



Escuela Politécnica
Nacional

UN GESTOR DE TAREAS BASADO EN KANBAN PARA LA PLANIFICACIÓN OPERATIVA

Desarrollo del Backend

Bryan Marcelo Tapia Vergara



bryan.tapia03@epn.edu.ec



CONTENIDO

01

Contexto y objetivos

02

Metodología

03

Diseño

04

Implementación

05

Gestión

06

Resultados, conclusiones y recomendaciones

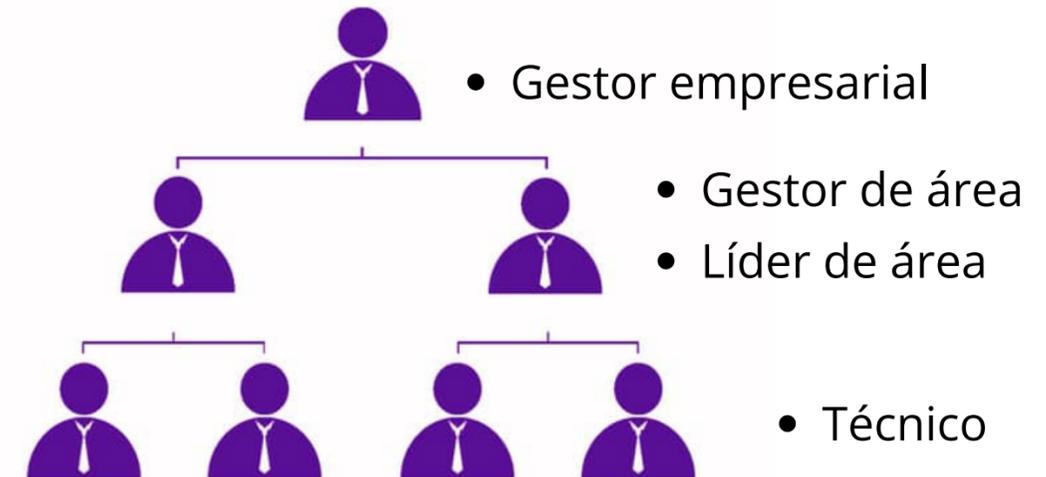
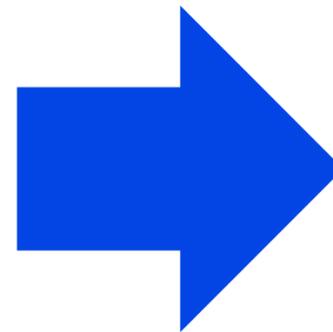
CONTEXTO Y OBJETIVOS

CONTEXTO



PROBLEMÁTICA

- Gestión operativa.
- Gestionar tareas es un reto.
- Varias herramientas.



SOLUCIÓN

- Sistema de gestión de tareas.
- Método Kanban.
- Seguimiento y control.
- Lógica de negocio en API.

OBJETIVOS

Objetivo general: Desarrollar una API que permita exponer las reglas de negocio de un sistema para gestión de tareas basado en Kanban.



Objetivo específico 1

Definir el contrato de la API de acuerdo con las necesidades del Front-End.



Objetivo específico 2

Diseñar los endpoints a través de una arquitectura REST.



Objetivo específico 3

Implementar y exponer la lógica de negocio a través de una API.

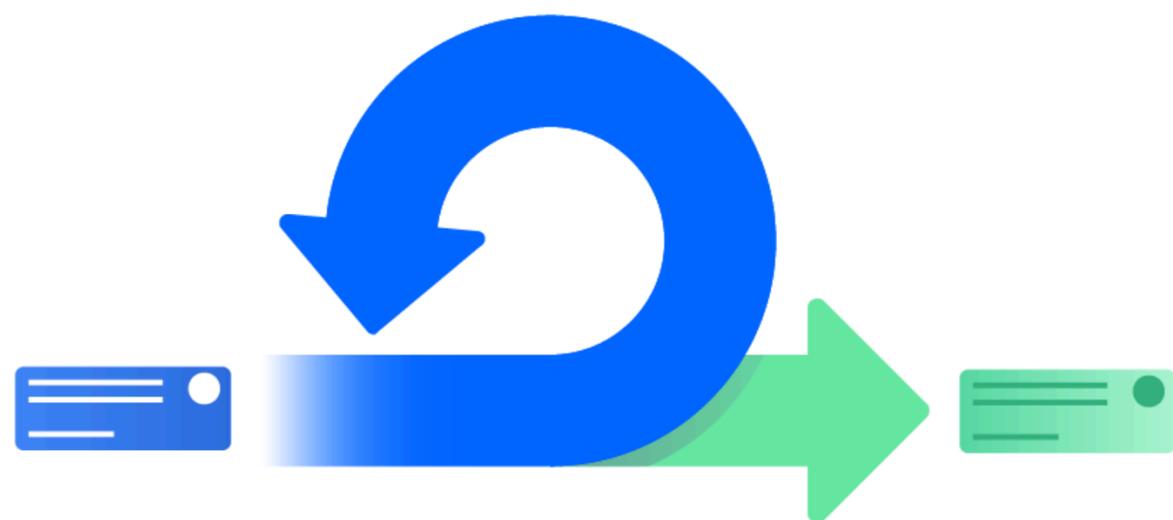
METODOLOGÍA

ENFOQUES

01

SCRUM

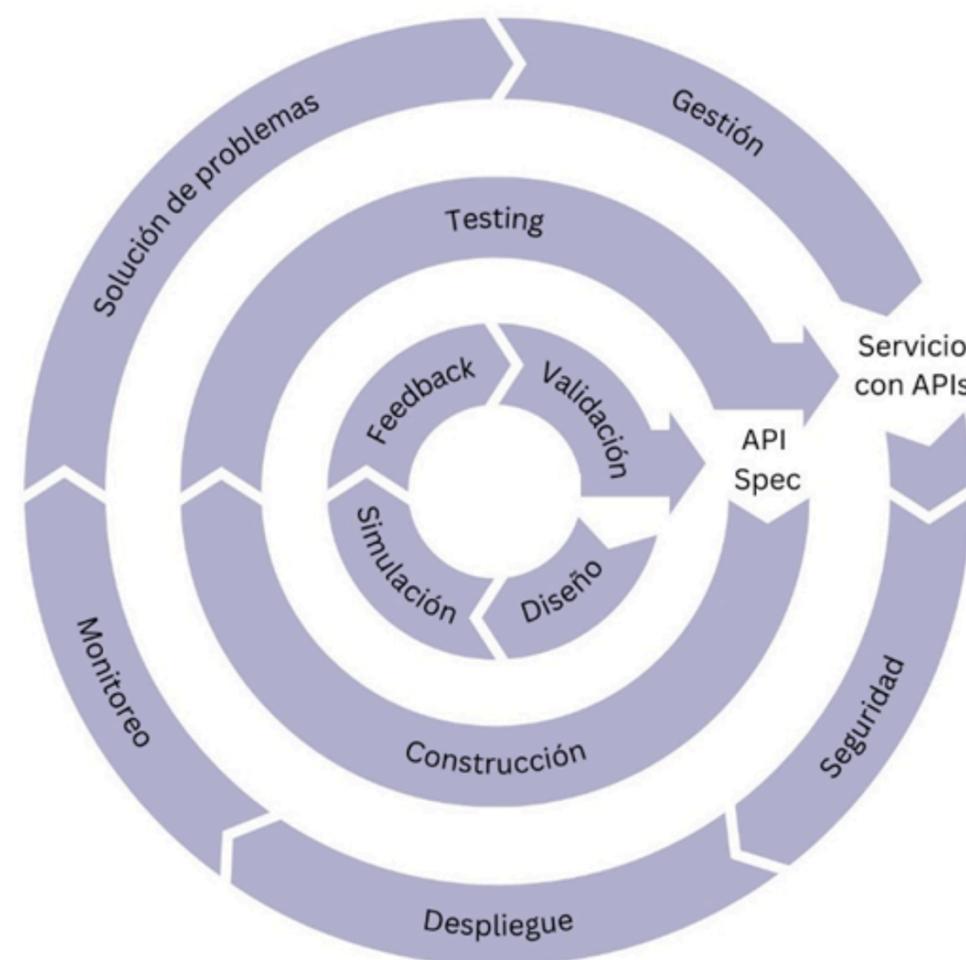
- Permite la gestión del proyecto.
- 14 Sprints.
- 15 historias de usuario para el componente.



02

SPEC-DRIVEN DEVELOPMENT

- Guía proceso de desarrollo.
- Sigue principio API-First a través de especificaciones.
- Estructura centrada en fases del proceso de desarrollo.



07

DISEÑO

DISEÑO



Estructura de la API basada en HUs.

