

**2024-2025学年 第二学期**

《APP开发与实践》期末作业

《学生基本信息管理》操作手册说明书

院 系： 国际工商管理学院

专 业： 信息管理与信息系统

学 号： 0221147010

姓 名：刘蕊

2025年6月

**目录**

[**1. 系统概述** 3](#_Toc199089042)

[**2. 运行环境要求** 3](#_Toc199089043)

[**3. 程序启动与登录指南** 3](#_Toc199089044)

[**3.1 启动流程** 3](#_Toc199089045)

[**3.2** **登录界面** 3](#_Toc199089046)

[**3.2.1 核心组件** 3](#_Toc199089047)

[**3.2.2** **布局设计** 4](#_Toc199089048)

[**3.2.3** **用户交互流程** 4](#_Toc199089049)

[**3.3 认证机制** 6](#_Toc199089050)

[**3.3.1 数据源** 6](#_Toc199089051)

[**3.3.2 认证流程** 6](#_Toc199089052)

[**3.3.3** **输入验证** 7](#_Toc199089053)

[**3.3.4 错误处理与用户反馈** 7](#_Toc199089054)

[**4. 主界面功能** 7](#_Toc199089055)

[**4.1 主界面构成** 8](#_Toc199089056)

[**4.1.1 核心组件** 8](#_Toc199089057)

[**4.1.2 布局设计** 9](#_Toc199089058)

[**4.1.3** **用户交互流程** 9](#_Toc199089059)

[**4.2 功能详解** 16](#_Toc199089060)

[**4.2.1 数据源** 16](#_Toc199089061)

[**4.2.2** **搜索功能** 16](#_Toc199089062)

[**4.2.3** **表格过滤** 17](#_Toc199089063)

[**4.2.4** **新增记录** 17](#_Toc199089064)

[**4.2.5** **编辑记录** 17](#_Toc199089065)

[**4.2.6** **删除记录** 17](#_Toc199089066)

[**4.2.7** **查看统计** 17](#_Toc199089067)

[**5. 数据可视化功能** 17](#_Toc199089068)

[**5.1 功能概述** 17](#_Toc199089069)

[**5.2 界面构成** 18](#_Toc199089070)

[**5.2.1** **窗口设置** 18](#_Toc199089071)

[**5.2.2 图表区域** 18](#_Toc199089072)

[**5.3 功能实现详解** 18](#_Toc199089073)

[**5.3.1** **性别分布饼图** 18](#_Toc199089074)

[**5.3.2 院系人数柱状图** 20](#_Toc199089075)

[**5.3.3** **切换机制** 21](#_Toc199089076)

[**6.附件** 21](#_Toc199089077)

**1. 系统概述**

本系统是基于PyQt6框架开发的学生信息管理工具，集成用户认证、数据管理、可视化统计三大核心模块。系统采用分层架构设计，前端界面与后端数据处理分离，通过Pandas实现Excel数据操作，Matplotlib/PyQtGraph完成图表渲染，用户可通过直观的图形界面完成从登录认证到数据维护的全流程操作。

**2. 运行环境要求**

| 组件 | 版本要求 | 备注说明 |
| --- | --- | --- |
| Python | ≥3.8 | 推荐使用3.10版本 |
| PyQt6 | 6.5.0+ | 核心GUI框架 |
| Pandas | 2.0.0+ | Excel数据处理 |
| Matplotlib | 3.7.0+ | 图表绘制 |
| OpenPyXL | 3.1.0+ | Excel文件读写 |

安装命令：pip install pyqt6 pandas matplotlib openpyxl

**3. 程序启动与登录指南**

**3.1 启动流程**

1. 双击main.py或通过命令行执行：python main.py。
2. 首次启动将自动创建数据目录./data/，包含：

用户认证文件：user.txt（每行格式：用户名:密码）

示例数据文件：学生数据示例.xlsx

* 1. **登录界面**

登录界面是“学生基本信息管理系统”的入口，旨在通过用户身份验证保护学生数据的安全性。LoginDialog 类继承自 QDialog，基于 PyQt6 的 QDialog 构建，提供直观的交互体验，通过合理的布局和输入验证提升用户体验。

**3.2.1 核心组件**

1. 窗口标题：设置为“登录”。

self.setWindowTitle("登录")

1. 用户名输入框：创建标签“用户名”，创建输入框，使用 QLineEdit，允许用户输入用户名，限制用户名最多可输入 20 个字符，最终绑定输入变化事件，动态验证输入。

self.username\_label = QLabel("用户名：")  
self.username\_input = QLineEdit()  
self.username\_input.setMaxLength(20)  
self.username\_input.textChanged.connect(self.validate\_input)

