

UACM



Diseño de software

Puntos de vista

SAR

Valadez Carmona Guadalupe Yamileth

Rodríguez Cervantes Kevin Manzur

Cruz Ovando Cristela Adelaida

HISTORIAL DE VERSIONES

FECHA	VERSIÓN	DESCRIPCIÓN	AUTOR@S
30/01/2025	0.50	<ul style="list-style-type: none"> Versión preliminar del análisis de los puntos de vista para la versión del programa 6.50 	Guadalupe Yamileth, Manzur Rodríguez, Cristela Adelaida
31/01/2025	1.50	<ul style="list-style-type: none"> Nuevo formato del documento. 	Guadalupe Yamileth, Manzur Rodríguez, Cristela Adelaida
21/02/2025	2.00	<ul style="list-style-type: none"> Actualización del documento, siguiente el 'Estándar de Documentación V - 2.00' 	Manzur Rodríguez
07/03/2025	2.10	<ul style="list-style-type: none"> 5.1 Introducción. 5.2 Punto de vista contextual. 	Guadalupe Yamileth
14/03/2025	2.20	<ul style="list-style-type: none"> 5.3 Punto de vista de la composición. 	Manzur Rodríguez
23/03/2025	2.30	<ul style="list-style-type: none"> 5.4 Punto de vista lógico. 	Cristela Adelaida
23/03/2025	2.40	<ul style="list-style-type: none"> 5.5 Punto de vista de la dependencia. 	Guadalupe Yamileth
23/03/2025	2.50	<ul style="list-style-type: none"> 5.6 Punto de vista informativo. 	Manzur Rodríguez
27/03/2025	2.60	<ul style="list-style-type: none"> Indicaciones para no desarrollar los puntos de vista 5.6 y 5.13 indicados en el estándar IEEE 1016 2009 	Guadalupe Yamileth, Manzur Rodríguez, Cristela Adelaida
10/04/2025	2.70	<ul style="list-style-type: none"> Corrección del formato del documento. 	Manzur Rodríguez
11/04/2025	2.80	<ul style="list-style-type: none"> Se utiliza el formato corregido. 	Manzur Rodríguez
11/04/2025	2.90	<ul style="list-style-type: none"> 5.7 Punto de vista de uso de patrones. 	Cristela Adelaida
25/04/2025	3.00	<ul style="list-style-type: none"> 5.8 Punto de vista de la interfaz. 	Guadalupe Yamileth
30/04/2025	3.10	<ul style="list-style-type: none"> Diagramas actualizados: clases, paquetes, estados. 	Manzur Rodríguez
02/05/2025	3.20	<ul style="list-style-type: none"> 5.9 Punto de vista de la estructura. 	Manzur Rodríguez
02/05/2025	3.30	<ul style="list-style-type: none"> 5.10 Punto de vista de la interacción. 	Cristela Adelaida
03/05/2025	3.40	<ul style="list-style-type: none"> Actualización del formato del documento. Corrección del apartado de 'versiones', con relaciones a las entregas realizadas. 	Manzur Rodríguez
04/05/2025	3.50	<ul style="list-style-type: none"> Apartado donde se indican los documentos y las versiones utilizadas para la elaboración del documento (Repositorio). 	Guadalupe Yamileth, Manzur Rodríguez, Cristela Adelaida
09/05/2025	3.60	<ul style="list-style-type: none"> 5.11 Punto de vista de dinámica de estados. 	Guadalupe Yamileth

INDICE

1. Introducción	1
2. Punto de vista contextual	2
2.1. Componentes	2
2.2. Diagrama	3
2.3. Caso de usos detallado	3
2.3.1 Comunidad -> Mostrar	3
2.3.2 SAR -> Comprobar	4
2.3.3 SAR -> Permitir / Denegar	5
2.3.4 SAR -> Leer	6
2.3.5 Vigilante -> Imprimir Acceso Temporal	7
2.3.6 Vigilante -> Registrar	8
2.3.7 Visitante -> Validación Acceso Temporal	9
2.4. Diagrama de Contexto	11
3. Punto de vista de la composición	11
3.1. Componentes	11
3.2. Diagrama de paquetes	12
3.3. Diagrama de componentes	14
3.4. IDEF0	15
3.4.1 Sistema actual	15
3.4.2 Nuevo sistema	15
3.5. Diagrama GANTT	16
3.5.1 Estimación de Tiempo	17
3.6. Entidad de diseño	17
3.6.1 Pratrón	17
3.6.2 Framework	17
3.7. Diagrama HIPO	17
4. Punto de vista lógico	18
4.1. Propósito	18
4.2. Problemas de diseño	18
4.3. Elementos de diseño	18
4.3.1 Entidades de diseño	18
4.3.2 Relaciones de diseño	19

4.3.3 Atributos de diseño.....	19
4.3.4 Restricciones de diseño.....	19
4.4. Ejemplos de idiomas.....	19
4.5. Diagrama de clases UML	19
4.6. Diagrama de objeto UML	20
5. Punto de vista de la dependencia.....	20
5.1. Problemas de diseño	21
5.1.1 Elementos de diseño	21
5.1.2 Atributos de dependencias.....	22
5.2. Ejemplos de Idiomas	22
6. Punto de vista de uso de patrones	23
6.1.1 Problemas de diseño	24
6.1.2 Elementos de diseño	24
6.1.3 Ejemplos de idiomas.....	24
6.2. Diagrama de Estructura Compuesta UML	24
7. Punto de vista de la interfaz.....	25
7.1. Descripción General	25
7.2. Atributo de interfaz.....	26
7.2.1 Interfaces Externas:	26
7.2.2 Interfaces Internas.....	26
7.2.3 Relaciones principales:.....	26
7.3. Ejemplos de idiomas.....	27
7.3.1 Vistas.....	27
7.3.2 Controladores	27
8. Punto de vista de la estructura	28
8.1. Consideraciones	28
8.2. Componentes.....	28
8.3. Diagrama de clases.....	28
8.3.1 Modelos.....	29
8.3.2 Service.....	29
8.3.3 Controller	29
8.4. Diagrama de estructura.....	30
8.4.1 Relación.....	30

9. Punto de vista de la interacción.....	31
9.1.1 Problemas de diseño	31
9.1.2 Elementos de diseño	31
9.2. Diagrama de secuencia UML	32
9.3. Diagrama de comunicación UML	32
10. Punto de vista de la dinámica del estado.....	33
10.1.1 Propósito.....	33
10.1.2 Problemas de diseño	33
10.1.3 Transformación dinámica del estado del QR.....	34
11. Definiciones, acrónimos y abreviaturas.....	35
12. Bibliografía	36

1. Introducción

Se define varios puntos de vista de diseño para su uso en SDD. Ilustra la realización de estos puntos de vista de diseño en términos de selecciones de lenguaje de diseño, relaciona las preocupaciones de diseño con los puntos de vista y establece nombres neutrales de lenguaje (notación y método) para estos puntos de vista.

En la **Tabla 1** se resumen estos puntos de vista de diseño. Para cada punto de vista, se enumeran su nombre, los problemas de diseño y los lenguajes de diseño adecuados. Para cada punto de vista se proporcionan descripciones breves que relacionan un conjunto mínimo de entidades de diseño, relaciones de diseño, atributos de entidad de diseño y restricciones de diseño. También se enumeran referencias adicionales pertinentes al uso de cada punto de vista

Resumen de los puntos de vista del diseño		
Punto de vista del diseño	Problemas de diseño	Ejemplos de lenguajes de diseño
Contexto (5.2)	Sistemas, servicios y usuarios	IDEF0, diagrama de casos de uso UML, Diagrama de contexto del análisis estructurado
Composición (5.3) Se puede refinar en nuevos puntos de vista, tales como: descomposición funcional (lógica) y descomposición en tiempo de ejecución (física)	Composición y montaje modular de sistemas en términos de subsistemas y componentes (enchufables), compra vs. construcción, reutilización de componentes	Lógico: Diagrama de paquetes UML, Diagrama de componentes UML, Lenguajes de descripción de arquitectura, IDEF0, Gráfico de estructura. HIPO Físico: Diagrama de implementación de UML
Lógica (5.4)	Estructura estática (clases, interfaces y sus relaciones) Reutilización de tipos e implementaciones (clases, tipos de datos)	Diagrama de clases UML, diagrama de objetos UML
Dependencia (5.5)	Interconexión, compartición y parametrización	Diagrama de paquetes UML y diagrama de componentes
Información (5.6) con superposición de distribución de datos y superposición volumétrica física	Información persistente	IDEF1X, diagrama entidad-relación, diagrama de clases UML


Patrones (5.7)	Reutilización de patrones y plantilla de marco disponible	Diagrama de estructura compuesta UML
Interfaz (5.8)	Definición de servicios, acceso a servicios	Lenguajes de definición de interfaz (IDL), diagrama de componentes UML
Estructura (5.9)	Componentes internos y organización de las materias, componentes y clases de diseño	Diagrama de estructura UML, diagrama de clases
Interacción (5.10)	Comunicación de objetos, mensajería	Diagrama de secuencia UML, diagrama de comunicación UML
Dinámica de estados (5.11)	Transformación dinámica del estado	Diagrama de la máquina de estado UML, diagrama de estado (Harel), tabla de transición de estado (matriz), autómatas, red de Petri
Algoritmo (5.12)	Lógica procedimental	Tabla de decisión, diagrama de Warnier, JSP, PDL
Recursos (5.13) Se puede refinar en puntos de vista basados en recursos con posibles superposiciones	Utilización de recursos	Perfil en tiempo real UML, diagrama de clases UML, lenguaje de restricción de objetos UML (OCL)

Tabla 1

2. Punto de vista contextual

Representa los servicios prestados por un sujeto de diseño con referencia a un contexto explícito. Ese contexto se define por referencia a los actores que incluyen a los usuarios y otras partes interesadas, que interactúan con el sujeto del diseño en su entorno. El punto de vista Contexto proporciona una perspectiva de "caja negra" sobre el tema del diseño.

2.1. Componentes

Información de símbolos	
	Representa un actor.

		Representación de un sistema.
--	--	-------------------------------

Tabla 2

2.2. Diagrama

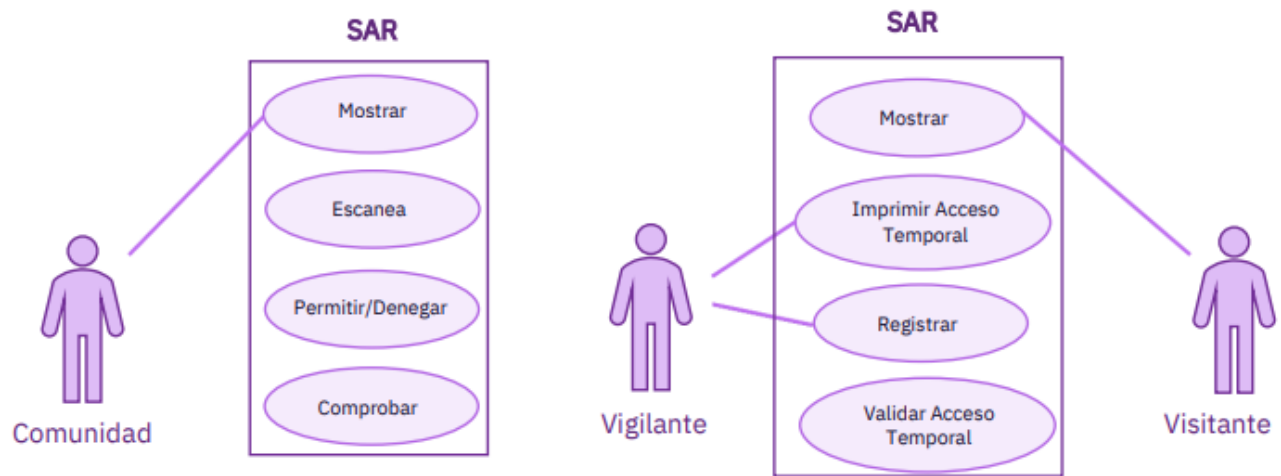


Diagrama de casos de uso

Guadalupe Yamileth

2 horas

Diagrama 1

2.3. Caso de usos detallado.

2.3.1 Comunidad -> Mostrar

Nombre de caso de Uso:	Comunidad -> Mostrar
Actor principal:	Comunidad
Precondiciones:	<ul style="list-style-type: none"> La comunidad debe mostrar su código QR (ubicado en su credencial) al escáner QR. Si no está disponible el escáner instalado fijamente en la entrada, el vigilante realizará la lectura del QR, con un escáner portátil.
Postcondiciones o Garantías de Éxito:	<ul style="list-style-type: none"> El código QR debe pertenecer a la UACM. El código QR debe ser visible completamente.

