**Technical Specification Sheet**

**Product Documentation**

*The programs are more prototype than product in their current state*

Appointment Preference System

1. System Description

This project is a project preference management interface that allows the user to search, select and view project details. Users can select a certain number of projects and view a progress bar and list of selected projects after selection. Administrators can download forms and manage project quantity scores.

**The preferencing model** adopted by Appointment Preference System is a model based on the **allocation of finite resources**, where users are allocated a fixed total number of points (e.g., 10 points) and must trade-off between multiple items. The core idea of the model is that by limiting the total number of points allocated, users are prompted to make explicit rankings and trade-offs regarding the importance of each item. This preference model has the following characteristics:

1. Core features of the preference model
2. **Limited resources**: Users have a fixed total score (e.g., 10), which means that they cannot arbitrarily assign too many points to each project. The total score limit forces users to carefully consider the importance and priority of each item.
3. **Preference prioritization is made explicit**: because of the limited number of scores, users must explicitly choose the items they value most and give them higher scores. Those items considered less important will receive fewer or lower scores. In this way, the user's preferences are represented as relative weights for each item.
4. **Trade-offs in score assignment**: When users assign scores between items, they should consider not only the stand-alone value of an item, but also the impact on other items after assigning a high score to that item. Each high score assigned to an item means fewer scores available for assignment to other items. This trade-off reflects the user's **preference ranking**, i.e., which items are considered most important by the user.
5. **Properties of a zero-sum game**: under the fixed total score model, there is a zero-sum between increasing and decreasing scores. If one item receives a higher score, the other items must therefore receive lower scores. This reflects the trade-offs that users have to make when resources are limited, forcing them to clearly express priorities.
6. Specific manifestations of preference modeling

* **High Score Preference**: certain items are considered extremely important by users and receive a high score (e.g., 6-7). These items are prioritized higher in the user's mind.
* **Medium score preference**: some items are considered more important by the user, but less than the prioritized items and will receive a medium score (e.g., 3-5).
* **Low Score Preference or Ignored**: some items may receive only a small number of scores or even not be assigned scores (e.g., 0-2) because they do not meet the user's core needs.

1. Benefits and Challenges

Benefits:

**Clear expression of preferences**: Users have to make trade-offs with limited resources, which can clearly reflect users' prioritization of different items.

**Simplified decision making**: by enforcing a total score limit, users' choices are more focused and do not result in all items being treated equally.

Challenges:

**Insufficient flexibility**: Some users may find the total score limit too strict to rationally allocate their preferences across items.

**Cognitive load**: Users need to make complex trade-offs between multiple items, which may increase their cognitive load, especially if the number of items is high.

User story: [Appointment Preference System user stories](https://docs.google.com/spreadsheets/d/1XHC7Md2xk-OWQvvGMNbcslDmw10_Eo51/edit?usp=sharing&ouid=101034790233204394702&rtpof=true&sd=true)

1. System Diagram
   1. users

flow chart：[user flow chart.pdf](https://drive.google.com/open?id=1Mfh245SxjhXYd2jFFrnnC33jGpGpV_9u)

* 1. admin

flow chart：[admin flow chart.pdf](https://drive.google.com/open?id=1VUHwEDrjSMONxhVivfQDE5KvOdiAa81W)

1. System Details
2. Project structure：
3. app/pages: Contains multiple HTML files ( 1.html, 2.html, 3.html, admin.html ). These pages are static files provided for the front-end to display front-end content.
4. public:
5. data.json: stores data to manage the number of projects and scores.
6. project.csv: This may be a CSV file storing project information for use with project data.
7. total.xlsx: This is an Excel file used for data analysis, statistics or presenting reports.
8. index.js: The main entry file for the project, usually used to configure the server and routing, and to handle requests and responses.
9. node\_modules: This is where all installed npm packages are stored, including the project's dependency libraries.
10. package-lock.json & package.json:
11. package.json: records meta information about the project and required dependencies.
12. package-lock.json: locks specific dependency versions to ensure consistent installation results.
    1. technology stack:

• Node.js: Used to build backend servers.

• Express: Simplifies the creation of servers and API routing in Node.js.

• CORS: Allows cross-origin requests, ensuring that the frontend application can access the backend API.

• fs: Node.js' file system module, used for reading and writing files.

• csv-parser: A data parser used for reading CSV files.

• XLSX: Used for manipulating Excel files (reading and writing).

* 1. Environment Requirements

Node.js: Make sure you have Node.js (version 14 or higher) installed on your system.

npm: Node.js package management tool, usually installed with Node.js.

1. System setup
2. environment requirements：

* Node.js: Ensure that Node.js (version 14 or higher) is installed on your system.
* npm: The Node.js package management tool, usually installed along with Node.js.

1. Installation of dependencies:

Open a terminal in the project root directory and run the following command to install the required dependencies:

***npm install express cors csv-parser xlsx***

1. Start the server:

In the project root directory, start the server with the following command:

***cd app***

***node .\index.js***

The server will listen on port 3001.

1. Access to applications:

Other devices on the LAN can use the application by accessing http://<your IP address>:3001. For example, if the IP address of the server is 192.168.1.10, it can be accessed at http://192.168.1.10:3001.

