**PHẦN 1: MỞ ĐẦU**

**1. Lý do chọn đề tài**:  
 Tuyển sinh là một công việc quan trọng và có ảnh hưởng trực tiếp đến sự phát triển của các trường đại học cũng như sự nghiệp học tập của sinh viên. Hồ sơ tuyển sinh của sinh viên chứa đựng các thông tin cơ bản và quan trọng, giúp các cơ sở giáo dục đánh giá và chọn lọc ứng viên phù hợp. Tuy nhiên, quá trình quản lý và xử lý hồ sơ tuyển sinh hiện nay ở nhiều trường vẫn chủ yếu thực hiện theo cách thủ công, thông qua việc nhập liệu và lưu trữ hồ sơ bằng giấy hoặc các bảng tính đơn giản.

Việc không sử dụng hệ thống tự động để quản lý hồ sơ tuyển sinh có thể gây ra một số vấn đề lớn. Đầu tiên, việc nhập liệu thủ công tốn rất nhiều thời gian và công sức. Nếu mỗi năm trường phải xử lý hàng nghìn hồ sơ, việc nhập thông tin thủ công vào hệ thống sẽ không chỉ làm giảm hiệu quả công việc mà còn dễ gây ra sai sót, thiếu chính xác trong dữ liệu. Điều này không chỉ ảnh hưởng đến chất lượng tuyển sinh mà còn làm chậm quá trình xét tuyển và ra quyết định, từ đó ảnh hưởng đến trải nghiệm của cả ứng viên và các cán bộ tuyển sinh.

Thêm vào đó, việc quản lý hồ sơ bằng phương pháp thủ công còn gặp phải những khó khăn trong việc lưu trữ và truy xuất thông tin khi cần thiết. Hồ sơ giấy hoặc dữ liệu trong các bảng tính đơn giản sẽ rất khó để tra cứu nhanh chóng, đặc biệt khi cần phải phân tích, so sánh các dữ liệu hoặc đưa ra các báo cáo. Trong suốt quá trình thực hiện đề tài nhóm đã đặt ra những thử thách và khó khăn trên do đó lý do chọn đề tài để giúp các trường đại học có thể tối ưu hóa công việc tuyển sinh một cách nhanh chóng và chính xác cũng như là cơ hội để ứng dụng các kiến thức đã được học từ môn Lập trình Python. Việc thực hiện đề tài này không chỉ giúp nhóm nâng cao năng lực chuyên môn mà được trải nghiệm cảm giác ngồi vào vị của những bộ óc thiên tài, người tạo ra hệ thống quản lí hồ sơ sinh viên.

**2. Mục tiêu đề tài**:

Mục tiêu của đề tài này là nghiên cứu và phân tích dữ liệu hồ sơ tuyển sinh của sinh viên, nhằm tìm hiểu về các yếu tố ảnh hưởng đến quyết định tuyển chọn và ứng dụng công nghệ trong việc tối ưu hóa quá trình xử lý và quản lý hồ sơ. Cụ thể, các mục tiêu chính của đề tài bao gồm:

**2.1. Phân tích các yếu tố trong hồ sơ tuyển sinh**:

* Nghiên cứu và phân tích các yếu tố quan trọng trong hồ sơ tuyển sinh, bao gồm:
  + - **Tên, tuổi, giới tính**: Các thông tin cơ bản giúp phân loại và xác định đặc điểm của sinh viên
    - **Điểm đạt được trong bài kiểm tra đầu vào**: Phân tích mức độ ảnh hưởng của điểm số bài kiểm tra đầu vào đến khả năng trúng tuyển của sinh viên.
    - **Tỷ lệ điểm cuối kỳ trung học của học sinh**: Xem xét mối liên hệ giữa kết quả học tập ở cấp trung học và khả năng trúng tuyển vào trường.
    - **Học sinh được chấp nhận hay bị từ chối**: Phân tích kết quả tuyển sinh để tìm hiểu các yếu tố quyết định đến việc chấp nhận hoặc từ chối hồ sơ.

**2.2. Đánh giá quy trình tuyển sinh**:

* + Đánh giá hiệu quả và tính minh bạch của quy trình tuyển sinh hiện tại, từ việc thu thập và nhập dữ liệu đến việc đưa ra quyết định tuyển chọn. Điều này sẽ giúp nhận diện các vấn đề trong quy trình hiện tại, chẳng hạn như khả năng xảy ra sai sót khi xử lý dữ liệu hoặc tốn thời gian trong quá trình tuyển sinh.

**2.3. Đề xuất giải pháp tự động hóa và cải tiến quy trình**:

* + Đề xuất các giải pháp công nghệ để tự động hóa quá trình nhập liệu, phân tích và quản lý hồ sơ tuyển sinh. Điều này nhằm giúp giảm thiểu sai sót, tiết kiệm thời gian và nâng cao hiệu quả của công tác tuyển sinh.
  + Phát triển một hệ thống quản lý hồ sơ tuyển sinh có khả năng lưu trữ, truy xuất và phân tích dữ liệu một cách nhanh chóng và chính xác.

