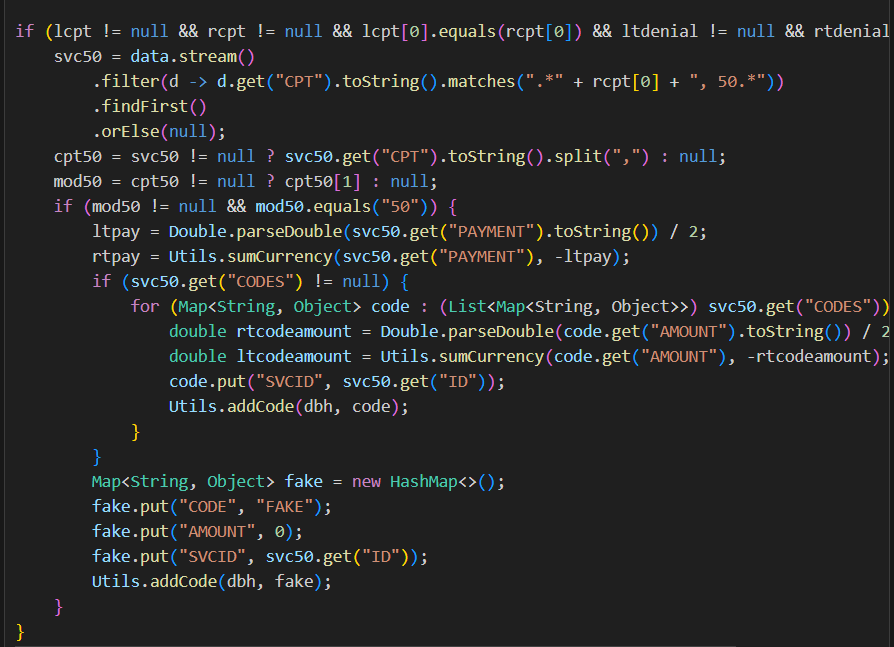


1. Code concepts similar to what is used in posting rules



Code Explanation

  
  
Converted Java code :

* Code is converted in a generic way and uses all the generic coding concepts for conversion
* Detects only the utility function call syntax and translates the code in the exact same way
* Using the copilot generated explanation as a prompt to also generates the same java code which was generated directly from perl.

**Conclusion :**

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Scenario | Observation | Hiccups |
| 1 | Generic code generation in any language | Code generation from scratch works if the prompts are correct. | 1. Doesn’t really look for reusable functions available in the code base. Instead tries to create from scratch 2. Prompts need to be accurate in order to generate the code which we want it to write |
| 2. | Database connections | Able to generate a good database connection template code. |  |
| 3. | CSV parser | Generates a good csv parser. Could improve if prompted in detail. |  |
| 4. | Code Explain | Generates the decent code explanation which is more on the technical perspective |  |
| 5. | Comments | Generates a good detailed documentation for functions which includes inputs, outputs and possible exceptions if available | 1. Generic prompt won’t work properly. However a little bit of detail in the prompt would do the job |
| 6. | Code conversion | The conversion happens on a one-to-one basis irrespective of the design.  Better approach would be to convert small pieces of code which does not involve usage of reusable function. | 1. Codebase is not scanned for availability of reusable functions. 2. Converts the code exactly the same way how the original code was written. 3. Might miss out module imports 4. Possibilities of bugs |
| 7. | Unit tests (Perl) | Its able to generate a decent unit test |  |
| 8. | SQL Queries | Queries are translated as it is and the generic template will be used to execute the query. | 1. Not design specific |
| 9. | Regular expression generation & conversion | Converts the regular expression using string matcher by default. But can use Pattern Matcher modules if prompted in detail |  |
| 10. | JUnit Tests | Generates decent JUnit test cases for the selected code covering multiple positive and negative scenarios |  |

* Copilot works better when generating code from scratch and behaves more like a developer assistant which provides code suggestions.
* Code conversions from one language to another can be done in a generic way if both the languages follow a similar design. Otherwise copilot will generate code in a generic way and later the developer has to manually make changes, Assuming that the generated code has no bug leakage which is unsure