1. API Description:

• POST /api/data

**Purpose**: Update quantity and scores.

**Return**: The updated data object.

• GET /api/projects

**Purpose**: Retrieve project CSV data.

**Return**: A data object containing project data in JSON format.

• POST /api/submit

**Purpose**: Submit project scores and related information.

**Return**: Success message.

• GET /api/Allproject

**Purpose**: Retrieve all projects and scores in Excel data.

**Return**: A data object containing Excel data in JSON format.

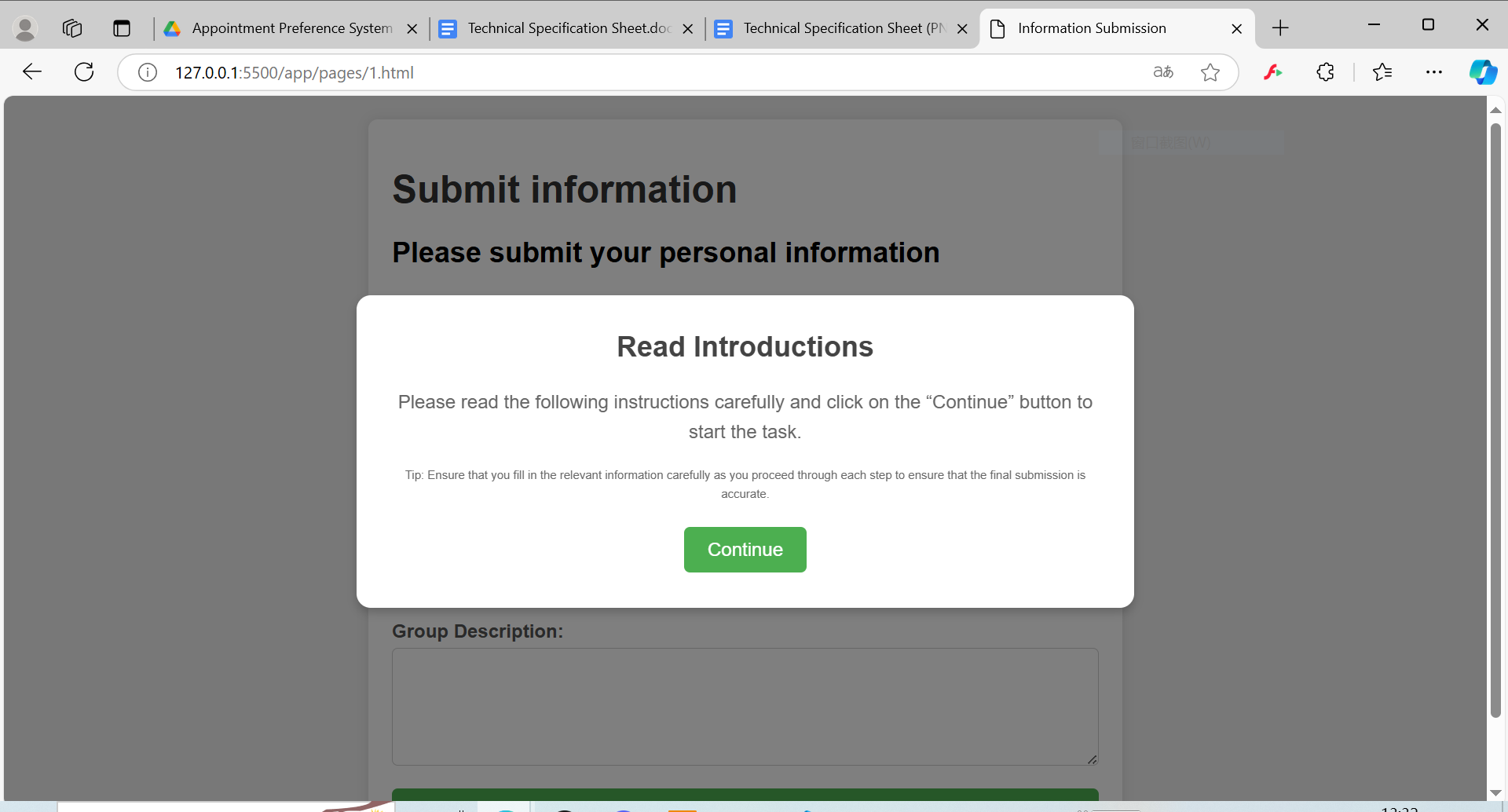
