UNIVERSIDAD LAICA ELOY ALFARO DE MANABÍ

Facultad de Ciencias de la Vida y Tecnologías

Carrera de Software

Trabajo Autónomo

Segundo Parcial

Docente: **John Cevallos**

Asignatura: **Aplicación para el Servidor Web**

Nivel: **Quinto**

Objetivo General

Extender el sistema desarrollado en el primer parcial mediante la implementación de una arquitectura de microservicios robusta que integre autenticación centralizada, pasarelas de pago con webhooks, inteligencia artificial conversacional mediante MCP (Model Context Protocol), y orquestación de eventos con n8n. El proyecto debe demostrar competencias avanzadas en desarrollo distribuido, integración de sistemas externos, interoperabilidad entre equipos y automatización de procesos de negocio.

Condiciones y Reglas

1. Generales

- Se mantienen los grupos de 3 integrantes conformados en el primer parcial.
- El proyecto del primer parcial debe mantenerse como base. Los componentes REST, GraphQL, WebSocket y Frontend desarrollados previamente deben integrarse con los nuevos pilares.
- Cada integrante puede utilizar el lenguaje de programación de su elección para implementar los nuevos microservicios, siempre que se cumplan los requisitos arquitectónicos.
- Las arquitecturas aplicadas deben sustentarse en documentación verificable (artículos, libros o páginas oficiales).
- El código debe subirse al mismo repositorio grupal en GitHub utilizado en el primer parcial.
- Cada grupo debe coordinarse con AL MENOS otro grupo para la integración de webhooks bidireccionales (requisito del Pilar 2).
- No se requiere publicación de la solución en hosting, pero debe ser demostrable localmente.

2. Arquitectura de la Aplicación Requerida: Los 4 Pilares

El sistema debe implementar obligatoriamente los siguientes cuatro pilares arquitectónicos que extienden la funcionalidad del primer parcial:

Pilar 1: Microservicio de Autenticación (15%)

Objetivo: Separar la autenticación en un servicio independiente, evitando el antipatrón de llamadas constantes al servicio de autenticación en cada request.

Componentes requeridos:

1. **Auth Service independiente:** Microservicio dedicado exclusivamente a la gestión de autenticación.
2. **JWT con access y refresh tokens:** Implementación de tokens de acceso de corta duración y tokens de renovación.
3. **Validación local:** Los demás servicios deben validar tokens localmente (verificando firma y expiración) sin consultar al Auth Service en cada petición.
4. **Base de datos propia:** Tablas para usuarios, refresh tokens, y tokens revocados.
5. **Seguridad:** Rate limiting en login, blacklist de tokens revocados.

Endpoints mínimos: POST /auth/register, POST /auth/login, POST /auth/logout, POST /auth/refresh, GET /auth/me, GET /auth/validate (interno).

Pilar 2: Webhooks e Interoperabilidad B2B (20%)

Objetivo: Implementar un sistema de pagos con abstracción de pasarela y comunicación bidireccional entre grupos mediante webhooks, simulando integración empresarial real.

Componentes requeridos:

1. **Payment Service Wrapper:** Microservicio que abstrae la pasarela de pago mediante el patrón Adapter.
    - Interface PaymentProvider abstracta que define el contrato.
    - Adapters implementados: StripeAdapter, MercadoPagoAdapter (opcional), y MockAdapter (obligatorio para desarrollo).
    - Normalización de webhooks de diferentes pasarelas a un formato común.
2. **Registro de Partners:** API para que otros grupos registren sus webhooks.
    - POST /partners/register - Registrar URL de webhook + eventos suscritos.
    - Generación de secret compartido para firma HMAC.
3. **Autenticación HMAC:** Todos los webhooks deben firmarse y verificarse usando HMAC-SHA256.
4. **Eventos bidireccionales:** Comunicación en ambas direcciones con el grupo partner.

Requisito de integración: Cada grupo debe coordinarse con al menos otro grupo para implementar webhooks bidireccionales. Por ejemplo: Grupo A (Hotel) notifica reserva confirmada → Grupo B (Tours) ofrece paquetes relacionados → Grupo B notifica tour comprado → Grupo A actualiza itinerario del huésped.

Contrato de eventos sugeridos: booking.confirmed, payment.success, order.created, service.activated, tour.purchased, etc.

Pilar 3: MCP - Chatbot Multimodal con IA (20%)

Objetivo: Implementar un asistente de inteligencia artificial conversacional que procese diferentes tipos de entrada (texto, imágenes, PDFs) y ejecute acciones de negocio mediante herramientas MCP.

Componentes requeridos:

1. **AI Orchestrator:** Microservicio que orquesta las interacciones con el modelo de lenguaje.
2. **LLM Adapter abstracto:** Interface que permite intercambiar proveedores de IA (Gemini, OpenAI, etc.) sin modificar la lógica de negocio. Debe implementarse usando el patrón Strategy.
3. **MCP Server con Tools:** Servidor de herramientas que el modelo de IA puede invocar.
4. **Chat UI:** Interfaz de chat en el frontend O integración con Telegram/WhatsApp via n8n.

