WORDS TRANSLATOR USING PYTHON

A PROJECT REPORT

Submitted For the Partial Fulfillment of the Requirement for the Degree of Master of Computer Applications

By

DHANANJAYAN E

A19101PCA6008

Under the Guidance of

Mr. S. Jawahar, MCA, M.Phil. (Ph.D.)
Assistant Professor, Department of Computer Applications,

INSTITUTE OF DISTANCE EDUCATION
UNIVERSITY OF MADRAS
CHENNAI - 600 005

JUNE-2022

BONAFIDE CERTIFICATE

This is to certify that the report entitled **WORD TRANSLATOR USING PYTHON** being submitted to the University of Madras, Chennai By **DHANANJAYAN E** (**A19101PCA6008**) for the Partial Fulfillment for the award of degree of M.C.A is a bonafide record of work carried out by him/her under my guidance and supervision

Name and Designation of the Guide

Mr. S.Jawahar, MCA, M.Phil. (Ph.D.)
Assistant Professor, Department of Computer Applications,

Co-Ordinator

Dr.S.SASIKALA, M.C.A,M.Phil.,Ph.D.
Associate Professor of Computer Science IDE, University ofMadras
Chepauk, Chennai-5

Date: 06.08.2022

Submitted for the Viva-Voce Examination held on 06.08.2022 at centre IDE, University of Madras.

Examiners

1. Name :

Signature:

2. Name :

Signature :

SPACE ZEE IT SERVICES

13, Ramakrishna Mutt Rd,
Venkatesa Agraharam, Mylapore,
Chennai, Tamil Nadu 600004

January 01, 2022

To

The Head of the Department,
Master of Computer Application,
University of Madras,
Chennai, Tamil Nadu 600005.

Dear Sir/Madam,

Sub: Main Project Acceptance letter.

This is to confirm and certify that "Dhananjayan E" (Reg No: A19101PCA6008) from University of Madras has been permitted to do a project in "Space Zee IT Services" on "WORDS TRANSLATOR USING PYTHON" from January -2022 to June -2022.

During this period the candidate is given permission to use the company's properties while also stick to our terms and conditions for which the candidate is supposed to sign a code of conduct agreement

Date: 06-30-2022
Place: Chennai

Space Zee IT Services
Chennai-600004

SPACE ZEE IT SERVICES

13, Ramakrishna Mutt Rd,
Venkatesa Agraharam, Mylapore,
Chennai, Tamil Nadu 600004

January 01, 2022

To

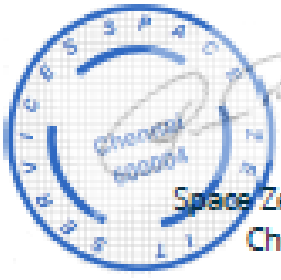
The Head of the Department,
Master of Computer Application,
University of Madras,
Chennai, Tamil Nadu 600005.

Dear Sir/Madam,

Sub: Main Project Acceptance letter.

This is to confirm and certify that **"Dhananjayan E" (Reg No: A19101PCA6008)** fr University of Madras has been successfully completed the project in "Space Zee Services" on **"WORDS TRANSLATOR USING PYTHON"** from **January - 2022 to Ju -2022.**

Date: 07-23-2022
Place: Chennai



Space Zee IT Services
Chennai-600004

TABLE OF CONTENT

ABSTRACT

In this project, a language will be chosen from a list of options in which the text is to be entered, and also the language in which the text is to be translated is also selected from the list of options. After selecting the languages, the translate button will be clicked to translate the text
For spreading new ideas, information and knowledge, the language translation is necessary. To effectively communicate between different cultures, language translation is important. This project helps in translating the text in other languages easily. Let's start developing the project of Language Translator in Python.

The objective of this Python project is to translate a piece of text into another language. You need to install, translate and import two modules: tkinter, translate. Basic knowledge of tkinter is required along with the knowledge of functions in python

This project requires knowledge of the tkinter module. Basic knowledge of functions in python is also required.

CHAPTER 1

INTRODUCTION

CHAPTER-1

INTRODUCTION

INTRODUCTION OF PROJECT

The objective of this Python project is to translate a piece of text into another language. You need to install, translate and import two modules: tkinter, translate. Basic knowledge of tkinter is required along with the knowledge of functions in python.

