Release Notes 2021-04-05 : : This is another work-in-progress I'm pushing out to assess interest and it is used in works already put online in note form. The software is not complete but I'm using it to optimize diet as described herein. The core software a c++ program that allow entry and validation of arbitrary events with word definitions completely up to the user except for a few reserved words. If there are competing products I'll include those for comparison once I am aware they exist or start using those.

MUQED: a Multi-Use Quantitative Event Diary For Dog Diet Analysis

Mike Marchywka*
306 Charles Cox Dr, Canton GA 30115
(Dated: April 5, 2021)

MUQED is a simple text based system to chronicle events such as inputs and outcomes suitable for a variety of data exploration projects. A terse text format is designed for simple entry as events occur while producing fully machine readable sanity checked data files. While designed for human entry, in theory an automated device, such as a barcode reader or dispenser involved in the events, may be able to generate conforming input directly. MUQED has been generalized from software to collate manually entered dog diets and all the examples are based on this real world application. A complete system is illustrated from data entry to report generation. The system outline includes elements of quantitative diaries, bill of materials functionalities, inventory control, data input validation etc. Nothing is terribly sophisticated but a lot of tedious code needed to be assembled to make the system work. It is not known how it compares to any similar systems although the text based approach with existing email integration (Mikemail) allows for a variety of user interfaces to be added with simple code on various platforms such as phones or small devices. No code is needed at all on systems with email capability as long as the user is happy with text data entry but in the future a dedicated MIME type and custom handler could improve user interaction options. The target platform for the current code is a low end linux system such as a laptop capable of running Ubuntu 16 or above. Data entry and storage can all be local or decentralized with communication through email allowing access via existing IMAP accounts without a dedicated web site or interface and without static IP addresses for the "server".

*Electronic address: marchywka@hotmail.com; to cite or credit this work, see BibTeX in Appendix K

Contents

1. HIGHLIGHTS

1. **Users: Home, Academic, Professional Diary Authors** Anyone wishing to record sanity checked event diaries - from high school students to retired PhD's - for later analysis by machine and human.
2. **Uses: Research, Logging, Text Processing** Data types may include Diet and health results, any events you would "log" such as maintence or service, event recording from automated event sources, ad hoc data exploration from any event lists. May be useful as a command line utility for some computer related tasks although time resolution currently is limited to minutes The architecture allows for concise data files to be expanded in various decompositions - more complicated and aware than sed but not quite a bill-of-materials system either.
3. **Simple Data Entry may be automated** Users enter data based on names they create which can be local shorthand expanded to more well known or canonical terms later. Text based allows easy keyboarding and interface to automated devices or GUI.
4. **First step, List items** Make an initial list of items of interest. Typical usage would require a list of nouns as "inputs" or "outcomes."
5. **Then Record Events** Text entry aided by templates derived from prior inputs - most useful for recurring events.
6. **Validate** Insure the nouns are known with legal adjectives to allow error correction quickly. Edit data file or improve noun file as needed.
7. **Explore Data** Data can be expanded into various space separated value files that should easily import into R or Excel etc.
8. **Use Related Tools** The time based quantitative event format is a little unusual and some tools are provided to deal with that. Currently this is just a few grophing programs to create SVG output with dates on the x axis.
9. **Interface to Data Source** The simple text format should work with various automated event generators with minimal translation. Examples contemplated include barbode or various symbol readers, software generated events, environmental sensors such as thermometers or with motion detection. A facility is included for referncing data files for video for example.
10. **Should adapt to GUI** The file formats should allow a menu driven system to be populated with allowed entries although there is currently no need in my current usage.

2. INTRODUCTION

In the beginning of the personal computer era, there was some anticipation that people would buy computers to organize their recipes. While this application never was a big driver, MUQED sort of satisfies that prophecy although right now it only tracks ingredients and not the cooking instructions. The simple low resource approach makes this system look like something that could have been run on a TRS-80 for many personal and small businesss applications in the 1980's. Further, the internal implementation is based largely on a set of string grids (parsed SSV files) similar to VisiCalc or now Excel.

As primitive as the code may seem and regardless of any competing approaches, today voice and other input formats with AI interpretation may have benefits, the system does make it possible to record a lot of time based inputs and outputs with some validation at time of entry, produce machine readable data files, etc without the need to design confining forms or DB structures right away.

It was not clear what to call this set of functions although in a search for a name the term "quantitative diary" appears in some literature [2] and a quick google search turns up various apps that use the term. The system described here need not run on the data collection platform and should communicate with any system that has email capability.

The original motivation was to record diets for a group of dogs and compare to their overall health outcomes. This sounded simple and is something a computer should be easily able to do but the little details add up. Data entry after the chaos of feeding a group of dogs can be complicated and ideally both planned and post hoc diet events should be accomodated (making a menu first and just recording amount eaten may be a lot easier to keep accurate but stuff still happens). While diet optimization itself is of general interest, during this work a still unresolved issue came up with dilated cardiomyopathy in dogs thought to be related to their diets [6]. A system like this used by pet owners who record all ingested items including treats and medicines along with health outcomes may be quite good for hypothesis generation.

The process was developed over a few years after initial indications that significant good and bad effects could be observed depending on what the dogs were fed. Meals were generally "made up" at the time of feeding and no *a priori* recipes were made. Ingredients could include commercial kibbles, canned foods, foods generally made for human consumption, vitamins and supplements, medicines , and treats. With 10 or so dogs and more than 10 varying ingredients it can be difficult to remember what each one got for the few minutes to get to a keyboard but with the right entry approach and shared meals it is not impossible with an acceptable error rate. Besides formulating the meals, there is always the problem of who actually ends up eating what as some may not be hungry and some eat each others' foods. Typically however meals and additions tend to be smiilar for most of the dogs and from day to day. The variations are exceptions that are easy to rememebr if the "base" is known. Ultimately then a lot of the work was over with simple copy from prior day and paste. As the dogs ate similar meals it was possible to define daily recipes and just note how much each actually ate although error bounds on this amount could vary a lot as they ate each others' food. The problem with error bounds, due to memory or sloppy measurement, or observational limitations, was not easy to fix during entry and remains a bit open. Once a simple syntax was available for defining a meal or recipe, indicating share amounts was not difficult. Ideally, perhaps a barcode reader could be used on most ingredients with manual entry of quantities during preparation but even then after the fact entry may be less distracting as just keeping track of who gets which bowl can be a challenge. Entered data should be as direct as possible to minimize conversion and interpretation errors and reduce the influence of assumptions on later analysis. For example, if teaspoons are measured you want to record an amount as teaspons rather than the target units of say milligrams. If errors occur, you want to note those. In any case, the first workable system came up with a terse line oriented text file format that could be easily updated in the old fashioned text editor "vi" [1]

Thinking outloud
Note wikipedia only seems to have BibTeX info on select pages...

. So, for example, if a "morning snack" was made on 2020-09-27 using various food items plus some vitamins and a dog named "Beauty" ate 3/8 of the batch, the data file may contain two lines like this,

```
2020-09-27 & AMSNACK 0505AM b7ngnc cbbroth carrot garlic 11PC .5 eggo3 2.0 bulk lecithin 900mg bulk taurine
70mg bs B-3 500mg nutricost pantothenate carlson glycinate 2.5mg Cu 400mg tryptophan 650mg leucine
1300mg lysinehcl 650mg threonine \\
2020-09-27 & Beauty .375 AMSNACK ( .5 eggo3 b20ngnc carrot 0630AM .25 AMSNACK 0930AMSNACK 1.0 PMDINNER .25
    ) PMSNACK \\
```

The first line defines the contents of the morning or AM snack as well as its earliest serving time , 5:05AM, and creates the recipe name "AMSNACK." The second indicates that Beauty consumed some of it with parenthesis around the anticipated consumption later in the day. The words and quantitites can be largely common, well known terms or shorthand known only to the person doing the entry. In this case, a word like "b7ngnc" means " ground beef containing 7 percent fat with no other things added." Carrot means carrot. Note in both of these no quantities are given because there was no interest in tracking their amounts although with interspersed purchase information a run rate and estimate could be figured out later if it was important. For more critical items, quantities precede the item - for example, the "AMSNACK" contained 400mg of tryptophan although this is a derived quantity and the actual amount was measured in teaspoons. Observations or concerns can be noted on similar dated lines with keywords like "NOTE" in place of a recipe or consumer name.

This article describes the software to handle this line oriented format of recipe(or "macro") descriptions and consumption records augmented with various features for ad hoc observations and exceptions. The addition of reminder or mnemonic words to the format and template helped greatly with the memory issues and may even be easier than drop down menu selections.

[1] https://en.wikipedia.org/wiki/Vi

2.1. Motivation, Scientific Rationale and Precedents for Utility of Complex Data in Informal Settings

TODO : I thought this was a good idea but not sure it is good here Uncontrolled multivariate "real world" situations may not sound like a good setting for scientific investigation. However, there are many compelling reasons to record or design complicated events such as feeding and outcome that can be explored based on existing theory and historical precedents. Folklore and informal hypotheses routinely turn into perceived facts and documenting the complexity of these situations first should demonstrate how questionable conclusions cam be. Food associations that exist as "common knowledge" may not translate into clinical trial successes because assumptions about the variable that matters are wrong and there is no easy way to find the missing ones. Complicated systems may not respond similarly to small changes in one variable depending on the state of the others. However, accidental discoveries are sometimes known to be proven correct much later.

There are plenty of accounts of trial and error guided by only limited theory eventually producing useful products in the laboratory [1] but it may be possible to extend "trial-and-error" to a more complicated but real world setting. Today, many diseases are investigated with single entities against model diseases. These are easier to understand than real cases but there is no asurance the models reflect cause and effect in a real disease or that the context for the single entity is even useful. A more complete understanding of real conditions, including simple old age, may be better at isolating important factors as simple as diet. Single agent controlled trials are not expected to be optimal but the hope is to find small incremental benefits by averging out the other factors with statistics. This often results in ambiguous or even misleading results even with good faith blinding and randomization. An alternative would be to "take your best shot" considering many components and generate an error signal when you " see what happens." This introduces a larger parameter space but with a good initial guess and decent error signals may be a more tractable way to achieve an optimized result. In the event of unexpected adverse events, as with the current issues with cardiomyopathy in dogs, the diary provides some data that may not otherwise be available.

Even in a well controlled setting against real diseases, such as clinical trials, awareness of the unexpected can turn a failure into a success as is the well known case of Viagra [4] [3].

Anemia may create thyroid hormone intolerance among those with hypothyroidism [11]. Controlled tests typically are more difficult and may be best appraoched when the important variables are known or suspected.

Additionally MUQED is designed for informal settings with the expectation that a high school student could use this at home. Even if imprescise, if enough people create isolated case data that can be aggregated perhaps new and interesting observations will emerge.

During the course of this work, two issues emerged of potential relevance. The unresolved link between dog foods and heart problems is mentioned elsewhere[6] and the recently emerged covid-19 is thought to be treated with various vitamins but evidence is lacking. Proponents of various vitamins may be able to organize efforts using this software to collate overall diet data with all health outcomes among those willing to share. In fact, I have used the legacy system to offer ideas based on the apparent importanance of some nutrients to older dogs their overall environment [10]. as age appears to be a big risk factor for severe covid.

2.2. Data Loss and false equivalence of false equivalents

One problem with exploratory work and hypothesis generation is complete ignorance about what is important to control and observe. This is especially a problem with food and food association studies where a given food may be equated with one trendy molecule, sich as citrus fruit with vitamin C or more recently with flavinoids. Even molecules can be nutritionally different depending on their overall environment [9].

So, if there is a desire to analyze a diet in terms of just one group of elements such as vitamins or amino acids, you may want to know about the "other stuff" that makes them more soluble and certainly not equate proteins or foods with a mixture of amino acids. Ideally then the diary must track the raw details and let translation occur later when caveats can be prominent.

Different recipes, components, explosions and analyses may not even be mutually consistent and breakdown in different schemes may be important to understand results.

Thinking outloud
need to say more about false equivalents of components versus sum with digestive issues

2.3. Quality Assurance, Logistics and Commercial Issues

A medium scale effort with several users may require some help with logistics and even non-profit groups may have sponsors who would like an accounting of money and products. These data streams can also help validate the target data as they should all be more or less consistent with each other. Small errors in a spoon or scale or technique may show up in larger integrated measures. Right now, little of this is accomodated except for the following,

1. Recognition of " BomTe*X* " and the utility of a "bill of materials" feature similar to BibTeX .
2. Skeleton for an inventory line type (INV*) specification as yet completely TBD
3. Outline ideas for a "CONSUMER" record that could record "outcome" nouns but also track affiliation, breed and demographic info, pictures,etc. Optional validation and auto update on reference.

2.4. Final Products- reports and hypotheses

The utility of the data entry and file contents eventually need to be judged by the ability to get useful information out. Successful data extraction and analysis verify the software but some issues remain unresolved. Most of the diet history and similarly recorded outcomes like weight or vomting can be included quantitatively and subsequently put into a portable CSV or SSV format for use with any existing data analysis packages. But error bounds and subjective events can be noted in human readable form for later exploitation. Two sample plots from the legacy system using related analysis code are included [8] to illustrate the nature of the data and some issues with presentation.

