Roorkee College of Engineering, Roorkee

A

PROJECT REPORT

ON

IRIS: Desktop Voice Assistant

Submitted in the partial fulfilment of the requirement for the award of degree of Bachelor of Technology in Computer Science & Engineering

Submitted To:

Mr Yudhveer Moudgil

Submitted By:

Anant Kumar
Siddharth Sharma

DECLARATION

We do hereby declare that the report entitled “IRIS-Desktop Voice Assistant” submitted by us to Roorkee College of Engineering, Roorkee in partial of the requirement for the award of the degree of B.TECH in COMPUTER SCIENCE AND ENGINEERING is a record of bona fide project work carried out by us under the guidance of Mr Yudhveer Moudgil and Department of Computer Science and Engineering.

Place: Roorkee

Siddharth Sharma

Anant Kumar

CERTIFICATE

This is to certify that the project entitled **"IRIS: Desktop Voice Assistant"** is a bona fide work done by Mr Siddharth Sharma, Mr Anant Kumar of 7th Semester B.Tech in Computer Science and Engineering from **Roorkee College of Engineering, Roorkee** under the guidance of Mr Yudhveer Moudgil in the partial fulfilment of the requirement of the award for the Degree of B.TECH in COMPUTER SCIENCE AND ENGINEERING in Roorkee College of Engineering, Roorkee.

Project Guide:

Mr Yudhveer Moudgil

Department Of Computer
Science and Engineering

(Head of the Department)

ACKNOWLEDGEMENT

We had a great experience working on this project and we got to learn a plethora of new skills through this project. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them. We are highly indebted to the teachers and especially **Mr Yudhveer Moudgil** for their guidance and constant supervision as well as providing necessary information regarding the project and also for their support in completing the project.

We would like to express our gratitude towards our parents and friends for their kind cooperation and encouragement which help us in the completion of the project.

Place - Roorkee　　　　Siddharth Sharma
Anant Kumar

ABSTRACT

The project aims to develop a personal-assistant for Linux-based systems. IRIS draws its inspiration from virtual assistants like Cortana for Windows, and Siri for iOS. It has been designed to provide a **user-friendly** interface for carrying out a variety of tasks by employing certain **well-defined commands**. Users can interact with the assistant either through **voice commands** or using **keyboard input**.

As a personal assistant, IRIS assists the end-user with *day-to-day* activities *like searching queries in google, searching topics in Wikipedia, live weather conditions, word meanings, scheduling messages and tasks, display news.*

CONTENT

PROBLEM STATEMENT

We are all well aware about Cortana, Siri, Google Assistant and many other virtual assistants which are designed to aid the tasks of users in Windows, Android and iOS platforms. But to our surprise, there's no such virtual assistant available for the paradise of Developers i.e. Linux platform.

PURPOSE

This Software aims at developing a personal assistant for Linux-based systems. The main purpose of the software is to perform the tasks of the user at certain commands, provided in either of the ways, speech or text. It will ease most of the work of the user as a complete task can be done on a single command. IRIS draws its inspiration from Virtual assistants like Cortana for Windows and Siri for iOS. Users can interact with the assistant either through voice commands or keyboard input.

PRODUCT GOALS AND OBJECTIVES

Currently, the project aims to provide the Linux Users with a Virtual Assistant that would not only aid in their daily routine tasks like searching the web, extracting weather data, vocabulary help and many others but also help in automation of various activities.

In the long run, we aim to develop a complete server assistant, by automating the entire server management process - deployment, backups, auto-scaling, logging, monitoring and make it smart enough to act as a replacement for a general server administrator.

PRODUCT DESCRIPTION

As a personal assistant, IRIS assists the end-user with day-to-day activities like searching queries in google, searching topics in Wikipedia, live weather conditions, word meanings, scheduling messages and tasks, display news.

SCOPE

Presently, IRIS is being developed as an automation tool and virtual assistant. Among the Various roles played by IRIS are:

1. Search Engine with voice interactions.
2. Reminder and To-Do application.
3. Vocabulary App to show meanings and correct spelling errors.
4. Weather Forecasting Application.

There shall be proper Documentation available on its Official Github repository for making further development easy and we aim to release our virtual assistant as an Open Source Software where modifications and contributions by the community are warmly welcomed.

