“Portal de Oportunidades de Becas”

2025

Nombre	No. Lista	Código
Christopher Jose Flores Alfaro	16	20200261

Docente:
Ing.Carlos Amilcar Chávez Iraheta

Fecha: 23 de junio de 2025

1. DOCUMENTO DE VISIÓN Y ALCANCE

1.1. INTRODUCCIÓN

Este documento describe la visión y alcance del sistema Portal de Oportunidades de Becas, cuyo propósito es centralizar en un solo sitio web la información más relevante sobre becas nacionales e internacionales para estudiantes salvadoreños. Servirá como base para coordinar esfuerzos de desarrollo y garantizar el cumplimiento de objetivos clave.

1.2. CONTEXTO DE NEGOCIO

1.2.1. Antecedentes

En El Salvador, miles de estudiantes carecen de acceso fácil y oportuno a información sobre becas. La mayoría de las oportunidades están dispersas en redes sociales, portales institucionales o no se publican oficialmente. Esta situación limita las posibilidades educativas de muchos jóvenes.

1.2.2. Fase del problema

La falta de un repositorio digital actualizado, confiable y accesible para consultar becas, hace que muchos estudiantes pierden oportunidades valiosas. Además, no existe un sistema automatizado que avise sobre nuevas convocatorias o cambios en los requisitos.

1.2.3. Objetivos de negocio

ID	Descripción del objetivo de negocio
ON-1	Crear un portal web intuitivo y accesible desde múltiples dispositivos, disponible 24/7.
ON-2	Incorporar notificaciones personalizadas sobre nuevas oportunidades de becas.
ON-3	Permitir búsquedas avanzadas por nivel académico, país, institución, o área de estudio.
ON-4	Implementar la solución antes del 30 de noviembre del presente año.

1.3. VISIÓN DE LA SOLUCIÓN

1.3.1. Fase de visión

El Portal de Oportunidades de Becas será un sitio web dinámico donde los estudiantes salvadoreños podrán acceder a una base de datos centralizada de becas. Permitirá realizar búsquedas, recibir notificaciones personalizadas y leer experiencias reales de beneficiarios.

1.3.2. Características del sistema

ID	Descripción	Prioridad	Objetivo de negocio
CAR-01	Permitir a los estudiantes buscar becas por filtros personalizados (nivel, país, área).	Alta	ON-3
CAR-02	Ofrecer notificaciones sobre becas nuevas relacionadas al perfil del estudiante.	Alta	ON-2
CAR-03	Mostrar requisitos, fechas límite y beneficios de cada beca.	Alta	ON-1
CAR-04	Incluir testimonios y reseñas de estudiantes que han ganado becas.	Media	ON-1, ON-4
CAR-05	Incluir un panel administrativo para agregar, modificar y eliminar becas.	Alta	ON-1, ON-4
CAR-06	Permitir exportar las oportunidades de becas en PDF o compartirlas por redes sociales.	Media	ON-1
CAR-07	Garantizar la seguridad de datos personales de los usuarios registrados.	Alta	ON-1
CAR-08	Disponibilidad 24/7 con tiempos de recuperación menores a 5 minutos en caso de fallos.	Alta	ON-1, ON-4
CAR-09	Permitir registro y autenticación de usuarios mediante correo y redes sociales.	Alta	ON-1, ON-2

1.4. ALCANCE

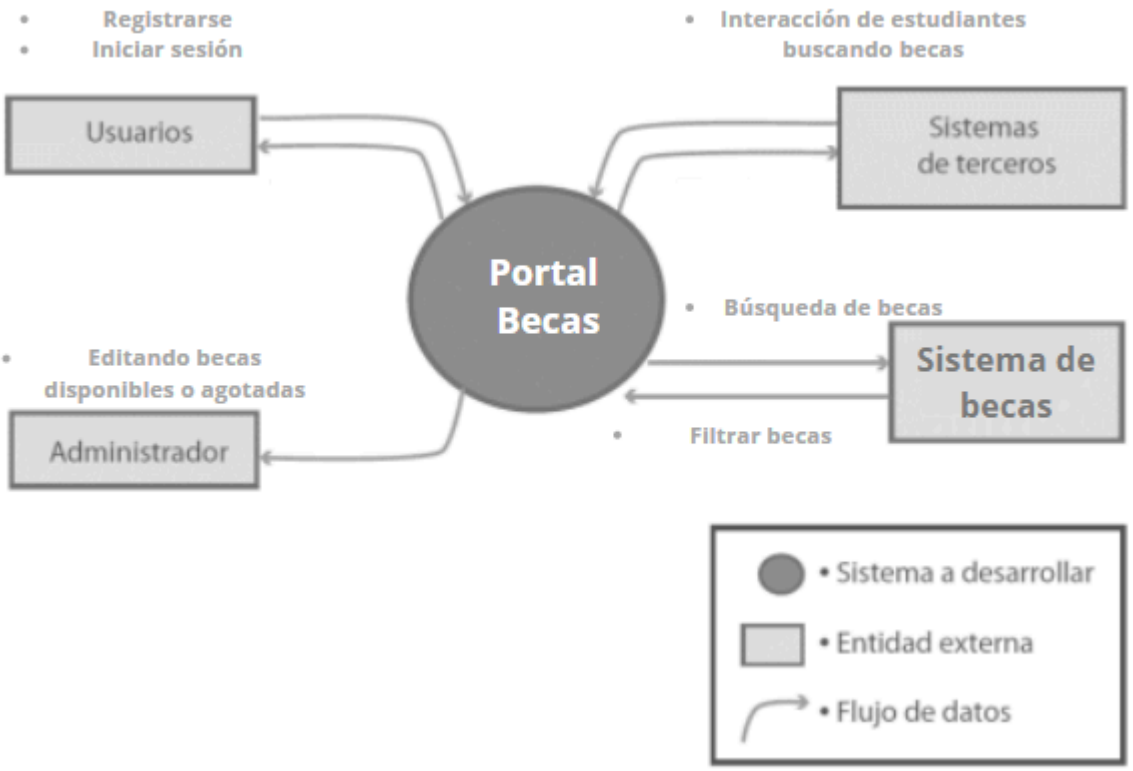
Número de Entrega	Tema Principal	ID de Características Incluidas
1.0	Funcionalidad básica del portal	CAR-01, CAR-02, CAR-03, CAR-05, CAR-07, CAR-09
2.0	Mejora visual y funciones secundarias	CAR-04, CAR-06
3.0	Robustez y disponibilidad	CAR-08

1.5. CONTEXTO DEL SISTEMA

1.5.1. Interesados

Nombre	Descripción	Responsabilidades
Christopher Flores	Líder de proyecto	Definir visión, supervisar desarrollo y entregables.
Estudiantes salvadoreños	Usuarios finales	Consultar y aplicar a becas.
Universidades e instituciones	Ofertantes de becas	Publicar y actualizar oportunidades.
Administrador del portal	Encargado de contenido	Gestionar publicaciones, validar información.
Desarrollador web	Técnico encargado del desarrollo	Diseñar e implementar el portal.
Equipo de infraestructura	Hosting/soporte técnico	Garantizar disponibilidad del sistema.

1.5.2. Diagrama de contexto



1.5.3. Entorno de operación

- Navegadores compatibles: Google Chrome , Firefox , Edge
- Dispositivos móviles: Android , iOS
- Plataforma: Servidor Linux o Windows, compatible con PHP, Python o Node.js
- Base de datos: MySQL o PostgreSQL
- Hosting: deberá soportar HTTPS, backup automático y disponibilidad
- Evitar uso de tecnologías obsoletas

1.6. INFORMACIÓN ADICIONAL

Los criterios de aceptación del proyecto incluyen:

- Sistema funcional y accesible públicamente en 8 meses.
- APIs de integración con instituciones de becas publicadas y documentadas en 10 meses.
- Protocolo de pruebas de aceptación aprobado al 100%.
- Entrega completa del código fuente, manual técnico y manual de administración.

