DITA: SIMPLE VOICE ASSISTANT

CHAPTER 14 : SPEECH RECOGNITION

By

Jessie Angelica	001202100035
Omega Voice Rambu Diki Rija	001202100137
Rizvany Aisyah Zulfikar Putri	001202100047
Yusi Roziana Oktaviani Tumanggor	001202100032

A Report Submitted

in Partial Fulfillment of the Requirements

for the Artificial Intelligence Class

Cikarang, Bekasi, Indonesia

TABLE OF CONTENTS

THEME

For this Artificial Intelligence Project, we chose Speech Recognition from Chapter 14 as the theme. Speech Recognition also known as automatic speech recognition (ASR) is a capability that enables a program to process human speech into a written format. This speech recognition focuses on the translation of speech from verbal format to a text in real-time.

Speech recognition is a type of biometric recognition, which is a computer process that identifies what someone is saying based on the intonation of the voice which is limited to digital print form. Speech Recognition allows a device to recognize and understand spoken words by digitizing the words and matching the digital signal with a certain pattern stored in a device.

With this technology, our lives are much easier and very helpful when we are in a state of urgency and have difficulty typing something.

PROBLEM

This real-time speech recognition and voice assistant technology have been popular and incredibly helpful in various life aspects. It is one of the most beneficial technologies of 2-way communication with an intelligent assistant. Many people find it hard to discover something, which takes a long time and not efficient. Moreover, some people are lazy to type on their devices when they are busy.

With the help of speech recognition, a digital assistant can understand and comprehend what the user or human is speaking and utilizes voice commands to access digital devices more efficiently and assist those who struggle with the latest information. Users can get the info without manually inputting any information into a machine. The program would allow users to input spoken words, then allocate the information search on the web to get the desired knowledge. This technology would make it easier for people to do something or gain any information and make business easier for those who struggle with many tasks. By leveraging speech recognition technology, we can make all things more accessible and efficient, ultimately making the job done faster.

For instance, when people need today's news, they don't have to worry; otherwise, say what news portal people want to go to, and it will let them access the information on the web. It can give information in real-time such as the day, date, and even the hour. The other example is the intelligent assistant employees to be more productive in the meeting room by telling him the results he wants, giving the receipt ideas for those people while cooking, and helping to search the location while people drive the car.

OBJECTIVES

- To generate and visualize various audio signals. It understands and reads an audio signal from a file and works with it through optimum AI/ML model training.
- To work with speech signals. The audio files are recorded using a microphone, and the system tries to understand the words that are being captured and stores the digitized versions.
- To recognize spoken words and transform audio signals to the frequency domain. The voice assistant takes audio signals as input and admits spoken words. It can receive spoken input and provide the user's needed information by directing it to Google. It can predict speech faster and more accurately.
- To successfully implement smart recognition that can recognize patterns in the system by using reinforcement learning from its mistakes and improve its performance. It enables speech recognition with different accents and dialects.
- To help advance the field of AI and solve the human struggle surrounding us. It makes everything accessible and efficient, ultimately creating more productivity and easier people work.

METHOD

Speech Recognition is programmed by Artificial narrow intelligence (ANI), which performs only singular tasks. In this section, to improve our project we use a data collection method. Because data collection is one of the most crucial stages of developing an AI/ML model. The execution of this stage determines the performance of the model when it is applied in the real world. One of the most effective methods of data collection is crowdsourcing, to help businesses learn more about the method in order to choose the most suitable path for an AI/ML project. Crowdsourcing is a large group of dispersed participants all over the world who produce any goods and services in exchange for compensation or as a voluntary job. In AI data collection, the "service" that the crowdsourced group provides is data collection.

Crowdsourcing different types of voice recordings is the first step to build an all-inclusive voice AI. This method exposes ML algorithms to different tones, genders, accents, and dialects. Over time, smart algorithms learn from these extensive data sets to better understand and respond to users' questions. Repeating questions when engaging with an automated system is frustrating and could potentially lead to abandonment. With crowdsourced voice recordings covering a wide range of accents, modal and non modal phonation, genders, and more, you can negate these types of situations. This approach goes a long way to deliver enhanced customer experiences by providing near-human-like conversations.

