# Project Proposal
# Nature Edge Sustainable Community

Ruotong Sun, Xinran Liu, Juncheng Zhang, Huiyang Liu, Shizheng Wang, Te Qi

## 1 Problem State

Since sustainable development is urgently needed, Nature's Edge Property Management Company has realized that meeting the goal of sustainable communities in Article 11 of SDG has become a major problem for them. In the community, Nature's Edge Property Management Company has already provided some convenient facilities, such as fitness equipment, community hospitals, etc. However, according to the community survey, the utilization rate of these facilities is extremely low, making it difficult to achieve the goal of sustainable communities. Additionally, the community has problems with high energy consumption and high carbon emissions, which are inconsistent with the concept of sustainable development.

## 2 Vision

GreenVista Software Company's aims to develop a comprehensive community service website for the existing community to solve the problems of Nature's Edge Property Management Company. Our vision is to build a community comprehensive service website for Nature's Edge Property Management Company that meets user needs in terms of functionality, security, and reliability. We will provide a sustainable community template website with community resource integration, owner resource consumption control, sustainability guide, and employment recruitment.

There are four modules on the website. The community service queries support map navigation, making an appointment or inquiry door-to-door services, and other services. The resource management section allows owners to check energy consumption and automatically make early warnings. In the electronic library, user can search for books and read them online, which achieves the goal of the education aspect of SDG 4. In the business docking users can enter the community recruitment platform, which meets the need of SDG Article 8.

From the perspective of functions, community resource map navigation, artificial intelligence, and electronic libraries are attractive to users, while automated resource monitoring and energy statistics can provide users with fast and convenient sustainable energy consumption management. Moreover, our projects enhance security through the use of encryption methods and the implementation of rights management systems, providing different services to our clients and their clients. The reliability of our projects can be guaranteed through manual and automated testing. In addition, the website also provides sustainability guidance and is equipped with an artificial intelligence assistant to help users make a sustainable living. The emerging platform will be in sharp contrast to the existing traditional communities and help build a sustainable smart community in an all-round way.

## 3 Benefits and Deliverables

Our deliverable is a Sustainability Community Software: based on the requirements of SDG 11, a complete software has been developed to promote the sustainable development of the community. The software should have a range of features to support the needs of community service, resource management, community engagement, etc.

- **Functional implementation**

  Ensure that the software implements all the functions listed in the project plan such as sustainability guidelines, health checks, owner resource management functions, job search functions, etc. In the process of development, the project may be developed or expanded according to the needs of continuous optimization has obtain the best results.

- **Quality assurance**

  To ensure that the actual quality of the project meets expectations. This means ensuring that the performance, stability, and security of the software are at the expected level. We will use software testing, code reviews, performance testing, etc., to ensure quality.

- **Comprehensive testing**

  All functions of the software are thoroughly tested before delivery. This includes functional testing, performance testing, user experience testing, etc., to ensure that the software is complete, stable, and meets user needs at the time of delivery.

- **Positive feedback**

  Receiving positive feedback during the testing phase ensures that the needs and expectations of users are fully understood and met before the software is delivered. Positive feedback can be obtained through communication with community members and user feedback collection, and the software can be adjusted and improved in time.

# 4 Evaluation criteria

Some of the evaluation criteria required to implement the project are listed below:

- **Security and Ethical standards**

  This standard aims to create a safe and regulated use environment, strictly abide by ethical standards, and ensure the security of users' data. In the collection of personal data, resolutely ensure the security of the privacy of users' personal information, such as the sleep quality and health monitoring of community-related service functions, will be carried out under the premise of users' priority to allow the lawful collection of personal data, and the collected data will be privatized for users and their related information, and users can also change the collected data. The function of the software and the relevant data displayed ensure that the content can be read healthily and safely, and do not use false and illegal data.

- **Project-related management criteria**

  —Project modular management. The project will be divided into 7 functional modules for implementation and management, which make it easy to deal with some problems that may be encountered in the future, and will also facilitate the optimization of the project in the later stage.

  —Milestones will serve as project metrics. The overall project development process will be carried out step by step, implementing module functions, testing, and collecting data feedback within the specified time.

