

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT**

**THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN NHẬP MÔN LẬP TRÌNH PYTHON**



**BÁO CÁO CUỐI KÌ**

**ĐỀ TÀI**

**APP TIỆN ÍCH ĐA NĂNG CHO MÁY TÍNH**

**ALL\_IN\_ONE UTILITY**

**MÃ MÔN HỌC:**  **241IPPA233277**

**Giảng viên hướng dẫn: TS. Phan Thị Thể.**

**Thực hiện: Nhóm 4. Lớp thứ 5, tiết 1-4**

Lê Đức Việt - 23162113  
 Nguyễn Văn Tâm - 23162088  
 Lê Trí Viễn - 23162112

Phạm Tiến Đạt - 23162018

Thành Phố Hồ Chí Minh, Tháng 11 Năm 2024

|  |  |
| --- | --- |
| **ĐH SƯ PHẠM KỸ THUẬT TP.HCM**  **KHOA CNTT**  \*\*\*\*\*\*\* | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Độc lập – Tự do – Hạnh Phúc**  \*\*\*\*\*\*\* |

**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

Họ và tên Sinh viên 1 : Lê Đức Việt MSSV 1: 23162113

Họ và tên Sinh viên 2 : Nguyễn Văn Tâm MSSV 2: 23162088

Họ và tên Sinh viên 3: Lê Trí Viễn MSSV 3: 23162112

Họ và tên Sinh viên 4: Phạm Tiến Đạt MSSV 4: 23162018

Ngành: An toàn Thông tin

Tên đề tài: Ứng dụng tiện ích, đa chức năng cho máy tính.

Họ và tên Giáo viên hướng dẫn: TS. Phan Thị Thể

**NHẬN XÉT**

1. Nội dung đề tài & khối lượng thực hiện

1. Khuyết điểm

1. Đề nghị cho bảo vệ hay không ?
2. Đánh giá loại :
3. Điểm :

Tp*. Hồ Chí Minh, ngày tháng 11 năm 2024*

Giáo viên hướng dẫn

*(Ký & ghi rõ họ tên)*

**KẾ HOẠCH THỰC HIỆN**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Thời gian** | **Công việc** | **Ghi chú** |
| **1** | 15/10 – 20/10 | Tìm hiểu và chọn đề tài. Phân tích yêu cầu và xác định mục tiêu đồ án. | Tìm hiểu đề tài và các yêu cầu kỹ thuật cần thiết để chuẩn bị kiến thức. |
| **2** | 21/10 – 23/10 | Nghiên cứu tài liệu và thu thập thông tin. | Xác định nguồn tài liệu và dữ liệu |
| **3** | 24/10 – 26/10 | Xây dựng và phân tích mô hình lý thuyết. | Thiết kế sợ lược cách chạy của đồ án. |
| **4** | 27/10 – 29/10 | Thiết kế hệ thống: giao diện hiển thị, cơ sở dữ liệu, luồng xử lý. | Họp nhóm trên gg meet thống nhất nội dung. |
| **5** | 30/10 – 10/11 | Lập trình và triển khai chức năng | Tiến hành thực hiện hóa ứng dụng. |
| **6** | 11/11 – 14/11 | Kiểm thử sản phẩm | Đảm bảo hệ thống hoạt động đúng yêu cầu, kiểm thử các tính năng |
| **7** | 15/11 – 20/11 | Viết báo cáo và hoàn thiện đồ án | Hoàn thành các công đoạn cuối cùng, viết báo cáo chi tiết, cụ thể cho bài báo cáo. |

|  |  |  |
| --- | --- | --- |
| Ý kiến giáo viên phản biện |  | Người viết kế hoạch |
| *(ký và ghi rõ họ tên)* |  | *(ký và ghi rõ họ tên)* |
|  |  | *Việt Lê Đức Việt* |
|  |  |  |

**BẢNG PHÂN CÔNG VIỆC, ĐÁNH GIÁ**

|  |  |  |  |
| --- | --- | --- | --- |
| **Họ và tên** | **Công việc** | **Thời gian thực hiện** | **Đánh giá** |
| Lê Đức Việt 23162113 | Phân tích, thiết kế, tiến hành thực hiện hóa ứng dụng (code giao diện chính, hàm dự báo thời tiết), viết báo cáo word (phân tích hoạt động của code, ...), làm ppt báo cáo. | Từ ngày 27/10/2024 đến ngày 11/11/2024 | Hoàn thành 100% |
| Nguyễn Văn Tâm 23162088 | Phân tích, thiết kế, tiến hành thực hiện hóa ứng dụng (code hàm xem lịch, đồng hồ), viết báo cáo word (đặt tả yêu cầu, cơ sở lí thuyết, ...), làm ppt báo cáo. | Từ ngày 27/10/2024 đến ngày 11/11/2024 | Hoàn thành 100% |
| Lê Trí Viễn  23162112 | Phân tích, thiết kế, tiến hành thực hiện hóa ứng dụng (code hàm chụp ảnh màn hình), viết báo cáo word (phân tích cơ sở lí thuyết cần dùng), làm ppt báo cáo. | Từ ngày 27/10/2024 đến ngày 11/11/2024 | Hoàn thành 100% |
| Phạm Tiến Đạt 23162018 | Phân tích, thiết kế, tiến hành thực hiện hóa ứng dụng (code chức năng trò chơi, hàm thoát ứng dụng, vào file), viết báo cáo word (phần kết quả đạt được, kết luận), làm ppt báo cáo. | Từ ngày 27/10/2024 đến ngày 11/11/2024 | Hoàn thành 100% |

**MỤC LỤC**

**[PHẦN MỞ ĐẦU 1](#_Toc9070)**

[1. Lý do chọn đề tài 1](#_Toc19479)

[2. Cấu trúc đồ án 1](#_Toc22702)

[3. Phương pháp nghiên cứu 2](#_Toc17703)

**[PHẦN NỘI DUNG 3](#_Toc12315)**

**[CHƯƠNG 1 : CƠ SỞ LÍ THUYẾT 3](#_Toc30703)**

[1.1. Giới thiệu tổng quan về python và sơ lược về đồ án. 3](#_Toc27072)

[1.2. Giới thiệu và cách sử dụng của các thư viện sử dụng trong đồ án. 4](#_Toc30099)

[1.2.1. Thư viện tkinter 4](#_Toc14005)

[1.2.2. Thư viện psutil 4](#_Toc27697)

[1.2.3. Thư viện platform 4](#_Toc11526)

[1.2.4. screen\_brightness\_control 4](#_Toc8924)

[1.2.5. ctypes 5](#_Toc17356)

[1.2.6. comtypes 5](#_Toc5971)

[1.2.7. pycaw 5](#_Toc174)

[1.2.8. geopy 5](#_Toc17402)

[1.2.9. timezonefinder 6](#_Toc14960)

[1.2.10. requests 6](#_Toc6832)

[1.2.11. pytz 6](#_Toc24822)

[1.2.12. tkcalendar 7](#_Toc837)

[1.3. API là gì ? 7](#_Toc32356)

**[CHƯƠNG 2 : PHÂN TÍCH VÀ THIẾT KẾ VÀ TIẾN HÀNH CODE 9](#_Toc25238)**

[2.1. Đặc tả yêu cầu và chức năng chính của ứng dụng. 9](#_Toc580)

[2.2. Lên ý tưởng, xác định yêu cầu của người dùng, thiết kế sơ lược giao diện tương tác với người dùng 13](#_Toc6045)

[2.3. Tiến hành thực hiện hóa ứng dụng. 16](#_Toc11993)

[2.4. Giải thích code và cách hoạt động của chương trình. 17](#_Toc1530)

[2.4. Tiến hành chạy, kiểm thử và chỉnh sửa. 37](#_Toc521)

**[CHƯƠNG 3: KẾT QUẢ ĐẠT ĐƯỢC VÀ KẾT LUẬN 38](#_Toc16825)**

[3.1. Code minh họa. 38](#_Toc32381)

[3.1.1. File code minh họa : 38](#_Toc20073)

[3.1.2. file ảnh dung trong chương trình 38](#_Toc7916)

[3.2. Demo chương trình 38](#_Toc13244)

[3.3. Kết quả được đạt: 39](#_Toc19636)

[3.3.1. Giao diện Chính (All\_in\_one Utility): 39](#_Toc31810)

[3.3.2. Ứng dụng Thời Tiết: 40](#_Toc15397)

[3.3.3. Ứng dụng Đồng Hồ và lịch : 41](#_Toc5897)

[3.3.4.Ứng dụng đổi màu giao diện: 41](#_Toc25637)

[3.3.5. Ứng dụng game ludo: 42](#_Toc6947)

[3.3.6. Ứng dụng Chụp Màn Hình (Screenshot): 42](#_Toc13350)

[3.3.7. Trình Quản Lý File (File): 42](#_Toc9305)

[3.3.8. Trình Duyệt Chrome (Chrome): 43](#_Toc633)

[3.3.9. App đọc báo nhanh báo mới. 43](#_Toc31393)

[3.3.10. Thoát Ứng Dụng (Close App): 43](#_Toc16001)

[3.4. Hạn chế. 44](#_Toc523)

[3.4.1. Phụ thuộc vào kết nối mạng cho Dự báo Thời Tiết 44](#_Toc3423)

[3.4.2. Vấn đề về hiệu năng với hình ảnh lớn và tải tài nguyên. 44](#_Toc7996)

[3.4.3. Tính Năng Đổi Chế Độ (Sáng - Tối) Còn Đơn Giản 44](#_Toc27910)

[3.4.4. Phụ thuộc vào các thư viện bên ngoài. 44](#_Toc23739)

[3.4.5. Không lưu trữ trạng thái và tùy chỉnh của người dùng. 44](#_Toc6356)

[3.5. Hướng phát triển. 45](#_Toc16273)

**[PHẦN KẾT LUẬN 46](#_Toc1067)**

**[TÀI LIỆU THAM KHẢO 47](#_Toc28181)**

# MỤC LỤC HÌNH ẢNH

[Hình 2.1 : Giao diện cơ bản của app đa tiện ích. 14](#_Toc7822)

[Hình 2.2 : Giao diện của app dự báo thời tiết khi người dùng tương tác. 15](#_Toc16795)

[Hình 2.3 : Giao diện của tiện ích xem lịch. 15](#_Toc28710)

[Hình 2.4 : Giao diện của trò chơi quay xúc sắc. 15](#_Toc20636)

[Hình 2.5 : Giao diện của xem tiện ích đồng hồ. 16](#_Toc4454)

[Hình 2.6: Khai báo các thư viện dùng trong chương trình 17](#_Toc30006)

[Hình 2.7 : Tạo giao diện chính cho ứng dụng 18](#_Toc12588)

[Hình 2.8 : Hàm covertTime()- hàm chuyển đổi thời gian 21](#_Toc31045)

[Hình 2.9 : Hàm none() - hiển thị các thông số về pin 21](#_Toc22136)

[Hình 2.10 : Hàm lấy giá trị của hệ thống về âm thanh và chỉnh sửa 22](#_Toc25114)

[Hình 2.11 : Hàm lấy giá trị của hệ thống về độ sáng và chỉnh sửa 24](#_Toc19066)

[Hình 2.12 : Hàm thực hiện chức năng dự báo thời tiết cho một thành phố 25](#_Toc20662)

[Hình 2.13 : Cấu trúc của file dữ liệu dự báo thời tiết gửi về (.json) 27](#_Toc7260)

[Hình 2.14 : Hàm thực hiện chức năng hiển thị đồng hồ 28](#_Toc2312)

[Hình 2.15 : Hàm thực hiện chức năng xem lịch 30](#_Toc21527)

[Hình 2.16 : Hàm thực hiện chức năng chuyển đổi chế độ sáng tối 32](#_Toc32739)

[Hình 2.17 : Hàm thực hiện chức năng trò chơi (game Ludo) 34](#_Toc21766)

[Hình 2.18 : Hàm thực hiện chức năng chụp ảnh màn hình 36](#_Toc3518)

[Hình 3.1 : Giao diện chính sau khi hoàn thành 39](#_Toc13856)

[Hình 3.2 : Ứng dụng dự báo thời tiết khi hoàn thành 40](#_Toc5261)

[Hình 3.4 : Ứng dụng xem lịch và đồng hồ 41](#_Toc17664)

[Hình 3.5 : Giao diện khi chuyển đổi chế độ sáng tối 41](#_Toc6438)

[Hình 3.5 : Trò chơi Ludo 42](#_Toc21026)

# LỜI CẢM ƠN

Để hoàn thành bài tiểu luận cuối kỳ này, em xin gửi lời cảm ơn chân thành tới:

BGH Trường Đại Học Sư Phạm Kỹ Thuật HCM đã tạo điều kiện về cơ sở vật chất với hệ thống thư viện hiện đại, đa dạng các loại sách, tài liệu thuận lợi cho việc tìm kiếm, nghiên cứu thông tin.

