**My Audio Email**

**Outline Design Specification**

**version: 1.00**

**Document information**

| **Name of Deliverables** | My Audio Email Outline Design Specification |
| --- | --- |
| **Author** | Ziyu Wang |
| **Course** | COM6103 Team Software Project |
| **Team** | Team02 |
| **Status** | First draft |
| **Version** | V1.00 |

**Change history**

| **Version** | **Change Describe** | **Changer** | **Date** | **Note** |
| --- | --- | --- | --- | --- |
| V1.00 | Create | Ziyu Wang | 26-03-2023 |  |
|  |  |  |  |  |

Catalog

[1 Introduction](#_heading=h.gjdgxs) **4**

[1.1 Purpose](#_heading=h.30j0zll) 4

[1.2 Scope of application](#_heading=h.3znysh7) 4

[1.3 Readership](#_heading=h.tyjcwt) 4

[2 System functional design](#_heading=h.3dy6vkm) **4**

[2.1 System overview](#_heading=h.1t3h5sf) 4

[2.2 Functional details](#_heading=h.4d34og8) 4

[2.2.1 Login](#_heading=h.2s8eyo1) 4

[2.2.1.1 Outline](#_heading=h.dfmqqcu303zy) 5

[2.2.1.2 View detail](#_heading=h.b4564e5rubak) 5

[2.2.1.3 Event processing](#_heading=h.ucc0adypp1qc) 5

[2.2.1.4 Processing details](#_heading=h.qzxfecqld47i) 5

[2.2.2 MenuList](#_heading=h.79aawnq0x74n) 6

[2.2.2.1 Outline](#_heading=h.kgf8hfgzn6h5) 6

[2.2.2.2 View detail](#_heading=h.lqo45zp37iur) 7

[2.2.2.3 Event processing](#_heading=h.yh0t12137dgv) 7

[2.2.2.4 Processing details](#_heading=h.p0e9yhw9t6e3) 7

[2.2.3 Inbox](#_heading=h.17dp8vu) 7

[2.2.3.1 Outline](#_heading=h.bmarcobeuj05) 7

[2.2.3.2 View detail](#_heading=h.ijw7kiid3iai) 7

[2.2.3.3 Event processing](#_heading=h.3sgnnwyh1eni) 8

[2.2.3.4 Processing details](#_heading=h.4vqbkmpnw0a2) 8

[2.2.4 Compose](#_heading=h.y5mz5yn8v1bq) 9

[2.2.5 Setting Voice](#_heading=h.ahoa8h653pbw) 9

# Introduction

## Purpose

Prepare this outline design specification for my audio email outline design based on the FEATURE LIST, aiming to be accurate, comprehensive and unambiguous in content. It facilitates control and management of the software development process and allows programmers to move into the detailed design phase; it also demonstrates the commonality of the software to gain wider application. Once completed, this outline design specification will serve as the basis and foundation for the project team's detailed design and development testing.

## Scope of application

This document applies to my audio mail, changes to this document will affect my audio mail development, testing.

## Readership

Project team members, client, advisor. The document may not be read or used by anyone other than the person to whom it is addressed without the written permission of the project leader.

# System functional design

## System overview

Individuals with very busy schedules sometimes find it difficult to catch up on emails. It is estimated that the average worker spends more than 11 hours a week on emails. This number of hours increases with senior executives or very busy individuals and as such, they miss important emails and deadlines, hire an executive assistant, or look for other creative ways to fix this problem. My Audio Email, is an application that attempts to fix this problem by converting a email of user into speech. Essentially, the application should work as a regular email client that allows users to read and send emails but in addition, the application should be able to read the emails to the user, using a chosen voice. The application can be developed as windows or web application. The email uses SMTP and IMAP protocols, which means that most email accounts can use the application.

## Functional details

### Login

#### Outline

1. **Functional description**

Log in by entering your email account and password, you can also choose your work or personal email, default choose personal email.

Users need to manually request an app password from a different email provider

1. **Start-up conditions**

Enter the ip and port number to access the web page (enter the domain name in the official environment)

#### View detail

1. **view**

Coming soon~~

Img 2.2.1.2 Login view

1. **Input item**

| **ID** | **Item Name** | **Contents** | **Note** |
| --- | --- | --- | --- |
| 1 | UserName | Username of the login user |  |
| 2 | Password | Password of the logged in user |  |
| 3 | Mailbox type | Mailbox type of the logged in user | Work/Personal |
| 4 | SMTP | The SMTP server of the account |  |
| 5 | IMAP | The IMAP server of the account |  |

1. **Control item**

| **ID** | **Item Name** | **Method** | **Initial state** | **Note** |
| --- | --- | --- | --- | --- |
| C1 | Login | Button | Enable |  |
| C2 | Reset | Button | Enable | Clear input box |
| C3 | Add | Button | Enable | Add new account |

#### Event processing

| **ID** | **Event** | **Processing content** | **MSG-ID** |
| --- | --- | --- | --- |
| 1 | Page initialization | Initial display of the login page | None |
| 2 | Click [Login] | Verify username and password for login operation | None |
| 3 | Click [Reset] | Clear input | None |
| 4 | Click [Add] | Input the account info | None |

#### Processing details

1. Handling process

If the username and password are stored in the database, they are checked in the backend. Call Jmail to retrieve the email information of user after the checks have passed.