To master a language one should start creating projects on it and practically use the language. Here we are going to create yet another interesting project using Python. We are creating Python Language Translation project.

A language translator is a very handy application that helps us to communicate in different languages by translating our language to the desired language. In earlier times, when there were no Language Translation Applications, it was very difficult for people to communicate with people coming from different parts of the world. Today we can create our own language translation project using python

Our objective is to create a Language Translator which would help us translate a word, sentence or even a paragraph to another language. We will try to incorporate as many languages as possible. We will be using Tkinter Module to build our GUI for the project and googletrans library to present us with a number of languages that are a part of it. Also, we will use the TextBlob library for processing textual data.

Often we are faced with a problem where either we cannot understand a piece of the critical text, or we need to provide information to speakers of another language. Although online translation tools are available, it is not always possible to access them, or it may be better to provide a static translated page for our clients.

In this tutorial, we look at the different translation APIs available, how to use them in Python, and how we can use BeutifulSoup to translate text within HTML webpages or XML documents.

To do our translation we use the translate-api python library. This provides an effective interface to the many possible translation APIs that are available.

It is worth noting that more specific translators also support a professional field for their translations — this ensures that the translation matches those which have been derived from documents (a corpus) within the specified field.

Two examples are given below and provided using the professional field argument when specifying a query.

we have chosen a translation API we load our script and address it as follows.

```
import translators as ts
ts.<api_name>(..)                # e.g. ts.google
```

Selecting a Language

There are a number of languages between which we may translate. To check if the two we are interested in, we can select our translator and run the following command:

we can upload file with any formats and we upload document with required language file

ts._<translator_name>.language_map

This gives us a dictionary of keys for each included language, and values representing what they may be translated into.

In the case of Google.

If we decide to use the Google API for our translation, we can use the help function: help(ts.google) to decide on which arguments we require. The more common ones are listed below.

- query_text: this is the default string we are translating
- from_language: this defaults to ‘auto’ and uses the apo to ‘guess’ the language
- to_language: by default, all translations are to English (en)
- timeout: the amount of time we want to wait before it gives up. The default is None (not specified)
- sleep_seconds: If doing multiple requests it may be worth spacing these out to prevent flooding the server and getting them rejected.

OBJECTIVE

In this work, we translate the uploaded file into given language, we used French dictionary to translate from English to French

What needs to be done?

1. Read the input text file, find words list text file and dictionary csv file
2. Find all words that is in the find words list, that has a replacement word in the dictionary
3. Replace the words in the input text file
4. Save the processed file as output

What is the expected output?

1. Unique list of words that was replaced with French words from the dictionary
2. Number of times a word was replace
3. Time taken to process.

CHAPTER 2

SYSTEM ANALYSIS

CHAPTER-2

SYSTEM ANALYSIS

EXISTING SYSTEM

Machine translation is a process which uses neural network techniques to automatically translate text from one language to the another, with no human intervention required.

In today's machine learning tutorial, we will understand the architecture and learn how to train and build your own machine translation system. This project will help us automatically translate English to produce French sentence

DISADVANTAGES

1. Slow Speed

We discussed above that Python is an interpreted language and dynamically-typed language. The line by line execution of code often leads to slow execution.

The dynamic nature of Python is also responsible for the slow speed of Python because it has to do the extra work while executing code. So, Python is not used for purposes where speed is an important aspect of the project.

2. Not Memory Efficient

To provide simplicity to the developer, Python has to do a little tradeoff. The Python programming language uses a large amount of memory.

PROPOSED SYSTEM

CNRI proposes to undertake a research effort called **Computer Programming for Everybody** (CP4E). This effort intends to improve the state of the art of computer use, not by introducing new hardware, nor even (primarily) through new software, but simply by *empowering all users* to be computer programmers.

Recent developments in computer and communication hardware have given many people access to powerful computers, in the form of desktops, laptops, and embedded systems. It is time to give these users more control over their computers through education and supporting software. If users have a general understanding of computers at the level of software design and implementation, this will cause a massive surge in productivity and creativity, with a far-ranging impact that can barely be anticipated or imagined.

