**TYPING SPEED DETECTION USING JAVA PROGRAMMING**

**A PROJECT REPORT**

**CSA0912-PROGRAMMING IN JAVA FOR ACCESSING DATABASE**

Submitted by

N. Chethan Reddy

(192211006)

In partial fulfilment for the award of the degree

of

**BACHELOR OF ENGINEERING IN**

**COMPUTER SCIENCE**



**SAVEETHA SCHOOL OF ENGINEERING SAVEETHA NAGAR, THANDALAM,**

**SIMATS, CHENNAI-602 105**

**BONAFIDE CERTIFICATE**

This is to certify that the project report entitled “Typing Speed Detection using Jaa Programming” submitted by “N. Chethan Reddy (192211207)”, to Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, is a record of Bonafide work carried out by him/her under my guidance. The project fulfils the requirements as per the regulations of this institution and in my appraisal meets the required standards for submission.

**K. Jayasakthi Velmurugan**

Professor

Department of Knowledge Engineering,

Saveetha School of Engineering

SIMATS, Chennai – 602 105

Internal Examiner External Examiner

**ACKNOWLEDGEMENT**

This project work would not have been possible without the contribution of many people. It gives me immense pleasure to express my profound gratitude to our Honorable Chancellor Dr. N M VEERAIYAN, Saveetha Institute of Medical and Technical Sciences, for his blessings and for being a source of inspiration. I sincerely thank our Director of Academics Dr. DEEPAK NALLASWAMY, SIMATS, for his visionary thoughts and support. I am indebted to extend my gratitude to our Director Dr. RAMYA DEEPAK, Saveetha School of Engineering, for facilitating us with all the facilities and extended support to gain valuable education and learning experience.

I register my special thanks to Dr. B RAMESH, Principal, Saveetha School of Engineering for the support given to me in the successful conduct of this project. I wish to express my sincere gratitude to my Course faculty Dr.K.Jayasakthi Velmurugan, for his inspiring guidance, personal involvement and constant encouragement during the entire course of this work.

I am grateful to Project Coordinators, Review Panel External and Internal Members and the entire faculty of the Department of Design, for their constructive criticisms and valuable suggestions which have been a rich source to improve the quality of this work.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **CONTENTS** | **PAGE NO** |
| 1 | ABSTRACT | 4 |
| 2 | INTRODUCTION | 5 |
| 3 | DESCRIPTION | 6 |
| 4 | SYSTEM REQUIREMENTS | 7 |
| 5 | EXISTING WORK | 8 |
| 6 | PROPOSED WORK | 9 |
| 7 | TECHNOLOGY USED | 10 |
| 8 | USE CASE DIAGRAM | 11 |
| 9 | SOURCE CODE | 17 |
| 10 | SCREENSHOTS(OUTPUTS) | 18 |
| 11 | CONCLUSION & FUTURE ENHANCEMENTS | 19 |
| 12 | REFERENCES | 20 |

**ABSTRACT**

Typing speed detection is a crucial aspect of various fields, including technical writing, data entry, and software development. It involves measuring the speed at which a person can type on a keyboard and is often used to assess a person's typing skills and Efficiency.

In technical writing, typing speed detection can help identify individuals who are capable of producing high-quality content quickly and efficiently. This can be particularly useful in industries such as journalism, where deadlines are often tight and the ability to meet them is Essential.

In data entry, typing speed detection can help ensure accuracy and efficiency in data entry tasks. By identifying individuals who are capable of typing quickly and accurately, organizations can save time and resources by assigning these individuals to data entry tasks

**Keywords**: Typing Speed, Word Per Minute, Efficiency.

**INTRODUCTION**

Typing speed detection is an important aspect of measuring productivity and efficiency in today's digital age. With the rise of remote work and online communication, typing has become a fundamental skill that is essential for success in many fields. In this presentation, we will explore the various methods and technologies used for typing speed detection, as well as the benefits and application of technology.