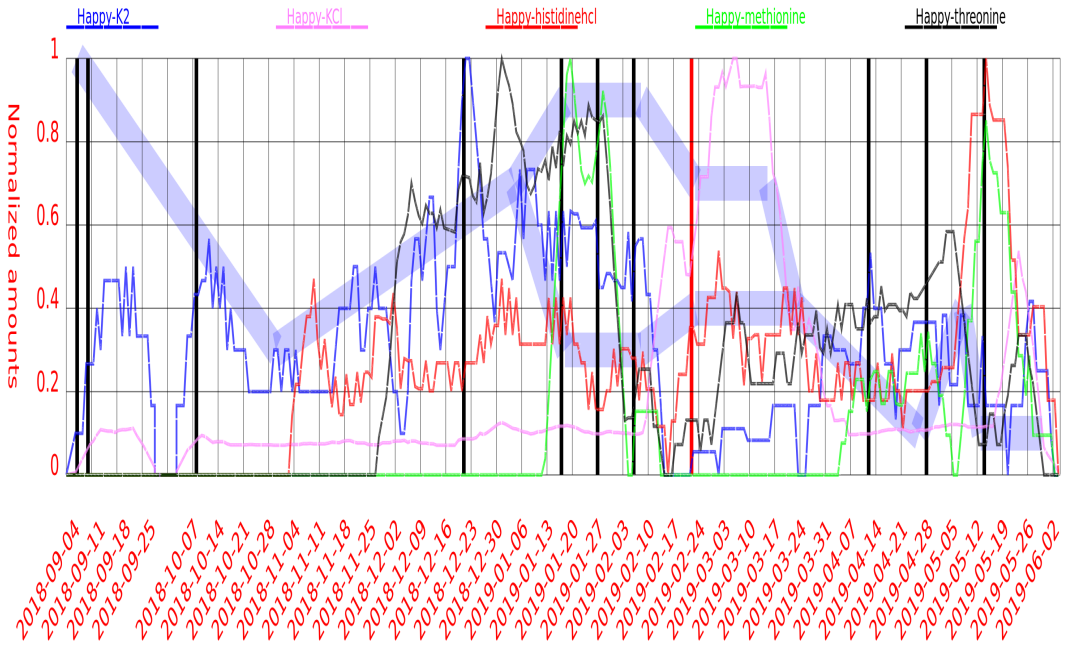


FIG. 1: One plot derived from data collected with the legacy system. Smoothed daily consumption of various vitamins is shown along with a thick blue ribbon derived from subjective observations noted in human readable form.

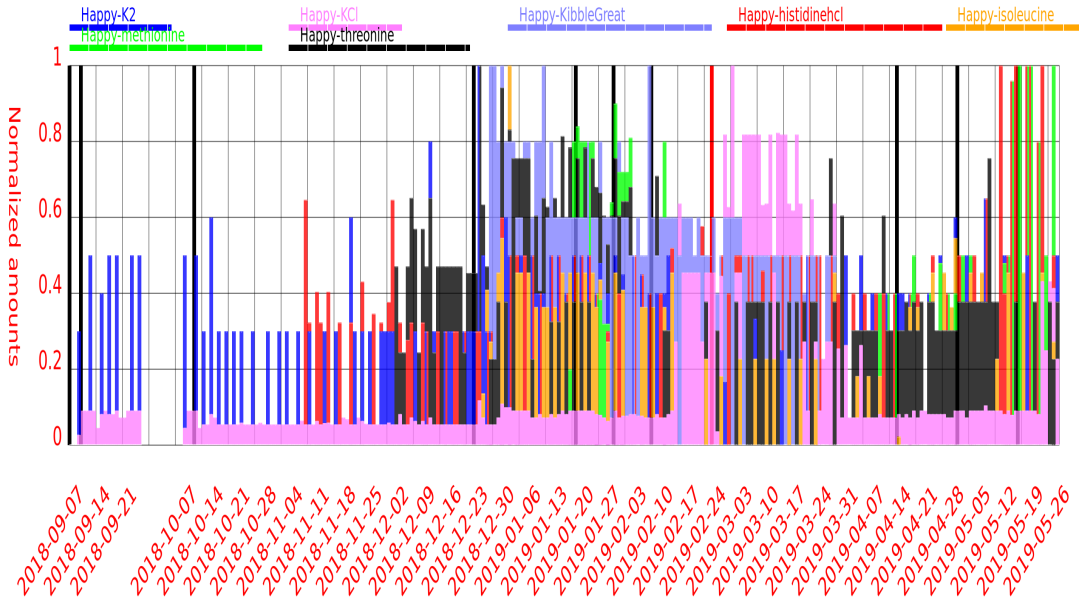


FIG. 2: Another plot showing daily amounts of select nurients consumed in histogram form. One feeding theory involved rotating nutrients which can interfere or compete with each other. This made it necessary to find smoothing algorithms implemented either in the report generator or plotting code. However, time-of-day relationships also became critical and the system allowed time information to be retained.

The legacy or original system did have some useful features including,

1. Template generation for each day derived from a fixed list of likely meals, ingredients, and consumers.
2. Dates described as day serial numbers for ease of period calculations.
3. Macro or recipe concepts allowing meals to be defined and split among the dogs or consumers.
4. Comment and DMEL features for adhoc observations that may be made sysematic .
5. Mnemonic words are allowed to remind user of possible ingredients- these are often actual ingredients prefixed with "no". This is a similar concept to a menu.
6. Sanity checking for some data entry errors that may not be clear on graphs or statistics.
7. Hard coded vocabulary as recompiling was just as easy as adding to a data file and more flexible.

Several problems arose however leading to the following changes in MUQED which continue to be refined along with the implementation and code overall,

1. Parameters were defined to allow nouns and units and features to be loaded from data files with adequate flexibility and extensible features with recompilation.
2. Unit conversion features were included to allow data entry in raw terms with converion to meaningful uniform units saved until later also allowing corrections to conversion factors like density at any time.
3. "of" syntax for success of feeding(" 10 percent of 200 grams ") as quantity eaten alone was not always the most useful for health indications.
4. Expansion features for most nouns to be exploded into components to the extent these are known.
5. Noun groups for reporting and possible menu tree population. This is most useful when details are known, multivitamins for example, althouh exploding by rank as on dog food labels is possible- see next feature.
6. Noun attribute skeletons for "design rule checking" although most of this is expected to occur during later analysis. If meals are designed *a priori* this may be useful to avoid errors with medicines and some nutrients. Common diet deign rules may include total amount of some time period (NOAEL) and interactions with other nutrients in temporal or spatial proximity. Although **TODO : the former may require knowledge of consumer attributes like weight.**
7. Content description by rank for qualitative decomposition into exploded parts a such as may occur with ingredient lists. Inert ingredients can be tracked to some extent and they are not always inert especially with dogs.
8. Automated extraction of a template for today's entries based on previous day as diets tended to be quite consistent as long as mnemonic null words were allowed. This may not be the case for other applications however. **TODO : Maybe add fixed contributions from a file. Need to consider an instance in a multiuser server setting not just local usage.**
9. **TODO : Better support for quantitative and qualitative error bounds or DMEL issues and addition in quadrature.**
10. Integration with Mikemail for demo or remote usage and evaluate the email form concept for exchange and modification of data files and form based data.

MUQED was created from scratch based on the concern that many idiosyncratic problems would turn up in the real world, or real kitchen, deployment that would not likely be incorporated in simiar lab or consumer open source software packages. Currently however there are not a lot of idiosyncratic features and likely satisfactory open source packages exist to do similar tasks. While a comparison with any of these would be useful a quick google search for some name candidates did not immediately turn up anything open source.**TODO : Compare to competing products regardless of source and business issues.**

1. DearScholar [5] https://peterkruyen.net/dearscholar.html https://github.com/pmkruyen/dearscholar This looks like a more typical mobile app and their messaging may work better with Mikemail **TODO : find a real name for Mikemail** . MUQED use of email provides some security and allows the correspondences to be integrated in situ with other communications if you don't like SMS/MMS.

2.5. Evolution and Development Paths

Probably a variety of "types" or "forms" could be added to accomodate specific types of data. One obvious archtype after a complete "recipe" would be a consumer. Hopefully after working through a recipe and consumer and maybe some additional archtypes, a general pattern will emerge for end users to extend.

3. OVERVIEW

The MUQED ecosystem can be described as containing the following central components, ignoring some of the other support functions for now. There is no real fixed relationship among the building blocks but their interactions weill become clear as the system is described.

1. Diet Diary Data File - this is the primary diary data that can be edited by a human (in vi nonetheless) and validated against various supporting files.
2. Diet Diary Form - A C++ class that deals with the diary data and the supporting files to validate input and generate output reports files.
3. muqed_util - A general purpose utliity program containing a diet diary form that supports a set of commands to make it useful locally and remotely.
4. mudaily.txt - A bash script that invokes muqed_util to perform repetitive functions on a given installation such as validating the input and generating reports.
5. Mikemail - A system for exchanging data that allows integration with existing corresponences with potential for later development using custom MIME types. This will result in longer response times compared to a web server but the tradeoff is expcected to be beneficial overall given the nature of this task. Bscially a fancy replacement for fetchmail and procmail, can also be used for remote monitoring and control.
6. muqed.tex,muqed.pdf - This file , the documentation for these components that overall comprise MUQED .

A MUQED installation would likely include the source code, some initial unit and nouns files, sample data files, and an actual user guide and distinct developer guide. The installation is envisioned to be either fully local or in a

server mode with communication over email (Mikemail). The server mode is largely the local mode integrated into IMAP polling code to interpret requests and point to specific data and supporting files for various user groups. In local mode, a user can edit the files with any text editor and run a command line program to execute MUQED functions. Each application or deployment of MUQED will likely require application specific noun files and a different data file. A dynamic set of units and nouns is used to validate and add meaning to the target diary or observation file contents. At each event, a user enters a sequence of times and quantitities related to each noun for each consumer. With the dog diet diary, this is some amount of a given recipe eaten by a given dog at a given time and date. In theory meals could be pre-planned and mostly entered a priori but in normal usage it was easier to enter after the feeding.

Setup requires building the relevant code or registering with a given server and populating the support files with baseline terms. The stock units file should contain most SI units, some common english and kitchen units along with some conveinence terms. Noun conventions could vary substantially among users and code words for specific items could be alternatively entered as generic nouns and differentiating adjectives.

Once setup, routine additions to the nouns may come up. Consumers are auto added as encountered and users need to check reports for typographical errors. Most of the daily usage is getting a template for a new day, entering the recipes and consumption as well as any outcomes or observations, having the file validated, edit until ok, and then add to the diary file.

Reports are generated using either the related software or any standard data analysis code capable of reading space separated tabular data.

4. RUNNING THE TEST CODE

Thinking outloud
This is documentation of the current code sitting in the Mikemail project and not likely to resemble a distribution but early work will begin from here

The test code consists of the Mikemail build script "run_mikemail" and a set of scripts designed to test the MUQED class in stand alone mode,

Code: Demo Scripts

```
marchywka@happy:/home/documents/cpp/proj/mikemail$ ls scripts
demo-commit.txt demo-markup.txt demo-report.txt demo-template.txt
```

Compile the stand alone form. The include path needs to pick up part of the mjm header library. The boost functions are really only needed in the server mode (and are included in later c++ versions) to make file paths secure.

Code: Building Standalone Diet Form

```
 ./run_mikemail -build-diet
g++ -std=gnu++11 -DTEST_MJM_DIET_DIARY_FORM -I. -I../../mjm/hlib -I../../mjm/num -Wall -gdwarf-3 -O0 -x c++
     mjm_diet_diary_form.h -o mjm_diet_diary_form.out -lpthread -lreadline -lboost_filesystem -
    lboost_system
```

Create a "load_demo.txt" script used by the other scripts to load constant stuff for the demo. Then run the report generator script and look in the "./out".

