

Stanisław Starzyński - CPNITS portfolio

student number: 5053811

outflow profile: WEB

submission date 22.01.2025

Table of contents:

Introduction.....	5
Week 1 - Functions & Variables.....	5
Assignment - Scratch game.....	5
CS50 - week 0.....	6
Indoor Voice.....	6
Playback Speed.....	7
Making Faces.....	7
Einstein.....	7
Tip Calculator.....	7
Evaluation.....	8
Week 2 - Conditionals & Loops.....	8
Assignment - Duck Pond Shootout.....	8
CS50 - week 1 & 2.....	13
Deep Thoughts.....	13
Home Federal Savings Bank.....	13
File Extensions.....	14
Math Interpreter.....	14
Meal Time.....	15
camel Case.....	15
Coke Machine.....	16
Just setting up my twttr.....	16
Vanity Plates.....	16
Nutrition Facts.....	18
Evaluation.....	18
Week 3 - Exceptions & Libraries.....	19
Assignment - none.....	19
CS50 - week 3 & 4.....	19
Fuel Gauge.....	19
Felipe's Taqueria.....	19
Grocery List.....	20
Outdated.....	21
Emojize.....	22
Frank, Ian and Glen's Letters.....	22
Adieu, Adieu.....	23
Guessing Game.....	23
Little Professor.....	24
Bitcoin Price Index.....	25
Evaluation.....	26
Week 4 - Unit Testing.....	26
Assignment - none.....	26
CS50 - week 5.....	26
Testing my twttr.....	26
Back to the Bank.....	27

Re-requesting a Vanity Plate.....	28
Refueling.....	29
Evaluation.....	31
Week 5 - File I/O, Regex.....	31
Assignment - none.....	31
CS50 - week 6&7.....	31
Lines of Code.....	31
Pizza Py.....	32
Scourgify.....	33
CS-50 P-shirt.....	33
NUMB3RS.....	34
Watch on Youtube.....	35
Working 9 to 5.....	36
Regular, um, Expressions.....	38
Response Validation.....	39
Evaluation.....	39
Week 6 - OOP.....	40
Assignments - Scavenger Hunt.....	40
Nerdy humor.....	40
What's your number?.....	41
Undutchify.....	42
Enigma.....	43
It's hip to be square.....	44
CS50 - week 8.....	45
Seasons of Love.....	45
Cookie Jar.....	46
CS50 Shirtificate.....	48
Evaluation.....	49
Week 7 - SQL.....	50
Assignment - none.....	50
CS50x - week 7.....	50
Songs.....	50
Movies.....	51
Evaluation.....	54
Week 8, 9 - Pico Project.....	55
Assignments - Pico.....	55
Onboard.....	55
Server.....	56
Evaluation.....	61
Week 1 - Web.....	61
Assignments - SQL, Murder Mystery, Custom Database.....	61
SQL.....	61
Murder Mystery.....	62
Custom Database and ERD.....	64

Evaluation.....	67
Week 2 - Web.....	68
Assignments - webpage.....	68
Evaluation.....	77
Week 3 - Web.....	78
Assignments - Grid and Flex.....	78
CSS Grid Garden and Flexbox Froggy.....	78
Layout.....	79
Evaluation.....	85
Week 4 - Web.....	85
Assignments - Flask.....	85
Vanity Plates.....	85
Evaluation.....	92
Week 7+ - Final Project.....	92
The idea.....	92
My website's structure.....	93
Link to the video:.....	121
Evaluation:.....	122
What Worked Out.....	122
What Didn't Work Out.....	122
What Could Be Improved in the Future.....	122
Final Project Reflection:.....	123
Final reflection.....	124
What Worked.....	125
What Didn't Work.....	125
Future Improvements.....	125
Conclusion.....	125

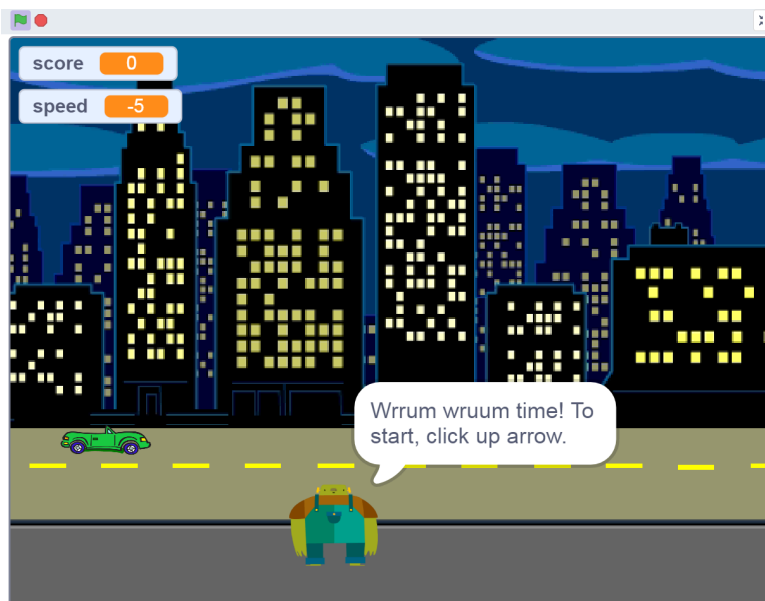
Introduction

This portfolio is a testament to the work I completed during the *Creative Programming for Non-IT Students* course. Through these pages, you will witness my progress and growth in the world of programming. Each week is accompanied by a personal reflection, offering insights into my learning process and helping better understand the journey. At the conclusion of this portfolio, you will find a summative reflection on the entire minor, detailing my overall experience and the outcomes of this endeavor.

Week 1 - Functions & Variables

Assignment - Scratch game

Link to the game: <https://scratch.mit.edu/projects/1063735976>





CS50 - week 0

Indoor Voice

```
text = input()
text = text.lower()
print(text)
```

<https://submit.cs50.io/check50/3f228273bfad2933d38f6e4ca9e81bd1258e16f8>

Playback Speed

```
txt = input()
txt = txt.replace(" ", "...")
print(txt)
```

<https://submit.cs50.io/check50/e2b4b6271a742852818949194dd73344433de11a>

Making Faces

```
def convert(conv):
    conv = conv.replace(":", "😊")
    conv = conv.replace(":", "😞")
    return conv
def main():
    function = convert(input())
    print(function)
main()
```

<https://submit.cs50.io/check50/d110e0aa64084dcf6f567d0fddb57c6fd68382bf>

Einstein

```
eq = int(input("m:"))
c = 300000000
E = eq * c ** 2
E = int(E)
print("E: ", E)
```

<https://submit.cs50.io/check50/edde129cf1d29e8f8177e844e5d771288c22552e>

Tip Calculator

```
def main():
    dollars = dollars_to_float(input("How much was the meal? "))
    percent = percent_to_float(input("What percentage would you like to tip? "))
    tip = dollars * percent
    print(f"Leave ${tip:.2f}")

def dollars_to_float(d):
    return float(d.replace("$", ""))
```

```
def percent_to_float(p):
    return float(p.replace("%", ""))/100

main()
```

<https://submit.cs50.io/users/stanislawstarzynski/cs50/problems/2022/python/tip>

Evaluation

About learning:

First week of CPNITS wasn't intense and I liked how it allowed me to even better understand what I'm going to do in the following weeks.

While it wasn't intense, I already got a chance to make a game of my own (which I really enjoyed). Having an easy to use tool at my hand made me more eager to try hard. The final game ended up being very simple but I was still happy with it because it was my first time programming.

On the other hand starting with python and CS50 was very overwhelming and even though topics covered were very simple I had to try hard to wrap my head around it.

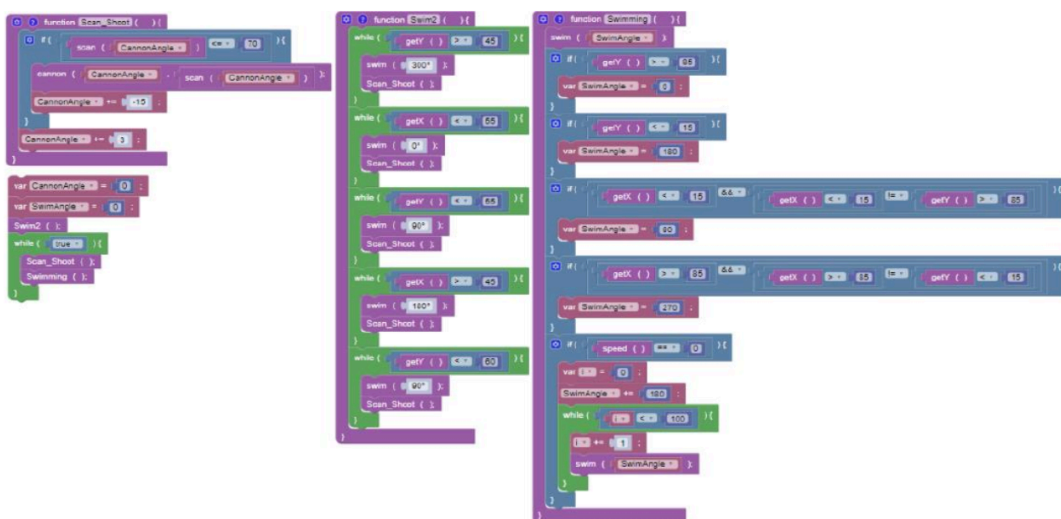
About relevance:

I find learning something from the very beginning very relevant to my course of study, because there are always new things one can learn to better their work.

Week 2 - Conditionals & Loops

Assignment - Duck Pond Shootout

Blocks Image:



Blockly Games : Pond



Player	Rook
Counter	Sniper

Documentation

▶ Run Program

Blockly Games : Pond



Player	Rook
Counter	Sniper

Documentation

✕ Reset

Blockly Games : Pond



Blockly Games : Pond



Blockly Games : Pond



Java Script (Copied):

```
var CannonAngle, SwimAngle, i;

// Describe this function...
function Scan_Shoot() {
  if (scan(CannonAngle) <= 70) {
    cannon(CannonAngle, scan(CannonAngle));
    CannonAngle += -15;
  }
  CannonAngle += 3;
}

// Describe this function...
function Swim2() {
  while (getY() > 45) {
    swim(300);
    Scan_Shoot();
  }
  while (getX() < 55) {
    swim(0);
    Scan_Shoot();
  }
}
```

```

while (getY() < 55) {
    swim(90);
    Scan_Shoot();
}
while (getX() > 45) {
    swim(180);
    Scan_Shoot();
}
while (getY() < 60) {
    swim(90);
    Scan_Shoot();
}
}

// Describe this function...
function Swimming() {
    swim(SwimAngle);
    if (getY() > 85) {
        SwimAngle = 0;
    }
    if (getY() < 15) {
        SwimAngle = 180;
    }
    if (getX() < 15 && getX() < 15 != getY() > 85) {
        SwimAngle = 90;
    }
    if (getX() > 85 && getX() > 85 != getY() < 15) {
        SwimAngle = 270;
    }
    if (speed() == 0) {
        i = 0;
        SwimAngle += 180;
        while (i < 100) {
            i += 1;
            swim(SwimAngle);
        }
    }
}

CannonAngle = 0;
SwimAngle = 0;
Swim2();
while (true) {
    Scan_Shoot();
    Swimming();
}

```

```
}
```

CS50 - week 1 & 2

Deep Thoughts

```
answer = input("What is the answer to the great question of life, the
universe, and everything else? ")
answer = answer.lower()
answer = answer.lstrip(" ")
answer = answer.rstrip(" ")
if answer == "42" or answer == "forty-two" or answer == "forty two":
    print("Yes")
else:
    print("No")
```

<https://submit.cs50.io/check50/8b07c3ec00576e7e78ccaeb7d3583a92b02e7a83>

Home Federal Savings Bank

```
import string
greetings = input("Greeting: ").strip().lower()
def main():
    if saidhello(greetings):
        print("$0")
    elif saidh(greetings):
        print("$20")
    else:
        print("$100")

def saidhello(hello):
    if greetings.startswith("hello"):
        return True

def saidh(h):
    if greetings.startswith("h"):
        return True

main()
```

<https://submit.cs50.io/check50/0698581c4e679b8751f3e59c7caceffb2c2bf3b2>

File Extensions

```
file_name = input("File name: ")
file_name = file_name.lower().strip()
if file_name.endswith(".gif"):
    print("image/gif")
elif file_name.endswith(".jpg"):
    print("image/jpeg")
elif file_name.endswith(".jpeg"):
    print("image/jpeg")
elif file_name.endswith(".png"):
    print("image/png")
elif file_name.endswith(".pdf"):
    print("application/pdf")
elif file_name.endswith(".txt"):
    print("text/plain")
elif file_name.endswith(".zip"):
    print("application/zip")
else:
    print("application/octet-stream")
```

<https://submit.cs50.io/check50/fa9cfd998d0afce508c8d560d310b91239915455>

Math Interpreter

```
equation = input("arithmetic expression: ")
xyz = equation.split()
if xyz[1] == "+":
    outcome = int(xyz[0]) + int(xyz[2])
    print(float(outcome))
elif xyz[1] == "-":
    outcome = int(xyz[0]) - int(xyz[2])
    print(float(outcome))
elif xyz[1] == "*":
    outcome = int(xyz[0]) * int(xyz[2])
    print(float(outcome))
elif xyz[1] == "/":
    xyz[2] != "0"
    outcome = int(xyz[0]) / int(xyz[2])
    print(float(outcome))
```

<https://submit.cs50.io/check50/054769ee942b4fc10dff3a6b0a85a93b58bdf8b>

Meal Time

```
def main():
    answer = input("What time is it? ")
    # input is always like ##:##
    float_time = convert(answer)
    # float_time between 7 and 8 = breakfast
    if float_time >= 7 and float_time <= 8:
        print("breakfast time")
    elif float_time >= 12 and float_time <= 13:
        print("lunch time")
    elif float_time >= 18 and float_time <= 19:
        print("dinner time")

def convert(time):
    # time is a string like ##:## or #:## and must convert to a float
    # 7:30 would be 7.5
    hour, minute = time.split(":")
    hour = float(hour)
    minute = float(minute)
    minute = minute/60.0
    return hour + minute

if __name__ == "__main__":
    main()
```

<https://submit.cs50.io/check50/88c5eb0fdd95cf21c0002792e93a07d60d75378c>

camel Case

```
# prompts the user for input of a variable in camelCase

camelCase = input("camelCase: ").strip()
snake_case = []

# Then returns the users in put as Snake case
for letter in camelCase:
    if letter.isupper():
        letter = "_" + letter.lower()
        camelCase = snake_case.append( letter)

else:
    camelCase = snake_case.append(letter)

print("".join(snake_case))
# Assume the users input will be CamelCase
```

<https://submit.cs50.io/check50/6a8c275e5ca1184283ebd24cf064a9606feaa6fa>

Coke Machine

```
amountdue = 50

while amountdue > 0:
    insertcoin = int(input(f"Amount Due: {amountdue}\nInsert Coin: "))
    if insertcoin == 5 or insertcoin == 10 or insertcoin == 25:
        amountdue = amountdue - insertcoin
    else:
        amountdue = int(amountdue)
change = -int(amountdue)

if amountdue <= 0:
    print(f"Change Owed: {change}")
```

<https://submit.cs50.io/check50/bd5c8e561f0dd9f199ceddf6968a3ac6d0682fd9>

Just setting up my twttr

```
text = input("Input: ")

print("Output: ", text.translate({ord(i): None for i in 'aeiouAEIOU'}))
```

<https://submit.cs50.io/check50/9d5c29dfac7400563f9bd8f32a2bae8025409d7f>

Vanity Plates

```
def main():

    plate = input("Plate: ")

    if is_valid(plate):
        print("Valid")
    else:
        print("Invalid")

def is_valid(s):
```



```

# input s is always uppercase
s = s.strip().upper()

# len(s) between 2 and 6 - TESTED
if len(s) < 2 or len(s) > 6:

    return False

# No periods, spaces, or punctuation marks are allowed. TESTED

if not s.isalnum():
    return False

letters = 0
numbers = 0

for c in s:
    if c.isdigit():
        # a number
        if letters < 2:
            # s must start with 2 or more letters
            return False

        if numbers == 0 and c == '0':
            # first number cannot be 0
            return False

        numbers += 1

    else:
        # a letter
        if numbers > 0:
            # numbers at the end
            return False

        letters += 1

return True

if __name__ == "__main__":
    main()

```

<https://submit.cs50.io/check50/9566b24f8c0094a2c1ff1acab680908080690874>

Nutrition Facts

```
fruit = input("Item: ").lower().replace(" ", "")

fruits = {
    "apple": 130,
    "avocado": 50,
    "banana": 110,
    "cantaloupe": 50,
    "grapefruit": 60,
    "grapes": 90,
    "honeydewmelon": 50,
    "kiwifruit": 90,
    "lemon": 15,
    "lime": 20,
    "nectarine": 60,
    "orange": 80,
    "peach": 60,
    "pear": 100,
    "pineapple": 50,
    "plums": 70,
    "strawberries": 50,
    "sweetcherries": 100,
    "tangerine": 50,
    "watermelon": 80
}

if fruits.get(fruit) is not None:
    print(fruits[fruit])
```

<https://submit.cs50.io/check50/0d82b8eb0456f7764a9e7dbbb8f3a33c92bcb8eb>

Evaluation

About learning:

Second week turned out to be much more intense. I got to program in a team of 3 and there were 2 weeks of CS50 to be done.

Working in a team is never easy but I believe I am pretty good at it. Mathijs (1 of my teammates) was very eager to work and already had some experience with programming. He was the person that actually came up with most of the code and while I was trying to help the best I could I was simply not catching up with his speed so I focused more on the tactics we could potentially use and trying to understand the logic behind the program we were making. Natalia, the other teammate of ours, wasn't that much present but she did communicate with us, so I chose to be understanding about it. In the end, I believe I learned from Mathijs a lot during those few hours of working together.

CS50 this week wasn't easy to me but the 1st week really helped me understand the logic behind it better.

About relevance:

I imagine myself in the future to frequently be working in a team. This experience taught me how to learn from my teammates and work more effectively within a team, especially when facing deadlines.