2.Requerimiento de la arquitectura

2.1 Drivers funcionales

Para este caso de estudio, los requerimientos funcionales que determinarán las interacciones entre los usuarios y el sistema han sido especificados utilizando la técnica de casos de uso. Esta técnica permite modelar cómo cada tipo de usuario interactúa con el sistema para cumplir sus objetivos.

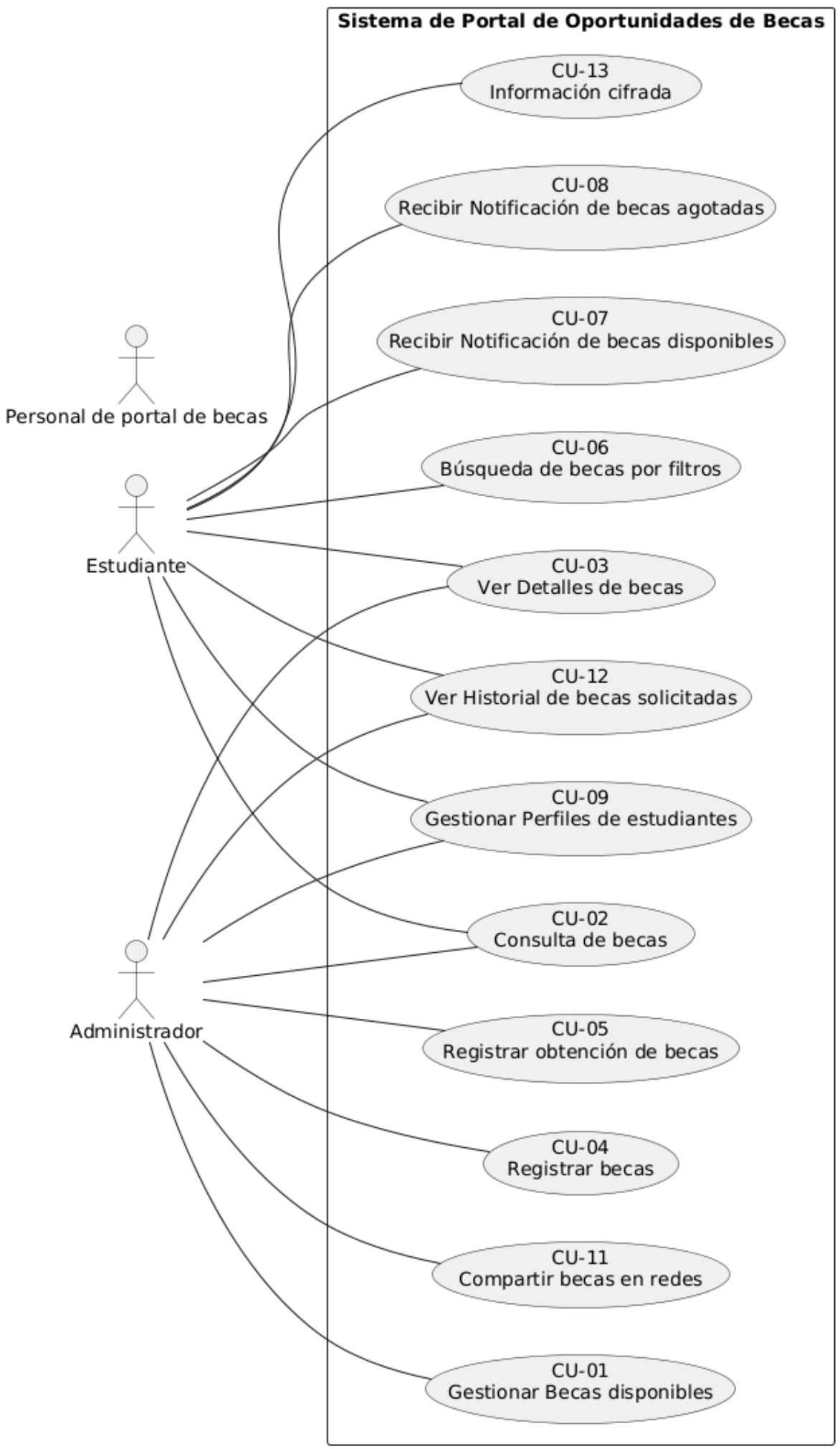
A continuación, se resumen los principales casos de uso que actúan como drivers funcionales:

ID	Caso de uso	Descripción resumida
CU-01	Registro de usuario	Permite que docentes y estudiantes puedan registrarse en la plataforma mediante un formulario.
CU-02	Inicio de sesión	Los usuarios pueden autenticarse usando su correo y contraseña para acceder a su cuenta.
CU-03	Gestión de becas	Permite a los administradores de contenido agregar, modificar, eliminar y publicar

		información de becas en el portal.
CU-04	Visualizar detalles de beca	Permite a los usuarios ver información completa sobre una beca específica, incluyendo requisitos y fechas de aplicación.
CU-05	Consulta sobre becas	Permite a los usuarios buscar y filtrar becas según criterios como nivel académico, área de estudio e institución.

2.1.1. Modelo de casos de uso

A continuación, se presentan los casos de uso derivados de las características del sistema descritas anteriormente:


Sistema de Portal de Oportunidades de Becas
CU-13
Información cifrada
CU-08
Recibir Notificación de becas agotadas
CU-07
Recibir Notificación de becas disponibles
CU-06
Búsqueda de becas por filtros
CU-03
Ver Detalles de becas
CU-12
Ver Historial de becas solicitadas
CU-09
Gestionar Perfiles de estudiantes
CU-02
Consulta de becas
CU-05
Registrar obtención de becas
CU-04
Registrar becas
CU-11
Compartir becas en redes
CU-01
Gestionar Becas disponibles
Personal de portal de becas
Estudiante
Administrador

ID	Descripción del Caso de Uso	Característica Asociada
CU-06	El estudiante busca oportunidades de becas aplicando filtros como nivel académico, país o área.	CAR-01
CU-07	El usuario se registra en el portal ingresando datos personales y creando una cuenta.	CAR-09
CU-08	El sistema envía notificaciones sobre nuevas becas relevantes según el perfil del usuario.	CAR-02
CU-09	El administrador del portal gestiona el contenido: añade, edita o elimina becas.	CAR-05
CU-10	Los estudiantes leen testimonios y experiencias de otros beneficiarios de becas.	CAR-04
CU-11	El estudiante comparte una beca a través de redes sociales desde el portal.	CAR-06
CU-12	El sistema muestra requisitos detallados, fechas límite y beneficios de cada oportunidad.	CAR-03
CU-13	El sistema protege la información del usuario usando protocolos de seguridad.	CAR-07
CU-14	El sistema garantiza disponibilidad continua del portal (24/7) y recuperación en caso de fallos.	CAR-08

2.1.2. Elección de casos de uso primarios

ID	Caso de Uso Primario	Justificación
CU-01	Buscar oportunidades de becas	Es el objetivo principal del sistema.
CU-03	Recibir notificaciones	Mejora la experiencia personalizada del estudiante.
CU-04	Gestionar contenido del portal	Garantiza que la información esté actualizada.