Xin chân thành cảm ơn giảng viên bộ môn đã giảng dạy tận tình, để nhóm em có đủ kiến thức và vận dụng chúng vào bài tiểu luận này.

Ngoài ra, em xin cảm ơn các bạn cùng tham gia bài tiểu luận, đã nhiệt tình thực hiện và hoàn thành tốt nhiệm vụ được giao cho mỗi bạn để có được bài tiểu luận hoàn chỉnh.

Do chưa có nhiều kinh nghiệm làm đề tài cũng như những hạn chế về kiến thức, trong bài tiểu luận sẽ không tránh khỏi những sai sót, em mong có thể nhận được lời nhận xét của thầy để em có thể hoàn thiện bài và rút kinh nghiệm trong những bài sau.

Lời cuối cùng, em xin kính chúc thầy/cô nhiều sức khỏe, thành công và hạnh phúc.

Em xin chân thành cảm ơn!

Đại diện nhóm

Việt

Lê Đức Việt

# PHẦN MỞ ĐẦU

## 1. Lý do chọn đề tài

Trong thời đại công nghệ hiện nay, các tiện ích và ứng dụng cho máy tính đóng vai trò quan trọng trong việc nâng cao hiệu quả làm việc, tối ưu hóa thời gian và tăng cường trải nghiệm người dùng. Đề tài "Một số tiện ích cho máy tính" nhằm phát triển một ứng dụng cung cấp các tính năng cần thiết và tiện lợi cho người sử dụng. Ứng dụng bao gồm các tiện ích đa dạng như quản lý hệ thống, điều chỉnh âm thanh và độ sáng, cập nhật thời tiết, lịch, chụp màn hình và nhiều chức năng hữu ích khác. Đề tài này được chọn vì nó đáp ứng nhu cầu thực tế, có khả năng ứng dụng cao, và có tiềm năng phát triển thêm nhiều tiện ích khác trong tương lai.

## 2. Cấu trúc đồ án

Đồ án được chia thành các chương như sau:

**Chương 1:** Cơ sở lý thuyết - Giới thiệu về các thư viện và công nghệ được sử dụng trong ứng dụng, như tkinter để tạo giao diện người dùng, psutil để quản lý tài nguyên hệ thống, screen\_brightness\_control để điều chỉnh độ sáng, và các thư viện khác.

**Chương 2:** Phân tích và tiến hành thực hiện đồ án - Mô tả chi tiết về yêu cầu chức năng, thiết kế hệ thống và mô hình giao diện của các tiện ích. Trình bày quá trình cài đặt, các đoạn mã code chính và cách triển khai ứng dụng.

**Chương 3:** Kết quả đạt được và kết luận - Đánh giá hiệu quả của ứng dụng, các tính năng đã hoàn thiện và các hạn chế còn tồn tại. Tóm tắt những kết quả đạt được và đưa ra hướng phát triển trong tương lai.

## 3. Phương pháp nghiên cứu

Để thực hiện đề tài, nhóm sử dụng các phương pháp nghiên cứu sau:

Nghiên cứu lý thuyết: Tìm hiểu các tài liệu về lập trình Python và các thư viện sử dụng trong đồ án như tkinter, psutil, platform, screen\_brightness\_control, pycaw, geopy, timezonefinder, requests, pytz, và tkcalendar.

Phân tích yêu cầu: Xác định các yêu cầu chức năng cần thiết cho ứng dụng tiện ích máy tính nhằm đáp ứng nhu cầu của người dùng.

Thiết kế và phát triển: Sử dụng tkinter để xây dựng giao diện, áp dụng các thư viện khác để cài đặt các chức năng như kiểm tra thông tin hệ thống, quản lý pin, điều chỉnh âm thanh, kiểm tra thời tiết, và nhiều tính năng tiện ích khác.

Kiểm thử và đánh giá: Kiểm thử các chức năng của ứng dụng trên nhiều hệ thống khác nhau để đảm bảo tính ổn định và độ chính xác, sau đó thu thập ý kiến phản hồi để cải tiến và hoàn thiện ứng dụng.

# PHẦN NỘI DUNG

# CHƯƠNG 1 : CƠ SỞ LÍ THUYẾT

## 1.1. Giới thiệu tổng quan về python và sơ lược về đồ án.

Python là một ngôn ngữ lập trình bậc cao, được biết đến với cú pháp đơn giản, dễ hiểu và mạnh mẽ. Python hỗ trợ lập trình hướng đối tượng, hàm, và cả lập trình chức năng. Với thư viện phong phú và cộng đồng người dùng rộng lớn, Python là một công cụ hữu ích cho nhiều lĩnh vực như phát triển ứng dụng, khoa học dữ liệu, học máy, và tự động hóa.

**Sơ lược về đồ án**

Đồ án này là một ứng dụng đa chức năng trên nền tảng Windows, bao gồm các tiện ích hỗ trợ như:

**Giám sát Hệ thống**: Hiển thị các thông tin về hệ thống như tên máy, hệ điều hành, dung lượng RAM, và CPU.

**Điều chỉnh Âm lượng và Độ sáng:** Cho phép người dùng điều chỉnh mức âm lượng và độ sáng màn hình trực tiếp từ giao diện ứng dụng.

**Tiện ích Dự báo Thời tiết:** Người dùng có thể nhập tên thành phố để xem thông tin thời tiết hiện tại như nhiệt độ, độ ẩm, và tốc độ gió.

**Ứng dụng Đồng hồ và Lịch:** Hiển thị đồng hồ số và lịch trong các cửa sổ phụ.

**Ứng dụng Khác:**

Chơi Game Ludo: Một trò chơi đơn giản tích hợp trong ứng dụng.

Chụp Ảnh Màn Hình: Hỗ trợ người dùng chụp và lưu ảnh màn hình.

Truy cập Google và Facebook: Mở trình duyệt và truy cập vào các trang web như Google hoặc Facebook.

**Chuyển Đổi Chế Độ Sáng/Tối:** Cho phép người dùng chuyển đổi giữa hai chế độ giao diện sáng và tối.

Ứng dụng được phát triển bằng tkinter - một thư viện giao diện đồ họa của Python. Sự kết hợp các chức năng này cung cấp một công cụ hữu ích giúp người dùng quản lý hệ thống, giám sát và điều chỉnh thiết bị một cách thuận tiện.

## 1.2. Giới thiệu và cách sử dụng của các thư viện sử dụng trong đồ án.

### 1.2.1. Thư viện tkinter

Tính năng: Cung cấp giao diện người dùng đồ họa (GUI) cho các ứng dụng.

Ứng dụng: Xây dựng các ứng dụng desktop đơn giản với các widget như nút bấm, hộp thoại, menu, khung hiển thị, v.v.

### 1.2.2. Thư viện psutil

Tính năng: Giám sát hệ thống và truy cập thông tin về CPU, RAM, đĩa, mạng.

Ứng dụng: Tạo các công cụ giám sát hệ thống, hiển thị tài nguyên hệ thống trong thời gian thực.

### 1.2.3. Thư viện platform

Tính năng: Lấy thông tin về hệ điều hành và nền tảng đang chạy ứng dụng.

Ứng dụng: Đoán thông tin phần cứng và hệ điều hành để tối ưu hóa hoặc điều chỉnh hoạt động ứng dụng phù hợp.

### 1.2.4. screen\_brightness\_control

**Tính năng**: Điều chỉnh độ sáng màn hình máy tính.

**Ứng dụng**: Quản lý ánh sáng để tối ưu trải nghiệm người dùng.

### 1.2.5. ctypes

**Tính năng**: Gọi các hàm C từ Python.

**Ứng dụng**: Tăng hiệu năng và dùng các thư viện C sẵn có.

**Ví dụ** :



### 1.2.6. comtypes

**Tính năng**: Kết nối với thư viện COM của Windows.

**Ứng dụng**: Điều khiển Microsoft Office, Windows Media Player, v.v.

**Ví dụ** :



### 1.2.7. pycaw

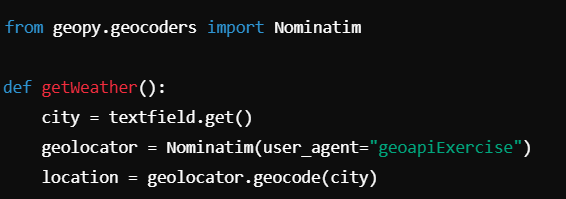
**Tính năng**: Điều chỉnh âm lượng cho từng ứng dụng trên Windows.

**Ứng dụng**: Tạo công cụ quản lý âm thanh tùy chỉnh.

### 1.2.8. geopy

**Tính năng**: Tìm vị trí địa lý và đo khoảng cách.

**Ứng dụng**: Tích hợp bản đồ, định vị người dùng, tìm địa điểm.

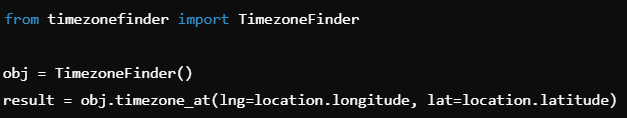
**Ví dụ** :  


### 1.2.9. timezonefinder

**Tính năng**: Xác định múi giờ theo tọa độ.

**Ứng dụng**: Đồng bộ thời gian chính xác trên toàn cầu.

**Ví dụ** :



### 1.2.10. requests

**Tính năng**: Gửi và nhận dữ liệu qua HTTP.

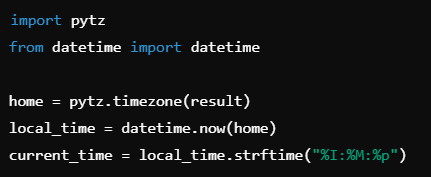
**Ứng dụng**: Gọi API, tự động tải dữ liệu từ web.

### 1.2.11. pytz

**Tính năng**: Làm việc với múi giờ quốc tế.

**Ứng dụng**: Xử lý thời gian đa vùng miền trong ứng dụng.

**Ví dụ** :

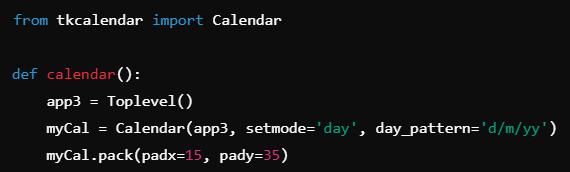


### 1.2.12. tkcalendar

**Tính năng**: Widget chọn ngày tháng cho Tkinter.

**Ứng dụng**: Tạo form nhập liệu với lựa chọn ngày trực quan.

**Ví dụ** :



## 1.3. API là gì ?

**Định nghĩa :**

API (Application Programming Interface) là một tập hợp các quy tắc và giao thức cho phép các phần mềm hoặc ứng dụng khác nhau giao tiếp và tương tác với nhau. API như một cầu nối, giúp các ứng dụng trao đổi dữ liệu hoặc thực hiện các chức năng một cách dễ dàng mà không cần phải biết chi tiết bên trong của hệ thống kia.