Week 3 - Exceptions & Libraries

Assignment - none

CS50 - week 3 & 4

Fuel Gauge

```
while True:
    try:
        fraction = input("Fraction: ")
        x, y = fraction.split("/")
        x = int(x)
        y = int(y)
        if x > y:
            continue
    except (ValueError, ZeroDivisionError):
        pass
    else:
        break
percent = int(round(x / y * 100))
if 1 < percent < 99:
    print(f"{percent}%")
elif percent >= 99:
    print("F")
elif percent <= 1:
    print("E")
```

<https://submit.cs50.io/check50/0b8c02df79081d3a2068c2e3c8340705b8f73461>

Felipe's Taqueria

```
menu = {
    "Baja Taco": 4.25,
    "Burrito": 7.50,
    "Bowl": 8.50,
    "Nachos": 11.00,
    "Quesadilla": 8.50,
```

```

    "Super Burrito": 8.50,
    "Super Quesadilla": 9.50,
    "Taco": 3.00,
    "Tortilla Salad": 8.00
}
total = 0
while True:
    try:
        item = input("Item: ").lower().title()
        total = float(total + menu[item])
        if total == False:
            pass
        else:
            print(f"Total: ${total:.2f}")
    except EOFError:
        break
    except KeyError:
        pass

```

<https://submit.cs50.io/check50/1382f7b57ecd200cb1209779872e935bef2d21cc>

Grocery List

```

grocery_list = {}

def list_add(grocery):
    if grocery in grocery_list:
        grocery_list[grocery] = grocery_list[grocery] + 1
    else:
        grocery_list.update({grocery : 1})

while True:
    try:
        grocery_input = input().strip().upper()
        list_add(grocery_input)

    except EOFError:
        break

try:
    for grocery in grocery_list:
        list_sorted = dict(sorted(grocery_list.items()))
    for grocery in list_sorted:
        print(list_sorted[grocery], grocery)
except NameError:
    print("")
except KeyError:

```

```
print("")
```

<https://submit.cs50.io/check50/22c77e78e8fe0a5e9cbb9ff59e67acbba3f69434>

Outdated

```
def main():
    date_to_convert = get_date("Date: ")
    print(date_to_convert)

def get_date(prompt):
    months = [
        "January",
        "February",
        "March",
        "April",
        "May",
        "June",
        "July",
        "August",
        "September",
        "October",
        "November",
        "December"
    ]
    while True:
        try:
            date = str(input(prompt)).strip()
            dateList = date.split("/")
            if len(dateList) == 3:
                month = dateList[0]
                if len(month) != 2:
                    month = "0" + month
                if int(month) <= 12:
                    day = dateList[1] if len(dateList[1]) == 2 else "0" +
dateList[1]
                    if int(day) <= 31:
                        newDate = dateList[2] + "-" + month + "-" + day
                        return newDate
            else:
                dateList = date.split(",")
                month, day = dateList[0].split(" ")
                index_of_list = months.index(month) + 1
                if index_of_list < 10:
                    month = "0" + index_of_list
```

```

        else:
            month = str(index_of_list)
            day = day if len(day) == 2 else "0" + day
            if int(day) <= 31:
                newDate = str(dateList[1]).strip() + "-" + month + "-" +
day
                return newDate
    except(ValueError, KeyError, KeyboardInterrupt):
        pass
if __name__ == "__main__":
    main()

```

<https://submit.cs50.io/check50/f3feedfb26802256040586c49502526e54615e8b>

Emojize

```

import emoji

emotka = input()

print(emoji.emojize(f"output: {emotka}", language = "alias"))
True

```

<https://submit.cs50.io/check50/a8342b66a6e8e85c6673326f76a03f7000fc569c>

Frank, Ian and Glen's Letters

```

from pyfiglet import Figlet
figlet = Figlet()
fonts = figlet.getFonts()
import sys
import random

if len(sys.argv) == 3:
    if sys.argv[1].strip() in ["-f", "--font"] and sys.argv[2].strip() in
fonts:
        tekst = input("Input: ")
        figlet.setFont(font = sys.argv[2])
        print(figlet.renderText(tekst))
    else:
        sys.exit("Invalid usage")
elif len(sys.argv) == 1:
    tekst = input("Input: ")
    randomfont = random.choice(fonts)
    figlet.setFont(font = randomfont)

```

```
print(figlet.renderText(tekst))
else:
    sys.exit("Invalid usage")
```

<https://submit.cs50.io/check50/1d71ca73269c9207fcbddc7b0bcf3f6fcab77e27>

Adieu, Adieu

```
import inflect
p = inflect.engine()
import sys
namelist = []
while True:
    try:
        name = input("Name: ").strip()
        namelist = namelist + [name]

    except EOFError:
        namelist = p.join(namelist)
        print(f"Adieu, adieu, to {namelist}")
        break
    except KeyError:
        pass
```

<https://submit.cs50.io/check50/1cde51a91d2642794400ae23ddea72b5c1089fd2>

Guessing Game

```
import random

while True:
    level = input("Level: ")
    if level.isnumeric() == True:
        level = int(level)
    else:
        continue
    if level > 0:
        break
    else:
        continue
randomnr = random.randrange(1, level+1, 1)
while True:
    guess = input("Guess: ")
    if guess.isnumeric() == True:
        guess = int(guess)
```

```

    if guess < randomnr:
        print("Too small!")
        continue
    elif guess > randomnr:
        print("Too large!")
        continue
    else:
        print("Just right!")
        break
else:
    continue

```

<https://submit.cs50.io/check50/b773b92d6aaaadffb78a132b65d598d9837b12cb>

Little Professor

```

import random

def main():
    level = get_level()
    cycle = 0
    score = 0
    while cycle < 10:
        x = generate_integer(level)
        y = generate_integer(level)
        tries = 3
        tried = 0
        while tried < tries:
            try:
                equation = int(input(f"{x} + {y} = "))
            except ValueError:
                print("EEE")
                tried += 1
                continue

            answer = x + y
            if equation != answer:
                print("EEE")
                tried += 1
            else:
                score += 1
                cycle += 1
                tried = 0
                break
        if tried == tries:
            print(f"{x} + {y} = {answer}")
            cycle += 1

```



```

        tried = 0
    print(f"Score: {score}")

def get_level():
    while True:
        try:
            n = int(input("Level: "))
            if n == 1 or n == 2 or n == 3:
                return n
            else:
                continue
        except ValueError:
            continue

def generate_integer(level):
    if level == 1:
        randomnr = random.randint(0, 9)
        return randomnr

    elif level == 2:
        randomnr = random.randint(10, 99)
        return randomnr

    elif level == 3:
        randomnr = random.randint(100, 999)
        return randomnr
    else:
        raise ValueError

if __name__ == "__main__":
    main()

```

<https://submit.cs50.io/check50/e1bc62dab8e2f6bdfa5d5371a923140fed284df8>
 check: 11/12

Bitcoin Price Index

```

import sys
import requests
import json

try:
    n = float(sys.argv[1])
except IndexError:
    sys.exit("Missing command-line argument")
except ValueError:
    sys.exit("Command-line argument is not a number")

```

```

try:
    response =
requests.get("https://api.coindesk.com/v1/bpi/currentprice.json")
    price_data = response.json()
    rate = price_data["bpi"]["USD"]["rate"]

except requests.RequestException:
    print("fail")
rate = rate.replace(",", "")
amount = float(rate) * n
print(f"${amount:,.4f}")

```

<https://submit.cs50.io/check50/a10b0c28ccb7bf091e16361d7b8930c5739aea64>

Evaluation

About learning:

This week turned out to be more challenging when it comes to CS50 exercises but luckily there was no official assignment to be done. It took me substantially longer to do the exercises listed and I struggled for a long time with an exercise called *Professor*. I tried to get advice on the exercise on discord but after multiple redos I was getting nowhere. It was at that point I realized how helpful and necessary writing pseudocode is. I tried reaching out to my colleague but he was unable to help me get 100% green *Check50*. My plan is to reach out to some colleagues from outside of CPNITS minor who would be able to help me score 100% on the exercise.

About relevance:

This week I had to ask for help from other more experienced colleagues. I believe in everyone's work there is going to be a time when one cannot fix a problem on their own and has to start asking around for help. This week taught me how to do it and learn from others.

Week 4 - Unit Testing

Assignment - none

CS50 - week 5

Testing my twttr

```

def main():
    str = input("Input: ").strip()

    print("Output: " + shorten(str))

```

```
def shorten(word):
    output = ''
    remove = ('a', 'e', 'i', 'o', 'u')
    for letter in word:
        if letter.lower() in remove:
            output = output
        else:
            output = output + letter
    return output

if __name__ == "__main__":
    main()
```

```
#importing function from twttr and pytest
from twttr import shorten
import pytest

#defining a test function
def test_letters():
    assert shorten("twitter") == "twtr"

def test_capitalletters():
    assert shorten("Twitter") == "Twtr"

def test_nrandletters():
    assert shorten("Twitter2024") == "Twtr2024"

def test_capitalletteronly():
    assert shorten("TWITTER") == "TWTR"

def test_punctuatuonletters():
    assert shorten("k Twitter..") == "k Twtr.."
```

<https://submit.cs50.io/check50/216ee1cb310e76cd4825e61606b9a25cf34859bb>

Back to the Bank

```
import string

#asking for input and printing respective amount
def main():
    greeting = input("Greeting: ").strip().lower()
    output = value(greeting)
    print(output)
```

```
#checking input
def value(greeting):
    if greeting.startswith("hello"):
        return int("0")
    elif greeting.startswith("h"):
        return int("20")
    else:
        return int("100")

if __name__ == "__main__":
    main()
```

```
#importing pytest and a relevant function
import pytest
from bank import value

#defining test functions
def test_hello():
    assert value("hello") == 0
    assert value("hello mrs and mr") == 0
    assert value("Hello Mrs and Mr") == 0

def test_h():
    assert value("hey") == 20
    assert value("hey sha") == 20
    assert value("Hey sha") == 20

def test_other():
    assert value("wassup") == 100
    assert value("13+2") == 100
    assert value("Wassup") == 100
```

<https://submit.cs50.io/check50/2d45ca5754888e66e9f7a92f868e4f9d1613d0cb>

Re-requesting a Vanity Plate

plates.py - same code as before

```
#importing pytest and a relevant function
import pytest
from plates import is_valid

#defining test functions
def test_plates():
    assert is_valid("WE111") == True
    assert is_valid("WEES") == True
```

```

def test_plateswronglenght():
    assert is_valid("WE111111") == False
    assert is_valid("W33333S") == False

def test_plateswrongstart():
    assert is_valid("111111") == False
    assert is_valid("W33333S") == False

def test_plate0placement():
    assert is_valid("WE01") == False

def test_platessigns():
    assert is_valid("WE.11") == False
    assert is_valid("WE-11") == False
    assert is_valid("WE11-") == False

def test_platenrplacement():
    assert is_valid("1101") == False
    assert is_valid("WE1EW") == False

```

<https://submit.cs50.io/check50/e857684793543e44659e5c559364fbbb91d96338>

Refueling

```

def main():
    while True:
        try:
            fraction = input("Fraction: ")
            percentage = convert(fraction)
            break
        except (ValueError, ZeroDivisionError):
            pass
    print(gauge(percentage))

#convert expects a str in X/Y format as input, wherein each of X and Y is an
integer, and returns that fraction as a percentage rounded to the nearest
int between 0 and 100, inclusive. If X and/or Y is not an integer, or if X
is greater than Y, then convert should raise a ValueError. If Y is 0, then
convert should raise a ZeroDivisionError.
def convert(fraction):
    x, y = fraction.split("/")
    x = int(x)
    y = int(y)

    if y == 0:
        raise ZeroDivisionError
    elif x > y:
        raise ValueError

```

```

    elif x.is_integer() == False:
        raise ValueError
    elif y.is_integer() == False:
        raise ValueError

    percentage = int(round(x / y * 100))
    return percentage

#gauge expects an int and returns a str
def gauge(percentage):
    if 1 < percentage < 99:
        return(f"{percentage}%")
    elif percentage >= 99:
        return("F")
    elif percentage <= 1:
        return("E")

if __name__ == "__main__":
    main()

```

```

#importing pytest and relevant functions
import pytest
from fuel import convert, gauge

#defining test functions
def test_fractions():
    assert convert("1/3") == 33
    assert convert("0/3") == 0
    assert convert("3/3") == 100
def test_zerodivision():
    with pytest.raises(ZeroDivisionError):
        convert("1/0")
def test_valuerror():
    with pytest.raises(ValueError):
        convert("cat/dog")
def test_gauge():
    assert gauge(50) == "50%"
    assert gauge(0) == "E"
    assert gauge(100) == "F"
    assert gauge(1) == "E"
    assert gauge(99) == "F"

```

<https://submit.cs50.io/check50/3ce8a190659f148ba1250cf0502e32b9720727c2>

Evaluation

About learning:

Learning unit testing was pretty straight forward, especially in comparison to the previous week. Gladly I didn't encounter any bigger issues while doing CS50 exercises for this week. I managed to get a good understanding of the basics of unit testing, however doing more complicated tests than asserting an outcome or assuming an error still seems dark magic to me (luckily it isn't needed for CS50).

About relevance:

I don't see any relevance to my main course of study. Still, programming is something I would like to continue after CPNITS and because unit testing is greatly important in a world of programming, I find this learning experience very relevant to my person.

Week 5 - File I/O, Regex

Assignment - none

CS50 - week 6&7

Lines of Code

```
import sys

#expects a name or a path as a sys.argv[1]
try:
    path = sys.argv[1]
#doesn't allow for no arg written
except:
    sys.exit("Too few argument lines")
#doesn't allow for arg longer than 1 line
if len(sys.argv)>2:
    sys.exit("Too many argument lines")
#allows only for arg to end with .py
elif sys.argv[1].endswith(".py") == False:
    sys.exit("Not a python file")
#opens a file (if possible), removes empty lines, comment lines and lines
with spaces only
try:
    with open(path, "r") as content:
        contentlist = []
        contentlist = content.readlines()
        for line in contentlist[:]:
            if len(line) == 0:
```

```

        contentlist.remove(line)
    elif line.strip().startswith("#"):
        contentlist.remove(line)
    elif line.isspace() == True:
        contentlist.remove(line)
#counts the lines and prints the amount
    counter = 0
    for line in contentlist:
        if line:
            counter += 1
    print(counter)
#allows only for existing file or path names to be entered
except FileNotFoundError:
    sys.exit("File does not exist")

```

<https://submit.cs50.io/check50/ca67a01d675d021f7b498e75a5b3ba4dcd843209>

Pizza Py

```

import sys, os
#importing tabulate and csv
import tabulate, csv
#expects a name or a path as a sys.argv[1]
try:
    path = sys.argv[1]
#doesn't allow for no arg written
except:
    sys.exit("Too few argument lines")
#doesn't allow for arg longer than 1 line
if len(sys.argv)>2:
    sys.exit("Too many argument lines")
#allows only for arg to end with .py
elif sys.argv[1].endswith(".csv") == False:
    sys.exit("Not a csv file")
#opens a file (if possible)
try:
    with open(path, "r") as menu:
        #creates an empty list
        list = []
#reads content and appends to the list
        body = csv.DictReader(menu)
        for line in body:
            list.append(line)
#prints the table in an expected way
        print(tabulate.tabulate(list, headers = "keys", tablefmt = "grid"))
except FileNotFoundError:
    sys.exit("File does not exist")

```


<https://submit.cs50.io/check50/37c6705ea7e2853dd330c2eb6d5310ba4c320db9>

Scourgify

```
import sys, os, csv

#Expects the user to provide two command-line arguments
try:
    file1 = sys.argv[1]    #the name of an existing CSV file to read as
    input, whose columns are assumed to be, in order, name and house
    file2 = sys.argv[2]    #the name of a new CSV to write as output, whose
    columns should be, in order, first, last, and house
except len(sys.argv)<3:
    sys.exit("Too few command-line arguments")
except len(sys.argv)>3:
    sys.exit("Too many command-line arguments")
if file1.endswith(".csv") == False and file2.endswith(".csv") == False:
    sys.exit("Not a csv file")

list = [] #empty list

#open a file and split the name and latter append to the list
try:
    with open(file1, "r") as source:
        hogwartlist = csv.DictReader(source)
        for row in hogwartlist:
            last, first = row["name"].split(",")
            house = row["house"]
            list.append([first.strip(), last.strip(), house])
except FileNotFoundError:    #when the file does not exist
    sys.exit("File does not exist")

#open a new csv and write expected output
with open(file2, "w") as output:
    writer = csv.writer(output)
    writer.writerow(["first", "last", "house"])
    for row in list:
        writer.writerow(row)
```

<https://submit.cs50.io/check50/5d73732d31882e1b2058a8f3c191b1e14f60395c>

CS-50 P-shirt

```
import sys, os
from PIL import Image, ImageOps

#expects 2 command-line arguments
try:
```

```

    input = sys.argv[1]    #the name (or path) of a JPEG or PNG to read
                            (i.e., open) as input
    output = sys.argv[2]   #the name (or path) of a JPEG or PNG to write
                            (i.e., save) as output
except:
    sys.exit("Too few command-line arguments")

#split path/file into name+extension
inputsplit = os.path.splitext(input)
outputsplit = os.path.splitext(output)

#exit via sys.exit if the input's names do not end in .jpg, .jpeg, or .png,
case-insensitively
if inputsplit[1] not in(".jpg", ".jpeg", ".png"):
    sys.exit("incorrect extension")
#exit via sys.exit if the input's names do not end in .jpg, .jpeg, or .png,
case-insensitively
elif outputsplit[1] not in(".jpg", ".jpeg", ".png"):
    sys.exit("incorrect extension")
#else if the input's name does not have the same extension as the output's
name: sys.exit
elif inputsplit[1] != outputsplit[1]:
    sys.exit("Input and output have different extensions")
elif len(sys.argv) > 3:
    sys.exit("Too many command-line arguments")
#tries to open the shirt image and get its size as tuple
try:
    shirt = Image.open("shirt.png")
    size = shirt.size
#tries to resize and crop the input with ImageOps.fit
    with Image.open(input, "r") as original:
        resized_original = ImageOps.fit(original, size)
#overlay the shirt with Image.paste
        resized_original.paste(shirt, shirt,)
#saves the results with Image.save
        resized_original.save(fp = output)
#except if the specified input does not exist exit via sys.exit
except FileNotFoundError:
    sys.exit("File does not exist")

```

<https://submit.cs50.io/check50/06f32de1793a002e13fc06b3a0d97b821659bcde>

NUMB3RS

```

import re
import sys

```

```
def main():
    print(validate(input("IPv4 Address: ")))

def validate(ip):
    if
re.search(r"^((25[0-5]\.|2[0-4][0-9]\.|1[0-9][0-9]\.|[1-9][0-9]\.|[0-9]\.){3
}((25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])){1}$", ip):    #search
for correct form of IP4
        return(True)    #return True if found
    else:
        return(False)    #return False if not found

if __name__ == "__main__":
    main()
```

```
import pytest
from numb3rs import validate

def test_validate_too_big():
    assert validate("1000.300.100.1") == False
    assert validate("30.300.50.13") == False
def test_validate_not_nr():
    assert validate("m.30.cat.1") == False

def test_validate_not_nr():
    assert validate("3.30.255.1") == True
```

<https://submit.cs50.io/check50/bedba6d1f1f676a6486f200ddc83789cebcc3103>

Watch on Youtube

```
import re
import sys

def main():
    print(parse(input("HTML: ")))

def parse(s):
    if match := re.search(r'^<iframe (?:(width="(?:\d)+" height="(?:\d)+"
)?src="https?:\/\/(?:www\.)?youtube\.com\/embed\/([a-zA-Z0-9]+)"(?:.)*><\/if
rame>$', s, re.IGNORECASE):
        return(f"https://youtu.be/{match.group(1)}")
    ...
```

```
if __name__ == "__main__":
    main()
```

<https://submit.cs50.io/check50/de3b62a864f36bc29b7773d7fcad09402665bf4b>

Working 9 to 5

```
import re
import sys

def main():
    print(convert(input("Hours: ")))

def convert(s):
    if matches := re.search(r'^(\d(?:\d)?)(?:\d\d)? ([PA]M) to (\d(?:\d)?)(?:\d\d)? ([PA]M)$', s): #searching for matches

#catching times
        hour1 = int(matches.group(1))
        min1 = matches.group(2)
        PA1 = matches.group(3)
        hour2 = int(matches.group(4))
        min2 = matches.group(5)
        PA2 = matches.group(6)

#check for ValueError
        if hour1 > 12:
            raise ValueError
        if hour1 < 0:
            raise ValueError
        if hour2 > 12:
            raise ValueError
        if hour2 < 0:
            raise ValueError

#converting times
        if PA1 == "AM":
            hour1 = hour1
            if hour1 == 12:
                hour1 -= 12
        elif PA1 == "PM":
            hour1 = hour1 + 12
            if hour1 == 24:
                hour1 -= 12
        if PA2 == "AM":
```

```

        hour2 = hour2
        if hour2 == 12:
            hour2 -= 12
    elif PA2 == "PM":
        hour2 = hour2 + 12
        if hour2 == 24:
            hour2 -= 12
#checks if hour is smaller than 10, if yes converts it to 0x format
    if hour1 < 10:
        hour1 = f"0{hour1}"

    if hour2 < 10:
        hour2 = f"0{hour2}"

    if min1 == "":
        hour1 = f"{hour1}:00"
    else:
        #check if minutes are in correct
format
        minutes1 = int(min1.strip(":"))
        if minutes1 >= 60:
            raise ValueError
    if min2 == "":
        hour2 = f"{hour2}:00"
    else:
        #check if minutes are in correct
format
        minutes2 = int(min2.strip(":"))
        if minutes2 >= 60:
            raise ValueError
#returns time in converted version
    return(f"{hour1}{min1} to {hour2}{min2}")
    else:
        raise ValueError

if __name__ == "__main__":
    main()

```

```

import pytest
from working import convert

def test_regular():
    assert convert("10:50 PM to 8 AM") == "22:50 to 08:00"

def test_12ampm():
    assert convert("12 PM to 12 AM") == "12:00 to 00:00"

```

```

    assert convert("12 AM to 12 PM") == "00:00 to 12:00"

def test_ValueError():
    try:
        assert convert("13 PM to 1 AM") == "25:00 to 01:00"
        assert convert("10:60 AM to 11:30 AM") == "10:60 to 11:30"
    except ValueError:
        print("Wrong time input")

def test_omit_to():
    try:
        assert convert("10 AM 1 PM") == "10:00 to 13:00"
    except ValueError:
        print("wrong input")

```

<https://submit.cs50.io/check50/61cc138135d7895b08cc3bc3bfde8f23446a656d>

Regular, um, Expressions

```

import re
import sys

def main():
    print(count(input("Text: ")))

def count(s):
    #using word boundry find all relevant ums, case insensitive
    umlist = re.findall(r'\bum\b', s, re.IGNORECASE)
    #count the list and make an int
    count = int(len(umlist))
    #return int
    return count

if __name__ == "__main__":
    main()

```

```

import pytest
from um import count

def test_regular():
    assert count("Hey, um, what's up?") == 1
    assert count("Hey") == 0

```

```
def test_uminword():
    assert count("You might need an umbrella today.") == 0

def test_combined():
    assert count("Um, you must think, um, I am dumb.") == 2

def test_empty():
    assert count("") == 0
```

<https://submit.cs50.io/check50/b420854aac99cd6408f0112968e240dbaf8e1a75>

Response Validation

```
from validator_collection import checkers, errors

#define main function
def main():
    print(check(input("What's your e-mail address? ")))

#define e-mail check function
def check(s):
    email = checkers.is_email(s)
    if email == True:
        return("Valid")
    else:
        return("Invalid")

if __name__ == "__main__":
    main()
```

<https://submit.cs50.io/check50/985a193a22be51e9f617d18ced5b842aef6b963e>

Evaluation

About learning:

This week problem sets took a substantial amount of time for me to finish, especially the File I/O ones. I tried to be more consistent using pseudo code, which helped make the more complex problems more approachable.