TECHNOLOGIES USED

FRONTEND FRAMEWORK

- Tkinter (Python Library)

BACKEND STACK

- Python
- Docker Container

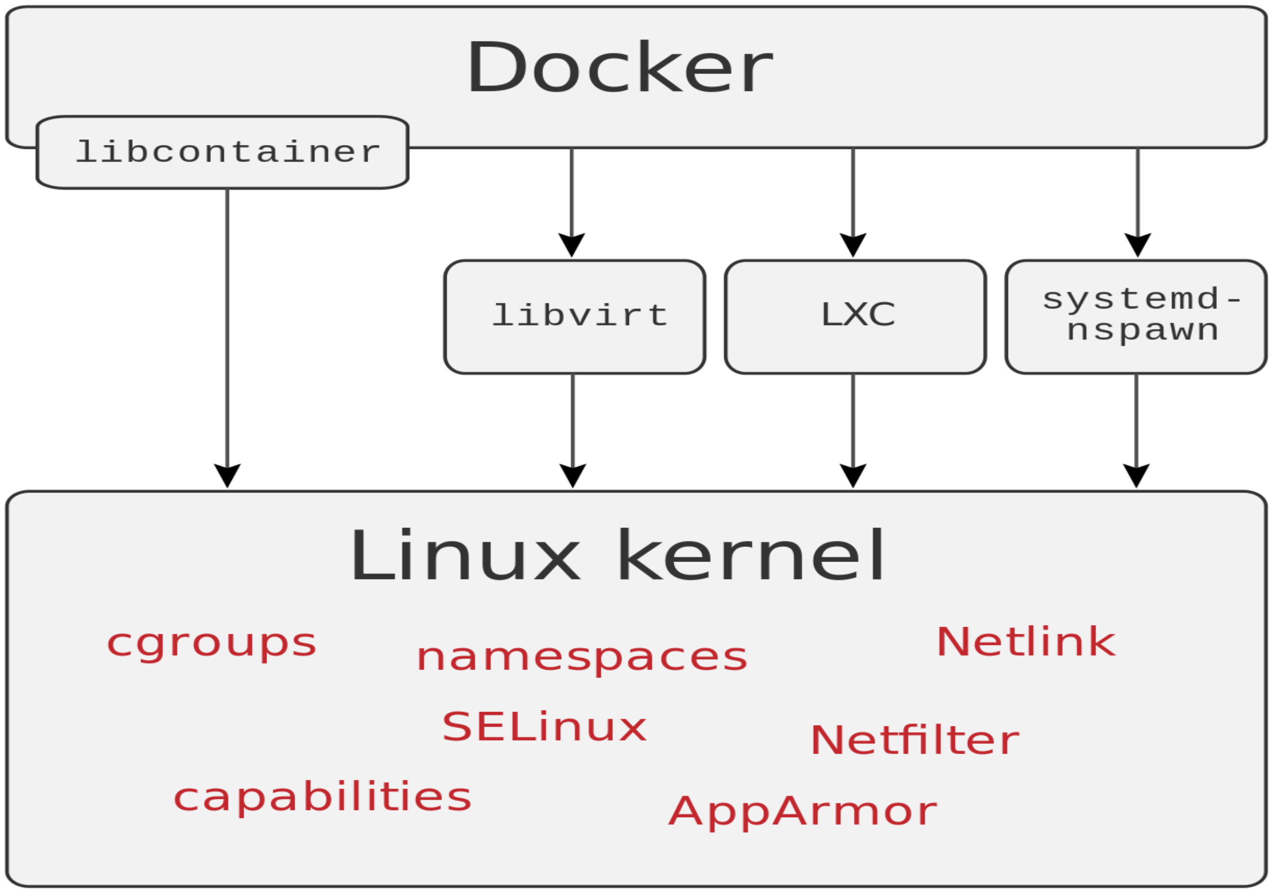
DATABASE

- SQLite

DOCKER CONTAINER

Docker is a computer program that performs operating-system-level virtualization. It is used to run software packages called containers. Containers are isolated from each other and bundle their own application, tools, libraries and configuration files; they can communicate with each other through well-defined channels. All containers are run by a single operating-system kernel and are thus more lightweight than virtual machines. Containers are created from images that specify their precise contents. Images are often created by combining and modifying standard images downloaded from public repositories.