- **About the project practicality standard**

  This standard is committed to ensuring that unregistered visitors (anonymous users) can use some of the functions of the website and some of the reading rights to browse the website, and to ensuring the relevant rights of customers who use the product by registered users (owners will be able to use most of the service functions of the software, and properties can use the management functions of the software, etc.).

# 5 Deadlines/Plan/Approach

The Gantt chart describes the detailed division of labor arrangements for each team member under the corresponding work package, and the team will flexibly adjust the work arrangements during the development process to ensure that it does not deviate from the project goals and meet user needs.

Technical details: The project is expected to adopt Web development with front-end and back-end separation and adopt an agile development model. The front end uses react as the framework introduces ant design, and uses the nextjs framework with Zustand state management to ensure the reliability and aesthetics of the project. The backend uses Flask + SQLALchemy. The project uses Figma for UI design to ensure that the project is fully functional, the webpage conforms to the user's usage logic, and the webpage design is beautiful and reasonable. The project uses GitHub as a repository management tool, pnpm for software dependency package management, and built-in eslint to ensure that development members have consistent code styles and are easy to read. After the functions are established, interface documents are carefully written to reduce the cost of front-end and back-end docking.

# 6 Cost/Budget/Team Requirements

- 100,000 for server building
- 1.5 million for software development and supporting hardware development and adaptation
- 200,000 models used for software big data and machine learning
- 200,000 for information security enhancement

| | | | |
|---|---|---|---|
| **PROJECT / GROUP NAME** | **Group 14** | | |
| **Start Date** | 2024.03.04 | **Finish Date** | 2024.7.10 |
| **Aim / Objective** | Project design: The purpose of this work package is to think about what problems need to be solved from the customer's perspective and to provide solutions to these problems from the software development team's perspective. It is necessary to provide clear project description documents, complete processes for specific functions, precise UI design, etc. Later, the architecture design will be connected and some details will be explained and corrected. | | |
| **Work package Manager** | Liu Xinran | | |
| **Contributors to this package** | Sun Ruotong<br>Wang Shizheng<br>Qi Te<br>Liu Huiyang<br>Zhang Juncheng | | |
| **Description / Activities** | Task 1.1 Define user requirements<br>• 1.1.1 Gather ideas from team members on possible project topics through brainstorming.<br>• 1.1.2 Discuss the feasibility and relevance of each option to the topic.<br>• 1.1.3 Vote anonymously for the most suitable solution.<br>• 1.1.4 Convert the plan into requirements and specify the functions that may need to be implemented.<br>Task 1.2 Specify functionalities<br>• 1.2.1 Hold a brainstorming to think about all the functional according to the user requirement.<br>• 1.2.2 Evaluate the feasibility and necessity of each function.<br>• 1.2.3 Refined description of all the functions involves, such as combining a function with the corresponding user story.<br>• 1.2.4 Check duplications and conflicts among functions.<br>Task 1.3 Design UI<br>• 1.3.1 Determine the overall website structure and components.<br>• 1.3.2 Determine the theme style and color scheme.<br>• 1.3.3 Design the individual interface for each function.<br>• 1.3.4 Determine the size and proportion of each element.<br>• 1.3.5 The design draft is finally presented using design software such as Figma and public to everyone on the team.<br>Task 1.4 Optimize UI<br>• 1.4.1 Check the correspondence between functions and designs, ensuring that each function has a demonstration in UI design. | | |
| **Milestones** | | | Week |

| | | |
|---|---|---|
| | M 1.1 Complete project design and determine tasks<br>M 1.2 Determine the detailed function of each task<br>M 1.3 Complete prototype UI design draft<br>M 1.4 Complete the detailed UI design, including all functions<br>M 1.5 Confirm that the UI design contains all functions | **2**<br>**6**<br>**9**<br>**11**<br>**16** |
| **Deliverables** | | Week |
| | D 1.1 Deliver idea of project design and task description documents<br>D 1.2 Deliver detailed project design description, including documentation of all functions<br>D 1.3 Deliver project UI design draft<br>D 1.4 Deliver the completed UI design | **2**<br>**6**<br>**11**<br>**16** |