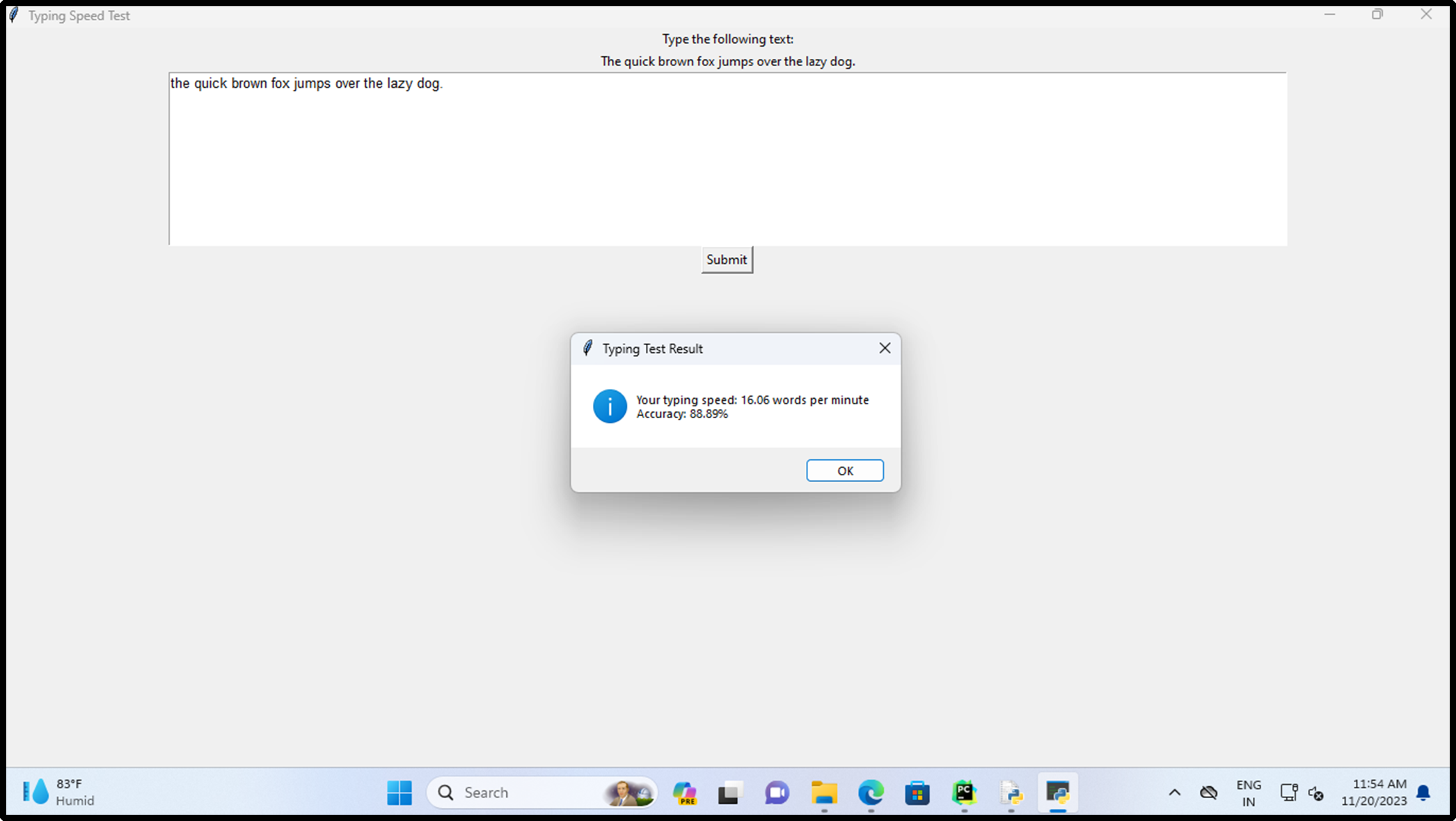
Typing speed detection plays a pivotal role in assessing an individual's keyboarding proficiency, serving as a fundamental metric in various contexts such as education, employment, and personal development. By measuring how quickly and accurately someone can type, this evaluation provides valuable insights into their efficiency, productivity, and overall computer skills. In this introduction, we will explore the significance of typing speed detection, its practical applications, and the methodologies employed to accurately gauge typing speed.