1. 密码输入框：标签为“密码”，使用 QLineEdit创建输入框，并设置为密码模式（setEchoMode(QLineEdit.Password)），以隐藏输入字符，保护隐私。同时限制密码至少 6 个字符，通过代码逻辑验证。最后绑定输入变化事件。

self.password\_label = QLabel("密码：")  
self.password\_input = QLineEdit()  
self.password\_input.setEchoMode(QLineEdit.EchoMode.Password)  
self.password\_input.textChanged.connect(self.validate\_input)

1. 登录按钮：创建“登录”按钮，初始状态为禁用（setEnabled(False)），防止无效提交。最后绑定点击事件，当用户名和密码输入均满足要求时，按钮启用。

self.login\_button = QPushButton("登录")  
self.login\_button.setEnabled(False)  
self.login\_button.clicked.connect(self.check\_login)

1. 提示信息区域：

使用 QLabel 显示动态提示，例如“用户名或密码错误！剩余尝试次数：X”或“登录成功！”。

1. 页脚：显示开发者信息：“刘蕊 0221147010”，通过 QLabel 居中显示在窗口底部。
   * 1. **布局设计**
2. 布局方式：采用 QVBoxLayout 垂直布局，确保组件从上到下排列。
3. 样式：通过 PyQt6 的样式表美化界面。
4. 窗口大小：固定为适合内容的尺寸，确保界面紧凑。
   * 1. **用户交互流程**
5. 界面初始化：

