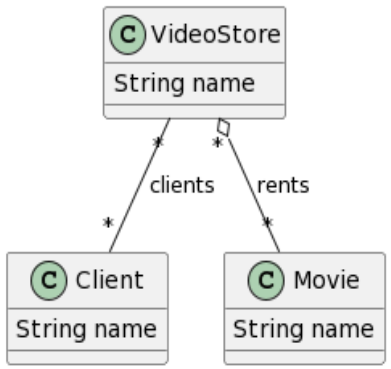
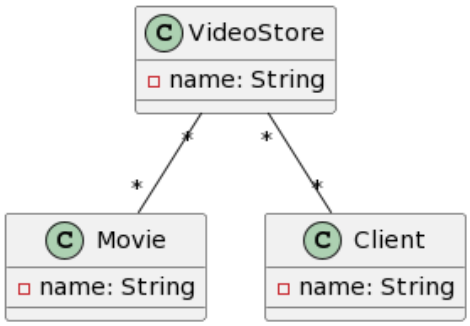
Model: JT2 (VideoStore_2)

Date: 2023/02/12

User intent	ChatGPT output
	

Observations

- Results generated are not deterministic. Even when we regenerate, results are different in different sessions.
- Attributes and classes in this simple example are generated correctly. However, associations are not.
- Several iterations with explicit requests for modification are needed to approximate the user intent model.

Chat Log

Summary:

- In the first session, no cardinality is added in the references, and the type of association is not correct (aggregation). When we try to regenerate the answer, the aggregation type disappears and cardinalities in all ends are added. There is a new association, which is wrong, between Movie and Client, since nothing is said in the input about this association. Association between VideoStore and Movie since correct (it could be aggregation, but it can also be ok like this), but cardinality between Client and VideoStore has been made up (according to the cardinality, one can be Client of only one VideoStore). Attributes are OK in both cases.
- In the second session, we try the same prompt as in session 1. This time, associations are worse than before, since Client and Movie inherit from VideoStore. This is semantically very wrong. Attributes are OK. We regenerate the answer. Now the association between Movie and VideoStore is worse than before, since there is “1” multiplicity in the “Movie” end. There are two associations between VideoStore and Client (“has” and “rents from”), while there should only be one. Attributes are OK. The input was exactly the same as in Session 1, but results have been different.
- In the third session, we input the same prompt. Yet a different solution from the previous two sessions is provided, this time it is closer to the expected solution, but cardinalities in associations were wrong. We tried to explain in the next prompt how cardinalities should be, but they were wrong again. In the third prompt, we re-explained the issue with the cardinalities, and this time they were correct.

Session 1

Prompt 1

Draw a UML diagram of a Video Store. There are also clients and movies in the system, all of which have a name. Video stores rent out movies, where the same movie can be offered in any video store. Video stores have clients. Please do not add any operation in the UML diagram. Draw the UML diagram in PlantUML.

ChatGPT Output 1

Here's a UML class diagram for the Video Store system you described, using PlantUML syntax:

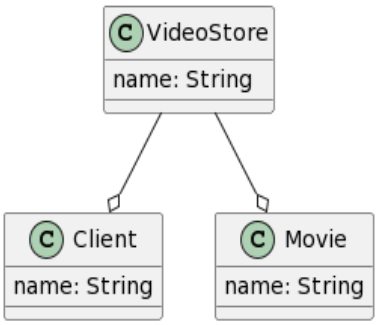
kotlin Copy code

```
@startuml
class VideoStore {
  name: String
}

class Client {
  name: String
}

class Movie {
  name: String
}

VideoStore --o Client
VideoStore --o Movie
@enduml
```

The above diagram depicts three classes: VideoStore, Client, and Movie, with the following properties:

- VideoStore has a name attribute of type String.
- Client has a name attribute of type String.
- Movie has a name attribute of type String.

The relationships between the classes are depicted using associations. The associations between the VideoStore and Client classes and VideoStore and Movie classes are represented by a solid line with an arrowhead pointing from the VideoStore class to the Client/Movie class, indicating that a VideoStore can have many Clients/Movies, but a Client/Movie can only belong to one VideoStore. The association is also marked with an "o" to indicate that it's a one-to-many relationship.



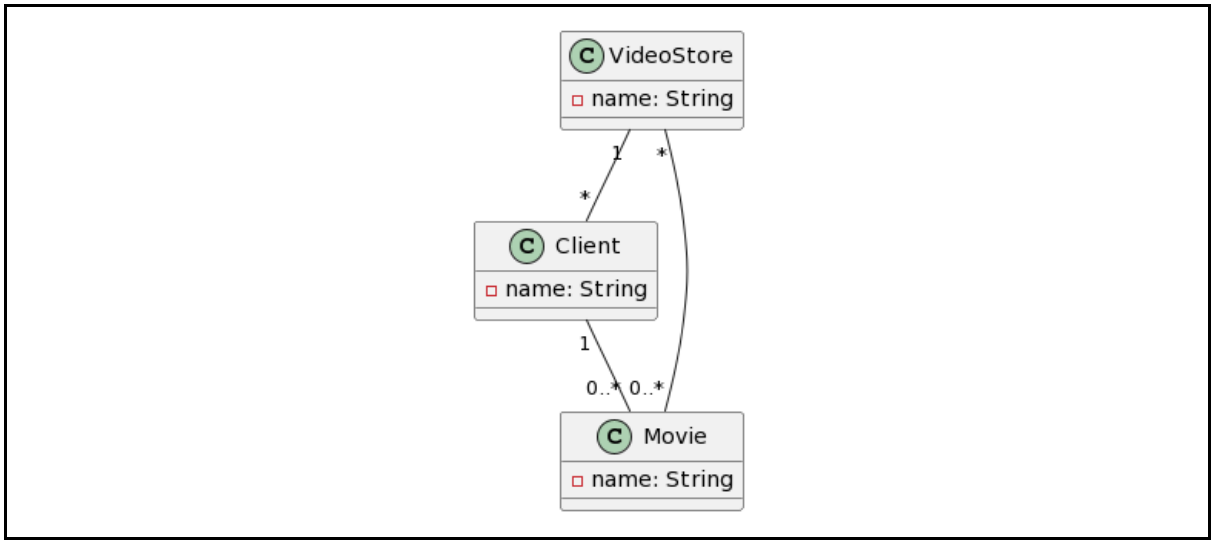
ChatGPT Output 2 (Regenerated Response)

Here is a UML class diagram in PlantUML syntax representing a Video Store system with clients and movies:

kotlin Copy code

```
@startuml
class VideoStore {
  - name: String
}

class Client {
  - name: String
}

class Movie {
  - name: String
}

VideoStore "1" -- "*" Client
Client "1" -- "0..*" Movie
Movie "0..*" -- "*" VideoStore
@enduml
```

This diagram shows that a `VideoStore` has a name and can have multiple `Client` objects. A `Client` also has a name and can rent out zero or more `Movie` objects. A `Movie` can be offered in any number of `VideoStore` objects.



Session 2

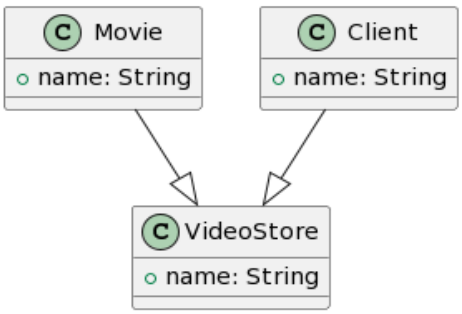
Prompt 1
Draw a UML diagram of a Video Store. There are also clients and movies in the system, all of which have a name. Video stores rent out movies, where the same movie can be offered in any video store. Video stores have clients. Please do not add any operation in the UML diagram. Draw the UML diagram in PlantUML.

ChatGPT Output 1
Here is a UML class diagram of a Video Store using PlantUML syntax:

kotlin Copy code

```
@startuml
class Movie {
  +name: String
}

class Client {
  +name: String
}

class VideoStore {
  +name: String
}

Client --|> VideoStore
Movie --|> VideoStore
@enduml
```

In this diagram, the class Movie has an attribute name of type String, which represents the name of the movie. The class Client also has an attribute name of type String, which represents the name of the client. The class VideoStore has an attribute name of type String, which represents the name of the video store. The relationships between the classes Client and VideoStore and between Movie and VideoStore are represented by a dotted line with an arrow pointing from the client or movie to the video store, indicating that a client rents movies from a video store, and a video store offers movies for rent.



ChatGPT Output 2 (Regenerated Response)

Here is a UML class diagram for a Video Store, using PlantUML syntax:

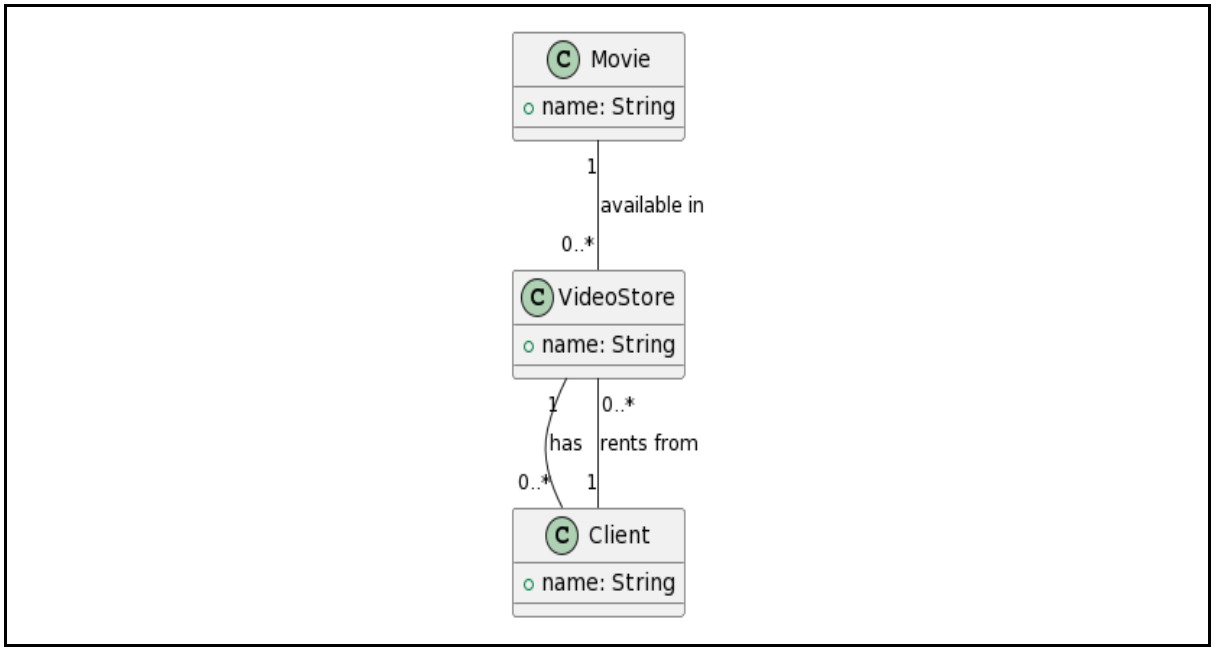
typescript Copy code

```
@startuml
class VideoStore {
  +name: String
}

class Client {
  +name: String
}

class Movie {
  +name: String
}

VideoStore "1" -- "0..*" Client : has
Client "1" -- "0..*" VideoStore : rents from
Movie "1" -- "0..*" VideoStore : available in
@enduml
```

Explanation:

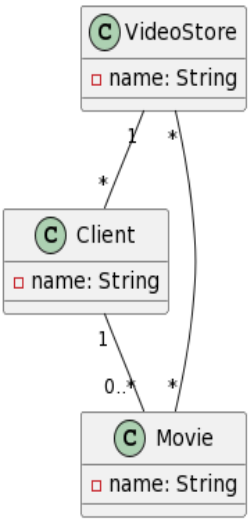
- The `VideoStore` class has a `name` attribute of type `String`.
- The `Client` class has a `name` attribute of type `String`.
- The `Movie` class has a `name` attribute of type `String`.
- The `VideoStore` class has a on ith the `Client` class, indicated by the `"1" -- "0..*"` notation, store can have many clients.

Regenerate response



SOLUCION DE CHATGPT:

```
@startuml
class VideoStore {
  -name: String
}

class Client {
  -name: String
}

class Movie {
  -name: String
}

VideoStore "1" -- "*" Client
Client "1" -- "0..*" Movie
VideoStore "*" -- "*" Movie
@enduml
```



Session 3

Prompt 1

Draw a UML diagram of a Video Store. There are also clients and movies in the system, all of which have a name. Video stores rent out movies, where the same movie can be offered in any video store. Video stores have clients. Please do not add any operation in the UML diagram. Draw the UML diagram in PlantUML.

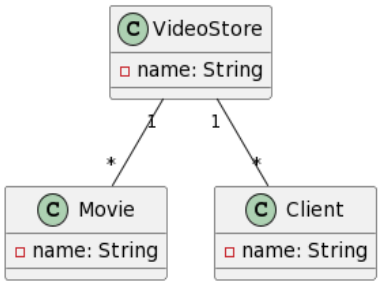
ChatGPT Output 1

Here's a UML class diagram for a video store system in PlantUML:

kotlin Copy code

```
@startuml
class VideoStore {
  -name: String
}

class Movie {
  -name: String
}

class Client {
  -name: String
}

VideoStore "1" -- "*" Movie
VideoStore "1" -- "*" Client
@enduml
```

This diagram represents the three main classes in the system: `VideoStore`, `Movie`, and `Client`. The relationship between the classes is depicted using arrows. The arrow pointing from `VideoStore` to `Movie` signifies that a `VideoStore` can have multiple `Movie`s. The arrow pointing from `VideoStore` to `Client` signifies that a `VideoStore` can have multiple `Client`s. The "1" and "*" annotations indicate the cardinality of the relationships, meaning that there is one `VideoStore` instance but multiple `Movie` and `Client` instances.



Prompt 2 (Trying to fix cardinalities in the associations)

Please, redo the diagram, this time considering that the same movie can be in several video stores, and the same client can be a client of several video stores. Please do not add any operation in the UML diagram. Draw the UML diagram in PlantUML

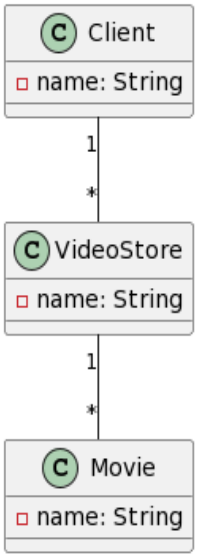
ChatGPT Output 2

Sure, here's a revised UML class diagram in PlantUML:

kotlin Copy code

```
@startuml
class VideoStore {
  -name: String
}

class Movie {
  -name: String
}

class Client {
  -name: String
}

VideoStore "1" -- "*" Movie
Client "1" -- "*" VideoStore
@enduml
```

This diagram shows that a `Movie` can be in multiple `VideoStore`s, represented by the arrow pointing from `Movie` to `VideoStore` with a cardinality of "1" on the `Movie` side and "" *on the* `VideoStore` *side. Additionally, a* `Client` *can be a client of multiple* `VideoStore`*s, represented by the arrow pointing from* `Client` *to* `VideoStore` *with a cardinality of "1" on the* `Client` *side and* "" on the `VideoStore` side.



Prompt 3 (Trying to fix cardinalities in the associations, again)

The diagram is not right, there are problems with the multiplicities. Please note that video stores can have any number of movies, and the same movie can be in any number of video stores. Likewise, video stores can have any number of clients, and the same client can be a client of any number of video stores. Please do not add any operation in the UML diagram. Draw the UML diagram in PlantUML.

