# 个人报告__马佳瑶



## 1. 概述

本项目实现了一个简易的图书管理系统，完成了管理员和普通读者的权限划分。

- 读者板块功能包括注册、登录、借书还书、图书查询、查看个人信息和欠款图书信息。
- 管理员板块功能涵盖了添加新书、添加新读者信息、对图书信息的增删改、查询所有或指定读者信息、查询所有到期未归还的图书信息，以及查询所有读者的欠款情况。

## 2. 项目的可行性研究与需求分析

系统预计完成时间为15天，开发小组为3人。

本项目可以利用现有的数据库管理系统SQL Server以及后端框架Flask进行开发，使用 css 和script优化html页面。其中SQL Server是本次数据库课程主要教学的内容，而Flask框架是一种兼具轻量级和灵活性的后端框架。

使用的相关工具需满足版本需求如下：

**数据库系统**

- SQL Server 2019

**开发工具**

- PyCharm2023.2

**其他支持要素**

- PowerDesigner: 绘制物理数据模型
- 飞书：项目管理、协同合作

# 3. 项目实训的基本原理和采用的主要方法与技术

## 3.1 基本原理

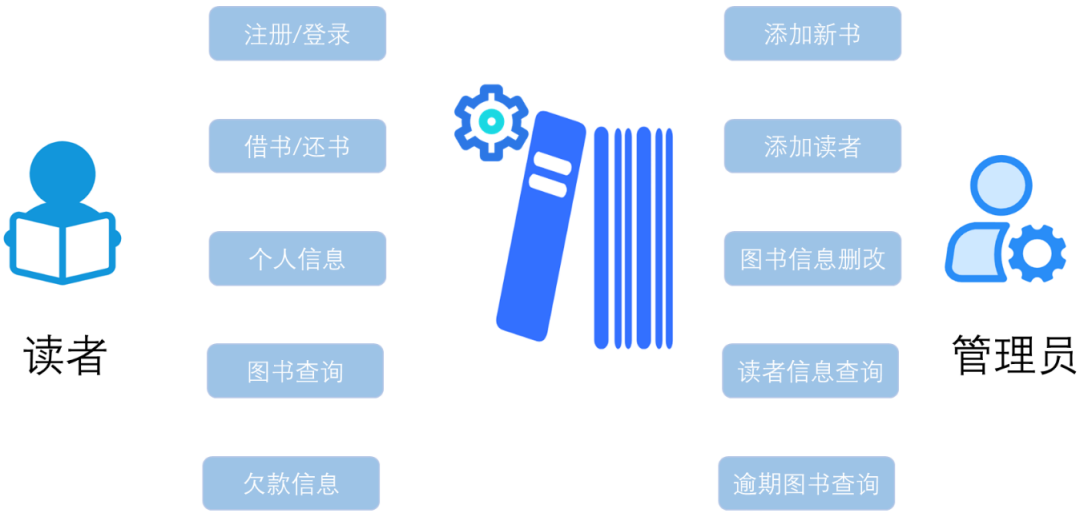
1. **用户中心设计**: 系统以用户需求为中心，围绕管理员和读者的操作习惯和需求进行设计。
2. **数据驱动**: 系统功能实现基于数据库中的数据，通过数据的增删改查（CRUD）操作来实现业务逻辑。
3. **接口交互**: 前后端通过API接口进行数据交互，实现数据的同步和用户界面的动态更新。

## 3.2 主要方法与技术


1. **HTML/CSS/JavaScript**:
    - 用于构建用户界面，提供交互式网页体验。
2. **Flask**:
    - 作为后端框架，处理HTTP请求、业务逻辑和服务器端的渲染。
3. **Pyodbc**:
    - 作为Python与数据库之间的桥梁，实现数据库的连接和操作。
4. **数据库设计**:
    - 使用ER图和物理数据模型来设计数据库结构，确保数据的完整性和一致性。
5. **版本控制**:
    - 使用Git等版本控制系统，进行代码的版本管理和团队协作。