• GET /api/download

**Purpose**: Download the total.xlsx file.

**Return**: File download.

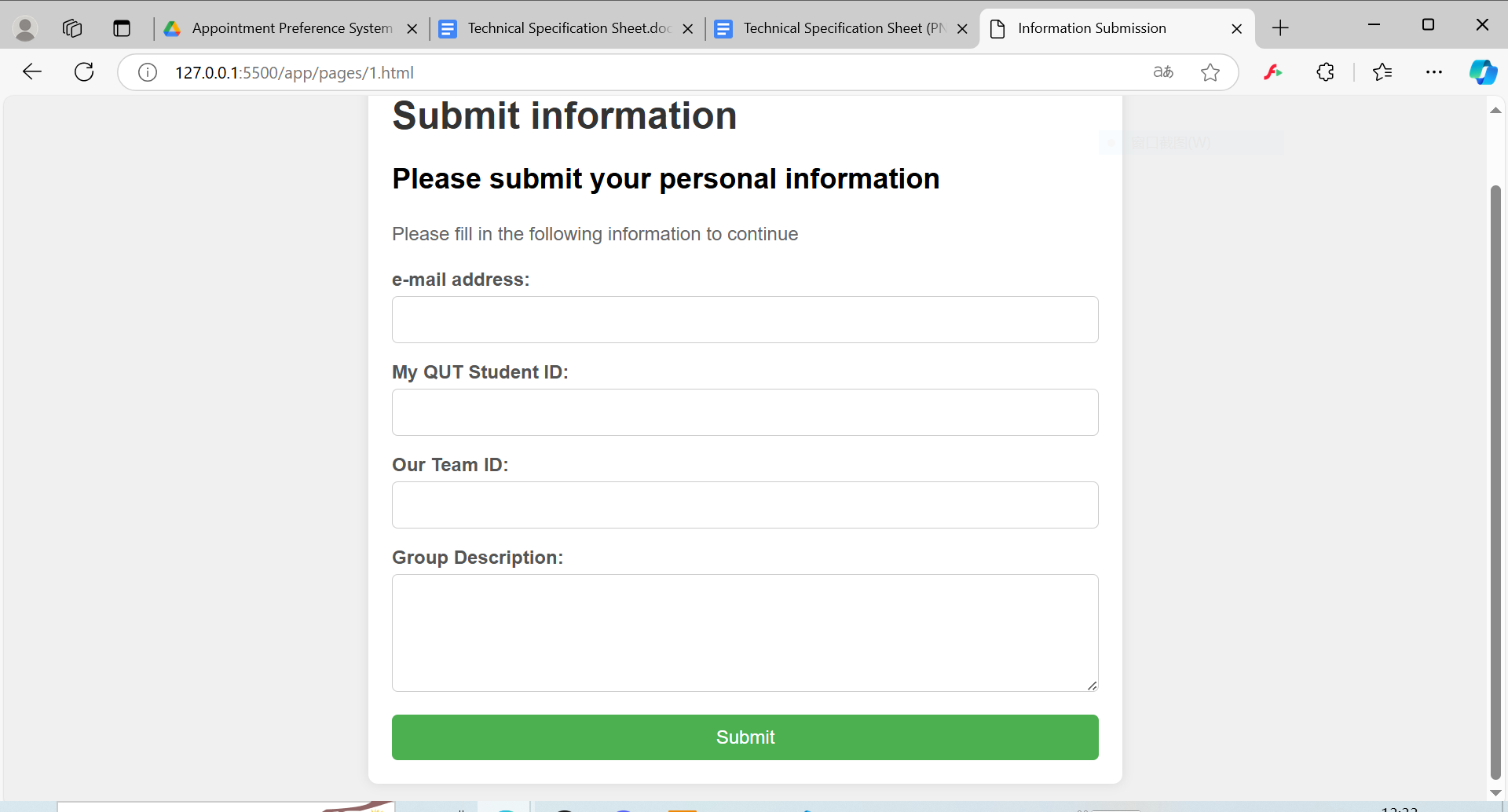
1. System Testing
2. User Information Form:

Modal: A pop-up window used to display informational instructions, requiring the user to read before proceeding.



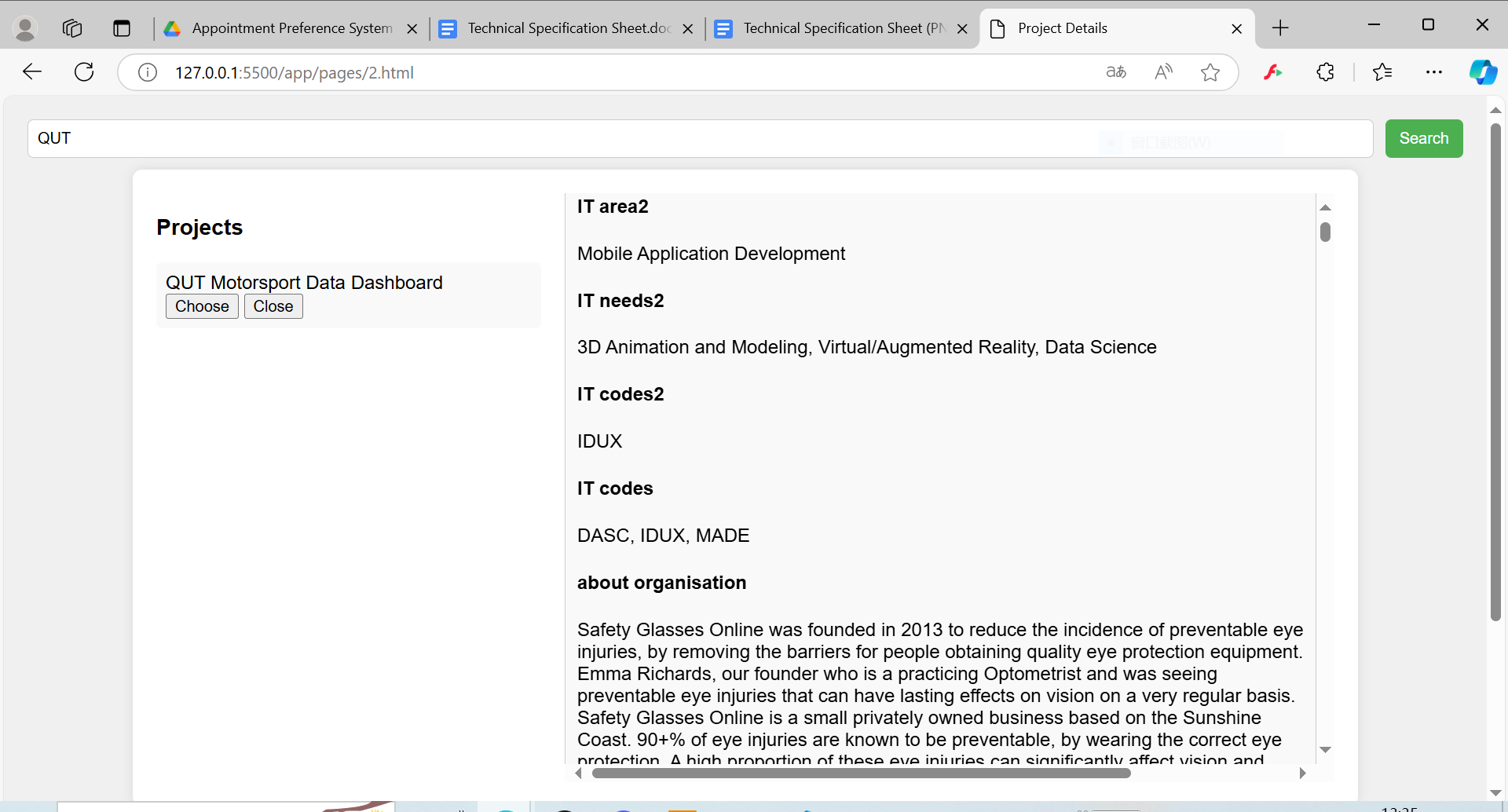
Title and Description: Contains the main title of the page and information to guide the user.