**2.4.Tạo báo cáo tổng hợp và trực quan hóa dữ liệu**:

* + Phân tích dữ liệu để tạo ra các báo cáo tổng hợp, trực quan hóa các thông tin quan trọng từ hồ sơ tuyển sinh như sự phân bố điểm số, tỷ lệ trúng tuyển theo từng tiêu chí, và sự phân bố các yếu tố như giới tính, độ tuổi.

**2.5. Đề xuất các cải tiến trong quyết định tuyển chọn**:

* + Dựa trên phân tích dữ liệu, đề xuất các yếu tố cần được chú trọng trong quá trình tuyển sinh để đưa ra quyết định tuyển chọn công bằng và hiệu quả hơn.

Với mục tiêu này, bạn không chỉ giúp người đọc hiểu rõ hơn về các yếu tố tác động đến quyết định tuyển sinh mà còn chỉ ra cách tối ưu hóa và cải thiện quy trình tuyển sinh thông qua việc ứng dụng công nghệ và phân tích dữ liệu.

**3. Đối tượng và phạm vi nghiên cứu**

● Tập dữ liệu thông tin được lấy từ Kaggle, nó chứa **157** hàng hồ sơ nhập học của sinh viên, bao gồm các hàng trùng lặp, giá trị bị thiếu và một số dữ liệu không nhất quán (ví dụ: giá trị ngoại lệ, giá trị không thực tế). Nó lý tưởng để thực hành các bước chuẩn bị dữ liệu phổ biến trước khi áp dụng các thuật toán học máy.

● 157 hồ sơ nhập học của sinh viên tại các thành phố cư trú tại Pakistan, bộ dữ liệu mô phỏng hệ thống hồ sơ tuyển sinh đại học, trong đó hồ sơ tuyển sinh của mỗi sinh viên bao gồm điểm thi, tỷ lệ phần trăm trung học và tình trạng tuyển sinh. Dữ liệu chứa các lỗi thực tế thường gặp trong dữ liệu thô, cung cấp kinh nghiệm thực tế trong việc xử lý dữ liệu.

**PHẦN 2: NỘI DUNG**

**CHƯƠNG 1: TÓM TẮT CƠ SỞ LÝ THUYẾT**

**1.1. Giới thiệu về Lập trình Python**

Python là ngôn ngữ lập trình bậc cao với nhiều tính năng, được thiết kế để dễ

học và dễ sử dụng. Với cú pháp các câu lệnh đơn giản, trực quan Python đã trở

thành ngôn ngữ, công cụ phổ biến trong nhiều lĩnh vực từ phát triển phần mềm,

phân tích dữ liệu, đến trí tuệ nhân tạo.

Những nhận định về định nghĩa khái quát của Python:

● *“Python is an interpreted, object-oriented, high-level programming*

*language with dynamic semantics.”* (Python là một ngôn ngữ lập trình

thông dịch, hướng đối tượng, bậc cao, với cú pháp động.)

—Python Software Foundation

● *“The Zen of Python: Simple is better than complex. Readability counts.*

*Beautiful is better than ugly.”* (Quan điểm của Python: Đơn giản tốt

hơn phức tạp. Đọc hiểu quan trọng. Đẹp hơn xấu.)

—Tim Peters, The Zen of Python

Ngôn ngữ này không chỉ được sử dụng rộng rãi trong việc phân tích dữ liệu thông qua các thư viện hỗ trợ như Pandas, Numpy và Matplotlib mà còn có khả năng cung cấp nền tảng vững chắc cho các lập trình viên, trang bị các kỹ năng thực tế để giải quyết các vấn đề và phát triển các ứng dụng.

**1.2. Giới thiệu về các thư viện được sử dụng để phân tích dữ liệu**

Python là ngôn ngữ nổi tiếng trong phân tích dữ liệu nhờ sở hữu các thư viện hỗ trợ, đơn giản hóa việc xử lý và trực quan dữ liệu. Một số thư viện có thể kể đến

như:

**1.2.1. NumPy**

NumPy (Numerical Python) là thư viện mạnh mẽ để làm việc với các mảng đa chiều. Nó cung cấp các hàm toán học hiệu suất cao, giúp thực hiện các phép tính khoa học như đại số tuyến tính, tính toán ma trận và xử lý dữ liệu lớn nhanh chóng. NumPy là nền tảng của nhiều thư viện khác trong hệ sinh thái Python.

**1.2.1. Pandas**

Pandas nổi bật với khả năng xử lý dữ liệu dạng bảng thông qua cấu trúc DataFrame và Series. Thư viện này hỗ trợ đọc, ghi dữ liệu từ nhiều định dạng như CSV, Excel, hoặc SQL. Các thao tác như làm sạch dữ liệu, tổng hợp và chuyển đổi dữ liệu được thực hiện dễ dàng với Pandas, giúp tăng hiệu suất trong phân tích dữ liệu.

**1.2.2. Matplotlib**

Matplotlib là thư viện tiêu chuẩn để trực quan hóa dữ liệu. Người dùng có thể tạo nhiều loại biểu đồ như biểu đồ đường, cột, tròn hoặc tán xạ. Với khả năng tùy chỉnh cao, Matplotlib giúp biểu diễn dữ liệu một cách trực quan, hỗ trợ việc khám phá và trình bày thông tin hiệu quả.

