

---

Senior Projects Spring 2024

Bard Undergraduate Senior Projects

---

Spring 2024

## GreenGame: A Carbon Emissions Game

Maihan Naimi  
*Bard College*

Follow this and additional works at: [https://digitalcommons.bard.edu/senproj\\_s2024](https://digitalcommons.bard.edu/senproj_s2024)

 Part of the [Educational Technology Commons](#)



This work is licensed under a [Creative Commons Attribution-NonCommercial-No Derivative Works 4.0 License](#).

---

### Recommended Citation

Naimi, Maihan, "GreenGame: A Carbon Emissions Game" (2024). *Senior Projects Spring 2024*. 201.  
[https://digitalcommons.bard.edu/senproj\\_s2024/201](https://digitalcommons.bard.edu/senproj_s2024/201)

This Open Access is brought to you for free and open access by the Bard Undergraduate Senior Projects at Bard Digital Commons. It has been accepted for inclusion in Senior Projects Spring 2024 by an authorized administrator of Bard Digital Commons. For more information, please contact [digitalcommons@bard.edu](mailto:digitalcommons@bard.edu).

GreenGame: A Carbon Emissions Game

Senior Project Submitted to  
The Division of Science, Math, and Computing  
of Bard College

by

Maihan Naimi

Annandale-on-Hudson, New York

May 2024



## Table of Contents

<b>Chapter 1: Introduction.....</b>	<b>1</b>
<b>1.1 Introduction .....</b>	<b>1</b>
<b>Chapter 2: Game Mechanics and Requirements.....</b>	<b>4</b>
<b>2.1 Game Setup.....</b>	<b>4</b>
Registration and Role Assignment.....	5
Setting Desired Reduction .....	5
Regulator’s Role and Setting Carbon tax.....	6
Company Operations.....	8
Next Round.....	9
Game Ended .....	10
<b>2. 2 Game Requirements.....</b>	<b>12</b>
Functional Requirements.....	12
Use Case Diagram .....	13
<b>Chapter 3: Details of Implementation.....</b>	<b>15</b>
<b>3. 1 Frontend.....</b>	<b>15</b>
API Folder .....	16
Pages Folder.....	16
Components Folder .....	17
<b>3.2 Backend.....</b>	<b>23</b>
Models Directory .....	23

Game Routes Directory .....	24
Utils Directory.....	24
<b>3.3 Database.....</b>	<b>26</b>
GameSession Schema.....	27
Users Schema.....	28
<b>3.4 Interaction of System Components .....</b>	<b>29</b>
<b><i>Chapter 4: Results and Evaluation .....</i></b>	<b><i>31</i></b>
<b>4.1 Correctness .....</b>	<b>31</b>
<b>4.2 Speed .....</b>	<b>32</b>
<b>4.3 Security.....</b>	<b>33</b>
<b>4.4 Client Evaluation .....</b>	<b>34</b>
<b><i>Bibliography .....</i></b>	<b><i>36</i></b>

# Chapter 1: Introduction

## 1.1 Introduction

Climate change is one of the most severe threats to the survival of humanity and the planet. The increase in concentrations of greenhouse gasses, in particular carbon dioxide (CO<sub>2</sub>), is altering the global climate change in the planet's atmosphere. Peter R. Orszag, former Congressional Budget Office Director, states that human activities such as industrial processes, deforestation, and the burning of fossil fuels are the primary contributors to these rising greenhouse gas levels (Orszag, 2007). These critical changes take various forms: increased global temperature, rising sea levels, Antarctic ice melting, and severe, frequent weather events like droughts and hurricanes. Also, biodiversity, water resources, agriculture productivity, human health, and safety are all in danger under the shadow of climate change.

The potential economic consequences of climate change are intense. Agriculture-dependent regions deal with concerns about food security since growing seasons have altered and water availability has lessened. Densely populated coastal areas, which are vital economically, are threatened by sea-level rise and storm surges, which can endanger infrastructure and people. Generally, climate change may worsen existing inequalities. Since communities with fewer resources are less able to adapt to environmental changes and are more vulnerable to disasters, such differences call for policies that address not only climate change mitigation but also adaptation strategies that focus on the most vulnerable populations.

Dealing with the crisis of climate change calls for global cooperation and broad policy structures. Reducing emissions is an important step, and according to Orszag (2007), employing incentive-based policies like carbon taxing or cap-and-trade systems can efficiently reduce CO<sub>2</sub> emissions by making it economically advantageous to reduce carbon footprints. These policies take advantage of the market's power to discover the most financially efficient efforts to reduce emissions.

Advances in technology are necessary to address climate change. For instance, replacing traditional sources of energy, such as fossil fuels, with renewable energies, such as solar, wind, and nuclear power, can significantly decrease greenhouse gases. Also, using technologies for capturing and storing carbon provides substantial ways to deal with CO<sub>2</sub> emissions directly by storing CO<sub>2</sub> underground instead of releasing it into the atmosphere.

We must acknowledge that addressing the complicated nature of climate change and its policies is a significant challenge. However, the potential of interactive games to educate and engage people about these policies is important. This report will describe a game used in university classrooms to teach students about climate change remediation policies. This digital version not only makes the game more accessible to a wider audience but also significantly enhances its engagement. Originally, the game existed only as a hands-on, paper-based activity. Lacked the user-friendliness that a digital approach can provide, making its educational content less engaging and time-consuming. The purpose of converting this traditional, paper-based game into a digital format is to educate students about complex environmental policies, such as carbon emissions.

Also, this digitized version will encourage a sense of hope and optimism in our ability to tackle climate change. According to Nekrasova et al. (2022), research shows that integrating environmental simulation games into the early stages of learning can greatly enrich an individual's environmental culture. GreenGame and other environmental digital games enable users to fully engage in simulated decision-making activities that explore various ways of reducing CO<sub>2</sub>. Learning through interactive regulation principles like these can be fun while helping students develop essential skills. These principles offer a hands-on way to teach strategic planning and critical thinking skills.

Sometimes, traditional learning methods fail to fully engage students in understanding the real-world implications of climate change policies. Interactive learning tools like simulation games can effectively bridge this gap in educational methodologies. The original "Pollution Game" (Corrigan, 2011) lays the groundwork for students to investigate pollution control dynamics. To

take the notion further, Gautam Sethi, a Bard College professor, designed a more concentrated version called the Carbon Emissions Game or Carbon Abatement Game. This version puts more emphasis on the process of carbon abatement policies, such as carbon taxes and emissions trading, which provides a more practical approach to comprehending these critical topics.

GreenGame digitizes Sethi's Carbon Emissions Game, moving it from paper to web format. The GreenGame digitizes Sethi's Carbon Emissions Game from a physical, paper-based game to a web format. The transition to a digital platform offers increased accessibility and a more engaging interactive experience. To simulate economic and environmental decision-making processes. The GreenGame uses cutting-edge technology. By engaging students in different roles, the game brings theoretical aspects of carbon abatement strategies to life, which enhances students' and users' engagement and grasp of complex policy implications and impacts in a safe environment.

This paper aims to explore the development and implementation of the GreenGame, which originated from a paper-based game known as the Carbon Emissions Game. GreenGame specifically digitizes the carbon tax game mode of the carbon emission game. The game allows players to take on the roles of regulators, companies, and facilitators. This hands-on approach not only helps break down the theoretical aspects of carbon management and environmental policies but also improves participants' engagement through dynamic interactions and immediate feedback on the consequences of their decisions.

Through this detailed examination, we will outline the core aspects of GreenGame, its technical implementation or development, and its crucial role in educating users on how to navigate global climate change challenges. Additionally, the evaluation section of the paper will assess the effectiveness of GreenGame as a learning tool by taking into account the client's feedback. Through this detailed examination, the paper aims to showcase how digital simulations like GreenGame can play a crucial role in environmental education, equipping users with the knowledge and skills necessary to navigate the complexities of global climate challenges.



