|  |  |
| --- | --- |
| **ELECTRONIC ASSIGNMENT COVERSHEET** | Murdoch_land_RGB |

|  |  |
| --- | --- |
| **Student Number:** | 35517678 |
| Surname: | New |
| Given name: | Wei Nern |
| Email: | 35517678@student.murdoch.edu.au |
|  |  |
| **Unit Code:** | ICT373 |
| Unit name: | Software Architecture |
| Enrolment mode: | Internal |
| Date: | 9/6/2025 |
| Assignment number: |  |
| Assignment name: |  |
| Tutor: | Andy Lee |

|  |
| --- |
| **Student’s Declaration:**   * Except where indicated, the work I am submitting in this assignment is my own work and has not been submitted for assessment in another unit. * This submission complies with Murdoch University's academic integrity commitments. I am aware that information about plagiarism and associated penalties can be found at <http://our.murdoch.edu.au/Educational-technologies/Academic-integrity/>. If I have any doubts or queries about this, I am further aware that I can contact my Unit Coordinator prior to submitting the assignment. * I acknowledge that the assessor of this assignment may, for the purpose of assessing this assignment:   + reproduce this assignment and provide a copy to another academic staff member; and/or   + submit a copy of this assignment to a plagiarism-checking service. This web-based service may retain a copy of this work for the sole purpose of subsequent plagiarism checking, but has a legal agreement with the University that it will not share or reproduce it in any form. * I have retained a copy of this assignment. * I will retain a copy of the notification of receipt of this assignment. If you have not received a receipt within three days, please check with your Unit Coordinator. |

|  |
| --- |
| I am aware that I am making this declaration by submitting this document electronically and by using my Murdoch ID and password it is deemed equivalent to executing this declaration with my written signature. |
| **Optional Comments to Tutor:**  *e.g. If this is a group assignment, list group members here* |

*If you can, please insert this completed form into the body of* ***each*** *assignment you submit. Follow the instructions in the Unit Information and Learning Guide about how to submit your file(s) and how to name them, so the Unit Coordinator knows whose work it is.*

***Start your assignment on the next page.***



Pastime Connection

*Pastime with like-minded people and foster a strong community.*

**Author:** New Wei Nern

**Murdoch Student ID**: 35517678

**Date:** 4/6/2025

**Student Note:** This is a GitHub [repository](https://github.com/nwnisworking/ICT373-Assignment-1/tree/V2) that contains the code.

**Files:**

Contents

[Problem Description 5](#_Toc200905655)

[Solution Approach 6](#_Toc200905656)

[Structure Design 8](#_Toc200905657)

[Testing 12](#_Toc200905658)

# Problem Description

We want to develop a website using HTML, CSS, and JavaScript that collects user data and ensures all the relevant fields are properly validated before submitting the data back to our server. The information sent by the user will be used to match them to a compatible community. Since it is a simple networking website, server-side validation is not required. The form must submit the following fields:

* User’s full name
* Phone number (with prefix code for Australia, Dubai, Malaysia, and Singapore)
* Date of birth (day, month, and year)
* Favourite pastime (from a provided list)

The fields must adhere to the following constraints:

* Name must contain only alphabetical letters, spaces, hyphens and apostrophes.
* Phone must contain a space between the prefix and number.
* Date must be validated (without the use of JavaScript’s Date object)
* Date must not be in the future or invalid (e.g., Leap year or non-existent date)
* Pastimes must be selected from a specific list
* All the fields must not be empty