On a shorter term, the quantity and quality of available computer software will improve drastically, as the imagination and labor of millions is applied to the problem. Inventive users will be able to improve the software that supports them in their tasks, and share their improvements with their colleagues or--via the Internet--with others far away who are faced with the same tasks and problems. The ability to modify or customize software is important in crisis situations, when experts cannot be appealed to for help. It is also important for day-to-day activities:

The two major research goals are the development of a prototype of a new *programming curriculum* and matching prototype software comprising a highly user-friendly *programming environment.* We envision that the typical target audience will consist of high school and (non-CS major) undergraduate college students, although younger students and adults will also be considered. Course and software will normally be used together, so they should be tightly tuned to each other; each will also be usable on its own.

We will also explore the role of programming in the future. We are rapidly entering an age where information appliances, wearable computers, and deeply networked, embedded CPUs in everyday objects offer users control over their physical and information environments. End-user programmability will be the key to unlocking the potential of these technologies. (This is a common theme of DARPA's Information Technology Expeditions.)

The research effort will not be done in isolation. We will engage academic research groups as well as several leading high schools. We will also build a larger community by making our course materials and software freely available on the Internet.

We plan to start by basing both components on Python, a popular free interpreted object-oriented language [Python] [Lutz] [Watters]. Originally developed at CWI in Amsterdam, Python is currently being developed and maintained by CNRI. Python is extremely suitable for teaching purposes, without being a "toy" language: it is very popular with computer professionals as a rapid application development language.

Python combines elements from several major programming paradigms (procedural, functional and object-oriented) with an elegant syntax that is easy on the eyes and easy to learn and use. While we believe that Python is a good starting point, undoubtedly we will learn that improvements are possible. As part of our research, we will evaluate the effectiveness of Python for education and use by beginners, and design improvements or alternatives.

ADVANTAGES

1. Easy to Read, Learn and Write

Python is a **high-level programming language** that has English-like syntax. This makes it easier to read and understand the code. Python is really easy to **pick up** and **learn**, that is why a lot of people recommend Python to beginners. You need less lines of code to perform the same task as compared to other major languages like **C/C++** and **Java**.

2. Improved Productivity

Python is a very **productive language**. Due to the simplicity of Python, developers can focus on solving the problem. They don't need to spend too much time in understanding the **syntax** or **behavior** of the programming language. You write less code and get more things done.

3. Interpreted Language

Python is an interpreted language which means that Python directly executes the code line by line. In case of any error, it stops further execution and reports back the error which has occurred.

CHAPTER-3

REQUIREMENT AND SPECIFICATION

CHAPTER-3

REQUIREMENT

AND

SPECIFICATION

SOFTWARE SPECIFICATION

- Front End : PYTHON
- Tool : JUPYTER NOTEBOOK (ANACONDA3)

MODULES

- Creating modules (seqtools).
- Namespaces.
- Attributes and the dot operator.
- Glossary.
- Python with tkinter module

CHAPTER 4

MODULES

DESCRIPTION

CHAPTER 4

MODULES DESCRIPTION

Creating modules

All we need to create a module is a text file with a .py extension on the filename:

We can now use our module in both scripts and the Python shell. To do so, we must first import the module. There are two ways to do this:

1. f**rom seqtools import remove_at**
2. **import seqtools**

In the first example, remove_at is called just like the functions we have seen previously. In the second example the name of the module and a dot (.) are written before the function name.

Notice that in either case we do not include the .py file extension when importing. Python expects the file names of Python modules to end in .py, so the file extention is not included in the import statement.

The use of modules makes it possible to break up very large programs.

NAMESPACES

A namespace is a syntactic container which permits the same name to be used in different modules or functions (and as we will see soon, in classes and methods).

Each module determines its own namespace, so we can use the same name in multiple modules without causing an identification problem.

We can now import both modules and access question and answer in each:

If we had used from module1 import * and from module2 import * instead, we would have a naming collision and would not be able to access question and answer from module1.

ATTRIBUTES AND THE DOT OPERATOR

Variables defined inside a module are called attributes of the module. They are accessed by using the dot operator (.). The question attribute of module1 and module2 are accessed using module1.question and module2.question.

Modules contain functions as well as attributes, and the dot operator is used to access them in the same way. seqtools.remove_at refers to the remove_at function in the seqtools module.