## Chapter 2: Game Mechanics and Requirements

### 2.1 Game Setup

The game begins with the users selecting the game mode in the homepage (Figure 2.1): either a trading game, a carbon tax game, or command and control game. For the purposes of this paper, we will focus exclusively on the carbon tax game. In the carbon tax scenario, the game is centered around managing emissions through the strategic imposition of taxes. This setup allows participants to engage in decision-making processes that mirror real-world economic and environmental strategies for managing carbon emissions. Once the game mode is selected, the facilitator determines the number of companies that will participate (Figure 2.2) in the game session, ranging from two to twelve.

### Welcome to the Carbon Game



*Figure 2.1 List of game modes available to play in the GreenGame*

Number of Companies

*Figure 2.2 Select number of companies to play in the GreenGame*

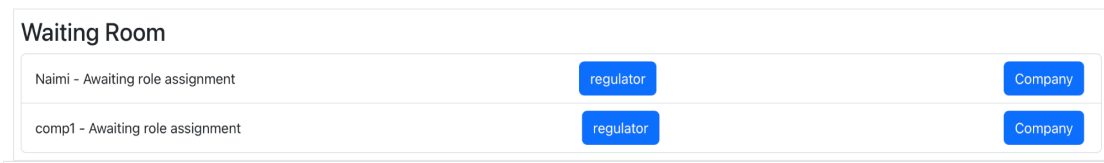
### *Registration and Role Assignment*

After receiving a 6-character unique code (Figure 2.3) from the facilitator, players proceed to register for the game. This code ensures that all participants enter the same game session. Once participants enter their usernames along with the invite code provided, they move to a waiting room, where the facilitator can view each participant, ensuring everyone is present before proceeding.



*Figure 1.3 The location of invite code in GreenGame (facilitator and regulator views).*

Once all participants are registered and gathered in the waiting room (Figure 2.4), the facilitator assigns roles within the game: one player is designated as the regulator, tasked with setting and adjusts the carbon tax based on game dynamics and emission data. The other players are assigned roles as companies, each responsible for managing their emissions and responding strategically to the carbon tax set by the regulator.

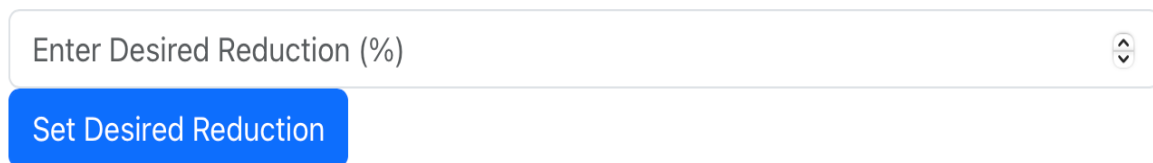


*Figure 2.4 Waiting room and role assignment (facilitator view).*

### *Setting Desired Reduction*

The facilitator starts by setting a desired percentage (Figure 2.5) for reducing carbon emissions based on the initial emissions. This decision significantly influences the game's strategic framework. For instance, if the combined initial emissions of all participating companies total

1312 tons per year, and the facilitator targets a 20% reduction, the game system automatically computes this desired reduction. This automated calculation results in a target abatement of 262 tons per year, thereby reducing the post-regulation emissions to 1050 tons per year. The facilitator's establishment of the target abatement not only delineates the objectives for the companies but also sets the stage for the regulator. The regulator's role is to set a carbon tax that motivates all the companies to work together to hit the emission reduction target.

The image shows a user interface element for setting a target. It consists of a light gray rounded rectangular input field with the placeholder text "Enter Desired Reduction (%)". To the right of the input field is a small vertical scroll bar icon. Below the input field is a solid blue rectangular button with the white text "Set Desired Reduction".

*Figure 2.5 Setting desired reduction to establish emission reduction target (facilitator view).*

### *Regulator's Role and Setting Carbon tax*

At the start, the regulator gets all the key information from the database. This includes the total Initial Emissions and the Target Abatement, which set the goals for how much the companies need to reduce their emissions. This data sets the stage for the regulator to decide upon the optimal tax rate that will assist in meeting the emissions reduction goal.

The regulator is presented with "noisy data" (Figure 2.7) for each company, which shows each company's potential to reduce emissions and the associated costs. However, this data isn't clear-cut—it includes uncertainties that simulate the real-world challenge of making decisions with incomplete information. For example, if the original data (Figure 2.6) suggests a company can reduce emissions by a certain amount at a specific cost, the noisy data might present this potential as slightly higher or lower, and the costs as more or less expensive. This variance requires the regulator to employ strategic thinking to set an effective Carbon Tax rate that motivates companies to meet the Target Abatement.

Alongside the noisy data, the regulator also receives Marginal Abatement Cost (MAC) graphs (Figure 2.9), which shows the relationship between potential emission reductions and their associated costs. These graphs illustrate the costs or savings associated with different emission reduction strategies and indicate the potential emission reductions each strategy could achieve. These visual graphs help the regulator understand the financial and environmental trade-offs of various options. For example, a MAC graph might show that investing in solar energy has a higher upfront cost but leads to significant reductions in emissions. This contrasts with a strategy like switching to High penetration wind, which might have a lower initial cost but offers smaller reductions in emissions. By examining these graphs, the regulator can decide to set a higher carbon tax rate to encourage more substantial investments in solar energy, aligning with a goal to maximize emission reductions efficiently.

Company 1 Data

Option	Abatement Potential (tons per year)	Average Cost (\$ per ton)
EFI	43	\$11
DSM	61	\$17
RCP	109	\$23
SGB	28	\$37
GLG	36	\$38
LPW	108	\$50
GEP	34	\$61
NEP	68	\$120
CSP	39	\$360
GPR	26	\$473

Figure 2.6 Original Data (facilitator and company views)

Company 1 Data

Option	Abatement Potential (tons/year)	Total Cost (\$)	Average Cost (\$/ton)
EFI	47	\$473	\$10
DSM	63	\$1037	\$16
RCP	97	\$2507	\$26
GLG	38	\$1368	\$36
SGB	25	\$1036	\$41
LPW	116	\$5400	\$47
GEP	37	\$2074	\$56
NEP	70	\$8160	\$117
CSP	36	\$14040	\$390
GPR	28	\$12298	\$439

Figure 2.7: Noisy Data (regulator view)

To influence how companies reduce their emissions, the regulator can set the Carbon Tax (Figure 2.10) in two main ways. The first is to manually set the Carbon Tax by analyzing the noisy data and Marginal Abatement Cost (MAC) graphs. With this method, the regulator can determine a tax rate that they believe will effectively encourage companies to achieve the Target Abatement. This approach requires a deep understanding of how different tax rates might influence company behaviors and emission levels. Alternatively, the regulator can opt for a more straightforward method to automatically calculate the optimal tax rate. Which is accessible via a button called “AUTO” in the game interface. This function automatically suggests a Carbon Tax rate based on the noisy data, simplifying the decision-making process by calculating the most

effective tax rate to achieve the desired emission reductions. This tool is particularly useful for regulators who may not feel confident in their ability to manually analyze the data and assess the impact of various tax rates.

Graph for Company 1

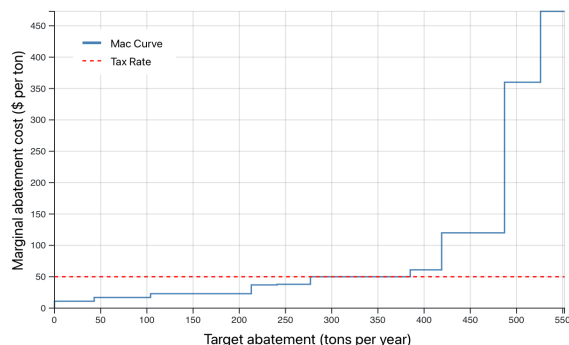


Figure 2.8: Mac graph of original data for companies and facilitator (the dotted line represent tax rate)

Company 1 Graph

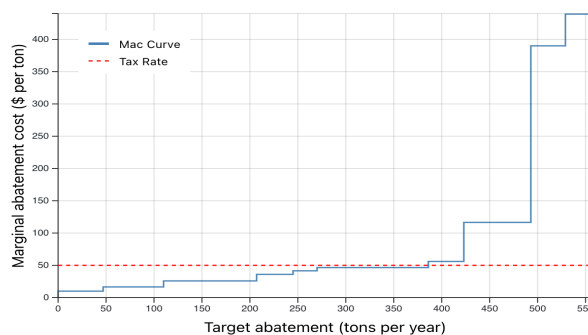


Figure 2.9 Mac graph of noisy data for regulator (the dotted line represent tax rate)

### Company Operations

Once the regulator sets the carbon tax, companies start their part of the game. Each company, having been assigned their role, receives specific information about their emission levels and the carbon tax rate determined by the regulator. This data is crucial as it helps them plan how to reduce emissions effectively. Alongside this, the companies in the game are able to see their Marginal Abatement Cost (MAC) graphs (Figure 2.8), which help them understand the costs or savings associated with various emission reduction strategies and the potential volume of emissions that could be reduced if these strategies are implemented.