	<ul style="list-style-type: none"> La credencial debe encontrarse en buenas condiciones, para visualizar la información.
Escenario Principal:	Acceso 1
Excepciones o Flujos Alternativos:	<ul style="list-style-type: none"> El usuario debe portar su credencial de identificación perteneciente a la UACM. Constancia de inscripción proporcionada por la UACM, el cual contenga su código QR.
Requisitos especiales:	El código QR debe ser visible para el escáner.
Frecuencia:	Alta, la comunidad mostrará su código QR cada vez que dese acceder al plantel.
Temas Abiertos:	<ul style="list-style-type: none"> Se mostrará la información más relevante del estudiante (nombre, carrera y una foto del mismo para corroborar que es la persona que dice ser. En caso de que el QR no pertenezca a un alumno, aparecerá un mensaje de error

2.3.2 SAR -> Comprobar

	Comprobar
Nombre de caso de Uso:	SAR -> Comprobar
Actor principal:	Comunidad
Precondiciones:	<ol style="list-style-type: none"> Un usuario, ya mostró su código QR El escáner extrae el identificador dentro de la URL. Con las características del identificador, se determinara que DB realizara la comprobación Retornar una respuesta.
Postcondiciones o Garantías de Éxito:	<ul style="list-style-type: none"> La información del QR debe ser enviada en texto plano. La información del QR no excederá un tamaño de 250 caracteres. El área de sistemas nos indicará que tablas de la DB, contienen la información necesaria para determinar si el usuario que desea entrar pertenece a la comunidad.

	<ul style="list-style-type: none"> El área de sistemas nos indicará que tablas de la DB, contienen la información de los visitantes, donde se almacena la duración de los QR. Contar con un usuario con permisos, para realizar las consultas a la DB.
Escenario Principal:	<ul style="list-style-type: none"> Funcionamiento dentro de la clase “Usuario”. La cual compruebe si la información enviada por el lector cumple ciertas condiciones, las cuales nos permitirán determinar si se realizó una lectura del código QR de forma correcta.
Excepciones o Flujos Alternativos:	<ul style="list-style-type: none"> Si el QR, no pertenece a la UACM, denegara el acceso. Si la información recibida no cumple cierta condición, indicara que es necesario volver a leer el QR.
Requisitos especiales:	El QR guarna una dirección web, la cual contiene la matrícula del usuario.
Frecuencia:	<ul style="list-style-type: none"> Alta, se van a realizarán un número elevado de consultas por día. En ciertas horas, aumenta el número de consultas.
Temas Abiertos:	<ul style="list-style-type: none"> Como se retornará el resultado de la consulta, la cual determina si pertenece a la UACM. Que pasa, si la DB, no se encuentra disponible o está saturada.

2.3.3 SAR -> Permitir / Denegar

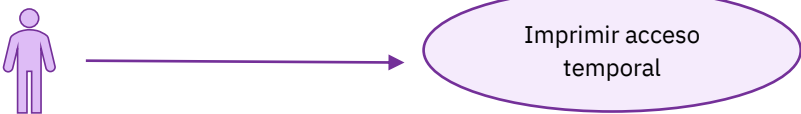
Permitir/Denegar	
Nombre de caso de Uso:	SAR -> Permitir / Denegar
Actor principal:	SAR
Precondiciones:	Respuesta de la consulta realizada a la DB.
Post condiciones o Garantías de Éxito:	<ul style="list-style-type: none"> Un usuario de la comunidad deberá estar activo en el sistema para que se le pueda permitir el acceso, de lo contrario se denegará la entrada. Si es un visitante, su QR debe encontrarse activo.
Escenario Principal:	Vista utilizada en la entrada principal.

Excepciones o Flujos Alternativos:	Si la respuesta de la consulta presenta algún error, indicara al vigilante “Sistema de Estudiante temporalmente inactivo”.
Requisitos especiales:	Como se mostrará el mensaje para permitir o denegar el acceso.
Frecuencia:	Alta, se van a realizarán un número elevado de consultas por día.
Temas Abiertos:	<ul style="list-style-type: none"> • Manera en que se mostrara el mensaje. • Si el mensaje, desaparecerá después de un tiempo. • Mientras se encuentre el mensaje activo, no se podrá leer otro QR.


2.3.4 SAR -> Leer

<div>Leer</div>	
Nombre de caso de Uso:	SAR -> Leer
Actor principal:	Escáner.
Precondiciones:	<ul style="list-style-type: none"> • QR legible. • Extraer la dirección web guardada en el QR de manera correcta.
Post condiciones o Garantías de Éxito:	<ul style="list-style-type: none"> • Que las entradas, cuenten con un lector QR portátil.
Escenario Principal:	Acceso 1.
Excepciones o Flujos Alternativos:	En caso de un error al leer el escáner, se deberá intentar de nuevo.
Requisitos especiales:	Conexión estable con el escáner QR.
Frecuencia:	Alto, van a leerse varios QR al día, en especial en los principales horarios de entrada.
Temas Abiertos:	Si el escáner utilizado, requiere un driver o API específico para su funcionamiento, o funciona al conectarlo a una computadora.


2.3.5 Vigilante -> Imprimir Acceso Temporal

	
Nombre de caso de Uso:	Vigilante -> Imprimir Acceso Temporal
Actor principal:	Vigilante
Precondiciones:	<ol style="list-style-type: none"> 1. Haber registrado al visitante. 2. El registro, debe tener un identificador único. 3. Que dicho registro, tenga un código QR asignado. 4. El registro, se encuentre guardado en la DB. <ul style="list-style-type: none"> • Acceso a la DB. • Vista única, para la opción de impresión. Solo se imprimirán los que no sobrepasen las 4 horas de haberse registrado.
Post condiciones o Garantías de Éxito:	<ul style="list-style-type: none"> • Registro guardado en la DB. • QR asignado al registro. • Contar con una impresora. • Contar con un escáner, para la identificación única del usuario.
Escenario Principal:	Vista única para impresión.
Excepciones o Flujos Alternativos:	Posible error con la impresora. En dicha condición, el personal de sistema o mantenimiento intervendrá.
Requisitos especiales:	<ul style="list-style-type: none"> • SI el QR ya venció, no permitir la impresión. Dentro de la UI, el registro ya no tendrá opción para imprimir. • En la DB, almacenara cuanto tiempo dura el código QR. • Después de las 7:00 p.m., la opción se deshabilitará, y se prohibirá cualquier acceso a visitantes.
Frecuencia:	Medio-Bajo. Varía en cuestión de la actividad del plantel.
Temas Abiertos:	Se en la impresión, se imprimirá todos los datos del registro, los cuales se imprimirá en una hoja completa (tamaño carta).

2.3.6 Vigilante -> Registrar

	
Nombre de caso de Uso:	Vigilante -> Registrar
Actor principal:	Vigilante
Precondiciones:	<ul style="list-style-type: none"> • Llenar todos los campos obligatorios. • Tener una cita previamente agendada. • Si no tiene una cita, dentro del apartado motivo, se especificará el motivo de la visita. • El visitante, obligatoriamente debe contar con una identificación oficial (INE).
Post condiciones o Garantías de Éxito:	<p>Los campos siguientes se consideran obligatorios:</p> <ul style="list-style-type: none"> • Nombre • Motivo • Identificación oficial (puede haber excepciones).
Escenario Principal:	Acceso 1
Excepciones o Flujos Alternativos:	Si un usuario de la comunidad, olvido su credencial, este proporcionará su matrícula, se comprobará si existe y pertenece al usuario. De ser correcto, se le permitirá el acceso, y se guardará en el registro de visitantes.
Requisitos especiales:	<ul style="list-style-type: none"> • Contar con una identificación oficial (INE), por parte del visitante, pueden existir casos especiales, en la cuales no sea requerida. • EL sistema, debe tener habilitada la opción para registrar. La opción se habilitará a las 6:59 a.m. • Después de las 7:00 p.m., la opción se deshabilitará, y se prohibirá cualquier acceso a visitantes.
Frecuencia:	Medio-Bajo. Varía en cuestión de la actividad del plantel.
Temas Abiertos:	Si el usuario pertenece a la comunidad, y este olvido su credencial, que información se llenaría en "Motivo".

2.3.7 Visitante -> Validación Acceso Temporal

	
Nombre de caso de Uso:	Visitante -> Validación Acceso Temporal
Actor principal:	Visitante
Precondiciones:	<ul style="list-style-type: none"> El visitante debe estar registrado, y su QR debe encontrarse activo. El visitante debe estar registrado y disponible durante el período de tiempo designado. El sistema valida, si el QR está activo.
Post condiciones o Garantías de Éxito:	<ul style="list-style-type: none"> El vigilante obtiene acceso al registro del visitante. El sistema valida que el acceso esté habilitado durante la fecha y hora programadas. El acceso del visitante, se encuentra registrado en la DB. El sistema verifica que el acceso solicitado esté habilitado en ese momento (puede ser un evento, una reunión, acceso a un edificio, uso de un servicio, etc.).
Escenario Principal:	Acceso 1
Excepciones o Flujos Alternativos:	<ul style="list-style-type: none"> E1: La identificación del visitante (código QR) no es válida. El sistema muestra un mensaje de error y solicita un nuevo escaneo o ingreso de datos. E2: El acceso solicitado no está habilitado en la fecha o la hora actuales. El sistema informa al visitante que el acceso no es posible debido a que no está disponible. E3: El visitante no está registrado. El sistema niega el acceso e informa al vigilante sobre el error de registro.

	<ul style="list-style-type: none">• E4: El visitante intenta acceder fuera del horario permitido. El sistema informa que el acceso está restringido fuera del horario autorizado.
Requisitos especiales:	<ul style="list-style-type: none">• Acceso a una red para validar el estado de acceso (en tiempo real si es necesario).
Frecuencia:	Medio-Bajo. Varía en cuestión de la actividad del plantel.
Temas Abiertos:	<ul style="list-style-type: none">• Consideraciones de seguridad adicionales para garantizar que no se usen medios de identificación fraudulentos.• Posibles problemas de validación por falta de conexión o fallas en el sistema.

CONFIDENTIAL

2.4. Diagrama de Contexto

Diagrama de contexto, para el Sistema de Acceso Rápido (SAR), sistema el cual se va a desarrollar.



Diagrama de contexto general

Guadalupe Yamileth

1 hora

Diagrama 2

3. Punto de vista de la composición

Describe la forma en que el sujeto de diseño se estructura (recursivamente) en partes constituyentes y establece los roles de esas partes.