2.2. DRIVERS DE ATRIBUTOS DE CALIDAD

Los siguientes atributos de calidad se identificaron como relevantes y fueron priorizados con base en su importancia para el negocio y complejidad de implementación, los cuales se identificaron usando la técnica de escenarios, y su priorización.

ID	Categoría	Escenario	Prioridad
EAC-01	Desempeño	Un estudiante busca becas filtrando por área y nivel; el sistema responde en menos de 2 segundos.	Alta (A,A)
EAC-02	Disponibilidad	El sistema sufre una caída. Debe recuperarse completamente en menos de 5 minutos.	Alta (M,A)
EAC-03	Seguridad	Se intercepta una conexión en proceso de registro. Los datos están cifrados y no se accede a información útil.	Alta (A,B)
EAC-04	Usabilidad	Un estudiante llena mal el formulario de beca. El sistema indica claramente los errores y cómo corregirlos.	Media (M,A)
EAC-05	Escalabilidad	El sistema mantiene el desempeño con muchos estudiantes simultáneamente buscando becas.	Alta (A,A)
EAC-06	Mantenibilidad	Se agregan nuevos tipos de becas. La base de datos y el portal se actualizan sin necesidad de recompilar.	Media (M,B)

2.3. Drivers de restricciones

Las restricciones son condiciones impuestas al diseño y desarrollo del sistema que deben respetarse obligatoriamente. Estas pueden ser establecidas tanto por el cliente como por el equipo de desarrollo, e influyen directamente en las decisiones arquitectónicas. A continuación, se presentan las restricciones identificadas para el proyecto Portal de Oportunidades de Becas:

Tipo	Descripción
Del cliente	- Tiempos de entrega y presupuestos establecidos. - Véase la sección 1.5.3 del documento de visión y alcance.
De la organización de desarrollo	- Los ingenieros disponibles están familiarizados con los frameworks de Java: Java Server Faces (JSF), Spring e Hibernate.

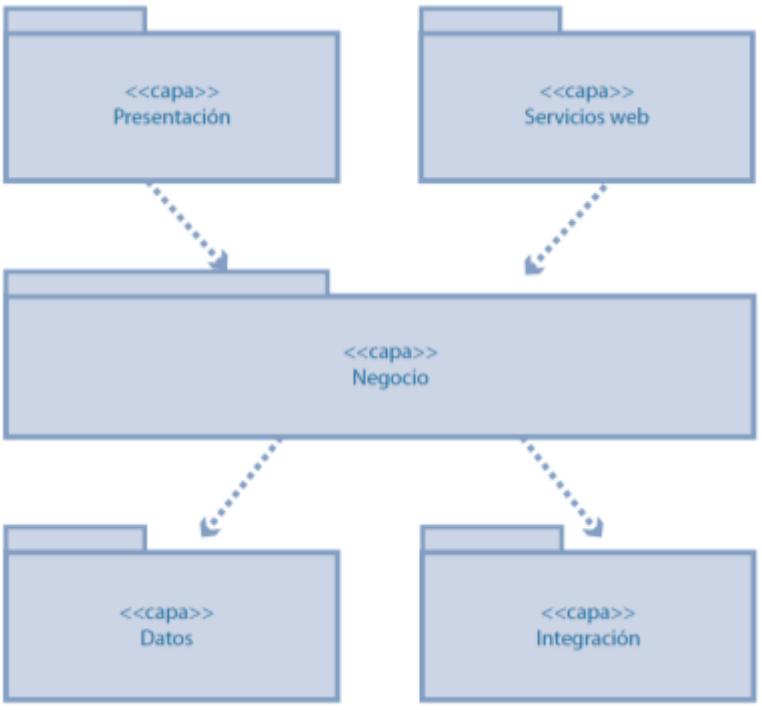
3.Diseño de la arquitectura

3.1 Primera iteración: estructuración general del sistema

Elemento a descomponer	Sistema
Drivers elegidos para la iteración	Dado que es la iteración inicial, se hace una estructuración general del sistema. Para hacerlo se toma en cuenta el conjunto de casos de uso primarios y escenarios de atributos de calidad, así como las siguientes restricciones (véase sección 1.5.3 del documento de visión): ● Uso desde navegadores web y dispositivos móviles. ● Integración de sistemas externos por medio de servicios web. ● Integración de servidor de BD legado.

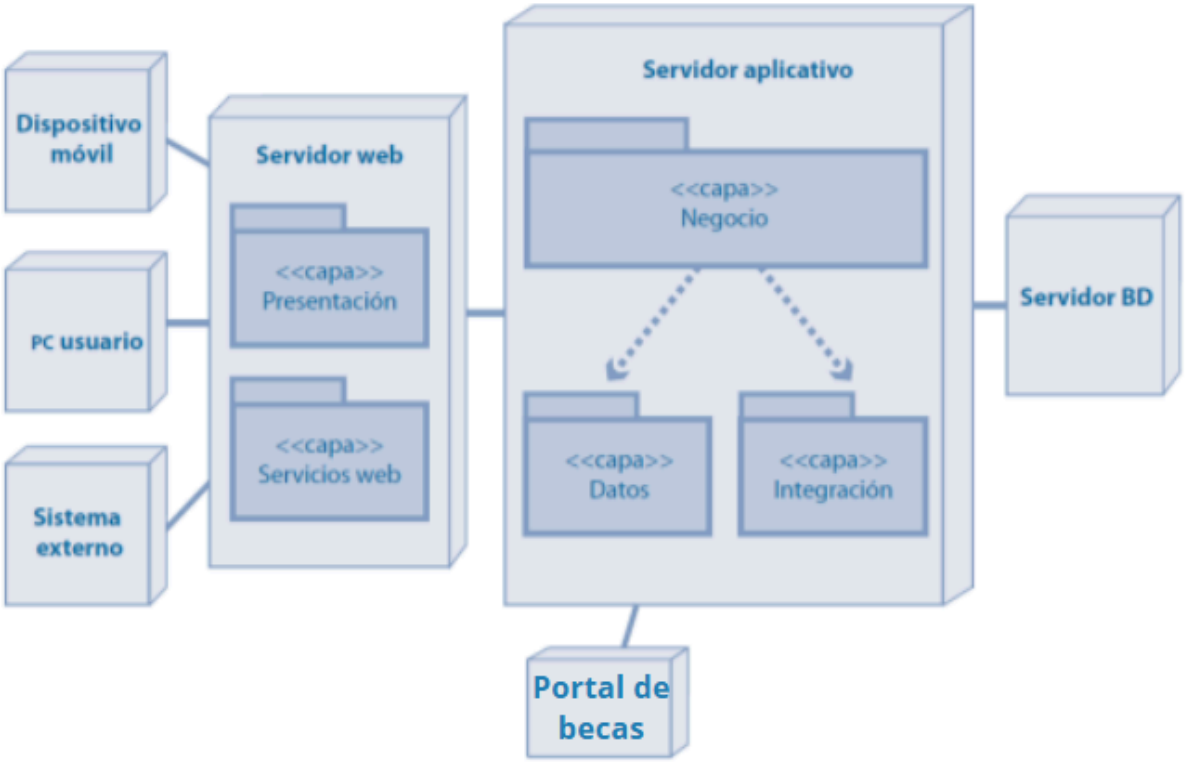
Conceptos de diseño	Justificación
Estilo o patrón arquitectónico de capas	Las capas permiten aislar de forma lógica distintas responsabilidades del sistema: los aspectos relacionados con la interacción con el usuario (capa de presentación), con la conexión de sistemas externos (capa de integración), el manejo de la lógica de negocio (capa de negocio) y la persistencia de los datos (capa de datos). Esto permite realizar cambios en la interfaz de usuario y asignar componentes a los ingenieros de manera más simple.
Estilo o patrón arquitectónico de N-tercios	Con la elección de este estilo se estructura el sistema desde un punto de vista de implantación, pues las capas de la aplicación se ubican en distintos “tercios” (nodos físicos). Esto permite aumentar la seguridad y facilita escalar el sistema y su mantenimiento.
Frameworks JSF, Spring e Hibernate	Estos se apegan a las capas definidas y son aquellos con los que el equipo de desarrollo está familiarizado.