About relevance:

Focusing not only on doing things correctly but also thinking about producing readable and understandable code this time around was very relevant to my course of studies since I mostly focus on collaborative work. I believe it is important for me to code in a way that could prospect in collaboration with other coders in the future.

Week 6 - OOP

Assignments - Scavenger Hunt

Nerdy humor

```
#import libraries
import pyjokes, sys, random

def main():
    get_a_joke(get_list_of_jokes())

def get_list_of_jokes():
    try:
        #get a list of supported languages by pyjokes
        supported_languages = pyjokes.get_languages()

        #if sys.argv lenght is 3
        if len(sys.argv) == 3:
            #try extracting the relevant part of an argument
            lang = sys.argv[1].split("=")
            cat = sys.argv[2].split("=")
            if lang[1] not in supported_languages:
                sys.exit("Why not choose Klingon?") #language not supported
            alljokes = pyjokes.get_jokes(language=lang[1], category=cat[1])

        #if sys.argv lenght is 2
        elif len(sys.argv) == 2:
            lang = sys.argv[1].split("=")
            if lang[0] == "language": #check if agrv[1] is a language or a
                category
                if lang[1] not in supported_languages:
                    sys.exit("Why not choose Klingon?") #language not
                    supported
                alljokes = pyjokes.get_jokes(language=lang[1],
                category="all")
            elif lang[0] == "category":
                alljokes = pyjokes.get_jokes(language="en",
                category=lang[1])
            else:
                sys.exit("Why not choose Klingon?") # Invalid arg

        #if sys.argv lenght is 1
        elif len(sys.argv) == 1:
            alljokes = pyjokes.get_jokes(language="en", category="all")
        else:
            sys.exit("You're hilarious!") #Too many arg
```



```

        return(alljokes)
#handle exceptions
except (AttributeError, IndexError, KeyError):
    sys.exit("Why not choose Klingon?")
except UnboundLocalError:
    sys.exit("You're hilarious!")
except:
    sys.exit("You're hilarious!")

#print a random joke from alljokes
def get_a_joke(s):
    joke = random.choice(s)
    print(joke)

if __name__ == "__main__":
    main()

```

What's your number?

```

#import lib
import json, requests, sys
def main():
    #ask for passowrd with input
    password = input("What's your password? ")
    #get a nr with collect()
    number = collect(password)
    #print my number
    print(f"My number is: {number}")

def collect(password):
    try:
        #ask for the nr
        response =
requests.get(f'https://cpnits.com/s-hunt/whatsyournumber.php?password={password}')
    #get a dict with json
    numberdict = response.json()
    #get a value from a key - your number
    number = int(numberdict.get('your number'))
    return(number)
#except errors, which are going to be printed in a specified form
except:
    response = str(response).split(" ")
    error = response[1].replace(">", "")

```

```

        error = error.replace("[", "")
        error = error.replace("]", "")
        sys.exit(f"Error {error}")

if __name__ == "__main__":
    main()

#my password: 7969-uz8k47re9zaqi
#my number: 613

```

Undutchify

```

import re
#define main
def main():
    #get a transcript which isn't empty
    while True:
        transcript = get_transcript()
        if transcript.isspace() is False:
            break
    #count he
    totalhe = start_counting(transcript)
    #print total
    print(f"He's found in the transcript: {totalhe}.")

    #get transcript
    def get_transcript():
        transcript = input("Transcript: ")
        return(transcript)

    #count he and return
    def start_counting(s):
        helist = re.findall(r'\bhe\b', s, re.IGNORECASE)
        hèlist = re.findall(r'\bhè\b', s, re.IGNORECASE)
        amountofhe = int(len(helist))
        amountofhè = int(len(hèlist))
        totalhe = amountofhe + amountofhè
        return(totalhe)

if __name__ == "__main__":
    main()

```

Enigma

```
def main():
    # Read the key from the key.txt file
    with open("key.txt", "r") as key_file:
        key = key_file.read().strip()

    # Read the gibberish from the gibberish.txt file
    with open("gibber.txt", "r") as gibber_file:
        gibber = gibber_file.read().strip()

    # Ungibber the gibberish
    readable_text = ungibber(key, gibber)

    # Print the ungibberished readable text
    print(readable_text)

def ungibber(key, gibber):
    #standard alphabet
    standard_alphabet =
    "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
    #make a canvas allowing to ungibber the text using translate function
    translation_canvas = str.maketrans(key, standard_alphabet)
    #translate using the canvas
    decrypted_text = gibber.translate(translation_canvas)
    #return decrypted text
    return(decrypted_text)

if __name__ == "__main__":
    main()

#password: 7969-uz8k47re9zaqi
#decrypted text under:
'''
It was all very well to say "Drink me," but the wise little Alice was not
going to do that in a hurry. "No, I'll look first," she said, "and see
whether it's marked 'poison' or not"; for she had read several nice little
histories about children who had got burnt, and eaten up by wild beasts and
other unpleasant things, all because they would not remember the simple
rules their friends had taught them: such as, that a red-hot poker will burn
you if you hold it too long; and that if you cut your finger very deeply
with a knife, it usually bleeds; and she had never forgotten that, if you
drink much from a bottle marked "poison," it is almost certain to disagree
with you, sooner or later.
So she called softly after it, "Mouse dear! Do come back again, and we won't
talk about cats or dogs either, if you don't like them!" When the Mouse
```

heard this, it turned round and swam slowly back to her: its face was quite pale (with passion, Alice thought), and it said in a low trembling voice, "Let us get to the shore, and then I'll tell you my history, and you'll understand why it is I hate cats and dogs."

By this time she had found her way into a tidy little room with a table in the window, and on it (as she had hoped) a fan and two or three pairs of tiny white kid gloves: she took up the fan and a pair of the gloves, and was just going to leave the room, when her eye fell upon a little bottle that stood near the looking-glass. There was no label this time with the words "DRINK ME," but nevertheless she uncorked it and put it to her lips. "I know something interesting is sure to happen," she said to herself, "whenever I eat or drink anything; so I'll just see what this bottle does. I do hope it'll make me grow large again, for really I'm quite tired of being such a tiny little thing!"

'''

```
#importing relevant library and function
import pytest
from enigma import ungibber

#defining test functions
def test_enigma1():
    assert ungibber('UFZmORzdjClPoSiJVNXLaxuDqyIsvWpMkAcHQnfetGYKwgTbhErB',
    '"hdq," XUjm LdO Wimi, "LdO FOXL uUq Li ODJPUjS jL jX Li mi jL."'') ==
    '"Why," said the Dodo, "the best way to explain it is to do it."'

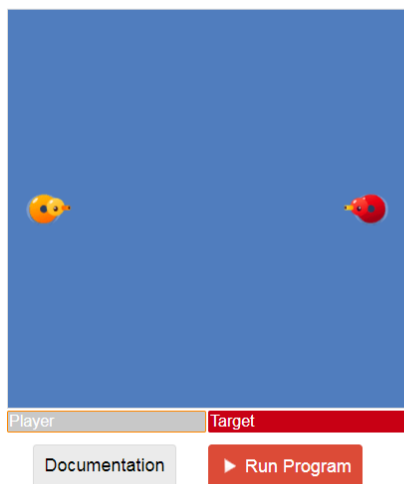
def test_enigma2():
    assert ungibber('mSwhQYIXRqyxMHVAzCWltdcTjkNsepEPriGoUuKaFZbOJfLBDngv',
    'FX, G WXVtxhH'l xRyQ lXm1!') == 'Oh, I shouldn't like that!'

def test_enigma():
    assert ungibber('XZUMqzTvKJHBagiFobplEmOINfusCePxSrwhGVtQncdyLAYWRDkj',
    'Avqbq Oqbq Miibp XBB biEgM lvq vXBB, ZE1 lvqN Oqbq XBB BiUHqM') == 'There
    were doors all round the hall, but they were all locked'
```

It's hip to be square

Link to the code and game: <https://blockly.games/pond-tutor?lang=en&level=9#djj63a>

Screen dump:



Pond
Logic
Loops
Math

```
while (true) {
  swim ( 90° );
  while ( getY ( ) >= 99 ) {
    stop ( );
    swim ( 0° );
    while ( getX ( ) >= 89 ) {
      stop ( );
      swim ( 270° );
      while ( getY ( ) <= 1 ) {
        stop ( );
        swim ( 180° );
        if ( getX ( ) <= 1 ) {
          swim ( 90° );
        }
      }
    }
  }
}
```

CS50 - week 8

Seasons of Love

```
from datetime import date, timedelta
import operator, sys, inflect
p = inflect.engine()

def main():
    ageint = age_in_min(str(input("Date of birth: ")))
    agestr = (int_to_str(ageint)).capitalize()
    print(f"{agestr} minutes")
    #print how old they are in minutes
    #print(...)

def age_in_min(s):
    #what day is it today
    today = str(date.today())
    #format dates to usable form
    try:
        formated_date = date.fromisoformat(today)
        formated_date_of_birth = date.fromisoformat(s)
    #sys.exit when incorrect date format
    except ValueError:
        sys.exit("Invalid date")
```

```

    #subtract the dates and get the age in sec
    age_in_sec = int(timedelta.total_seconds(operator.sub(formated_date,
formated_date_of_birth)))
    return(age_in_sec/60)

def int_to_str(s):
    s = int(s)
    agestr = p.number_to_words(s, andword="")
    return(agestr)

if __name__ == "__main__":
    main()

```

```

from seasons import age_in_min
import pytest

def test_age_in_min():
    assert age_in_min("2000-08-22") == 12702240.0

```

<https://submit.cs50.io/check50/a598003827d3b4f702938244bf4fbd16771a39fa>

Results for cs50/problems/2022/python/seasons generated by check50 v3.3.11
:) seasons.py and test_seasons.py exist
:) Input of "1999-01-01" yields "Five hundred twenty-five thousand, six hundred minutes" when today is 2000-01-01
:) Input of "2001-01-01" yields "One million, fifty-one thousand, two hundred minutes" when today is 2003-01-01
:) Input of "1995-01-01" yields "Two million, six hundred twenty-nine thousand, four hundred forty minutes" when today is 2000-01-01
:) Input of "2020-06-01" yields "Six million, ninety-two thousand, six hundred forty minutes" when today is 2032-01-01
:) Input of "1998-06-20" yields "Eight hundred six thousand, four hundred minutes" when today is 2000-01-01
:) Input of "February 6th, 1998" prompts program to exit with sys.exit
:) seasons.py passes all checks in test_seasons.py
To see more detailed results go to <https://submit.cs50.io/check50/4933dba844f018c7b0fff4bead029c7bfab6bb26>

Cookie Jar

```

class Jar:

    # __init__ should initialize a cookie jar with the given capacity
    def __init__(self, capacity=12):
        self._capacity = capacity

    # If capacity is not a non-negative int, though, __init__ should instead
    raise a ValueError
    if capacity < 0:
        raise ValueError ("Wrong capacity")
    self._size = 0

    # __str__ should return a str with n🍪, where n is the number of
    cookies in the cookie jar
    def __str__(self):
        return ("🍪" * self.size)

```

```

    # deposit should add n cookies to the cookie jar
    def deposit(self, n):
        # If adding that many would exceed the cookie jar's capacity, though,
        deposit should instead raise a ValueError
        if n > self.capacity:
            raise ValueError("Too many cookies in the jar")
        elif self.size + n > self.capacity:
            raise ValueError("Too many cookies in the jar")
        self._size += n

    # withdraw should remove n cookies from the cookie jar
    def withdraw(self, n):
        # If there aren't that many cookies in the cookie jar, though, withdraw
        should instead raise a ValueError
        if self.size < n:
            raise ValueError("Not enough cookies in the jar")
        self._size -= n

    # capacity should return the cookie jar's capacity
    @property
    def capacity(self):
        return self._capacity

    # size should return the number of cookies actually in the cookie jar,
    initially 0
    @property
    def size(self):
        return self._size

def main():
    jar = Jar()
    jar.deposit()
    jar.withdraw()
    print(jar)

if __name__ == "__main__":
    main()

```

```

from jar import Jar
import pytest

def test_init():
    jar = Jar()
    assert jar.capacity == 12
    jar2 = Jar(3)
    assert jar2.capacity == 3

```

```

def test_str():
    jar = Jar()
    assert str(jar) == ""
    jar.deposit(1)
    assert str(jar) == "🍪"
    jar.deposit(11)
    assert str(jar) == "🍪🍪🍪🍪🍪🍪🍪🍪🍪🍪🍪🍪"

def test_deposit():
    jar = Jar()
    jar.deposit(1)
    assert jar.size == 1
    jar.deposit(6)
    assert jar.size == 7

def test_withdraw():
    jar = Jar()
    jar.deposit(5)
    jar.withdraw(2)
    assert jar.size == 3
    jar.withdraw(1)
    assert jar.size == 2

```

<https://submit.cs50.io/check50/4eca8904276963ee06bc5ae49b31acd62c53da17>

Results for cs50/problems/2022/python/jar generated by check50 v3.3.11

- :) jar.py exists
- :) Jar's constructor initializes a cookie jar with given capacity
- :) Jar's constructor raises ValueError when called with negative capacity
- :) Empty jar prints zero cookies
- :) Jar prints total number of cookies deposited
- :) Jar's deposit method raises ValueError when deposited cookies exceed the jar's capacity
- :) Jar's withdraw method removes cookies from the jar's size
- :) Jar's withdraw method raises ValueError when withdrawn cookies exceed jar's size
- :) Implementation of Jar passes all tests in test_jar.py
- :) test jar.py contains at least four functions

CS50 Shirtificate

```

from fpdf import FPDF, Align
pdf = FPDF()

def main():
    name = input("Name: ")
    get_pdf(name)

def get_pdf(name):

```



```

pdf.add_page()
pdf.set_font("helvetica", "B", 50)
pdf.cell(0, 60, "CS50 Shirtificate", new_x="LMARGIN", new_y="NEXT",
align="C")
pdf.image("shirtificate.png", w=pdf.epw)
pdf.set_font_size(20)
pdf.text(x=47.5, y=140, text=f"{name} took CS50")
pdf.output("shirtificate.pdf")

if __name__ == "__main__":
    main()

```

<https://submit.cs50.io/check50/005b7dd4586697092befe4d9228f2dcfac22f1d2>

check50

cs50/problems/2022/python/shirtificate

:) shirtificate.py exist

```

Log
checking that shirtificate.py exists...

```

:) shirtificate.py creates a PDF called shirtificate.pdf

```

Log
running python3 shirtificate.py...
sending input John Harvard...
checking that program exited with status 0...
checking that shirtificate.pdf exists...

```

Evaluation

About learning:

This week was an intense learning experience. Scavenger Hunt thoroughly tested newly acquired programming knowledge and problem solving skills. The day turned out to be even tougher than expected. Because of some issues with passwords, we were given 11 hours instead of the planned 8 hours. I had to work for 10 hours with small breaks to solve the problems presented. It was not only a great test of my knowledge but also perseverance and dedication. After finishing I was hugely relieved and satisfied with the work I've done. CS50's theme of the week was Object Oriented Programming. It happened to be the most difficult week of CS50 for me. I found classes to be especially hard to understand but at the same time very important. Keeping that in mind I am planning on further investigating their usage and characteristics.

About relevance:

Working on projects with a tight deadline is not uncommon in the professional world of media, but not only. This week was a great test of my ability to do just that. I learned how I react to such situations, what are my limits and how to further improve.

Week 7 - SQL

Assignment - none

CS50x - week 7

Songs

1.

```
SELECT name FROM songs;
```

2.

```
SELECT name
FROM songs
ORDER BY tempo;
```

3.

```
SELECT name
FROM songs
ORDER BY duration_ms DESC
LIMIT 5;
```

4.

```
SELECT name
FROM songs
WHERE danceability > 0.75 AND energy > 0.75 AND valence > 0.75;
```

5.

```
SELECT AVG(energy)
FROM songs;
```

6.

```
SELECT name
FROM songs
WHERE artist_id =
(
    SELECT id
    FROM artists
    WHERE name = 'Post Malone'
);
```

7.

```
SELECT AVG(energy)
FROM songs
JOIN artists ON songs.artist_id = artists.id
WHERE artists.name = 'Drake';
```

8.

```
SELECT name
FROM songs
WHERE name LIKE '%feat.%';
```

Results for cs50/problems/2024/x/songs generated by check50 v3.3.11

```
:) SQL files exists
:) answers.txt exists
:) 1.sql produces correct result
:) 2.sql produces correct result
:) 3.sql produces correct result
:) 4.sql produces correct result
:) 5.sql produces correct result
:) 6.sql produces correct result
:) 7.sql produces correct result
:) 8.sql produces correct result
:( answers.txt includes reflection
   answers.txt does not contain a sufficiently long reflection
```

Movies

1

```
SELECT title
FROM movies
WHERE year=2008;
```

2

```
SELECT birth
FROM people
WHERE name='Emma Stone';
```

3

```
SELECT title
FROM movies
WHERE year>=2018
ORDER BY title;
```

4

```
SELECT COUNT(movie_id)
FROM ratings
WHERE rating=10;
```

5

```
SELECT title, year
FROM movies
WHERE title LIKE 'Harry Potter%'
ORDER BY year;
```

6

```
SELECT AVG(rating)
FROM ratings
JOIN movies ON movie_id=id
WHERE year=2012;
```

7

```
SELECT title, rating
FROM movies
JOIN ratings ON movie_id=id
WHERE year=2010
ORDER BY rating DESC, title;
```

8

```
SELECT name
FROM people
WHERE id IN
(
    SELECT person_id
    FROM stars
    WHERE movie_id =
        (
            SELECT id
            FROM movies
            WHERE title='Toy Story'
        )
);
```

9

```
SELECT name
FROM people
WHERE id IN
(
    SELECT person_id
    FROM stars
    WHERE movie_id IN
        (
            SELECT id
            FROM movies
            WHERE year=2004
        )
)
ORDER BY birth;
```

10

```
SELECT name
FROM people
WHERE id IN
```

```
(
  SELECT person_id
  FROM directors
  WHERE movie_id IN
  (
    SELECT movie_id
    FROM ratings
    WHERE rating >= 9
  )
);
```

11

```
SELECT title
FROM movies
JOIN ratings ON movie_id=id
WHERE id in
(
  SELECT movie_id
  FROM stars
  WHERE person_id=
  (
    SELECT id
    FROM people
    WHERE name='Chadwick Boseman'
  )
)
ORDER BY rating DESC LIMIT 5;
```

12

```
SELECT title
FROM movies
WHERE id IN
(
  SELECT movie_id
  FROM stars
  WHERE person_id=
  (
    SELECT id
    FROM people
    WHERE name='Bradley Cooper'
  )
)
AND id IN
(
  SELECT movie_id
  FROM stars
  WHERE person_id=
```

```
(
    SELECT id
    FROM people
    WHERE name='Jennifer Lawrence'
)
);
```

13

```
SELECT name
FROM people
WHERE id IN
(
    SELECT person_id
    FROM stars
    WHERE movie_id IN
    (
        SELECT movie_id
        FROM stars
        WHERE person_id=
        (
            SELECT id
            FROM people
            WHERE name='Kevin Bacon' AND birth=1958
        )
    )
)
AND name != 'Kevin Bacon';
```

<https://submit.cs50.io/check50/4ffebdc5156158607f1275e0b3f921fa19d7fab4>

Evaluation

About learning:

This week I didn't encounter any issues. The murder exercise was the most challenging one for me and I had to try solving it again after taking a break but it was very rewarding when I finally solved it.

About relevance:

SQL is a very handy and pretty easy to learn programming language. I believe learning it can be very beneficial when working with databases of all kinds. Another relevant aspect of this week was logical thinking and problem solving which I had to exercise finding a thief.

