|  |
| --- |
| ATAM: Explorando el Universo |
| *Proyecto: Explorando el Universo* |
|  |
| **Revisión*: 1.0*** |
| **20/01/2025** |

|  |
| --- |

Índice

[**Ficha del documento 3**](#_heading=h.1fob9te)

[**1.ATAM 5**](#_heading=h.tyjcwt)

[Acciones 5](#_heading=h.3dy6vkm)

[Resultados Esperados 6](#_heading=h.1ksv4uv)

[**2.Objetivos de negocio 7**](#_heading=h.1t3h5sf)

[**3.Arquitectura 8**](#_heading=h.4d34og8)

[**4.Enfoque arquitectónico 10**](#_heading=h.2s8eyo1)

[**5.Árbol de utilidad 12**](#_heading=h.17dp8vu)

[**6.Análisis de enfoque arquitectónico 13**](#_heading=h.26in1rg)

[**7.Brainstorming y Priorización de Escenarios 17**](#_heading=h.lnxbz9)

[**8.Análisis de los Enfoques Arquitectónicos 18**](#_heading=h.35nkun2)

# **Ficha del documento**

| **Fecha** | **Revisión** | **Autor** | **Modificación** |
| --- | --- | --- | --- |
| *20/01/2025* | *01* | *Raquel R* | * *Formato al documento* |
| *21/01/2025* | *02* | *Raquel R* | * *Trabajo en el punto 1, 4 y 5* |

Documento validado por las partes en fecha:

| Por el cliente |  | Por la empresa suministradora |
| --- | --- | --- |
| [Firma] |  | [Firma] |
| Sr./Sra. |  | Sr./Sra. |

# 

# 1.ATAM

Este documento tiene como objetivo analizar las decisiones arquitectónicas tomadas para el proyecto 'Explorando el Universo' y su influencia en los aspectos clave de calidad del sistema. "Explorando el Universo" es una plataforma de aprendizaje en línea diseñada para ofrecer cursos de astronomía a estudiantes de niveles básico y medio, con el propósito de fomentar la curiosidad y el interés en la astronomía mediante contenido interactivo, evaluaciones, foros de discusión y recursos multimedia.

Para evaluar las decisiones arquitectónicas del sistema, utilizaremos el método ATAM (Architecture Tradeoff Analysis Method), una técnica que permite identificar riesgos y compromisos, además de analizar cómo estas decisiones impactan el desempeño y las características no funcionales de la plataforma. A lo largo del documento, se detallarán las cuatro fases del método ATAM aplicadas a este proyecto, considerando aspectos clave como la gestión de los cursos, la interacción entre estudiantes y tutores, y la administración de recursos multimedia.

También se revisarán posibles puntos débiles en la arquitectura y su posible impacto en el funcionamiento del sistema, proporcionando una visión clara del enfoque seguido junto con los evaluadores e involucrados en el proyecto. La aplicación del método ATAM nos permitirá asegurar una experiencia de aprendizaje enriquecedora y accesible para todos los usuarios.

# **2.Objetivos de negocio**

**Los objetivos del negocio son:**

**Accesibilidad Universal:**

* Garantizar que la plataforma sea completamente accesible para todos los usuarios, incluyendo personas con discapacidades visuales y auditivas, cumpliendo con los estándares WCAG 2.1 Nivel AA.
* Asegurar la compatibilidad con una amplia gama de dispositivos (computadoras, tabletas y teléfonos inteligentes) y navegadores web populares.

Experiencia de Usuario Personalizada:

* Proporcionar a los estudiantes un acceso rápido y sencillo a sus cursos, permitiéndoles retomar su progreso justo donde lo dejaron mediante enlaces directos al contenido más reciente.
* Facilitar la navegación intuitiva y la comprensión del contenido mediante una interfaz clara y organizada.

Disponibilidad Continua:

* Hay que asegurar que la plataforma educativa esté disponible las 24 horas del día, permitiendo el acceso a los contenidos en cualquier momento y lugar.
* Definir horarios específicos para la comunicación con los profesores, de lunes a viernes de 8:00 a 17:00, garantizando el soporte académico dentro de esos periodos.