ChatGPT Output 3

Here's the corrected UML class diagram in PlantUML:

kotlin Copy code

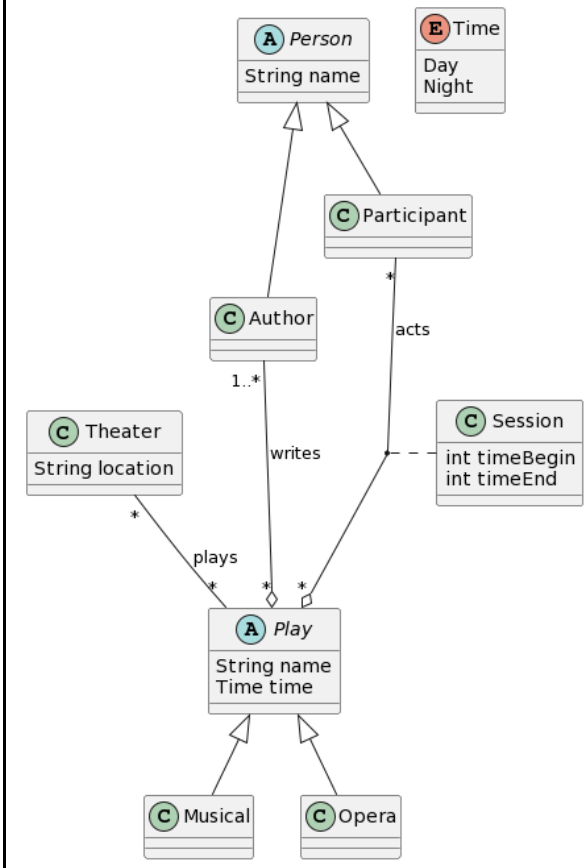
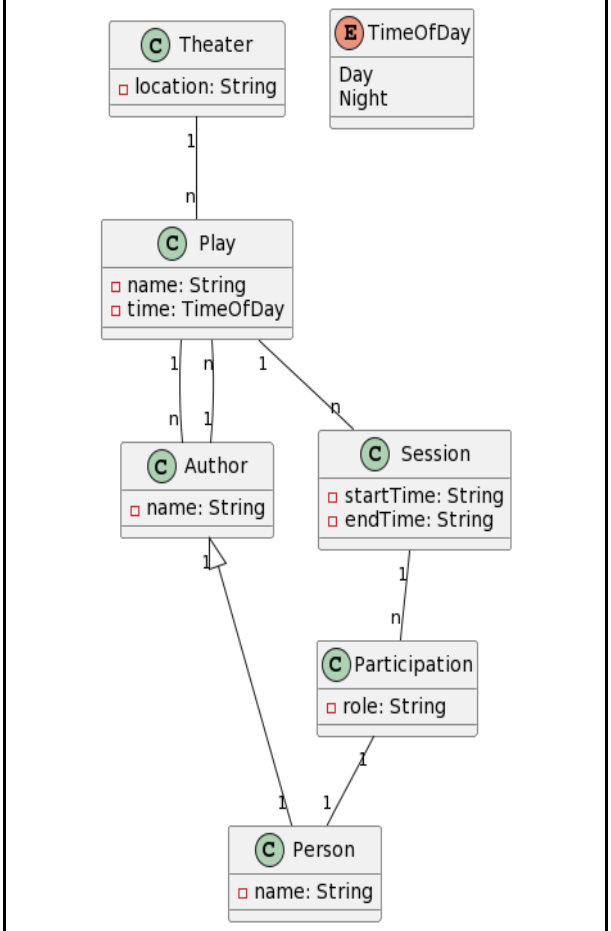
```
@startuml
class VideoStore {
  -name: String
}

class Movie {
  -name: String
}

class Client {
  -name: String
}

VideoStore "*" -- "*" Movie
VideoStore "*" -- "*" Client
@enduml
```

This diagram accurately represents the relationships between the classes, with the arrow pointing from `VideoStore` to `Movie` and the arrow pointing from `VideoStore` to `Client` both having a cardinality of "*" on both ends, meaning that a `VideoStore` can have any number of `Movie`s and `Client`s, and the same `Movie` and `Client` can be associated with any number of `VideoStore`s.

VideoStore
name: String
Movie
name: String
Client
name: String
*
*
*
*

Model: JT8 (Theater_4)

Date: 2023/02/13

User intent	ChatGPT output
	

Observations

- Results generated are not deterministic. Even when we regenerate, results are different in different sessions.
- For this model, some results have serious flaws (such as Participant inheriting from Author, or having the Person class without associations).
- The information in some associations is also specified as attributes (of type array of the target class). The information of these attributes is redundant.
- Sometimes, regenerating the solution does not have any positive effect towards the user intent.
- Enumeration classes are not generated unless said explicitly.
- Association classes are not generated.
- Inheritance relations are not properly dealt with. Sometimes they are not generated. When they are, the parent class is not abstract, and attributes in children and parent class are repeated.
- In some executions, some important class is not generated (Participant in Session 3).

- The composition association is not obtained from the context.
- Sometimes, two bidirectional associations are defined between two classes with different multiplicities. There should be only one bidirectional association.
- In it hard to approximate to the user intent in this example

Chat Log

Summary:

- In the first session, we try to create the UML diagram, but the version is not correct at all. There are serious flaws, such as a Participant inheriting from an Author, or a Person not having any children. The association class has not been created. Also, the model seems like a mixture of design and implementation models: the information about the associations is also registered as attributes (attributes whose type is an array of the target class). Some cardinalities are not correct, and there is no enumeration type. I asked ChatGPT to improve the solution. The only thing it did was change the array type of the attributes with a List type. Musical and Opera classes are not defined.
- In the second session, we input the same prompt. Again, association class is not generated, and there are unnecessary attributes with redundant information about the associations. There are semantically wrong references, such as between Participant and Person. Some cardinalities in associations are wrong. We click on the button to regenerate response. This time, inheritance relationships are properly created, but the same attribute is present in the subclasses and in the parent class (name). The class that should be an association class, has no association at all. Musical and Opera classes are not defined.
- In the third session, with the same input, the Participant class is not even generated. There is no inheritance, no enumeration class and cardinalities are wrong. In the second iteration, I try to force the creation of Participant class and the associations. The class is created, but there is no association class. It would be correct if there was a composition association with the proper cardinalities. Anyway, let's leave it like this for now. In the next prompt, we try to create the inheritance relationships, but the solution provided by ChatGPT is wrong. There is only one inheritance association (instead of two) and its direction is wrong. In the fourth prompt, we say it explicitly, but the same solution is provided. In the fifth prompt, we try for ChatGPT to generate the Enum class and to fix issues regarding cardinalities. The enumeration class is created, and there are now two associations between Play and Author that should be merged in one. Musical and Opera classes are not defined.

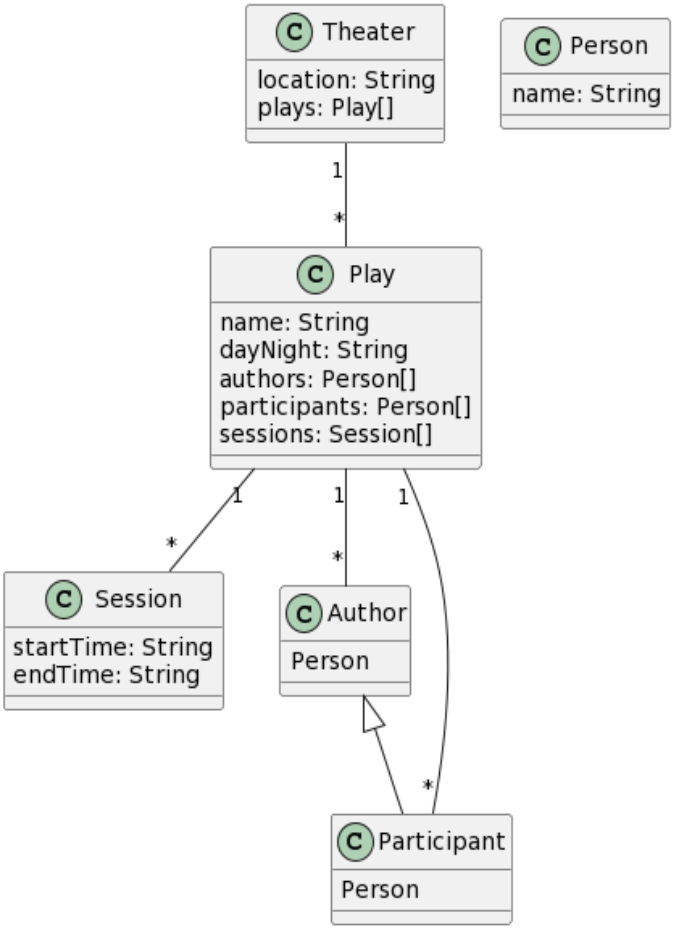
Session 1

Prompt 1
Draw a UML diagram of a Theater. Theaters have a location, and they can play any play, which can be musicals or operas. Plays have a name and can be played either at day or night. Plays are written by one or more authors, who have a name. There are many participants who act in a play, participants also have a name. A particular participant acts in a particular session of a play, where each session begins and ends at a specific time. Both authors and participants are persons. Please do not add any operation in the UML diagram. Draw the UML diagram in PlantUML.

ChatGPT Output 1

Here is the UML diagram for a Theater using PlantUML syntax

```
@startuml
class Theater {
  location: String
  plays: Play[]
}

class Play {
  name: String
  dayNight: String
  authors: Person[]
  participants: Person[]
  sessions: Session[]
}

class Person {
  name: String
}

class Session {
  startTime: String
  endTime: String
}

class Author {
  Person
}

class Participant {
  Person
}

Theater "1" -- "*" Play
Play "1" -- "*" Author
Play "1" -- "*" Participant
Play "1" -- "*" Session
Author <|-- Participant
@enduml
```



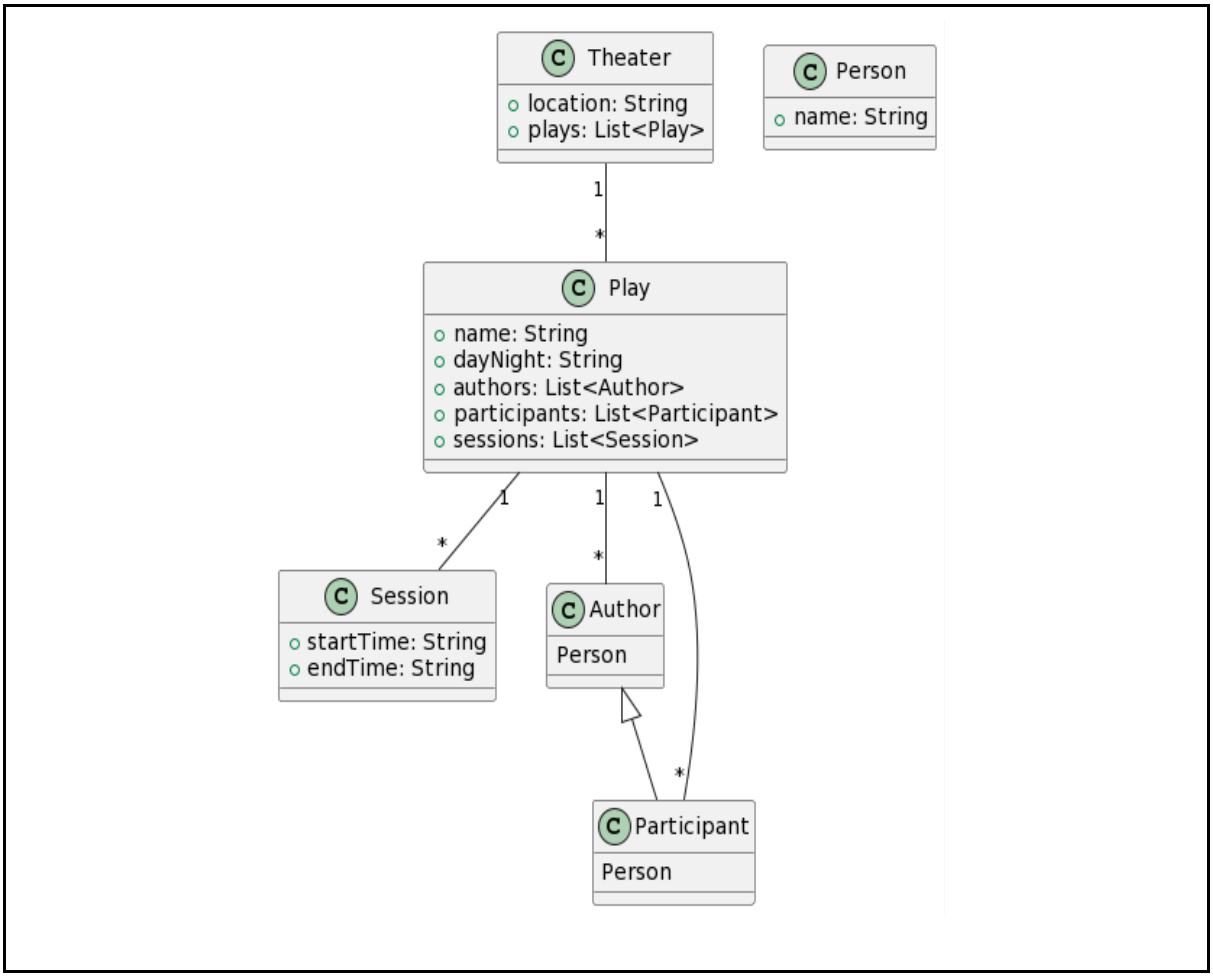
Prompt 2

I didn't see the option to regenerate the solution, so I explicitly wrote:
can you please improve the given solution?

ChatGPT Output 2 (Regenerated Response)

Sure, here is an improved version of the UML diagram:

```
@startuml
class Theater {
  +location: String
  +plays: List<Play>
}

class Play {
  +name: String
  +dayNight: String
  +authors: List<Author>
  +participants: List<Participant>
  +sessions: List<Session>
}

class Person {
  +name: String
}

class Session {
  +startTime: String
  +endTime: String
}

class Author {
  Person
}

class Participant {
  Person
}

Theater "1" -- "*" Play
Play "1" -- "*" Author
Play "1" -- "*" Participant
Play "1" -- "*" Session
Author <|-- Participant
@enduml
```

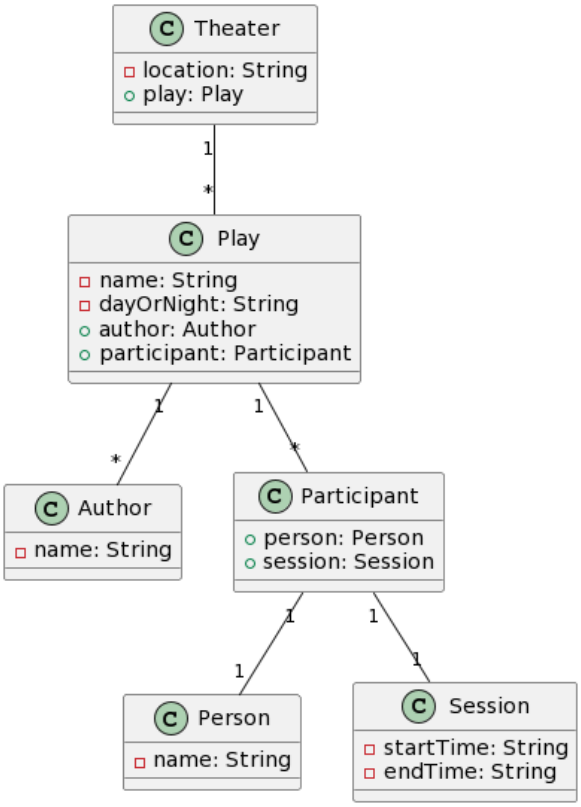


Session 2

Prompt 1
Draw a UML diagram of a Theater. Theaters have a location, and they can play any play, which can be musicals or operas. Plays have a name and can be played either at day or night. Plays are written by one or more authors, who have a name. There are many participants who act in a play, participants also have a name. A particular participant acts in a particular session of a play, where each session begins and ends at a specific time. Both authors and participants are persons. Please do not add any operation in the UML diagram. Draw the UML diagram in PlantUML.

ChatGPT Output 1
Here is a UML class diagram in PlantUML for a Theater system: @startuml class Theater {

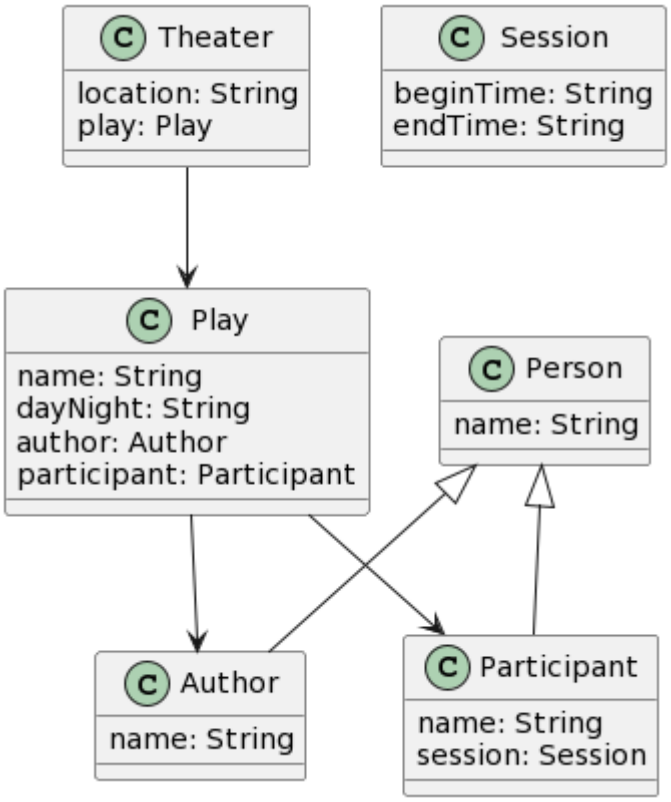
```
 -location: String
 +play: Play
}

class Play {
 -name: String
 -dayOrNight: String
 +author: Author
 +participant: Participant
}

class Author {
 -name: String
}

class Person {
 -name: String
}

class Participant {
 +person: Person
 +session: Session
}

class Session {
 -startTime: String
 -endTime: String
}

Theater "1" -- "*" Play
Play "1" -- "*" Author
Play "1" -- "*" Participant
Participant "1" -- "1" Person
Participant "1" -- "1" Session
@enduml
```



ChatGPT Output 2 (Regenerated Response)

```
@startuml
class Theater {
 location: String
 play: Play
}
```

```
class Play {
  name: String
  dayNight: String
  author: Author
  participant: Participant
}

class Author {
  name: String
}

class Participant {
  name: String
  session: Session
}

class Session {
  beginTime: String
  endTime: String
}

class Person {
  name: String
}

Person <|-- Author
Person <|-- Participant
Play --> Author
Play --> Participant
Theater --> Play
@enduml
```



Session 3

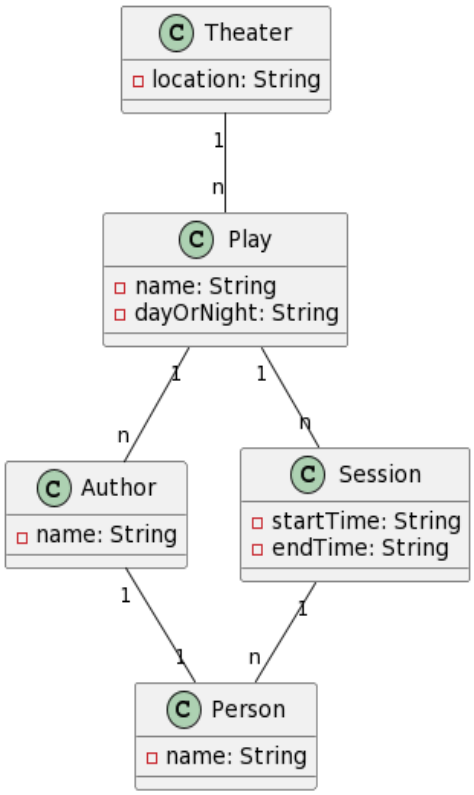
Prompt 1

Draw a UML diagram of a Theater. Theaters have a location, and they can play any play, which can be musicals or operas. Plays have a name and can be played either at day or night. Plays are written by one or more authors, who have a name. There are many participants who act in a play, participants also have a name. A particular participant acts in a particular session of a play, where each session begins and ends at a specific time. Both authors and participants are persons. Please do not add any operation in the UML diagram. Draw the UML diagram in PlantUML.

