Personal Programming Project Report

Title: Minecraft Crash Log Analyzer
Provide a descriptive title for your project.

1. **(5pts) Honor Code and LLM Usage for this Report.**

I have neither given nor received unauthorized assistance on this assignment.

During the preparation of this assignment, I used VSCode Copilot in code generation to utilize python libraries like Flask without taking the significant amount of time to understand it fundamentally. After using this tool, I reviewed and edited the content as needed to ensure its accuracy and take full responsibility for the content in relation to grading.

2. **(15pts) Learning Objectives:**
   List the learning objectives from your proposal. In your own words explain whether you met those objectives and how (50-100 words each objective). Also describe if you learned something different than expected or anything additional.

Learn how to process information from existing websites:
Yes, I was able to meet this objective as I technically was able to process information from Google Gemini's API/website to use in my program, and I also learned to use online hosted web services like Render to remotely host the python code for my website, as GitHub alone would not be able to run it.

Build interactive web applications using JavaScript frameworks like React or Angular:
Yes, I think I somewhat met this objective since in my website I was able to use JavaScript to add file handling so users could submit files to be analyzed by Gemini. Although I did not incorporate frameworks, I was still able to build a barebones interactive website, as it is not just a static webpage.

3. **(15pts) Timeline:**
   Outline how you spent time on your project. Break down the time into specific tasks or milestones. Here is an adjustable schedule to get you started. Actual Details should be 50-100 words each and should compare or reflect on differences from your proposal.

Time	**Task**	**Expected Details from Proposal**	**Actual Details**
Hour 1-2	Research and gather resources	My goal is to create a website interface that allows users to upload .log or .txt files that are Minecraft crash logs. I also want to use API calls to	Learned how to use gemini api calling pretty quickly. Still needed to figure out how to host a website.

Time	Task	Expected Details from Proposal	Actual Details
		knowledgeable LLMs like Gemini to analyze the crash logs to inform the user, since they are all variable in nature to parse through traditional means. This means that I have to learn the basics of making a website/frontend, file uploading, and using Gemini's API.	
Hour 3-4	Design the project structure and plan	I would create all the files I need, figure out the libraries I need, perhaps use version control like Git, make comments in each of the files that outline functions and structures.	Designed the crash_analyzer.py and the basic html website
Hour 5-6	Start coding the basic functionalities	In this part I would at least have the website done, and as a temporary measure it would be a webpage that allows file uploading and maybe spits out the file back to the user as a test.	Learned about website hosting on github, used that. Problem was its used for hosting static websites, have to change how my program worked so the python scripts were hosted on a backend.
Hour 7-8	Test and debug the initial version	In this part I would at least have the website done, and as a temporary measure it would be a webpage that allows file uploading and maybe spits out the file back to the user as a test.	Got the program working and was able to locally host the website with gemini api call functionality before moving it over to github.

Time	Task	Expected Details from Proposal	Actual Details
Hour 9-10	Refine and add advanced features	I would refine the initial conditions given to the Gemini API call, giving it context, examples, specific instructions for output. I could also tell it to make sure the given file is correct and not a log. If I have time, I could consider deploying the website onto something like Microsoft Azure.	Used Render as a free service to host the backend scripts which worked. Made process seamless as pushing new commits to github made the render webservice rebuild and deploy showing new changes which was cool.
Additional			

4. **(55pts) Final Product Description:**
   Include your proposed MVP, Target, and Reach versions.
   i. Minimum Viable Product (MVP):

Webpage that allows file uploading that accesses Gemini's API to analyze the file. This site would be locally hosted on my machine. Target audience would be Minecraft players trying to determine the reason for their game crashing especially with mods.

   ii. Target Product:

Webpage that allows file uploading that accesses Gemini's API to analyze the file. This site would be locally hosted on my machine. The site would give instructions on how to retrieve their log to upload to the site, and the API output would be fine-tuned to tell the user their issue and fix without extra text. Target audience would be Minecraft players trying to determine the reason for their game crashing, especially with mods.

