main.cpp

```
#include "headers.hpp"
#include "tools.hpp"
#include "player.hpp"
#include "map.hpp"
#include "object.hpp"
#include "client.hpp"
#include "toaster.hpp"

#include <chrono>

void loginPage(sf::RenderWindow& window, Player& player, Toaster&
toaster);
void mainLoop(sf::RenderWindow& window, Player& player, Toaster&
toaster);



int main()
{
    //Window
    sf::RenderWindow window(sf::VideoMode(WIDTH, HEIGHT), "Program",
    sf::Style::Close, sf::ContextSettings(24, 8, 8));

    window.setFramerateLimit(60);


    Toaster toaster;

    Player player(40, 21, window, toaster);

    loginPage(window, player, toaster);


    // Game loop
    mainLoop(window, player, toaster);

    return 0;
}


void loginPage(sf::RenderWindow& window, Player& player, Toaster&
toaster)
{
    // background image
    sf::Texture login_tex, signup_tex;
    login_tex.loadFromFile("sprites/loginpage.jpg");
    signup_tex.loadFromFile("sprites/signuppage.jpg");

    sf::Sprite bg_sprite(login_tex);

    // font
    sf::Font input_font;
    if (!input_font.loadFromFile("Fonts/Roboto-Regular.ttf"))
```

```
{
    std::cerr << "Error Loading File.\n";
    return;
}

bool logging_in = true;
bool enter_pressed = false;

// text box and text

TextBox username(v2f(194, 249), v2f(461, 70), "", input_font);
TextBox password(v2f(194, 355), v2f(461, 70), "", input_font);

password.hidden = true;

TextBox* text_boxes[3] = { nullptr, &username, &password };


int box_focused = 1;

sf::Clock clock;

v2f enter_position(193, 469), enter_size(462, 61);
v2f switch_position(530, 185), switch_size(130, 30);

while (window.isOpen())
{
    float dt = clock.restart().asSeconds();

    string typed_text = "";
    int backspace_counter = 0;


    sf::Event event;
    while (window.pollEvent(event))
        if (event.type == sf::Event::Closed)
            window.close();
        else if (sf::Keyboard::isKeyPressed(sf::Keyboard::Escape))
            window.close();
        else if (event.type == sf::Event::TextEntered) {
            // actual typing
            if (event.text.unicode > 32 && event.text.unicode <
            127) {
                typed_text += event.text.unicode;
            }

            // backspaces
            if (event.text.unicode == '\b')
                backspace_counter++;
            if (event.text.unicode == 127) // ctrl backspace
                backspace_counter = -100;

            // tab
            if (event.text.unicode == '\t' && box_focused)
            {
                box_focused = (box_focused + 1) % 3;
                if (box_focused == 0)
                    box_focused = 1;

                text_boxes[box_focused]->turnOnCursor();
            }

            //enter
            if (event.text.unicode == '\r')
                enter_pressed = true;

        }
        else if (event.type == sf::Event::MouseButtonPressed) {
            // Check if mouse click is within the text box
            v2i mousePos = sf::Mouse::getPosition(window);

            box_focused = 0;
            for (int i = 1; i < 3; i++)
            {
                if (text_boxes[i]->inBox(mousePos))
                {
                    box_focused = i;
                    text_boxes[i]->turnOnCursor();
                }

            }


            if (inBounds(enter_position, enter_size, mousePos))
                enter_pressed = true;

            if (inBounds(switch_position, switch_size, mousePos))
            {
                logging_in ^= true;
                if (logging_in) bg_sprite.setTexture(login_tex);
                else bg_sprite.setTexture(signup_tex);
                text_boxes[1]->clearText();
                text_boxes[2]->clearText();
                box_focused = 1;
            }
        }


    if (enter_pressed)
    {
        string error;
        if (logging_in)
        {
            if (player.client.tryLogIn(username.getString(),
            password.getString(), error))
            {
                toaster.toast("Connection Successful!");
                return;
            }
            toaster.toast(error);

        }

        else // signing up
        {
            if (player.client.trySignUp(username.getString(),
            password.getString(), error))
            {
                toaster.toast("Signup Successful!");
                logging_in = true;
                bg_sprite.setTexture(login_tex);
                text_boxes[1]->clearText();
                text_boxes[2]->clearText();
                box_focused = 1;
            }
            else
                toaster.toast(error);
        }
    }

    enter_pressed = false;


    window.clear(sf::Color::Red);

    window.draw(bg_sprite);


    //box highlight
    if (box_focused)
    {
        if (typed_text.size())
            text_boxes[box_focused]->addText(typed_text);
        if (backspace_counter)
            text_boxes[box_focused]->backspace(backspace_counter);
    }

    for (int i = 1; i < 3; i++)
        text_boxes[i]->draw(window, i == box_focused);

    toaster.drawToasts(window, dt);


    window.display();

    }
}

void mainLoop(sf::RenderWindow& window, Player& player, Toaster&
toaster)
{
    v2i screen_center(WIDTH / 2, HEIGHT / 2);


    int frame_count = 0;
    sf::Clock clock;



    Player::HitInfo* hits = new Player::HitInfo[WIDTH];

    std::thread udpThread(&Player::listenToServer, &player);

    player.setFocus(true);

    while (window.isOpen())
    {
        float dt = clock.restart().asSeconds();

        sf::Event event;
        while (window.pollEvent(event))
            if (event.type == sf::Event::Closed)
                player.quitGame();
            else if (player.window_focused && event.type ==
            sf::Event::MouseButtonPressed)
                player.shootGun(event.mouseButton.button ==
                sf::Mouse::Left);
            else if (event.type == sf::Event::LostFocus)
                player.setFocus(false);
            else if (event.type == sf::Event::GainedFocus)
                player.setFocus(true);
            else if (player.window_focused && event.type ==
            sf::Event::MouseMoved)
            {
                v2i current_pos = sf::Mouse::getPosition(window);

                player.rotateHead(current_pos.x - screen_center.x,
                    current_pos.y - screen_center.y, dt);

                sf::Mouse::setPosition(screen_center, window);
            }
            else if (event.type == sf::Event::KeyReleased)
            {
                if (event.key.code == sf::Keyboard::Space)
                    player.respawn();
            }


        //if (frame_count % 100 == 0)
        //    cout << (1 / dt) << "\n";
        player.updateServer();


        player.handleKeys(dt);


        // Graphics
        window.clear(sf::Color::Red);

        player.map.drawSky(); // Sky

        player.map.drawGround();


        player.shootRays(hits); // populate hits[]

        // World

        {
            std::lock_guard<std::mutex> lock(player.mtx);

            player.drawWorld(hits, dt);


            //std::cout << "Elapsed time: " << elapsed.count() * 1000
            << " ms" << std::endl;
        }


        if (player.debug_mode)
        {
            player.rotateHead(1, 0, 0.3);
        }

        player.drawGun(dt); // Gun

        player.drawCrosshair(dt); // Crosshair

        player.drawDeathScreen(dt);


        toaster.drawToasts(window, dt);

        toaster.drawLeaderboard(window, player.leaderboard, dt);

        window.display(); // Render to screen

        frame_count++;
    }
```

```
    delete[] hits;
}
```