ChatGPT Output 1

Here's a UML class diagram that represents the relationship between a Theater, Play, Author, Person, and Session, as described in your description:

```
@startuml
class Theater {
  -location: String
}

class Play {
  -name: String
  -dayOrNight: String
}

class Author {
  -name: String
}

class Person {
  -name: String
}

class Session {
  -startTime: String
  -endTime: String
}

Theater "1" -- "n" Play
Play "1" -- "n" Author
Author "1" -- "1" Person
Play "1" -- "n" Session
Session "1" -- "n" Person
@enduml
```



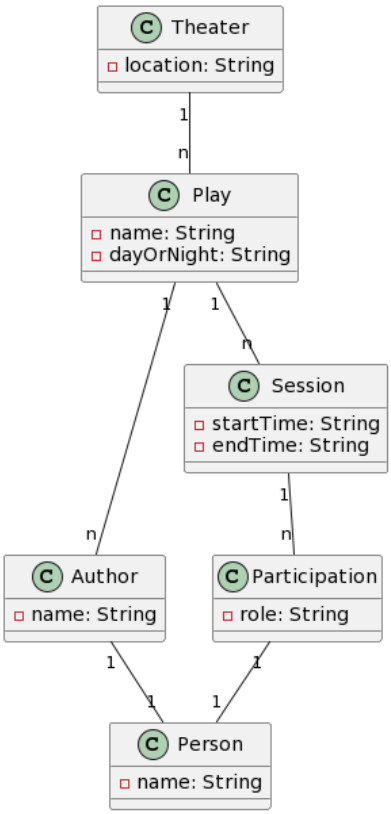
Prompt 2 (Trying to force the creation of Participant class with proper associations)

Can you please improve the model? Please take into account that participants are the ones acting in the plays. Any play can have any number of participants, and any participant can act in any number of plays. Furthermore, each participant acts in specific sessions of a play, where the session has a begin time and an end time

ChatGPT Output 2

Certainly! Here's an updated UML class diagram that takes into account the additional information you provided:

```
@startuml
class Theater {
  -location: String
}

class Play {
  -name: String
  -dayOrNight: String
}

class Author {
  -name: String
}

class Person {
  -name: String
}

class Session {
  -startTime: String
  -endTime: String
}

class Participation {
  -role: String
}

Theater "1" -- "n" Play
Play "1" -- "n" Author
Author "1" -- "1" Person
Play "1" -- "n" Session
Session "1" -- "n" Participation
Participation "1" -- "1" Person
@enduml
```

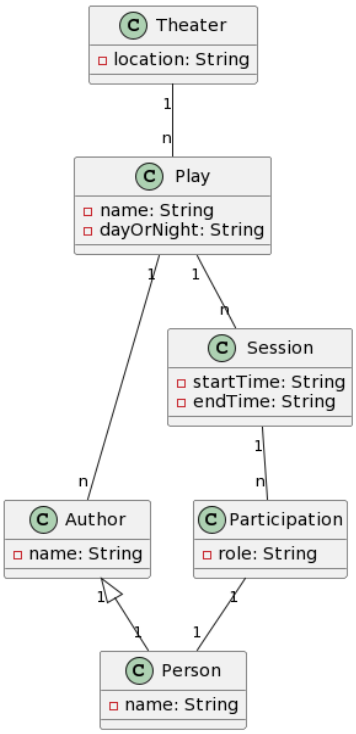


Prompt 3 (Trying to force the creation of inheritance associations)

Thank you, the diagram is a bit better now. However, both author and participants are persons, so the relationship between these classes should be different. Could you please correct this?

ChatGPT Output 3

```
@startuml
class Theater {
  -location: String
}

class Play {
  -name: String
  -dayOrNight: String
}

class Person {
  -name: String
}

class Author {
  -name: String
}

class Session {
  -startTime: String
  -endTime: String
}

class Participation {
  -role: String
}

Theater "1" -- "n" Play
Play "1" -- "n" Author
Author "1" <|-- "1" Person
Play "1" -- "n" Session
Session "1" -- "n" Participation
Participation "1" -- "1" Person

@enduml
```

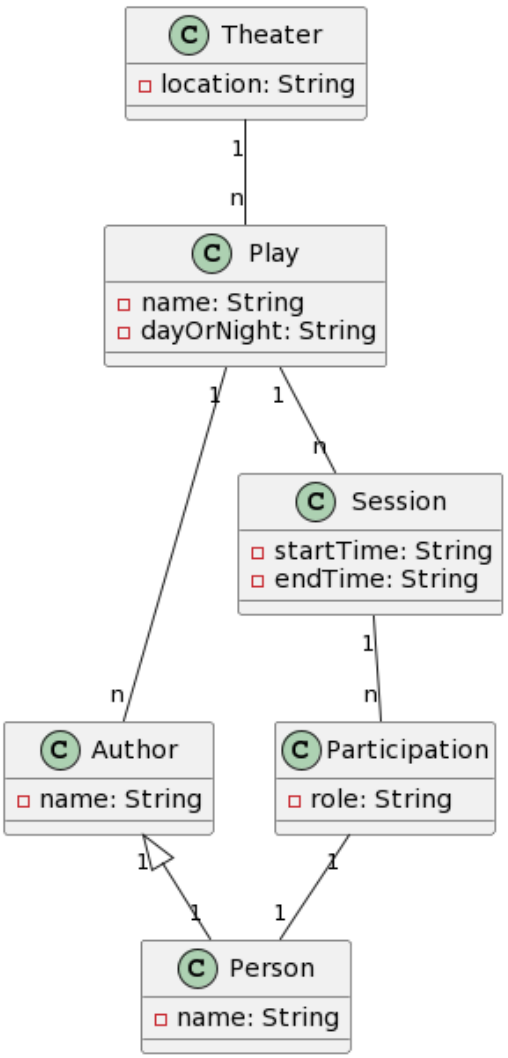


Prompt 4 (Trying to correct the inheritance associations)

Thanks, but you got the inheritance relationship wrong. Author must inherit from person, and participant must inherit from person. Please draw the diagram again

ChatGPT Output 4

```
@startuml
class Theater {
  -location: String
}

class Play {
  -name: String
  -dayOrNight: String
}

class Person {
  -name: String
}

class Author {
  -name: String
  --
}

class Session {
  -startTime: String
  -endTime: String
}

class Participation {
  -role: String
}

Theater "1" -- "n" Play
Play "1" -- "n" Author
Author <|-- Person
Play "1" -- "n" Session
Session "1" -- "n" Participation
Participation "1" -- "1" Person

@enduml
```

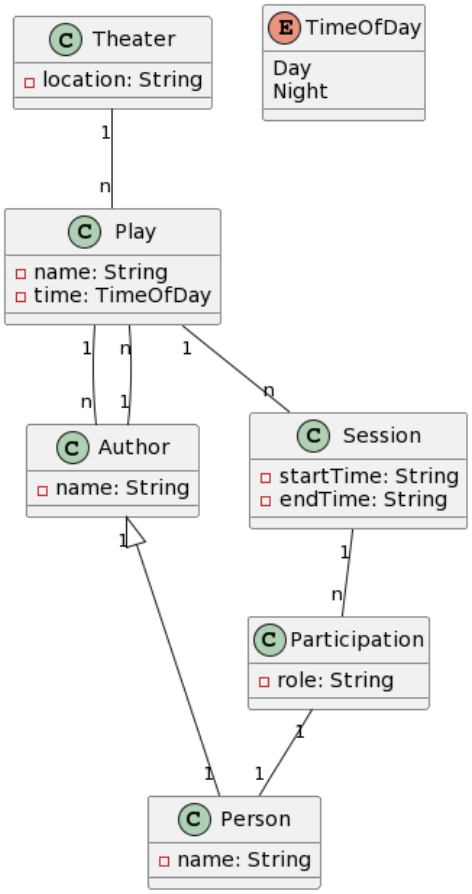


Prompt 5 (Trying to correct many details)

As I said before, Plays can happen at day or night. That means that we only have those two options, so it is better to have an enumeration class for this. Regarding authors, one can be the author of more than one play. Besides, plays can take place in more than one theater. Finally, participants can act in any number of plays. Please improve the UML diagram

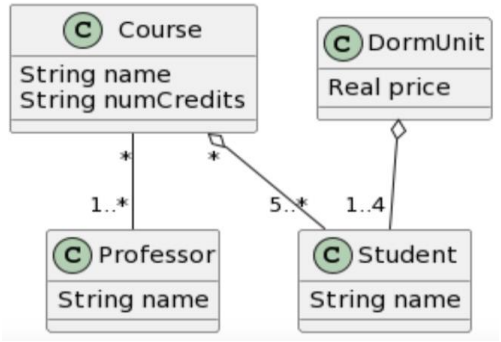
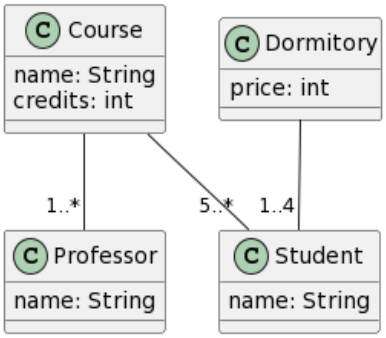
ChatGPT Output 5

```
@startuml
class Theater {
  -location: String
}

class Play {
  -name: String
  -time: TimeOfDay
}

enum TimeOfDay {
  Day
  Night
}

class Person {
  -name: String
}

class Author {
  -name: String
}

class Session {
  -startTime: String
  -endTime: String
}

class Participation {
  -role: String
}

Theater "1" -- "n" Play
Play "1" -- "n" Author
Author "1" -- "n" Play
Author "1" <|-- "1" Person
Play "1" -- "n" Session
Session "1" -- "n" Participation
Participation "1" -- "1" Person
@enduml
```



Model: LB4 (Student)

Date: 2023/02/13

User intent	ChatGPT output
	

Observations

- Results generated are not deterministic. Even when we regenerate, results are different in different sessions. There is not a clear rule.
- When explicitly asking ChatGPT to improve a solution, it creates new classes that are not mentioned in the initial input.
- When explicitly asking ChatGPT to improve a solution and we give details about a cardinality, the model is worsened by the duplication of associations.
- It is not trivial to give the proper indications for obtaining a “good” model.
- Sometimes, even when explicitly asking ChatGPT not to define operations, it does include them.
- Aggregation associations are not created.
- It is hard for a non experienced user to generate a diagram if she does not the diagram that has to be created.

Chat Log

Summary:

- In the first session, all four classes are properly created, with their attributes. Associations’ cardinalities are not right, and there is a duplicated association (between Course and Student). Aggregation association is not created, despite the fact that we explicitly mentioned “aggregated” in the input. We ask ChatGPT to improve the solution, and now it generates classes that were not in our initial input. This is wrong.
- In the second session, we enter the same prompt. The result is now improved, although aggregation associations are not generated and a cardinality is especially wrong. We try to improve the solution with a clarification regarding the cardinality. However, the result is not only not improved, but it is worsened. There are now two binary associations between two classes that were not there before.
- In the third session we input again the same prompt. The model is again different. This time it has different errors. Despite we explicitly asked not to include any operation, it added operations. Cardinalities are inverted in some binary relationships.

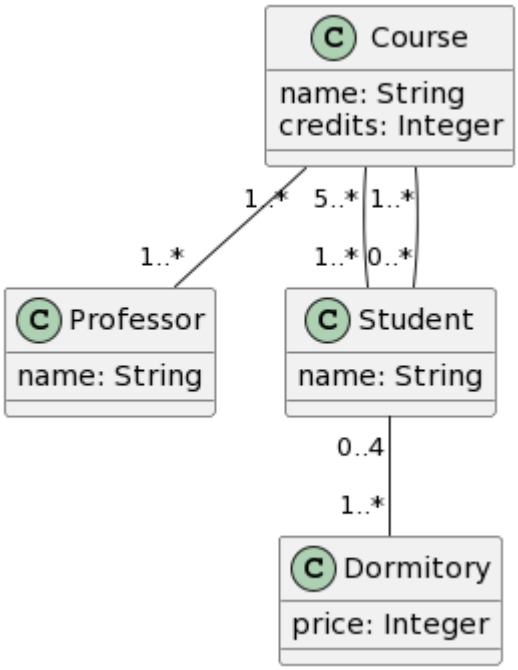
Session 1

Prompt 1

Hi, ChatGPT. We want to model some entities of a course. We know that courses have a name and a specific number of credits. Each course can have one or more professors, who have a name. Professors could participate in any number of courses. For a course to exist, it must aggregate, at least, five students, where each student has a name. Students can be enrolled in any number of courses. Finally, students can be accommodated in dormitories, where each dormitory can have from one to four students. Besides, each dormitory has a price. Please, draw a UML diagram using PlantUML. Do not include any operation.

ChatGPT Output 1

```
@startuml
class Course {
  name: String
  credits: Integer
}

class Professor {
  name: String
}

class Student {
  name: String
}

class Dormitory {
  price: Integer
}

Course "1..*" -- "1..*" Professor
Course "5..*" -- "1..*" Student
Student "0..*" -- "1..*" Course
Student "0..4" -- "1..*" Dormitory

@enduml
```



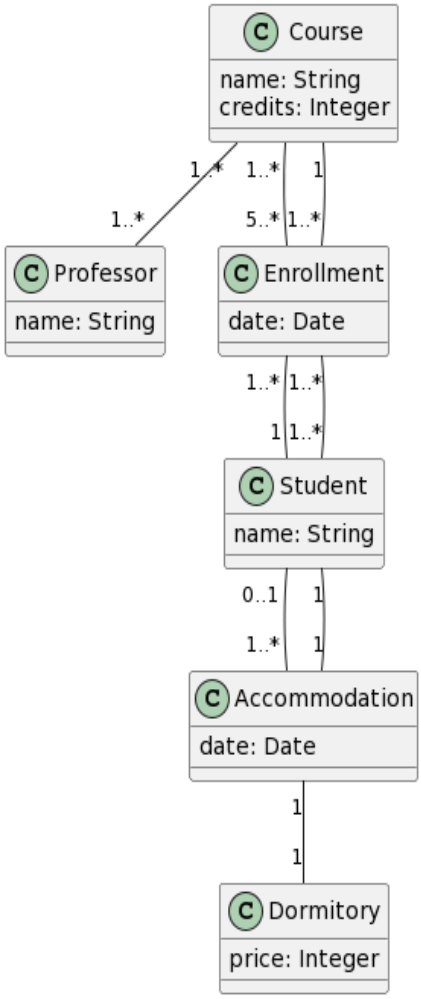
Prompt 2

We do not have the option to regenerate the diagram, so we input:

There are some flaws in this diagram, can you please provide an improved solution? Please, draw a UML diagram using PlantUML. Do not include any operation.