Elemento	Responsabilidades principales	Propiedades de calidad relacionadas
Capa de presentación	Mostrar interfaz web y móvil al estudiante. Gestionar formularios de búsqueda, registro y alertas.	Usabilidad, desempeño, seguridad
Capa de servicios web	Exponer API REST para compartir oportunidades o acceder a funcionalidades desde sistemas externos.	Interoperabilidad, disponibilidad, seguridad
Capa de negocio	Aplicar reglas para filtrar becas, generar notificaciones, validar acceso, controlar flujo.	Modificabilidad, mantenibilidad, desempeño
Capa de datos	Gestionar la consulta, almacenamiento y actualización de becas, usuarios y testimonios.	Desempeño, disponibilidad, seguridad
Capa de integración	Conectar con redes sociales, correo, y servicios externos para compartir o enviar alertas.	Interoperabilidad, escalabilidad



Elemento	Responsabilidades	Propiedades de calidad relacionadas
PC del usuario	Ejecuta el navegador web desde donde los estudiantes acceden al portal.	Internet Explorer 10+, Firefox 10+, Google Chrome 17+

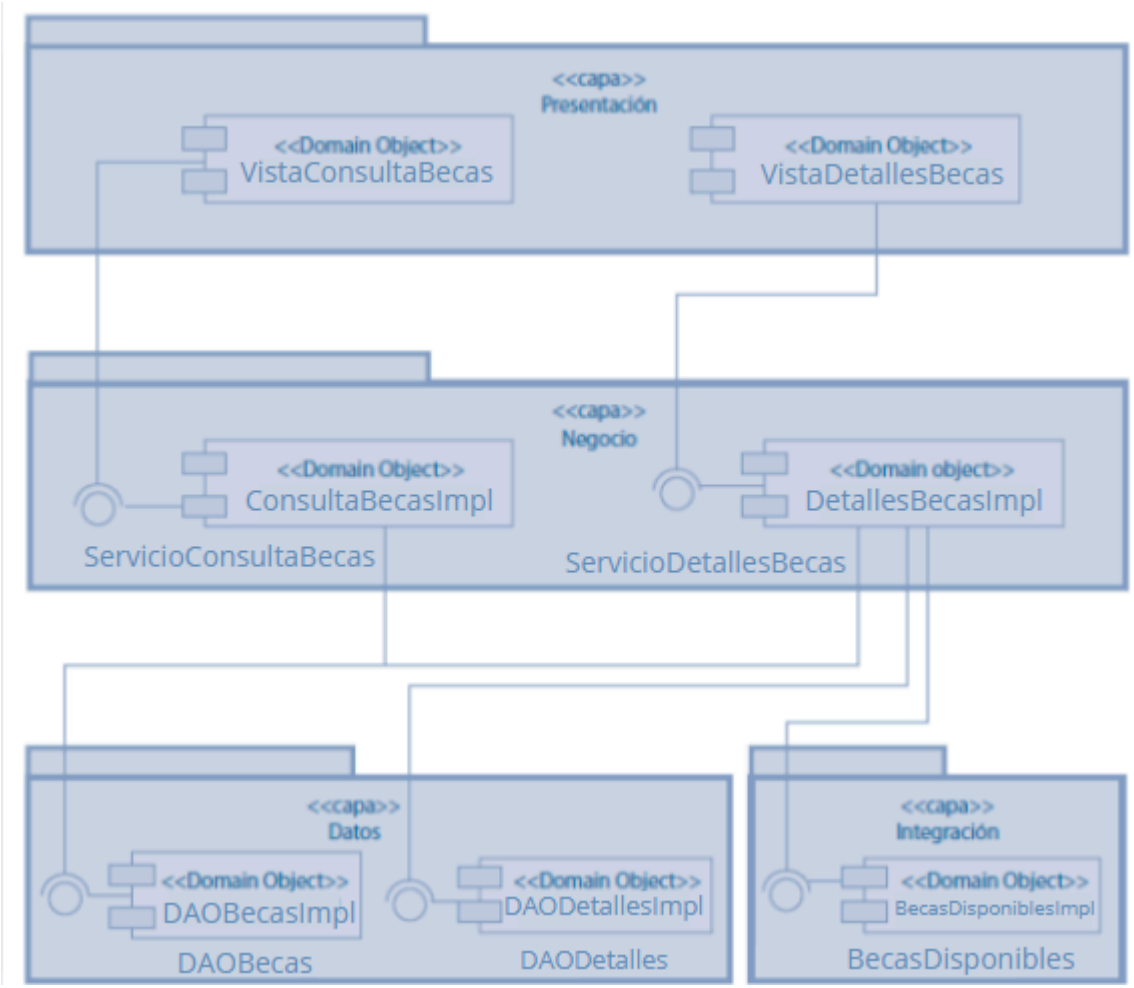
Dispositivo móvil	Permite el acceso responsivo al portal desde celulares o tablets, con igual funcionalidad.	iPhone/iPod Touch, iOS 6+, Android 4
Servidor web	Aloja la interfaz del sitio, gestiona las solicitudes HTTP y entrega contenido al usuario.	Tipo = Apache
Servidor aplicativo	Ejecuta la lógica de negocio del sistema, controla flujos, genera alertas, valida datos.	Lenguaje = PHP Memoria = Por definir Procesador = Por definir
Servidor BD	Administra la base de datos relacional con la información de becas, usuarios y testimonios.	RDBMS = MySQL



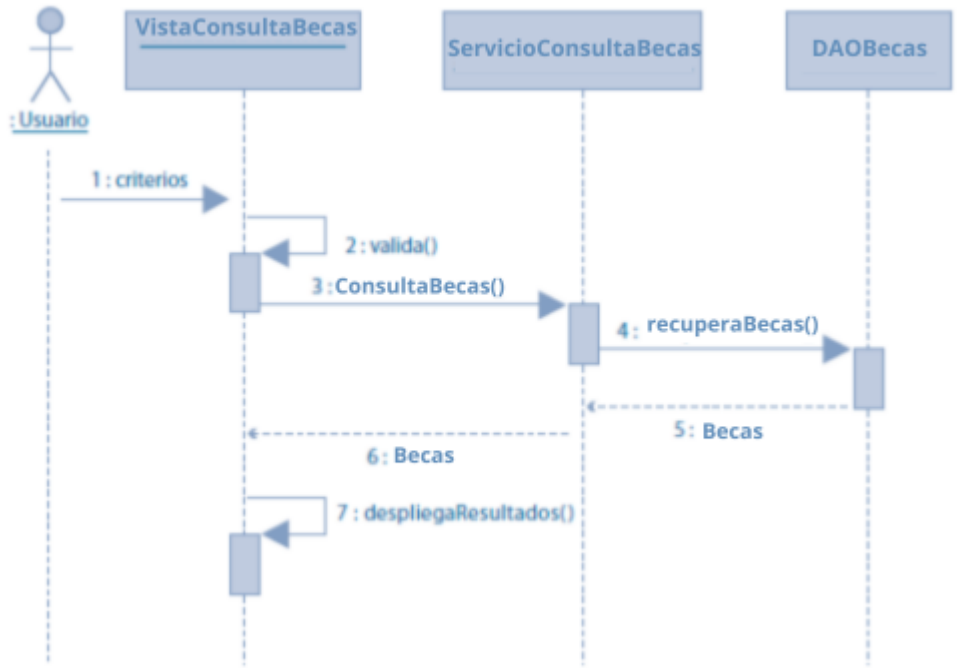
3.2 Segunda iteración: integración de la funcionalidad a las capas