These almost human-like interactions encourage more engagement. ML algorithms will continue to train based on the data sourced and structured in workflows, including collecting, annotating, transcribing, and tagging voice recordings. Through different stages of validation, this technology will only keep getting better, ensuring accuracy and saliency. A voice assistant that learned from crowdsourced voice recordings will understand what you say (the first time we say it) even if we're not a native speaker. It works by collecting the speech data, transcribing it to text, validating it, and then annotating it to derive greater value from the data – for example, to help the voice assistant understand the user's intent.

AI will match the question or instruction with the appropriate response, and the dialogue will continue in this manner. Human-machine engagements where the speaker was pleased with the outcome are added to the voice dataset (and the intelligent algorithms will get better by learning from it).

Project Explanation

1. Project Source Code

   In our project, we created a speech assistant with Python called Dita: Simple Voice Assistant. Our program is inspired by one of the source codes on Youtube and GitHub which is Alexis-Speech-Assistant. We use methods and code similar to the source code and modify the code slightly according to the tools and devices we use. Here is the source code link :

   - Youtube : https://youtu.be/x8xjj6cR9Nc
   - GitHub : https://github.com/bradtraversy/alexis_speech_assistant

2. Project Prerequisites

   To implement a voice assistant in python requires us to have a basic knowledge of the python programming and speech_recognition library.

   - Install venv module - to create and manage virtual environments
   - speech_recognition – to recognize the speech
   - ctime from the time library – to get what time it is.
   - pyttsx3 – is a text-to-speech conversion library
   - webbrowser – is a convenient web browser controller to search for something
   - Random - used to generate random numbers between 0 and 1.

3. Importing Modules

We first import all the necessary Python libraries required for our voice assistant project in Visual Studio Code.

```
1   import speech_recognition as sr
2   import webbrowser
3   import time
4   import random
5   import pyttsx3
6   from time import ctime
```

4. Define functions

Create different functions which are going to be used in our project.

a) setName()

This function is used to set the user's name in the person's object.

```
class person:
    name = ''
    def setName(self, name):
        self.name = name
```

b) there_exists()

This function is used to set words or sentences that can trigger the AI to speak or do something.

```
13    def there_exists(terms):
14        for term in terms:
15            if term in voice_data:
16                return True
```

c) SR.Recognizer()

This function allows recognition of speech from the microphone. It makes easy to transcribe an audio file. It also shows us recognition results in an easy-to-understand format.

```
r = sr.Recognizer()
```

d) record_audio()

This function allows the program to get what people are saying by listening to audio and converting it to text.

```
def record_audio(ask = False):
    with sr.Microphone() as source: # microphone as source
        if ask:
            speak(ask)
        audio = r.listen(source) # listen for the audio via source
        voice_data = ''
        try:
            voice_data = r.recognize_google(audio) # convert audio to text
        except sr.UnknownValueError:   # error: recognizer does not understand
            speak('Sorry, I did not get that')
        except sr.RequestError:
            speak('Sorry, my speech service is down')  # error: recognizer is not connected
        return voice_data
```

e) speak()

This function is created to make an audio file to be played. Based on the GitHub source code, we use playsound to play the audio file. But when we run this code, the app doesn't play any sound.

```
def speak(audio_string):
    tts = gTTS(text=audio_string, lang='en') # text to speech(voice)
    r = random.randint(1,20000000)
    audio_file = 'audio' + str(r) + '.mp3'
    tts.save(audio_file) # save as mp3
    playsound.playsound(audio_file) # play the audio file
    print(f"kiri: {audio_string}") # print what app said
    os.remove(audio_file) # remove audio file
```

- First Problem

We use pyttsx3 for loading a speech engine driver implementation from the pyttsx3.drivers module. After construction, the program uses the engine object to register and unregister event callbacks; produce and stop speech; get and set speech engine properties; start and stop event loops.

```
engine = pyttsx3.init('sapi5')
voice_data = engine.getProperty('voices')
engine.setProperty('voice', voice_data[1].id)

def speak(audio):
    engine.say(audio)
    print(audio)
    engine.runAndWait()
```

To get this code, we used another source code from Youtube and collaborated it with our code so that the AI would make a sound.