Ví dụ để hiểu rõ hơn về chức năng :

Khi bạn sử dụng ứng dụng thời tiết trên điện thoại, ứng dụng này kết nối tới một API của dịch vụ thời tiết để nhận thông tin về nhiệt độ, độ ẩm, và dự báo. API này đóng vai trò như người "phiên dịch" giữa ứng dụng và hệ thống dữ liệu thời tiết.

Ưu điểm của API :

Tích hợp dễ dàng: Giúp ứng dụng kết nối và trao đổi dữ liệu nhanh chóng.

Linh hoạt: Cho phép sử dụng các dịch vụ từ nhiều nguồn khác nhau.

Tiết kiệm: Tận dụng mã nguồn có sẵn, tiết kiệm thời gian và chi phí.

Bảo mật cao: Cung cấp cơ chế xác thực và quyền truy cập.

Trải nghiệm tốt: Tạo sự liền mạch cho người dùng.

Dễ mở rộng: Giúp nâng cấp, thêm tính năng mà không làm gián đoạn hệ thống.

# CHƯƠNG 2 : PHÂN TÍCH VÀ THIẾT KẾ VÀ TIẾN HÀNH CODE

**2.1. Đặc tả yêu cầu và chức năng chính của ứng dụng.**

**2.1.1. Lên ý tưởng :**

Ứng dụng được phát triển bằng Python, sử dụng các thư viện như tkinter, psutil, và platform để cung cấp các tính năng hữu ích cho người dùng. Python là một ngôn ngữ lập trình mạnh mẽ, có thể hỗ trợ nhiều thư viện và công cụ tiện ích giúp tối ưu hóa việc phát triển các ứng dụng. Nhóm đã quyết định sử dụng thư viện tkinter để tạo giao diện đồ họa đơn giản, dễ sử dụng, với khả năng hiển thị thông tin về hệ thống máy tính (như CPU, RAM, và tình trạng pin), điều chỉnh độ sáng màn hình, âm lượng, và nhiều tính năng hữu ích khác.

Ngoài các tính năng cơ bản, nhóm cũng phát triển thêm một số tiện ích như dự báo thời tiết, chơi game, xem lịch, đồng hồ, và các ứng dụng giải trí khác. Các tính năng này giúp tạo ra một ứng dụng đa chức năng và dễ sử dụng, phục vụ nhu cầu của người dùng một cách nhanh chóng và hiệu quả.

**2.1.2. Yêu cầu của người dùng:**

Ứng dụng cần phải dễ sử dụng, có giao diện trực quan và các tính năng dễ tiếp cận. Người dùng muốn ứng dụng có thể cung cấp thông tin chính xác về hệ thống máy tính của họ, chẳng hạn như trạng thái pin, nhiệt độ CPU, thời gian sử dụng còn lại của pin, cũng như các thông tin về phần cứng máy tính. Các tính năng của ứng dụng cần được cập nhật một cách nhanh chóng và chính xác.

Giao diện người dùng cần phải đẹp mắt và dễ nhìn, không gây ảnh hưởng đến các ứng dụng khác đang chạy trên máy tính.

**2.1.3.** **Yêu Cầu Chức Năng :**

**Ứng dụng sẽ bao gồm các chức năng chính sau:**

**Giám sát thông tin hệ thống máy tính:**

Hiển thị thông số hệ thống: Cung cấp các thông tin cơ bản về hệ thống máy tính người dùng như:

Tên máy (hostname), hệ điều hành (OS), và các thông số kỹ thuật khác.

Thông tin CPU: Tần số, số lõi, mức độ sử dụng, nhiệt độ (nếu có).

Thông tin RAM: Dung lượng tổng, dung lượng đã sử dụng, dung lượng còn lại.

Thông tin về ổ đĩa cứng: Tổng dung lượng, dung lượng đã sử dụng và dung lượng còn lại.

**Giám sát tình trạng pin:**

Hiển thị thông tin pin: Phần trăm pin còn lại, thời gian sử dụng còn lại và trạng thái sạc (đang sạc hay không).

**Điều chỉnh các thiết bị phần cứng:**

Điều chỉnh độ sáng màn hình: Cho phép người dùng thay đổi độ sáng màn hình thông qua thanh trượt.

Điều chỉnh âm lượng: Cho phép người dùng điều chỉnh âm lượng hệ thống thông qua thanh trượt.

**Dự báo thời tiết:**

Nhập tên thành phố: Người dùng nhập tên thành phố và ứng dụng sẽ hiển thị thông tin thời tiết hiện tại của thành phố đó.

Thông tin thời tiết: Nhiệt độ, tình trạng thời tiết (nắng, mưa, v.v.), tốc độ gió, độ ẩm, áp suất.

**Hiển thị đồng hồ và lịch:**

Đồng hồ: Hiển thị thời gian chính xác (giờ, phút, giây) theo thời gian thực.

Lịch: Hiển thị lịch theo tháng, năm, cho phép tìm kiếm và thao tác ngày tháng dễ dàng.

**Ứng dụng giải trí:**

Trò chơi quay xúc xắc: Người dùng có thể chơi trò chơi quay xúc xắc ngay trong ứng dụng.

Đọc báo: Cho phép người dùng duyệt web đọc tin tức hoặc tìm kiếm thông tin qua trình duyệt tích hợp trong ứng dụng.

**2.1.4. Yêu Cầu Giao Diện Người Dùng (UI)**

**Giao diện chính:** Giao diện người dùng sẽ chia thành ba phần chính:

**Phần 1:** Hiển thị thông số kỹ thuật của hệ thống máy tính (CPU, RAM, ổ đĩa, hệ điều hành, v.v.).

**Phần 2:** Hiển thị tình trạng pin và các thông tin về pin (phần trăm còn lại, thời gian sử dụng còn lại).

**Phần 3:** Các tiện ích con như dự báo thời tiết, xem đồng hồ, xem lịch, trò chơi, đọc báo, v.v.

**Giao diện tiện ích con:** Mỗi khi người dùng chọn một tiện ích (ví dụ: Dự báo thời tiết), ứng dụng sẽ mở một cửa sổ mới, cho phép người dùng tương tác với tiện ích đó. Giao diện các tiện ích con cần phải đơn giản, dễ sử dụng và thân thiện với người dùng.

**Tính dễ sử dụng:** Giao diện phải trực quan và dễ hiểu, giảm thiểu các thao tác phức tạp để người dùng có thể dễ dàng truy cập và sử dụng các tính năng của ứng dụng.

**2.1.5. Yêu Cầu Kỹ Thuật và Công Cụ Sử Dụng**

**Ngôn ngữ lập trình:** Python 3.x, được chọn vì dễ sử dụng, dễ tiếp cận với các thư viện mạnh mẽ và phổ biến.

**Thư viện và công cụ chính:**

**tkinter:** Thư viện chính để xây dựng giao diện đồ họa (GUI) cho ứng dụng.

**psutil:** Thư viện dùng để giám sát hệ thống, lấy thông tin về CPU, RAM, ổ đĩa và tình trạng pin.

**platform:** Lấy thông tin hệ điều hành và cấu hình máy tính.

**requests:** Dùng để gửi yêu cầu HTTP, lấy dữ liệu từ các API (ví dụ: API thời tiết).

**geopy và timezonefinder:** Xác định vị trí và múi giờ.

**pycaw, screen\_brightness\_control:** Điều chỉnh âm lượng và độ sáng màn hình.

**tkcalendar:** Hiển thị lịch trong ứng dụng.

**pyautogui và random:** Cung cấp các chức năng điều khiển chuột, tạo số ngẫu nhiên cho các trò chơi (xúc xắc).

**Yêu cầu về hiệu suất:** Ứng dụng phải hoạt động mượt mà và không ảnh hưởng đến hiệu suất hệ thống hoặc các ứng dụng khác đang chạy.

**Tính mở rộng:** Ứng dụng cần được thiết kế sao cho dễ dàng mở rộng với các tính năng mới trong tương lai mà không cần thay đổi nhiều mã nguồn hiện tại.

**2.1.6. Yêu Cầu An Toàn và Bảo Mật**

**Bảo mật thông tin người dùng:** Ứng dụng không thu thập thông tin cá nhân của người dùng, không lưu trữ dữ liệu nhạy cảm.

**Quản lý lỗi và thông báo:** Ứng dụng phải có khả năng thông báo cho người dùng khi gặp sự cố (ví dụ: không thể kết nối đến API thời tiết, lỗi hệ thống, v.v.) và xử lý lỗi hợp lý.

## 2.2. Lên ý tưởng, xác định yêu cầu của người dùng, thiết kế sơ lược giao diện tương tác với người dùng

**Lên ý tưởng :**

Python là một ngôn ngữ lập trình bậc cao, do đó nó hổ trợ cho người dùng rất nhiều thư viện tiện lợi, giúp người dùng có thể dễ dàng sử dụng.

Tận dùng điều đó, nhóm chúng em có nhiều ý tưởng cho đồ án cuối kì lần này. Sử dụng thư viện tkinter để tạo giao diện cơ bản, đơn giản, đẹp mắt, dùng thư viện psutil, platform giám sát hệ thống và lấy các thông tin của CPU, dùng một số thư viện để điều chỉnh độ sáng tối của màn hình, điều chỉnh âm lượng, quan sát trạng thái của pin máy tính, ...

Trên đó là những ý tưởng cơ bản để xây dựng ứng dụng ban đầu, trong quá trình thực hiện đồ án, nhóm đã phát sinh thêm nhiều ý tưởng như phát triển thêm tiện tích dự báo thời tiết, chơi game, ...

**Yêu cầu của người dùng:**

Ứng dụng dễ sử dụng, đáp ứng các nhu cầu của người dùng nhanh chóng, cập nhật thông tin chính xác, cụ thể, rõ ràng. Hiển thị cụ thể các thông số kĩ thuật của máy tính người dùng, tình trạng pin, thời gian sử dụng còn lại của pin, ...

Giao diện hài hòa, thích hợp, không gây ảnh hướng đến các phần mềm, ứng dụng khác.

Các chức năng cụ thể :

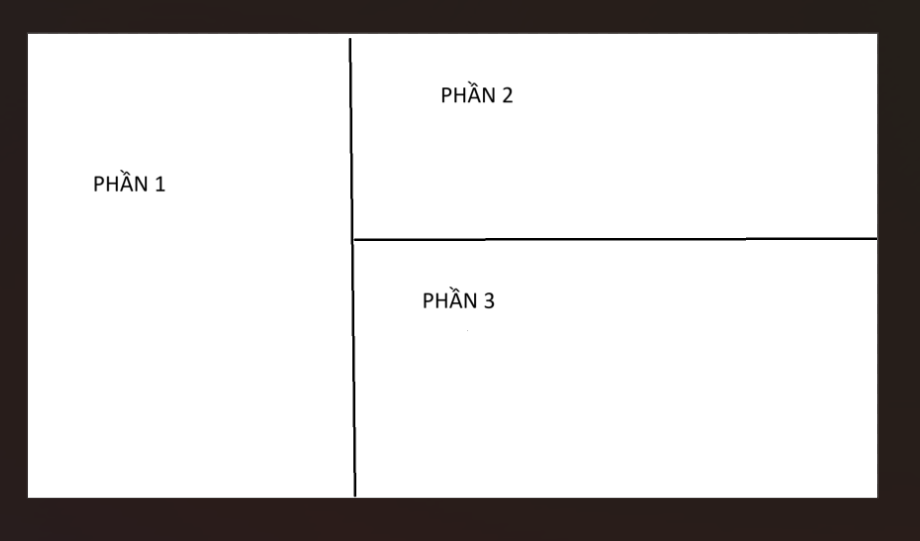
- Dự báo thời tiết : nhập vào tên của một thành phố bất kỳ, cho biết nhiệt độ hiện tại, thực trạng nắng mưa, tốc độ gió, đổ ẩm, áp suất, ...