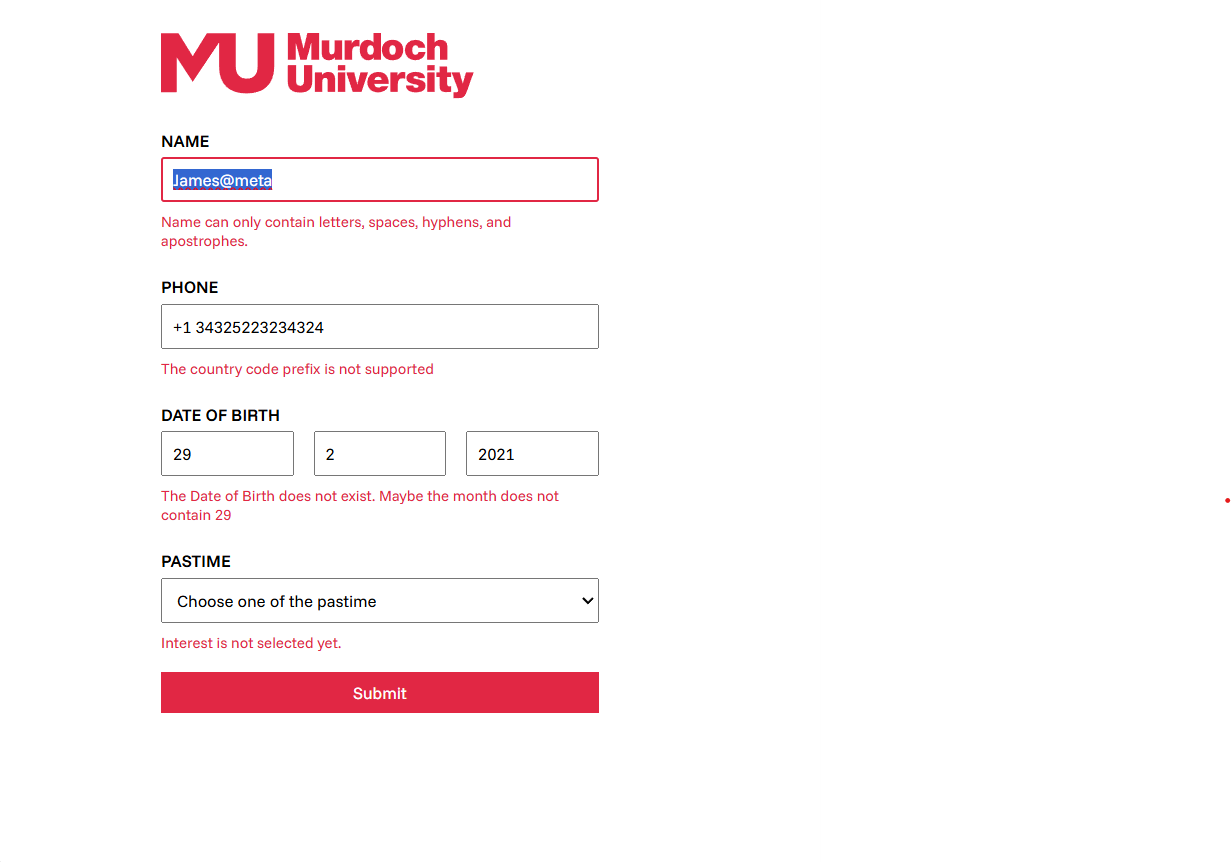
# Solution Approach

The approach to the solution was implemented using plain HTML, CSS, and JavaScript. Validation occurs within the JavaScript file, which manages DOM interactions, conditional logic, and event handling. The user is informed of any input errors through on-screen prompts, guiding them to correct the information before submission. This provides an interactive and user-friendly experience.

A red and white background with text

AI-generated content may be incorrect.

The design uses Murdoch University signature red as the primary colour, creating a strong contrast between the form and the tagline, allowing the users to quickly grasp the intention of the website. The layout is split into two sections: a left panel for users to enter their personal details, and a right panel that features a message encouraging them to enrol into the community. The simplistic design of the input fields allows the submit button to stand out - until the user encounters an input error, which draws their attention appropriately.



# Structure Design

As the requirement states, the use of JavaScript Date API is prohibited, so another API was used in its place: Performance API. By using performance.timeOrigin, we can reverse engineer to get the current date, month, and year.

Below is a function to calculate whether the year is a leap year.  
IsLeapYear(year)

# Leap year can be divided by 4, else it doesn’t count as a leap year

If year is divisible by 4 then

# It’s not a leap year if it’s divisible by 100 unless

# the year is divisible by 400

If year is divisible by 100 then

If year is divisible by 400 then

Return true

Else

Return false

Else

Return true

Else

Return false

Below is a snippet of the code written in pseudocode for calculating the date, month, and year.

TIME = performance.timeOrigin

# 1 second consists of 1000 milliseconds

TOTAL\_SEC = TIME / 1000

# 1 day consists of 24 hours x 60 minutes x 60 seconds = 86400 seconds

TOTAL\_DAYS = TOTAL\_SEC / (24 \* 60 \* 60)

# This gets the number of days it has elapsed since 1970.

Days\_Elapsed = TOTAL\_DAYS

# 1970 is the start of the Epoch

Year = 1970

Month = 0

Date = 0

While True

# Check if the

Is\_Leap = IsLeapYear(Year)

Days = Is\_Leap then 366 else 365

# We do not need to count the months yet! Only increment the year.

If Days\_Elapsed minus Days is more than 0

Days\_Elapsed -= Days

Year++

# We are at the current year. We need to figure out the month and date.

Else

# We are calculating the months now.

For I is 1, If I is less-equals to 12, I++

# The number of days the current month has

Days\_In\_Month = 0

# Check whether February is a leap year.

If I equals to 2

Days\_In\_Month = Is\_Leap ? 29 : 28

# Basically, anything divisible by 2 before August have 30 days

Else If I less equal to 7

Days\_in\_Month = I divisible 2 equals to 0 then 30 else 31

# Anything divisible by 2 onwards after July is 31 days

Else

Days\_in\_Month = I divisible 2 equals to 0 then 31 else 30

# Still not the current month

If Days\_Elapsed minus Days\_In\_Month is more than 0

Days\_Elapsed = Days\_In\_Month

Else

Month = I

Date = Floor(Days\_Elapsed + 1)

Escape While

The next step is to process each of the input fields. The code loops through an array containing each fields along with its validators to validate the input values. If a value is valid, the field’s name and value are appended to a URLSearchParams object. Otherwise, the error will display on the affected field and an on\_input event is added to clear input tip when the user modifies the field. The first invalid field is focused to guide the user’s attention.

