**Evaluator with State**

**By Trinity McCann**

**CS441 with Brian Hare**

**5/1/2025**

### Table of Contents

[**LLM Choice and the Reason Why**](#_6a6dd8hg0z0w) **1**

[**LLM’s Helpfulness**](#_yfqez19owvqj) **1**

* **Figure 1: One Simple Prompt**
* **Figure 2: User Input**
* **Figure 3: Remove Test**
* **Figure 4: Full Testing**

[**Semester Reflection on LLMs**](#_ihyg9pv8kqbw) **2**

[**Additional Skills Needed to Make Use of LLMs**](#_ecvh0mw1znyk) **3**

[**Big Picture Takeaway**](#_k0q9dh4grdxa) **3**

[**Advice for Future Students/Faculty**](#_6haw6zt0eoc6) **4**

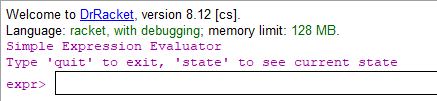
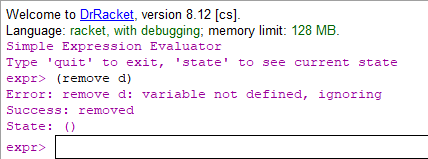
### LLM Choice and the Reason Why

As with the past two assignments for this course, I have selected Claude by Anthropic as the tool of choice for this project. I originally chose Claude at the beginning of the semester due to community consensus generally agreeing that it is the best LLM for code generation. Certain models of ChatGPT, Deepseek, and others have outshone it temporarily in smaller categories, but Claude reigns supreme when computer science is on the line.

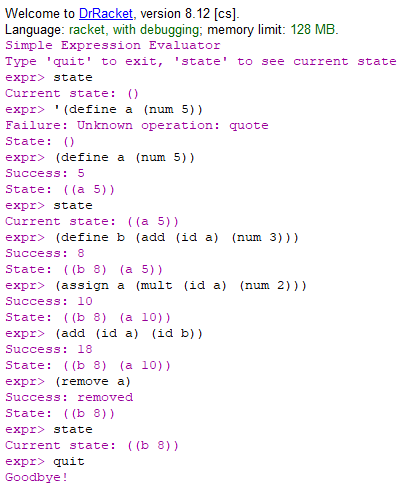
My relatively positive, though minorly frustrating, experience with Claude in the past has held me to coming back to it for each of the three projects. The current Claude model, and the one used in this project, is Sonnet 3.7 which has been out since February 24th, 2025. I was not able to utilize 3.7 for the first project, as it was due before it was released, but I did utilize it for the last project and now this one.

### LLM’s Helpfulness

I had mentioned previously in project 1 that I was blown out of the water by how well Claude was able to perform my prompts. To say that now would be none other than the greatest understatement of all time. Claude has outdone itself and I am even brought to say every other LLM I have interacted with. I wouldn't consider myself a “vibe coder”, someone who relies on LLMs to create grand projects and alleviate them of bug fixes, but I have tested with them seeing as I am a computer science major and they will have a direct, significant impact on my life. I have even downloaded a local running, albeit primitive due to lack of RAM, version of Deepseek on my personal computer. 

Claude successfully completed the entirety of this project in a singular prompt. With zero misunderstandings, missing parentheses, compile errors during runtime, or logic errors during manual testing. I sat in shock after testing, in disbelief that it was fully functional. I tested everything suggested in our document outlining the requirements for submission. After that, I went back and tested another two times. It proved to be a fully sufficient project that covered all guidelines. 

I cannot give it higher marks on this project. I simply wrote a few sentences. Quoted the project directly as this project, unlike the last few ones, had a lot of small specifications like specific terms, grammar, and rules to adhere to. I didn’t want to misphrase something or have it be unclear.

I have a few theories as to why this project might’ve gone better than the previous one. First, it could be that Claude’s 3.7 model has received minor update pushes since its initial release, which could result in performance improvements. Secondly, the specifications of this assignment being very straightforward rules fit exceedingly well into an LLM’s natural logical processing. That is to say that, because the evaluator rules are simple logical facts instead of things up to interpretation to the developer, an LLM would have an easier time meeting the specified requirements. My last theory is that, perhaps, giving Claude a starting point of *working* code, helped it to see the final project in its correct fulfillment. In my experience previously, giving Claude a starting point often proved more harmful than beneficial, but usually it was code that was not fully desired. There is a chance that a strong foundation of reliable code will help the LLM.

There is a chance that none of these theories are correct or, more likely the case, that all of them had part of something to do with this wild success. Regardless as to the reasons why, I would give Claude a 10/10 for helpfulness on this project. I truly can’t think of how it could improve from being better than taking only a single prompt except for maybe reading my mind and doing it automatically, although I fear that is for a much more dystopian timeline that I pray is not ours. ‘

Attached to this document in full is all of the testing showing the full functionality of Claude’s code. Submitted alongside this document inside of the GitHub repository are the other requirements for submission including this report, all chat logs with Claude, and the full Racket code file. Unlike previous submissions, this assignment focuses on our reflections over the entirety of the semester. The next sections will cover semester reflections, key takeaways, and advice as outlined in the submission requirements as opposed to code breakdown and testing that were required in previous projects.