启动程序时，LoginDialog 实例化并显示。

用户名和密码输入框为空，登录按钮禁用。

图形用户界面, 应用程序

AI 生成的内容可能不正确。

1. 输入用户名和密码：

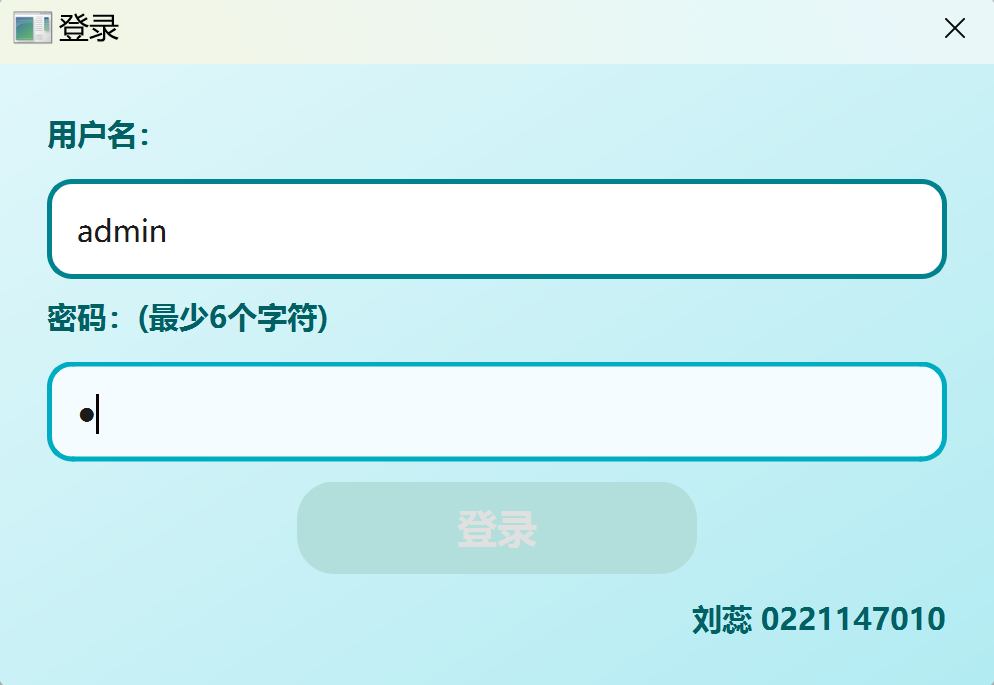
用户在输入框中键入内容。

每次输入变化时，检查输入是否满足条件：

用户名不为空且不超过20个字符。

密码长度至少6个字符。

如果条件满足，启用登录按钮；否则保持禁用状态。



1. 点击登录按钮：

点击后执行认证逻辑。

根据认证结果显示提示：

成功：显示“登录成功！”并关闭对话框，打开主界面。

图形用户界面, 应用程序

AI 生成的内容可能不正确。

失败：显示“用户名或密码错误！剩余尝试次数：X”，X 为剩余次数。

图形用户界面, 文本, 应用程序

AI 生成的内容可能不正确。

如果失败次数达到 5 次，显示“已达最大尝试次数！请重新启动程序”，并关闭窗口。



**3.3 认证机制**

认证机制是登录功能的核心，确保只有合法用户可以访问系统，它通过读取外部文件 user.txt 中的凭据，结合失败尝试限制，实现安全性和用户体验的平衡。

**3.3.1 数据源**

1. 文件路径：./data/user.txt。
2. 文件格式：每行存储一对用户名和密码，用冒号分隔（如 admin:123456）。
3. 加载逻辑：

使用def load\_accounts(self)读取文件内容，逐行读取，忽略空行。接着每行按冒号分割，创建键值对（用户名：密码）。最后返回字典，如 {"admin": "123456"}。

错误处理：如果文件不存在，显示错误：“用户文件未找到！请确保路径正确”。返回空字典，确保程序不会崩溃。

**3.3.2 认证流程**

1. 获取输入：从 QLineEdit 获取用户名和密码的文本内容。
2. 验证凭据：检查用户名是否存在于返回的字典中。如果存在，比较输入的密码与字典中的密码是否一致。
3. 处理结果：若成功，显示成功消息“登录成功！”，调用 accept() 关闭对话框，允许主窗口打开；若失败，增加失败计数器 ，初始值为 0。失败次数小于 5：显示警告“用户名或密码错误！剩余尝试次数：X”。失败次数达到 5：显示消息“已达最大尝试次数！请重新启动程序”。调用 reject() 关闭对话框，终止登录尝试。为了安全性，密码以隐藏模式输入，防止泄露；失败尝试限制（5 次）防止暴力破解。
   * 1. **输入验证**

监听用户名和密码输入框的信号，获取用户名和密码输入（username\_input.text() 和 password\_input.text()）。检查条件为“用户名非空且长度 ≤ 20”（len(username) > 0）和“密码长度 ≥ 6”len(password) >= 6），如果两个条件都满足，启用登录按钮，否则禁用。目的是防止用户提交无效输入，提高交互效率。

if len(username) > 0 and len(password) >= 6:  
 self.login\_button.setEnabled(True)  
else:  
 self.login\_button.setEnabled(False)

如果密码非空但长度不足 6 个字符，更新密码标签为“密码：(最少6个字符)”；否则恢复默认标签“密码：”

if len(password) > 0 and len(password) < 6:  
 self.password\_label.setText("密码：(最少6个字符)")  
else:  
 self.password\_label.setText("密码：")

**3.3.4 错误处理与用户反馈**

1. 文件读取错误：如果 user.txt 不可访问，程序不会崩溃，而是显示错误消息并允许用户检查文件路径。
2. 认证失败：通过动态提示（剩余尝试次数）引导用户，避免盲目尝试。
3. 尝试次数限制：5 次失败后强制关闭，增强安全性，同时提示用户重新启动程序。

**4. 主界面功能**

主界面是“学生基本信息管理系统”的核心操作平台，旨在为用户提供一个直观、高效的管理学生数据的环境。它集成了数据的读取、搜索、编辑、新增、删除以及统计等多种功能，基于 PyQt6 的 QMainWindow 构建，通过精心设计的布局和交互逻辑，确保用户能够轻松完成各项任务。

**4.1 主界面构成**

**4.1.1 核心组件**

1. 窗口标题“学生基本信息管理”

self.setWindowTitle("学生基本信息管理")

1. 搜索栏位于界面顶部，由一个文本输入框（QLineEdit）和一个“搜索”按钮（QPushButton）组成，“搜索”按钮配有放大镜图标，点击后触发搜索功能。输入框提示文本为“输入姓名搜索”，引导用户操作，clicked\_connect绑定搜索功能。

search\_layout = QHBoxLayout()  
self.search\_input = QLineEdit()  
self.search\_input.setPlaceholderText("输入姓名搜索")  
self.search\_button = QPushButton("搜索")  
self.search\_button.clicked.connect(self.search\_data)  
search\_layout.addWidget(self.search\_input)  
search\_layout.addWidget(self.search\_button)  
layout.addLayout(search\_layout)

1. 学生信息表格

使用 QTableWidget 动态展示学生数据。

self.table = QTableWidget()