After all fields are processed, the code checks whether the form is valid. If every field is valid, the form’s action URL will be appended with the URLSearchParams object. The form is submitted programmatically. If any field is invalid, no submission will occur.

Form Submit Event

Prevent form submission

Define fields by element and validator

# Assume the inputs are valid

Is\_valid = true

# Set to focus the first invalid input

Has\_selected = false

# Search parameter

Search\_Param = new SearchParam

For Element and validator in fields

Result, Message = validator.call(element)

Input\_Tip = Element.query(“small”)

If there is no element input event

Element input = ()=>Input\_Tip.text = “”

# Validator results in false

If no Result

Is\_valid = false

If Input\_text no text

Input\_text.text = Message

# Only focus on the first invalid field

If Has\_Selected is false

Element.focus()

Has\_Selected = True

Else

Search\_Param.append(element.name, element.value)

If Is\_valid

Form.action+= Search\_Param

Form.submit()

# Testing

Below is a boundary test to ensure that all relevant fields correctly validate their values. Fields with valid values have empty error messages. By default, all fields **should not** be empty; therefore, tests with empty values are omitted.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Description | Data | Expected Result | Actual Result | Test Passed |
| name\_1 | Name with an apostrophe | De’Lacy |  |  | Passed |
| name\_2 | Name with a hyphen | Smith-Doe |  |  | Passed |
| name\_3 | Name a with space | Alice Smith |  |  | Passed |
| name\_4 | Name with random symbol | Alice@smith | Name can only contain letters, spaces, hyphens, and apostrophes. | Name can only contain letters, spaces, hyphens, and apostrophes. | Passed |
| phone\_1 | Phone number without () | +65 12345678 |  |  | Passed |
| phone\_2 | Phone number without a + | 65 12345678 |  |  | Passed |
| phone\_3 | Phone number without country prefix | 12345678 | Phone number did not match the format | Phone number did not match the format | Passed |
| phone\_4 | Phone number that is not supported | +1 12345678 | The country code prefix is not supported. Only 60, 61, 65, and 971 are supported. | The country code prefix is not supported. Only 60, 61, 65, and 971 are supported. | Passed |
| year\_1 | The current year | 2025 | Oops! You can’t be born this year or in the future. | Oops! You can’t be born this year or in the future. | Passed |
| year\_2 | The future year | 2027 | Oops! You can’t be born this year or in the future. | Oops! You can’t be born this year or in the future. | Passed |
| year\_3 | Non-numeric characters | F34@ | Year can only contain numbers. | Year can only contain numbers. | Passed |
| year\_4 | A normal, valid year | 2020 |  |  | Passed |
| month\_1 | Non-numeric characters | F3 | Month can only contain numbers | Month can only numbers | Passed |
| month\_2 | A non-existent month that exceeds more than 12 | 14 | Month field exceeds the range of valid month (1 to 12) | Month field exceeds the range of valid month (1 to 12) | Passed |
| month\_3 | A normal, valid month | 2 |  |  | Passed |
| date\_1 | Non-numeric characters | F3 | Date can only contain numbers | Date can only numbers | Passed |
| date\_2 | A non-existent date that exceeds more than 31 | 49 | Date field exceeds the range of valid date (1 to 31) | Date field exceeds the range of valid date (1 to 31) | Passed |
| date\_3 | A normal, valid date | 28 |  |  | Passed |
| dob\_1 | A year that only has 365 days | 29 2 2025 | The Date of Birth does not exist. Maybe the month does not contain 29 | The Date of Birth does not exist. Maybe the month does not contain 29 | Passed |
| dob\_2 | A valid date that accounts for leap year | 29 2 2024 |  |  | Passed |