Code: Generate Reports

```
./run_mikemail -diet-template demo load_demo.txt

 2719 ./scripts/demo-report.txt
 2720 cat out/dog_used.txt
 2721 ls -al out
ls -al out
total 1672
drwxrwxr-x 2 marchywka marchywka 4096 Sep 27 20:24 .
drwxrwxr-x 14 marchywka marchywka 4096 Sep 28 06:58 ..
-rw-rw-r-- 1 marchywka marchywka 34609 Sep 28 06:59 dog_daily.txt
-rw-rw-r-- 1 marchywka marchywka 223921 Sep 28 06:59 dog_glob.txt
-rw-rw-r-- 1 marchywka marchywka  799 Sep 28 06:59 dog_used.txt
-rw-rw-r-- 1 marchywka marchywka 2425 Sep 27 20:51 template.txt
-rw-rw-r-- 1 marchywka marchywka 1432752 Sep 28 06:59 xxx
```

The reports have lines of the form as shown below. The three output fileshow a daily summary of diet by dog expanded to some degree ("daily"), a glob of consumption records for each consumption event("glob") , and an summary of the total amounts of each product mentioned for sanity check and inventory control("used"). The first report, dog_daily.txt, is likely the most useful for data analysis. Several options exist to make it easier to use with standard analysis tools or it can be used with linc_graph

Code: Report Lines

```
il -n2 out/*.txt
==> out/dog_daily.txt <==
18527 2020-09-22 Chipper B-3 13.125 mg Cu 0.9375 mg b7ngnc 0.375 - carrot 0.375 - citrate 0.046875 tsp
    ctbroth 0.375 - eggo3 0.09375 - garlic 0.375 - kcl 0.046875 tsp lecithin 450 lecu leucine 237 mg
    lysinehcl 243.75 mg pantothenate 93.75 mg taurine 168.75 mg threonine 120 mg tryptophan 75 mg
18533 2020-09-28 Beauty B-3 26.25 mg Cu 0.9375 mg b7ngnc 0.375 - carrot 0.375 - citrate 0.09375 tsp ctbroth
     0.375 - eggo3 0.1875 - garlic 0.375 - kcl 0.09375 tsp lecithin 900 lecu leucine 243.75 mg lysinehcl
    487.5 mg pantothenate 187.5 mg taurine 337.5 mg threonine 243.75 mg tryptophan 150 mg

==> out/dog_glob.txt <==
18533 2020-09-28 5.08333 Beauty lysinehcl 487.5 mg - 487.5 mg
18533 2020-09-28 5.08333 Beauty threonine 243.75 mg - 243.75 mg

==> out/dog_used.txt <==
valine 5.45 gram
worm 2 -
```

After seeing how the "out-of-the-box" demo files work, they can be edited to demonstrate how new data are included. A template file allows for each new entry to be made more easily when the current data is expected to be a small perturbation of the prior day's. Generate a template file,

Code: Make Template for New Entry

```
 2783 ./scripts/demo-template.txt
 2785 cat out/template.txt
```

Edit the file out/template.txt to make it look like this, and then submit it for markup or validation,

Code: Markup New Entry

```
cat out/template.txt

2020-09-27 & AMSNACK 0505AM b7ngnc cbbroth carrot garlic 11PC .5 eggo3 2.0 bulk lecithin 900mg bulk taurine
     70mg bs B-3 500mg nutricost pantothenate carlson glycinate 2.5mg Cu 400mg tryptophan 650mg leucine
    1300mg lysinehcl 650mg threonine \\
2020-09-27 & Beauty .375 AMSNACK .5 eggo3 b20ngnc carrot 0630AM .25 AMSNACK 0930AMSNACK 1.0 PMDINNER .25
    PMSNACK \\


marchywka@happy:/home/documents/cpp/proj/mikemail$ ./scripts/demo-markup.txt
marchywka@happy:/home/documents/cpp/proj/mikemail$ cat out/markup.txt
2020-09-28 & AMSNACK 0505AM b7ngnc cbbroth carrot garlic 11PC .5 eggo3 2.0 bulk lecithin 900mg bulk taurine
     70mg bs B-3 500mg nutricost pantothenate carlson glycinate 2.5mg Cu 400mg tryptophan 650mg leucine
    1300mg lysinehcl 650mg threonine \
2020-09-28 & Beauty .375 AMSNACK .5 eggo3 b20ngnc carrot 0630AM .25 AMSNACK 0930AMSNACK 1.0 PMDINNER .25
    PMSNACK \
#  m_w= m_pos=17 m_msg= unused adjectives at end of line
```

Note that there is a new line beginning wiht a pound sign describing a problem. The other items beyond "AMSNACK? are not defined and treated as adjectives leading to this message. For a partial day entry, just comment them out (by adding a left parenthesis after "AMSNACK" and a right one just before the trailing backslash. NOTE parenthesis must be seaparated from other words by a space) and try again. **TODO : the trailing backslash is not consistent double or single but not needed now anyway** . Diary entries may also contain "ignore" nouns or mnemonic placeholders that sort of function as a menu in this setting. For example, the template may contain a word like " notyrosine " which simply reminds the user tyrosine could have been given and that it is entered by adding a quantity and removing the "no." If these are indicated as "ignore" nouns in the noun file, they will be flagged if retained with a non-zero quantity suggestive of an error.

Code: Fixed Entry

```
marchywka@happy:/home/documents/cpp/proj/mikemail$ vi out/template.txt
marchywka@happy:/home/documents/cpp/proj/mikemail$ cat out/template.txt
2020-09-28 & AMSNACK 0505AM b7ngnc cbbroth carrot garlic 11PC .5 eggo3 2.0 bulk lecithin 900mg bulk taurine
     70mg bs B-3 500mg nutricost pantothenate carlson glycinate 2.5mg Cu 400mg tryptophan 650mg leucine
    1300mg lysinehcl 650mg threonine \\
2020-09-28 & Beauty .375 AMSNACK ( .5 eggo3 b20ngnc carrot 0630AM .25 AMSNACK 0930AMSNACK 1.0 PMDINNER .25
    PMSNACK ) \\
marchywka@happy:/home/documents/cpp/proj/mikemail$ ./scripts/demo-markup.txt
marchywka@happy:/home/documents/cpp/proj/mikemail$ cat out/markup.txt
2020-09-28 & AMSNACK 0505AM b7ngnc cbbroth carrot garlic 11PC .5 eggo3 2.0 bulk lecithin 900mg bulk taurine
     70mg bs B-3 500mg nutricost pantothenate carlson glycinate 2.5mg Cu 400mg tryptophan 650mg leucine
    1300mg lysinehcl 650mg threonine \
2020-09-28 & Beauty .375 AMSNACK ( .5 eggo3 b20ngnc carrot 0630AM .25 AMSNACK 0930AMSNACK 1.0 PMDINNER .25
    PMSNACK ) \
```

With no errors, it can now be added to the file. The legacy local script would copy a template file to the clipboard and the user would just paste it into the diary data file with vi. **TODO : locally, you can just copy and paste the template into the main data file. The remote version needs a merge facility.** Run the commit script and verify the data file contains today's entry,

Code: Add New Entry and Verify

```
 2865  ./scripts/demo-commit.txt
 2866  cat out/xxx
 2867  cat demo/data.txt
 2868  ./scripts/demo-report.txt
 2869  history
marchywka@happy:/home/documents/cpp/proj/mikemail$ cat demo/data.txt | tail
2020-09-22 & Happy .125 AMSNACK ( 0930AMSNACK .25 PMDINNER .125 PMSNACK ) \
2020-09-22 & Claire .125 AMSNACK ( 0930AMSNACK 1.0 PMDINNER .125 PMSNACK ) \
2020-09-22 & Spicey 0630AM .125 AMSNACKB ( 1230PM ctbrothbs .0625 PMSNACK ) \
2020-09-22 & ( Dexter .05 PMSNACK ) \
2020-09-22 & Autumn .125 AMSNACK ( 2 PMDINNER .125 PMSNACK .375 RPMSNACKB ) \
2020-09-22 & Andy .375 AMSNACKB ( .375 PMSNACKB 1640PM 50mg doxycycline canned .25 RPMSNACKB ) \
2020-09-22 & Chipper .375 AMSNACKB ( .375 PMSNACKB .25 RPMSNACKB ) \

2020-09-28 & AMSNACK 0505AM b7ngnc cbbroth carrot garlic 11PC .5 eggo3 2.0 bulk lecithin 900mg bulk taurine
     70mg bs B-3 500mg nutricost pantothenate carlson glycinate 2.5mg Cu 400mg tryptophan 650mg leucine
    1300mg lysinehcl 650mg threonine
2020-09-28 & Beauty .375 AMSNACK ( .5 eggo3 b20ngnc carrot 0630AM .25 AMSNACK 0930AMSNACK 1.0 PMDINNER .25
    PMSNACK )
```

5. RUNNING AND USING A LOCAL INSTALL

For actual usage, a script "mudaily.txt" is provided that provides most common features needed for daily usage. This could be pointed to the test code directories but likely would be used with an installed or deployed set of locations.

For example, /home/scripts/script_data/cases which also ends up with additional files for Mikemail and development files,

Code: Copy data files into an install directory

```
marchywka@happy:/home/documents/latex/proj/muqed$ ls -al /home/scripts/script_data/cases
total 2284
drwxrwxrwx 3 marchywka marchywka 4096 Oct 6 06:34 .
drwxrwxrwx 3     777 root       4096 Oct 4 04:59 ..
-rw-rw-r-- 1 marchywka marchywka 3594 Oct 4 04:59 aka_nouns.txt
-rw-rw-r-- 1 marchywka marchywka  72 Oct 4 04:59 badcmd.txt
-rw-rw-r-- 1 marchywka marchywka 5423 Oct 4 04:59 canon_nouns_compatible.txt
-rw-rw-r-- 1 marchywka marchywka 5599 Oct 4 07:39 canon_nouns.txt
-rw-rw-r-- 1 marchywka marchywka 3646 Oct 4 04:59 diary.txt
-rw-rw-r-- 1 marchywka marchywka 346 Oct 4 04:59 dumm.txt
-rw-rw-r-- 1 marchywka marchywka 241 Oct 4 04:59 dum_nouns.txt
-rw-rw-r-- 1 marchywka marchywka  27 Oct 4 04:59 ignores.txt
-rw-rw-r-- 1 marchywka marchywka 2254904 Oct 5 07:37 newest_data.txt
-rw-rw-r-- 1 marchywka marchywka 122 Oct 4 04:59 newuser.txt
-rw-rw-r-- 1 marchywka marchywka   2 Oct 4 04:59 null.txt
drwxrwxr-x 2 marchywka marchywka 4096 Oct 6 05:51 out
-rw-rw-r-- 1 marchywka marchywka  26 Oct 4 04:59 register.txt
-rw-rw-r-- 1 marchywka marchywka  22 Oct 4 04:59 reserveds.txt
-rw-rw-r-- 1 marchywka marchywka 1089 Oct 4 04:59 units.txt
-rw-rw-r-- 1 marchywka marchywka  58 Oct 4 04:59 users.txt
-rw-rw-r-- 1 marchywka marchywka  22 Oct 4 04:59 welcome.txt
```

Setup the paths in "mudaily.txt" or use whatever variables exist for data organization. **mrddir** is the data file directory. As all data files are dynamic they are in the same dir - diary data along with nouns and units etc. **tempdir** is for temperorary files that are easy to recreate. **outdir** is for outputs like reports.

Code: Confirugation variables in mudaily.txt

```
. includes
# really not needed as should have been done
. setup

tmpdir="$SE_TEMP_DIR"
mrddir="$SE_SCRIPT_DATA/cases"
outdir="$SE_SCRIPT_DATA/cases/out"
data_fil="$mrddir/newest_data.txt"
templatefil="$tmpdir/cases_template.txt"
markupfil="$tmpdir/cases_markup.txt"
# on path
mjm_exe="mjm_diet_diary_form.out"
runlog="$tmpdir/run_log.txt"
```

As with the legacy system, daily usage consists of just running one command. In this case, "mudaily.txt -daily". This opens "vi" on a marked up copy of the diary data file with a template on the clipboard. The user can paste it at the end of the file or just ignore it. Any new data are entered and errors corrected. Since it overwrites the original data file if changes were made, it makes a backup relying on a script called "backupdir" to provide the directory for backups. The backup uses a script called "newbackup" which by default picks names based on day only. Note that the backups are based on day only and recovery requires restart of entire day's inputs. Adding "-time" in the script makes the names include time down to seconds.

Code: Confirugation variables in mudaily.txt

```
 2322 mudaily.txt -daily
marchywka@happy:/home/documents/latex/proj/muqed$ ls `backupdir`/*mudaily*
/home/marchywka/mnt/bk/static/jcasesmudaily-2020Y-10+04.zip
/home/marchywka/mnt/bk/static/jcasesmudaily-2020Y-10+05.zip
/home/marchywka/mnt/bk/static/jcasesmudaily-2020Y-10+06.zip

cat `which mudaily.txt` | grep newbackup
#. newbackup -filter filter_dudaily -base "casesmudaily"
. newbackup -time -filter filter_dudaily -base "casesmudaily"
. newbackup -filter filter_daily -base "casesdaily"
```

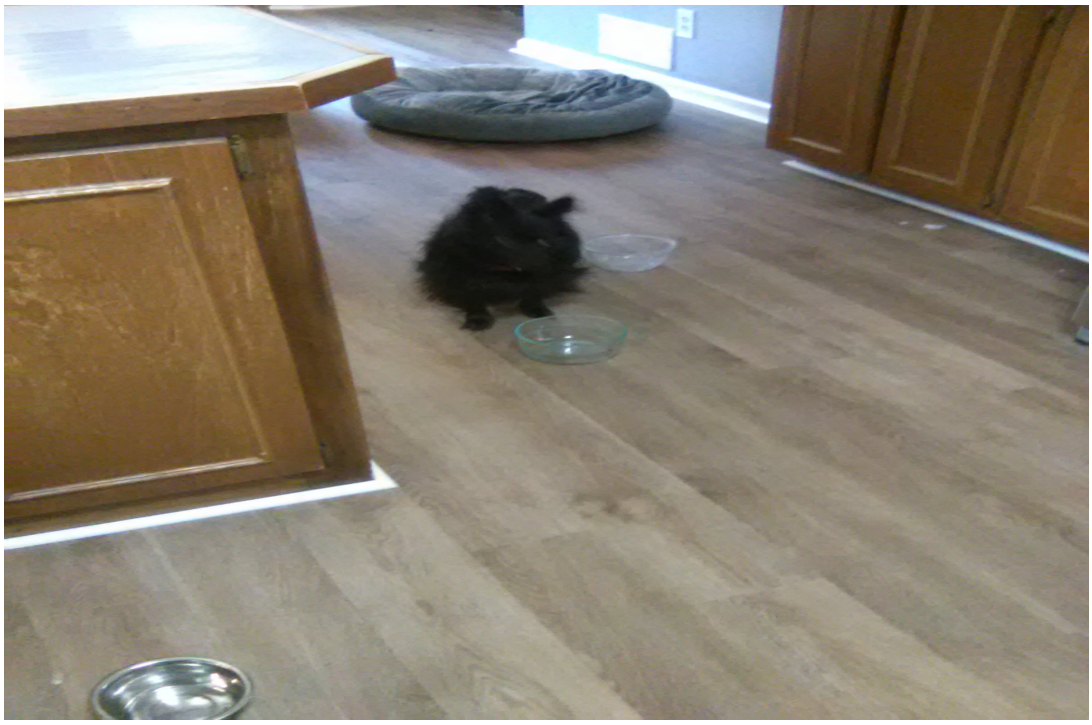
Other options include full error checking (by default just prior week is markedup up), noun maintenance, error dump, and report generation. The report option produces the 3 default reports and prints the number of items for each date as shown below,