With this information in hand, companies are further supported by access to "optimal values." (Figure 2.10). These are strategic hints that become available once the carbon tax is set, illustrating the most cost-effective methods for reducing emissions. Companies can use these hints to guide their strategies or decide based on their own insights into the tax implications. This interactive feature of the game makes it engaging and user-friendly, offering companies insights into cost-effective strategies to reduce emissions, the associated costs, and the expected

outcomes in terms of emission reduction and tax liabilities. Companies have the choice to either follow these recommendations or develop their strategies based on personal analysis.

Hide Optimal Values

**Optimal Values**

Optimal Abatement: 385 tons per year

Optimal Abatement Cost: \$11821 \$ per year

Optimal Emissions: 167 tons per year

Optimal Tax Payment: \$8350 \$ per year

*Figure 2.10 - Optimal Values*

### *Next Round*

After the first round of actions in the GreenGame, where companies submit their actions in response to the carbon tax, both the facilitator and the regulator receive the round details (Figure 2.11) to assess progress. This assessment is crucial as it determines the need for adjustments in the next round. If the companies have not met the target emissions, and instead paid taxes on their excess emissions, it suggests that the initial carbon tax rate set by the regulator might have

Round Details				
Round 1				
In the previous round, the carbon tax of \$50 per ton generated a combined abatement of 244 tons per year. Please set the carbon tax for this new round above.				
Industry as a whole is abating 244 tons/year				
Industry as a whole is emitting 428 tons/year				
Company	Abatement (tons per year)	Abatement Cost (\$)	Emissions (tons per year)	Taxes Paid (\$)
comp1	244	\$5,108	428	\$21,400
<b>Total</b>	<b>244</b>	<b>\$5,108</b>	<b>428</b>	<b>\$21,400</b>

*Figure 2.11 Round detail (regulator view).*

been too low to effectively encourage sufficient abatement actions. This scenario triggers the facilitator to initiate the next round of the game.

In this subsequent round, the regulator, now equipped with a better understanding of the companies' abatement potentials and their previous responses to the tax, reassesses the carbon tax rate. The decision involves a detailed comparison of the actual abatement achieved against the target abatement set at the game's start. If the actual abatement falls below the target, indicating the tax was too low, the regulator may decide to increase the tax rate. Conversely, if the abatement was higher than the target, suggesting the tax was possibly too high, the regulator might lower the tax to balance the economic burden on the companies.

The regulator's decision on how much to adjust the tax is based on their decision-making and strategic analysis of the round details. Once a new carbon tax rate is determined, this information is communicated to the companies, enabling them to adjust their strategies for the new round. Simultaneously, all the decisions and adjustments made by the regulator are also shared with the facilitator. This allows the facilitator to monitor the overall progress and effectiveness of the game mechanics, ensuring that the session progresses towards its environmental goals efficiently. This cycle of action, review, and adjustment can continue for multiple rounds, adapting the carbon tax as needed to guide the companies towards achieving the desired level of emissions reduction.

### *Game Ended*

As the game progresses through various rounds, each involving strategic decisions and adjustments to the carbon tax rate by the regulator, the ultimate goal is to reach the target abatement set at the beginning of the session. The game concludes when this target is met, marking the end of the game session. At this point, the game announces that the "Target Abatement has been Reached" and provides a summary of the game's outcomes to all participants (Figure 2.12). This summary includes key data points such as the total initial emissions, which in this scenario are 1312 tons per year. It also details the desired reduction percentage that was aimed for, the target abatement of 262 tons per year that has been

successfully achieved, the post-regulation emissions which are now reduced to 1050 tons per year, and the number of rounds it took to reach this goal, which in this case is 2 rounds. Additionally, the final carbon tax rate applied in the last round, \$56, is disclosed.

This end-of-game report not only serves as a closure to the current game session but also acts as a crucial feedback tool. It allows participants to review how their decisions impacted the game's outcomes and understand the effectiveness of the strategies they employed. This reflective phase is essential for learning and deepening the understanding of environmental policy dynamics.



Figure 2.12 end of the game report and action details (facilitator, regulator, and companies' views)



## 2.2 Game Requirements

As demonstrated, GreenGame works through a series of steps which starts with the game setup by the facilitator, moves to role assignments and target abatement setup, and ends with decision-making by the regulator and responses from companies. This setup not only mimics real-world environmental policy planning on carbon taxation but also shows the complex details and the strong support needed from the game's architecture.

To make sure the game works properly, we need a solid foundation of game requirements. These functional requirements are designed to ensure that every part of the game meets educational and simulation goals and also provides a smooth, user-friendly experience that keeps participants engaged.

### *Functional Requirements*

Functional requirements detail specific actions and operations the game system must execute to function correctly (Sommerville, 2011, p. 85). In GreenGame, these include enabling the facilitator to set up game parameters such as the number of companies and the type of game (carbon tax or trading). These requirements include the registration process where players sign up and enter the game using a unique invite code. This code ensures they are correctly assigned to their roles as either regulators or companies. Functional requirements also ensure that the game can dynamically generate and display real-time data such as target emissions and carbon tax, which enables the regulator to make decisions based on current game conditions and company responses. Further elaborating on the essential requirements, GreenGame ensures that:

1. Users should have the ability to log in or create an account to access the game.
2. Users should have the ability to create a new game session or join an existing one.
3. The system should have the ability to automatically assign the role of facilitator to the user who creates the game.

4. The facilitator should have the ability to assign roles of regulators and companies to other players.
5. The system should have the ability to dynamically generate real-time data based on the game's initial settings.
6. The facilitator should have the ability to set a target for emissions reduction
7. The regulator should have the ability to set the carbon tax rate.
8. The facilitator should have the ability to view all abatement options and their costs for each company.
9. The system should have the ability to provide the regulator with a 'noisy' version of data.
10. The facilitator and regulator should have the ability to see the cumulative results of all actions taken by companies in terms of emissions reduction.
11. Each company should have the ability to see its specific options for reducing emissions.
12. Companies should have the ability to choose different strategies for emission reduction or decide to pay taxes instead.
13. The system should have the ability to automatically calculate the results of the game rounds based on the actions and strategies implemented by the players.

### *Use Case Diagram*

To visualize the game's requirements and bridge the gap between conceptual requirements and practical application, a use case diagram plays a crucial role. The use case diagram shown in Figure 2.13 provides a schematic representation of the essential interactions between the users and the system within the game. In these use cases facilitator, regulator, and company participants are identified as key roles. Each role interacts with various processes to drive the GreenGame forward. The Facilitator initiates the game by creating a new session. They set the

desired emissions reduction targets to guide the simulation. The Regulator, important in influencing gameplay dynamics, is responsible for two critical actions: setting the carbon tax rate and reviewing game data which reflects the outcomes of each round. The company participant joins the game session and engages with the system by submitting their strategic actions and decisions. These use cases collectively define the user requirements. They dictate how each role interacts with the system to facilitate a smooth gaming experience.

