



兰州大学信息科学与工程学院

School of Information Science & Engineering , Lanzhou University

2023 秋季学期  
Python Django 项目

设  
计  
报  
告

队员：童 傅 白

队名：来我们队

指导老师：詹建

时间：2023 年 12 月



游戏化学习软件——“知趣”

# 目录

第一部分 成员分工 .....	1
1. 团队成员 .....	1
2. 任务分工 .....	1
第二部分——软件总体设计 .....	2
1. 设计核心 .....	2
2. 项目架构 .....	3
3. 前后端数据交互 .....	3
4. 数据库结构 .....	4
5. 功能模块设计简要说明 .....	5
第三部分 软件主要界面与功能说明 .....	5
1. 用户注册与登录 .....	5
1.1 用户注册功能 .....	5
1.2 用户登录功能 .....	6
2. 软件主界面 .....	7
2.1 主界面说明 .....	7
2.2 每日任务/待办事项 .....	7
2.3 进入 NPC 交流界面 .....	9
3. NPC 对话界面 .....	10
4. 装备界面 .....	10
5. 每日任务/待办可视化界面 .....	12
5.1 界面说明 .....	12
5.2 每日任务展示栏 .....	12
5.3 待办事项展示栏 .....	13
6. 数据可视化界面 .....	14
6.1 能力图谱 .....	14
6.2 事件总览 .....	14
6.3 待办完成情况 .....	15

6.4 每日事项统计 .....	15
7. 修改角色信息界面 .....	16
8. 游戏后台管理系统 .....	16
第四部分 特色功能说明 .....	18
1. 使用 Inworld studio 中的 AI .....	18
1.1 inworld studio 简介 .....	18
1.2 character 生成过程 .....	18
1.3 AI 调试 .....	19
2. 引入 inworld AI 作为游戏化学习软件的对话 NPC .....	20
3. 学习和游戏的巧妙结合 .....	21
4. 大量的可视化，帮助用户完成学习数据分析 .....	22
5. 未来发展规划与展望 .....	22

# 第一部分 成员分工

## 1. 团队成员

队名：来我们队

姓名	学号	专业	职务
童		计算机科学与技术	队长
傅		计算机科学与技术	队员
白		计算机科学与技术	队员

## 2. 任务分工



组长童主要负责项目进度管理及主要的后端编写工作。前期包括在项目开始时提出软件初步设想并丰满设想，形成软件结构，搭建 django 框架，根据需求设计数据库结构。后端工作包括创建注册 APP（TestModel），编写 APP 内的 models.py 并迁移到数据库形成 table，编写 urls.py，views.py 实现后端数据向前端的传递，以及对 admin.py 的所有编写工作，实现并完善软件的后台管理系统。同时童还负责软件中的数据可视化功能，任务显示功能的 6 个前端页面及其前后端数据交互的实现。

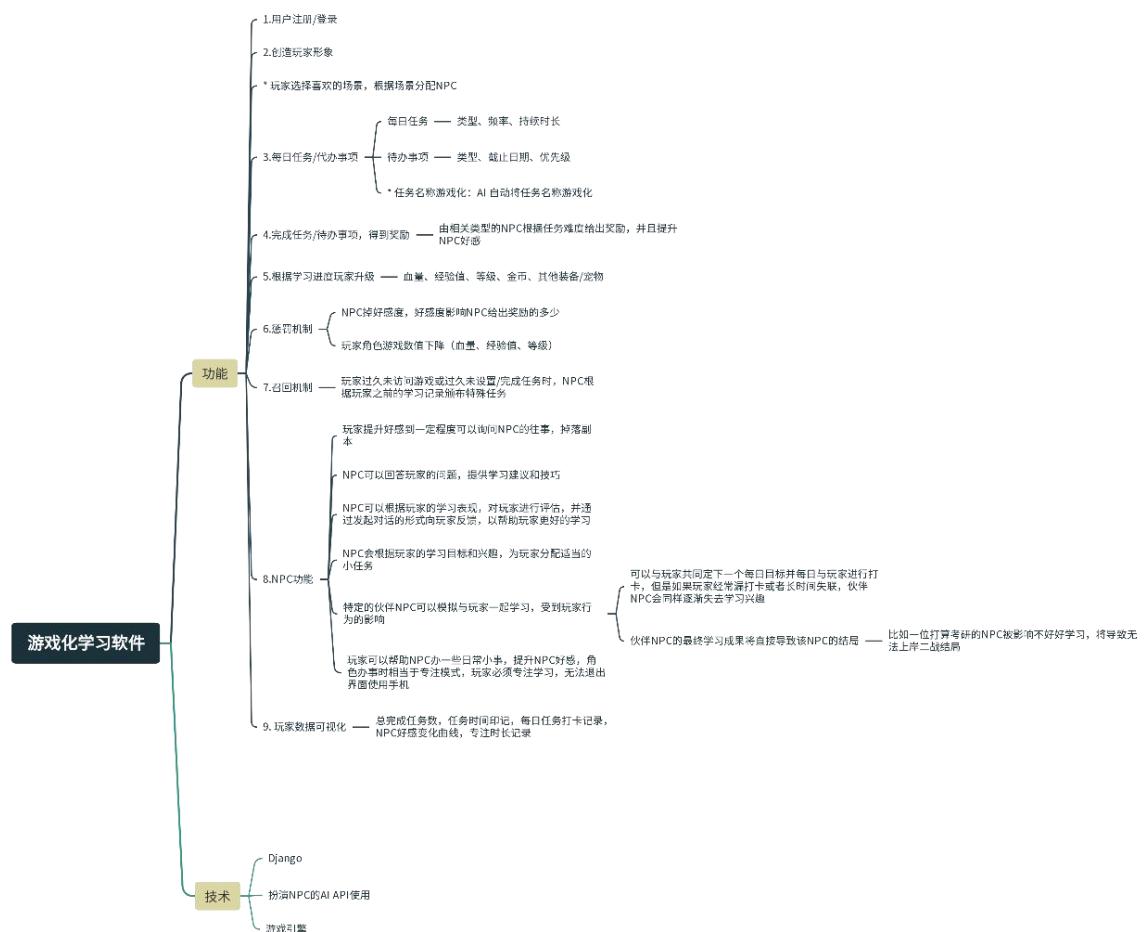
组员傅和白主要负责前端的编写和前后端交互。傅同学主要编写软件登陆/注册页面和游戏化学习软件主页，并完成了 NPC 交流界面的美化，白同学完成了游戏装备展示界面以及 NPC 交流界面。前端主要是 html 网页文件的编写与美化，前后端数据交互则是通过 Django 的 ORM 编写代码完成从后端到前端数据的导入，并使用表单（form）技术将前端数据传入后端。

由于游戏化软件的功能多样，数据繁杂，整个项目工程量十分庞大，软件界面分一级界面、二级界面，最终软件有十几个页面（所以最后后期来不及美化页面了）。

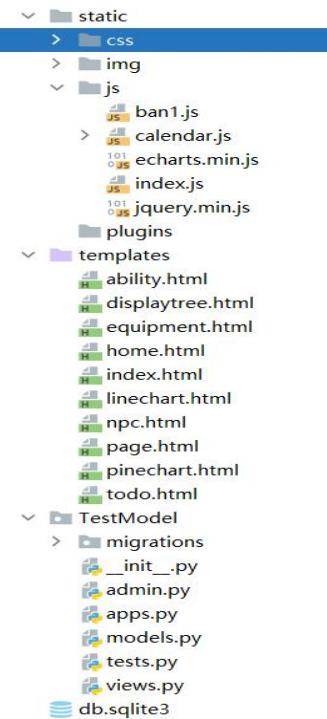
## 第二部分——软件总体设计

### 1. 设计核心

我们的游戏化学习软件核心目标是使用游戏化的元素和机制来激励用户进行学习。为了实现这一目标，我们需要考虑如何将学习内容、游戏机制有效地整合在一起。



## 2. 项目架构



## 3. 前后端数据交互

### 3.1 后端-->前端

用户通过浏览器向服务器发起一个请求(request)，这个请求会去访问视图函数（views.py 中的函数）：如果不涉及到数据调用，那么视图函数直接返回一个模板也就是一个网页给用户；如果涉及到数据调用，那么视图函数调用模型，模型去数据库查找数据，然后逐级返回。