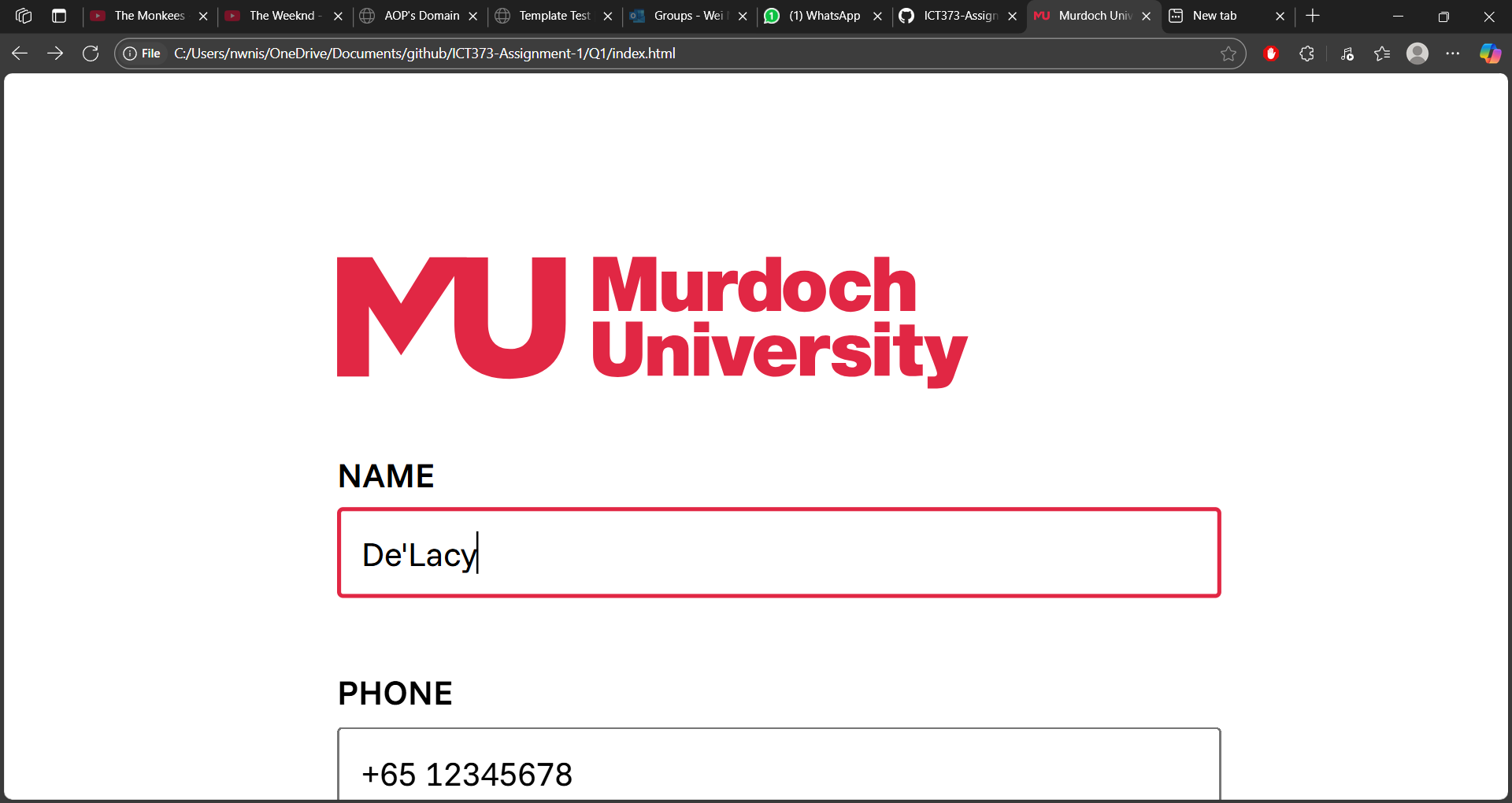
**Test ID**: name\_1

**Description**: Name with an apostrophe

**Data**: De’Lacy

**Expected Result**:

**Actual Result**:



**Test ID**: name\_2

**Description**: Name with a hyphen

**Data**: Smith-Doe

**Expected Result**:

**Actual Result**:

A screen shot of a computer

AI-generated content may be incorrect.

**Test ID**: name\_3

**Description**: Name with a space

**Data**: Alice Smith

**Expected Result**:

**Actual Result**: A screen shot of a computer

AI-generated content may be incorrect.

**Test ID**: name\_4

**Description**: Name with random symbol

**Data**: Alice@Smith

**Expected Result**: Name can only contain letters, spaces, hyphens, and apostrophes

**Actual Result**: Name can only contain letters, spaces, hyphens, and apostrophesA screenshot of a computer

AI-generated content may be incorrect.

**Test ID**: phone\_1

**Description**: Phone number without ()

**Data**: +65 12345678

**Expected Result**:

**Actual Result**:

A screen shot of a computer

AI-generated content may be incorrect.

**Test ID**: phone\_2

**Description**: Phone number without a +

**Data**: 65 12345678

**Expected Result**:

**Actual Result**: A screen shot of a computer

AI-generated content may be incorrect.