# 4. 实现项目的过程与步骤

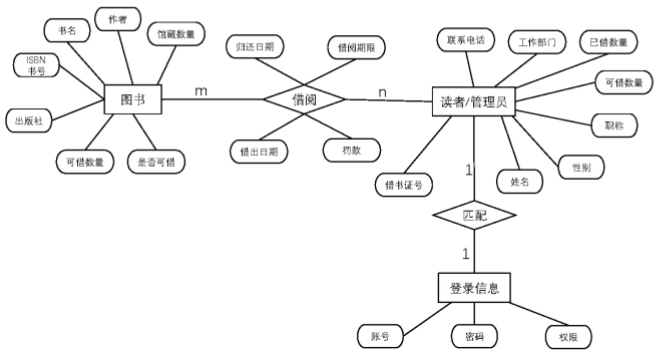
## 4.1 项目总体设计图




## 4.2 概要设计

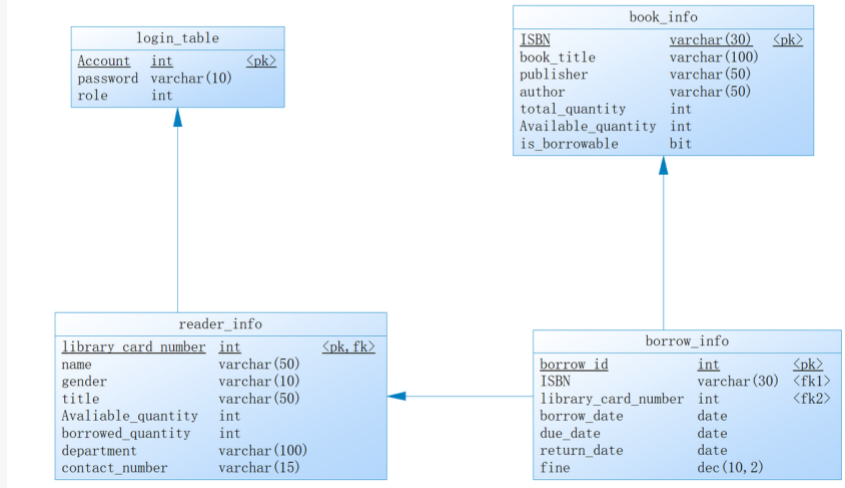
ER图

**ER图**




该系统的处理流程和数据数据流如下图所示

**物理数据图**



## 4.3 实现的子系统功能

### 管理员模块（admin.py）

- **add_reader函数**

  **功能：** 添加读者信息

  **参数：** library_card_number, name=None, gender=None, title=None, available_quantity=10, borrowed_quantity=0, department=None, contact_number=None

  调用此函数时library_card_number一定要给出，其余可以省略。

  其中available_quantity(可借数量)默认为10，borrowed_quantity(已借书数量)默认为0、

  **返回值：**

  - 成功

    ```
    {
        "success":True,
        "data":{
                "library_card_number": library_card_number,
                "name": name,
                "gender": gender,
                "title": title,
                "available_quantity": available_quantity,
                "borrowed_quantity": borrowed_quantity,
                "department": department,
                "contact_number": contact_number
                }
    }
    ```

  - 失败

    ```
    except pyodbc.DatabaseError as e:
        ...
        return error(301, '添加读者失败: ' + str(e))
    except Exception as e:
        return error(401, "错误: " + str(e))
    ```

- delete_reader函数

  功能：删除指定读者信息

  参数：library_card_number

  返回值：

  - 成功

    ```
    {
        "success":True,
        "data":{"library_card_number":library_card_number}
    }
    ```

  - 失败

```
except pyodbc.DatabaseError as e:
    ...
    return error(301,'删除读者信息失败:' + str(e))
except Exception as e:
    return error(401,"错误"+str(e))
```

- print_all_reader_info函数

  **功能**：打印整个reader_info表格

  **参数**：无

  **返回值**：

  - 成功

    ```
    {
        "success":True,
        "data":{"readers":[数组元素]}
    }
    # 数组元素
    {
    "library_card_number": library_card_number,
    "name": name,
    "gender": gender,
    "title": title,
    "available_quantity": available_quantity,
    "borrowed_quantity": borrowed_quantity,
    "department": department,
    "contact_number": contact_number
    }
    ```

    其中all_reader_info是一个字典类型，内容为reader_info表格

  - 失败

    ```
    except pyodbc.DatabaseError as e:
        ...
        return error(301,'删除读者信息失败:' + str(e))
    except Exception as e:
        return error(401,"错误"+str(e))
    ```