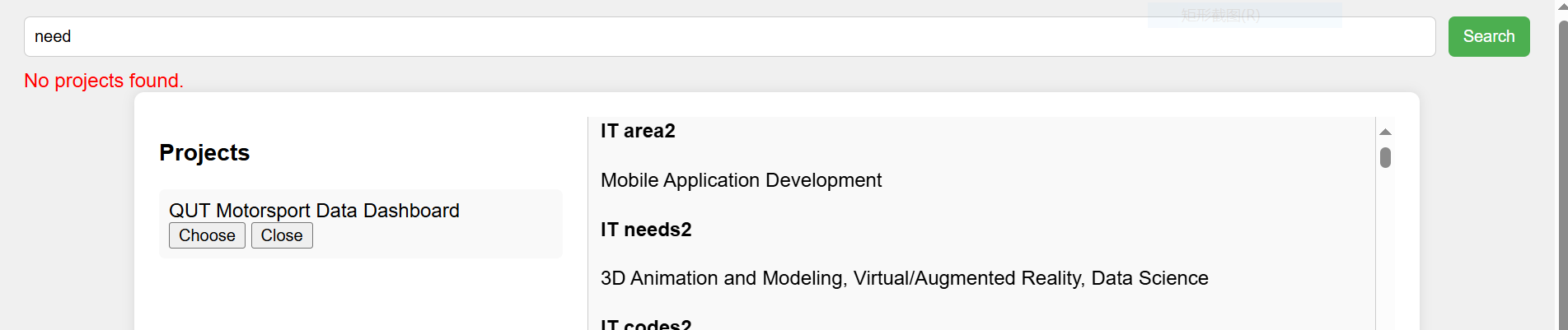
Information Submission Form: A form that users need to fill out, including fields for email address, student ID, team ID, and team description.



1. search project:

Users can search for projects by entering keywords in the search box. Clicking the "Search" button will filter the project list based on the entered keyword, and an error message will be displayed if no matching projects are found.