Code: Other options for mudaily.txt

```
marchywka@happy:/home/documents/latex/proj/muqed$ mudaily.txt
Usage /home/scripts/mudaily.txt
 -help
 -daily
 -full
 -reports
 -nouns
 -errors

mudaily.txt -reports 2020-09-01 2020-09-05
start 2020-09-01 end 2020-09-05
/home/scripts/links/mjm_diet_diary_form.out
output to /tmp/run_log.txt
     9 2020-09-01
     9 2020-09-02
     9 2020-09-03
     9 2020-09-04
     9 2020-09-05
output in /home/scripts/script_data/cases/out
```

6. WORKFLOW



6.1. Workflow Legacy

1. Run the script "daily.txt". This puts a new template on the clipboard and opens vi to the cases.tex file containing note as well as the diary data file. User pastes template and modifies after feeding. A meal menu approach could be implemented with meals designed apriori but after the fact "touch up" required anyway so all entry of recipes and amounts was done after feeding.
2. Feed the dogs and try to remember the recipe and who ate what. This is easier than it sounds once the template is in place but still subject to errors.
3. Fix the template to match the events. Close the file in vi.
4. The daily.txt script then runs LaTeX on the file, embedded in a larger LaTeX document, and copies it to a backup file system.
5. From time to time, use "run_snacks" to make intermediate files and validate the entries. Fix typos, add new nouns or words to the code and recompile.
6. Run linc_graph or R or other codes to produce tables and graphs.**TODO : maybe document some R scripts**

6.2. Workflow Local

For purely local use, the user has many options. The overall objective is to add entries to the data file that record activity as it occurs. This may involve simply using "vi" to add entries along with updates to the nouns files. With pre-planned meal recipes and no regard for wasted of uneated foods, the entries can be made a priori although maybe commented out until they are actually realized. The design goal workflow is more or less as follows,

1. User generates a "template file" or a copy of the last day's data file entries but all commented out with parentheses.
2. User edits the template, vi works well, to reflect activity as it occurs.

3. User asks MUQED to validate the edited template file.
4. User examines markups or issues with the template file and iterates.
5. User appends the template file to the data file and maybe updates the noun file as new ingredients are used.
6. User runs reports as desired.

However, one alternative is a hybrid of new and old methods that cleans up the data file by parsing and write back out in a more uniform format,

1. User invokes a script similar to legacy system, "mudaily.txt -daily" . This invokes "vi" on a partially markedup copy of the diet data file with a template for the current day on the clipboard. This is the actual data file with errors and comments on lines beginning with "##"
2. User pastes template at end of file if needed and modifies.
3. User searches for "$\hat{}$##" to locate all error messages and resolve if desired.
4. User exits editor, the modified file is written over the original data file.
5. User repeats as desired to fix errors or enter more data.
6. User updates noun files as needed, "mudaily.txt -nouns" .
7. User searches for older unresolved errors, "mudaily.txt -full" .
8. User generates snippets of unresolved errors, "mudaily.txt -errors" .
9. User generates reports as outlined elsewhere or uses mudaily.txt for example, "mudaily.txt -reports 2020-09-01 2020-09-10" .

6.3. Workflow Mikemail

MUQED can also be run with a remote installation to allow various people to make diary entries and share data or other parts of MUQED . In this setting, Mikemail functions to provide interaction with users for which a normal web interface could be of less utility. Reasons for this include lack of a static IP address for the "server", long running calculations between request and response, need for a record of interaction stored with other correspondences, and preference for simple text or custom UI on top of email in place of idiosyncratic web pages. Mikemail provides fully text based interaction using largely a CSV or SSV format that could also be fed into a UI or adapted to an alternative source code format similar to LaTeX . XML would be an option but it would make the need for a GUI almost a reqruirement.

1. User sends email to mjm_mailproc@yahoo.comwithout any meaninful content. Mikemail generates a response message containing an SSV form requesting the user ton fill in blanks to register.
2. User receives blank SSV form, edits it , and mails back.
3. User receives confirmation of registration.
4. User requests template of some specific diary **TODO : implement a means to pick one doh**
5. User edits received template, mails back requesting markup (AKA validation AKA proofreading) .
6. User recieves a coorected and marked up version explaining what the server thought. User edits and repeats and eventually commits as a new entry to the data file .
7. User requests various reports etc.

7. THE DIET DIARY FORM

The Diet Diary Form is the basic mechanisms for handling MUQED data. Implemented as a single c++ headler file with support from an mjm header library, it can be compiled alone with a limited test UI or intergrated into a larger app such as Mikemail . The diet diary form is parsed into words within lines as a "ragged" grid of words similar to most SSV or CSV files. It is interpretted using other files that give meanings to the entries.

7.1. Syntax : Actual Data File

The data file or form is designed for terse user entry of as much data as possible without apriori restrictions from a menu or validation system. Arbitrary text is allowed. Although this does create a typo and validity hazard these can usually be fixed in a text editor after a quick form validation. A more sophisticated version may be able to provide some autocomplete functionality if the support files are set up properly. Alternatively, the supporting noun files may contain enough information to populate a menu tree for a GUI.

A short snippet of the included data file, "demo/data.txt", is as follows,

First note that any text between parens is a comment and quickly omited during parsing. As the day progresses, the commented entries can be deleted or included with modifications. The above indicates Greta had a variety of items around six AM while Beauty had 3/8 of the AMSNACK around the time it was made a little after 5 in the morning.

The words within each line are organized as follows, ,

Code:

```
2020-09-22 & AMSNACK 0505AM b7ngnc cbbroth carrot garlic 11PC .5 eggo3 2.0 bulk lecithin 900mg bulk taurine
     70mg bs B-3 500mg nutricost pantothenate carlson glycinate 2.5mg Cu 400mg tryptophan 650mg leucine
    1300mg lysinehcl 650mg threonine \
2020-09-22 & AMSNACKB 0505AM b7ngnc cbbroth carrot garlic .5 11PC .25 eggo3 1.0 bulk lecithin 450mg bulk
    taurine 35mg bs B-3 250mg nutricost pantothenate carlson glycinate 2.5mg Cu 200mg tryptophan 632mg
    leucine 650mg lysinehcl 320mg threonine \
2020-09-22 & NOTE Greta some human readable obervation here \
2020-09-22 & Greta 0600AM SnAg 200mg arginine 2mg glycinate Cu ( .25 AMSNACK 0930AMSNACK 1.0 PMDINNER .25
    PMSNACK ) \
2020-09-22 & Beauty .375 AMSNACK .5 eggo3 b20ngnc carrot 0630AM .25 AMSNACK ( 0930AMSNACK 1.0 PMDINNER .25
    PMSNACK ) \
```

1. **Date :** The first word is a date which should be in the format YYYY-MM-DD although the final implementation may use the Linux "date" command to see if it can be coerced to this form.

2. **Optional Ampersand :** Legacy code stored data in partial LaTeX format and this is ignored although attempt to preserve if present.

3. **Line Type {***consumer*|*recipe*|**NOTE|DMEL|COMMENT|TODO : INV*,... } :** The line type indicates the significance of the rest of the line. Consumers and recipes are distinguished by character class as consumers should be proper names capitalized but recipe or macro are digits and upper case. **TODO : verify this and make sure nouns not checked here. Maybe make this more rational so consumer names can have serial numbers or suffix etc.** In this code, Beauty and Greta are two consumers and their diets include a recipe or macro called "AMSNACK" . The nouns or consumed items can be entered in local shorthand and converted to more common terms or components later as described in the NOUN file.

   The free-form text line types are mostly to record subjective general or exceptional information about some of the day's entries,

   (a) **COMMENT :** is used to record observations of miscellaneous information about the day's events.

   (b) **DMEL :** is used to note exceptions or problems encountered that violate norms.

   (c) **TODO : INV* : an inventory control record that may include inventory count, orders, costs, arrivals, etc TBD**

   (d) **NOTE :** lines have typically been in the form of consumer name followed by free form text for later semi-automated collation.

   For the consumer and recipe lines, the remainder of the line describes consumption by the consumer on the given date, **TODO : The "recipes" completely lack instruction and are just ingredient lists yet MAllaid and Amadori have not left the building lol**

   (a) **TIME {DIGIT1-4{A|P}M |{DIGIT1-2:DIGIT3-4 } } :** Two time formats are supported, a conventional 24 hour specification with 2 hour digits, a colon, and 2 minute digits. A redundant version with 4 digits of 24 hour time followed by AM or PM is also supported.

Items on the consumer and recipe lines describe attributes of consumption such as times and quantities involved for different nouns. The time attached to a recipe may reflect its time of creation but should be the earliest time it can be served. As the line is parsed from left to right, the time cursor only increments and later times cause earlier ones to be ignored. This allows for a time to be given for a recipe like "AMSNACK" without being repeated for each consumer. If leftovers are served later, the time cursor for each consumer can be later than the recipe time and will be considered the time of event.**TODO : verify that nested expnsions work so that a new entry could be made with just a later time. For example, define BSNACK as "BSNACK 2300PM ASNACK" TODO : perhaps add a preparation time adjective for some nouns**

(b) **ADJ1-n :** A sequence of words that are not listed as nouns in the noun file **TODO : or database** . Adjectives are asembled until a NOUN or recipe is found and then the accumulated adjectives are evalated along with the noun at the latest time specification encountered.

(c) **of :** A reserved word that separates one group of quantifying adjectives from another. Those to the left describe the amount consumed, those to the right the amount offered or provided.

(d) **NOUN :** A word in the noun or recipes list to which the

(e) **KEYWORD :** A reserved word such as **of :** that modifies the interpretation of the event. See appendix for details.

4. **Optional Double Backslash :** Legacy code stored data in partial LaTeX format and this is ignored although attempt to preserve if present.

7.2. Supporting File : Units

All units except for a few hard coded one (pct, rank) are defined in this file. It provides various attributes for standard and local units allowing flexibility in going from data entry to reporting. Metric, English, and miscellaneous kitchen and invented units are included for illustration.

7.3. Supporting File : Nouns

Nouns may be artibtrary text strings including those of the same syntax as units and quantifiers. If they are included in the noun list, they can not ne used in other ways. Now, the nouns are mostly food components but can be outcomes such as weight or anything else that can be measured at some time point.

The nouns right now are read from a text file and kept in memory.**TODO : but they are well somewhat isolated and coulb be read from a DB with cachable hits and misses** .

7.4. Supporting File : Ignores

Initial entry may contain more words of value that are not yet entered into appropriate data files. The code may ignore those listed here until other things are fixed. **TODO : This is confusing. These are NOT nouns to ignore as mnemonics such as " notyrosine " but rather typically common adjectives that have yet to be allowed for a large group of nouns.**

7.5. Supporting File : Reserveds

Not sure this has a lot of meaning yet as there is nothing to do with them although the legacy code recognizes consumer specific "NOTE" and more generic COMMENT and obervations about data quality or exceptions (DMEL, data-model-error-log).

8. OUTPUT FILES

MUQED code converts the files into more text files that should work with many data processing packages or home brew scripts. The diet form entries may be expanded somewhat for report generation allowing totals of various components to be obtained. This may include specific vitamins or more gross analyses.

Three different report types are supported.

1. Data Glob, line for each date consumer time iten
2. Daily, one line for each date consumer summed for the day.
3. Inventory , a total of all items consumed.

The Glob report is just a list of each item consumed for each consumer-time along with redundant values for debugging. Times are given in 24 hour time with hour fractions rather than minutes for ease of analysis.

Code: Snippet from the Glob report

```
head out/dog_glob.txt
18510 2020-09-05 5 Greta Cu 5 mg - 5 mg
18510 2020-09-05 5 Greta arginine 400 mg - 400 mg
18510 2020-09-05 5 Greta ctbroth 1 - - 1 -
18510 2020-09-05 6.5 Greta ctbroth 1 - - 1 -
18510 2020-09-05 6.5 Greta Cu 2 mg - 2 mg
18510 2020-09-05 6.5 Greta SnAg 1 - - 1 -
18510 2020-09-05 6.5 Greta arginine 200 mg - 200 mg
18510 2020-09-05 8 Greta ctbroth 1 - - 1 -
18510 2020-09-05 8 Greta Cu 2 mg - 2 mg
18510 2020-09-05 8 Greta SnAg 1 - - 1 -
```

The Daily report is probably the most common output to use. Total daily consumption of each item is reported for each consumer. Several options exist to make the output more user or machine readable. Each line can include just the items consumed by that consumer or a list of all items consumed that is contant through the report.