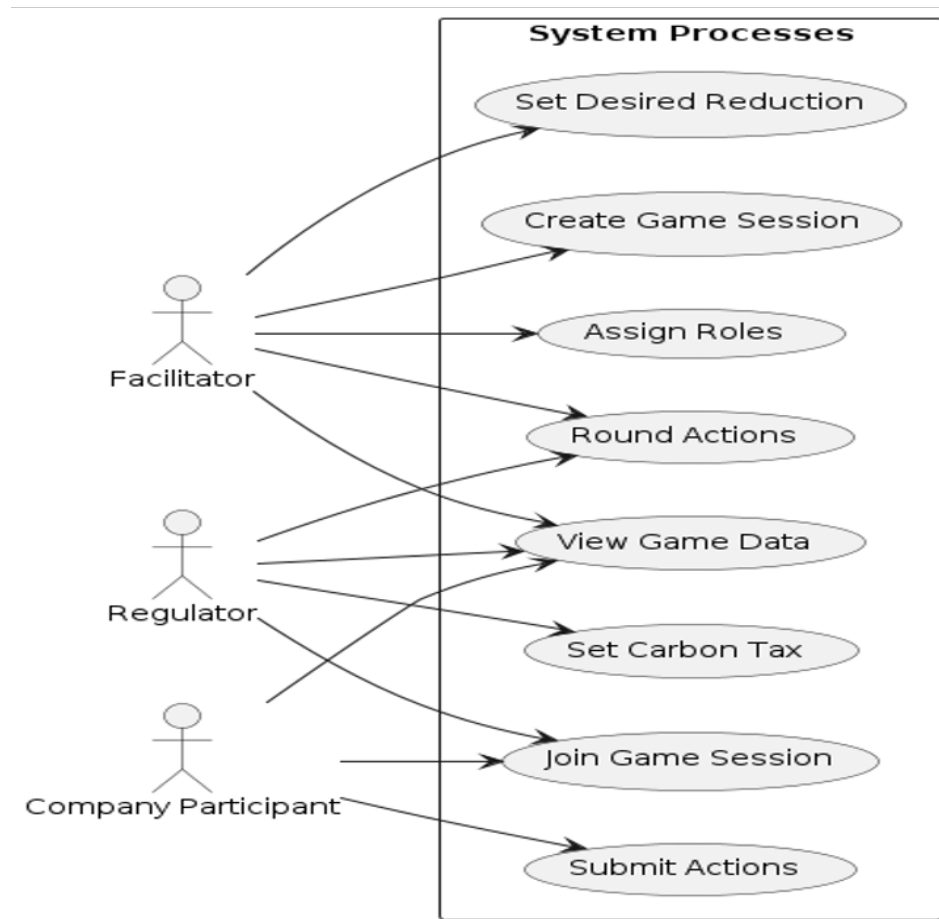


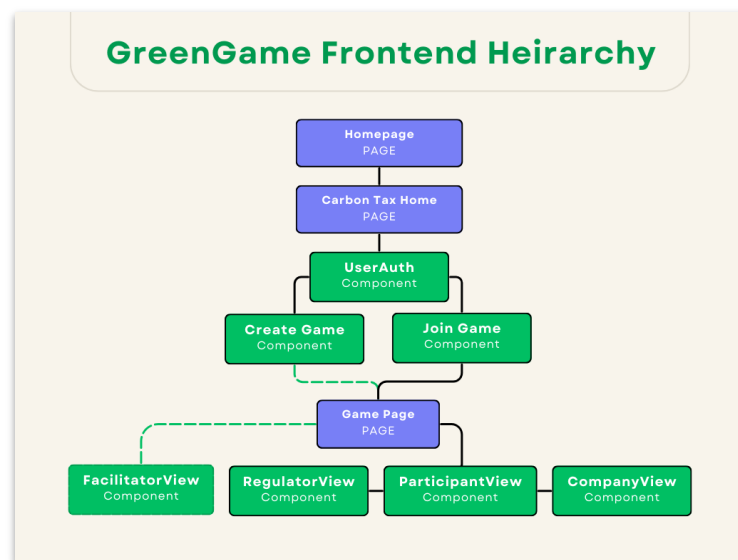
Figure 2.13 Use case diagrams for the GreenGame

## Chapter 3: Details of Implementation

In this chapter, we delve into the technical specifics of the GreenGame project, focusing on the three core components that form the backbone of any modern web application: the backend, the frontend, and the database. Each component serves a distinct purpose and interacts with the others to provide a robust and efficient system. A key goal of this chapter is to demonstrate how these components are designed and integrated to fulfill the specific game requirements. This is to make sure that everything works well together to support the game's features.

### 3.1 Frontend

The frontend of an application refers to the client-side interface and experience that users interact with directly. It is built using technologies that run on the user's browser, providing the graphical user interface (GUI) through which users can interact with the application. For GreenGame, the frontend is developed using React, a popular JavaScript library designed for building fast and reactive user interfaces. This component is crucial for ensuring that the game is visually appealing, user-friendly, and responsive to user inputs, thereby enhancing the overall user experience. Figure 3.1 illustrates the hierarchy of the GreenGame frontend, which demonstrates how these components work together.



*Figure 3.1* - This diagram shows the hierarchical structure of the GreenGame's frontend. It shows the navigation flow from the Homepage to the different components. Dashed lines indicate the path a user follows when creating a game, leading to the FacilitatorView, while solid lines indicate the path for users joining an existing game session, leading to their respective views as Regulator, Participant, or Company.

The frontend is structured into three main folders: API, Pages, and Components, that organize and manage different aspects of the application:

### *API Folder*

This folder contains modular code that allows for quicker and easier communication between the frontend and the backend. It includes **api.js** and **storage.js**, which handle the fetching of data from the server and the management of local data storage, respectively. The **api.js** file is tasked with making API calls, allowing the frontend to send and receive data from the backend. On the other hand, **storage.js** manages local storage and session storage, ensuring that user data and game state are persistently stored across sessions.

### *Pages Folder*

This part of the frontend architecture deals with the various pages that users navigate through within the game. It includes files such as **Homepage**, **Carbon Tax Home**, and **Game Page**. Each page is designed to carry out specific functions and interactions:

- **HomePage.js** serves as the primary landing page for the GreenGame. It is designed to welcome users and guide them towards different game modes such as Carbon Tax, Carbon Trading, and Command and Control. The page utilizes the **useNavigate** hook from React Router, which is a navigation library within the React ecosystem. React hooks are features that allow the use of state and other React functionalities in functional components. The **useNavigate** hook specifically enables the application to change views based on user interactions, for instance moving from homepage to carbon tax page, without the need for manual URL adjustments in the browser. This setup enhances user experience by providing straightforward access to various game pages and modules.
- **CarbonTaxHome.js** serves as the entry point for handling carbon tax-related game activities. It includes user authentication through the **UserAuth** component of the game, allowing players to log in or register using their usernames. Once authenticated, users can

either create new game sessions or join existing ones using the **CreateGame** and **JoinGame** components, respectively.

- **GamePage.js** acts as the central hub for active game sessions within the carbon tax segment of the GreenGame. This page dynamically displays content based on the user's role—whether they are facilitators, participants, regulators, or companies. It handles state management, data fetching, and navigation with the help of various hooks and helper functions from React. The page fetches and updates session data regularly, ensuring that all users see the most current game state. Based on the game's status and the user's role, it adjusts the displayed content, providing appropriate views for interaction and management of the game session. If the game session ends, it shows a summary of outcomes.

### *Components Folder*

The GreenGame frontend architecture is designed to make sure users have a great experience while playing. At the core of this design is the Components folder, which includes specific React components responsible for different parts of the game. This folder includes components like `UserAuth`, `JoinGame`, `CreateGame`, `GameData`, `CompanyView`, `FacilitatorView`, `RegulatorView`, `ParticipantView`, `MacGraph`, `RoundData`, `WaitingRoom`, and `OptimalCalculations`. Each component focuses on certain tasks such as user authentication, session management, showing data, and handling live game interactions. By dividing the frontend into these specific components, the application keeps the development, testing, and updates organized and clear. This modular approach makes the code easier to manage and understand. We next explore the specific roles and functions of each component within this module.

- The **CreateGame** component provides the functionality for creating new game sessions. It allows users, typically facilitators, to set up a game by specifying the number of participating companies. The component uses an API call to create a game session on the backend and stores this session data locally using the game's storage API, which includes storing the game session, the session ID, and the user's role as 'facilitator'. Once the game

session is successfully created, it triggers an **onGameCreated** event to notify other components of the new game's initiation. This process is central to starting a new game and setting the stage for all subsequent in-game activities and interactions.