Elemento a descomponer	Drivers elegidos para la iteración
Capas del sistema (Presentación, Servicios Web, Negocio, Datos, Integración)	**Casos de uso primarios**: • CU-02 (Inicio de sesión) • CU-03 (Gestión de becas) • CU-05 (Consulta de becas)

Concepto de Diseño	Justificación
Domain Object	Encapsula la lógica y datos relacionados con entidades clave como usuarios, becas y evaluaciones, favoreciendo el mantenimiento y reutilización.
Data Mapper	Facilita la separación entre objetos de dominio y lógica de acceso a datos, especialmente al utilizar una base de datos NoSQL o MySQL
Explicit Interface	Mejora la claridad de las interacciones entre componentes al definir interfaces públicas explícitas.
Encapsulated Implementation	Asegura que los detalles internos estén ocultos, permitiendo modificar la implementación sin afectar a otros componentes.



Elemento	Responsabilidad	Propiedades de calidad relacionadas
ServicioConsultaBecas (Presentación)	Recopilar los filtros ingresados por el estudiante (nivel, país, área) y enviarlos a la capa de servicios web.	HTML + CSS. Validaciones básicas. Interfaz responsiva.
DAOBecas (Presentación)	Mostrar al usuario una lista de oportunidades de becas filtradas y ordenadas.	Generada dinámicamente desde datos REST. Responsive.
VistaConsultaBecas (Servicios Web)	Procesar login, emitir tokens de sesión y conectarse a la lógica de negocio para validación.	API REST. Métodos POST /login, POST /logout. Seguridad por token o sesión.
VistaDetallesBecas(Servicios Web)	Exponer métodos para buscar, listar y ver detalles de becas.	API REST. Métodos GET /clases, POST /clases, DELETE /clases/{id}.
BecasDisponiblesImpl (Negocio)	Validar credenciales, administrar roles (estudiante, admin) y estado de sesión.	Lógica desacoplada de presentación. Validación con cifrado.
GestorBecas (Negocio)	Aplicar reglas para filtrar, clasificar y retornar resultados de becas.	Encapsula reglas de negocio. Métodos reutilizables.
DAODetallesImpl (Datos)	Traducir objetos de dominio Usuario al formato persistente y viceversa.	Implementado con consultas a MySQL o colecciones en MongoDB.
ServicioDetallesBecas (Datos)	Gestionar consultas y operaciones CRUD sobre la base de datos de becas.	Optimizado con índices. CRUD completo.
NotificacionesService (Integración)	Enviar alertas personalizadas por correo o push notificando nuevas becas al estudiante.	Interfaz con APIs externas. Usa HTTP REST. Resiliente ante errores de red.

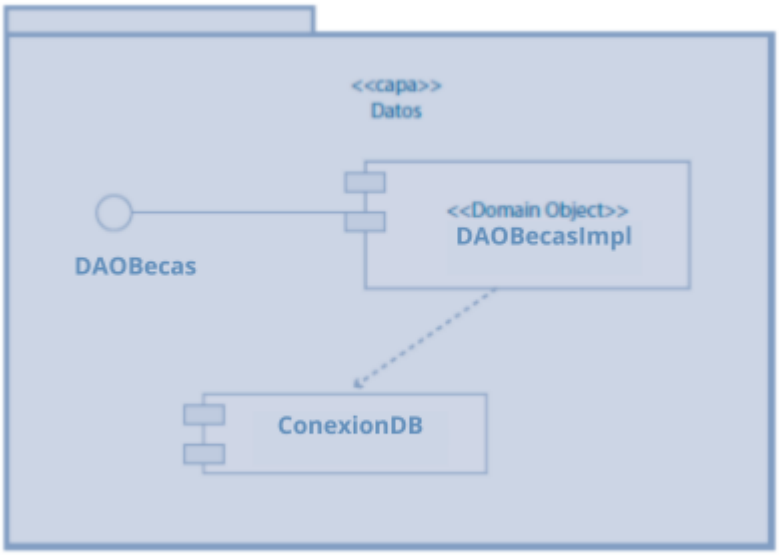


3.3. Tercera iteración: desempeño en capa de datos

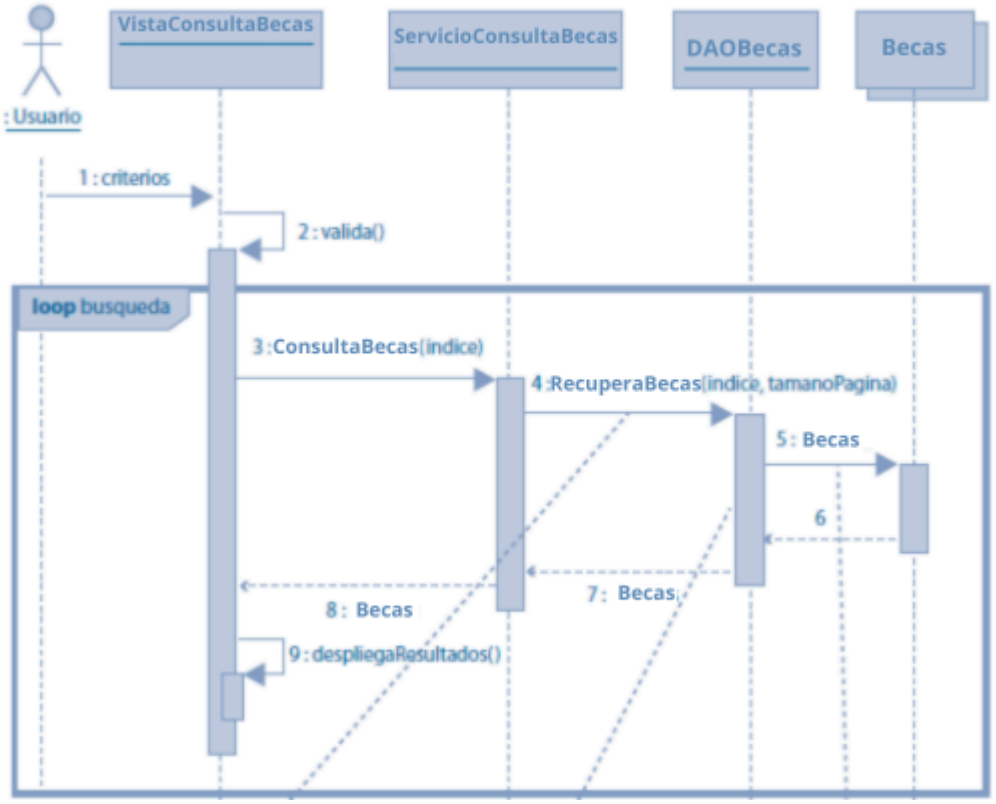
Elemento a descomponer	Drivers elegidos para la iteración
Capa de datos	**Escenario de atributo de calidad EAC-01 (Desempeño)**: Un estudiante realiza búsqueda de becas con filtros , y el sistema debe responder en mitad segundos bajo muchos usuarios concurrentes.

