TYTS - First Iteration

Tianyang Yang (ty2388)
Xi Lin (xl2793)
Ruolin Jia (rj2537)
Lixing Zhang (lz2636)
Benhao Chen(bc2801)

Github

Repository
https://github.com/hreatx/TYTS

Demo version
https://github.com/hreatx/TYTS/releases/tag/v1.0

CI files & directory

Pre-commit config: TYTS/.pre-commit-config.yaml
Pre-commit report : TYTS/CI_report/pre_commit_report
Post-commit config : TYTS/.travis.yml
Post-commit report: TYTS/CI_report/post_commit_report

Unittest

TYTS/testBuddingWidget.py
TYTS/testEventController.py
Please see *TYTS/CI_report/post_commit_report* in the above directory for the test coverage information. You can search for “coverage report” in that file to locate the output of the code coverage.

User stories and conditions of satisfaction

We accomplished three user stories in our revised proposal. The user can operate on the account, smile to collect the money and change different appearances of Budding. The original user stories and conditions of satisfaction are listed below:

- As a player, I want to create a new account of my own so that I can access to the system and check my private data.
  Acceptance Criteria:
  (a) The player can create a new account/password in database
  (b) The player can login with correct account/password information
    (i) If the password is not correct, reject attempt to login.
  (c) Retrieve the data of the pet using the account-specific information
- As a player, I want to smile to earn virtual cash so that I can gain virtual money
  Acceptance Criteria:
  (d) When the user clicks “making money” button, the program can detect user’s smile within 3 seconds
  (e) After detecting a smile, the player will see his/her money is increased.
- As a player, I want to pat Budding, so Budding can change the appearance
  Acceptance Criteria:
  (a) Budding is clickable.
  (b) When the player clicks Budding, the picture of budding will change the appearance

We are glad that currently, the project goes well and we do not need to change our proposal. And we are planning to add some more functions to our pudding game.

Our CI Plan

Our pre-commit focuses on the code-smell check and static analysis of our code. Each of our members shared the same rules about writing code with quality. Each member needs to comply with these rules in order to be able to commit to the repository. We use Pre-Commit framework.

Our post-commit also includes static analysis by using Pylint. However, we intend to let post-commit to prioritize unit tests. We have one team member serving specifically the test engineer. He is responsible to write test cases, maintain test code and the Travis environment. We think it is efficient to put these tasks in post-commit altogether and let one member to own this whole process.

TA Feedback

1. Since the database runs locally, one user cannot access to an account registered on another computer, otherwise the DB file has to be copied for sharing account information.
   Solution: adopt MySQL server rather than SQLite3 as database system.
2. The key of Microsoft Cloud is currently explicitly shown in the code (and the code is in a public repository). This is dangerous because everyone who reads this code will have access to a private cloud service account.
   Solution: For security, we should load key from file (e.g. a JSON file). And this file should not appear in github repo (use gitignore file to ignore this file).