Seguridad de la Información:

* Implementar medidas de seguridad robustas, como cifrado de datos en tránsito y en reposo, controles de acceso y autenticación multifactorial (MFA), protegiendo la información personal de los usuarios.

Escalabilidad y Mantenimiento Eficiente:

* Diseñar la arquitectura del sistema de manera modular para facilitar futuras actualizaciones, nuevas funcionalidades y correcciones de errores sin afectar la operatividad existente.
* Asegurar un código bien documentado y estructurado para optimizar el tiempo de desarrollo y mantenimiento.

Fomento del Aprendizaje Interactivo:

* Ofrecer recursos multimedia enriquecidos, como videos educativos con subtítulos, simulaciones interactivas y foros de discusión, para mejorar la experiencia de aprendizaje.
* Proporcionar herramientas de evaluación efectivas que permitan a los estudiantes medir su progreso en los distintos módulos del curso.

Cumplimiento de Regulaciones y Estándares:

* Hay que asegurar que la plataforma cumpla con las normativas de accesibilidad web y seguridad de datos, evitando riesgos legales y garantizando la inclusión de todos los usuarios.

# 

## **3. Arquitectura**

# La arquitectura del sistema "Explorando el Universo" sigue un modelo de tres capas para garantizar escalabilidad, seguridad y mantenibilidad. Se utilizarán Django para la lógica de negocio y PostgreSQL como sistema de gestión de bases de datos, además de JavaScript para mejorar la interactividad del frontend. A continuación, se detallan las capas y cómo satisfacen los requerimientos establecidos.

### 3.1. Capa de Presentación (Frontend)

#### Componentes:

# Aplicación Web Responsiva: Desarrollada con HTML, CSS y JavaScript, posiblemente utilizando un framework o biblioteca como React.js para construir una interfaz de usuario dinámica y fluida. Esta capa se encargará de la presentación del contenido educativo.

# Interfaz Accesible: La interfaz de usuario será accesible y cumplirá con los estándares WCAG 2.1 Nivel AA, garantizando que las personas con discapacidades visuales y auditivas puedan interactuar con el contenido. Esto incluirá subtítulos en videos y un diseño que asegure un buen contraste.

# Integración: Se integrará con servicios de terceros para la gestión de foros, comentarios y notificaciones.

#### Función:

# Permitir el inicio de sesión de estudiantes y profesores.

# Gestión de perfiles con opciones de modificación y eliminación.

# Facilitar la búsqueda e inscripción en cursos, además de la visualización del contenido y la realización de evaluaciones.

# Presentar certificados de finalización una vez que los cursos han sido aprobados.

#### Contribución a los Objetivos del Negocio:

# Mejora la experiencia del usuario con una interfaz intuitiva y accesible.

# Garantiza la compatibilidad con dispositivos móviles, tabletas y computadoras de escritorio.

# Soporta un alto número de usuarios simultáneos sin degradar el rendimiento.

### 3.2. Capa de Lógica de Negocio (Backend)

#### Componentes:

# Servidor de Aplicaciones con Django: La lógica de negocio se implementará utilizando Django, permitiendo la creación de una arquitectura RESTful para gestionar la autenticación de usuarios, la inscripción en cursos y la creación de evaluaciones.

# Mecanismos de Seguridad: Se implementarán sistemas de autenticación seguros (como JWT y OAuth 2.0) para proteger las sesiones de los usuarios.

# Módulos de Evaluaciones: Se incluirán módulos para gestión de evaluaciones, tanto manuales como automáticas, con generación de reportes de rendimiento.

