UNIVERSIDAD NACIONAL DE SAN ANTONIO ABAD DEL CUSCO

FACULTAD DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA,

INFORMÁTICA Y MECÁNICA

ESCUELA PROFESIONAL DE INGENIERÍA INFORMÁTICA Y DE SISTEMAS

“Proyecto de Carga Académica Docente ”

CURSO: INGENIERÍA DE SOFTWARE I

DOCENTE: William Zamalloa

PRESENTADO POR:

- CALLAPIÑA RODRIGUEZ JOSUE CRISTIAN 164235
- NAJOR JOSUE VALDEIGLESIAS DUEÑAS 151830
- JOSE LUIS CRUZ CARRION 170431
- CASILLA TTITO EVANDIR SAUL 164238
- TRUJILLO TORBISCO LUIS ANDERSON 083221
- LOAIZA MONROY BRUNO WALDIR 153097
- PUMA MAMANI NILSON MAURIÑO 151822
- HUAMAN QUISPE JEMY SANDRO 194519
- ASENCIO ARQQUE JHOEL FELIZ 163806

CUSCO – PERÚ

2023

1) Presentación de proyecto.

El presente proyecto de Gestión de Carga Académica Docente de la UNSAAC. En el entorno universitario, la eficiente asignación de responsabilidades docentes es esencial para garantizar una educación de calidad. Este proyecto tiene como objetivo principal mejorar la planificación y distribución de la carga académica, maximizando así el rendimiento y el bienestar de nuestros docentes.

2) Organigrama del proyecto.

Lider de proyecto
- Najor Josue Valdeiglesias Dueñas

Desarrolladores
- Mamani Zela Nicholas Edward
- Callapiña Rodriguez Josue Cristian
- Dexa Kacha Reno Max
- Jose Luis Cruz Carrion
- Casilla Ttitto Eandir Saul
- Trujillo Torbisco Luis Anderson
- Loayza Monroy Bruno Waldir
- Puma MamaniI Nilson Mauriño
- Huaman Quispe Jemy Sandro
- Asencio Arque Jhoel Feliz

3) Plan del proyecto. (Cronograma a todo nivel)

Semanas 1-2: Definición y Planificación Inicial

Día 1-2: Reunión de inicio del proyecto, definir objetivos y alcance.

Día 3-4: Identificar requisitos del programa de carga académica.

Día 5-7: Desarrollar una estructura inicial del proyecto y asignar roles.

Semana 2: Revisar y ajustar el plan según comentarios del equipo.

Semanas 3-4: Diseño y Arquitectura

Días 1-5: Diseñar la arquitectura del programa.

Días 6-10: Crear prototipos de las interfaces de usuario.

Días 11-12: Revisión y ajuste del diseño.

Días 13-14: Preparar documentación de diseño.

Semanas 5-8: Desarrollo

Días 1-4: Configurar el entorno de desarrollo.

Días 5-9: Desarrollar módulos básicos del sistema.

Días 10-12: Integrar módulos y realizar pruebas preliminares.

Días 13-16: Refinar y ajustar funcionalidades.

Semanas 9-12: Pruebas y Mejoras

Días 1-5: Realizar pruebas exhaustivas del sistema.

Días 6-9: Corregir errores y optimizar el rendimiento.

Días 10-12: Preparar la documentación del usuario.

Días 13-14: Entrenamiento del personal y prueba de aceptación del usuario.

Días 15-16: Preparar para la implementación.

Semanas 13-16: Implementación y Entrega

Días 1-2: Implementar el programa de carga académica en el entorno de producción.

Días 3-7: Monitoreo y ajuste post-implementación.

Días 8-14: Preparar y entregar la documentación final del proyecto.

Días 15-16: Celebrar la finalización del proyecto y realizar una revisión post-proyecto.