3.1. Componentes

Información de símbolos

	Carpeta pública del sistema, se encuentran las imágenes, estilos y archivos necesarios para el funcionamiento correcto de plugin.
	Carpeta de 'Services', contiene clases específicas que no requieren de Modelos, Acceso a base de datos o controladores para funcionar.
	Carpeta la cual contiene todos los controladores.
	Carpeta la cual contiene los modelos.
	Carpeta que contiene todas las vistas.
	Representa la carpeta donde se encuentran las estructuras de las tablas para DB.
	Representación de una vista.

Tabla 3

3.2. Diagrama de paquetes

Mediante el diagrama de paquetes UML, se detallará el funcionamiento de sistema. El diagrama nos permite ver:

- Funcionamiento del sistema.

- Interfaces gráficas.
- Modelos y controladores.

Como el sistema no tiene acceso completo a la información almacenada en la (DB), se tomó la decisión de representarlo cómo un sistema aparte.

El diagrama de paquetes seguirá la arquitectura Modelo-Vista-Controlador.

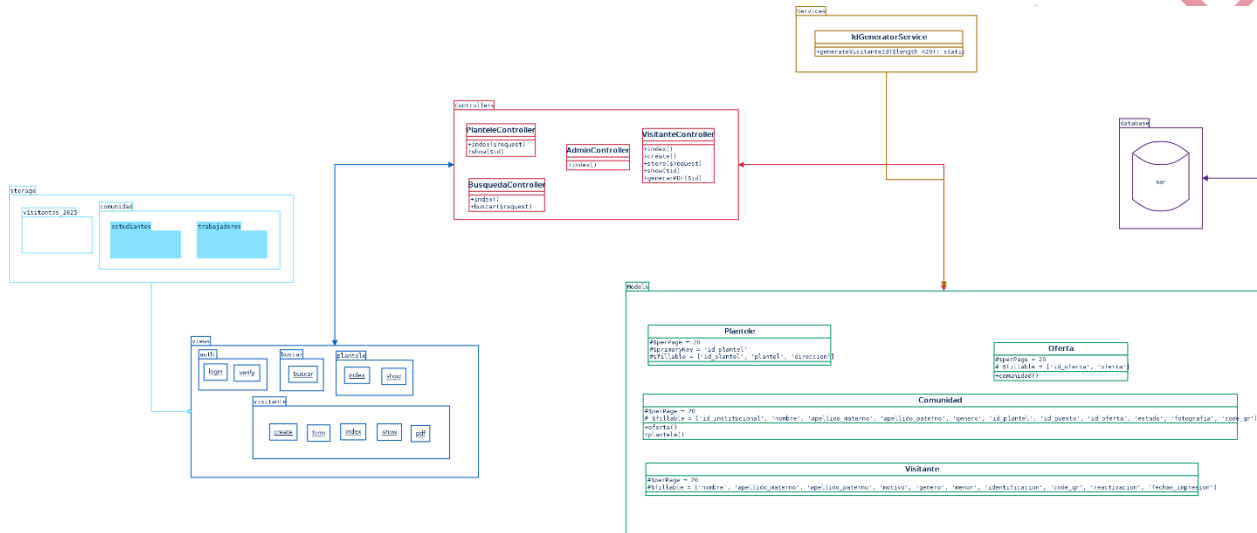


Diagrama general de paquetes

Manzur Rodríguez
Diagrama 3

1.5 horas

Información de las carpetas:

- **Database:** Dentro de esta carpeta, contiene subcarpetas, las cuales nos ayudan a establecer la estructura (*migrations*) de la base de datos, plantar las semillas de información (*seeders*) en las tablas DB y la generadora de semillas (*factories*). Esto nos ayuda a llenar con información de prueba la base de datos, para realizar las pruebas de funcionalidad.
- **Services:** Contiene el servicio para generar 'Id' compuestos por letras mayúsculas, minúsculas y números. Este servicio es llamado por el controlador '*VisitanteController*'.
- **Controllers:** Controladores del sistema, estos controlan de que manera tiene que mostrarse las vistas y que información vas a mostrar o solicitar al usuario.
- **Models:** Los modelos son llamados por los controladores, los modelos son los que se conectan a la base de datos del sistema.
- **Views:** Dependiendo del controlador, una o varias vistas son mostradas en el sistema.
- **Storage:** Almacena los estilos del sistema, archivos de los Plugins, es donde se almacenan las identificaciones de los visitantes y las fotografías de la comunidad. Los QR también se almacenan en dichas carpetas.

3.3. Diagrama de componentes

El paquete ‘controller’, hace uso de 1 o varios modelos. En el diagrama, mencionamos todos los componentes que existen en cada paquete, los 3 paquetes, corresponden al MVC.

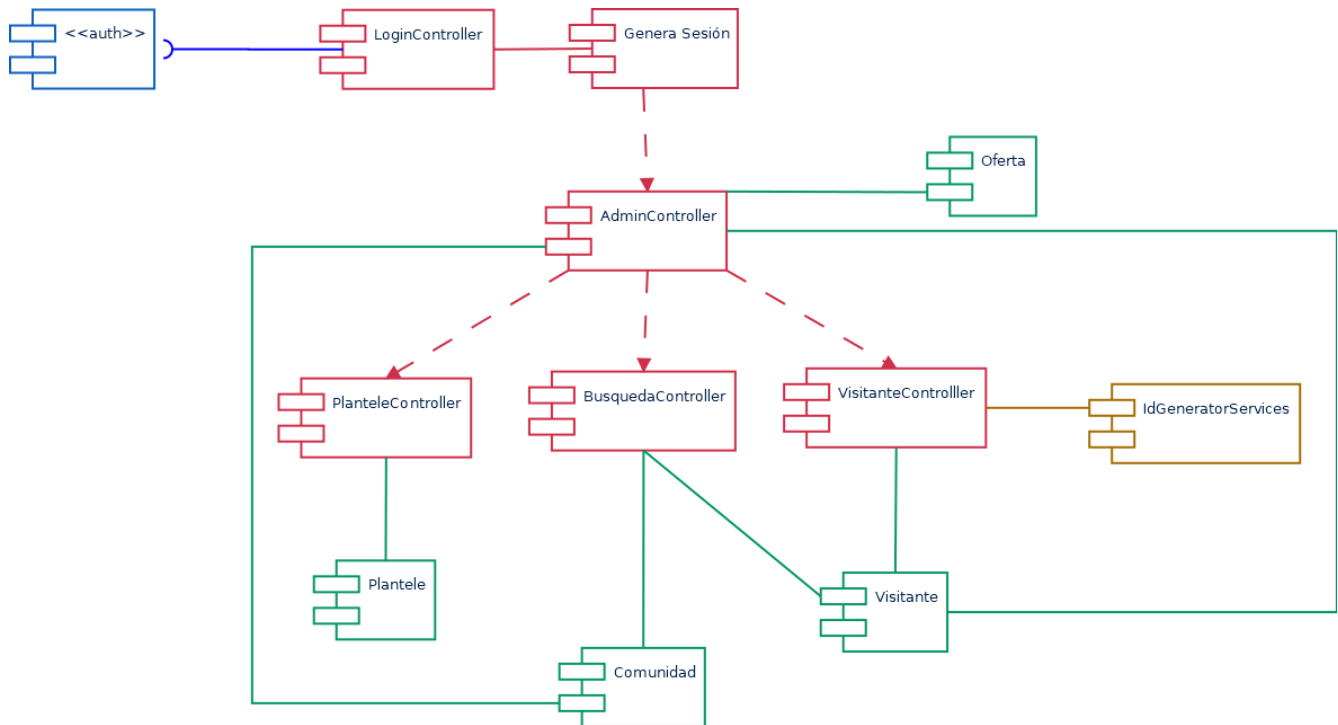


Diagrama de componentes

Manzur Rodriguez

2 horas

Diagrama 4

- **Auth:** Vista en la cual el usuario tiene que ingresar ‘correo electrónico’ y ‘contraseña’ correctas para iniciar sesión en el sistema.
- **LoginController:** Controlador que verifica si los datos ingresados en el formulario de iniciar sesión son correctos, comparándolos con los almacenados en la base de datos.
- **Genera Sesión:** Una vez que se comprobaron que los datos son correctos, el sistema genera una sesión en el sistema. Esta es necesaria para visualizar todas las vistas.
- **AdminController:** Vista principal del sistema, esta muestra gráfica de los visitantes que accedieron en el día, además permite navegar por las funcionalidades del sistema.
- **Oferta:** Modelo que se conecta con la base de datos, para retornar la información de la oferta académica que cuenta la UACM.
- **PlanteleController:** Envía a la vista la información obtenida de la DB.
- **Plantele:** Lee la información almacenada en la DB de los planteles y sedes de la AUCM.

- **BusquedaController:** Formulario que trabaja en conjunto con el escáner de QR. Para consultar si el identificador almacenado en el QR es válido, el sistema lo comprueba en los modelos ‘Comunidad’ y ‘Visitante’.
- **VisitanteController:** Retorna a la vista la lista de visitantes registrados el mismo día de utilización, permite generar un QR con información específica, el cual se le entrega al visitante. Recibe la información ingresada de un formulario (encontrada en su vista *create*) y la procesa para su almacenamiento en la DB. Además, utiliza el servicio ‘IdGeneratorService’ para generar un identificador de longitud 20.
- **Visitante:** Utilizado para leer y registrar información en la DB.

3.4. IDEF0

Se describe de forma general el proceso de entrada de un usuario (Comunidad o Visitante) al plantel de la UAMC. La entrada está regulada por los vigilantes de seguridad (*personal humano*), quienes verificarán las credenciales de manera manual.

3.4.1 Sistema actual

El sistema que sigue actualmente la universidad, para permitir el acceso a estudiantes, personal administrativo, docentes, trabajadores al plantel, se basa en el uso de la credencial, que previamente se les proporciona, como identificación oficial por parte de la UACM.



Diagrama IDEF0

Manzur Rodriguez

2 horas

Diagrama 5

3.4.2 Nuevo sistema

A pesar que el sistema actual es funcional, presenta cuestiones importantes en el tema de seguridad esto, a que no se comprueba la información de dichas credenciales, por parte de los vigilantes.

El sistema (SAR), busca mejorar la seguridad, en el acceso al plantel, y al mismo tiempo, ayudar a que los vigilantes, tengan mejores herramientas, para dejar entrar a un usuario al plantel.