#### Función:

# Procesar solicitudes de autenticación y recuperación de contraseña.

# Administrar la gestión de perfiles, cursos, evaluaciones y contenido.

# Implementar políticas de seguridad y protección de datos, incluyendo cifrado de contraseñas y validación de acceso.

# Asegurar la disponibilidad 24/7 mediante estrategias de balanceo de carga y alta disponibilidad.

#### Contribución a los Objetivos del Negocio:

# Asegura el cumplimiento de estándares de accesibilidad y seguridad.

# Permite la gestión eficiente de los recursos y el almacenamiento seguro de datos.

# Facilita la realización de actualizaciones y mejoras de manera modular.

### 3.3. Capa de Acceso a Datos (Base de Datos)

#### Componentes:

# Base de Datos PostgreSQL: Utilizará PostgreSQL como sistema de gestión de bases de datos. Esta elección permitirá manejar grandes volúmenes de datos de manera eficiente y ofrecer funcionalidades avanzadas.

# Integración con Django ORM: Se utilizará el ORM de Django para interactuar con la base de datos de forma sencilla y segura, facilitando operaciones CRUD (crear, leer, actualizar, eliminar) en las entidades del sistema.

#### Función:

# Almacenar información de usuarios, perfiles, inscripciones, cursos y evaluaciones.

# Facilitar la recuperación de información en tiempo real para dashboards y reportes.

# Gestionar los registros de actividad para auditoría y control de acceso.

#### Contribución a los Objetivos del Negocio:

# Garantiza la seguridad e integridad de los datos almacenados.

# Proporciona una estructura flexible para adaptarse a futuras mejoras y necesidades.

# Asegura la disponibilidad de la información para usuarios de distintas zonas horarias.

# 

# **4.Investigación y Análisis**

2.1 Identificación de enfoques arquitectónicos  
Se han identificado los siguientes enfoques arquitectónicos para el sistema:

1. Autenticación basada en JWT (JSON Web Token)
   * Implementación de un sistema de autenticación basado en JWT, garantizando sesiones seguras y una gestión eficiente de usuarios.
   * Seguridad mejorada mediante la implementación de OAuth 2.0 para la autenticación con cuentas institucionales (Google/Microsoft).
2. Arquitectura de tres capas
   * Dividida en:
     + Capa de Presentación: interfaz de usuario basada en React.js.
     + Capa de Lógica de Negocio: manejo de lógica con Flask.
     + Capa de Datos: almacenamiento en MySQL.
   * Facilita la escalabilidad y mantenibilidad del sistema.
3. Cifrado de datos sensibles
   * Implementación del estándar AES-256 para la protección de información confidencial, como contraseñas y datos personales de estudiantes y profesores.
   * Encriptación en tránsito mediante SSL/TLS para garantizar la seguridad en la comunicación de datos.
4. Notificaciones automáticas
   * Uso de notificaciones push y por correo electrónico para recordar fechas importantes como inscripciones, evaluaciones y certificados de finalización.
   * Integración con servicios como Firebase Cloud Messaging (FCM) para la gestión eficiente de notificaciones.
5. Modularidad en el código
   * Separación de funcionalidades clave como:
     + Gestión de cursos y perfiles.
     + Evaluaciones automáticas y manuales.
     + Administración de contenidos y certificados.
   * Facilita la ampliación y el mantenimiento del código.
6. Integración con infraestructura existente
   * Conexión con sistemas de autenticación de la institución para una experiencia de usuario fluida.
   * Sincronización con plataformas educativas como Google Classroom o Moodle para una mejor gestión de contenido.
7. Respaldo automático
   * Sistema de respaldo automatizado de la base de datos cada 12 horas para evitar la pérdida de información crítica.
   * Almacenamiento de respaldos en entornos seguros y accesibles.
8. Compatibilidad multiplataforma
   * Desarrollo con tecnologías web responsivas (React.js) para garantizar accesibilidad desde diferentes dispositivos y navegadores (Chrome, Firefox, Edge).
   * Optimización para dispositivos móviles a través de PWA (Progressive Web Apps).
9. Tolerancia a fallos
   * Implementación de estrategias de recuperación ante fallos mediante balanceo de carga y replicación de base de datos.
   * Monitoreo continuo del sistema con herramientas como Prometheus y Grafana para detectar incidentes rápidamente.
10. Cumplimiento de estándares de accesibilidad

* Diseño de interfaz conforme a los estándares WCAG 2.1 AA, asegurando la inclusión de personas con discapacidades visuales o motoras.
* Pruebas de usabilidad para garantizar que el sistema sea accesible para todos los usuarios.