Code: Snippet from the Daily report

```
 head out/dog_daily.txt
18510 2020-09-05 Greta Cu 17 mg SnAg 6 - arginine 2400 mg b7ngnc 0.2 - canned 2 - carrot 0.2 - ctbroth 11.2
     - eggo3 0.05 - prednisone 5 mg salmon 0.2 - shrimp 17.964 grams
18510 2020-09-05 Beauty 01PC 0.75 - B-3 96.25 mg K1 6 mg KibbleAmJrLmUHP 0.9 - KibbleLogic 0.6 - arginine
    600 mg b20ngnc 1.5 - b7ngnc 2 - biotin 6.25 mg carrot 1.25 - citrate 0.3125 tsp ctbroth 2 - eggo3
    0.8125 - garlic 1.25 - kcl 0.3125 tsp lecithin 4800 lecu leucine 406.25 mg lysinehcl 2600 mg
    pantothenate 687.5 mg salmon 2.2625 - shrimp 20.2095 grams taurine 1800 mg threonine 1300 mg tryptophan
     800 mg tyrosine 850 mg
18510 2020-09-05 Happy 01PC 0.125 - B-3 21.875 mg K1 1 mg KibbleAmJrLmUHP 0.075 - KibbleLogic 0.05 -
    arginine 100 mg b20ngnc 0.125 - b7ngnc 0.5 - biotin 1.25 mg carrot 0.375 - citrate 0.078125 tsp ctbroth
     0.5 - eggo3 0.078125 - garlic 0.375 - kcl 0.078125 tsp lecithin 1050 lecu leucine 121.25 mg lysinehcl
    568.75 mg pantothenate 156.25 mg salmon 0.4 - shrimp 2.2455 grams taurine 393.75 mg threonine 325 mg
    tryptophan 175 mg tyrosine 150 mg
```

The Daily report can be modified by passing flags with bits defined as below. Originally the file was designed so that each line is self contained and provides date and consumer information first followed by nouns and amount consumed without units. Units are now optional and flags are provided to output all fields or just those that had entries. If all fields are uniform across lines, a header makes sense and a "normal" ssv file can be generated.

Code: Bit fields for the report formats

```
push_all=Bit(flags,0); // include all fields used anywhere
occurance_order=Bit(flags,1); // order fields as encountered instead of alpha
auto_range= Bit(flags,2); // adjust inventory totals to best fit suffix
include_units= auto_range || Bit(flags,3);
push_zero=!Bit(flags,4); // include entries with value of zero
labels_hdr=Bit(flags,5); // create a header line with field labels
labels_each=!labels_hdr; // include field names on each line
fill_units=true; // zero entries may not have units, borrow sfx
```

The inventory or used report is just a total of all nouns eaten although offered may be a better guide for inventory management.

Code: Snippet from the Inventory report

```
head out/dog_used.txt
01PC 12.1875 -
B-1 168.75 mg
B-100 4.1375 -
B-3 1.6275 gram
Cu 80.9375 mg
D-3 20 iu
Iodine 3.375 mg
K1 116.4 mg
K2 76.875 mg
KibbleAmJrLmUHP 22.581 -
```

FIG. 3: IMG20210101_042018.jpg

FIG. 4: IMG20210101_042026.jpg

FIG. 5: IMG20210101_044532.jpg

FIG. 6: IMG20210101_044559.jpg

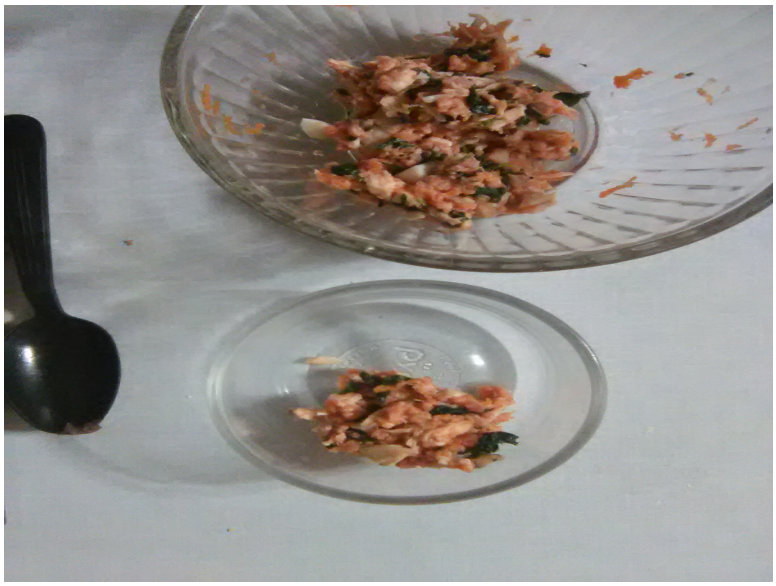


FIG. 7: IMG20210101_045347.jpg

FIG. 8: IMG20210101_045354.jpg

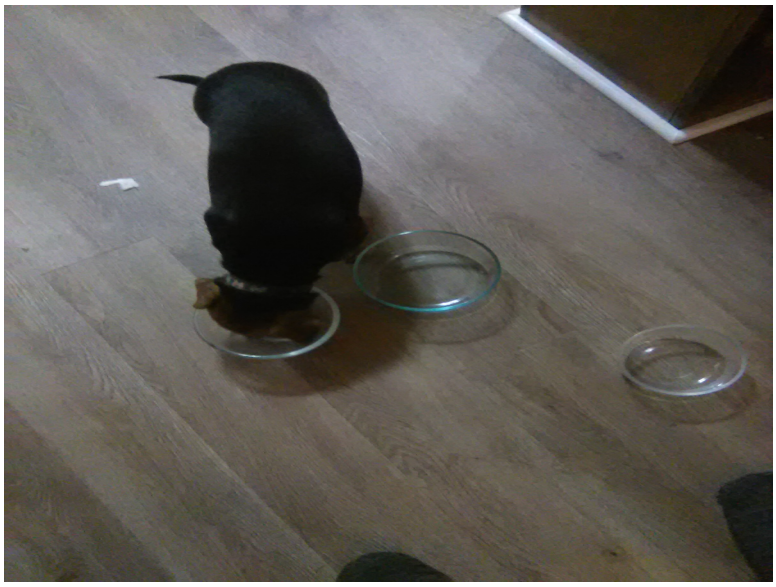


FIG. 9: IMG20210101_045514.jpg

9. CONCLUSIONS

While MUQED is quite primitive, there are many seemingly primitive utilities on Linux for processing text. This is specialized to keeping a log or diary of events and consumption for correlation with outcomes and ad hoc observations. The open format avoids problems with menu systems or menu plus complaint box. With batch validation, exchange formats like email are allowed and creation of custom GUI should not be difficult for various applications. The legacy code performed well and allowed for many insightful obsevations and hopefully more can be made if MUQED is completely implemented.

10. SUPPLEMENTAL INFORMATION

10.1. Computer Code

11. BIBLIOGRAPHY

[1] TA Ban. The role of serendipity in drug discovery. *Dialogues in Clinical Neuroscience*, 8(3):335–44, 2006. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3181823/.

[2] Y Ciere, D Jaarsma, A Visser, R Sanderman, E Snippe, and J Fleer. Studying learning in the healthcare setting: the potential of quantitative diary methods. *Perspectives on Medical Education*, 4(4):203–7, 2015. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4530538/.

[3] HA Ghofrani, IH Osterloh, and F Grimminger. Sildenafil: from angina to erectile dysfunction to pulmonary hypertension and beyond. *Nature Reviews. Drug Discovery*, 5(8):689–702, 2006. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC7097805/.

[4] Irwin Goldstein, Arthur L. Burnett, Raymond C. Rosen, Peter W. Park, and Vera J. Stecher. The serendipitous story of sildenafil: An unexpected oral therapy for erectile dysfunction. *Sexual Medicine Reviews*, 7(1):115–128, jan 2019. URL: https://doi.org/10.1016/j.sxmr.2018.06.005, doi:10.1016/j.sxmr.2018.06.005.

[5] Peter Kruyen. Dearscholar: A mobile application to conduct qualitative and quantitative diary research. *J. Open Source Softw.*, 5:2506, 2020. URL: https://www.semanticscholar.org/paper/DearScholar%3A-A-mobile-application-to-conduct-and-Kruyen/59113a5b4b838730defd4ebc6caa317140390b27.

[6] WD Mansilla, CPF Marinangeli, KJ Ekenstedt, JA Larsen, G Aldrich, DA Columbus, L Weber, SK Abood, and AK Shoveller. Special topic: The association between pulse ingredients and canine dilated cardiomyopathy: addressing the knowledge gaps before establishing causation1. *Journal of Animal Science*, 97(3):983–97, 2019. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC6396252/.

[7] M.J. Marchywka. Adding citation features to shares for e-commerce, news, and other non-academic sites. Technical Report MJM-2019-003, not institutionalized , independent, 306 Charles Cox , Canton GA 30115, October 2019. version 0.00 likely to change significantly.

[8] M.J. Marchywka. Canine heartworm treated with doxycycline, ivermectin and various supplements. Technical Report MJM-2019-001, not institutionalized , independent, 306 Charles Cox , Canton GA 30115, February 2019. May be recycled in appropriate media.

[9] M.J. Marchywka. Less trendy properties of common molecules. Technical Report MJM-2019-004, not institutionalized , independent, 306 Charles Cox , Canton GA 30115, October 2019. May be recycled in appropriate media.

[10] M.J. Marchywka. On the age distribution of sars-cov-2 patients. Technical Report MJM-2020-002, not institutionalized , independent, 306 Charles Cox , Canton GA 30115, May 2020. May be recycled in appropriate media.

[11] K.M. Mohamed Shakir, David Turton, Brian S. Aprill, Almond J. Drake, and Radm John F. Eisold. Anemia: A cause of intolerance to thyroxine sodium. *Mayo Clinic Proceedings*, 75(2):189–192, feb 2000. URL: https://doi.org/10.4065/75.2.189, doi:10.4065/75.2.189.

Acknowledgments

1. Pubmed eutils facilities and the basic research it provides.
2. Free software including Linux, R, LaTex etc.
3. Thanks everyone who contributed incidental support.

Appendix A: Legacy Components and Tests

daily.txt updates cases.tex with a template on clipboard to paste just before "DIETEND". This then compiles and makes a backup. export.txt then copies to a test location,

```
 2035 ./export.txt
 2036 daily.txt
 2037 ./export.txt
 2038 daily.txt
 2039 history
marchywka@happy:/home/documents/latex/proj/cases$ cat export.txt
#!/bin/bash
cat cases.tex | grep "^20[12][0-9]-[01][0-9]-[0-8][0-9] " | grep "&" | awk '{ if ($1> "2017-04-21") print
    $0; } ' > /home/documents/cpp/proj/mikemail/diet/newest_data.txt
```

The snacks project then makes data files,

```
2097 vi mjm_snacks.h
 2098 ./run_snacks -opt -compile
 2099 ./run_snacks -run-dog = 2>&1
 2100 cat /home/data/inter/snacks_collated.ssv | grep Greta| sed -e 's/vmap.*//g' > x1


 linc_graph
```

could be used to make plots and tables but there were no units.

```
 2038 ./run_linc_graph -opt -compile
 2039 ./mjm_linc_graph.out -cmd "read-ragged in $SE_DATA_EXCHANGE/snacks_collated.ssv" -cmd "read-ragged p
    pin_greta.txt" -cmd "set-param datemin 2019-10-01" -cmd "set-param datexmax 2021-03-01" -cmd "set-
    param period 1" -cmd "set-param histogram_like 0x000" -cmd "add-ragged p nfoods *" -cmd "add-ragged p
    foods tyrosine vitamina arginine optizn valine tryptophan threonine" -cmd "add-ragged p filter wide 3
    uniform " -cmd "add-ragged p path-filter * wide" -cmd "add-ragged p path-filter oliveoil unity" -cmd "
    snacks-txt-svg-i gretaxxx.svg 66 in p" -quit 2>&1 | grep stubbed
```

However, the units were added with the

```
mjm_diet_diary_form that
```

is part of mikemail,

```
 2085 ./mjm_linc_graph.out -cmd "read-ragged in /home/documents/cpp/proj/mikemail/dog_daily.txt" -cmd "read
    -ragged p pin_greta.txt" -cmd "set-param datemin 2019-10-01" -cmd "set-param datexmax 2021-03-01" -cmd
     "set-param period 1" -cmd "set-param histogram_like 0x000" -cmd "add-ragged p nfoods *" -cmd "add-
    ragged p foods tyrosine vitamina arginine optizn citrate tryptophan threonine" -cmd "add-ragged p
    filter wide 3 uniform " -cmd "add-ragged p path-filter * wide" -cmd "add-ragged p path-filter oliveoil
     unity" -cmd "snacks-txt-svg-i gretaxxx.svg 322 in p" -quit
```