**Test ID**: phone\_3

**Description**: Phone number without a country prefix

**Data**: 12345678

**Expected Result**: Phone number did not match the format

**Actual Result**: Phone number did not match the format A screen shot of a computer

AI-generated content may be incorrect.

**Test ID**: phone\_4

**Description**: Phone number that is not supported

**Data**: +1 12345678

**Expected Result**: The country code prefix is not supported. Only 60, 61, 65, and 971 are supported.

**Actual Result**: The country code prefix is not supported. Only 60, 61, 65, and 971 are supported. A screen shot of a computer

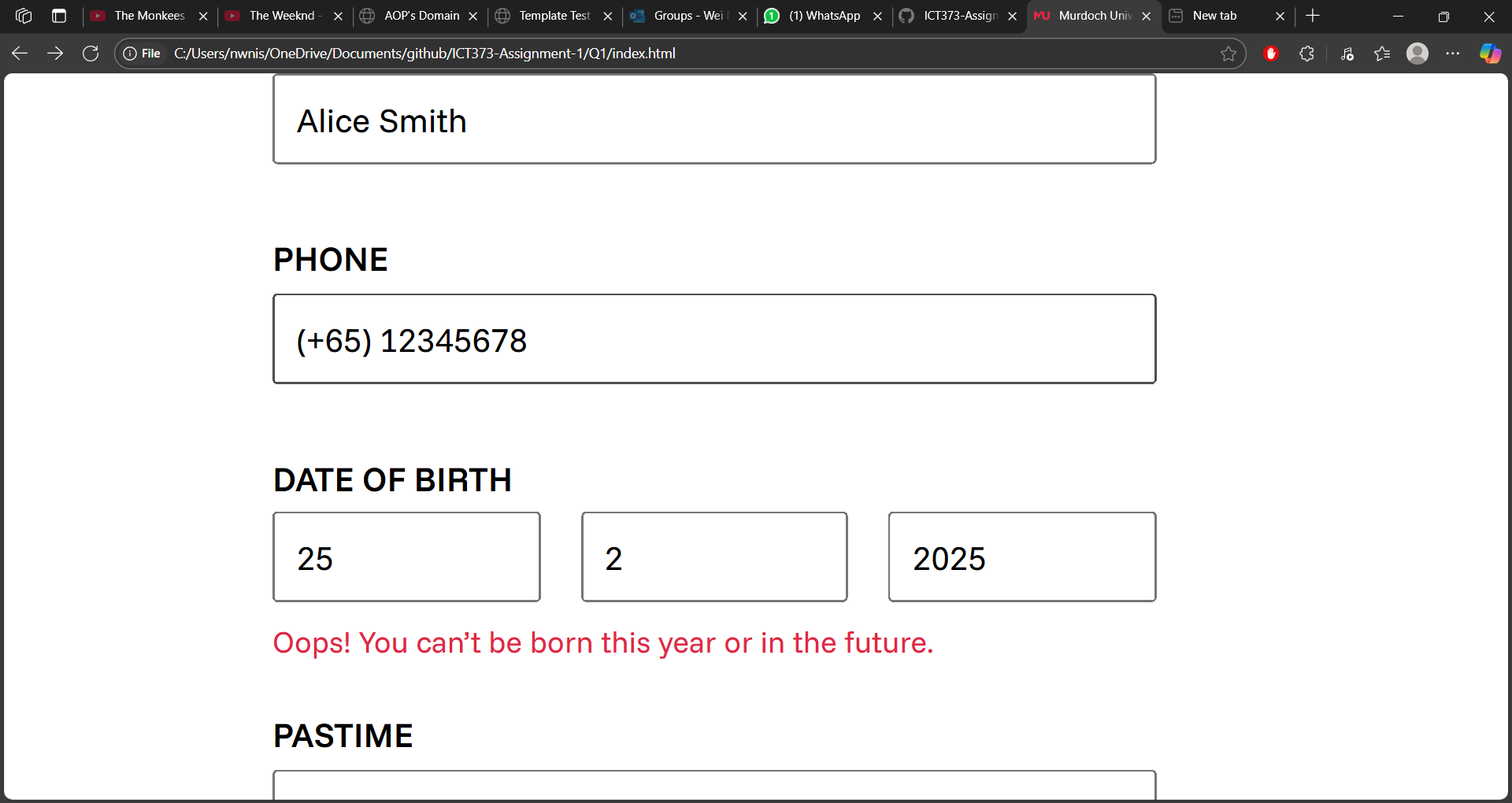
AI-generated content may be incorrect.