FilterHeader 自定义表头，支持列过滤，设置列名（姓名、性别、民族、系别、专业和省份）。

header = FilterHeader(self.table)

self.table.setHorizontalHeader(header)

setSectionsClickable(True)启用点击表头，on\_section\_clicked生成上下文菜单，显示唯一值并支持多选过滤。

class FilterHeader(QHeaderView):  
 def \_\_init\_\_(self, parent):  
 super().\_\_init\_\_(Qt.Orientation.Horizontal, parent)  
 self.setSectionsClickable(True)  
 self.sectionClicked.connect(self.on\_section\_clicked)  
 self.filters = {}  
 self.parent = parent

apply\_filter 和 clear\_filters函数用于管理过滤状态，更新表格显示，具体代码均在FilterHeader类下。另外，为优化性能，使用setSortingEnabled(False) 和 setUpdatesEnabled(False)，在批量更新时禁用排序和刷新。

1. 操作按钮区

QHBoxLayout 包含四个按钮：新增、编辑、删除、查看统计。

button\_layout = QHBoxLayout()

每个按钮绑定图标和点击事件。

self.add\_button.clicked.connect(self.add\_record)  
self.edit\_button.clicked.connect(self.edit\_record)  
self.delete\_button.clicked.connect(self.delete\_record)  
self.stats\_button.clicked.connect(self.show\_statistics)

新增时，使用 QInputDialog.getText顺序提示输入字段（姓名、性别等），使用验证函数检查输入：姓名：2-10 个汉字；性别：仅“男”或“女”；院系：必须在现有值中；专业：2-15 个汉字；民族/省份：可选，最多 6/10 个汉字。有效输入后，创建新 DataFrame 行，追加到 self.df 并保存。遇到错误输入时，无效输入弹出警告，允许重试或取消，另外，保存失败时显示错误。

编辑时，根据 currentRow 和 currentColumn 判断编辑范围：单元格编辑：修改单一字段；整行编辑：更新所有字段。使用 get\_original\_row\_index 映射显示行到原始索引。最后验证输入，更新 DataFrame 并保存。

删除时，判断选中单元格或整行：单元格，检查是否必填字段，弹出选项（删除字段或整行）；整行，确认后删除。使用 drop 删除 DataFrame 行，保存文件。安全机制为，QMessageBox 确认操作，防止误删，以及必填字段（姓名等）不可删除。

