

# Análisis Arquitectónico del Sistema CIED

## *Cálculo Integral + Ecuaciones Diferenciales*

Sistema CIED - Documentación Técnica

January 19, 2026

### **Abstract**

Este documento presenta un análisis completo del sistema CIED (Cálculo Integral + Ecuaciones Diferenciales), una aplicación web educativa desarrollada en Flask que combina un repositorio de errores matemáticos con un sistema de evaluación interactiva basado en inteligencia artificial. Se detalla su arquitectura, componentes principales, flujo de funcionamiento y tecnologías utilizadas.

### **Contents**

# 1 Introducción al Sistema CIED

## 1.1 Propósito y Alcance

El sistema CIED es una plataforma educativa innovadora diseñada para apoyar el aprendizaje de técnicas avanzadas de cálculo integral y ecuaciones diferenciales. Combina dos componentes principales:

1. **Repositorio de Errores:** Base de conocimiento estructurada de errores comunes cometidos por estudiantes
2. **Sistema de Quiz Interactivo:** Plataforma de evaluación con generación dinámica de ejercicios usando IA

## 1.2 Contexto Educativo

- **Curso:** Cálculo Integral y Ecuaciones Diferenciales (MATE-1214)
- **Nivel:** Universitario avanzado
- **Metodología:** Aprendizaje basado en errores con retroalimentación inmediata
- **Tecnología:** Integración de IA para personalización de ejercicios

## 1.3 Objetivos Pedagógicos

- Identificar y catalogar patrones de error comunes
- Proporcionar retroalimentación específica y actionable
- Generar ejercicios adaptativos usando IA
- Facilitar el aprendizaje autónomo y la remediación

# 2 Arquitectura General

## 2.1 Patrones de Diseño Implementados

El sistema CIED sigue las mejores prácticas de desarrollo web moderno:

- **Application Factory Pattern:** Para configuración flexible
- **Blueprint Pattern:** Para organización modular de rutas
- **Repository Pattern:** Para acceso a datos estructurado
- **Service Layer Pattern:** Para separación de lógica de negocio

## 2.2 Estructura de Directorios

```

1 cied/
2     app/                               # Aplicaci n Flask
3         __init__.py                   # Application factory
4         blueprints/                 # Rutas organizadas
5             core.py                  # Landing y syllabus
6             errors.py                # Gest i n de errores
7             quiz.py                 # Sistema de quiz
8         services/                  # L gica de negocio
9             errors_repo.py          # Repositorio de errores
10            llm_generator.py        # Generador LLM
11            quiz_week01.py          # L gica Week 01
12            mediator.py            # Sistema de remediaci n
13            weeks.py                # Framework extensible
14        templates/                # Plantillas Jinja2
15        static/                  # Recursos est ticos
16    data/errors/                 # Base de errores (JSON)
17    docs/syllabus/               # Documentaci n pedag gica
18    scripts/                    # Utilidades de deployment
19    tests/                      # Suite de pruebas
20    config.py                  # Configuraci n centralizada

```

Listing 1: Estructura del proyecto CIED

## 2.3 Tecnologías Principales

Table 1: Tecnologías del Sistema CIED

Componente	Tecnología	Versión
Backend	Python Flask	2.3.3
Base de Datos	JSON (Filesystem)	-
Frontend	HTML5 + CSS3 + MathJax	-
IA Generativa	Google Gemini API	1.x
Renderizado LaTeX	MathJax	3.x
Testing	pytest	7.4.0
Deployment	systemd + Gunicorn	-

## 3 Componentes Principales

### 3.1 Repositorio de Errores

#### 3.1.1 Estructura de Datos

Cada error se almacena como un documento JSON estructurado:

```

1 {
2     "id": "integracion-partes-eleccion-uv",
3     "curso": "MATE-1214",
4     "tema": "T cnicas de integraci n",
5     "subtema": "Integraci n por partes",
6     "titulo": "Elecci n incorrecta de uvdydv",