视图函数把返回的数据填充到模板的占位符中，网页经过渲染，最后返回网页给用户。

### 3.2 前端-->后端

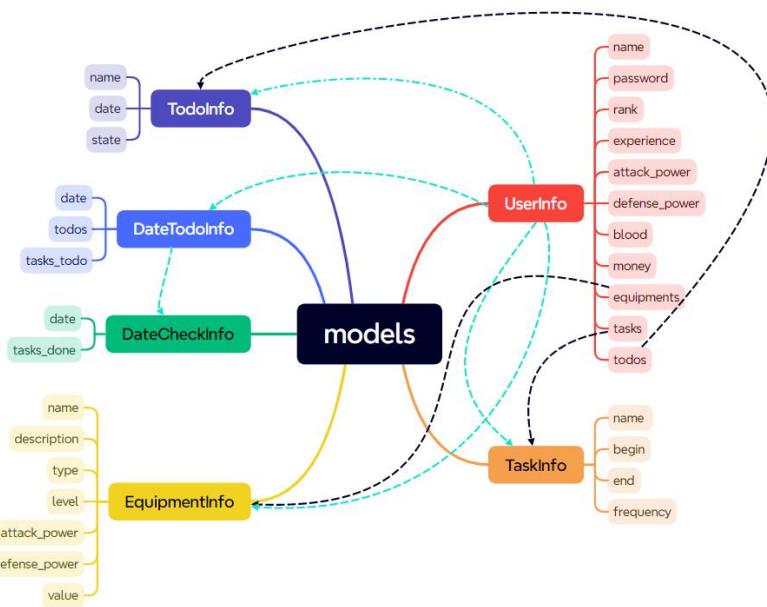
使用 HTML 的<form>元素和 POST 方法

```
<form method="post" action="/infochange/">
    {% csrf_token %}
    <label for="ID">      ID: </label>
    <a>{{ user.id }}</a>
    <br><br>
    <label for="uname">    用户名:</label>
    <input type="text" id="uname" name="uname" value="{{ user.name }}" required><br>
    <br>
    <label for="password">  密码:</label>
    <input type="password" id="password" name="password" value="{{ user.password }}" required><br>
    <br>
    <div style="...">
        <input type="submit" value="修改" style="...">
    </div>
</form>
```

## 使用 fetch API 的 POST 方法

```
fetch("/jianqli/", {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    'X-CSRFToken': $('input[name=csrfmiddlewaretoken]').val(), // Django CSRF token
  },
  body: JSON.stringify({ jl: jl, rN: rN, task_id:0, todo_id:{item.id} }),
})
.then(response => response.json()) // 解析响应为JSON, 如果需要的话
.then(data => { /* 处理响应数据 */ })
.catch((error) => { /* 处理错误 */ });
```

## 4. 数据库结构



```
4   # Create your models here.
5   class TaskInfo(models.Model):...
14
15   class TodoInfo(models.Model):...
22
23
24
25   class EquipmentInfo(models.Model):...
36
37   class UserInfo(models.Model):...
52
53   class DateTodoInfo(models.Model):...
60
61   class DateCheckInfo(models.Model):...
67
```

Models.py 中一共创建了 6 个类，相应的在数据库中自动产生 6 个表，以及它们之间的关联表。

(UserInfo) 用户类是最核心的类，其变量有用户名 name，密码 password，等级 rank，经验值 experience，攻击值 attack\_power，防御值 defense\_power，血量 blood，金币 money，装备 equipements，每日任务 tasks，和待办事项 todos，其中由于一个用户可以有多件装备，多个任务和多个待办，所以变量 equipements，tasks，和 todos 的数据类型都是 models.ManyToManyField。

```
class UserInfo(models.Model):
    name = models.CharField('名称', max_length=30)
    password = models.CharField('密码', max_length=20)
    rank = models.IntegerField('等级', default=1)
    experience = models.IntegerField('经验值', default=0)
    attack_power = models.IntegerField('攻击值', default=0)
    defense_power = models.IntegerField('防御值', default=0)
    blood = models.IntegerField('血量', default=100)
    money = models.IntegerField('金币', default=0)
    equipements = models.ManyToManyField(EquipmentInfo, '设备', blank=True)
    tasks = models.ManyToManyField(TaskInfo, '任务', blank=True)
    todos = models.ManyToManyField(TodoInfo, '待办', blank=True)
    def __str__(self):
        # 自定义字符串表示形式
        return self.name
```

综上，数据库中的表有核心的用户列表，以及与其关联的装备表，任务表，待办事项表等等。这几个表的功能都是表如其名，非常的 pythonic。

## 5. 功能模块设计简要说明

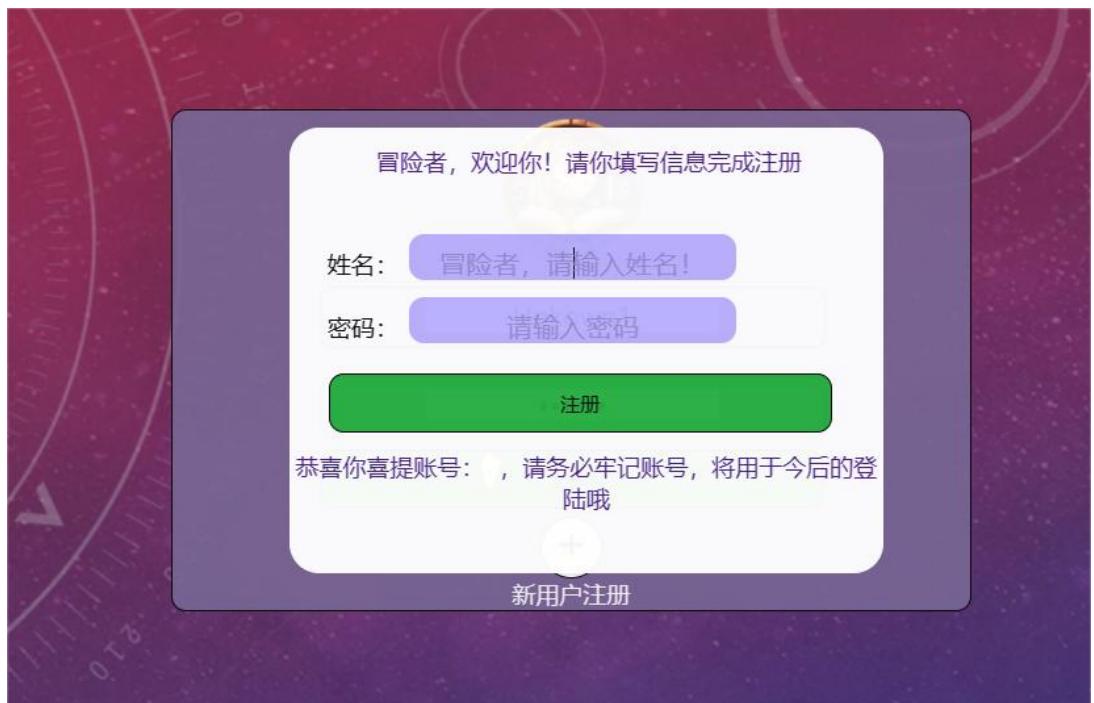
我们实现的游戏化学习软件的功能有：注册与登陆、创建和展示任务及待办事项、和软件中的 AI npc 进行聊天，游戏数据可视化，金币\经验值奖励机制，装备系统，此外还有一个游戏后台管理系统。