4) Glosario de términos propios de la implementación

- Laravel: Es un popular framework de desarrollo web en PHP que sigue el patrón de diseño MVC (Modelo-Vista-Controlador). Ofrece un conjunto de herramientas y funciones que simplifican y aceleran el proceso de desarrollo web.
- ORM: En lugar de escribir consultas SQL directamente para interactuar con la base de datos, un ORM proporciona métodos y herramientas para manipular los datos utilizando objetos y clases en el lenguaje de programación que estés utilizando.
- PHP: es un lenguaje de programación ampliamente utilizado en el desarrollo web, especialmente adecuado para crear aplicaciones dinámicas y sitios web interactivos, con una amplia gama de herramientas y una comunidad activa que lo respalda.

- CRUD: Es un acrónimo que se refiere a las operaciones básicas de manipulación de datos en un sistema de gestión de bases de datos o en la persistencia de datos en general
- MySQL: Es un sistema de gestión de bases de datos relacional de código abierto. Se utiliza para almacenar, organizar y gestionar datos. Ofrece capacidades para manipular datos estructurados, consultas complejas y es ampliamente utilizado en aplicaciones web.
- HTTPS: Es el acrónimo de Hypertext Transfer Protocol Secure. Es un protocolo de comunicación que se utiliza para transferir datos de forma segura a través de Internet. HTTPS utiliza una capa adicional de seguridad mediante el cifrado de la información transmitida entre el cliente y el servidor, lo que ayuda a proteger la integridad y privacidad de los datos.
- Servidor web: Es un software que se ejecuta en un servidor y maneja las solicitudes de los clientes (navegadores web) enviando los recursos solicitados, como páginas web, imágenes, archivos, etc. Algunos ejemplos populares son Apache, Nginx, Microsoft IIS, entre otros.
- UML (Unified Modeling Language): Es un lenguaje estándar utilizado para modelar sistemas de software. Proporciona notaciones y diagramas que permiten visualizar, especificar, construir y documentar los artefactos de un sistema de software. Los diagramas UML incluyen diagramas de clases, de casos de uso, de secuencia, entre otros, que ayudan a comprender la estructura y el comportamiento de un sistema.
- Casos de Uso: En UML, los casos de uso representan la funcionalidad o los servicios que un sistema proporciona a sus actores (usuarios o sistemas externos). Los casos de uso describen las interacciones entre el sistema y los actores, mostrando cómo el sistema responde a diferentes acciones. Son utilizados para capturar los requisitos del sistema desde la perspectiva del usuario, identificando las diferentes formas en que los usuarios interactúan con el sistema y qué funcionalidades ofrece.

5) Documento de Requerimientos

a. Requerimientos Funcionales

1. Gestión de Usuarios:

- **RF1:** El sistema debe permitir el registro de usuarios con roles específicos: docentes, revisores, presidente de la comisión, y jefe de departamento.
- **RF2:** Los usuarios deben poder iniciar sesión con sus credenciales.

2. Gestión de Portafolios:

- **RF3:** Los docentes deben poder cargar sus portafolios, seleccionando el formato correspondiente (teórico o práctico).
- **RF4:** El sistema debe permitir la revisión de documentos dentro del portafolio por parte de la comisión revisora.
- **RF5:** Se debe notificar a los docentes sobre observaciones o faltas encontradas durante la revisión.

3. Planificación y Asignación:

- **RF6:** El sistema debe permitir la planificación de las fechas de revisión.
- **RF7:** Asignar docentes a revisores y especificar roles dentro de la comisión.

4. Generación de Informes:

- **RF8:** El sistema debe generar informes al concluir cada revisión, indicando el cumplimiento o no según los resultados de la revisión.
- **RF9:** Los informes deben ser enviados automáticamente al jefe de departamento.

5. Seguimiento de Carga Académica:

- **RF10:** El sistema debe llevar un registro de la carga académica de cada docente en cada semestre.
- **RF11:** Permitir la actualización de la carga académica para reflejar cambios semestrales.

b. Requerimientos No Funcionales

1. Usabilidad:

- **RNF1:** La interfaz debe ser intuitiva y fácil de usar para los distintos roles de usuarios.
- **RNF2:** La respuesta del sistema debe ser rápida, manteniendo tiempos de carga mínimos.