```

```

7     "descripcion_corta": "...",
8     "prerrequisitos": [...],
9     "sintomas": [...],
10    "patron_error": {
11      "latex": "\int_{x_0}^{x_1} f(x) dx",
12      "explicacion": ...
13    },
14    "deteccion": {
15      "tipo": "Análisis de dificultad relativa",
16      "reglas": [...]
17    },
18    "remediacion": {
19      "estrategia": "Aplicar regla LIATE",
20      "pistas": [...],
21      "mini_leccion": {
22        "latex": "...",
23        "nota": ...
24      }
25    },
26    "ejercicio_correctivo": [...],
27    "verificacion": {...},
28    "metadata": {
29      "tags": ["integración por partes", "LIATE"]
30    }
31 }

```

Listing 2: Estructura de un error en CIED

### 3.1.2 Repositorio de Errores (ErrorsRepository)

- **Carga Perezosa:** Los errores se cargan bajo demanda
- **Cache Simple:** Implementación de cache en memoria
- **Búsqueda Inteligente:** Por título, tags y contenido
- **Validación Robusta:** Manejo de errores de parsing

## 3.2 Sistema de Quiz Interactivo

### 3.2.1 Framework de Semanas (WeekSpec)

El sistema implementa un framework extensible para agregar nuevas semanas:

```

1 class WeekSpec:
2     """
3         Especificación genérica para una semana del curso.
4         Define estructura completa para implementar quiz semanal.
5     """
6
7     def __init__(
8         self,
9         week_id: str,
10        title: str,
11        subtitle: str,
12        temas: List[str],
13        técnicas: List[str],

```

```

14     descripcion: str,
15     quiz_templates: List[Dict[str, Any]],
16     error_tags: Optional[List[str]] = None
17   ):
18     # Atributos de metadata y configuración
19     pass

```

Listing 3: Clase WeekSpec - Framework Extensible

### 3.2.2 Generador LLM (LLMGenerator)

Sistema de generación dinámica de ejercicios:

- **Prompt Engineering:** Prompts específicos por técnica matemática
- **Fallback Seguro:** Si falla la IA, retorna None
- **Provider Actual:** Google Gemini (extensible a otros)
- **Configuración Segura:** API keys via variables de entorno

### 3.2.3 Política Pedagógica 50/50

```

1 # Configuración pedagógica del switch para mezclar ejercicios
2 SEED_RATIO = 0.5 # Proporción de ejercicios basados en semillas
3 LLM_RATIO = 0.5 # Proporción de ejercicios generados dinámicamente
4
5 # Validación: las proporciones deben sumar 1.0
6 assert SEED_RATIO + LLM_RATIO == 1.0

```

Listing 4: Configuración Pedagógica

## 3.3 Sistema de Remediación

Implementa lógica determinística para acciones de remediación:

```

1 def remediate(error_id: str, context: Optional[Dict] = None) -> Dict:
2     """
3         Determina acción de remediación basada en patrones de error.
4         No requiere LLMs ni bases de datos externas.
5     """
6     if "innecesaria" in error_id.lower():
7         return {
8             "action": RemediationAction.REINFORCE,
9             "reason": "Error indica falta de comprensión conceptual"
10            "b sica"
11        }
12
13    if "antiderivada" in error_id.lower() or "derivada" in error_id.
14       lower():
15        return {
16            "action": RemediationAction.RETRY,
17            "reason": "Error en cálculo algebraico, permitir reintento"
18        }
19

