|  |  |
| --- | --- |
|  |  |

Project Documentation

**Mail Management System**

**Documentation**

Maximilian Müller, D953

Dimitrios Tsigaropoulos, E098

Dr. Lamya Abdullah

Content

[1 Project 1](#_Toc185795223)

[1.1 Features 1](#_Toc185795224)

[1.2 Getting Started 1](#_Toc185795225)

[2 Code-Level Documentation 1](#_Toc185795226)

[2.1 File Structure 1](#_Toc185795227)

# Project

This Project is an assignment for the programming course.

The goal is a command-line based Mail-Management-System.

## Features

User creation, login, logout, folder-management, mail-management, contact-management, mail-writing.

## Getting Started

Download all components and make sure that they are arranged in the file structure shown below. Or clone the git project: <https://github.com/MaxNize/MMSAssignment.git> .

No further resources are needed then Python. Every library used is a standard library of Python. Please use Python 3.10.4 or above.

To run the program, run the ‘App.py’ file.

Your command line will be the in- and output panel.

# Code-Level Documentation

## File Structure

Src/

├─ App.py

├─ Contact.py

├─ ContactManagementSystem.py

├─ db

│  ├─ data.db

│  └─ db.py

├─ Folder.py

├─ FolderManagementSystem.py

├─ Mail.py

├─ MailManagementSystem.py

├─ Programm.py

├─ README.md

├─ SearchManagementSystem.py

├─ System.py

├─ ToDo.txt

├─ User.py

├─ UserManagementSystem.py

└─ \_\_init\_\_.py

## Usage of files

### App.py

App.py is used to start the whole Program.

### Contact.py

Contact.py contains the class contact. Contact has the attributes, name (is a given name from the user like a nickname), “firstName”, “lastName”, “mail”, “username” and “phone”. Theis attributes are stored in the database and are manipulated in ContactManagementSystem.py.

### ContactManagementSystem.py

ContactManagementSystem.py has three imports, first:” import datetime” is used provides classes for working with dates and times, second: “import System” because ContactManagementSystem.py is a child of System.py and the last import: “import Contact” is used to manipulate the data in the database. The Main functions are to add (“addNewContactQ()”), delete(“deleteContactQ()”) , edit(“editContactQ()”) and send(“sendMailToContact()”) a Mail to a contact. The different variations of the “baseQuestion“- function shows all features the user can use via a menu on the consol. Depending on which feature the user selects in the console the Menu updates itself “updateBaseQuestion”-function.

### Folder.py

Folder.py imports “Mail” since the folder contains mails and if the folder is manipulated it also effects the mails. The Class “Folder” contains the attribute:”name” and a list were every mail in the folder is saved. Folder.py also contains three functions which allows create(“createMail”), delete(“deleteMail()”) and get(“getMailBySubject()”) a mail.

### FolderManagementSystem.py

FolderManagementSystem.py is used to manage the various fodders themselves. FolderManagementSystem.py imports “System”, since it is also a child of it. FolderManagementSystem.py has a menu through which the user can use the features that the “baseQuestion()“-function contains. FolderManagementSystem.py shows all the folders the user has and lets him enter the folders. In the folder itself through “updateBaseQuestion()”-function the menu updates gives the user the option to delete the folder(“deleteFolder()”) and shows all the mails in the folder. The mails can be selected which allow the user to enter the “MailManagementSystem.py”.

### Mail.py

Mail.py contains the class Mail. Mail has the attributes: “subject”, “to”, “sender”, “bcc”, “cc”, “content”, “attachmentsPath”, and “timestamp”. These attributes define the structure of a mail and are used by “FolderManagementSystem.py” managing mail functionalities.