Elemento	Responsabilidad	Propiedades
ClaseMapper	Optimizar consultas de búsqueda y filtrado de becas. Implementa patrones Lazy y Eager según necesidad del escenario.	Implementación con índices en campos clave, uso de agregaciones eficientes (Mysql).
UsuarioMapper	Refinado para garantizar búsqueda eficiente por correo electrónico durante login.	Añadido índice por correo. Consulta optimizada, carga selectiva de datos.
ConexiónBD	Administrar recursos de conexión a base de datos mediante un pool, permitiendo múltiples accesos concurrentes sin bloqueo.	Uso de biblioteca de conexión persistente (PHP)

VistaMaterializadaClases	Estructura de soporte para almacenar temporalmente resultados de búsqueda de becas más solicitadas.	Actualización periódica, acceso directo en menos de 1 segundo, uso interno transparente para el usuario.



Interfaz	Detalles / Descripción
ClaseMapper	Clase[] consultarClases(filtros) → Recupera la lista de becas aplicando filtros con índices optimizados.
UsuarioMapper	Usuario buscarPorCorreo(correo) → Retorna un usuario según su correo, usando índice único en la base de datos.
ConexiónBD	Conexion obtenerConexion() → Devuelve una conexión activa desde el pool para ejecución segura y concurrente.
VistaMaterializadaClases	Clase[] obtenerClasesFrecuentes() → Retorna resultados precargados de búsquedas populares para consulta rápida.



4.Documentación

En esta sección se documenta la arquitectura resultante del proceso de diseño. Este método establece que una buena documentación arquitectónica debe contener vistas organizadas en tres categorías principales:

- **Vistas de módulos:** representan la estructura lógica del sistema.
- **Vistas de componentes y conectores:** representan la arquitectura en tiempo de ejecución.
- **Vistas de localización:** representan cómo se asignan elementos del software a los elementos físicos.

Adicionalmente, se puede generar documentación complementaria como justificación de decisiones, descripción de interfaces, y diagramas de contexto de alto nivel.

4.1 Generar una lista de vistas candidatas

Para identificar las vistas necesarias, el arquitecto realiza una matriz que cruza los interesados del sistema con las categorías de vistas, lo cual permite identificar qué vistas son necesarias, para quién y con qué nivel de detalle deben elaborarse.

Interesado	**Capas (Módulos)**	**Descomposición (Módulos)**	**Uso (Módulos)**	**Modelo de Datos**	**Cliente-Servidor (C&C)**	**Implantación (Localización)**	**Instalación**	**Asignación de trabajo**
Desarrollador backend	Detallada (d)	Detallada (d)	Media (m)	Detallada (d)	Media (m)	Media (m)	Media (m)	Detallada (d)
Desarrollador frontend	Detallada (d)	Detallada (d)	Detallada (d)	Baja (b)	Media (m)	Baja (b)	Detallada (d)	Detallada (d)
Administrador del portal	Baja (b)	Baja (b)	Baja (b)	Media (m)	Baja (b)	Media (m)	Baja (b)	Baja (b)
Estudiante (usuario final)	Baja (b)	No aplica	Baja (b)	Baja (b)	Baja (b)	No aplica	No aplica	No aplica

Líder de proyecto	Media (m)	Media (m)	Medi a (m)	Media (m)	Media (m)	Media (m)	Media (m)	Detallada (d)

4.2 Combinar las vistas

Durante esta etapa, se analizan las vistas candidatas con el objetivo de reducir su número y evitar redundancias, siempre que dicha combinación no afecte su comprensión ni mantenimiento.

Vista	Estilos combinados	Detalle	Justificación
Vista 1	Capas, Descomposición, Uso	Muy detallada	Las capas son pocas y de semántica conocida, por lo que se pueden combinar.
Vista 2	Modelo de datos	Muy detallada	Información distinta a las anteriores. Documenta entidades clave.
Vista 3	Cliente-servidor	Muy detallada	Útil para representar flujos de casos de uso.
Vista 4	Implantación e Instalación	Muy detallada	Se puede consolidar en una sola sin complejidad adicional.

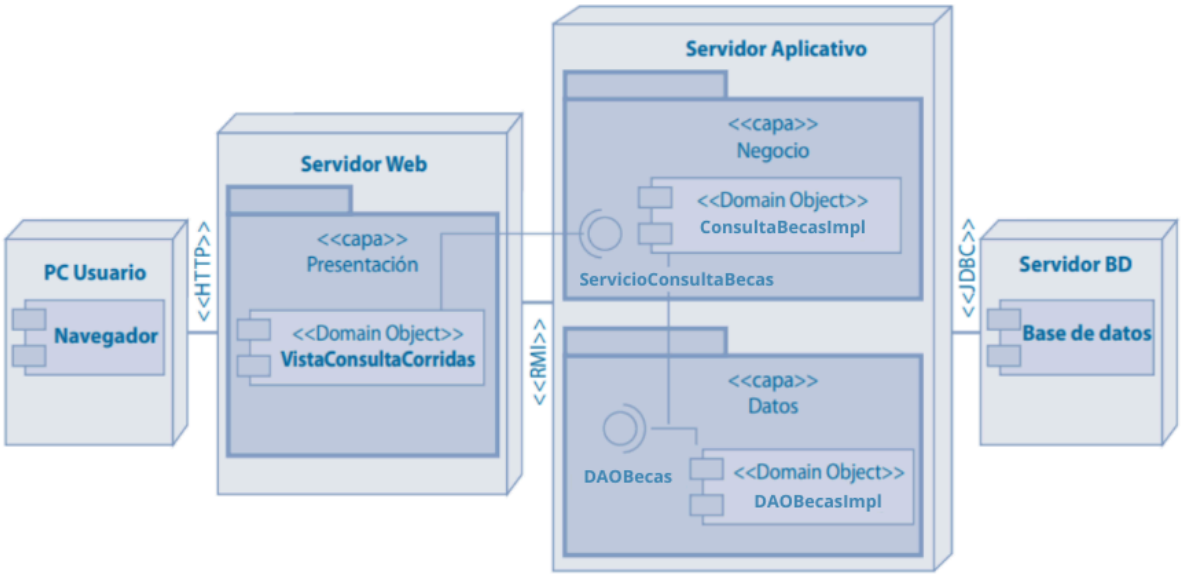
4.3 Priorizar las vistas

Una vez identificadas y combinadas las vistas, se establece un orden de prioridad para su documentación, basado en su utilidad durante el desarrollo del sistema.

Prioridad	Vista	Motivo de prioridad
1	Vista 1 (Capas + Uso + Descomposición)	Fundamental para guiar el diseño lógico y dividir responsabilidades.
	Vista 2 (Modelo de datos)	Esencial para diseñar base de datos y lógica de acceso.
2	Vista 3 (Cliente-servidor)	Ilustra los flujos y casos de uso primarios del sistema.
	Vista 4 (Implantación + Instalación)	Ayuda a configurar servidores y despliegue del sistema.

4.4 Ejemplo de vista

Propósito	Describir la interacción entre los componentes durante la búsqueda de becas.
Estilo arquitectónico	Cliente-servidor con invocación síncrona
Escenario representado	Un estudiante realiza una búsqueda de becas aplicando filtros desde su navegador.
Participantes	Navegador del estudiante, Servidor web, Capa de negocio, Capa de datos
Conectores	HTTP/HTTPS para navegadores, invocaciones internas entre capas, consultas SQL hacia la base de datos.
Variabilidad	El servidor puede escalar horizontalmente. Las búsquedas pueden usar caché para resultados comunes.
Restricciones	Tiempo de respuesta menor a 2 segundos. Soportar hasta muchos usuarios concurrentes.



a) Elementos

Nombre de la vista: Distribución física del sistema
Estilo arquitectónico: Vista de localización
Propósito: Describir cómo y dónde se ejecuta el sistema de oportunidades de becas.

