**EcoMechanica Software Requirements Document:**

**Table of Contents:**

1)Header…………………………………………………………………1

2) Product Perspective……………………………………………….2

3) Product Functions…………………………………………………3

4) User Characteristics………………………………………………3

5) Constraints…………………………………………………………..3

6) System Features……………………………………………………3

7)System Qualities……………………………………………………6

8) Task Matrix……………………………………………………………6

**Product Perspective**

* *This project was conceived as a game designed to educate primary and secondary school students on the environmental damage inflicted by industrial factories.*

**Product Functions**

* There are 4 main functions in our game:
  + 1) Resource Extraction:
  + 2) Manufacturing New Resources:
  + 3) Selling or Researching with these Resources:
  + 4) To avoid creating pollution as much as possible:

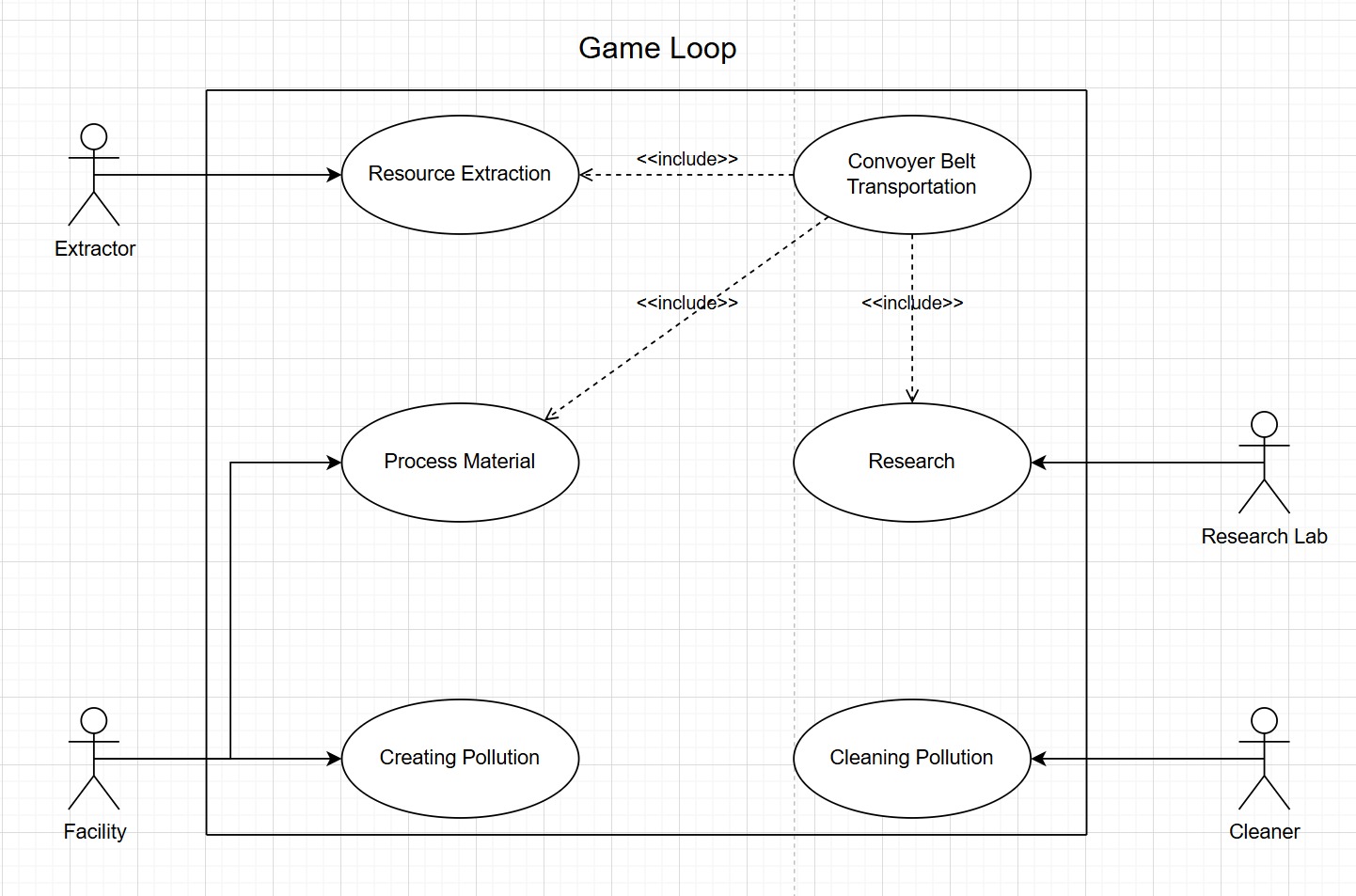
**User Characteristics**

* *The game is designed to appeal to learners across age groups, from schoolchildren to university-level youth . The game is designed to be intuitive and accessible, requiring minimal background knowledge for young learners and college students alike.*