**Test ID**: year\_1

**Description**: The current year

**Data**: 2025

**Expected Result**: Oops! You can’t be born this year or in the future.

**Actual Result**: Oops! You can’t be born this year or in the future. 

**Test ID**: year\_2

**Description**: The future year

**Data**: 2027

**Expected Result**: Oops! You can’t be born this year or in the future.

**Actual Result**: Oops! You can’t be born this year or in the future. A screenshot of a computer

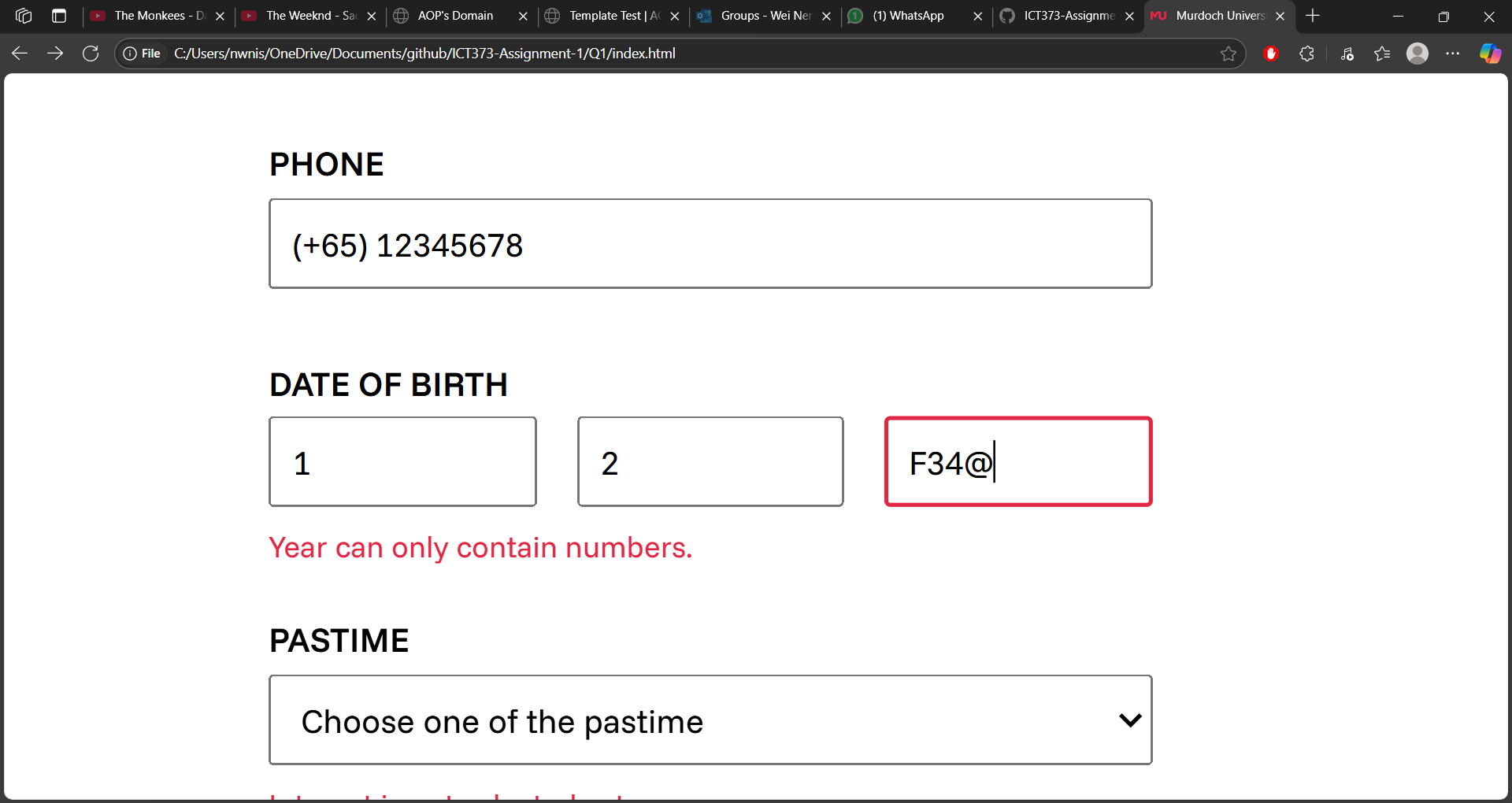
AI-generated content may be incorrect.