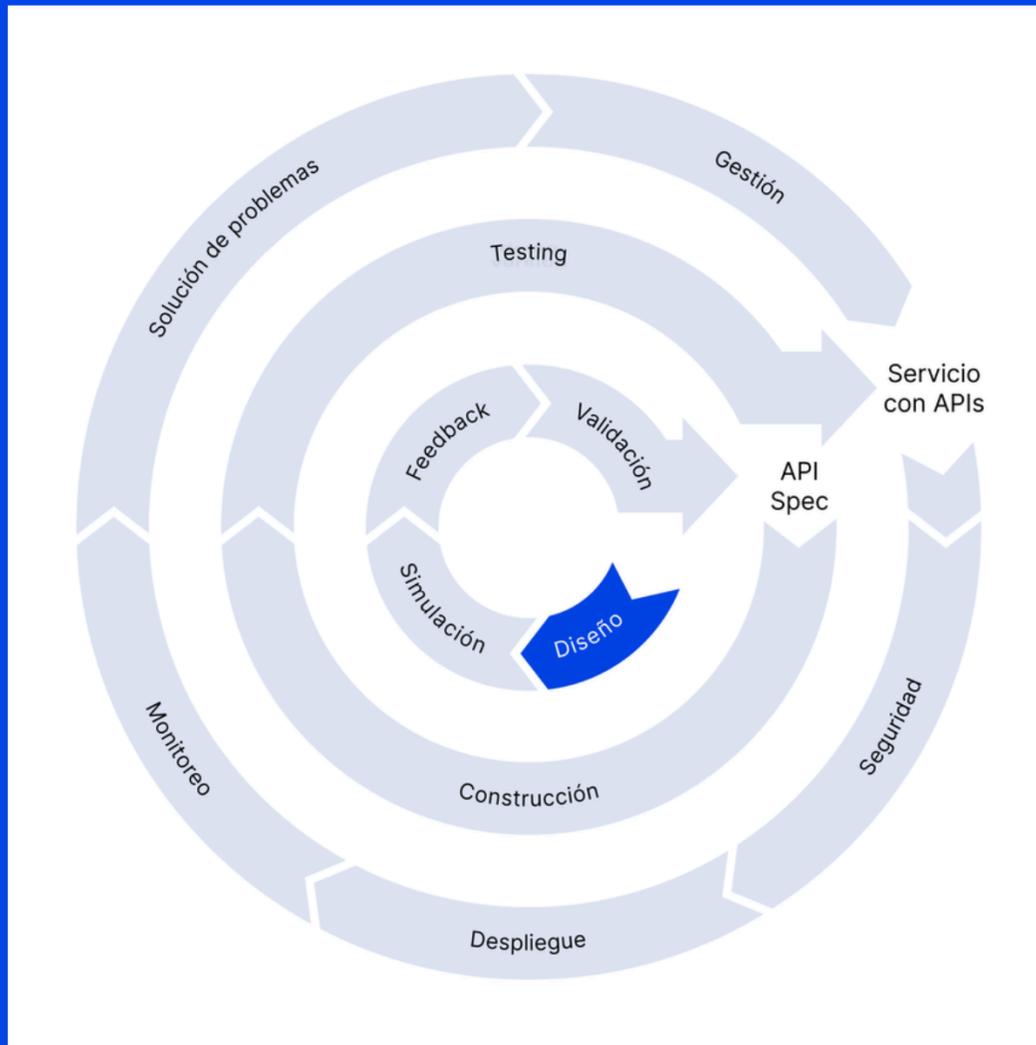
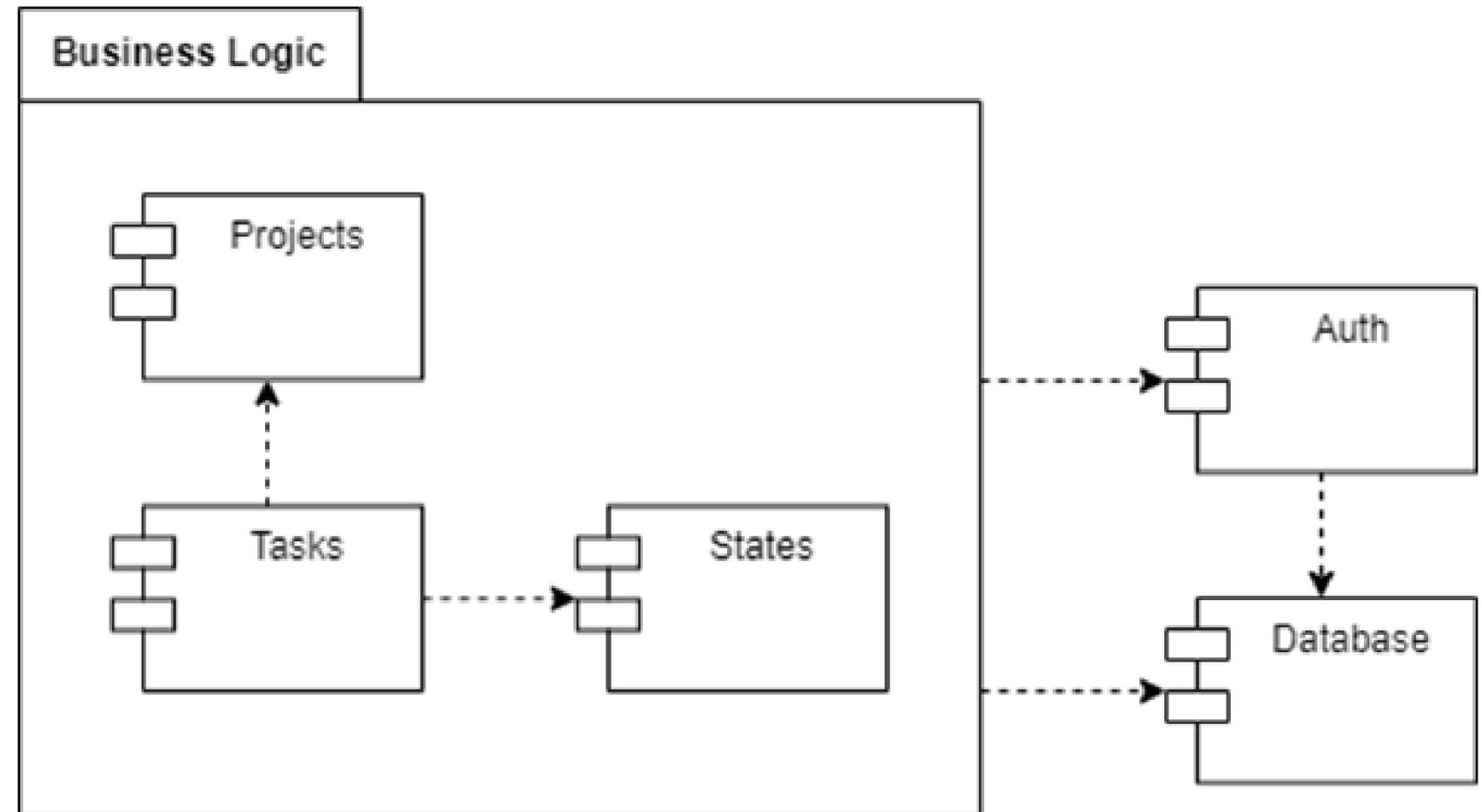