| | | | |
|---|---|---|---|
| **PROJECT / GROUP NAME** | **Group 14** | | |
| **Start Date** | 2024.03.18 | **Finish Date** | 2024.07.10 |
| **Aim / Objective** | Architecture and interface design: Connect with the project design department, review project design-related documents from the perspective of the development team, and discuss the feasibility of solutions and functions and more details. Design the connections and details of each part of the entire software from the architectural level to ensure high cohesion and low coupling of each part of the software, while maintaining high scalability to cope with changing needs. Provide complete interface documents and database design to connect subsequent code development processes and ensure the feasibility of architectural design. | | |
| **Work package Manager** | Sun Ruotong | | |
| **Contributors to this package** | Liu Xinran<br>Wang Shizheng<br>Qi Te<br>Liu Huiyang<br>Zhang Juncheng | | |
| **Description / Activities** | Task 2.1 Evaluate function feasibility<br>•2.1.1 Re-read the existing feature list.<br>•2.1.2 Evaluate the feasibility and necessity of function from a software engineer's view.<br>•2.1.3 Re-discuss the unclear functions and deal with the controversy.<br>Task 2.2 Conduct functional descriptions<br>•2.2.1 Explain the tasks that have a specific meaning in the project.<br>•2.2.2 Further explain functions as engineering description.<br>•2.2.3 Use tools like flowcharts to describe functions with complex interaction logic.<br>Task 2.3 Design interface documentation<br>•2.3.1 Design interfaces for user registration, login, etc.<br>•2.3.2 Design the interface for displaying articles.<br>•2.3.3 Design interfaces for other functions.<br>o 2.3.3.1 Determined the data involved in the function<br>o 2.3.3.2 Map variables to the front and back ends<br>•2.3.4 Organize interfaces and find identical or similar interfaces to merge and split.<br>Task 2.4 Design database<br>•2.4.1 List the data required for each function.<br>•2.4.2 Combine identical data and assign easy-to-understand and unique variable names to the data.<br>•2.4.3 Abstract data relationships and create entity relationship diagrams.<br>•2.4.4 Generate ER diagrams according to relational database tables.<br>•2.4.5 Test database availability and reliability<br>Task 2.5 Select technology stack | | |

|  |  |  |
|---|---|---|
|  | • 2.5.1 Determine the appropriate technology stack selection based on the knowledge mastered by team members and combined with project functional requirements. |  |
| **Milestones** |  | Week |
|  | M 2.1 Confirm that all functions are feasible and necessary | **5** |
|  | M 2.2 Map all functional descriptions into professional terms | **7** |
|  | M 2.3 Draw a flow chart | **9** |
|  | M 2.4 Organize the required interfaces and design all interface | **11** |
|  | M 2.5 Design, create, and implement database |  |
|  | M 2.6 Determine technology selection and technology stack | **13** |
|  |  | **16** |
| **Deliverables** |  | Week |
|  | D 2.1 Deliver a complete flowchart or ER diagram | **9** |
|  | D 2.2 Deliver complete interface documentation | **11** |
|  | D 2.3 Deliver a complete design database | **13** |
|  | D 2.4 Delivery technology selection and technology stack description | **16** |

| | | | |
|---|---|---|---|
| **PROJECT / GROUP NAME** | **Group 14** | | |
| **Start Date** | 2024.04.01 | **Finish Date** | 2024.07.10 |
| **Aim / Objective** | Implementation of front-end information acquisition function: Carry out code development in parallel and implement front-end functions and interfaces according to project design documents and architectural design. This work package will realize the functions of community service inquiry, educational resource integration, and sustainable development guide. Combine architectural design and back-end development to ensure interface consistency. | | |
| **Work package Manager** | Wang Shizheng | | |
| **Contributors to this package** | Sun Ruotong<br>Liu Xinran<br>Qi Te<br>Liu Huiyang<br>Zhang Juncheng | | |
| **Description / Activities** | Task 3.1 Implement interface<br>•3.1.1 Design request interception rules and add request header information appropriately<br>•3.1.2 Complete the front-end request api according to the interface document<br>•3.1.3 Find relevant articles to display as demos<br>Task 3.2 Implement the sustainable development guide function<br>•3.2.1 Draw page elements based on design draft<br>•3.2.2 Write logic for functions<br>•3.2.3 Properly display the obtained back-end data<br>•3.2.4 Find relevant articles to display as demos<br>Task 3.3 Implement community service query function<br>•3.3.1 Draw page elements based on design draft<br>•3.3.2 Handle query requests appropriately<br>•3.3.3 Find relevant articles to display as demos<br>Task 3.4 Realize the integration function of community education resources<br>•3.4.1 Draw page elements based on design draft<br>•3.4.2 Write logic for functions<br>•3.4.3 Find relevant articles to display as demos<br>Task 3.5 Optimize completed code<br>•3.5.1 Some code may need to be refactored<br>Task 3.6 Joint debugging of front-end and back-end<br>•3.6.1 Accept errors reported by testers and analyze whether they are front-end code problems<br>•3.6.2 Assist with backend staff to fix bugs | | |
| **Milestones** | | | Week |

