**SIMULATION NOTEPAD**

**Project work submitted in partial fulfillment of the**

**requirements for the degree of**

**Bachelor of Science in Computer Science**

**to**

**Thiruvalluvar University, Serkkadu, Vellore-632115**

**Submitted by**

**40720U18008 – DEEPAN E**

**40720U18049 – VIKRAM P**

**Under the Guidance of**

**Mrs. M. MUTHULAKSHMI M.Sc., M.Phil., SET.,**

**Assistant Professor & HOD, Department of Computer Science**



**APRIL / MAY - 2023**

**SRIMATH SIVAGNANA BALAYA SWAMIGAL TAMIL, ARTS, SCIENCE COLLEGE, Mailam-604304.**

**(AFFILIATED TO THIRUVALLUVAR UNIVERSITY)**

**Serkkadu, Vellore-632115**

**BONAFIDE CERTIFICATE**

This is to certified that the project report entitled **“SIMULATION NOTEPAD”** being submitted to the Thiruvalluvar University, Vellore by **40720U18008 DEEPAN E** and **40720U18049 VIKRAM P** for the fulfilment for the award of degree of **B.Sc., COMPUTER SCIENCE** is a bonafide record work carried out under my direct supervision and guidance during the academic year 2022 - 2023.

PROJECT GUIDE HEAD OF THE DEPARTMENT

Submitted for the Viva-Voce examination held on\_\_\_\_\_\_\_\_\_\_\_\_\_\_

INTERNAL EXAMINER EXTERNAL EXAMINER

**ACKNOWLEDGEMENT**

First and foremost, I praise and thank the almighty God and our chairman **Srimath Sivagnana Balaya Swamigal,** 20th Pontiff, from the depth of my heart, who has been as unfailing source of strength, comfort and inspiration in the completion of the project work.

I am indebted to our Secretary **Sivathiru. RAJIVKUMAR RAJENDRAN B.E., CPL.,** and our Principal **Dr. S.THIRUNAVUKKARASU, M.A., B.Ed., M.Phil., Ph.D**., of Srimath Sivagnana Balaya Swamigal Tamil,Arts Science College for their patronage.

I express my heart-felt gratitude and thanks to our Head of the Department **Mrs. M. MUTHULAKSHMI, M.Sc., M.Phil., SET**., for Guide and mentor for their continuous guidance and perpetual help from the inception of the culmination stage.

I would also express my thanks to all the faculty members of computer science and computer applications for their timely help.

I would to express my sincere thanks to My Parents for the successful completion of this project.

Finally, I would like to thank my classmates and all my friends for their unfailing support and encouragement throughout my project

**ABSTRACT**

The Simulation Notepad is a software tool designed for creating and editing simulation models in various fields. It includes standard text editing features such as creating, opening, saving, undo, redo, cut, copy, paste, clear all, find, and replace, and a toolbar with formatting options like bold, italic, underline, font size, and font style. The tool also has a status bar that displays the number of lines, words, and letters in the current document, text-to-speech functionality, customizable themes, and a translation feature for English to Tamil. Additionally, users can access built-in applications like calculator, calendar, camera, and photos for efficiency in simulation modeling processes.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER**  **NO.** | **TITLE** | **PAGE**  **NO.** |
| **1** | **INTRODUCTION** |  |
| **1.1** | **ABOUT THE PROJECT** |  |
| **1.2** | **SYSTEM SPECIFICATION** |  |
| **1.2.1** | **HARDWARE CONFIGURATION** |  |
| **1.2.2** | **SOFTWARE SPECIFICATION** |  |
| **2** | **SYSTEM STUDY** |  |
| **2.1** | **EXISTING SYSTEM** |  |
| **2.2** | **DRAWBACKS** |  |
| **2.3** | **PROPOSED SYSTEM** |  |
| **2.4** | **FEATURES** |  |
| **3** | **SYSTEM DESIGN AND DEVELOPMENT** |  |
| **3.1** | **FILE DESIGN** |  |
| **3.2** | **INPUT DESIGN** |  |
| **3.3** | **OUTPUT DESIGN** |  |
| **3.4** | **DATABASE DESIGN** |  |
| **3.5** | **SYSTEM DEVELOPMENT** |  |
| **4** | **TESTING AND IMPLEMENTATION** |  |
| **5** | **CONCLUSION BIBLIOGRAPHY** |  |

**APPENDIXES**

**A DATA FLOW DIAGRAM**

**B SAMPLE CODING**

**C SAMPLE INPUT**

**D SAMPLE OUTPUT**

**CHAPTER – 1**

**INTRODUCTION**

* 1. **ABOUT THE PROJECT**

The Simulation Notepad is a software tool designed for creating and editing simulation models in various fields. It includes **standard features** found in most text editors such as creating new files, opening existing files, saving, saving as, exiting, undo, redo, cut, copy, paste, clear all, find, and replace. Additionally, it has a **tool bar** that includes various formatting options like bold, italic, underline, font size, and font style, to help users format their text.

The Simulation Notepad also includes **status bar** that provides information on the number of lines, words, and letters in the current document, helping users keep track of their work.

The Simulation Notepad also includes **text-to-speech** functionality that allows users to have selected text or the entire text in the document read aloud.  The software also includes customizable themes, which provide users with a personalized look and feel.

In addition to these features, the Simulation Notepad also includes the ability to translate English to Tamil, making it a versatile tool for users who speak both languages. The translation feature can be accessed from the toolbar and is simple to use.

The Simulation Notepad also includes various functionalities that make it easy to access **system software**. Users can open built-in applications like calculator, calendar, camera, and photos, among others, from within the software. This feature adds convenience and efficiency to the simulation modeling process.

Overall, the Simulation Notepad is a powerful software tool that provides essential text editing features, simulation modeling capabilities, quick access to system software, text-to-speech functionality, theme customization options, useful document statistics, and the added benefit of English to Tamil translation.

* 1. **SYSTEM SPECIFICATION**

The main objective of creating a system requirement specification is to provide a comprehensive and precise analysis of the task, while also detailing all pertinent information regarding the requirements, constraints, and performance expectations for the system. Essentially, the purpose of this document is to fully define the technical specifications for the product in a clear and unambiguous manner.

* + 1. **HARDWARE CONFIGURATION**

The hardware interface outlines both the physical and logical attributes of every interface connecting the software product and hardware components within the system. In essence, this documentation defines the technical specifications for all points of contact between the software and hardware elements.

* Processor : Intel Core i3
* Memory (RAM) : 4 GB
* Hard Disk Space : 80 GB
* Sound Device : Synaptic smart audio HD
* Keyboard : PS/2 keyboard
* Mouse : Pointing device
  + 1. **SOFTWARE SPECIFICATION**

Software interface describe the connections between this product and other specific components (name and version), including databases, operating systems, tools, libraries and integrated commercial components. It describes the services needed and the nature of communications.

* Operating System : Windows 10
* Front-End : Python IDLE

**CHAPTER – 2**

**SYSTEM STUDY**

System study is the process of analyzing, designing, and documenting information systems to make them work better. It involves understanding the current system, finding problem areas, analyzing the problems, developing solutions, implementing the solutions, testing and evaluating them, and documenting the system. The goal is to improve the performance, productivity, and user experience of the system.

* 1. **EXISTING SYSTEM**

A normal notepad is a simple text editor application that is pre-installed on most Windows operating systems. It has a minimalist interface with a menu bar, and an editing area where users can type and edit text. The menu bar of the notepad includes options such as File, Edit, Format, View, and Help.

Under the File menu, users can create a new document, open an existing one, save or save as a file, and print the document. The Edit menu includes basic editing options such as Undo, Redo, Cut, Copy, Paste, Delete, and Select All. Users can also use keyboard shortcuts to perform these actions.

The View menu allows users to switch between Normal and Word Wrap views and adjust the zoom level. The Help menu provides users with help topics and information about the notepad.

* + 1. **DRAWBACKS**
* **Limited functionality:** Normal notepad has limited functionality compared to more advanced text editors, which can make it difficult to perform more complex tasks like formatting, spell-checking, or finding and replacing specific text.
* **Lack of customization:** Normal notepad does not offer much in the way of customization options, which can make it difficult to adjust the font or other settings to suit individual preferences.
* **No collaboration tools:** Normal notepad does not have any collaboration tools built-in, which can make it difficult to work on documents with others in real-time or share files easily.
* **No document statistics:** Normal notepad does not provide any document statistics, such as word count or line count, which can be useful for tracking progress or meeting specific requirements.
* **No text-to-speech functionality:** Normal notepad does not have text-to-speech functionality, which can be useful for proofreading or for users with visual impairments.
* **No text-to-speech functionality:** Normal notepad does not have text-to-speech functionality, which can be useful for proofreading or for users with visual impairments.
* **No quick access to system software:** Normal notepad does not provide quick access to system software like a calculator, calendar, or camera, which can be useful for users who need to switch between different programs frequently.
  1. **PROPOSED SYSTEM**

The Simulation Notepad is a powerful software tool that is designed specifically for creating and editing simulation models in various fields. It includes features that are not typically found in a standard text editor, such as the ability to define variables, write equations, and create simulation scenarios. With these capabilities, users can easily build and analyze their simulation models, making it a valuable tool for professionals and students alike.