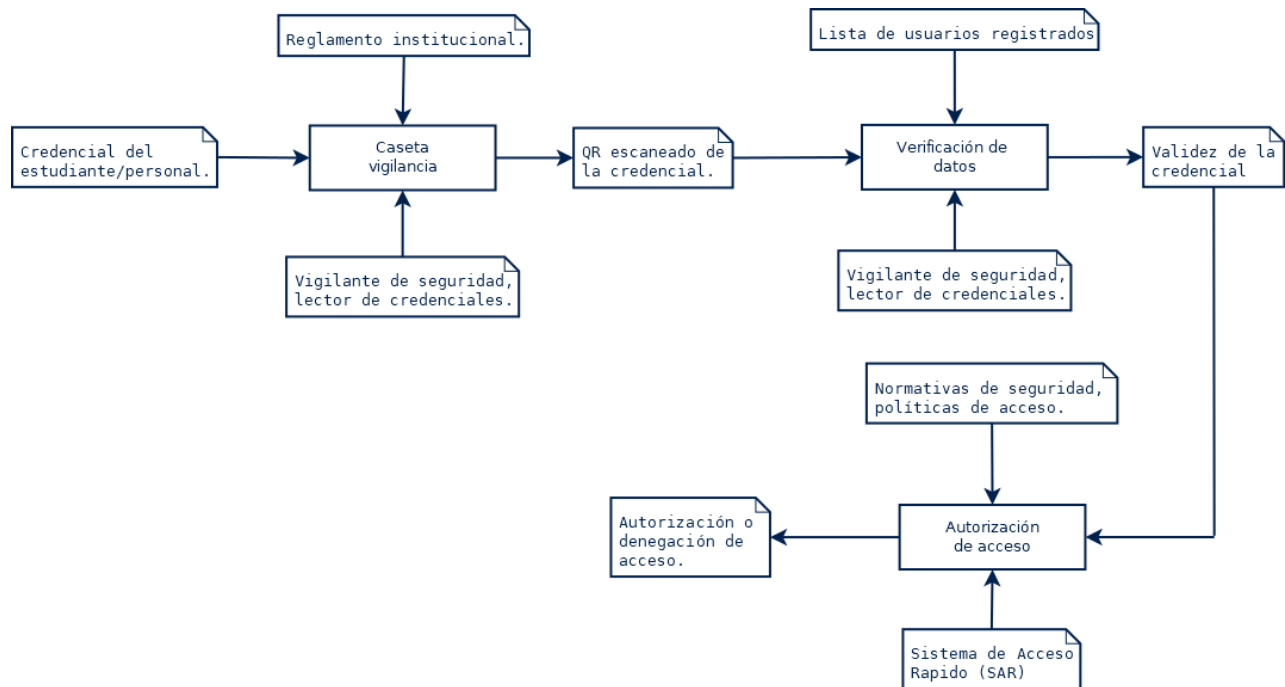


Diagrama IDEF0 SAR

Manzur Rodríguez

2 horas

Diagrama 6

3.5. Diagrama GANTT

Para realizar la planificación, supervisar su evolución, retroalimentaciones o actualizaciones por parte del cliente o gerente del proyecto, se utilizará la herramienta ‘GanttProjet’.

Utilizaremos la versión gratuita de la herramienta, por lo tanto, las actualizaciones o correcciones, se realizarán en las reuniones. Con el objetivo, que todo el equipo de desarrollo tenga conocimiento de alguna actualización o modificación de los requisitos.

Esta herramienta nos permitirá estimar el costo, el personal que será asignado y el cronograma para el desarrollo.

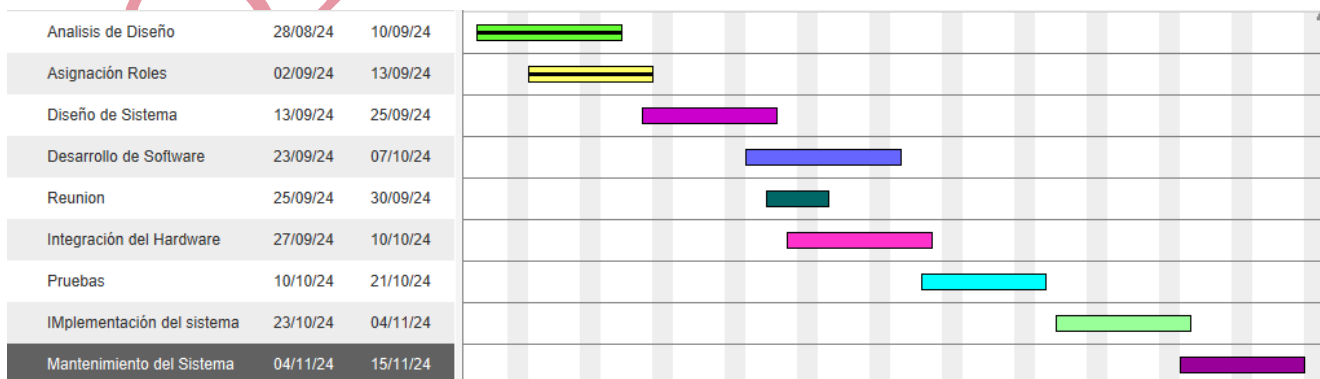


Diagrama de Gantt 01

Manzur Rodríguez

3 horas

Diagrama 7

Las etapas de desarrollo se asignará el trabajo de manera equitativa, el tiempo estimado de desarrollo, que tiene el equipo:

3.5.1 Estimación de Tiempo

- **Análisis de Requerimientos:** Duración estimada de 2 semanas.
- **Diseño del Sistema:** Duración estimada de 3 semanas.
- **Desarrollo:** Duración estimada de 8 semanas.
- **Pruebas:** Duración estimada de 4 semanas.
- **Implementación:** Duración estimada de 2 semanas.
- **Soporte Post-Implementación:** Duración estimada de 2 semanas.

Utilizando la aplicación antes mencionada, establecimos los roles del equipo de desarrollo, las fechas límite para la codificación. Con esto, podemos determinar de manera visual si un desarrollador tiene sobrecarga de trabajo.

3.6. Entidad de diseño

3.6.1 Patrón

El proyecto será diseñado utilizando el patrón de desarrollo Modelo – Vista – Controlador (MVC). Se escogió dicho patrón, con el objetivo de facilitar las actualizaciones, o mejoras posteriores del sistema.

3.6.2 Framework

Se utilizará el Framework 'Laravel', en su versión 11.47

3.7. Diagrama HIPO

Usando el diagrama HIPO, modelaremos las acciones a realizar por parte de (SAR). Derivado del análisis estructurado y la técnica de diseño (SADT). Este método nos ayudara a realizar el análisis del sistema y a promover una buena comunicación entre el análisis previo del sistema, por parte de los desarrolladores y el cliente.

Utilizaos los rectángulos y líneas para representar procesos, funciones, trabajos o tareas, así como las conexiones existentes entre las funciones y el entorno externo.

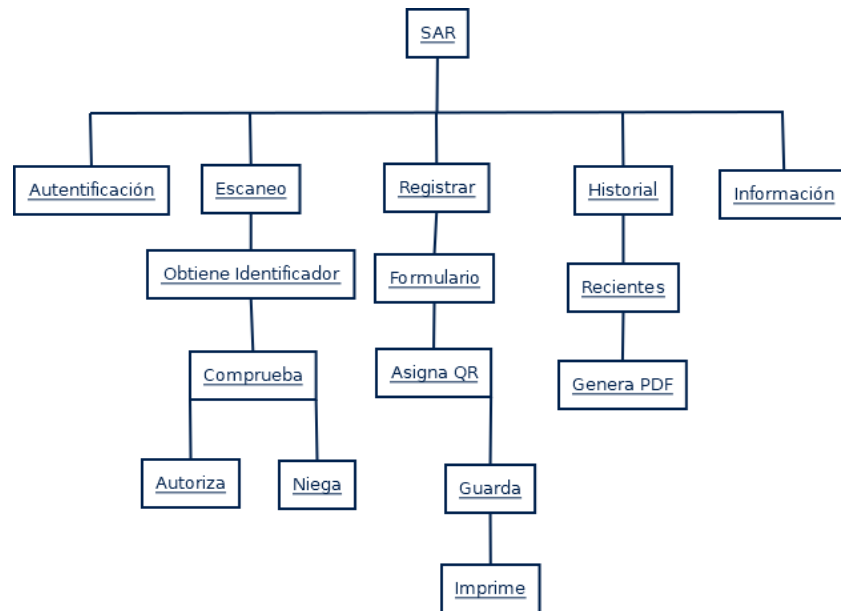


Diagrama HIPO

Manzur Rodríguez

1.5 Horas

Diagrama 8

4. Punto de vista lógico

4.1. Propósito

El propósito del punto de vista lógico en este diagrama es identificar los objetos dentro del sistema de control de acceso mediante QR y sus relaciones estáticas. Se diseñan clases y objetos que interactúan para representar la autenticación de usuarios y visitantes mediante códigos QR.

4.2. Problemas de diseño

El diagrama de objeto, los principales problemas de diseño abordados incluyen:

- Identificación de entidades claves: Se define objetos como **Usuario**, **Visitante**, **Login**, **Sistema**, **GeneradorQR**, **QRcode**, **Lector** y **Escaneable**.
- Reutilización de abstracciones: Se presentan objetos que pueden ser utilizados en distintas instancias del sistema, como **Login** para diferentes usuarios o **QRCode** generado para cada visitante.
- Interacción entre los elementos: Se requiere definir cómo se comunican los objetos entre sí, como la relación entre el **GeneradorQR**, el **QRCode** y el **Lector**.

4.3. Elementos de diseño

4.3.1 Entidades de diseño

- Clases y objetos: Se identifican clases implícitas en el diagrama como **Usuario**, **Visitante**, **Login**, **Sistema**, **GeneradorQR**, **QRCode**, **Lector** y **Escaneable**.

- **Atributos:** Algunos atributos incluyen usuario y contraseña en Login, código en QRCode y motivo en Visitante.
- **Métodos:** Aunque no se especifican explícitamente, el GeneradorQR tiene un método para generar códigos QR y Lector uno para escanearlos.

4.3.2 Relaciones de diseño

- **Asociación:** Tiene una relación entre Usuario y Sistema, indicando que el sistema maneja datos del usuario.
- **Generalización:** Escaneable es una clase abstracta que puede representar cualquier objeto escaneable, como un código QR.
- **Dependencia:** Lector depende de Escaneable, lo que indica que Lector no tiene sentido sin un objeto que pueda escanear.

4.3.3 Atributos de diseño

- **Nombre y rol:** Cada objeto tiene un identificador y atributos relevantes.
- **Cardinalidad:** Se infiere que puede haber múltiples Usuarios, Visitantes y QR Codes, pero el Sistema actúa como un contenedor único.

4.3.4 Restricciones de diseño

- **Multiplicidad:** Un Usuario visitante puede autenticarse varias veces, y no generar múltiples QRCode.
- **Navegabilidad:** La relación entre Lector y QRCode sugiere una dirección clara: Lector escanea QRCode, pero no al revés.

4.4. Ejemplos de idiomas

- **Diagrama de objetos UML:** Representa instancias específicas de las clases, como Usuario con matrícula 20231025 y Visitante con motivo = 'Reunión'.
- **Diagramas de clases UML:** Se utilizan para representar la estructura general, definiendo Usuario y Visitante como subclases de una clase Persona, por ejemplo.