Diagrama de componentes



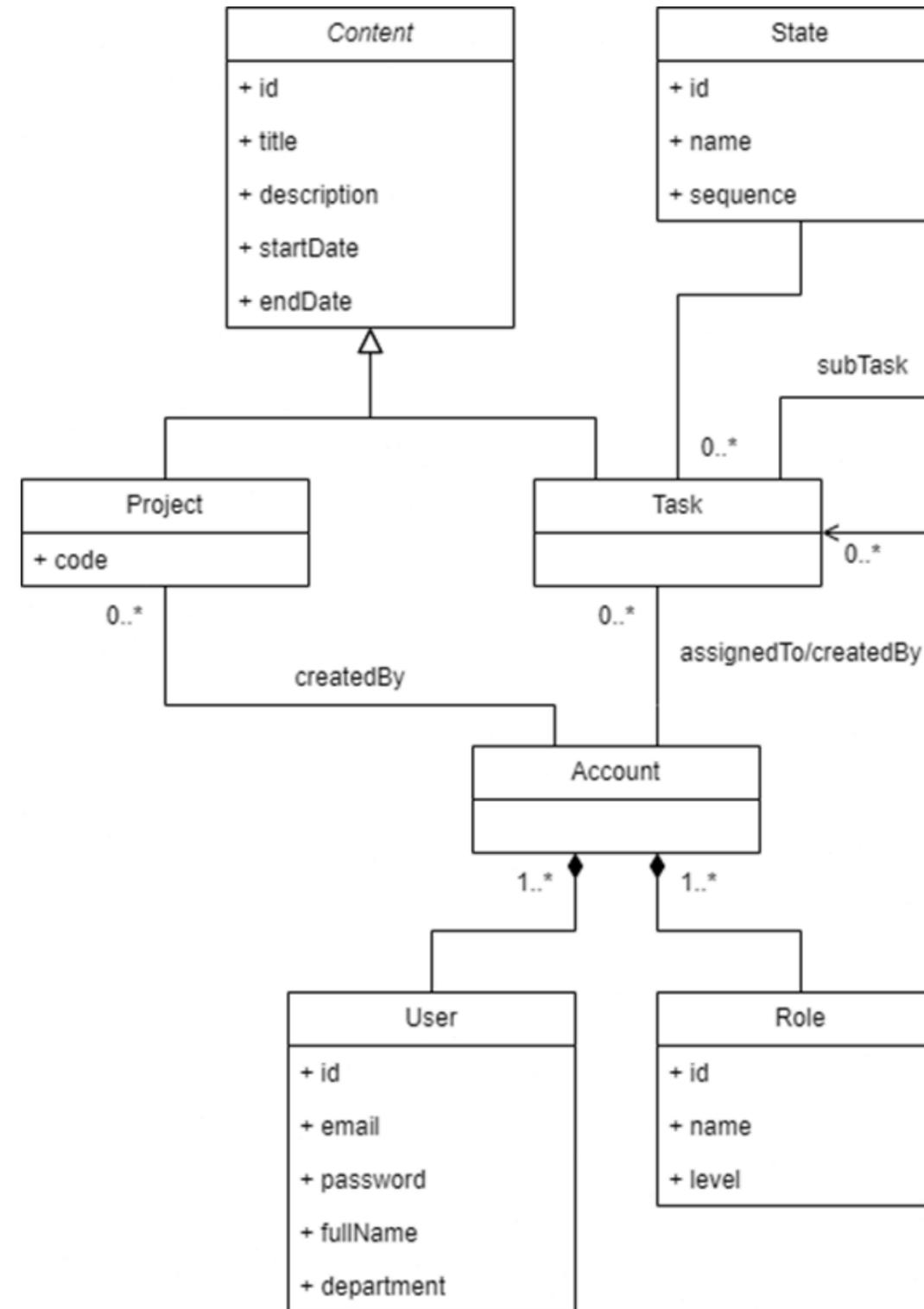
DISEÑO

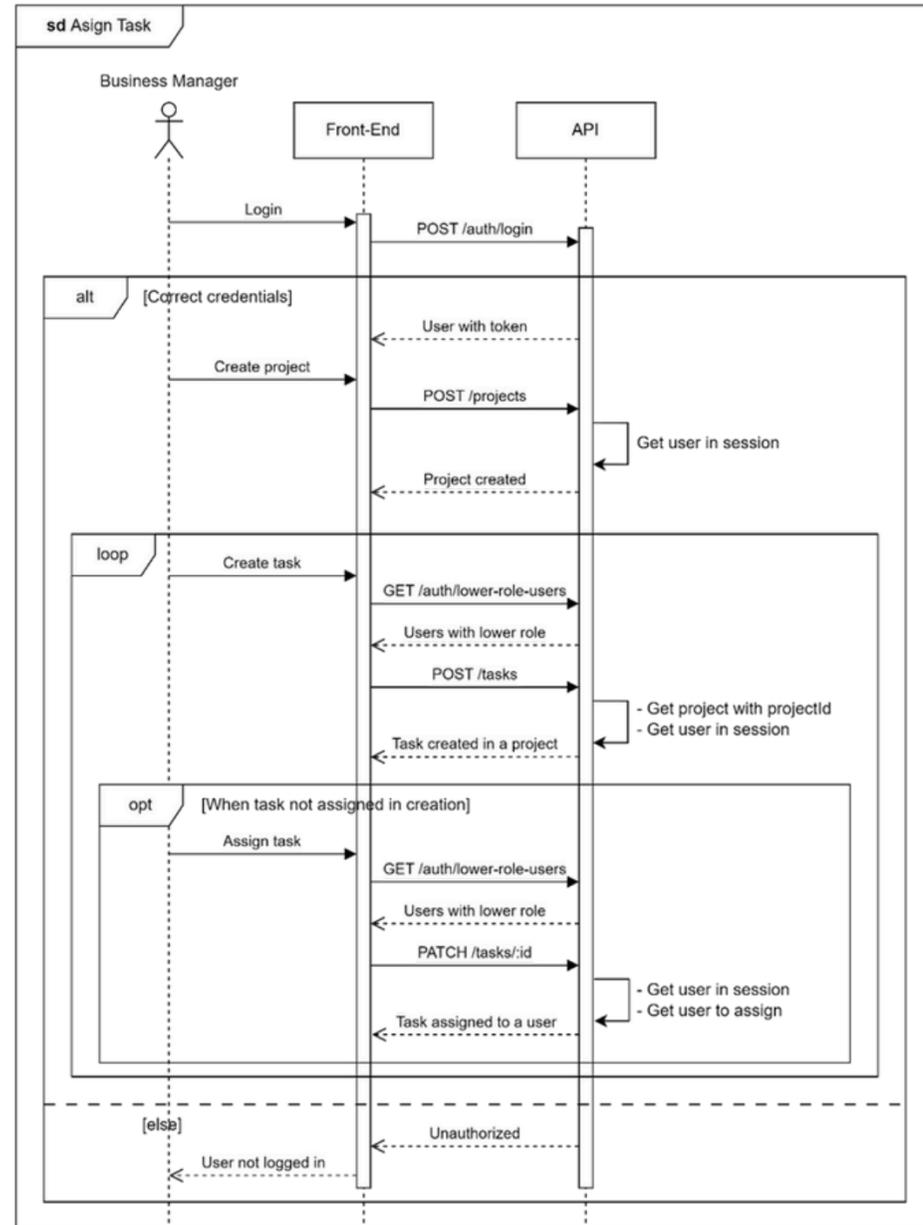


Estructura de la API con entidades y atributos.

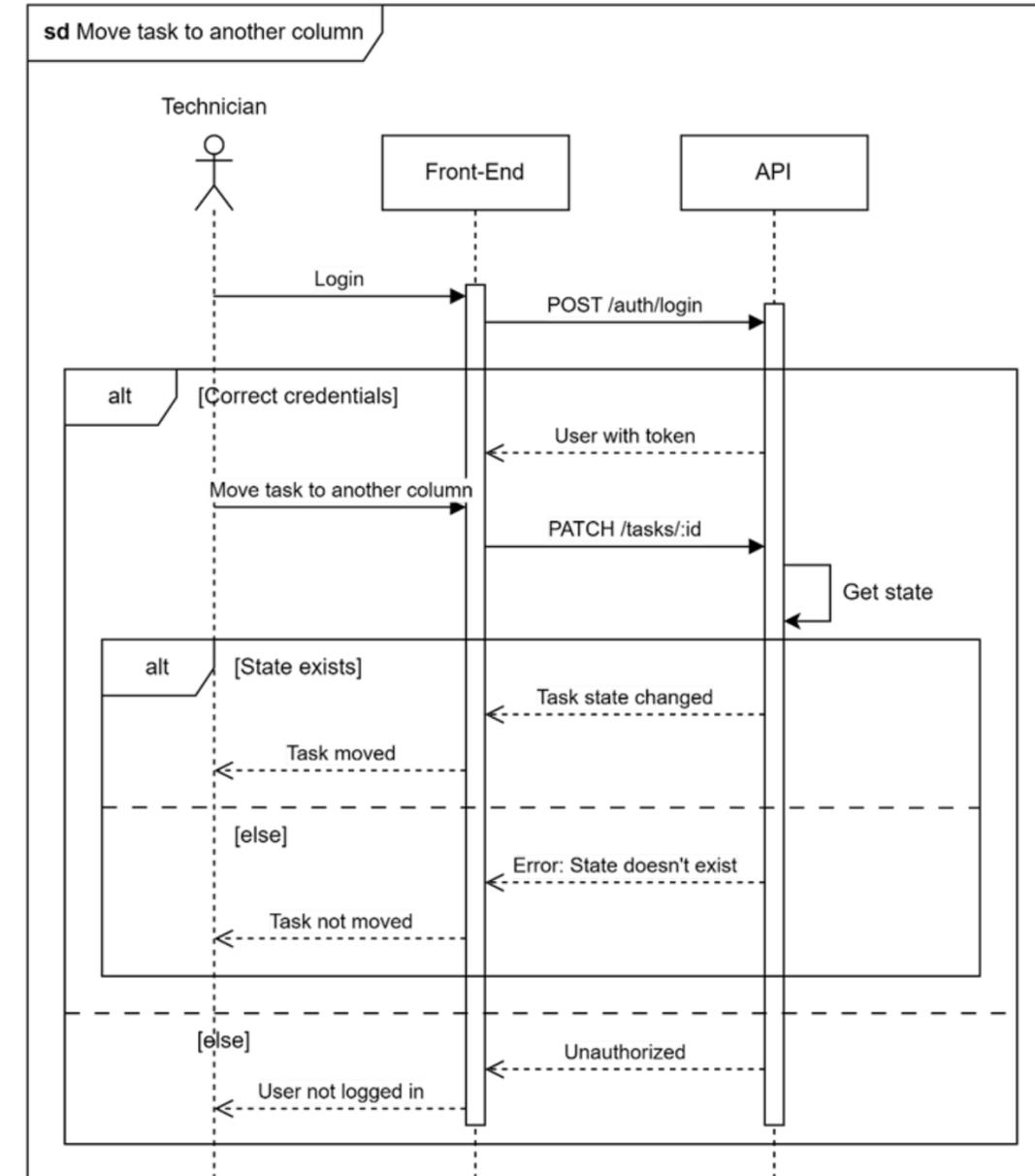


Diagrama de clases de dominio





01 Diagrama de secuencia para la asignación de una tarea

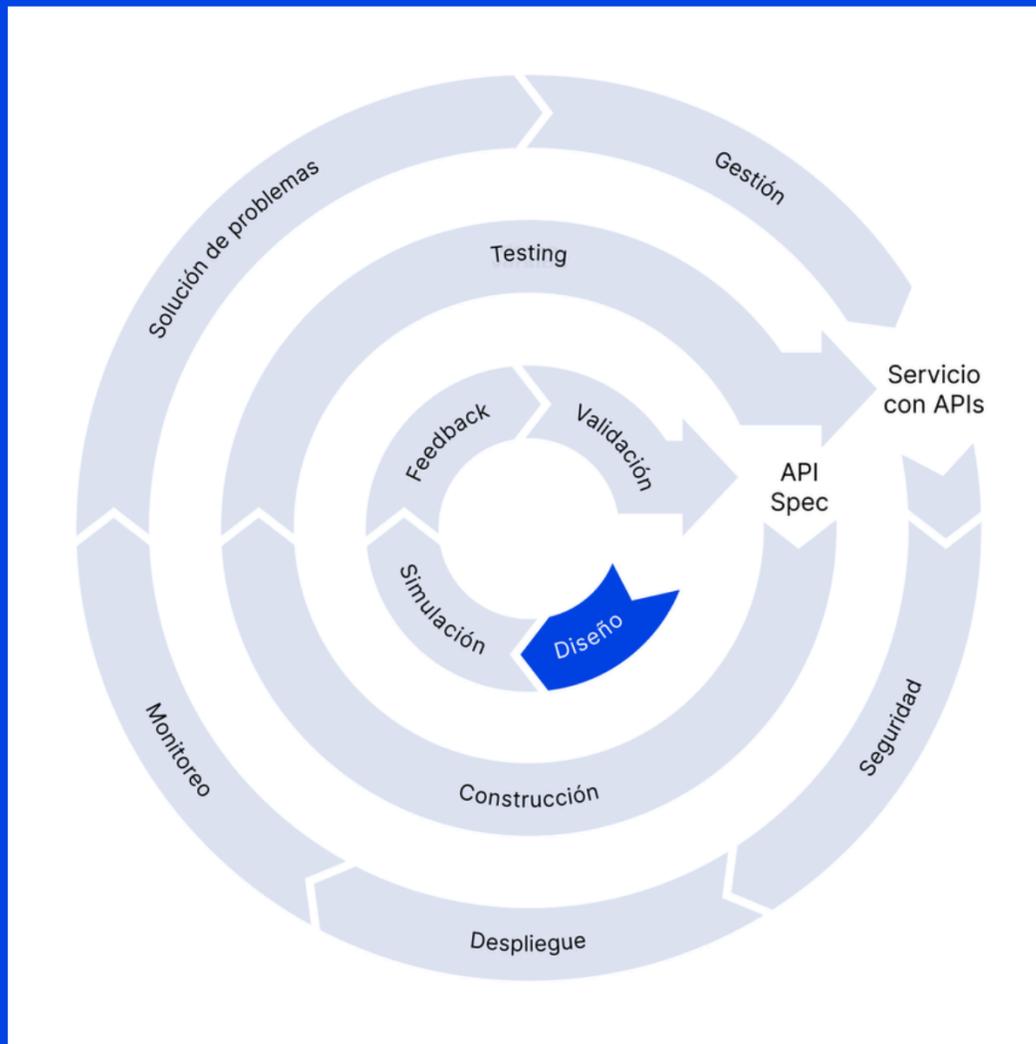


02 Diagrama de secuencia para el cambio de columna de una tarea

DISEÑO



Especificaciones se documentan en contrato de la API.



