## Complete the Tasks

Make sure you take notes as you code while completing the tasks. These notes will help you to answer the questionnaire once you have completed all the tasks.

## Warm Up Scenario:

1) Add a new `Feature Mapping` named `.feature-to-folder` file within the `Items` directory.

    - Ensure that the feature *`Data`* is defined in the Feature Model using the `Feature Model View Window` found at the bottom left.
    - Map the feature *`Data`* into the new `.feature-to-folder` file to create its mapping.

By completing these steps, you've successfully mapped the feature *`Data`* to the `Items` directory.

2) Add a new `Feature Mapping` named `.feature-to-file` file within the `IO` directory.

    - Ensure that the feature *`Membership`* is defined in the Feature Model using the `Feature Model View Window` found at the bottom left.
    - Map the feature *`Membership`* to the file `Controller.java` into the new `.feature-to-file` file to create its mapping.

By completing these steps, you've successfully mapped the feature *`Membership`* to the file within the *`IO`* directory.

3) Navigate all the usages of the feature *`Search`* using the Feature Model.

    - To ensure exactly where the features have been annotated in the source code. Use the `Feature Model View Window` at the bottom left, by right-clicking on the feature *`Search`* and selecting the `Find Usages`.

By completing these steps, you've successfully found the feature *`Search`* used in all places `6 results`.

4) Adding a feature to code mapping.

    - The feature *`Review`* is implemented in the `Rentable()` class. Your task is to identify the corresponding code fragment and annotate it with the feature *`Review`*.

      ***Hint**: You will need a begin and end annotation for the feature annotation.*

## Scenario 1: Task*: Importing Data from Text Files*

Task: Implementing ImportData Feature

**Objective:**

Your task is to add a new feature called `DataLogging` to the DARTPlus system, specifically implementing the `ImportData` functionality. This feature will enable the system to read data from the text file `dartData.txt` containing information about employees, songs, games, and customers, and register them accordingly. Please note that employees and customers have dedicated array lists (employees and customers), whereas rentable items (songs and games) can be added to the same array list (i.e., items).

Integration Steps:
1. Feature Addition:

- Open the feature model of the DARTPlus system.

- Add a new feature `DataLogging` to the `feature model`.

- Create two sub-features under `DataLogging` named `ImportData` and `ExportData`.

2. Code Implementation and Annotation:

- Navigate to the class `Manager` in the directory `Persons` in the DARTPlus source code.

- Implement a method named `readFile()` within the `Manager` class.

- Design the method to read data from a txt file (called `dartData.txt`) and parse each line to extract relevant information about employees, songs, games, and customers.

***Note:*** The `dartData.txt` you can find in GitHub repository.

-Register the Employees in array list **`employees of type employee`**, Songs and Games in **`Items of type Rentable`**, and Customers in the **`customers of type Customer`**.

**Help:**
For file reading, you can use BufferedReader:

```
public void readFile(ArrayList<Employee> employees, ArrayList<Customer>
customers, ArrayList<Rentable> items, Scanner input) {
 try {
   File dartData = new File("dartData.txt");
   FileReader fr = new FileReader(dartData);
   BufferedReader br = new BufferedReader(fr);
    String line;
   while((line = br.readLine()) != null) {
      String[] dartInfo = line.split(";");// parse data
//register employees, songs, and games based on the first word of
the line (employee, game, song)
//For registering, use methods this.registerEmployee(),
items.add(new Game or Song), and customers.add()
}
 catch (Exception var12)
    { System.out.println("File does not exist"); }
```

- You are supposed to **annotate** the **implemented code** with **embedded feature annotations**.

3. Menu Option Addition:

- Navigate to the Controller class.

- Add a menu option for data importing in the manager's menu options within the Controller Class. For that, go to the `managerLoggedIn()` method in the `Controller` class, and add a print statement adding a new option `System.out.println("b. Import data from text file");`

- Lastly, put a new case in the body of the `managerLoggedIn()` method that calls the `readFile()method`.

- You are supposed to **annotate** the **implemented code** with **embedded feature annotations**.

Expected Outcome: By completing this task, you will enable the DARTPlus system to import data from the text file, enhancing its functionality and data management capabilities. Remember to annotate the code appropriately to maintain consistency between the feature model and codebase.

## Scenario 2: Task: *Allowing Customers to also Rent Movies*

**Objective:**

Clone the `Song` class to create a `Movie` class in the DARTPlus system. Add a `feature-to-file mapping` for `RentItem` to `Movie.java` and implement the functionality of renting movies. Details written below.

**Tasks:**

1. Clone `Song` Class:

- Locate the Song `class` in the DARTPlus codebase.

- Clone the `Song` class to create a new class called `Movie` (in the same folder, i.e., `Items`).

- Within the Movie class replace `artist` by `director`.

2. Map the rentable items `Game and Song` including the cloned file, to **RentItem.**

3. Implement Movie Renting Functionality:

- Add a menu option for renting Item (movie) in the `customerLoggedIn` options within the `Controller` Class.

- Ensure that the implementation aligns with the existing rental functionality for `Games` and `Songs`. By finding the usages of the *`RentItem`* feature in the `feature model View Window` and updating the existing feature implementation to accommodate `Movie` rentals.

Expected Outcome:

- Feature-to-file mapping **created** and **updated** to include Movie.java for the RentItem feature.
- Implementation of movie rental functionality within the Movie class, allowing users to rent movies.