

# Cinema visit

• Read the guideline before start

You have opened your own cinema. To have a better idea of what's going on in the cinema you decided to keep a record of events in the cinema. For this purpose you have to create such modules:

- 1. In directory app create package cinema. In this package create modules:
  - o bar.py inside this module create CinemaBar class that describes work of cinema bar. This class should have only one static method sell\_product, that takes product name of the product that customer wants and customer Customer instance, that means customer. The sell\_product method sells a product to the customer and displays which product was sold and to whom.

```
cb = CinemaBar()
customer = Customer("Bob", "popcorn")
cb.sell_product(customer=customer, product=customer.food)
# Cinema bar sold popcorn to Bob.
```

o hall.py - inside this module create CinemaHall class that describes actions during the movie session. Its \_\_init\_\_ method takes and stores ONLY the number of the hall in the cinema. This class should have only one method movie\_session, that takes movie\_name, customers - list of a customers (Customer instances), cleaning\_staff - cleaner (Cleaner instance). This method prints about movie start, calls customers method watch\_movie, prints about movie end, calls cleaner method clean\_hall. So, we are expecting that everything listed above will be performed in movie session function.

```
hall = CinemaHall(hall_number=5)
movie_name = "Madagascar"
customers = [
    Customer(name="Bob", food="Coca-cola"),
    Customer(name="Alex", food="popcorn")
]
cleaning_staff = Cleaner(name="Anna")
hall.movie_session(movie_name=movie_name, customers=customers, cleaning_staff=cleaning_staff)
```

- 2. In directory app create package people . In this package create modules:
  - o customer.py inside this module create Customer class, its <u>\_\_init\_\_</u> method takes and stores name, food food that customer wants to buy in cinema bar. This class should have only one method watch\_movie, this method takes movie and prints what movie customer is watching.

```
bob = Customer(name="Bob", food="popcorn")
bob.watch_movie(movie="Madagascar")
# Bob is watching "Madagascar".
```

cinema\_staff.py - inside this module create cleaner class, its \_\_init\_\_ method takes and stores
 name . This class should have only one method clean\_hall , this method takes hall\_number number of hall that cleaner have to clean and prints that cleaner is cleaning that hall.

```
anna = Cleaner(name="Anna")
anna.clean_hall(hall_number=5)
# Cleaner Anna is cleaning hall number 5.
```

In the module <code>main.py</code> you have to import all this classes. Classes should be imported by absolute path, that starts with 'app.' with keyword 'from'. Write a function <code>cinema\_visit</code> that takes <code>movie</code>, <code>customers</code> - a list of customers, elements are dicts with 'name' and desired 'food' of a customer, <code>hall\_number</code> - number of the hall in cinema, <code>cleaner</code> - name of the cleaner, that will clean the hall after movie session.

This function should create instances of <code>customer</code>, <code>cinemaHall</code>, and <code>cleaner</code>. First, the cinema bar should sell food to customers. To do this, you can use the <code>cinemaBar</code> class without creating an instance. Then, the cinema hall should schedule a movie session, and finally, a cleaner should clean the cinema hall. We expect each class to work with the provided data, accepting parameters in the correct order and having the necessary methods. No additional checks or error handling are needed!

Example (do not add it to main.py):

ſĊ

Important Note: Each method responsible for performing a task should only print a message using the print() function. There is no need to return anything or use the logging module.



#### Releases

No releases published Create a new release

### **Packages**

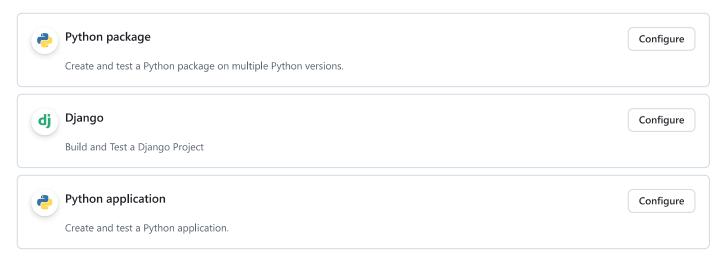
No packages published Publish your first package

#### Languages

Python 100.0%

## Suggested workflows

Based on your tech stack



More workflows

Dismiss suggestions