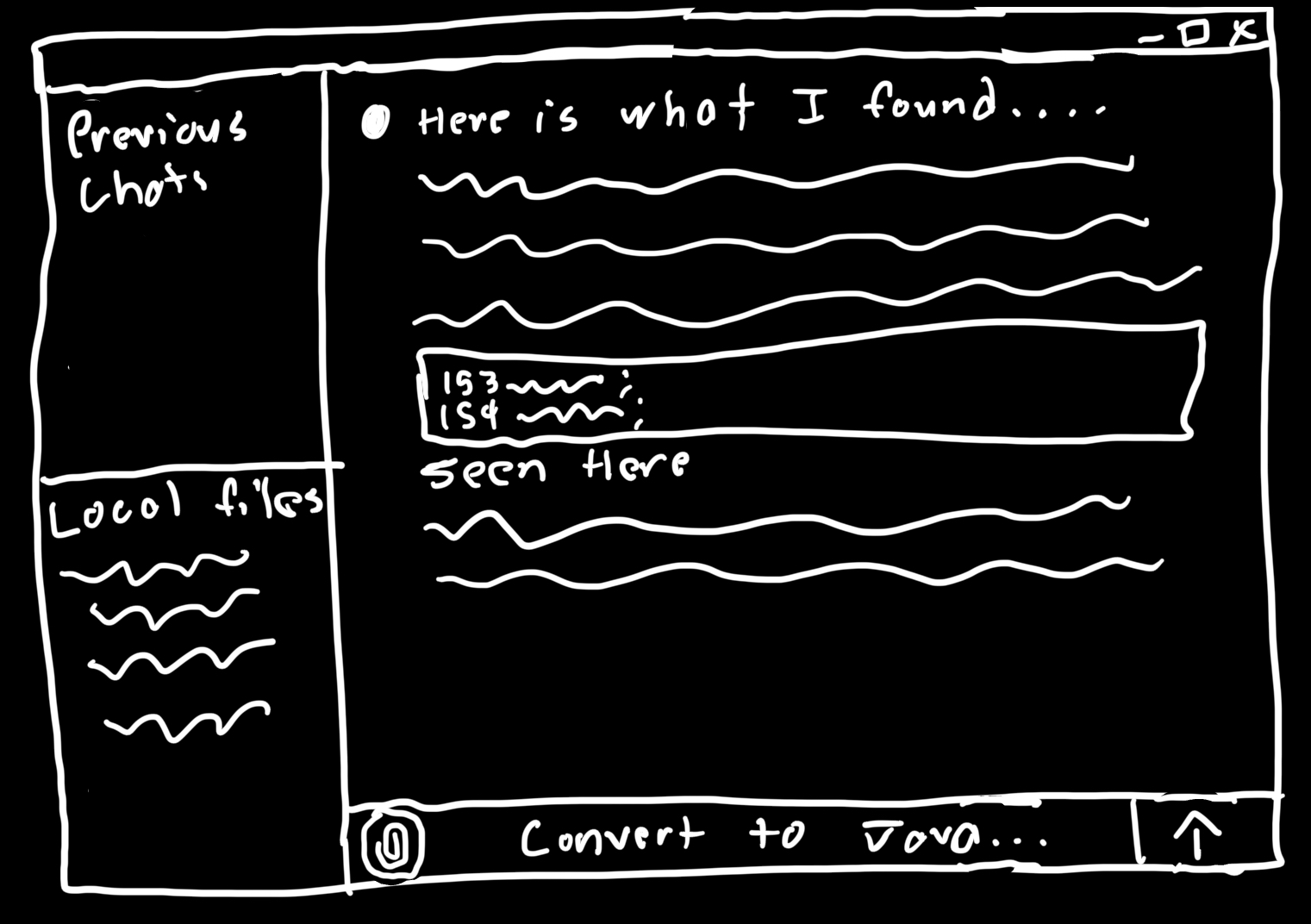
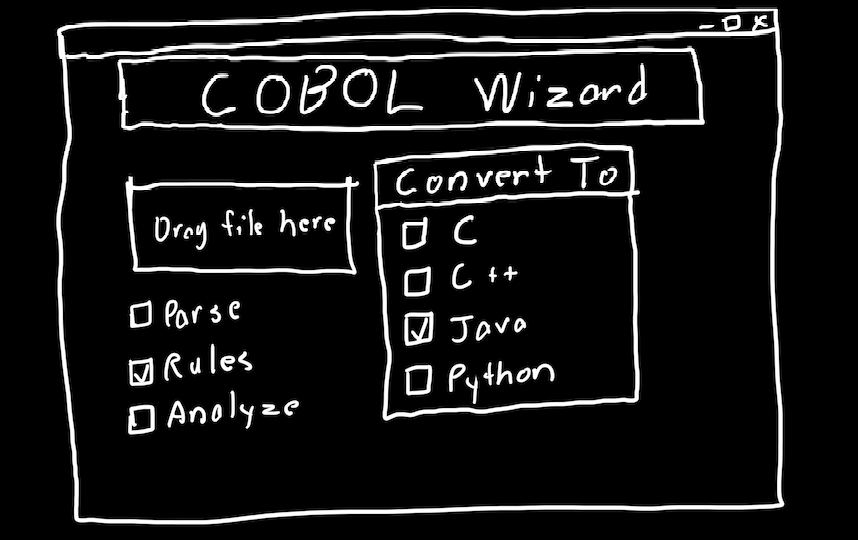
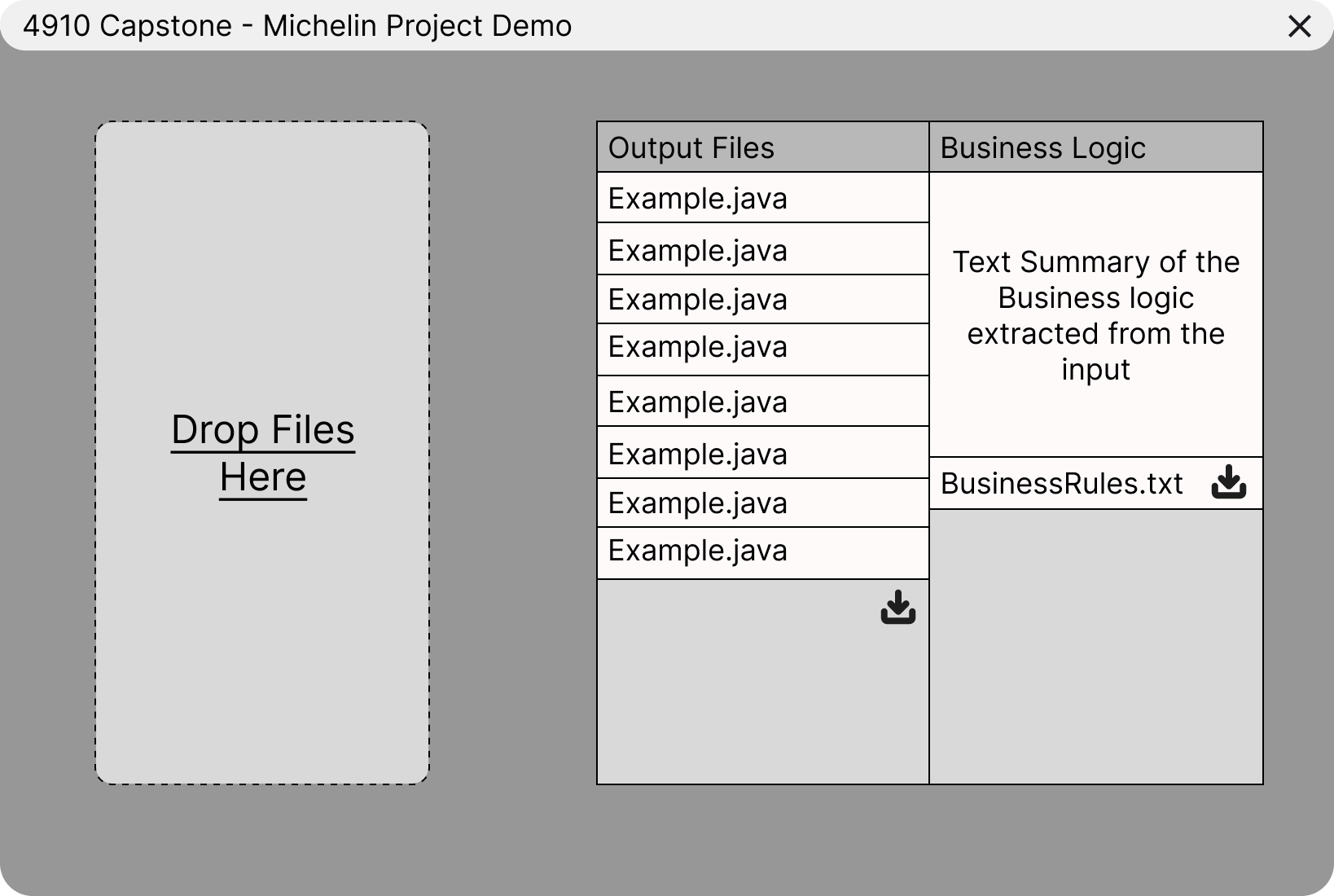
(AJ) So this is what I am thinking the product should look like on startup. This is after clicking on a desktop shortcut. This would just simply output files and maybe a text file that has the requested information with the converted file too. Maybe this is the prototype for a smaller system that we use for proof of concept?Or something like this, similar to current AI prompts. This might make the system more inviting.