- get_reader_info函数

  **功能**：打印指定读者的信息

  **参数**：library_card_number

  **返回值**：

  - 成功

```
{
    "success":True,
    "data":{
            "library_card_number": library_card_number,
            "name": name,
            "gender": gender,
            "title": title,
            "available_quantity": available_quantity,
            "borrowed_quantity": borrowed_quantity,
            "department": department,
            "contact_number": contact_number
            }
}
```

其中reader_info为一个字典类型，内容为借书证号为library_card_number的读者信息

- 失败

```
    return error(20, "未找到该借书证号对应的读者信息")
except pyodbc.DatabaseError as e:
    ...
    return error(301,'删除读者信息失败:' + str(e))
except Exception as e:
    return error(401,"错误"+str(e))
```

- get_overdue_books函数

  **功能**：查询所有到期未归还的图书信息

  **参数**：无

  **返回值**：

  - 成功

```
{
    "success":True,
    "data":{"overdue_books":[数组元素]}
}
# 数组元素
{
"borrow_id": borrow_id,
"library_card_number": library_card_number,
"ISBN": ISBN,
"borrow_date": borrow_date,
"due_date": due_date,
"return_date":return_date
}
```

其中overdue_books_info是一个字典类型，内容为所有到期未归还的图书信息

- 失败

```
except pyodbc.DatabaseError as e:
    ...
    return error(301,'删除读者信息失败:' + str(e))
except Exception as e:
    return error(401,"错误"+str(e))
```

- get_reader_fines函数

  **功能**：查询所有读者的欠款状况

  **参数**：无

  **返回值**：

  - 成功

    ```
    {
        "success":True,
        "data":{"readers_fines":[数组元素]}
    }
    # 数组元素
    {
    "library_card_number": library_card_number,
    "name": name,
    "total_fine": float(total_fine) if total_fine is not None else 0.0,

    }
    ```

    其中reader_fines_info是一个字典类型，内容为读者欠费状况

  - 失败

    ```
    except pyodbc.DatabaseError as e:
        ...
        return error(301,'删除读者信息失败:' + str(e))
    except Exception as e:
        return error(401,"错误"+str(e))
    ```

## 管理员测试模块（test_admin.py）

- 运行test_admin_reader()的参考结果

```
Initial reader info:
Connection successful
{'library_card_number': 2, 'name': '张三', 'gender': '男', 'title': '学生', 'available_quantit
y': 10, 'borrowed_quantity': 0, 'department': '计算机科学与技术', 'contact_number': '123456789
01'}

Adding a new reader:
Connection successful
{'library_card_number': 1, 'name': '张三', 'gender': '男', 'title': '教授', 'available_quantit
y': 10, 'borrowed_quantity': 0, 'department': None, 'contact_number': '12345678901'}

Reader info after adding a new reader:
Connection successful
{'library_card_number': 1, 'name': '张三', 'gender': '男', 'title': '教授', 'available_quantit
y': 10, 'borrowed_quantity': 0, 'department': None, 'contact_number': '12345678901'}
{'library_card_number': 2, 'name': '张三', 'gender': '男', 'title': '学生', 'available_quantit
y': 10, 'borrowed_quantity': 0, 'department': '计算机科学与技术', 'contact_number': '123456789
01'}

Updating reader info:
Connection successful
读者信息更新成功
{'library_card_number': 1, 'name': '李四', 'gender': None, 'title': '副教授', 'available_quant
ity': 10, 'borrowed_quantity': 0, 'department': None, 'contact_number': '09876543210'}

Reader info after updating:
Connection successful
{'library_card_number': 1, 'name': '李四', 'gender': '男', 'title': '副教授', 'available_quant
ity': 10, 'borrowed_quantity': 0, 'department': None, 'contact_number': '09876543210'}
{'library_card_number': 2, 'name': '张三', 'gender': '男', 'title': '学生', 'available_quantit
y': 10, 'borrowed_quantity': 0, 'department': '计算机科学与技术', 'contact_number': '123456789
01'}


Getting reader info for library_card_number=1:
Connection successful
{'library_card_number': 1, 'name': '李四', 'gender': '男', 'title': '副教授', 'available_quant
ity': 10, 'borrowed_quantity': 0, 'department': None, 'contact_number': '09876543210'}

Getting reader info for library_card_number=200 (non-existent):
Connection successful
{'success': False, 'status': 20, 'message': '未找到该借书证号对应的读者信息'}

Deleting reader with library_card_number=1:
Connection successful
{'success': True, 'data': {'library_card_number': 1}}

Reader info after deleting the reader:
Connection successful
{'library_card_number': 2, 'name': '张三', 'gender': '男', 'title': '学生', 'available_quantit
y': 10, 'borrowed_quantity': 0, 'department': '计算机科学与技术', 'contact_number': '123456789
01'}
```

