Week3-Thu-meeting

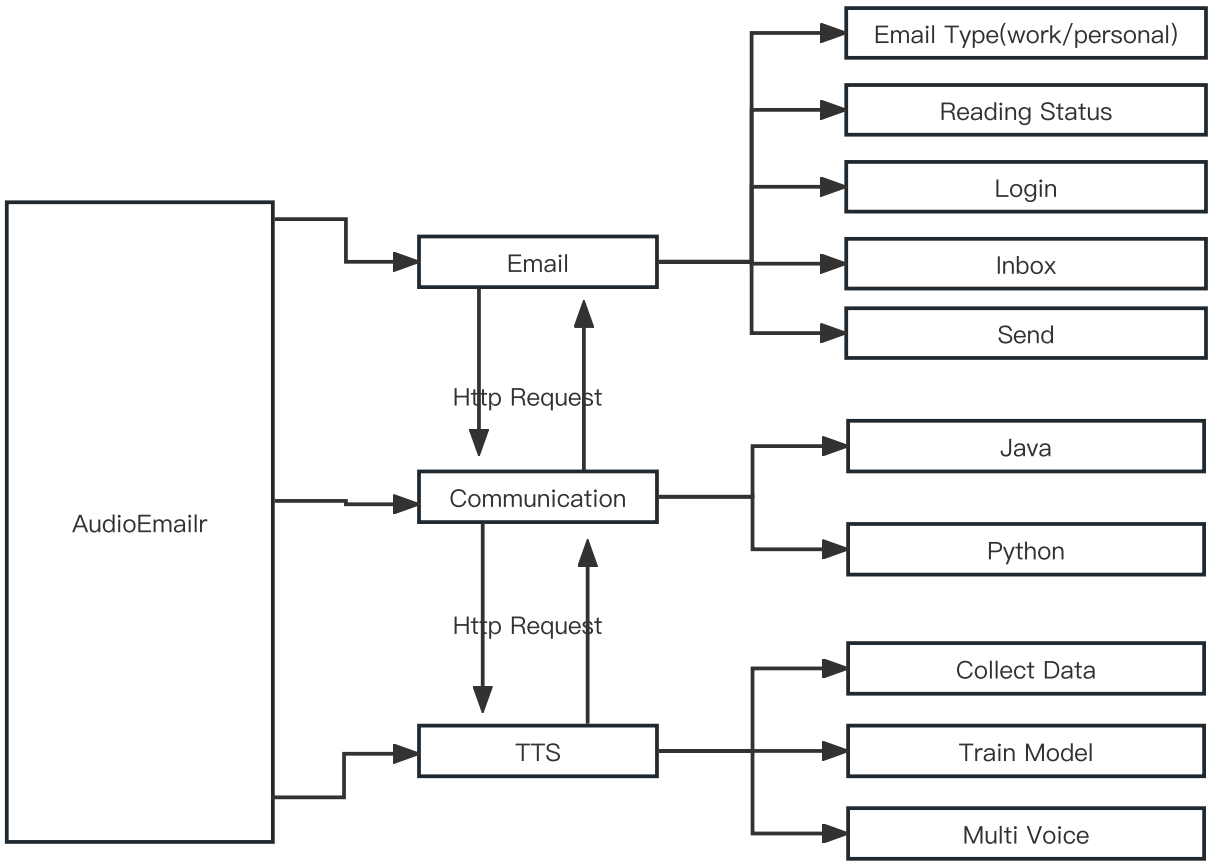
Agenda

- Questions of last meeting(on Tuesday)
- Module/Function division
- Document job
- What should to do in this stage
  - TTS Part
  - Email Part
- Trello
- Git

Questions

- The report should contain evidences of the meetings
- What is peer programming
- Does the advisor say that if we are considering using google docs to save documents then the link needs to be provided to them?
- Personal or business email

Module/Function division



If you have any comments, I can make adjustments to him.

Document

What documentation needs to be prepared during the project?

What should to do in this stage

Is there anything else you think needs to be changed. And assigning tasks.