1. Escalabilidad horizontal

* Implementación de una arquitectura que permita el escalado horizontal mediante la adición de nodos según la demanda de usuarios simultáneos.
* Uso de contenedores con Docker y orquestación con Kubernetes para facilitar el despliegue en diferentes entornos.

# 5.Árbol de utilidad

En esta sección del documento, se presentarán los escenarios de calidad en forma de un árbol de utilidad.

Imagen que contiene Diagrama

Descripción generada automáticamente

# 

# 6.Análisis de enfoque arquitectónico

En este análisis se evalúa cómo el Modelo en Capas impacta los factores de calidad clave como fiabilidad, rendimiento, disponibilidad, seguridad y escalabilidad, y cómo su implementación en "Explorando el Universo" responde a estos atributos.

El Modelo en Capas (también conocido como arquitectura de n-capas) es una aproximación en la que se dividen las responsabilidades del sistema en diferentes capas, generalmente:

1. Capa de presentación: Interacción con el usuario.
2. Capa de lógica de negocio: Procesamiento de la información.
3. Capa de acceso a datos: Gestión de la persistencia y consultas a la base de datos.
4. Capa de almacenamiento: Base de datos o sistemas de archivos.

Impacto del Modelo en Capas en los Factores de Calidad

1. Fiabilidad

* Impacto de la capa de presentación: La capa de presentación es crucial para mostrar la información de manera coherente y sin errores. Un diseño robusto y bien validado puede prevenir errores de presentación que afecten la percepción de fiabilidad. Además, el correcto manejo de los inputs del usuario reduce los errores al comunicar con el sistema.
* Impacto de la capa de lógica de negocio: Esta capa es la encargada de ejecutar las reglas del negocio y gestionar la lógica de la aplicación. Un diseño bien estructurado en esta capa asegura que el sistema opere sin fallos, ya que centraliza los procesos y puede manejar excepciones y errores de manera eficaz.
* Impacto de la capa de acceso a datos: Si la capa de acceso a datos está bien optimizada, las operaciones con la base de datos serán consistentes y fiables, lo cual es crucial para mantener la integridad de la información a lo largo del tiempo.
* Impacto de la capa de almacenamiento: La fiabilidad de la capa de almacenamiento depende de la correcta implementación de backups, recuperación ante desastres y control de versiones. Esta capa es esencial para garantizar que los datos no se pierdan y se mantengan consistentes.

2. Disponibilidad

* Impacto de la capa de presentación: La capa de presentación debe estar optimizada para asegurar que los usuarios puedan acceder al sistema sin tiempos de espera prolongados. Si bien la disponibilidad de esta capa no depende directamente de otros componentes, una buena implementación en la interfaz puede mejorar la accesibilidad del sistema.
* Impacto de la capa de lógica de negocio: La lógica de negocio debe estar diseñada para asegurar que las operaciones se completen rápidamente y sin interrupciones, incluso en situaciones de alta carga. Si se gestiona correctamente, puede garantizar que el sistema siga funcionando sin errores y esté disponible para todos los usuarios.
* Impacto de la capa de acceso a datos: El acceso rápido y eficiente a la base de datos es fundamental para mantener la disponibilidad. Técnicas como la replicación de bases de datos o el uso de sistemas de caching pueden mejorar significativamente la disponibilidad, ya que permiten una recuperación rápida en caso de fallo.
* Impacto de la capa de almacenamiento: Esta capa debe ser diseñada para garantizar la disponibilidad constante de los datos, utilizando tecnologías como bases de datos distribuidas y sistemas de respaldo.