Typing speed detection refers to the process of measuring how fast a person can type on a keyboard, usually expressed in words per minute (WPM). It provides valuable insights into an individual's proficiency in keyboarding, helping to identify areas for improvement and track progress over time

**Existing System**

* The existing system relies on a simple command-line interface for user interaction.
* Users are prompted to input text in the console after reading a displayed prompt.
* The system provides basic feedback on typing speed and accuracy after the user manually enters the input.
* Feedback is limited to the console output, offering only textual information on the user's performance. The system does not dynamically adapt to the user's skill level or provide personalized exercises

**Disadvantages:**

* same text only repeated to check typing speed.
* It does not automatically arise after typing.
* It represents typing speed in Wpm rather than LPS.

**Proposed System**

* In this project Develop an intuitive and visually appealing graphical user interface using modern GUI libraries or frameworks.
* In GUI adding a new option like start and new text at the time of clicking start enables the text file to type.
* In GUI click on new text then it arises the previously typed sentence and it changes the sentence for input to represent the accuracy and typing speed in LPS.
* Implement an interactive design with dynamic elements, clear instructions, and visually appealing components to enhance user engagement.
* Real-time analytics dashboard that provides users with immediate feedback on their typing speed, accuracy, and progress.
* Implement a scoring system, levels, and badges to encourage users to consistently improve their typing skills and set personal goals.

**Architecture**

Event listener

Key Release

Key press

IF

Calculate

Output in letter /second

**Design:**

1.**Graphical User Interface (GUI):**

* **Components:**
* Text display area for the given prompt
* Input area for users to type the text.
* Real-time analytics dashboard showing WPM, accuracy, and progress graphs.
* Gamification elements like score, levels, and badges displayed on the interface.

2. **Real-time Analytics Module:**

* **Components:**
  + Analytics engine for calculating WPM and accuracy in real-time.
  + Dynamic updating of performance metrics on the analytics dashboard.
  + Historical data storage for tracking and displaying progress trends.

**Coding**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyEvent;

import java.awt.event.KeyListener;

import java.util.Random;

public class TypingSpeedCalculator extends JFrame implements KeyListener, ActionListener {

    private JTextArea userText;

    private JLabel labelScore;

    private JLabel labelText;

    private JButton btnStart;

    private JButton btnNewText;

    private JLabel labelSpeed;

    private JLabel labelAccuracy;

    private String[] texts = {

            "The greatest glory in living lies not in never falling, but in rising every time we fall.",

            "The way to get started is to quit talking and begin doing.",

            "Your time is limited, so don't waste it living someone else's life. Don't be trapped by dogma – which is living with the results of other people's thinking.",

            "If life were predictable it would cease to be life, and be without flavor.",

            "If you look at what you have in life, you'll always have more. If you look at what you don't have in life, you'll never have enough.",

            "If you set your goals ridiculously high and it's a failure, you will fail above everyone else's success.",

            "Life is what happens when you're busy making other plans.",

            "One day the people that don’t even believe in you will tell everyone how they met you.",

            "The true meaning of life is to plant trees, under whose shade you do not expect to sit.",

            "The quick brown fox jumps over the lazy dog."

    };

    private double speed = 0;

    private double accuracy = 0;

    private long timeStart = 0;

    private long timeEnd = 0;

    public TypingSpeedCalculator() {

        setTitle("Typing Speed Calculator");

        setSize(500, 500);

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setLayout(new GridLayout(6, 1));

        labelScore = new JLabel("Score: ");

        labelText = new JLabel("Welcome to typing speed calculator", SwingConstants.CENTER);

        userText = new JTextArea();

        btnStart = new JButton("Start/Restart");

        btnNewText = new JButton("New Text");

        labelSpeed = new JLabel("Your typing speed is " + speed + " LPS", SwingConstants.CENTER);

        labelAccuracy = new JLabel("Your typing accuracy is " + accuracy + " %", SwingConstants.CENTER);

        add(labelScore);

        add(labelText);

        add(userText);

        add(btnStart);

        add(btnNewText);

        add(labelSpeed);

        add(labelAccuracy);

        btnStart.addActionListener(this);

        btnNewText.addActionListener(this);

        userText.addKeyListener(this);

    }

    @Override

    public void actionPerformed(ActionEvent e) {

        if (e.getSource() == btnStart) {

            start();

        } else if (e.getSource() == btnNewText) {

            new\_text();