Docker is developed primarily for Linux, where it uses the resource isolation features of the Linux kernel such as cgroups and kernel namespaces, and a union-capable file system such as OverlayFS and others to allow independent containers to run within a single Linux instance, avoiding the overhead of starting and maintaining virtual machines (VMs). The Linux kernel's support for namespaces mostly isolates an application's view of the operating environment, including process trees, network, user IDs and mounted file systems, while the kernel's cgroups provide resource limiting for memory and CPU.



Docker can use various interfaces to access virtualisation features of the kernel. A Docker container, unlike a virtual machine, does not require or include a separate operating system. Instead, it relies on the kernel's functionality and uses resource isolation for CPU and memory, and separate namespaces to isolate the application's view of the operating system. Docker accesses the Linux kernel's virtualization features either directly using the libcontainer library, which is available as of Docker 0.9, or indirectly via libvirt, LXC (Linux Containers).

COMPONENTS

The Docker software is a service consisting of three components:

- **Software:** The Docker daemon, called dockerd, is a persistent process that manages Docker containers and handles container objects. The daemon listens for requests sent via the Docker Engine API. The

Dockerinterfac e clientthat allows users to interact with Docker daemons. program, called docker, provides a command-lin e

- **Objects:** Docker objects are various entities used to assemble an application in Docker. The main classes of Docker objects are images, containers, and services.
    - A Docker container is a standardized, encapsulated environment that runs applications. A container is managed using the Docker API or CLI
    - A Docker image is a read-only template used to build containers. Images are used to store and ship applications.[34]
    - A Docker service allows containers to be scaled across multiple Docker daemons. The result is known as a *swarm*, a set of cooperating daemons that communicate through the Docker API.
- **Registries:** A Docker registry is a repository for Docker images. Docker clients connect to registries to download ("pull") images for use or upload ("push") images that they have built. Registries can be public or private. Two main public registries are Docker Hub and Docker Cloud. Docker Hub is the default registry where Docker looks for images. Docker registries also allow the creation of notifications based on events.

Tools

- **Docker Compose** is a tool for defining and running multi-container Docker applications. It uses YAML files to configure the application's services and performs the creation and start-up process of all the containers with a single command. The docker-compose CLI utility allows users to run commands on multiple containers at once, for example, building images, Commands scalin g containers, related to runningimage manipulation, containers that or were stopped, and more. user-interactive options, are not relevant in Docker Compose because they address one container. The **docker-compose.yml** file is used to define an application's services and includes various configuration options. For example, the build option defines configuration options suchdefault as Dockerthe Dockerfile command, path, and the more. command]The optionfirst public allows version one to of override Docker

  Compose (version 0.0.1) was released on December 21, 2013. The first production-ready version (1.0) was made available on October 16, 2014.
- **Docker Swarm** provides native clustering functionality for Docker containers, which turns a group of Docker engines into a single virtual Docker engine.] In Docker 1.12 and higher, Swarm mode is integrated with Docker Engine. The swarm CLI utility allows users to run Swarm containers, create discovery tokens, list nodes in the cluster, and more. The docker node CLI utility allows users to run various commands to manage nodes in a swarm, for example, listing the nodes in a swarm, updating nodes, and removing nodes from the swarm. Docker manages

swarms using the Raft Consensus Algorithm. According to Raft, for an update to be performed, the majority of Swarm nodes need to agree on the update.

INTEGRATION

Docker can be integrated into various infrastructure tools, including Amazon Web Services, Ansible CFEngine, Chef Google Cloud Platform, IBM Bluemi x, HPE Helion Stackato, Jelastic, Jenkins,Kubernetes, Microsoft Azure, OpenStack Nova, OpenSVC, Oracle Container Cloud Servic e, Puppet, ProGet, Salt,Vagrant, and VMware vSphere Integrated Containers.

The Cloud Foundry Diego project integrates Docker into the Cloud Foundry PaaS.

Nanobox uses Docker (natively and with VirtualBox) containers as a core part of its software development platform.

Red Hat's OpenShift PaaS integrates Docker with related projects (Kubernetes, Geard, Project Atomic and others) since v3 (June 2015).

The Apprenda PaaS integrates Docker containers in version 6.0 of its product.

Jelastic PaaS provides managed multi-tenant Docker containers with full compatibility to the native ecosystem.

The Tsuru PaaS integrates Docker containers in its product in 2013, the first PaaS to use Docker in a production environment.