**Test ID**: year\_3

**Description**: Non-numeric characters

**Data**: F34@

**Expected Result**: Year can only contain numbers.

**Actual Result**: Year can only contain numbers. 

**Test ID**: year\_4

**Description**: A normal, valid year

**Data**: 2020

**Expected Result**:

**Actual Result**: A screenshot of a computer

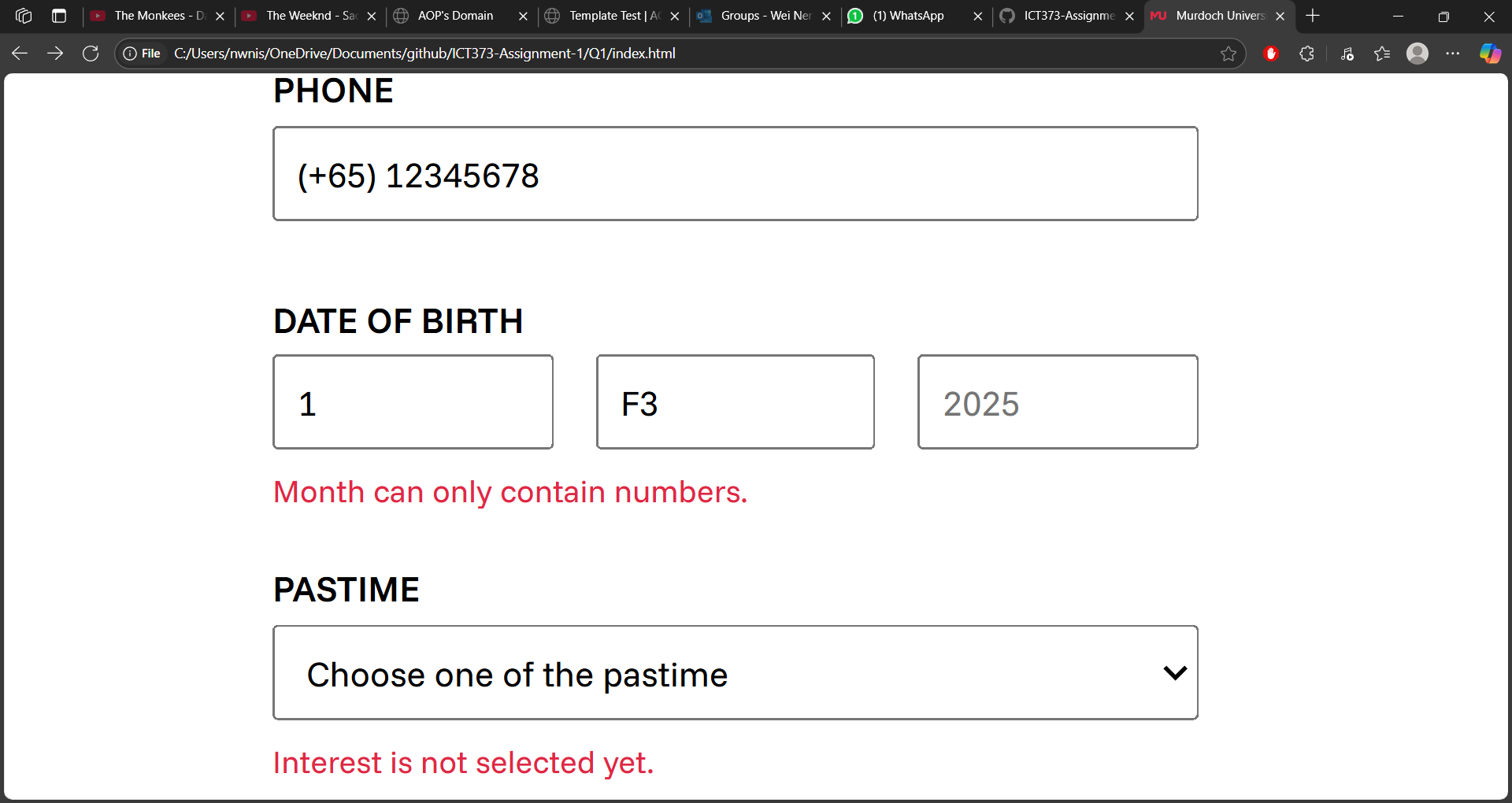
AI-generated content may be incorrect.