3. Rendimiento

* Impacto de la capa de presentación: La capa de presentación debe ser ligera y rápida para que el sistema sea ágil. Un diseño optimizado, con técnicas como el uso de cargas perezosas (lazy loading) o la compresión de archivos, ayudará a reducir el tiempo de carga de las páginas.
* Impacto de la capa de lógica de negocio: El rendimiento de la lógica de negocio puede ser impactado por algoritmos ineficientes o por una mala gestión de los procesos. El uso de técnicas de optimización, como la ejecución en paralelo y el uso adecuado de recursos, puede mejorar el rendimiento general del sistema.
* Impacto de la capa de acceso a datos: El rendimiento de esta capa depende en gran medida de cómo se gestionan las consultas a la base de datos. Consultas mal optimizadas pueden generar cuellos de botella. Es importante implementar índices adecuados, utilizar técnicas de caching y minimizar el número de consultas necesarias para cada operación.
* Impacto de la capa de almacenamiento: La velocidad de acceso a los datos es esencial para el rendimiento general del sistema. El uso de tecnologías de almacenamiento rápido y la correcta optimización de la base de datos son cruciales para mejorar el rendimiento.

4. Escalabilidad

* Impacto general del modelo en capas: La arquitectura en capas facilita la escalabilidad del sistema, ya que permite escalar cada capa de manera independiente según sea necesario. Por ejemplo, la capa de acceso a datos puede ser optimizada para manejar grandes volúmenes de datos, mientras que la capa de presentación puede ser distribuida para manejar más usuarios simultáneamente. Esta independencia en las capas es fundamental para la escalabilidad.
* Impacto de la capa de presentación: A medida que aumentan los usuarios, la capa de presentación debe ser capaz de manejar más peticiones de forma eficiente. Se puede escalar horizontalmente mediante balanceo de carga para distribuir las solicitudes de los usuarios.
* Impacto de la capa de lógica de negocio: La lógica de negocio puede ser escalada verticalmente (mejorando el hardware) o horizontalmente (distribuyendo el procesamiento entre servidores). El patrón en capas facilita la expansión de esta capa sin afectar las otras.
* Impacto de la capa de acceso a datos: La capa de acceso a datos es clave para la escalabilidad. El uso de bases de datos distribuidas, particionadas o el uso de técnicas de caching avanzadas puede mejorar la escalabilidad de esta capa y reducir la carga sobre la base de datos.
* Impacto de la capa de almacenamiento: La capacidad de almacenamiento puede ampliarse mediante el uso de bases de datos distribuidas o almacenamiento en la nube. Esto permitirá que el sistema pueda manejar una mayor cantidad de datos sin comprometer su rendimiento.

5. Seguridad

* Impacto de la capa de presentación: La capa de presentación debe estar diseñada para prevenir ataques como Cross-Site Scripting (XSS) y asegurar que los datos ingresados por los usuarios sean validados y sanitizados correctamente. Esto previene vulnerabilidades de seguridad en la interfaz del usuario.
* Impacto de la capa de lógica de negocio: La lógica de negocio es responsable de implementar reglas de validación de acceso y permisos. El controlador debe asegurarse de que solo los usuarios autorizados puedan ejecutar ciertas acciones. Además, el manejo de la lógica de acceso y autenticación se realiza aquí.
* Impacto de la capa de acceso a datos: La capa de acceso a datos debe proteger los datos sensibles mediante cifrado tanto en tránsito como en reposo. Asimismo, las consultas a la base de datos deben ser protegidas contra inyecciones SQL y otros tipos de ataques.
* Impacto de la capa de almacenamiento: En la capa de almacenamiento, se deben implementar medidas de seguridad como cifrado de datos sensibles, control de acceso a la base de datos y auditoría constante de la infraestructura de almacenamiento.

Fortalezas del Modelo en Capas

* Escalabilidad: La arquitectura permite escalar cada capa de manera independiente, lo que facilita la adaptación a mayores volúmenes de usuarios y datos.
* Mantenimiento y Flexibilidad: La separación clara entre las capas facilita el mantenimiento y la evolución del sistema. Cualquier cambio en una capa no afectará directamente a las otras.
* Seguridad Mejorada: Cada capa puede ser asegurada de manera independiente, lo que reduce el riesgo de ataques y aumenta la protección de los datos.

