# Documentation for Auto Email Classification

1. **Overview**

The submitted code is the backend server for the Auto Email Classification. Written in C#, the program utilizes Google’s Gmail API to classify a user’s emails based on the analysis of email content. The classification is simulated by adding a custom label to the emails whose content has the highest term frequency of the entered topic words. Although this program is still at its prototype stage, it proves the concept that statistical language model can be used to help business users classify emails much more effectively than the existing key word match approaches. The current application architecture can be augmented with more powerful statistical language models than term frequency. As an experiment, another function was developed to return a rank list that provides finer granularity of classification.

1. **Implementation**

The implementation of the program contains three major components:

1. Gmail API authentication & authorization. The scope for access of a test user’s Gmail account is defined statically at the beginning of the program. For our testing purposes, full access of the account is requested. API credentials are loaded and passed to the Gmail API when the Main function begins to execute (Line 25 to 39).
2. Email retrieval from the test user’s Gmail account. For testing purposes, the program only retrieves the top 10 emails and parse out each email’s subject and content that will be fed into the statistical language model for classification (Line 48 – 60). Three helper functions, StripHTML (Line 94 – 97), DecodeBase64String (Line 185 – 193) and GetNestedBodyParts (Line 195 – 221), were provided to parse the email body.
3. Email classification. For each email retrieved from the test user’s Gmail account, the program computes the term frequency of each word in the topic (Line 98 to 117), and based on the value of term frequency threshold the email gets classified into “has topic coverage” or “has no topic coverage”. The threshold value was set arbitrarily for testing purposes (Line 184). If the aggregation of term frequencies is over the threshold, the ModifyMessage method is called to add the label with the topic text to the email via Gmail API (Line 185 to 190). The detailed implementation of ModifyMessage can be found in Line 75 to 96. If there’s an error with adding the label to email, the error details will be printed in the console window.
4. **Usage of software**
5. Download the complete source code of the project from a public Github repository: <https://github.com/digitalvelocity/AutoEmailClassification> or clone the repo from <https://github.com/digitalvelocity/AutoEmailClassification.git>
6. Create folder for storing client credentials: on a Windows PC, create the following folder: C:\temp\AutoEmailClassification\Credentials
7. Install .NET Framework 4.6 if it is not already installed on the PC. Here’s the link to download the installer: <https://www.microsoft.com/net/download/dotnet-framework-runtime/net462>
8. Run the console app with arguments, which represent the topic user is interested in. e.g. “EmailClassifierConsole.exe missed call”. Here “missed call” is the topic to find.
9. Gmail API for .NET can be found here: https://developers.google.com/gmail/api/quickstart/dotnet