   iii. Reach Version:

Webpage that allows file uploading that accesses Gemini's API to analyze the file. This site would be hosted on a cloud platform like Microsoft Azure. The site would give instructions on how to retrieve their log to upload to the site, and the API output would be fine-tuned to tell the user their issue and fix without extra text. Target audience would be Minecraft players trying to determine the reason for their game crashing, especially with mods. The site might also look fancy and modern. There would be a file storage system such that the LLM can build context from previous logs,

   iv. (20pts) Description of final product including target audience, user story, problem statement, key features, technical details and technologies used. (100 – 150 words)

Key features would be a website, a file uploader box, some text that tells the user what the purpose of the site is, and an output box that returns the response from the Gemini API call. There could be instructions to tell the user how to retrieve their crash logs.

Target audience would just be Minecraft players, for this it would be tailored to a child, teenager, young adult audience.
Platform would just be a webpage that needs internet access.

v. (20pts) Provide a YouTube link to your video demonstration (1–2 minutes, narrated). **Important Note:** Do not upload your video file directly. Instead, upload your video to YouTube and include the video link clearly here in your report. The level of difficulty and detail of the project should be reasonable for 10 hours of work with LLM support. The project should not be something an LLM can solve without significant effort by the developer.
(Be sure to have someone else test that your link is working.)
https://youtu.be/V3EKhOz56EY

vi. (15pts) Any input files, coding files, and test files should be uploaded. Provide a list here of file names and purposes, or any links to live sites or artifacts. Remember code should also be commented. A README file should be created and uploaded so that we have the option to follow your instructions to run your project.

Project Repository & Code Submission Details:
Project Repository (code.vt: https://code.vt.edu/ or (Git-hub only if you are part of virtual global collaboration)): Your repository should be well-organized, documented, and easy to navigate. At a minimum, include the following structure:

- **code/** – All source code files for your project (organized by component or module if applicable).
- **data/** – Any input files, datasets, or configuration files used by your program.
- **tests/** – Test scripts or files demonstrating how your code was verified.
- **docs/** – Supporting materials such as screenshots, reports, or documentation.
- **report/** – This final report document.
- **README.md** – A detailed file describing:
  - Project overview and purpose
  - Video link of your project
  - Installation and setup instructions
  - How to run the program and reproduce results
  - Technologies or libraries used
  - Author(s) and contribution summary

Required:

- Maintain a logical directory structure, do not store all files at the root level.
- Include comments in your code to explain logic and design decisions.

- Keep your repository **private** until grades are released, then you may make it public.

Share access with the following personnel (Add them as collaborators):

GTA Name	Section	Professor
Mona Moghadampanah	83484	P. Sullivan
Yue Shen	83485	O. Emebo
Abdullah Al Noman	83486	O. Emebo
Suraj Vishwanath	83487	P. Sullivan
Juno Bartsch	91578	S. Nizamani

5. **(10pts) Consultation and Use of LLMs:**
   Each student must create a unique project but is allowed to consult with other people and use Large Language Models (LLMs). Describe how you incorporated these resources into your project:

   - **Consultation Description:**
     Describe how you ended up seeking advice or feedback from peers, mentors, or online communities.

I used online communities and forums to help diagnose problems that I had in my code such as StackExchange.

   - **Use of LLMs:**
     Explain how you ended up utilizing LLMs to assist with coding, debugging, learning technologies and concepts, or generating ideas.

I used LLMs in the coding process itself, helping with working with Flask in Python as well as helping with the JavaScript portion of my HTML website file. It also commented on the file that uses Flask.

I also used LLMs in debugging how to have the Gemini output in Markdown since originally it was straight text but most LLMs give their output in markdown syntax.

I also used LLMs in my project itself, incorporating Google Gemini's API to help with the analysis process of the website.