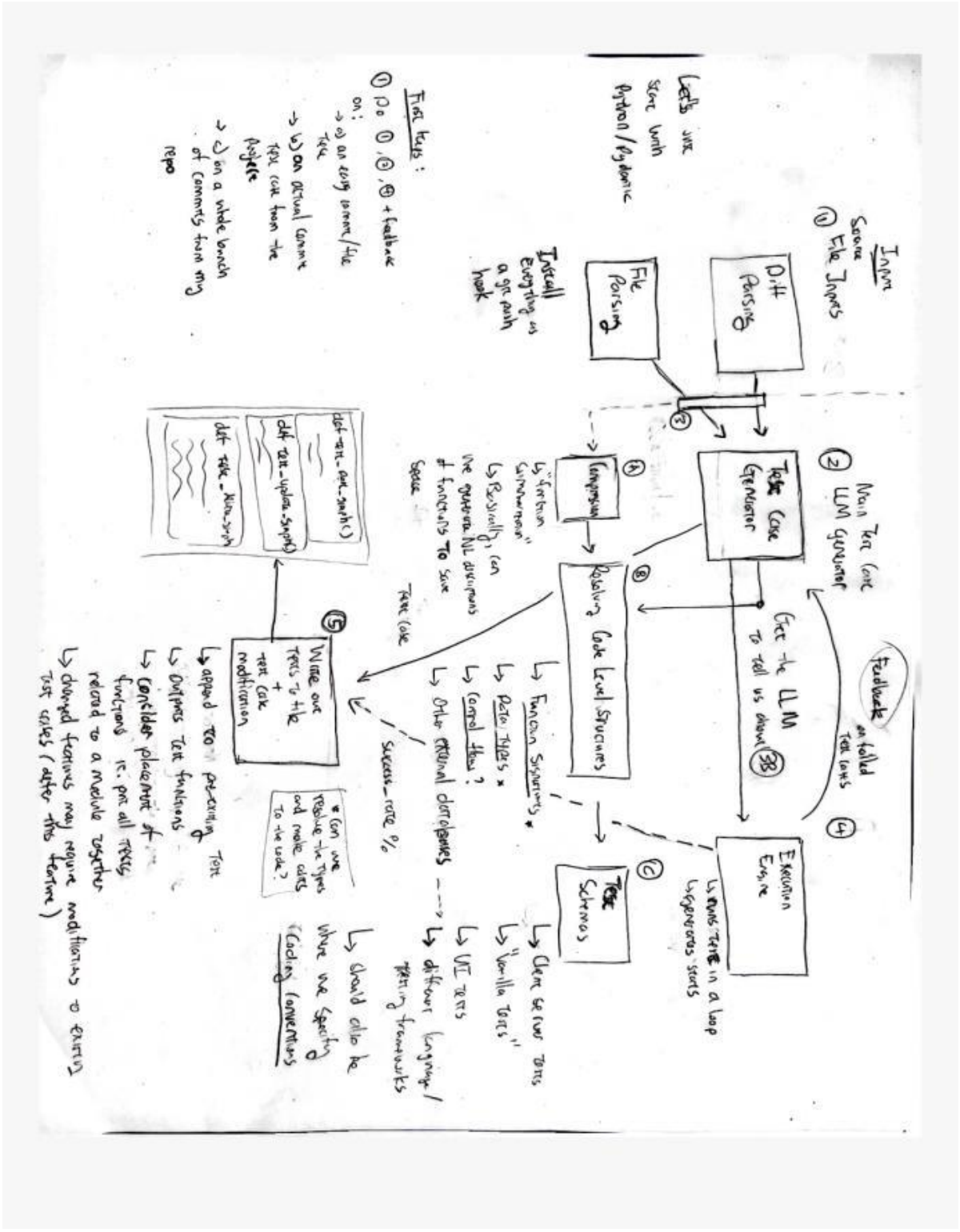
Compilation of screenshots and pictures of notes taken by members of the team during the meeting on 17th October 2023.



Code coverage: static analysis tools - execution path, if statements, can even use LLM itself to assess the branches and then generate test cases, split up the steps

90% accuracy - 1. if it compiles 90% of the time after several loops, without human intervention 2. if it actually covers 90% of the test cases we want - may up to further review (can't let gpt generate test metrics and then assess 90% coverage based on that)