- Xem lịch, đồng hồ hiển thị : thao tác trên giao điện để xem lịch trong các tháng, hoặc các năm.

- Các ứng dụng giải trí : đọc báo, trò chơi, yêu cầu thao tác dễ dàng, dễ sử dụng, tiện lợi, truy cập vào web tìm kiếm thông tin cần thiết, ...

**Thiết kế giao diện:**

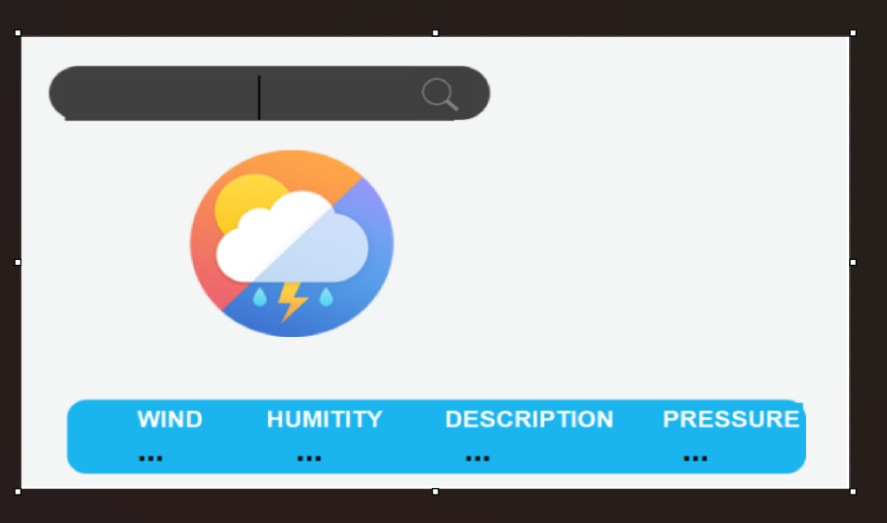
Giao diện được chia làm 3 phần chính cơ bản: **phần 1** dùng để chứa các thông số của máy tính người dùng, **phần 2** chứa các thông tin như thời lượng pin(%), thời gian sửa dụng còn lại, ..., **phần 3** chứa các tiện ích, app con cho người dùng tương tác. Có 9 tiện ích con bên trong app bao gồm : dự báo thời tiết của 1 thành phố trên thế giới, xem đồng hồ, xem lịch, trò chơi quay xúc sắc, chụp ảnh màn hình, trình duyệt tìm kiếm, web đọc báo, ...



Hình 2.1 : Giao diện cơ bản của app đa tiện ích.

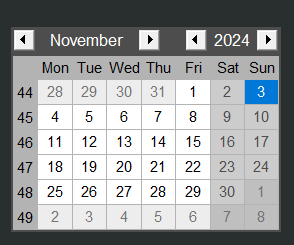
Khi người dùng chọn vào 1 tiện ích trong phần 3 thì ứng dụng sẽ tiếp tục xuất hiện ra màn hình cửa sổ giúp người dùng có thể tương tác với tiện ích đó.

Ví dụ : trong tiện ích sẽ có app dự báo thời tiết cho 1 thành phố nào đó tại thời điểm ngay lúc đó, cửa sổ mới hiện ra và có giao diện được thiết kế như hình bên dưới. Người dùng chỉ cần nhập tên thành phố, tìm kiếm và quan sát kết quả.

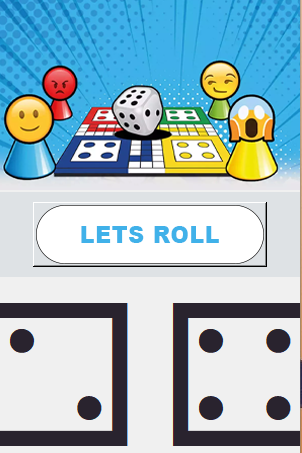


Hình 2.2 : Giao diện của app dự báo thời tiết khi người dùng tương tác.

Tương tự như thế giao diện của một số tiện ích con trong ứng dụng được thiết kế như các hình ảnh bên dưới:



Hình 2.3 : Giao diện của tiện ích xem lịch.



Hình 2.4 : Giao diện của trò chơi quay xúc sắc.



Hình 2.5 : Giao diện của xem tiện ích đồng hồ.

## 2.3. Tiến hành thực hiện hóa ứng dụng.

Tiến hành thực hiện theo các bước :

**Bước 1 :** Xác định đây là một ứng dụng giao diện đồ họa (GUI).

**Bước 2 :** Tham khảo các mẫu hình ảnh cần sử dụng trong ứng dụng và tổng hợp.

**Bước 3 :** Tiến hành code phần giao diện chính cho ứng dụng (thực hiện theo từng phần 1,2,3,...) bao gồm các hình ảnh, kích thước của cửa sổ để thao tác, các label, các ô văn bản, các nút, ...

**Bước 4:** Tiến hành code phần giao diện cho các tiện ích con (các thành phần cũng tương tự như trên, ...)

**Bước 5 :** Xây dựng từng chức năng, từng tiện ích theo từng hàm dễ dàng chỉnh sửa và lưu trữ.

## 2.4. Giải thích code và cách hoạt động của chương trình.

**Khai báo các thư viện cần thiết :**



Hình 2.6: Khai báo các thư viện dùng trong chương trình

- tkinter: Thư viện dùng để tạo giao diện người dùng (GUI) trên Python. Cung cấp các công cụ để tạo cửa sổ, nút bấm, nhãn, v.v.

- ttk: Một phần mở rộng của tkinter, cung cấp các widget với giao diện đẹp và phong cách hiện đại hơn.

- messagebox: Thư viện con trong tkinter, hiển thị hộp thoại thông báo cho người dùng.

- filedialog: Dùng để mở hộp thoại chọn tệp tin từ hệ thống.

- platform: Lấy thông tin về hệ thống (như hệ điều hành, phiên bản).

- psutil: Thư viện giám sát hệ thống, cho phép truy cập vào thông tin về CPU, bộ nhớ, đĩa, và mạng.

- screen\_brightness\_control: Thư viện dùng để điều chỉnh độ sáng màn hình.

- ctypes, comtypes, và pycaw: Các thư viện hỗ trợ thao tác âm thanh trên hệ điều hành Windows.

- geopy: Thư viện hỗ trợ lấy vị trí địa lý (địa chỉ, tọa độ).

- timezonefinder: Thư viện xác định múi giờ từ vị trí.

- datetime: Làm việc với thời gian và ngày tháng.

- requests: Thư viện gửi yêu cầu HTTP, dùng để lấy dữ liệu từ API thời tiết.

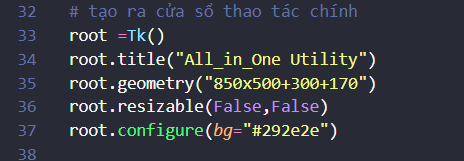
- pytz: Quản lý múi giờ.

- strftime: Định dạng thời gian thành chuỗi.

- tkcalendar: Một tiện ích lịch trong tkinter.

- pyautogui, subprocess, webbrowser, và random: Các thư viện cung cấp các chức năng điều khiển chuột, mở chương trình, mở trình duyệt, và tạo số ngẫu nhiên.

**Tạo cửa sổ chính và cài đặt giao diện :**



Hình 2.7 : Tạo giao diện chính cho ứng dụng

- root = Tk(): Tạo cửa sổ chính của ứng dụng.

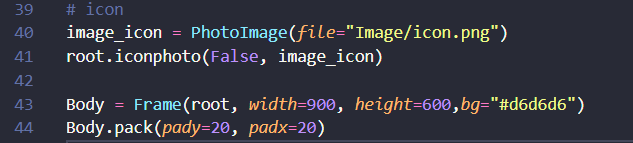
- root.title("All\_in\_One Utility"): Đặt tiêu đề cho cửa sổ.

- root.geometry("850x500+300+170"): Đặt kích thước và vị trí của cửa sổ.

- root.resizable(False, False): Vô hiệu hóa khả năng thay đổi kích thước.

- root.configure(bg="#292e2e"): Đặt màu nền cho cửa sổ.

**Thiết lập icon và khung chính:**

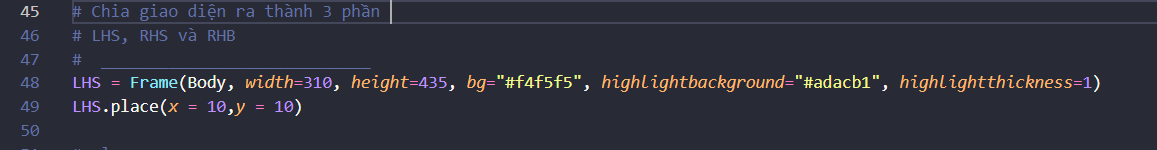


- PhotoImage(file="Image/icon.png"): Tải ảnh từ tệp icon.png làm icon cho ứng dụng.

- Frame(root, width=900, height=600, bg="#d6d6d6"): Tạo một khung (frame) trong cửa sổ chính để chứa các thành phần của ứng dụng.

- Body.pack(pady=20, padx=20): Đặt khung Body vào cửa sổ chính với khoảng đệm xung quanh.

**Tạo khung bên trái (LHS) và đặt các nhãn thông tin hệ thống:**



- Frame(Body,width=310,height=435, bg="#f4f5f5", highlightbackground="#adacb1", highlightthickness=1): Tạo một khung con bên trong Body để hiển thị thông tin hệ thống.

- place(x=10, y=10): Đặt khung này ở vị trí (10, 10) trong Body.

**Logo và Thông tin hệ thống :**



- photo = PhotoImage(file="Image/laptop.png"): Tải ảnh logo từ tệp laptop.png.

- Label(LHS, image=photo, background="#f4f5f5"): Tạo nhãn chứa hình ảnh logo.

- place(x=2, y=20): Đặt nhãn logo trong khung LHS.

**my\_system = platform.uname()**



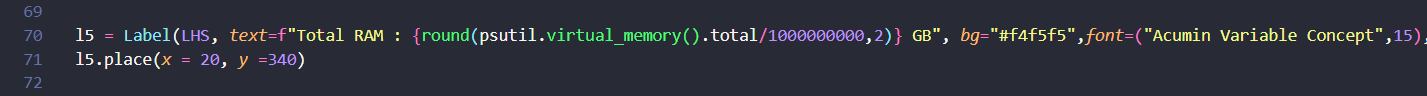
- platform.uname(): Lấy thông tin về hệ thống hiện tại (máy chủ, phiên bản, hệ điều hành, loại máy).



- Label(LHS, text=my\_system.node, ...): Tạo nhãn hiển thị tên máy.

- place(x=20, y=200): Đặt nhãn này ở vị trí (20, 200) trong LHS.

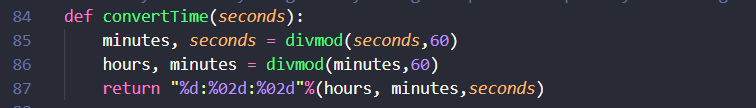
- Tương tự như trên, các nhãn l2, l3, l4, và l5 lần lượt hiển thị phiên bản hệ thống, hệ điều hành, loại máy, và tổng dung lượng RAM.