Week 8, 9 - Pico Project

Assignments - Pico

Onboard

```
import time
import machine
import sys
import datetime

led = machine.Pin("LED", machine.Pin.OUT)
unit = 0.15

def get_now_time() -> str:
    # get date time
    # Get the local time as a tuple and convert it to a datetime object
    local_time_tuple = time.localtime(time.time())
    local_time = datetime.datetime(local_time_tuple[0],
local_time_tuple[1], local_time_tuple[2], local_time_tuple[3],
local_time_tuple[4], local_time_tuple[5])
    return (f"The time is {local_time}")

def read_celsius() -> float:
    # base on
https://www.halvorsen.blog/documents/technology/iot/pico/pico\_temperature\_sensor\_builtin.php
    # The internal temperature sensor is connected to an internal ADC
    pin #4
    adcpin = 4
    sensor = machine.ADC(adcpin)
    # The built-in Analog-to-Digital Converter (ADC) has a 16 bit
    resolution ( $2^{16} = 65536$  different levels), producing values from 0 to
    65535. The read_u16() function gives a value between 0 and 65535
    adc_value = sensor.read_u16()
    # convert read_u16 value to Voltage Signal 0-3.3v
    volt = (3.3/65535)*adc_value
    # convert Voltage to degrees Celsius (formula is documented in the
    Raspberry Pi Pico RP2040 datasheet)
    temp = 27 - (volt - 0.706)/0.001721
    return round(temp, 1)

def morse_dit():
    led.on()
    time.sleep(unit)
```

```

        led.off()
        time.sleep(unit)
def morse_dah():
    led.on()
    time.sleep(3*unit)
    led.off()
    time.sleep(unit)
def morse_letterspace():
    # one space is at the end of each letter, makes total of 3
    time.sleep(3*unit)

def morse_wordspace():
    # one space is at the end of each letter, then 3 letterspaces
    time.sleep(7*unit)

def morse_pico_pi():
    # p = dit dah dah dit  i = dit dit  c = dah dit dah dit  o = dah dah dah
    p = dit dah dah dit  i = dit dit
    # say "pico pi" using Morse code
    pico_pi =
morse_dit();morse_dah();morse_dah();morse_dit();morse_letterspace();mors
e_dit();morse_dit();morse_letterspace();morse_dah();morse_dit();morse_da
h();morse_dit();morse_letterspace();morse_dah();morse_dah();morse_dah();
morse_wordspace();morse_dit();morse_dah();morse_dah();morse_dit();morse_
letterspace();morse_dit();morse_dit()

def main():
    while True:
        # check if temp too hot
        temp = read_celsius()
        if temp >= 35:
            print("T00 HOT: {temp} degrees Celsius")
            sys.exit()
        morse_pico_pi() # the morse thingy
        print("Morse: .--. .. -.-. --- .--. ..")
        print(f"Pico Pi's temperature is {temp} degrees Celsius")
        print(get_now_time())
        time.sleep(5)

if __name__ == "__main__":
    main()

```

Server

```
# Import necessary libraries
```



```

import network
from time import sleep
import urequests
import machine
import sys
import socket # network connections ( the Pico with other devices over
the network)

# Wi-Fi network name (SSID) without a password
ssid = 'Orange_Swiatlowod_BC80' # Replace with your Wi-Fi SSID
password = 'of7icMYNYboK7tVaJn' # Repllace with your wifi password

def connect():
    # Create the WLAN object and activate it
    wlan = network.WLAN(network.STA_IF)
    wlan.active(True)
    wlan.connect(ssid, password)

    # Set a timeout for connection attempts (e.g., 15 seconds)
    max_attempts = 15
    attempt = 0

    # Loop until connected or timeout reached
    while not wlan.isconnected() and attempt < max_attempts:
        print(f'Waiting for connection... (Attempt {attempt +
1}/{max_attempts})')
        sleep(1)
        attempt += 1

    # Check if connected after the loop
    if wlan.isconnected():
        # Get and print Internal IP address
        internal_ip = wlan.ifconfig()[0]
        print("Internal IP:", internal_ip)

        # Get and print the External IP Address
        try:
            response = urequests.get("https://httpbin.org/ip")
            external_ip = response.json().get("origin", "Unknown")
            print("External IP:", external_ip)
            response.close()
            return internal_ip # Return IP if successful
        except Exception as e:
            print("Error fetching external IP:", e)
            return None
    else:

```

```

        print("Failed to connect to Wi-Fi.")
        return None

def check_internet():
    try:
        response = urequests.get("https://www.google.com")
        if response.status_code == 200:
            print("Internet connection successful!")
            response.close()
            return True
        else:
            print("Failed to connect to the internet.")
            response.close()
            return False
    except Exception as e:
        print("Error connecting to the internet:", e)
        return False

""" P3- Web server: socket & trouble shooting
Simple first server version """

def my_html():
    html = f"""
        <!DOCTYPE html>
        <html>
        <head>
        <link rel="shortcut icon" href="data:image/x-icon;"
type="image/x-icon">
        </head>
        <body>
        <h1>Hello from the Pico!</h1>
        </body>
        </html>
        """
    return str(html)

# Function to extract the path from request headers
def get_path(headers):
    request_line = headers[0] # First line contains "GET /path?query
HTTP/1.1"
    path_with_args = request_line.split()[1] # Extracts "/path?query"
    path = path_with_args.split('?')[0] # Take the path part (before
'?')
    return path

```

```

# Function to extract query parameters as a dictionary
def get_args(headers):
    request_line = headers[0]
    path_with_args = request_line.split()[1]

    if '?' in path_with_args:
        query_string = path_with_args.split('?')[1] # Get the query
part (after '?')
        args = {}
        for param in query_string.split('&'):
            if '=' in param:
                key, value = param.split('=')
                args[key] = value
            else:
                args[param] = None # Handle parameters without '='
        return args
    else:
        return {} # No query parameters

# Function gets as input the IP address of the WiFi and creates a simple
web server using sockets
def server(ip):
    # Define socket address (the internal IP address) and port (80
standard fot HTTP, that way web browsers will know this IP expects a web
server response)
    address = (ip, 80)
    # creates new socket object (listns to incomming connections)
    connection = socket.socket()

    # allows the same adress to be reused (IMPORTANT LINE TO MAKE
CONNECTION RESTART POSSIBLE)
    connection.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    # Connects the socket to the IP address, now the socket knows where
to listen for incoming connections
    connection.bind(address)
    connection.listen(1) # 1 number of connections waiting in line
before being accepted
    print(f"Server running on {ip}")

    # Allows the server to wait for multiple client requests
    while True:
        try:
            # client is a socket that will be used with this specific
one client (separate line)
            # the adress of the client so the server knows where tp send
the response

```

```

        client, addr = connection.accept()
        print("Client connected:", addr)

        # Recieve data from client, reads up to 1024 bytes from
clients request
        headers = client.recv(1024).decode().split("\r\n") # convert
the data to str (readable)
        print("Request recieved:", headers[0])

        # get path and arg from the header
        path = get_path(headers)
        args = get_args(headers)

        # Print path and args
        print('PATH:', path)
        print('ARGS:', args)

        # HTTP response format
        response = "HTTP/1.1 200 OK\nContent-Type: text/html\n\n" +
my_html()
        client.send(response.encode())
        client.close()

    except KeyboardInterrupt:
        # IMPORTANT SO YOU CAN STOP THE SERVER WITH Ctrl+C
        print(f"server {ip} closed by: Ctrl+C")
        sys.exit(1)

    except Exception as e:
        print(f"server {ip} closed by: {e}")
        sys.exit(1)

def main():
    # Connect to the Wi-Fi network and retrieve the internal IP address
    local_ip = connect()
    if local_ip is None:
        # Exit if Wi-Fi connection fails
        sys.exit(1)

    # Check if there is an active internet connection
    valid_internet = check_internet()
    if valid_internet:
        print("Internet connection confirmed.")
    else:
        print("No internet access.")

```

```
# Start the server with the internal IP address
server(local_ip)

if __name__ == "__main__":
    main()
```

Evaluation

About learning:

I experienced less issues than I was expecting working on the pico project. The biggest problem I had was the lack of communication from my partner. I managed to do the first part of the assignment on my own but not getting any reply from my teammate made me worried about the outcome. Luckily, Natalia finally answered and she stepped up trying to make up for the first part of the assignment offering to do the rest. I agreed on doing a smaller part of it (so I can still learn more) and leave the rest to her. This time she did not disappoint me. The code she has sent to me was well written and described.

About relevance:

Since collaboration is a big part of the work I would like to do in the future, working with a partner helped me exercise it.

Week 1 - Web

Assignments - SQL, Murder Mystery, Custom Database

SQL

```
import sqlite3

conn = sqlite3.connect('movies.db')

cursor = conn.cursor()

#Write an SQL query to add the following movies in the movies table:
cursor.execute('''INSERT OR IGNORE INTO movies (id, title, year)
                VALUES (80684, "Star Wars: Episode V - The Empire Strikes
Back", 1980),
                (86190, "Star Wars: Episode VI - Return of the Jedi", 1983),
                (120915, "Star Wars: Episode I - The Phantom Menace", 1999),
                (121765, "Star Wars: Episode II - Attack of the Clones",
2002),
                (121766, "Star Wars: Episode III - Revenge of the Sith",
2005)''')
```

```

#Write an SQL query to add the following people in the people table
cursor.execute('''INSERT OR IGNORE INTO people(id, name, birth)
                VALUES (191, "Ewan Gordon McGregor", 1971),
                (204, "Natalie Portman", 1981),
                (159789, "Hayden Christensen", 1981),
                (184, "George Lucas", 1944)''')

#Write an SQL query linking the first three people of question 2 as stars of
the film Revenge of the Sith
cursor.execute('''INSERT OR IGNORE INTO stars (movie_id, person_id)
                VALUES (121766, 191),
                (121766, 204),
                (121766, 159789)''')

#Write an SQL query linking George Lucas as director for all Star Wars films
cursor.execute('''INSERT OR IGNORE INTO directors (movie_id, person_id)
                VALUES (80684, 184),
                (86190, 184),
                (120915, 184),
                (121765, 184),
                (121766, 184)''')

#Write an SQL query where all Star Wars films are rated 8.0
cursor.execute('''INSERT OR IGNORE INTO ratings (movie_id, rating)
                VALUES (80684, 8),
                (86190, 8),
                (120915, 8),
                (121765, 8),
                (121766, 8)''')

#Write an SQL query where the film Star Wars "Return of the Jedi" with a
rating of 8.0 becomes a 10.0
cursor.execute('''
UPDATE ratings
SET rating = 10.0
WHERE movie_id = 86190
''')

conn.commit()
conn.close()

```

Murder Mystery

Queries:

```

SELECT *
FROM 'crime_scene_report'
WHERE date = 20180115 AND type = "murder" AND city = "SQL City"

```

```
SELECT *
FROM 'person'
WHERE address_street_name='Franklin Ave' and name LIKE '%Annabel%'
```

```
SELECT *
FROM 'person'
WHERE address_street_name='Northwestern Dr'
ORDER BY address_number DESC
```

```
SELECT *
FROM 'interview'
WHERE person_id=14887 OR person_id=16371
```

```
SELECT *
FROM 'get_fit_now_member'
WHERE membership_status='gold' AND id LIKE '48Z%'
```

```
SELECT *
FROM 'drivers_license'
WHERE id=173289 OR id=423327
```

```
SELECT name
FROM 'person'
WHERE license_id=423327
```

Congrats, you found the murderer! But wait, there's more... If you think you're up for a challenge, try querying the interview transcript of the murderer to find the real villain behind this crime. If you feel especially confident in your SQL skills, try to complete this final step with no more than 2 queries. Use this same INSERT statement with your new suspect to check your answer.

```
SELECT *
FROM 'interview'
WHERE person_id=67318
```

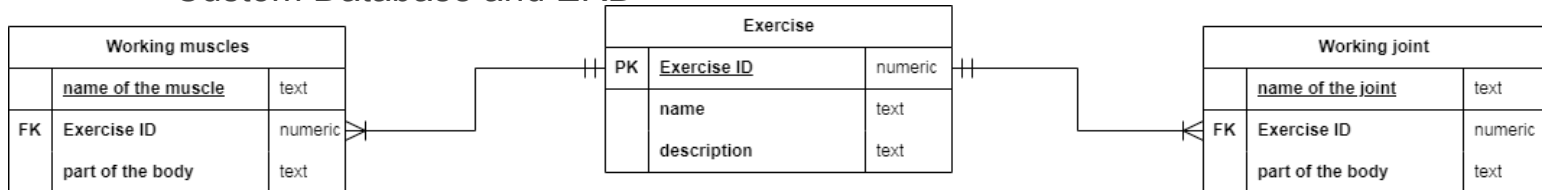
```
SELECT *
FROM 'facebook_event_checkin'
WHERE event_name='SQL Symphony Concert' AND date LIKE '201712%'
ORDER BY person_id
```

```
SELECT *
```

```
FROM 'person'
WHERE id=24556 OR id=99716
```

Congrats, you found the brains behind the murder! Everyone in SQL City hails you as the greatest SQL detective of all time. Time to break out the champagne!

Custom Database and ERD



```
import sqlite3

# Connect to the database
conn = sqlite3.connect('my_database.db')
cursor = conn.cursor()

# Enable foreign keys
cursor.execute("PRAGMA foreign_keys = ON;")

# Create tables
cursor.executescript("""
-- Create the Exercise table
CREATE TABLE IF NOT EXISTS Exercise (
    Exercise_ID INTEGER PRIMARY KEY, -- Primary Key
    name TEXT NOT NULL,
    description TEXT
);

-- Create the Working muscle table
CREATE TABLE IF NOT EXISTS Working_muscle (
    Muscle_ID INTEGER PRIMARY KEY, -- Primary Key
    name_of_the_muscle TEXT NOT NULL,
    Exercise_ID INTEGER, -- Foreign Key to Exercise
    part_of_the_body TEXT,
    FOREIGN KEY (Exercise_ID) REFERENCES Exercise (Exercise_ID) ON
DELETE CASCADE
);

-- Create the Working joint table
```



```

CREATE TABLE IF NOT EXISTS Working_joint (
    Joint_ID INTEGER PRIMARY KEY, -- Primary Key
    name_of_the_joint TEXT NOT NULL,
    Exercise_ID INTEGER, -- Foreign Key to Exercise
    part_of_the_body TEXT,
    FOREIGN KEY (Exercise_ID) REFERENCES Exercise (Exercise_ID) ON
DELETE CASCADE
);
""")

# Insert data into Exercise table
cursor.executemany("""
INSERT OR IGNORE INTO Exercise (Exercise_ID, name, description)
VALUES (?, ?, ?);
""", [
    (1, 'Deadlift (Barbell)', 'A compound exercise targeting the back,
glutes, and hamstrings.'),
    (2, 'Back Squat (Barbell)', 'A lower-body exercise targeting the
quadriceps, hamstrings, and glutes.'),
    (3, 'Standing Calf Raises', 'A lower-body exercise focusing on the
calves.'),
    (4, 'Standing Good Mornings (Barbell)', 'A hip hinge exercise
targeting the hamstrings, glutes, and lower back.'),
    (5, 'Hip Thrust (Barbell)', 'A glute exercise that primarily targets
the glutes and hamstrings.'),
    (6, 'Bench Press (Barbell)', 'An upper-body exercise targeting the
chest, shoulders, and triceps.'),
    (7, 'Overhead Press (Barbell)', 'An upper-body exercise targeting
the shoulders, triceps, and upper chest.'),
    (8, 'Pistol Squat', 'A single-leg squat focusing on the quadriceps,
hamstrings, and glutes.'),
    (9, 'Pull-Up', 'An upper-body exercise focusing on the back and
biceps.'),
    (10, 'Bodyweight Rows', 'A bodyweight exercise targeting the back
and biceps.'),
    (11, 'Leg Lifts', 'A core exercise targeting the lower abs and hip
flexors.')
])

# Insert data into Working_muscle table
cursor.executemany("""
INSERT OR IGNORE INTO Working_muscle (name_of_the_muscle, Exercise_ID,
part_of_the_body)
VALUES (?, ?, ?);
""", [
    ('Hamstrings', 1, 'Lower Body'),

```

```

('Glutes', 1, 'Lower Body'),
('Back', 1, 'Lower Body'),
('Quadriceps', 2, 'Lower Body'),
('Hamstrings', 2, 'Lower Body'),
('Glutes', 2, 'Lower Body'),
('Calves', 3, 'Lower Body'),
('Hamstrings', 4, 'Lower Body'),
('Glutes', 4, 'Lower Body'),
('Lower Back', 4, 'Lower Body'),
('Glutes', 5, 'Lower Body'),
('Hamstrings', 5, 'Lower Body'),
('Chest', 6, 'Upper Body'),
('Shoulders', 6, 'Upper Body'),
('Triceps', 6, 'Upper Body'),
('Shoulders', 7, 'Upper Body'),
('Triceps', 7, 'Upper Body'),
('Chest', 7, 'Upper Body'),
('Quadriceps', 8, 'Lower Body'),
('Glutes', 8, 'Lower Body'),
('Back', 9, 'Upper Body'),
('Biceps', 9, 'Upper Body'),
('Back', 10, 'Upper Body'),
('Biceps', 10, 'Upper Body'),
('Lower Abs', 11, 'Upper Body'),
('Hip Flexors', 11, 'Lower Body')
])

# Insert data into Working_joint table
cursor.executemany("""
INSERT OR IGNORE INTO Working_joint (name_of_the_joint, Exercise_ID,
part_of_the_body)
VALUES (?, ?, ?);
""", [
    ('Hip Joint', 1, 'Lower Body'),
    ('Knee Joint', 1, 'Lower Body'),
    ('Ankle Joint', 1, 'Lower Body'),
    ('Hip Joint', 2, 'Lower Body'),
    ('Knee Joint', 2, 'Lower Body'),
    ('Ankle Joint', 2, 'Lower Body'),
    ('Ankle Joint', 3, 'Lower Body'),
    ('Hip Joint', 4, 'Lower Body'),
    ('Knee Joint', 4, 'Lower Body'),
    ('Spine', 4, 'Lower Body'),
    ('Hip Joint', 5, 'Lower Body'),
    ('Knee Joint', 5, 'Lower Body'),
    ('Shoulder Joint', 6, 'Upper Body'),

```

```
('Elbow Joint', 6, 'Upper Body'),
('Shoulder Joint', 7, 'Upper Body'),
('Elbow Joint', 7, 'Upper Body'),
('Knee Joint', 8, 'Lower Body'),
('Ankle Joint', 8, 'Lower Body'),
('Shoulder Joint', 9, 'Upper Body'),
('Elbow Joint', 9, 'Upper Body'),
('Shoulder Joint', 10, 'Upper Body'),
('Elbow Joint', 10, 'Upper Body'),
('Hip Joint', 11, 'Lower Body'),
('Spine', 11, 'Upper Body')
])

# Commit and close the connection
conn.commit()
conn.close()
```

Evaluation

About learning:

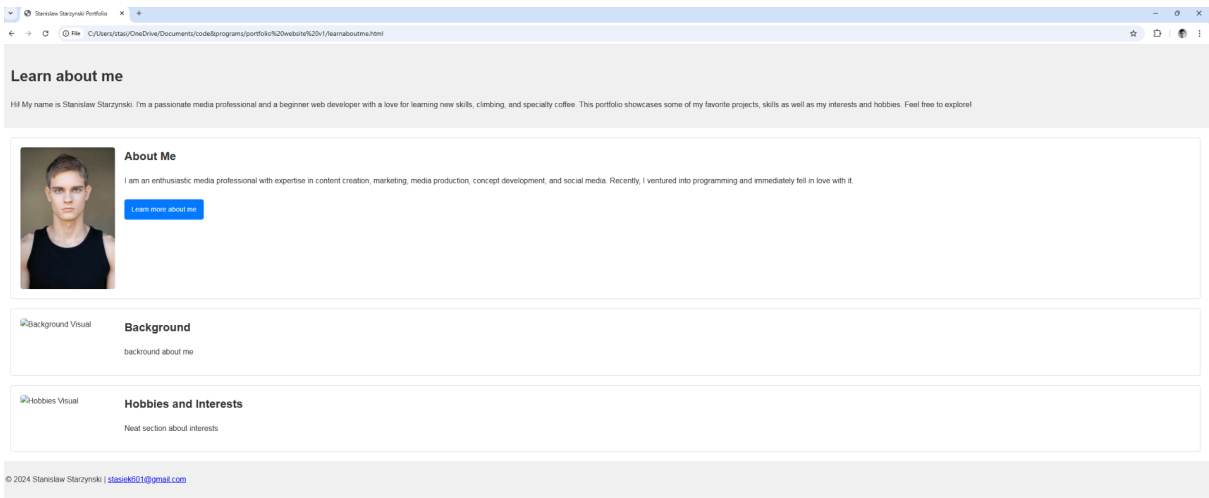
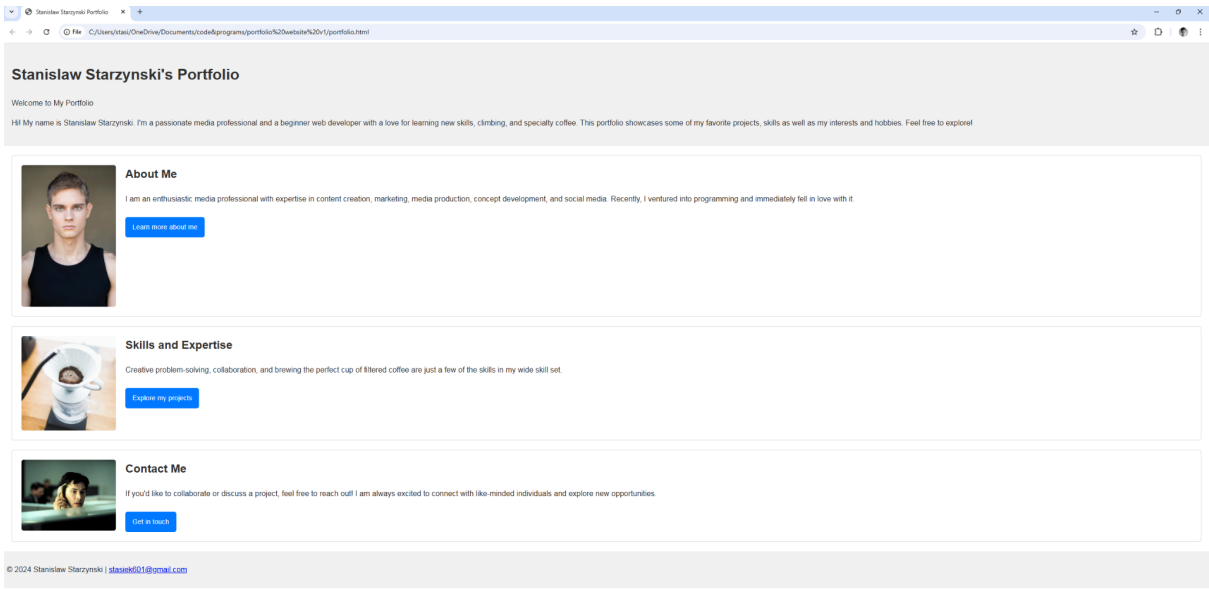
First week of the second part of the minor was pretty straight forward. I got to revise my SQL basics and later create my first database. The second party was the only thing I had problems with. I had lots of ideas for the database but I didn't know how to apply my ideas in practice. In the end, I managed to do a simplistic strength exercise database and an ERD describing the relationships within the database.