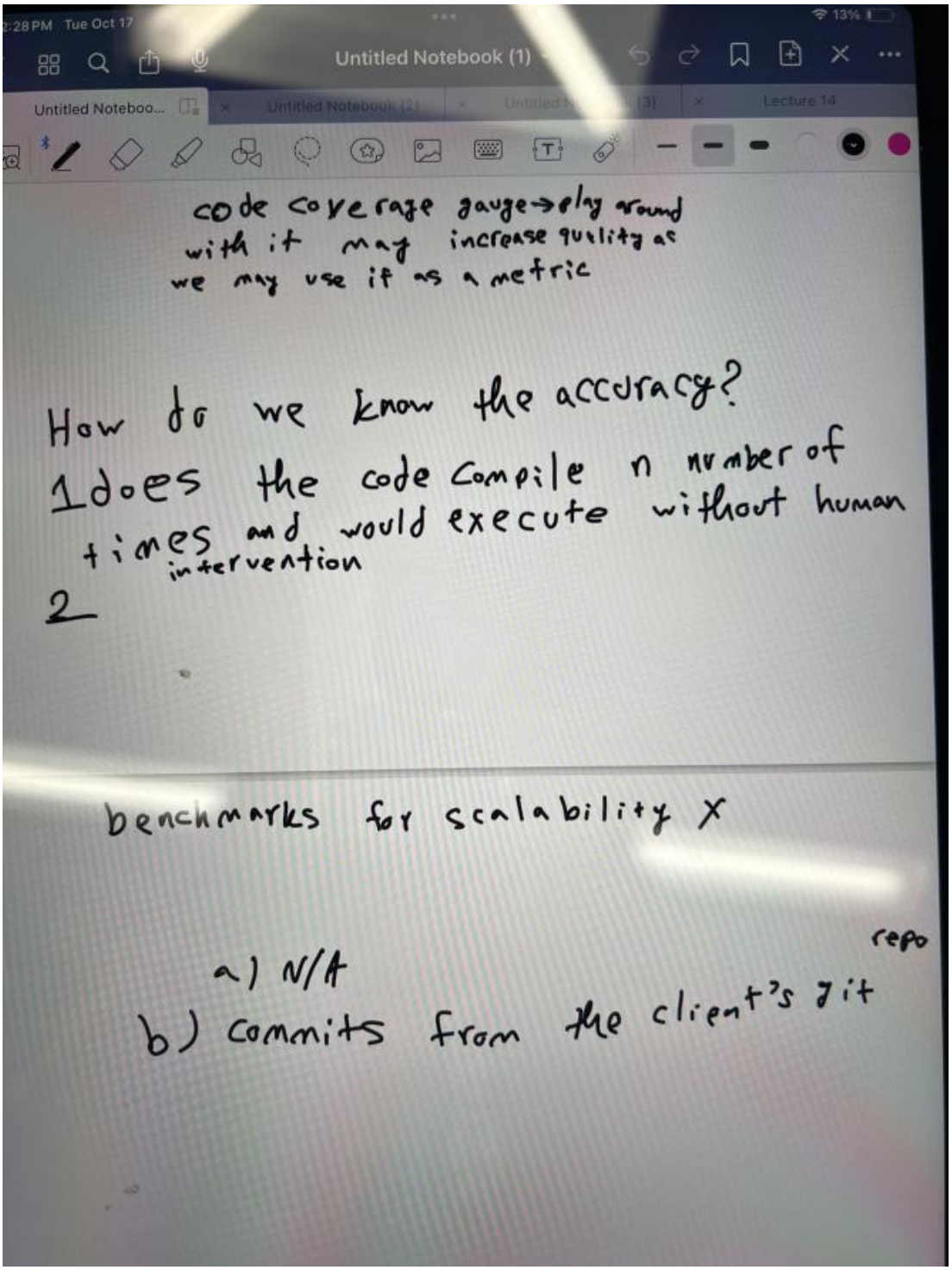
Efficiency (minimize money spent on LLMs, charge per token, cost per tokens in source file) more important than scalability, mostly run locally

Interface - UI not important, print out ascii metrics table

Data - trust openai with all the source code

In sample test accuracy
Out of sample test accuracy

0.1: assess on diffs and commits


2:28 PM Tue Oct 17
13%
Untitled Notebook (1)
Untitled Noteboo...
Lecture 14
code coverage gauge → play around with it may increase quality as we may use it as a metric
How do we know the accuracy?
1 does the code compile n number of times and would execute without human intervention
2
benchmarks for scalability X
a) N/A
b) commits from the client's git repo

Oct 17

4218 - Must compile
- Test case coverage
- Write generated test cases + result to output JS
- Client: John
- Stakeholders: John + Open Source Devs
- Works with Github, Python, Pytest, Pydantic, we based, using Open AI GPT 4 or 3.5
- Code in Python
- Use LLM to output report of different code coverage results
- 90% accuracy
  1) Does the code compile 90% of the time (with iterative runs)
  2) 90% test case coverage

-

11)
a) N/A
b) Provide documentation (maybe even just a ma
c) Cost and accuracy
~~Cost~~
Total LLM cost / Number of tokens
d) N/A
e) N/A

Notes

Coding & Testing:

- might not use waterfall and use agile instead
- 90% accuracy based on ensuring that the test cases compile 90% of the time.
- testing pulls from chatgpt's database so code and tests come from there.
- No scalability issues as it will be run from the user's computer and code.
- Errors in final output should be relooped in and rerun.

Project Constraints:

- LLM's charge per token so there is a cost bottleneck. Need to find ways to reduce this.
- Source code / Input errors will produce test cases that compile but when run will give a test case failure so that we know when source code has bugs.

- Project must generate a specific type of python file as output.
- Must run in github w/ gitcommit.

Interface: User git commits, test file generated, ascii table w/ the test cases, compile (Y or N), compile time, etc.

Users: John Peng & open-source developers.

Security & Risks → aware that uploading code on chatgpt means it will keep some of that code. Not much data privacy and security.

Documentation: will build a user guide / documentation on how to use the bot and its functionalities.

client john
stakeholders developers open source
community and john

github python and pytest should generate
pydantic compatible python source code
openai gpt4 and gpt 3.5 models

enforces typechecking so u cant put
whatever
webbased

0.1: assess on diffs and commits
run in the same cases when adding features but also add new tests to check for overfitting

Source: commits from clients repo

What if input source has errors: exactly what we want - another metrics, number of actual errors found

Usability: Python code will not come statically typed

Documentation: readme, configurations

Agile development