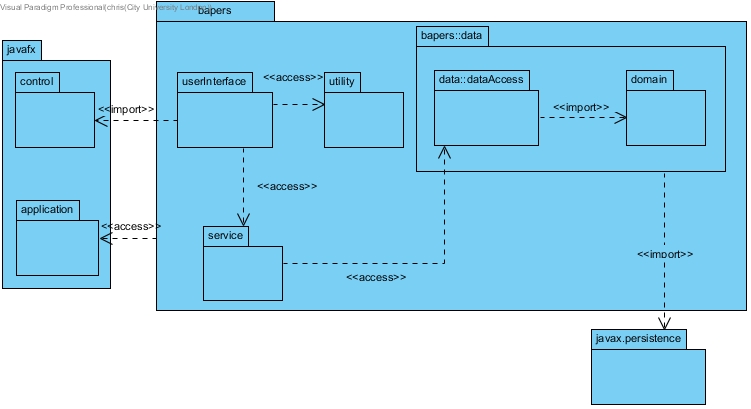
# 1.0 Design Class

## 1.1 Package Diagram



DB connectivity is modelled with the expectation of using the eclipselink library, which is an implementation of JPA, hence the import to javax.persistence. I have simplified the diagram by not showing the persistence unit, as this will be autogenerated by the chosen IDE, making it redundant to show it in the class diagram.

The persistence unit will need to include the following property: <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/bapers?zeroDateTimeBehavior=convertToNull&amp;autoReconnect=true&amp;useSSL=true"/>

The userInterface will utilise javaFX; each form will have three separate files, a bapers::fxml::form.fxml, bapers::styles::form.css and bapers::FormController.java. The fxml file details objects within the form, and their respective location, and the controller class will describe the behaviour of the forms objects. However, for the sake of brevity, I have simplified this for the class diagram by bundling all three files into a single boundary class, and making the userInteface package, where they will be stored.

## 1.2 Simplified Design Class Diagram

The following diagram presents the full class diagram; however, all classes have been stripped of their members. This is to concisely show all associations. Full details will be provided in section 1.3.

Please note that the javax.persistence package appears more than once. This was to allow for neater formatting of the diagram, all repeated instances should be treated as part of the same package. This also applies to any other repeated packages/classes.

### 1.2.1

This diagram excludes all entities from the bapers::data::domain package, as the associations for that package will be shown in the detailed diagram in section 1.3.

## 1.3 Packages

### 1.3.1 bapers::data::domain

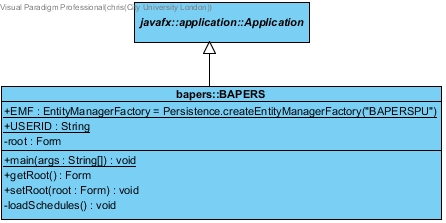
The purpose of the domain package is to store the entity classes, which are a direct mapping from the MySQL entities, to java objects.

To show one to many relationships, I have used java::util::List as a collection class. This means that each occurrence of java::util::List, should be treated as a separate object owned by the source class.

### 1.3.2 bapers::data::dataaccess

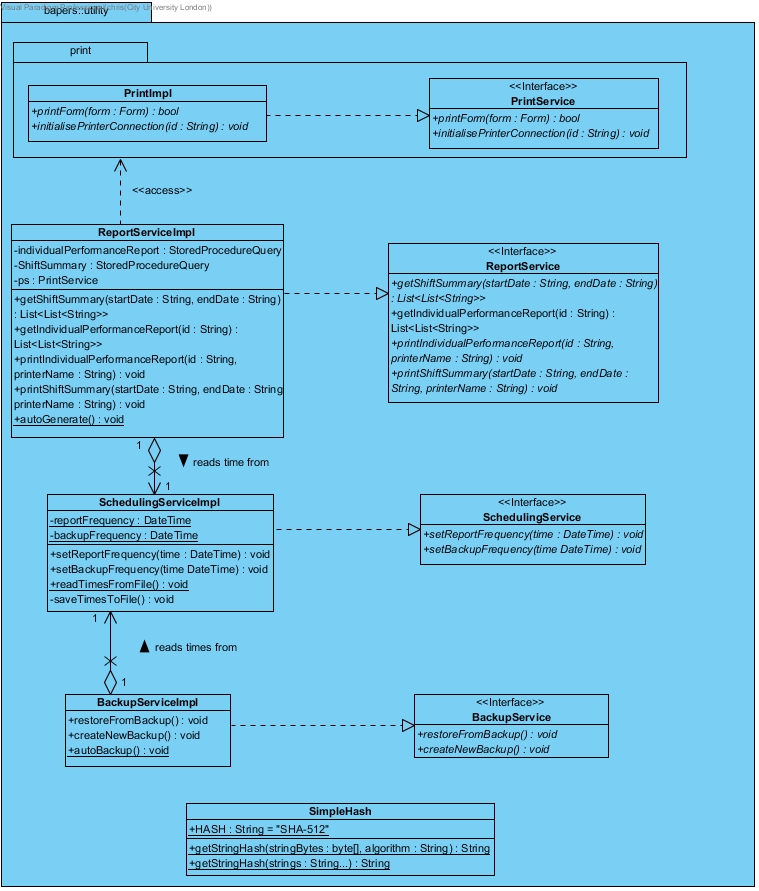
This package holds the JPA classes that interface with the database, using the entities in the domain package.

### 1.3.3 bapers



The only class inside the bapers package is BAPERS.java, which extends from javafx.application.Application. This is the main entry point for the application. It first loads the persistence unit, then runs the automatic backup and report generation services under new threads. Finally the user interface is launched, and the entry form is set to the login page.

### 1.3.4 bapers::utility



The utility package holds all the classes that do not directly talk to the classes in bapers::dataaccess. This class. Some of these classes also provide services that are intended to be run from the duration of the program.

### 1.3.5 bapers::service

The service package contains all the classes that interface with bapers::dataaccess, to direct input/output from/to the user to the database.

### 1.3.6 bapers::userInterface

This package holds all of the simplified gui forms which allow the user to interact with the system.