SUB-PROCESSES/CHILD PROCESS

A subprocess is a process started by another program. There are two major procedures for creating a child process: the fork system call (preferred in Unix-like systems and the POSIX standard) and the spawn (preferred in the modern (NT) kernel of Microsoft Windows, as well as in some historical operating systems).

A child process inherits most of its attributes, such as file descriptors, from its parent. In Unix, a child process is typically created as a copy of the parent, using the fork system call. The child process can then overlay itself with a different program (using exec) as required.

Each process may create many child processes but will have at most one parent process; if a process does not have a parent this usually indicates that it was created directly by the kernel. In some systems, including Linux-based systems, the very first process (called init) is started by the kernel at booting time and never terminates (see Linux startup process); other parentless processes may be launched to carry out various daemon tasks in userspace. Another way for a process to end up without a parent is if its parent dies, leaving an orphan process; but in this case, it will shortly be adopted by init.

When a child process terminates, some information is returned to the parent process. When a child process terminates before the parent has called wait, the kernel retains some information about the process, such as its exit status, to enable its parent to call wait later. Because the child is still consuming system resources but not executing it is known as a zombie process.

Python

Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. The main reasons why we chose Python for this project are:

#1 Easy to code: Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in python language and anybody can learn python basics in a few hours or days. It is also a developer-friendly language.

#2 Free and Open Source : Python language is freely available at the official website and you can download it from the given download link below click on the Download Python keyword.

Since it is open-source, this means that source code is also available to the public. So you can download it as, use it as well as share it.

#3 High-level Language: Python is a high-level language. When we write programs in python, we do not need to remember the system architecture, nor do we need to manage the memory.

#4 Large Standard Library: Python has a large standard library which provides a rich set of module and functions so you do not have to write your own code for every single thing. There are many libraries present in python for such as regular expressions, unit-testing, web browsers, etc.

#5 Dynamically Typed Language: Python is a dynamically-typed language. That means the type (for example- int, double, long, etc.) for a variable is decided at run time not in advance because of this feature we don't need to specify the type of variable.

Features in IRIS

1. Queries from the web:

Making queries is an essential part of one’s life, and nothing changes even for a developer working on Linux. We have addressed the essential part of a netizen’s life by enabling our voice assistant to search the web. Here we have used Node JS and Selenium framework for extracting the result from the web as well as displaying it to the user. IRIS supports a plethora of search engines like Google, Bing and Yahoo and displays the result by scraping the searched queries.

In order to make queries from different search engines, the given format should be adopted:

<search engine name> <query>

IRIS supports Google, Bing and Yahoo, which should precede the desired query.

2. Accessing youtube videos

Videos have remained as a main source of entertainment, one of the most prioritized tasks of virtual assistants. They are equally important for entertainment as well as educational purposes as most teaching and research activities in present times are done through Youtube. This helps in making the learning process more practical and out of the four walls of the classroom.

IRIS implements the feature through a subprocess module which is handled by the main Golang service. This service initiates the subprocess for Node JS which serves the Selenium WebDriver, and scraps the searched YouTube query.

In order to access videos from youtube format is:

youtube *<video you want to search for>*

3. Get weather for a location

Getting live weather conditions about a place remains an important task of virtual assistants. It helps the user charter the course of their action. IRIS addresses this issue with the help of Python.

In order to access the live weather condition format is:

Weather *<city> <state/country>*

4. Retrieve images

Users could get images directly through the IRIS interface. This implementation is done using the Selenium WebDriver. The images are derived as iframes from the entire web code received from Google images. These are formatted according to use and displayed in a compact manner in the IRIS interface.

In order to retrieve image format is:

Image *<image you want to search>*

5. Dictionary meaning

One of the usages of the web is to find word meaning and its usage in our day to day life. Instead of going through the bulky books, our users can simply search for it using the voice assistant and get the meaning within a fraction of seconds.

For retrieving the meaning of a word format is,

meaning *<word>*

7. Set Reminders

One of the main features of a voice assistant is to set a reminder for the user accordingly. IRIS is no different when it comes to this. The user can set reminders to be notified about a task at a particular time. This will help users, especially developers to schedule their time and resources easily. All the user have to do is to input ***Set reminder*** to the assistant**.** A form will be displayed. Fill the form with the required details and click on **set reminder** button.