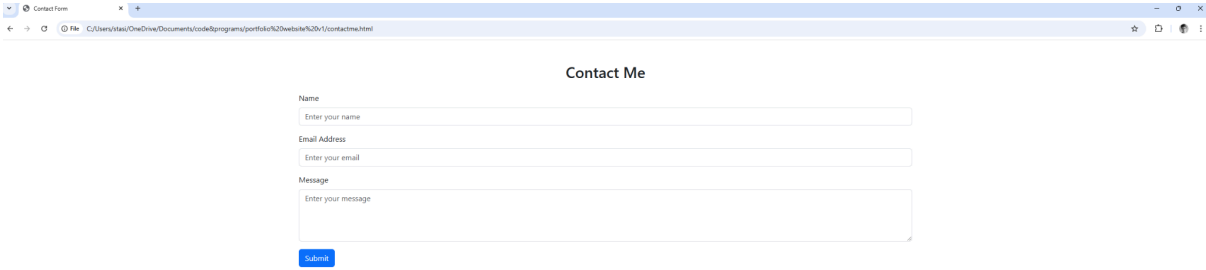
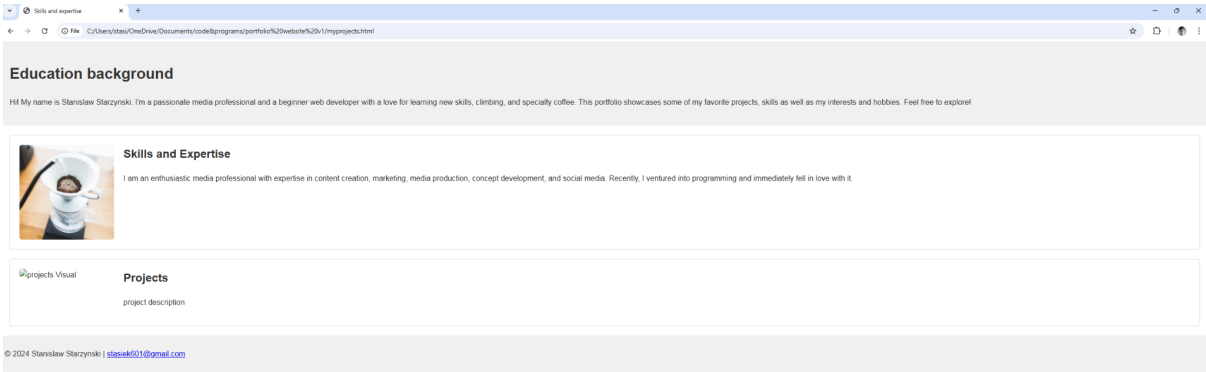
About relevance:

Since everything online depends on a database of some kind I believe learning to be proficient in SQL and knowing how to make an ERD for it may turn out to be very relevant in my professional future.

Week 2 - Web

Assignments - webpage





images	28.11.2024 14:41	Folder plików	
contactme.html	25.11.2024 17:42	Chrome HTML Do...	2 KB
learnaboutme.html	24.11.2024 17:30	Chrome HTML Do...	3 KB
myprojects.html	28.11.2024 13:29	Chrome HTML Do...	2 KB
portfolio.css	24.11.2024 17:27	CSS Source File	3 KB
portfolio.html	25.11.2024 16:57	Chrome HTML Do...	3 KB

```
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Metadata and styling -->
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Stanislaw Starzynski Portfolio</title>
  <link href="portfolio.css" rel="stylesheet">
</head>
<body>
  <!-- Header section -->
  <header>
    <h1>Stanislaw Starzynski's Portfolio</h1>
    <p>Welcome to My Portfolio</p>
    <p>
```

Hi! My name is Stanislaw Starzynski. I'm a passionate media professional and a beginner web developer with a love for learning new skills, climbing, and specialty coffee.

This portfolio showcases some of my favorite projects, skills as well as my interests and hobbies. Feel free to explore!

```
</p>
</header>

<!-- About Me section -->
<section>
  
  <div class="content">
    <h2>About Me</h2>
    <p>
      I am an enthusiastic media professional with expertise
in content creation, marketing, media production, concept development,
and social media. Recently, I ventured into programming and immediately
fell in love with it.
    </p>
    <a href="learnaboutme.html">Learn more about me</a>
  </div>
</section>

<!-- Skills and Expertise section -->
<section>
  
  <div class="content">
    <h2>Skills and Expertise</h2>
    <p>
      Creative problem-solving, collaboration, and brewing the
perfect cup of filtered coffee are just a few of the skills in my wide
skill set.
    </p>
    <a href="myprojects.html">Explore my projects</a>
  </div>
</section>

<!-- Contact Me section -->
<section>
  
  <div class="content">
    <h2>Contact Me</h2>
    <p>
      If you'd like to collaborate or discuss a project, feel
free to reach out!
```

I am always excited to connect with like-minded individuals and explore new opportunities.

</p>

Get in touch

</div>

</section>

<!-- Footer with contact information -->

<footer class="contact">

<p>

(c) 2024 Stanislaw Starzynski |

<!-- Email link for direct contact -->

stasiek601@gmail.com

</p>

</footer>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<!-- Metadata and styling -->

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Stanislaw Starzynski Portfolio</title>

<link href="portfolio.css" rel="stylesheet">

</head>

<body>

<!-- Header section -->

<header>

<h1>Learn about me</h1>

<p>

Hi! My name is Stanislaw Starzynski. I'm a passionate media professional and a beginner web developer with a love for learning new skills, climbing, and specialty coffee.

This portfolio showcases some of my favorite projects, skills as well as my interests and hobbies. Feel free to explore!

</p>

</header>

<!-- About Me section -->

<section>

 <div class="content">
 <h2>About Me</h2>
 <p>
 I am an enthusiastic media professional with expertise
 in content creation, marketing, media production, concept development,
 and social media. Recently, I ventured into programming and immediately
 fell in love with it.
 </p>
 Learn more about me
 </div>
</section>

<!-- Background section -->
<section>

 <div class="content">
 <h2>Background</h2>
 <p>
 background about me
 </p>
 </div>
</section>

<!-- Hobbies section -->
<section>

 <div class="content">
 <h2>Hobbies and Interests</h2>
 <p>
 Neat section about interests
 </p>
 </div>
</section>

<!-- Footer with contact information -->
<footer class="contact">
 <p>
 (c) 2024 Stanislaw Starzynski |
 <!-- Email link for direct contact -->
 stasiek601@gmail.com
 </p>
</footer>
</body>
</html>

```



```

<!DOCTYPE html>
<html lang="en">
<head>
 <!-- Metadata and styling -->
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width,
initial-scale=1.0">
 <title>Skills and expertise</title>
 <link rel="stylesheet" href="portfolio.css">
</head>
<body>
 <!-- Header section -->
 <header>
 <h1>Education background</h1>
 <p>
 Hi! My name is Stanislaw Starzynski. I'm a passionate media
 professional and a beginner web developer with a love for learning new
 skills, climbing, and specialty coffee.
 This portfolio showcases some of my favorite projects,
 skills as well as my interests and hobbies. Feel free to explore!
 </p>
 </header>

 <!-- Skills and expertise -->
 <section>

 <div class="content">
 <h2>Skills and Expertise</h2>
 <p>
 I am an enthusiastic media professional with expertise
 in content creation, marketing, media production, concept development,
 and social media. Recently, I ventured into programming and immediately
 fell in love with it.
 </p>
 </div>
 </section>

 <!-- Projects section -->
 <section>

 <div class="content">
 <h2>Projects</h2>
 <p>
 project description
 </p>
 </div>
 </section>

```

```

 </div>
 </section>

 <!-- Footer with contact information -->
 <footer class="contact">
 <p>
 (c) 2024 Stanislaw Starzynski |
 <!-- Email link for direct contact -->
 stasiek601@gmail.com
 </p>
 </footer>
</body>
</html>

```

```

<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width,
initial-scale=1.0">
 <title>Contact Form</title>
 <!-- Bootstrap CSS -->
 <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.mi
n.css" rel="stylesheet">
</head>
<body>
 <div class="container mt-5">
 <h2 class="text-center mb-4">Contact Me</h2>
 <form>
 <div class="mb-3">
 <label for="name" class="form-label">Name</label>
 <input type="text" class="form-control" id="name"
placeholder="Enter your name" required>
 </div>
 <div class="mb-3">
 <label for="email" class="form-label">Email
Address</label>
 <input type="email" class="form-control" id="email"
placeholder="Enter your email" required>
 </div>
 <div class="mb-3">
 <label for="message" class="form-label">Message</label>

```

```

 <textarea class="form-control" id="message" rows="4"
placeholder="Enter your message" required></textarea>
 </div>
 <button type="submit" class="btn
btn-primary">Submit</button>
</form>
</div>

<!-- Bootstrap JS Bundle -->
<script

src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bund
le.min.js"
></script>
<script>
 document.addEventListener('DOMContentLoaded', function() {

document.querySelector('form').addEventListener('submit', function(e) {
 alert('Looking forward to your message, ' +
document.querySelector('#name').value);
 e.preventDefault();
 });
 });
</script>
</body>
</html>

```

```

/* General styling for the body */
body {
 font-family: Arial, sans-serif;
 margin: 0;
 padding: 0;
 line-height: 1.6; /* Improves readability */
 color: #333; /* Dark text color for contrast */
}

/* Styling for the header */
header {
 padding: 20px; /* Adds space around the text */
 background-color: #f4f4f4; /* Light gray background for
header */
}

/* Styling for sections */
section {

```

```

 margin: 20px; /* Adds space outside the section */
 padding: 20px; /* Adds space inside the section */
 background-color: #fff; /* White background for content
 clarity */
 border: 1px solid #ddd; /* Light gray border */
 border-radius: 5px; /* Rounded corners for aesthetics */
 display: flex; /* Flex container to align image and content
 */

 gap: 20px; /* Space between image and content */
 align-items: flex-start; /* Aligns content to the top */
 }

 /* Styling for images */
 section img {
 width: 200px;
 height: auto; /* Maintain aspect ratio */
 border-radius: 5px; /* Optional: rounded corners for images
 */
 }

 /* Styling for the content container */
 .content {
 flex: 1; /* Allow text to take up the remaining space */
 text-align: left; /* Align text to the left */
 }

 /* Ensure section headers are included in the layout */
 .content h2 {
 margin-top: 0; /* Remove extra margin at the top of headings
 */
 }

 /* Styling for links */
 .content a {
 display: inline-block; /* Make the link act like a button */
 margin-top: 10px; /* Add space above the link */
 padding: 10px 15px; /* Add padding inside the link */
 color: white; /* White text color */
 background-color: #007BFF; /* Blue background */
 text-decoration: none; /* Remove underline */
 border-radius: 5px; /* Rounded corners */
 font-size: 0.9rem; /* Slightly smaller text */
 }

 /* Add hover effect for links */
 .content a:hover {

```

```
 background-color: #0056b3; /* Darker blue for hover */
 }

 /* Footer styling for contact section */
 .contact {
 text-align: left;
 padding: 10px;
 background-color: #f4f4f4; /* Matches the header for
consistency */
 margin-top: 20px; /* Adds space between the last section and
footer */
 }
```

## Evaluation

### About learning:

This week I struggled making a website. Understanding HTML comes harder to me than learning Python or SQL. I used notes to code which I later realised is not as handy as a dedicated visual studio. At the end of the week I was happy to understand the structure better and even apply some extra functions and design to my website.

On the other hand I am not satisfied with the content on the website. At the moment it looks more like a draft. Taking additional responsibilities on my shoulders significantly decreased the amount of time I can spend on coding.

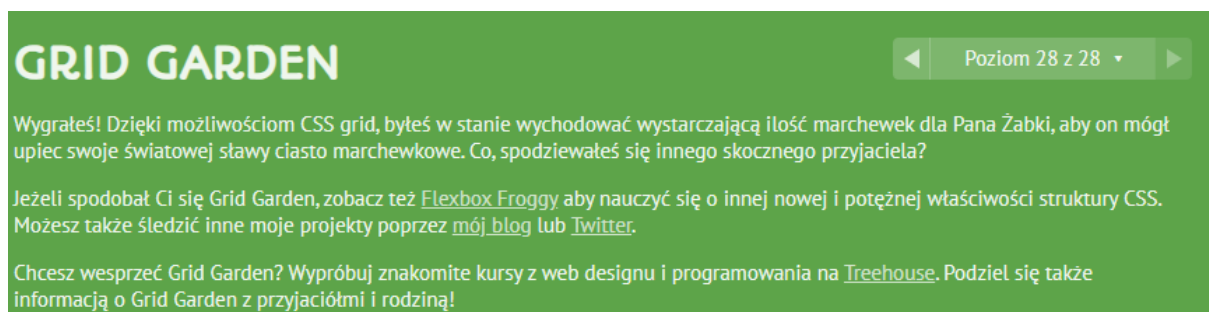
### About relevance:

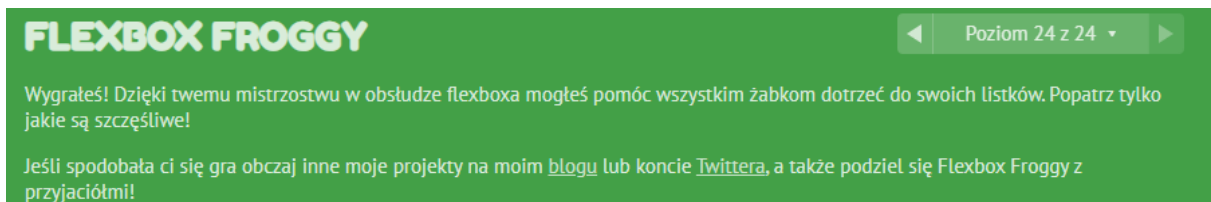
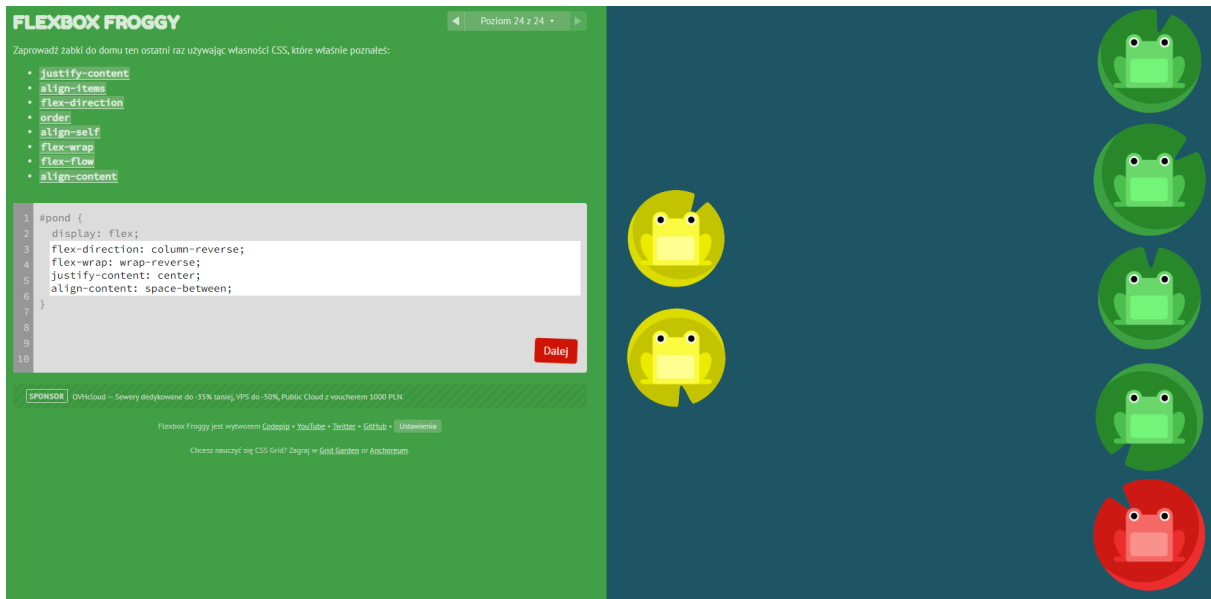
In the future I am aspiring to have at least a personal website where I can showcase my projects, work, and interests. The assignment was essentially a draft of what I would like to host online in some time.

# Week 3 - Web

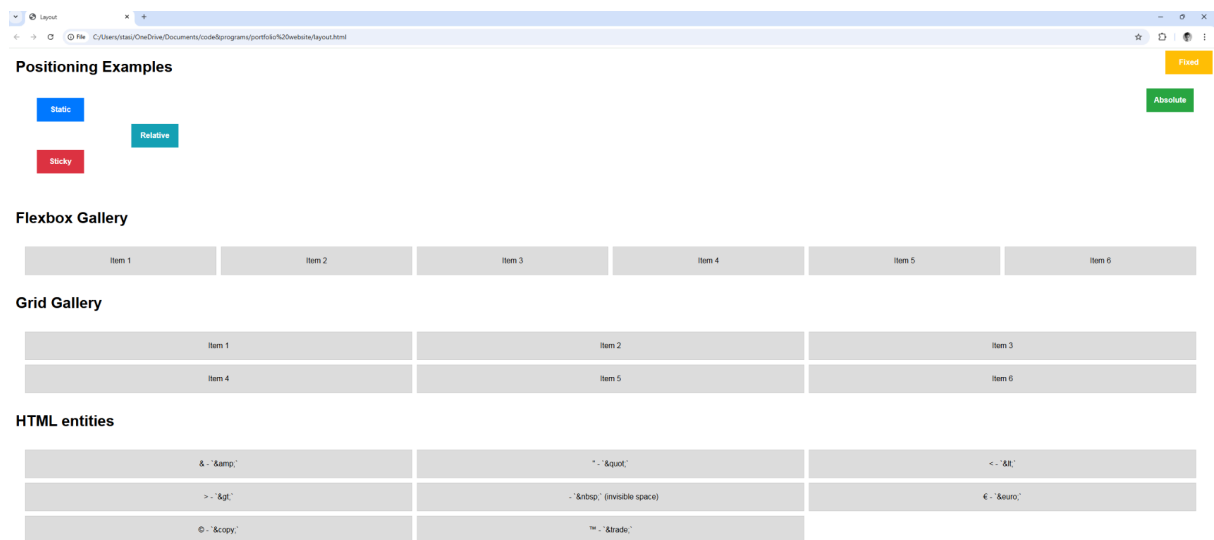
## Assignments - Grid and Flex

### CSS Grid Garden and Flexbox Froggy





## Layout



```

<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width,
initial-scale=1.0">
 <title>Layout</title>

```

```

 <link href="css/layout.css" rel="stylesheet">
</head>
<body>
 <h1>Positioning Examples</h1>
 <div class="positioning-container">
 <div class="position-static item">Static</div>
 <div class="position-relative item">Relative</div>
 <div class="position-absolute item">Absolute</div>
 <div class="position-fixed item">Fixed</div>
 <div class="position-sticky item" style="top:
10px;">Sticky</div>
 </div>

 <h1>Flexbox Gallery</h1>
 <div class="flexbox-gallery">
 <div class="item">Item 1</div>
 <div class="item">Item 2</div>
 <div class="item">Item 3</div>
 <div class="item">Item 4</div>
 <div class="item">Item 5</div>
 <div class="item">Item 6</div>
 </div>

 <h1>Grid Gallery</h1>
 <div class="grid-gallery">
 <div class="item">Item 1</div>
 <div class="item">Item 2</div>
 <div class="item">Item 3</div>
 <div class="item">Item 4</div>
 <div class="item">Item 5</div>
 <div class="item">Item 6</div>
 </div>