4.5. Diagrama de clases UML

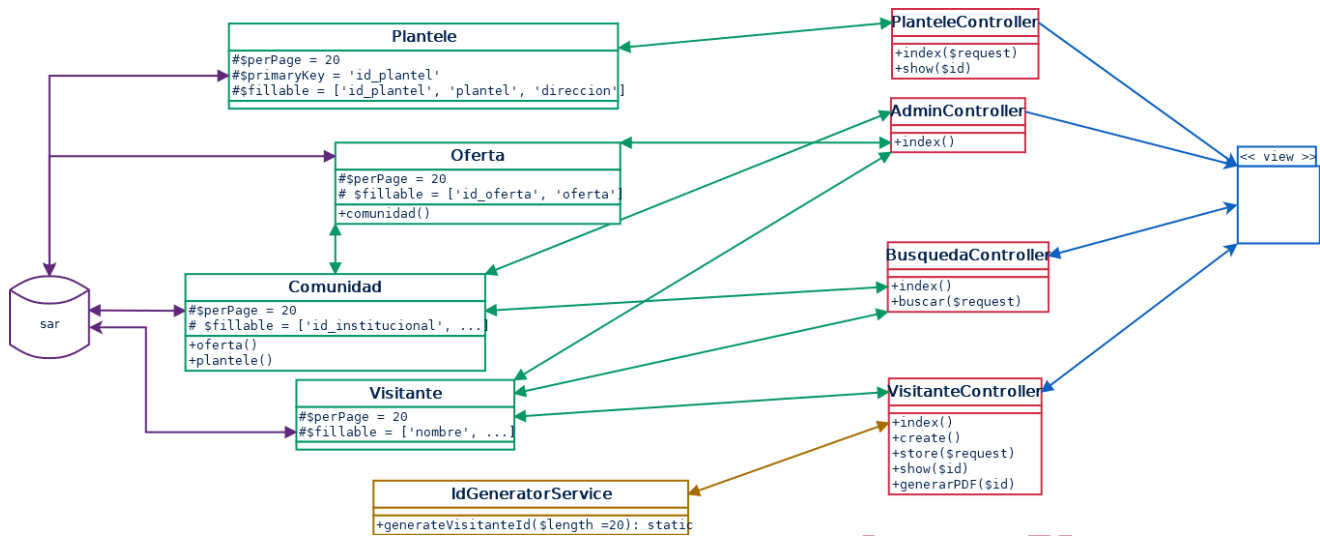


Diagrama clase

Kevin Manzur
Diagrama 9

1.5 Horas

4.6. Diagrama de objeto UML

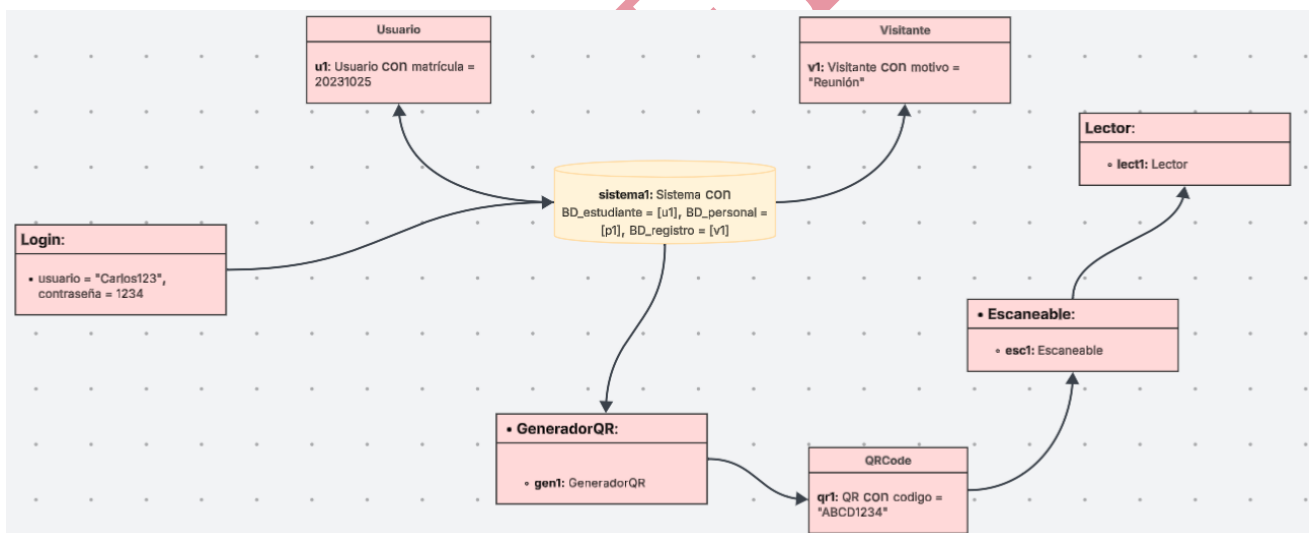


Diagrama objeto

Cristela Cruz Ovando
Diagrama 10

3.5 Horas

5. Punto de vista de la dependencia.

La vista de dependencias tiene como objetivo describir las relaciones entre los diferentes elementos del sistema, mostrando cómo interactúan y dependen unos de otros. En este apartado, se detallan las conexiones entre los principales componentes del sistema de acceso a la UACM mediante QR, incluyendo módulos de autenticación, gestión de usuarios y validación de accesos.

Esta vista es esencial para comprender el nivel de acoplamiento entre los elementos del software, identificar posibles mejoras en la arquitectura y garantizar que el diseño del sistema sea flexible y mantenible. Para ello, se presentan los diagramas de componentes y paquetes, los cuales ilustran las interacciones y agrupaciones lógicas dentro del sistema.

5.1. Problemas de diseño

En el diseño del sistema de acceso a la UACM mediante QR, se han identificado los siguientes problemas de diseño:

- › **Acoplamiento entre componentes:** La interdependencia entre controladores y la base de datos puede generar problemas de mantenimiento y escalabilidad.
- › **Gestión de autenticación:** La seguridad en la comunicación entre el componente de Login y la base de datos requiere especial atención para evitar vulnerabilidades.
- › **Manejo de sesiones:** La conexión entre los componentes Home y QR debe garantizar que los accesos sean seguros y eficientes.

5.1.1 Elementos de diseño

Los principales elementos de diseño incluyen:

- › **Componentes del sistema:** Representados en el diagrama de componentes, incluyen los módulos principales como Admin, Login, Búsqueda, Visitante y los controladores correspondientes.
- › **Relaciones entre componentes:** Definen cómo interactúan los diferentes elementos del sistema, incluyendo la base de datos como punto central de la gestión de información.
- › **Estructura modular:** Utilizando Laravel, los controladores se encargan de la lógica de negocio y se comunican con la base de datos mediante modelos.

Diagrama de Componentes

El diagrama de componentes muestra la estructura del sistema, detallando los módulos principales y su interacción con los controladores y la base de datos.

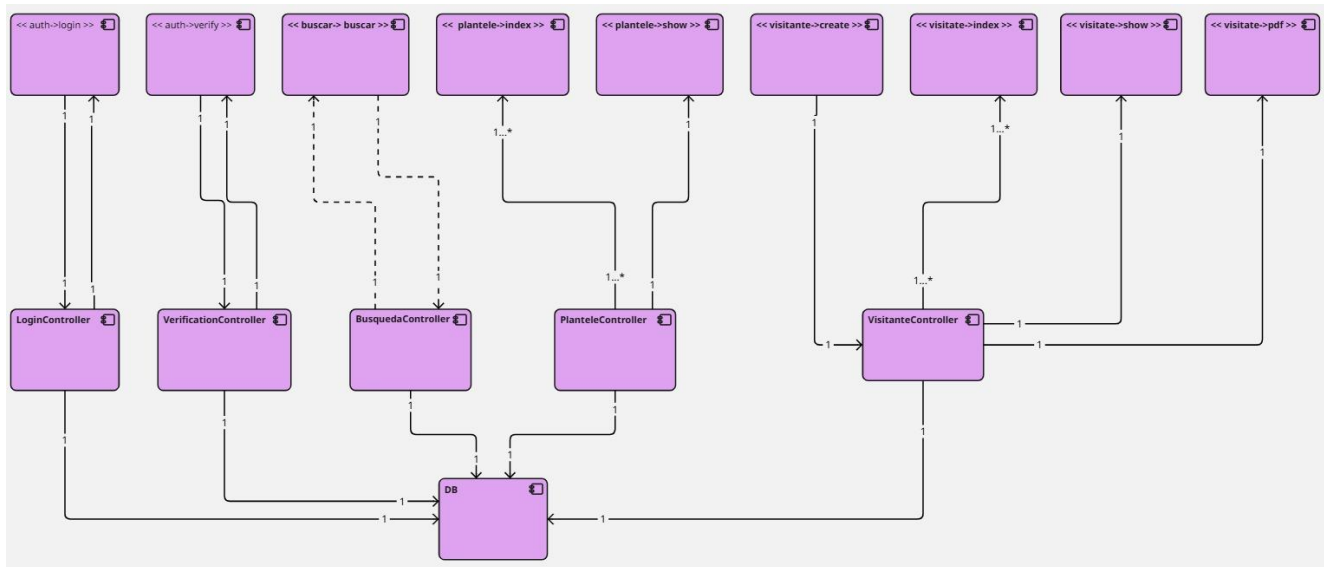


Diagrama de componentes

Guadalupe Yamileth

1.5 Horas

Diagrama 11

5.1.2 Atributos de dependencias

Las dependencias dentro del sistema están determinadas por:

- › **Tipo de dependencia:** Comunicación entre controladores y base de datos, dependencias entre módulos de usuario y autenticación.
- › **Direccionalidad:** Se identifican relaciones unidireccionales y bidireccionales según la interacción de los componentes.
- › **Grado de acoplamiento:** Se busca minimizar dependencias innecesarias para mejorar la escalabilidad del sistema.

5.2. Ejemplos de Idiomas

El sistema está desarrollado en PHP, con Laravel como framework. Se utilizan estándares de desarrollo web, incluyendo HTML, CSS y JavaScript para la interfaz de usuario.

Diagrama de Paquetes

El diagrama de paquetes organiza los componentes en módulos lógicos, reflejando la estructura del sistema y facilitando su mantenimiento.

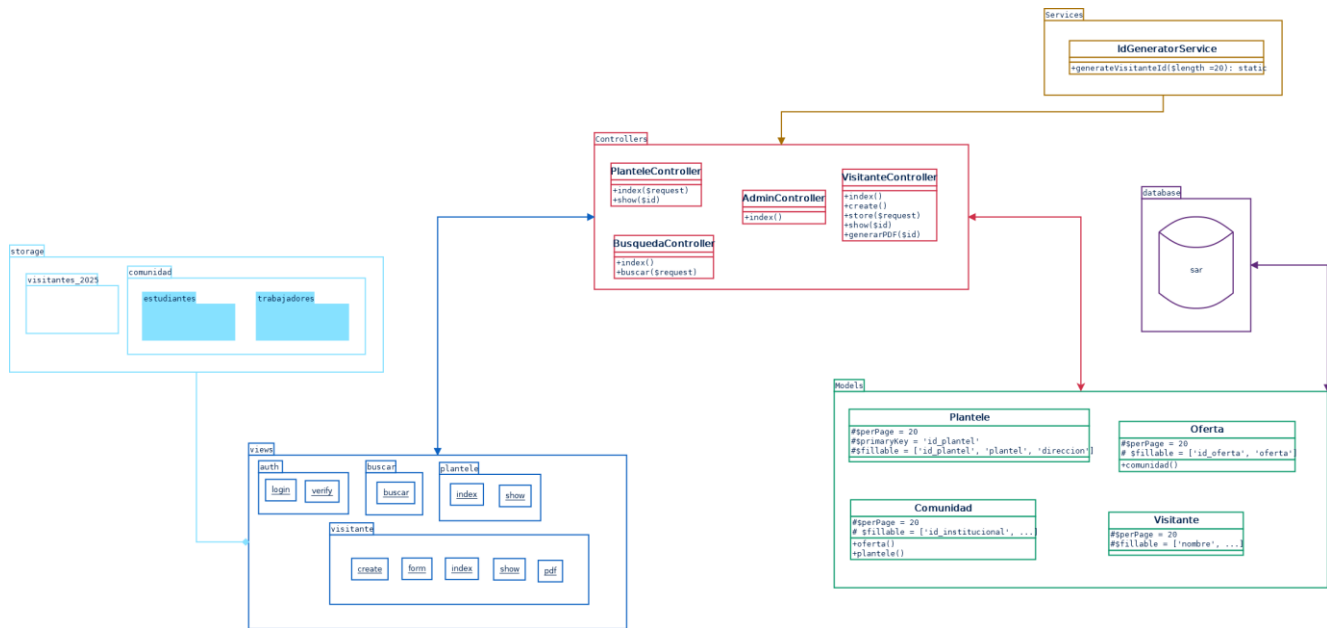


Diagrama general de paquetes

Manzur Rodriguez

3 horas

Diagrama 12

6. Punto de vista de uso de patrones

El patrón **Modelo-Vista-Controlador (MVC)** permite organizar el desarrollo de software separando la lógica de datos, la interfaz y el control del sistema. En el proyecto actual de **SAR**, este patrón mejora la claridad y el mantenimiento del código.

