Data Persistence Tech Report

Kieran A J Roper

Keywords: data persistence, full stack development, storage, databases, JDL, JHipster

1 WHAT IS PERSISTENCE?

Persistence is ”continued or prolonged existence or occurrence; duration; continuance.”[2] In terms of data, persistence is achieved by ensuring data is stored in non-volatile memory. In addition, data should be backed-up - ideally in multiple ways.

2 IMPLEMENTATION

2.1 Which data should persist?

Any data that is useful to store between use sessions should utilise data persistence.

- User settings
  - Across user sessions on the same device and across different devices.
  - Examples: food preferences (vegetarian), blacklisting other users, befriending other users, notification settings (Important with GDPR).
- User generated content
  - e.g. images, text input, messages.
  - Other versions of the same content, e.g. will we store 100x100 versions of 1000x1000 images for thumbnails / lower DPI users, or will we process ad hoc?

2.2 Which data will not persist?

Any data that does not end up getting committed will not need to persist. For example, editing settings, but not clicking ’Save Changes’ will not persist in the database. There are similar situations when similar data may persist to better facilitate the user experience; e.g. when filling out a longer job application form - data will persist between sessions to give the candidate more time to complete it. With our project, we do not require such data persistence due to the, relatively, short time required to create a post, send messages, etc.

2.3 Other considerations

When data persists, we must ensure it is handled securely and within GDPR. There must be options to restrict access to, or delete persistant data. There must be appropriate user-access controls. For example, the question is will we allow anyone (That means anyone on the internet, including bots.) to view all posts, or only their friends, or only authenticated logged-in users. We must also consider how we will ensure persistent data is locked away from both bad actors and accidental viewers.

Data persistence for sake of user experience is something that should be considered during improvement phases. This could involve adding data persistence with a draft-style system for creating posts.

3 UTILISING DATA PERSISTENCE

Our current tech stack uses JHipster. This uses PostgreSQL in the back-end. JHipster has a useful tool called JDL-Studio. This helps in the development of JHipster Domain Language (JDL)[5] files, and their UML equivalents. This is how we will design our relations between objects and entities within the back-end of our project. This will allow us to more easily connect different data together, and implement them more easily into our deployment.

4 BACKING-UP DATA

Whilst it may not be necessary to back-up data in our, relatively, small-scale implementation; it never hurts to have a fall-back. All of our code is stored locally on our machines, plus on the University of Birmingham GitLab repository. The PostgreSQL knowledgebase includes information on backing up databases[6].

5 HANDS-ON

There are numerous guides accessible online discussing file storage and other data persistence using JHipster[7][8].

REFERENCES

[1]This LATEX document template was created by Harry Cooke.
[2]Oxford English Dictionary - Persistence `https://www.oed.com/view/Entry/141466`
[3]MongoDB - An Introduction to Data Persistence `https://www.mongodb.com/databases/data-persistence`
[4]Spring Developer - A Spring Data’s Guide to Persistence `https://youtu.be/gmDMkFA03UM`
[5]JHipster Domain Language (JDL) `https://www.jhipster.tech/jdl/intro`
[6]PostgreSQL: Chapter 26. Backup and Restore - Part III. Server Administration `https://www.postgresql.org/docs/current/backup.html`
[7]ippon - Create a simple file storage through database with JHipster *https://blog.ippon.tech/create-a-simple-file-storage-through-database-with-jhipster/*
[8]Get Started with JHipster 7 `https://youtu.be/6lf64CctDAQ`