2. Seguridad:

- **RNF3:** La información del portafolio y los informes debe ser confidencial y accesible solo para usuarios autorizados.
- **RNF4:** Se debe implementar un sistema robusto de autenticación y autorización.

3. Escalabilidad:

- **RNF5:** El sistema debe ser escalable para manejar un aumento en el número de usuarios y documentos.

4. Compatibilidad:

- **RNF6:** El sistema debe ser compatible con los navegadores web más comunes.

c. Matriz de Rastreo de los Requerimientos (Rastreabilidad)

ID Requerimiento	Descripción	Requerimientos Asociados	Fuente	Diseño	Prueba
RF1	Registro de Usuarios	-	Cliente	Módulo de autenticación y autorización	Verificar que el sistema permite el registro de usuarios con roles específicos. Comprobar que los roles asignados son correctos.
RF2	Inicio de Sesión	-	Cliente	Módulo de autenticación y autorización	Verificar que los usuarios pueden iniciar sesión con sus credenciales correctamente.
RF3	Carga de Portafolios	-	Cliente	Módulo de gestión de documentos	Verificar que los docentes pueden cargar sus portafolios y seleccionar el formato correspondiente.
RF4	Revisión de Documentos	RF3	Cliente	Módulo de revisión de documentos	Verificar que la comisión revisora puede revisar los documentos dentro del portafolio.
RF5	Notificación de Observaciones	RF4	Cliente	Módulo de notificaciones	Verificar que las notificaciones sobre observaciones o faltas se envían correctamente a los docentes.
RF6	Planificación de Fechas	-	Cliente	Módulo de planificación	Verificar que el sistema permite la planificación de las fechas de revisión.

RF7	Asignación de Docentes	RF6	Cliente	Módulo de asignación	Verificar que el sistema permite asignar docentes a revisores y especificar roles dentro de la comisión.
RF8	Generación de Informes	RF4, RF7	Cliente	Módulo de generación de informes	Verificar que el sistema genera informes al concluir cada revisión, indicando el cumplimiento o no según los resultados de la revisión.
RF9	Envío Automático de Informes	RF8	Cliente	Módulo de notificaciones	Verificar que los informes se envían automáticamente al jefe de departamento
RF10	Registro de Carga Académica	-	Cliente	Módulo de seguimiento de carga académica	Verificar que el sistema lleva un registro de la carga académica de cada docente en cada semestre.
RF11	Actualización de Carga Académica	RF10	Cliente	Módulo de seguimiento de carga académica	Verificar que el sistema permite la actualización de la carga académica para reflejar cambios semestrales.
RNF1	Interfaz Intuitiva	-	-	Diseño de interfaz de usuario	-
RNF2	Respuesta Rápida del Sistema	-	-	-	-
RNF3	Confidencialidad de Información	-		-	-
RNF4	Sistema de Autenticación y Autorización	-	-	-	-

RNF5	Escalabilidad	-	-	Arquitectura escalable	-
RNF6	Compatibilidad con Navegadores	-	-		-

Esta matriz de rastreo ayuda a establecer la relación entre los requerimientos funcionales y no funcionales, permitiendo un seguimiento claro del cumplimiento de cada uno de ellos a lo largo del desarrollo del sistema.

