EcoBazaar

Initial Phase: Brainstorming Website Ideas and Major Features

1. **Brainstorming website ideas:**

- What type of website (e.g., e-commerce, storefront, educational and learning platform, streaming and entertainment site, etc.)?

Type of website : e-commerce.

- Who are the target users ?

General users who are eco-responsible.

- What services or information will be provided?

Secondhand, recycled, eco designed products.

2. **Website features**

- Identify the main features of your website

Look for products by categories, by name, by price, by ratings, by popularity, ect.

Users can add their own products, delete their own products, update their own products, add products to their cart, make an order.

- Define user roles (e.g., administrator, regular user, guest, etc.) and their rights.

Regular users (CRUD) : add (their own products), view the entire list of products, update (their own products), delete (their own products), add items in the cart, remove items from the cart, make an order.

Administrator users : manage products, manage users.

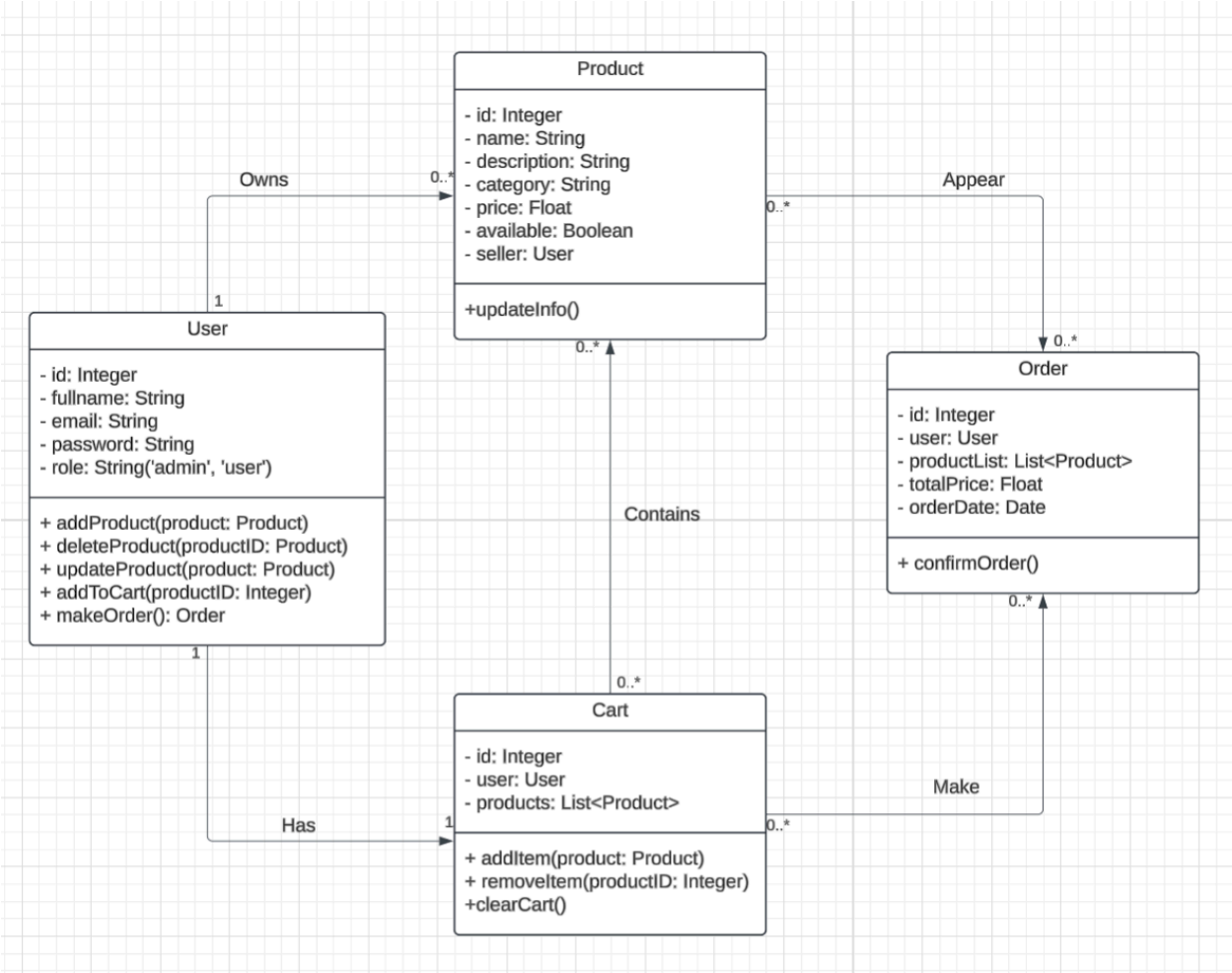
2nd phase: design

1. **Designing minimalist and eco-responsible architecture:**

▪ Website design diagrams (suggested tool: https://plantuml.com/fr/)