In Chapter 7 we introduced the find function from the string module. The string module contains many other useful functions:

>>> import string>>> string.capitalize('maryland')'Maryland'**>>>** string.capwords("what's all this, then, amen?")"What's All This, Then, Amen?"**>>>** string.center('How to Center Text Using Python', 70)' How to Center Text Using Python '**>>>** string.upper('angola')'ANGOLA'>>>

You should use pydoc to browse the other functions and attributes in the string module.

Almost everything in Python is an object. Every object has certain attributes and methods. The connection between the attributes or the methods with the object is indicated by a “dot” (”.”) written between them.

If methods of the class are like eats(), runs(), sleeps() we can write Fido.eats(), Fiido.runs(), Fido.sleeps() and say Fido has attributes like Fido.size = tall, Fido.hair_color = brown

GLOSSARY

The default Python prompt of the interactive shell. Often seen for code examples which can be executed interactively in the interpreter.

Can refer to:

- The default Python prompt of the interactive shell when entering the code for an indented code block, when within a pair of matching left and right delimiters (parentheses, square brackets, curly braces or triple quotes), or after specifying a decorator.
- The Ellipsis built-in constant.

A tool that tries to convert Python 2.x code to Python 3.x code by handling most of the incompatibilities which can be detected by parsing the source and traversing the parse tree.

PYTHON WITH TKINTER

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps –

- Import the Tkinter module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user

CHAPTER-5

SOFTWARE DESCRIPTION

CHAPTER-5

SOFTWARE DESCRIPTION

Python Jupyter Notebook (anaconda3)

The Jupyter Notebook application allows you to create and edit documents that display the input and output of a Python or R language script. Once saved, you can share these files with others.

NOTE: Python and R language are included by default, but with customization, Notebook can run several other kernel environments.

This page provides a brief introduction to Jupyter Notebooks for AEN users.

For information on the notebook extensions available in AEN, see Using Jupyter Notebook extensions.

The Examples folder in Jupyter Notebook contains several types of Notebook examples created in Python—and one with R language—kernel environments.

Open any example notebook to experiment and see how it works.

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Python Jupyter Notebook (anaconda3) Environment

the kernel environment to create your new notebook in.

NOTE: Customizable Python and R Language kernel environments are automatically created for you during project creation.

- Your project's default conda env kernels are a cloned copy of the root environment. You can customize them and install and delete additional packages.
- Root environment is managed by your Administrator. You cannot make or save any changes to it.
- You can switch between Python, R language and any other custom kernels in the notebook as you work in your notebook. For more information, see Using the Synchronize Environments extension.

The Synchronize Environments extension allows you to apply a Python, R language or any other custom environment inside your current notebook session, without needing to start up several Notebook instances using each of the selected environments.The following extensions are available for use with AEN's Jupyter Notebook application:

- Synchronize Environments with Jupyter from the Kernel menu.
- Locking adds multi-user capability from the Lock button.
- Revision Control Mechanism (RCM) adds Status, Checkout and Commit buttons.
- Conda environment and package management tab.
- Conda notebook adds conda management inside Notebook from the Kernel > Conda Packages menu option.
- Anaconda Cloud integration from the Publish to cloud button.
- Notebook Present turns your notebook into a PowerPoint-style presentation.

- Conda environments list—export, clone or delete an environment in the action column, or create a new environment by clicking the plus + icon. Switch to an environment by clicking it; packages for that environment are displayed below in the installed packages list.
- Conda available packages list—for the selected environment in currently configured channels.

Contributors

Project Jupyter is an engaged and respectful community made up of people from all over the world. Your involvement helps us to further our mission and to create an open platform that serves a broad range of communities, from research and education, to journalism, industry and beyond.

Naturally, this implies diversity of ideas and perspectives on often complex problems. Disagreement and healthy discussion of conflicting viewpoints is welcome: the best solutions to hard problems rarely come from a single angle. But disagreement is not an excuse for aggression: humans tend to take disagreement personally and easily drift into behavior that ultimately degrades a community. This is particularly acute with online communication across language and cultural gaps, where many cues of human behavior are unavailable. We are outlining here a set of principles and processes to support a healthy community in the face of these challenges.