The Mikemail code is still hardcoded in places but is the stuff documented here,

```
2480 g++ -std=gnu++11 -DTEST_MJM_DIET_DIARY_FORM -I. -I../../mjm/hlib -I../../mjm/num -Wall -gdwarf-3 -O0 -
    x c++ mjm_diet_diary_form.h -lpthread -lreadline -lboost_filesystem -lboost_system

 2484 cat load_stuff.txt | ./a.out 2>&1 | tee xxx
 2485 ls dog*
 2486 history
cat load_stuff.txt
#load-form diet/diary.txt
load-form diet/newest_data.txt
#load-form /home/documents/latex/proj/cases/x4
# need to load units to compile nouns
load-units diet/units.txt
load-nouns diet/aka_nouns.txt
load-nouns diet/canon_nouns.txt
#load-units diet/units.txt
load-ignores diet/ignores.txt
```

```
load-reserveds diet/reserveds.txt
dates 2002-01-01 2020-12-31
#eval
# first flag is for report second is for output choices
eval 8 8
#parse
#markup
#blank
#template

The code in the diet form test seciotn for eval is,

if (cmd=="eval")
{
IdxTy flags=atoi(cmdp1.c_str());
IdxTy flagout=atoi(cmdp2.c_str());
StrTy daily="dog_daily.txt";
StrTy glob="dog_glob.txt";
StrTy usedd="dog_used.txt";
const bool dump_dmglob=Bit(flagout,0);
const bool dump_rest=Bit(flagout,1);
const bool save_dmglob=Bit(flagout,2);
const bool save_daily=Bit(flagout,3);
const bool save_used=Bit(flagout,4);
IdxTy idx=3;
if (save_dmglob)
{ if (cip.wif(idx)!="") if (cip.wif(idx)!="-") glob=cip.wif(idx); ++idx; }
if (save_daily)
{ if (cip.wif(idx)!="") if (cip.wif(idx)!="-") daily=cip.wif(idx); ++idx; }
if (save_used)
{ if (cip.wif(idx)!="") if (cip.wif(idx)!="-") usedd=cip.wif(idx); ++idx; }
```

Appendix B: MUQED Commands

Boiler plate commands not in the main command map may not be visible from help, **TODO : these look like mostly integrated now, ignore this**

1. **err err**
2. **print print**
3. **status status**
4. **template template**
5. **test test**

1. **? ?**
2. **about about**
   Print a short list of credits and information about the app to stdout.
3. **add-ragged add-ragged** NAME WORD0 WORD1 ...
   Add the line WORD0,WORD1... to the ragged entry named NAME .
4. **banner banner**
   Kind of obsolete, print some configuration info and stats.
5. **blank blank**
   Does nothing in muqed_util, see the form code.
6. **clear clear** Currently does nothing.

7. **cm cm**

   Dump the counter timer times, does nothing as nothing is timed right now.

8. **commit commit** FNAME FLAGS

   Append the file FNAME to the diary and save it.

9. **dates dates** STARTDATE ENDATE

   Set the start and endates for further analysis. Both should be LEXIDATE format (YYYY-MM-DD).

10. **dump dump**

    Calls the diary dump method and dumps to stderr.

11. **dump-ragged dump-ragged** NAME FLAGS WFLAGS

    Write the ragged named by NAME to stdout in a way modified by FLAGS and WFLAGS.

tabsep	flags&2	output with a tab seprate
ignore_hash	flags&4	remove comments from output
debug	flags&8	generate a lot of information messages
csv	flags&16	separate with comma even if tabsep is set
use_space	wflags&1	instead of "—" output with space sep unless use_s set
add_seq	wflags&2	do not add line numbers on output
add_quote	wflags&4	put quotes around each field
add_escapes	wflags&8	escape required characters
use_s=	wflags&16	if set with use_space, use the instance default sep.
debug_parse	wflags&32	output info message

12. **eval eval**

$$flags$$

$$flagout$$

$$FILE1FILE2FILE3$$

    Evaluate the diary file over a specified time period and generate upto 3 reports which can be saved to files or dumped to stderr. All parameters are optional but position dependent and blanks are allowed. The file names are used for selected reports in the order glob,daily,used. It is probably better to use the member values rather than change values with parameters here. **TODO : start using and explain m_flp**

```
const StrTy cmd=cip.cmd();
IdxTy flags=atoi(cip.p1.c_str());
IdxTy flagout=atoi(cip.p2.c_str());
const StrTy startdate=m_startdate; // cip.wif(3);
const StrTy enddate=m_enddate; // cip.wif(4);
StrTy daily= m_flp.dog_daily(); // "dog_daily.txt";
StrTy glob=m_flp.glob(); // "dog_glob.txt";
//StrTy usedd="dog_used.txt";
StrTy usedd=m_flp.dog_used(); // "dog_used.txt";
const bool dump_dmglob=Bit(flagout,0);
const bool dump_rest=Bit(flagout,1);
const bool save_dmglob=Bit(flagout,2);
const bool save_daily=Bit(flagout,3);
const bool save_used=Bit(flagout,4);
```

13. **eval-week eval-week** See **eval**. Generate reports for the last week. May be empty if no entries for the prior week.

14. **expand** **expand** type state

    Set the expansion type to "type" and numeric state to "state." By default, all macro definitions will be expanded and nouns will expand according to their defintiions. Those can be overridden to some extent here. **TODO : maybe make a more detailed override system so users don't need to edit the noun files.** A macro only has one "Expansion" elaborated by the line that defines it. Nouns may have multiple analyses leading to an ordered list of ingredients of unknown quantity or perhaps a nutritional analysis or a completely known breakdown into specific molecules. All of these may be included with a default set for each noun. This evaluation time **expand** command allows for uniformaly expanding all nouns that have a qualifying expansion.

Bit	Variable	Impact
0	try_an_expand	expand all nouns that have an expansion
1	have_pref_expand	
2	skip_all	skip all expansions and just report each entry by canonical name

```
{ m_expand_type=e; m_expand_state=s;
MM_ERR(MMPR4(e,s,m_expand_type,m_expand_state))
const bool skip_all_expand=Bit(m_expand_state,2);
bool try_an_expand=Bit(m_expand_state,0)||cn.expand();// (choice!="");
const bool have_pref_expand=Bit(m_expand_state,1)||(m_expand_type.length()!=0);
<<MMPR4(cn.expand(),cn.expand_lines(),skip_all_expand,m_expand_state)
m_expand_state=0;
IdxTy m_expand_state;
```

```
{ m_expand_type=e; m_expand_state=s;
MM_ERR(MMPR4(e,s,m_expand_type,m_expand_state))
const bool have_pref_expand=Bit(m_expand_state,1)||(m_expand_type.length()!=0);
const bool have_exp_type=cn.have_parts(m_expand_type);
const bool have_exp_line_type=cn.have_line_parts(m_expand_type);
if (have_exp_type || have_exp_line_type) choice=m_expand_type;
StrTy m_expand_type;
```

15. **get-param** **get-param** key

    Print the value for "key" to stdout. MUQED contains a general parameter table m_flp

16. **help** **help**

    Basically just list commands

17. **list** **list**

    List summaries of values of most items in muqed_util.

18. **load** **load** FILE This does nothing now, use one of the other load commands

19. **load-finder** **load-finder** FILE

    Load the configuration for the diet diary's resource finder. This allows it to validate file names where a noun is supposed to point to an existing file.

20. **load-form** **load-form** FILE Load, or actually append, FILE to the internal diet diary file.

21. **load-ignores** **load-ignores** FILE

    Load a list of words the diary parser can ignore. This is largely a temporary list while migrating nouns.

22. **load-nouns** **load-nouns**

    Load or append a list of noun specifications.

23. **load-recipes** **load-recipes**

    Load or append a list of recipes specifications.

24. **load-reserveds** **load-reserveds**

    Load or append a list of reserved word specifications.

25. **load-units load-units**

    Load or append a list of unit specifications.

26. **markup markup** FILE FLAGS

    Markup the diary document and save in FILE or dump to stderr if FILE is blank. FLAGS not currently used.

27. **markup-week markup-week**

    As above except for only the last week.

28. **missing missing** FLAGS

    Dump a list of nouns needed for expansions that are missing. DUMP entries for them to stderr based on the setting of FLAGS

29. **nouns nouns** flags

    Indexes or compiles the currently loaded nouns and optionally dumps a list of nouns to stderr. Largely for debugging the noun file.

30. **parse parse**

    Does nothing, a copied command from the standalone diary form test code.

31. **parse-week parse-week** Does nothing, a copied command from the standalone diary form test code.

32. **quit quit**

    MUQED continues to read commands until the input stream closes or until this command is received.

33. **read-ragged read-ragged** NAME FILE FLAGS

    Read lines of ragged NAME from FILE with options chose by FLAGS. Only space, comma, or tab delimiters are allowed. Qouting and handling of special input quirks can be selected

```
const bool load_string=(cmd=="string-ragged");
const bool load_parsed=((flags&1)==0);
const bool tabsep=((flags&2)!=0);
const bool ignore_hash=((flags&4)!=0);
const bool debug=((flags&8)!=0);
const bool csv=((flags&16)!=0);
const bool pdftext=((flags&32)!=0);
const bool ignore_2delims=((flags&64)!=0);
```

34. **set-param set-param** KEY VALUE

    Set the value for key KEY to VALUE in m_flp;

35. **source source** FILE

    Read and execute commands from FILE and return back to this stream for further input.

36. **string-ragged string-ragged**

    See **read-ragged**

37. **template template** FILE FLAGS

    Parses the last week (from current date) of entries and generates a blank form from last entry. Note that no dates may be parsed if data has not been entered in 7 days but a blank form will stillbe returned if any day has had data.

38. **transpose-ragged transpose-ragged** dname sname flags Transpose ragged sname and store in dname. Limited use here, included for fixing foreign files.

39. **write-ragged write-ragged** file ragged flags

    Writer ragged to file with options chosen by flags.

Appendix C: Unit Descriptors

A units file may look like the one below,

Code: Snippet from a units file.

```
microgram base gram system si up mg type mass convert gram,1e-6 aliases mcg
milligram base gram system si up gram down microgram type mass convert gram,1e-3 aliases mg
kilogram base gram system si down gram type mass convert gram,1e3 aliases kg
teaspoon base milliliter system kitchen type volume convert milliliter,4.929 aliases tsp
"quarter teaspoon" base milliliter sytem kitchen up teaspon type volume convert milliliter,1.23225 aliases
    qtsp
"g/tsp" type density units grams/teaspoon
# need a syntax for spaces doh
"g/qtsp" type density units grams/qtsp
gram base gram system si,mks up kilogram down milligram type mass convert gram,1 aliases g,grams
lecu base gram system arb type mass convert gram,1.2,mg,1200
ounce base gram system english up pound type mass convert gram,28.3495 aliases oz,ounces
pound base gram system english down ounce type mass convert gram,453.592 aliases pounds,lb,lbs
milliliter base liter system si,cgs type volume convert liter,1e-3 aliases ml,cc,mL
"cubic centimeter" base liter system si,cgs type volume convert liter,1e-3 aliases ml,cc,mL
```

Each line begins with a unit name and usually will contain an **aliases** keyword followed by a comma separated list of synonyms which are usually abbreviations. Both a **base** and **type** keyword exist to aid with manipulation. The base is the name of another unit to which this can be converted for ultimate conversion to other units of the same **type** although **TODO : it may make more sense to specific a base unit for a given type and not for each unit definition.** A **system** is provided for a user defined list of memberships. **up** and **down** allow for changing units to normalize the quantity value. **convert** and maybe instead a **equals** list define more conversions for cases wheere direct convertion may be desired.

1. **RATU** An expression of products and quotients of units raised to integer powers. Unlike the DQ, there is no quantity associated with it.
2. **SFX**
3. **TEXT** A quoted character string of relevance to the user
4. **UNIT**

1. **aliases aliases** SFX1,SFX2...

   A list of synonums for this unit that may include alternative names or abbreviations usually used as suffixes. **TODO : spaces will probably work with quotes but probably need only single suffix per alias keyword. Don't use spaces...**

2. **base base** UNIT

   A unit to which this one can be converted that should allow all units of the same type to be compared. Units from different systems may still have the same base.

3. **definition definition** RATU

   A rational expression of units which define the expansion of complex quantities.

4. **down down** UNIT

   The name or alias of a unit to which this suffix can be converted when the value is too low to be "in range."

5. **system system** TEXT

   A user defined catagory that may be useful for reporting at some point. Names can be detailed, CGS or MKS for example or broader SI, English, kitchen,arbitrary etc.

6. **text text** TEXT

   Free form text comments for user

7. **type type** TEXT

   A single word descriptions of the quantity it defines such as mass, volume, density, etc.

8. **up up** UNIT

   The name or alias of a unit to which this suffix can be converted when the value is too high to be "in range."

Appendix D: Noun Descriptors

The noun descriptors define the allowed set of things that can be chronicled along with more details about them. Descriptors include sensible constraints on the modifiers, expansions into component parts, and miscellaneous attributes and grouping for later analysis.

