# Architecture of a WIS Learning

Workgroup number: E8.01
Repository: https://github.com/javiervaz01/Acme-Toolkits
Date: 2022/05/26


| Name | Corporate e-mail |
|---|---|
| Carrasco Núñez, Alejandro | alecarnun@alum.us.es |
| Durán Terrero, Andrés | anddurter@alum.us.es |
| López Benítez, Pablo Delfín | pablopben@alum.us.es |
| Núñez Moreno, Pablo | pabnunmor@alum.us.es |
| Robledo Campa, Pablo José | pabrobcam@alum.us.es |
| Vázquez Monge, Francisco Javier | fravazmon@alum.us.es |

# 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 number | Date | Description |
|---|---|---|
| v1 | 2022/06/01 | Initial version |

# 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 a Web Information System is made of, the interactions inside them and how to develop those components.

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

- The browser can be considered a layer itself. It is the software that the end users will use to send requests (using the HTTP protocol) to a server by means of user interactions (e.g., by clicking on a button), and to render the obtained responses (in HTML+CSS+JS).

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

- The application itself, which implements the features that are requested by the customer. In our case, it is composed of views, controllers, services, entities and repositories. This is not an imposed decision for every WIS, but an architectural pattern our application follows, since Acme Framework uses Spring under the hood —which follows the Model-View-Controller (MVC) architectural pattern—.

- The database server. This is the part responsible for managing data. 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 list of SQL operations to be performed on the database in the context of a particular transaction.

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

- First of all, the user makes user interaction (e.g.., clicks on a button or enters a website). The Application server (more precisely, the HTTP server) will receive the request, and passes it onto the servlet server

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

- Once the transaction has started, a transaction will be opened in the database server component, therefore creating a journal in the staging area —or several independent journals, in case there are more requests taking place concurrently—. The queries that don't involve modifying data can be executed immediately, while those which can potentially modify data are always staged. Concurrent requests will get their data from either the staging or the data area depending on their journals.

- Finally, 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.

Finally, and as a valuable lesson for our near future, we have learned that the difference between a regular programmer and an engineer is that the latter 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 therefore, having a general idea about the internal architecture a WIS may have— will be a determining inspiration for us to decide to develop our own frameworks, either if they are a more complete solution or only a small 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 have before. We had already worked in some other subjects with WIS, but we think we have gone a step further thanks to this subject. We hope to have achieved the expected goals regarding Web Information Systems, however we seek learning more. Therefore, we hope to become real experts in the field and learn everything about it, not only the basics, in our future.

# Bibliography

Intentionally blank.