 <h1>HTML entities</h1>
 <div class="grid-gallery">
 <div class="item">& - `&`</div>
 <div class="item">" - `&quot;`</div>
 <div class="item">< - `&lt;`</div>
 <div class="item">> - `&gt;`</div>
 <div class="item"> - `&nbsp;` (invisible
space)</div>
 <div class="item">€ - `&euro;`</div>
 <div class="item">© - `&copy;`</div>
 <div class="item">™ - `&trade;`</div>
 </div>

</body>

```



```
</html>
```

```
/* Flexbox Gallery */
.flexbox-gallery {
 display: flex;
 justify-content: space-between; /* Evenly space the items */
 flex-wrap: wrap;
 gap: 10px; /* Adds spacing between items */
 padding: 20px;
}
.flexbox-gallery .item {
 background-color: #e0e0e0; /* Light gray background for items */
 padding: 20px; /* Space inside each item */
 flex: 1 1 calc(16% - 10px); /* Flexibly size items, ensuring equal
spacing */
 text-align: center; /* Center-align the text */
 border: 1px solid #ccc; /* Border for individual items */
 box-sizing: border-box; /* Includes padding and border in size
calculation */
}

/* Grid Gallery */
.grid-gallery {
 display: grid;
 grid-template-columns: repeat(3, 1fr); /* 3 equal columns */
 gap: 10px; /* Space between grid items */
 padding: 20px;
}
.grid-gallery .item {
 background-color: #e0e0e0; /* Light gray background for items */
 padding: 20px; /* Space inside each item */
 text-align: center; /* Center-align the text */
 border: 1px solid #ccc; /* Border for individual items */
 box-sizing: border-box; /* Includes padding and border in size
calculation */
}

/* Positioning Examples */
.positioning-container {
 margin: 20px;
 padding: 20px;
 position: relative; /* Needed for absolute positioning to work
within this container */
}
```

```

 height: 200px;
 }

.positioning-container .item {
 width: 100px;
 height: 50px;
 text-align: center;
 line-height: 50px;
 margin: 5px;
 color: white;
 font-weight: bold;
}

/* Specific Positioning */
.position-static {
 position: static; /* Default behavior */
 background-color: #007bff;
}

.position-relative {
 position: relative;
 left: 200px; /* Moves 200px to the right from its original position
*/
 background-color: #17a2b8;
}

.position-absolute {
 position: absolute; /* Positioned relative to the nearest positioned
ancestor */
 top: 0px;
 right: 0px;
 background-color: #28a745;
}

.position-fixed {
 position: fixed; /* Stays in the same place relative to the viewport
*/
 top: 0;
 right: 0;
 background-color: #ffc107;
 z-index: 1000;
}

.position-sticky {
 position: sticky; /* Sticks to the top when scrolling, within its
parent container */

```

```

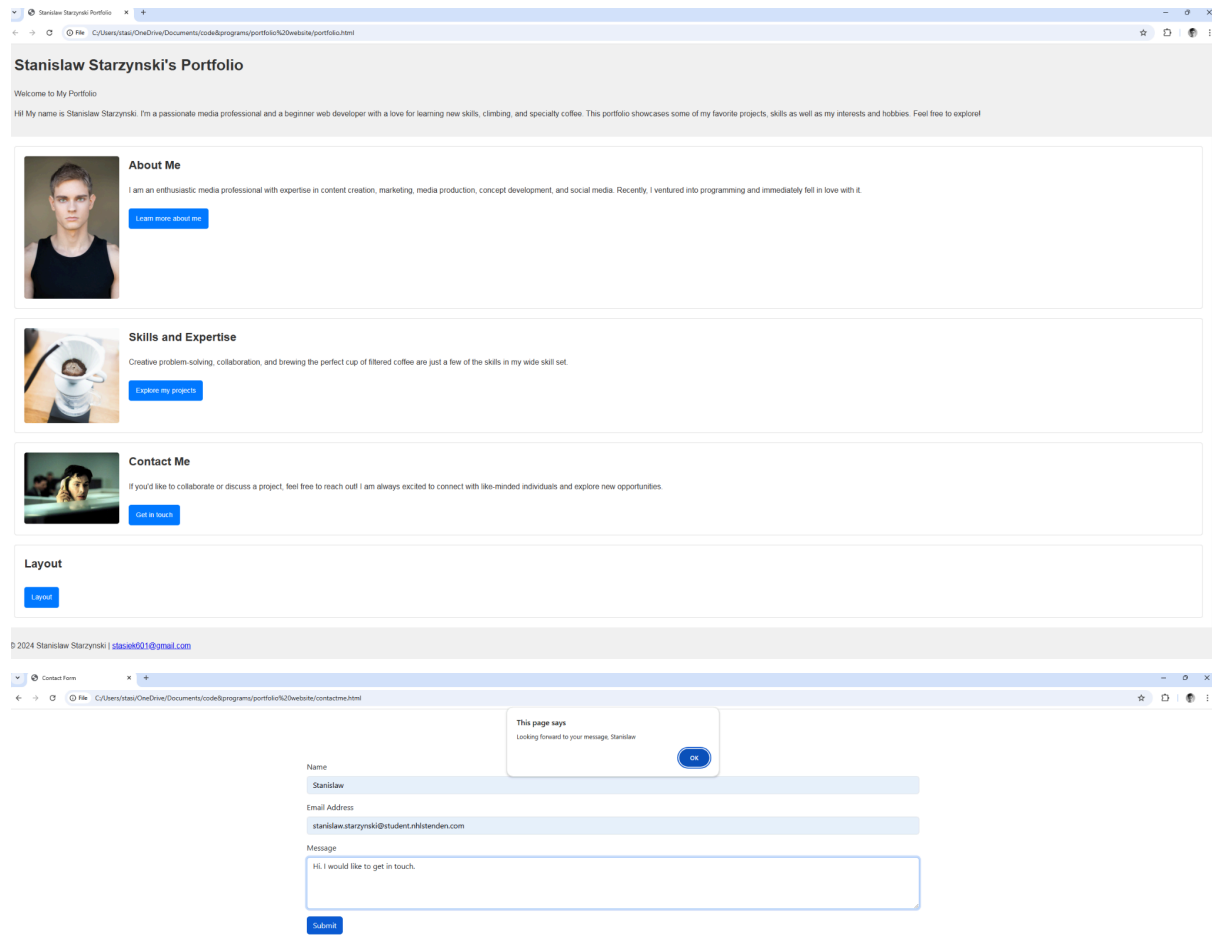
 top: 10px;
 background-color: #dc3545;
}

/* Additional Styling */
body {
 font-family: Arial, sans-serif;
 margin: 20px;
}
h2 {
 text-align: center;
}

/*
margine - changes the margine space around the content
width - changes the width of the content printed on the website
height - changes the hight
padding - changes the space between content and the outside border
color - changes the color
background-color - changes the background color
border - creates a visible boundary around an element

Positions:
Static - Default position, no special positioning.
Relative - Moved a little from its normal position.
Absolute - Completely removed and placed somewhere else.
Fixed - Stays fixed relative to the viewport.
Sticky - Moves normally but sticks when it hits a boundary.
Inherit - Copies position from the parent element.
*/

```



CRUD stands for Create, Read, Update, Delete.

## C - Create

This is when you add new data to the database.

In SQL, this is done using an **INSERT** statement.

## R - Read

This is when you view or retrieve data from the database.

In SQL, this is done using a **SELECT** statement.

## U - Update

This is when you modify existing data in the database.

In SQL, this is done using an **UPDATE** statement.

## D - Delete

This is when you remove data from the database.

In SQL, this is done using a **DELETE** statement.

## Evaluation

### About learning:

Playing flexbox and grid games was fun and educational at the same time. While the games and reading Mondriaan were pretty easy, putting flexbox and grid in use was a little bit harder for me.

It wasn't the first time I encountered CRUD and it was already clear to me.

### About relevance:

Flex and grid greatly improve my ability to manipulate the look of the website, enhancing my design abilities. Even though design isn't my strong suit, it is an important part of my studies and the newly learned functionalities are certainly going to help me elevate my designing skills of a website.

## Week 4 - Web

### Assignments - Flask

#### Vanity Plates

##### **vanity.db:**

- This is the SQLite database where license plate data is stored.
- It contains a table (**numberplates**) that holds all the plates submitted through the website.

##### **app.py:**

- This is the main Python file for the Flask application.
- It contains the routes (URLs) and logic to handle requests (what happens when a user submits a form or visits a page).

##### **database.py:**

- A helper Python file for managing database operations like fetching data (**READ**) and adding data (**CREATE**).

##### **\_\_pycache\_\_:**

- A folder automatically created by Python to store compiled versions of the code for faster execution.

#### static:

- A folder where static files like images, CSS (styling), or JavaScript (frontend functionality) would be stored.

#### templates:

- A folder for HTML files (webpages).
- These templates define how the website looks and work with Flask to display data dynamically.
- Files inside:
  - `base.html`: The layout template that other pages (like `valid.html`) extend.
  - `index.html`: The homepage where all license plates are listed.
  - `invalid.html`: A page displayed if the license plate is invalid.
  - `valid.html`: A page displayed if the license plate is valid.

### app.py

This file contains the main Flask application. Flask is a web framework that makes it easy to create a web server and handle requests and responses.

#### Key Components in `app.py`:

##### Setup and Configuration:

```
from flask import Flask, render_template, request
app = Flask(__name__)
app.config['TEMPLATES_AUTO_RELOAD'] = True
```

- This initializes the Flask app and enables auto-reloading of templates when they change.

##### Database Connection:

```
from database import Database
db = Database('vanity.db')
```

- It imports the `Database` class from `database.py` and connects to a SQLite database file named `vanity.db`.

##### Handling Web Routes:

```
@app.route('/', methods=['GET', 'POST'])
```

- Defines the behavior of the home page (/ route). Depending on the request type:
  - **GET Request:**
    - Displays the list of plates stored in the database.
  - **POST Request:**
    - Validates a submitted license plate and either:
      - Stores it in the database (if valid).
      - Shows an error page (if invalid).

### Running the Server:

```
if __name__ == "__main__":
 app.run(debug=True)
```

- Starts the web server and allows you to visit your application in a web browser

### Base.html

#### <head> Section

```
<head>
 <title>{% block title %}Number plate validator{% endblock %}</title>
 <!-- Bootstrap CSS -->
 <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.m
in.css">
</head>
```

The `<head>` section contains metadata and links to stylesheets or other resources for the webpage.

#### <title>:

```
<title>{% block title %}Number plate validator{% endblock %}</title>
```

The title of the webpage is displayed in the browser tab.

`{% block title %}` and `{% endblock %}` are placeholders where child templates (like `index.html`) can provide their own titles.

If a child template doesn't override the title, the default is "Number plate validator".

## <body> Section

The <body> section contains the visible content of the webpage.

### Navigation Bar

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
 Number plate validator
</nav>
```

This code creates a **navigation bar** at the top of the page using Bootstrap.

- **<nav>**: Defines the navigation bar.
- **navbar navbar-expand-lg navbar-light bg-light**: These are Bootstrap classes that style the navigation bar (e.g., light background, expandable on larger screens).
- **<a class="navbar-brand" href="/">**:
  - This is the clickable title of the website, which links back to the homepage (/ route).

### Main Content Area

```
<div class="container mt-5">
 <!-- Image -->

 {% block content %}
 {% endblock %}
</div>
```

#### <div class="container mt-5">:

- Creates a container (a centered box) to hold the main content of the page.
- **mt-5**: Adds a top margin (spacing above the container).

### Image:

```

```

- Displays an image named **pizza.png** stored in the **static** folder.
- **url\_for('static', filename='pizza.png')** dynamically generates the correct URL for the image.

#### {% block content %} {% endblock %}:

- This is a placeholder for additional content provided by child templates.
- For example:



- `index.html` could replace this block with a list of license plates.
- `valid.html` could replace it with a success message.

## index.html

```
{% extends "base.html" %}
```

- This line means that `index.html` is using `base.html` as its "parent" template.
- All the shared elements from `base.html` (like the navigation bar, Bootstrap styling, and JavaScript) will automatically appear in this file.
- `index.html` only defines the specific content for the `content` block.

```
{% block content %}
```

This is where the content specific to the home page (`index.html`) goes. It overrides the `{% block content %}` placeholder in `base.html`.

```
<h1 class="text-center mb-4">Number plate validator</h1>
```

- Displays a large heading: **"Number plate validator"**.
- **Classes:**
  - `text-center`: Centers the text horizontally.
  - `mb-4`: Adds a margin below the heading to create some space.

```
<form method="POST" action="/">
 <div class="form-group">
 <label for="plate">Enter your number plate:</label>
 <input type="text" class="form-control" id="plate" name="plate"
placeholder="Enter plate">
 </div>
 <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

This section is a **form** that lets users enter a number plate and submit it to the server.

Key elements:

- **<form>:**
  - `method="POST"`: Specifies that the form will send the data using a POST request when the user clicks the Submit button.
  - `action="/"`: The form data will be sent to the `/` route, which is handled by the Flask app in `app.py`.
- **<div class="form-group">:**
  - A Bootstrap class that groups the label and input field together.

- **<label for="plate">:**
  - The text label "Enter your number plate."
  - **for="plate"** links it to the input field with **id="plate"**.
- **<input>:**
  - A text box for the user to type in their number plate.
  - Attributes:
    - **class="form-control"**: Adds Bootstrap styling to the input box.
    - **id="plate"**: A unique identifier for this input field.
    - **name="plate"**: The name used to send the data to the Flask app.
    - **placeholder="Enter plate"**: Displays placeholder text inside the input field.
- **<button>:**
  - A button to submit the form.
  - **btn btn-primary**: Bootstrap classes that style the button (e.g., blue color).

```
<div class="mt-5">
 <h2>Previously entered number plates:</h2>

 {% for plate in plates %}
 {{ plate[1] }}
 {% endfor %}

</div>
```

This section displays a list of number plates that were previously submitted and saved in the database.

**<div class="mt-5">:**

- A container for this section.
- **mt-5**: Adds a top margin for spacing.

```

 {% for plate in plates %}
 {{ plate[1] }}
 {% endfor %}

```

**{% for plate in plates %}:**

- A loop that goes through all the number plates passed from the Flask app (**plates**).
- For each **plate**, the second column in the database row (**plate[1]**) is displayed in a list item (**<li>**).

**{{ plate[1] }}:**

- Inserts the number plate text into the HTML.

```
{% endblock %}
```

This marks the end of the `content` block, completing the specific content for `index.html`.

### `valid.html` and `invalid.html`

These files display if the input was respectively, valid or invalid.

### `database.py`:

```
import sqlite3
from itertools import groupby

class Database:
 def __init__(self, db_path):
 self.db_path = db_path

 def get_connection(self):
 # Establishes a connection to the SQLite database
 conn = sqlite3.connect(self.db_path)
 conn.row_factory = sqlite3.Row # Sets the row_factory to
 retrieve rows as dictionaries
 return conn

 def get_numberPlates(self):
 with self.get_connection() as conn:
 cursor = conn.cursor()
 # Executes a SELECT query to retrieve all rows from the
 "numberplates" table
 cursor.execute("SELECT * FROM numberplates")
 return cursor.fetchall() # Returns all fetched rows as a
 list of dictionaries

 def add_numberPlates(self, plate):
 with self.get_connection() as conn:
 # Executes an INSERT query to add a new number plate to the
 "numberplates" table
 conn.execute("INSERT INTO numberplates (plate) VALUES (?)",
 (plate,))
```

`get_connection()`: Opens a connection to the SQLite database.

**get\_numberPlates()**: Fetches all number plates from the database.

**add\_numberPlates()**: Adds a new number plate to the database.

## Evaluation

### About learning:

Learning about Flask seemed overwhelming at the beginning but with time I realised how much functionality it adds to the websites I can make in the future. It helps in making more creative projects and links python (my favourite programming language) which helped motivate me to get a better grasp on it.

### About relevance:

As a creative media student I find Flask very relevant to my studies. I believe with repetition this framework can also become easy in use. I am definitely going to use Flask in my future web projects.

## Week 7+ - Final Project

### The idea

#### **TRAINING WEBSITE**

Product runs on a website. It is a tool/environment to help a person improve their flexibility. Learn about FLEXIBILITY, the basic flexibility positions. Find out what workout plan suits best your current flexibility level. Get a template to follow in order to improve your chosen flexibility position. Create a profile to track your progress and to be able to access the extra (paid for) content. A contact form to reach me included.

Input: fill a form assessing your flexibility level, upload your progress,

Output: workout plan for the given level of flexibility, progress library

Workout plan templates (custom made) included in SQL DB, customizable profile SQL DB (basic information, progress) [NOT SURE if this implementation of the SQL DB makes sense]

#### **MoSCoW:**

-Must have: presentation of the basic flexibility positions, workout plans for various levels of flexibility leading to the flexibility positions, survey linking a person to the workout plan for their given level of flexibility, contact form

-Should have: profiles, progress tracking (ability to upload pictures with description)

-Could have: paywall to access extra content

-Won't have: comment section, customisable workout plans templates, videos (unless it's included in the next version of paid for content)

# My website's structure

Th test_app.py	21.01.2025 15:21	Python file	7 KB
requirements.txt	21.01.2025 13:58	Dokument tekstowy	2 KB
Procfile	21.01.2025 13:59	Plik	1 KB
Th app.py	21.01.2025 13:58	Python file	13 KB
templates	16.01.2025 15:36	Folder plików	
static	21.01.2025 14:06	Folder plików	
instances	21.01.2025 14:01	Folder plików	
__pycache__	21.01.2025 15:21	Folder plików	
.pytest_cache	21.01.2025 15:14	Folder plików	


## templates:

about.html	21.01.2025 14:01	Chrome HTML Do...	2 KB
contact.html	21.01.2025 14:02	Chrome HTML Do...	2 KB
layout.html	21.01.2025 14:02	Chrome HTML Do...	2 KB
login.html	21.01.2025 14:02	Chrome HTML Do...	1 KB
positions.html	21.01.2025 14:03	Chrome HTML Do...	5 KB
profile.html	21.01.2025 14:03	Chrome HTML Do...	1 KB
register.html	21.01.2025 14:03	Chrome HTML Do...	1 KB
survey.html	21.01.2025 14:04	Chrome HTML Do...	8 KB

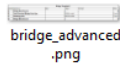
## static:

images	21.01.2025 14:06	Folder plików	
uploads	21.01.2025 14:05	Folder plików	
styles.css	21.01.2025 14:04	CSS Source File	2 KB

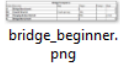
## images:



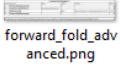
bridge.png



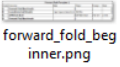
bridge\_advanced.png




bridge\_beginner.png



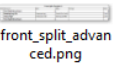
forward\_fold\_advanced.png



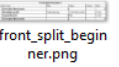
forward\_fold\_beginner.png




forward-fold.png



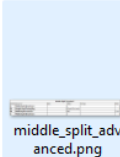
front\_split\_advanced.png



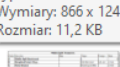
front\_split\_beginner.png




front-split.png



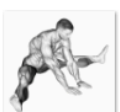
middle\_split\_advanced.png



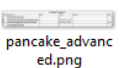
middle\_split\_beginner.png



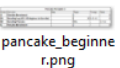
middle-split.png



pancake.png



pancake\_advanced.png



pancake\_beginner.png

## instances:

database.db	21.01.2025 14:00	SQLite	32 KB
-------------	------------------	--------	-------

app.py:

```
Import necessary modules
from flask import Flask, render_template, request, redirect, url_for,
session, flash
from flask_sqlalchemy import SQLAlchemy
from flask_mail import Mail, Message
from werkzeug.security import generate_password_hash,
check_password_hash
from functools import wraps
import os

Initialize Flask application
app = Flask(__name__)

Configuration
app.secret_key = os.environ.get('SECRET_KEY', 'your_default_secret_key')

Set database file path using SQLAlchemy
basedir = os.path.abspath(os.path.dirname(__file__))
app.config['SQLALCHEMY_DATABASE_URI'] =
f'sqlite:/// {os.path.join(basedir, "instance", "database.db")}'

Configure email settings
app.config['MAIL_SERVER'] = 'smtp.gmail.com'
app.config['MAIL_PORT'] = 587
app.config['MAIL_USE_TLS'] = True
app.config['MAIL_USE_SSL'] = False
app.config['MAIL_USERNAME'] = os.environ.get('MAIL_USERNAME')
app.config['MAIL_PASSWORD'] = os.environ.get('MAIL_PASSWORD')
app.config['MAIL_DEFAULT_SENDER'] =
os.environ.get('MAIL_DEFAULT_SENDER')

Initialize extensions
db = SQLAlchemy(app)
mail = Mail(app)

Allowed file extensions for photo uploads
ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'gif'}