Laravel, un framework basado en MVC, facilita este tipo de desarrollo al ofrecer herramientas para enrutamiento, autenticación y manejo de base de datos. Usar Laravel con MVC permite construir un sistema eficiente, seguro y escalable para controlar el acceso de estudiantes, personal y visitantes mediante escaneo de códigos QR.

Laravel **separa y organiza** claramente:

- **Modelo** → Datos.
- **Vista** → Interfaz.
- **Controlador** → Lógica.

Todo esto **automatiza el patrón MVC** para que solo nos enfoquemos en la lógica del proyecto.

6.1.1 Problemas de diseño

Al desarrollar el sistema **SAR**, pueden surgir problemas clave como la necesidad de **reutilizar ideas de diseño**, aplicar **estilos arquitectónicos** adecuados y aprovechar **plantillas de frameworks**. El patrón **MVC**, junto con Laravel, ayuda a organizar el código, facilita el mantenimiento y mejora la eficiencia del desarrollo al resolver estos problemas de forma estructurada.

6.1.2 Elementos de diseño

En el sistema **SAR** usando MVC y Laravel:

- **Entidades de diseño:** incluyen clases, roles, conectores (como rutas), y el uso de Laravel como plantilla de marco.
- **Relaciones de diseño:** se reflejan en asociaciones entre clases y la colaboración entre vista, controlador y modelo.
- **Atributos de diseño:** como los nombres claros de clases y funciones.
- **Restricciones de diseño:** se aplican para que la vista no acceda directamente al modelo, respetando la estructura MVC.

6.1.3 Ejemplos de idiomas

Se utilizan **idiomas visuales** como los diagramas UML para representar la estructura del sistema.

- **Diagrama de estructura compuesta UML:** muestra cómo los componentes internos (modelo, vista, controlador) se relacionan dentro del sistema.

6.2. Diagrama de Estructura Compuesta UML

El diagrama de estructura compuesta muestra cómo interactúan los componentes internos dentro del patrón MVC.

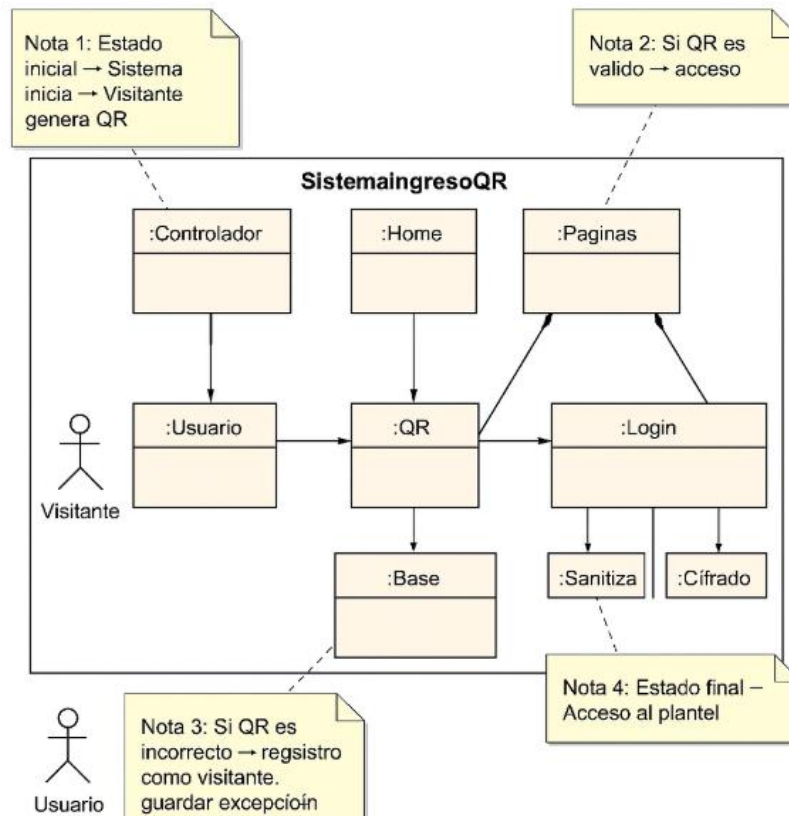


Diagrama de estructura compuesta UML

Cristela Cruz Ovando

3.5 Horas

Diagrama 13

7. Punto de vista de la interfaz

El propósito de este punto de vista es describir las interfaces externas y las interacciones entre los componentes principales del sistema SAR (Sistema de Acceso de la UACM), enfocándose en las conexiones con sistemas externos y en las relaciones internas entre sus módulos de software.

7.1. Descripción General

El sistema SAR utiliza una arquitectura basada en el patrón MVC (Modelo-Vista-Controlador), donde:

- › Las vistas representan la capa de presentación para los usuarios (Login, Home, QR, Usuario y Visitante).
- › Los controladores gestionan la lógica de negocio asociada a cada vista.
- › El sistema interactúa con una Base de Datos MySQL para el almacenamiento de datos de usuarios, visitantes y accesos.
- › El Framework Laravel se emplea para facilitar la comunicación entre las vistas y los controladores, así como para administrar la seguridad y la organización de rutas.

7.2. Atributo de interfaz

7.2.1 Interfaces Externas:

Base de Datos (MySQL):

- › El sistema realiza operaciones de lectura y escritura en una base de datos externa, donde se almacenan registros de visitantes, usuarios y permisos de acceso.

Framework: Laravel:

- › Laravel funge como un sistema de apoyo externo que maneja la estructura MVC del proyecto, proporcionando mecanismos de enrutamiento, validación de datos, autenticación de usuarios y generación de respuestas HTTP.

7.2.2 Interfaces Internas

La interacción interna del sistema se modela mediante el Diagrama de Componentes UML. Cada vista tiene una relación directa con su controlador respectivo, y los controladores acceden a la base de datos para realizar operaciones necesarias.

7.2.3 Relaciones principales:

Vistas	Controladores
Visitante->create Visitante->index Visitante->show Visitante->pdf	VisitanteController
Plantele->index Plantele->show	PlanteleController
Buscar->buscar	BusquedaController
Auth->login	LoginController
Auth->verify	VerificationController

Tabla 4

7.3. Ejemplos de idiomas

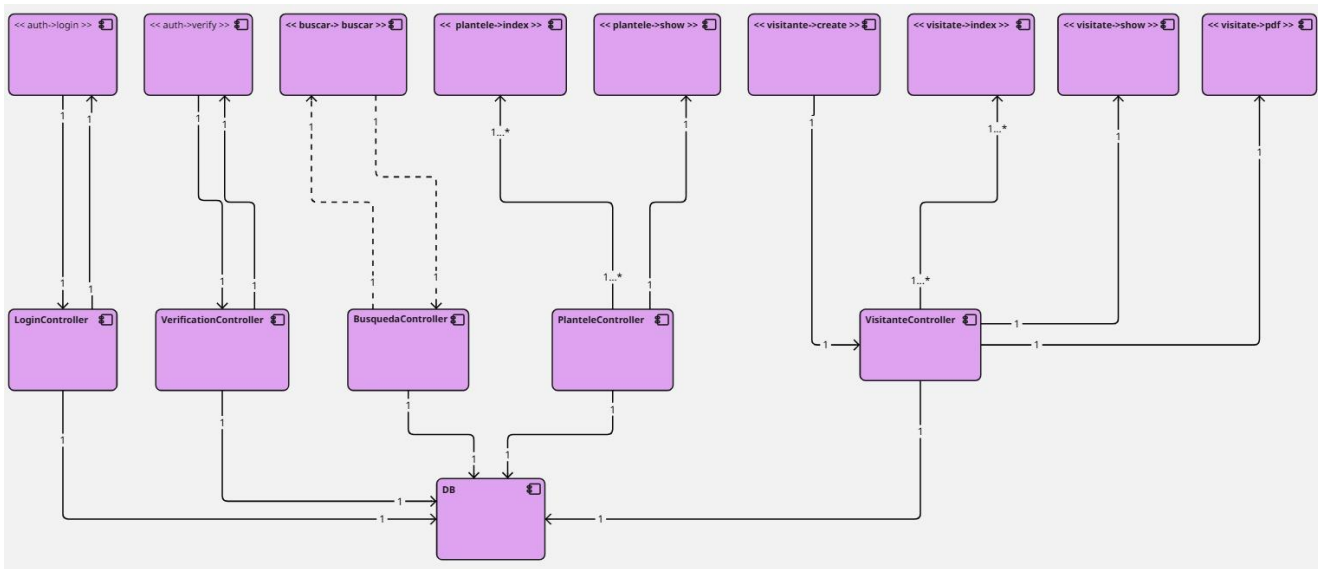


Diagrama de componentes de las vistas

Guadalupe Yamileth

2 horas

Diagrama 18

7.3.1 Vistas

- › **Auth->login:** Permite el ingreso de usuarios autorizados al sistema. Se conecta a LoginController para validar credenciales.
- › **Auth->verify:** Gestiona de la verificación del correo electrónico para cualquier usuario que se haya registrado recientemente con la aplicación.
- › **Buscar->buscar:** Conexión con el escáner, obtiene el identificador almacenado en el QR y comprueba su existencia en la base de datos.
- › **Plantele->index:** Lista de planteles universitarios u oficinas de la UACM.
- › **Plantele->show:** Información específica de un plantel.
- › **Visitante->create:** Permite registrar nuevos visitantes y gestionar accesos temporales.
- › **Visitante->show:** Información de registro del visitante.
- › **Visitante->pdf:** Generación del pdf del visitante.

7.3.2 Controladores

- › **LoginController, VerificationController:** Encargados de brindar seguridad al sistema.
- › **VisitanteController, PlanteleController, BusquedaController:** Controladores responsables de la lógica de negocio, conexión con la base de datos y procesamiento de las solicitudes de las vistas.
- › **DB:** Base de Datos externa donde se almacenan los registros de usuarios, visitantes y accesos.

8. Punto de vista de la estructura

El diagrama de estructura nos ayudara a comprender la composición interna del Sistema de Acceso Rápido (S.A.R.). A continuación, se describe como se compone el sistema y la relación entre ellos.

8.1. Consideraciones

- La base de datos se representará como un sistema aparte.
- El escáner QR que se utilizara para el funcionamiento del sistema, este no se requiere modelar como un sistema aparte, o modelar como un Interfaz de Programación de Aplicaciones (**API**) conjunto de reglas y protocolos que permiten la conexión entre el sistema y el escáner.
- El diagrama no está representando las conexiones existentes para la **autenticación** de un usuario (*Función de ingresar a sistema*), debido a que se utiliza un plugin el cual se encarga de realizar la autenticación del usuario y generar una sesión.

8.2. Componentes

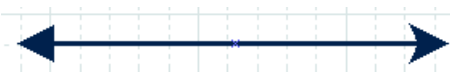


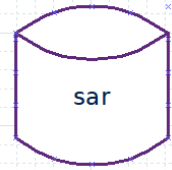
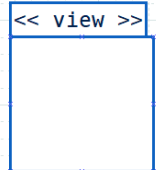
Información de símbolos	
	Indica que reciben y envían información entre ambos.
	Indica que solo se va a envía información.
	Indica dependencia de un componente.
	Representa la base de datos, representación como un sistema aparte.
	Representa todas las vistas implementadas en el sistema.