- The **JoinGame** component enables users to enter an existing game session by providing a session ID, which is a 6-digit unique invite code. This component interacts with the backend through an API call to join a specific game session, ensuring the user's ID is correctly associated with the session. Upon successful joining, it updates the local session storage with the game session details, assigns the user role as 'participant', and triggers the **onGameJoined** event, a callback function that handles further actions such as navigating to the game page. This functionality is for players who wish to participate in ongoing game sessions and play the game as regulators or companies.
- The **WaitingRoom** component acts as a pre-game lobby for the GreenGame, where participants wait before the game starts. It displays a list of all participants, along with their usernames and roles, which could be 'Regulator', 'Company', or 'Awaiting role assignment'. This component uses API calls to fetch and update participant details, storing them in session storage to maintain consistency across the game session. Additionally, facilitators can assign roles to participants directly from this component, ensuring everyone is properly positioned before the game commences. This functionality helps streamline the setup process and prepares participants for the game.
- The **GameData** component acts as a central hub in the game, managing essential data and interactions for facilitators and companies. It starts by pulling the initial game session details from local storage, which preserves user data and settings between sessions. This component controls the game's core dynamics—tracking user roles, current game round, emissions details, and calculations related to carbon tax and abatement strategies.
  - For facilitators, **GameData** provides a comprehensive overview of the game's progress, including total emissions, desired reduction targets, and the outcome of various rounds.

- Companies interact with this component differently; they receive tailored information relevant to their specific circumstances, such as their own emissions data and potential abatement options. Companies can input their chosen strategies directly through this interface, which the component then processes to update game state—calculating costs, potential savings, and overall compliance with set regulations.
- Also, the component renders detailed data for each company, including their abatement options and associated costs, neatly organized in tables. This allows the Facilitator, Companies and Regulator to view and assess the impact of abatement strategies in a structured format. For deeper analysis, the component integrates graphical representations of data through the MacGraph component -explained later in the chapter.
- The **GameData** component ensures that all game data is up-to-date by reacting to changes in game sessions and user interactions. This involves recalculating emissions and abatements as new data becomes available or as players submit new actions. By maintaining a constant update loop, **GameData** ensures that all participants have access to the latest information.
- The **GameDataRegulator** component is specifically designed for the regulator's role in the GreenGame. It focuses on monitoring and managing carbon tax rates and generating noisy data to add complexity and realism to the game. This component retrieves the initial game session data from local storage, so it ensures the regulator can continue their work with the most recent game state, which includes details on emissions and rounds.

The component features a unique function for generating noisy data, which introduces variability into the abatement potential and cost of options available to companies. This is achieved using predefined standard deviations for each abatement option. This noise is added to the original data by modifying the abatement potential and costs within a defined range, ensuring that the game remains challenging and that strategies cannot be too easily predicted or optimized without consideration of potential variations. Below is the pseudocode



for the noisy data generation, that will provide a clearer understanding of how the function operates.

```

IF noise already added THEN
  STOP PROCESS
ENDIF

FOR each company IN session data
  FOR each abatement option IN company's options
    RETRIEVE noise settings for option
    IF noise settings exist THEN
      GENERATE noise for abatement potential within defined standard deviation range
      GENERATE noise for cost within defined standard deviation range
      ADJUST abatement potential by noise, ensure it is equal to or more than 1
      ADJUST cost by noise, ensure it is equal to or more than 1
      CALCULATE new total cost from the product of adjusted abatement potential and cost
    ENDIF
  ENDFOR
ENDFOR

MARK noise as added
RETURN updated session data

```

- Another significant functionality of the **GameDataRegulator** component is the ability to dynamically calculate and set an optimal carbon tax based on the current game data. This involves gathering the average cost of all the abatement options of all companies into a single list which is then sorted in ascending order and the median cost is then selected and set as the new carbon tax rate.
- Additionally, the regulator has the capability to manually adjust the carbon tax via an input field. Also, this component renders detailed data for each company, including their abatement options and associated costs organized in tables using the noisy data. This allows the regulator to view and assess the impact of noise and the efficacy of abatement strategies in a structured format. For deeper

analysis, the component integrates graphical representations of data through the **MacGraph**, enhancing the visual feedback on the economic and environmental implications of the game's dynamics

- The **FacilitatorView** component is crucial for overseeing and managing game sessions. It enables the facilitator to control important gameplay aspects, such as setting the desired reduction in emissions. Facilitators can input a percentage reduction target, which the component uses to calculate and set new target emissions for the game session. This update is achieved through an API call, ensuring the game reflects the new targets accurately. Additionally, facilitators can initiate the next round of the game.
- The **CompanyView** component is specifically designed for company participants within the game, providing them with a user interface that displays vital game data relevant to their role. This component dynamically renders information such as carbon tax and their target emissions that are crucial for the companies to make informed decisions during the game. It interacts with the game's session data to refresh the interface as the game progresses and new taxes are enforced by the regulator, ensuring that the company players are always aware of the current game conditions. Additionally, the **CompanyView** incorporates elements like the **GameData** component to visually present this data, and it handles user-specific interactions, ensuring that each company sees data pertinent to their own situation in the game. This setup helps maintain an engaging and informative experience for each participant.
- The **RegulatorView** component in the game serves as the main interface for the regulator role. It provides a comprehensive dashboard that displays crucial game information and updates in real-time. The component includes **GameDataRegulator**, which facilitates the management of carbon tax rates and the observation of company emissions data. Additionally, it integrates **RoundData** to display round-specific details.
- The **MacGraph** component in the game visualizes the Marginal Abatement Cost (MAC) Curve, which helps players understand the cost-effectiveness of different emission reduction strategies. Using D3.js library for dynamic graphing, it sorts abatement options

by cost-effectiveness, calculates cumulative potentials and costs, and updates these visuals in real-time to reflect game changes. The graph shows the relationship between potential emission reductions and their associated costs, enhancing strategic decision-making by illustrating the current carbon tax rate compared to the costs of abatement options.

- The **ParticipantView** component provides a user interface for players in the GreenGame. It integrates the WaitingRoom component, ensuring players are held in a preparatory area before the game starts. Players can view their session ID, which they can copy to the clipboard for easy sharing with other players. This component also dynamically updates to ensure it reflects the current state of the game session, pulling data as needed from the server to keep the participant's view accurate and up-to-date.
- The **RoundData** component provides detailed insights into the actions taken by companies during each game round. This component uses data from the game session to calculate and present these metrics, organizing them by company and round. Each round's data includes the total emissions abated, the cost of these abatements, and the carbon tax implications, all structured within an accordion UI that allows users to expand or collapse the detailed statistics for each round. This setup helps regulator and facilitator track performance. The RoundData also integrates an alert system that provides feedback on the effectiveness of the carbon tax rate from the previous round, guiding the adjustment of future tax rates.
- The **UserAuth** component is a straightforward React component designed for user authentication within an application. It provides a single text input field for users to enter their username and a button to either log in or sign up.

## 3.2 Backend

While the frontend serves as the face of the application, the backend handles the core logic and data management. The backend of an application refers to the server-side logic and operations that process data and handle interactions with the database. It is responsible for managing the application's core functionalities and integrates with the frontend to deliver a unified experience. For GreenGame, the backend is developed using Node.js, a runtime environment that allows JavaScript to run on the server side, and Express.js, a framework that simplifies the setup of servers and routes, enhancing development efficiency. Node.js handles multiple connections simultaneously, which is ideal for real-time applications such as GreenGame. The backend architecture shown in **Figure 3.2** has three key directories and a server configuration file. The

routes directory handles HTTP requests, the model's directory defines database schemas, and the utils directory contains utility functions that support various backend operations. In the backend, the server.js file configures the server settings, initializes middleware, and specifies routes, which serves as the entry point for the server-side application. This structure ensures that the backend is

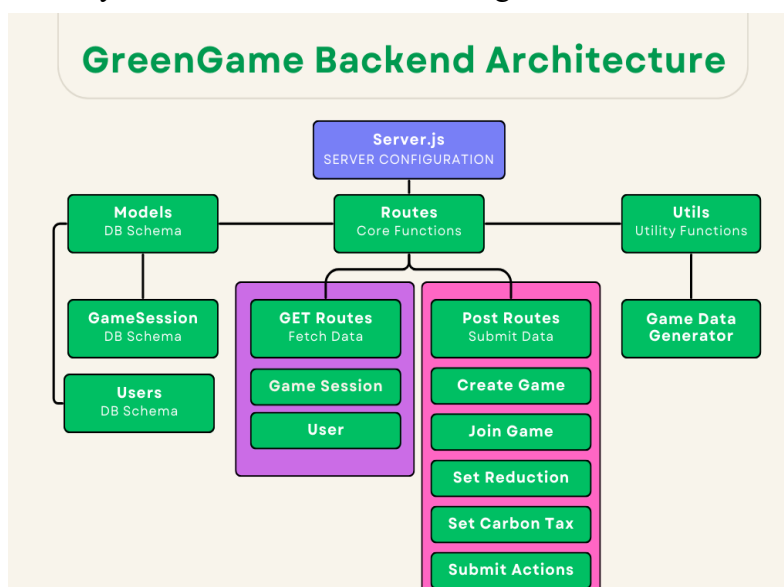


Figure 3.2 GreenGame backend architecture.

not only well-organized but also modular, making it easier to maintain and scale.