**1.2.3. Tkinter**

Tkinter là thư viện tích hợp sẵn trong Python, cho phép xây dựng giao diện đồ họa (GUI). Với Tkinter, người dùng có thể tạo ứng dụng trực quan, giúp hiển thị và tương tác với dữ liệu dễ dàng. Dù đơn giản hơn các công cụ như PyQt, Tkinter vẫn đủ mạnh mẽ để phát triển các ứng dụng phân tích dữ liệu cơ bản. Các thư viện trên không chỉ hỗ trợ việc xử lý và trực quan hóa dữ liệu, mà còn giúp người lập trình Python thực hiện các phân tích phức tạp với độ chính xác và hiệu quả cao. Chúng tạo nên một hệ sinh thái linh hoạt, đáp ứng tốt nhu cầu của người làm phân tích dữ liệu.

**CHƯƠNG 2: MÔ TẢ DỮ LIỆU HỒ SƠ SINH VIÊN**

**2.1. Kiểu dữ liệu**

**Bảng 1:** Thống kê các kiểu dữ liệu của các đặc trưng:

|  |  |  |
| --- | --- | --- |
| **Đặc trưng** | **Giải thích** | **Kiểu dữ liệu** |
| **Name** | Tên của học sinh(Tên tiếng Pakistan) | string |
| **Sex** | Giới tính(Nam/Nữ) | string |
| **Age** | Tuổi của học sinh(Tuổi của học sinh (một số trường hợp ngoại lệ và giá trị bị thiếu). | float |
| **Admission Test Score** | Điểm đạt được trong bài kiểm tra đầu vào (bao gồm các trường hợp ngoại lệ và giá trị bị thiếu). | float |
| **High School Percent** | Tỷ lệ điểm cuối kỳ trung học của học sinh (bao gồm các trường hợp ngoại lệ và giá trị bị thiếu) | float |
| **City** | Thành phố cư trú tại Pakistan | string |
| **Admission Status** | Học sinh được chấp nhận hay bị từ chối | True/False |

**2.2. Lĩnh vực ứng dụng-Hạn chế của dữ liệu**

**Lĩnh vực ứng dụng của dữ liệu trong hồ sơ quản lý sinh viên**

Hồ sơ quản lý sinh viên là một phần quan trọng trong hệ thống quản lý giáo dục, giúp các cơ sở giáo dục lưu trữ và theo dõi thông tin của sinh viên từ khi nhập học cho đến khi tốt nghiệp. Dữ liệu trong hồ sơ quản lý sinh viên có thể được ứng dụng trong nhiều lĩnh vực khác nhau, bao gồm:

1. **Quản lý thông tin cá nhân và học tập:** Dữ liệu trong hồ sơ quản lý sinh viên bao gồm thông tin cá nhân như họ tên, ngày sinh, địa chỉ, thông tin liên hệ, cũng như kết quả học tập, điểm số, quá trình học tập. Điều này giúp nhà trường theo dõi sự tiến bộ của sinh viên và hỗ trợ trong việc cung cấp các dịch vụ học thuật phù hợp.
2. **Hỗ trợ quyết định quản lý:** Các trường học sử dụng dữ liệu sinh viên để đưa ra các quyết định về việc tuyển sinh, phân bổ lớp học, phân công giảng viên, cũng như quản lý tài chính (học phí, học bổng, các khoản hỗ trợ). Dữ liệu còn có thể giúp đưa ra các dự báo về số lượng sinh viên sẽ tốt nghiệp trong năm, từ đó giúp lập kế hoạch giảng dạy và tài chính hiệu quả hơn.
3. **Cung cấp các dịch vụ sinh viên:** Hồ sơ quản lý sinh viên có thể cung cấp thông tin để cung cấp các dịch vụ sinh viên như hỗ trợ tài chính, tư vấn học tập, hỗ trợ nghề nghiệp, các câu lạc bộ và hoạt động ngoại khóa. Thông qua phân tích dữ liệu, các trường có thể cung cấp các dịch vụ cá nhân hóa để hỗ trợ sinh viên đạt được thành công trong học tập.
4. **Phân tích và nghiên cứu giáo dục:** Dữ liệu sinh viên cũng có thể được sử dụng cho các nghiên cứu giáo dục, nhằm phân tích các xu hướng học tập, đánh giá chất lượng giảng dạy, và cải tiến chương trình đào tạo. Các nghiên cứu này có thể giúp cải thiện phương pháp giảng dạy và tạo ra các chương trình học phù hợp với nhu cầu thay đổi của sinh viên.
5. **Xây dựng hệ thống đánh giá và kiểm tra:** Dữ liệu trong hồ sơ quản lý sinh viên giúp xây dựng các hệ thống đánh giá, kiểm tra và xếp hạng sinh viên. Đây là yếu tố quan trọng trong việc đo lường hiệu quả của chương trình giảng dạy, cũng như làm cơ sở để cấp chứng chỉ, bằng cấp cho sinh viên.

