# **Requirements Analysis/Assumptions**

This programming task has been written as a full-fledged perl project. I have named the project as **MailStore.**

Requirements are taken from email/github and they are pasted as is. I have provided my comments/assumptions based on my requirement analysis for each items in the below list.

1. input will be given as a Maildir
   * **[Babu]:** Input can be given as Maildir path using --maildir option in mailstore.pl script or if not given; the path is taken from config file. But the user input gets the precedence.
2. Every message gets its own directory, with a unique name that can be computed from the message itself
   * **[Babu]:** I am using messageId of the message (which is unique) to create the name of the foldername in target path and the Target path(storage path) is picked from the config file
3. text and html parts, except for attachments, get written as UTF-8 to body.txt and body.htm (if there's more than one of either, only save one this way)
   * **[Babu]:** I am checking content-type to identify the media type, and if it is text/plain then I am storing it in body.txt file, if the message content-type is text/html then I am storing it in body.html file. If there are more than one text/plain or text/html, then I am ignoring it as I am storing only the first one.
   * Attachments are separately handled.
4. all other attachments are written to files with unique names computable from the message
   * **[Babu]:** message with multi-part content has attachments, I am saving each attachment to the message folder named uniquely (messageId + PartNumber + count), and messageId is the unique name of the message.
5. a file called manifest is written in some easy-to-load format that maps from the structure of the source message to the files written out
   * **[Babu]:** I am creating a manifest file named "manifest.yaml" (YAML format) which holds the mapping.
   * My **assumption** is that we need two fields as a mapping group to identify message and its content, so for every message extracted, I am storing two fields = (messageId, Stored\_location) in the manifest file. This manifest file is then stored in the target location (destination path).
6. all created files are mode read/write only for the current user
   * **[Babu]:** I am using 0600 permission mode while creating files on the disk.
7. We are particularly interested in seeing the design of your solution. We will evaluate your code not only based on correctness but also based on design and organization.
   * **[Babu]:** I have created a Class Diagram and it is stored under MailStore/CodeTest/Design-Docs. I have created two formats pdf and png, you can see whichever you want.
8. Implement the program in one or more Perl 5 modules
   * **[Babu]:** Fully object-oriented implementation using Moose. There are more than one modules (please refer class diagram)
9. Target whatever version of perl you like, using whatever libraries you like, as long as we can install them from the CPAN
   * **[Babu]:** I am using perl-5.24.0
   * I have used many CPAN libraries. To make it easy for you to install and run the same versions of those modules, I have created Build.PL script in which I have mentioned all the modules and their versions I am using. After downloading the project you can simply run “Perl Build.PL” and "./Build installdeps" from the main folder 'MailStore/CodeTest/MailStore'. It will install all required CPAN modules from the internet.
10. Include at least a simple set of tests using the standard test libraries.
    * **[Babu]:** Yes, I have used Test::More, and all the testscripts are under folder MailStore/CodeTest/MailStore/t/scripts/
    * I have created a script "runall.t" (using App::Prove module) under t/ folder. This script recursively goes to all the subdirectories and identifies the test scripts and then executes it. Finally it gives a summary report of all the failed and passed result. This script is very handy in running all testscripts in a single shot. To run it, Go to MailStore/CodeTest/MailStore/t and run "perl runall.t"
11. Also include an executable program that takes the paths to its input Maildir as arguments and responds to --help and any other relevant options.
    * **[Babu]:** Yes, the script is here MailStore/CodeTest/MailStore/bin/mailstore.pl
    * This script can accept Maildir path or can run without arguments (as it picks it up from the config)
12. Also include a program that, pointed at one of these generated directories, can print the message's from/to/subject/date, the structure of the original message, and the path to the filename in which each part can be found.
    * **[Babu]:** same script can be used: “MailStore/CodeTest/MailStore/bin/mailstore.pl” with different options –dir-path or --message-id
    * For attachments, just their names will be printed out, anyhow path is already displayed as another parameter in the report.

# **Additional tasks (Perl Standard Activities)**

* 1. All modules are fully object-oriented implemented using Moose
  2. Ran perlcritic level 3 (Harsh) for all modules/test scripts/program
  3. Ran perltidy for all modules/scripts/program
  4. For unit testing, I have created runall.t file that recursively runs all test scripts and gives summary result.
  5. There are no hardcoded values, all parameters will be picked up from config files located under /etc
  6. Logging used, which writes the log to /log/app.log file
  7. Implemented singleton using Class::Singleton module for config and logger.
  8. Manifest file is written in YAML format
  9. Implemented Build using Module::Build for the project. This is to facilitate ease of use to download and install the libraries. Please run “Perl Build.PL” and “./Build installdeps”
  10. Executable scripts are kept under /bin folder
  11. Introduced proper comments wherever required in all modules

**NOTE:** Please see **ReadMe.txt** file for more details for the project structure