Elemento físico	Descripción técnica
Cliente	Dispositivo del usuario (PC, tablet, móvil) con navegador web
Navegador	Chrome, Firefox u otro navegador compatible
Servidor Web	Apache/Nginx con motor PHP o entorno Node.js
Base de datos	Servidor con MySQL o MariaDB
Red	Conexión vía HTTP/HTTPS
Proceso de instalación	Descargar archivos, configurar .env, importar base de datos, ejecutar servidor

b) Relaciones

Nombre	**Responsabilidad**	**Propiedades**
Request/Response (L)	Permite la comunicación entre los módulos internos del sistema que proporcionan y requieren servicios.	Comunicación = Síncrona Protocolo = MI (Method Invocation o Invocación a método) Modo = Local (L)
Request/Response (R)	Permite la comunicación entre el usuario y el servidor web del portal de becas.	Comunicación = Síncrona Protocolo = HTTP, HTTPS, RMI (Remote Method Invocation o Invocación a método remoto) Modo = Remoto (R)

c) Interfaces

Interfaz: IAuthService

Nombre del método	Login
Precondiciones	El usuario debe estar registrado previamente en el sistema.
Poscondiciones	Se establece una sesión activa y se retorna el objeto Usuario si las credenciales son correctas.

Parámetros y retorno		
Nombre	**Tipo**	**Descripción**
Correo	String	Correo electrónico del usuario
Contraseña	String	Contraseña en texto plano
Valor de retorno	Usuario	Objeto que representa al usuario autenticado

Excepciones	
Nombre	**Descripción**
UsuarioNoEncontradoExcep	El correo ingresado no pertenece a ningún usuario registrado
CredencialesInvalidasExcep	La contraseña ingresada no es válida

Nombre del método	Logout
Precondiciones	Debe existir una sesión activa asociada al token.
Poscondiciones	Se cierra la sesión activa correspondiente al token dado.

Parámetros y retorno		
Nombre	**Tipo**	**Descripción**
Token	String	Token de sesión generado tras login
Valor de retorno	Boolean	“true” si se cerró la sesión correctamente

Excepciones	
Nombre	**Descripción**
TokenInvalidoExcep	El token proporcionado no es válido o expiró

Interfaz: IBecasService

Nombre del método	listarBecas
Precondiciones	El usuario debe estar autenticado.

Poscondiciones	Se devuelve una lista de becas disponibles.

Parámetros y retorno		
Nombre	**Tipo**	**Descripción**
Ninguno	Ninguno	Ninguno
Valor de retorno	Clase[]	Lista de objetos Becas disponibles

Excepciones	
Nombre	**Descripción**

Excepciones	
AccesoDenegadoExcep	El usuario no tiene permisos para ver las becas

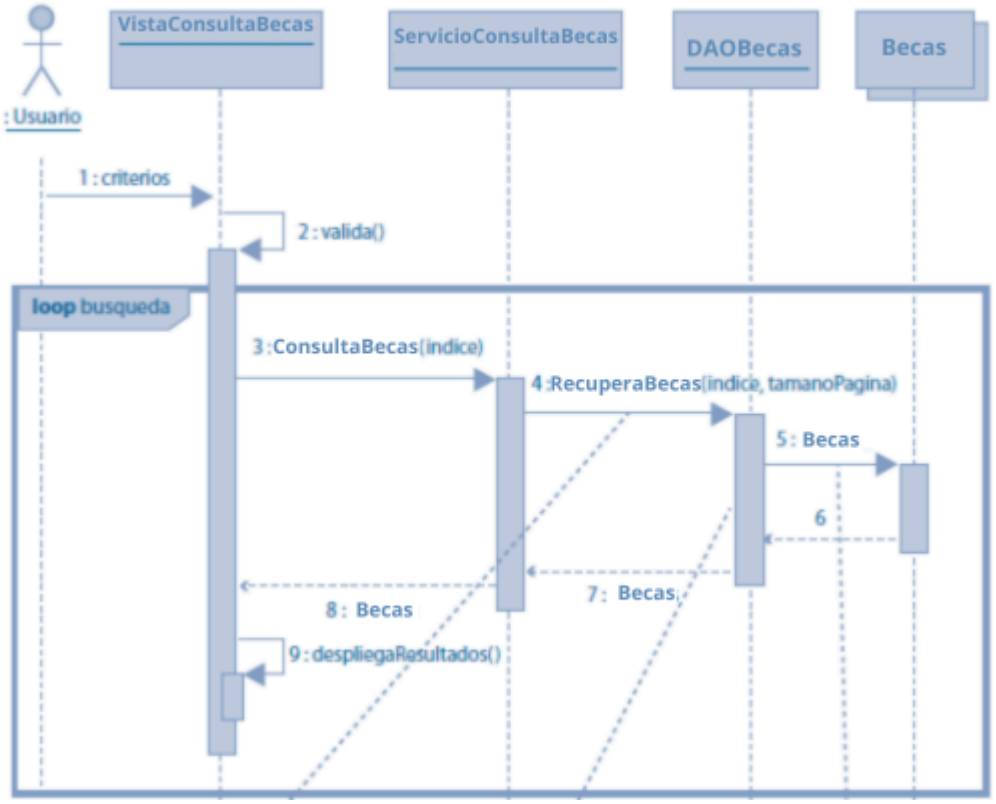
Nombre del método	crearBeca
Precondiciones	El usuario debe tener rol de alumno.
Poscondiciones	Se guardan las becas disponibles en la base de datos.

Parámetros y retorno		
Nombre	**Tipo**	**Descripción**
datos	BecaDTO	Objeto con la información de la Beca
Valor de retorno	Beca	Objeto Beca creado

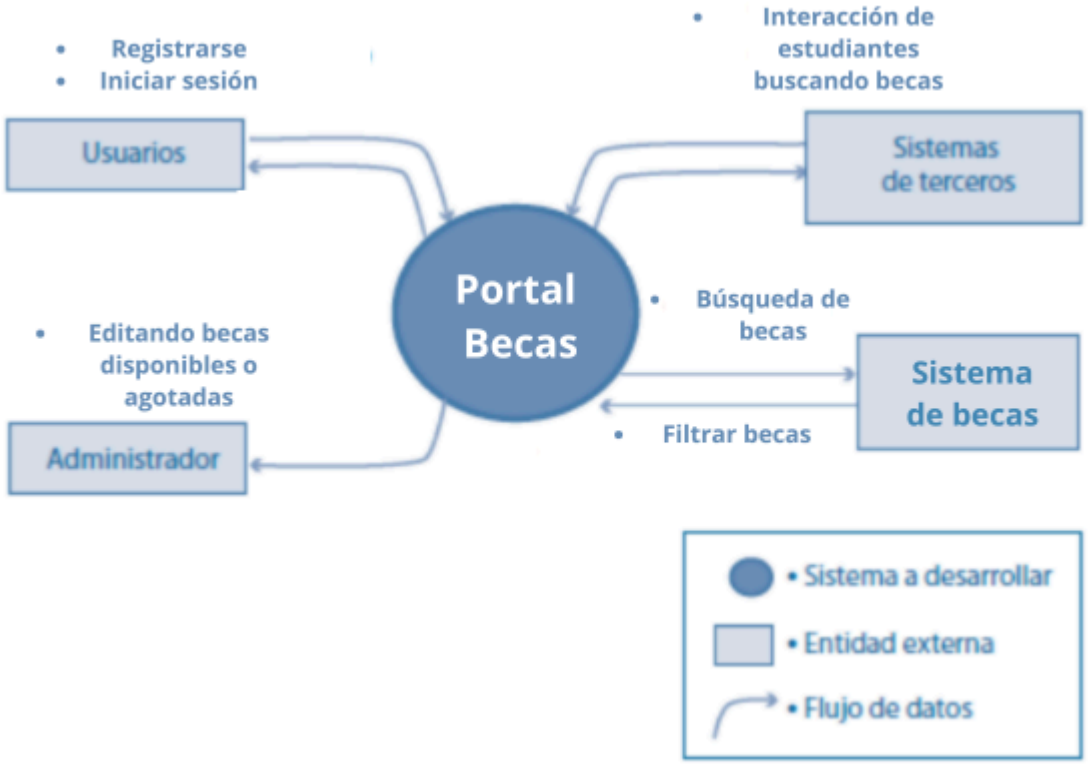
Excepciones	
Nombre	**Descripción**
PermisoDenegadoExcep	El usuario no tiene permisos filtrar las clases
DatosInvalidosExcep	La información de entrada no es válida