|  |  |  |
|---|---|---|
|  | M 3.1 Interface routing design completed | **5** |
|  | M 3.2 Successfully displayed and obtained backend data | **7** |
|  | M 3.3 Successfully implemented community service inquiry function | **10** |
|  | M 3.4 Realize the integration function of community education resources | **12** |
|  | M 3.5 Optimize and refactor existing code | **14** |
|  | M 3.6 Complete front-end and back-end joint debugging | **16** |
| **Deliverables** |  | Week |
|  | D 3.1 Deliver front-end API part code | **6** |
|  | D 3.2 Deliver and implement part of the code for sustainable development guidance | **8** |
|  | D 3.3 Deliver part of the community service query function code | **10** |
|  | D 3.4 Deliver community education resource integration functions | **12** |
|  | D 3.5 Deliver optimized and refactored code | **14** |
|  | D 3.6 Deliver all back-end code after joint debugging | **16** |

| | | | |
|---|---|---|---|
| **PROJECT / GROUP NAME** | **Group 14** | | |
| **Start Date** | 2024.04.01 | **Finish Date** | 2024.07.10 |
| **Aim / Objective** | Front-end interactive function logic implementation: Carry out code development in parallel and implement front-end functions and interfaces according to project design documents and architectural design. This work will realize the functions of user system, owner resource management, and community employment platform. Combine architectural design and back-end development to ensure interface consistency. | | |
| **Work package Manager** | Qi Te | | |
| **Contributors to this package** | Sun Ruotong<br>Liu Xinran<br>Wang Shizheng<br>Liu Huiyang<br>Zhang Juncheng | | |
| **Description / Activities** | Task 4.1 Implement interface<br>•4.1.1 Design request interception rules and add request header information appropriately<br>•4.1.2 Complete the front-end request api according to the interface document<br>Task 4.2 Implement user system functions<br>•3.2.1 Split user-related functionality into appropriate components<br>•3.2.2 Draw page elements according to design draft<br>•3.2.3 Realize the functions of each component<br>•3.2.4 Test functions such as user registration and login<br>Task 4.3 Realize owner resource management function<br>•4.4.1 Obtain owner-related content data<br>•4.4.2 Use charts to demonstrate<br>•4.4.3 Call the artificial intelligence module to analyze some data and finally give reasonable suggestions.<br>Task 4.4 Realize community employment platform function<br>•4.4.1 Obtain existing recruitment information<br>•4.4.2 Display recruitment information<br>•4.4.3 Collect job application information filled in by users and send it to the backend<br>Task 4.5 Optimize completed code<br>•4.5.1 Some code may need to be refactored<br>Task 4.6 Joint debugging of front-end and back-end<br>•4.6.1 Accept errors reported by testers and analyze whether they are front-end code problems<br>•4.6.2 Assist with backend staff to fix bugs | | |
| **Milestones** | | | Week |