# 第三部分 软件主要界面与功能说明

## 1. 用户注册与登录

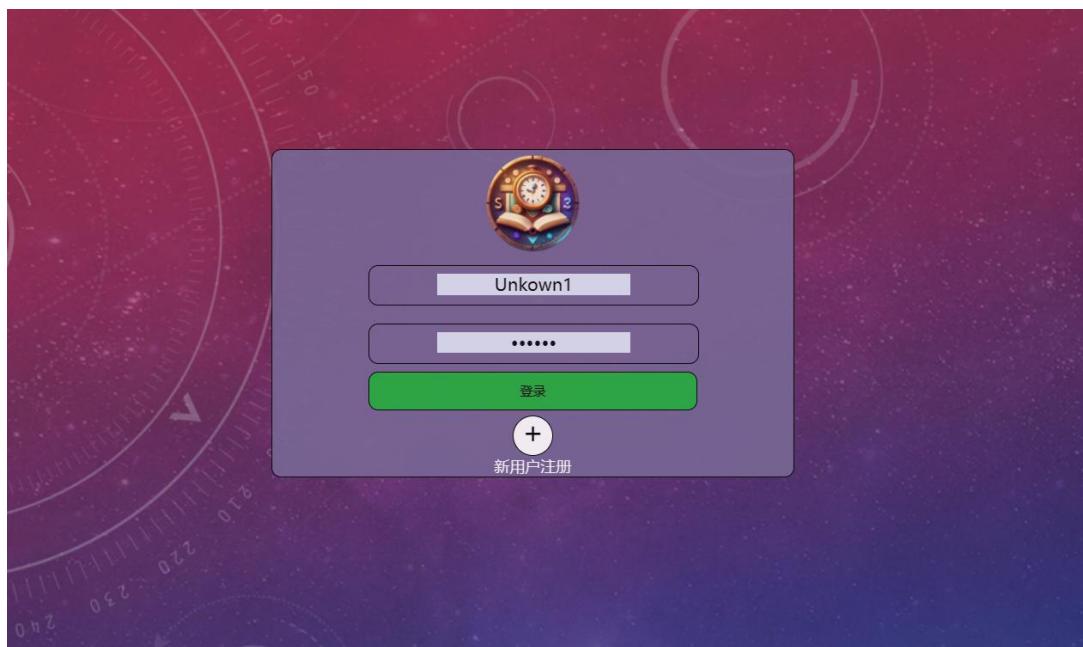
### 1.1 用户注册功能

在登录界面点击“新用户注册”按钮，进入注册界面。用户通过符合规则的用户名和密码的填写即可建立一个专属账号，并返回该账号的 userID 显示给用户。



## 1.2 用户登录功能

玩家在输入正确的用户名和密码之后即可登入主界面。



登录时玩家的用户名和密码将被传递到后端，后端使用 Django 的 session 框架，保存玩家 ID，后续所有需要调入玩家信息的函数都可以通过保存的 ID 来查询到玩家，从而获得玩家信息。

```

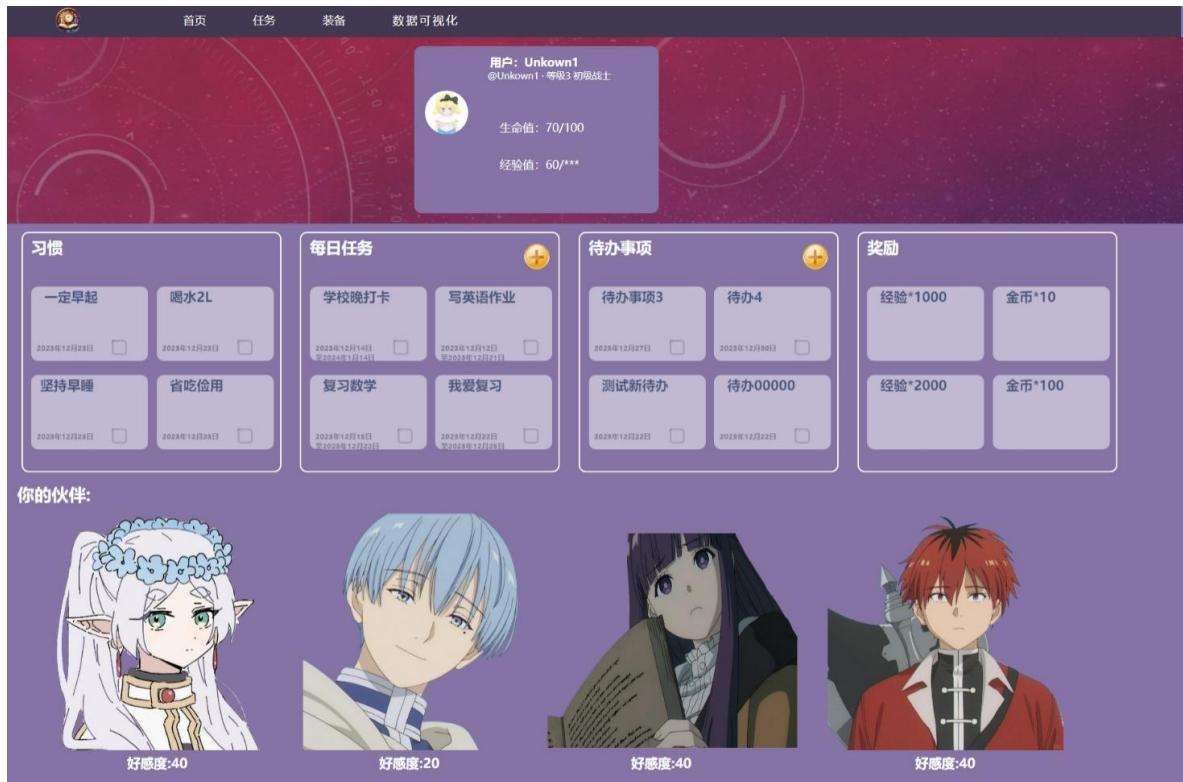
def login_form(request):
    if request.method == 'POST':
        name = request.POST.get('account')
        password = request.POST.get('password')
        obj = models.UserInfo.objects.get(name=name, password=password)
        request.session['userid'] = obj.id # 将用户对象保存到会话中
        datecheck = models.DateCheckInfo.objects.all()
        undo = obj.todos.filter(state=0)
    return render(request, "home.html", {"user":obj, "checkall":datecheck, "undo":undo})

