

Analytics: All features related to code generation with Git integration module and MOSS for flexibility

Original Code

```

1 def handle_user_input(question):
2
3     response = st.session_state.conversation({'question':question})
4     st.session_state.history = response['history']
5
6     for k, msg in enumerate(st.session_state.history):
7         if k % 10 == 0:
8             st.write(user_template.replace("{{MSG}}", msg.content), unsafe_allow_h
9         else:
10            st.write(bot_template.replace("{{MSG}}", msg.content), unsafe_allow_h
11

```

Your Code (for Comparison)

```

69
70
71     return conversation_chain
72
73 def handle_user_input(question):
74
75     response = st.session_state.conversation({'question':question})
76     st.session_state.chat_history = response['chat_history']
77
78     for i, message in enumerate(st.session_state.chat_history):
79         if i % 2 == 0:
80             st.write(user_template.replace("{{MSG}}", message.content), unsaf
81         else:
82             st.write(bot_template.replace("{{MSG}}", message.content), unsafe
83
84
85
86 def main():
87     load_dotenv()
88     st.set_page_config(page_title='Chat with Your own PDFs', page_icon='book
89
90     st.write(css, unsafe_allow_html=True)
91
92     if "conversation" not in st.session_state:
93         st.session_state.conversation = None
94
95     if "chat_history" not in st.session_state:

```



Comparison Result

Amount of overlap/plagiarism: 8.47%

Number of overlapping tokens: 301

Commit – 9:00 AM	AI-generated 9:30 AM	AI-generated 9:40 AM	Commit 10:00 AM
1a	Code snippet AA	Code snippet BB	1b

Spec:

- Log all events from Autocomplete, Sonarlint Autofix, Docify when user accept AI-generated code based on events [defined here](#) by Manh.
 - o Code (**other.autocomplete_loc_accepted – LOCX(count lines)**) - Code snippet AA and Code snippet BB above - **ĐẠT**
 - o Current git info (branch and HEAD commit) - “1a” above - **ĐẠT**
- When there is a new commit, log Git commit events from IDE for files. Can we turn this into a setting for that profile and enforced by PM for three levels:
 - o Allowing productivity tracking on all files within workspace
 - o Allowing productivity tracking on all files allowed by Code Context
 - o Productivity tracking not allowed
- Can you do Git diff in the IDE or have to handle in Backend?
 - If yes: other.code_added and 1b and 1a commit hash content
 - If no: other.code_committed and 1b and 1a commit hash the in the backend - Compare (other_code_committed of 1a,other_code_committed of 1b) -> other.code_added
- Backend service to: **PHÚC**
 - o Compare a database of other.autocomplete_loc_accepted of HEAD == 1a to **other.code_added – TLOC (count lines)** using MOSS
 - In other.code_added, for example if line 10-20 is matched to other.autocomplete_loc_accepted then we count those lines as

other.autocomplete_loc_committed. Only count for line 10-20 once, meaning if it is already counted as matched once, it cannot be counted again.

- But if line 8-25 also got matched then we use 8-25, not 10-20
- If 8-15 matched, and 10-20 matched, then we count 8-20 as AI Generated
- Count other.autocomplete_loc_committed to calculate other.autocomplete_number_loc_committed **LOCY**
- MOSS is not for commercial use, use this instead <https://dolos.ugent.be/about/algorithm.html>
<https://github.com/dodona-edu/dolos>
- Return **programming language** as well to calculate saving MM

- All events have to be attached to the right user profile
- Clear storage of these file snippets after calculation is done

AI4Software: Effectiveness Measurement Methodology