### *Models Directory*

In the GreenGame backend, the model's directory includes data schemas essential for interacting with the database to manage game states and user data. This directory contains two primary files: **GameSession.js** and **User.js**, both utilizing Mongoose, a Node.js library that provides object

modeling for MongoDB; which is a NoSQL database that adopts a document-oriented data model, that is highly suitable for handling large volumes of structured and unstructured data. It stores data in flexible documents where fields can vary from one document to another. and the data structure can be changed over time.

### *Game Routes Directory*

The **GameRoutes** in the backend are crucial for managing HTTP requests, methods used by web browsers and clients to communicate with servers. The backend handles two primary types of HTTP requests: GET, used to retrieve data without altering the server's state, and POST, used to submit data that can change the server's state. In the game the `GameRoutes.js` processes requests related to game sessions, user interactions, and game settings, by using the `Express.js` framework.

`GameRoutes` defines nine POST routes: these routes carry out specific instructions such as allowing users to create new game sessions and join existing game sessions. Also, allowing facilitators to assign roles such as regulator or company to the participants within a session, set the desired target emissions reduction for the game. It allows regulator to set or update carbon tax and allow companies to submit their abatement and emissions actions. Additionally, `GameRoutes` also defines two GET routes: these routes retrieve game session details and for user details. These routes interact robustly with the MongoDB database through Mongoose models such as `GameSession` and `User`, ensuring efficient and secure data interactions. This routing structure enhances `GreenGame`'s scalability and maintainability.

### *Utils Directory*

The **utils GameDataGenerator** module in the `GreenGame` backend plays a vital role in generating dynamic and statistically modeled game data to enhance gameplay realism. This module leverages the **d3-random** library, a part of the D3 (Data-Driven Documents) JavaScript

library that specializes in generating random numbers following specific distributions. The **generateNormalData** function in this module is designed to create data that follows a normal distribution, where values cluster around a central mean (average). This function takes two parameters: the mean, which is the central value most data points will cluster around, and the standard deviation, which measures how much the values deviate from the mean. This function is used in the GreenGame for generating realistic abatement potential and cost data for different abatement options. utilizing the **generateNormalData** function, the **generateAbatementOptions** function in GreenGame is designed to create a diverse set of abatement options for companies participating in the game. An abatement option refers to a specific method or technology that can reduce emissions of pollutants, like carbon dioxide.

### Game Data Generation Pseudocode

```

FUNCTION generateAbatementOptions(numberOfCompanies)
  DEFINE optionDetails WITH predefined means and standard deviations for each option

  SHUFFLE optionDetails TO ensure variety
  SELECT first 10 options from shuffled optionDetails

  INITIALIZE results array
  FOR EACH option IN selected options
    CALCULATE adjustedAbatementMean AS option.abatementMean DIVIDED BY numberOfCompanies
    CALCULATE adjustedAbatementSD AS option.abatementSD DIVIDED BY numberOfCompanies
    CALCULATE adjustedCostMean AS option.costMean DIVIDED BY numberOfCompanies
    CALCULATE adjustedCostSD AS option.costSD DIVIDED BY SQRT(numberOfCompanies)

    GENERATE abatementPotential USING generateNormalData WITH adjustedAbatementMean,
    adjustedAbatementSD

    GENERATE costPerTon USING generateNormalData WITH adjustedCostMean, adjustedCostSD
    COMPUTE cost AS costPerTon MULTIPLIED BY abatementPotential

    CREATE option result WITH optionName, abatementPotential, AND rounded cost
    ADD option result TO results array
  END FOR

  RETURN results array

```

## END FUNCTION

Initially, the function establishes an object of potential abatement options, each defined by typical values for how much pollution they can reduce (abatement potential) and their associated costs. This object represents various technologies or strategies that might be used in the real world to cut emissions, such as energy efficiency improvements or renewable energy installations (here I have a source given by Gautam for all the option names used in this game) The function starts by randomly selecting a subset of these options to ensure each game session offers different strategies. It then adjusts the potential and costs of these options based on the number of companies playing. This adjustment ensures the options are scaled appropriately for the game's size. For each selected option, **generateNormalData** is employed to generate values that follow a normal distribution, based on the adjusted means and standard deviations. Each option's total cost is calculated by multiplying its randomly generated abatement potential by its cost per unit.

### 3.3 Database

While the GreenGame's frontend provides an engaging interface for users, the database serves as its foundational support, actively storing and organizing essential game data. In web applications, databases manage data storage, retrieval, and handling. This ensures that information persists across user sessions and interactions. For GreenGame MongoDB, a NoSQL database known for its robust performance and adaptability performs this role. It handles the game's varied data, such as session details, user profiles, and gameplay actions, in a flexible format tailored to meet the game's complex data requirements. This database setup is crucial for a smooth data flow between the frontend and backend. It effectively supports the game's functionality and enhances user experience. The setup consistently maintains game states, preferences, and progress, ensuring a seamless gaming experience for users.

As demonstrated in Figure 3.3, the GreenGame database structures its data using schemas that outline the organization and shape of the stored information. At the root, there are two primary schemas: **GameSession** and **Users**.