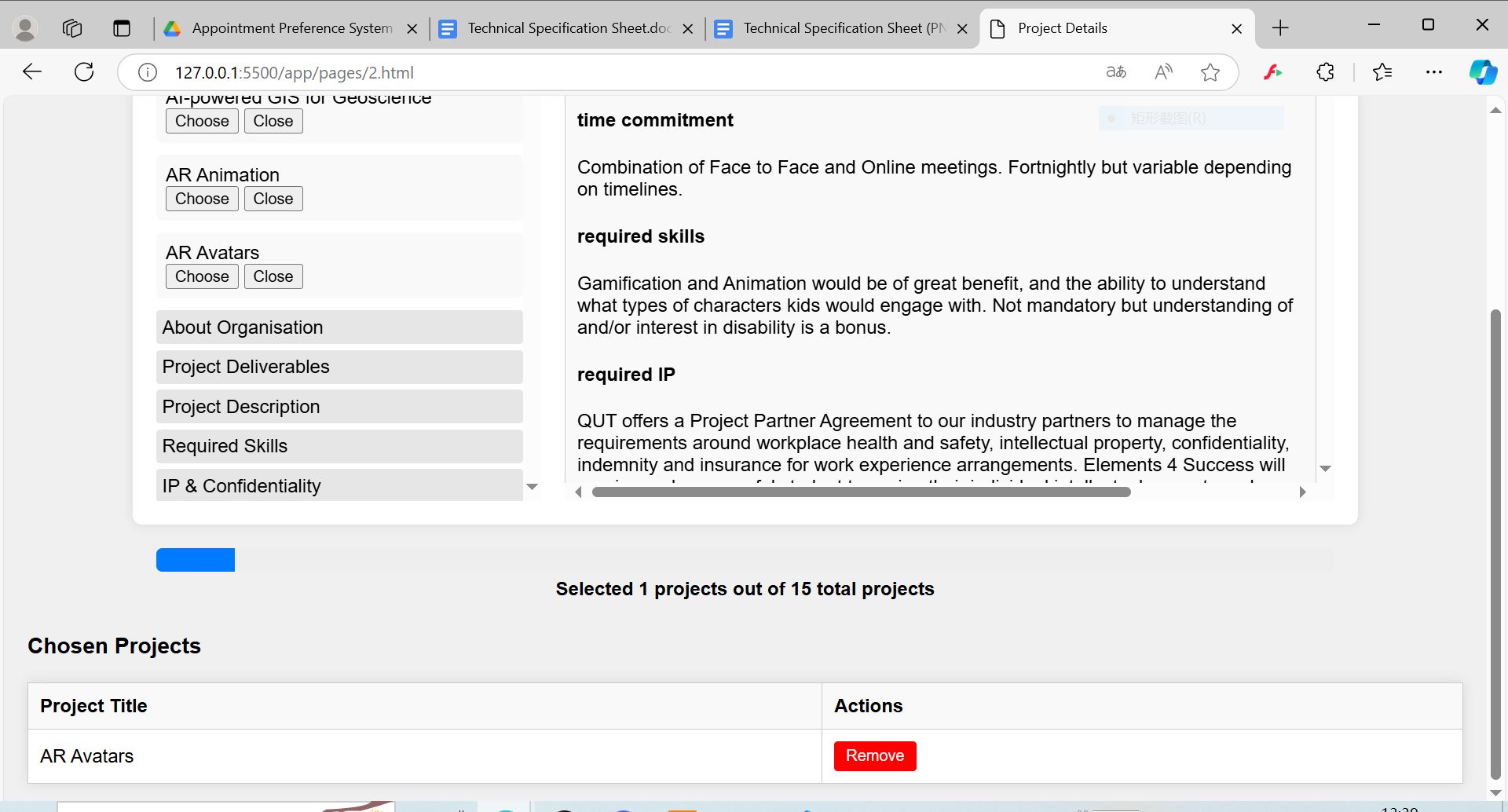


1. Select Project:

• Clicking on the project title allows users to view detailed information about the project.

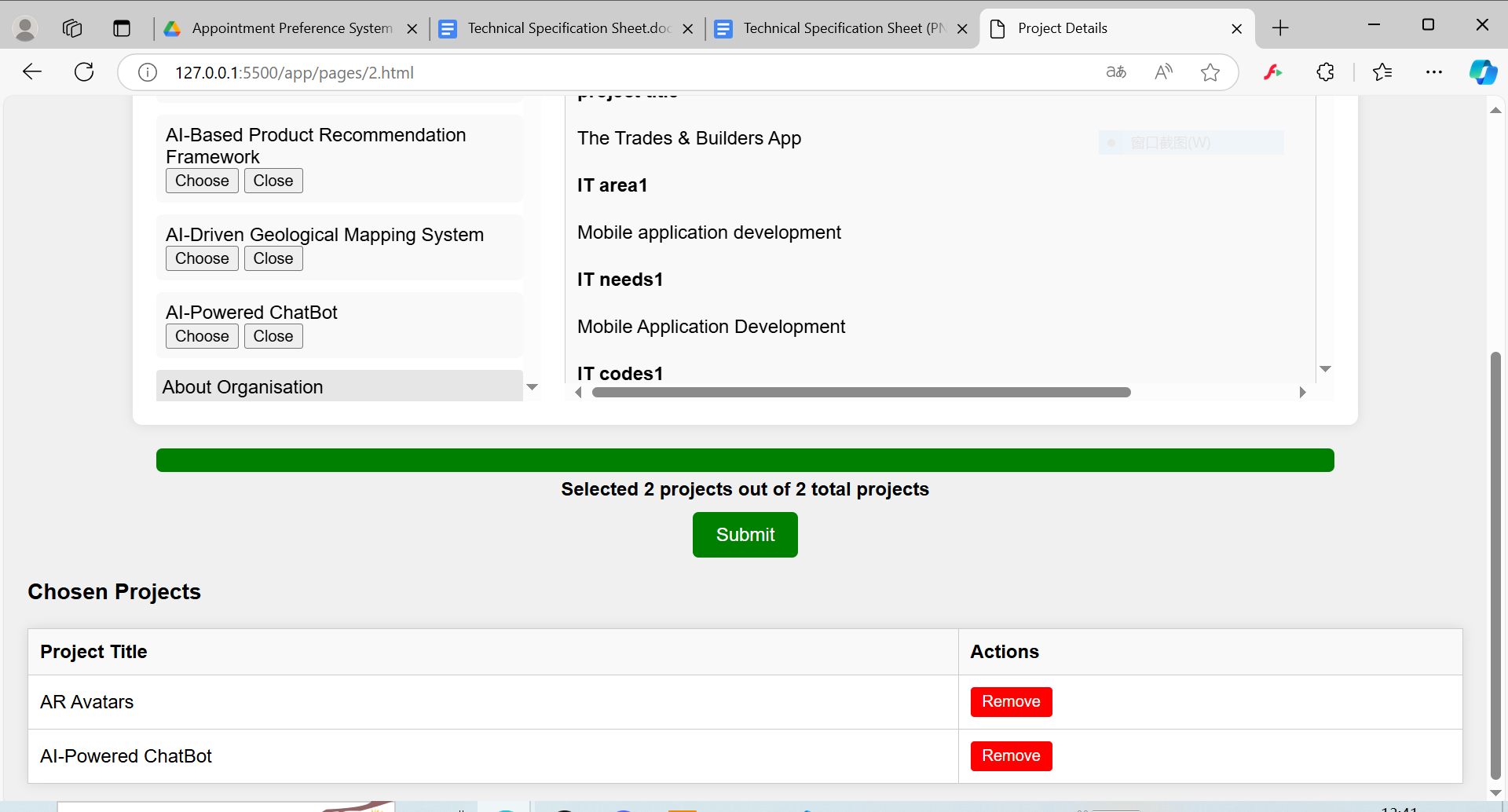
• Clicking the "Choose" button adds the project to the selection list.