Especificaciones de la API

The screenshot shows an OpenAPI 3.0 editor interface with a sidebar on the left and a code editor on the right. The sidebar lists the structure of the API specification, and the code editor shows the corresponding YAML code. Red arrows point from specific parts of the code to labels on the right side of the image.

```
1  openapi: '3.0.0'
2  info:
3    version: '1.0.0'
4    title: 'Optiplan API'
5    description: This is an API for a kanban-based project
6  servers:
7    - url: localhost:3000
8    - url: http://backendoptiplanbd.azurewebsites.net
9  paths:
10   /projects:
11     post:
12       summary: Add a new project
13       description: Add a new project
14       security:
15         - BearerAuth: []
16       requestBody:
17         description: Create a new project
18         content:
19           application/json:
20             schema:
21               $ref: "Projects/ProjectDto.yaml"
22       responses:
23         201:
24           description: Project created
25           content:
26             application/json:
27               schema:
28                 $ref: "Projects/ProjectResponse.yaml"
29         401:
30           description: Unauthorized
```

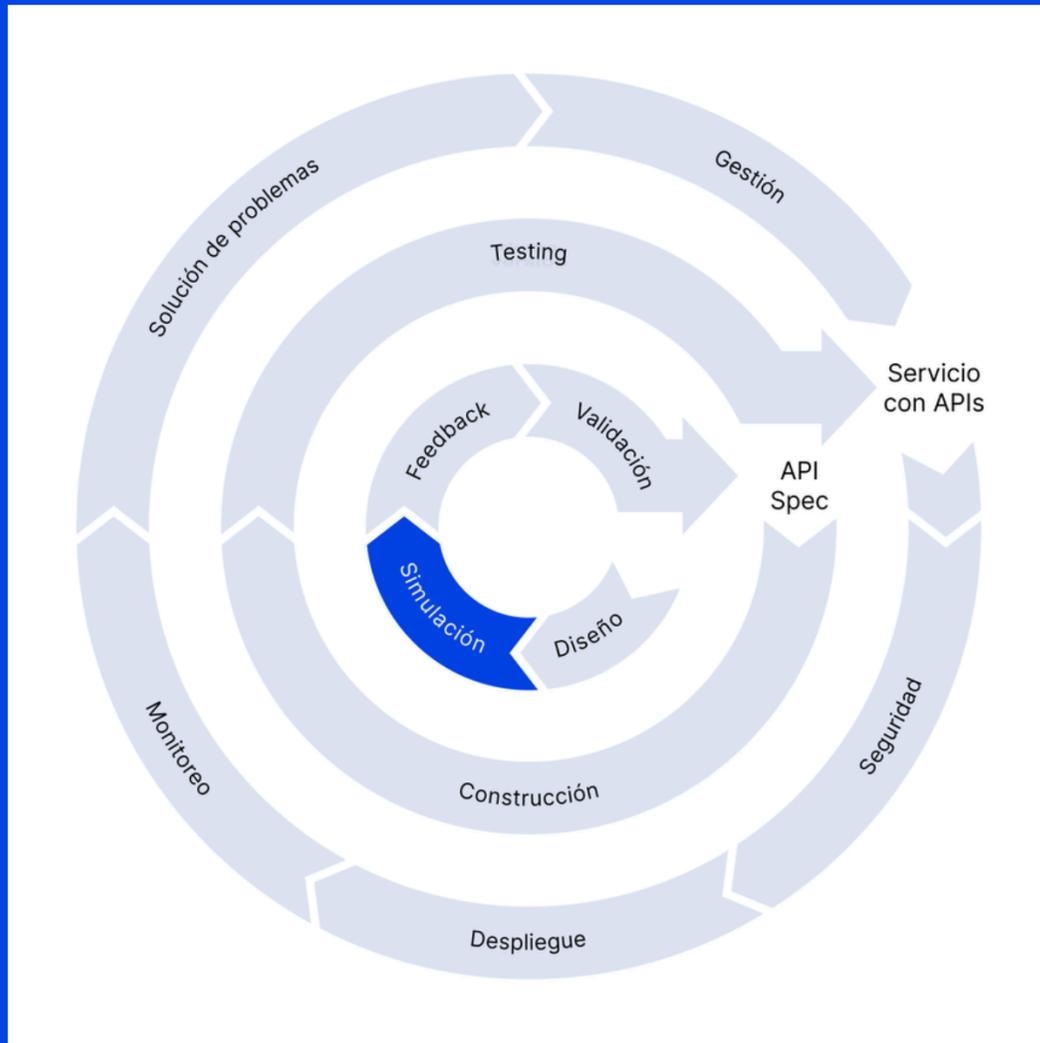
Labels on the right side of the image, connected to the code by red arrows:

- API Info (points to line 3)
- Servidores (points to line 7)
- Ruta (points to line 10)
- Descripción (points to line 13)
- Autorización (points to line 15)
- Petición (points to line 18)
- Respuestas (points to line 23)

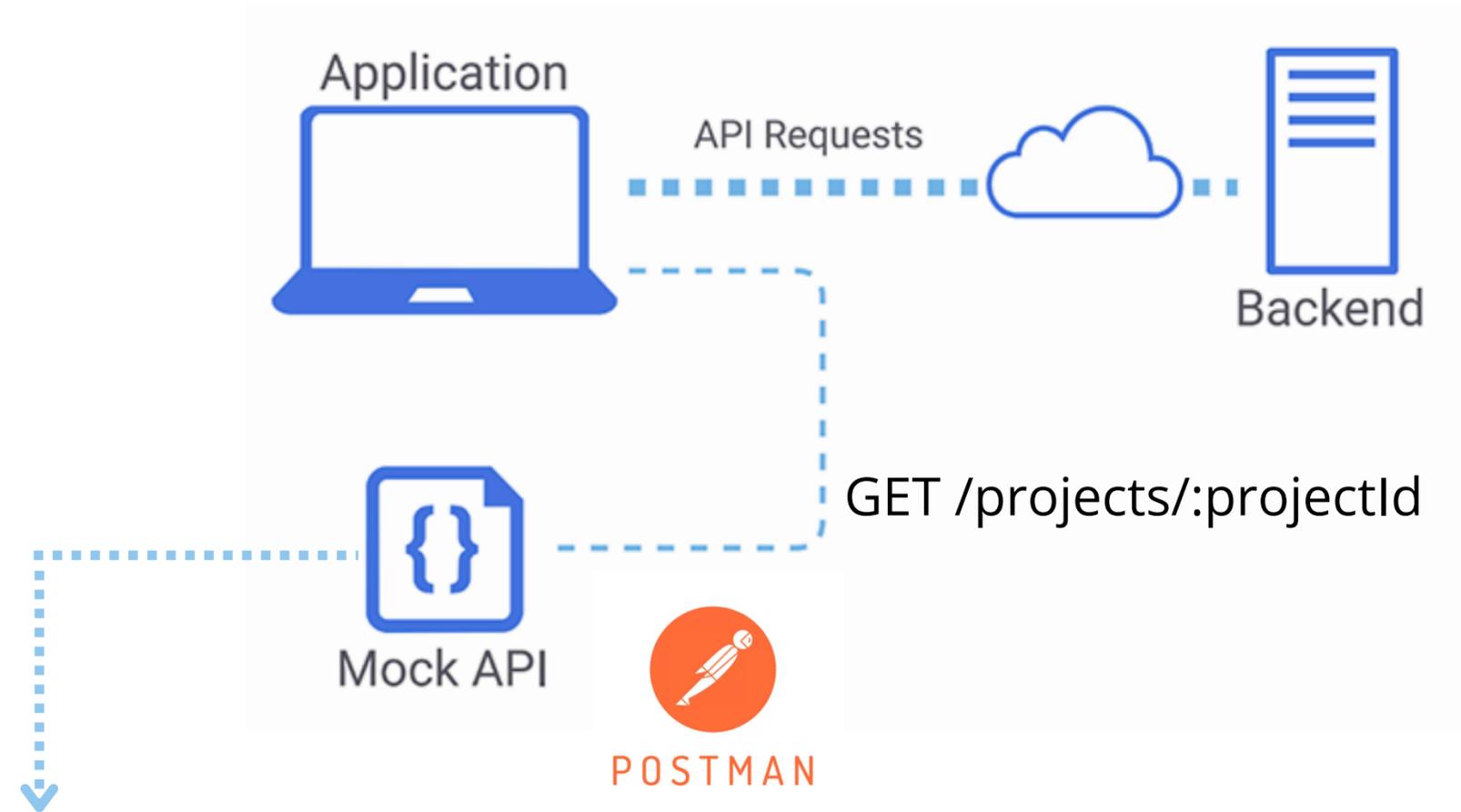
SIMULACIÓN



Simulación de la API en Postman.