(Troy) What I had in mind was something similar to AJ’s first picture above. I envision an interface that allows a member of Michelin’s IT department to select a Cobol file. The interface would then automatically extract business rules from the file and present them in a preview pane for review. There might be some way for the user to make changes to a particular business rule in the preview pane if needed (e.g., if the AI wasn’t able to interpret the rule correctly). The user would then click a “generate” button to create a new file that is compatible with the new Oracle system. My understanding of this project is that Michelin wants to automate the process as much as possible, but they would still need to have some degree of control, hence why the application would require manual file selection, and the user would be able to make amendments if needed. There would also be an option for batch file conversion.

(Carson) I think we should take an existing Generative AI Model like CodeBERT which is pre-trained for code interpretation and fine tune it (look at AWS for hardware) to understand and interpret COBOL. Using the COBOL from ordering and invoicing which already has the business rules extracted we can train the model to hopefully recognize and extract business rules from the COBOL. Using this AI our app would take in COBOL and interpret it which could be used to output information to the user such as graphical structure representations as well as generate java code to accomplish the same tasks as the output. From the research I’ve been doing today I don’t think a COBOL analyzer or parser is strictly necessary but might be helpful to use later.



(Caroline) I like the first graphic AJ has. I think that there are two main end goals: the ability to convert the documents and the ability to extract the business rules out of the documents. I’m looking into the options to do each. We can either use two different tools as a two-pronged solution and put it into a GUI or we can see if we can find something that can do both together. Doing more research tomorrow.

(Jacob)

* Point system to specific COBOL file(s) to extract rules from
* Accurately extract business rules from the given COBOL files
* Respond to English business rule queries (lest we invent a new language to query the system; response format is flexible)
* Produce responses to queries using extracted business rules
* *Stretch:* GUI for interaction loop
* *Stretch:* Convert semantic business rules to new database format
* *Stretch:* Validate translated business rules for correctness

Selection criteria:

1. Accuracy: How accurately the solution answers business rule queries
2. Comprehensiveness: How much of the overall system goals the solution provides
3. Responsiveness: On average, how long the solution takes to produce a result
4. Implementation: How complicated the solution is to implement.
5. Cost: Financial cost of using the system
6. Computational intensity: Compute time needed to implement/make changes to the solution
7. Documentation: Measure of documentation quality for the included in the solution frameworks
8. Extensibility: How easily the solution can be modified later to reach stretch goals.