Comprehensive Report for Task 4

Introduction

This report provides a detailed overview of Task 4, where the objective was to create an "Advertising Agency" with a list of "Campaign" instances. The task involved implementing the provided `LegalEntity` interface and adding methods to manage the campaigns, including functionality to save and load the campaigns to and from a file. This document describes the functionality and goals of each class involved in the implementation.

Description of Classes

Campaign Class

Functionality

The `Campaign` class represents an individual advertising campaign. Each campaign has a name and a budget associated with it. This class encapsulates the data related to a campaign and provides methods to access this data.

Goal

The primary goal of the `Campaign` class is to serve as a blueprint for creating campaign objects. These objects store the essential details of an advertising campaign, such as the campaign's name and budget. By serializing this class, we can easily save and retrieve campaign data from a file, ensuring data persistence.

Methods

- **Constructor**: Initializes the campaign with a name and budget.

```
public Campaign(String name, double budget) {
    this.name = name;
    this.budget = budget;
}
```

- **getName()**: Returns the name of the campaign

```
public String getName() {
    return name;
}
```

Implementation Details

The `Campaign` class implements the `Serializable` interface, which allows campaign objects to be written to and read from a file. This is crucial for saving the state of the campaign list managed by the advertising agency.

AdvertisingAgency Class

Functionality

The `AdvertisingAgency` class manages a list of campaigns. It implements the `LegalEntity` interface, which requires the class to provide methods for retrieving the address and VAT number of the agency. Additionally, this class provides methods to add, delete, save, and load campaigns.

Goal

The main goal of the `AdvertisingAgency` class is to manage the lifecycle of advertising campaigns within the agency. This includes adding new campaigns, deleting existing ones, and ensuring the campaigns' data can be persisted and restored through file operations.

Methods

- **Constructor**: Initializes the agency with an address and VAT number, and creates an empty list of campaigns.

```
public AdvertisingAgency(String address, String vatNumber) {
    this.address = address;
    this.vatNumber = vatNumber;
    this.campaigns = new ArrayList<>();
}
```

- **getAddress()**: Returns the address of the agency.

```
@Override
public String getAddress() {
    return address;
}
```

- **getVatNumber()**: Returns the VAT number of the agency.

```
@Override
public String getVatNumber() {
    return vatNumber;
}
```

- **addCampaign(Campaign campaign)**: Adds a campaign to the agency's list if it is not already present.

```
public void addCampaign(Campaign campaign) {
    if (!campaigns.contains(campaign)) {
        campaigns.add(campaign);
    }
}
```

- **deleteCampaign(Campaign campaign)**: Deletes a campaign from the agency's list if it exists.

```
public boolean deleteCampaign(Campaign campaign) {
    return campaigns.remove(campaign);
}
```

- **saveCampaignsToFile(String filename)**: Saves the list of campaigns to a file using serialization.

```
public void saveCampaignsToFile(String filename) throws IOException {
    try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(filename
        oos.writeObject(campaigns);
    }
}
```

- **loadCampaignsFromFile(String filename)**: Loads the list of campaigns from a file using deserialization.

```
public void loadCampaignsFromFile(String filename) throws IOException, ClassNotFoundEx
    try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(filename)))
        campaigns = (List<Campaign>) ois.readObject();
    }
}
```

- **getCampaigns()**: Returns the list of campaigns managed by the agency.

```
public List<Campaign> getCampaigns() {
    return campaigns;
}
```

The `AdvertisingAgency` class uses `ObjectOutputStream` and `ObjectInputStream` to perform file operations. By writing the list of campaigns to a file, we ensure that the state of the campaigns can be persisted across sessions. When the agency is reinitialized, it can load the campaigns from the file, restoring its previous state.

In summary, the `Campaign` class and the `AdvertisingAgency` class work together to manage advertising campaigns within an agency. The `Campaign` class provides a template for creating campaign objects, while the `AdvertisingAgency` class manages these objects and ensures their persistence through file operations. The implementation adheres to the requirements of Task 4, providing a robust solution for managing and persisting campaign data in an advertising agency context.