Mock de la API



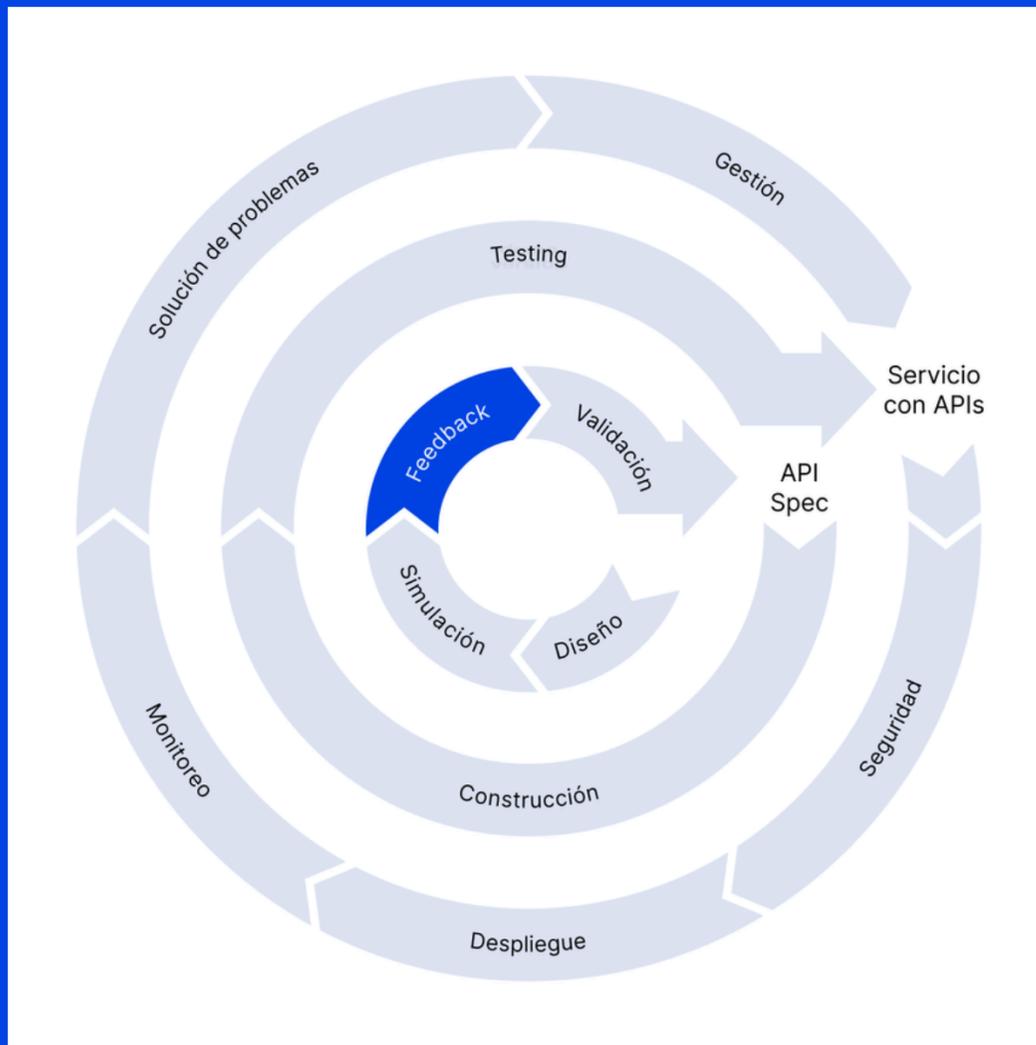
Respuesta del mock

```
1 {
2   "title": "Transformación Digital de la Empresa",
3   "id": "a1b2c3d4-e5f6-7890-ab12-cd34ef567890",
4   "description": "Implementación de nuevas tecnologías para optimizar procesos y mejorar la eficiencia operativa.",
5   "startDate": "2024-06-01",
6   "endDate": "2024-12-31",
7   "code": "DIG-001",
8   "priorityOrder": 1,
9   "createdBy": "Roberto García"
10 }
```

FEEDBACK



Revisión y retroalimentación del equipo y PO.



Reuniones

El equipo revisa endpoints y documentación para definir correcciones.



VALIDACIÓN

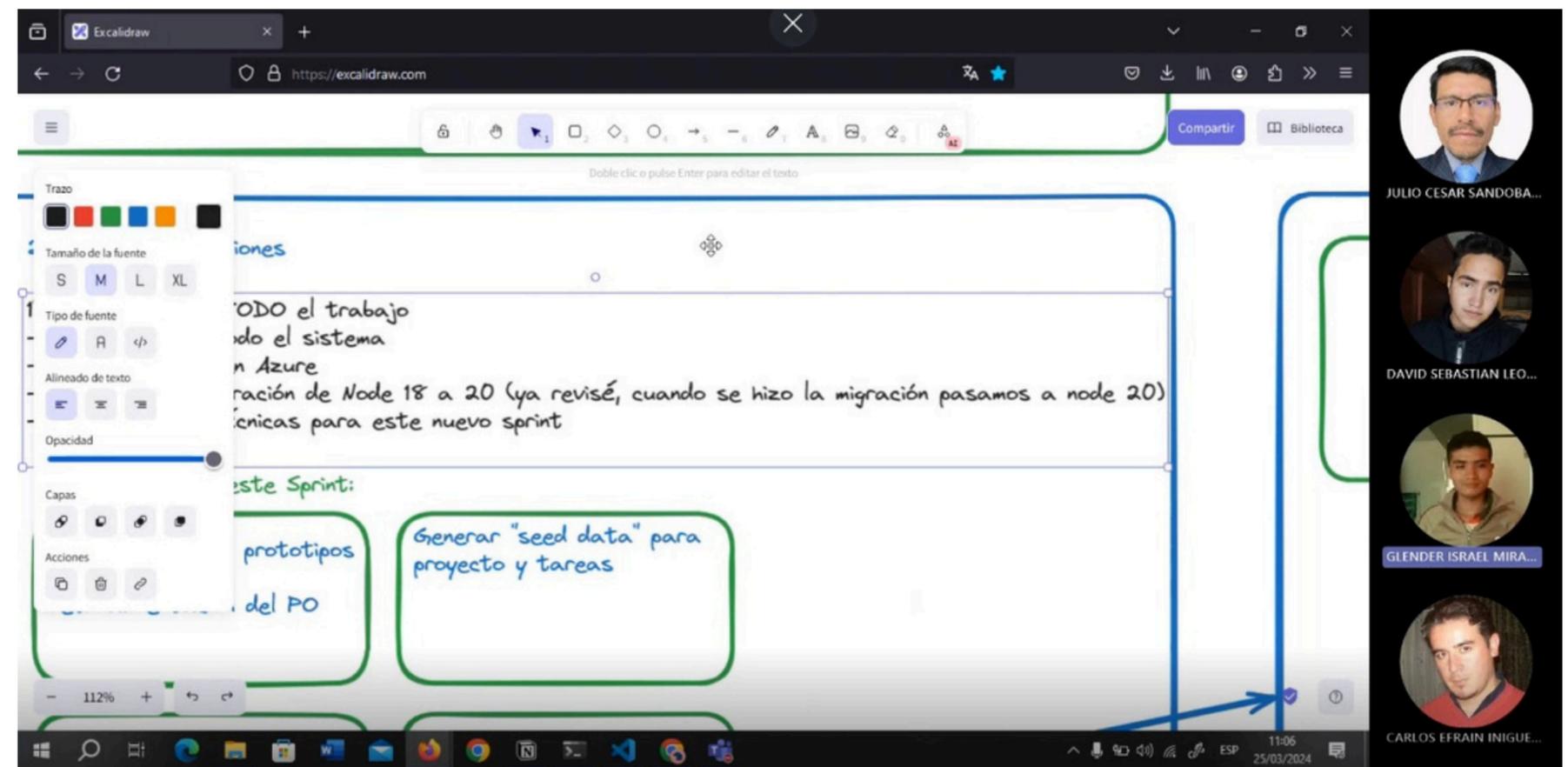


La API debe cumplir con los requisitos del negocio.



Revisión final

Última revisión de las especificaciones antes de implementación.