添加读者信息

验证

更新读者信息

验证

查询指定读者信息(存在)

查询指定读者信息(不存在)

删除读者信息

验证

- 运行test_admin_book_borrow()参考结果

首先插入几条示例借阅信息(已更新在github仓库中的sql文件中)

```sql
-- 插入借阅信息
INSERT INTO borrow_info (library_card_number, ISBN, borrow_date, due_date,
return_date, fine)
VALUES
(2, '978-3-16-148410-0', '2023-01-01', '2023-01-15', '2023-01-14', 0.00),--过期已
还
(2, '978-0-12-345678-9', '2023-02-01', '2023-02-15', NULL, 12.00),--过期未还
(2, '978-1-23-456789-0', '2023-01-01', '2025-01-15', NULL, 0.00) --未过期且未还
```

```
Connection successful
{'borrow_id': 4, 'library_card_number': 2, 'ISBN': '978-0-12-345678-9', 'borrow_date': '2023-0
2-01', 'due_date': '2023-02-15', 'return_date': None}
Connection successful
{'library_card_number': 2, 'name': '张三', 'total_fine': 12.0}
```

只有过期未还的借阅信息需要被打印出来，结果相符。

## 图书查询模块（book_search.py）

- **search_books 函数**

**功能：** 搜索图书，搜索关键词可以是书籍的ISBN号、作者、出版社和书名

**参数：** 输入字符串，可以是 ISBN，author，publisher 或者 book_title

**返回值：**

- 成功

```python
for book in overdue_books:
    overdue_books_data.append({
        "borrow_id": book.borrow_id,
        "library_card_number": book.library_card_number,
        "ISBN": book.ISBN,
        "borrow_date": book.borrow_date,
        "due_date": book.due_date,
        "overdue_days": book.overdue_days,
        "fine": book.fine
     })
return success(books)
```

- 失败

```python
if not results:
    return error(404, "图书信息未找到")

except pyodbc.DatabaseError as e:
    ...
    return error(301, '查询失败: ' + str(e))
except Exception as e:
    return error(401, "错误: " + str(e))
```

**运行测试：**

查找成功：

{'success': True, 'data': [{'ISBN': '978-3-16-148410-0', 'book_title': '图书1', 'publisher': '出版社A', 'author': '作者A', 'total_quantity': 5, 'available_quantity': 5, 'is_borrowable': True}]}

没有图书记录：{'success': False, 'status': 404, 'message': '图书信息未找到'}

## 逾期图书查询模块（overdue_books.py ）

- **get_overdue_books 函数**

  **功能**：查询未归还图书信息

  **参数**： library_card_number, name=None, gender=None, title=None, available_quantity=10, borrowed_quantity=0, department=None, contact_number=None

  调用此函数时library_card_number一定要给出，其余可以省略。

  其中available_quantity(可借数量)默认为10，borrowed_quantity(已借书数量)默认为0、

  **返回值**：

  - 成功

```
for row in results:
    books.append({
        "ISBN": row.ISBN,
        "book_title": row.book_title,
        "publisher": row.publisher,
        "author": row.author,
         "total_quantity": row.total_quantity,
        "available_quantity": row.available_quantity,
        "is_borrowable": row.is_borrowable
})
return success(books)
```

  - 失败

```
except pyodbc.DatabaseError as e:
    ...
    return error(301, '查询失败: ' + str(e))
except Exception as e:
    return error(401, "错误: " + str(e))
```

  **运行测试**：

  查找成功：

{'success': True, 'data': [{'borrow_id': 2, 'library_card_number': 2, 'ISBN': '978-0-12-345678-9', 'borrow_date': '2023-02-01', 'due_date': '2023-02-15', 'overdue_days': 490, 'fine': Decimal('12.00')}]}