Fundamentally, we are committed to fostering a productive, harassment-free environment for everyone. Rather than considering this code an exhaustive list of things that you can’t do, take it in the spirit it is intended - a guide to make it easier to enrich all of us and the communities in which we participate.

Whether you are a new, returning, or current contributor to Project Jupyter's subprojects or IPython, we welcome you.

Project Jupyter has seen steady growth over the past several years, and it is wonderful to see the many ways people are using these projects. As a result of this rapid expansion, our project maintainers are balancing many requirements, needs, and resources. We ask contributors to take some time to become familiar with our contribution guides and spend some time learning about our project communication and workflow.

The Contributor Guides and individual project documentation offer guidance. If you have a question, please ask us. Community Resources provides information on our commonly used communication methods.

Jupyter has seen wonderful growth over the past decade. As we have grown, the projects now span multiple GitHub organizations. Jupyter projects may be found in these organizations:

- jupyter
- ipython
- jupyterhub
- jupyterlab
- jupyter-widgets
- jupyter-server
- jupyter-xeus

Jupyter Notebook

JupyterLab: A Next-Generation Notebook Interface:

JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.

Jupyter Notebook: The Classic Notebook Interface:

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience

Jupyter Notebook vs. Google Colab

Google Colab's major differentiator from Jupyter Notebook is that it is cloud-based and Jupyter is not. This means that if you work in Google Collab, you do not have to worry about downloading and installing anything to your hardware.

It also means that you can rest easy knowing that your work will autosave and backup to the cloud without you having to do anything.

Google Colab is great for people who need to work across multiple devices — such as one computer at home and one at work, or a laptop and a tablet — since it syncs seamlessly across devices.

History and License

Python was created in the early 1990s by Guido van Rossum at Stichting Mathematisch Centrum in the Netherlands as a successor of a language called ABC. Guido remains Python's principal author, although it includes many contributions from others.

In 1995, Guido continued his work on Python at the Corporation for National Research Initiatives in Reston, Virginia where he released several versions of the software.

In May 2000, Guido and the Python core development team moved to BeOpen.com to form the BeOpen PythonLabs team. In October of the same year, the PythonLabs team moved to Digital Creations. In 2001, the Python Software Foundation was formed, a non-profit organization created specifically to own Python-related Intellectual Property. Zope Corporation is a sponsoring member of the PSF.

All Python releases are Open Source. Historically, most, but not all, Python releases have also been GPL-compatible; the table below summarizes the various releases.

Note

GPL-compatible doesn't mean that we're distributing Python under the GPL. All Python licenses, unlike the GPL, let you distribute a modified version without making your changes open source. The GPL-compatible licenses make it possible to combine Python with other software that is released under the GPL; the others don't.

Packages

A python package is a collection of modules. Modules that are related to each other are mainly put in the same package. When a module from an external package is required in a program, that package can be imported and its modules can be put to use.

Any Python file, whose name is the module's name property without the .py extension, is a module.

A package is a directory of Python modules that contains an additional __init__.py file, which distinguishes a package from a directory that is supposed to contain multiple Python scripts. Packages can be nested to multiple depths if each corresponding directory contains its own __init__.py file. When you import a module or a package, the object created by Python is always of type module.

When you import a package, only the methods and the classes in the __init__.py file of that package are directly visible.

For example, let's take the datetime module, which has a submodule called date. When datetime is imported, it'll result in an error

The correct way to use the date module is shown below:

```
from datetime import date
print date.today()
```

Add-ins

An add-in is a customization, such as a collection of tools on a toolbar, that plugs into an ArcGIS Desktop application (that is, ArcMap, ArcCatalog, ArcGlobe, and ArcScene) to provide supplemental functionality for accomplishing custom tasks.

In ArcGIS 10, add-ins are authored using .NET or Java along with Extensible Markup Language (XML). The XML describes the customizations, while the .NET or Java classes provide the custom behavior. The ArcObjects software development kit (SDK) includes an Add-Ins Wizard that integrates with development environments—such as Eclipse, Microsoft Visual Studio, and the free Express Editions of Visual Studio—to simplify development.

ArcGIS 10.1 introduces Python to the list of languages for authoring Desktop add-ins, providing you with an easy solution to extend desktop functionality. To simplify the development of Python add-ins, you must download and use the Python Add-In Wizard to declare the type of customization. The wizard will generate all the required files necessary for the add-in to work.