6) Documentos de Diseño

a. Arquitectura funcional del sistema

i. Interfaz de usuario (capa de presentación)

- ➔ Desarrollo de una interfaz de usuario amigable para que los usuarios (administradores, docentes) gestionen la carga académica.
- ➔ Formularios para agregar, editar y eliminar cursos, asignar docentes, y gestionar detalles como horarios, aulas, etc.

ii. Lógica de Negocio (Capa de Aplicación)

- ➔ Desarrollo de la lógica para calcular la carga horaria total de los docentes.
- ➔ Implementación de reglas de asignación de cursos a docentes, considerando preferencias y restricciones.
- ➔ Validaciones para asegurar que los cursos no excedan el límite de aforo y que las fechas y horarios sean coherentes.

ii. Persistencia de Datos (Capa de Datos)

- ➔ Implementación de modelos en Laravel para representar las entidades como cursos, docentes, aulas, etc.
- ➔ Configuración y utilización de Eloquent ORM para interactuar con la base de datos.
- ➔ Desarrollo de migraciones para crear y actualizar la estructura de la base de datos.

b. Arquitectura técnica del sistema

i. Topología de software

- ➔ Laravel Framework

- Utilización de Laravel como el marco de desarrollo principal para PHP.
- Uso de controladores para gestionar la lógica de negocio.
- Utilización de middleware para manejar la autenticación y autorización.
- Uso de Eventos y Listeners para gestionar eventos del sistema.
- Vistas y plantillas Blade para la presentación de datos.

➔ Base de Datos (MySQL)
- Configuración de una base de datos para almacenar la información sobre cursos, docentes, aulas, etc.
- Implementación de un sistema de respaldo y recuperación.
- Uso de transacciones para garantizar la integridad de los datos.

➔ Eloquent ORM
- Utilización de Eloquent, el ORM de Laravel, para interactuar con la base de datos y realizar operaciones CRUD.

➔ Servidor de Aplicaciones
- Configuración para gestionar la carga y mejorar la seguridad.
- Uso de HTTPS para la comunicación segura entre el cliente y el servidor.

ii. Topología del hardware

➔ Servidor Web
- Configuración de un entorno de equilibrio de carga para manejar grandes cantidades de tráfico.
- Implementación de un plan de monitorización y escalabilidad.

➔ Base de Datos
- Implementación en un servidor de base de datos dedicado con configuración para el rendimiento óptimo.
- Estrategias de respaldo y recuperación para garantizar la disponibilidad.

➔ Entornos de Desarrollo, Pruebas y Producción
- Desarrollo en entornos locales.
- Despliegue utilizando servicios como Laravel Forge, Envoyer o soluciones de orquestación de contenedores.

➔ Red
- Configuración de una red segura y eficiente entre el servidor web y la base de datos.

c. Documentos basados en UML

i. Casos de uso

ii. Modelo de Objetos (si fuese el caso)

iii. Diagramas secuenciales

d. Descripción de la metodología ágil utilizada de manera específica (sprints, historias, etc)

Roles en SCRUM:

Product Owner:
Identifica y prioriza las necesidades y características clave del proyecto en colaboración con los usuarios finales y los administradores académicos.
Define las historias de usuario y los criterios de aceptación.
Participa activamente en las reuniones de planificación de sprint y proporciona la visión general del producto.

Scrum Master:
Facilita las ceremonias de SCRUM y garantiza la adhesión a los principios y prácticas ágiles.
Elimina obstáculos y fomenta la comunicación efectiva entre los miembros del equipo.
Apoya la mejora continua del equipo y promueve un entorno de trabajo colaborativo.
Equipo de Desarrollo:

Profesionales encargados de diseñar, desarrollar e implementar las funcionalidades del proyecto.
Se autoorganizan para completar las historias de usuario durante los sprints.
Participan activamente en la planificación de sprint y las reuniones diarias.
Ceremonias de SCRUM:

Planificación del Sprint:

El equipo, junto con el Product Owner, planifica las historias de usuario a abordar durante el sprint.
Se establece la meta del sprint y se determinan las tareas necesarias para alcanzarla.
La duración típica del sprint puede ser de dos semanas.
Reunión Diaria o Daily Standup:

Sesiones cortas diarias donde el equipo comparte actualizaciones sobre el progreso,

los obstáculos encontrados y los planes para el próximo día.