## 读者信息查询模块（reader_query.py）

- **get_reader_info 函数**

  **功能**： 查询读者个人信息和读者借书信息

  **参数**： 输入 library_card_number

  **返回值**：

- 成功

```
reader_data = {
    "library_card_number": reader_info.library_card_number,
    "name": reader_info.name,
    "gender": reader_info.gender,
    "title": reader_info.title,
    "available_quantity": reader_info.available_quantity,
    "borrowed_quantity": reader_info.borrowed_quantity,
    "department": reader_info.department,
    "contact_number": reader_info.contact_number
    }
...
for book in borrowed_books:
    borrowed_books_data.append({
        "borrow_id":book.borrow_id,
        "ISBN": book.book_ISBN,
        "book_title": book.book_title,
        "author": book.author,
        "borrow_date": book.borrow_date,
        "due_date": book.due_date,
        "return_date": book.return_date,
        "fine": book.fine
     })

return success({
    "reader_info": reader_data,
    "borrowed_books": borrowed_books_data,
})
```

- 失败

```
if not reader_info:
    return error(404, "读者信息未找到")

except pyodbc.DatabaseError as e:
        cursor.rollback()
        cursor.close()
        cnxn.close()
        return error(301, '查询失败: ' + str(e))
except Exception as e:
        return error(401, "错误: " + str(e))
```

**运行测试**:

查询成功:

{'success': True, 'data': {'reader_info': {'library_card_number': 2, 'name': '张三', 'gender': '男', 'title': '学生', 'available_quantity': 10, 'borrowed_quantity': 0, 'department': '计算机科学与技术', 'contact_number': '12345678901'}, 'borrowed_books': [{'ISBN': '978-0-12-345678-9', 'book_title': '图书2', 'author': '作者B', 'borrow_date': '2023-02-01', 'due_date': '2023-02-15', 'return_date': None, 'fine': Decimal('12.00')}, {'ISBN': '978-1-23-456789-0', 'book_title': '图书3', 'author': '作者C', 'borrow_date': '2023-01-01', 'due_date': '2025-01-15', 'return_date': None, 'fine': Decimal('0.00')}]}}

没有该读者记录: {'success': False, 'status': 404, 'message': '读者信息未找到'}

## 登录注册模块（login.py）

- **register 函数**

  **功能**：进行注册

  **参数**：name->要注册的姓名，password->要注册的密码，title->职称，phone_number->电话号码，department->系别，gender->性别，is_root->是否为管理员

  **返回值**：

  - 成功

```
{
  "success":True,
  "data":account
}
```

  - 失败（代码与说明）
    - 1 :姓名为空错误
    - 2 :密码为空错误
    - 3 :姓名长度过长
    - 4 :密码长度过长
    - 10:权限设置错误
    - 20:身份长度过长
    - 21:性别不为男也不为女
    - 22:系别长度过长
    - 23:电话长度过长
    - 301:数据库操作错误
    - 401:意外错误
- **login函数**

  **功能**：登录

  **参数**：account->账号，password->密码

  **返回值**：

  - 成功

```
{
  "success":True,
  "data":{
    "role":role,
    "name":name
  }
}
```

  - 失败
    - 1 ： 用户名或密码错误
    - 301：数据库部分错误
    - 401：意外错误
    -

## 借书还书模块（borrowbook.py）

- **borrow_book函数**

  **功能**：借书

  **参数**：library_card_number->借书号/账号,isbn->借书的isbn号,due_date->到期时间,begin_date->起始时间，默认为当天

  **返回值**：

  - 成功

```
{
  "success":True,
  "data":{
          "library_card_number":library_card_number,
          "isbn":isbn,
          "begin_date":begin_date,
          "due_date":due_date
      }
}
```

  - 失败
    - 1： isbn不存在错误
    - 2： library_card_number不存在错误
    - 3： 借书超过上限错误
    - 4： 有未交罚金或有未在时间内归还的图书
    - 5： 重复借同一本书
    - 6： 书籍不可借
    - 301：数据库操作错误
    - 401：意外错误

- **return_book函数**

  **功能**：还书，并提醒快过期和已经过期的书单

  **参数**：library_card_number->借书号/账号,isbn->借书的isbn号

  **返回值**：

  - 成功

```
{
  "success":True,
  "data":[
    [isbn,due_date,fine],
    [isbn,due_date,fine],
    ...
  ]
}
```