- psutil.virtual\_memory().total: Lấy tổng dung lượng RAM trong hệ thống.

- round(..., 2): Làm tròn dung lượng RAM thành số thập phân với 2 chữ số.

**Hàm convertTime() :**



Hình 2.8 : Hàm covertTime()- hàm chuyển đổi thời gian

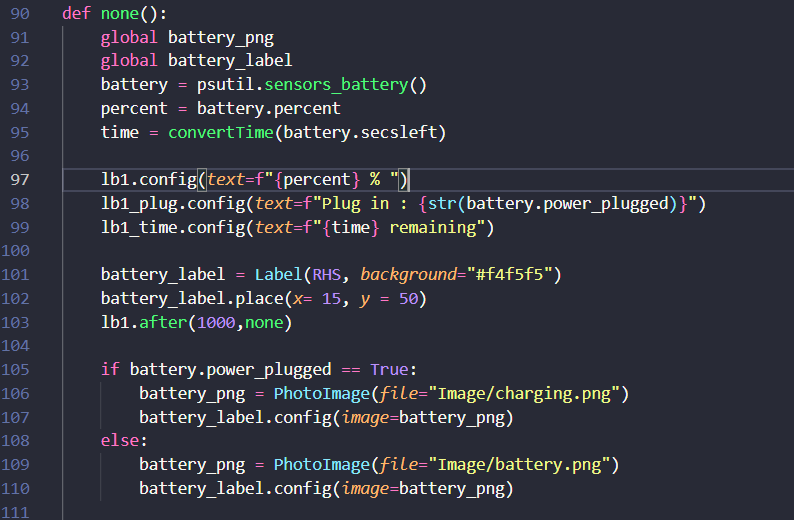
**- Yêu cầu** : Hàm này chuyển đổi thời gian từ giây sang định dạng giờ phút giây, từ đó có thẻ hiển thị ra giao diện cho người dùng quan sát.

- **Cách hoạt động**:

+ minutes, seconds = divmod(seconds, 60),hours, minutes = divmod(minutes, 60) : Sử dụng hàm divmod để chia số giây thành phút và giây, sau đó chia phút thành giờ và phút.

+ return "%d:%02d:%02d" % (hours, minutes, seconds) : Trả về chuỗi định dạng giờ, phút, giây với các giá trị số được định dạng cho đúng (ví dụ: luôn có hai chữ số cho phút và giây).

**Hàm none() :**



Hình 2.9 : Hàm none() - hiển thị các thông số về pin

**- Yêu cầu** : Hàm này truy cập vào hệ thống lấy thông tin kĩ thuật của pin như phần trăm còn lại, thời lượng sử dụng còn lại và cập nhật giao diện người dùng.

**- Cách hoạt động**:

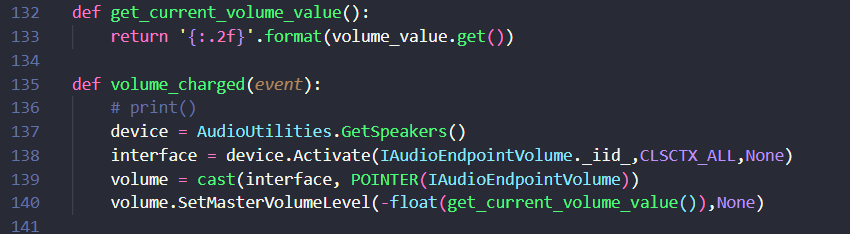
+ battery = psutil.sensors\_battery() : Sử dụng thư viện psutil để lấy thông tin về pin, bao gồm phần trăm pin và thời gian còn lại.

+ percent = battery.percent và time = convertTime(battery.secsleft): Cập nhật các nhãn trên giao diện với thông tin pin.

+ lb1.config(text=f"{percent} % "),lb1\_plug.config(text=f"Plug in : {str(battery.power\_plugged)}"): Thiết lập hình ảnh của pin dựa trên trạng thái sạc (đang sạc hay không).

+ lb1.after(1000, none) : Gọi lại chính nó sau 1000 ms (1 giây) để liên tục cập nhật thông tin.

**Hàm get\_current\_volume\_value và hàm volume\_changed**



Hình 2.10 : Hàm lấy giá trị của hệ thống về âm thanh và chỉnh sửa

**- Yêu cầu** : Hàm này truy cập vào hệ thống và lấy giá trị âm lượng hiện tại và định dạng nó, sau đó giúp cho người dùng có thể thay đổi âm lượng khi trượt thanh âm lượng.

- **Cách hoạt động**:

+ volume\_value.get(): Lấy giá trị hiện tại từ biến liên kết với thanh trượt (mức âm lượng hiện tại).

+ '{:.2f}'.format(...): Định dạng giá trị thành chuỗi thập phân có 2 chữ số (ví dụ: 20.00).

+ device = AudioUtilities.GetSpeakers():

Lấy thông tin về thiết bị loa (speakers) hiện tại.

+ interface = device.Activate(...):

Kích hoạt giao diện IAudioEndpointVolume, giúp điều chỉnh âm lượng của thiết bị.

+ volume = cast(interface, POINTER(IAudioEndpointVolume)):

Chuyển đổi giao diện thành đối tượng IAudioEndpointVolume để có thể sử dụng các phương thức điều chỉnh âm lượng.

+ volume.SetMasterVolumeLevel(-float(get\_current\_volume\_value()), None):

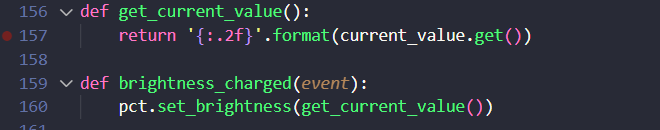
Cập nhật mức âm lượng:

-float(get\_current\_volume\_value()): Lấy giá trị hiện tại từ thanh trượt, chuyển thành số thực và đảo dấu (do API yêu cầu giá trị âm cho mức âm lượng).

None: Giá trị không cần thay đổi tham số bổ sung.

+ Đặt mức âm lượng chính bằng giá trị âm lượng được lấy từ thanh trượt (được định dạng để là số âm).

**Hàm get\_current\_value và brightness\_charged:**



Hình 2.11 : Hàm lấy giá trị của hệ thống về độ sáng và chỉnh sửa

**- Yêu cầu** : Hàm này truy cập vào hệ thống, lấy thông tin về trạng thái của màn hình hiện tại, định dạng lại và thay đổi độ sáng khi thanh trượt độ sáng được điều chỉnh.

- Cách hoạt động:

**def get\_current\_value():**

Hàm này trả về giá trị hiện tại của thanh trượt điều chỉnh độ sáng (current\_value) dưới dạng chuỗi định dạng có hai chữ số thập phân.

+ current\_value.get(): Lấy giá trị hiện tại từ biến liên kết với thanh trượt.

+ '{:.2f}'.format(...): Định dạng giá trị thành chuỗi có 2 chữ số thập phân (ví dụ: 50.00).

**def brightness\_charged(event):**

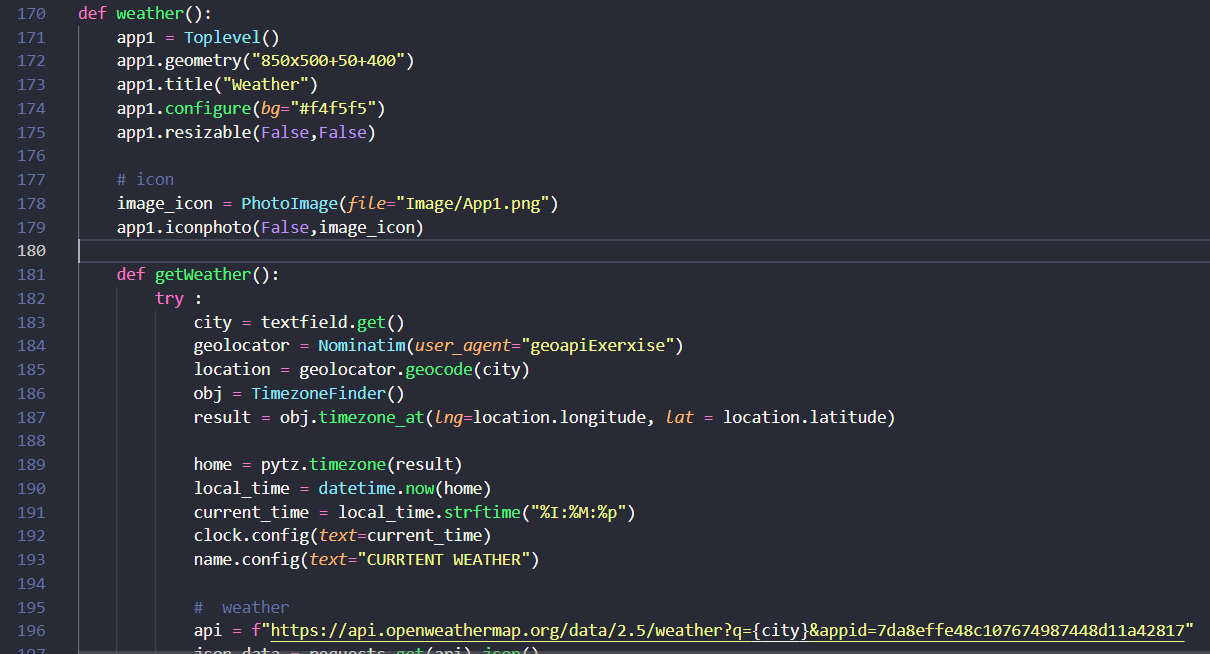
Hàm này được gọi mỗi khi giá trị của thanh trượt độ sáng thay đổi. Nó điều chỉnh độ sáng màn hình dựa trên giá trị từ thanh trượt.

+ pct.set\_brightness(get\_current\_value()):

+ Gọi phương thức set\_brightness() của đối tượng pct để đặt độ sáng màn hình.

+ Giá trị đầu vào được lấy từ get\_current\_value(), chuyển đổi giá trị trên thanh trượt thành định dạng cần thiết cho phương thức set\_brightness.

**Hàm weather:**



Hình 2.12 : Hàm thực hiện chức năng dự báo thời tiết cho một thành phố

**- Yêu cầu** : Hàm này cho người dùng nhập vào một tên thành phố, sau đó hệ thống sẽ gửi đi một đường dẫn url (api) để lấy thông tin về tình hình thời tiết hiện tại của thành phố đó, dữ liệu sẽ được gửi về hệ thống với định dạng fille .json, hệ thống tiến hành xử lí dữ liệu và hiẻn thị cho người dùng, cửa sổ mới để hiển thị thông tin thời tiết.

- **Cách hoạt động**:

**Giao diện chính của ứng dụng thời tiết**

+ app1 = Toplevel(): Tạo một cửa sổ mới trên giao diện chính (sử dụng Toplevel để tạo một cửa sổ phụ).

+ app1.geometry("850x500+50+400"): Đặt kích thước của cửa sổ là 850x500 pixel và vị trí hiển thị trên màn hình là tọa độ (50, 400).

+ app1.title("Weather"): Đặt tiêu đề của cửa sổ là "Weather".

+ app1.configure(bg="#f4f5f5"): Đặt màu nền của cửa sổ thành màu xám nhạt (#f4f5f5).

+ app1.resizable(False, False): Tắt chức năng thay đổi kích thước của cửa sổ.

**Icon của ứng dụng**

+ image\_icon = PhotoImage(file="Image/App1.png"): Tải một hình ảnh từ tệp "Image/App1.png" làm biểu tượng của ứng dụng.

+ app1.iconphoto(False, image\_icon): Đặt hình ảnh làm biểu tượng của cửa sổ ứng dụng.

