




HISTORIAS DE USUARIOS

Nombre de la HU:		Integración y creación de APIs con Node
Objetivo de la HU		Como inversionista, quiero crear una aplicación que consuma APIs externas de cualquier tema que deseen implementar, usando axios y que ejecute cronjobs para enviar emails de notificación de interacciones, con el fin de aprender a integrar servicios externos y automatizar procesos. Aceptando ideas creativas e innovadoras.
TASK1	Configurar el entorno de desarrollo (Node.js + editor).	
	<ul style="list-style-type: none">• Crear un proyecto base con <code>npm init -y</code>.• Agregar el tsconfig con <code>tsc --ini</code>• Modificar el tsconfig para uso de node• Instalar dependencias iniciales: express, axios, node-cron, nodemailer• Instalar dependencias de desarrollo con <code>@types/<librería></code>	
TASK2	Crear la estructura base de la app con Express.	
	<ul style="list-style-type: none">• Crear un archivo principal <code>app.js</code> o <code>index.js</code>.• Configurar el servidor Express en un puerto (ej. 3000).• Crear la carpeta <code>/routes</code> para manejar endpoints.• Probar el servidor con un endpoint básico (<code>GET /ping</code>).	
TASK3	Integrar consumo de APIs externas con axios.	
	<ul style="list-style-type: none">• Crear un servicio en <code>/services/apiService.js</code>.• Implementar un método que realice una petición GET a una API pública (ej. clima, criptomonedas, noticias).	

	 Exponer un endpoint en Express (/api/data) que retorne la información procesada de la API.
TAS	Configurar cronjobs para envío automático de emails
	<ul style="list-style-type: none"> • Configurar node-cron en el proyecto. • Implementar una tarea que se ejecute cada día a una hora definida (ej. 8:00 AM). • Usar nodemailer para enviar un correo con el resumen de los datos obtenidos desde la API externa. • Colocar la lógica de envío en /services/emailService.js.
TAS	Manejo de errores y logs.
	<ul style="list-style-type: none"> • Usar try/catch en llamadas axios y envío de emails. • Mostrar mensajes amigables cuando ocurra un error. • Agregar logs con console.log o una librería (ej. winston) para registrar las ejecuciones de cronjobs.
TAS	Pruebas básicas y validación.
	<ul style="list-style-type: none"> • Probar el endpoint /api/data en Postman o el navegador. • Verificar que el cronjob dispare el envío de emails correctamente. • Validar que la aplicación sigue corriendo sin interrumpirse ante errores de red o fallos de correo
	Criterios de aceptación. <ul style="list-style-type: none"> • El servidor Express se levanta sin errores. • El endpoint /api/data devuelve información real desde una API externa. • El cronjob envía un correo automáticamente en el horario configurado. • La aplicación maneja errores sin detenerse y muestra mensajes claros. • La estructura está organizada en rutas y servicios, siguiendo buenas prácticas.
	Story Points: 20



Actividad de cierre:

Al finalizar esta historia de usuario, serás capaz de construir una aplicación con Express que consume APIs externas y automatiza procesos con cronjobs de envío de emails. Esto te permitirá dominar integraciones en Node.js y sentar bases sólidas para proyectos más complejos que incluyan automatización, notificaciones y consumo de servicios externos.



P – J – H - B	B	F
S1 - TASK	S1 - TASK	S1 – TASK
S2 – SPIKE - RETORSPECTIVE	S2 - TASK	S2 – TASK
S3 – HU - CALIFICABLE	S3 – SPIKE RETRO	S3 – SPIKE RETRO
	S4 - TASK	S4 – TASK
	S5 – HU - NOTA	S5 – SPIKE RETRO
		S5 – TASK
		S7 – HU - NOTA

critério	recuerda	comprende	practica	analiza	Evalua
Tarea 1	2	3	7	8	10
Tarea 2					



www.riwi.io



301 732 53 27



Cl. 16 # 55 - 129