Entradas multimodales (mínimo 2 tipos):

- **Texto (obligatorio):** Comandos y consultas en lenguaje natural.
- **Imagen:** OCR de documentos, clasificación de productos, análisis de fotos.
- **PDF:** Extracción de datos de facturas, contratos, catálogos.
- **Audio (bonus):** Transcripción y procesamiento de notas de voz.

MCP Tools mínimos (5 herramientas):

- 2 tools de consulta (ej: buscar_productos, ver_reserva, obtener_cliente).
- 2 tools de acción (ej: crear_reserva, registrar_cliente, procesar_pago).
- 1 tool de reporte (ej: resumen_ventas, estadisticas_diarias).

Pilar 4: n8n - Event Bus (15%)

Objetivo: Centralizar la orquestación de eventos externos del sistema utilizando n8n como Event Bus visual.

Principio fundamental: *"Todo evento externo pasa por n8n"*

Workflows obligatorios:

1. **Payment Handler:** Recibe webhook de pasarela de pago → Valida payload → Activa servicio/reserva → Notifica via WebSocket → Envía email de confirmación → Dispara webhook al grupo partner.
2. **Partner Handler:** Recibe webhook de grupo partner → Verifica firma HMAC → Procesa según tipo de evento → Ejecuta acción de negocio → Responde ACK.
3. **MCP Input Handler (opcional si hay Chat UI):** Recibe mensaje de Telegram/Email → Extrae contenido y adjuntos → Envía a AI Orchestrator → Responde por el mismo canal.
4. **Scheduled Tasks:** Cron job que ejecuta tarea programada (reporte diario, limpieza de datos, recordatorios, health checks).

3. Integración de Tecnologías

Requisito crítico: La integración entre los 4 pilares y los componentes del primer parcial es un parámetro fundamental de evaluación. El sistema debe demostrar:

- **Comunicación fluida:** Los microservicios deben comunicarse correctamente a través del API Gateway.
- **Consistencia de datos:** Las operaciones deben mantener integridad entre servicios.
- **Flujo end-to-end:** Demostrar al menos un flujo completo desde entrada de usuario hasta notificación final.
- **Integración con P1:** El Core REST, GraphQL y WebSocket del primer parcial deben seguir funcionando e integrarse con los nuevos pilares.
- **Interoperabilidad B2B:** Webhooks funcionales con al menos otro grupo.

4. Funcionalidades Mínimas Requeridas

1. Sistema de autenticación completo con JWT y refresh tokens.
2. Pasarela de pago funcional (puede usar MockAdapter para desarrollo).
3. Webhooks bidireccionales con grupo partner operativos.
4. Chatbot multimodal que procese al menos 2 tipos de entrada.
5. 5 MCP Tools funcionales integrados con el negocio.
6. 1 workflow de n8n operativo.
7. Notificaciones en tiempo real via WebSocket.
8. Al menos una tarea programada (cron job) ejecutándose.
9. Dashboard actualizado con los nuevos módulos.
10. Manejo de errores estructurado en todos los servicios.

5. Documentación

1. **README.md actualizado:** Descripción del proyecto, arquitectura completa incluyendo nuevos pilares, diagrama de componentes.
2. **Instrucciones de instalación:** Pasos claros para levantar todos los servicios (preferiblemente con docker-compose).
3. **Documentación de APIs:** Endpoints de Auth Service, Payment Service, AI Orchestrator.
4. **Guía de integración para partners:** Cómo registrar webhooks, formato de eventos, ejemplos de payload.

5. **Documentación de MCP Tools:** Descripción de cada herramienta, parámetros, ejemplos de uso.
6. **Workflows de n8n exportados:** Archivos JSON de los workflows con descripción de cada uno.

Rúbrica Técnica y de Soft Skills (100%)

Categoría	Peso	Detalle de Evaluación
Pilar 1: Auth Service	**15%**	JWT funcional (5%), Refresh tokens (3%), Validación local (4%), Seguridad (3%)
Pilar 2: Payment + Webhooks B2B	**20%**	Wrapper + Adapters (8%), Registro partners (4%), HMAC (4%), Bidireccional (4%)
Pilar 3: MCP Chatbot Multimodal	**20%**	LLM Adapter abstracto (5%), 5 Tools (6%), Entradas multimodales (6%), Logs (3%)
Pilar 4: n8n Event Bus	**15%**	Payment Handler (5%), Partner Handler (4%), MCP Input (3%), Scheduled Task (3%)
Integración con P1	**10%**	Comunicación entre servicios (4%), Consistencia datos (3%), Flujo E2E (3%)
Frontend Extendido	**5%**	Nuevos módulos (Chat, Pagos) (3%), UX integrada (2%)
Documentación	**5%**	README completo (2%), API docs (2%), Partner guide (1%)
Trabajo Colaborativo	**5%**	Commits semanales (2%), Distribución equitativa (2%), Comunicación (1%)
Presentación y Demo	**5%**	Demo funcional (3%), Explicación arquitectura (2%)
TOTAL	**100%**	

Factor de Participación Individual

El aporte individual se evaluará mediante los commits semanales realizados al repositorio grupal durante las 5 semanas de desarrollo del segundo parcial.