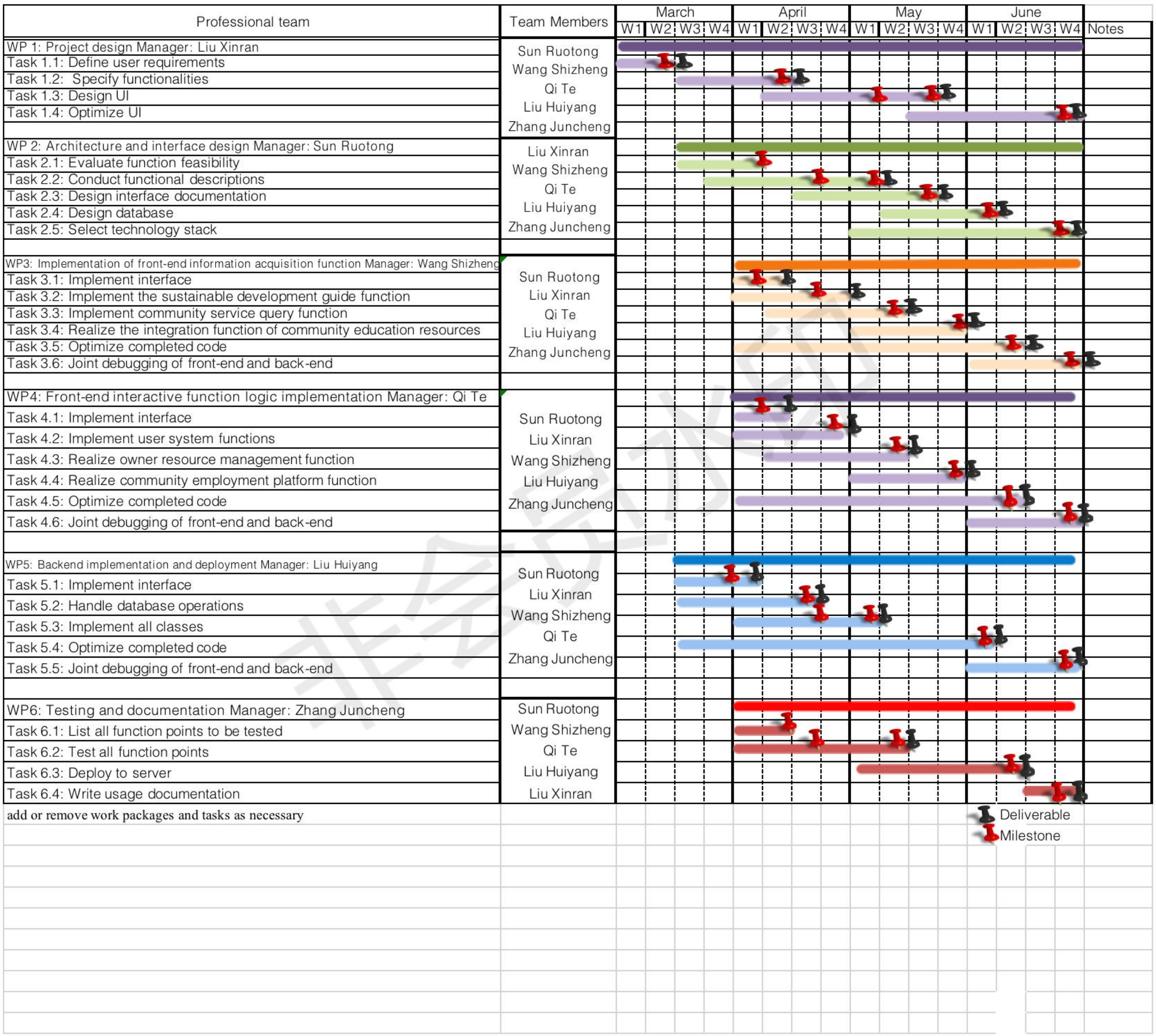
| | | |
|---|---|---|
| | M 4.1 Interface routing design completed | **5** |
| | M 4.2 Successfully implemented user system functions | **8** |
| | M 4.3 Successfully implemented owner resource management function | **10** |
| | M 4.4 Successfully implemented community employment platform function | **12** |
| | M 4.5 Optimize and refactor existing code | **14** |
| | M 4.6 Complete front-end and back-end joint debugging | **16** |
| **Deliverables** | | Week |
| | D 4.1 Deliver front-end API part code | **6** |
| | D 4.2 Deliver part of the system function code to the user | **8** |
| | D 4.3 Deliver part of the code that implements the owner resource management function | **10** |
| | D 4.4 Deliver part of the community employment platform function code | **12** |
| | D 4.5 Deliver optimized and refactored code | **14** |
| | D 4.6 Deliver all back-end code after joint debugging | **16** |