ChatGPT Output 2 (Regenerated Response)

```
@startuml
class Course {
  name: String
  credits: Integer
}

class Professor {
  name: String
}

class Student {
  name: String
}

class Dormitory {
  price: Integer
}

class Enrollment {
  date: Date
}

class Accommodation {
  date: Date
}

Course "1..*" -- "1..*" Professor
Course "1..*" -- "5..*" Enrollment
Enrollment "1..*" -- "1" Course
Enrollment "1..*" -- "1" Student
Student "1..*" -- "1..*" Enrollment
Student "0..1" -- "1..*" Accommodation
Accommodation "1" -- "1" Student
Accommodation "1" -- "1" Dormitory
@enduml
```



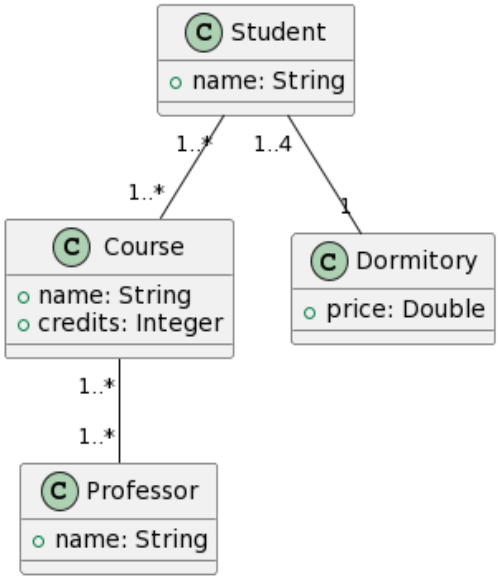
Session 2

Prompt 1

Hi, ChatGPT. We want to model some entities of a course. We know that courses have a name and a specific number of credits. Each course can have one or more professors, who have a name. Professors could participate in any number of courses. For a course to exist, it must aggregate, at least, five students, where each student has a name. Students can be enrolled in any number of courses. Finally, students can be accommodated in dormitories, where each dormitory can have from one to four students. Besides, each dormitory has a price. Please, draw a UML diagram using PlantUML. Do not include any operation.

ChatGPT Output 1

```
@startuml
class Course {
  +name: String
  +credits: Integer
}

class Professor {
  +name: String
}

class Student {
  +name: String
}

class Dormitory {
  +price: Double
}

Course "1..*" -- "1..*" Professor
Student "1..*" -- "1..*" Course
Student "1..4" -- "1" Dormitory
@enduml
```

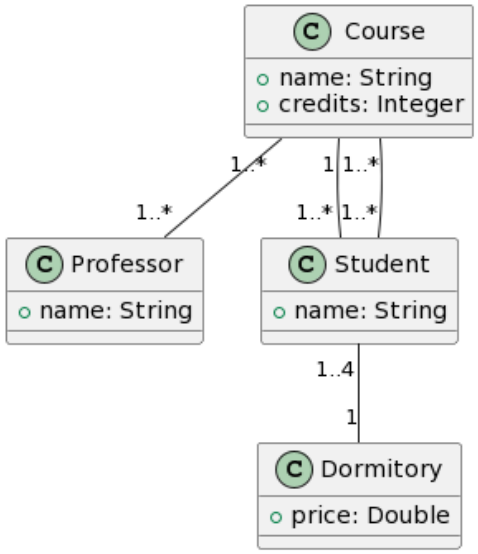


Prompt 2

Thank you. I told you that each course must have, at least one students. can you please consider this in the model?

ChatGPT Output 2

```
@startuml
class Course {
  +name: String
  +credits: Integer
}

class Professor {
  +name: String
}

class Student {
  +name: String
}

class Dormitory {
  +price: Double
}

Course "1..*" -- "1..*" Professor
Course "1" -- "1..*" Student
Student "1..*" -- "1..*" Course
Student "1..4" -- "1" Dormitory
@enduml
```



Session 3

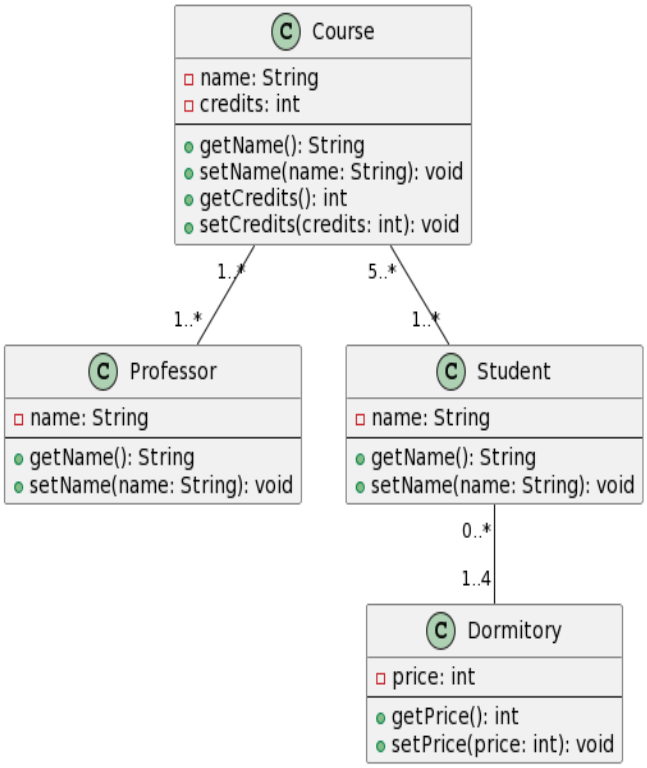
Prompt 1

Hi, ChatGPT. We want to model some entities of a course. We know that courses have a name and a specific number of credits. Each course can have one or more professors, who have a name. Professors could participate in any number of courses. For a course to exist, it must aggregate, at least, five students, where each student has a name. Students can be enrolled in any number of courses. Finally, students can be accommodated in dormitories, where each dormitory can have from one to four students. Besides, each dormitory has a price. Please, draw a UML diagram using PlantUML. Do not include any operation.

ChatGPT Output 1

```
@startuml
class Course {
  -name: String
  -credits: int
  --
  +getName(): String
  +setName(name: String): void
```

```
  +getCredits(): int
  +setCredits(credits: int): void
}

class Professor {
  -name: String
  --
  +getName(): String
  +setName(name: String): void
}

class Student {
  -name: String
  --
  +getName(): String
  +setName(name: String): void
}

class Dormitory {
  -price: int
  --
  +getPrice(): int
  +setPrice(price: int): void
}

Course "1..*" -- "1..*" Professor
Course "5..*" -- "1..*" Student
Student "0..*" -- "1..4" Dormitory
@enduml
```



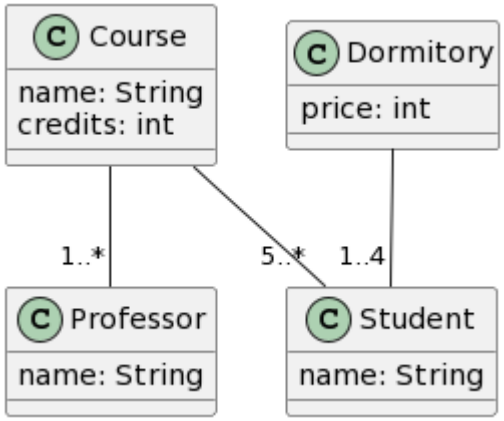
Prompt 2 (Trying to improve associations and remove operations)

Thanks. I told you not to include any operation, though. Can you please remove them? Besides, associations' multiplicities are wrong. A dormitory must have from 1 to 4 students (and not the other way around). Besides, each course must have at least 5 students (and not the other way around). Can you please improve the model with these instructions?

ChatGPT Output 2

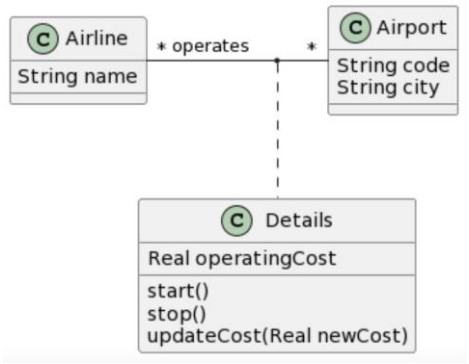
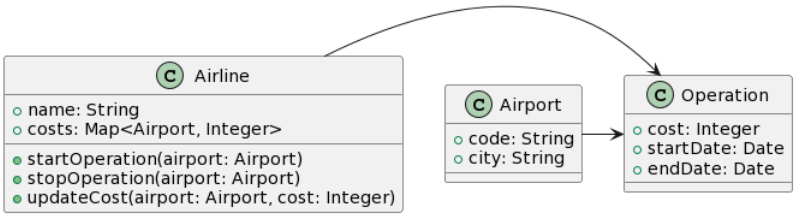
```
@startuml
class Course {
  name: String
```

```
  credits: int
}

class Professor {
  name: String
}

class Student {
  name: String
}

class Dormitory {
  price: int
}

Course -- "1..*" Professor
Course -- "5..*" Student
Dormitory -- "1..4" Student
@enduml
```



Model: LB5 (Airport)

Date: 2023/02/13

User intent	ChatGPT output
	

Observations

- It is not able to create association classes.
- Sometimes, redundant attributes with information of a relationship are created.
- Since association classes are not added, when there are operations that should be in an association class, these are created in one of the classes of the association.

Chat Log

Summary:

- In session 1, the association class is not generated. In fact, the solution given is quite different from the user intent. So we ask ChatGPT to improve the solution by adding an association class. It does not add any association class, but it changes the model. The model seems more accurate, but operations are for instance not included. Besides, multiplicities are not included.
- In session 2, for the same input, a very different solution is obtained. There is a System class without any association to any other class. Where does this come from? The operations that should be added in an association class are added in Airline.
- In session 3, for the same input, only two classes are created. For this reason, we ask explicitly for the creation of an association class, but it is not created. Instead, there is a new class that has associations with the former two classes. Operations are located in an incorrect class.

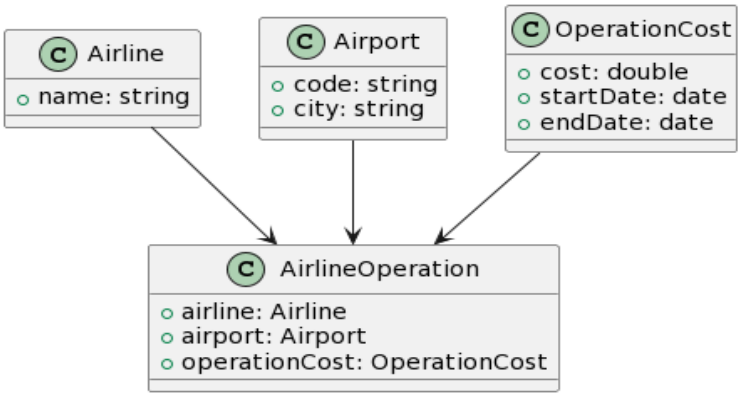
Session 1

Prompt 1

Many different airlines can operate in different airports. An airline has a name, and an airport has a code and is in a city. Each airline has associated a cost for operating in a specific airport. Besides, the operation of an airline in a specific airport can be started and stopped at any time, and its operating cost can be updated at any time. Please, draw a UML diagram for this system. Represent the diagram using PlantUML

ChatGPT Output 1

```
@startuml
class Airline {
  +name: string
}

class Airport {
  +code: string
  +city: string
}

class OperationCost {
  +cost: double
  +startDate: date
  +endDate: date
}

class AirlineOperation {
  +airline: Airline
  +airport: Airport
  +operationCost: OperationCost
}

Airline --> AirlineOperation
Airport --> AirlineOperation
OperationCost --> AirlineOperation
@enduml
```

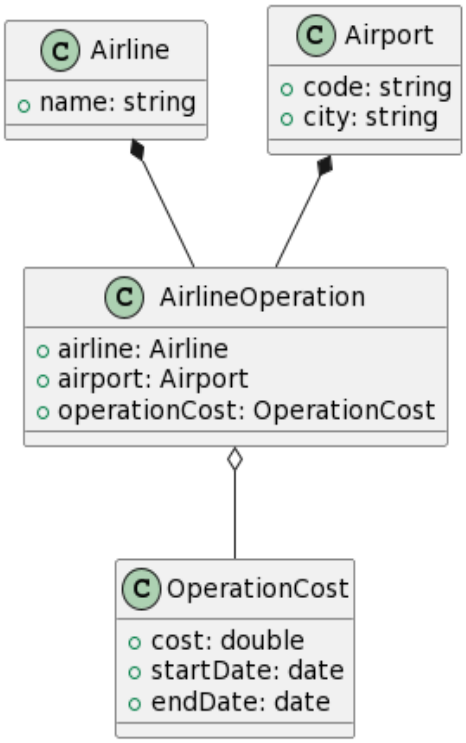


Prompt 2

jtroya@uma.es
Can you please provide the solution using an association class between airport and airline?

ChatGPT Output 2 (Regenerated Response)

```
@startuml
class Airline {
  +name: string
}

class Airport {
  +code: string
  +city: string
}

class OperationCost {
  +cost: double
  +startDate: date
  +endDate: date
}

class AirlineOperation {
  +airline: Airline
  +airport: Airport
  +operationCost: OperationCost
}

Airline *-- AirlineOperation
Airport *-- AirlineOperation
AirlineOperation o-- OperationCost
@enduml
```



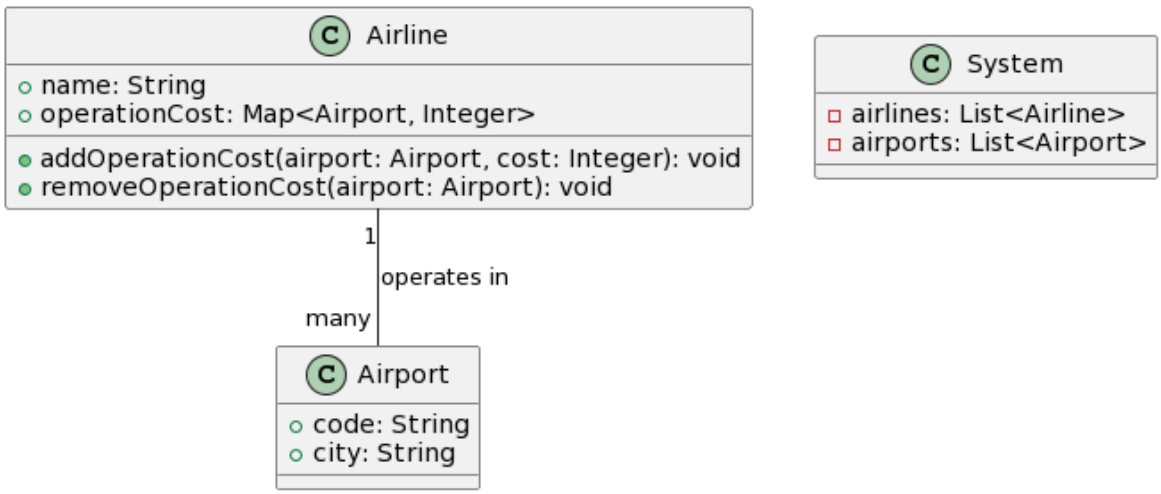
Session 2

Prompt 1

Many different airlines can operate in different airports. An airline has a name, and an airport has a code and is in a city. Each airline has associated a cost for operating in a specific airport. Besides, the operation of an airline in a specific airport can be started and stopped at any time, and its operating cost can be updated at any time. Please, draw a UML diagram for this system. Represent the diagram using PlantUML

ChatGPT Output 1

```
@startuml
class Airline {
  +name: String
  +operationCost: Map<Airport, Integer>
  +addOperationCost(airport: Airport, cost: Integer): void
  +removeOperationCost(airport: Airport): void
}

class Airport {
  +code: String
  +city: String
}

class System {
  -airlines: List<Airline>
  -airports: List<Airport>
}

Airline "1" -- "many" Airport: operates in
@enduml
```



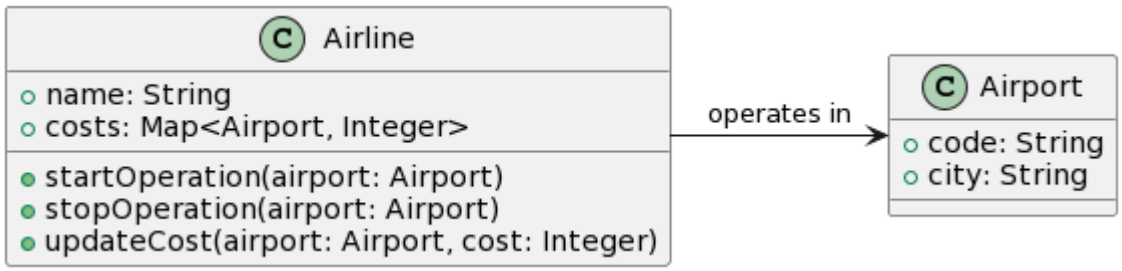
Session 3

Prompt 1

Many different airlines can operate in different airports. An airline has a name, and an airport has a code and is in a city. Each airline has associated a cost for operating in a specific airport. Besides, the operation of an airline in a specific airport can be started and stopped at any time, and its operating cost can be updated at any time. Please, draw a UML diagram for this system. Represent the diagram using PlantUML

ChatGPT Output 1

```
@startuml
class Airline {
  +name: String
  +costs: Map<Airport, Integer>
  +startOperation(airport: Airport)
  +stopOperation(airport: Airport)
  +updateCost(airport: Airport, cost: Integer)
}

class Airport {
  +code: String
  +city: String
}

Airline -> Airport : operates in
@enduml
```



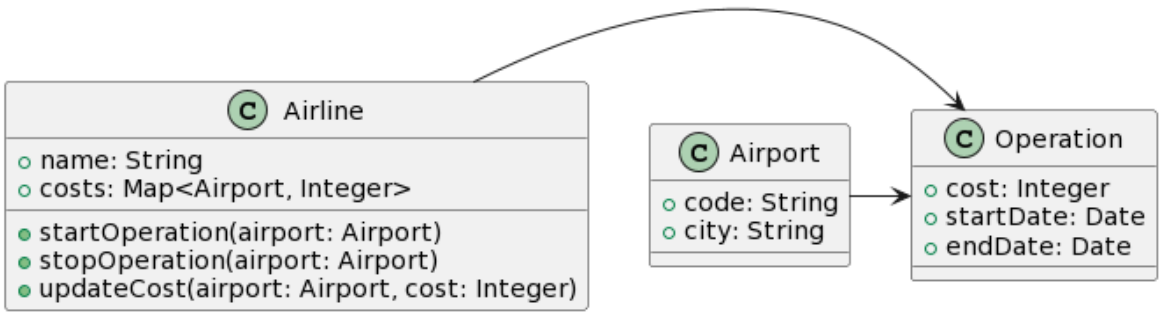
Prompt 2 (Trying to improve associations and remove operations)

The information about the relationship of airlines and airports should be in an association class. Can you please add an association class between airline and airport?