Tabla 5

8.3. Diagrama de clases

Previamente se presentó el [Diagrama de paquetes UML], explicando como estaría conformado el sistema (Back-end). Describe los diferentes paquetes, realizamos uan breve descripción del funcionamiento de cada una.

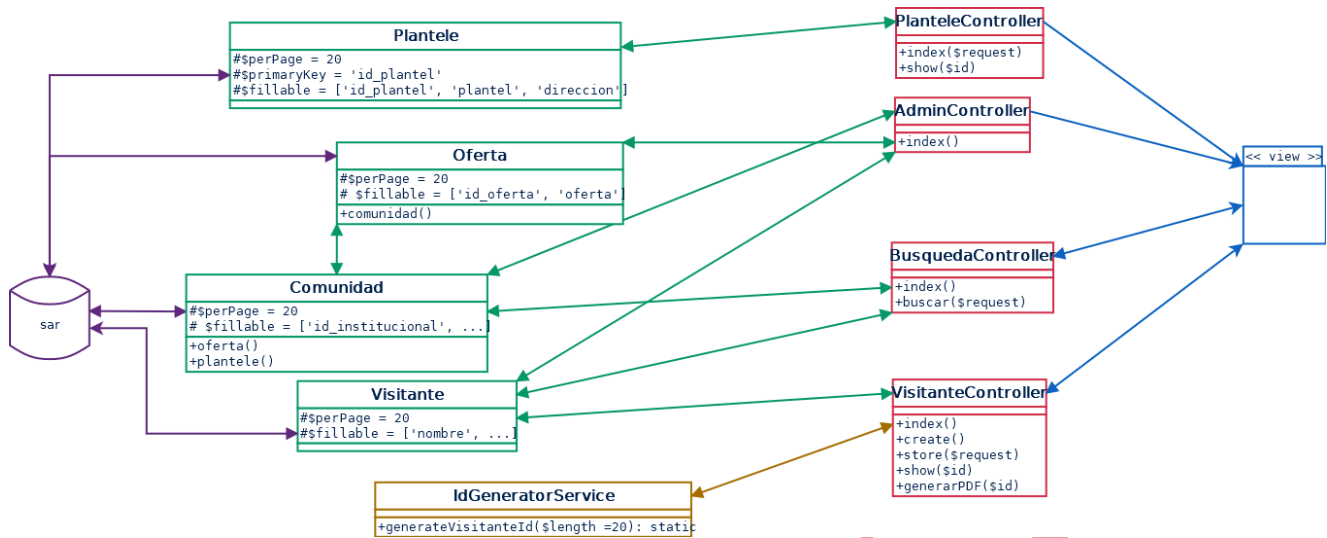


Diagrama de clases

Manzur Rodriguez

1 Hora

Diagrama 19

8.3.1 Modelos

- **Plantele:** El atributo '*primaryKey*' establece el '*id*' por el cual será buscado en la base de datos. Se tiene que establecer el nombre del identificador, dado que por defecto el sistema busca '*id*'. El atributo '*fillable*' indica los campos que tiene la tabla en la base de datos.
- **Oferta:** El atributo '*fillable*' indica los campos que tiene la tabla en la base de datos.
- **Comunidad:** El atributo '*fillable*' indica los campos que tiene la tabla en la base de datos. El modelo actual se relaciona con el modelo '*Oferta*', esto a que en la base de datos existe una relación por medio de un atributo.
- **Visitante:** El atributo '*fillable*' indica los campos que tiene la tabla en la base de datos.

8.3.2 Service

- **IdGeneratorService:** Genera un código compuesto por (*números, letras mayúsculas y minúsculas*), la longitud por defecto se establece en el parámetro.

8.3.3 Controller

- **Plantele:** Recibe como parámetro el número de la página actual. Funciones para ver la lista de planteles de forma general, y ver la información de un plantel en específico.
- **AdminController:** Su método '*index()*' envía información a la vista, para ello hace uso de los controladores '*Oferta, Comunidad, Visitante*'. Esta es utilizada para mostrar la cantidad de usuarios que entraron por día y generar una gráfica de barras.
- **BusquedaController:** Recibe un '*id*' con una longitud máxima de 20, el cual buscara en la base de datos, para ello hace uso del controlador '*Comunidad*' y '*Visitante*'. Si la búsqueda es exitosa, envía la información del registro que coincida con exactitud a la vista.

- **VisitanteController:** Tiene la funcionalidad para crear un nuevo registro de un visitante, ver todos los visitantes registrados el mismo día, ver la información de un visitante en específico, generar un QR único para cada visitante y generar un PDF con la información del visitante y su QR.

8.4. Diagrama de estructura

Representación visual que muestra la organización y relaciones entre los componentes del sistema. Este diagrama nos ayudara a ilustran cómo los módulos o clases se conectan entre sí y cómo se llama un módulo a otro.

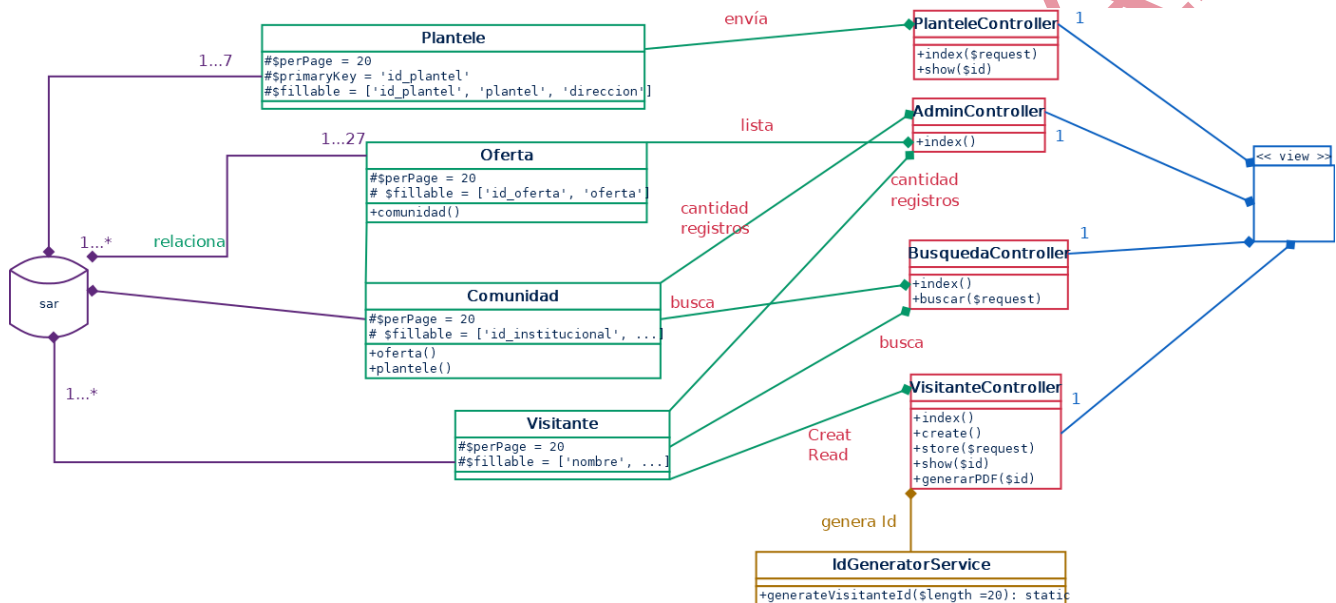


Diagrama de clases

Manzur Rodriguez

1 Hora

Diagrama 20

8.4.1 Relación

- **View:** Para que estas sean mostradas, es necesario que exista uno o varios controladores.
- **PlanteleController:** Utiliza el modelo '*Plantele*' para que recia de la DB la lista de planteles de la UACM, esta información se envía a la vista.
- **AdminController:** Llama al modelo 'Oferta' para que le envíe la lista de oferta académica que tiene la UACM, con esta se clasificaran los datos para generar gráficas. Los modelos 'Comunidad' y 'Visitante' solo envían el número de usuarios que ingresaron al plantel en un día específico. Todo esto lo envía a una vista.
- **BusquedaController:** De su vista recibe un '*identificador*' de tipo Sting, el cual buscará en el modelo 'Comunidad' o 'Visitante', en caso de encontrarlo, el modelo enviará información del usuario al que le pertenece el 'identificador' y esta será mostrada en su vista correspondiente.
- **VisitanteController:** Este realiza las operaciones (Create y Read) en relación con un visitante. Para realizar la operación de **Cread**, este necesita el servicio de '*IdGeneratorService*', el cual le genera

un ‘*identificador*’ (compuesto por letras mayúsculas, minúsculas y números) único, este se le asigna al visitante que se está registrando.

9. Punto de vista de la interacción

El punto de vista de la interacción en el sistema de control de acceso mediante código QR se representa mediante diagramas de secuencia UML y diagrama de comunicación UML. Estos diagramas describen cómo los actores (usuario y visitante) interactúan con los diferentes componentes del sistema (registro, lector, sistema y base de datos) durante el proceso de validación de acceso.

9.1.1 Problemas de diseño

Se aborda la asignación de responsabilidades entre los objetos involucrados, particularmente mediante la colaboración entre componentes como el lector, el sistema central y la base de datos. Las secuencias muestran cómo se envían y reciben mensajes para realizar validaciones, registrar visitantes y proporcionar una respuesta al actor. Se modelan eventos como la lectura del código QR, la búsqueda en la base de datos, la verificación de registros existentes y la emisión del resultado (acceso concedido o denegado).

9.1.2 Elementos de diseño

Los elementos utilizados incluyen *clases* (como lector, sistema, registro, base de datos), *eventos* (escanear QR, solicitar validación), *condiciones* (usuario o visitante), y la sincronización entre mensajes enviados y recibidos. La temporización y simultaneidad se observan en la respuesta inmediata del sistema a múltiples entradas concurrentes, permitiendo un flujo continuo de validación.

