Systems Privacy Project 1: Implementing Data Privacy

Professor	Kevin Gallagher
Week	4
Due	2nd November, 2025 at 23:59
Version	1.1

Project Description

There are many different ways to protect privacy. Some are required by law, some are recommended by law, and some offer more protection than the law has yet to mandate. Depending on the amount of protection, a company may lose the utility that the data they collected provides. At the same time, without enough protection and there is bound to be a data breach, compliance investigations, and a public relations nightmare.

In this project your group will be implementing several different methods of protection, as well as attempting to attack some. First you will go through simple de-identification of a dataset. Then you will attack it to see how de-identification may not be enough. Then you will implement k-anonymity for the same dataset, and attempt the attack again. Finally, you will implement a differential privacy middleware that will accept queries from clients and respond using Laplace differential privacy.

This project contains four parts, a database, a server, a client, and a setup script. Each of these parts is described in detail below.

The Database

In order to complete this project, you will need to write code changes to an application that relies on a database. In order for this to work, you will need a working localhost database. The application is expecting a PostgreSQL database to exist on the localhost at port 5432, with the username "postgresql", the password "password", and the database name "pds_proj_1".

The easiest way to get this to work is to install docker (if you do not already have it installed) and use docker to run a postgresql container. To do so, see the installation instructions at:
https://docs.docker.com/engine/install/
After installing, you simply need to run the command:

```
sudo docker run --name testPostgres -e POSTGRES_PASSWORD=password \
-e POSTGRES_DB=pds_proj_1 -d -p 5432:5432 postgres
```

This command assumes you are running Linux. The command may be different if you are running Windows. After running this command, the database should be set up and you can move on to running the server.

The Server

In this project you will need to modify a Java Spring Boot application to contain a de-identified version of a database, to perform a linkage attack, to implement k-anonymity, and finally to implement differential privacy. This application has 4 data classes (MedData, WorkData, CountRequest, and CountResponse), and you will likely need to add more. Two of these have both a repository and controller to connect to the database. The project has multiple endpoints for each of these classes. You should see the respective controller classes for details about which endpoints you can use. You will need to add more.

For the de-identification, linkage attack, and k-anonymity sections of this project, you will not need the CountRequest and CountResponse classes. These are more useful for the Differential Privacy portion of the project.

Unfortunately, the modifications needed to the server are extensive. The goals of these modifications will be outlined in the Objectives section. However, each modification should be made in a separate class. For example, when you are de-identifying the MedData, it should be placed into a new class (I recommend calling it MedDataDeidentified) so that you have all versions of the data available at the end of the project. This is not something you should do in the real world, but it will help me grade the project at the end.

You can find the code in the Project1Server.zip file on CLIP. To run the code you need to navigate to the folder that contains the pom.xml file, then run the following commands:

```
mvn clean compile package
mvn spring-boot:run
```

Please ensure your PostgreSQL container is running before you run these commands. You can interact with the REST API using your web browser at the address:

`http://127.0.0.1:8000`

Please use one of the endpoints specified in the controller classes.

The Setup Script

While having an application and a database are useful, it is difficult to know if our solution works unless we have data inside said database. To aid in this, I have written a small setup script in Python that will populate the database with false but realistic data. You can find this script in the Project1Setup.zip file on CLIP. In order to run this, you will need to extract the files. You will then need to go into the resulting directory and run the following commands:

```
python3 -m venv .
source bin/activate
pip3 install sqlalchemy
pip3 install psycopg2-binary
python3 populate_tables.py
```

This will fill the database with 1000 MedData records and 1000 WorkData records.

The Client

In order to test your code for the differential privacy objectives you will need a REST client to connect to the server. This will be provided at a later date and this section will be updated. The skeleton code for this client will be found in the Project1Client.zip file.

To run this code, you will need to run the following commands:

```
mvn clean compile package
mvn spring-boot:run
```

Objectives

By the end of this project, you should have experience with:

- Implementing data de-identification.
- Performing a linkage attack.
- Implementing k-anonymity.
- Implementing Differential Privacy.
- Writing a technical report.

Data De-Identification

In this project there are two database tables: med_data and work_data. The med_data table contains an id field, a name field, an age field, an address field, an email field, a gender field, a postal_code field, and a diagnosis field. The work_data table has an id field, a fName field, an lName field, a postal_code field, an education field, a gender field, a workplace field, and a department field. To learn more about these tables you can see their respective Classes in the src/main/java/pt/unl/fct/pds/proj1server/model directory.

For the first portion of this project, you must de-anonymize the med_data database (and only the med_data database). Keep in mind that this may require creating a new model in the server code, and destroying and recreating the database.

There are various legitimate paths to achieve this goal. In my view, the easiest is to create a new class in the model folder (I would call it MedDataDeidentified and map it to the table med_data_deidentified) which contains only the non-identifiable attributes, and then run the python data insertion script again, except have it insert data into both versions of the med_data database: med_data and med_data_deidentifiede de-identified.