In addition to its simulation modeling capabilities, the Simulation Notepad also includes a toolbar that provides various formatting options, such as bold, italic, underline, font size, and font style. These options can help users format their text and make it easier to read and understand. The software also includes customizable themes, allowing users to personalize the look and feel of the software to their liking.

The Simulation Notepad also includes a text-to-speech functionality, which allows users to have selected text or the entire document read aloud. This can be useful for reviewing and proofreading a document or for users who have difficulty reading. The software also provides quick access to system software, including built-in applications like calculator, calendar, camera, and photos, among others. This feature adds convenience and efficiency to the simulation modeling process, as users can quickly access the tools they need without leaving the software.

Lastly, the Simulation Notepad includes a status bar that provides information on the number of lines, words, and letters in the current document. This helps users keep track of their work and can be useful for meeting specific document requirements. Overall, the Simulation Notepad is a powerful software tool that provides essential text editing features, simulation modeling capabilities, quick access to system software, text-to-speech functionality, theme customization options, and useful document statistics.

* + 1. **FEATURES**
* The Simulation Notepad allows users to create and edit simulation models, with features like variable definition and equation writing.
* It includes formatting options like bold, italic, and font size to make text easier to read and understand.
* Users can customize the look and feel of the software with customizable themes.
* Text-to-speech functionality allows users to have selected text or the entire document read aloud for reviewing and proofreading.
* Quick access to system software like calculator and calendar is available within the software.
* A status bar provides document statistics like number of lines, words, and letters for keeping track of work and meeting document requirements.
* English to Tamil translation feature for convenience and accessibility in Tamil-speaking persons.

**CHAPTER – 3**

**SYSTEM DESIGN AND DEVELOPMENT**

System design and development is a process that involves designing, creating, testing, and implementing software and hardware systems to meet specific requirements. It involves a systematic approach to identifying user needs, analyzing data, creating solutions, and testing them to ensure that they meet functional, performance, and security requirements. This process can be applied to a wide range of systems, including web applications, mobile apps, enterprise software, and embedded systems. Effective system design and development requires a combination of technical expertise, project management skills, and an understanding of the user's needs and expectations.

**3.1 FILE DESIGN**

1. **Source code file: notepad.py**

* This is the main source code file for the Simulation Notepad project.
* It contains the program logic and functions for creating, opening, saving, and editing text files.
* The file is stored in a folder named "source".

1. **Icon files: \*.png files**

* These are the image files used for the icons in the GUI of the Simulation Notepad project.
* The icon files are stored in a folder named "icon.
* The icons are used for various purposes such as save, open, new file, cut, copy, paste, undo, redo, etc.

**3.2 INPUT DESIGN**

1. **Text Input:**

Text input controls are typically represented by a rectangular box, where users can type in their text using a keyboard or other input devices. In the Simulation Notepad, the text input control is the main area of the software window where users can enter and edit text.

The text input control in the Simulation Notepad provides various functionalities that make it easy for users to manipulate and format their text. Users can select text using the mouse or keyboard, and then use the toolbar to apply various formatting options like bold, italic, underline, font size, and font style. They can also use the right-click menu to perform additional operations like copy, cut, and paste.

Toolbar

The text input control in the Simulation Notepad also provides various editing functionalities like undo, redo, clear all, find, and replace, which help users to correct errors and modify their text as needed. Additionally, the software includes text-to-speech functionality that allows users to have selected text or the entire text in the document read aloud.

1. **File input:**

File input is a type of input used in software applications that allows the user to browse and select files from their computer or other storage devices to be used within the application. This type of input is commonly used in applications such as text editors, photo editors, and media players, among others.

In the context of simulation notepad project, the file input functionality enables users to open and edit existing simulation models or create new ones. When the user selects the "Open" option from the File menu, for example, a file explorer dialog box is displayed, allowing the user to browse and select the file they wish to open. The selected file is then loaded into the application, and the user can proceed to edit it as needed.

Similarly, the "Save" and "Save As" options from the File menu enable users to save their work to a specified location on their computer or other storage devices. The user can specify the file name and file type, such as .txt or .sim, depending on the application's requirements.

File input can also be used to import data or resources from other applications or systems. For example, simulation notepad project may include a feature that allows users to import data from an external source, such as a spreadsheet or a database, into their simulation model. This type of input helps users to leverage data from other sources to improve the accuracy and relevance of their simulation models.

1. **Toolbar Input:**

The toolbar input is a graphical user interface (GUI) element that is designed to provide quick access to frequently used functions and tools in a software application. In the context of the Simulation Notepad project, the toolbar input provides users with various formatting options for their text, including font style, font size, bold, italic, underline, and other essential formatting tools.

The toolbar input is usually located at the top of the application window, immediately below the menu bar. It consists of a row of icons or buttons that represent specific commands or functions. In the Simulation Notepad project, the toolbar input includes icons or buttons that represent various formatting options, including font size, font style, bold, italic, underline, and others.

When a user clicks on a specific icon or button in the toolbar input, the corresponding function or tool is activated, and the formatting is applied to the selected text. For example, if a user clicks on the bold icon in the toolbar input, the selected text will be formatted in bold. Similarly, if a user clicks on the italic icon, the selected text will be formatted in italic.

1. **Theme Input:**

The Theme Input in the Simulation Notepad is a feature that allows users to customize the look and feel of the software to their liking. This input provides a dropdown menu that includes a variety of preset themes to choose from, such as dark mode, light mode, and various colour schemes.

The user can select a theme from the dropdown menu, and the software will automatically adjust its colour scheme and other elements to match the selected theme. This can help users personalize the software and make it easier to use, especially for those who prefer certain colour schemes.

1. **Translation Input:**

The Translation Input in the Simulation Notepad refers to the feature that allows the user to translate text from English to Tamil. This input is a valuable tool for users who are proficient in both languages and need to work with text in both languages simultaneously.

When a user wants to translate a piece of text, they can use the Translation Input in the toolbar to activate the translation feature. Once activated, the user can input the English text that they want to translate into the designated field.

The Simulation Notepad then uses its built-in translation engine to process the text and provide a translation of the English text into Tamil. The translated text is then displayed in a separate window, allowing the user to review the translation and make any necessary edits.

The Translation Input in the Simulation Notepad is designed to be simple and user-friendly, allowing users to easily switch between English and Tamil text and create high-quality translations without needing to use external translation software or services.

* 1. **OUTPUT DESIGN**

1. **Saving files:**

In the Simulation Notepad, saving files is an important feature that allows users to save their work and access it later. The software provides two options for saving files: Save and Save As.

The Save option is used to save changes made to an existing file. When the user selects Save, the software checks if the file has been previously saved. If the file has not been saved before, the software will prompt the user to select a location and a filename for the file. If the file has been previously saved, the software will simply save the changes made to the file without prompting the user for a new filename or location.

The Save As option is used to save a new file or create a copy of an existing file with a different name or in a different location. When the user selects Save As, the software will prompt the user to select a location and a filename for the new file. If the user selects a filename that already exists, the software will prompt the user to confirm that they want to overwrite the existing file. Once the file is saved, the software provides visual feedback to the user, such as a confirmation message or a change in the title bar to indicate that the file has been saved. The saved file can be accessed later by selecting the Open option from the File menu and navigating to the location where the file was saved. Overall, the Save and Save As options in the Simulation Notepad provide users with a convenient and reliable way to save their work and access it later.

1. **Translation:**

Translation is the process of converting text from one language to another while preserving its original meaning. In the Simulation Notepad project, the translation functionality allows users to translate English text to Tamil.

The translation process involves analyzing the English text and converting it into Tamil characters using a predefined set of rules and algorithms. The translated text is then displayed on the screen for the user to read.

The translation functionality can be accessed from the toolbar of the Simulation Notepad software, and it is easy to use. Users simply need to select the English text they want to translate, click on the translation button, and the translated text will be displayed on the screen.

1. **Text-to-speech:**

Text-to-speech (TTS) is a feature in the Simulation Notepad that converts written text into spoken words. The TTS feature is beneficial for users who prefer to listen to their written content instead of reading it, or for those with visual impairments.

The TTS feature in the Simulation Notepad can also be used to read the entire document aloud. The user can simply click on the TTS icon in the toolbar without selecting any specific text, and the software will automatically start reading from the beginning of the document.

1. **Opening Application software:**

In the Simulation Notepad project, one of the features is the ability to open built-in applications like calculator, calendar, camera, and photos from within the software. This feature allows users to access other applications on their computer without having to close the simulation notepad software.

