Python Project

# Abstract

To develop and chart machine downtime over a specified time frame utilizing the skill matrix as a basis for the requirements.

# Skill Matrix

|  |  |  |
| --- | --- | --- |
| **Skill** | **Location in Code** | **Notes** |
| Output text (print with formatting) |  |  |
| Output formatted numeric data |  |  |
| Simple variables (string, integer, float) |  |  |
| Complex Variables (objects, multi-part) |  |  |
| Conditionals (standard if/then/else, switch) |  |  |
| Loops (for loops and/or while loops, iterators) |  |  |
| Array (single dimension) |  |  |
| Multidimensional array or dataframe |  |  |
| Function with input (takes input) |  |  |
| Function with output (returns output) |  |  |
| Data/Math Library (pandas, numpy) |  |  |
| Data Library (plotting) |  |  |

# Problem Statement

Tasked with creating a web module that can track machine status in a production shop. The module will include Status of the machines, hoe many times each machine goes down, and for how long. The reporting with be graph based with downtime in hours, and the number of times it went down. Statistical analysis will be done on the final numbers, extrapolating the mean down time, how many times all machines went down, finally giving a total loss on downtime. A special rule must be used to only calculate downtime excluding weekends. Below are the formulas:

# Methodology

Once the models are created for the application, the business logic for calculating the down time will utilize Numpy. Numpy has a function built in that will check to see if a date is a business day or not. By giving Numpy a date range we can extrapolate only the business weekday. Once we have the number of business days the machine is down, we then calculate the date time difference into hours. This allows for a quick and easy calculation. If a machine does not have a finished downtime report i.e.(not repaired) then that time is not calculated in the report.

When the user wants to run a report, they simply go to the predetermined date ranges and run the report. They are then sent to the URL with the report they wanted to run. A for loop allows the script to cycle through all the data needed for calculations. This will display chart data of the different machines that have downtime reports attached to them within that date range. The various charts available are Life, Current Year, with Current Year Q1-Q4.

The charts are created with a JavaScript library called Charts.js. This allows for extreme customization on the chart and displaying of the data. Basically, the chart object is comprised of an JSON API that handles all the data and options.

The information can allow upper management to see if a machine needs to be replaced before it becomes a serious problem. Currently we rely on our Machine Maintenance’s memory to see if a machine needs to be replaced. With concrete evidence all stake holders are held accountable for the health of the machine shop.

# Solution

By utilizing two for loops we can extrapolate the data and calculate the machine health.

# Conclusion

Finding a glaring problem and having the skills to implement a solution that can better a company you work for is an awesome feeling.

# Appendix