Noun definition criteria are left to the user and a given set of items could be described by one noun with many modifiers or individual nouns with no modifiers. This makes it easier for data entry without making reporting more difficult. Nouns may include special purpose words that may not be normally considered as nouns.

A noun definition consists of a single line in a text file. With many modifiers or ingredient lists this may be quite long. Users may source the file from some scripting language that uses variables and then generate the MUQED file from the related script or generation software.

The first word is always the noun name and the rest of the words are keywords followed by any corresponding parameter values. The name "word" may actually be a set of synonyms separated by commas although the entire list must be quoted if it includes a space. Use of non-alpha is discouraged as it may not be supported later due to ambiguiuty with other word types. The first example is for "MACRONOUN" which is the entry for expansions of macro or recipe consumption. Right now it has no interesting capabilities. The second line for "11PC" demonstrates a problem with the early system of essentially unconstrained noun names- this could also be a quantity with a units suffix but instead is shorthand for a combination of potassium chloride and citric acid. The **part** keyword allows components to be specified and with expansion enable the components rather than the 11PC are reported. Note that slang, local jargon, and trivial names can be used to facilitate entry but then can be removed from reporting. In the case of some poducts like "optizn" a specific kind of zinc forumlation is denoted but most reports may just want "total zinc." For most reports the **canon** term will be used and many special cases can map to one such word.

A noun descriptor line can contain content between parens which is ignored similar to the diary entries. This is useful as a comment that could be uncommented later.

Code: TODO : these are old need to check for obsolete Selected entries from the nouns file.

```
cat demo/canon_nouns.txt | more | awk '{ if (NF>3) print $0;}'

MACRONOUN canon MACRONOUN adj delta adj small
11PC canon 11PC runits serving expand 1 serving .25tsp part citrate,.25tsp part KCl,.25tsp
citrate canon citrate runits tsp
KCl canon kcl runits tsp
B-12 canon B-12 runits mg
B-3 canon B-3 adj bs
D-3 canon D-3 adj kroger runits IU punit IU,1iu serving 1iu
B-6 canon B-6 adj sundown
bor-l-immune canon bor-l-immune adj drops
canned canon canned group generic
spinach canon spinach adj raw adj birdseye adj frozen adj fresh
Cu canon Cu group metal adj glycinate adj carlson
optizn canon optizn group metal
water canon water group drink
pasta canon pasta group humanfood
scraps canon scraps group humanfood group vague
doxycycline canon doxycycline group antibiotic
garlic canon garlic adj minced
LESI canon Iodine serving 1mg startdate 2017-09-06 text "LESI Life Extensions Sea Iodine 1000mcg"
lecithin canon lecithin serving 1200lecu runits lecu
lipoicacid canon lipoicacid adj drop adj drops adj natural adj best adj naturals adj bestnaturals
ketasalmonbrothbs canon salmon adj sockeye,chile,farmed adj pink adj coho adj kroger adj pacific adj wild
salmons canon salmon adj sockeye,chile,farmed adj pink adj coho adj kroger adj pacific adj wild
shrimp canon shrimp runits grams
SnAg canon SnAg startdate 2020-09-06 text "SnAg solder dissolved in arginine or lysinehcl with sweeter
    taste and easier to eat"
taurine canon taurine runits mg
vitamina canon vitamina adj bronson adj iu
weight canon weight group outcome
vomit canon vomit group outcome
treat canon treat group generic
```

1. Word Types

1. **ADJ** Any literal character string except although special characters and spaces may not be well supported. These should typically be modifiers that help describe a NOUN.

2. **BIBTEX** Any literal character string except although special characters and spaces may not be well supported. This is expected to allow the user to locate citation information in whatever database is associated with a project. One form would be "file:entry" for a local set of bib files.

3. **BOMTEX** Any literal character string except although special characters and spaces may not be well supported. This is expected to allow the user to locate "bill of materials" information in whatever database is associated with a project. One form would be "file:entry" for a local set of bib files. This is a convenience for re-ordering or allowing others to source critical components. See also https://tug.org/pipermail/texhax/2019-October/024083.html

4. **DQ** A dimensional quantity in the form of a number followed by units.

5. **FLAGS** Typically an integer that may be expressed in hex format or sometimes as an "OR" of bit definitions such as "a—b—c"

6. **LEXIDATE** A lexical date of the form YYYY-MM-DD or (maybe) one that can be converted by the linux "date" command. Implemenations and caching may vary so performance will be enhanced with conforming input even if alternatives are accepted.

7. **NOUN** Any literal character string although special characters and spaces may not be well supported. These should typically be items of interest , consumed or whatever is being tracked.
8. **SFX** Any literal character string although special characters and spaces may not be well supported. Used to go with a quantity to describe the amount involved in a diary entry- typically things like mass or volume but may be conversions with complicated dimensionality
9. **TEXT** Any literal character string as desired by the user . Free form text .
10. **URL** Should be a valid url to point to a resource for additional information of functions.
11. **WORD** Any literal character string although special characters and spaces may not be well supported. These are generally defined by the user.

2. Keywords and their Parameters

Implementation Dev Note : The noun descriptor keywords are hard coded in a map that converts them to integers for processing.

1. Basics: **canon**,**flags**,**vflags**,**ignore**

   Define a common name for the noun and some attribute flags to define how it is quantified for reporting.

2. Quantity Details : **serving**,**default-qty**,**punit**,**units**,**runits**,**density**,**active** Provide more details on how an instance of this noun is reported. The norm is quantity but data files and text may be supported allowing images or supporting documents to be referenced and inclusion of free text not suitable for a **NOTE** endtry.

3. Disambiguation : **modifier**,**adj** ,**adjective**,**not**,**mfg**,**UPC**

   Multiple nouns can have the same name but details may vary between them. Adjectives can specify one versus another or exclude one. These may be specific or quirky idiosyncratic words. UPC should be unambiguous but may need additional details. Multiple code types may be supported.

4. Limitations : **startdate**,**enddate**

   Hard date limits may exist for a given noun indeed maybe even a given UPC if forumlation is changed.

5. Organization : **group**, **conflict**

   Nouns may be grouped on arbitrary criteria. Conflicts are similarly user defined. While there is no MUQED hierarchy, users can create naming conventions that contain such.

6. Documentation : **text**,**url**,**bibtex**,**bomtex**

   As writing and sharing results are integral objectives, the nouns include some information for those wishing to repicate results or have enough details to evaluate. These fields provide human readable unstructured text with more structured pointers to recognized standards for further information. User is free to create naming conventions except of course for url.

7. Decomposition : **part**,**rpart**,**lpart**,**ingredients**,**expand**

   Foods and most other items can be decomposed in multiple ways to either facilitate purchase and reproduction, or analysis. In CAD systems these are precise but in this case incomplete information such as ingredient lists may be all that is available. Components may be generic nutrients or idiosyncratic nouns with limited quantity information. These tools provide some ways to deal with these limitations.

8. Miscellaneous : **misc**

   TODO : *not implemented yet maybe a list of key value pairs of something, mix in json for hierarchy but want human readable compatible with the Ragged classes*

1. **active active** TEXT

   Indicates that this is a formulation containing a single intended active ingredient. This has not been thought through and currently does nothing other than keep the string value. This comes up where a product such as a metal salt may be specified in terms of metal content. Normally this is not ambiguous because of the way servings and components are defined. The **canon** feature should take care of the primary ingredient and measures are defined preciesly where known. **Implementation Dev Note : The value is retained in m_active but not used for anything.**

2. **adj adj** ADJ1,ADJ2...

3. **adjective adjective** ADJ1,ADJ2...

   The keyword is followed by a common separated list of adjectives that modify the noun. If spaces are used the entire set or subset containing the spaces needs to be enclosed in quotes.

   Allow user specified adjectives to distinguish this noun from others with the same name. Users may code nouns to include various attributes or use a generic noun name with a long list of adjectives. This may include packaging, manufacturer, quirky features, etc.

4. **bibtex bibtex** BIBTEX

   As a goal of this work is to publish and some NOUN's may be critical, a user field for a local or other BIBTEX entry. This is free form text but suggest something like "file:entry".

5. **bomtex bomtex** BIBTEX

   As a goal of this work is to replicate, some NOUN's may be purchased or sourced carefully. This field allows for a "bill of mateials" entry [7]. Actually free form for now.

6. **canon canon** NOUN

   The initial noun name(s) are terms the user enters to denote whatever this entry describes. They all map for reporting purposes to the NOUN given here. Otherwise, they are all reported by name unless and expansion is specified in which case the components are reported if possible. The legacy version recognized multiple NOUN's could map to a single one for reporting purposed, this field existed before.

7. **conflict conflict** NOUN1,NOUN2...

   Several NOUN's have conflicts with others for food recipes. Some may compete or interfere with each other, some drug interactions may be severe. Some may conflict for other reasons including taste or other compatibility issues. **TODO : there is no mechanism to indicate why or importance etc** Any other NOUN's with such relationships could be listed here. **Implementation Dev Note : this does nothing right now**

8. **default-qty default-qty** DQ See **serving Implementation Dev Note : set a dimensioned quantity , m_default, to a value DQ**

9. **density density** DQ

   The user typically will enter an easy direct quantity such as teaspoons but reports likely make more sense in weight or even moles. Density depends on formulation and various quirks and can vary a lot. The density for this noun which may be overridden by a similar entry in the diary file. If the user enters in a volume type and reporting requests mass this allows for the conversion. Units should have same base units as reporting and entry type or will not convert with current code. See "Units " documentation.

   Implementation Dev Note : set a dimensioned quantity , m_density, to a value DQ

10. **enddate enddate** LEXIDATE

    The last date on which this entry is valide to allow for changes in suppliers or specific types. **TODO : Note that this may not be checked unless multiple nouns with the same name exist.**

11. **enum enum** WORD1,WORD2.....

    A list of key(odd) value (even) pairs mapping allowed diary entry keys into evaluation values for the noun. Values can be numeric or string . Normally, this would be entered with an equals sign between key and value... **TODO : want to make a compatible error bounds system with adjectives and numbers at some point**

12. **expand expand** FLAGS

    An integer composed of flag bits that determine how this NOUN should be expanded for reporting. If bit zero if reset, no expansion is performed. If bit 3 and 0 are set, "rank" units are interpretted according to bits 1 and 2.

```
grep m_expand ../mikemail/mjm_compiled_noun.h
bool expand() const { return Bit(m_expand,0); } // 0
bool expand_lines() const { return Bit(m_expand,3); } // 3
IdxTy expand_rank() const { return (m_expand>>1)&3; } //1,2
```

13. **flags flags** FLAGS

```
m["ignore"]=1<<IGNORE; // mnemomnic or placeholder must have zero qty.
m["syn"]=1<<SYNONYM; // accomodates a synonum from foreign ingredient list
m["synonym"]=1<<SYNONYM;
m["dummy"]=1<<DUMMY; // machine generated to replace a missing one.
m["units"]=1<<NEED_UNITS; // there is no concept of a serving must have units
m["dmel"]=1<<DMEL; // something was wrong with this before earlier entries dmel
m["corrected"]=1<<DMEL; // something was wrong with this before earlier entries dmel
m["default"]=1<<WIN_TIE; // pick this silently in case of a tie
m["alpha"]=1<<EXP_EVAL; // evaluation is exceptional ( not a number )
m["text"]=1<<EXP_EVAL; // evaluation is exceptional ( not a number )
// text here is part of flag string, other text is a description
// of the noun in human format
```

14. **group group** WORD1,WORD2,...

    A flat list of groups to which this NOUN belongs. This is completely up to the user. **TODO : Future code may support these as reporting words and use a hierarchial naming convention although a JSON like or other syntax may make sense too .**

    Thinking outloud
    Note these could be used to populate menus for a GUI app to make a template or entry,

    TODO : output group memberships useful to user for validation and utility for menu or reporting design

15. **ignore ignore**

    A redundant word which achieves the same effect as setting the "ignore" flag. When included, the "ignore" flag is set and the noun is just a mnemonic placeholder. A diary entry can include this NOUN with the expectation it will be edited into a related NOUN that was consumed. The legacy code would use a form like "nofoo" to mean ' no foo." The NOUN is eliminated from all output except error checking. Non-zero quantities are reported as errors. By default a lack of quantitifiers is taken as zero and if quantifiers are present they need to evaluate to zero.

16. **ingredients ingredients**

    Ome of several ways of indicating the component parts into which the NOUN can be decomposed. **TODO : Need to develop a syntax for various limitations and nesting in the list that is common on product labels.**

17. **lpart lpart** LINE1,LINE2...

    Decompose the NOUN into a series of other NOUNS described here. **TODO : This sounded like a good idea but the creation of private nouns is probably pointless. Not fully explored yet**

18. **mfg mfg** TEXT

    Free form text to distinguish NOUN's from different sources. May be used according to user overall needs probably treated like other adjectives.

19. **misc misc TODO : Pick a semantics for this thing**

20. **modifier modifier** ADJ1, ADJ2... See **adjective** .

21. **not not** ADJ1, ADJ2, ... When multiple NOUNS have the same name for diary entry eliminate this one from consideration if the entry contains any of these words. See **adjective** .

22. **part part**

23. **part part** WORD,NOUN1,DQ1,NOUN2,DQ2...