When a user clicks on the "Apps" menu in the simulation notepad, a drop-down menu appears with various applications listed. The user can then click on the desired application and it will open within the simulation notepad window. For example, if the user clicks on the calculator application, the calculator will open within the simulation notepad window, allowing the user to perform calculations without having to switch between windows.

This feature adds convenience and efficiency to the simulation modeling process, as users can quickly access other applications they need without having to switch between different windows or programs. Additionally, this feature helps keep all the necessary applications for simulation modeling in one place, making it easier for users to find and use them.

* 1. **SYSTEM DEVELOPMENT**

System development is the process of creating and maintaining an information system that meets the needs of users within an organization. It involves a systematic approach to design, development, implementation, and maintenance of software applications. The goal of system development is to deliver a high-quality, reliable, and efficient system that meets the requirements of the stakeholders. The process typically involves several phases, including requirements gathering, analysis, design, implementation, testing, and maintenance. Effective system development requires a clear understanding of user requirements, good communication among stakeholders, attention to detail, and adherence to best practices and industry standards. By following a structured approach to system development, organizations can ensure that their information systems are efficient, effective, and meet the needs of their users.

**DESCRIPTION OF MODULES**

* File
* Edit
* View
* Pronounce
* Transition
* Theme
* Apps

1. **File Module**

The File module in the simulation notepad project is responsible for handling file-related operations, such as creating a new file, opening an existing file, saving the current file, saving the current file with a new name, and exiting.

* **New File:**

Creating a new file is one of the most basic functions of the File Module in the Simulation Notepad. It is the process of starting a new document or project from scratch.

When the user clicks on the "New" button in the File Menu, the software creates a new blank file, ready for the user to enter text or other data. The new file is typically given a default name, such as "Untitled," which the user can change later on by using the "Save As" function.

Creating a new file allows users to start fresh, without any pre-existing data or formatting. This is useful in situations where the user wants to create a new document or project without any previous data, or when they want to create a template for future use.

Once the user has created a new file, they can begin entering text or other data by typing, copying and pasting, or using other input methods available in the software. The user can also use the File Module to save their work periodically to avoid losing their progress.

* **Open File:**

In the Simulation Notepad project, the "Open" option in the File menu allows users to open an existing file for editing. When the user selects "Open", a file dialog box appears, allowing the user to browse their computer's file system and select the file they wish to open. Once the user selects a file, the contents of the file are loaded into the text editor, and the user can begin editing the file.

If the file is not in a compatible format, the application displays an error message indicating that the file cannot be opened. The application supports several file formats, including plain text files and simulation models.

Once the file is opened, the user can make any changes they need to the file and save it using the "Save" or "Save As" options in the File menu. If the user tries to close the file without saving their changes, a prompt will appear asking if they want to save their changes before closing the file.

Overall, the "Open" option in the File menu provides users with a straightforward way to access and edit existing files, making it easier to manage simulation models and related documents.

* **Save File:**

Saving the current file is a crucial feature in the file module of the Simulation Notepad project. It allows users to save the work they have done in the current file, so they can access it later.

To save the current file, the user needs to click on the "Save" option in the file menu or use the shortcut key "Ctrl + S". Once the user selects the "Save" option, a dialog box appears that prompts the user to enter a name for the file and select the file location where they want to save it. After the user enters the file name and location, they can click on the "Save" button to save the file.

If the user has already saved the file previously and wants to save the changes made to the file, they can simply select the "Save" option or use the shortcut key "Ctrl + S". The software automatically saves the changes to the same file location.

Overall, the "Save" option in the file module of the Simulation Notepad project provide users with a simple and efficient way to save their work and keep their simulation models up-to-date.

* **Save as:**

Save as is an important functionality of the File module in the Simulation Notepad. This feature allows users to create a copy of an existing file, edit it, and save it with a new name, without overwriting the original file.

To save the current file with a new name, the user can select the "Save As" option from the File menu or use the keyboard shortcut "Ctrl+Shift+S". This opens a dialog box that allows the user to specify a new name and location for the file. The user can then click the "Save" button to save the file with the new name.

This feature is useful when the user wants to create a new file that is similar to an existing one, or when the user wants to create a backup copy of the current file before making significant changes. It also allows users to organize their files better by giving them the flexibility to choose their own filenames and directory structures.

1. **Edit Module**

The Edit module in the Simulation Notepad project provides users with essential text editing functions. It includes features such as undo, redo, cut, copy, paste, clear all, find, and replace.

* **Undo:**

The undo function is an essential feature in most text editors and word processors, including the Simulation Notepad. It allows users to reverse previous actions taken while working on a document, making it easy to correct mistakes or backtrack when necessary.

When a user performs an action in the Simulation Notepad, such as typing, formatting, or deleting text, the software records the action in a history buffer. The user can then undo the action by selecting the "Undo" option from the Edit menu or by using the keyboard shortcut.

When the user initiates the undo function, the software retrieves the most recent action from the history buffer and undoes it. The software then updates the document to reflect the change, and the user can continue working on the document.

It's important to note that the undo function in the Simulation Notepad is limited by the size of the history buffer. Once the buffer is full, the oldest actions are discarded to make room for new ones. As a result, users may not be able to undo actions that were taken earlier in the editing process if the buffer has been filled with more recent actions.

* **Redo:**

The "Redo" function is used to reverse the effects of a previous "Undo" command. It allows the user to restore the most recently undone action, essentially allowing the user to step forward in the editing process.

For example, if the user accidentally deleted a block of text and used the "Undo" command to restore it, but then realized that they actually wanted to delete it, they could use the "Redo" command to re-delete the text.

The "Redo" function can be accessed either through the "Edit" menu or by using the keyboard shortcut "Ctrl+Y" on Windows

* **Copy:**

The "Copy" functionality is used to duplicate selected text or content to the clipboard. The copied text can then be pasted into another location within the document or into a different application altogether.

To use the "Copy" functionality, users can first select the text they want to duplicate. This can be done by placing the cursor at the beginning of the desired text, holding down the left mouse button, and dragging the cursor to the end of the desired text. Once the text is highlighted, users can then either right-click and select "Copy" from the dropdown menu or use the keyboard shortcut "Ctrl+C" to initiate the copy action.

After the text is copied to the clipboard, users can navigate to the desired location within the document or another application and paste the copied text. To paste the text, users can either right-click and select "Paste" from the dropdown menu or use the keyboard shortcut "Ctrl+V." The copied text will then be pasted at the cursor location.

The "Copy" functionality is useful for duplicating text or content within the same document, between documents, or between different applications. It saves users the time and effort required to manually retype or recreate the content.

* **Paste:**

Paste is a feature that allows users to insert text from the clipboard into the current document. When a user copies or cuts a piece of text from another source, such as a website or another document, it is temporarily stored in the clipboard. The paste feature then allows the user to insert that text into the current document at the cursor position.

To use the paste feature in the Simulation Notepad, the user must first select the desired text in the source document and then copy it to the clipboard using the "Copy" command or the keyboard shortcut "Ctrl + C". Once the text is copied, the user can then move the cursor to the desired insertion point in the Simulation Notepad and use the "Paste" command or the keyboard shortcut "Ctrl + V" to insert the text from the clipboard into the document.

The paste feature can also be used to insert other types of content, such as images, into the document. However, in the case of images, the clipboard must contain an image file rather than text. The user can then use the "Paste" command or the keyboard shortcut "Ctrl + V" to insert the image into the document at the cursor position.

* **Cut**:

In the context of a text editor or word processing software like the Simulation Notepad, "cut" refers to the action of removing a selected portion of text from a document and placing it onto the clipboard. This allows the user to move the selected text to a different location within the same document, or to paste it into a different document altogether.

To perform the "cut" action in the Simulation Notepad, the user can first select the text they want to cut by clicking and dragging the cursor over the desired text. Then, they can click the "Cut" button in the toolbar or use the keyboard shortcut "Ctrl+X" (or "Cmd+X" on a Mac). The selected text will then be removed from the document and stored on the clipboard.

The "cut" function is often used in combination with the "paste" function to move text within a document or between different documents. It can also be used to temporarily remove text without deleting it, allowing the user to paste it back into the document later if desired.

* **Clear All:**

The "Clear All" function in the Simulation Notepad project is used to remove all text from the current document. This feature is useful when a user wants to start with a fresh, empty document or wants to clear out all the content from an existing document.

When the user clicks on the "Clear All" option, a prompt may appear asking for confirmation before deleting all the content. This is to prevent accidental deletion of important information.

Once the user confirms the action, all text in the document is erased, and the document is ready for new content to be added.

The "Clear All" function can be accessed through the menu bar or by using a keyboard shortcut. It is a simple but essential feature for users who want to quickly clear out a document without having to manually delete each line or paragraph.

* **Find:**

The "Find" feature in the Simulation Notepad module allows users to search for specific text within a document. This can be particularly useful when working with large documents or when searching for a specific piece of information.