=====
DATABASE MODELS
=====

user in the system
class User(db.Model):
```

```

 id = db.Column(db.Integer, primary_key=True)
 username = db.Column(db.String(80), unique=True, nullable=False)
 email = db.Column(db.String(120), unique=True, nullable=False)
 password = db.Column(db.String(200), nullable=False)

user progress entries
class Progress(db.Model):
 id = db.Column(db.Integer, primary_key=True)
 user_id = db.Column(db.Integer, db.ForeignKey('user.id'),
 nullable=False)
 photo = db.Column(db.String(120), nullable=False)
 description = db.Column(db.Text, nullable=True)

represents flexibility training templates
class TrainingTemplate(db.Model):
 id = db.Column(db.Integer, primary_key=True)
 position_name = db.Column(db.String(100), nullable=False)
 template_level = db.Column(db.String(20), nullable=False)
 image_path = db.Column(db.String(255), nullable=False)
 description = db.Column(db.Text, nullable=True)

=====
HELPER FUNCTIONS
=====

checks if a file is valid based on its extension
def allowed_file(filename):
 return '.' in filename and filename.rsplit('.', 1)[1].lower() in
ALLOWED_EXTENSIONS

decorator to ensure a user is logged in before accessing certain
routes
def login_required(f):
 @wraps(f)
 def decorated_function(*args, **kwargs):
 # check if 'user_id' exists in the session
 if 'user_id' not in session:
 flash('Please log in to access this page.')
 return redirect(url_for('login'))
 return f(*args, **kwargs)
 return decorated_function

=====
ROUTES
=====
@app.route("/")

```

```

def about():
 """Displays the About page."""
 return render_template("about.html")

@app.route("/positions")
def positions():
 """Displays the positions page."""
 return render_template("positions.html")

@app.route("/survey")
def survey():
 """Displays the survey page."""
 return render_template("survey.html")

@app.route("/contact")
def contact():
 """Displays the contact page."""
 return render_template("contact.html")

@app.route('/submit-contact', methods=['POST'])
def submit_contact():
 """Handles contact form submission."""
 if request.method == 'POST':
 email = request.form['email']
 message = request.form['message']

 # validate form data
 if not email or not message:
 flash('All fields are required.', 'error')
 return redirect(url_for('contact'))

 try:
 # send email
 msg = Message('Contact Form Submission',
recipients=[app.config['MAIL_USERNAME']])
 msg.body = f"From: {email}\n\nMessage:\n{message}"
 mail.send(msg)
 return redirect(url_for('contact', success=1))
 except Exception as e:
 print(f"Error: {e}")
 flash('An error occurred while sending your message. Please
try again.', 'error')
 return redirect(url_for('contact'))

@app.route('/register', methods=['GET', 'POST'])
def register():

```



```

if request.method == 'POST':
 try:
 username = request.form['username']
 email = request.form['email']
 password = request.form['password']

 # validate inputs
 if not username or not email or not password:
 flash('All fields are required.')
 return redirect(url_for('register'))

 if len(password) < 8:
 flash('Password must be at least 8 characters long.')
 return redirect(url_for('register'))

 if User.query.filter_by(email=email).first():
 flash('Email already registered.')
 return redirect(url_for('register'))

 # save new user
 hashed_password = generate_password_hash(password)
 new_user = User(username=username, email=email,
password=hashed_password)
 db.session.add(new_user)
 db.session.commit()
 flash('Registration successful! Please log in.')
 return redirect(url_for('login'))
 except Exception as e:
 db.session.rollback()
 print(f"Error During Registration: {e}")
 flash('An error occurred during registration. Please try
again.')
 return redirect(url_for('register'))

return render_template('register.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
 if request.method == 'POST':
 email = request.form['email']
 password = request.form['password']

 user = User.query.filter_by(email=email).first()
 if user and check_password_hash(user.password, password):
 session['user_id'] = user.id
 session['username'] = user.username

```

```

 flash('Login successful!')
 return redirect(url_for('profile'))

 flash('Invalid email or password.')
 return redirect(url_for('login'))

 return render_template('login.html')

@app.route('/logout')
def logout():
 session.clear()
 flash('Logged out successfully.')
 return redirect(url_for('login'))

@app.route('/profile', methods=['GET', 'POST'])
@login_required
def profile():
 if request.method == 'POST':
 photo = request.files.get('photo') # Use `.get()` to avoid
 # crashes if 'photo' is not in request
 description = request.form.get('description', '')

 # Validate the uploaded photo
 if photo and allowed_file(photo.filename):
 # Create uploads directory if it doesn't exist
 upload_folder = os.path.join('static', 'uploads')
 os.makedirs(upload_folder, exist_ok=True)

 # Save the file
 photo_path = os.path.join(upload_folder, photo.filename)
 photo.save(photo_path)

 # Save the progress entry to the database
 new_progress = Progress(user_id=session['user_id'],
 photo=photo_path, description=description)
 db.session.add(new_progress)
 db.session.commit()

 flash('Progress photo added successfully!')
 else:
 flash('Invalid file type. Please upload a valid image.')

 # Fetch user's progress entries from the database
 progress_entries =
 Progress.query.filter_by(user_id=session['user_id']).all()
 return render_template('profile.html', username=session['username'],

```

```

progress_entries=progress_entries)

@app.errorhandler(404)
def page_not_found(e):
 """handles 404 errors with a custom page."""
 return render_template('404.html'), 404

@app.route('/submit-survey', methods=['POST'])
def submit_survey():
 position = request.form.get('position')
 level = None

 if position == "Forward Fold":
 question = request.form.get('forward_fold_question')
 level = "Advanced" if question == "yes" else "Beginner"
 elif position == "Bridge":
 question = request.form.get('bridge_question')
 level = "Advanced" if question == "yes" else "Beginner"
 elif position == "Front Split":
 question_1 = request.form.get('front_split_question_1')
 if question_1 == "no":
 level = "Beginner"
 else:
 question_2 = request.form.get('front_split_question_2')
 level = "Advanced" if question_2 == "yes" else "Beginner"
 elif position == "Middle Split":
 question = request.form.get('middle_split_question')
 level = "Advanced" if question == "less_30cm" else "Beginner"
 elif position == "Pancake":
 question = request.form.get('pancake_question')
 level = "Advanced" if question == "yes" else "Beginner"

 template = TrainingTemplate.query.filter_by(position_name=position,
template_level=level).first()

 return render_template('survey.html', template=template)

=====
RUN THE APP
=====
if __name__ == '__main__':
 with app.app_context():
 db.create_all()
 # Manually seed data
 if TrainingTemplate.query.count() == 0:

```

```

templates = [
 TrainingTemplate(
 position_name='Forward Fold',
 template_level='Beginner',

image_path='static/templates/forward_fold_beginner.png',
 description='Forward Fold template for beginners.'
),
 TrainingTemplate(
 position_name='Forward Fold',
 template_level='Advanced',

image_path='static/templates/forward_fold_advanced.png',
 description='Forward Fold template for advanced
practitioners.'
),
 TrainingTemplate(
 position_name='Bridge',
 template_level='Beginner',
 image_path='static/templates/bridge_beginner.png',
 description='Bridge template for beginners.'
),
 TrainingTemplate(
 position_name='Bridge',
 template_level='Advanced',
 image_path='static/templates/bridge_advanced.png',
 description='Bridge template for advanced
practitioners.'
),
 TrainingTemplate(
 position_name='Front Split',
 template_level='Beginner',

image_path='static/templates/front_split_beginner.png',
 description='Front Split template for beginners.'
),
 TrainingTemplate(
 position_name='Front Split',
 template_level='Advanced',

image_path='static/templates/front_split_advanced.png',
 description='Front Split template for advanced
practitioners.'
),
 TrainingTemplate(
 position_name='Middle Split',

```

```

 template_level='Beginner',

image_path='static/templates/middle_split_beginner.png',
 description='Middle Split template for beginners.'
),
 TrainingTemplate(
 position_name='Middle Split',
 template_level='Advanced',

image_path='static/templates/middle_split_advanced.png',
 description='Middle Split template for advanced
practitioners.'
),
 TrainingTemplate(
 position_name='Pancake',
 template_level='Beginner',
 image_path='static/templates/pancake_beginner.png',
 description='Pancake template for beginners.'
),
 TrainingTemplate(
 position_name='Pancake',
 template_level='Advanced',
 image_path='static/templates/pancake_advanced.png',
 description='Pancake template for advanced
practitioners.'
),
]
db.session.bulk_save_objects(templates)
db.session.commit()
print("Training templates have been seeded.")
app.run(debug=True, host='0.0.0.0', port=5000)

```

styles.css:

```

/* General Reset */
body {
 /* Use a flexbox layout for the body to allow content to be
vertically aligned and the footer to stay at the bottom. */
 display: flex;
 /* Stack child elements vertically. */
 flex-direction: column;
 margin: 0;
 /* Ensure the body takes up the full height of the viewport. */
 min-height: 100vh;
 font-family: 'Open Sans', sans-serif; /* Apply font to the entire
body */
}

```

```

}

header {
 /* Set a dark background color for the header. */
 background: #333;
 /* Use white text color. */
 color: #fff;
 /* Add padding around the header content. */
 padding: 1rem 0;
 text-align: center;
}

header h1 {
 margin: 10;
}

nav ul {
 list-style: none;
 padding: 0;
 margin: 0;
 display: flex;
 /* Center-align the navigation links. */
 justify-content: center;
 background: #444;
}

nav ul li {
 /* Add horizontal spacing between navigation items. */
 margin: 0 1rem;
}

nav ul li a {
 /* Remove underlines from links. */
 text-decoration: none;
 color: #fff;
 font-weight: bold;
 transition: color 0.3s ease;
}

/* Add hover effect for navigation links. */
nav ul li a:hover {
 color: #ffa500;
}

main {
 padding: 1.5rem;
}

```

```

/* Wrapper for the content above the footer */
.wrapper {
 flex: 1; /* Pushes the footer to the bottom */
}

/* Footer styling */
footer {
 background-color: #333;
 color: #fff;
 text-align: center;
 /* Add padding around the footer content. */
 padding: 1rem 0;
 /* Ensure the footer stays at the bottom of the page. */
 margin-top: auto; /* Ensures it stays at the bottom */
}

```

about.html:

```

{% extends "layout.html" %}

{% block title %}About Us{% endblock %}

{% block content %}
<section id="about">
 <h2>About Flexibility Training</h2>
 <p>Welcome to the Flexibility Training Website! Our platform is
designed for individuals with some prior experience in training who are
looking to enhance their range of motion in foundational flexibility
positions. Whether you're refining your Forward Fold, Bridge, Front
Split, Middle Split, or Pancake, we're here to support your journey.</p>
 <p>Through our interactive survey, you'll receive personalized
training templates tailored to your current level and goals. Use these
templates to structure your practice effectively and track your progress
over time.</p>
 <p>Our progress tracking feature allows you to monitor improvements
easily. Simply log into your account (or register if you're new), upload
your progress photos, and add descriptions. Revisit your profile
regularly to celebrate your achievements and ensure you're on the right
path.</p>
 <p>We understand that some concepts and specialized vocabulary may
seem unfamiliar. If you're unsure about a position or term, don't worry!
Reach out to us through the contact form, and we'll be happy to
assist.</p>
 <p>Take your flexibility training to the next level with us. Join
today and work toward mastering these essential positions!</p>

```

```


</section>
{% endblock %}

```

contact.html:

```

{% extends "layout.html" %}

{% block title %}Contact Us{% endblock %}

{% block content %}

<section id="contact">
 <h2>Contact Us</h2>
 <p>If you have any questions or need assistance, please get in touch
with us using the form below.</p>
 <form action="/submit-contact" method="post">
 <label for="email">Your Email:</label>
 <input type="email" id="email" name="email" required>

 <label for="message">Your Message:</label>
 <textarea id="message" name="message" rows="5"
required></textarea>

 <button type="submit">Send Message</button>
 </form>
 <p>You can also reach us at support@flexibil
itytraining.com.</p>
</section>

<script>
 // Check if the URL contains the `success` query parameter
 const urlParams = new URLSearchParams(window.location.search);
 if (urlParams.has('success')) {
 alert('Your message has been sent successfully!');
 // Remove the query parameter from the URL (optional, for better
user experience)
 history.replaceState({}, document.title,
window.location.pathname);
 }
</script>

{% endblock %}

```



layout.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width,
initial-scale=1.0">
 <title>{% block title %}Flexibility Training{% endblock %}</title>
 <link rel="stylesheet" href="{{ url_for('static',
filename='styles.css') }}">
</head>
<body>
 <header>
 <h1>Flexibility Training</h1>
 <nav>

 About
 <a href="{{ url_for('positions')
}}">Positions
 Survey
 Contact

 {% if 'user_id' in session %}
 <a href="{{ url_for('profile')
}}">Profile
 <a href="{{ url_for('logout')
}}">Logout
 {% else %}
 Login
 <a href="{{ url_for('register')
}}">Register
 {% endif %}

 </nav>
 </header>

 <main class="wrapper">
 {% block content %}
 <!-- Page-specific content goes here -->
 {% endblock %}
 </main>

 <footer>
 <p>© 2025 Flexibility Training. All rights reserved.</p>
 </footer>
</body>
```

```
</html>
```

login.html:

```
{% extends 'layout.html' %}
{% block title %}Login{% endblock %}
{% block content %}
<h2>Login</h2>
<form action="/login" method="post">
 <label for="email">Email:</label>
 <input type="email" id="email" name="email" required>
 <label for="password">Password:</label>
 <input type="password" id="password" name="password" required>
 <button type="submit">Login</button>
</form>
{% endblock %}
```

positions.html:

```
{% extends "layout.html" %}

{% block title %}Flexibility Positions{% endblock %}

{% block content %}
<section id="positions">
 <h2>Basic Flexibility Positions</h2>
 <div class="positions-container">
 <div class="position">
 <h3>Forward Fold</h3>

 <p>The Forward Fold is a foundational position that not only
targets the hamstrings but also decompresses the lower back and engages
the entire posterior chain. Begin by standing tall with your feet
hip-width apart, grounding through your heels. Initiate the movement by
hinging at the hips rather than the lower back, maintaining a slight
bend in the knees if needed. Let gravity guide your upper body as you
fold forward, reaching your hands toward your toes, shins, or the floor.
Focus on creating length in the spine and avoiding any excessive
rounding in the upper back. This position is a gateway to deeper forward
folds and is an excellent tool for improving hamstring flexibility and
overall body awareness.</p>
 </div>

 <div class="position">
 <h3>Bridge</h3>
```

```

```

```
<p>The Bridge is a cornerstone of backbending practice that
combines spinal extension with activation of the glutes and posterior
chain. Begin by lying on your back with your knees bent and feet flat on
the ground, about hip-width apart. Position your arms by your sides,
palms pressing into the floor. Engage your glutes and core as you press
through your heels to lift your hips toward the ceiling. Imagine
creating a gentle arc through your spine while keeping your knees
aligned and resisting the tendency to splay outward. This position
builds the foundation for advanced backbends, helping to open the chest
and strengthen the spinal erectors while promoting active
flexibility.</p>
```

```
</div>
```

```
<div class="position">
```

```
<h3>Front Split</h3>
```

```

```

```
<p>The Front Split, or Hanumanasana, is a hallmark position
for hamstring and hip flexor flexibility. Begin in a lunge position,
ensuring your front knee is stacked over your ankle and your back leg
extends behind you. Gradually slide your front foot forward and your
back knee backward, allowing your pelvis to descend toward the ground.
The goal is to keep the hips square and avoid collapsing to one side.
Use props such as yoga blocks or cushions to support your hands or hips
as you work toward full extension. Focus on active engagement in both
legs, using the muscles to hold the position rather than simply relying
on passive stretching. This position not only improves flexibility but
also enhances strength and control in the hip joints.</p>
```

```
</div>
```

```
<div class="position">
```

```
<h3>Middle Split</h3>
```

```

```

```
<p>The Middle Split is a benchmark for lateral hip
flexibility and inner thigh mobility. Start by sitting on the floor with
your legs extended wide apart. Ensure that your toes point upward and
your knees remain straight to maintain proper alignment. Gradually slide
your feet outward as you lean your torso slightly forward. Engage your
quadriceps and glutes to protect the hip joints and actively pull your
legs apart. Avoid collapsing into your lower back; instead, think about
lengthening your spine forward. This position is excellent for
```

```

developing strength and flexibility in the adductors and preparing for
deeper lateral splits.</p>
</div>

<div class="position">
 <h3>Pancake</h3>

 <p>The Pancake is an advanced forward fold that emphasizes
hip rotation and hamstring length. Begin in a seated straddle position
with your legs spread wide apart. Engage your core and tilt your pelvis
forward to initiate the fold from your hips rather than your back. Reach
your arms forward, aiming to rest your chest closer to the ground. Focus
on keeping your knees and toes pointed upward to maintain active
engagement in the legs. This position not only stretches the inner
thighs and hamstrings but also builds strength and control in the hip
flexors, making it a staple in active flexibility training.</p>
</div>

</div>
</section>
{% endblock %}

```

profile.html:

```

{% extends 'layout.html' %}
{% block title %}Profile{% endblock %}
{% block content %}
<h2>Welcome, {{ username }}!</h2>
<p>Upload your progress photos and track your flexibility journey.</p>

<form action="/profile" method="post" enctype="multipart/form-data">
 <label for="photo">Upload Progress Photo:</label>
 <input type="file" id="photo" name="photo" required>
 <label for="description">Description:</label>
 <textarea id="description" name="description" rows="4"></textarea>
 <button type="submit">Add Progress</button>
</form>

<h3>Your Progress</h3>

 {% for entry in progress_entries %}

 <p>{{ entry.description }}</p>

{% endfor %}

{% endblock %}
```

register.html:

```
{% extends 'layout.html' %}
{% block title %}Register{% endblock %}
{% block content %}
<h2>Register</h2>
<form action="/register" method="post">
 <label for="username">Username:</label>
 <input type="text" id="username" name="username" required>
 <label for="email">Email:</label>
 <input type="email" id="email" name="email" required>
 <label for="password">Password:</label>
 <input type="password" id="password" name="password" required>
 <button type="submit">Register</button>
</form>
{% endblock %}
```

survey.html:

```
{% extends "layout.html" %}

{% block title %}Flexibility Survey{% endblock %}

{% block content %}
<section>
 <h2>Find Your Flexibility Level</h2>
 <form action="/submit-survey" method="POST" id="flexibility-survey">
 <!-- Step 1: Choose a position -->
 <div id="question-1" class="question">
 <label for="position">Which position would you like to
improve on?</label>
 <select id="position" name="position" required>
 <option value="" disabled selected>Select a
position</option>
 <option value="Forward Fold">Forward Fold</option>
 <option value="Bridge">Bridge</option>
 <option value="Front Split">Front Split</option>
 <option value="Middle Split">Middle Split</option>
 <option value="Pancake">Pancake</option>
 </select>
 </div>
 </form>
</section>
{% endblock %}
```

```

 </select>
 <button type="button"
onclick="showNextQuestion()">Next</button>
 </div>

 <!-- Placeholder for dynamic questions -->
 <div id="dynamic-questions"></div>

 <!-- Submit button (hidden initially) -->
 <div id="submit-container" style="display: none;">
 <button type="submit">Get Your Template</button>
 </div>
</form>

<!-- Template Display Section -->
{% if template %}
<section id="template-display">
 <h3>Your Training Template</h3>
 <h4>{{ template.position_name }} - {{ template.template_level
}}</h4>

 <p>{{ template.description }}</p>
</section>
{% endif %}
</section>

<script>
 // Declare an object to track the current state of the survey
 let currentState = {};

 function showNextQuestion() {
 // Get the selected position value from the dropdown
 const position = document.getElementById('position').value;

 // Save the selected position to the current state
 currentState.position = position; // Save the position

 // Get the dynamic questions container element
 const dynamicQuestions =
document.getElementById('dynamic-questions');

 dynamicQuestions.innerHTML = ""; // Clear previous questions

 // Hide the submit button initially
 document.getElementById('submit-container').style.display =

```

```

"none";

 if (position === "Forward Fold") {
 addQuestion(
 "forward_fold_question",
 "Can you reach your ankles in a Forward Fold position?",
 ["Yes", "No"]
);
 } else if (position === "Bridge") {
 addQuestion(
 "bridge_question",
 "Can you perform a Feet Elevated Bridge?",
 ["Yes", "No"]
);
 } else if (position === "Front Split") {
 addQuestion(
 "front_split_question_1",
 "Can you perform Forward Fold reaching the floor?",
 ["Yes", "No"],
 handleSecondFrontSplitQuestion
);
 } else if (position === "Middle Split") {
 addQuestion(
 "middle_split_question",
 "How far from the floor are you in the Middle Split
position?",
 ["30cm or more", "Less than 30cm"]
);
 } else if (position === "Pancake") {
 addQuestion(
 "pancake_question",
 "Can you touch the floor with your head in a seated
Pancake position?",
 ["Yes", "No"]
);
 }
}

/**
 * Function to dynamically add a question to the form.
 * @param {string} name - The name of the input field for the
question.
 * @param {string} question - The question text to be displayed.
 * @param {Array} options - An array of answer options for the
question.
 * @param {Function|null} onChangeHandler - Optional callback to