```

```

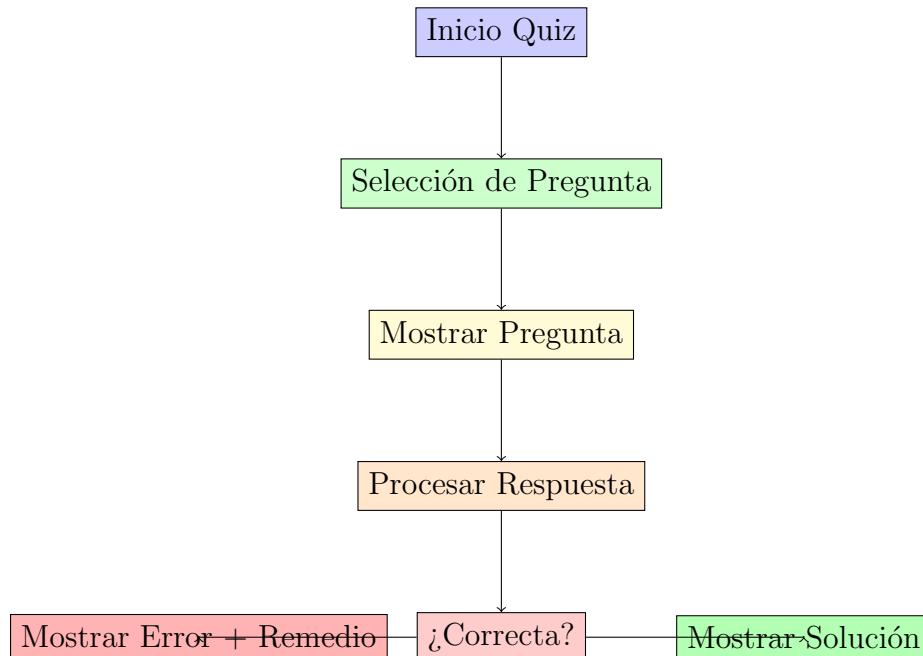
18     return {
19         "action": RemediationAction.HALT,
20         "reason": "Error requiere una lista pedagógica detallada"
21     }

```

Listing 5: Sistema de Remediación Determinístico

## 4 Flujo de Funcionamiento

### 4.1 Flujo del Quiz



### 4.2 Algoritmo de Selección de Ejercicios

- Generar Número Aleatorio:** Entre 0.0 y 1.0
- Decidir Fuente:**
  - Si  $\text{random} < \text{SEED\_RATIO}$  → Usar ejercicio seed
  - Si  $\text{random} \geq \text{SEED\_RATIO}$  → Generar con LLM
- Validar Disponibilidad:** Si LLM falla, usar seed como fallback
- Marcar Origen:** Para trazabilidad pedagógica

### 4.3 Procesamiento de Respuestas

- Validar Entrada:** Verificar que se seleccionó opción válida
- Comparar Respuesta:** Verificar si coincide con opción correcta
- Generar Retroalimentación:**

- Correcta: Mostrar procedimiento completo
  - Incorrecta: Mostrar error específico + explicación
4. **Acción de Remediación:** Basada en tipo de error cometido

## 5 API y Endpoints

### 5.1 Endpoints Principales

Table 2: API Endpoints del Sistema CIED

Método	Endpoint	Descripción	Respuesta
GET	/	Página principal	HTML
GET	/syllabus	Syllabus del curso	HTML
GET	/health	Health check	JSON
GET	/quiz/week01	Quiz Semana 1	HTML
POST	/quiz/week01	Procesar respuesta	HTML
GET	/errors	Lista errores	JSON
GET	/errors/{id}	Error específico	JSON
GET	/errors/search?q={query}	Búsqueda de errores	JSON

### 5.2 Ejemplo de Uso de la API

```
1 curl http://localhost:8082/health
```

Listing 6: Consultar Health Check

```
1 curl "http://localhost:8082/errors/search?q=integracion"
```

Listing 7: Buscar errores relacionados con integración

## 6 Configuración y Deployment

### 6.1 Variables de Entorno

Table 3: Variables de Entorno Principales

Variable	Descripción	Default	Requerida
FLASK_ENV	Entorno de ejecución	development	No
GEMINI_API_KEY	API Key de Gemini	-	Sí (para LLM)
HOST	IP de binding	0.0.0.0	No
PORT	Puerto de escucha	8082	No
SECRET_KEY	Clave secreta Flask	dev-key	No
LOG_LEVEL	Nivel de logging	INFO	No