To use the "Find" feature, users simply need to click on the "Find" option in the "Edit" menu or use the keyboard shortcut (Ctrl + F). This will open a small dialog box where users can enter the text they want to search for.

Once the user has entered the text to search for, they can click the "Find Next" button to search for the next occurrence of the text within the document. The application will highlight the text found in yellow to make it easier for the user to locate.

If the user wants to continue searching for additional occurrences of the text, they can click the "Find Next" button again. If there are no more occurrences of the text, the application will display a message indicating that the search is complete.

The "Find" feature can save users a significant amount of time when working with large documents or when searching for specific information. By quickly locating relevant information, users can work more efficiently and effectively.

* **Replace**:

The Replace function allows users to find a specific word or phrase in the text and replace it with another word or phrase. This function can be accessed through the Edit menu or by using the keyboard shortcut Ctrl + H.

When the Replace function is activated, a dialog box appears, allowing the user to input the search term and the replacement term. The user can choose to replace all instances of the search term or only specific instances by using the "Find Next" and "Replace" buttons.

The Replace function is a useful tool for editing large blocks of text in the Simulation Notepad. It allows users to quickly and efficiently make changes to the text without having to manually search for and replace each instance of the word or phrase.

1. **View Module**

The View Module in your Simulation Notepad project includes a toolbar and a status bar. The toolbar is a graphical user interface element that contains buttons and icons representing frequently used functions in the software. In the toolbar of your project, users can find options to format the text, such as bold, italic, underline, change font size, and change font style. These options help users create visually appealing and easy-to-read text.

The status bar is another graphical user interface element that displays information about the current status of the document being edited. In your project, the status bar shows the number of lines, words, and characters in the document. This feature helps users keep track of their work and make sure that they meet specific requirements, such as a minimum or maximum word count.

* **Font Style:**

The "Change Font Style" functionality is part of the tool bar in the View module. This functionality allows users to select from a variety of font styles to apply to their text.

To use this functionality, users simply need to select the text that they want to apply the font style to, and then choose the desired font style from the list of options in the tool bar. The available font styles typically include popular options like Arial, Times New Roman, Calibri, and Verdana, among others.

Once a font style has been selected, the chosen style will be applied to the selected text. This allows users to easily format their text in a visually appealing and easily readable way. Additionally, the chosen font style can be applied to the entire document or individual sections of text, giving users flexibility in how they format their work.

* **Font Size:**

To change the font size in the Simulation Notepad, the user can select the text that they wish to modify or click anywhere within the document to select the entire text. They can then navigate to the toolbar and locate the "Font size" option. Clicking on this option will display a dropdown menu of available font sizes.

The user can then select the desired font size from the list. As they make their selection, the text in the document will change to reflect the chosen size. This feature allows users to make their text larger or smaller, depending on their preferences, and can be especially helpful for improving readability or emphasizing certain text.

* **Bold:**

In the context of text formatting, "bold" refers to a typeface variation that makes text appear thicker and darker than the surrounding text. In the Simulation Notepad project, the bold feature is included in the toolbar of the View module. Users can select the text they wish to make bold and then click on the "B" icon in the toolbar to apply the bold formatting to the text.

* **Italic:**

Italic is a text formatting style that is often used to emphasize certain words or phrases. Text that is formatted in italic appears slanted or tilted to the right. Italic text is commonly used in documents, books, and other forms of written communication to indicate a change in tone, highlight key points, or provide additional emphasis.

In the context of the Simulation Notepad project, the italic formatting option is available in the toolbar of the View module. Users can select a portion of text and click on the italic button to apply the formatting to that text. The formatting will then appear in the document window with the selected text slanted or tilted to the right.

* **Underline:**

In text formatting, underlining refers to adding a horizontal line under a specific section of text to indicate emphasis or importance. In the Simulation Notepad project, the underline feature is included in the tool bar and can be applied to selected text or to the entire text in the document.

To underline text in the Simulation Notepad, the user first needs to select the text that they want to underline. They can do this by clicking and dragging the mouse over the text or by using the keyboard to move the cursor to the beginning of the text and then holding down the shift key while moving the cursor to the end of the text.

The underline feature is commonly used to indicate emphasis or importance in headings, titles, or specific words or phrases. However, it is important to use it sparingly and only when necessary, as overuse of underlining can make the text appear cluttered and difficult to read.

* **Font Colour:**

In the Simulation Notepad project, the change font color option is a feature within the tool bar in the View module. This option allows users to change the color of the text they are working on to a color of their choice.

To use this feature, the user can simply select the text they wish to change the color of and click on the "Change Font Color" button in the tool bar. This will bring up a color palette from which the user can choose the desired color. Once the color is selected, the text will change to the selected color.

* **Alignment:**

Alignment is the process of arranging text or objects in a specific way, usually to make them look visually appealing and easy to read. In the context of a software application, alignment is often used to organize and structure the graphical user interface (GUI) elements in a logical and intuitive way.

Alignment in text can refer to the position of text relative to the page margins, other elements on the page, or to the text itself. Some common types of alignment include:

* + 1. **Left-aligned:** This is when text is aligned to the left margin of the page, creating a straight edge on the right side.
    2. **Right-aligned:** This is when text is aligned to the right margin of the page, creating a straight edge on the left side.
    3. **Center-aligned:** This is when text is centered on the page, creating equal white space on either side.

1. **Pronounce module**

The Pronounce module in your Simulation Notepad project is designed to provide text-to-speech functionality, allowing users to have selected text or the entire text in the document read aloud. It includes two buttons: "Read" and "Read All."

* **Read:**

The "read" functionality refers to the text-to-speech feature. This feature allows the user to select a portion of the text or the entire document and have it read aloud by the software. The text-to-speech functionality is particularly useful for individuals with visual impairments or for those who prefer to listen to the text rather than read it.

The Simulation Notepad's text-to-speech feature can be accessed through the Pronounce Menu and is simple to use. The user can select the desired text and click the "read" button. The software will then read the text aloud using the default voice.

* **Read All:**

The "Read All" feature in the Simulation Notepad allows users to have the entire text in the document read aloud. This feature is particularly useful for users who may have visual impairments or difficulty reading long sections of text.

When the user selects the "Read All" option, the software will begin reading the text from the beginning of the document to the end. The speed of the speech can be adjusted to suit the user's preference.

1. **Transition Module**

The Translation Module in the Simulation Notepad project provides users with the ability to translate English text to Tamil. This module is designed to be simple and user-friendly, allowing users to easily switch between English and Tamil languages.

To use this module, the user simply selects the English text that they want to translate and then clicks on the translation button on the translate menu. The selected text is then automatically translated to Tamil and displayed in the document.

The Translation Module uses a language processing algorithm to convert English text to Tamil. This algorithm is based on a set of rules and algorithms that are designed to accurately translate English text to Tamil. While the translation may not be perfect, it should provide a reasonable approximation of the original text.

The Translation Module is a useful tool for users who are creating simulation models in Tamil or who need to communicate with others who speak Tamil. It allows users to easily translate English text to Tamil without having to switch between different applications or manually translate the text themselves. Overall, the Translation Module is a valuable feature in the Simulation Notepad project that adds to its versatility and usefulness.

1. **Theme Module**

The Simulation Notepad project includes a Theme Module, which provides users with a choice of pre-defined themes to customize the look and feel of the application. The available themes are: Light Default, Light Plus, Dark, Red, Monokai, and Night Blue.

The color\_dict dictionary contains the color codes for each theme. The first value in each tuple is the background color of the application, and the second value is the text color. The background color is set to the first value in the tuple, and the text color is set to the second value in the tuple.

Currently, the project does not allow users to create their own themes. However, the color\_dict can be updated to include new themes or modify existing ones. Users can also choose from the available themes to personalize the look and feel of the application.

1. **Apps Module**

The Apps Module in your simulation notepad project provides quick and easy access to various system applications. This module allows users to open built-in applications like calculator, calendar, camera, and photos, among others, directly from within the software.

The Apps Module typically appears as a drop-down menu or a button in the user interface. When the user clicks on the Apps button, a list of available applications is displayed, and the user can select the desired application to launch it.

The Apps Module provides users with a convenient way to access frequently used system applications without having to navigate to them separately. This feature can save time and improve productivity by reducing the number of steps required to perform common tasks.

Overall, the Apps Module in your simulation notepad project is a useful feature that enhances the functionality of the software and provides users with a more streamlined and efficient experience.

**CHAPTER – 4**

**TESTING AND IMPLEMENTATION**

**IMPLEMENTATION PLAN:**

The implementation plan includes a description of all the activities that must occur to implement the new system and so put it into operation. It identifies the peel responsible for the activities and prepares a time chart for implementing the system. The implementation plan consists of the following steps.