**Hạn chế của dữ liệu trong hồ sơ quản lý sinh viên**

Mặc dù dữ liệu trong hồ sơ quản lý sinh viên có thể mang lại nhiều lợi ích, nhưng việc sử dụng và lưu trữ dữ liệu này cũng gặp phải một số hạn chế và thách thức:

1. **Vấn đề bảo mật và quyền riêng tư:** Hồ sơ sinh viên chứa đựng nhiều thông tin cá nhân nhạy cảm như họ tên, ngày sinh, địa chỉ, điểm số, và tình trạng tài chính. Nếu không được bảo mật đúng cách, dữ liệu này có thể bị xâm nhập hoặc lộ ra ngoài, gây rủi ro cho sinh viên và tổ chức giáo dục. Việc tuân thủ các quy định bảo vệ dữ liệu như GDPR hoặc các luật bảo mật dữ liệu khác là một yếu tố quan trọng cần phải đảm bảo.
2. **Chất lượng và độ chính xác của dữ liệu:** Một trong những vấn đề lớn trong việc quản lý hồ sơ sinh viên là độ chính xác và tính toàn vẹn của dữ liệu. Nếu dữ liệu không được cập nhật kịp thời hoặc có sai sót (ví dụ như nhập sai điểm số, thiếu thông tin liên lạc), sẽ ảnh hưởng đến quá trình ra quyết định của trường học và thậm chí có thể ảnh hưởng đến quyền lợi của sinh viên.
3. **Khả năng tích hợp và chia sẻ dữ liệu:** Các hệ thống quản lý hồ sơ sinh viên đôi khi không thể tích hợp dễ dàng với các hệ thống khác trong trường học hoặc trong hệ thống giáo dục quốc gia. Điều này có thể dẫn đến việc phân mảnh dữ liệu, khó khăn trong việc chia sẻ thông tin giữa các bộ phận, hoặc giữa các trường học, ảnh hưởng đến hiệu quả công tác quản lý và cải tiến dịch vụ.
4. **Phân biệt đối xử và thiên vị:** Việc phân tích dữ liệu để đánh giá năng lực sinh viên cũng có thể dẫn đến các vấn đề về phân biệt đối xử hoặc thiên vị, đặc biệt khi có sự can thiệp của các yếu tố chủ quan hoặc dữ liệu không đầy đủ. Các thuật toán và hệ thống phân tích có thể không phản ánh chính xác thực tế, dẫn đến quyết định sai lầm trong việc hỗ trợ học tập và phân bổ các cơ hội học bổng, tài trợ.
5. **Chi phí và nguồn lực để duy trì hệ thống:** Việc duy trì hệ thống hồ sơ sinh viên đòi hỏi các trường học phải đầu tư vào công nghệ, nhân lực và bảo trì hệ thống. Điều này có thể tốn kém, đặc biệt đối với các trường học có ngân sách hạn hẹp. Hơn nữa, các công nghệ mới và quy trình bảo mật thay đổi liên tục, khiến cho các trường phải liên tục cập nhật và duy trì hệ thống.
6. **Sự phụ thuộc vào công nghệ:** Trong một số trường hợp, nếu hệ thống quản lý hồ sơ sinh viên gặp sự cố hoặc bị tấn công, việc truy cập vào dữ liệu có thể bị gián đoạn. Điều này có thể gây ảnh hưởng lớn đến hoạt động giảng dạy, học tập, và các dịch vụ hỗ trợ sinh viên.
7. **Vấn đề về công bằng trong phân tích dữ liệu:** Các thuật toán sử dụng để phân tích dữ liệu sinh viên đôi khi có thể không phản ánh đầy đủ các yếu tố văn hóa, xã hội và cá nhân của từng sinh viên. Việc đánh giá sinh viên chỉ dựa vào dữ liệu có thể bỏ qua các yếu tố quan trọng khác như hoàn cảnh gia đình, sức khỏe, hoặc những khó khăn ngoài trường học, điều này có thể dẫn đến những quyết định không công bằng.

**2.3. Kết luận**

Dữ liệu trong hồ sơ quản lý sinh viên là một công cụ mạnh mẽ để cải thiện công tác quản lý giáo dục, giúp tối ưu hóa quy trình học tập và cung cấp dịch vụ hỗ trợ sinh viên. Tuy nhiên, việc sử dụng dữ liệu này cũng cần phải được thực hiện một cách cẩn trọng và có trách nhiệm. Cần có các biện pháp bảo vệ dữ liệu chặt chẽ, đồng thời duy trì chất lượng, độ chính xác và công bằng trong việc sử dụng dữ liệu để đảm bảo quyền lợi của sinh viên và hiệu quả của hệ thống giáo dục.

**Chương 3: PHÂN TÍCH DỮ LIỆU SINH VIÊN Ở THÀNH PHỐ CƯ TRÚ TẠI PAKISTAN**

**3.1. Khởi tạo dữ liệu:**

**3.1.1. Hàm show\_list\_page(page):**

**Mục đích:** Hàm này có mục đích hiển thị danh sách dữ liệu trong DataFrame dưới dạng bảng trong giao diện người dùng. Nó hỗ trợ phân trang để hiển thị dữ liệu một cách hợp lý và dễ dàng hơn.

