approach#1

```
class AddressDao {
  public Address getAddress(int addressNo,
            SessionFactory sessionFactory) {

  }
  public void saveAddress(Address address,
            SessionFactory sessionFactory) {
  }
}
```

```
class ProductDao {
  public Product getProduct(int productNo,
            SessionFactory sessionFactory) {
  }
  public int saveProduct(Product product,
            SessionFactory sessionFactory) {
  }
}
```

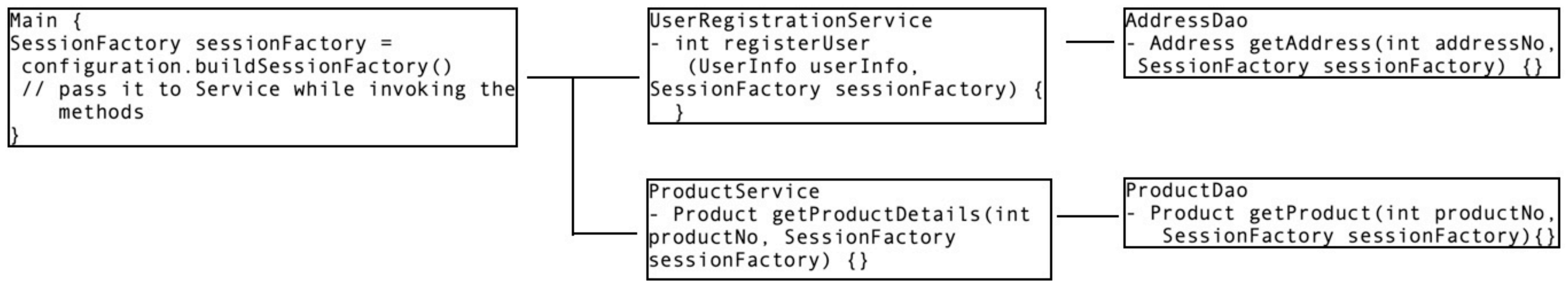
```
class Test {
  public static void main(String[] args) {
    SessionFactory sessionFactory = configuration.buildSessionFactory();

    AddressDao addressDao = new AddressDao();
    addressDao.getAddress(1, sessionFactory);


    ProductDao productDao = new ProductDao();
    productDao.getProduct(10, sessionFactory);

  }
}
```

This approach of passing SessionFactory as a parameter in calling the Dao methods is dis-encouraged, because the api methods of the Dao classes is exposed to the Persistence-tier technology (Hibernate Framework).
In-future if we change the Persistence-tier technology lets say from Hibernate Framework to Jpa api, the again we need to modify all the methods of all the dao classes.
In addition, since the Business logic classes will invoke the Daos, now its the responsibility business logic class to pass SessionFactory, so that business logic is also exposed to persistence-tier. A change in persistence-tier not only effects persistence-tier components, even business-tier components also will be effected and incurs huge maintainance.



```
Main {
SessionFactory sessionFactory =
 configuration.buildSessionFactory()
 // pass it to Service while invoking the
    methods
}
```

```
UserRegistrationService
- int registerUser
   (UserInfo userInfo,
SessionFactory sessionFactory) {
  }
```

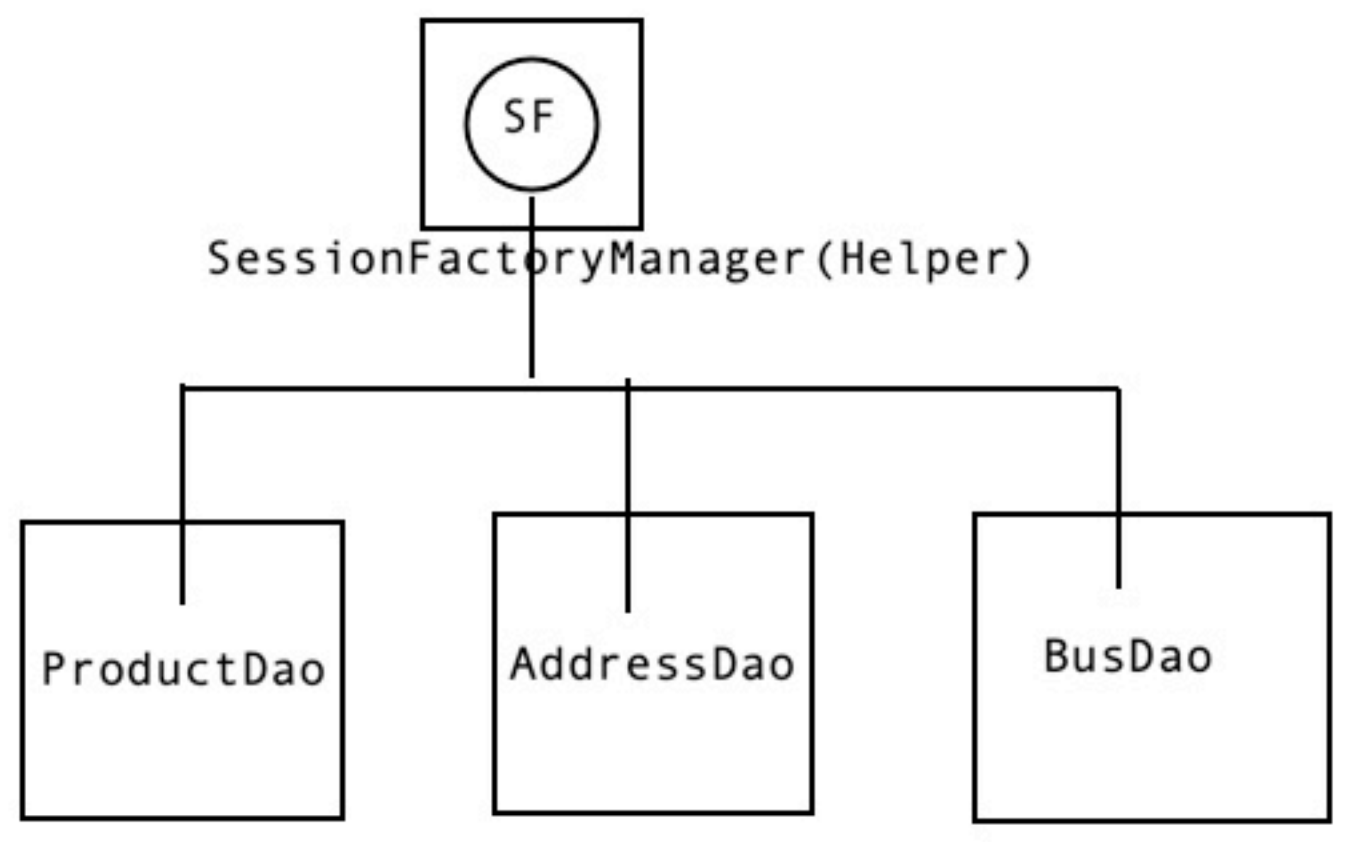
```
AddressDao
- Address getAddress(int addressNo,
 SessionFactory sessionFactory) {}
```

```
ProductService
- Product getProductDetails(int
productNo, SessionFactory
sessionFactory) {}
```

```
ProductDao
- Product getProduct(int productNo,
   SessionFactory sessionFactory){}
```

In the above if we are switching from Hibernate to Jpa, not only the Dao even the business-tier Service components also should be modified.

Then how to manage the SessionFactory?
Let us consider in our application we are using only one Database for simplicity and understand how to manage SessionFactory.


SF
SessionFactoryManager(Helper)
ProductDao
AddressDao
BusDao