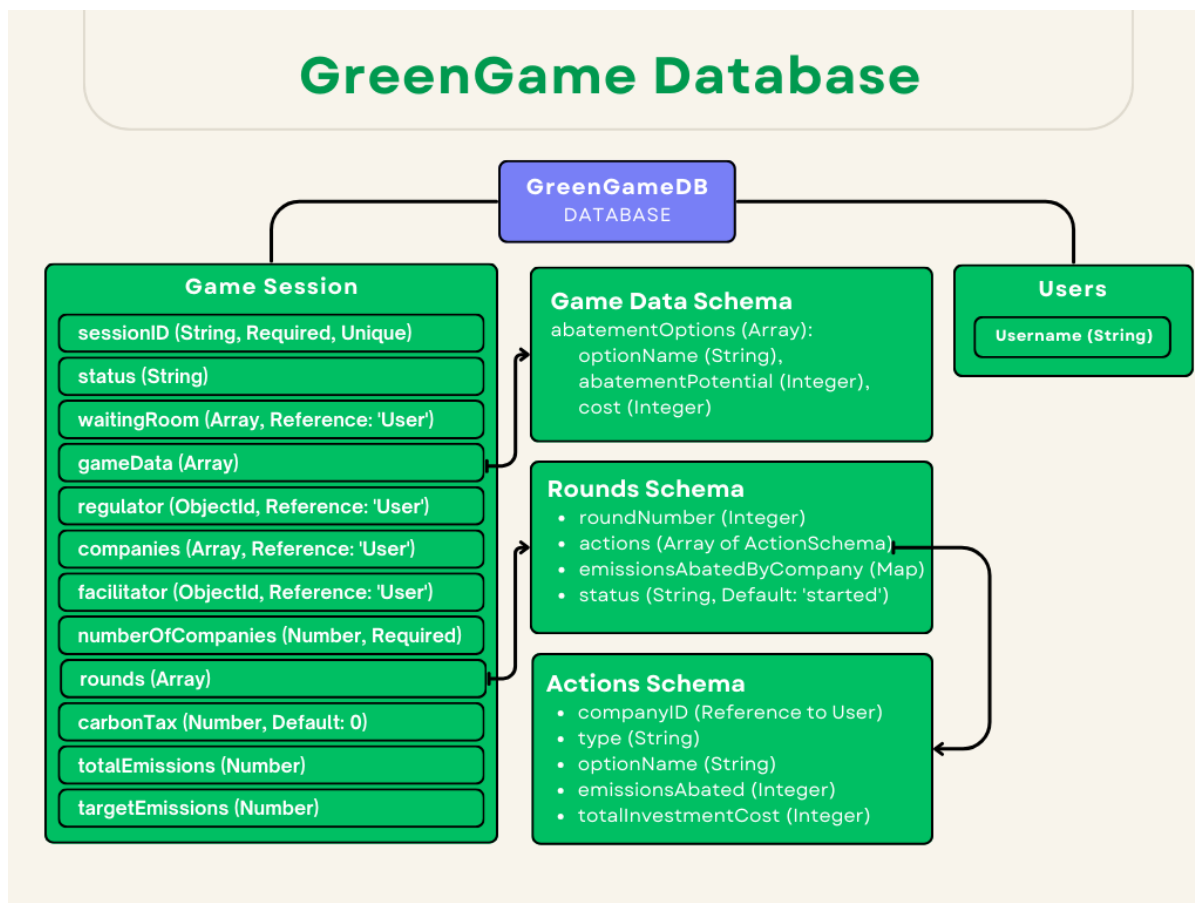


Figure 3.3 illustrates the relational structure of the GreenGame database with GameSession as the central schema that connects to Users, Game Data Schema, Rounds Schema, and Actions Schema. Lines indicate the schema relationships within a game session, which details user roles and game mechanics.

### GameSession Schema

The **GameSession** schema is a comprehensive blueprint detailing various aspects of a game's lifecycle. It records the unique session ID, overall game status, and a waiting room to manage user participation before the game starts. It ties directly to individual users through references, capturing roles like facilitator and regulator, as well as the list of participating companies. The schema also tracks the number of companies, the rounds of the game, including actions taken by companies, and overarching game data, which aggregates various abatement options—strategies



for reducing emissions. Also, the carbon tax rate, total emissions, and target abatement are in this schema.

### *Users Schema*

Meanwhile, the **Users** schema is simpler; it focuses on individual participants. It ensures each user in the game has a unique username. The authentication process in the game is supported in this schema, which allows users to log into the system and take part in the game sessions. In order to preserve the integrity and flow of game data, these schemas work together to support the entire GreenGame, from user management to tracking sessions and gameplay dynamics.

### 3.4 Interaction of System Components

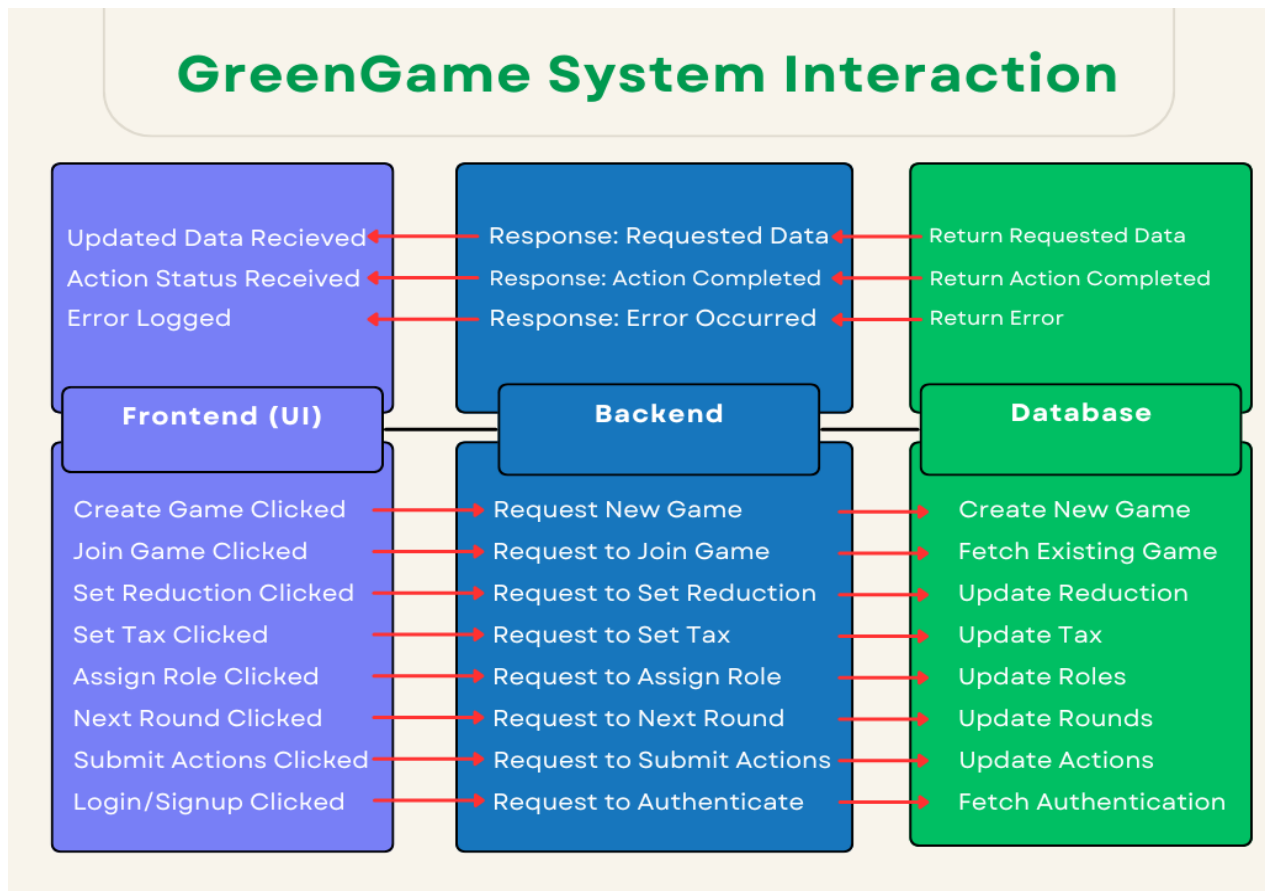


Figure 3.4 illustrates the workflow of user interactions within the GreenGame. It shows communication among the frontend UI, backend services, and database.

The GreenGame frontend, backend, and database work in unison. As shown in Figure 3.4, there is a dynamic interaction among these components. So, when a user on the frontend of the game clicks "Join Game," it actively sends a request to the backend, signaling a player's intention to join a game session. The backend responds back by retrieving the necessary game details from the database and then updating the frontend to confirm the player's participation.

Similarly, for example, if a player clicks "Set Tax", it sends a request to the backend to update the carbon tax rate. The backend receives the request and updates the corresponding records in the database and then, it confirms the action's completion back to the frontend. When it is processed the player can immediately see the updated tax rate reflected in the game interface.

If there's an error at any point, for example, maybe the database can't update due to a connection issue, then the backend catches this and logs the error. It also sends a response back to the frontend, so the user is aware that something went wrong. This direct and organized interaction between the frontend, backend, and database, as shown in Figure 3.4, ensures that all parts of the system are synchronized and informed. It also makes sure that the game data is consistent across different players' views.

## Chapter 4: Results and Evaluation

In the previous chapter, we explored GreenGame's technical setup and implementation. We walked through how the backend, frontend, and database all played a role in building the application. For every component, we explained what it does to support and enhance the game. Here we examine how well GreenGame functions in real-world scenarios. To do this, we will examine the client's feedback and my observations on the project. In particular, we will examine three key areas: the correctness of the game compared to the paper-based version, the responsiveness and speed of the system itself, and the security measures implemented to protect the system and its stored data. Also, we consider how the client assesses GreenGame. This perspective shows how close the game is to their requirements and educational goals. The purpose of this comprehensive evaluation is to evaluate the overall effectiveness of GreenGame and identify areas where improvements are needed to ensure that the game is able to achieve the intended educational outcomes.