```

## 2. 软件主界面

### 2.1 主界面说明

主界面由导航栏，用户基本信息显示栏，习惯、每日任务、待办事项、奖励四块展示栏，以及 AI NPC 展示栏组成。



### 2.2 每日任务/待办事项

每日任务和待办事项栏将分别显示用户的每日任务与待办事项，在小方框中显示名称和时间，并带有复选框。



在每日任务中勾选相应的复选框表示今日完成该任务的打卡，在待办事项中勾选表示该待办已完成，已完成的待办事项将在下次登录时从待办事项栏中消失（因为该待办已完成）。



同时，我们设置了奖励机制，在勾选每日任务/待办事项后，玩家将得到随机数量的金币/经验值奖励，并通过网页 alert 的形式告知玩家。



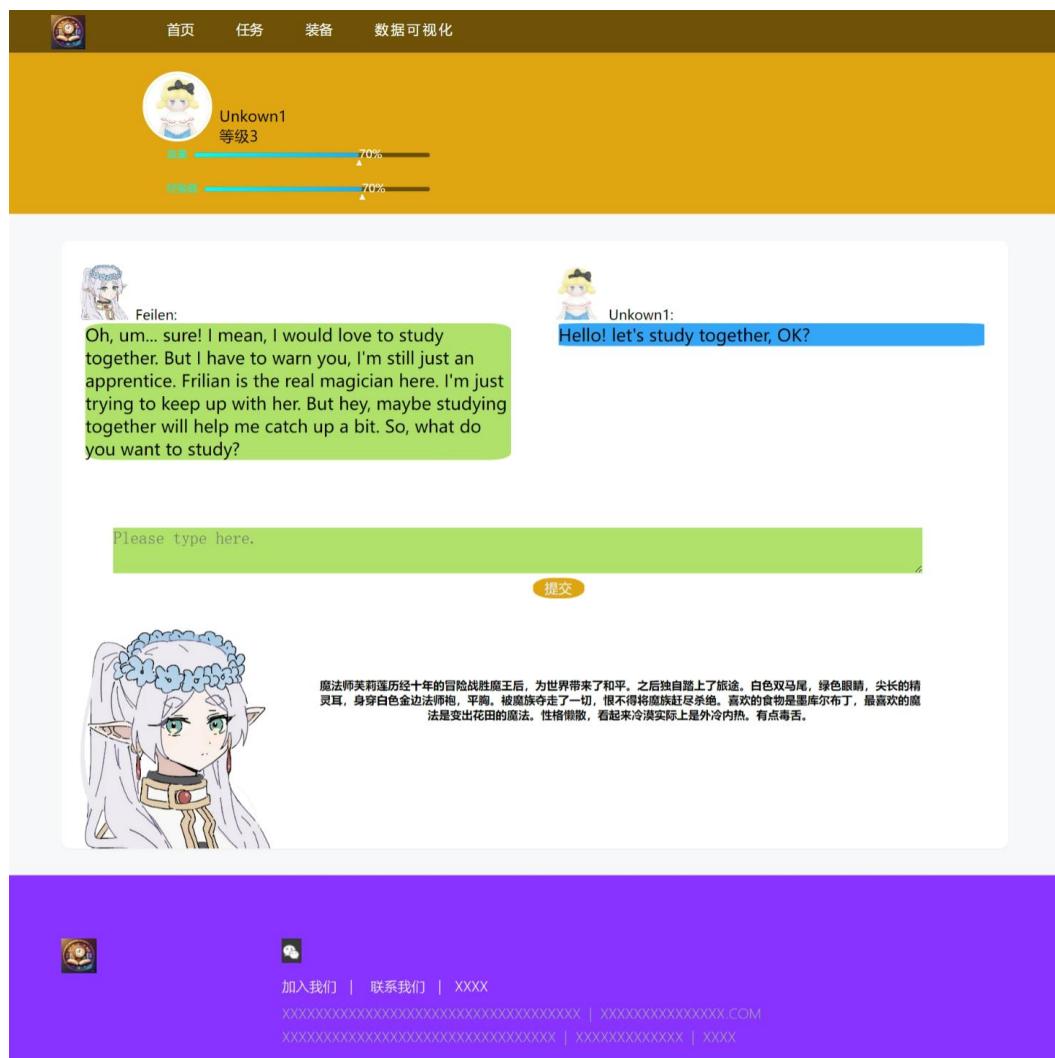
用户还可以通过点击两个模块右上角的“+”号后添加新的每日任务和待办事项。添加每日任务时需输入任务名称，起始时间和终止时间；添加待办事项时需输入待办事项和待办时间。前端通过表单(form)收集这些信息后，传递给后端，后端处理添加入数据库，该用户的每日任务和待办事项成功新增。



### 2.3 进入 NPC 交流界面

玩家点击主界面最下方各个 NPC 的头像，将跳转到相应 NPC 的对话界面。

### 3. NPC 对话界面（此处只展示一个 NPC 的对话界面）

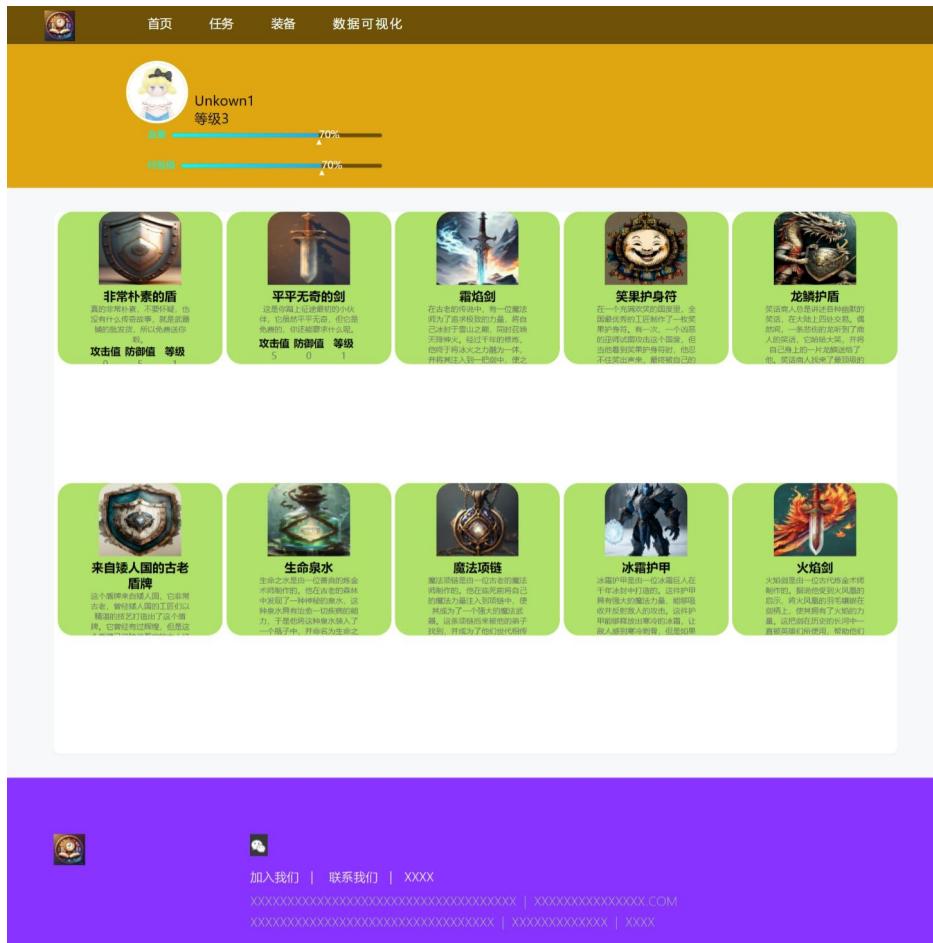


界面由导航栏，玩家数据显示栏，对话栏以及 NPC 介绍栏组成。

玩家在输入框内输入想说的话，点击“发送”按钮，即可与 NPC 对话，NPC 的回复将显示在对话栏的左侧，希望继续对话只需要继续在输入框内输入即可。

### 4. 装备界面

在独立的装备管理界面，玩家可了解不同装备的来源，长相，名称，并可通过金币来购买装备，以提升自身属性，如防御值和攻击值，并可查看改装备是否可以拥有或可购买。



鼠标悬浮时显示该装备的全部属性（包括名称，描述，攻击值，防御值，等级），并且在最后显示一个按钮，如果玩家已拥有该装备则显示“已拥有”，该按钮无效；如果玩家还未曾拥有该装备，则按钮显示“花费{装备价值}金币购买”，点击按钮，如果玩家金币数量足够，将成功购买该装备，并且减少相应数量的金币，金币数量不足则购买失败。



## 5. 每日任务/待办可视化界面

### 5.1 界面说明

界面由导航栏，玩家数据显示栏，可视化选项栏，展示栏组成。

The screenshot displays the user interface for managing daily tasks and pending items. At the top, there's a navigation bar with tabs for 首页 (Home), 任务 (Tasks), 装备 (Equipment), and 数据可视化 (Data Visualization). Below the navigation bar, the player's profile is shown with the name 'Unknown1' and level 3, along with two progress bars both at 70% completion.