Source : https://youtu.be/M4KeqY4nZEM

- Second Problem

We use speech rate control to control the speed of the program''s speech from very fast to normal.

```
engine.setProperty('rate', 130)
```

```
engine = pyttsx3.init('sapi5')
voice_data = engine.getProperty('voices')
engine.setProperty('rate', 130)
engine.setProperty('voice', voice_data[1].id)

def speak(audio):
    engine.say(audio)
    print(audio)
    engine.runAndWait()
```

f) respond()

This function is used to take the user instruction from the user and perform the user-given task. We'll also print the instructions which the user will provide. To run this function repeatedly, we're putting it in an indefinite while loop which only gets terminated when the user triggers it by saying "Thank you."

```
def respond(voice_data):
    # 1: greeting
    if there_exists(['hey','hi','hello']):
        greetings = [f"how can I help you {person_obj.name}", f"what's up? {person_obj.name}",
                     f"I'm listening {person_obj.name}", f"how can I help you? {person_obj.name}",
                     f"hello {person_obj.name}"]
        greet = greetings[random.randint(0,len(greetings)-1)]
        speak(greet)

    # 2: name
    if there_exists(["what is your name","what's your name","tell me your name"]):
        if person_obj.name:
            speak("my name is Dita")
        else:
            speak("my name is Dita. what's your name?")

    if there_exists(["my name is"]):
        person_name = voice_data.split("is")[-1].strip()
        speak(f"okay, i will remember that {person_name}")
        person_obj.setName(person_name) # remember name in person object

    # 3: greeting
    if there_exists(["how are you","how are you doing"]):
        speak(f"I'm very well, thanks for asking {person_obj.name}")
```

```
# 4: time
if there_exists(["what's the time","tell me the time","what time is it"]):
    time = ctime().split(" ")[3].split(":")[0:2]
    if time[0] == "00":
        hours = '12'
    else:
        hours = time[0]
    minutes = time[1]
    time = f'{hours} {minutes}'
    speak(time)

# 5: search google
if there_exists(["search for"]):
    search = voice_data.split("for")[-1]
    url = f"https://www.google.com/search?q={search}"
    webbrowser.get().open(url)
    speak(f'Here is what I found for {search} on google')

# 6: search for location
if  there_exists(["find location"]):
    location = record_audio('What is the location')
    url = 'https://www.google.com/maps/place/' + location + '/&amp;'
    webbrowser.get().open(url)
    speak('Here is the location of ' + location)

if 'thank you' in voice_data:
    speak('You are welcome')
    exit()
```

g) time.sleep()

This function is used to add delay in the execution of a program. We can use the python sleep function to halt the execution of the program for a given time in seconds.

```
time.sleep(1)
```

5. Create person object

   This object is an instance of a class name. It is a collection of attributes (variables) and methods. We use the person's object to perform actions.

```
person_obj = person()
```

6. Start program

   It is the default function that gets started when you run the program from the interface.

```
speak('Hello')
```

7. While loop

   In this while loop, the AI will keep running the program and recognizing the user's voice as long as the system is not stopped yet.

```
while 1:
    voice_data = record_audio()
    respond(voice_data)
```

EXPECTED RESULTS

In this section, we will discuss the predictions made and the result outcomes based on the project source code. The expected results will be presented along with the potential implications and limitations of the findings. The anticipated outcomes aim to provide a clear understanding of the project trial and their implications for future implementation and practice.

This AI Voice Assistant Speech Recognition project is expected to fulfill the objectives and solve the problems mentioned before. From the project code trial, the AI is expected to greet us, telling us about its name, telling us about the current clock time, searching for something on Google, and showing us a map of a certain location.

The AI is able to respond based on certain keywords written on the code shown in the previous section. It also shows the confidence level of the AI's speech recognition along with the transcripts of the several words recognized by the AI on the terminal output.

The implications of AI speech recognition voice assistants are vast and wide-ranging, and they are transforming the way we interact with technology. Here are some of the key implications we got from our findings:

1. Increased convenience: AI voice assistants make it easier to perform tasks and access information by simply using voice commands, which can save time and increase efficiency. In this project, we can ask for help simply by saying hello and saying the trigger words for the tasks, for example we want to ask about the current clock time by saying 'what time is it?' and the AI will give its answer for our request.

2. Improved accessibility: Voice assistants can improve accessibility for individuals with disabilities, such as visual or motor impairments, by providing an alternative means of interacting with technology.