### 4.1 Correctness

In order to assess the correctness of GreenGame, it is essential to evaluate how game mechanics work in the digital version. The primary purpose of GreenGame is to provide users with a good understanding of carbon taxation policies. The game's development has mainly focused on getting these dynamics right to ensure realistic decision-making by players. The testing and client feedback during initial increments worked to guide GreenGame to a high degree of accuracy. Based on our inspection, the primary game mechanics and functionalities including:

- Setting desired reduction
- Setting carbon taxes on emissions
- Generating noisy data for the regulator
- Generating and displaying company abatement options (game data)
- Generating and displaying MAC graphs based on game data for all roles
- Allowing companies to submit abatement and emission decisions

are correctly working as intended. Furthermore, the carbon tax game of the GreenGame incorporates all of the features agreed upon with the client including real time updates and simultaneous user access. This compliance with client requirements increases the educational value of the game and allows students to explore complex policy impacts safely.

## 4.2 Speed

When it comes to developing applications speed is another factor that determines how well an application performs. In this section, we gain an understanding of the game's performance in real-world situations, focusing on how quickly it loads, how well it responds, and how stable the overall system is.

GreenGame is hosted on a trial version of a hosting service suitable for initial testing. However, an upgrade to a professional hosting server such as premium version of Google Cloud Computing, is anticipated for the future, mainly to support a high number of users. This upgrade is necessary when we venture to extend the outreach of the game.

During our initial testing of the application, we faced challenges regarding the applications load time and responsiveness. When navigating from joining a game to reaching a specific role view it took approximately 20 to 25 seconds load time. This was because each component made individual requests from the server every five seconds to fetch updated data. The Node.js server, which manages the backend operations of our web applications, could handle rapid updates. But the clients' browsers struggled with the large amount of incoming data, which led to decrease performance and increase load times.

To fix the problem, we decided to centralize all data requests instead of each component requesting data individually. So, we shifted this responsibility to the parent component. By doing so, we significantly reduced the number of server requests. This lowered the network load and enhanced the user interface's responsiveness.

As mentioned in earlier chapters, GreenGame is developed using React. This tool is mainly used for developing user interfaces (UIs) for single-page applications (SPA). SPAs are implemented

to operate in a single browser view and they cannot handle page refreshes by clicking the refresh button in the web browsers. So, upon refreshing the game users will encounter a white screen containing the error “cannot get the URL.” This problem first arose when we deployed our application to Google Cloud and accessed it through a web browser. After examining the problem when we were testing it, we learned it is because the server does not know how to respond to direct browser requests for different paths, a common feature in traditional multi-page applications. In order to fix this issue, we planned to use advanced route handling strategies provided by frameworks such as Next.js, which is an extension of React.

Another challenge with using React, which is not an issue but has an impact in the long term on our application, involves managing data flow from parent components to child components. While passing data down through components is a fundamental React feature, it causes difficulties as the application scales up and the component hierarchy deepens. This can lead to several problems, such as code duplication, the increased processing load on the browser, and difficulties in maintaining code due to the complexity of tracking where data changes and which components need to re-render. To address these challenges, we can utilize the Redux library, a robust solution for data management across all components of an application. It ensures that data transitions are predictable and centralizes data management, making it easier to understand, debug, and manage complex interactions and data updates across the application. These technical enhancements are crucial in ensuring that GreenGame is functional and efficient enough to meet the demands of educational environments effectively.

### **4.3 Security**

When GreenGame was being developed, the main idea was to create and hone its primary functions. During this stage, the only security measures implemented were the configuration of server settings with the CORS (Cross-Origin Resource Sharing) library. This step is vitally important for controlling how game resources are distributed across different domains since the game's backend and frontend are hosted on separate domains. By implementing CORS, we were able to restrict requests to only those from approved origins. This is important for keeping the

server of the game safe against unauthorized interactions. This is also essential for securing the communication between the game's server and the front end.

However, the backend routes in GreenGame were designed to be straightforward, handling only the essential tasks of receiving requests and sending responses. They currently lack sophisticated access control mechanisms that would restrict access based on user roles. For instance, the functionality for setting the carbon tax, which is intended exclusively for the regulator role, could potentially be accessed by unauthorized roles such as companies due to a lack of stringent access controls. This oversight could compromise the intended functionality of our API by allowing unauthorized access to critical functionalities.

Additionally, the data fetched from the server is stored unencrypted in session storage and local storage, making it accessible to anyone who can access a user's device. For example, users in the role of regulator are presented with noisy data; however, they can find and retrieve the original data without any restrictions if they access the session storage of the application. This oversight could provide regulators with an unfair advantage in the game. To address these security vulnerabilities, plans are underway to implement encryption using the JWT (JSON Web Token) library, which will secure stored data by transforming it into a string of incomprehensible characters and numbers, decipherable only through a decryption process.

As GreenGame was primarily focused on functionality during the initial stages, comprehensive security measures are not fully integrated. Looking ahead, as the game progresses to broader user engagement and starts to handle more sensitive data, the importance of strengthening security measures becomes paramount. Future security strategies will include implementing more robust user authentication mechanisms and further enhancing data encryption to protect user interactions and maintain the integrity of the game.

#### **4.4 Client Evaluation**

For the GreenGame, the client provided insightful feedback, highlighting strengths and areas for improvement. The client expressed that they would want a more polished and user-friendly

interface for the game and that the design can be both minimalistic and functional. They mentioned that "The user interface needs a fair bit of work. I'd like the UI to be sparse and appealing, similar to Apple's webpages." The client desires dropdown elements in the application that would appear upon hovering the mouse; this hovering will make the application interface look cleaner while making it easy to access additional information.

Moreover, on user experience (UX), the client stated that the game is intuitive and engaging. However, they mentioned their familiarity might bias their perception of the game's UX. However, in order to objectively evaluate the UX, they suggested: "I would love to try out the paper version in my class first, and then the e-version to make my own observations and compare the two gameplays," This approach of direct comparison and gathering student feedback, will provide a more rounded understanding of how the game functions. Also, the client stated that they will use this game in the near future in their classroom to test and see the UX of the game in an actual classroom setting.

Also, the client's feedback on the educational effectiveness of the game stated that a primary goal of this game is for students to walk in the shoes of regulators and private companies that are being regulated for carbon emissions. While this goal was partially met in the paper version, they anticipate that the e-version will significantly increase the game's effectiveness. They stated that, this is primarily because the paper version required a lot of down-time both within and between rounds. In the paper version, within rounds, some players would make their decisions faster than others and get bored while others were figuring out their best decision. Between rounds, the companies would wait for the regulator to tabulate everyone's decisions and then make their own decision. The e-version bypasses much of this making gameplay significantly smoother.



## Bibliography

- ClimateWorks Centre. (n.d.). *How to read a marginal abatement cost curve*. Retrieved from <https://www.climateworkscentre.org/resource/how-to-read-a-marginal-abatement-cost-curve/>
- Corrigan, J. R. (2011). The Pollution Game: A Classroom Game Demonstrating the Relative Effectiveness of Emissions Taxes and Tradable Permits. *Journal of Economic Education*, 42(1), 70-78. Retrieved: <https://doi.org/10.1080/00220485.2011.536491>
- InTeGrate. (n.d.). *Unit 6: Carbon Emissions Game*. SERC at Carleton College. Retrieved from [https://serc.carleton.edu/integrate/teaching\\_materials/carbon\\_emissions/unit6.htm](https://serc.carleton.edu/integrate/teaching_materials/carbon_emissions/unit6.htm)
- Nekrasova, M., Sablukov, A., Novikov, A., Yushkova, S., Seregina, T., & Kindzerskaya, M. (2022). Environmental simulation games for transport workers. *Transportation Research Procedia*, 63, 2186-2193. Retrieved from Environmental simulation games for transport workers – ScienceDirect
- Sommerville, I. (2011). *Software engineering* (9th ed.). Addison-Wesley. Retrieved from <https://engineering.futureuniversity.com/BOOKS%20FOR%20IT/Software-Engineering-9th-Edition-by-Ian-Sommerville.pdf>
- GitHub: <https://github.com/Maihan07/GreenGame-Capstone.git>