Fomenta la colaboración y la transparencia, permitiendo a cada miembro del equipo conocer el estado del proyecto.
Revisión de Sprint:

Al final del sprint, el equipo demuestra las funcionalidades completadas al Product Owner y otras partes interesadas.
Se recopila la retroalimentación para ajustar y mejorar las futuras iteraciones del proyecto.
Retrospectiva del Sprint:

El equipo reflexiona sobre el sprint, identifica áreas de mejora y define acciones concretas para implementar cambios positivos en el próximo sprint.
Incluye una revisión de procesos, herramientas y relaciones interpersonales.
Artefactos de SCRUM:

Backlog del Producto:

Lista priorizada de todas las características, historias de usuario y mejoras planificadas para el proyecto.
El Product Owner es responsable de mantener y actualizar continuamente el backlog.
Backlog del Sprint:

Subset del backlog del producto que se selecciona para abordar durante el sprint.
Se establece durante la planificación del sprint y se mantiene constante durante el sprint.
Incremento:

Conjunto de funcionalidades totalmente desarrolladas y probadas al final de cada sprint.
Representa un avance tangible y utilizable del producto.

La implementación de SCRUM en el proyecto de Gestión de Carga Académica Docente Universitario proporcionará una estructura ágil y colaborativa, permitiendo entregas continuas y adaptación a medida que evolucionan los requisitos y las necesidades de los usuarios. La transparencia y la comunicación efectiva serán pilares fundamentales para el éxito del proyecto.

e. Modelo lógico de la base de datos

- **Entidades Principales:**

  Docente:
  Atributos: Código Docente (clave primaria), Nombre, Apellido,Dirección,Celular.

Asignatura:
Atributos: Código Asignatura(clave primaria), Nombre, Categoría(teórica, práctica, teórica/práctica),Pre-requisito,Créditos.

Portafolio:
Atributos: Código Portafolio(clave primaria), Código Docente(clave foránea), Código Asignatura(clave foránea).

Revisor:
Atributos: Código Docente(clave foránea).

Comisión Revisora:
Atributos: Código Comision(clave primaria), Código Docente (clave foránea).

- **Relaciones:**

Docente-Portafolio:
Relación: Un docente puede tener varios portafolios (uno por semestre).
Restricciones: Cada portafolio pertenece a un único docente.

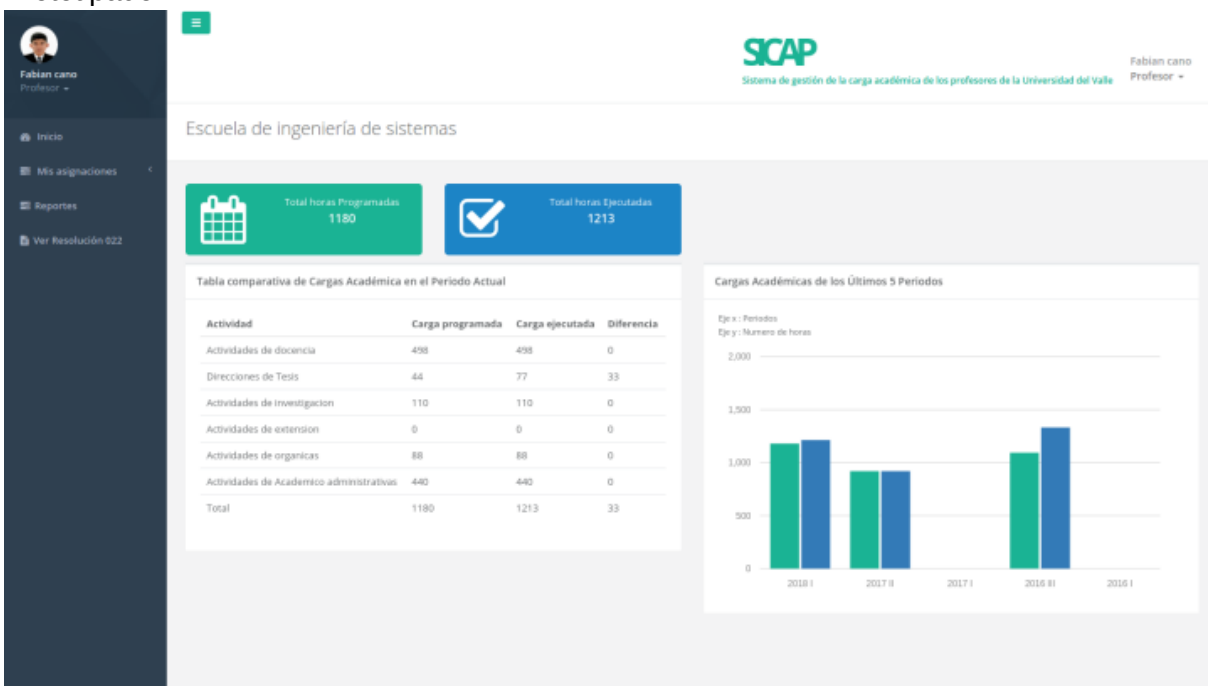
Asignatura-Portafolio:
Relación: Un portafolio incluye una o varias asignaturas.
Restricciones: Cada asignatura puede estar en varios portafolios (diferentes semestres).