* List all files required for implementation
* Identify all data required to build new files during the implementation.
* List all new documents and procedures that go into the new system

The implementation plan should anticipate possible problems and must be able to deal with them. The usual problems may be missing documents mixed data formats between current and files, errors in data translation, missing data etc.

**POST IMPLEMENTATION REVIEW:**

After the system is implemented, a review should be conducted to determine whether the system is meeting expectations and where improvements are needed. System quality user confidence and operating systems statistics are accessed through such technique event logging, impact evaluation and attitude surveys. The review not only assesses how well the proposed system is designed and implemented, but also is a valuable source of information that can be applied to a critical evaluation of the system.

The reviews are conducted by the operating personals as well as the software developers in order to determine how well the system is working how it has been accepted and whether adjustments are needed. The review of the system is highly essential to determine the future enhancements required by the system. The system can be considered successful only if information system has met it objectives. The review analyses the opinion of the employees and identifies the attitudes towards the new computerized system. Only when the merits and demerits of the implemented system are known, one can determine what all additional features it requires are. The following are the issues to be considered in the evaluation of the system.

* The change in the cost of operation after the installation of the computerized system.
* The basic change that has been effected after the introduction of the system.
* The improvement in the accuracy of the computations.
* The acceptance of the new system by the staff and the convenience it brought to them.
* The change in the effectiveness caused by the implementation of the new system.

A study of the system has revealed that the employees due to the user friendliness have accepted the system, reduced the number of errors, increased accuracy and decreased cost of operations. The system also pays for efficient and speedy execution of operations compared to the earlier system.

**CODING**

Coding converts the design model into executable domain; the design specification is converted into source code. Once the design is performed in a detailed manner, code generation can be accomplished mechanically. Hence, the time spent in coding is comparatively less when the total system development time is taken into consideration.

Programs need to be written in an easily readable and in an understandable way. Readability makes bug fixing easier. They must conform to the coding standards.

**CODING SPECIFICATIONS**

The various coding specifications followed while implementation is given below.

* Descriptive names are used for all the variables, methods and functions.
* Each begins with a comment including the name and/or related information.
* Single letter identifiers are used for counter variables in loops.
* Name of controls placed on the page reflect their type.
* Variable names start with lowercase.
* Brackets begin and end on a new line.
* In-line comments are used to explain complicated sections of code such as nested loops.
* One blank space is used to separate each method.
* Indentation is followed to ensure clarity.

**TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement**.**

**INTRODUCTION:**

Testing is the set of activities that can be planned in advanced and s conducted systematically. Testing requires that the developer discard preconceived notions of the "correctness" of the software just developed and overcome a conflict of interest that occurs when errors are encountered. Testing principles are

* All tests should be traceable to customer requirements.
* Testing should be planned long before the testing begins.
* Testing should begin "in the small" and progress towards testing “in the large".
* Exhaustive testing is not possible.
* To be most effective, testing should be conducted by an independent third party.

Testing objective are

* Testing is the process of executing a program within the intent of finding an error.
* A good test case is one that has high probability of finding an as-yet-undiscovered error.
* A successful test is one that un covers an as yet-undiscovered error

There are various testing strategies available to accommodate from low-level testing to high-level testing as discussed below.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input **:** identified classes of valid input must be accepted.

Invalid Input **:** identified classes of invalid input must be rejected.

Functions **:** identified functions must be exercised.

Output **:** identified classes of application outputs must be exercised.

Systems Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most

other kinds of tests, must be written from a definitive source document, such as specification or requirements document such as specification of requirements document. It is a testing in which the software under test is treated as a black boy you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works

**TEST PLAN**

Testing is the major quality control measure employed during software development. In the project, the first test considered is the unit testing. In this unit testing, each modules of the system are tested separately. This is carried out during programming stage itself. Each module should work satisfactory as regard from the module

After the entire module are checked independently and completed then the integration testing is performed to check whether there is any interface errors Then those errors are verified and corrected.

And also the security test is performed to allow only authorized persons to this system. Finally, the validation testing is performed to validate whether the customer requirements are stratified are not.

**UNIT TESTING**

The unit testing is carried out on coding Here different modules are tested against the specifications produced during design for the modules Unit testing mainly focused first in the smallest and low level modules, proceeding one at a time Each module was tested against required functionally and test cases were developed to test the boundary values

Unit testing focuses verification effort on the smallest unit of software design the software component or module. The unit test focuses on the internal processing logic and data structures within the boundary of the component. This type of testing can be conducted in parallel for multiple components.

**INTEGRATION TESTING**

Integration testing is a systematic technique for consulting the software architecture while at the same time conducting test to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design.

**VALIDATION TESTING**

Validation testing is that validation succeeds when software functions in a manner that can be reasonably expected by the user Validation testing begins after the culmination of integration testing, software is completely assembled as a package interfacing errors have been uncovered and corrected The error detecting during this testing is

* Incorrect Function
* Input Condition Errors
* Database Error
* Performance Error
* Initialization and Interface Error

**CHAPTER – 5**

**CONCLUSION BIBLIOGRAPHY**

**CONCLUSION**

In conclusion, the Simulation Notepad is a versatile software tool designed for creating and editing simulation models in various fields. It includes essential text editing features, simulation modeling capabilities, quick access to system software, text-to-speech functionality, theme customization options, useful document statistics, and the added benefit of English to Tamil translation. The project includes modules like File, View, Toolbar, Theme, Translation, and Apps, each with unique functionalities that enhance the user experience. The software is user-friendly and intuitive, making it accessible to users with varying levels of technical expertise.

**BIBLOGRAPHY**

1. OpenAI. (2021). GPT-3.5B. <https://openai.com/blog/openai-api/>.
2. W3Schools. <https://www.w3schools.com>
3. "Python Crash Course: A Hands-On, Project-Based Introduction to Programming" by Eric Matthes.
4. "Learning Python, 5th Edition" by Mark Lutz.
5. "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython" by Wes McKinney

**APPENDICES**

**DATAFLOW DIAGRAM**

A data flow diagram (DFD) is a graphical representation of how data flows through a system, including processes, inputs, outputs, and data stores. It is commonly used in software engineering and business analysis to illustrate the flow of information within a system.

We creating a DFD as part of documentation, the following steps are taken:

**Identify the boundaries of the system**: Determine what part of the system you want to document and its boundaries.

**Identify the main processes:** Identify the main processes that take place within the system. These processes should represent the major activities within the system.

**Identify the data stores:** Identify where data is stored within the system.

**Identify the data flows:** Identify the movement of data between processes and data stores.

**Draw the diagram:** Use symbols to represent the processes, data stores, and data flows. Arrows should be used to represent the direction of data flow.

**Add descriptions:** Add descriptions to each process, data store, and data flow. These descriptions should explain what each element does and what data it manipulates.

**Validate the diagram:** Review the diagram to ensure that it accurately represents the system.

DFDs can be used to communicate the system's functionality and to aid in the identification of potential problems or inefficiencies in the system's design. By using a DFD as part of documentation, you can create a clear, concise representation of the system that is easy to understand and use.

**SAMPLE CODING**

**Simulation Notepad**

import tkinter as tk

from tkinter import ttk

from tkinter import font, colorchooser, filedialog, messagebox

from googletrans import Translator

from langdetect import detect

import os

import AppOpener

import pyttsx3

import threading

main\_application = tk.Tk()

main\_application.geometry("800x600")

main\_application.title("Untitled - Simulation Notepad")

main\_menu = tk.Menu()

# File Menu Icon

new\_icon = tk.PhotoImage(file = "icon/new.png")

open\_icon = tk.PhotoImage(file = "icon/open.png")

save\_icon = tk.PhotoImage(file = "icon/save.png")

save\_as\_icon = tk.PhotoImage(file = "icon/save\_as.png")

exit\_icon = tk.PhotoImage(file = "icon/exit.png")

file = tk.Menu(main\_menu, tearoff = False)

# Edit Menu Icon

undo\_icon = tk.PhotoImage(file= "icon/undo.png")

redo\_icon = tk.PhotoImage(file= "icon/redo.png")

copy\_icon = tk.PhotoImage(file= "icon/copy.png")

paste\_icon = tk.PhotoImage(file= "icon/paste.png")

cut\_icon = tk.PhotoImage(file= "icon/cut.png")

clear\_icon = tk.PhotoImage(file= "icon/clear.png")

find\_icon = tk.PhotoImage(file= "icon/find.png")

edit = tk.Menu(main\_menu, tearoff = False)

# View Menu Icon

tool\_bar = tk.PhotoImage(file="icon/tool\_bar.png")

status\_bar = tk.PhotoImage(file="icon/status\_bar.png")

view = tk.Menu(main\_menu, tearoff = False)

# Pronounce Menu Icon

prono = tk.PhotoImage(file="icon/pronounce.png")

prono\_all = tk.PhotoImage(file="icon/pronounce\_all.png")

stop\_ = tk.PhotoImage(file="icon/stop.png")

pronounce = tk.Menu(main\_menu, tearoff = False)