Despite the numerous benefits of this AI speech recognition voice assistant project, there are also some limitations that we found. Here are some of the key limitations we got from our findings:

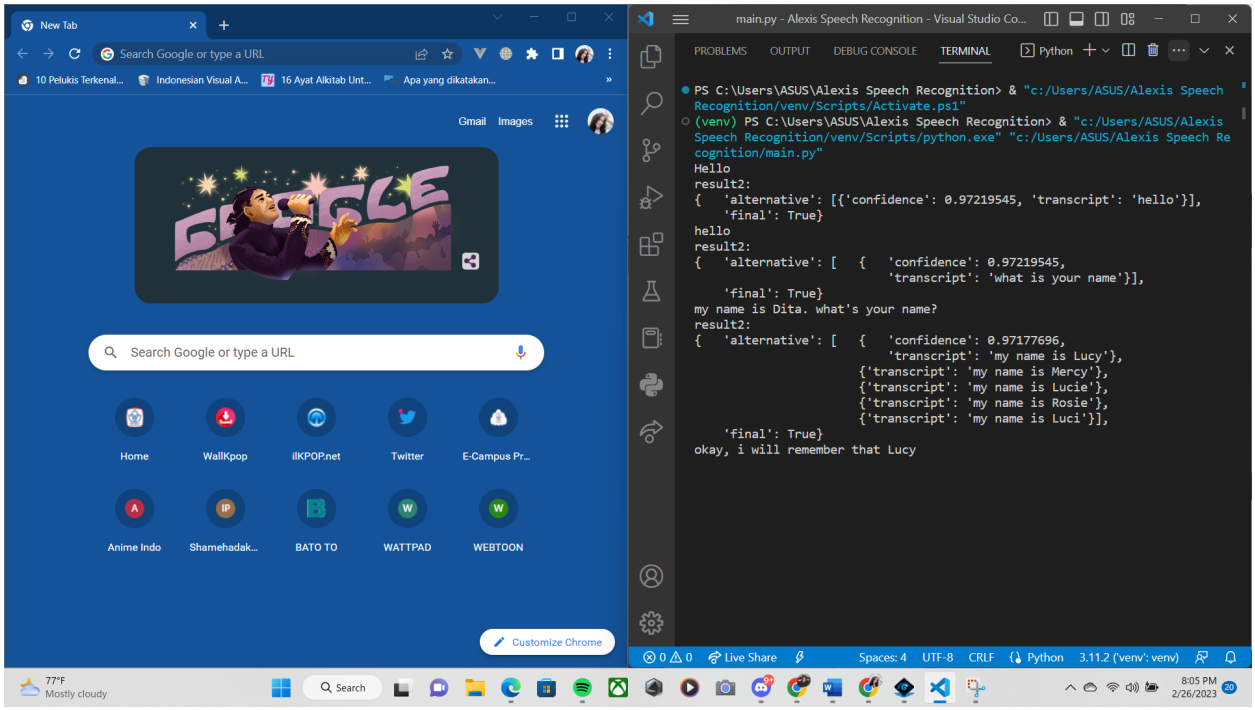
1. Accuracy: While AI speech recognition has made significant advancements in recent years, this project is still not 100% accurate, and users may experience frustration with misinterpretations or errors in transcription. The further confidence level rate of this AI only can go up to 97% accuracy.