ChatGPT Output 2

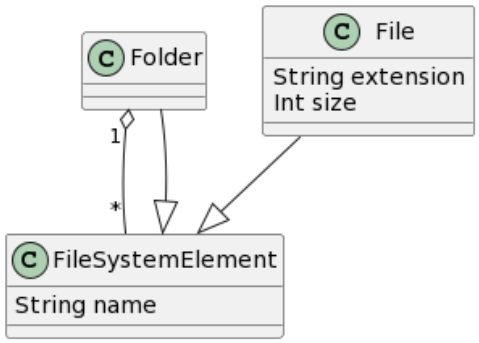
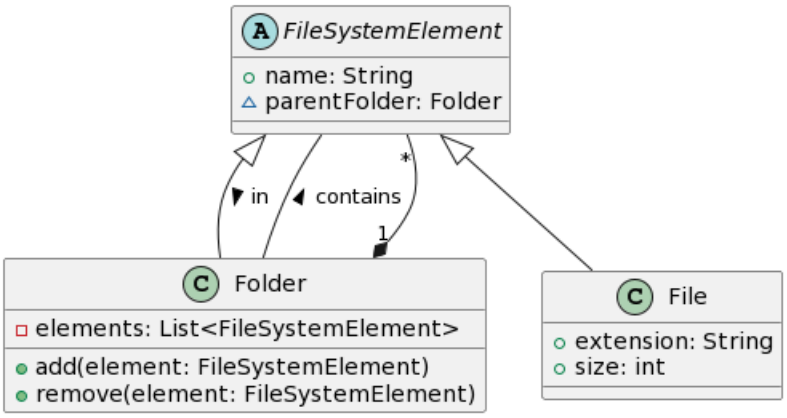
```
@startuml
class Airline {
  +name: String
```

```
  +costs: Map<Airport, Integer>
  +startOperation(airport: Airport)
  +stopOperation(airport: Airport)
  +updateCost(airport: Airport, cost: Integer)
}

class Airport {
  +code: String
  +city: String
}

class Operation {
  +cost: Integer
  +startDate: Date
  +endDate: Date
}

Airline -> Operation
Airport -> Operation

@enduml
```



Model: JC3 (FileSystem_3)

Date: 2023/03/01

User intent	ChatGPT output
	

Observations

- It is not able to create aggregation relationships, even when the word “aggregates” is explicitly written in the input text or when we explicitly ask to replace composition with aggregation.
- It sometimes includes attributes that are not derived from the input text, but they are simply “invented”.
- When asking to kindly improve the solution, ChatGPT can very likely worsen the solution, even inventing some new concepts.
- Even when explicitly trying ChatGPT to add an inheritance relationship, it does not manage to add it. Although once it added it.

Chat Log

Summary:

- In session 1, the diagram contains some information semantically wrong and some information that was not stated in the text. For instance, it adds a composition association between a file system element and a folder, or a composition association between a file and a file system element. Besides, no inheritance relation is added between Folder and file system element. After asking to improve the solution, ChatGPT outputs a solution where some more information comes from nowhere, such as different types of files.
- In session 2, we give the same input and get a completely different solution. In this case, Folder, again, does not inherit from file system element, and again, there are attributes that should not been there. Then I said explicitly that folder is a kind of file system element, and it did only not add the inheritance from folder and FileSystemElement, but it added the inheritance relationship in the wrong direction(!). Besides, now, the same inheritance relation from File to FileSystemElement is repeated.

- In session 3, we give the same input and get a solution similar to the one in the first session. Then, I explicitly asked it to include an inheritance relationship, which it did. I also asked to replace composition with aggregation, and it did not do it.

Session 1

Prompt 1

I would like the UML class diagram in PlantUML notation for the following scenario. We have file system elements, which can be a folder and a file. All file system elements have a name. Every file system element must belong to a folder, and folders can aggregate any number of file system elements. Files have information about their extension and their size. Please draw the UML class diagram in PlantUML.

ChatGPT Output 1

Here is the PlantUML notation for the UML class diagram:

```
@startuml

class FileSystemElement {
   -name: String
   -parentFolder: Folder
}

class Folder {
   -name: String
   -parentFolder: Folder
   -elements: List<FileSystemElement>
   +add(element: FileSystemElement)
   +remove(element: FileSystemElement)
}

class File {
   -extension: String
   -size: Integer
}

FileSystemElement *-- Folder : belongs to >
FileSystemElement <|-- File : is a >
Folder *-- "0..*" FileSystemElement : contains >
File *-- "1" FileSystemElement : is a >

@enduml
```

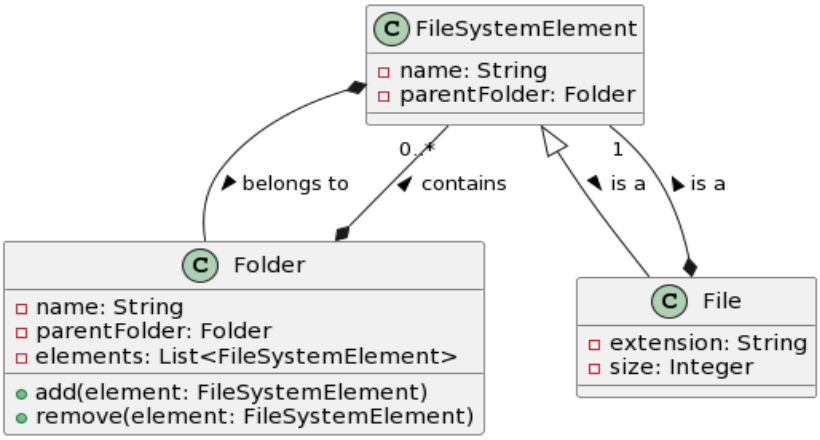
In this diagram, FileSystemElement is an abstract class that has two private attributes: name and

parentFolder. The Folder and File classes inherit from FileSystemElement.

Folder has an additional attribute elements, which is a list of FileSystemElement objects. It also has two public methods: add(element: FileSystemElement) and remove(element: FileSystemElement), which allow adding and removing elements to/from the elements list.

File has two private attributes: extension and size.

The relationships between classes are represented with arrows. FileSystemElement has a unidirectional association with Folder, which means that every FileSystemElement belongs to a Folder. Folder has a bidirectional aggregation relationship with FileSystemElement, which means that a Folder contains zero or more FileSystemElement objects. File has a unidirectional inheritance relationship with FileSystemElement, which means that every File is a FileSystemElement. Finally, File has a unidirectional association with FileSystemElement, which means that a File is contained in a FileSystemElement.



Prompt 2

I didn't see the option to regenerate the solution, so I explicitly wrote:
can you please improve the given solution?

ChatGPT Output 2 (Regenerated Response)

Sure, here is an improved version of the UML class diagram in PlantUML notation for the scenario you provided:

@startuml

```
abstract class FileSystemElement {
   -name: String
   -parentFolder: Folder
   {abstract}
}

class Folder {
   -elements: List<FileSystemElement>
   +add(element: FileSystemElement): void
   +remove(element: FileSystemElement): void
   +contains(element: FileSystemElement): boolean
}

class File {
   -extension: String
   -size: long
}

class TextFile {
   -charset: String
}

class ImageFile {
   -resolution: String
}

class AudioFile {
   -duration: long
}

FileSystemElement *-- Folder : contains >
FileSystemElement <|-- File : is a >
File <|-- TextFile : is a >
File <|-- ImageFile : is a >
File <|-- AudioFile : is a >

@enduml
```

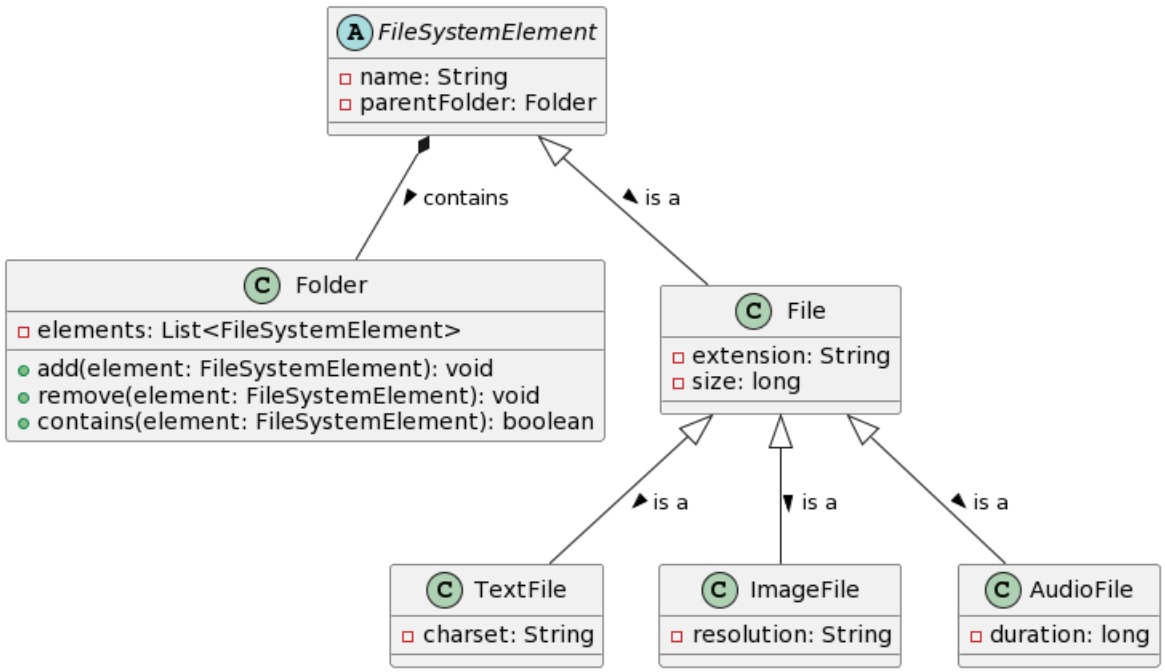
In this improved version, FileSystemElement is still an abstract class that has two private attributes: name and parentFolder. I added the {abstract} keyword to indicate that it is an abstract class.

Folder has an additional private attribute elements, which is a list of FileSystemElement objects. I removed the name attribute from Folder as it is already present in FileSystemElement. I also changed the visibility of add and remove methods to public, and added a contains method to check if a given FileSystemElement is present in the folder.

File has two private attributes: extension and size. I changed the type of the size attribute to long to support larger file sizes.

I also added three new classes TextFile, ImageFile, and AudioFile, which inherit from File. Each of these classes has an additional private attribute that is specific to its type: charset for TextFile, resolution for ImageFile, and duration for AudioFile.

Finally, I updated the relationships between the classes to reflect the new additions. FileSystemElement still has a unidirectional association with Folder, which means that every FileSystemElement belongs to a Folder. Folder still has a bidirectional aggregation relationship with FileSystemElement, which means that a Folder contains zero or more FileSystemElement objects. File now has three subclasses: TextFile, ImageFile, and AudioFile. Each of these subclasses inherits from File and has a unidirectional inheritance relationship with it.



Session 2

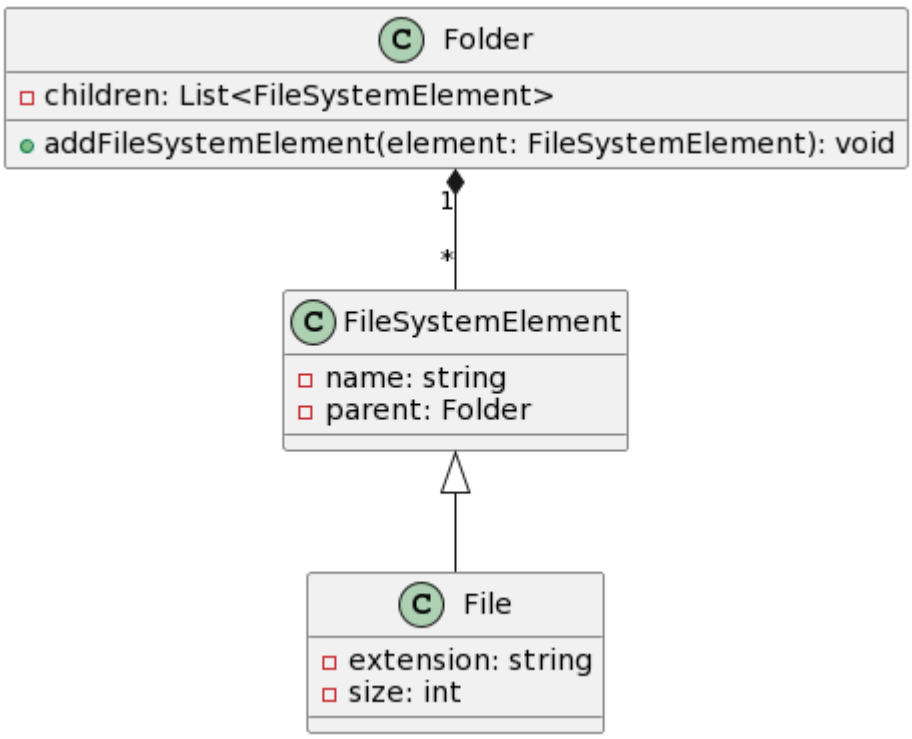
Prompt 1

I would like the UML class diagram in PlantUML notation for the following scenario. We have file system elements, which can be a folder and a file. All file system elements have a name. Every file system element must belong to a folder, and folders can aggregate any number of file system elements. Files have information about their extension and their size. Please draw the UML class diagram in PlantUML.

ChatGPT Output 1

Sure, here's a UML class diagram in PlantUML notation for the given scenario:

```
@startuml
class FileSystemElement {
  - name: string
  - parent: Folder
}

class Folder {
  - children: List<FileSystemElement>
  + addFileSystemElement(element: FileSystemElement): void
}

class File {
  - extension: string
  - size: int
}

Folder "1" *-- "*" FileSystemElement
FileSystemElement <|-- File

@enduml
```



Prompt 2 (Trying to improve inheritance)

I said before that folder is a kind of file system element, can you please improve the diagram?

ChatGPT Output 2

Sure, based on your feedback, here's an updated UML class diagram in PlantUML notation for the given scenario:

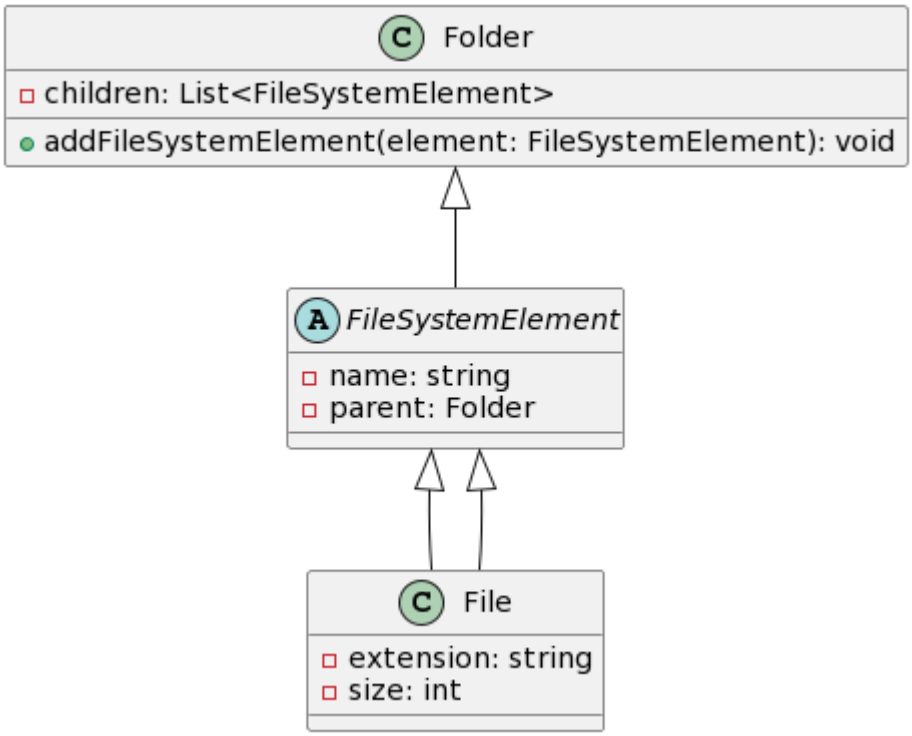
```
@startuml
abstract class FileSystemElement {
 - name: string
 - parent: Folder
}

class Folder {
 - children: List<FileSystemElement>
 + addFileSystemElement(element: FileSystemElement): void
}

class File extends FileSystemElement {
 - extension: string
 - size: int
}

Folder <|-- FileSystemElement
FileSystemElement <|-- File

@enduml
```

In this updated diagram, Folder is no longer a concrete class, but a subclass of FileSystemElement. FileSystemElement is still an abstract class, representing a file system element.