# Translator Menu Icon

trans\_icon = tk.PhotoImage(file="icon/pronounce.png")

trans = tk.Menu(main\_menu, tearoff = False)

# Theme Color

light\_theme = tk.PhotoImage(file="icon/light\_default.png")

light\_plus\_theme = tk.PhotoImage(file="icon/light\_plus.png")

dark\_theme = tk.PhotoImage(file="icon/dark.png")

red\_theme = tk.PhotoImage(file="icon/red.png")

monokai\_theme = tk.PhotoImage(file="icon/monokai.png")

night\_theme = tk.PhotoImage(file="icon/night\_blue.png")

color\_theme = tk.Menu(main\_menu, tearoff=False)

theme\_choose = tk.StringVar()

# System App Icon

calculator\_icon = tk.PhotoImage(file = "icon/calculator.png")

calendar\_icon = tk.PhotoImage(file = "icon/calendar.png")

chrome\_icon = tk.PhotoImage(file = "icon/chrome.png")

camera\_icon = tk.PhotoImage(file = "icon/camera.png")

photos\_icon = tk.PhotoImage(file = "icon/photos.png")

apps = tk.Menu(main\_menu, tearoff = False)

# System Apps Function

def calculator\_fun(event = None):

os.system("calc.exe")

apps.add\_command(label="Calculator", image=calculator\_icon, compound=tk.LEFT, command=calculator\_fun)

def calendar\_fun(event = None):

AppOpener.open("calendar")

#os.system("calendar.exe")

apps.add\_command(label="Calendar", image=calendar\_icon, compound=tk.LEFT, command=calendar\_fun)

def chrome\_fun(event = None):

AppOpener.open("google chrome")

apps.add\_command(label="Chrome", image=chrome\_icon, compound=tk.LEFT, command=chrome\_fun)

def camera\_fun(event = None):

AppOpener.open("Camera")

apps.add\_command(label="Camera", image=camera\_icon, compound=tk.LEFT, command=camera\_fun)

def photos\_fun(event = None):

AppOpener.open("Photos")

apps.add\_command(label="Photos", image=photos\_icon, compound=tk.LEFT, command=photos\_fun)

# Theme Menu

color\_icons = (light\_theme, light\_plus\_theme, dark\_theme, red\_theme, monokai\_theme, night\_theme)

color\_dict = {

'Light Default' : ('#000000',"#ffffff"),

'Light Plus' : ('#474747',"#e0e0e0"),

'Dark' : ('#c4c4c4',"#2d2d2d"),

'Red' : ('#2d2d2d',"#ffe8e8"),

'Monokai' : ('#d3b774',"#474747"),

'Night Blue' : ('#ededed',"#6b9dc2")

}

theme\_choose.set('Light Default')

main\_menu.add\_cascade(label="File", menu=file)

main\_menu.add\_cascade(label="Edit", menu=edit)

main\_menu.add\_cascade(label="View", menu=view)

main\_menu.add\_cascade(label="Pronounce", menu=pronounce)

main\_menu.add\_cascade(label="Translator", menu=trans)

main\_menu.add\_cascade(label="Theme", menu=color\_theme)

main\_menu.add\_cascade(label="Apps", menu=apps)

tool\_bar\_label = ttk.Label(main\_application)

tool\_bar\_label.pack(side=tk.TOP,fill=tk.X)

# Font Style Box

font\_tuple = font.families()

font\_family = tk.StringVar()

font\_box = ttk.Combobox(tool\_bar\_label,width=30,textvariable=font\_family,state="readonly")

font\_box["values"] = font\_tuple

font\_box.current(font\_tuple.index("Arial"))

font\_box.grid(row=0,column=0,padx=5)

# Font Size Box

size\_variable = tk.IntVar()

font\_size = ttk.Combobox(tool\_bar\_label,width=20,textvariable=size\_variable,state="readonly")

font\_size["values"] = tuple(range(8,100,2))

font\_size.current(4)

font\_size.grid(row=0,column=1,padx=5)

# Bold Button

bold\_icon = tk.PhotoImage(file="icon/bold.png")

bold\_btn = ttk.Button(tool\_bar\_label,image=bold\_icon)

bold\_btn.grid(row=0,column=2,padx=5)

# Italic Button

italic\_icon = tk.PhotoImage(file="icon/italic.png")

italic\_btn = ttk.Button(tool\_bar\_label,image=italic\_icon)

italic\_btn.grid(row=0,column=3,padx=5)

# UnderLine Button

underline\_icon = tk.PhotoImage(file="icon/underline.png")

underline\_btn = ttk.Button(tool\_bar\_label,image=underline\_icon)

underline\_btn.grid(row=0,column=4,padx=5)

# Font Color Button

font\_color\_icon = tk.PhotoImage(file="icon/font\_color.png")

font\_color\_btn = ttk.Button(tool\_bar\_label,image=font\_color\_icon)

font\_color\_btn.grid(row=0,column=5,padx=5)

# Align Left

align\_left\_icon = tk.PhotoImage(file="icon/align-left.png")

align\_left\_btn = ttk.Button(tool\_bar\_label,image=align\_left\_icon)

align\_left\_btn.grid(row=0,column=6,padx=5)

# Align Center

align\_center\_icon = tk.PhotoImage(file="icon/align-center.png")

align\_center\_btn = ttk.Button(tool\_bar\_label,image=align\_center\_icon)

align\_center\_btn.grid(row=0,column=7,padx=5)

# Align Left

align\_right\_icon = tk.PhotoImage(file="icon/align-right.png")

align\_right\_btn = ttk.Button(tool\_bar\_label,image=align\_right\_icon)

align\_right\_btn.grid(row=0,column=8,padx=5)

# Text Editor

text\_editor = tk.Text(main\_application, undo=True)

text\_editor.config(wrap="word", relief=tk.FLAT)

scroll\_bar = tk.Scrollbar(main\_application)

text\_editor.focus\_set()

scroll\_bar.pack(side=tk.RIGHT,fill=tk.Y)

text\_editor.pack(fill=tk.BOTH,expand=True)

scroll\_bar.config(command=text\_editor.yview)

text\_editor.config(yscrollcommand=scroll\_bar.set)

# Font Family and Function

font\_now = "Arial"

font\_size\_now = 12

def change\_font(main\_application):

global font\_now

font\_now = font\_family.get()

text\_editor.configure(font=(font\_now,font\_size\_now))

def change\_size(main\_application):

global font\_size\_now

font\_size\_now = size\_variable.get()

text\_editor.configure(font=(font\_now,font\_size\_now))

font\_box.bind("<<ComboboxSelected>>",change\_font)

font\_size.bind("<<ComboboxSelected>>",change\_size)

# Bold Function

# print(tk.font.Font(font=text\_editor["font"]).actual())

def bold\_fun():

text\_get = font.Font(font=text\_editor["font"])

if text\_get.actual()["weight"] == 'normal':

text\_editor.configure(font=(font\_now,font\_size\_now,"bold"))

if text\_get.actual()["weight"] == 'bold':

text\_editor.configure(font=(font\_now,font\_size\_now,"normal"))

bold\_btn.configure(command=bold\_fun)

# Italic Function

def italic\_fun():

text\_get = font.Font(font=text\_editor["font"])

if text\_get.actual()["slant"] == 'roman':

text\_editor.configure(font=(font\_now,font\_size\_now,"italic"))

if text\_get.actual()["slant"] == 'italic':

text\_editor.configure(font=(font\_now,font\_size\_now,"roman"))

italic\_btn.configure(command=italic\_fun)

# UnderLine Function

def under\_line\_fun():

text\_get = font.Font(font=text\_editor["font"])

if text\_get.actual()["underline"] == 0:

text\_editor.configure(font=(font\_now,font\_size\_now,"underline"))

if text\_get.actual()["underline"] == 1:

text\_editor.configure(font=(font\_now,font\_size\_now,"normal"))

underline\_btn.configure(command=under\_line\_fun)

# Color Choose

def color\_choose():

color\_var = colorchooser.askcolor()

text\_editor.configure(fg=color\_var[1])

font\_color\_btn.configure(command=color\_choose)

# Align Left

def align\_left():

text\_get\_all = text\_editor.get(1.0,"end")

text\_editor.tag\_config("left",justify=tk.LEFT)

text\_editor.delete(1.0,tk.END)

text\_editor.insert(tk.INSERT,text\_get\_all,"left")

align\_left\_btn.configure(command=align\_left)

# Align Center

def align\_center():

text\_get\_all = text\_editor.get(1.0,"end")

text\_editor.tag\_config("center",justify=tk.CENTER)

text\_editor.delete(1.0,tk.END)

text\_editor.insert(tk.INSERT,text\_get\_all,"center")

align\_center\_btn.configure(command=align\_center)

# Align Right

def align\_right():

text\_get\_all = text\_editor.get(1.0,"end")

text\_editor.tag\_config("right",justify=tk.RIGHT)