  - 失败
    - 1： 借书记录不存在错误
    - 301：数据库操作错误
    - 401：意外错误

# 5. 主要成果及小结

## 5.1 个人遇到的困难与获得的主要成果

### 5.1.2 遇到的困难

- 对于项目的整体把握比较模糊，导致上手很慢

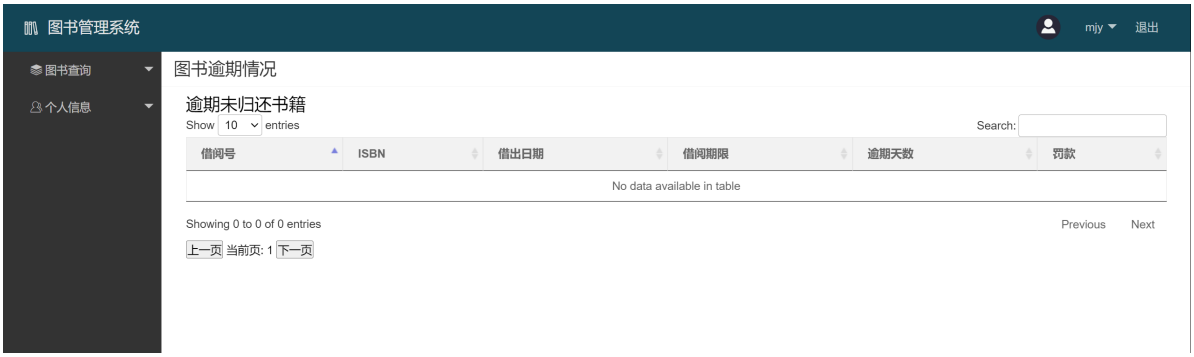
对flask路由，html，数据库等各单个模块的掌握不熟练；对结合这几个工具综合运用的项目比较磕磕绊绊。

- 对于如何在login.py文件以外的地方获取当前用户的 account 的实现有疑问，尝试了几种方法未能成功，最终在队友的提醒下用了传参的方法。
- 期末已经结束，要克服内心的抗拒和外界的诱惑静下来做期末项目，有点难

### 5.1.3 主要的成果

实现了读者个人信息查询，包括读者个人信息和读者借书信息：


实现了读者图书借阅逾期情况的查询：




实现了图书查询功能（支持模糊查询）：


## 5.2 结果分析与个人小结

### 结果分析

经过为期15天的紧张开发，我们的简易图书管理系统项目取得了以下成果：

1. **功能完善**：系统不仅实现了基本的图书管理、用户管理、借阅管理等功能，还额外提供了图书查询、个人信息查看、逾期图书提醒等便捷功能。
2. **技术应用**：项目中成功应用了HTML/CSS/JavaScript构建前端界面，使用Flask框架处理后端逻辑，并通过Pyodbc与数据库进行交互。此外，ER图和物理数据模型的运用确保了数据库设计的合理性。
3. **团队协作**：在开发过程中，团队成员之间通过飞书等工具进行了有效的沟通和协作，共同解决了项目中遇到的技术难题。
4. **问题解决**：虽然在开发过程中遇到了对整体项目把握不足、技术模块掌握不熟练等问题，但通过不断学习和团队合作，最终克服了这些困难。

### 个人小结

在这次简易图书管理系统的项目，我获得了宝贵的经验，也对团队合作有了更深刻的理解。

首先，我要感谢我的队友们。在整个项目过程中，他们不仅展现了出色的专业能力，还在我遇到困难时耐心地提供指导。他们的技术支持和心理慰藉让我深受感动。

在技术方面，通过这次项目实践，加深了我对Web开发的理解。从前端的页面设计到后端的逻辑处理，再到数据库的交互，每一个环节都让我对技术有了更深入的掌握。虽然在学习Flask、数据库操作等技术时遇到了挑战，但通过不断尝试和改进，我逐步解决了这些问题，也在这个过程中提升了自己的技术能力。

这次项目也锻炼了我的时间管理和自我控制能力。我学会了如何合理安排时间，集中精力完成项目任务。

# 参考文献

[1]. Flask. (n.d.). Flask Documentation. Retrieved June 29, 2024, from https://flask.palletsprojects.com/

[2]. Python Software Foundation. (n.d.). Python documentation. Retrieved June 29, 2024, from https://docs.python.org/3/