Nombre del método	eliminarBeca
Precondiciones	El usuario debe iniciar sesion para visualizar los tipos de beca
Poscondiciones	LaBeca es eliminada de la base de datos.

Parámetros y retorno		
Nombre	**Tipo**	**Descripción**
Id	String	Identificador único de la clase
Valor de retorno	Boolean	“true”si se eliminó correctamente



Excepciones	
Nombre	**Descripción**
BecaNoEncontradaExcep	No se encontró ninguna Beca con ese ID
PermisoDenegadoExcep	El usuario no tiene autorización para eliminarla



Guía de variabilidad

La guía de variabilidad identifica los puntos del sistema que pueden modificarse o configurarse sin afectar toda la arquitectura. Es útil para adaptaciones futuras.

Elemento variable	Descripción	Opciones / Alternativas	Impacto
Sistema de autenticación	Método de validación de usuarios.	Login tradicional / Google OAuth / LDAP	Medio: cambia lógica del Controlador Login
Almacenamiento	Motor de base de datos.	MySQL / PostgreSQL / SQLite	Bajo si se mantiene SQL estándar
Visualización de notas	Cómo se muestran las notas a estudiantes.	Texto plano / Gráfica / PDF descargable	Bajo: cambia solo capa de presentación

Módulo de tareas	Estructura de entrega de tareas.	Archivos adjuntos / Texto plano / Enlace a recurso externo	Bajo: afecta solo lógica y validaciones

Descripción de decisiones arquitectónicas

Las decisiones arquitectónicas son fundamentales para definir la dirección técnica del proyecto. A continuación se documentan las más relevantes tomadas para EduSoft:

Decisión	**Motivo / Justificación**
Uso de arquitectura cliente-servidor	Permite separar responsabilidades y facilita el mantenimiento del sistema.
División por capas (presentación, servicios, lógica, datos)	Mejora la modularidad, escalabilidad y reutilización del código.
Uso de MySQL como sistema de almacenamiento	Base de datos confiable, ampliamente usada, con buena documentación y soporte en entornos educativos.
Desarrollo web con PHP	Tecnología accesible, conocida por el equipo, adecuada para proyectos escolares y fácil despliegue.
Interfaz web responsive	Garantiza la accesibilidad desde distintos dispositivos (PC, móviles, tablets).
Controladores individuales por módulo	Facilita la organización del código y la asignación de responsabilidades dentro del equipo de desarrollo.

5.Evaluación

5.1. Realización de la evaluación

La evaluación de la arquitectura se realiza utilizando el método ATAM (Architecture Tradeoff Analysis Method). Este método permite identificar riesgos, sensibilidades y puntos de compromiso en las decisiones de diseño arquitectónico.

Para este caso de estudio, se presenta la fase de familiarización, que permite al equipo de evaluación y arquitectura explorar los aspectos más relevantes del diseño sin intervención externa de otros interesados.

5.1.1. Identificación de las decisiones arquitectónicas

Durante el diseño de la arquitectura del sistema Portal de Oportunidades de Becas, se tomaron las siguientes decisiones principales:

Decisión
El sistema se estructura en capas: presentación, servicios web, negocio, integración y datos.
Se utilizan los patrones arquitectónicos Capas y N-tercios para organizar lógica e implantación.
Se seleccionaron frameworks familiares al equipo: Spring Boot, Hibernate, React.
Se diseñaron componentes reutilizables desacoplados mediante interfaces explícitas.
El acceso a datos se optimiza con patrones como Lazy y Eager acquisition.
Se incluye un servicio de integración con sistemas externos (redes sociales, correo, APIs REST).
La lógica de negocio se encapsula en gestores específicos (gestor de becas, gestor de usuarios).

5.1.2. Generación del árbol de utilidad

El siguiente árbol resume la estructura de atributos de calidad identificados como más relevantes para la arquitectura de el portal de becas:

Atributo de Calidad	Escenario Descrito
Funcionalidad	Registro, login, gestión de becas, filtrar las becas,
Desempeño (EAC-01)	Estudiante hace clic en “Buscar becas” con usuarios conectados simultáneamente.
Disponibilidad (EAC-02)	Ocurre una falla interna durante el uso normal del sistema.
Seguridad (EAC-03)	Un atacante intercepta el tráfico durante la carga del sistema.
Usabilidad (EAC-04)	Estudiante llena mal un formulario de inicio de sesión.
Facilidad de prueba (EAC-05)	QA realiza pruebas unitarias sobre componentes de lógica y persistencia.
Modificabilidad (EAC-06)	Se agrega soporte para modo oscuro en la interfaz.

5.1.3. Análisis de las decisiones arquitectónicas

A continuación se analizan las decisiones clave en relación con los escenarios propuestos:

ID	Escenario	Análisis
EAC-01	El estudiante realiza búsqueda de becas con sobrecarga en el sistema.	La arquitectura en capas + separación lógica mejora el desempeño; sin embargo, es necesario configurar índices de búsqueda y optimizar peticiones.
EAC-02	Ocurre una falla durante el uso del sistema. Se requiere recuperación .	La separación en capas facilita reinicios rápidos. Se recomienda definir procesos automáticos de respaldo y recuperación.

EAC-03	Un atacante intercepta datos durante la subida de datos sobre las becas disponibles.	El uso de HTTPS y cifrado de contraseñas mitiga el riesgo. Falta reforzar políticas de tokens o autenticación más robusta.
EAC-04	Usuario llena mal un formulario de registro o inicio de sesión.	Se aplican validaciones de entrada y retroalimentación visual. Esto mejora la experiencia del usuario y previene errores.

5.2 Resultados de la evaluación

A partir del análisis ATAM, se identifican riesgos, no-riesgos, y recomendaciones para futuras iteraciones del diseño arquitectónico.

Tipo	Descripción
R1	No se especifica claramente el mecanismo de respaldo ante caídas del sistema (failover o replicación).
R2	Riesgo de bajo rendimiento si se supera el número de usuarios simultáneos previstos.
R3	No se han modelado escenarios para ataques de denegación de servicio o sesiones maliciosas.
R4	Dependencia directa de servicios de notificación externos sin control de contingencia.
NR1	Separación clara de capas favorece la mantenibilidad y escalabilidad.
NR2	Reutilización de componentes con interfaces explícitas.
NR3	Uso de arquitectura cliente-servidor facilita implementación distribuida.