<table>
<tr><td>PROJECT / GROUP NAME</td><td colspan="3">Group 14</td></tr>
<tr><td>Start Date</td><td>2024.03.25</td><td>Finish Date</td><td>2024.07.10</td></tr>
<tr><td>Aim / Objective</td><td colspan="3">Backend implementation and deployment: Carry out code development in parallel, and implement back-end functions and interfaces based on project design documents and architecture design. Design and implement back-end code structure based on project architecture and entity relationships to ensure code readability and high performance. Connect architecture design and front-end development to ensure interface consistency.</td></tr>
<tr><td>Work package Manager</td><td colspan="3">Liu Huiyang</td></tr>
<tr><td>Contributors to this package</td><td colspan="3">Sun Ruotong<br>Liu Xinran<br>Wang Shizheng<br>Qi Te<br>Zhang Juncheng</td></tr>
<tr><td>Description / Activities</td><td colspan="3">Task 5.1 Implement interface<br>•5.1.1 Design backend routing paths<br>•5.1.2 Complete the back-end request api according to the interface<br>Task 5.2 Handle database operations<br>•5.1.1 Check and test whether relational database tables meet back-end usage requirements<br>•5.1.2 Establish an appropriate connection with the database and encapsulate basic operations such as addition, deletion, modification, and query<br>Task 5.3 Implement all classes<br>•5.3.1 Establish corresponding simple class diagrams based on the ER diagram and functional description given in the architectural design<br>•5.3.2 Implement code according to the designed class diagram<br>•5.3.3 Maintain high cohesion and low coupling of class implementation while maintaining code readability and scalability<br>Task 5.4 Optimize completed code<br>•5.1.1 Some code may need to be refactored<br>Task 5.5 Joint debugging of front-end and back-end<br>•5.5.1 Accept errors reported by testers and analyze whether they are back-end code problems<br>•5.5.2 Assist with frontend staff to fix bugs</td></tr>
<tr><td>Milestones</td><td colspan="2"></td><td>Week</td></tr>
<tr><td></td><td colspan="2">M 5.1 Interface routing design completed</td><td>4</td></tr>
<tr><td></td><td colspan="2">M 5.2 Test database meets requirements</td><td>7</td></tr>
<tr><td></td><td colspan="2">M 5.3 Design class diagram based on ER diagram</td><td>7</td></tr>
<tr><td></td><td colspan="2">M 5.4 Implement and optimize all classes</td><td>9</td></tr>
<tr><td></td><td colspan="2">M 5.5 Optimize and refactor existing code</td><td>13</td></tr>
</table>

| | | |
|---|---|---|
| | M 5.6 Complete front-end and back-end joint debugging | **16** |
| **Deliverables** | | Week |
| | D 5.1 Deliver the backend api part of the code<br>D 5.2 Deliver a database that establishes a connection to the backend<br>D 5.3 Deliver all backend classes<br>D 5.4 Deliver optimized and refactored code<br>D 5.5 Deliver all back-end code after joint debugging | **5**<br>**7**<br>**9**<br>**13**<br>**16** |