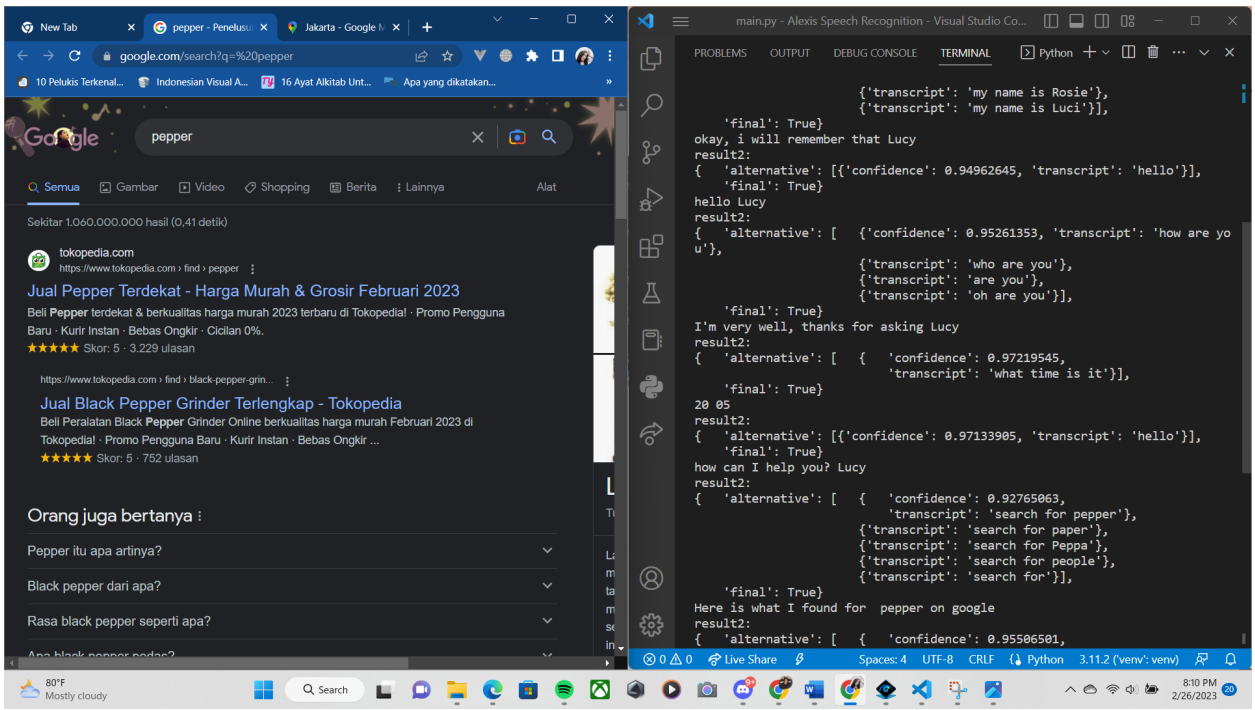
2. Language and dialect limitations: AI speech recognition voice assistants may struggle to accurately interpret regional dialects, accents, or languages that differ significantly from the language models they were trained on. In this project, we are using United States English as the AI basic language, so the result may be inaccurate if we are using words with other languages or other English accents.

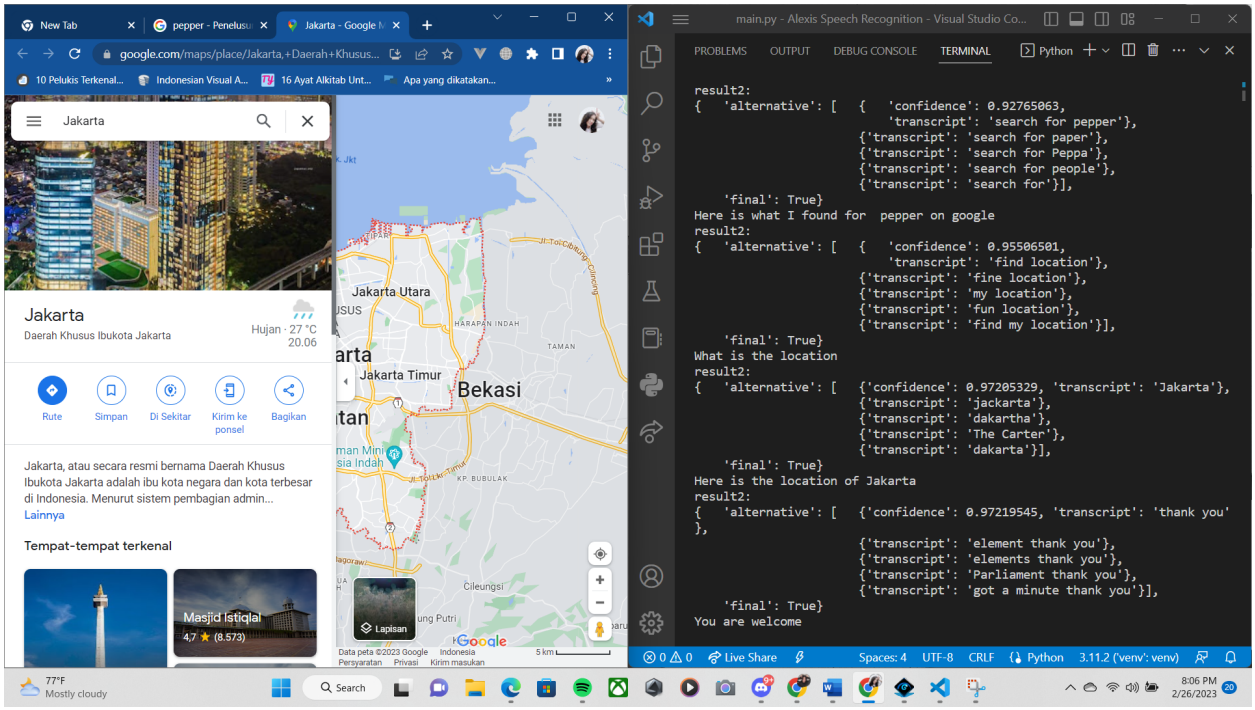
3. Limited flexibility: While voice assistants are capable of completing a wide range of tasks, they are limited by the specific commands and functions they have been programmed to perform. This AI project is programmed to greet us, telling us about its name, telling us about the current clock time, searching for something on Google, and showing us a map of a certain location. We are trying to collect more data information to expand the dataset as much as we can aim to.

