**Architecture of a WIS**

| **Group:** C1.024 | |
| --- | --- |
| **Repository:** https://github.com/davvarmunUS/Acme-ANS-D01 | |
| Student #1  **ID:**  21015312M  **UVUS:**  davvarmun  **Name:**  Vargas Muñiz, David  **Roles:**  Project Manager, Developer | Student #2  **ID:**  77862000P  **UVUS:**  antlopcub  **Name:**  López Cubiles, Antonio José  **Roles:**  Developer |
| Student #3  **ID:**  17478227K  **UVUS:**  enrpermil  **Name:**  Pérez Milla, Enrique  **Roles:**  Developer | Student #4  **ID:**  47561989J  **UVUS:**  jesmarver  **Name:**  Martín Vergara, Jesús  **Roles:**  developer, analyst, operator |
| Student #5  **ID:**  Masked ID number  **UVUS:**  uvus  **Name:**  Surnames, Name  **Roles:**  role1, role2, role3 |  |
| **Date:** place month day, year | |

**1. Resumen Ejecutivo**

En términos generales, se contaba con conocimientos básicos de desarrollo web (HTML, CSS, JavaScript) y la idea de una arquitectura Cliente-Servidor, pero con un enfoque muy elemental y sin profundizar en modelos avanzados ni en la organización a gran escala de los componentes de un Web Information System. Asimismo, se hacía un uso limitado de frameworks, tanto en el front-end (por ejemplo, Bootstrap, jQuery) como en el back-end (Express.js, Django), sin entender plenamente cómo se integraban en una arquitectura completa. En consecuencia, la visión global de cómo estructurar y relacionar los distintos módulos de un proyecto (interfaz, lógica de negocio, base de datos) era difusa y se basaba más en la repetición de tutoriales que en la aplicación de principios de diseño y organización.

**2. Tabla de Versiones**

| Versión | Fecha | Descripción |
| --- | --- | --- |
| 1 | 18/02/2025 | Creación del informe |

**3. Introducción**

Antes de cursar la asignatura, nuestro grupo tenía una idea muy general y algo desestructurada sobre la Arquitectura WIS (Web Information Systems). Principalmente, sabíamos que un sistema de información en la web no se reducía únicamente a desarrollar un sitio con HTML y CSS, sino que implicaba la interacción con bases de datos, el manejo de la lógica de negocio y la necesidad de mantener una comunicación fluida entre el cliente y el servidor.

**4. Contenidos**

**1. Conocimientos de desarrollo web básicos**

* Nos defendíamos con HTML**,** CSS y JavaScript para crear interfaces sencillas.
* Estábamos familiarizados con la idea de un “front-end” que muestra la información al usuario y un “back-end” que gestiona los datos y la lógica.
* Reconocíamos la importancia de contar con un sistema de gestión de bases de datos (generalmente SQL), pero no profundizamos en la optimización o diseño relacional avanzado.

**2. Concepto general de ‘Arquitectura Cliente-Servidor’**

* Entendíamos que hay una parte “servidor”, donde residen los datos y la lógica principal, y una parte “cliente”, que es el navegador o la aplicación que interactúa con el usuario.
* Sin embargo, no conocíamos modelos más sofisticados (como multicapa o microservicios) ni sabíamos cómo se organizaban los componentes dentro de un Web Information System.

**3. Manejo de frameworks e integración**

* Para proyectos sencillos, habíamos usado frameworks de front-end (como Bootstrap) o alguna librería de JavaScript (por ejemplo, jQuery) pero sin profundizar en su arquitectura interna.
* A nivel de back-end, conocíamos algún framework básico (como Express.js en Node o Django en Python), pero realmente no entendíamos a fondo cómo encajaba esto en la arquitectura global de un sistema WIS.

**4. Falta de visión global**

* Sabíamos que era importante definir una estructura clara para un proyecto web, pero no teníamos claro cómo se relacionaban formalmente los distintos módulos (interfaz, lógica de negocio, base de datos) ni cómo organizar de manera óptima la comunicación entre ellos.
* La mayor parte de la estructura la hacíamos por inercia o imitando tutoriales que encontrábamos, sin comprender en detalle los principios teóricos de diseño y organización de una arquitectura WIS.

**5. Conclusión**

* Antes de la asignatura, nuestra perspectiva se centraba en "hacer que funcione" en lugar de "diseñar la arquitectura".
* Entendíamos de forma superficial algunos conceptos clave (separación cliente-servidor, uso de bases de datos, importancia de la lógica de negocio), pero no teníamos una visión formal de cómo se concibe, diseña y mantiene un Web Information System a gran escala.

En resumen, nuestro grupo partía de nociones bastante prácticas y “autodidactas”, sin un marco teórico sólido que nos permitiera hablar con propiedad de la Arquitectura WIS. Esta asignatura nos brinda la oportunidad de formalizar esos conocimientos y entender la relevancia de diseñar y estructurar los sistemas web de manera coherente y escalable.

**6. Bibliografía**

Intencionalmente en blaco.