**Cách xây dựng:**

* Hàm sử dụng pandas.DataFrame.iloc để trích xuất dữ liệu của từng trang dựa trên số dòng a (số dòng mỗi trang) và trang hiện tại b.
* ttk.Treeview được sử dụng để hiển thị dữ liệu dưới dạng bảng.
* Dữ liệu được chèn vào bảng bằng cách sử dụng phương thức tree.insert() trong vòng lặp.
* Phân trang được cài đặt với các nút "Trang trước" và "Trang sau". Mỗi khi nhấn nút, trang sẽ thay đổi và hàm được gọi lại.

**Cách hoạt động:**

* Dữ liệu từ DataFrame sẽ được chia thành các trang với a dòng mỗi trang.
* Người dùng có thể duyệt qua các trang bằng cách nhấn vào các nút "Trang trước" và "Trang sau".
* Các cột trong bảng là: "STT", "Name", "Age", "Gender", "Admission Test Score", "High School Percentage", "City", "Admission Status".

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

*Hình 1. Code để tạo danh sách có phân trang*

**3.1.2. Hàm anvao():**

**Mục đích:** Hàm này được gọi khi người dùng muốn quản lý hồ sơ tuyển sinh, với khả năng lọc dữ liệu dựa trên các tiêu chí như "City" và "Gender".

**Cách xây dựng:**

* Hàm tạo hai ttk.Combobox để cho phép người dùng chọn thành phố và giới tính.
* Một nút "Tìm kiếm" được tạo ra để tìm kiếm các hồ sơ phù hợp dựa trên lựa chọn của người dùng.
* Bảng ttk.Treeview được sử dụng để hiển thị dữ liệu sau khi tìm kiếm.

**Cách hoạt động:**

* Sau khi người dùng chọn giá trị từ các Combobox và nhấn "Tìm kiếm", dữ liệu từ DataFrame sẽ được lọc theo thành phố và giới tính.
* Các kết quả tìm kiếm sẽ được hiển thị trong bảng Treeview. Nếu không có dữ liệu phù hợp, thông báo "Không có dữ liệu thỏa mãn yêu cầu" sẽ được hiển thị trong bảng.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

*Hình 2. Code tạo hàm khi người dùng ấn vào QLHS*

**3.1.3. Hàm datacombobox():**

**Mục đích:** Hàm này có mục đích lọc và hiển thị danh sách các thí sinh trong DataFrame (df) dựa trên lựa chọn của người dùng từ hai Combobox: "City" và "Gender". Sau khi lọc dữ liệu, hàm sẽ cập nhật bảng (Treeview) với các thí sinh thỏa mãn các điều kiện lọc.

**Cách hoạt động:**

* Hàm tạo ra một cơ chế lọc dữ liệu từ DataFrame dựa trên các giá trị mà người dùng chọn trong các Combobox.
* Sau khi lọc, hàm sẽ cập nhật lại bảng (Treeview) hiển thị danh sách thí sinh.
* Nếu không có thí sinh nào thỏa mãn điều kiện lọc, sẽ hiển thị thông báo không có dữ liệu phù hợp.

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Mô tả được tạo tự động

*Hình 3. Code lấy dữ liệu từ file csv để chèn vào combobox*

**3.1.4. Hàm show\_tkinter\_new():**

**Mục đích:**

* Mở cửa sổ mới để thêm hồ sơ thí sinh vào DataFrame.

**Cách xây dựng:**

* Sử dụng Toplevel để tạo một cửa sổ mới.
* Các trường nhập liệu (Entry) và chọn giá trị (Combobox) được tạo cho các thông tin như "Name", "Age", "Gender", "Admission Test Score", "High School Percentage", "City", và "Admission Status".

**Cách hoạt động:**

* Người dùng điền thông tin vào các trường nhập liệu và nhấn nút "Lưu".
* Nếu tất cả các thông tin được điền đầy đủ, dữ liệu sẽ được thêm vào DataFrame.
* Sau khi lưu, cửa sổ sẽ đóng và danh sách tuyển sinh sẽ được cập nhật.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

*Hình 4. Code tạo tkinter mới*

**3.1.5. Hàm save()**

Là một hàm con của hàm show\_tkinter\_new()

**Mục đích:**

* Lưu thông tin mới nhập vào DataFrame sau khi người dùng nhấn "Lưu" trong cửa sổ thêm hồ sơ.

**Cách xây dựng:**

* Hàm nhận giá trị từ các trường nhập liệu trong Toplevel và kiểm tra tính hợp lệ.
* Dữ liệu mới được thêm vào DataFrame và sau đó cập nhật danh sách thí sinh hiển thị trên giao diện chính.

**Cách hoạt động:**

* Kiểm tra xem người dùng đã điền đầy đủ thông tin chưa. Nếu không, hiển thị cảnh báo.
* Tạo một DataFrame mới từ dữ liệu nhập và thêm nó vào DataFrame chính.
* Cập nhật lại danh sách thí sinh và đóng cửa sổ thêm hồ sơ.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, phần mềm

Mô tả được tạo tự động

*Hình 5. Code lưu dữ liệu vào chèn vào file csv*

**3.1.6. Hàm delete\_selected\_row(tree):**

**Mục đích:**

* Xóa dòng dữ liệu được chọn từ Treeview trong giao diện.

**Cách xây dựng:**

* Hàm nhận đối số là tree (cái Treeview cần xóa dòng).