Folder inherits the name and parent attributes from FileSystemElement, and adds an attribute children (a list of FileSystemElement objects), and a method addFileSystemElement to add a file system element to the folder's children.

File is a concrete class extending FileSystemElement and adding two attributes: extension (a string) and size (an integer).

The relationships between the classes remain the same as in the previous diagram, with a generalization relationship between FileSystemElement and File, indicating that File is a subtype of FileSystemElement, and a specialization relationship between Folder and FileSystemElement, indicating that Folder is a kind of FileSystemElement.



Session 3

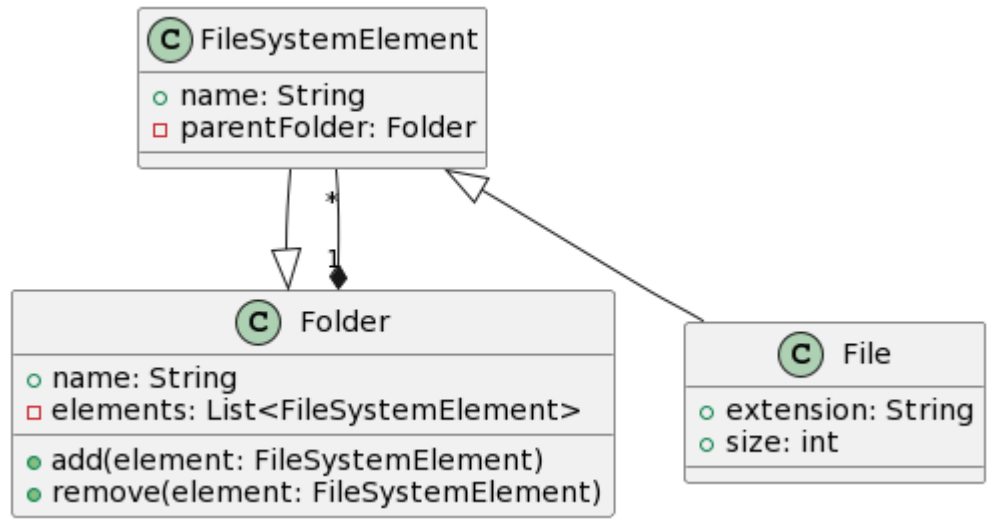
Prompt 1

I would like the UML class diagram in PlantUML notation for the following scenario. We have file system elements, which can be a folder and a file. All file system elements have a name. Every file system element must belong to a folder, and folders can aggregate any number of file system elements. Files have information about their extension and their size. Please draw the UML class diagram in PlantUML.

ChatGPT Output 1

Here's the PlantUML notation for the UML class diagram:

```
@startuml

class FileSystemElement {
  +name: String
  -parentFolder: Folder
}

class Folder {
  +name: String
  -elements: List<FileSystemElement>
  +add(element: FileSystemElement)
  +remove(element: FileSystemElement)
}

class File {
  +extension: String
  +size: int
}

FileSystemElement --|> Folder
FileSystemElement <|-- File
Folder "1" *-- "*" FileSystemElement

@enduml
```



Prompt 2 (Trying to improve inheritance and to include aggregation relationship)

Please, do not use composition but aggregation relationships. Note that Folder must inherit from FileSystemElement, and not the other way around. Please improve the class diagram in PlantUML notation

ChatGPT Output 2

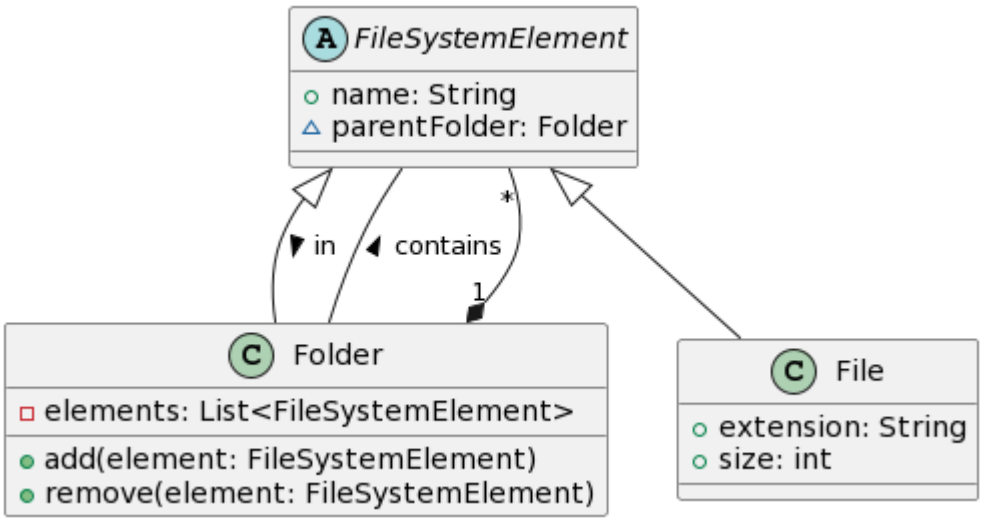
Here's the updated PlantUML notation for the UML class diagram with aggregation relationships:

```
@startuml

abstract class FileSystemElement {
  +name: String
  ~parentFolder: Folder
}

class Folder {
  -elements: List<FileSystemElement>
  +add(element: FileSystemElement)
  +remove(element: FileSystemElement)
}

class File {
  +extension: String
  +size: int
}

Folder -- FileSystemElement : contains >
FileSystemElement <|-- File
FileSystemElement <|-- Folder : in >
Folder "1" *-- "*" FileSystemElement

@enduml
```

Explanation:

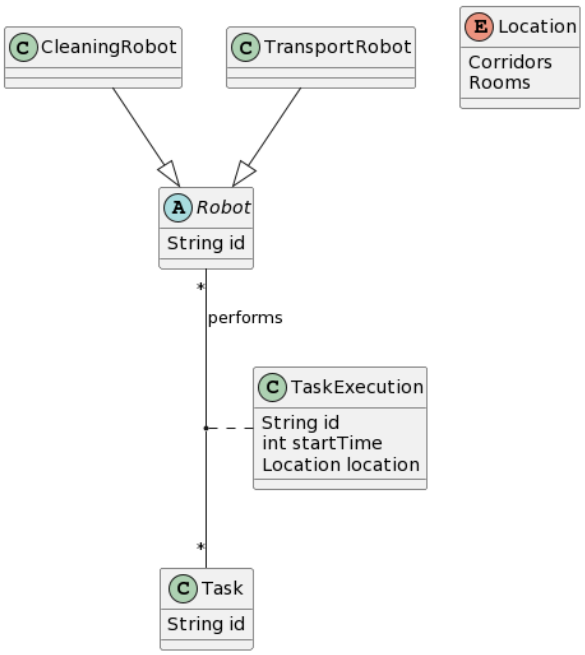
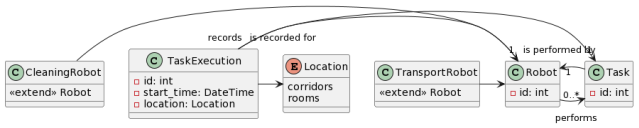
FileSystemElement is now an abstract class that contains the name attribute and a reference to the parentFolder, which is now a private field. This is to indicate that a file system element can be a child of only one folder.
Folder inherits from FileSystemElement, and contains a list of elements that can be any type of FileSystemElement.

File has the same attributes extension and size as before.
The relationships between classes are now aggregation relationships. Folder is connected to FileSystemElement with a contains relationship (a folder can contain many file system elements), and FileSystemElement is connected to File and Folder with an in relationship (a file or folder can belong to only one folder).
The multiplicity of the contains relationship is indicated by the 1 and * symbols on the Folder end of the line. This means that a folder can contain any number of file system elements.



Model: JC6 (Robots_2)

Date: 2023/03/01

User intent	ChatGPT output
	

Observations

- Association classes are not generated.
- The syntax in the diagram can be incorrect: for instance, a class that has a reference to an Enumeration class (such as in Session 2).
- There are attributes that are made up, no information about them was given in the prompt.
- When explicitly asking ChatGTP to add multiplicities in the associations, it didn't add them, but it assigned names and roles to the associations.

Chat Log

Summary:

- In Session 1, the diagram created was quite wrong. The hierarchy relation in robots was right, but robots were not related with tasks. The location attribute was of Location type instead of enumeration. **I then hit "Regenerate response" button.The solution was fairly improved**. Now there was an enumeration type, and task executions were related to tasks, although the problem is it was not an association class.
- In Session 2, we give the same text. There is no hierarchy in robots now, but simply associations, which is wrong. Again, there is no association class. There is the Enumeration class, which is correct, but there is an association of a class with an enumeration, which is wrong. **I then hit "Regenerate**

response" button.The solution was worsened. There is no reference now between a specific robot and the generic robot. And there is still a reference between a class and an enumeration class.

- Again in Session 3 we introduced the same text. Now the inheritance relationships were properly established, but the subclasses had completely invented attributes (batteryLevel and maxPayload). Since no multiplicities are added, I asked ChatGPT to add multiplicities to the diagram. It didn't add any multiplicities, but names and roles to the relationships.

Session 1

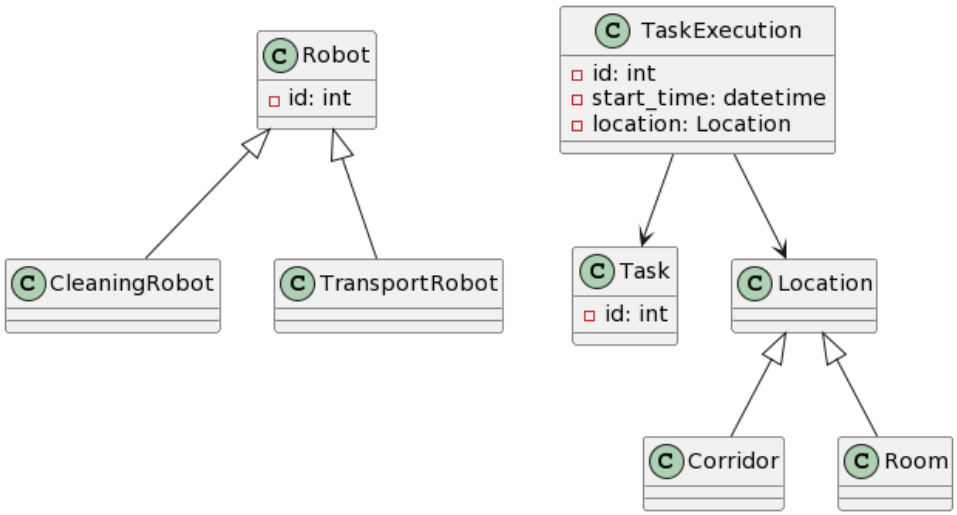
Prompt 1

I'd like you to create a UML class diagram using PlantUML notation of the following system. There are robots that have an id. Robots can be of type cleaning robot and transport robot. Robots can perform tasks, which have an id. When a robot performs a task, the task execution is recorded. A task execution has an id, a start time and a location. The location can be either corridors or rooms.

ChatGPT Output 1

```
@startuml

class Robot {
  - id: int
}

class CleaningRobot {
}

class TransportRobot {
}

class Task {
  - id: int
}

class TaskExecution {
  - id: int
  - start_time: datetime
  - location: Location
}

class Location {
}

class Corridor {
}

class Room {
}
```

```
Robot <|-- CleaningRobot
Robot <|-- TransportRobot

TaskExecution --> Task
TaskExecution --> Location

Location <|-- Corridor
Location <|-- Room

@enduml
```

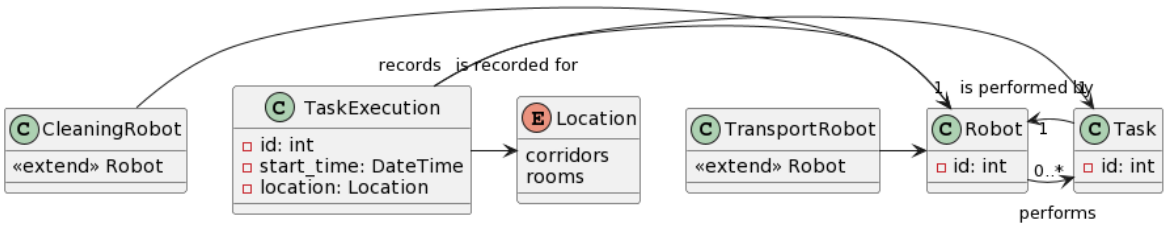


Prompt 2

ChatGPT Output 2 (Regenerated Response)

```
@startuml
class Robot {
  - id: int
}

class CleaningRobot {
  <<extend>> Robot
}

class TransportRobot {
```

```
  <<extend>> Robot
}

class Task {
  - id: int
}

class TaskExecution {
  - id: int
  - start_time: DateTime
  - location: Location
}

enum Location {
  corridors
  rooms
}

CleaningRobot -> Robot
TransportRobot -> Robot

Robot "performs" -> "0..*" Task
Task "is performed by" -> "1" Robot

TaskExecution "records" -> "1" Robot
TaskExecution "is recorded for" -> "1" Task

TaskExecution -> Location

@enduml
```



Session 2

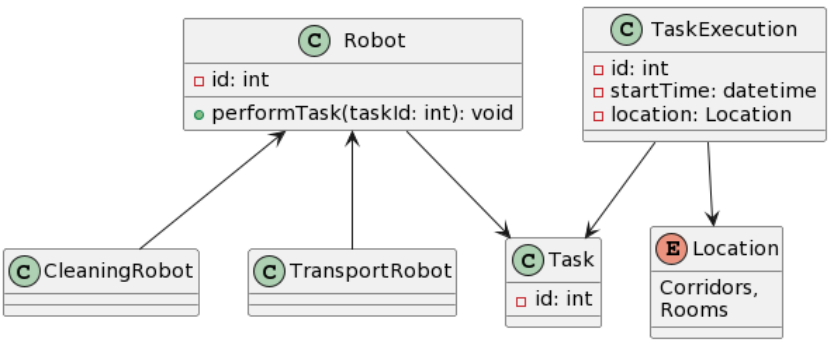
Prompt 1

I’d like you to create a UML class diagram using PlantUML notation of the following system. There are robots

that have an id. Robots can be of type cleaning robot and transport robot. Robots can perform tasks, which have an id. When a robot performs a task, the task execution is recorded. A task execution has an id, a start time and a location. The location can be either corridors or rooms.

ChatGPT Output 1

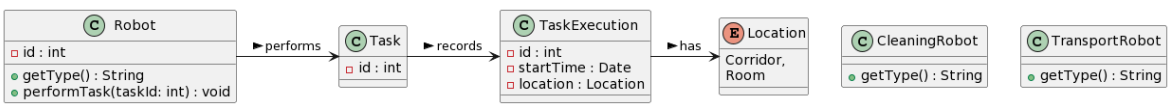
```
@startuml
class Robot {
   -id: int
   +performTask(taskId: int): void
}

class CleaningRobot {
}

class TransportRobot {
}

class Task {
   -id: int
}

class TaskExecution {
   -id: int
   -startTime: datetime
   -location: Location
}

enum Location {
   Corridors,
   Rooms
}

Robot --> Task
Robot <-- CleaningRobot
Robot <-- TransportRobot
TaskExecution --> Location
TaskExecution --> Task
@enduml
```



Prompt 2

ChatGPT Output 2 (regenerated response)

```
@startuml
class Robot {
 -id : int
 +getType() : String
 +performTask(taskId: int) : void
}

class CleaningRobot {
 +getType() : String
}

class TransportRobot {
 +getType() : String
}

class Task {
 -id : int
}

class TaskExecution {
 -id : int
 -startTime : Date
 -location : Location
}

enum Location {
 Corridor,
 Room
}

Robot -> Task : performs >
Task -> TaskExecution : records >
TaskExecution -> Location : has >
@enduml
```



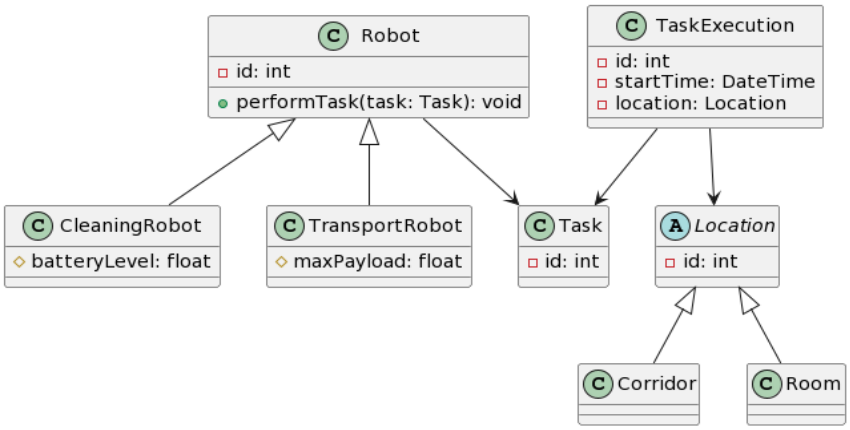
Session 3

Prompt 1

I'd like you to create a UML class diagram using PlantUML notation of the following system. There are robots that have an id. Robots can be of type cleaning robot and transport robot. Robots can perform tasks, which have an id. When a robot performs a task, the task execution is recorded. A task execution has an id, a start time and a location. The location can be either corridors or rooms.