| | | | |
|---|---|---|---|
| **PROJECT / GROUP NAME** | **Group 14** | | |
| **Start Date** | 2024.04.08 | **Finish Date** | 2024.07.01 |
| **Aim / Objective** | Testing and documentation: This work requires comprehensive testing against the function points in the original project design to ensure the usability and fault tolerance of each function. Propose the problematic parts as early as possible and provide a rough modification plan, and notify the team members responsible for this part to repair them. After the software has been tested and no problems exist, a usage document will be written to explain how to use each part of the function. Finally, the software needs to be deployed to the server | | |
| **Work package Manager** | Zhang Juncheng | | |
| **Contributors to this package** | Sun Ruotong<br>Liu Xinran<br>Wang Shizheng<br>Qi Te<br>Liu Huiyang | | |
| **Description / Activities** | Task 6.1 List all function points to be tested<br>•6.1.1 Automate testing of content that can be automated<br>•6.1.2 Design multiple interactive tests for function points with interactive requirements<br>Task 6.2 Test all function points<br>•6.2.1 When a function point that does not meet expectations occurs, record all relevant test conditions to ensure the reproducibility of the exception.<br>•6.2.2 Remind team members who develop corresponding function points to make repairs or improvements<br>Task 6.3 Deploy to server<br>•6.3.1 Ensure that local testing of software no longer performs unexpectedly<br>•6.3.2 Package and deploy the software to the given server<br>•6.3.3 Perform functional testing again and repeat Task 6.2 when problems arise<br>Task 6.4 Write usage documentation<br>•6.4.1 Declare the development environment and deployment environment of the entire software<br>•6.4.2 Compare the listed function points for a detailed introduction<br>•6.4.3 Reread the document, look for content that is difficult to understand or make sense, and rewrite that part of the document | | |
| **Milestones** | | Week | |
| | M 6.1 List all test points and design interactive test plans | 6 | |
| | M 6.2 Test all function points | **7** | |
| | M 6.3 Confirm that no accidents will occur during local testing | **10** | |

|  |  |  |
|---|---|---|
|  | M 6.4 Deploy to server<br>M 6.5 Complete usage documentation | **14**<br>**16** |
| **Deliverables** |  | Week |
|  | D 6.1 Deliver tests and correct improved function points and code<br>D 6.2 Deliver successfully deployed project products<br>D 6.3 Deliver usage documentation including development and deployment environment, etc. | **10**<br><br>**14**<br>**16** |



| Professional team | Team Members | March |  |  |  | April |  |  |  | May |  |  |  | June |  |  |  | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 |  |
| WP 1: Project design Manager: Liu Xinran | Sun Ruotong<br>Wang Shizheng<br>Qi Te<br>Liu Huiyang<br>Zhang Juncheng |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 1.1: Define user requirements |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 1.2:  Specify functionalities |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 1.3: Design UI |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 1.4: Optimize UI |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| WP 2: Architecture and interface design Manager: Sun Ruotong | Liu Xinran<br>Wang Shizheng<br>Qi Te<br>Liu Huiyang<br>Zhang Juncheng |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 2.1: Evaluate function feasibility |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 2.2: Conduct functional descriptions |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 2.3: Design interface documentation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 2.4: Design database |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 2.5: Select technology stack |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| WP3: Implementation of front-end information acquisition function Manager: Wang Shizheng | Sun Ruotong<br>Liu Xinran<br>Qi Te<br>Liu Huiyang<br>Zhang Juncheng |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 3.1: Implement interface |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 3.2: Implement the sustainable development guide function |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 3.3: Implement community service query function |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 3.4: Realize the integration function of community education resources |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 3.5: Optimize completed code |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 3.6: Joint debugging of front-end and back-end |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| WP4: Front-end interactive function logic implementation Manager: Qi Te | Sun Ruotong<br>Liu Xinran<br>Wang Shizheng<br>Liu Huiyang<br>Zhang Juncheng |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 4.1: Implement interface |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 4.2: Implement user system functions |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 4.3: Realize owner resource management function |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 4.4: Realize community employment platform function |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 4.5: Optimize completed code |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 4.6: Joint debugging of front-end and back-end |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| WP5: Backend implementation and deployment Manager: Liu Huiyang | Sun Ruotong<br>Liu Xinran<br>Wang Shizheng<br>Qi Te<br>Zhang Juncheng |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 5.1: Implement interface |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 5.2: Handle database operations |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 5.3: Implement all classes |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 5.4: Optimize completed code |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 5.5: Joint debugging of front-end and back-end |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| WP6: Testing and documentation Manager: Zhang Juncheng | Sun Ruotong |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 6.1: List all function points to be tested | Wang Shizheng |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 6.2: Test all function points | Qi Te |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 6.3: Deploy to server | Liu Huiyang |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Task 6.4: Write usage documentation | Liu Xinran |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

add or remove work packages and tasks as necessary

Deliverable

Milestone