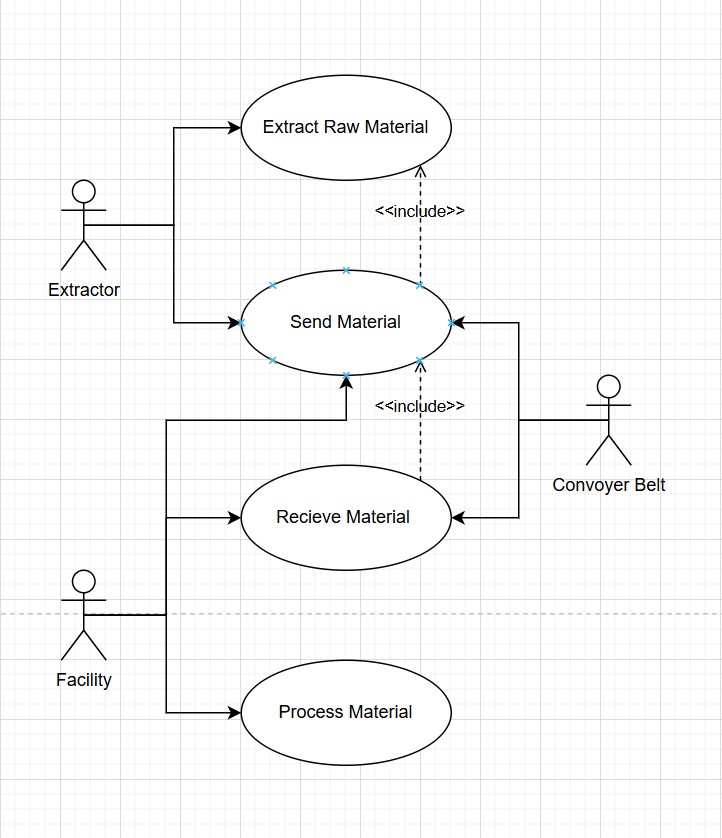
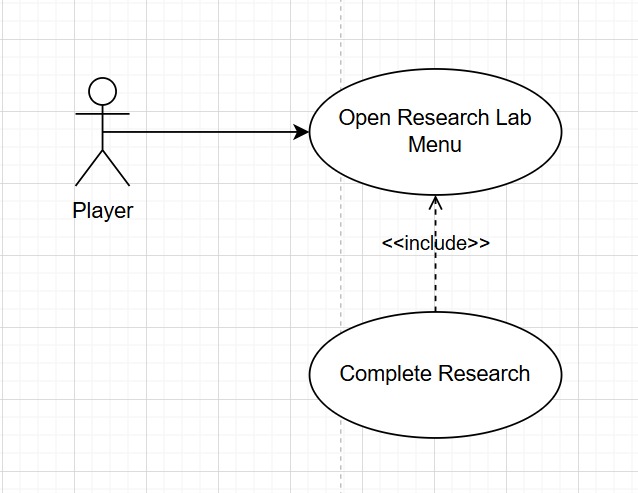
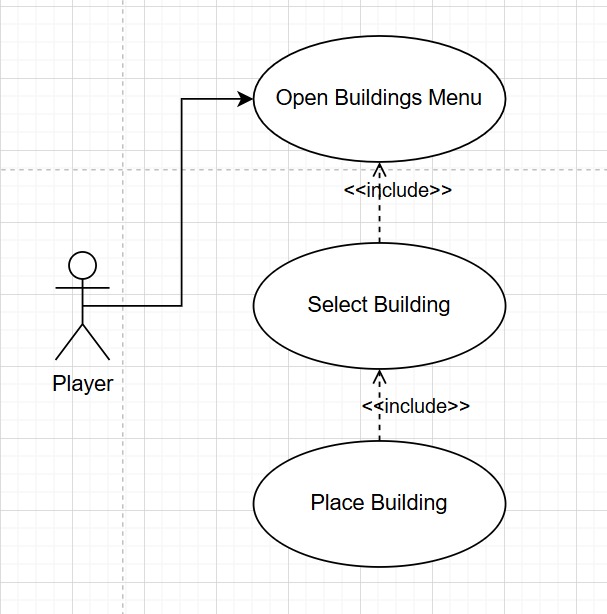
**Constraints**