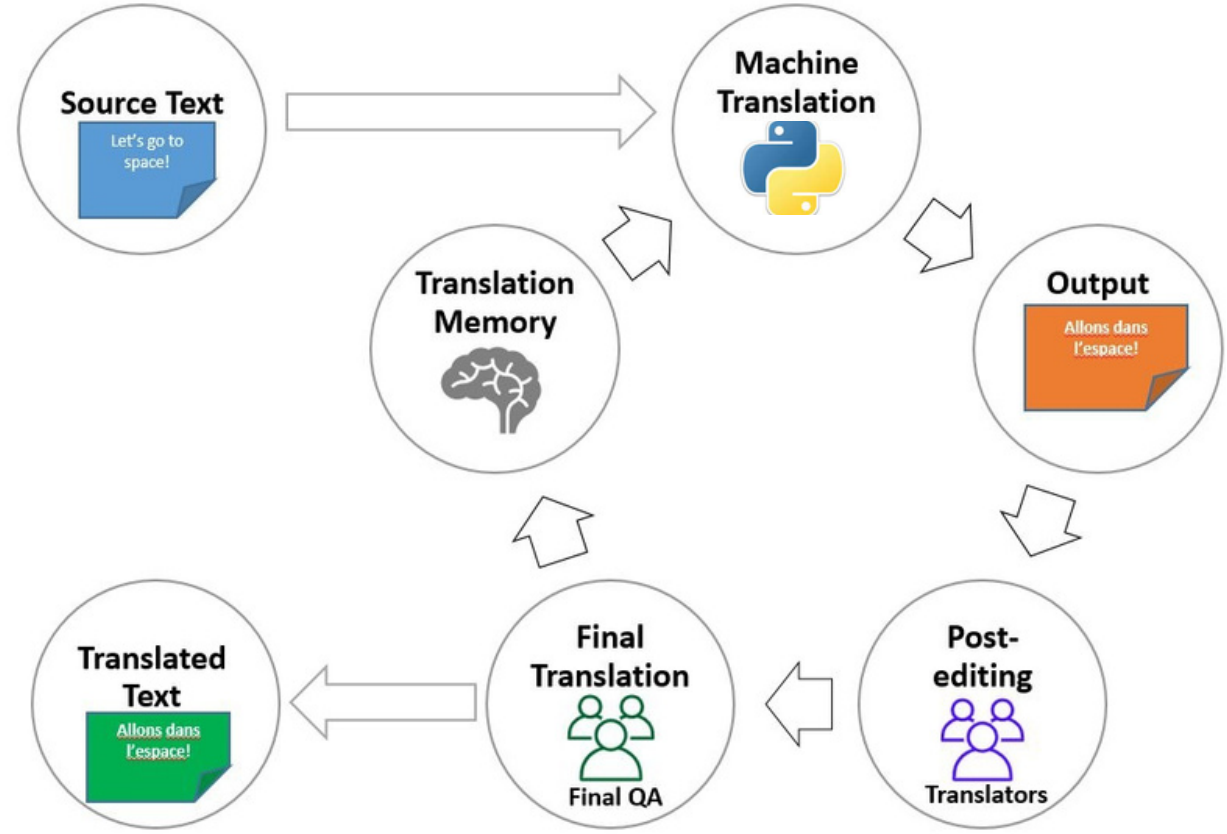
- **config.xml**—An Extensible Markup Language (XML) file defining the static add-in properties (for example, author, version, caption, category, and so on).
- **Python script**—The Python script (**.py** file) containing your business logic.
- **Resource files**—Items, such as images, and in some cases, data that is used to support your add-in.

CHAPTER-6

DIAGRAMS

CHAPTER-6

SYSTEM ARCHITECTURE



6.1.2 ARCHITECTURE EXPLANATION

The string translate() method returns a string where each character is mapped to its corresponding character in the translation table.

translate() method takes the translation table to replace/translate characters in the given string as per the mapping table.

The translation table is created by the static method maketrans().