ChatGPT Output 1

```
@startuml
class Robot {
  - id: int
  + performTask(task: Task): void
}

class CleaningRobot {
  # batteryLevel: float
}

class TransportRobot {
  # maxPayload: float
}

class Task {
  - id: int
}

class TaskExecution {
  - id: int
  - startTime: DateTime
  - location: Location
}

abstract class Location {
  - id: int
}

class Corridor {
}

class Room {
}

Robot --> Task
Robot <|-- CleaningRobot
```



```
Robot <|-- TransportRobot
TaskExecution --> Task
TaskExecution --> Location
Location <|-- Corridor
Location <|-- Room
@enduml
```

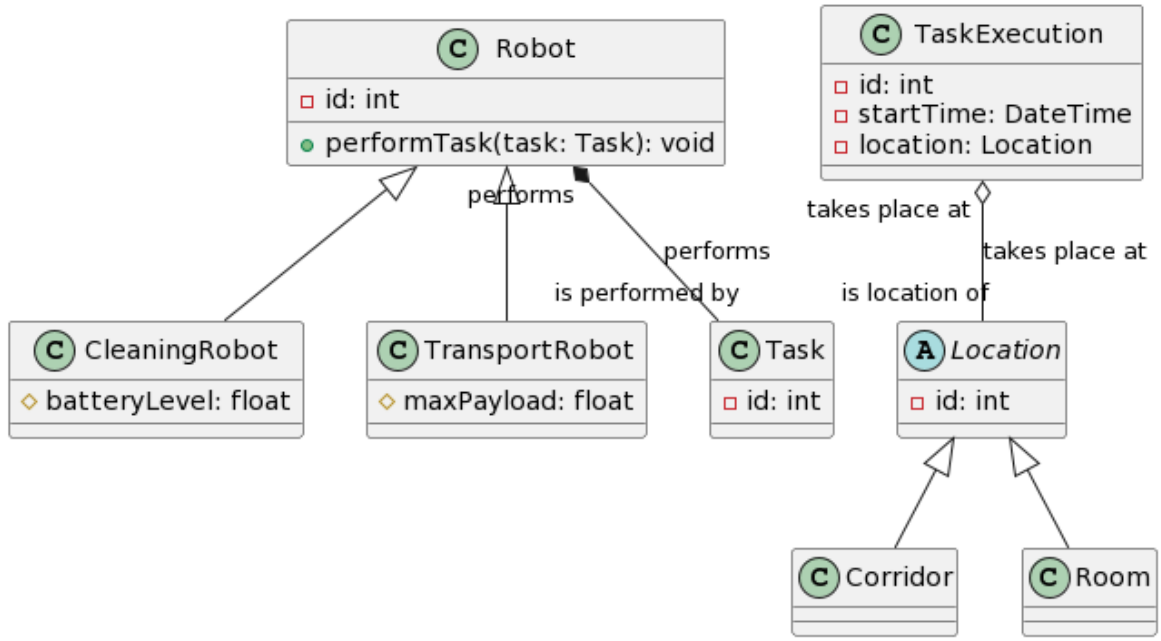
Prompt 2 (Trying to add multiplicities)

You added no multiplicities in the associations in the diagram. Can you please add multiplicities?

ChatGPT Output 2

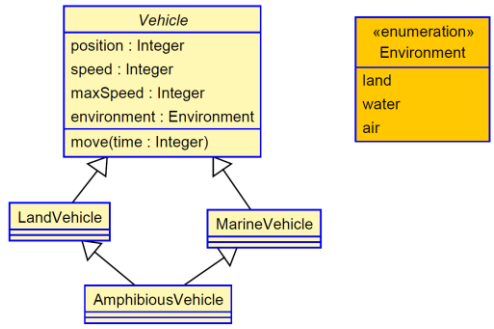
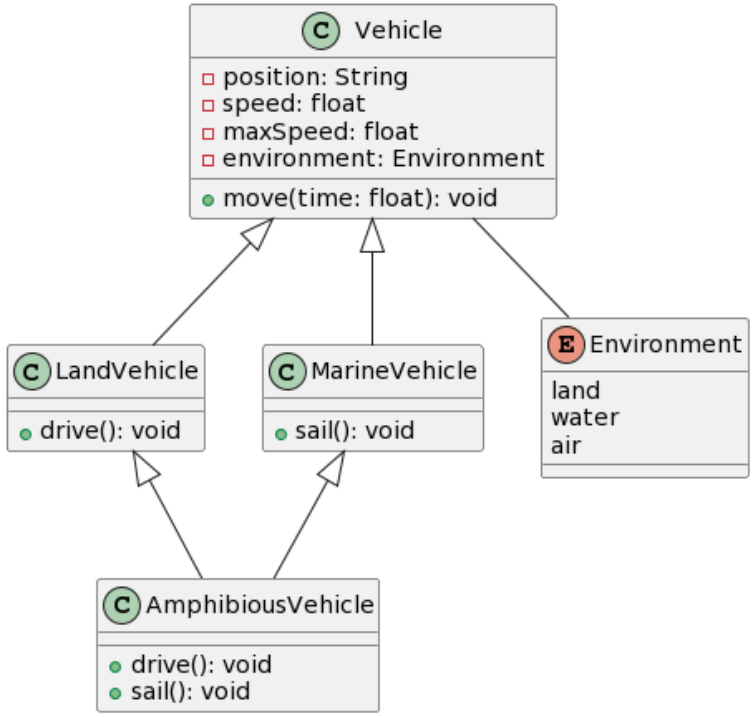
```
@startuml
class Robot {
 - id: int
 + performTask(task: Task): void
}

class CleaningRobot {
 # batteryLevel: float
}

class TransportRobot {
 # maxPayload: float
}

class Task {
 - id: int
}

class TaskExecution {
 - id: int
 - startTime: DateTime
 - location: Location
}

abstract class Location {
 - id: int
```

```
}

class Corridor {
}

class Room {
}

Robot "performs" *-- "is performed by" Task : performs
Robot <|-- CleaningRobot
Robot <|-- TransportRobot
TaskExecution "takes place at" o-- "is location of" Location : takes place at
Location <|-- Corridor
Location <|-- Room
@enduml
```



Model: AV1 (Amphibious)

Date: 2023/03/02

User intent	ChatGPT output
	

Observations

- Some attributes are made up, such information was not given in the prompt.
- Once, when asking explicitly to improve the solution, it returned exactly the same solution as the previous one.
- The syntax in the diagram can be incorrect: for instance, a class that has a reference to an Enumeration class (such as in Session 2).
- When explicitly asking to remove the previous reference, it did not remove it.

Chat Log

Summary:

- In the first session, the solution approached the user intent, but there was not multi-inheritance. Besides, there were attributes in some vehicles that had not been given in the text. There was also no enumeration class for the environment. I then asked to improve the solution (as the “regenerate solution” button was not available). Nothing was changed.
- In Session 2 I entered the same prompt, but the diagram was completely different, it was worse. The inheritance relationships were added in the wrong direction. The position was entered as a new class

that has latitude and longitude. This time, the environment was added as an enumeration, but this enumeration class had a reference to another class. I explicitly asked to improve these relationships. This time, the diagram was improved, and multi-inheritance was added. I tried to correct the relationships by explicitly saying it, as well as to remove the Position class. The attribute was corrected, but the association was not removed.

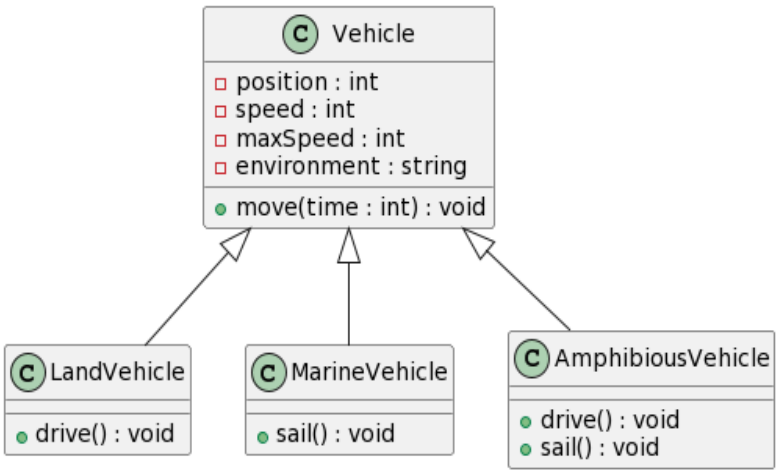
Session 1

Prompt 1

A Vehicle has a position, a speed and a maximum speed. Besides, it has an environment (with a value among “land”, “water” and “air”). Finally, it can move for a given time. Those are the features of a generic vehicle. Then, we have land vehicles and marine vehicles as specific types of vehicles, where both have all the same features as a vehicle. Finally, an amphibious vehicle can be both a land vehicle and a marine vehicle. Please, draw a UML class diagram with PlantUML notation.

ChatGPT Output 1

```
@startuml

class Vehicle {
  - position : int
  - speed : int
  - maxSpeed : int
  - environment : string
  + move(time : int) : void
}

class LandVehicle {
  + drive() : void
}

class MarineVehicle {
  + sail() : void
}

class AmphibiousVehicle {
  + drive() : void
  + sail() : void
}

Vehicle <|-- LandVehicle
Vehicle <|-- MarineVehicle
Vehicle <|-- AmphibiousVehicle

@enduml
```

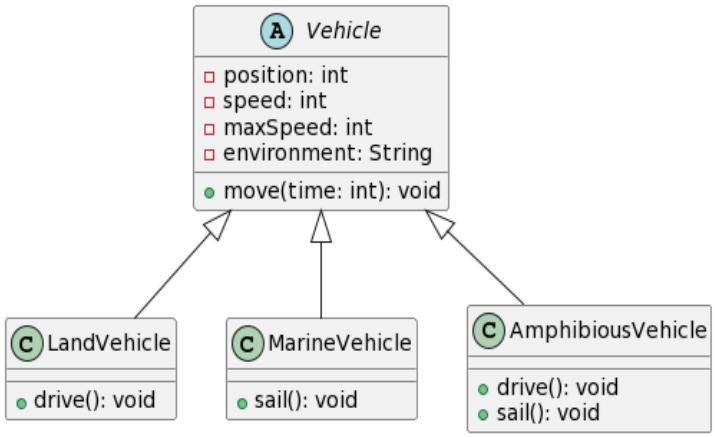


Prompt 2

can you please improve the solution?

ChatGPT Output 2

```
@startuml

abstract class Vehicle {
  - position: int
  - speed: int
  - maxSpeed: int
  - environment: String
  + move(time: int): void
}

class LandVehicle {
  + drive(): void
}

class MarineVehicle {
  + sail(): void
}

class AmphibiousVehicle {
  + drive(): void
  + sail(): void
}

Vehicle <|-- LandVehicle
Vehicle <|-- MarineVehicle
Vehicle <|-- AmphibiousVehicle

@enduml
```



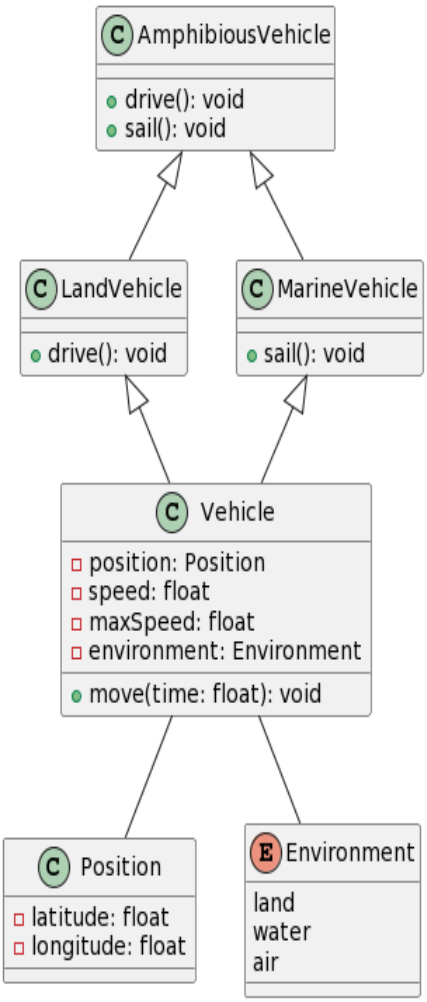
Session 2

Prompt 1

A Vehicle has a position, a speed and a maximum speed. Besides, it has an environment (with a value among “land”, “water” and “air”). Finally, it can move for a given time. Those are the features of a generic vehicle. Then, we have land vehicles and marine vehicles as specific types of vehicles, where both have all the same features as a vehicle. Finally, an amphibious vehicle can be both a land vehicle and a marine vehicle. Please, draw a UML class diagram with PlantUML notation.

ChatGPT Output 1

```
@startuml

class Vehicle {
  -position: Position
  -speed: float
  -maxSpeed: float
  -environment: Environment
  +move(time: float): void
}

class LandVehicle {
  +drive(): void
}

class MarineVehicle {
  +sail(): void
}

class AmphibiousVehicle {
  +drive(): void
  +sail(): void
}

class Position {
  -latitude: float
  -longitude: float
}

enum Environment {
  land
  water
  air
}

Vehicle -- Position
Vehicle -- Environment
LandVehicle <|-- Vehicle
MarineVehicle <|-- Vehicle
```



```
AmphibiousVehicle <|-- LandVehicle
AmphibiousVehicle <|-- MarineVehicle

@enduml
```

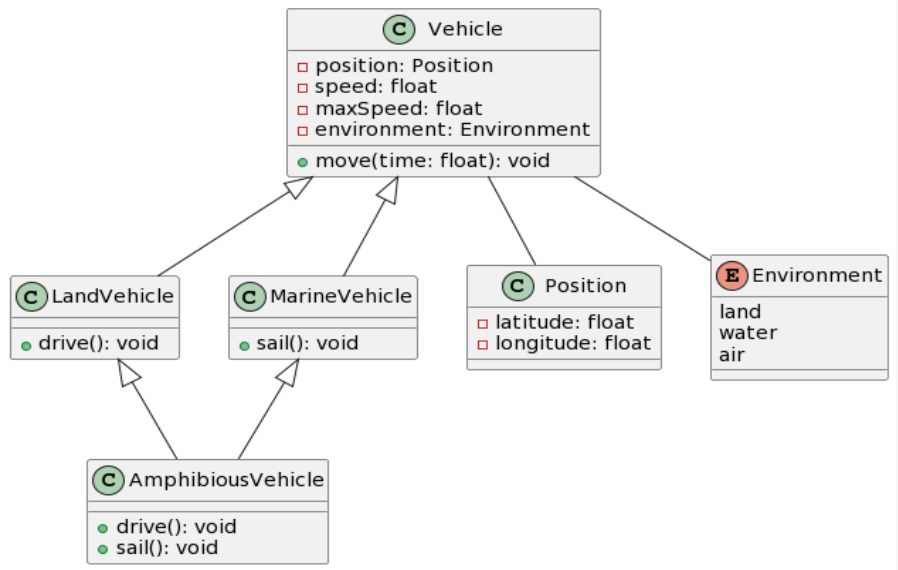
Prompt 2

You got the inheritance relationships wrong, they are all reversed. Can you please correct the relationships and improve the diagram?