**Hàm getWeather()**

Hàm này thực hiện các bước sau để lấy thông tin thời tiết của thành phố người dùng nhập vào:

Lấy thông tin địa lý của thành phố

+ city = textfield.get(): Lấy tên thành phố từ trường nhập liệu (textfield).

+ geolocator = Nominatim(user\_agent="geoapiExerxise"): Sử dụng thư viện geopy để tạo đối tượng định vị (Nominatim) phục vụ tìm kiếm thông tin địa lý.

+ location = geolocator.geocode(city): Lấy thông tin vị trí (kinh độ, vĩ độ) của thành phố.

+ obj = TimezoneFinder(): Sử dụng thư viện timezonefinder để tìm múi giờ dựa trên tọa độ.

+ result = obj.timezone\_at(lng=location.longitude, lat=location.latitude): Trả về múi giờ của thành phố.

**Lấy thời gian địa phương**

+ home = pytz.timezone(result): Tạo đối tượng múi giờ dựa trên kết quả tìm được.

+ local\_time = datetime.now(home): Lấy thời gian hiện tại của thành phố.

+ current\_time = local\_time.strftime("%I:%M:%p"): Định dạng thời gian theo kiểu 12 giờ (AM/PM).

+ clock.config(text=current\_time): Cập nhật thời gian trên giao diện.

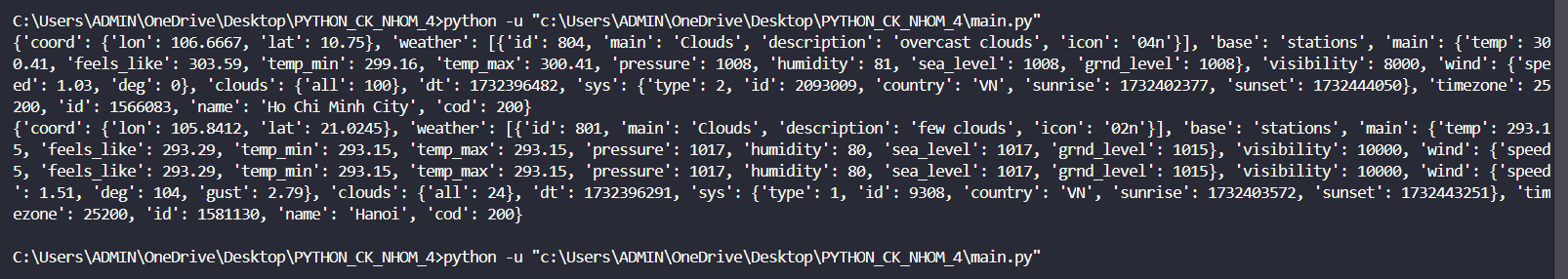
+ name.config(text="CURRENT WEATHER"): Đặt tiêu đề là "CURRENT WEATHER".

**Gọi API thời tiết**

+ api = f"https://api.openweathermap.org/data/2.5/weather?q={city}&appid=7da8effe48c107674987448d11a42817": URL API lấy thông tin thời tiết từ OpenWeatherMap, sử dụng tên thành phố làm tham số.

+ json\_data = requests.get(api).json(): Gửi yêu cầu đến API và nhận dữ liệu thời tiết ở định dạng JSON.

**Xử lý dữ liệu thời tiết**



Hình 2.13 : Cấu trúc của file dữ liệu dự báo thời tiết gửi về (.json)

**Trích xuất các thông tin từ JSON:**

+ condition = json\_data['weather'][0]['main']: Điều kiện thời tiết chính (ví dụ: "Cloudy").

+ description = json\_data['weather'][0]['description']: Mô tả chi tiết về thời tiết.

+ temp = int(json\_data['main']['temp'] - 273.15): Nhiệt độ hiện tại (chuyển đổi từ Kelvin sang độ C).

+ pressure = json\_data['main']['pressure']: Áp suất không khí.

+ humidity = json\_data['main']['humidity']: Độ ẩm.

+ wind = json\_data['wind']['speed']: Tốc độ gió.

**Cập nhật giao diện:**

+ t.config(text=(temp, "°C")): Hiển thị nhiệt độ.

+ c.config(text=(condition, "|", "FEELS LIKE", temp, "°C")): Hiển thị điều kiện thời tiết và cảm nhận nhiệt độ.

+ w.config(text=wind): Hiển thị tốc độ gió.

+ h.config(text=humidity): Hiển thị độ ẩm.

+ d.config(text=description): Hiển thị mô tả thời tiết.

+ p.config(text=pressure): Hiển thị áp suất.

**Xử lý lỗi**

+ except Exception as e:: Nếu có lỗi (ví dụ: nhập sai tên thành phố), hiển thị hộp thoại báo lỗi.

+ messagebox.showerror("Weather App", "Giá trị nhập vào không hợp lệ"): Hiển thị thông báo lỗi "Giá trị nhập vào không hợp lệ".

**Hàm clock :**



Hình 2.14 : Hàm thực hiện chức năng hiển thị đồng hồ

**- Yêu cầu** : Hàm này hiẻn thị đầy đủ thông tin giờ, phút, giây tịa thời điểm hiện tại, và mở một cửa sổ mới hiển thị đồng hồ.

- **Cách hoạt động**:

Hàm clock() này tạo một ứng dụng nhỏ hiển thị đồng hồ số với giao diện đơn giản. Dưới đây là phân tích từng thành phần và cách hoạt động của hàm:

**Giao diện chính của ứng dụng đồng hồ**

+ app2 = Toplevel(): Tạo một cửa sổ con mới (Toplevel) cho giao diện đồng hồ.

+ app2.geometry("850x110+300+10"): Đặt kích thước cửa sổ là 850x110 pixel, vị trí hiển thị tại tọa độ (300, 10) trên màn hình.

+ app2.title("Clock"): Đặt tiêu đề của cửa sổ là "Clock".

+ app2.configure(bg="#292e2e"): Đặt màu nền của cửa sổ là màu xám đậm (#292e2e).

+ app2.resizable(False, False): Vô hiệu hóa khả năng thay đổi kích thước cửa sổ.

**Icon của ứng dụng**

+ image\_icon = PhotoImage(file="Image/App2.png"): Tải biểu tượng từ tệp hình ảnh "Image/App2.png".

+ app2.iconphoto(False, image\_icon): Đặt hình ảnh này làm biểu tượng của cửa sổ đồng hồ.

**Hàm clock()**

Hàm con clock() bên trong được gọi đệ quy để cập nhật thời gian mỗi giây:

+ text = strftime('%H:%M:%S:%p'): Sử dụng hàm strftime từ thư viện time để lấy thời gian hiện tại: %H:%M:%S (hiển thị giờ, phút, giây ở định dạng 24 giờ ) , %p (hiển thị AM hoặc PM ) .

+ lb1.config(text=text): Cập nhật văn bản của nhãn (lb1) để hiển thị thời gian mới nhất.

+ lb1.after(1000, clock): Gọi lại chính nó sau 1000 ms (1 giây) để cập nhật thời gian liên tục.

**Giao diện đồng hồ**

+ lb1 = Label(...): Tạo một nhãn hiển thị thời gian trên cửa sổ: font=("digital", 50, "bold"): Đặt phông chữ là "digital", kích thước 50, kiểu chữ đậm, width=20: Đặt chiều rộng nhãn đủ lớn để chứa thời gian, bg="#f4f5f5", fg="#292e2e": Nền nhãn màu xám nhạt, chữ màu xám đậm.

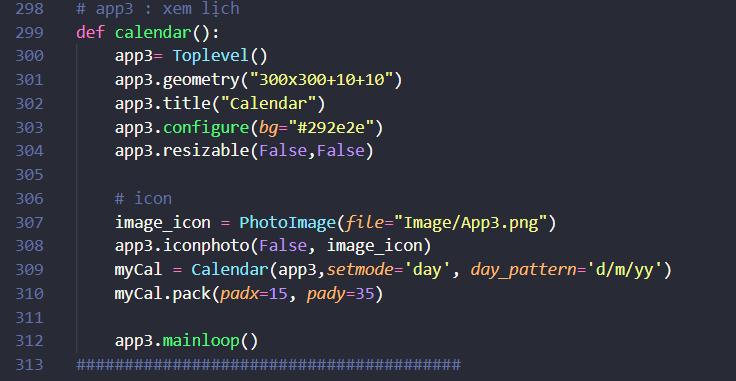
+ lb1.pack(anchor="center", pady=20): Đặt nhãn ở giữa cửa sổ, với khoảng cách trên/dưới là 20 pixel.

**Kích hoạt ứng dụng đồng hồ** clock():

Gọi hàm clock() lần đầu để khởi động việc hiển thị và cập nhật thời gian.

+ app2.mainloop(): Bắt đầu vòng lặp sự kiện của giao diện để giữ ứng dụng hoạt động.

**Hàm calendar :**



Hình 2.15 : Hàm thực hiện chức năng xem lịch

**- Yêu cầu** : Hàm này phải hiển thị đầy đủ về thứ, ngày, tháng, năm, các thao tác tìm kiếm ngày dễ sử dụng, mở một cửa sổ mới hiển thị lịch.

- **Cách hoạt động**:

Hàm calendar() tạo ra một ứng dụng con để hiển thị lịch ngày tháng, sử dụng thư viện Tkinter và widget Calendar từ thư viện tkcalendar. Dưới đây là phân tích chi tiết từng thành phần và cách hoạt động:

**Tạo giao diện chính của ứng dụng lịch**

+ app3 = Toplevel(): Tạo một cửa sổ con mới để hiển thị ứng dụng lịch.

+ app3.geometry("300x300+10+10"): Đặt kích thước cửa sổ là 300x300 pixel và vị trí hiển thị tại tọa độ (10, 10) trên màn hình.

+ app3.title("Calendar"): Đặt tiêu đề của cửa sổ là "Calendar".

+ app3.configure(bg="#292e2e"): Đặt màu nền của cửa sổ là màu xám đậm (#292e2e).

+ app3.resizable(False, False): Vô hiệu hóa khả năng thay đổi kích thước của cửa sổ.

**Icon của ứng dụng**

+ image\_icon = PhotoImage(file="Image/App3.png"): Tải biểu tượng từ tệp hình ảnh "Image/App3.png".

+ app3.iconphoto(False, image\_icon): Đặt hình ảnh này làm biểu tượng của cửa sổ lịch.

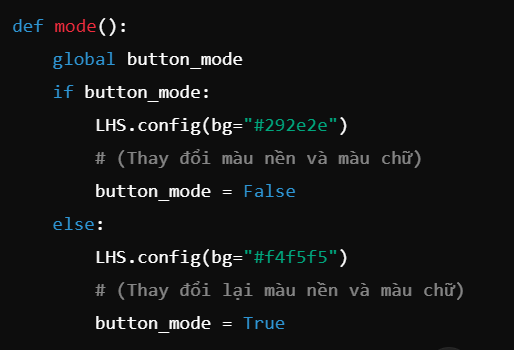
**Hiển thị lịch**

+ myCal = Calendar(...): Tạo một widget lịch bằng cách sử dụng Calendar từ thư viện tkcalendar. Các tham số quan trọng: app3: Xác định rằng widget lịch sẽ được hiển thị trong cửa sổ app3, setmode='day': Chế độ chọn ngày. Mặc định, người dùng chỉ có thể chọn một ngày cụ thể, day\_pattern='d/m/yy': Định dạng ngày hiển thị là ngày/tháng/năm (ví dụ: 26/11/24). myCal.pack(padx=15, pady=35): Đặt widget lịch vào giữa cửa sổ, với khoảng cách: padx=15: Lề trái/phải là 15 pixel. pady=35: Lề trên/dưới là 35 pixel.