4. Reliance on internet connectivity: AI speech recognition voice assistants typically require an internet connection to function, which can be problematic in areas with poor connectivity or limited access to the internet. This AI project is experiencing a delay in recognizing and giving responses to us when the connection is poor.

Results from the project source code trial:






New Tab
pepper - Penelusu
Jakarta - Google M
google.com/maps/place/Jakarta,+Daerah+Khusus...
10 Pelukis Terkenal...
Indonesian Visual A...
16 Ayat Alkitab Unt...
Apa yang dikatakan...
Jakarta
Jakarta
Daerah Khusus Ibukota Jakarta
Hujan · 27 °C
20.06
Rute
Simpan
Di Sekitar
Kirim ke ponsel
Bagikan
Jakarta, atau secara resmi bernama Daerah Khusus Ibukota Jakarta adalah ibu kota negara dan kota terbesar di Indonesia. Menurut sistem pembagian admin...
Lainnya
Tempat-tempat terkenal
Masjid Istiqlal
4,7 ★ (8.573)
Jakarta Utara
Jakarta Timur
Bekasi
Cileungsi
Lapisan
Data peta ©2023 Google
Indonesia
5 km
Persyaratan
Privasi
Kirim masukan
main.py - Alexis Speech Recognition - Visual Studio Co...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Python
result2:
{ 'alternative': [{ 'confidence': 0.92765063,
'transcript': 'search for pepper'},
{'transcript': 'search for paper'},
{'transcript': 'search for Peppa'},
{'transcript': 'search for people'},
{'transcript': 'search for'}],
'final': True}
Here is what I found for pepper on google
result2:
{ 'alternative': [{ 'confidence': 0.95506501,
'transcript': 'find location'},
{'transcript': 'fine location'},
{'transcript': 'my location'},
{'transcript': 'fun location'},
{'transcript': 'find my location'}],
'final': True}
What is the location
result2:
{ 'alternative': [{'confidence': 0.97205329, 'transcript': 'Jakarta'},
{'transcript': 'jackarta'},
{'transcript': 'dakartha'},
{'transcript': 'The Carter'},
{'transcript': 'dakarta'}],
'final': True}
Here is the location of Jakarta
result2:
{ 'alternative': [{'confidence': 0.97219545, 'transcript': 'thank you'
},
{'transcript': 'element thank you'},
{'transcript': 'elements thank you'},
{'transcript': 'Parliament thank you'},
{'transcript': 'got a minute thank you'}],
'final': True}
You are welcome
Live Share
Spaces: 4
UTF-8
CRLF
Python
3.11.2 ('venv': venv)
77°F
Mostly cloudy
Search
8:06 PM
2/26/2023

LOG HOURS

No	Student Name	Task Done and Log Hour Spent
1	Rizvany Aisyah	Expected Results (1 hour 40 minutes), Project Code Trial and Error (8 hours)
2	Jessie Angelica	Problem & Objective, Project Code Explanation (4 hours)
3	Omega Voice	Theme (1 hour 45 minutes)
4	Yusi Roziana Tumanggor	Method (3 hour 35 minutes), Project Code Trial and Error (8 hours)