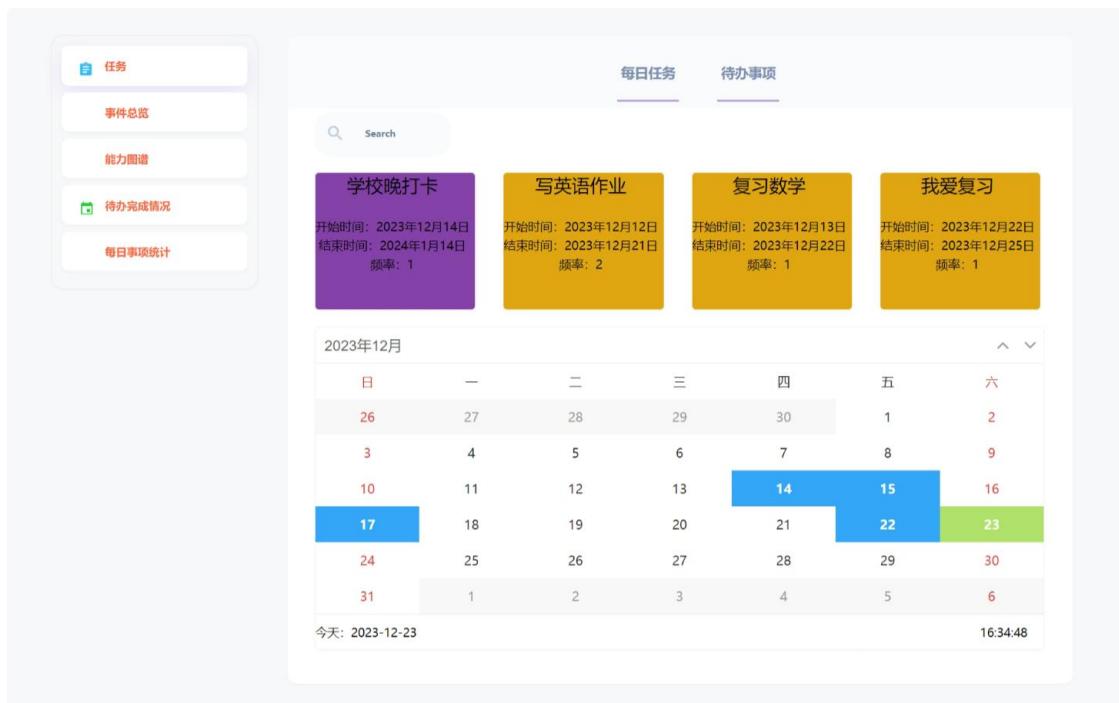
The main content area is divided into several sections:

- 左侧菜单 (Left Sidebar):** Includes '任务' (Tasks), '事件总览' (Event Overview), '能力图谱' (Ability Map), '待办完成情况' (Pending Task Status), and '每日事项统计' (Daily Task Statistics).
- 每日任务 (Daily Tasks):** A section listing four tasks with their start and end dates and frequencies:
  - 学校晚打卡 (School Night Check-in): Start: 2023-12-14, End: 2024-1-14, Frequency: 1
  - 写英语作业 (Write English Homework): Start: 2023-12-12, End: 2023-12-21, Frequency: 2
  - 复习数学 (Review Math): Start: 2023-12-13, End: 2023-12-22, Frequency: 1
  - 我爱复习 (I Love Review): Start: 2023-12-22, End: 2023-12-25, Frequency: 1
- 待办事项 (Pending Tasks):** A section showing a monthly calendar for December 2023. The date 23 is highlighted in green, indicating it's the current date (2023-12-23). Other dates are shown in red. The calendar includes columns for Sunday through Saturday.
- 底部信息 (Bottom Information):** Includes links for '加入我们' (Join Us) and '联系我们' (Contact Us), followed by placeholder text 'XXXXXXXXXXXXXXXXXXXX | XXXXXXXXXXXXXXX.COM' and 'XXXXXXXXXXXXXXXXXXXX | XXXXXXXXXXXXXXX | XXXX'.