**Test ID**: month\_1

**Description**: Non-numeric characters

**Data**: F3

**Expected Result**: Month can only contain numbers

**Actual Result**: Month can only contain numbers 

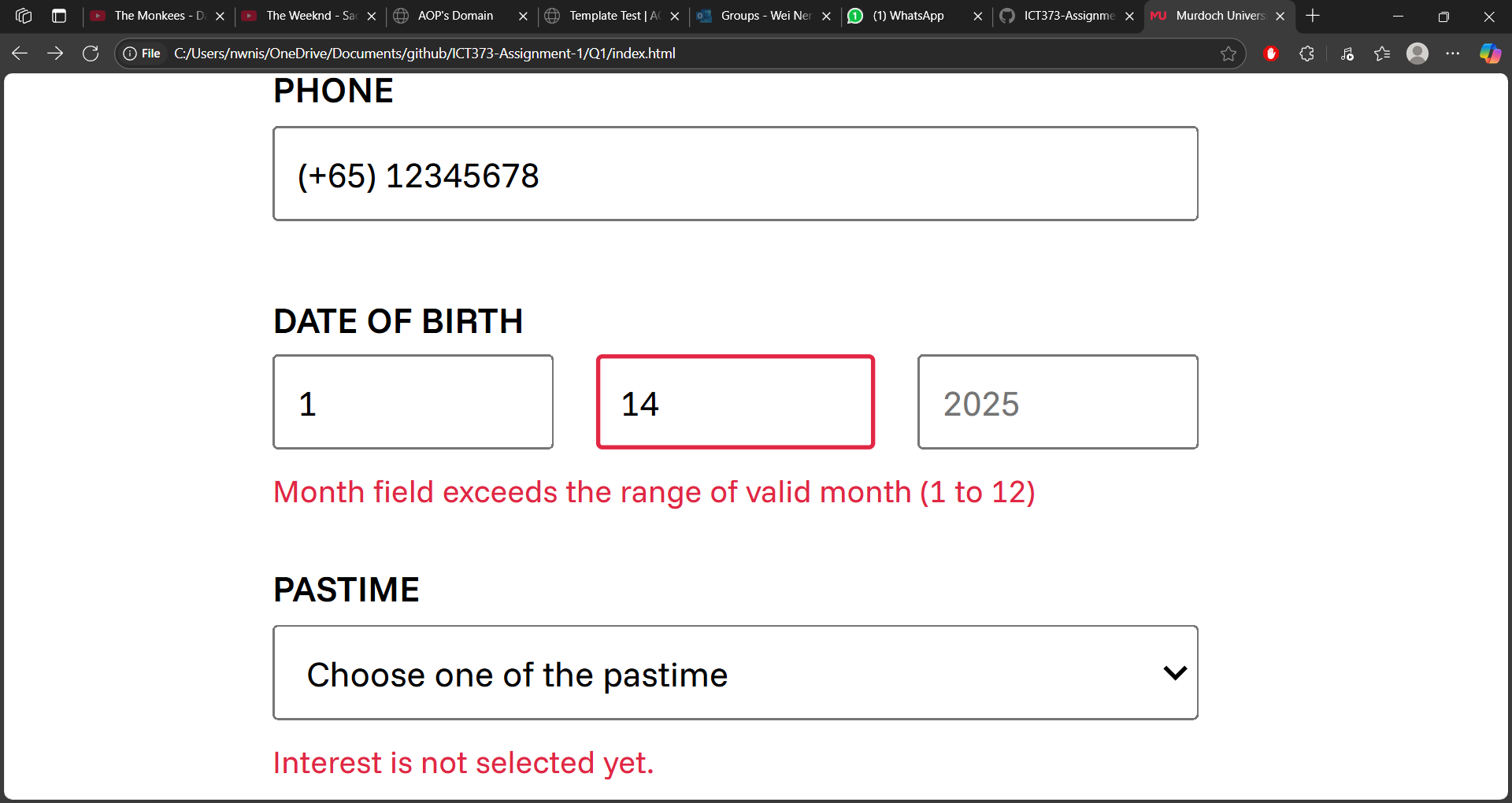
**Test ID**: month\_2

**Description**: A non-existent month that exceeds more than 12

**Data**: 14

**Expected Result**: Month field exceeds the range of valid month (1 to 12)

**Actual Result**: Month field exceeds the range of valid month (1 to 12)



**Test ID**: month\_3

**Description**: A normal, valid month

**Data**: 2

**Expected Result**:

**Actual Result**: A screenshot of a computer

AI-generated content may be incorrect.

**Test ID**: date\_1

**Description**: Non-numeric characters

**Data**: F3

**Expected Result**: Date can only contain numbers

**Actual Result**: Date can only contain numbers A screenshot of a computer screen

AI-generated content may be incorrect.

**Test ID**: date\_2

**Description**: A non-existent date that exceeds more than 31

**Data**: 49

**Expected Result**: Date field exceeds the range of valid date (1 to 31)

**Actual Result**: Date field exceeds the range of valid date (1 to 31) A screenshot of a computer

AI-generated content may be incorrect.

**Test ID**: date\_3

**Description**: A normal, valid date

**Data**: 29

**Expected Result**:

**Actual Result**: A screenshot of a computer screen

AI-generated content may be incorrect.

**Test ID**: dob\_1

**Description**: A year that only has 365 days

**Data**: 29 2 2025

**Expected Result**: The Date of Birth does not exist. Maybe the month does not contain 29

**Actual Result**: The Date of Birth does not exist. Maybe the month does not contain 29 A screenshot of a computer

AI-generated content may be incorrect.

**Test ID**: dob\_2

**Description**: A valid date that accounts for leap year

**Data**: 29 2 2024

**Expected Result**:

**Actual Result**: A screenshot of a computer

AI-generated content may be incorrect.