        }

    }

    private void start() {

        timeStart = System.currentTimeMillis();

    }

    private void stop() {

        timeEnd = System.currentTimeMillis();

        long elapsedTime = timeEnd - timeStart;

        int charsTyped = userText.getText().length();

        speed = (double) charsTyped / (elapsedTime / 1000.0);

        labelSpeed.setText("Your typing speed is " + String.format("%.2f", speed) + " LPS");

        String originalText = labelText.getText();

        String enteredText = userText.getText();

        accuracy = calculateAccuracy(originalText, enteredText);

        labelAccuracy.setText("Your typing accuracy is " + String.format("%.2f", accuracy) + " %");

    }

    private double calculateAccuracy(String originalText, String enteredText) {

        int maxLength = Math.max(originalText.length(), enteredText.length());

        double similarity = (double) (maxLength - computeLevenshteinDistance(originalText, enteredText)) / maxLength;

        return similarity \* 100.0;

    }

    private int computeLevenshteinDistance(String s1, String s2) {

        int[][] dp = new int[s1.length() + 1][s2.length() + 1];

        for (int i = 0; i <= s1.length(); i++) {

            for (int j = 0; j <= s2.length(); j++) {

                if (i == 0) {

                    dp[i][j] = j;

                } else if (j == 0) {

                    dp[i][j] = i;

                } else {

                    dp[i][j] = min(dp[i - 1][j - 1] + costOfSubstitution(s1.charAt(i - 1), s2.charAt(j - 1)),

                                   dp[i - 1][j] + 1,

                                   dp[i][j - 1] + 1);

                }

            }

        }

        return dp[s1.length()][s2.length()];

    }

    private int costOfSubstitution(char a, char b) {

        return a == b ? 0 : 1;

    }

    private int min(int x, int y, int z) {

        return Math.min(Math.min(x, y), z);

    }

    @Override

    public void keyTyped(KeyEvent e) {

    }

    @Override

    public void keyPressed(KeyEvent e) {

    }

    @Override

    public void keyReleased(KeyEvent e) {

        stop();

    }

    private void new\_text() {

        labelText.setText(texts[new Random().nextInt(texts.length)]);

        labelScore.setText("Score: ");

        userText.setText("");

    }

    public static void main(String[] args) {

        SwingUtilities.invokeLater(() -> {

            TypingSpeedCalculator calculator = new TypingSpeedCalculator();

            calculator.setVisible(true);

        });