    Create a decomposition or explosion called WORD for this noun when all or most of the quantities are reasonably well known. WORD can be empty creating a default explosion. The components are indicated by name followed by a quantity NOUN. If no units are given, units are synthesized as per serving amounts. The component NOUNS may or may not exist in the loaded NOUN descriptors. If a DQ is blank, it will be assigned a rank value based on position in the list. When most values are not known, using **rpart** or **ingredients** may be better.

24. **punit punit** SFX1,DQ1,SFX2,DQ2...

    A collection of private units or allowed suffixes with their dimensional meanings specific to this NOUN, for example measures of potency IU or international units. Trivial, idiosyncratic, or local names may be produced ("scoops" for example).

25. **rpart rpart** WORD,NOUN1,NOUN2...

    Produce a decomposition called WORD with quantities only given as rank with most common ingredient first. Useful when this is the only information available. **ingredients** may provide more cues common on ingredient labels when implemented.

26. **runits runits** DQ

    The amount of NOUN consumed will be converted to the units described by DQ ("reporting units") if possible given the conversion factors. Exhaustive searches will not be made but units will be transformed to "base" units which can be equated. If no units are given in diary entry, the quantity is assumed to be in servings which may then be converted to reporting units.

27. **serving serving** DQ

    If no units are given when an NOUN is entered on a diary line, the quantity will be interpreted as number of servings consumed described by DQ. Units may be volume or mass but may also be potency related or mnemonic of no larger significance. Should be used if decompositions are not given in intensive units.

28. **startdate startdate** LEXIDATE

    The first day on which this NOUN may be used when multiple nouns with the same name exist.

29. **text text** TEXT

    A free form text field for user comments about the NOUN

30. **UPC UPC** WORD1,WORD2...

    A comma separated lists of UPC codes that describe this item. Each word is a code followed by type (UPC,EAN,etc). These should be synonyms in different systems NOT alternatives.

31. **units units** WORD1,WORD2,...

    TODO : this seems to now be obsolete as the words are just added as adjectives

32. **url url** URL

    A link to a related website. Maybe a manufacturer or commentary about the item. Not used for much, suggest BIBTEX or BOMTEX.

33. **vflags vflags** FLAGS

    A set of value flags to describe how the NOUN is associated with some value. The default is a qauntity but free form and validated test may be included. **TODO : Eentually an enumerated type wii be availble but now it is free or a filename that must exist as a unique entry on a given search path. Search path can not be redundant, need to make canonical names**

```
m["free"]=1<<FREE; // free form human readable text
m["file"]=1<<RESOURCE; // the value is a data file that needs to exist
m["string"]=1<<VALUE_IS_VALUE; // eval to string rather than "1" or rest of phrase
```

Appendix E: About Entry Keywords

1. **density density** DQ A density to be usef for this entry only if this use the applicable NOUN had a different density from what was given.
2. **expand expand** expand Set the report expansion mode for this NOUN only.
3. **note note** TEXT Observation that is not allowed some other way.
4. **dmel dmel** TEXT Any comments or observations that were not expected or make the results interpretation exceptional.

Appendix F: About Mikemail

Mikemail was originally designed to download the contents of IMAP servers in a robust way to backup email. It evolved into an automated email handler similar to fetchmail or procmail. In this context, it provides for remote data collection with all the benefits of email over web forms. This may sound primitive compared to dropdown menus, autocomplete, and instant validation of web forms but in many cases validation of text input may take a while anyway. Security may be another concern as email can go over the wire as plain text but a simple frontend and specific MIME type could allow for encryption and signatures and then send as base64 attachments transparently to the user. A lot of data exchanged over the web would fit well into a CSV or SSV file format- tax forms, medical information, many subscriptions and applications- without the graphics creating more complexity. Many forms are of the "form" of a series of lines with names and numbers for each and a blank area for user input. The graphics don't contibute anything and rely on a lot of other resources on a computer. Many existing programs can deal with line oriented data quite well and Mikemail just makes it easier to avoid even this step. Even as computers become more capable and graphics more robust, various problems occur with the complicated cosmetics that do nothing to help the information flow. As currently envisioned, a normal CSV or SSV format may be common but a more sophisticated version using LaTeX syntax may be more useful providing human readable style cues without all the rendering.

Appendix G: Statement of Conflicts

No specific funding was used in this effort and there are no relationships with others that could create a conflict of interest. I would like to develop these ideas further and have obvious bias towards making them appear successful.

Appendix H: About the Authors and Facility

This work was performed at a dog rescue run by Barbara Cade and housed in rural Georgia. The author of this report ,Mike Marchywka, has a background in electrical engineering and has done extensive research using free online literature sources. I hope to find additional people interested in critically examining the results and verify that they can be reproduced effectively to treat other dogs.

Appendix I: Symbols, Abbreviations and Colloquialisms

TERM	definition and meaning
CSV	Comma Separated Value file format
DMEL	Data Model Error Log or a description of violations or exceptions that impact data utility
NOAEL	No Observed Adverse Event Level
SSV	Space Separated Value file format

Appendix J: General caveats and disclaimer

This document was created in the hope it will be interesting to someone including me by providing information about some topic that may include personal experience or a literature review or description of a speculative theory or idea. There is no assurance that the content of this work will be useful for any paricular purpose.

All statements in this document were true to the best of my knowledge at the time they were made and every attempt is made to assure they are not misleading or confusing. However, information provided by others and observations that can be manipulated by unknown causes may be misleading. Any use of this information should be preceded by validation including replication where feasible. Errors may enter into the final work at every step from conception and research to final editing.

Documents labelled "NOTES" or "not public" contain substantial informal or speculative content that may be terse and poorly edited or even sarcastic or profane. Documents labelled as "public" have generally been edited to be more coherent but probably have not been reviewed or proof read.

Generally non-public documents are labelled as such to avoid confusion and embarassment and should be read with that understanding. ‘

Appendix K: Citing this as a tech report or white paper

Note: This is mostly manually entered and not assured to be error free.
This is tech report MJM-2020-004.

Version	Date	Comments
0.00	2020-09-20	Create from empty.tex template
0.01	2020-10-11	version 0.01 MJM-2020-004
0.011	2020-12-25	added pictures version 0.011 MJM-2020-004
0.012	2021-04-05	added release notes, little progress MJM-2020-004
-	April 5, 2021	version 0.012 MJM-2020-004
1.0	20xx-xx-xx	First revision for distribution

Released versions,

Version	Date	URL
0.01	2020-10-11	https://www.linkedin.com/posts/marchywka_any-app-already-exists-for-this-or-automated-activity-67211
0.011	2020-12-25	https://www.linkedin.com/posts/marchywka_adding-artwork-activity-6748265617696059392-b3Ej
0.012	2021-04-05	https://www.academia.edu/attachments/66183407/download_file?s=sidebar
0.012	2021-04-05	https://www.researchgate.net/publication/350636753_MUQED_a_Multi-Use_Quantitative_Event_Diary_For_Dog

@TECHREPORT{mmarchywka-MJM-2020-004-0.012 ,
AUTHOR = {M.J. Marchywka},
TITLE = { MUQED: a Multi-Use Quantitative Event Diary For Dog Diet Analysis},
NUMBER = {MJM-2020-004},
VERSION = {0.012 April 5, 2021 public NOTES },
INSTITUTION = { not institutionalized , independent},
ADDRESS = {306 Charles Cox , Canton GA 30115},
NOTE = {Version 0.012 , may change signficantly if less than 1.00 },
DATE = {April 5, 2021},
DAY = {5},
MONTH = {4},
YEAR = {2021},
AUTHOR1EMAIL = {marchywka@hotmail.com},
AUTHOR1ID = {orcid.org/0000-0001-9237-455X},
PAGES = { 40 },
CONTACT = {marchywka@hotmail.com},
FILENAME = {muqed}
}

Supporting files. Note that some dates,sizes, and md5's will change as this is rebuilt.
This really needs to include the data analysis code but right now it is auto generated picking up things from prior build in many cases

```
3437 Apr 5 08:54 ./comment.cut 85ae6e8122f8ae54aa105ae1b8b62de4
16564 Jan 31 05:33 /home/documents/latex/bib/mjm_tr.bib 6595006469fa444bd3cdf6005b99b84b
7331 Jan 24 2019 /home/documents/latex/pkg/fltpage.sty 73b3a2493ca297ef0d59d6c1b921684b
7434 Oct 21 1999 /home/documents/latex/pkg/lgrind.sty ea74beead1aa2b711ec2669ba60562c3
7162 Nov 13 2015 /home/documents/latex/pkg/mol2chemfig.sty f5a8b1719cee30a4df0739275ac75f8a
1665 Mar 11 07:50 /home/documents/latex/share/content/mikemails.tex 492f120b024532860d6d00c8d725cf51
2857 Oct 4 2020 /home/documents/latex/share/includes/bibtex2.txt 9afee34eb8da693643444ddf4456aa2c
1050 Jun 30 2020 /home/documents/latex/share/includes/disclaimer-informal.tex 82
    adbad09c56f90ace278c22aaa14b08
425 Oct 11 17:20 /home/documents/latex/share/includes/disclaimer-status.tex
    b276f09e06a3a9114f927e4199f379f7
1286 Nov 14 2019 /home/documents/latex/share/includes/mycommands.tex 18011c7f850bd7ab15625df8328e3cf8
```

```
2901 Jun 17 2020 /home/documents/latex/share/includes/myskeletonpackages.tex
    fcfcd2e3c8d69d533932edaaa47f53a1
1489 Apr 5 06:47 /home/documents/latex/share/includes/recent_template.tex 8c5e794aaf78b21c2e21b8e7516353f2
443519 Dec 24 18:00 includes/claire.jpg 2a2525b640a35cc217910f0b899046bb
538873 Dec 22 11:08 includes/cover.jpg 17ef2dfe03bcc2a8e3d703878037601b
387493 Dec 24 20:13 includes/eating.jpg d0c57d67c3253cb96452015439cd32a8
449308 Dec 24 17:55 includes/frankie.jpg e647b082947e7ce52c47164007e53775
431862 Sep 25 2020 includes/happy_filtered_blue.png 38c0c4c6f267dd98dd7d257809e8c9b4
114891 Sep 25 2020 includes/happy_unfiltered.png f5ab3fd56bbb3d536d19bc9799bbcc5d
505271 Jan 1 06:38 includes/IMG_20210101_042018.jpg 6f982b0fb29ff44253243386d877bc56
503020 Jan 1 06:38 includes/IMG_20210101_042026.jpg 7eb6a645b08be1f88be44d5a6674db5a
573598 Jan 1 06:38 includes/IMG_20210101_044532.jpg 6aefb7848e8418c77116c8265ec1c5bc
516045 Jan 1 06:38 includes/IMG_20210101_044559.jpg fc214c1ae8db2b71f02c095e052c099c
509367 Jan 1 06:38 includes/IMG_20210101_045347.jpg d4ff7423cb0d1910a4713fac24869e40
461421 Jan 1 06:38 includes/IMG_20210101_045354.jpg ba6f559a20731b7e62b5aecdc43ee610
441320 Jan 1 06:38 includes/IMG_20210101_045514.jpg 656cf869e85155db2ba4e838675dcf84
587053 Dec 22 17:45 includes/ready.jpg 1be61e35a95bac34a506fc9fdecd61f2
586442 Dec 22 17:42 includes/vita.jpg fa8fee7152a5fa7b461d3810de3f6309
8598 Apr 5 08:54 ./muqed.aux 674463f905cedb2f4cd68565e63f9c15
3784 Apr 5 08:54 ./muqed.bbl e00b5d9a0468e719642d491deab7bced
5688 Apr 5 08:54 ./muqed.bib f020697332e87b75491948b5ba98e775
1164 Apr 5 08:54 muqed.blg 440e403d1d64dc0040e61beaea254a60
2602 Apr 5 08:55 ./muqed.bundle_checksums 96618348194a79b1ba54f6b85040eb1b
8938 Dec 27 15:26 ./muqed_commands.tex af7d8c91f6335b4d2da9305159240d08
3 Apr 5 08:54 ./muqed.last_page 90e2a51705594d033a3abe9d77b2b7ad
72388 Apr 5 08:54 muqed.log 72086b67bdb1cf1f0aa267426a956363
2752 Apr 5 08:54 ./muqed.out abb25a8e9e130701c5ddfb05858c3864
85669 Apr 5 08:47 ./muqed.tex c24fd74c18903b18f322b7c49f9e5df6
3509 Apr 5 08:54 ./muqed.toc 5823dbeff26c20690c0cb208c5eeffe0
2134 Jan 1 10:38 non_pmc_muqed.bib dae95e962492668118af63f72571cf49
26174 Jan 2 17:37 ./noun_syntax.tex 41d85cad9648ff490284644c23f9cbe8
3161 Jan 1 10:55 pmc_muqed.bib b7c7c9fe2206ec0db6b81b6abab119aa
748 Apr 5 08:53 ./releases.tex 593ef4be14f4ba7e711a11014637c825
31050 Jul 21 2011 /usr/share/texlive/texmf-dist/bibtex/bst/urlbst/plainurl.bst
    ffdaefb09013f5fd4b31e485c13933c1


7411482 Apr 5 08:54 muqed.pdf 734651698631c9ec1c8b995cb32fda9c
```