**Cách hoạt động:**

* Kiểm tra nếu có dòng nào được chọn trong Treeview. Nếu có, dòng đó sẽ bị xóa.
* Nếu không có dòng nào được chọn, hiển thị cảnh báo yêu cầu người dùng chọn dòng cần xóa.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

*Hình 6. Code xóa dòng trong treeview*

**3.1.7. Hàm bctk():**

**Mục đích:**

* Hiển thị các biểu đồ thống kê liên quan đến dữ liệu tuyển sinh bằng cách tạo các hàm con khác liên quan tới từng biểu dồ

**Cách hoạt động:**

* Người dùng nhấn vào từng nút để xem biểu đồ thống kê tương ứng.
* Các biểu đồ này sẽ được vẽ bằng cách sử dụng các thư viện như matplotlib và seaborn (chưa được gọi trong hàm nhưng có thể bổ sung sau này).

**3.1.8. Hàm kqxt():**

**Mục đích:**

* Cung cấp giao diện để tìm kiếm kết quả xét tuyển của thí sinh dựa trên các thông tin nhập vào.

**Cách xây dựng:**

* Tạo các trường nhập liệu (Entry) để người dùng nhập các thông tin như "Name", "Age", "Gender", "Admission Test Score", "High School Percentage", và "City".
* Tạo một nút "Tra cứu" để thực hiện tìm kiếm.

**Cách hoạt động:**

* Sau khi người dùng nhấn "Tra cứu", hàm show\_result() được gọi để tìm kiếm thông tin trong DataFrame.
* Hiển thị kết quả xét tuyển của thí sinh (nếu có).

**3.1.9. Hàm show\_result(name, age, gender, admission\_test\_score, high\_school\_percentage, city():**

**Mục đích:**

* Tìm kiếm kết quả xét tuyển của thí sinh dựa trên thông tin nhập vào.

**Cách xây dựng:**

* Hàm sử dụng các tham số để lọc dữ liệu trong DataFrame.

**Cách hoạt động:**

* Kiểm tra nếu có trường nào bị thiếu thông tin, nếu có, hiển thị cảnh báo yêu cầu nhập đầy đủ.
* Kiểm tra và chuyển đổi dữ liệu (tuổi, điểm thi, tỷ lệ tốt nghiệp) sang đúng kiểu dữ liệu.
* Tìm kiếm thí sinh trong DataFrame và lấy kết quả xét tuyển của họ.
* Hiển thị kết quả hoặc cảnh báo nếu không tìm thấy thí sinh.

**Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phông chữ

Mô tả được tạo tự động**

*Hình 7. Code lọc kết quả khi ấn tra cứu*

**3.1.10. Hàm display\_result(admission\_status):**

**Mục đích:**

* Hiển thị kết quả xét tuyển của thí sinh.

**Cách xây dựng:**

* Sử dụng Label để hiển thị kết quả trong giao diện.

**Cách hoạt động:**

* Hàm hiển thị kết quả xét tuyển trong một khung Label trên giao diện.

**3.1.11. Hàm hide\_content():**

**Mục đích:**

* Ẩn nội dung của frame3 (xóa các widget hiện tại) để chuyển sang giao diện khác.

**Cách xây dựng:**

* Dùng winfo\_children() để lấy tất cả các widget trong frame3 và xóa chúng.

**Cách hoạt động:**

* Khi người dùng nhấn nút "Thoát" hoặc chuyển sang một giao diện khác, hàm này sẽ ẩn tất cả các widget hiện tại trong frame3 và chuẩn bị cho giao diện mới.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

*Hình 8. Code ẩn nội dung*

**3.1.11. Kĩ thuật xử lý trong các hàm:**

**def show\_list\_page(page):**

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

*Hình 9. Kĩ thuật xử lý hàm def show\_list\_page(page):*

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm

Mô tả được tạo tự động

*Hình 10. Kĩ thuật xử lý hàm def show\_list\_page(page):*

**1. Quản lý Dữ liệu với Pandas DataFrame**

Đầu vào của hàm là một trang (page), và mục tiêu của hàm là hiển thị dữ liệu từ một DataFrame (df) theo từng trang. Đây là một ví dụ của việc phân trang dữ liệu:

* **Cách tính dữ liệu hiển thị**:
  + page: Trang hiện tại được yêu cầu.
  + a: Số lượng bản ghi muốn hiển thị trên mỗi trang.
  + b = page: Chỉ định trang hiện tại.
  + c = page \* a: Chỉ số bắt đầu của phần dữ liệu muốn hiển thị.
  + d = c + a: Chỉ số kết thúc của phần dữ liệu muốn hiển thị.
  + df.iloc[c:d+1]: Sử dụng phương thức iloc của pandas để lấy một phần dữ liệu từ chỉ số c đến d (lấy d+1 vì iloc không bao gồm chỉ số cuối cùng).