The syntax of the translate() method is:

String translate() Parameters

translate() method takes a single parameter:

- table - a translation table containing the mapping between two characters; usually created by maketrans()

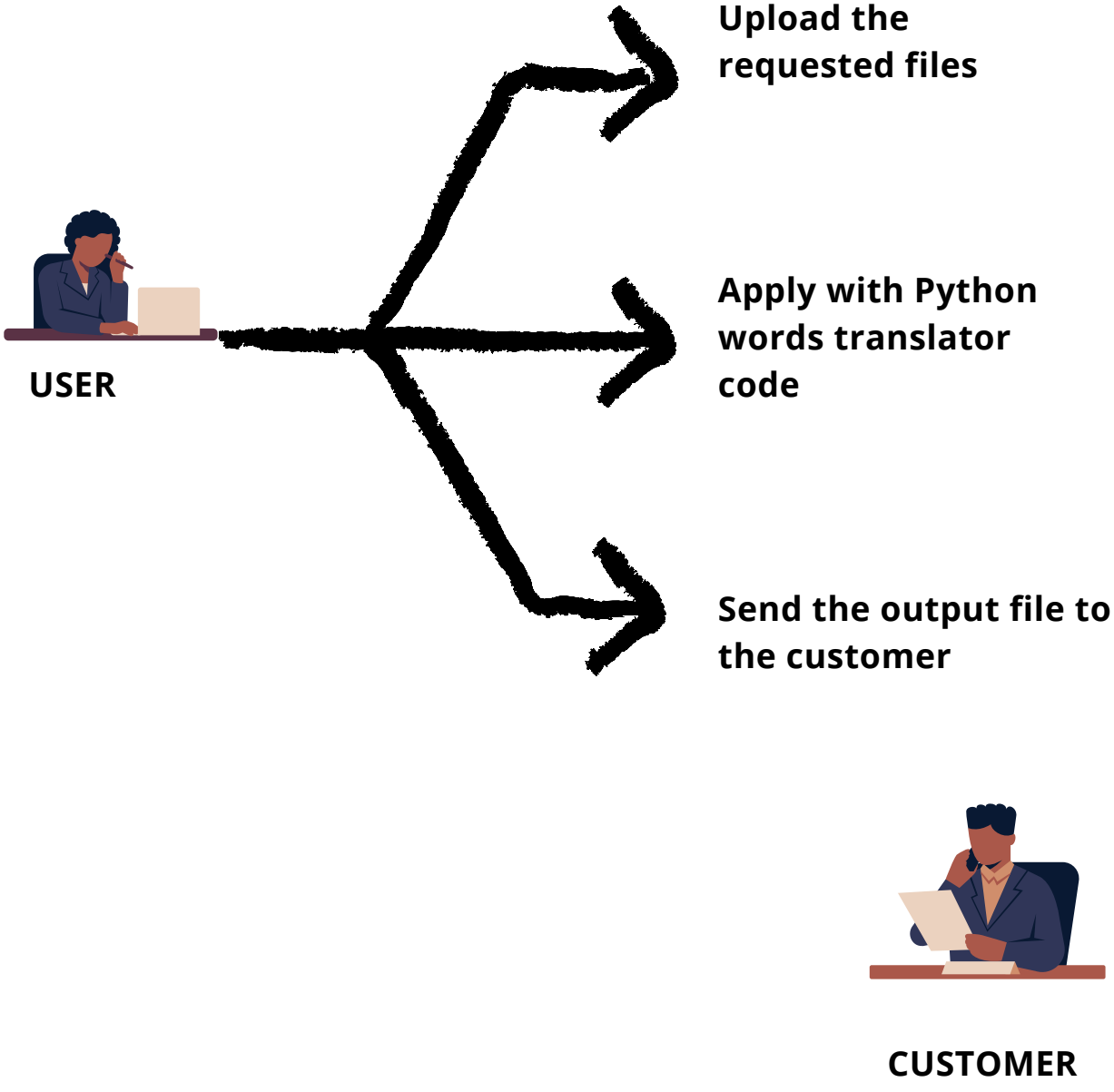
Return value from String translate()

translate() method returns a string where each character is mapped to its corresponding character as per the translation table.

we don't create a translation table from maketrans() but, we manually create the mapping dictionary translation.

This translation is then used to translate string to get the same output.

USECASE


Upload the requested files
Apply with Python words translator code
Send the output file to the customer
USER
CUSTOMER