查看统计为时，使用show\_statistics，进行实例化StatisticsWindow，传递 self.df。

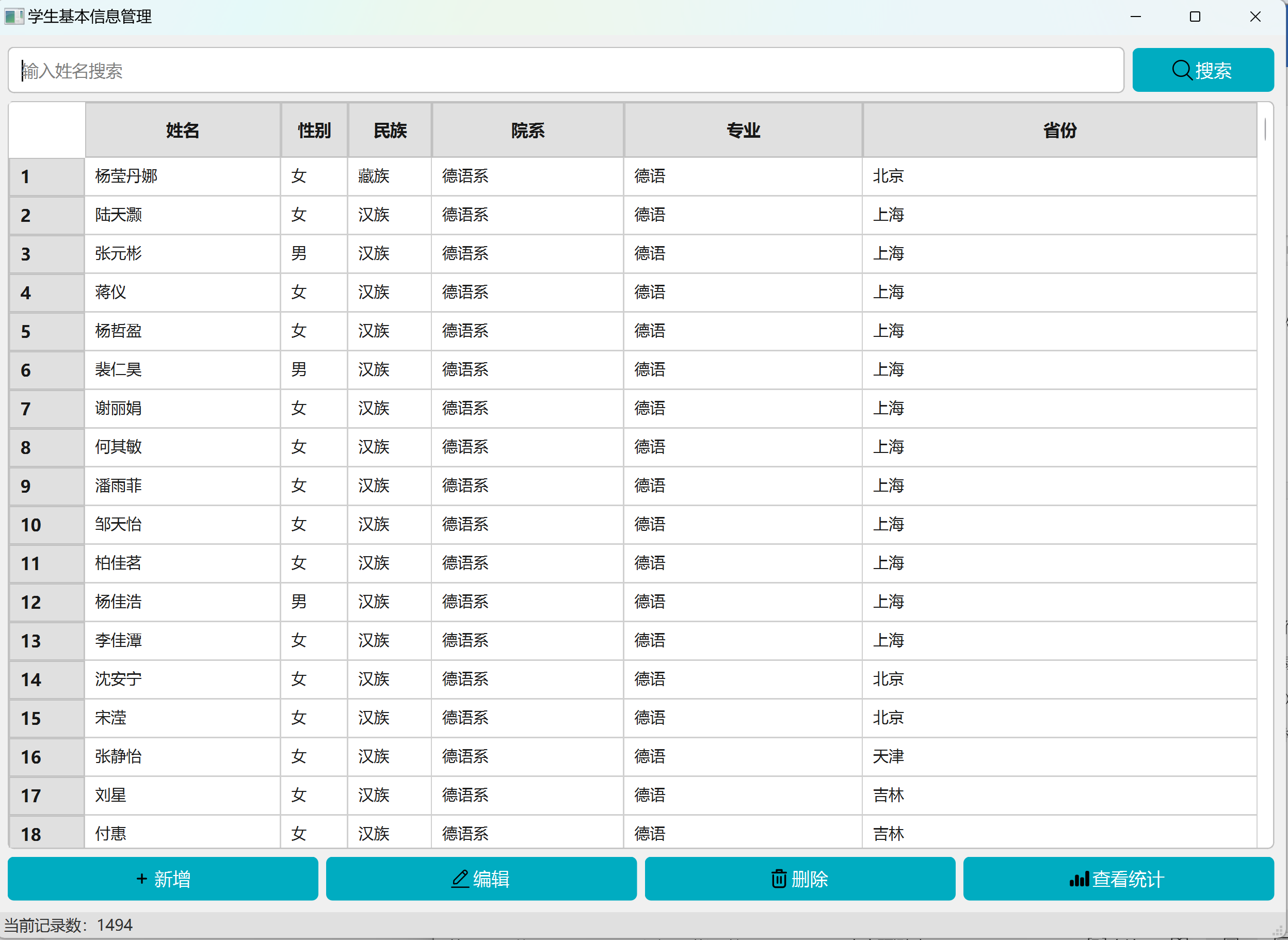
1. 状态栏

创建 QStatusBar 实例 self.status\_bar，用于显示实时记录数量，格式为“当前记录数：X”，通过 setStatusBar 集成到窗口底部。

self.status\_bar = QStatusBar()  
self.setStatusBar(self.status\_bar)

**4.1.2 布局设计**

1. 整体布局：采用 QVBoxLayout 垂直排列，从上到下依次为搜索栏、表格、按钮区和状态栏。
2. 搜索栏：通过 QHBoxLayout 将输入框和按钮水平排列。
3. 表格：自动填充中间空间，列宽根据内容动态调整。
4. 按钮区：四个按钮水平分布，视觉上整齐统一。
   * 1. **用户交互流程**
5. 初始化：登录成功后，使用 pd.read\_excel，主界面加载并显示学生数据示例.xlsx中的数据，状态栏显示初始记录数量。try-except 捕获 FileNotFoundError，显示错误提示并返回空 DataFrame。使用display\_data初始化表格行数和列数，遍历 DataFrame，转换缺失值（NaN）为“无”，批量设置 QtableWidgetItem，最后恢复更新和排序，确保数据正确显示。



1. 搜索：用户输入关键词，点击“搜索”按钮，系统在所有列中进行模糊搜索，更新表格显示匹配记录。具体逻辑为，获取 search\_input 文本，转为小写并去除首尾空格，使用 df.apply 逐行搜索，匹配任意列中包含搜索词的记录。接着使用mask 过滤 DataFrame，更新表格显示。另外，为优化显示，只搜索非空值（pd.notna），提高效率，以及使用 display\_filtered\_data 维护原始索引。背景图案

