Week 3 Group Coding Exercise

Group 3s submission for Computer Science 323 , Section 12, Week 3

Our Information

- Name: David Ulloa
- Email: davidulloamontesinos@csu.fullerton.edu
- Name: Emily Velasco
- Email: emilyvlsco@csu.fullerton.edu
- Name: Medrian Manuel Bacareza
- Email: medbac@csu.fullerton.edu

How to Run

1.) Put both automata.py and demo.py in same folder in location of your choosing. 2.) cd into that file from terminal 3.) Run python3 demo.py

Example: automata.py and demo.py are in file group_exercise_3 in path /Documents/group_exercise_3

```
cd Documents/group_exercise_3
python3 demo.py
```

Hardest Part of Group Exercise and Why?

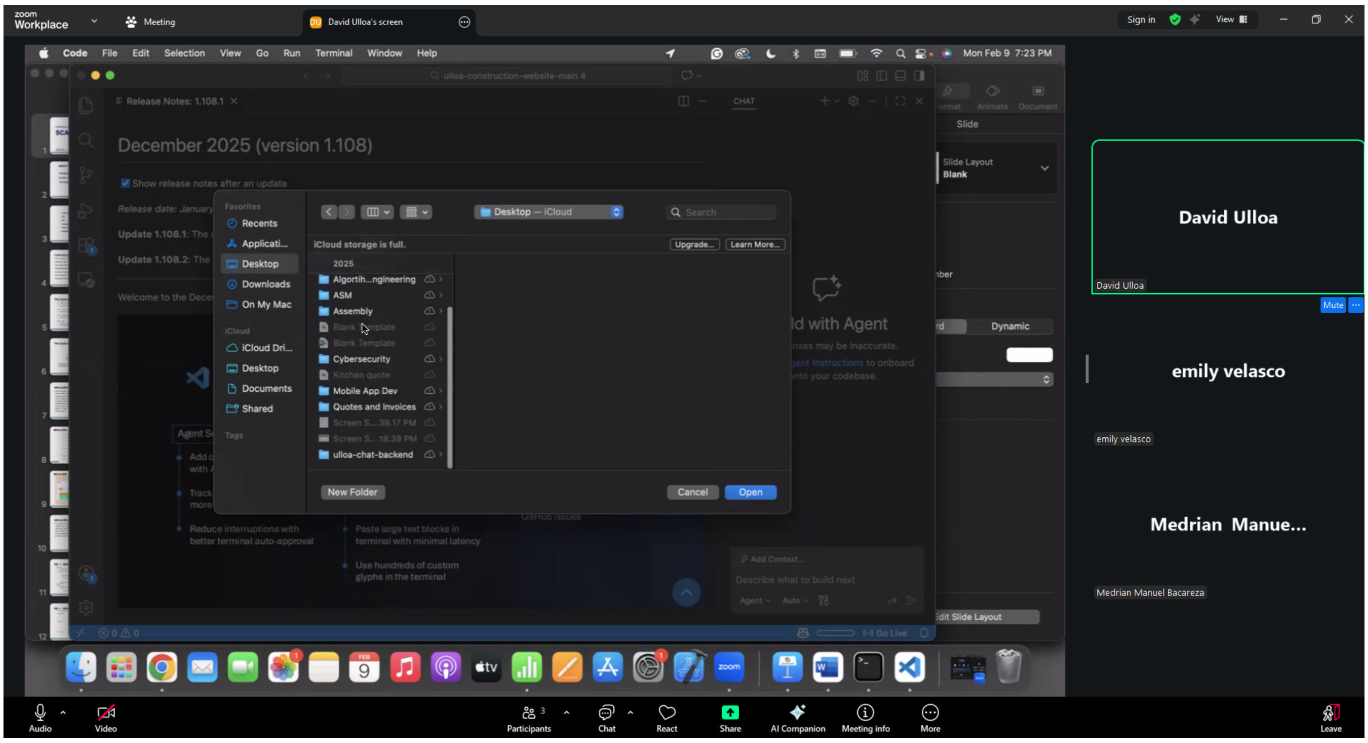
One of the hardest part of this exercise was the debugging after testing. We ran into a few "MISMATCH FOUND" errors and had to review our logic. The fix ended up being quite simple. We were able to fix it by including epsilon closures at each step and it resolved the issues. Another issue we ran into was with code syntax and formatting. Since we are not really familiar with python, we found errors with our code tabbing that causes logic errors. Given that in c++, parentheses are what keeps code organized, the tabbing format through us off, but we persevered. The last error we ran into in terms of our source code was syntax. When we were transfering the code from the doc over to our file, we left a line of comments without the proper formatting which causes another compilation error.

The other difficult part was understanding the outcomes of the program. We had to review the previous class slides and spend extra time understanding the contents of the code and what the outputs entailed. For example, we spent some time reviewing over the differences of subset construction and how the outputs demostrated how NFAs can be in multiple states at once called configurations. Another important concept was the idea of how each DFA state is also a subset of NFA states. Overall, this part was the most difficult part of the group coding assignment for us.

One Insight From the Trace?

In the trace for 'baab' shows the NFA stays in n0 through the first b, then it only branches when it hits the 'a' which starts the abb attempt.

Team + Proof



Files Included in Proof

1.) automata.py

2.) demo.py

3.) README.md 4.) zoom_prood.png