**Chạy vòng lặp giao diện**

+ app3.mainloop(): Bắt đầu vòng lặp sự kiện để giữ ứng dụng hoạt động.

**Hàm mode :**



Hình 2.16 : Hàm thực hiện chức năng chuyển đổi chế độ sáng tối

**- Yêu cầu** : Hàm này thực hiện chức năng thay đổi màu sắc trên giao diện máy tính, chuyển đổi giữa chế độ sáng và tối cho giao diện người dùng.

- **Cách hoạt động**:

Hàm mode() trong đoạn mã được sử dụng để thay đổi chế độ giao diện giữa Chế độ tối (Dark Mode) và Chế độ sáng (Light Mode). Dưới đây là phân tích cách hoạt động và từng thành phần:

+ Sử dụng biến toàn cục button\_mode để lưu trạng thái hiện tại: True: Ứng dụng đang ở chế độ sáng. False: Ứng dụng đang ở chế độ tối.

+ Khi gọi hàm: Nếu button\_mode là True: Chuyển sang chế độ tối. Nếu button\_mode là False: Chuyển sang chế độ sáng.

+ Thay đổi giao diện sang chế độ tối

Cấu hình các thành phần với màu tối:

if button\_mode:

LHS.config(bg="#292e2e")

myimage.config(bg="#292e2e")

l1.config(bg="#292e2e", fg="#d6d6d6")

...

apps.config(bg="#292e2e", fg="#d6d6d6")

button\_mode = False

Các thành phần chính như LHS, RHB, l1 đến l6, và apps được cấu hình với: Nền màu xám đậm (#292e2e): Phù hợp với chế độ tối. Chữ màu xám nhạt (#d6d6d6): Tăng độ tương phản trên nền tối.

button\_mode = False: Cập nhật trạng thái của chế độ thành "tối".

+ Thay đổi giao diện sang chế độ sáng

Cấu hình các thành phần với màu sáng:

LHS.config(bg="#f4f5f5")

myimage.config(bg="#f4f5f5")

l1.config(bg="#f4f5f5", fg="#292e2e")

...

apps.config(bg="#f4f5f5", fg="#292e2e")

button\_mode = True

Các thành phần được cấu hình lại với: Nền màu xám nhạt (#f4f5f5): Phù hợp với chế độ sáng. Chữ màu xám đậm (#292e2e): Dễ nhìn hơn trên nền sáng.

button\_mode = True: Cập nhật trạng thái của chế độ thành "sáng".

**Hàm game:**



Hình 2.17 : Hàm thực hiện chức năng trò chơi (game Ludo)

**- Yêu cầu** : Giao diện dễ dàng sử dụng, hấp dẫn, mở một cửa sổ mới để chơi trò chơi xúc xắc.

- Cách hoạt động:

**Tạo cửa sổ ứng dụng**

+ app5 = Toplevel(): Tạo một cửa sổ con (không phải cửa sổ chính).

+ app5.geometry("300x500+1170+170"): Đặt kích thước cửa sổ là 300x500 pixel và hiển thị tại tọa độ (1170, 170) trên màn hình.

+ app5.title("Ludo"): Đặt tiêu đề của cửa sổ là "Ludo".

+ app5.configure(bg="#dee2e5"): Đặt màu nền cửa sổ là màu xám nhạt (#dee2e5).

+ app5.resizable(False, False): Vô hiệu hóa khả năng thay đổi kích thước của cửa sổ.

**Thêm biểu tượng và hình ảnh nền**

+ image\_icon = PhotoImage(file="Image/App5.png"): Tải biểu tượng ứng dụng từ file "Image/App5.png".

+ ludo\_image = PhotoImage(file="Image/ludo back.png"): Tải hình nền cho giao diện trò chơi từ file "Image/ludo back.png".

+ Label(app5, image=ludo\_image).pack(): Hiển thị hình nền trong cửa sổ.

**Hiển thị kết quả xúc xắc**

+ label = Label(app5, text="", font=("times", 150)): Tạo một nhãn để hiển thị kết quả xúc xắc.

+ font=("times", 150): Sử dụng font chữ "Times" với kích thước lớn (150).

**Hàm tung xúc xắc**

+ def roll(): Hàm này được gọi khi người dùng nhấn nút tung xúc xắc.

+ Tạo danh sách ký tự Unicode cho xúc xắc: dice = ['\u2680','\u2681', '\u2682','\u2683','\u2684','\u2685']. Mỗi ký tự Unicode tương ứng với một mặt của xúc xắc (1 đến 6).

+ Chọn ngẫu nhiên hai mặt xúc xắc:

label.configure(text=f"{random.choice(dice)}{random.choice(dice)}", fg="#29232e")

+ random.choice(dice): Lấy ngẫu nhiên một mặt xúc xắc từ danh sách dice.

+ Kết quả của hai xúc xắc được nối lại và hiển thị trên nhãn. fg="#29232e": Đặt màu chữ đậm.

**Nút bấm tung xúc xắc**

+ btn\_image = PhotoImage(file="Image/ludo button.png"): Tải hình ảnh cho nút bấm từ file "Image/ludo button.png".

+ btn = Button(app5, image=btn\_image, bg="#dee2e5", command=roll): Tạo một nút bấm với:

+ image=btn\_image: Hiển thị hình ảnh nút.

+ command=roll: Gọi hàm roll() khi nút được nhấn.

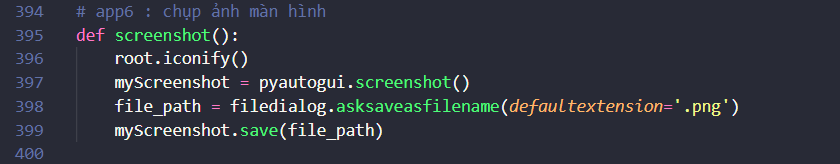
+ btn.pack(padx=10, pady=10):

+ Hiển thị nút trong giao diện với khoảng cách lề: padx=10: Khoảng cách lề trái/phải là 10 pixel, pady=10: Khoảng cách lề trên/dưới là 10 pixel.

**Vòng lặp chính**

+ app5.mainloop(): Bắt đầu vòng lặp sự kiện để duy trì hoạt động của giao diện.

**Hàm screenshot, file, chrome, close\_win :**



Hình 2.18 : Hàm thực hiện chức năng chụp ảnh màn hình

**- Yêu cầu** : Thao tác chụp ảnh màn hình nhanh gọn, sau đó lưu file ảnh vào máy của người dùng, đi đến thư mục trong máy tính, mở trình duyệt tìm kiếm google, tắt ứng dụng.

- Cách hoạt động:

**Hàm screenshot()**

+ root.iconify(): Thu nhỏ cửa sổ chính để không xuất hiện trong ảnh chụp màn hình.

+ myScreenshot = pyautogui.screenshot(): Sử dụng thư viện pyautogui để chụp ảnh màn hình toàn bộ màn hình.

+ filedialog.asksaveasfilename(defaultextension='.png'): Hiển thị hộp thoại để người dùng chọn đường dẫn và tên tệp để lưu ảnh chụp. Mặc định định dạng tệp là .png.

+ myScreenshot.save(file\_path): Lưu ảnh chụp màn hình vào vị trí người dùng đã chọn.

**Hàm file()**

+ subprocess.Popen(): Sử dụng thư viện subprocess để chạy lệnh mở File Explorer của Windows. r'explorer /select,"C:\path\of\folder\file"', explorer /select: Lệnh mở File Explorer và chọn tệp hoặc thư mục cụ thể. "C:\path\of\folder\file": Đường dẫn đến tệp hoặc thư mục cần mở. Đường dẫn này cần được thay thế bằng giá trị cụ thể trong mã thực tế.

**Hàm chrome()**

+ wb.register('chrome', None): Đăng ký trình duyệt Google Chrome với thư viện webbrowser.

+ wb.open('https://www.google.com/'): Mở URL https://www.google.com/ bằng trình duyệt đã đăng ký.

**Hàm close\_win()**

+ root.destroy(): Hủy toàn bộ cửa sổ và dừng vòng lặp sự kiện của ứng dụng.

## 2.4. Tiến hành chạy, kiểm thử và chỉnh sửa.

Sau khi tiến hành chạy thử, chỉnh sửa lại một số giao diện cho phù hợp, đẹp mắt, vị trí của các icon, hình ảnh cho hài hòa, chọn lại màu sắc, chỉnh sửa lại một số tiện ích.

# CHƯƠNG 3: KẾT QUẢ ĐẠT ĐƯỢC VÀ KẾT LUẬN

## 3.1. Code minh họa.

### 3.1.1. File code minh họa :



### 3.1.2. file ảnh dung trong chương trình



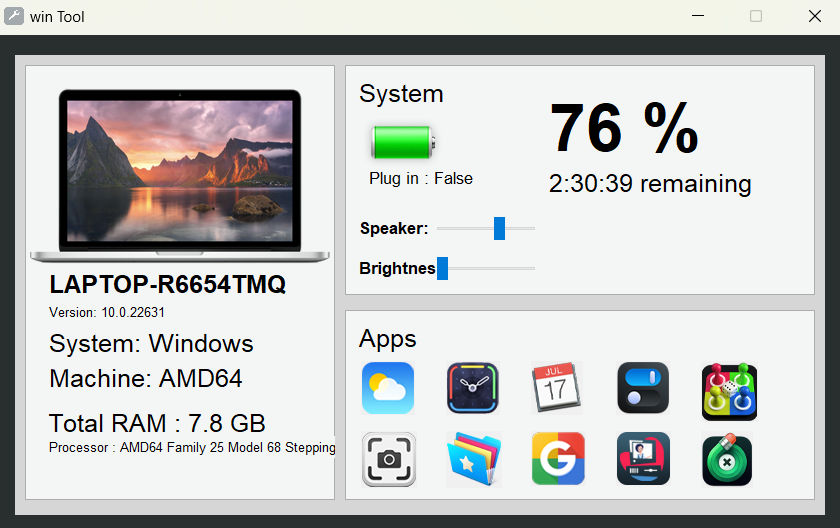
## 3.2. Demo chương trình

Truy cập link để có thể tải video về :

https://drive.google.com/file/d/1O\_fQsCjtL7Ud2BWHzL963tEJhV6TCXsc/view?usp=sharing

## 3.3. Kết quả được đạt:

### 3.3.1. Giao diện Chính (All\_in\_one Utility):



Hình 3.1 : Giao diện chính sau khi hoàn thành

Hiển thị thông tin chi tiết của hệ thống, bao gồm tên máy tính, phiên bản hệ điều hành, loại hệ thống, máy, RAM và bộ xử lý. Điều này giúp người dùng biết rõ cấu hình phần cứng và hệ điều hành của máy.

Theo dõi tình trạng pin với mức sạc (theo phần trăm), tình trạng kết nối sạc và thời gian sử dụng còn lại, rất hữu ích để quản lý thời lượng pin.

Tích hợp thanh trượt để điều chỉnh âm lượng và độ sáng màn hình, giúp người dùng dễ dàng thao tác mà không cần truy cập vào cài đặt của hệ điều hành.

Cung cấp các biểu tượng ứng dụng để mở nhanh các tính năng như thời tiết, đồng hồ, lịch, game, chụp màn hình, trình duyệt, và Facebook.