**2. Quản lý Giao diện Người Dùng (GUI) với Tkinter**

Mã sử dụng thư viện Tkinter để hiển thị dữ liệu dưới dạng bảng trong ứng dụng giao diện người dùng. Các phần chính của giao diện là:

* **Treeview Widget**: Treeview là một widget của Tkinter cho phép hiển thị dữ liệu dạng bảng hoặc cây. Trong trường hợp này, nó được sử dụng để hiển thị dữ liệu dạng bảng.
  + Các cột được xác định với columns và có các tiêu đề (heading) cho mỗi cột.
  + Các cột cũng được tùy chỉnh về kích thước và cách căn chỉnh (column).
  + Dữ liệu của mỗi bản ghi được thêm vào Treeview qua vòng lặp for:

Trong đó, iterrows() là phương thức của pandas dùng để lặp qua các dòng của DataFrame. index + 1 là số thứ tự của dòng trong bảng, và các giá trị từ row là các cột dữ liệu của mỗi bản ghi.

* **Nút Chuyển Trang**:
* Nếu trang hiện tại (b) lớn hơn 0, nút "Trang trước" sẽ được hiển thị và cho phép người dùng quay lại trang trước.
* Nếu chỉ số kết thúc (d) nhỏ hơn tổng số bản ghi trong DataFrame, nút "Trang sau" sẽ được hiển thị để chuyển sang trang tiếp theo.
* Các nút này đều được liên kết với các hàm lambda để gọi lại hàm show\_list\_page với trang mới.
* **Chỉ thị Trang**:
* Label page\_label hiển thị thông tin về trang hiện tại và tổng số trang, được tính toán như sau: total\_pages = (len(df) // a) + (1 if len(df) % a != 0 else 0)
* len(df) // a: Chia tổng số bản ghi cho số bản ghi mỗi trang để tính số trang cơ bản. 1 if len(df) % a != 0 else 0: Nếu có dư (tức là số bản ghi không chia hết cho a), cần thêm một trang nữa.

**Kĩ thuật xử lý hàm def datacombobox():**

**Ảnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động**

*Hình 11. Kĩ thuật xử lý hàm def datacombobox()*

1. **Lọc dữ liệu trong Pandas**:

* Việc lọc dữ liệu dựa trên các điều kiện (City và Gender) là một kỹ thuật phổ biến khi làm việc với dữ liệu trong Pandas. Cách sử dụng các điều kiện và phép toán logic (&) giúp chúng ta dễ dàng lọc dữ liệu theo nhiều tiêu chí cùng lúc.
* Hàm df[(df['City'] == laycity) & (df['Gender'] == laygender)] là một cách hiệu quả và dễ hiểu để lọc dữ liệu trong Pandas.

**2. Cập nhật giao diện người dùng (GUI) với Tkinter:**

* **Xóa dữ liệu cũ trong Treeview**: Trước khi hiển thị dữ liệu mới, việc xóa tất cả các mục hiện có trong Treeview là một kỹ thuật phổ biến để tránh việc dữ liệu mới bị chồng lên dữ liệu cũ.
* **Chèn dữ liệu vào Treeview**: Chúng ta sử dụng phương thức tree.insert() để thêm từng dòng dữ liệu vào Treeview. Điều này cho phép chúng ta linh hoạt trong việc hiển thị dữ liệu theo cấu trúc bảng.

**3**. **Kiểm tra và xử lý dữ liệu rỗng**:

* Việc kiểm tra nếu DataFrame lọc được có rỗng hay không (if not locdata.empty) giúp xử lý trường hợp không có dữ liệu thỏa mãn và đảm bảo rằng ứng dụng không bị lỗi khi không có dữ liệu.
* Nếu không có dữ liệu, thông báo "Không có dữ liệu thỏa mãn yêu cầu!" được hiển thị trong bảng, thay vì một bảng trống, giúp người dùng hiểu rõ hơn về trạng thái.

**4**. **Sử dụng iterrows() để lặp qua DataFrame**:

* Hàm iterrows() trong Pandas được sử dụng để lặp qua từng dòng trong DataFrame, giúp dễ dàng trích xuất giá trị của mỗi cột và chèn vào Treeview.

**Kĩ thuật xử lý hàm def save():**

**Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Hệ điều hành

Mô tả được tạo tự động**

*Hình 12. Kĩ thuật xử lý hàm def save()*

**1. Lấy dữ liệu từ giao diện người dùng (Tkinter Entry Widgets)**:

* Hàm .get() được sử dụng để lấy giá trị người dùng nhập từ các widget Entry trong Tkinter. Đây là một kỹ thuật cơ bản khi làm việc với các widget nhập liệu trong giao diện Tkinter.

**2.** **Kiểm tra tính hợp lệ của dữ liệu**:

* Hàm all() giúp kiểm tra xem tất cả các trường nhập liệu có giá trị hay không. Nếu không, sử dụng messagebox.showwarning để cảnh báo người dùng về việc thiếu thông tin. Đây là một kỹ thuật phổ biến để đảm bảo tính toàn vẹn của dữ liệu người dùng.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, Hệ điều hành

Mô tả được tạo tự động

*Hình 13. Thông báo khi người dùng nhập thiếu thông tin*

**3**. **Chỉnh sửa và cập nhật DataFrame**:

* **Tạo một DataFrame mới từ dữ liệu nhập liệu**: Dữ liệu nhập được đóng gói thành một DataFrame nhỏ (dfnew) và sau đó nối với DataFrame chính (df) sử dụng pd.concat(). Đây là một kỹ thuật phổ biến trong Pandas để thêm dữ liệu mới vào DataFrame.
* **ignore\_index=True**: Sử dụng ignore\_index=True để tự động reset chỉ số trong DataFrame, đảm bảo rằng không có chỉ số bị trùng hoặc lộn xộn.