Comisión Revisora-Revisor:
Relación: Una comisión revisora tiene varios revisores.
Restricciones: Cada revisor puede pertenecer a una única comisión revisora.

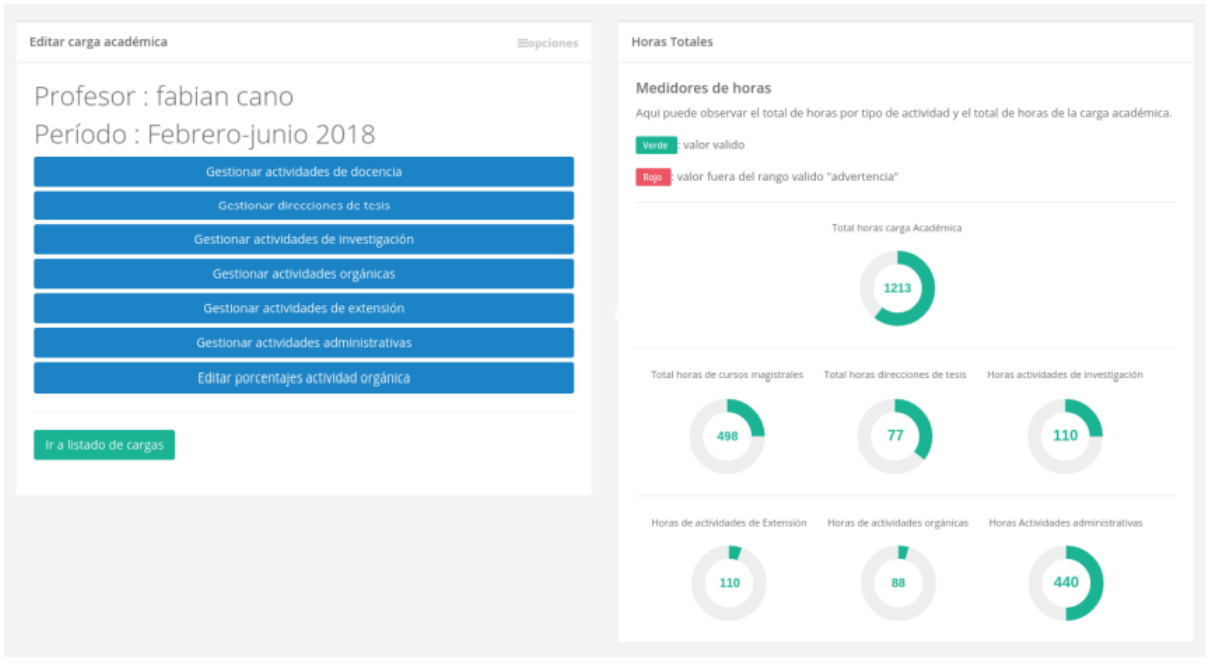
Comisión Revisora-Portafolio:
Relación: Una comisión revisora revisa varios portafolios.
Restricciones: Cada portafolio es revisado por una única comisión revisora.