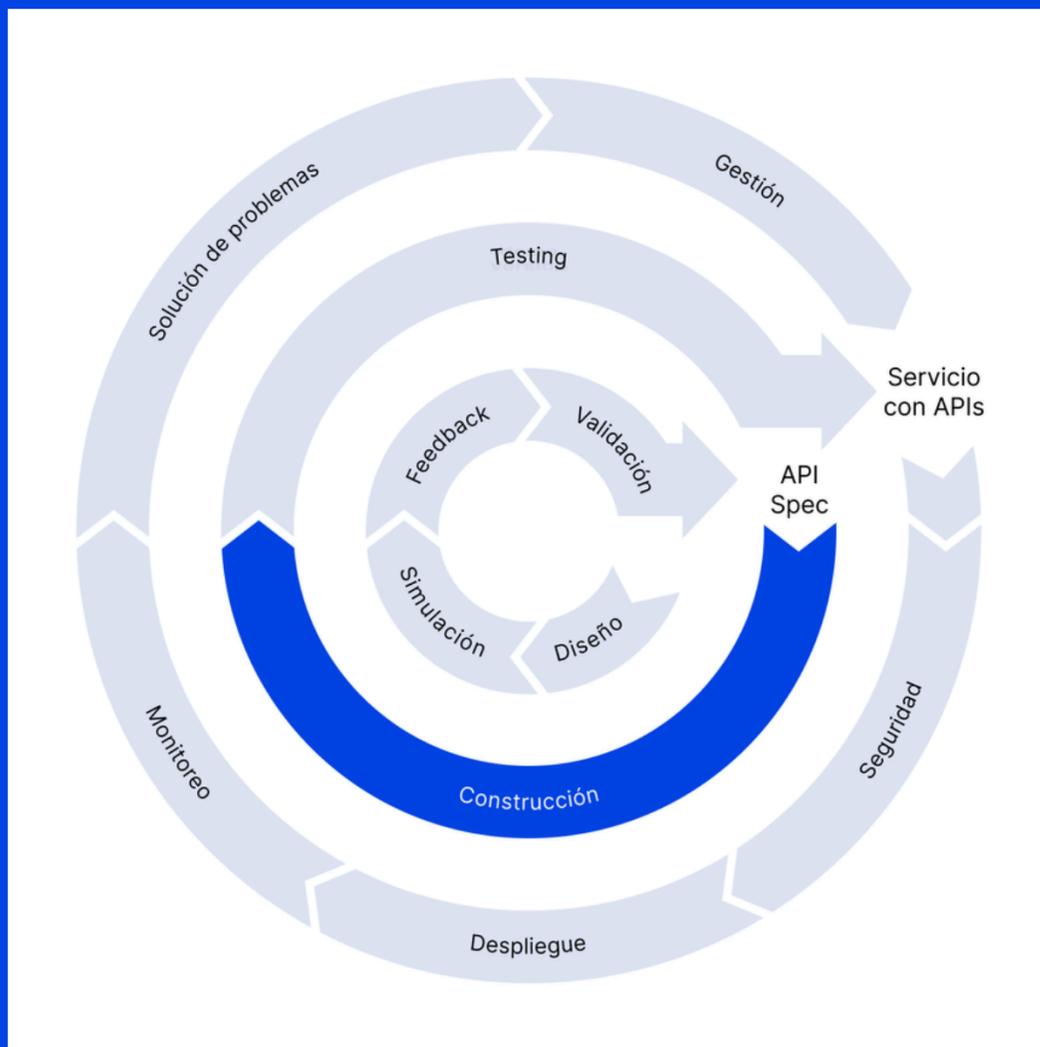


IMPLEMENTACIÓN

CONSTRUCCIÓN

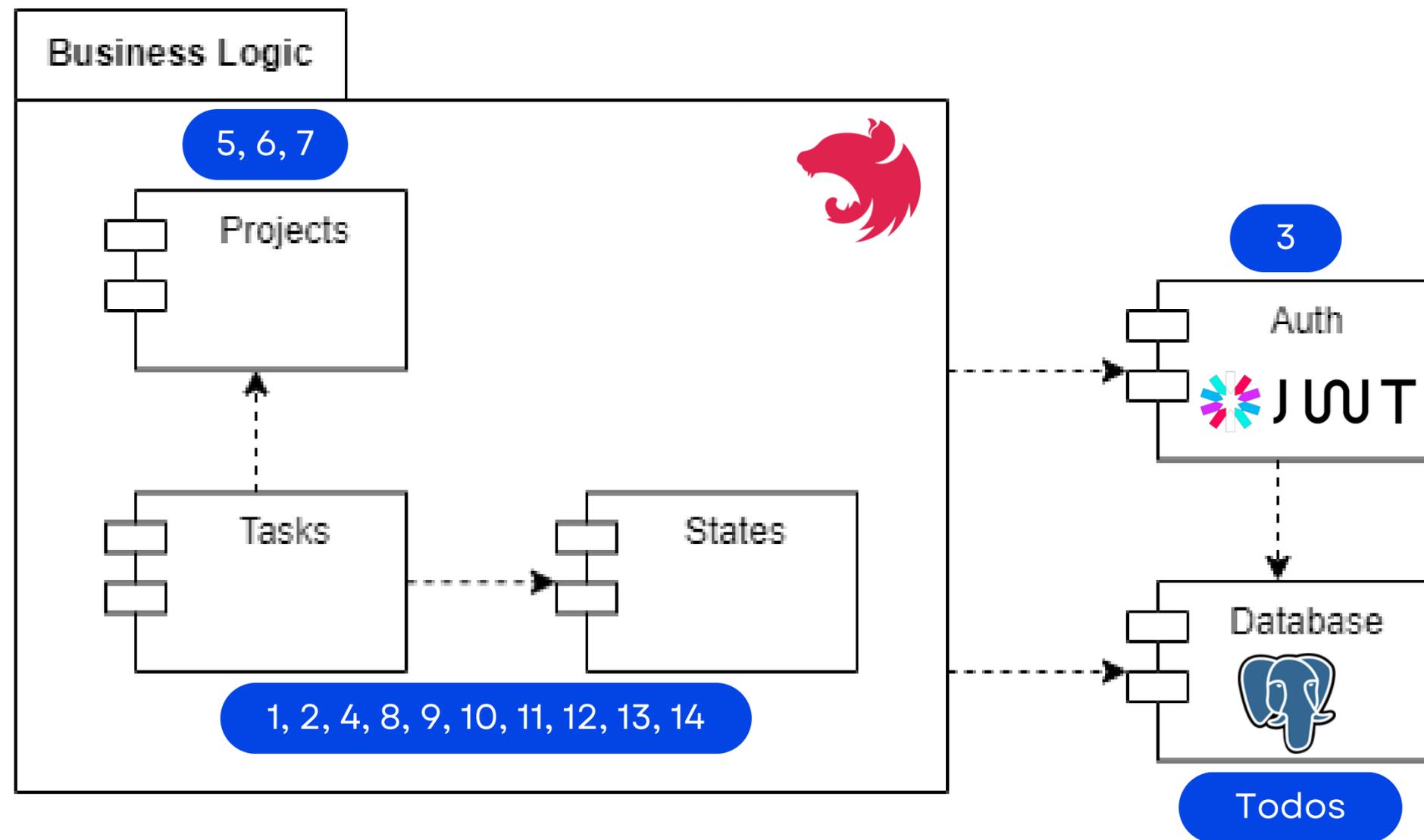


Implementación de funcionalidades de la API.

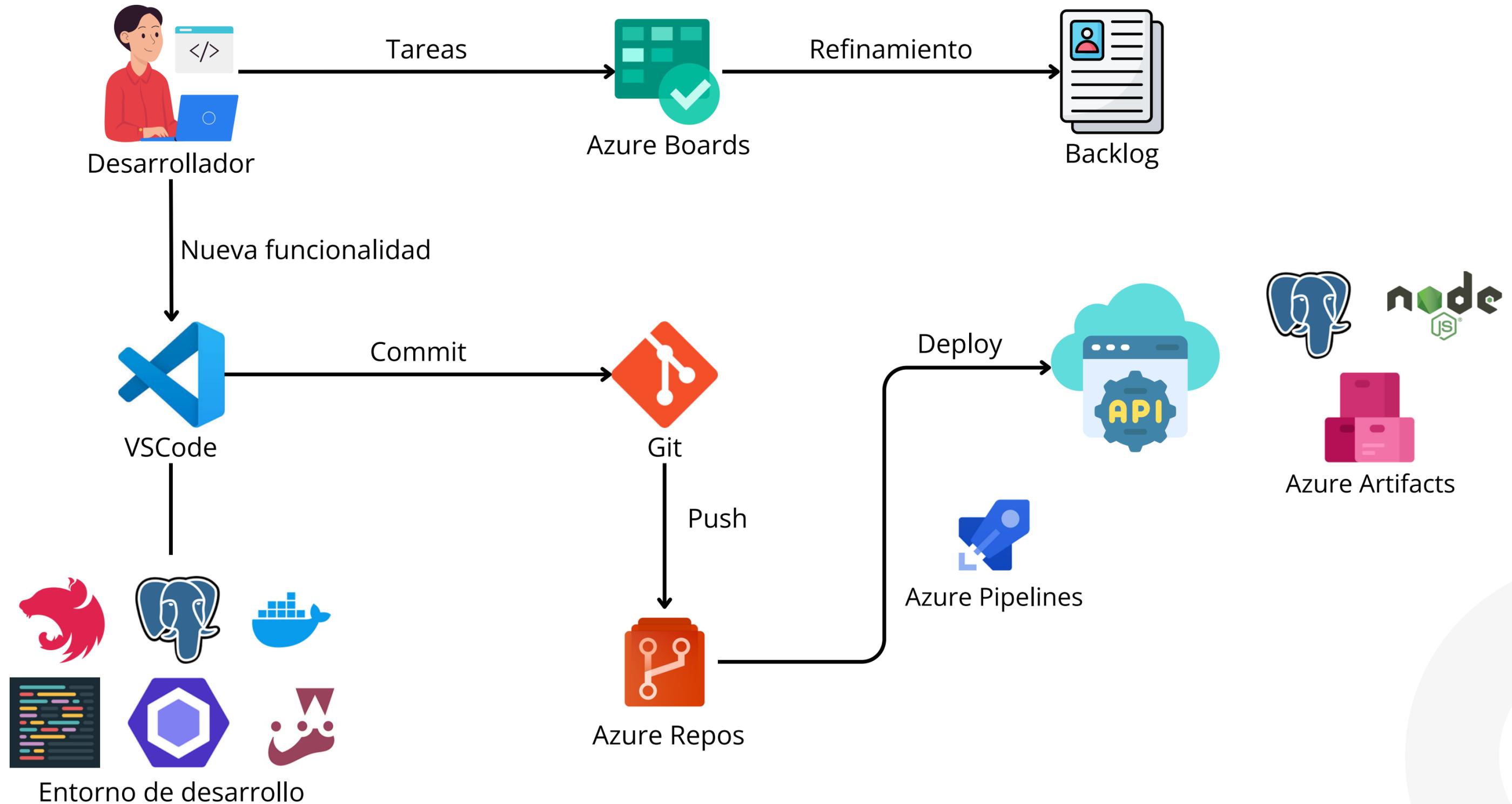


Componentes implementados

Funcionalidades agrupadas en módulos (NestJS).