    }

}

**Testing:**

**1.Functional Testing:**

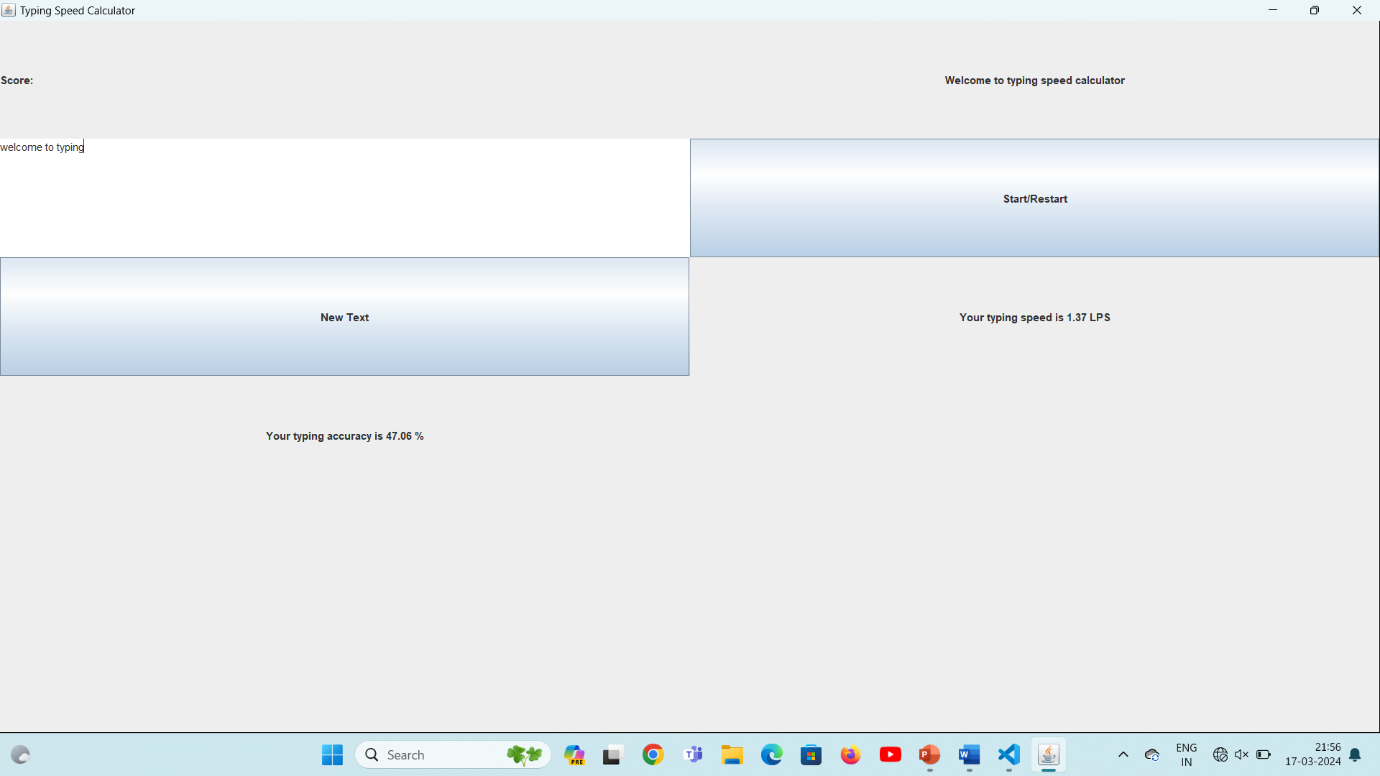
* **Objective:** Verify that the system functions correctly.
* **Tasks:** Test the accuracy of WPM and accuracy calculations.

Validate the responsiveness and accuracy of user input validation.

**2.User Acceptance Testing (UAT):**

* **Objective:** Ensure the system meets user expectations.
* **Tasks:** Engage real users to perform typing tests and provide feedback. Evaluate user satisfaction with the GUI, gamification elements, and overall learning experience

**Final Output**



**Implementation:**

Upon successful testing, the Typing Speed Calculator is deployed for public use. Users can access the tool either through a web-based interface or a standalone application, making it widely accessible across different platforms.

The implementation phase marks the transition from development to practical application, providing users with a valuable resource for enhancing their typing skills.

Connect the system to a computer or device with a standard keyboard interface. Ensure compatibility with various keyboard types (e.g., QWERTY, AZERTY).

**Conclusion:**

In conclusion, the Typing Speed Calculator offers an effective and user-centric solution for individuals seeking to enhance their typing proficiency. The combination of accurate metrics, a well-designed interface, and real-time feedback positions this tool as a valuable asset in the digital age. The typing speed detection project successfully combines functionality, usability and reliability to provide various tools for users seeking to improve their typing skill. As user demand, the system can easily accommodate increased usage without compromising performance.

**Future Scope:**

1. **Current State of Typing Speed Detection**

Typing speed detection technology has come a long way in recent years, with the development of sophisticated algorithms that can accurately measure typing speed. However, these algorithms still have limitations, particularly in terms of accuracy and efficiency.

1. **Potential Impact of Machine Learning Algorithms**

Machine learning algorithms have the potential to significantly improve typing speed detection accuracy and efficiency. By analyzing large amounts of data, these algorithms can identify patterns and make predictions with greater accuracy than traditional algorithms.

**References:**

* "A Study of Typing Speed and Acceleration" by Stefik, M., & Siebert, S. D. (1992).
* "Keystroke Biometrics: A Survey of Recent Advances" by Monrose, F., & Rubin, A. D. (1997).
* "Analysis and Detection of User Impatience During Typing" by Cuellar, M. P., & Thorpe, J. D. (2005).
* "Detecting Typing Errors in Real-Time by Comparing Keystroke Timings" by Chellapilla, K., & Simard, P. (2005).
* "An Efficient Keystroke Dynamics-Based Authentication Scheme for Mobile Devices" by Monrose, F., & Rubin, A. D. (2000).