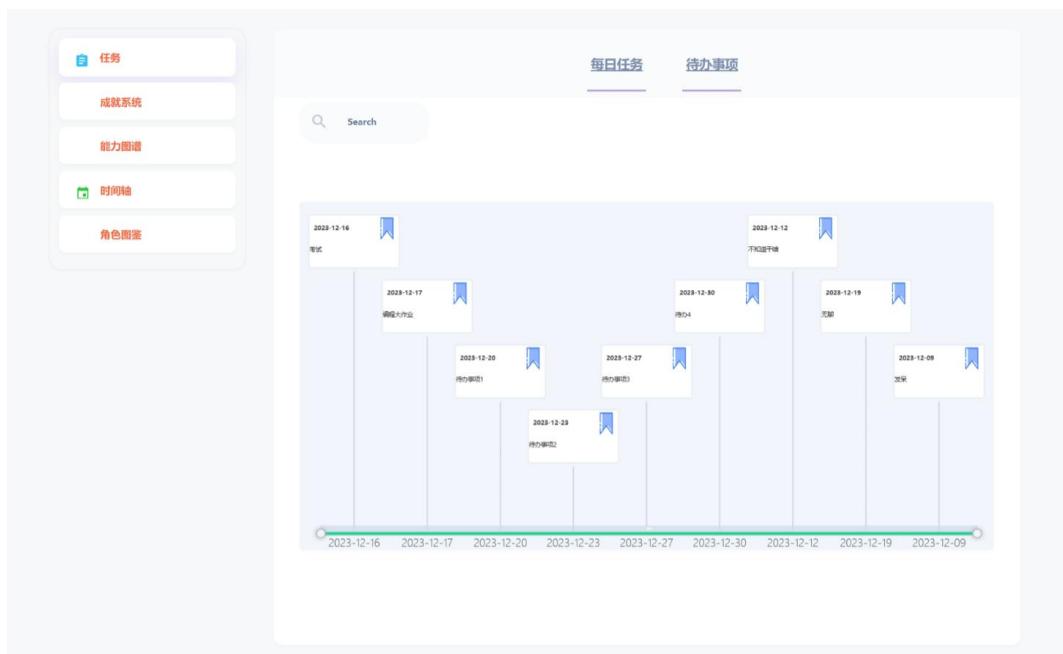
### 5.2 每日任务展示栏

最上方显示四个玩家最新的每日任务，并显示相关信息，点击将变色，同时下方的日历中该任务已打卡的日期会出现蓝色方框，以达成可视化该每日任务打卡情况的目的。

日历部件通过 js 实现，数据由后端导入，可实现上下翻页（选择月份），并且可以实时显示现在的时间（当时的日期将由绿色突出显示）。



### 5.3 待办事项展示栏

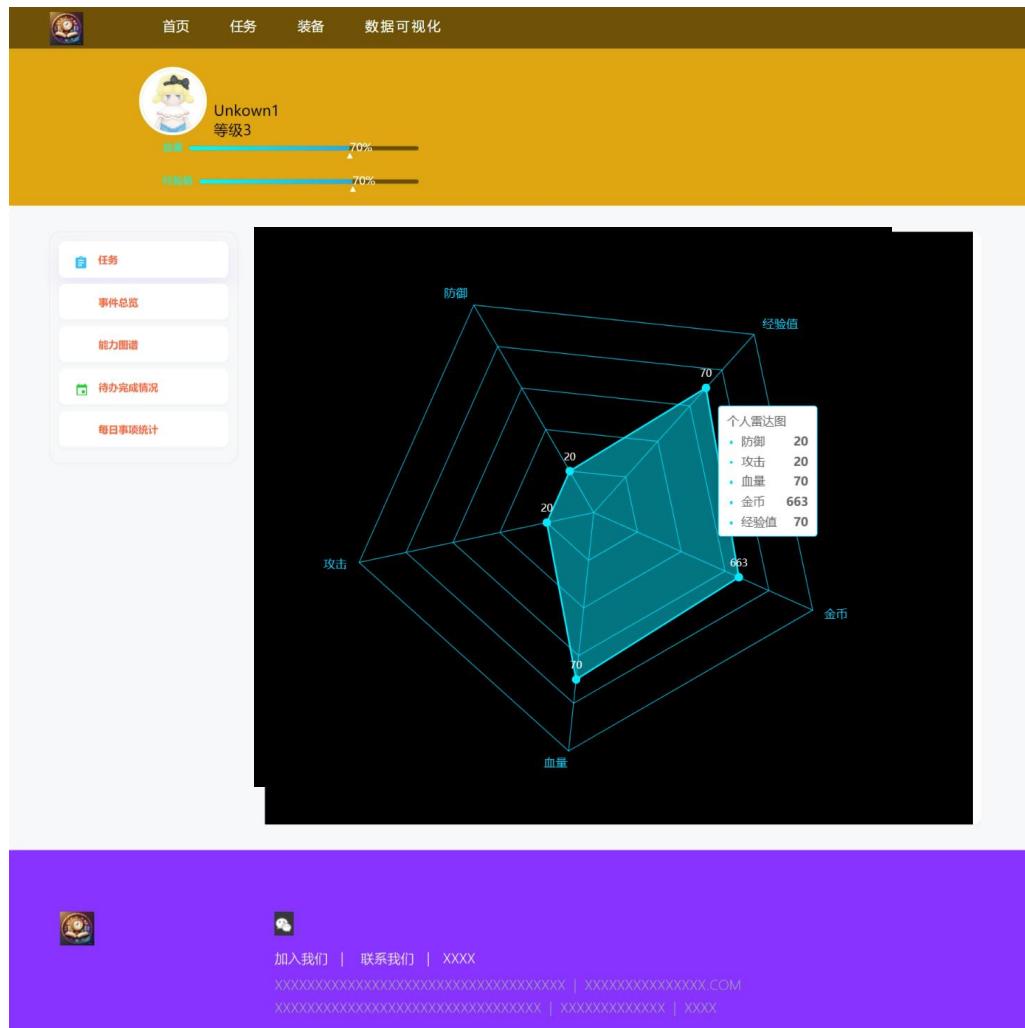


该界面以时间轴的形式展示了该玩家的所有待办事项，清晰明了，可以辅助玩家做出合理的时间规划。

## 6. 数据可视化界面

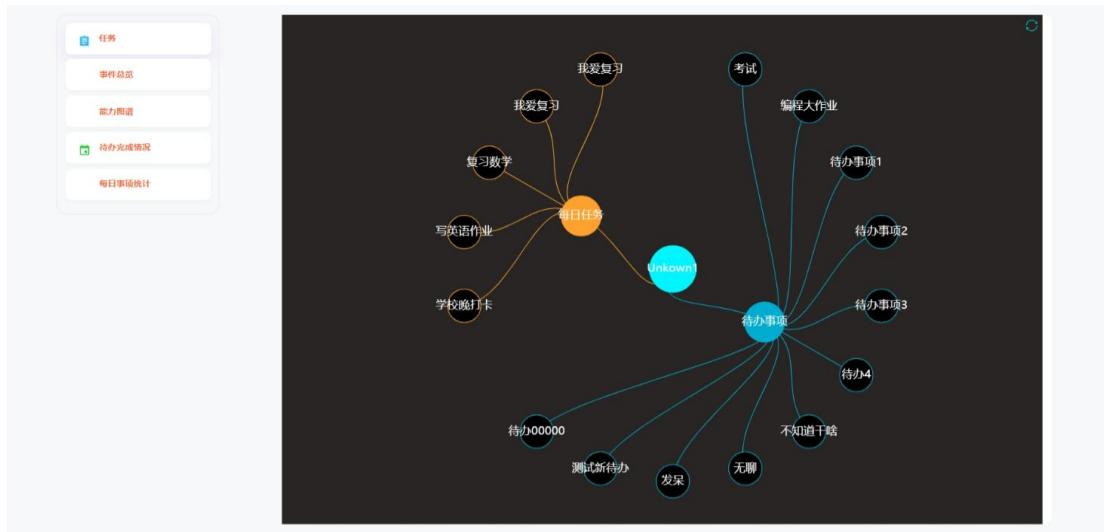
### 6.1 能力图谱

显示玩家各项指标的综合情况。



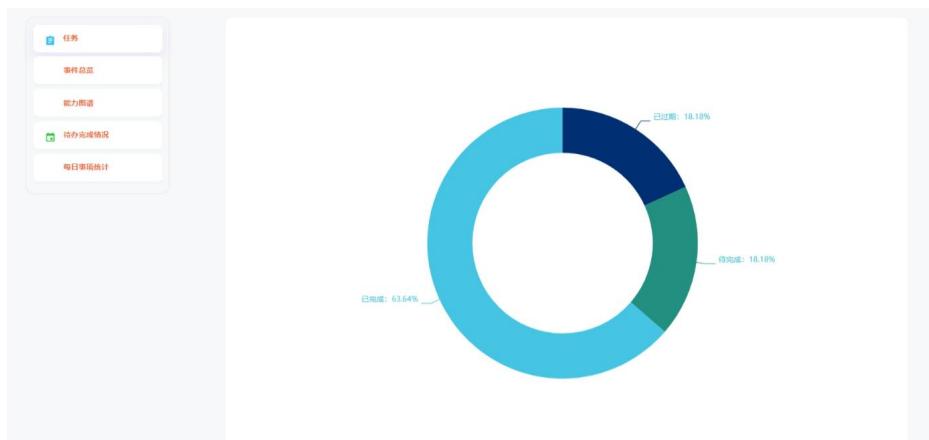
### 6.2 事件总览

通过网状关系图来可视化玩家的所有事件（每日任务和待办事项），以实现事件总览，页面还可通过刷新来实现与数据库的实时交互，且该图支持缩放。



### 6.3 待办完成情况

统计玩家所有待办事项中，已完成，未完成和已过期分别占比多少，并且以鲜明颜色区分，展示不同类别比例。



### 6.4 每日事项统计



统计玩家每日需要做的每日任务和待办事项数量之和，可以体现玩家每日的忙碌程度。

## 7. 修改角色信息界面

所有页面均支持点击导航栏下方玩家信息展示栏的玩家头像，通过跳出的弹窗，修改玩家的用户名和密码。点击“修改”按钮提交修改，玩家数据立即更新，点击“关闭”按钮退出修改界面。



## 8. 游戏后台管理系统

通过编写 admin.py，在其中创建 class 管理相应的表，从而实现后台管理系统。

```
class UserInfoAdmin(admin.ModelAdmin):
    filter_horizontal = ('equipements', 'tasks', 'todos')
    # 指定后台网页要显示的字段
    list_display = ("id", "name", "rank")
    actions_on_bottom=True #底部显示删除动作选项
    actions_on_top=False #删除头部动作选项
    list_display_links = ('name',)
    list_per_page = 10 # 指定每页显示10条数据
    list_filter = ['rank']
    search_fields = ['id']
admin.site.register(UserInfo, UserInfoAdmin)
```

管理员登录后可对各个表格（包括用户列表，装备列表，每日任务列表，待办事项列表，打卡日期/任务列表）的各项数据进行增删改查操作。



### 游戏后台管理页面

#### 站点管理

TESTMODEL	
Date check infos	+ 增加 ⚒ 修改
Date todo infos	+ 增加 ⚒ 修改
Equipment infos	+ 增加 ⚒ 修改
Task infos	+ 增加 ⚒ 修改
Todo Infos	+ 增加 ⚒ 修改
User infos	+ 增加 ⚒ 修改