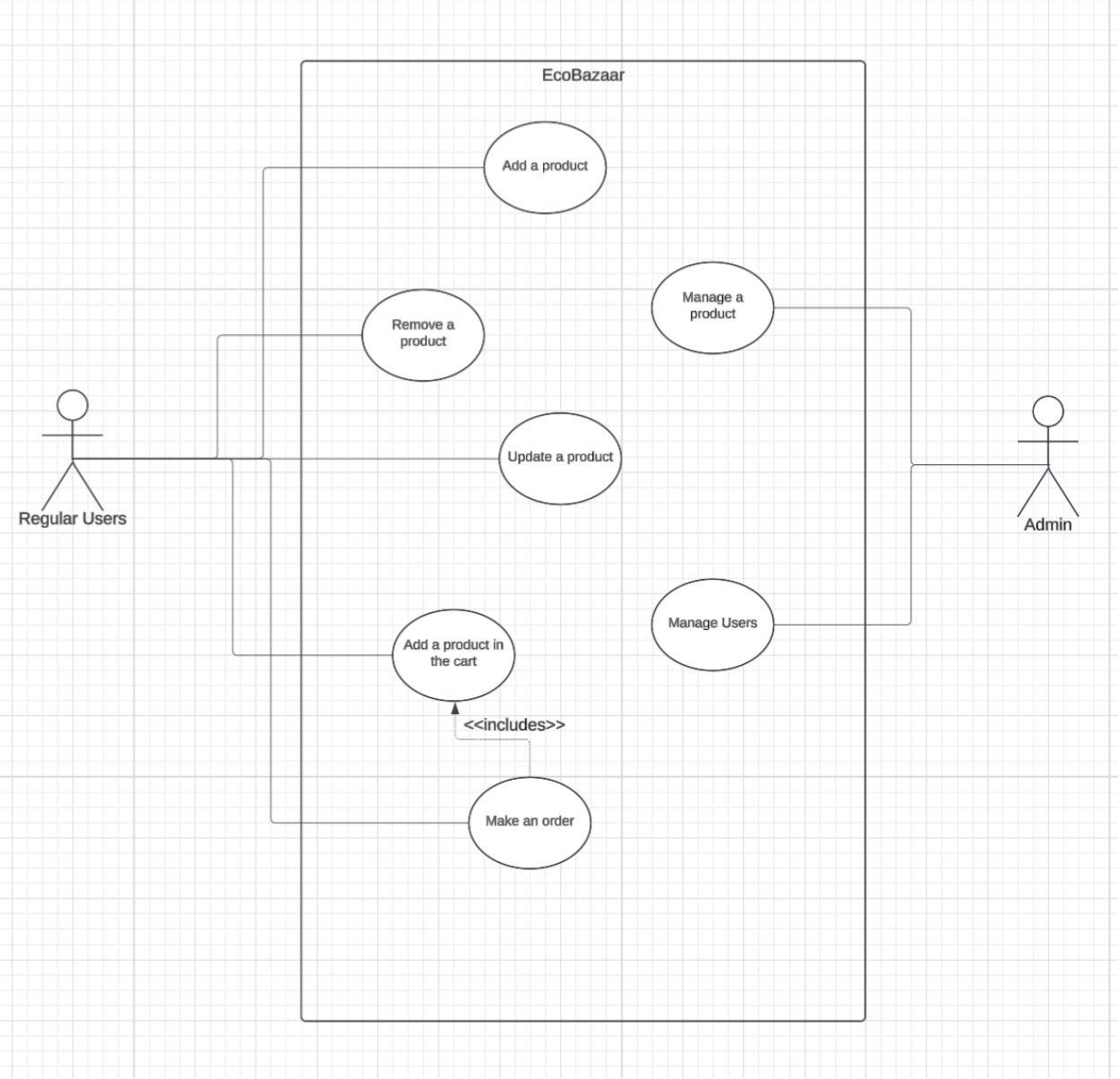
○ Class diagram, to represent the structure of objects and interactions.

Class diagram : (UML diagram which describe the structure of our project system)



○ Sequence diagram, to show the exchange of messages between the components of the site.

○ A Use Case Diagram, to model the interactions between the actors and the system.



▪ Wireframe (e.g., via **Figma**, **Canva**, **Sketch** or **Balsamiq**).

○ Design a mockup of the user interface.

○ Include screens for key features (e.g., homepage, …).

▪ Specify the pages to be expanded, with at least:

○ A home page informing about the site’s ecological measures.

○ A user management page (add, delete, edit).

○ An overview page of other data retrieved from the database.

○ Etc.

2. **Choose technologies:**

▪ Front-end and back-end technologies taking into account their resource consumption.

Eco-friendly tech stack for the front-end : HTML5, CSS3, Javascript.

Eco-friendly tech stack for the back-end : Express.js and Sqlite3

▪ Favor lightweight frameworks (e.g. Jekyll, Hugo, or basic HTML/CSS-based solutions).

Express.js lightweight framework to simplify request/response with the server.

3rd phase: front-end, back-end implementation and database access

Step 1: Front-end development

To be obtained: develop the visible part of the site by integrating green IT concepts, in particular the optimization (minification, compression, etc.) of CSS files, JavaScript files and images.

- **Suggested technologies:**
  - HTML5/CSS3 for structure and style.
  - JavaScript for interactivity (minimal use of heavy libraries like jQuery).
  - Lightweight framework like **TailwindCSS** or **Bootstrap** if needed.
- **Key instructions:**
  - Optimization of the weight of web pages by minimizing unnecessary resources (minification of files, limitation of heavy images).
  - Compressing images via tools like **TinyPNG** or **Squoosh**.
  - Use lazy loading to load elements (images, videos) only when needed.
  - Limit the use of complex animations and transitions (to reduce CPU consumption).

Step 2: Back-end development and database access

To get: Implement server-side logic, with management of data stored in a relational database.

- **Suggested technologies**:
    - **Node.js**, **Django** or **PHP** for the back-end.
    - **MySQL**, **PostgreSQL**, or **SQLite** for the database.
- **Key instructions**:
    - Setting up a database to manage users and any other relevant data (products, sustainability information, etc.)
    - CRUD (Create, Read, Update, Delete) of users and other entities:
        - Creation of a user table with fields such as name, email, date of registration.
        - Adding new entries via a form on the site.
        - Modify or delete users from a dedicated interface.
        - Display of registered users via a list connected to the database.
    - Minimize queries and optimizations of database access (paging of results, optimized queries).
    - Retrieve only the data that is necessary for the display.
    - Use persistent connections to minimize power consumption when accessing the database.