TypeRacer Project

Final project for first year students at Suleyman Demirel University

12.04.2016

German Ilyin
SDU
Kaskelen,
Abilaykhana 1/1

Overview

Build an IS that allows several users to compete in typing text.

Terminology

1. TypeRacer Project (TRP) - a set of all IT solutions that makes type racing project one working solution.
2. User - human that uses software
3. Player - user that does type racing
4. Admin - user that manages the server of typeracer
5. Client Application - desktop application that is used by players
6. Game - one round of type racing competition
7. Server - piece of software responsible for serving games, getting data from clients
8. Server Application - desktop application, used by admins to control the server
9. CRUD - an acronym for CREATE, RECEIVE, UPDATE, DELETE operations on some data

General Requirements

1. All code should comply to flake8 code style linter
2. PyQT5 should be used for User interface
3. Server should be able to handle at least 5 clients
4. The application must have client and server side
5. The communication between Client and Server should be done using TCP or UDP protocol
6. Server part should store data (user, user scores) in database. SQLite3, PostgreSQL or MariaDB can be used.
7. Server part should store texts on disk using simple text files

Client Application

Client application is a desktop application that is used by players to race. Client application should work only with the server, connecting to database or reading texts

from disk is not allowed. Player runs the client application, authenticates using existing username and password and enters waiting mode.

Waiting mode

1. During the waiting time user can chat with other users online.
2. User can see a list of users online

Game mode

1. Once game is started the UI should guide the player what to type, listen for player keystrokes and send the progress to server.
2. Once game is over show the statistics of the game
    a. Order of players
    b. Time, Accuracy, Speed of each player

Server Application

Server application is a desktop application that is used by admins to CRUD users and typing texts, chat with players, and start/stop the game.

Managing users

Requirements:

1. Admin should be able to:
    a. See the list of existing users
    b. Create a new user by giving username and password
    c. Delete one of the users
2. The data should be stored in database
3. The password should be stored using md5 hash function

Managing text

Requirements:

1. Admin should be able to:
    a. See the list of existing texts
    b. Create a new text by giving title, difficulty and text body
    c. Delete one of existing texts

2. The data should be stored in text files in "`texts`" folder

Managing games and chat

Requirements:

1. Admin should be able to:
    a. See the list of users who are online right now
    b. See chat messages
    c. Enter a chat message
    d. Start the game
        i. Select text to be used or specify to use some random text file
    e. Stop the game
2. During the game UI shows the status of the game, and progress of users
3. Once first user finishes, the countdown is started (30 seconds for example)
4. Once countdown is over, the game should be stopped automatically
5. Once game is over show the statistics of the game
    a. Order of players
    b. Time, Accuracy, Speed of each player

Milestones of project development

Suggestions on the steps and intermediary results. This will help you to finish the project on time.

I. Step 1 - no server yet

Create a client application with login screen, once you enter hardcoded login and password, UI switches to gaming mode, where you can type some hardcoded text. Once you finish typing, UI switches to game statistics.

II. Step 2 - no user/text management

Create server application that accepts client connections, and checks login and password (hardcoded on the server for now). Add chat to client application. Add chat to server application. Admin can enter chat messages, and all clients can enter and see all messages sent by all other users.

III. Step 3 - no user/text management

Now add the game functionality to client and server. It's the same with chat functionality. Clients enter text, all clients get notified of game status and progress.

IV. Step 4 - add user management

Add user management to server app, now check for login and password from database during authentication. Notify all clients when a new user logins or quits.

V. Step 5 - add text management

Add text management on server app. So that when game is started texts are used from texts folder.

VI. Step 6 - finish up everything

Finishing everything that is left. Refactor the code, make it pretty.

Competitive products to check out

1. http://play.typeracer.com/
2. https://typing.io/
3. http://stamina.ru/

Grading

1. Code style - 10pt
2. Compliance to Requirements - 50pt
    a. User management - 5pt
    b. Text management - 5pt
    c. Game process - 20pt
    d. Chat - 10pt
    e. Authentication - 10pt
3. Technical Defense - 40pt
4. Originality in approach and creativity - BONUS 20pt