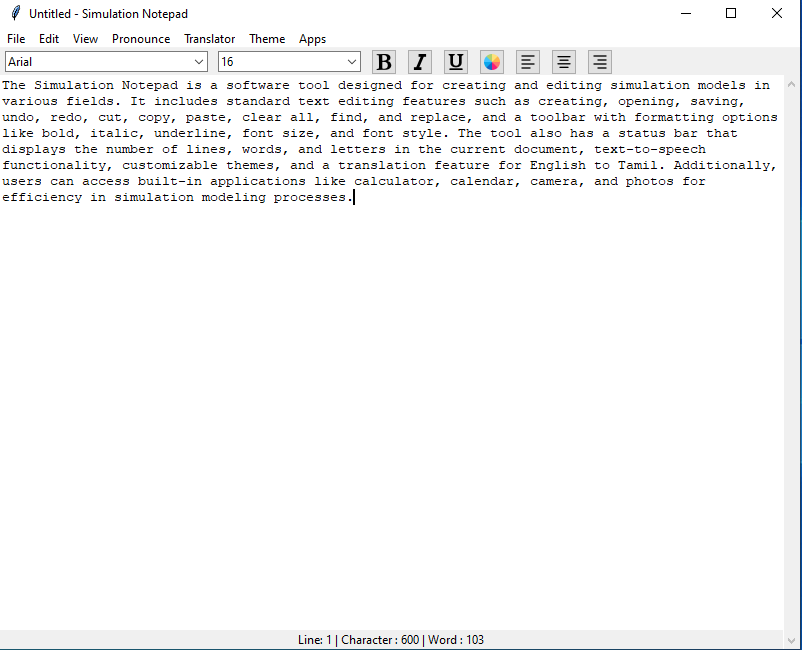
text\_editor.delete(1.0,tk.END)

text\_editor.insert(tk.INSERT,text\_get\_all,"right")

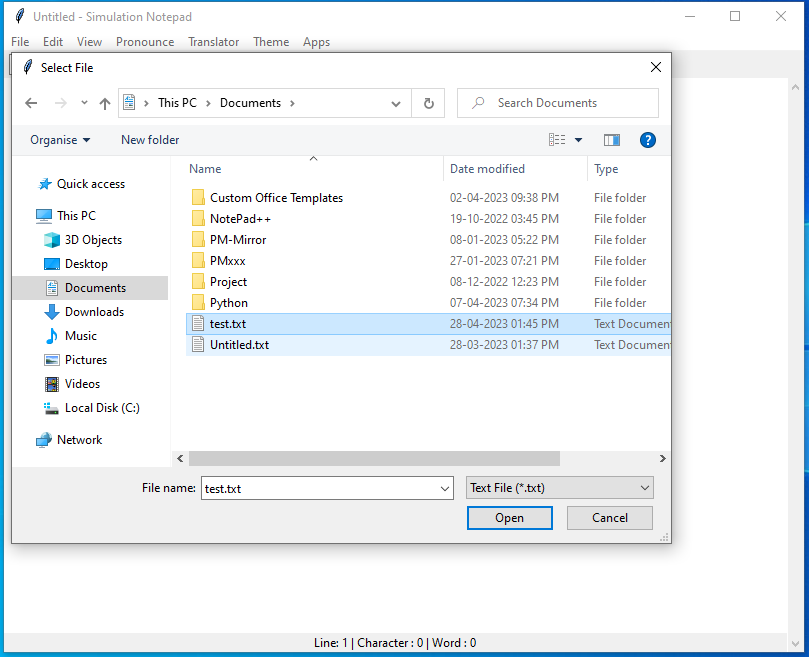
align\_right\_btn.configure(command=align\_right)