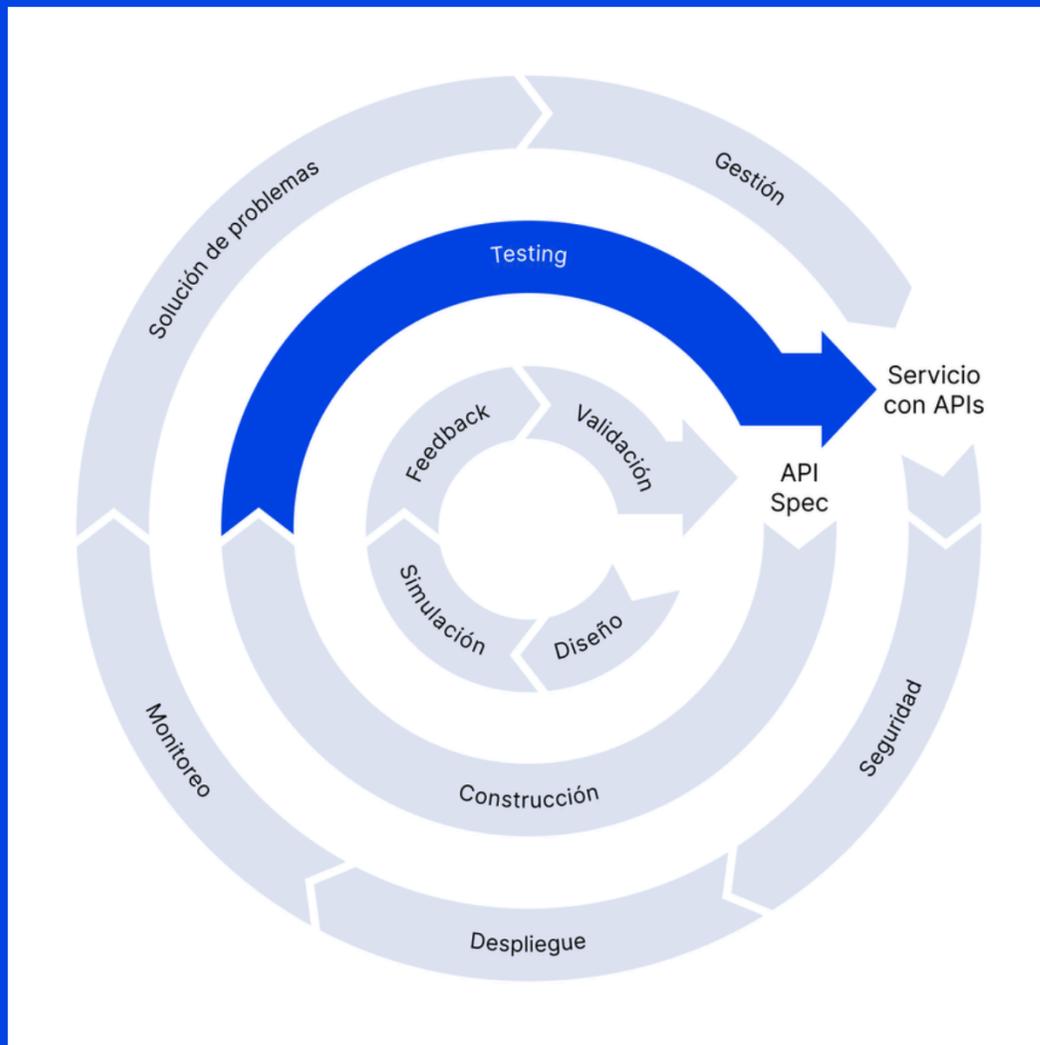
CONSTRUCCIÓN



TESTING

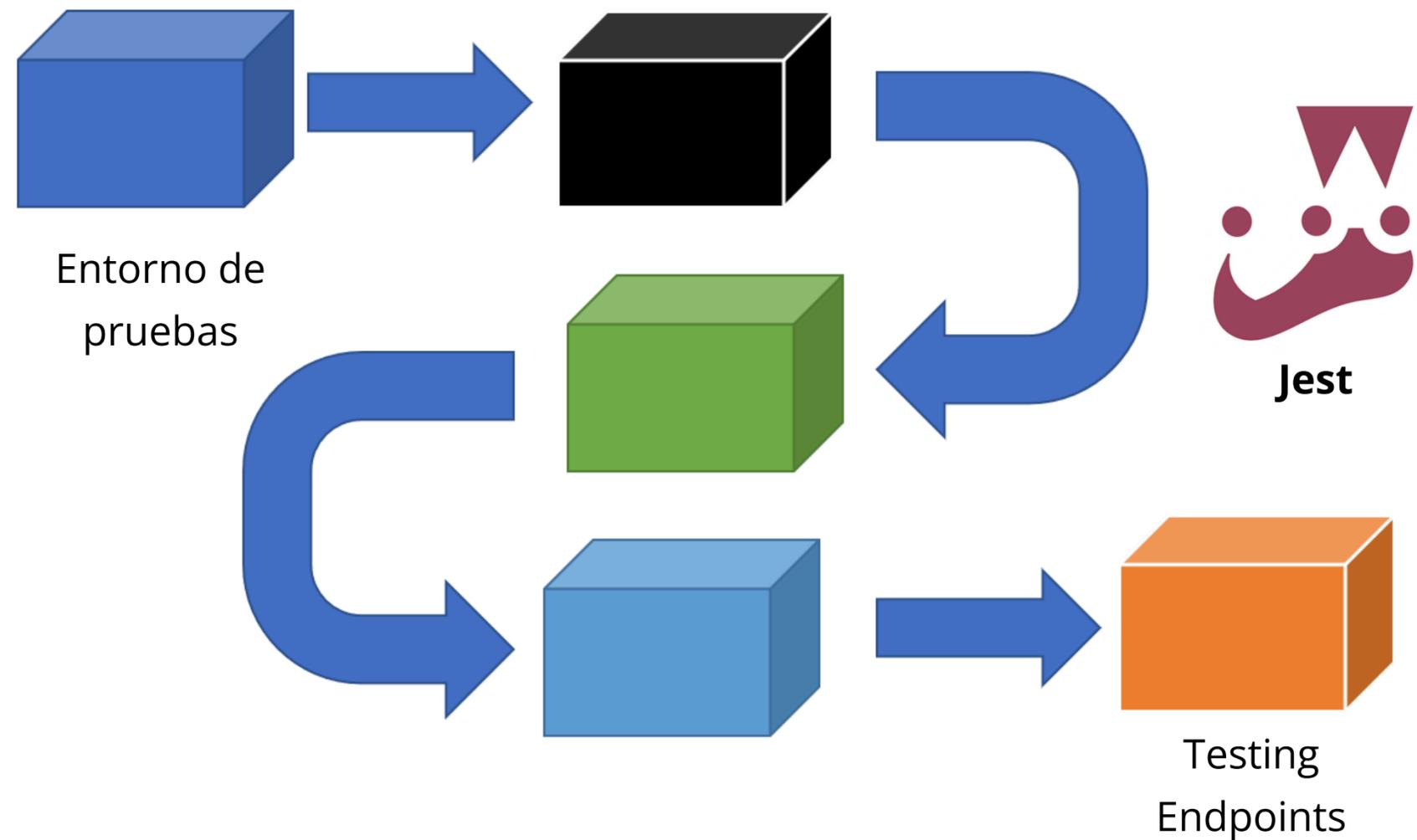


Pruebas end-to-end automatizadas de endpoints.



Pruebas end-to-end

- Las peticiones y respuestas deben ser las esperadas.
- 21 pruebas realizadas.

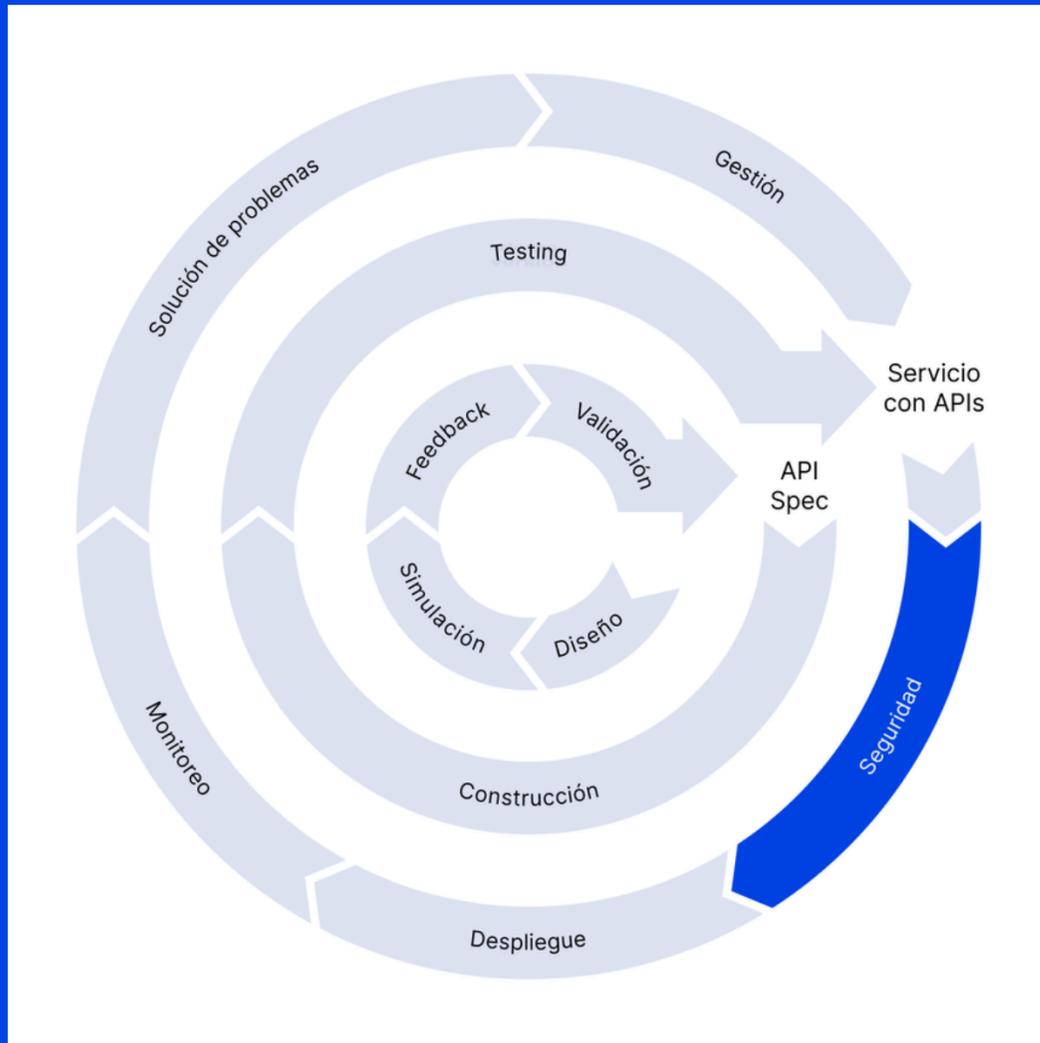


GESTIÓN

SEGURIDAD



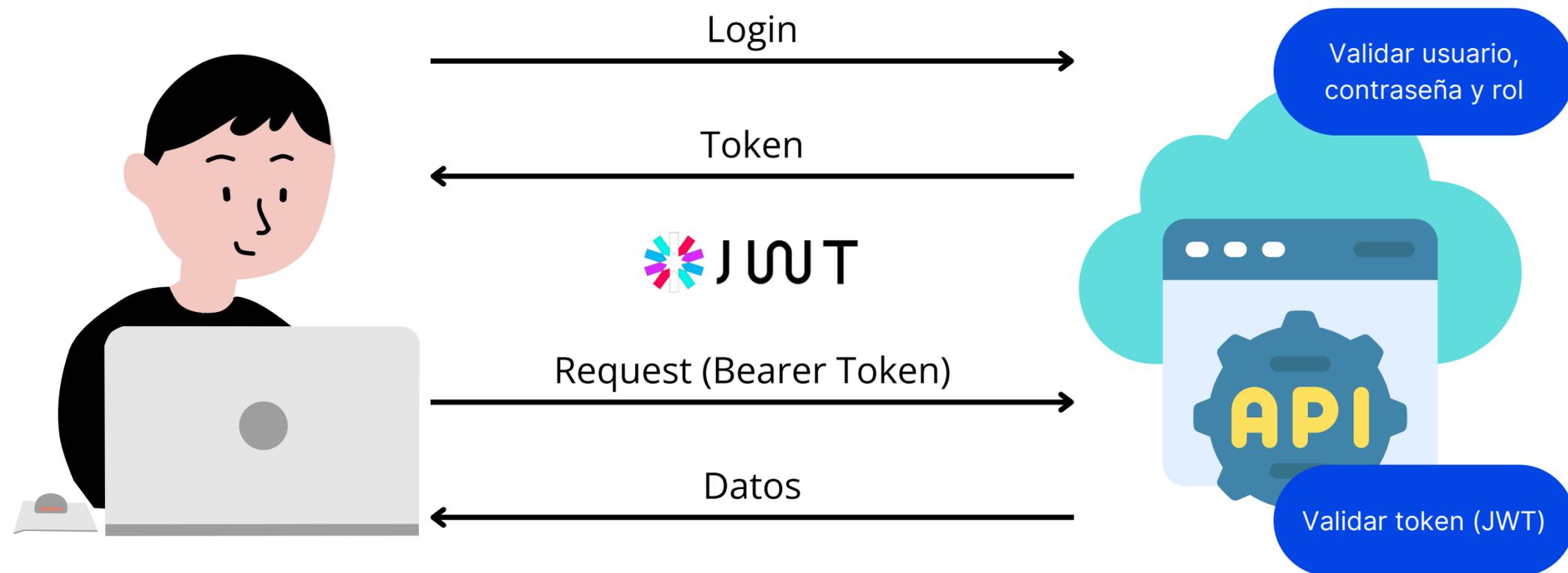
Protección de rutas de la API mediante roles.



