Deliverable D05: Architecture of a WIS learning report
-
Group E8.02

Github Repository: https://github.com/acme-recipes/Acme-Recipes

Members:

- Gregorio Ortega Soldado (greortsol@alum.us.es) 30271286C
- Alejandro Manuel Gestoso Torres (alegestor@alum.us.es) 30260633Q
- Jaime Stockwell Mendoza (jaistomen@alum.us.es) 30696480J
- Pablo Aurelio Sánchez Valenzuela (pabsanval1@alum.us.es) 30246142S
- Manuel Cabra Morón (mancabmor1@alum.us.es) 47561328L
- Fernando Claros Barrero (ferclabar@alum.us.es) 77868841H

Fecha: 30/08/2022

Table of Contents

Executive Summary

The following document consists of an explanation of what we have learnt along the Design and testing II subject about the architecture of a Web Information System (WIS).

Revision Table

Revision	Description	Date
v1.0	Sprint retrospective discussion	30/08/2022
v1.1	Revision of our base of knowledge	30/08/2022
v1.2	Final version	31/08/2022

Introduction

We have already been working on designing and implementing Web Information Systems through all these years studying our degree, but in this subject we have gone one step further. However, we have always approached it from a different perspective, so we have widened our range of knowledge about how to design these systems and how to ensure their quality by means of E2E testing.

Contents

Along the whole course, we have learned about the components that conform a Web Information System, the interactions between them and how to develop those components in order to make them work as a unique system.

Our WIS is structured in four layers: browser layer, application server layer, application layer and database server layer:

- We consider the browser a layer itself. It is the software that the final users will use to send requests (using the HTTP protocol) and interact with the system (e.g., by clicking on a button), and also to catch and render the obtained responses (in HTML+CSS+JS).

- The application server, which receives the requests sent from the browser, processes them and returns the associated responses (in HTML+CSS+JS). The internal structure is composed of the HTTP server, servlet server, and renderer.

- The application itself, which implements the functionalities that are requested by the customer. In our case, it is composed of a view component, a controller component, services, entities and repositories. This is not a mandatory decision for every WIS, but this is the architectural pattern that our application follows, since Acme Framework uses Spring in order to work, which follows the classical Model-View-Controller (MVC) architectural pattern.

- The database server. This is the part responsible for managing data and guaranteeing its persistence. It handles the transactions, which are determinant in our context, as well as the execution ,or staging, for a future execution of the queries. It has introduced to us the idea of the transaction journal: a set of SQL operations to be performed on the database in order to respond to a particular transaction.

With this idea in mind, we can now think about the interactions between the previously mentioned components of our WIS:

1. First of all, the user makes user interaction. The Application server will receive the request, and passes it onto the servlet server.

2. After that, the servlet server is in charge of determining which of the controllers should attend the request. Again, we can see this pattern in many other web information systems, not only this one. This controller will start a transaction and then do the corresponding procedure back to the user to serve the request.

3. Once the transaction has started, a transaction will be opened in the database server component, creating a journal in the staging area —or more than one independent journals, in case there are more requests taking place at the same time—. The queries that don’t involve modifying data can be executed almost immediately, while

those that can potentially modify our system data are always staged. Concurrent requests will get their data from either the staging or the data area depending on their journals.

4. Lastly, the workflow will be requested to commit if there are no errors or exceptions during its execution. Otherwise, it will be requested to roll back.

To conclude, we have acquired a lot of quality knowledge for our future, we now understand the difference between a regular programmer and an engineer. The engineer extracts a common factor when developing software, so that he/she does not repeat the same work twice. Indeed, knowing how the Acme Framework works and having a general idea about what a WIS internal architecture looks like will be a determining inspiration for us to decide to develop our own architecture patterns and frameworks to work with, either if they are a more complex solution or only a single piece of code.

Conclusions

After having taken this subject we believe that we have acquired some knowledge about working with Web Information Systems that we did not know before. We had already worked in some other subjects with this kind of WIS, but we think we have gone one step further doing the project for this subject. We hope to have accomplished the expected goals regarding Web Information Systems, however we look forward to learning more about this topic. Therefore, we hope to become real experts in the field and learn everything we can about it, not only the basics, in our future.