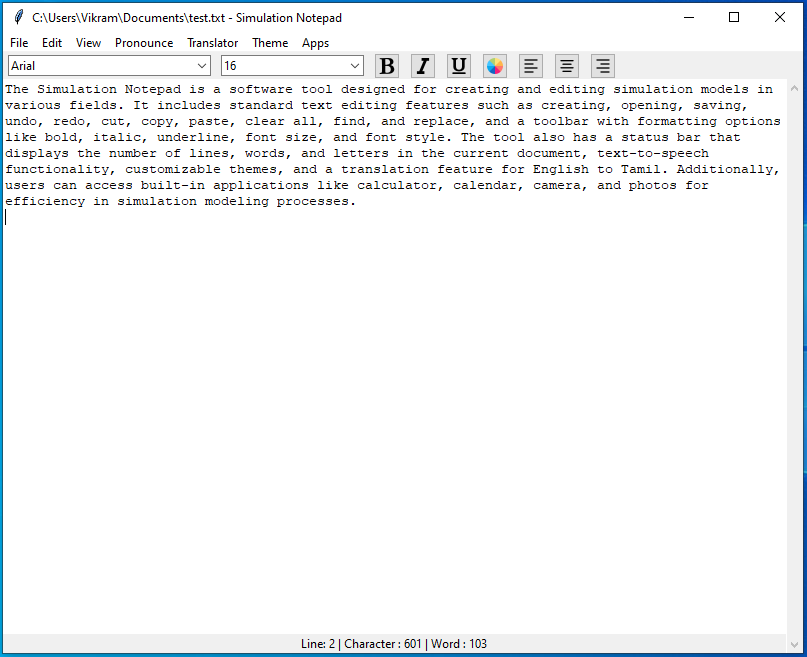
**SAMPLE INPUT AND OUTPUT**

1. **Text:**

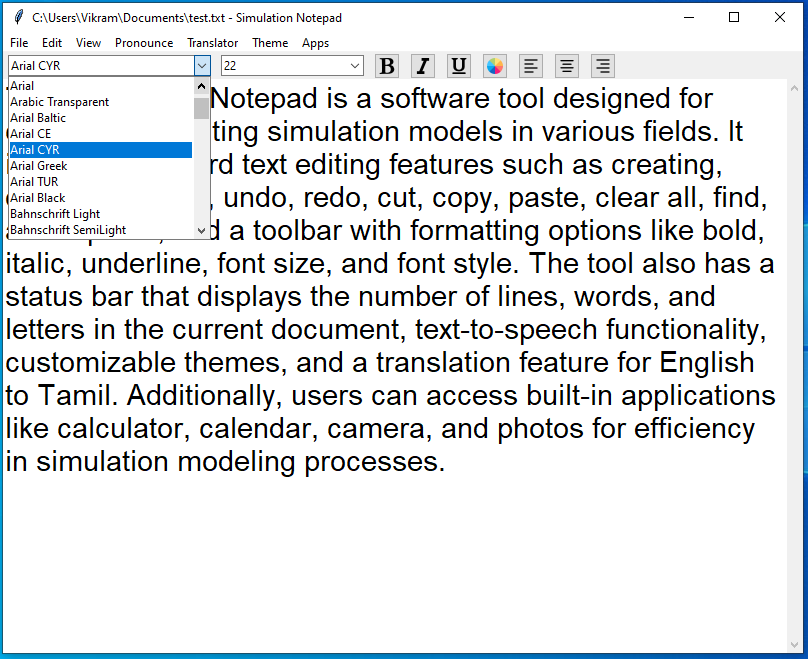


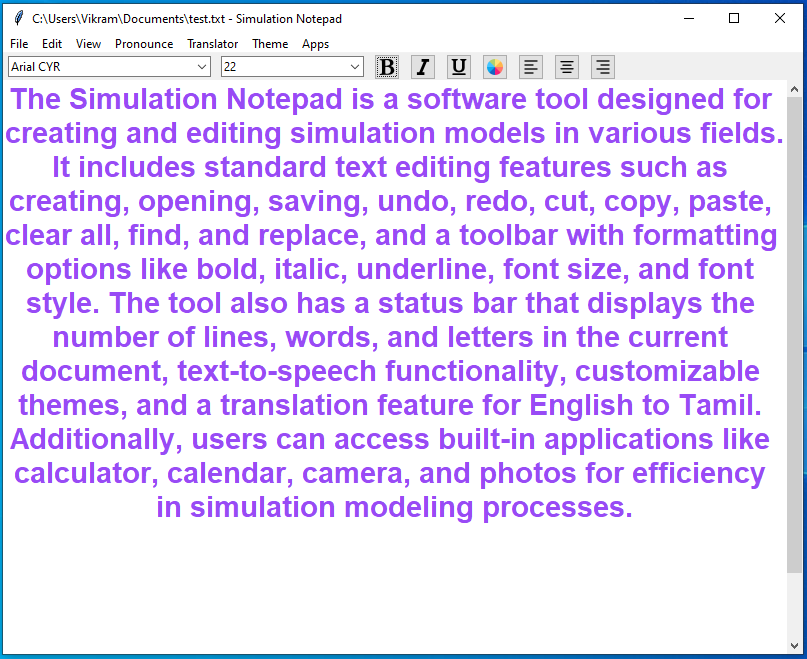
1. **File:**



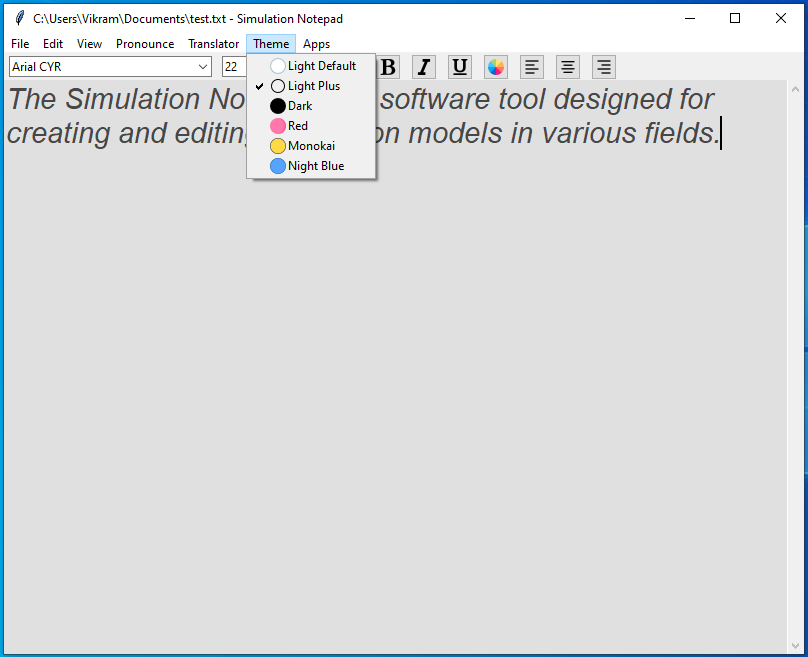


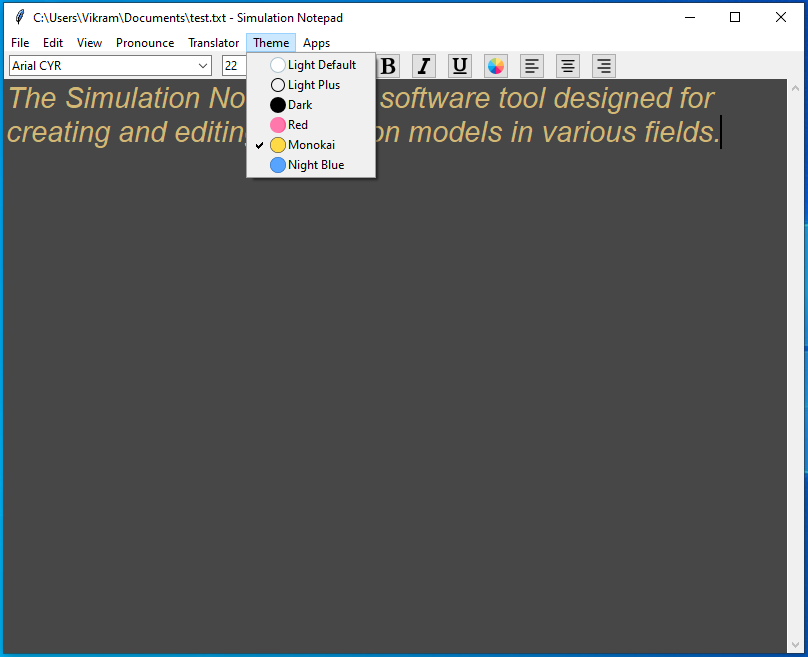
1. **Toolbar:**

****

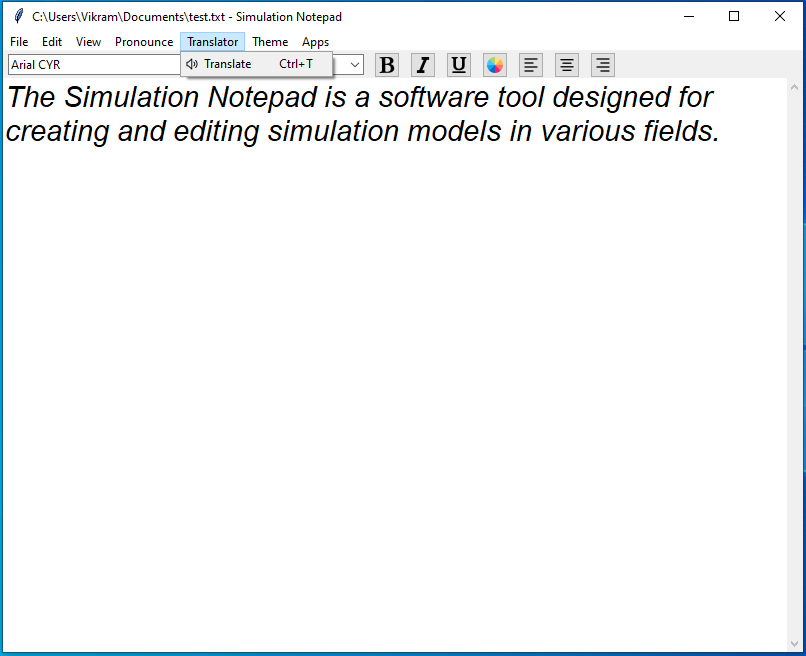
****

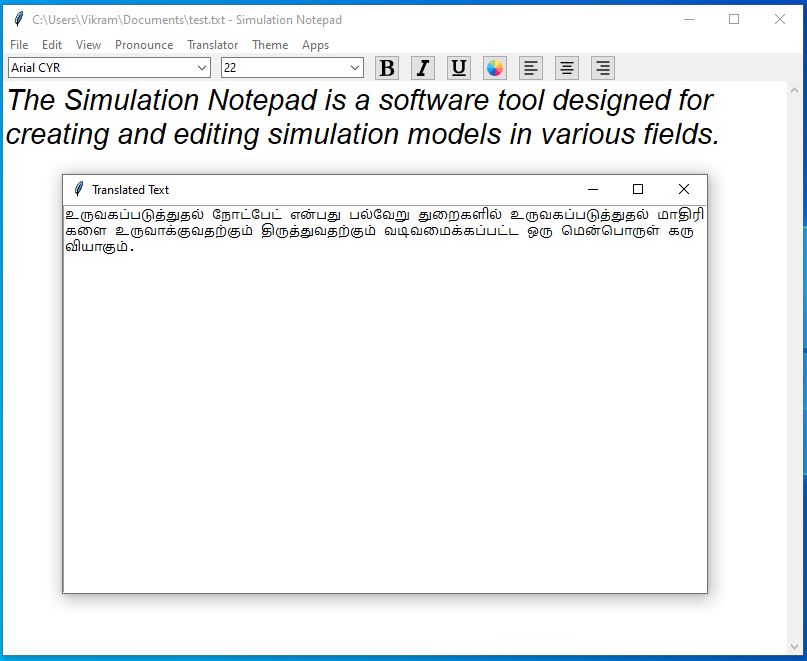
1. **Theme:**

****

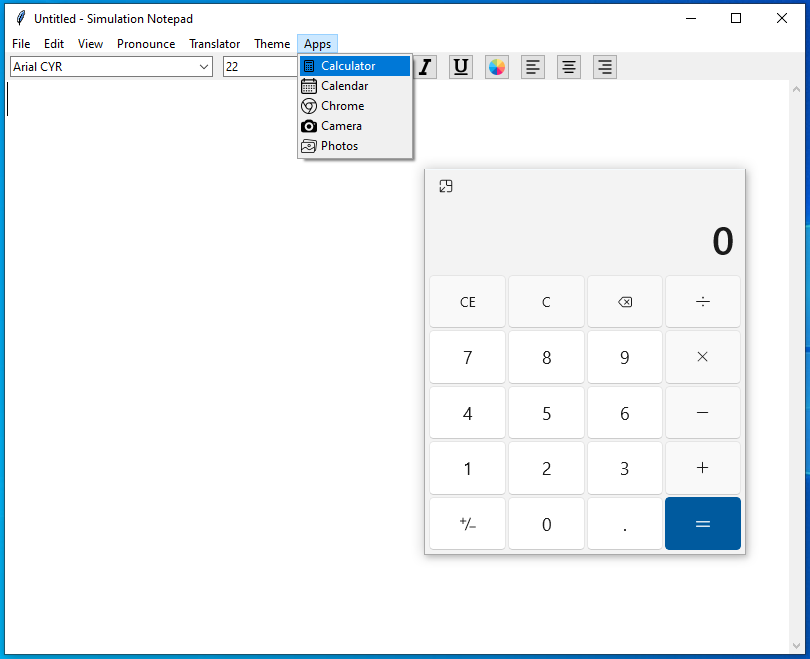
****

1. **Translation:**

****

****

1. **Apps:**

****