While we know from the theoretical lectures that de-identification isn't sufficient, this is where many companies in the modern world stop their data protection. As we will see in the next section, this could be a disaster waiting to happen.

Linkage Attack

Whenever two databases share large amounts of overlap, it is always possible to use one to attempt to re-identify another. This is called a linkage attack, and is quite simple to do in practice.

In this project we have two database tables, the now de-identified med_data table and the completely identified work_data table. Your job in this section of the project is to write the code to perform a linkage attack between these two tables, and run the attack. You should then report how you went about the attack, the number of matches you found, and how successful you believe your attack was.

k-Anonymity

In order to attempt to defend against linkage attacks, you will attempt to defend the database with *k*-anonymity. In order to turn a dataset *k*-anonymous, you first need to categorize all columns of the database as an identifier, a quasi-identifier, or sensitive data. Identifiers are to be suppressed (removed), quasi-identifiers are to be modified usually by generalization, top and bottom coding, or suppression, as discussed in class. Sensitive data should not be modified or touched.

You may use whatever algorithm you wish in order to achieve a *k*-anonymous dataset, but remember that we explicitly covered one of the simpler algorithms in our theoretical lecture. Your *k* should be configurable and you should describe how it is passed to the program so I can test it.

As previously, you will be modifying the med_data table to make it k-annonymous. You may use suppression, generalization, top and bottom coding, or other methods you may read about in order to achieve this. However, if you choose another method, please cite it. Keep in mind that these changes introduce practical problems, such as how you will represent ranges if you choose to generalize with ranges, etc. All of these decisions should be justified within the technical report you will turn in at the end of the project.

After you are finished implementing *k*-anonymmity. Try the linkage attack again. In your technical report, briefly discuss your new success rate, as well as the utility vs privacy trade-off.

Differential Privacy

As discussed in theoretical class 5, Differential Privacy is much better for a privacy and utility trade-off than k-anonymity if the data remains under the control of the organization. In this instance, the organization can use noise to make aggregate statistics available while still defending individuals whose data may make up the database.

In this portion of the project, you will be implementing a differential privacy layer in the software that will receive queries and respond with noisy results. Specifically, your project must support the following queries:

1. Count queries. The basis of this already exists in the controller classes.
2. Average queries.
3. Histographs/Grouped counts. (For example, average of the age given a specific diagnosis).

Your ε should be configurable and specified in your src/main/resources/application.properties file. For each query, you should respond with:

- The attribute(s) they queried on.
- The noised response.
- The sensitivity of the query.
- The updated privacy budget after the query was run.

As always, if there is not sufficient privacy budget to run a query, the query should not be run. Instead, you can return a HTTP Error Code 404.

Decisions about how you calculated your sensitivity and privacy budget, the decisions you made to implement differential privacy, and other details should be briefly reported in the technical report.

The Technical Report

As mentioned several times in this document, your group must turn in a technical report with your solution. This technical report should be no longer than 3 pages with 12 point Ariel fond. Brevity is a skill, so please prioritize what you want to tell me in the report. Compilation instructions or other information I need to run your code does not count towards this page limit. Citations also do not count towards this page limit. The report should be submitted as a PDF, and should only contain your student numbers, not names. **In order to ensure blind grading, I do not want to see your names on the report.**

Reports can be in English or in Portuguese. The choice of language will not impact your grade.

Submission

To submit, please send all of your code in a zip, 7z, or tar file to my email address, k.gallagher@fct.unl.pt, with the following subject:
"[PdS 25/26] Project 1 Submission - XXXXX and YYYYY"
where XXXXX and YYYYY are your student numbers. **Again, to ensure blind grading I'd like there to not be any student names in the code or in the email subject. I will be using a script to extract all files and information from the emails while keeping myself blind to the names of the students who sent them.**

Grading Rubric

To be added soon.

Frequently Asked Questions

Here is a list of questions I expect will be frequent about the lab, along with their respective answers.

1. **Is this exercise graded?**

   Yes. It is a graded project.

2. **Do we do the labs alone or in pairs?**

   I recommend doing the project in pairs.

3. **Where do I turn in the project?**

   See the section immediately above this one.

4. **What does this functionality of the web application do?**

   I am happy to clarify these doubts during a class or during office hours, or by email. However, this document is already long, and I would basically only be regurgitating information that can be found online.

5. **I really like/want to learn language x. Can I do my lab and project in that language?**

   Conditionally on approval.

6. **I found this bug/problem/issue with the report/code.**

   Skill issue on my end. Please let me know and I'll do my best to update the document and/or code to fix the issue as fast as possible. I'll also update the version number on top and add your name to the acknowledgements section below.

Acknowledgements

Credit for the base of the LaTeX template goes to Gilles Callebaut. Credit to students Diogo Nunes, Niklas Sander, and Francisco Coutinho for discovering unclear language and technical issues within the project.