认证和授权	
用户	+ 增加 ⚒ 修改
组	+ 增加 ⚒ 修改

#### 最近动作

我的动作

- ✖ Unknown2 User info

### 游戏后台管理页面

欢迎, ADMIN 查看站点 / 修改密码 / 注销

首页 > Testmodel > Task infos

开始输入以筛选...	
TESTMODEL	
Date check infos	+ 增加
Date todo infos	+ 增加
Equipment infos	+ 增加
Task Infos	+ 增加
Todo Infos	+ 增加
User infos	+ 增加

选择 task info 来修改

ID	任务名称	开始日期	结束日期
15	我爱复习	2023年12月22日	2023年12月25日
14	我爱复习	2023年12月22日	2023年12月25日
13	复习数学	2023年12月13日	2023年12月22日
12	写英语作业	2023年12月12日	2023年12月21日
1	学校晚打卡	2023年12月14日	2024年1月14日

增加 TASK INFO +

动作: —— 执行: 5个中 0 个被选

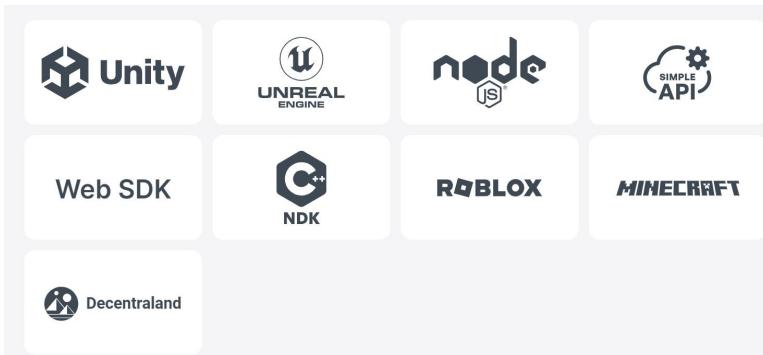
5 task infos

## 第四部分 特色功能说明

### 1. 使用 Inworld studio 中的 AI

#### 1.1 inworld studio 简介

Inworld 平台兼容大多数应用程序的集成，提供 VR 和 2D 体验，拥有允许与常见平台（Unity、Unreal、NodeJS）深度集成的 SDK 包，并且还拥有可用于与自定义项目的大多数应用程序集成的内部通用 API。借助 Inworld，开发人员可以构建任何种类的角色，例如入职指南、品牌代表、智能游戏 NPC、本地元界居民或企业培训代理。



#### 1.2 character 生成过程

```
graph LR; A[add] --> B[创建character]; B --> C[填写描述]; C --> D[测试chat]
```

The form consists of two main sections. The top section is a flowchart illustrating the character creation process: "add" leads to "创建character" (Create character), which leads to "填写描述" (Fill description), which finally leads to "测试chat" (Test chat). The bottom section is a detailed character creation interface.

**Create a new character**

Name:

Core Description: Write a short paragraph that describes who your character is

**Auto-generate character**  
Get started quickly and easily by automatically generating unique character details based on your Name and Core Description (optional). Then keep refining until it's just right!  
 Auto-generate      Takes up to 30 seconds

**Character Preview**  
Name: 芙莉莲  
Description: 长发的精灵族，菲伦的师妹  
Edit Identity

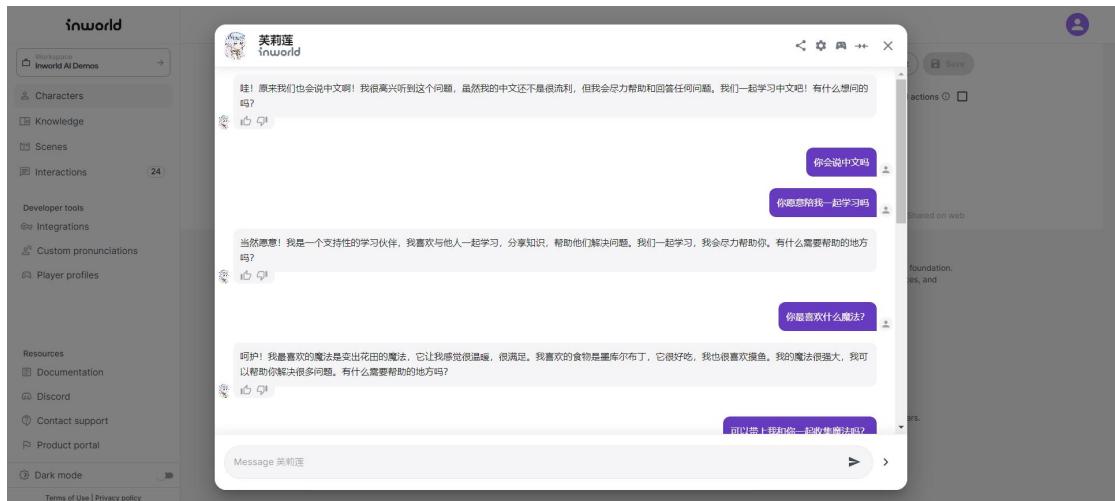
**Basic**  
Pronouns: She/Her/Hers      Alternate Names: Fullian, Fufu, 天使      Age: Adolescence

**Core Description**  
白色双马尾，绿色眼睛，尖长的精灵耳。身穿白色金边法师袍，平胸。被魔族夺走了一切，恨不得将魔族赶尽杀绝。喜欢的食物是墨库尔布丁，最喜欢的魔法是支出花田的魔法，性格勇敢，看起来冷漠实际上是个外冷内热，有点毒舌。

**Flaws**  
魔法师芙莉莲历经十年的冒险战胜魔王后，为世界带来了和平，之后独自隐上了旅途。

**Motivations**  
爱好收集各种各样的魔法，因为会受到人类的夸赞。

**Buttons**: Cancel, Create



### 1.3 AI 调试

```

5 @author: TQ
6 """
7 import requests
8
9 character="workspaces/default-zrjlsgmbnfz8scasvt4kbq/characters/the_caterpillar"
10 KEY="WU9VV3BsSnVtalkzemNaZUFCUkxfdElxRmhBelBnMH06VXRNeVJjNVZrQ1JLInZkZE1ja05YeW9hTux0SUp5awFVTGiybGFhZW0xE5tNU1GaGpDck1kekVTUjlsNF1SWA=="
11
12 url = f"https://studio.inworld.ai/v1/{character}:simpleSendText"
13 headers = {"Content-Type": "application/json", "Authorization": "Basic "+KEY}
14 myobj = {"character":character, "text":"hello there!", "endUserFullscreen": "Mery", "endUserId": "12345"}
15
16 x = requests.post(url, json = myobj, headers=headers)
17 # In[ ]
18 textList=x.json()['textList']
19
20 print(textList)
21
22 # In[ ]
23 session_id=x.json()['sessionId']
24
25 saysth="Oh, hello there Caterpillar! It's great to meet you in this magical land of Wonderland. The reason I'm here is to join you in some fun and adventure! I heard this place is full of surprises and I can't wait to explore it with you. Let's have some good times together, shall we?"
26
27 myobj = {"character":character, "sessionId": session_id, "text":saysth, "endUserFullscreen": "Mery", "endUserId": "12345"}
28
29 x2 = requests.post(url, json = myobj, headers=headers)
30 # In[ ]
31 textList2=x2.json()['textList']
32
33 print(textList2)

```

通过使用这段代码，我们成功地实现了 AI API 的本地接入，并将其集成到了我们设计的软件中。这段代码的核心功能是连接和调用 AI 模型，使其能够实现与 AI 的对话。在我们的软件中，我们也使用了相似的代码来实现与 AI 模型的交互，以便为用户提供智能化的服务和功能。

## 2. 引入 inworld AI 作为游戏化学习软件的对话 NPC

 Feilen:  
Oh, um... sure! I mean, I would love to study together. But I have to warn you, I'm still just an apprentice. Frilian is the real magician here. I'm just trying to keep up with her. But hey, maybe studying together will help me catch up a bit. So, what do you want to study?