Cada semana representa un 20% del factor individual, por lo que mantener actividad constante equivale al 100% del aporte individual.

Criterios de evaluación:

- Commits significativos y relacionados con las tareas asignadas.
- Constancia semanal en los aportes.
- Calidad del código y claridad en los mensajes de commit.
- Participación activa en la integración de los 4 pilares.

Ejemplo:

Si la nota base del proyecto es 8.5 y el estudiante solo registra commits en 4 de las 5 semanas, su factor será del 80%. **Nota final: 8.5 × 0.8 = 6.8**

Cronograma de Desarrollo Sugerido

Semana	Actividades Principales
Semana 1	**Setup y Auth Service:** Configuración de microservicios, implementación completa del Auth Service (JWT + refresh tokens), setup de n8n, coordinación con grupo partner.
Semana 2	**Payment Service:** Payment Wrapper completo con MockAdapter, estructura del AI Orchestrator, primeros MCP tools, registro de partner.
Semana 3	**MCP y n8n:** AI Orchestrator completo con LLM Adapter, 5 MCP tools funcionales, workflows de n8n para pagos y partners, pruebas de webhooks con partner.
Semana 4	**Integración:** Webhooks bidireccionales con partner funcionando, entradas multimodales en chatbot, frontend extendido (Chat UI, módulo de pagos), testing de integración.
Semana 5	**Presentación:** Refinamiento final, documentación completa, preparación de demo, ensayo de presentación oral, demo conjunta con grupo partner.

Recomendaciones Técnicas

Tecnologías Sugeridas

- **Auth Service:** NestJS con Passport, FastAPI con python-jose, o Express con jsonwebtoken.
- **Payment Service:** NestJS o FastAPI con SDKs oficiales de Stripe/MercadoPago.
- **AI Orchestrator:** Python con LangChain, TypeScript con Vercel AI SDK, o implementación directa con APIs de Gemini/OpenAI.
- **MCP Server:** Implementación según especificación MCP o adaptación propia del patrón tool-calling.
- **n8n:** Instalación via Docker (recomendado) o npm.
- **Base de datos:** PostgreSQL (recomendado), MySQL o MongoDB según el modelo de datos.
- **Frontend:** Angular, React o Vue con la librería de UI utilizada en P1.

Patrones de Diseño Recomendados

- **Adapter Pattern:** Para Payment Providers y LLM Providers.
- **Strategy Pattern:** Para intercambiar implementaciones de LLM.
- **Factory Pattern:** Para instanciar el provider correcto según configuración.
- **Observer Pattern:** Para el sistema de eventos y webhooks.

Gestión del Proyecto

- Utilizar GitHub Projects, Issues y Milestones para organización.
- Crear branches por feature y usar Pull Requests para revisión de código.
- Documentar decisiones arquitectónicas en el README o en un archivo ARCHITECTURE.md.
- Mantener comunicación constante con el grupo partner para la integración de webhooks.

Entrega Final - Semana 15

Componentes de la Entrega

1. **Repositorio GitHub:** Código completo de todos los microservicios y componentes.
2. **README.md actualizado:** Documentación técnica completa incluyendo arquitectura de los 4 pilares.
3. **Workflows de n8n:** Archivos JSON exportados de los 4 workflows obligatorios.
4. **Guía de integración:** Documento para partners explicando cómo integrar webhooks.
5. **Presentación oral:** 20-25 minutos con demo en vivo de los 4 pilares funcionando.
6. **Demo conjunta:** Demostración de webhooks bidireccionales con el grupo partner.

Criterios de Aceptación

1. Sistema completo y funcional con los 4 pilares operativos.
2. Auth Service funcionando con JWT y refresh tokens.
3. Payment Service procesando pagos (al menos con MockAdapter).
4. Webhooks bidireccionales funcionando con grupo partner.
5. Chatbot multimodal procesando al menos 2 tipos de entrada.
6. 5 MCP Tools ejecutando acciones reales en el sistema.
7. 4 workflows de n8n operativos.
8. Frontend integrado con los nuevos módulos (Chat UI, Pagos).
9. Código bien estructurado y documentado.
10. Participación equitativa de los integrantes (evidenciada en commits).
11. Demostración exitosa de un flujo end-to-end completo.

Nota importante: La integración con el grupo partner es un requisito obligatorio. Coordinen tempranamente para definir el contrato de eventos y realizar pruebas de integración antes de la presentación final.