### MailManagementSystem.py

MailManagementSystem.py is used to manage the various actions related to the user's mails. MailManagementSystem.py imports "System" since it inherits from it and imports “datetime” which provides classes for working with dates and times. The system provides a menu through the "baseQuestion()" function that allows the user to delete, move, answer, or forward mails. "updateBaseQuestion()" updates the menu to reflect the current mail, and the "updateTitle()" function updates the title based on the selected mail. The user can delete a mail with "deleteMailQ()", move it to another folder using "moveMailQ()", answer the mail through "answerMailQ()", or forward it with "forwardQ()".

### Programm.py

Programm.py is used to initialize and manage the core functionalities of the application. Programm.py imports several management systems, including “UserManagementSystem”, “MailManagementSystem”, “FolderManagementSystem”, “ContactManagementSystem”, and “SearchManagementSystem”. It also imports the database (“db”) to handle data storage. The file establishes connections between these systems, enabling them to interact with each other. Through the “setupUsers()”, “setupMails()”, and “setupContacts()” functions, Programm.py loads existing data from the database into the respective systems. The “mainloop()” function starts the user interface, allowing the user to interact with the system. The “save()” function saves all changes, including user data, emails, and contacts, back to the database for persistence.

### SearchManagementSystem.py

SearchManagementSystem.py is used to manage the search functionalities within the application. It imports the "System" class and the database (“db”) for retrieving mail data. The class offers users the ability to search received mails by different criteria, including sender, subject, and attachment. The “baseQuestion” function displays the available search options. Main functions like “searchBySender()”, “searchBySubject()”, and “searchByAttachment()” allow users to input search queries and retrieve matching results from the database. Once a search result is selected, the user can view the mail by opening it in the “MailManagementSystem.py”. The “specificQuestionnaire()” function handles the user's selection of a search option, triggering the corresponding search function.

### System.py

System.py is used as a base class for managing system-wide functionalities. It includes general text constants and patterns, such as “TEXTdivider”, “TEXTspacer”, and “emailPattern”, which are used across different parts of the program. The System class provides various utility methods, including “safeQuestion()”, which ensures that user input matches the expected type, and “TEXTheading()”, which formats and displays headings for user interactions. The class also includes safety checks for user responses (“checkSafetyQuestion()”) and email validation (“checkForMailpattern()”). The “baseQuestionnaire()” function presents a menu to the user, allowing them to select options and interact with the system. The class runs in a loop using “mainloop()”, continuing until the running variable is set to False.

### User.py

User.py contains the User class and its attributes: “username”, “firstName”, “lastName”, “mail”, and “pw”. The folders attribute holds the folders assigned to the user, which include default folders: "Inbox", "Sent", and "Trash" for a new user, or an empty list for an existing one. The contacts attribute stores the user's contacts. The “sendMail()” function enables the user to send an email by creating a mail object in the "Sent" folder. Additionally, the class has several getter and checking methods: “getFolders() “retrieves a list of folder names, “getFoldersForSave()” prepares a string for saving folder names, and “checkForExistingFolder()” checks if a folder exists. The “getFolder()” method retrieves a specific folder by name, while “getMailsOfFolder()” returns all the mails in a folder. The “checkPw()” and “getPw()” methods are used to validate and access the user's password.

### UserManagementSystem.py

The UserManagementSystem.py file defines the “UserManagementSystem” class, inheriting from “System.py”, to handle user management tasks such as user creation, login, and folder management. Key attributes include title (shows current user status), “baseQuestion” (menu options), and “users” (list of registered users). The “FolderManager”, “ContactsManager”, and “SearchManager” attributes manage folders, contacts, and email search respectively.

Main functions include checking for existing usernames (“checkForExistingUsername()”), retrieving users “(getUser()”), updating the title and menu (“updateTitle()”) and “updateBaseQuestion()”), and handling actions like logging out (“logout()”), writing emails (“writeMail()”), managing folders (“openFolder()”), and creating or deleting users (“createUserQ()”) and (“deleteUserQ()”). The “specificQuestionnaire()” method processes user input based on login status and user activity, offering relevant options for logged-in users or guests.