8. Sending Emails

Integrating mailing features to IRIS eases the job of mailing, which otherwise would have to be done by opening the concerned email address. With IRIS, you do not need to go for another tab to do one of the major task of your day to day affairs. The user can send emails to the desired receiver. He should input ***Send mail,*** after which a form will be displayed. Fill the form with the required details and click on the **send mail** button.

Why to use IRIS?

1. It fulfils the lack of a virtual assistant in Linux systems.
2. It has an easy to install and use interface.
3. It accepts inputs even through voice or keyboard.
4. It gives live weather updates.

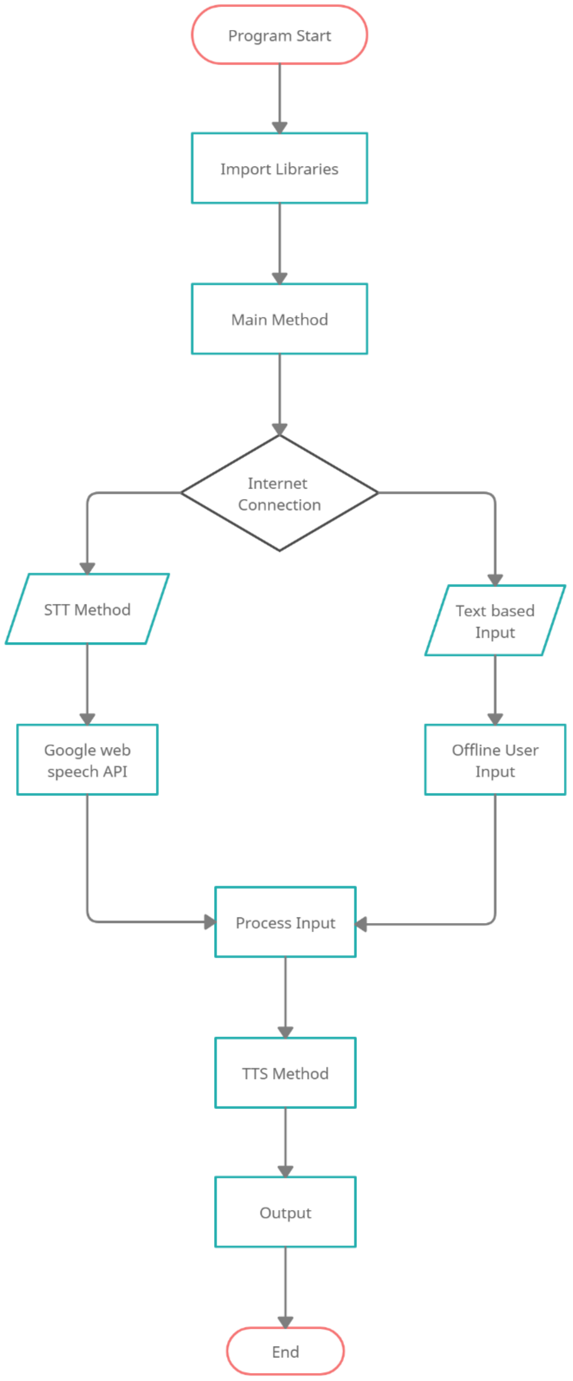
FUTURE PROSPECTIVE

Further, in the long run, IRIS is planned to feature auto deployment supporting elastic beanstalk, backup files, and all operations which a general Server Administrator does. The functionality would be seamless enough to replace the Server Administrator with IRIS.

System Requirements:

- Linux/ Windows
- Intel i3 or AMD equivalent
- 4 GB Ram
- 30 GB Memory

Data Flow Diagram:



Functional Requirements:

- Proper Internet Connection
- Python 3
- Pyttsx3 (Python Library)
- Chromium-based browser, like Chrome, Edge

Non-Functional Requirements:

The non-functional requirements of the system include:

- The system ensures safety, security and usability, which are observable during operation (at run time).
- The system is adaptable to different situations.
- The project is light on resources.

CONCLUSION

Through this voice assistant, we have automated various services using a single line command. It eases most of the tasks of the user like searching the web, retrieving weather forecast details, vocabulary help and medical related queries. We aim to make this project a complete server assistant and make it smart enough to act as a replacement for a general server administration. Further,in the long run, IRIS is planned to feature auto deployment supporting elastic beanstalk, backup files, and all operations which a general Server Administrator does. The functionality would be seamless enough to replace the Server Administrator with IRIS.