You may insre
yettext aege

Project Title: SLAIT – Source Language Assembly Inspector Tool

Team Members:
- Maria Linkins-Nielsen – mlinkinsniel2022@my.fit.edu
- Michael Bratcher – mbratcher2021@my.fit.edu

Faculty Advisor: Dr. Marius Silaghi – msilaghi@fit.edu
Client: Dr. Marius Silaghi – CSE Professor & Faculty Advisor

Progress of Current Milestone:

Task	Completion%	Michael	Maria	Reasoning/To-Do
1. Initial HTTP Test	50%	50%	50%	In the backend - Initial HTTP test has been configured, however, it needs to be adapted to receive file transfer. In Frontend - HTTP Frontend is ready for test once backend adaptations have been applied.
2. Update File Manager Widget Visuals	100%	100%	0%	Visuals have been minorly overhauled to be significantly more visually appealing. This also includes a system visual check and a dark theme swap toggle.
3. Create Widget Component for Visualization	100%	100%	0%	Frontend has structured the final major component, while not visually present, will allow ease of implementation once the http requirement has been met.

4. Finalize backend file upload and execution API	50% Type text here ype text here	0%	100%	Currently working on the API to automate the process of spinning up the Docker container, taking in the user's asm file and the line tracking file generated by the UI, running the asm file while tracking the specified registers and flags, then sending the output file with the output of the tracking process back to the front end. Currently, all components of this process operate independently; now I am working on automating the process so that it functions as a cohesive whole.
5. Implement Multi-Line Register Tracking in Backend	50%	0%	100%	The current Python script takes in a single-line string as input and returns an output file with asm file output and formatted register states at the specified line. The script just needs to be adapted to take in the line specifications as a text file instead of a single string.

Discussion of Milestone Tasks. You may inspire from the asm Jupiter kernel
https://github.com/ConnorNelson/asm-kernel

1.

Maria:

I started setting up the backend HTTP test in Flask and confirmed that the server is able to receive basic requests. However, I did not progress to the point of parsing or handling the separate file uploads. The next step will be implementing the multipart form logic the frontend requires for sending both the user's .asm file and the tracking file.

2.

Visuals on the frontend have been adjusted. All buttons are now in a uniform layout, following a pill shape with glow highlights for hover and click functions. Stylistically, the frontend has also been made significantly more uniform with a consistent purple theme throughout the page. In addition, a dark mode toggle has been added to the frontend, which will automatically default to a user's system default when accessing the page. Toggling from dark to light or vice versa will adjust the colors to be easier for the designated monotone option. This also applied to the text viewer, which is no longer a static black and changes with the mode.

3.

The component for the visualization tool has been created on the frontend project. While not currently visually apparent, this is the final major widget the project will need, and its component has been instantiated.

4.

I did not complete the automation pipeline during this milestone. My work mainly consisted of reviewing how the Docker container, *run_asm.sh*, and *parse_registers.py* currently run independently so I can plan the necessary steps to integrate them. Full automation—spinning up the container, calling the scripts, and returning an output file—remains a task for the upcoming milestone.

5.

I began looking through *parse_registers.py* to understand how line numbers flow into gdb, but I did not yet implement the logic for reading a text file or handling multiple breakpoints. This will be one of my primary backend tasks moving forward.

Contribution to Milestone Tasks:

Michael:

I worked on the frontend development tasks, which included the visual overhaul, the start of the component for visualization, and the HTTP frontend.

Maria:

During this milestone, my contributions were limited but focused on laying the groundwork for the backend integration. I began experimenting with the initial Flask HTTP setup and confirmed that the basic route is reachable from the frontend, but I did not yet complete the full file-parsing logic or connect it to the remaining backend components.

I also reviewed the existing backend scripts—*run_asm.sh* and *parse_registers.py*—to understand how they will eventually fit together. While I did not complete major functional updates, I identified the changes needed to support uploading two files (the asm file and the line-tracking text file) and the adjustments required for multi-line breakpoint handling.

Overall, my progress this milestone was more exploratory and preparatory than fully implemented. The main development work for the backend automation will take place in the next milestone.

1.   Plan for the next Milestone (task matrix) or
[skip if this is for Milestone 6]

Task	Michael	Maria
Finalize Backend HTTP Configuration	0%	100%
Finalize Backend Automation	0%	100%
Add "Template" Frontend Feature	100%	0%
Continue Visualization Widget Updates	100%	0%m

2. Discussion of each planned task for the next Milestone
   ○ Task 1: My goal for the next milestone is to fully implement the backend's ability to receive multipart POST requests from the frontend. This includes correctly parsing both uploaded files, validating them, and preparing them for use in the execution workflow. This is a continuation of the partial work I started during this milestone.
   ○ Task 2:. The next milestone will focus heavily on the backend automation that I did not complete this cycle. This includes linking the Docker execution steps with the register-tracking script, managing file flow between components, and packaging the results so they can be sent back to the frontend.
   ○ Task 3: This will be a continuation of the snippet focus on the program. This will include a number of pre code templates for users to select when using the text editor option to improve ease of use within the application.
   ○ Task 4: The Widget for visualization will be developed as much as possible with respect to the current lack of backend return data.
3. Date(s) of meeting(s) with Client during the current milestone: n/a
4. Client feedback on the current milestone
   ○ see Faculty Advisor Feedback below
5. Date(s) of meeting(s) with Faculty Advisor during the current milestone:
   ○
6. Faculty Advisor feedback on each task for the current Milestone

- ○ Task 1: You may inspire from the asm Jupiter kernel https://github.com/ConnorNelson/asm-kernel
- ○ Task 2: Type text here
- ○ Task 3:
- ○ Task 4:
- ○ Task 5:

7. Faculty Advisor Signature: _____ Date: _____

8. Evaluation by Faculty Advisor
   - ○ Faculty Advisor: detach and return this page to Dr. Chan (HC 209) or email the scores to pkc@cs.fit.edu
- ○ Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

Michael	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Maria	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

- ○ Faculty Advisor Signature: _____ Date: _____