* *A PC-only experience available in English exclusively.*

**System Features**



**diyagram, metin, taslak, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.**

* **i. Title**  Resource extraction

**ii. Main Actor** Extractor

**iii. Goal** Extracting raw material to sell, research or process

**iv. Preconditions** Extractor must be built in resources pile

**v. Main Flow** Extract the resources according to resource type, assign conveyor belts to transport resources to other buildings

**vi. Postconditions** Gain resources

**i. Name the pattern** Factory method pattern strategy pattern

**ii. Why** Factory method pattern strategy pattern for creating raw metarial according to its resource pile

**iii. Describe how** Each resource pile has its own extraction strategy. These strategies produce raw materials with certain rules and periods.The Extractor produces the appropriate product according to the strategy through a Factory Method. New production behaviors can be added without interfering with existing codes, making the system modular and sustainable.

* **i. Title** Researching

**ii. Main Actor** Researcher Lab

**iii. Goal** To unlock new recipes for being able to build new facilities

**iv. Preconditions** Must have supplied enough of resources

**v. Main Flow** Check if the resource is necessity for researching.

If the resource is not necessary sell it; otherwise advance the research progress

**vi. Postconditions** Unlock new recipes of processed materials

**i. Name the pattern** Command pattern

**ii. Why** To decide which action will be taken when material arrives to research lab

**iii. Describe how** When a material arrives, the Research Lab calls this command. Materials that can be used by existing Research are spent on research, otherwise they are sold.

* **i. Title**  Conveyor belts transportation

**ii. Main Actor** Conveyor belts

**iii. Goal**  Accurate conveying of products on conveyor belts.

**iv. Preconditions** Receiving material from a previous addressee

**v. Main Flow** Transports Arriving products to other belts or facilities or Research Lab

**vi. Postconditions** Sending material to a next addressee

**i. Name the pattern** Chain of responsibility pattern

ii. **Why** Because the structure to which the material will be transmitted varies depending on the context it is working in. In this case, each conveyor takes responsibility and performs the operation if there is a suitable target, otherwise it forwards it to the next one.

**iii. Describe how** Each conveyor belt will work as a chain link and will transfer the incoming products to the next structure or belt it is connected to. In this way, any changes to the conveyor network, connecting a new structure or directing the flow will be possible by simply editing the chain without changing the code. In this way, the system will become flexible and modular.

* **i. Title**  Cleaners

**ii. Main Actor** Cleaner

**iii. Goal** Enhancing the game's longevity

**iv. Preconditions** The pollution level must have increased.

**v. Main Flow** Decreases the pollutions increase

**vi. Postconditions** The pollutions extending level will decrease

**i. Name the pattern** Observer Pattern

**ii. Why** Because Cleaners must react according to the pollution level in the world. The pollution level is kept by a central system and Cleaner structures that observe this value must be triggered automatically.

**iii. Describe how** A central object called PollutionManager keeps track of the pollution level and updates itself by observing Cleaners

* For design Software Architecture we will use Layered Architecture as Unity uses it in itself and we will use it by 4 main layers Presentation Layer which will be Ui ,input and rendering layer Game Logic Layer which will run at back making the calculations Data layer for saving and research lab recipes and Utility layer for Event Triggering if we use a diagram to describe it in text [Player Input]-> [Presentation Layer(Unity)]->[Game Logic Layer(Simulation) ] -> [Data layer (Save and recipes)] ->[Utility Layer]. What benefits we got from this architecture? If we simplfy our examples in our use cases Factory building will be done in Logic layers condition on Presentation layer likewise Pollution Management Research Progression will be handled in Utility , Data and Presentation Layer and of course we are doing it a singleplayer game so no need to work on server – client dependencies

**System Qualities**

* The system will be user friendly with understandable buttons and menus , it can have some performance issues because there will be so much materials had been produced but it should be stable.
* The user's screen will feature interactive buttons and a map. Players can freely navigate the map using WASD controls, with each button clearly labeled to explain its function.
* This Project will mainly use Unitys libraries

**TASK MATRİX**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Finding Use cases | Making the UML Diyagram | *A debate regarding the project's intended audience* | Finding non Functional  Requirements | Finding constraints | Finding User Abilities |
| Umut Baran Boztaş | X |  | X | X |  | X |
| Mehmet Efe Palaz | X |  | X |  | X |  |
| Efe Selim Sürekli |  | X | X |  | X |  |
| Mehmet Fatih Akay | X | X | X |  |  |  |
| Kaan Behzetoğlu |  |  | X | X |  | X |