Autenticación y autorización

Roles:

- Administrador
- Gestor empresarial
- Gestor de área
- Líder de área
- Técnico



DESPLIEGUE



Despliegue continuo durante los 14 Sprints.

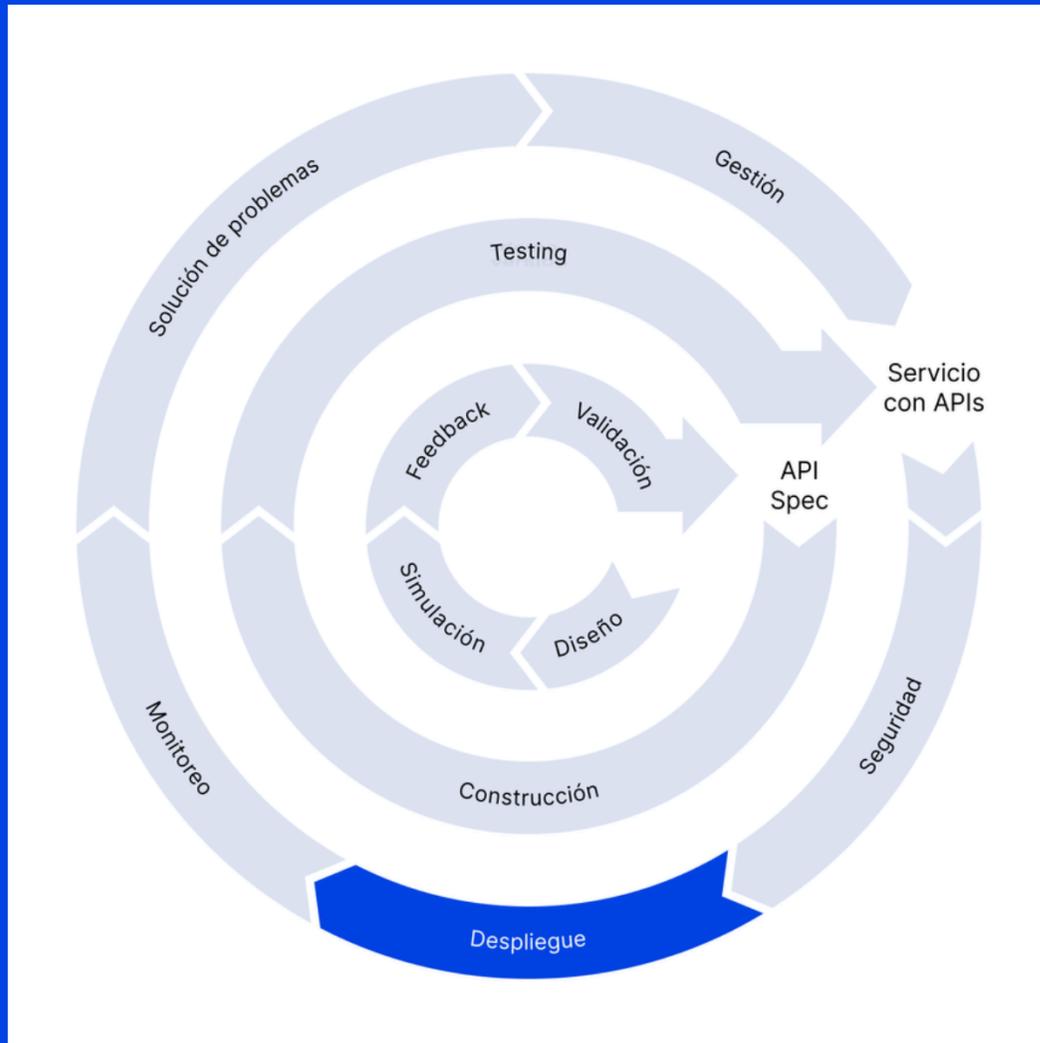
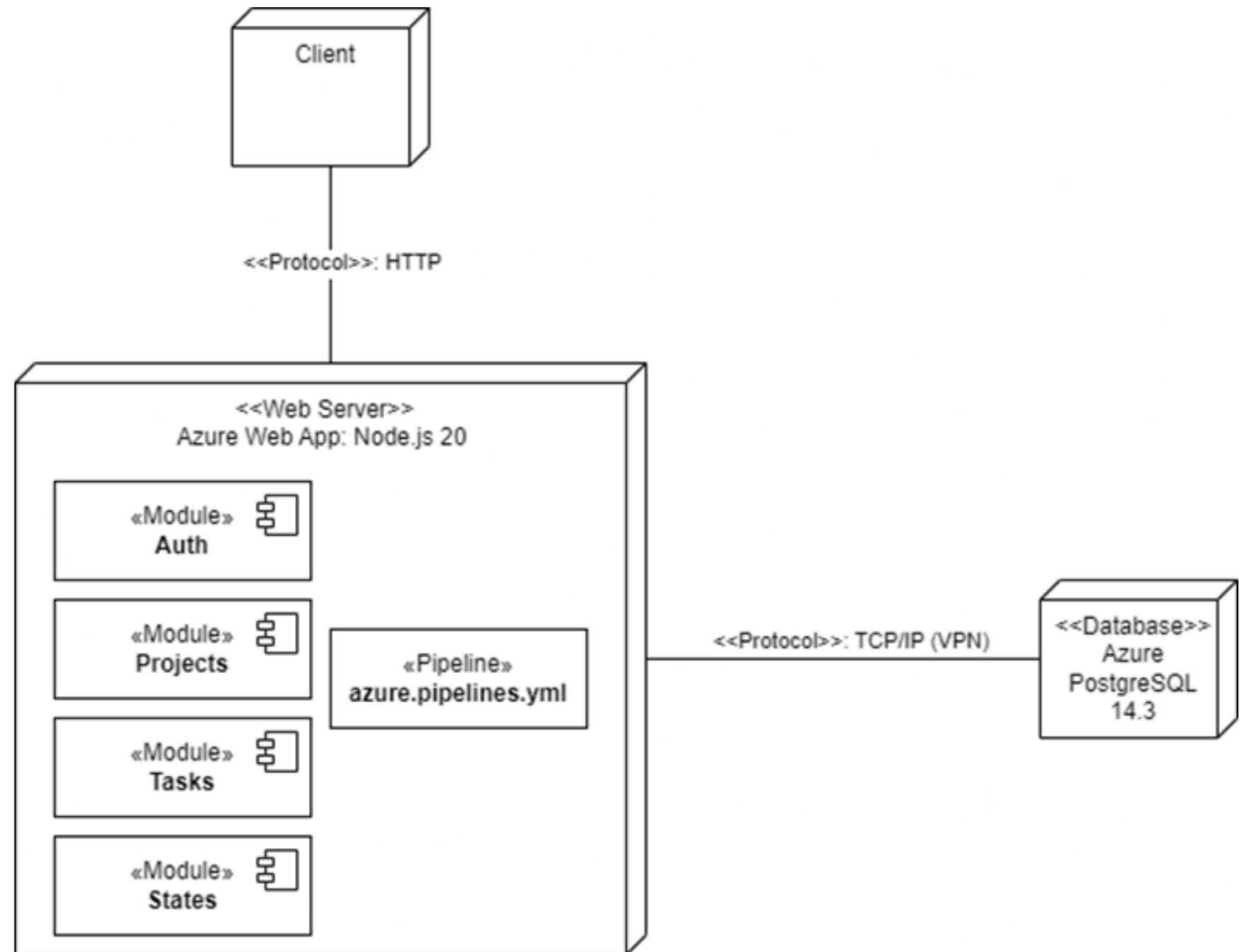


Diagrama de despliegue



RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

RESULTADOS

19

Endpoints
implementados



15

Historias de
usuario



14

Sprints



CONCLUSIONES



OE1: “Definir el contrato de la API de acuerdo con las necesidades del Front-End”

- Reuniones con el frontend.
- Acuerdos en formatos y respuestas.
- Socialización de la API mediante Postman.



OE2: “Diseñar los endpoints a través de una arquitectura REST”

- Definición de rutas y métodos.
- Jerarquía y organización de endpoints.



OE3: “Implementar y exponer la lógica de negocio a través de una API”

- Lógica de negocio implementada en 19 endpoints.
- Módulos de autenticación, proyectos, estados y tareas.

RECOMENDACIONES

01 Comunicación

- Comunicación clara y concisa.
- Socializar estructura de endpoints.
- Comprender requisitos.

02 Investigación

- Escoger herramientas y tecnologías adecuadas.
- Evitar el cargo cult.

03 Nuevas funcionalidades

- Analizar entorno empresarial.
- Integración, chats, IA.





**Escuela Politécnica
Nacional**

GRACIAS

POR SU ATENCIÓN

Septiembre, 2024

Bryan Marcelo Tapia Vergara



bryan.tapia03@epn.edu.ec

