DATABASE SYSTEMS

-MOTIVATION-

Hibernate

- Java framework, open-source, lightweight and ORM (Object-Relational Mapping) tool for the Java language which simplifies the buildout of Java application to interact with the database
- well-known JPA implementation (it extends it with extra functionalities)
- **PROS**
  - recognizes the mappings we create between objects and tables and guarantees that data is stored and retrieved from the database in accordance with the mappings
  - provide Dialect classes, so we don't need to write SQL queries in Hibernate, instead we use the methods provided by that API
  - supports annotations
  - supports caching mechanism, by this the number of round trips between an application and the database will be reduced and performance will be increased automatically
  - has its own object-oriented query language: HQL (Hibernate Query Language) which is database independent
  - allows you to develop object-oriented persistent classes with all known functionality like polymorphism, inheritance, and association
  - it is used in mapping Java data types with SQL data types and database tables
- **CONS**
  - is a little slower than pure JDBC
- Why Hibernate and not JPA?
  - Hibernate is an implementation of JPA guidelines
  - It helps in mapping Java data types to SQL data types
  - It is the contributor of JPA
  - JPA is only a specification, not an implementation

H2

- well-known in-memory database
- Spring Boot provides an excellent interaction with H2
- Java-based relational database management system
- it may be integrated in Java programs or used as a client-server application
- **PROS**
  - faster
  - internal use only (not accessible from outside)
  - more secure
  - supports standard SQL and JDBC API
  - it can use PostgreSQL ODBC driver too
  - Spring Boot provides an excellent interaction with H2
- **CONS**
  - has very bad concurrent behavior
  - not transactional

    - poor query optimizer
- In conclusion, we chose to use **H2 in the development of our application** because it is an in-memory database, which means that we are independent of the choice of an actual database, so we can test our app on a database without having an actual database.

PostgreSQL

- open-source database that has a strong reputation for its reliability, flexibility, and support of open technical standards
- unlike other RDMBS (Relational Database Management Systems), PostgreSQL supports both non-relational and relational data types, which makes it one of the most compliant, stable, and mature relational databases available today
- **PROS**
    - allows administrators to build fault-tolerant environment by protecting data integrity
    - mature Server-Side Programming Functionality
    - full support for client-server network architecture
    - object-oriented
- **CONS**
    - changes made for speed improvement requires more work
- In conclusion, we chose to use **PostgreSQL in production of our application** because it is robust, allows very complex queries, can handle complex datasets and has no data corruption. We also noticed that of all the RDBMS, PostgreSQL is one of the most flexible and is used for data persistence.