ASHESI
UNIVERSITY

ESG KPI Tracking System

- Cenviro -

Individual Contribution Reports

Date: December 2, 2025

Nerrisa Abunu’s **Report:**

For the Cenviro ESG KPI tracking system, I developed three core components: the ESG calculation engine, the data access layer, and the user authentication system.

ESG Calculation Engine

I implemented the Report class which calculates ESG scores from raw business metrics. The calculateEsgScore() method evaluates ten metrics across Environmental, Social, and Governance categories, normalizes each against industry benchmarks, and produces a composite score on a 0-100 scale. The normalization system I built handles different metric types intelligently, dividing ideal by actual for "lower is better" metrics (emissions, turnover) and actual by ideal for "higher is better" metrics (diversity, recycling rates).

I created specific calculation formulas for environmental metrics including energy consumption intensity (energy per employee per month), carbon emissions per employee, and water usage intensity. For social metrics, I implemented employee turnover rate, training hours per employee, the OSHA-standard injury rate formula, and women representation tracking. The system handles missing data gracefully by excluding null values rather than failing, and normalizes across different reporting periods for fair comparison.

Data Management System

I designed the ESGData class to handle all database operations for ESG metrics. The saveData() method uses an upsert pattern to insert or update data points while preventing duplicates. The getValueFromKpi() method retrieves and averages data across specified date ranges, which my Report class calculations depend on. I also implemented methods for mapping KPI names to database identifiers, generating summary statistics, and fetching the most recent values. These database operations provide the data foundation that the calculation engine operates on.

Authentication System

I built the complete user authentication workflow including registration and login functionality. The registration system validates all required fields, enforces eight-character minimum passwords, checks for duplicate emails, and creates both organization and user records. The login system authenticates credentials, validates input formats, and initializes secure sessions. Both systems implement SQL injection prevention and XSS protection to ensure security.

Conclusion

My work delivered the technical core of the Cenviro platform through ten industry-standard KPI calculations, six database operation methods, and secure authentication workflows. These components work together to enable organizations to measure and track their ESG performance accurately and securely.

Davis Amponsah’s **Report:**

Being part of a group project, my team created Cenviro, a web-based ESG (Environmental, Social, Governance) KPI tracking system, created using PHP, MySQL, HTML, and CSS. The system enables organizations to add ESG data, track the trends using a dashboard, and create downloadable ESG reports. The present report is dedicated to my personal work on the project, especially in the field of interface design, testing, and refactoring of the codebase to an object-oriented architecture instead of procedural PHP.

Personal Position and Technological Work.

In the group, I became the main contributor to the general page structure and the main application logic. On the front end, I created the HTML structure of semantic tags to make the page accessible and consistent. I also organized the ESG data input forms to have a tabbed interface on the Environmental, Social, and Governance pillars by applying the right input types to enhance validation and user-friendliness. I used a card-based dashboard design and a basic system of design that used uniform spacing, fonts, and color scheme. I also included responsive behavior to allow the layout to fit smaller screens, with a side-bar, and tables. At the back end, my primary technical input was the refactorification of the original procedural code into an Object-Oriented Programming (OOP) framework.

Difficulties and Problem-Solving.

The difficult part was to know how classes should relate with each other without closely integrating everything to the database. On the front end, one of the major challenges was ensuring that the dashboard was not overwhelming to the users. I managed to resolve this by progressive disclosure: high level metric cards on the dashboard with a possibility to dig deeper into detailed tables and reports.

Skills and Lessons Learned

This project enhanced my technical and teamwork capabilities. On the technical side, I enhanced my knowledge on PHP OOP, database structure with foreign and constraint keys, as well as simple security measures, including password hashing and input sanitization. I also got to know the importance of incremental refactoring and testing every significant change. On the project side, I got to know that it can save a lot of time in the future by planning the architecture upfront and documenting the responsibilities of the classes. Another lesson learned in the process of working in a group was the necessity of proper sharing of files and naming conventions in order to make the code easily understood and extended by other people.

Conclusion

In general, I worked on the organization of the user interface, the application of responsive styling, and spearheaded the move to an object-oriented backend.The project has provided me with a realistic experience of utilizing the concepts of software engineering to a real web application and has enhanced my confidence in approaching systems of a greater complexity in the future.

Derrick Fiagbedzi’s **Report:**

For this project, I took full responsibility for developing the System Settings module, which is managed through the *settings.php* page. This part of the system allows administrators to update the core configuration of the application, such as the organization’s name, the theme of the system, the selected timezone, and the dataset used for the dashboard. I designed the entire form structure myself, making sure the inputs were arranged clearly and captured all the required information. On the backend, I wrote the PHP logic that validates form inputs, especially ensuring that fields like the organization name were not left empty. I also implemented the update process in the database, using secure SQL queries, and made sure that existing system settings were automatically loaded into the form by fetching the values stored in the *system_settings* table. In addition to this, I added feedback messages for both successful updates and failed submissions, and I ensured that the page communicated correctly with our shared database connection file. This whole section of the project makes it possible for users to configure the system easily and personalize their experience without needing to modify any code.