   AI 生成的内容可能不正确。

未找到匹配项时，弹出提示“未找到匹配的记录”。

图形用户界面, 应用程序

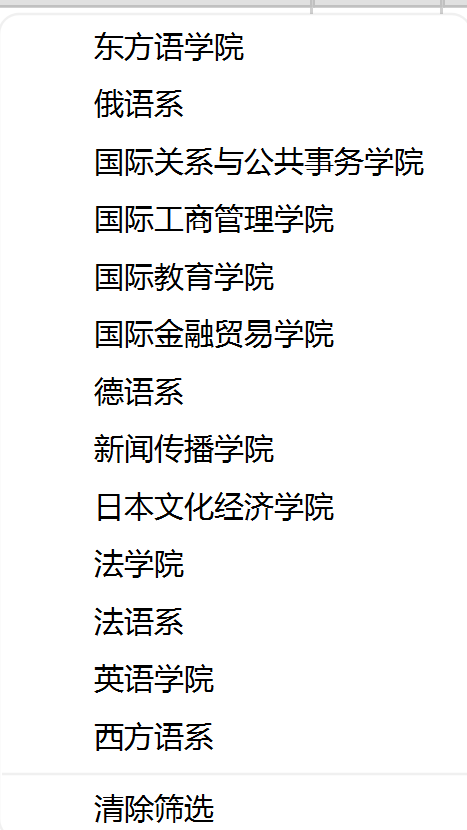
AI 生成的内容可能不正确。

1. 表格操作：点击列标题，弹出过滤菜单，选择值（如性别“女”）进行过滤。

支持多选过滤，选择性别“女”和院系“国际工商管理学院”。图形用户界面, 应用程序, 表格, Excel

AI 生成的内容可能不正确。

点击“清除过滤器”恢复全部数据。



1. 按钮操作：

新增：弹出对话框，输入新记录。

图形用户界面, 文本, 应用程序, 聊天或短信

AI 生成的内容可能不正确。图形用户界面, 文本, 应用程序, 聊天或短信

AI 生成的内容可能不正确。图形用户界面, 文本, 应用程序, 聊天或短信

AI 生成的内容可能不正确。文本

AI 生成的内容可能不正确。图形用户界面, 文本

AI 生成的内容可能不正确。图形用户界面, 文本, 应用程序, 聊天或短信

AI 生成的内容可能不正确。图形用户界面, 文本, 应用程序, 聊天或短信

AI 生成的内容可能不正确。应用程序

AI 生成的内容可能不正确。

编辑：选中行或单元格后，弹出编辑对话框。

图形用户界面, 文本, 应用程序, 聊天或短信

AI 生成的内容可能不正确。图形用户界面, 文本, 应用程序

AI 生成的内容可能不正确。图形用户界面, 文本, 应用程序, 聊天或短信

AI 生成的内容可能不正确。

若输入为空，则报错提示。

图形用户界面, 应用程序

AI 生成的内容可能不正确。

删除：选中后弹出确认框，确认后删除。

图形用户界面, 文本, 应用程序, 聊天或短信

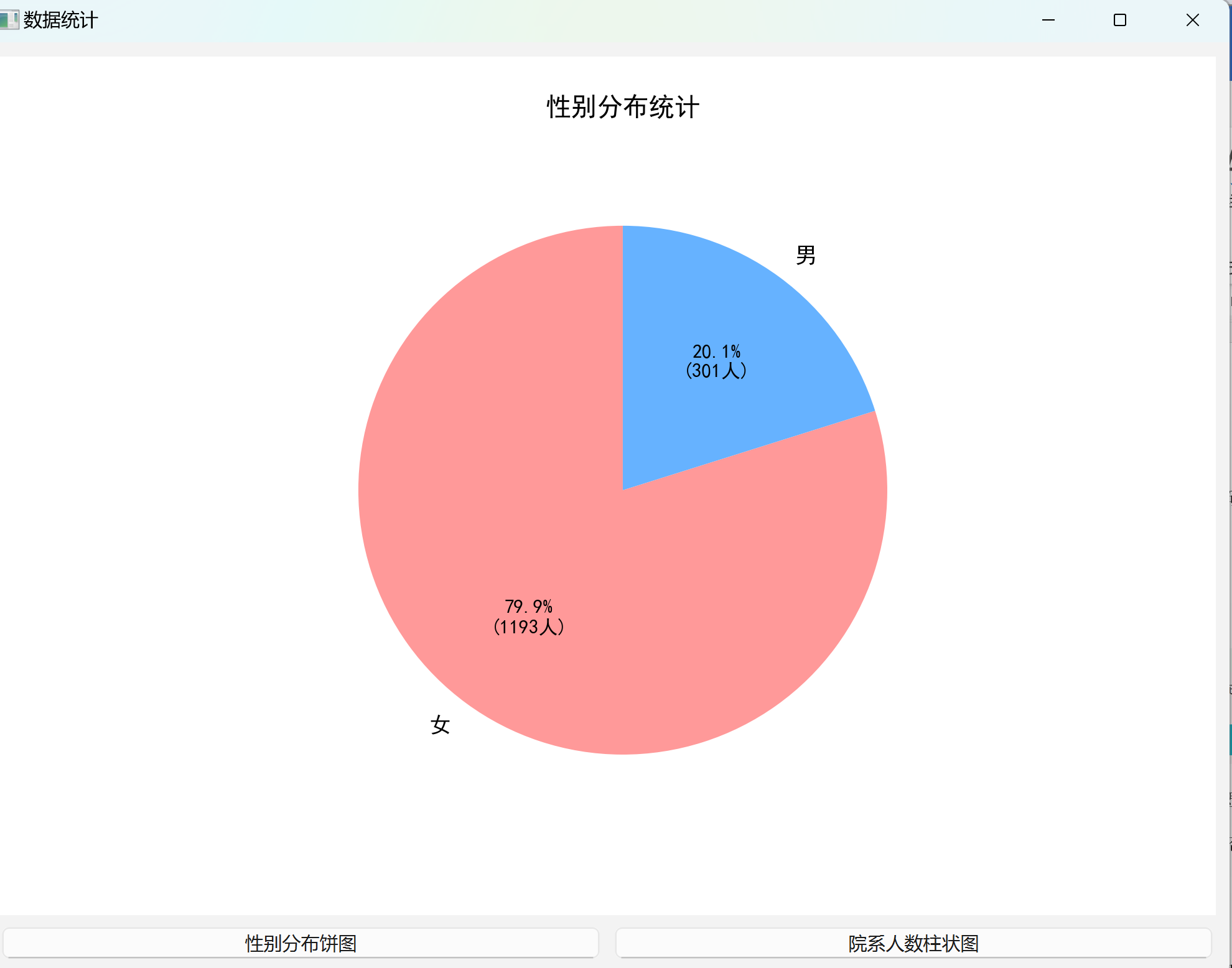
AI 生成的内容可能不正确。

图形用户界面, 应用程序

AI 生成的内容可能不正确。 图形用户界面, 文本, 应用程序, 聊天或短信

AI 生成的内容可能不正确。

查看统计：打开统计窗口，显示图表。



图表, 条形图

AI 生成的内容可能不正确。

（部分）

1. 状态更新：数据操作后，状态栏实时更新记录数量。

**4.2 功能详解**

**4.2.1 数据源**

1. 文件路径：./data/学生数据示例.xlsx
2. 包含姓名、性别、民族、院系、专业、省份6列，共1494行数据

加载数据集到DataFrame，初始化表格显示和状态栏。

self.file\_path = "data/学生数据示例.xlsx"  
self.df = self.load\_student\_data()  
self.display\_data(self.df)  
self.update\_status\_bar()

* + 1. **搜索功能**