**4**. **Cập nhật giao diện người dùng**:

* Hàm show\_list\_page(b) được gọi lại để làm mới Treeview, giúp dữ liệu mới (bao gồm bản ghi người dùng vừa nhập) được hiển thị ngay lập tức trong giao diện người dùng. Việc gọi lại hàm này cho phép làm mới giao diện mà không cần phải tải lại toàn bộ ứng dụng.

**5**. **Đóng cửa sổ nhập liệu**:

* Việc sử dụng destroy() để đóng cửa sổ Tkinter sau khi lưu dữ liệu là một kỹ thuật đơn giản nhưng hiệu quả để quản lý giao diện người dùng.

**Kĩ thuật xử lý hàm def delete\_selected\_row(tree):**

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

*Hình 14. Kĩ thuật xử lý hàm def delete\_selected\_row(tree):*

**1**. **Lấy mục được chọn từ Treeview**:

* tree.selection() là một phương thức của widget Treeview trong Tkinter, dùng để lấy danh sách các mục hiện tại đang được chọn. Mỗi mục trong Treeview có một ID duy nhất, được trả về bởi selection(). Phương thức này rất hữu ích khi cần thực hiện các thao tác như xóa, chỉnh sửa, hoặc xem thông tin chi tiết về các mục đã chọn.

**2**. **Kiểm tra tính hợp lệ của lựa chọn**:

* Việc kiểm tra nếu selected\_item có giá trị hay không (if selected\_item:) là một kỹ thuật để đảm bảo rằng người dùng đã chọn một mục trước khi tiến hành xóa. Nếu không có mục nào được chọn, sẽ có thông báo cảnh báo (messagebox.showwarning) yêu cầu người dùng phải chọn mục cần xóa.

**3**. **Xóa dòng trong Treeview**:

* tree.delete(selected\_item) xóa mục đã chọn khỏi Treeview. Phương thức delete() của Treeview là kỹ thuật để loại bỏ các mục không cần thiết trong bảng mà người dùng đang làm việc. Nó giúp làm sạch dữ liệu trong bảng mà không làm ảnh hưởng đến các mục khác.

**4**. **Sử dụng Messagebox để cảnh báo người dùng**:

* messagebox.showwarning() được sử dụng để hiển thị một cửa sổ cảnh báo, thông báo cho người dùng nếu họ cố gắng xóa mà không chọn dòng nào. Đây là một cách phổ biến trong Tkinter để tương tác với người dùng, giúp cải thiện trải nghiệm người dùng (UX).

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, Hệ điều hành

Mô tả được tạo tự động

*Hình 15. Thông báo khi chưa chọn dòng cần xóa.*

**Kĩ thuật xử lý hàm def show\_result(name, age, gender, admission\_test\_score, high\_school\_percentage, city):**

**Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Hệ điều hành

Mô tả được tạo tự động**

*Hình 16. Kĩ thuật xử lý hàm def show\_result(name, age, gender, admission\_test\_score, high\_school\_percentage, city):*

*Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động*

*Hình 16. Kĩ thuật xử lý hàm def show\_result(name, age, gender, admission\_test\_score, high\_school\_percentage, city):*

**1**. **Kiểm tra và xử lý dữ liệu nhập từ người dùng**:

* Hàm sử dụng kiểm tra điều kiện để đảm bảo tất cả các thông tin cần thiết đã được nhập. Khi nhập thiếu thông tin thì sẽ hiển thị thông báo

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, Hệ điều hành

Mô tả được tạo tự động

*Hình 17. Thông báo khi nhập thiếu thông tin*

* Hàm try-except giúp xử lý các lỗi khi người dùng nhập sai kiểu dữ liệu (ví dụ: nhập chữ thay vì số). Nếu nhập sai sẽ hiển thị thông báo

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

*Hình 18. Thông báo khi nhập sai dữ liệu*

**2**. **Tìm kiếm trong DataFrame**:

* Phương thức str.lower() giúp thực hiện so sánh không phân biệt chữ hoa và chữ thường.
* Các phép toán điều kiện kết hợp với & giúp tìm kiếm các bản ghi thỏa mãn nhiều điều kiện trong DataFrame.

**3**. **Xử lý kết quả tìm kiếm**:

* Hàm kiểm tra nếu kết quả tìm kiếm là rỗng (result.empty), giúp tránh việc thực hiện thao tác trên DataFrame trống. Nghĩa là nếu nhập sai dữ liệu trong df thì sẽ hiển thị thông báo

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, Hệ điều hành

Mô tả được tạo tự động

*Hình 19. Thông báo khi không tìm thấy thông tin*

* Nếu có kết quả, lấy giá trị Admission Status của thí sinh và hiển thị nó.

**4**. **Hiển thị thông báo lỗi với messagebox.showwarning()**:

* Các cảnh báo được hiển thị nếu dữ liệu nhập thiếu hoặc sai kiểu, giúp người dùng dễ dàng nhận biết vấn đề và sửa chữa.