9.2. Diagrama de secuencia UML

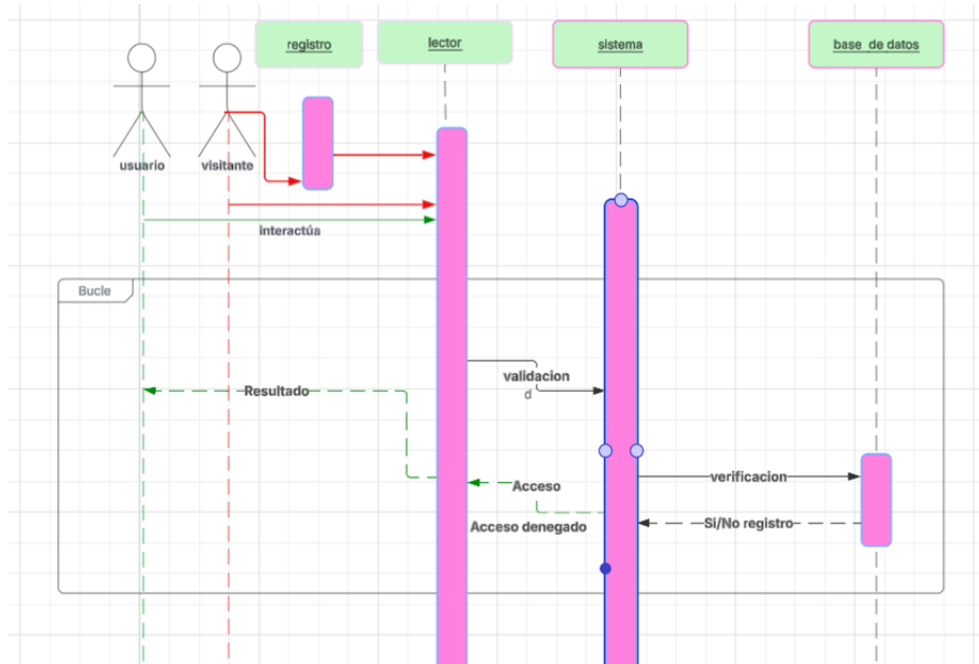


Diagrama de secuencia

Cristela Cruz Ovando

1.5 Horas

Diagrama 20

9.3. Diagrama de comunicación UML

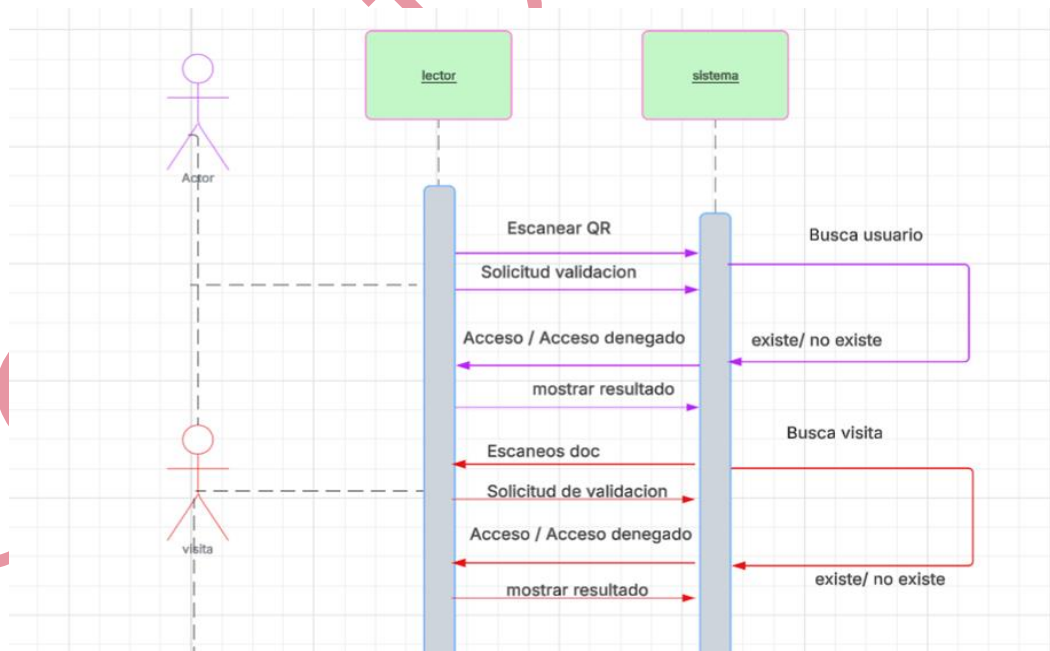


Diagrama de secuencia

Cristela Cruz Ovando

1.5 Horas

Diagrama 21

10. Punto de vista de la dinámica del estado

10.1.1 Propósito

Desde el punto de vista de la dinámica del estado, el sistema se analiza considerando cómo cambian los estados de los objetos principales en respuesta a eventos del usuario, interacciones con la base de datos o ejecución de controladores. Este enfoque permite entender mejor el ciclo de vida de entidades clave dentro del sistema, como los visitantes, usuarios y procesos de generación de identificadores o códigos QR.

10.1.2 Problemas de diseño

Uno de los elementos más representativos es la entidad **Visitante**, cuyo estado cambia durante su interacción con el sistema. La secuencia de estados por la que puede transitar un visitante es la siguiente:

- **Estado inicial:** No registrado.
- **Registro iniciado:** El visitante comienza el proceso mediante un formulario de registro.
- **Validación de datos:** El sistema verifica los datos ingresados (nombre, comunidad, etc.).
- **ID generado:** Se genera un identificador único a través del servicio IdGeneratorService.
- **Visitante registrado:** Se almacena la información del visitante en la base de datos.
- **Código QR generado:** El sistema genera un código QR con el identificador del visitante mediante el modelo QR.
- **PDF generado (opcional):** A solicitud del usuario o controlador, se crea un documento PDF con la información del visitante.
- **Estado final:** Visitante registrado y código/pase emitido.

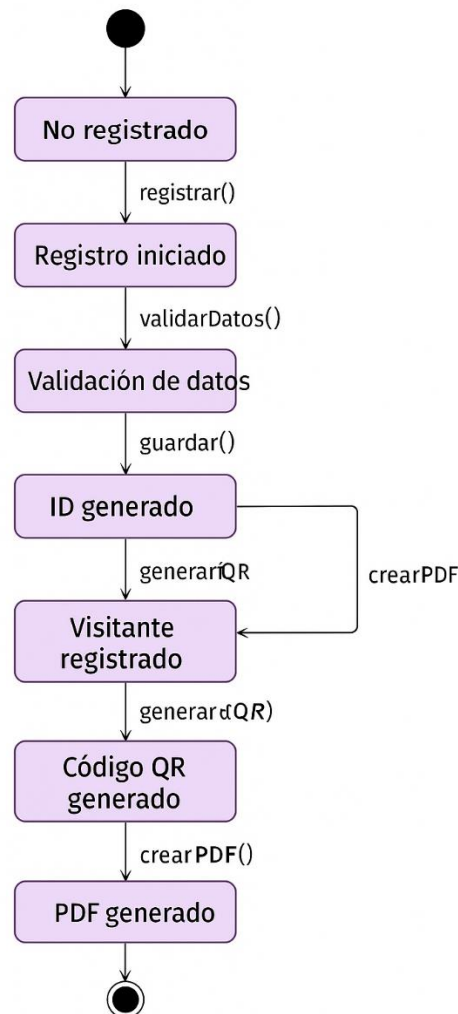


Diagrama de estados UML

Guadalupe Valadez

1 Hora

Diagrama 22

Desde la perspectiva dinámica, el objeto **QR** dentro del sistema de control de acceso de visitantes de la UACM representa un recurso clave que cambia de estado durante su ciclo de vida, y cuya validez y uso son fundamentales para garantizar la seguridad del ingreso a las instalaciones.

10.1.3 Transformación dinámica del estado del QR

El ciclo de vida de un código QR comienza desde el momento en que se genera para un visitante y puede concluir de distintas maneras, dependiendo de si es utilizado, escaneado o si expira antes de ser usado. El comportamiento esperado del objeto QR se describe en los siguientes estados:

- **QR No Generado:** Estado inicial donde aún no se ha solicitado la creación de un código QR.
- **QR Generado:** El sistema ha creado un código único, pero aún no ha sido vinculado a un visitante.
- **QR Asignado:** El código QR ha sido asociado a un visitante registrado en el sistema.

- **QR Escaneado:** El visitante ha presentado el código QR en un punto de control y fue leído correctamente.
- **QR Verificado:** El sistema validó que el QR pertenece a un visitante registrado y que es válido para el acceso.
- **QR Inválido:** El código QR escaneado no corresponde a ningún visitante válido o ha sido manipulado.
- **QR Expirado:** El código ha caducado por uso único o por tiempo de vigencia.
- **Estado Final:** El ciclo de vida del QR termina tras ser validado, rechazado o expirar.

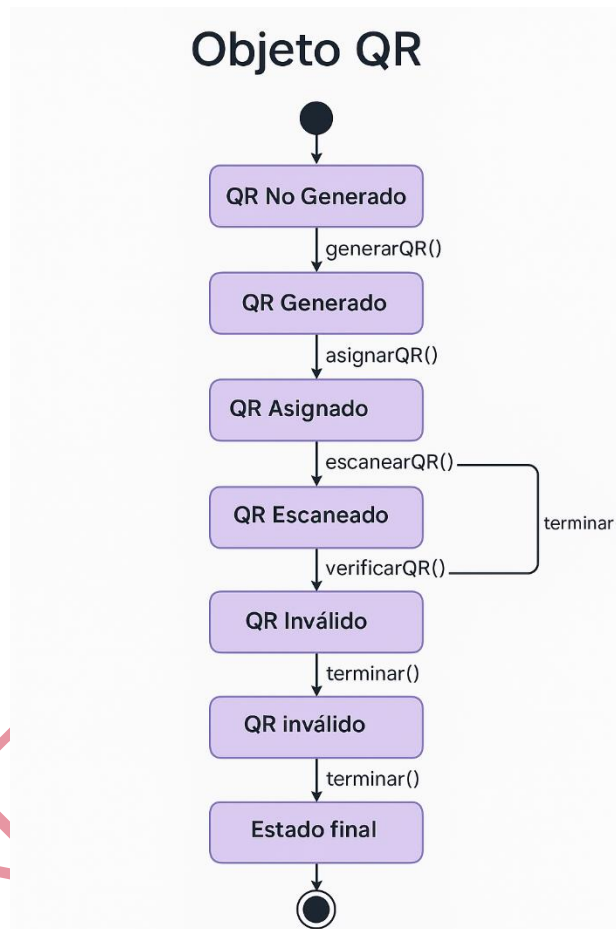


Diagrama de estados UML

Guadalupe Valadez

1 Hora

Diagrama 23

II. Definiciones, acrónimos y abreviaturas

- › **SAR:** “Sistema de Acceso Rápido” (SAR).
- › **UACM:** Universidad Autónoma de la Ciudad de México.
- › **Comunidad:** Estudiantes y trabajadores.
- › **Trabajadores:** Personal docente e investigador, Personal de administración y servicios y Personal de vigilancia.

- › **Campus:** Área de instalaciones universitarias donde se realizan actividades académicas y administrativas.
- › **Servidor:** Sistema informático que proporciona recursos y servicios a otros ordenadores a través de una red.
- › **Base de Datos:** Conjunto organizado de datos almacenados electrónicamente, permitiendo su gestión y actualización.
- › **Normativas:** Reglas y directrices establecidas por una autoridad para regular comportamientos y acciones.
- › **Políticas:** Normas que regulan las actividades y comportamiento dentro de la institución.
- › **UI (User Interface):** UI significa Interfaz de Usuario. Se refiere a la parte del software con la que los usuarios interactúan directamente. El diseño de UI se enfoca en la disposición visual y la presentación de los elementos en la pantalla.
- › **UX (Experiencia de Usuario):** UX Se refiere a la experiencia general del usuario al interactuar con el software. El diseño de UX abarca aspectos más amplios que solo la apariencia y se centra en cómo se siente el usuario durante el uso del producto.
- › **QA (Aseguramiento de la Calidad):** Es un proceso integral que se enfoca en asegurar que el software cumpla con los estándares de calidad y que funcione correctamente según los requisitos definidos.
- › **Formador:** Es un profesional encargado de capacitar a los usuarios, desarrolladores, y otros miembros del equipo sobre el uso de software, herramientas o metodologías específicas.
- › **Visitante:** Usuario final, el cual no pertenece de ninguna manera al plantel educativo.
- › **DB:** Base de datos de la UACM.
- › **Usuario:** Persona que desea acceder a la institución educativa.

12. Bibliografía

- A.U.S. Gustavo Torossi. Diseño de Sistemas. El proceso unificado de desarrollo de Software.
- Cervantes, Velasco, Castro; Arquitectura de Software. Conceptos y Ciclo de Desarrollo; Cengage Learning, 2016.