## 6.2 Deployment en Producción

El sistema está configurado para deployment con systemd:

- **Servidor WSGI:** Gunicorn para producción
- **Servicio Systemd:** Configurado en `cied.service`
- **Logging:** Configurado para rotación automática
- **Monitorización:** Endpoint `/health` para health checks

# 7 Escalabilidad y Extensión

## 7.1 Agregar Nueva Semana

El framework permite extender fácilmente a nuevas semanas:

1. Crear `quiz_week02.py` con templates de ejercicios
2. Configurar `WeekSpec` con metadata de la semana
3. Agregar rutas en blueprint `quiz`: `/quiz/week02`
4. Crear template HTML: `quiz_week02.html`
5. Registrar en sistema via `register_week()`

## 7.2 Extensión de Proveedores LLM

El sistema es extensible a múltiples proveedores de IA:

```

1 def _get_llm_model():
2     """Factory method para obtener modelo LLM disponible."""
3     # Intentar Gemini primero
4     if os.environ.get("GEMINI_API_KEY"):
5         return _get_gemini_model()
6
7     # Intentar OpenAI como fallback
8     if os.environ.get("OPENAI_API_KEY"):
9         return _get_openai_model()
10
11    # Intentar DeepSeek como ltimo recurso
12    if os.environ.get("DEEPSEEK_API_KEY"):
13        return _get_deepseek_model()
14
15    return None # Sin LLM disponible

```

Listing 8: Extensión a Múltiples Proveedores LLM

## 8 Consideraciones de Seguridad

### 8.1 Protección de API Keys

- **Variables de Entorno:** Nunca hardcodear claves en código
- **Archivo .env:** No versionar, incluir en .gitignore
- **Validación:** Verificar existencia antes de usar APIs
- **Fallback Seguro:** Funcionalidad completa sin LLM si falla

### 8.2 Validación de Entrada

- **Sanitización:** Validar todas las entradas de usuario
- **Límite de Tamaño:** Controlar tamaño de payloads
- **Tipo de Datos:** Verificar tipos de datos esperados
- **XSS Protection:** Escapar contenido dinámico en templates

## 9 Testing y Calidad

### 9.1 Suite de Pruebas

- **Framework:** pytest con cobertura
- **Pruebas Unitarias:** Para servicios y utilidades
- **Pruebas de Integración:** Para rutas y API
- **Validación de Datos:** Para estructura de errores JSON

### 9.2 Validación Pedagógica

Scripts específicos para validar contenido educativo:

- **validate\_week01.py:** Verifica integridad de Week 01
- **Verificación de Templates:** Consistencia de ejercicios
- **Validación de Errores:** Estructura JSON correcta

## 10 Conclusión

### 10.1 Logros Arquitectónicos

El sistema CIED demuestra una arquitectura robusta y extensible:

- **Modularidad:** Separación clara de responsabilidades

- **Escalabilidad:** Framework extensible para nuevas semanas
- **Resiliencia:** Funcionamiento completo sin dependencias externas
- **Seguridad:** Protección de credenciales y validación de entrada
- **Mantenibilidad:** Código bien estructurado y documentado

## 10.2 Ventajas Pedagógicas

- **Personalización:** Generación adaptativa de ejercicios
- **Retroalimentación Específica:** Errores catalogados con remedios
- **Aprendizaje Activo:** Interacción inmediata con corrección
- **Escalabilidad:** Fácil adición de nuevo contenido

## 10.3 Futuras Extensiones

- **Múltiples Proveedores LLM:** OpenAI, DeepSeek, Anthropic
- **Análisis de Aprendizaje:** Tracking de progreso estudiantil
- **Gamificación:** Sistema de puntos y logros
- **Multi-idioma:** Soporte para español e inglés
- **Móvil:** Aplicación responsive y PWA

El sistema CIED representa un ejemplo exitoso de integración de tecnología educativa moderna con principios de arquitectura de software sólida, ofreciendo una plataforma escalable para el aprendizaje de matemáticas avanzadas.