• Clicking the "Close" button removes the project from the selection list.



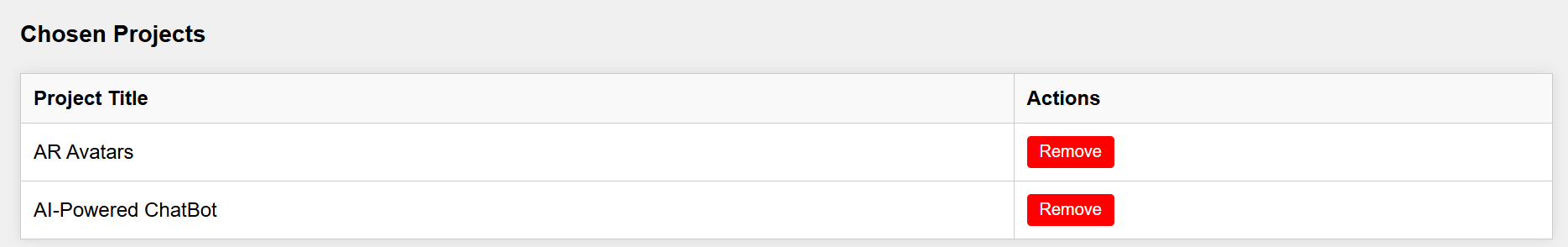
1. progress bar:

When the user selects a project, the progress bar will update in real time, showing the ratio of selected projects to the total number of projects. When the maximum selection limit is reached, the progress bar will turn green, and the "Submit" button will appear.



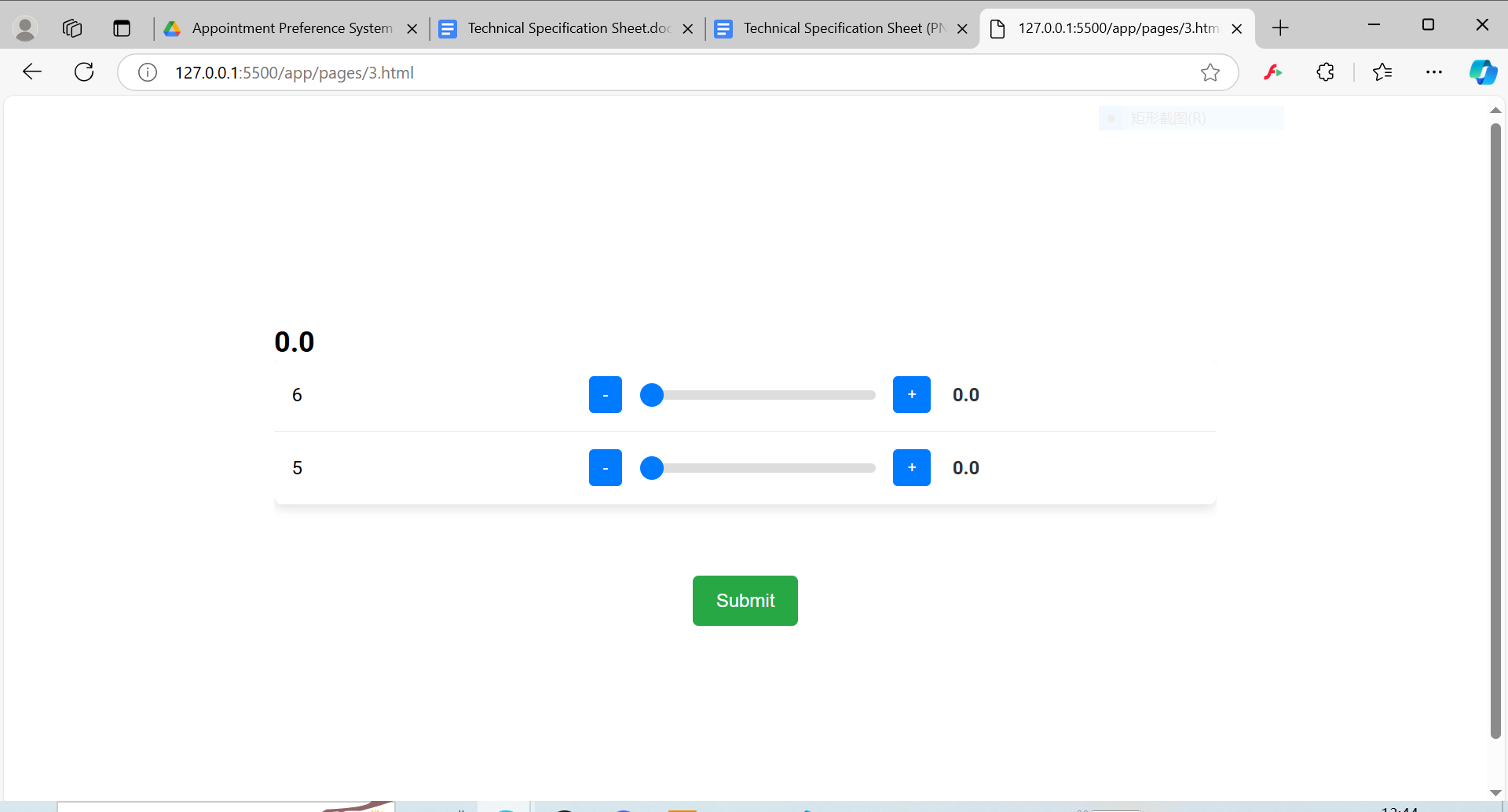
1. selection list:

The projects selected by the user will be displayed in the "Chosen Projects" table, and they can be removed from the selection by clicking the "Remove" button.



1. project scoring:

Users can assign scores to the selected projects, with a total score limit set for all projects.

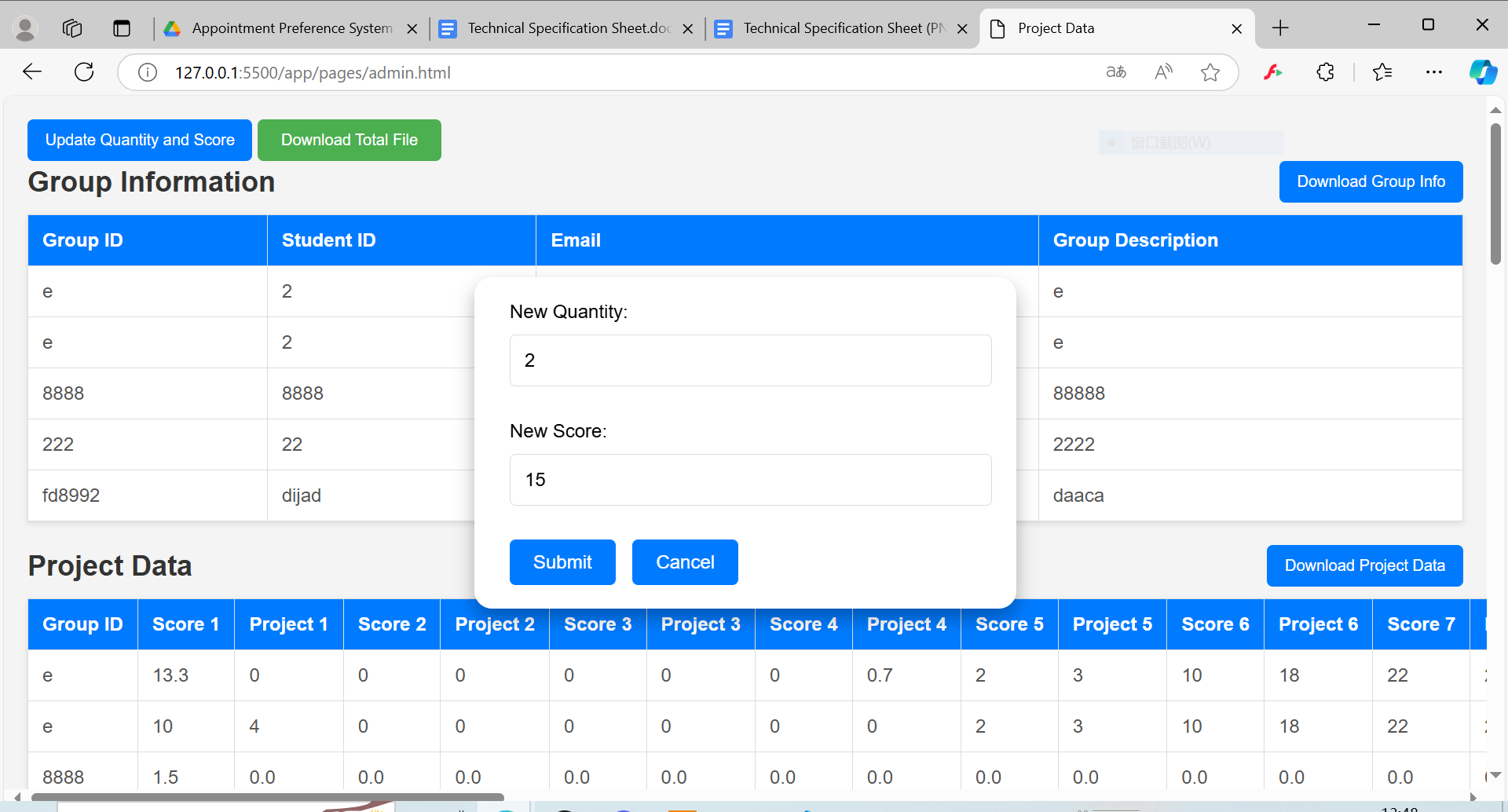


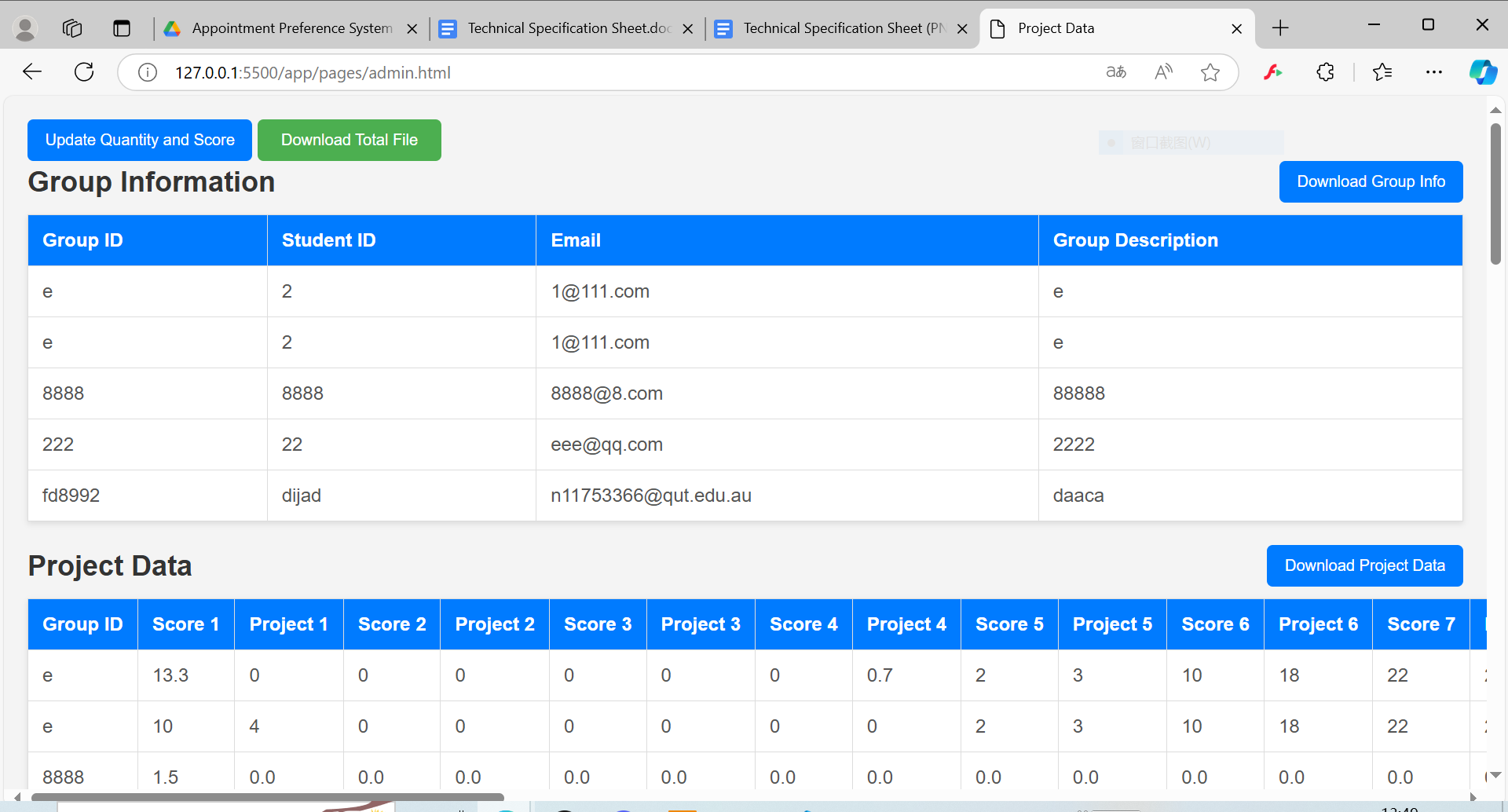
1. admin management:

Admin can limit the number of project preferences users can select and set a total score limit.

**Interface Description**• Update Quantity and Score Button: Clicking this button will open an update form where users can enter new quantities and scores.  
• Download Total File Button: Users can download the total score file.  
• Group Information Table: Displays the group ID, student ID, email, and group description.  
• Project Data Table: Displays detailed information about the projects, with dynamically generated column headers.  
• Selected Projects Table: Displays the projects chosen by each group along with their corresponding scores.

**Function Description**• Load Project Data: When the page loads, the loadProjectData function is called to retrieve all project data from the backend and populate the table. Data is loaded from http://localhost:3001/api/Allproject.  
• Update Quantity and Score: Clicking the "Update Quantity and Score" button opens a form where users can enter new quantities and scores. After submitting the form, the new data is sent via a POST request to http://localhost:3001/api/data.  
• Download Function: Clicking the download button allows users to download the relevant table data as an Excel file, processed using the XLSX library.





1. System troubleshooting

**This system can be deployed on a LAN to ensure that devices within the same network can access the application. It does not have the possibility of being deployed on a server, but it can be used as a preference method test.**

**All teams can submit as many times as they want n**

**The system does not record the latest submission**