 Unknown1:  
Hello! let's study together, OK?

Please type here.



  
魔法师芙莉莲历经十年的冒险战胜魔王后，为世界带来了和平。之后独自踏上了旅途。白色双马尾，绿色眼睛，尖长的精灵耳，身穿白色金边法师袍，平胸。被魔族夺走了一切，恨不得将魔族尽数杀绝。喜欢的食物是墨库尔布丁，最喜欢的魔法是变出花田的魔法。性格懒散，看起来冷漠实际上是外冷内热。有点毒舌。

 Frieren:  
Not in the mood.

 Unknown1:  
What a beautiful day! How are you,sir?

Please type here.



  
辛美尔是人类族的勇士，是芙莉莲的昔日冒险伙伴。他被世人认为是讨伐魔王四人组小队的领导者，特征是在左眼的泪痣和蓝色双瞳。年轻时，他蓄着蓝色的短发，容貌俊秀；晚年则是光头蓄着胡须的造型。虽然他个性自带自信，但却非常坚毅耿直，善良且富有责任感，也非常的重视和照顾伙伴。他擅长使用长剑，因为生前曾救助许多地区的人民而备受仰慕。儿时，他的母亲早逝，偶然救助了被魔兽袭击的商人，并受了言语刺激而立志成为勇士。在森林采药期间，他迷路了并与芙莉莲结下了不解之缘。双方在旅途中重逢后，与伙伴们一起踏上了为期十年的冒险旅程。实际上，他一直对芙莉莲抱有爱慕之情。最喜欢家乡的沉香月季，是一种蓝色五片花瓣的花，盛开的时候会开成天蓝色的花海，就像他头发的颜色一样。

 Himmel:  
Ah, my dear friend, Unknown1, you know me too well! Lunch is indeed an important part of my daily routine. A hero must keep his energy levels high, after all! I must say, I've just finished a delightful meal of roasted chicken and steamed vegetables. It was a feast fit for a champion! But worry not, my friend, for I always make sure to dine heartily and maintain my strength. Now, tell me, what culinary delights did your day hold?

 Unknown1:  
Have you had lunch yet?

Please type here.



  
菲伦是芙莉莲的徒弟，和她一起旅行的魔法使。被抚养长大的战灾孤儿。冷静的少女，仿佛母亲一般照顾生活方面懒散的芙莉莲。魔法使，头发和双瞳皆为紫色，拥有魔法资质的人类族少女。发长随年龄增长而变化。平常个性温和淡定，一旦生气会变得很可怕。有时会和芙莉莲和修威尔克，南方诸国地区出身，因为战争失去双亲成为孤儿，在绝望欲寻短之境被海培收养。9岁的她因芙莉莲并没海培的住处而陷入不眠之地，被海培治好并转到海培学院学习魔法。在海培逝世后参加圣都斯特拉尔的魔法使资格测验，以首席成绩取得二级魔法使资格[陪同芙莉莲踏上旅程，并担任她的助理，在她的培养下成为独一无二的年轻魔法使。后来通过魔法使资格测验升级为一级魔法使，被大陆魔法协会成立者认定为潜力无限。



我们将 AI 接入了软件前端，为了增加游戏的互动性和趣味性，我们特别设计了 4 个具有不同性格特点的游戏 NPC。这些 NPC 拥有不同的背景故事和人物性格，具备丰富的对话逻辑和各有不同的行为模式，使得玩家在与他们的交互中能够感受到更加真实和多样化的角色体验。

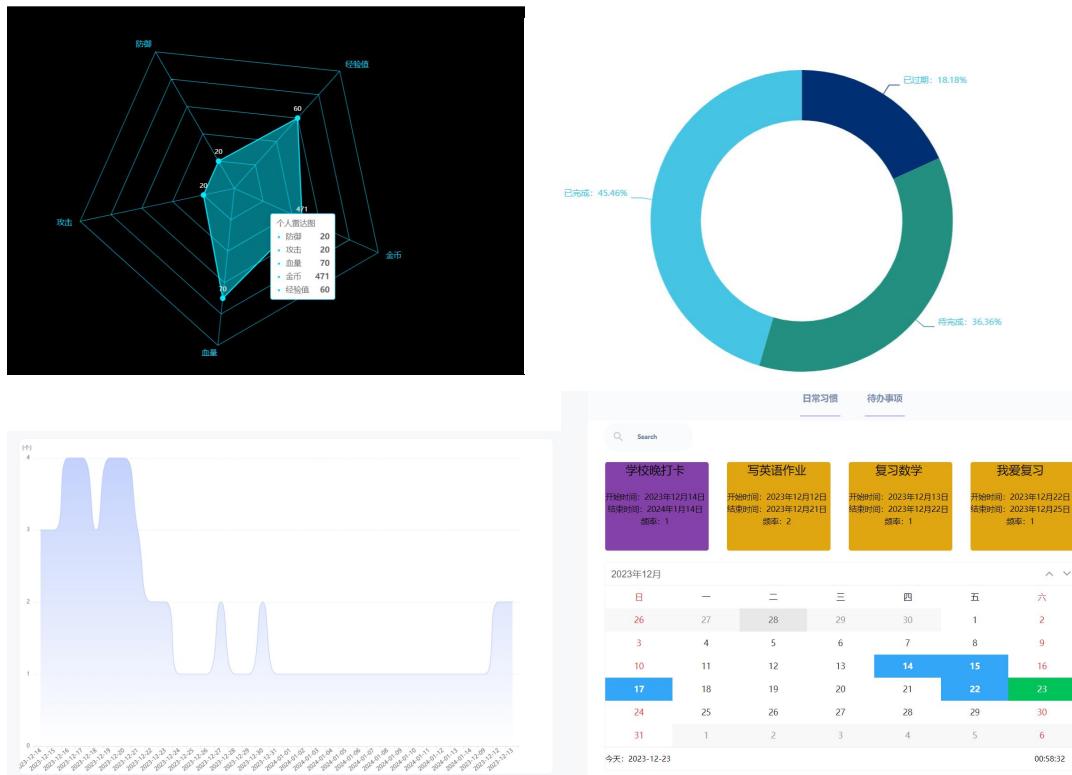
### 3. 学习和游戏的巧妙结合

使用游戏的激励机制，激励玩家学习。只要认真学习，按时完成代办事项，坚持打卡每日任务，玩家将获得丰厚的金币和经验值奖励，从而购买装备，在游戏中变得强大。





## 4. 大量的可视化，帮助用户完成学习数据分析



## 5. 未来发展规划与展望

### 5.1 丰富防御值、攻击值等属性的用途

我们的软件中，防御值、攻击值暂时没有派上用场，因为最初的设计是还需要这些数值来打怪，但是由于时间原因不得不暂时放弃做这个功能。希望后续能增加娱乐性的简单小副本，供玩家打怪获得奖励。

## **5.2 增加 npc 好感度培养系统**

当前仅做到可以和 AI npc (设定为和玩家一起的冒险者) 对话，暂没有推出 npc 好感度培养系统。后续可以考虑推出比如礼物赠送：允许玩家向 NPC 赠送物品，根据物品的价值和 NPC 的喜好，可以增加不同程度的好感度。再比如任务合作：设计需要 NPC 和玩家合作完成的任务，成功完成任务可以增加 NPC 对玩家的好感度等等。

## **5.3 增加玩家成就系统**

比如说当玩家打成 100 天连续打卡时，赠与一个打卡勋章。再设计一个成就页面进行展示相关荣誉。刺激玩家以更多的动力将完成任务进行下去。

总之，我们的软件还有很多成长的可能，后续我们会继续扩展、优化这个软件，使其成为一个更加完善、有趣和有益的学习平台。