### 3.3.2. Ứng dụng Thời Tiết:

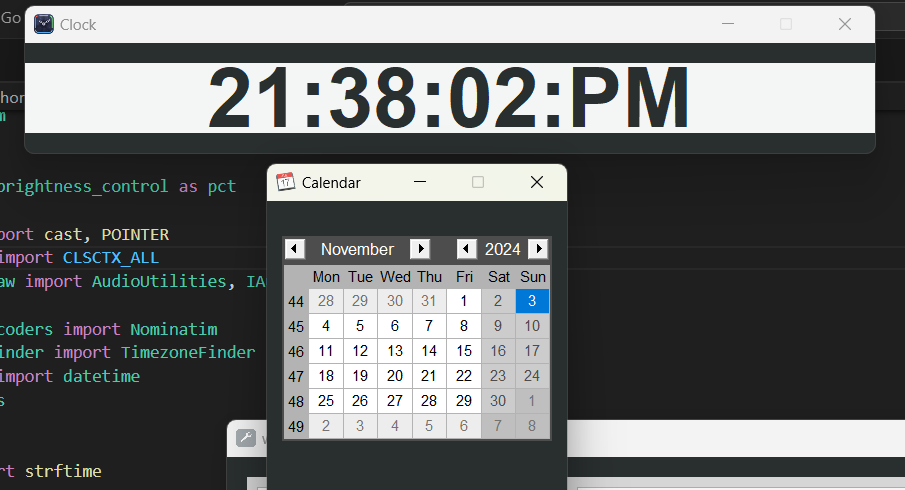


Hình 3.2 : Ứng dụng dự báo thời tiết khi hoàn thành

Cho phép người dùng tra cứu thời tiết hiện tại của một thành phố bằng cách nhập tên thành phố.

Hiển thị thời gian thực, nhiệt độ hiện tại, tình trạng thời tiết, độ ẩm, áp suất, và tốc độ gió. Tất cả thông tin này được trình bày rõ ràng và dễ hiểu, giúp người dùng nắm bắt nhanh tình hình thời tiết tại khu vực mong muốn.

### 3.3.3. Ứng dụng Đồng Hồ và lịch :

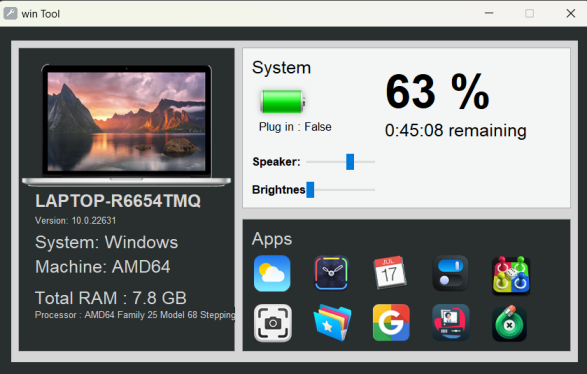
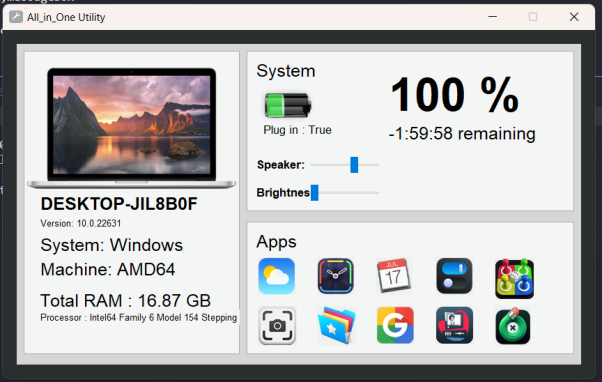


Hình 3.4 : Ứng dụng xem lịch và đồng hồ

Cung cấp một đồng hồ kỹ thuật số lớn hiển thị thời gian thực với định dạng giờ, phút, giây và AM/PM. Đồng hồ này có thiết kế dễ nhìn và có thể được sử dụng làm đồng hồ chính trên màn hình.

Hiển thị lịch tháng, cho phép người dùng dễ dàng xem ngày tháng hiện tại và chọn ngày cụ thể. Điều này rất hữu ích để quản lý thời gian và các công việc liên quan đến lịch trình.

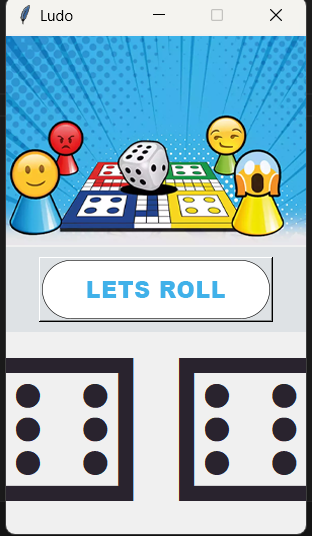
### 3.3.4.Ứng dụng đổi màu giao diện:

Hình 3.5 : Giao diện khi chuyển đổi chế độ sáng tối

Tính năng này giúp người dùng chuyển đổi giữa chế độ màu tối và sáng, phù hợp với sở thích cá nhân hoặc môi trường sử dụng.

### 3.3.5. Ứng dụng game ludo:



Hình 3.5 : Trò chơi Ludo

Cung cấp một trò chơi xúc xắc đơn giản, với giao diện vui nhộn và hình ảnh xúc xắc động mỗi khi người dùng nhấn nút "LET'S ROLL". Đây là một tính năng giải trí nhẹ nhàng cho người dùng.

### 3.3.6. Ứng dụng Chụp Màn Hình (Screenshot):

Cho phép người dùng chụp ảnh màn hình nhanh chóng. Khi nhấn vào biểu tượng chụp màn hình, ứng dụng tự động thu nhỏ cửa sổ chính và chụp toàn bộ màn hình.

Sau khi chụp, người dùng có thể chọn vị trí lưu tệp ảnh với định dạng .png. Tính năng này rất hữu ích cho những ai cần ghi lại thông tin nhanh chóng hoặc chia sẻ hình ảnh màn hình cho người khác.

### 3.3.7. Trình Quản Lý File (File):

Khi nhấn vào biểu tượng này, ứng dụng sẽ mở thư mục mà bạn đã chỉ định. Tính năng này tạo điều kiện thuận lợi cho người dùng truy cập nhanh vào các tài liệu hoặc thư mục thường dùng mà không cần phải tìm kiếm thủ công.

Có thể tuỳ chỉnh đường dẫn thư mục theo nhu cầu của người dùng, giúp tăng tính linh hoạt.

### 3.3.8. Trình Duyệt Chrome (Chrome):

Cung cấp một nút để mở nhanh trình duyệt Google Chrome và chuyển hướng đến trang chủ Google (hoặc trang bạn muốn).

Đây là một tính năng tiết kiệm thời gian, giúp người dùng truy cập Internet nhanh chóng mà không cần tìm biểu tượng Chrome trên desktop hoặc thanh tác vụ.

### 3.3.9. App đọc báo nhanh báo mới.

Khi nhấn vào biểu tượng này, ứng dụng sẽ mở trình duyệt và chuyển hướng tới trang báo mới.

Điều này giúp người dùng dễ dàng truy cập vào và cập nhật tin tức mới nhất và hot nhất trên thế giới.

### 3.3.10. Thoát Ứng Dụng (Close App):

Nút cuối cùng trong giao diện ứng dụng là biểu tượng thoát, cho phép người dùng đóng toàn bộ ứng dụng ngay lập tức.

Tính năng này giúp người dùng dễ dàng thoát khỏi ứng dụng khi không cần thiết, đóng tất cả các cửa sổ con để giải phóng bộ nhớ hệ thống.

## 3.4. Hạn chế.

### **3.4.1. Phụ thuộc vào kết nối mạng cho Dự báo Thời Tiết**

Tính năng dự báo thời tiết yêu cầu kết nối Internet để lấy dữ liệu từ API của OpenWeatherMap. Nếu không có kết nối mạng hoặc nếu API gặp sự cố, chương trình sẽ gặp lỗi và không thể hiển thị thông tin thời tiết.

### 3.4.2. Vấn đề về hiệu năng với hình ảnh lớn và tải tài nguyên.

Chương trình sử dụng nhiều hình ảnh cho các biểu tượng ứng dụng, nút và giao diện. Nếu kích thước hình ảnh quá lớn, có thể làm giảm hiệu năng và tốc độ tải giao diện.

### **3.4.3. Tính Năng Đổi Chế Độ (Sáng - Tối) Còn Đơn Giản**

Tính năng đổi chế độ sáng - tối chỉ áp dụng màu nền và màu chữ. Đối với một số yếu tố giao diện phức tạp hơn, tính năng này có thể không đáp ứng tốt.

### **3.4.4. Phụ thuộc vào các thư viện bên ngoài.**

Chương trình sử dụng nhiều thư viện bên ngoài như screen\_brightness\_control, pycaw, geopy, timezonefinder, và tkcalendar. Nếu người dùng không cài đặt đúng hoặc có sự không tương thích giữa các phiên bản thư viện, chương trình sẽ không hoạt động đúng.

### **3.4.5. Không lưu trữ trạng thái và tùy chỉnh của người dùng.**

Ứng dụng không lưu trữ tùy chỉnh của người dùng, chẳng hạn như mức âm lượng và độ sáng. Khi khởi động lại, tất cả thiết lập sẽ trở về mặc định.

## 3.5. Hướng phát triển.

Thêm phần kiểm tra kết nối mạng trước khi gọi API. Nếu không có kết nối, hiển thị thông báo lỗi thân thiện cho người dùng.

Cải tiến chế độ sáng - tối bằng cách tùy chỉnh nhiều thành phần hơn, bao gồm cả hình ảnh, màu biểu tượng và các phần tử khác như thanh trượt. Cũng có thể cung cấp nhiều hơn hai chế độ màu (ví dụ: thêm chế độ màu xanh, đỏ,...).

Thêm phần trợ giúp hoặc hướng dẫn sử dụng cho từng tính năng, hoặc cung cấp các gợi ý khi người dùng di chuột qua các biểu tượng.

# 

# PHẦN KẾT LUẬN

Chương trình python trên trên là minh chứng rõ ràng cho tính ứng dụng đa dạng và mạnh mẽ của ngôn ngữ lập trình Python trong việc phát triển các công cụ và ứng dụng hỗ trợ hệ thống. Với sự kết hợp của nhiều thư viện, chương trình cung cấp một bộ công cụ tiện ích, từ theo dõi thông tin hệ thống, điều chỉnh âm lượng, độ sáng, cho đến dự báo thời tiết, lịch, đồng hồ và giải trí. Python cho phép chúng ta dễ dàng tích hợp nhiều tính năng và giao diện trực quan vào một ứng dụng duy nhất, phục vụ nhu cầu hàng ngày của người dùng máy tính một cách linh hoạt và hiệu quả.   
 Chương trình này không chỉ là công cụ tiện ích mà còn thể hiện khả năng của Python trong việc phát triển ứng dụng giao diện người dùng (GUI) với Tkinter. Điều này cho thấy Python không chỉ là ngôn ngữ mạnh mẽ trong phân tích dữ liệu và phát triển web mà còn có thể được sử dụng để xây dựng các ứng dụng desktop hữu ích. Tổng kết lại, chương trình này là một minh họa sống động cho sự linh hoạt và tính ứng dụng to lớn của Python trong việc tạo ra các công cụ tiện ích, phục vụ đa dạng nhu cầu của người dùng trong môi trường số hiện đại.

**TÀI LIỆU THAM KHẢO**

1. Python với tkinter, thư viện tạo giao diện người dùng, truy cập ngày 1/11/2024. Tại link : <https://www.geeksforgeeks.org/python-gui-tkinter/>
2. API là gì, những đặc điểm của api. Truy cập ngày 1/11/2024. Tại link :

<https://topdev.vn/blog/api-la-gi/>

1. Những thư viện lấy dữ liệu đọc dữ liệu từ hệ thống. Truy cập ngày 30/**10/2024. Tại link : <https://www.w3schools.com/python/module_os.asp>**
2. Những quy tắc cơ bản để thiết kế một ứng dụng. Truy cập ngày 29/10/2024. Tại link : <https://winds.vn/thiet-ke-giao-dien-app/>