Informe:
Atributos: ID informe (clave primaria), ID comision (clave foránea), Resultado (cumple, no cumple), Observaciones, Otros detalles.
Relación: Un informe está asociado a una comisión revisora.
Restricciones: Cada comisión revisora genera varios informes (uno por cada revisión).

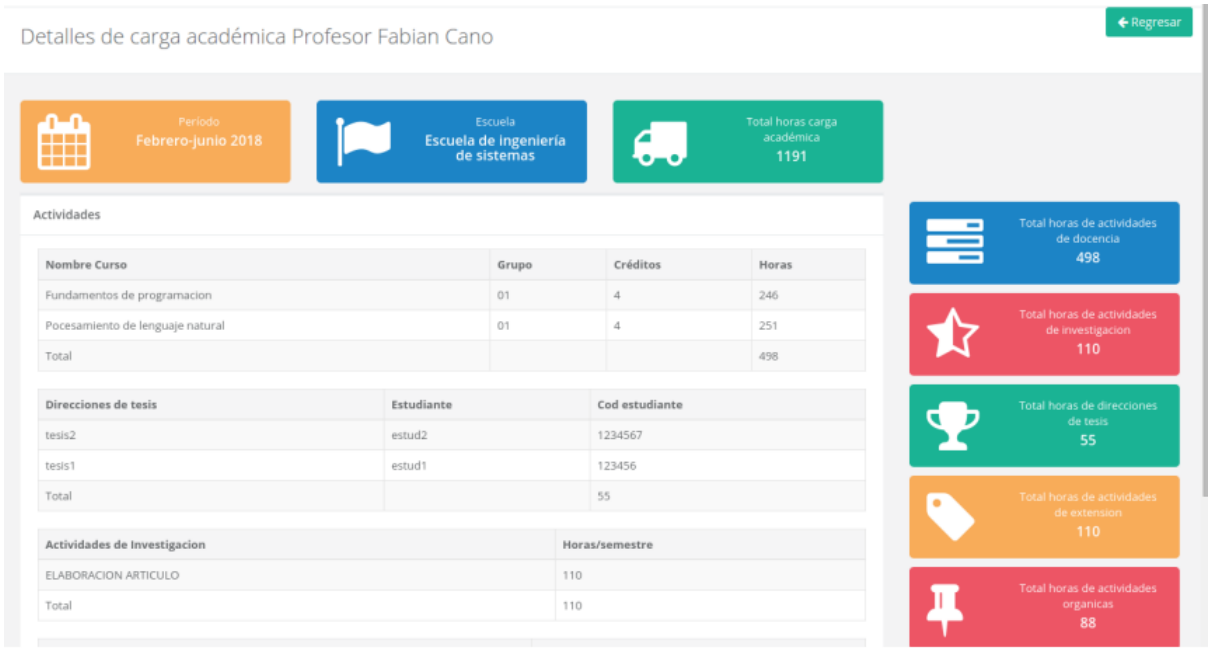
Prototipado



Vista de la carga académica docente(https://bibliotecadigital.univalle.edu.co/server/api/core/bitstreams/95a1ffcc-90d1-4b5b-b65f-4f3420a4855f/content)



Vista de carga académica, diseño si se completo de completado la carga de cada docente vista desde el coordinador(https://bibliotecadigital.univalle.edu.co/server/api/core/bitstreams/95a1ffcc-90d1-4b5b-b65f-4f3420a4855f/content)



Vista de los cursos y la carga vista desde un perfil del profesor (https://bibliotecadigital.univalle.edu.co/server/api/core/bitstreams/95a1ffcc-90d1-4b5b-b65f-4f3420a4855f/content)