获取输入框文本，转为小写，在 DataFrame 中模糊搜索，筛选匹配行，最后更新表格显示筛选结果。支持模糊搜索（如输入“张”匹配所有含“张”的记录），状态栏显示筛选后数量。



* + 1. **表格过滤**

点击列标题，显示该列唯一值菜单，选择值过滤表格，支持多选，可清除过滤器恢复数据。

* + 1. **新增记录**

弹出对话框，要求输入各字段。接着依次验证输入姓名、性别、民族、院系、专业、省份。最后有效输入追加到数据并保存。其中必填字段姓名、性别、院系、专业不为空，选填民族、省份可为空。

* + 1. **编辑记录**

单元格编辑，直接修改并保存；整行编辑：弹出对话框修改。

* + 1. **删除记录**

删除字段，设为“无”；删除整行，通过QMessageBox确认后移除，防止误删。

* + 1. **查看统计**

新窗口显示性别饼图（matplotlib实现）和院系别柱状图（pyqtgraph实现），支持切换图表。

**5. 数据可视化功能**

**5.1 功能概述**

“学生基本信息管理系统”的数据可视化功能通过 StatisticsWindow 类实现，以生成性别分布饼图和院系分布柱状图，为用户提供直观的学生数据统计分析。

**5.2 界面构成**

* + 1. **窗口设置**

初始化方法 (\_\_init\_\_)设置窗口标题。

self.setWindowTitle("数据统计")

使用 QVBoxLayout 包含 FigureCanvas（matplotlib 画布）、PlotWidget（pyqtgraph 部件）和按钮布局。

# 创建中央部件  
central\_widget = QWidget()  
self.setCentralWidget(central\_widget)  
layout = QVBoxLayout(central\_widget)

button\_layout（QHBoxLayout）包含“性别分布饼图”和“院系人数柱状图”按钮。

# 按钮切换图表  
button\_layout = QHBoxLayout()  
self.pie\_button = QPushButton("性别分布饼图")  
self.bar\_button = QPushButton("院系人数柱状图")

**5.2.2 图表区域**

matplotlib 画布 (self.canvas):用于绘制饼图，基于 FigureCanvasQTAgg 集成到 PyQt6。

self.figure = Figure()  
self.canvas = FigureCanvas(self.figure)  
layout.addWidget(self.canvas)

pyqtgraph 部件 (self.plot\_widget):用于绘制柱状图，提供高效的实时渲染。

self.plot\_widget = pg.PlotWidget()  
layout.addWidget(self.plot\_widget)