```

```

handle changes to the answer.
 */
 function addQuestion(name, question, options, onChangeHandler =
null) {
 // Get the dynamic questions container
 const dynamicQuestions =
document.getElementById('dynamic-questions');

 // Create a container for the new question
 const questionDiv = document.createElement("div");
 questionDiv.className = "question";

 // Create label
 const label = document.createElement("label");
 label.innerText = question;
 questionDiv.appendChild(label);

 // Create select dropdown
 const select = document.createElement("select");
 select.name = name; // Set the name for the select element
 select.required = true; // Make it a required field

 // Add an event listener for changes in the select dropdown
 select.onchange = function () {
 // Save the selected answer to the current state
 currentState[name] = this.value;

 // If a callback is provided, call it with the selected
answer
 if (onChangeHandler) onChangeHandler(this.value);

 // Enable submission for single-question positions
 document.getElementById("submit-container").style.display =
"block";
 };

 // Add default option
 const defaultOption = document.createElement("option");
 defaultOption.value = "";
 defaultOption.disabled = true;
 defaultOption.selected = true;
 defaultOption.innerText = "Select an option";
 select.appendChild(defaultOption);

 // Add other options
 options.forEach(option => {

```



```

 const opt = document.createElement("option");
 opt.value = option.toLowerCase();
 opt.innerText = option;
 select.appendChild(opt);
 });

 // Add the select dropdown to the question container
 questionDiv.appendChild(select);

 // Add the question container to the dynamic questions section
 dynamicQuestions.appendChild(questionDiv);
}

/**
 * Function to handle displaying the second question for Front
 * Split.
 * @param {string} answer - The answer to the first Front Split
 * question.
 */
function handleSecondFrontSplitQuestion(answer) {
 const dynamicQuestions =
document.getElementById('dynamic-questions');
 dynamicQuestions.innerHTML = ""; // Clear previous question

 // If the answer is "no", set the level to Beginner and show the
 submit button
 if (answer === "no") {
 currentState["front_split_level"] = "Beginner";
 document.getElementById("submit-container").style.display =
"block";
 } else {
 // If the answer is "no", set the level to Beginner and show
 the submit button
 addQuestion(
 "front_split_question_2",
 "Can you perform Couch Stretch staying upright with your
 pelvis posteriorly tilted?",
 ["Yes", "No"],
 () => {
 // Show the submit button after the second question
 is answered
 document.getElementById("submit-container").style.display = "block";
 }
);
 }
}

```

```
}
</script>
{% endblock %}
```

procfile:

web: gunicorn app:app

requirements:

asyncio==3.4.3

attrs==20.3.0

autopep8==2.3.1

backports.shutil-get-terminal-size==1.0.0

backports.shutil\_which==3.5.2

beautifulsoup4==4.12.3

black==24.10.0

blinker==1.9.0

braceexpand==0.1.7

Brotli==1.1.0

cachelib==0.13.0

certifi==2024.12.14

cffi==1.17.1

charset-normalizer==3.4.1

check50==3.3.11

clang-format==19.1.5

cli50==8.0.1

click==8.1.8

colorama==0.4.6

compare50==1.2.6

contourpy==1.3.1

cryptography==44.0.0

cssselect2==0.7.0

cycler==0.12.1

djhtml==3.0.7

EditorConfig==0.17.0

Flask==3.1.0

Flask-Mail==0.10.0

Flask-Session==0.8.0

Flask-SQLAlchemy==3.1.1

fonttools==4.55.3

greenlet==3.1.1

help50==3.0.5

icdiff==2.0.7

idna==3.10

inflect==7.0.0

iniconfig==2.0.0

intervaltree==3.1.0

itsdangerous==2.2.0

jellyfish==0.11.2  
Jinja2==3.1.5  
jsbeautifier==1.15.1  
kiwisolver==1.4.8  
lib50==3.0.12  
MarkupSafe==3.0.2  
matplotlib==3.10.0  
more-itertools==10.5.0  
msgspec==0.19.0  
mypy-extensions==1.0.0  
natsort==8.4.0  
numpy==1.26.4  
packaging==24.2  
pathspec==0.12.1  
pexpect==4.9.0  
pillow==10.4.0  
platformdirs==4.3.6  
pluggy==1.5.0  
psutil==6.1.1  
ptyprocess==0.7.0  
pycodestyle==2.12.0  
pycparser==2.22  
pydantic==1.10.21  
pydyf==0.11.0  
Pygments==2.18.0  
pyparsing==3.2.1  
pypdf==5.1.0  
pyphen==0.17.0  
pytest==8.3.4  
python-dateutil==2.9.0.post0  
python-magic==0.4.27  
pytz==2024.2  
PyYAML==6.0.2  
render50==9.2.7  
requests==2.32.3  
setuptools==75.6.0  
six==1.17.0  
sortedcontainers==2.4.0  
soupsieve==2.6  
SQLAlchemy==2.0.36  
sqlparse==0.5.3  
style50==2.10.4  
submit50==3.2.0  
termcolor==1.1.0  
tinycss2==1.4.0  
tinyhtml5==2.0.0  
tqdm==4.67.1  
typeguard==4.4.1

typing\_extensions==4.12.2  
tzlocal==5.2  
urllib3==2.3.0  
weasyprint==63.1  
webencodings==0.5.1  
websockets==14.1  
Werkzeug==3.1.3  
wheel==0.45.1  
zopfli==0.2.3.post1  
unicorn

test\_app.py:

```
import unittest
from app import app, db, User, Progress, TrainingTemplate
from werkzeug.security import check_password_hash
import os
import tempfile
from io import BytesIO

class FlaskAppTests(unittest.TestCase):
 def setUp(self):
 """
 SETUP PHASE

 1. Configure test environment
 2. Create temporary database
 3. Initialize application context
 4. Create database tables
 5. Add test data
 """
 # Test configuration
 app.config['TESTING'] = True
 app.config['WTF_CSRF_ENABLED'] = False
 app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

 # Create temporary database
 self.temp_db_file = tempfile.NamedTemporaryFile(suffix='.db',
delete=False)
 self.temp_db_file.close()
 app.config['SQLALCHEMY_DATABASE_URI'] =
f'sqlite:/// {self.temp_db_file.name}'

 # Setup application context
 self.app_context = app.app_context()
 self.app_context.push()

 # Initialize database
 db.create_all()
```

```

self.client = app.test_client()

Add test data
test_template = TrainingTemplate(
 position_name='Forward Fold',
 template_level='Beginner',
 image_path='static/templates/forward_fold_beginner.png',
 description='Test template'
)
db.session.add(test_template)
db.session.commit()

def tearDown(self):
 """
 CLEANUP PHASE

 1. Remove database session
 2. Drop all tables
 3. Remove context
 4. Delete temporary file
 """
 db.session.remove()
 db.drop_all()
 self.app_context.pop()
 os.unlink(self.temp_db_file.name)

def test_about_page(self):
 """
 TEST CASE: About Page Access

 GIVEN: A Flask application
 WHEN: User accesses the root URL ('/')
 THEN: Should return successful response (200)
 """
 response = self.client.get('/')
 self.assertEqual(response.status_code, 200)

def test_register(self):
 """
 TEST CASE: User Registration

 GIVEN: A new user's details
 WHEN: User submits registration form
 THEN:
 1. User should be created in database
 2. Password should be hashed
 3. Should redirect to login page
 """

```

```

response = self.client.post('/register', data={
 'username': 'testuser',
 'email': 'test@example.com',
 'password': 'testpassword123'
}, follow_redirects=True)

Verify user creation
user = User.query.filter_by(username='testuser').first()
self.assertIsNotNone(user)
self.assertEqual(user.email, 'test@example.com')
self.assertTrue(check_password_hash(user.password,
'testpassword123'))
self.assertIn(b'<title>Login</title>', response.data)

def test_register_validation(self):
 """
 TEST CASE: Registration Validation

 GIVEN: Invalid registration data
 TEST 1 - Short Password:
 WHEN: User submits registration with short password
 THEN: Should stay on registration page

 TEST 2 - Duplicate Email:
 WHEN: User submits registration with existing email
 THEN: Should stay on registration page
 """
 # Test short password
 response = self.client.post('/register', data={
 'username': 'testuser',
 'email': 'test@example.com',
 'password': 'short'
 }, follow_redirects=True)
 self.assertIn(b'<title>Register</title>', response.data)

 # Test duplicate email
 # First registration
 self.client.post('/register', data={
 'username': 'testuser1',
 'email': 'duplicate@example.com',
 'password': 'testpassword123'
 })
 # Second registration with same email
 response = self.client.post('/register', data={
 'username': 'testuser2',
 'email': 'duplicate@example.com',
 'password': 'testpassword123'
 }, follow_redirects=True)

```

```

 self.assertIn(b'<title>Register</title>', response.data)

def test_login_logout(self):
 """
 TEST CASE: Login and Logout

 GIVEN: A registered user

 LOGIN TEST:
 WHEN: User submits valid login credentials
 THEN: Should redirect to profile page

 LOGOUT TEST:
 WHEN: Logged-in user clicks logout
 THEN: Should redirect to login page
 """

 # Register user
 self.client.post('/register', data={
 'username': 'testuser',
 'email': 'test@example.com',
 'password': 'testpassword123'
 })

 # Test login
 login_response = self.client.post('/login', data={
 'email': 'test@example.com',
 'password': 'testpassword123'
 }, follow_redirects=True)
 self.assertIn(b'<title>Profile</title>', login_response.data)
 self.assertIn(b'Welcome, testuser!', login_response.data)

 # Test logout
 logout_response = self.client.get('/logout', follow_redirects=True)
 self.assertIn(b'<title>Login</title>', logout_response.data)

def test_profile_protected(self):
 """
 TEST CASE: Profile Page Protection

 GIVEN: An unauthenticated user
 WHEN: User tries to access profile page
 THEN: Should redirect to login page
 """

 response = self.client.get('/profile', follow_redirects=True)
 self.assertIn(b'<title>Login</title>', response.data)
 self.assertIn(b'<h2>Login</h2>', response.data)

def test_progress_upload(self):

```

```

"""
TEST CASE: Progress Photo Upload

GIVEN: An authenticated user
WHEN: User uploads a progress photo
THEN:
1. Photo should be saved
2. Progress entry should be created in database
"""

Setup authenticated user
self.client.post('/register', data={
 'username': 'testuser',
 'email': 'test@example.com',
 'password': 'testpassword123'
})
self.client.post('/login', data={
 'email': 'test@example.com',
 'password': 'testpassword123'
})

Create test image
image_data = BytesIO(b'fake image data')
image_data.name = 'test.jpg'

Test upload
response = self.client.post('/profile', data={
 'photo': (image_data, 'test.jpg'),
 'description': 'Test progress photo'
}, content_type='multipart/form-data', follow_redirects=True)

Verify upload
progress = Progress.query.first()
self.assertIsNotNone(progress)
self.assertEqual(progress.description, 'Test progress photo')

def test_survey_submission(self):
 """
 TEST CASE: Survey Submission

 GIVEN: Survey response data
 WHEN: User submits survey
 THEN: Should show appropriate template based on answers
 """
 response = self.client.post('/submit-survey', data={
 'position': 'Forward Fold',
 'forward_fold_question': 'no'
 }, follow_redirects=True)

```



```

self.assertEqual(response.status_code, 200)
self.assertIn(b'Forward Fold', response.data)

def test_contact_form(self):
 """
 TEST CASE: Contact Form

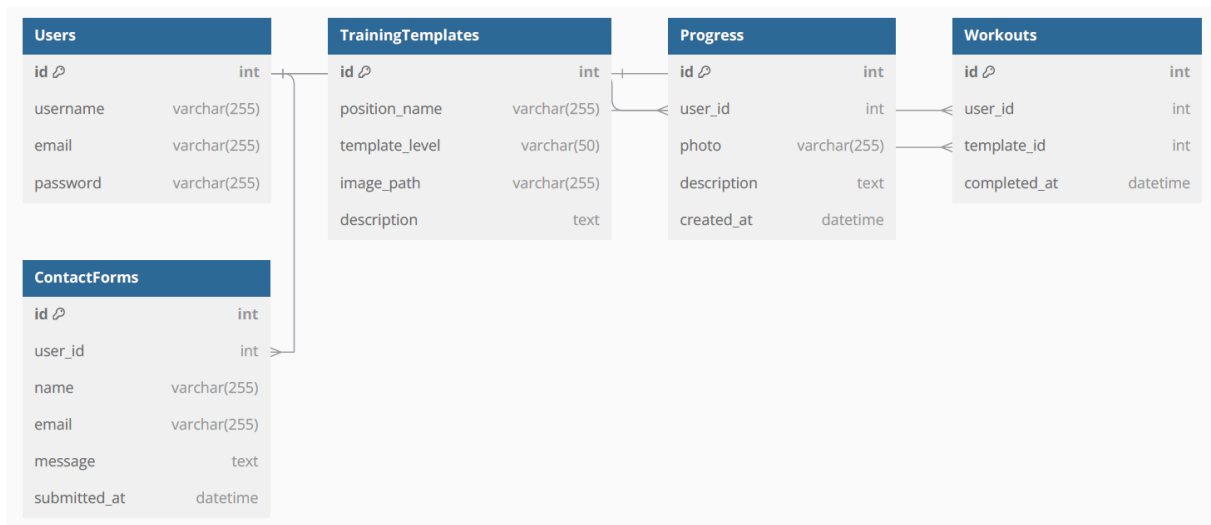
 GIVEN: Contact form data
 WHEN: User submits contact form
 THEN: Should process form successfully
 """
 app.config['TESTING'] = True

 response = self.client.post('/submit-contact', data={
 'email': 'test@example.com',
 'message': 'Test message'
 }, follow_redirects=True)
 self.assertEqual(response.status_code, 200)

if __name__ == '__main__':
 unittest.main()

```

database ERD:



Link to the video:

<https://youtu.be/5KGT9dXqKF8>

## Evaluation:

### What Worked Out

1. **Core Functionality:**  
The website successfully provides users with essential tools to learn about flexibility and access workout plans based on their flexibility level. The survey form effectively links users to appropriate workout plans, fulfilling a core requirement.
  2. **Content Delivery:**  
The presentation of basic flexibility positions and corresponding workout plans works well. Storing the templates in an SQL database ensures that the data is organized and easy to retrieve.
  3. **Contact Form:**  
The inclusion of a functional contact form allows users to reach out for inquiries, ensuring communication and engagement.
  4. **Deployment:**  
The project runs live on Render, making it accessible and user-friendly. This step marks a significant achievement in ensuring the website's usability beyond the local environment.
- 

### What Didn't Work Out

1. **Profiles and Progress Tracking:**  
While progress tracking was marked as a "should-have" feature, its implementation was only partially realized. Users can upload progress pictures, but the feature is not as robust or seamless as initially envisioned.
  2. **Paywall for Extra Content:**  
The paywall for premium content was not implemented.
  3. **Testing Environment:**  
Writing and executing unit tests proved challenging. While tests worked in the CS50 environment, local testing environments were problematic, limiting comprehensive quality assurance.
- 

### What Could Be Improved in the Future

1. **Enhanced Profiles:**
  - Implement detailed user profiles with fields for tracking progress, setting goals, and receiving personalized feedback.
  - Allow users to view a timeline of their progress, including uploaded images and notes.
2. **Robust Progress Tracking:**

- Introduce visual analytics, such as graphs, to showcase progress over time.
- Enable users to add custom tags or descriptions to their progress uploads for better organization.
- 3. **Premium Features:**
  - Develop and integrate a paywall system for advanced content, such as video tutorials, personalized coaching plans, or interactive guides.
  - Include an option for users to unlock customizable workout plans as part of the premium package.
- 4. **Mobile Responsiveness:**

Optimize the website for mobile devices to ensure a seamless experience across platforms, as many users may access workout tools on their phones.
- 5. **Improved Testing and Debugging:**
  - Set up a consistent local testing environment to ensure that unit tests and other automated checks can be executed reliably.
  - Introduce more comprehensive tests to cover edge cases, ensuring the site runs smoothly under different conditions.
- 6. **Database Refinement:**
  - Expand the SQL database structure to include more detailed workout metadata (e.g., exercise duration, difficulty levels) for better customization.
  - Optimize queries to ensure faster data retrieval, especially as the user base grows.
- 7. **User Engagement Features:**
  - Consider adding motivational features like badges or streaks to encourage consistency in training.
  - Introduce social features, such as sharing progress with friends or joining challenges, to build a community around the platform.
- 8. **Localization and Accessibility:**
  - Add multi-language support to cater to a global audience.
  - Ensure the website meets accessibility standards, such as screen reader compatibility and keyboard navigation.

## Final Project Reflection:

Working on the final project was a culmination of everything I learned throughout the Creative Programming for Non-IT Students minor. It was both the most challenging and rewarding endeavor of these few months. The project pushed me to apply theoretical knowledge in a practical setting, often testing my resilience and problem-solving abilities. At times, it felt as though I was stuck and would not be able to finalize the project. However, with perseverance, determination, and support from the resources I had, I successfully completed it.

One of the initial challenges I faced was ensuring consistency in tagging HTML files so that my CSS styles would render as intended. It was a meticulous process, and I often found myself troubleshooting minor errors that would break the design. This struggle taught me the importance of careful organization and attention to detail when working on front-end development.

The most time-consuming aspect of the project was writing JavaScript. For me, it remains one of the most complex programming languages to learn and apply effectively. Initially, I wasn't even certain how I wanted my form to look or function. As I started coding, I realized I needed the form to be more responsive and user-friendly, which meant going back to the drawing board and rewriting significant portions of my code. This iterative process was frustrating at times, but in the end, it was incredibly satisfying to see the form function as I envisioned.

Another significant hurdle was writing the unit testing file. I struggled with getting the testing environment to function properly. I experimented with multiple environments and different versions of software, encountering numerous roadblocks along the way. Eventually, I managed to pass all the tests within the CS50 environment, even though I couldn't get the tests to work in my local terminal. This experience reinforced the importance of adaptability and resourcefulness when dealing with technical challenges.

Finally, deploying the project to Render was another challenging aspect. Despite being unfamiliar with deployment tools at the beginning of the project, I managed to get everything up and running within a day. Seeing the website live and functional was one of the most satisfying moments of the entire process.

This project not only allowed me to consolidate my skills in HTML, CSS, JavaScript, and Flask but also taught me valuable lessons about debugging, troubleshooting, and perseverance. Each challenge I encountered helped me grow, not just as a programmer, but also in my ability to manage complex projects. Looking back, I feel proud of what I accomplished and grateful for the learning journey it took to get there.

## Final reflection

Completing the **Creative Programming for Non-IT Students** minor has been an enriching experience, blending technical learning with creativity and problem-solving. As a student of **Creative Business and Media Management**, this journey into programming has given me a broader perspective on technology's potential within my field.

One of the most valuable lessons I've learned is how programming can complement and enhance media strategies. For example, creating web-based tools, like the flexibility website for my final project, demonstrated how technology can be a bridge between user needs and innovative solutions. This website not only showcased my ability to develop and structure a product but also highlighted the role of user-centric design—a core principle in media and business.

Throughout the minor, I encountered challenges that tested my perseverance. For instance:

- **Learning New Concepts:** The step-by-step learning of programming languages such as Python, SQL, and JavaScript was both daunting and exciting. While initially overwhelming, the structured approach of weekly assignments and the CS50 curriculum allowed me to build confidence incrementally.
- **Team Collaboration:** Working with peers on certain assignments helped me appreciate the dynamics of teamwork, particularly in tech-focused projects. I learned how to contribute effectively, even when my technical skills were less advanced than those of my peers.
- **Problem-Solving Mindset:** Debugging errors, refining algorithms, and optimizing designs taught me that solutions often require patience and creative thinking—qualities essential in media management, where problem-solving is a daily task.

## What Worked

- **Practical Application:** Each week, I applied what I learned to real-world problems, reinforcing my understanding of how coding can solve complex challenges.
- **Structured Reflection:** Documenting my weekly progress provided an opportunity to internalize lessons and understand areas for improvement.
- **Final Project:** The website's creation showcased my ability to integrate technical skills with media management concepts, such as user engagement and content strategy.

## What Didn't Work

- **Initial Overwhelm:** Diving into technical concepts without prior experience was intimidating. This led to slower progress at the beginning.
- **Tool Limitations:** Some tools and environments were less intuitive, requiring extra effort to adapt.

## Future Improvements

- **Refining Technical Skills:** While I've gained a strong foundation, there's still room for improvement, particularly in advanced JavaScript and database management.
- **Enhanced Integration:** Exploring ways to incorporate programming into creative media campaigns or interactive content strategies could expand my skill set.
- **Collaboration Across Disciplines:** Bringing technical experts into creative projects earlier could lead to even more innovative results.

## Conclusion

This minor has been a bridge between the creative and technical worlds, empowering me to think beyond traditional boundaries in media and business. The technical skills I've acquired will allow me to approach future projects with more confidence, whether it's building digital tools, managing creative teams, or analyzing user data. In an industry that increasingly relies on technology, I'm excited to continue exploring how programming can transform ideas into impactful solutions.