I also developed the Company Profile module, found in the *company_settings.php* page. This part allows organizations to manage key company details. I created the full HTML layout for the form, including inputs for the company’s name, address, contact email, phone number, and description. On the backend, I wrote the logic responsible for loading the current company information from the database, validating any changes made by the user, and updating the records in the *company_info* table. I also added meaningful success and error messages to guide users whenever they updated the information. While building this page, I made sure it fit smoothly into the rest of the system and followed the same layout patterns so the interface felt unified and consistent. This page gives organizations a simple way to keep their company information accurate and up to date.

Across both modules, I contributed heavily to how the backend handles data. I wrote the SQL queries needed to update records in the database and ensured that the connection was opened and closed properly on every request. I also implemented error-handling using try/catch blocks so that the system would not crash unexpectedly. In addition, I sanitized the inputs to protect the system against invalid or harmful data. I maintained a consistent backend structure across the pages so that the entire application behaved like a single, organized system.

Difficulties and Problem-Solving

One of the main difficulties I faced was managing how each form interacted with the database while still keeping the code clean, organized, and easy to follow. I had to plan the logic in a way that avoided repeating the same code unnecessarily, which required careful structuring. Another challenge was figuring out how to ensure that each form loaded the correct existing values before updates. I solved this by fetching all the needed data at the top of the script and then inserting those values directly into the form using PHP. I also needed to make sure that the feedback messages were clear and helpful, which meant paying attention to how I structured the validation and error-handling sections.

Skills and Lessons Learned

Working on these modules allowed me to strengthen several important skills. I became more confident in handling PHP forms and writing proper server-side validation. I gained more experience with SQL, especially when updating and retrieving data from the database. I also learned the importance of writing backend code that is clean, modular, and easy for others to reuse or understand. Another skill I developed was improving user experience through well-written feedback messages and clear form layouts. This project also taught me the value of maintaining consistent code structure when working as a team, as well as how to combine HTML and PHP in a way that keeps the code readable. Finally, I learned how small design choices such as selecting the right input type or organizing form fields, can improve the overall usability of a system.

Conclusion

Overall, my main contribution was the development of the System Settings and Company Profile modules, where I worked on both the interface design and the backend functionality. These two sections play a major role in the administration of the application, and building them helped me improve my skills in PHP, MySQL, and web development as a whole. The experience gave me hands-on practice with building real, database-driven modules and also strengthened my teamwork and technical confidence.

Enyonam Attipoe's **Report:**

My role in this project was focused on data documentation and creating the data input field. I was in charge of designing and keeping track of all relevant data diagrams including Entity Relation diagrams, flow diagrams, and the data dictionary for our project database. I was also tasked with creating the data input page, allowing users to input relevant KPIs to each ESG category, in order to generate their ESG report.

Data Input

The page displays three tabs—Environmental, Social, and Governance—so users can easily enter the correct type of ESG data. These tabs are switched using the JavaScript function openTab(), which shows the selected tab and hides the others.

Inside each tab, a form is included that sends the information back to the same page using POST. Every form includes a hidden category field to show which ESG section the user is submitting. There is also a required date field for the reporting period. Below that, I list the KPI input fields for that category, along with short help descriptions to guide the user.

When the user clicks "Submit Data," the website runs validateForm() to check for errors. If everything looks good, the PHP code processes the entries, saves them using

$esgData->saveData(), and displays a success or error message using $message and $message_type.

In simple terms, this page lets the website collect ESG data in a clean, organized, and user-friendly way, while making sure all information is checked and saved correctly.

Data Documentation

To support the development of the ESG Tracking system, I created a data dictionary, an ER diagram, and a flow diagram. The data dictionary helped me clearly define all the attributes of the database, their validation rules, and example formats, while the ER diagram showed the relationships between users, KPIs, categories, and input records. The flow diagram mapped how data moves from user input to validation and storage. These tools gave me a clear understanding of the structure and behaviour of the system before writing any code.

Having these documents made the development process smoother and more efficient. They helped me avoid confusion, keep the database and code consistent, and speed up decision-making during development. Whenever I needed to check how something should work, I could refer back to the diagrams. Overall, these documents played an important role in keeping the project organized and ensuring its success.

Skills I Used

While working on this page, I used several important skills:

- **PHP programming**, especially handling POST requests and working with arrays and functions.
- **Backend logic design**, such as mapping KPI fields to KPI patterns and saving them in the database.
- **JavaScript validation**, including writing the validateForm() function to catch errors before submission.
- **HTML and CSS**, to design the form layout, tabs, and user-friendly interface.
- **Secure authentication handling**, using functions like requireLogin() and logout().

Lessons I Learned

From this process, I learned how important it is to keep my code organized, especially when dealing with many input fields. Using a mapping array made the code cleaner and easier to maintain. I also learned how useful JavaScript validation is in creating a smooth user experience, catching errors before they reach the server. Most importantly, I saw how backend and frontend code work together to create a reliable data entry system.

Conclusion

Overall, building this ESG Data Input page helped me understand how to handle secure user input, validate data, and store it properly in a database. I learned how to structure a clean, multi-tab form and how to map many different fields into one processing flow. The combination of PHP, JavaScript, HTML, and authentication functions allowed me to create a functional and user-friendly system. This project strengthened my technical skills and gave me a better understanding of full-stack development.