Daniyal Bhatti

Code Foo Document

1. Introduction:

My name is Daniyal and I'm currently a student at the Northern Alberta Institute of Technology working on my diploma. I’m in Digital Media and IT, with a focus on animation. The program I’m in has a wide variety of technical proficiencies for its students and I'm currently working on gaining experience to be more equipped in whichever field I enter.

I heard of Code Foo from a friend, and being a fan of video games since I was little I’ve always known of IGN and their reviews. While I didn’t know IGN provided internships for different disciplines, the opportunity to work with the company would be an amazing experience that could further advance the skills I want to learn for my career.

Some of my favorite games growing up have been series from Nintendo, as their consoles were the only ones we had in our household. So when it came to games outside of the Nintendo library, IGN was always an excellent resource for what games I should play and why. Even to this day before buying a new release, it’s always important to me to check what credible gaming reviewers like IGN have to say.

I think I should be picked for this year’s Code Foo because of my ability to learn on the go. Most of the coding skills I’ve shown have been learned in my own time and not through any classes or programs. I think my quick learning skills and creative perspective make me an asset worthy of being picked. I also always keep up to date with the current gaming landscape, either through new game releases or through emerging technologies being explored by developers.

If I was IGN, I would personally give myself a 10/10. “A New Perspective Joins A Classic”.

2. Hisui's New Power Plant:

To find out how many Voltorbs it would take to produce enough energy to power Jubilife Village, I started by comparing the average size of a voltorb to that of a Pikachu. Both Voltorb and Pikachu are electric-type pokemon and are about 1 foot in height making them similar in size. Though Voltorb on average weighs twice as much as Pikachu (Voltorb being an average of

22 lbs, while Pikachu is 13 lbs) meaning that it is very likely Voltorb may produce and store more electricity in its body. While it isn't confirmed how much electricity Voltorb can produce, Pikachu can learn a particular move known as 10,000,000 Volt Thunderbolt. Though this is a specific move and might not be accurate to how much a Pikachu can produce consistently, we can take a tenth of that and infer that Pikachu can produce a million volts consistently on demand. Voltorb is a bit heavier than Pikachu, and as a result, we can assume Voltorb would be generating slightly more than an average Pikachu. Given this, it is possible that Voltorb can produce 1.5 million volts on command. We can use this to interpret what the constant output of current would be for both of these Pokemon.

According to the Sneak Peek at Pokemon VHS, Pikachu produces a constant current of 1000 amperes. Once again we'll assume Voltorb can produce slightly more, being closer to 1500 amperes. Being that Jubilife Village is a small rural town, it wouldn't need massive amounts of energy to function. If one Voltorb can cleanly produce 1.5 million volts at a time, we can convert this to watts by multiplying its voltage and amperes outage. 1.5 million volts multiplied by 1500 amperes is 22,500,000,000 watts. That is 22,500 megawatts that Voltorb can produce cleanly, which would mean that one Voltorb should suffice in powering Jubilife Village consistently.

3. BackEnd Instructions:

For the programming problem, I chose to develop the API that can read the provided CSV file, store it in a SQL Database, and have the user search said Database using certain parameters.

The program allows the user to input the CodeFoo CSV file, and see it get added to a data table. The data from the CSV file is being read and is separating values between each comma. Those separated values are then being sorted into a table of 16 columns, which is the number of different values in the provided file.

The user can then click a button to submit the seen information into the SQL database with which the program is connected to. Rather than pulling from the file, the program takes the values of the CSV from the previously-created table and enters those values into the SQL database row by row. Since SQL has a character limit and a very specific input index, any of the values that would be too long for the database are changed to a "temp" value. These values wouldn't be needed to search the database so it is better to have them like this.

There is then a button to open another window which holds a table with the contents of the edited database. The three endpoints that allow the user to request data are the search terms parameters. These specific parameters are entered by the user, allowing them to search by ID, Media Type, and Name. The filtering of the data is evident in how the database can be sorted by if the given rating of an entry is higher or lower than an input value of the user.

I also provided a video demonstration of the program working along with the code on Github.

NOTE: The provided CVS document was not compatible with the SQL database. Multiple single quotes and long values between commas would be rejected by the program. To fix this I altered the document slightly to have it work quickly with the program. I believe that this was the correct approach to ensure the program achieved its intended purpose.