TTS

- Use python to send and receive http request, and send voice(demo)
- Research on training text-to-speech model
  - How to collect data
  - .....
- Alternate Plans (If anyone is interested)
  - How to use Microsoft api, and the fee Policy

Email

- Email Part
  - Demo of send and receive gmail
  - The general style of the front end

Trello

How to manage development progress with Trello.

Git

- Command Line
- GUI
- IDE

Command Line

- Git
  - One person/module , one branch(different version of the project),
    - On week 5, Ziyu will manually merge them into one branch.
    - ```
      # Create and switch branches
      git checkout -b mq
      # It is a combination of the following two commands
      git branch mq
      git checkout mq
      ```
  - Clone the code, change it and commit it
    - ```
      git clone  <url>
      # download the project

      git add .
      # add the change in current folders

      git commit -m "your comment"
      # commit the change and add some comment

      git pull
      ```

```
# Pull other people's updates to local, to know whether the local code conflict with the cloud code
# If your code have confict with the code in gitlab, you need to handle them manually
# Then you should do the following action again
# git add .
# git commit -m "your comment"
git push
# upload you change to the project
```

- After running `git pull` , you will see the following prompt in git if a conflict arises.
  - `CONFLICT(content): Merge conlict in README.md`
  - ```
    ~/COM6103_GitTest  main ✔
    ▶ git pull
    remote: Enumerating objects: 9, done.
    remote: Counting objects: 100% (9/9), done.
    remote: Compressing objects: 100% (4/4), done.
    remote: Total 7 (delta 2), reused 3 (delta 1), pack-reused 0
    Unpacking objects: 100% (7/7), 1.44 KiB | 490.00 KiB/s, done.
    From https://github.com/Crowds21/COM6103_GitTest
       8afcf4d..0d134d5  main       -> origin/main
     * [new branch]      wzy        -> origin/wzy
    hint: Pulling without specifying how to reconcile divergent branches is
    hint: discouraged. You can squelch this message by running one of the following
    hint: commands sometime before your next pull:
    hint:
    hint:   git config pull.rebase false  # merge (the default strategy)
    hint:   git config pull.rebase true   # rebase
    hint:   git config pull.ff only       # fast-forward only
    hint:
    hint: You can replace "git config" with "git config --global" to set a default
    hint: preference for all repositories. You can also pass --rebase, --no-rebase,
    hint: or --ff-only on the command line to override the configured default per
    hint: invocation.
    Auto-merging README.md
    CONFLICT (content): Merge conflict in README.md
    Automatic merge failed; fix conflicts and then commit the result.
    ```
  - In this example, open `REAME.md` , you can see
    ```

```
  2 # COM6103_GitTest
  1 nie
3   12312321 3
  1 ##  Basic
  2
  3 ```shell
  4 git clone ....
  5 # download the project
  6 git add .
  7 # add the change
  8 git commit -m "your comment"
  9 # commit the change and add some comment
 10 git push
 11 # upload you change to the project
 12
 13 ```
 14 <<<<<<< HEAD
 15  Test Test Test
 16
 17 =======
 18 123123123
 19 >>>>>>> 0d134d53fbfbd5a53892c87ff078fb8c6e467406
```

```
1 <<<<<<< HEAD
2  Test Test Test
3
4 =======
5 123123123
6 >>>>>>> 0d134d53fbfbd5a53892c87ff078fb8c6e467406
```

- And you need to decide what content to keep and what to delete. In this case, I keep all of them

```
1 Test Test Test
2 12312312
```

```
~/COM6103_GitTest  main ✗
▶ nvim README.md

~/COM6103_GitTest  main ✗
▶ git add .

~/COM6103_GitTest  main ✗
▶ git commit -m "solve conflict"
[main c7a1ee9] solve conflict

~/COM6103_GitTest  main ✔
▶ git push
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 1.57 KiB | 1.57 MiB/s, done.
Total 12 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/Crowds21/COM6103_GitTest.git
   0d134d5..c7a1ee9  main -> main
```

GUI and IDEA

Both Pycharm and Intellij have Git integration, so the overall process is similar to the one described above, and with a graphical interface, it's much easier to use.

Or you can use [GitHub Desktop]