ChatGPT Output 2

```
@startuml

class Vehicle {
  -position: Position
  -speed: float
  -maxSpeed: float
  -environment: Environment
  +move(time: float): void
}

class LandVehicle {
  +drive(): void
}

class MarineVehicle {
  +sail(): void
}

class AmphibiousVehicle {
  +drive(): void
  +sail(): void
}

class Position {
  -latitude: float
  -longitude: float
}
```

```
enum Environment {
 land
 water
 air
}

Vehicle <|-- LandVehicle
Vehicle <|-- MarineVehicle
LandVehicle <|-- AmphibiousVehicle
MarineVehicle <|-- AmphibiousVehicle

Vehicle -- Position
Vehicle -- Environment

@enduml
```

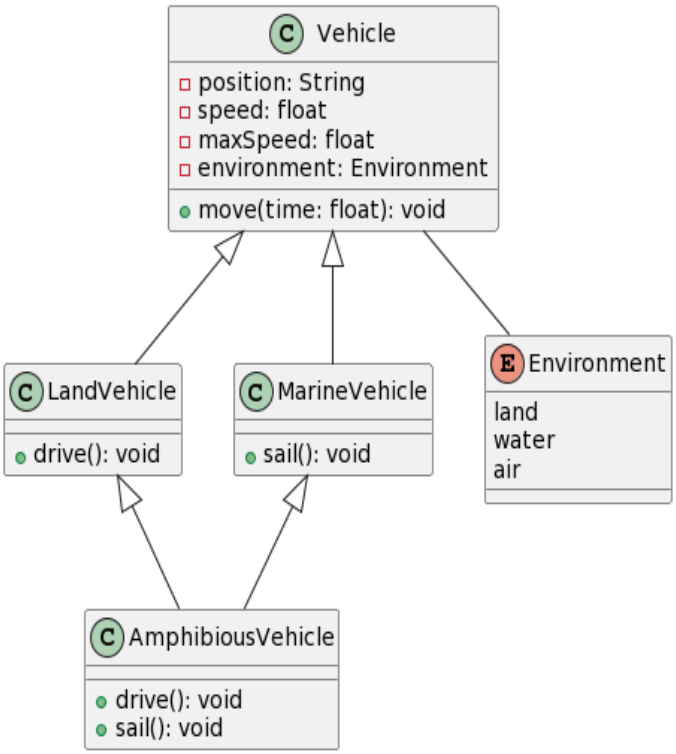


Prompt 3

The position of the vehicle must be of type String. Also, enumeration classes must not have an association with any other class. Please, improve the diagram considering these two aspects

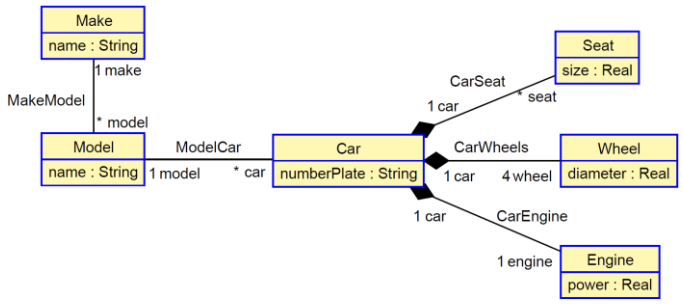
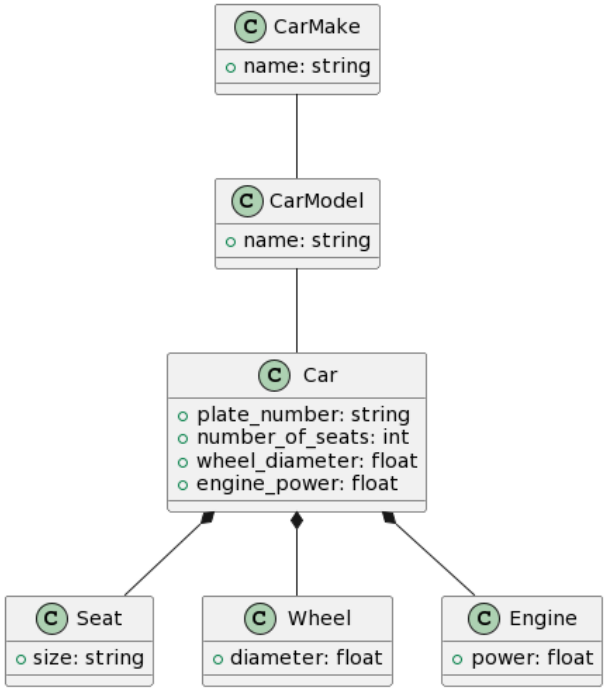
ChatGPT Output 3

```
@startuml

class Vehicle {
 -position: String
 -speed: float
 -maxSpeed: float
 -environment: Environment
 +move(time: float): void
}

class LandVehicle {
 +drive(): void
}

class MarineVehicle {
 +sail(): void
}

class AmphibiousVehicle {
 +drive(): void
 +sail(): void
}

enum Environment {
 land
 water
 air
}

Vehicle <|-- LandVehicle
Vehicle <|-- MarineVehicle
LandVehicle <|-- AmphibiousVehicle
MarineVehicle <|-- AmphibiousVehicle

Vehicle -- Environment

@enduml
```



Model: AV8 (Cars)

Date: 2023/03/02

User intent	ChatGPT output
	

Observations

- It is not able to create aggregation relationships, even when the word “aggregates” is explicitly written in the input text or when we explicitly ask to replace composition with aggregation.
- It sometimes includes attributes that are not derived from the input text, but they are simply “invented”.
- When asking to kindly improve the solution, ChatGPT can very likely worsen the solution, even inventing some new concepts.
- Even when explicitly trying ChatGPT to add an inheritance relationship, it does not manage to add it. Although once it added it.
- When explicitly asking to improve a certain aspect of the diagram, the new solution might improve that particular aspect (or not), but it will likely also change other parts in the diagram that were not specified in the prompt.

Chat Log

Summary:

- In Session 1, with the text introduced, the solution was quite good. It added two operations in Car that weren't asked. Also numSeats attribute in the Car was not necessary. All associations were simple, to compositions were added. But, in general, good solution.
- In Session 2 I entered the same prompt, and the solution was also quite good. This time, no multiplicities were added in the associations, and composition associations were added this time. There were several attributes in the car that were not needed (all except the first one). **After regenerating the response, the solution was worsened.** For instance, class Seat disappeared.
- In Session 3, with the same prompt, a similar solution was given. Now, attributes were correct but associations were simple (not a composition). I explicitly asked ChatGPT to convert the association to composition. It changed the associations to composition, but it also modified other things in the diagram, such as multiplicities or the addition of many attributes in Car.

Session 1

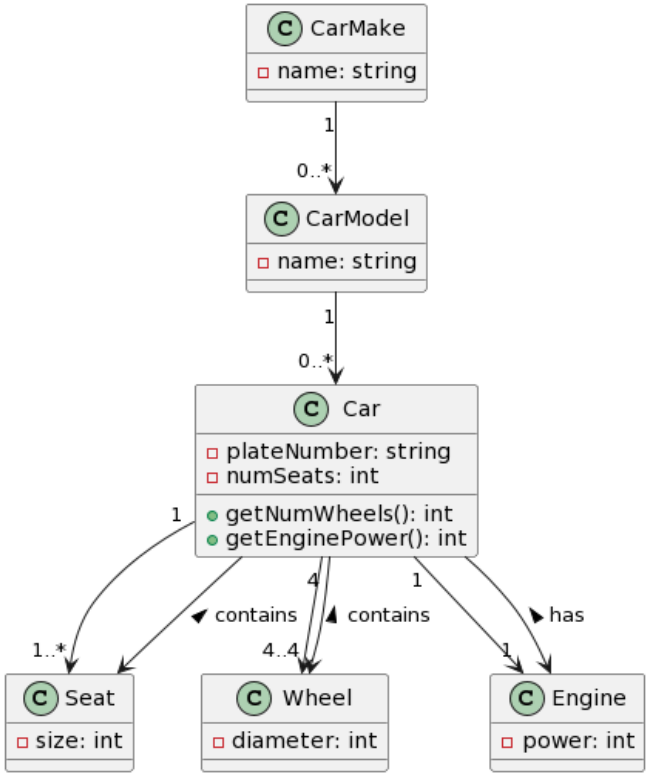
Prompt 1

A car make has a name, and it can have several models (each model has a name). For each model, there can be any number of cars, where each car has a plate number. Every car can have any number of seats, where each seat has a size. Also, all cars have 4 wheels, and each wheel has a diameter. Finally, all cars have one engine, where each engine has a specific power. Please draw the UML diagram using PlantUML notation.

ChatGPT Output 1

```
@startuml

class CarMake {
  - name: string
}

class CarModel {
  - name: string
}

class Car {
  - plateNumber: string
  - numSeats: int
  + getNumWheels(): int
  + getEnginePower(): int
}

class Seat {
  - size: int
}
```

```
class Wheel {
  - diameter: int
}

class Engine {
  - power: int
}

CarMake "1" --> "0..*" CarModel
CarModel "1" --> "0..*" Car
Car "1" --> "1..*" Seat
Car "4" --> "4..4" Wheel
Car "1" --> "1" Engine

Seat <-- Car : contains >
Wheel <-- Car : contains >
Engine <-- Car : has >

@enduml
```



Session 2

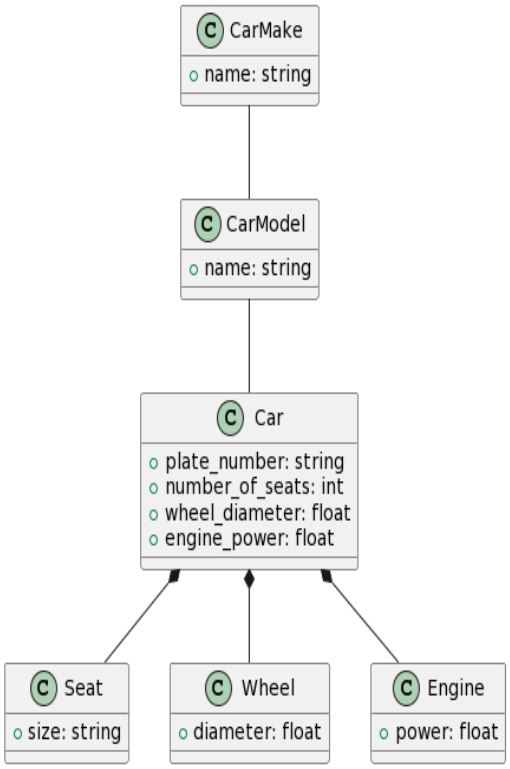
Prompt 1

A car make has a name, and it can have several models (each model has a name). For each model, there can be any number of cars, where each car has a plate number. Every car can have any number of seats, where each seat has a size. Also, all cars have 4 wheels, and each wheel has a diameter. Finally, all cars have one engine, where each engine has a specific power. Please draw the UML diagram using PlantUML notation.

ChatGPT Output 1

```
@startuml

class CarMake {
   +name: string
}

class CarModel {
   +name: string
}
```

```
class Car {
   +plate_number: string
   +number_of_seats: int
   +wheel_diameter: float
   +engine_power: float
}

class Seat {
   +size: string
}

class Wheel {
   +diameter: float
}

class Engine {
   +power: float
}

CarMake -- CarModel
CarModel -- Car
Car *-- Seat
Car *-- Wheel
Car *-- Engine

@enduml
```



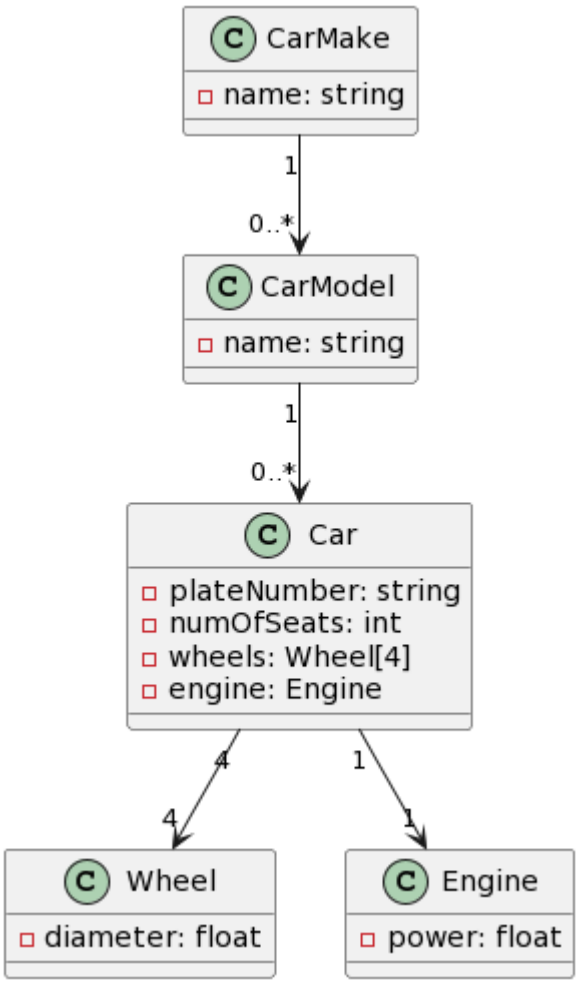
Prompt 2

I

ChatGPT Output 2 (Regenerate solution)

```
@startuml

class CarMake {
   - name: string
}

class CarModel {
   - name: string
}
```

```
class Car {
   - plateNumber: string
   - numOfSeats: int
   - wheels: Wheel[4]
   - engine: Engine
}

class Wheel {
   - diameter: float
}

class Engine {
   - power: float
}

CarMake "1" --> "0..*" CarModel
CarModel "1" --> "0..*" Car
Car "4" --> "4" Wheel
Car "1" --> "1" Engine

@enduml

@startuml

class CarMake {
   - name: string
}

class CarModel {
   - name: string
}

class Car {
   - plateNumber: string
   - numOfSeats: int
   - wheels: Wheel[4]
   - engine: Engine
}

class Wheel {
   - diameter: float
}
```



```
class Engine {
   - power: float
}

CarMake "1" --> "0..*" CarModel
CarModel "1" --> "0..*" Car
Car "4" --> "4" Wheel
Car "1" --> "1" Engine

@enduml
```

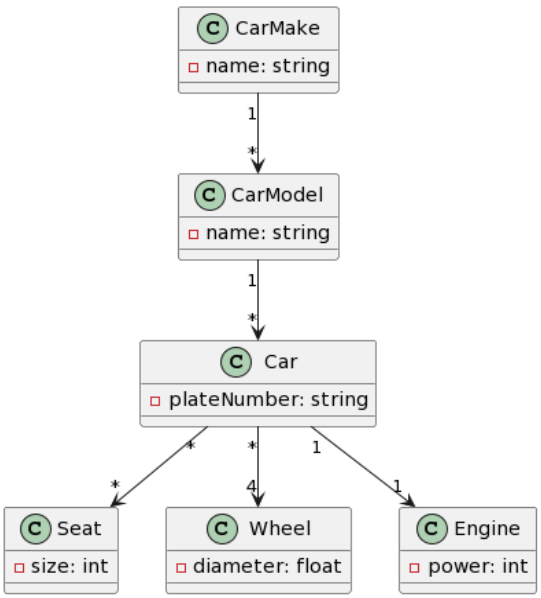
Session 3

Prompt 1

A car make has a name, and it can have several models (each model has a name). For each model, there can be any number of cars, where each car has a plate number. Every car can have any number of seats, where each seat has a size. Also, all cars have 4 wheels, and each wheel has a diameter. Finally, all cars have one engine, where each engine has a specific power. Please draw the UML diagram using PlantUML notation.

ChatGPT Output 1

```
@startuml

class CarMake {
  -name: string
}

class CarModel {
  -name: string
}

class Car {
  -plateNumber: string
}

class Seat {
  -size: int
}

class Wheel {
  -diameter: float
```



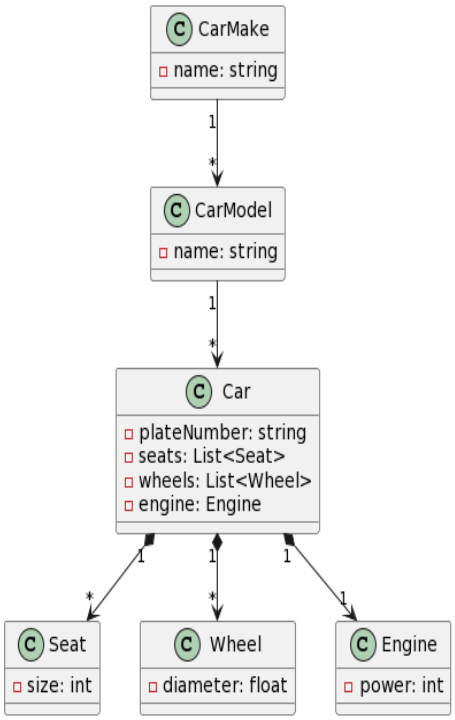
```
}

class Engine {
  -power: int
}

CarMake "1" --> "*" CarModel
CarModel "1" --> "*" Car
Car "*" --> "*" Seat
Car "*" --> "4" Wheel
Car "1" --> "1" Engine

@enduml
```

Prompt 2 (Trying to add composition associations)

the associations between the Car and Seat, Wheel and Engine should not be simple, but composition associations. Please improve the diagram

ChatGPT Output 2

```
@startuml

class CarMake {
  -name: string
}

class CarModel {
  -name: string
}

class Car {
  -plateNumber: string
  -seats: List<Seat>
  -wheels: List<Wheel>
  -engine: Engine
}

class Seat {
  -size: int
```



```
}

class Wheel {
  -diameter: float
}

class Engine {
  -power: int
}

CarMake "1" --> "*" CarModel
CarModel "1" --> "*" Car
Car "1" *--> "*" Seat
Car "1" *--> "*" Wheel
Car "1" *--> "1" Engine

@enduml
```