If the account information is not in the database, a request will be sent to the IMAP server corresponding to the account, and if the IMAP server can be successfully connected, the account information will be written to the database, if not, the account information will be incorrect.

(Module developers can add to the processing logic during development)

1. Database references

| user\_id | int(auto\_increment) |
| --- | --- |
| username | varchar(128) |
| password | varchar(128) |
| realname | varchar(128) |
| smtp | varchar(64) |
| imap | varchar(64) |
| type | char |
| device\_type | char |

system\_account table

| device\_id | int(auto\_increment) |
| --- | --- |
| user\_id | int |
| device\_content | varchar(256) |

device table

1. Database Output

### MenuList

#### Outline

* + - * 1. **Functional description**

Used for page jumping.

* + - * 1. **Start-up conditions** This menu will be displayed on all pages after login.

#### View detail

**View**

**Input Item**

| **ID** | **Item Name** | **Contents** | **Note** |
| --- | --- | --- | --- |
|  |  |  |  |

**Control item**

| **ID** | **Item Name** | **Method** | **Initial state** | **Note** |
| --- | --- | --- | --- | --- |
| C1 | Inbox | Button | Enable |  |
| C2 | Unread | Button | Enable |  |
| C3 | Write email | Button | Enable |  |
| C4 | Sent | Button | Enable |  |
| C5 | Spam | Button | Enable |  |
| C6 | Setting | Button | Enable | Click on this button to see the next three buttons |
| C7 | VoiceSetting | Button | Disable |  |
| C8 | AccountSetting | Button | Disable |  |
| C9 | Sync | Button | Disable |  |

#### Event processing

| **ID** | **Event** | **Processing content** | **MSG-ID** |
| --- | --- | --- | --- |
| 1 | Page initialization | Initial selected button is Inbox | None |
| 2 | Click [Inbox] | Show Inbox page |  |
| 3 | Click[Unread] | Show unread page |  |
| 4 | Click[Write email] | Show write email page |  |
| 5 | Click [Sent] | Show Sent page |  |
| 6 | Click[Spam] | Show Spam page |  |
| 7 | Click[Setting] | Expand the menu to show the options available |  |
| 8 | Click[VoiceSetting] | Show Voice setting page |  |
| 9 | Click[AccountSetting] | Show AccountSetting Page |  |
| 10 | Click[Sync] | Show Sync Page |  |

#### Processing details

### Inbox

#### Outline

1. **Functional description**

Default jump to inbox after user login, users can choose which emails they want to read. In addition to this, users can choose to read all emails they want,

1. **Start-up conditions**

Default jump after login

#### View detail

1. **view**

Coming soon~~

2.2.1.2 inbox view

1. **Input item**

| **ID** | **Item Name** | **Contents** | **Note** |
| --- | --- | --- | --- |
| 1 | DateTime | dd/mm/yyyy hh:mm |  |
| 2 |  |  |  |

1. **Control item**

| **ID** | **Item Name** | **Method** | **Initial state** | **Note** |
| --- | --- | --- | --- | --- |
| C1 | Read | Button | Enable | Start reading the selected emails |
| C2 | Set status | Button | Enable | Set “unread” to “read” |
| C3 | Delete | Button | Enable |  |
| C4 | Speed up | Button | Enable |  |
| C5 | Speed down | Button | Enable |  |
| C6 | Stop | Button | Enable | Stop reading |
| C7 | Confirm | Button | Enable | Filter the email by time |
| C8 | Chose | Multi-Check Box | Enable | Check the email for other operations |

#### Event processing

| **ID** | **Event** | **Processing content** | **MSG-ID** |
| --- | --- | --- | --- |
| 1 | Page initialization | Calling a third-party mailbox interface to get email messages |  |
| 2 | Click [Read] Button | Calling the TTS interface to convert emails to voice playback |  |
| 4 | Click [Set status] Button | Set the status of the selected email to read |  |
|  | Click [Delete] Button | Delete the selected email |  |
|  | Click [Speed up] Button | Increase the speed of reading |  |
|  | Click [Speed down] Button | Decrease the speed of reading |  |
|  | Click [Stop] Button | Stop reading |  |
|  | Click [Confirm] Button | Filter the email by date |  |
|  | Click [Chose] Check-Box | Select email |  |

#### Processing details

1. Handling process

Calling a third-party interface to obtain user email data and display it after a user has logged in. Emails are listed in descending order of receipt time.

1. Database references

| mail\_id |  |
| --- | --- |
| to\_user |  |
| from\_user |  |
| subject |  |
| content |  |
| read\_flag |  |
| create\_time |  |
| createby |  |
| delete\_flag |  |
| delete\_time |  |
| deleteby |  |

mail\_info table

TTS\_info table

1. Database Output

After clicking the read button, the data after TTS is stored in the database.

### Compose

1. **Functional description**

The user enters the recipient and the content of the email, and uploads the attachment. And Send the email to the corresponding IMAP server.

1. **Start-up conditions**

Default jump when click WriteEmail

### 

### Setting Voice