切换按钮:self.pie\_button 和 self.bar\_button 分别触发 plot\_pie\_chart 和 plot\_bar\_chart 方法。默认显示饼图，self.plot\_pie\_chart(df) 在初始化时调用。

self.pie\_button.clicked.connect(lambda: self.plot\_pie\_chart(df))  
self.bar\_button.clicked.connect(lambda: self.plot\_bar\_chart(df))  
button\_layout.addWidget(self.pie\_button)  
button\_layout.addWidget(self.bar\_button)

**5.3 功能实现详解**

* + 1. **性别分布饼图**

使用plot\_pie\_chart方法实现。

（1）数据准备

统计性别分布（例如“男”和“女”的数量）。

gender\_counts = df["性别"].value\_counts()

提取标签和数值。

labels = gender\_counts.index.tolist()  
sizes = gender\_counts.values.tolist()

（2）绘制逻辑

清除之前的内容，创建子图，设置标题，并自定义标签，显示百分比和人数。

self.figure.clear()  
ax = self.figure.add\_subplot(111)  
  
def func(pct, allvals):  
 absolute = int(np.round(pct / 100. \* np.sum(allvals)))  
 return f"{pct:.1f}%\n({absolute}人)"  
  
patches, texts, autotexts = ax.pie(sizes,  
 labels=labels,  
 autopct=lambda pct: func(pct, sizes),  
 startangle=90,  
 colors=['#FF9999', '#66B2FF'])  
  
plt.setp(autotexts, size=9, weight="bold")  
plt.setp(texts, size=10)  
  
ax.set\_title("性别分布统计", pad=20, size=12, weight="bold")  
self.canvas.draw()

（3）结果分析:

图表, 饼图

AI 生成的内容可能不正确。

女生占 79%，1193人，以较大粉色部分表示；男生占 20.1%，301人，以较小浅蓝色部分表示。

**5.3.2 院系人数柱状图**

使用plot\_bar\_chart方法实现。

（1）数据准备:

统计各院系人数。

dept\_counts = df["院系"].value\_counts()

提取院系名称和数量。

labels = dept\_counts.index.tolist()  
values = dept\_counts.values.tolist()

（2）绘制逻辑:

清除之前的内容，创建柱状图，x 轴为院系索引，高度为人数，添加数值标签，使用 pg.TextItem 在柱顶显示人数。调整显示范围，确保所有柱子可见。

self.plot\_widget.clear()

x = range(len(labels))  
bar = pg.BarGraphItem(x=x, height=values, width=0.6, brush='#00ACC1')  
self.plot\_widget.addItem(bar)  
  
# 添加数值标签  
for i, v in enumerate(values):  
 text = pg.TextItem(str(v), anchor=(0.5, 1.0))  
 text.setPos(i, v)  
 self.plot\_widget.addItem(text)  
  
# 设置坐标轴  
self.plot\_widget.getAxis('bottom').setTicks([[(i, label) for i, label in enumerate(labels)]])  
self.plot\_widget.setTitle("院系人数分布")  
self.plot\_widget.setLabel('left', '人数')  
  
# 调整显示范围  
self.plot\_widget.setRange(xRange=[-0.5, len(labels) - 0.5],  
 yRange=[0, max(values) \* 1.2])

（3）结果分析:

通过滑动鼠标滚轮可以放大或缩小饼图，由于无法全部展示院系人数分布（否则x轴标签会有所重叠），在此进行文字描述：英语学院164人，国际金融贸易学院 150人，日本文化经济学院 150人，国际工商管理学院 146人，国际教育学院 140人，新闻传播学院 136人，西方语系 115人，东方语学院 112人，德语系 100人，法学院 100人，法语系 70人，国际关系与公共事务学院 67人，俄语系43人。

图表, 条形图

AI 生成的内容可能不正确。

* + 1. **切换机制**

（1）绑定按钮事件

self.pie\_button.clicked.connect(lambda: self.plot\_pie\_chart(df))  
self.bar\_button.clicked.connect(lambda: self.plot\_bar\_chart(df))

（2）制图表显示。

button\_layout.addWidget(self.pie\_button)  
button\_layout.addWidget(self.bar\_button)

（3）用户可随时切换查看不同图表，界面保持响应。

**6.附件**

(1)数据文件用户凭据文件：./data/user.txt学生数据文件：./data/学生数据示例.xlsx(2)图标文件新增图标：./icons/add.png删除图标：./icons/delete.png编辑图标：./icons/edit.png搜索图标：./icons/search.png统计图标：./icons/stats.png(3)程序文件./main.py

./main.exe

（4）ui文件

登录对话框的ui文件：./main.ui

主窗口的ui文件：./main\_window.ui

统计窗口的ui文件：./statistics\_window.ui