Riesgos Identificados

* Interdependencia entre capas: Aunque el modelo en capas ayuda a mantener la modularidad, una mala comunicación entre las capas puede generar dependencias fuertes que dificulten el mantenimiento.
* Cuellos de botella en la capa de acceso a datos: Si la capa de acceso a datos no está optimizada correctamente, puede convertirse en un cuello de botella que afecte todo el sistema.

Recomendaciones

* Optimización de la capa de acceso a datos: Utilizar técnicas de caching, normalización de base de datos, y asegurar consultas eficientes.
* Escalabilidad de la capa de presentación: Implementar técnicas como balanceo de carga y procesamiento asíncrono para manejar grandes volúmenes de usuarios concurrentes.
* Reforzar la seguridad: Implementar medidas de seguridad como cifrado, autenticación multifactor y validaciones en la capa de presentación y lógica de negocio.

7. Brainstorming y Priorización de Escenarios

El brainstorming (tormenta de ideas) se llevó a cabo con los miembros del equipo para identificar y elaborar una lista de escenarios relevantes relacionados con la arquitectura del sistema. Durante esta fase, se discutieron los diferentes escenarios que podrían surgir a medida que se construye y despliega la plataforma educativa.

Escenarios Identificados:

# Inscripción de estudiantes en cursos.

# Registro y autenticación de usuarios.

# Gestión de contenidos por parte de los profesores.

# Interacción de estudiantes y profesores en foros de discusión.

# Recuperación de contraseñas.

# Acceso a la plataforma desde diferentes dispositivos.

# Evaluación del rendimiento del sistema bajo carga simulada.

# Priorización de Escenarios:

# Se utilizó una clasificación basada en la importancia y el impacto de cada escenario en el sistema. Los escenarios fueron evaluados según la probabilidad de ocurrencia y el impacto en la calidad del sistema:

# Alta Prioridad:

# Inscripción de estudiantes en cursos.

# Registro y autenticación de usuarios.

# Media Prioridad:

# Gestión de contenidos por parte de los profesores.

# Evaluación del rendimiento del sistema bajo carga simulada.

# Baja Prioridad:

# Recuperación de contraseñas.

# Interacción de estudiantes y profesores en foros de discusión.

# Acceso a la plataforma desde diferentes dispositivos.

# Esta priorización permitirá al equipo enfocar sus esfuerzos en los escenarios más críticos que podrían afectar la calidad y el rendimiento de la plataforma, asegurando que se aborden de manera eficiente durante el desarrollo e implementación.

# 8.Análisis de los Enfoques Arquitectónicos

#### Enfoque Arquitectónico Seleccionado: Arquitectura por Capas

* Descripción: La arquitectura por capas divide el sistema en tres componentes principales:
  + Capa de Presentación: Donde se encuentra la interfaz de usuario y las interacciones directas del usuario.
  + Capa de Lógica de Negocio: Que maneja la lógica y las reglas de negocio de la aplicación.
  + Capa de Acceso a Datos: Que se encarga de la gestión de la base de datos y la persistencia de los datos.

#### Evaluación de Factores de Calidad

1. Fiabilidad:
   * Impacto: La arquitectura por capas facilita el manejo de errores en la lógica de negocio, permitiendo una recuperación más fácil y asegurando que los componentes no se vean afectados entre sí.
2. Rendimiento:
   * Impacto: La adecuada optimización de la capa de acceso a datos permite una rápida recuperación de información, así como un manejo eficiente de las solicitudes de los usuarios simultáneamente.
3. Disponibilidad:
   * Impacto: La arquitectura permite implementar soluciones de alta disponibilidad y balanceo de carga, asegurando que el sistema esté accesible casi todo el tiempo, cumpliendo con los estándares del negocio.
4. Mantenibilidad:
   * Impacto: La separación clara de las capas permite que los futuros desarrollos y correcciones de errores se realicen de manera más sencilla, sin riesgo de afectar otras partes del sistema.
5. Seguridad:
   * Impacto: La implementación de medidas de seguridad en cada capa, como validaciones y controles de acceso, facilita la protección de datos sensibles.