### Semester Reflection on LLMs

Throughout this semester, I have truly utilized AI and LLMs for the first time. Before this semester I had occasionally poked around with some of the silly things they could generate, especially list making and brainstorming ideas, but overall found them to be rather repetitive and unhelpful. This semester, I had not one but two classes actively require me to utilize them on my code. Initially, I was not only skeptical but downright stubborn. I did not want AI writing my code for me! College is supposed to show *my* intellectual skill, not that of generated and copied text.

However, I believe these initial thoughts come from a place of pride, likely one that isn’t even misplaced. We as humans value our trades/crafts and can view new and innovative tools as threats to what we enjoy and take pride in. Machines or robots replacing our jobs is a complex notion and the blame is often misplaced on the tools and not the societal pressures and expectations that are actually what punish us. Innovation and tools, the mass efficiency of the working class, should come to their benefit, allowing them to be freed of their societal burdens, but yet it has only ever come to punish them.

To go to college isn’t just about learning and proving your education, although that would be nice. It is also about the degree that you hold up at the end of your years of study, about the certification that deems you fit to be sent out into the workforce in a career that matches your teachings. To not train future computer science graduates with what will become a necessity in their vocation would be a disservice to them, to your university, and to the entirety of the workforce. Overall, I am very proud to see that my coursework has led me to fully submerge myself into learning AI as I view it essential to graduate.

### Additional Skills Needed to Make Use of LLMs

I do believe that the use of AI, particularly for code generation, isn’t an end-all be-all. It’s a simple tool, albeit a powerful one, it can still only do so much. A large part of software development has long been known to have very little to do with the actual coding process and much more to do with project planning and management. Not to mention, just like text based AI generation, which requires significant edits to touch it up for your exact desires, or image based AI generation, which needs photoshop artists to go in and fix mistakes, code generation is similar.

Even if AI models get to the point of testing their own code to ensure no bugs or errors, I hardly doubt they’ll solve real world development plans. Logic errors are already considered to be more difficult as a developer to test for and fix than compile errors. When I’ve tried to generate a front-end website page, buttons didn’t work appropriately and numbers sometimes did not do what they were supposed to. These are things that are integral to the development lifecycle. So, if I were to be either a freelance or corporate dev, I could use AI to shorten the code generation process, but I would still need the skills to set up my IDE, perform necessary IDE/language/PC updates, test and accurately detect logic errors, and have the capability to make fixes as requested by my team, manager, and clients. The skill to create good prompts is also in the mix; the clients are paying you for your expertise in all of these aspects.

### Big Picture Takeaway

I think AI and large language models have taken almost everyone by surprise since ChatGPT’s public reveal in 2023. Since then, the world has been thrown into frantic panic with all industries playing catch up in a game they didn’t even know they were playing. Coding, art, writing, media creation, law, science, medicine, just about every industry you could think of is affected.

As such, it's more important now than ever for research institutes to start studying the short term and long term impacts, the uses, the dangers, what it is, and where it came from. Universities play a major role in the development of AI and its impact on our economy and culture.

### Advice for Future Students/Faculty

In my opinion, UMKC has taken the correct first steps. Banning AI use before anyone really knew what it was capable of was the correct first response. AI was dangerous, hard to track, and many people weren’t not experienced in what they were handling to be able to make appropriate judgements on it. I also believe that UMKC’s next steps were appropriate. Beginning and funding research into AI and its effects on the classroom and subsequently allowing it to be used while strictly monitored, proctored, and self reported by students is nothing less than genius. Students were still finding ways to use it anyways, students need these skills to be successful, and it provides large amounts of test data on AI through assignment submissions and in classroom feedback, allowing faculty to make appropriate decisions to be made.

I do not think classrooms will ever go back to banning AI, particularly in the computer science field. As much as I dislike hearing that, that is where the industry is taking us and UMKC owes its students and faculty that much. My earnest advice is to encourage, require, and enforce open communication about AI’s use. Become stricter on giving credit to not only what LLM was used, but the specific model, and creators of the model.

I think becoming too strict on the use of AI could become a slippery slope; if it’s truly being used as a tool, then where do professors draw the line? Grammarly already uses models trained on large data sets so, while not an LLM, it still uses the similar processes. The same argument against LLMs, that it isn’t your own creative work could also apply to the grammar edits it makes. Should we ban grammarly from all classes? And if grammarly is banned, what about the simple autocorrect on our PC softwares? That also isn’t your own original work if you truly didn’t know the difference between homophones.

I am very delighted and proud of UMKC for its take on AI. I feel I have graduated as a better student having gone through it being banned, discussed in classrooms, permitted, and required throughout my time here. I look forward to seeing how UMKC and other universities address AI in the future.