# Endpoints y Pruebas en Postman (SUPERVISOR + STANDARD)

Este documento contiene únicamente:

- listado de endpoints
- variables de Postman
- flujo de pruebas en Postman

---

## Base URL

- `{{BASE_URL}} = http://localhost:3000`

---

## Variables de Environment (Postman)

- `BASE_URL`
- `SUPERVISOR_INVITE_TOKEN`
- `SUPERVISOR_JWT`
- `STANDARD_JWT`
- `STANDARD_USER_ID`
- `TASK_ID_SUP`
- `TASK_ID_STD`
- `REQ_CREATE_STD`
- `REQ_UPDATE_STD`
- `REQ_COMPLETE_STD`
- `REQ_DELETE_STD`

---

## Endpoints Auth

- `POST /api/auth/supervisor-token/request`
- `POST /api/auth/register`
- `POST /api/auth/login`
- `GET /api/auth/me` *(Bearer token)*
- `PUT /api/auth/me/profile` *(Bearer token)*
- `PUT /api/auth/me/password` *(Bearer token)*
- `GET /api/auth/can-approve` *(Bearer token + permiso supervisor)*

### Payloads de referencia (Auth)

`POST /api/auth/supervisor-token/request`

```
{
  "name": "Supervisor Uno",
  "email": "sup1@test.com",
  "organizationalUnit": "Finanzas"
}
```

`POST /api/auth/register` (SUPERVISOR)

```
{
  "name": "Supervisor Uno",
  "email": "sup1@test.com",
  "password": "SupPass123",
  "organizationalUnit": "Finanzas",
  "role": "SUPERVISOR",
  "supervisorToken": "{{SUPERVISOR_INVITE_TOKEN}}"
}
```

`POST /api/auth/register` (STANDARD)

```
{
  "name": "Standard Uno",
  "email": "std1@test.com",
  "password": "StdPass123",
  "organizationalUnit": "Finanzas",
  "role": "STANDARD"
}
```

`POST /api/auth/login`

```
{
  "identifier": "sup1@test.com",
  "password": "SupPass123"
}
```

---

## Endpoints Tasks

- `GET /api/tasks` *(Bearer token)*
- `GET /api/tasks/unit-users` *(Bearer token)*
- `GET /api/tasks/change-requests/mine` *(Bearer token)*
- `POST /api/tasks/requests/create` *(Bearer token)*
- `POST /api/tasks/tasks/:taskId/requests/update` *(Bearer token)*
- `POST /api/tasks/tasks/:taskId/requests/complete` *(Bearer token)*

- `POST /api/tasks/tasks/:taskId/requests/delete` *(Bearer token)*
- `GET /api/tasks/change-requests/pending` *(Bearer token + supervisor)*
- `POST /api/tasks/change-requests/:requestId/decision` *(Bearer token + supervisor)*
- `GET /api/tasks/reports/snapshot` *(Bearer token)*

### Payloads de referencia (Tasks)

`POST /api/tasks/requests/create`

```
{
  "title": "Tarea Supervisor Directa",
  "description": "Creada por supervisor",
  "priority": "HIGH",
  "dueDate": "2026-03-01",
  "assignedToUserId": {{STANDARD_USER_ID}},
  "reason": "Inicial"
}
```

`POST /api/tasks/tasks/:taskId/requests/update`

```
{
  "title": "Tarea Supervisor Actualizada",
  "status": "IN_PROGRESS",
  "priority": "MEDIUM",
  "reason": "Ajuste"
}
```

`POST /api/tasks/tasks/:taskId/requests/complete`

```
{
  "reason": "Finalizada"
}
```

`POST /api/tasks/tasks/:taskId/requests/delete`

```
{
  "reason": "Depuración"
}
```

`POST /api/tasks/change-requests/:requestId/decision`

```
{
  "decision": "APPROVED",
  "reviewComment": "Aprobado"
}
```

---

## Flujo de pruebas Postman (orden recomendado)

### 1) Setup de sesión

1. `POST /api/auth/supervisor-token/request`
2. Copiar token de `previewUrl` a `SUPERVISOR_INVITE_TOKEN`
3. `POST /api/auth/register` (SUPERVISOR)
4. `POST /api/auth/register` (STANDARD)
5. `GET /api/auth/me` con ambos tokens
6. `GET /api/auth/can-approve`:
   - supervisor: `200`
   - standard: `403`

### 2) Preparar ids

1. `GET /api/tasks/unit-users` con supervisor
2. Tomar usuario `STANDARD` y guardar `STANDARD_USER_ID`

### 3) Flujo supervisor directo

1. `POST /api/tasks/requests/create` con supervisor
2. Guardar `TASK_ID_SUP` desde `response.request.id`
3. `POST /api/tasks/tasks/{{TASK_ID_SUP}}/requests/update`
4. `POST /api/tasks/tasks/{{TASK_ID_SUP}}/requests/complete`
5. `GET /api/tasks` para validar cambios

### 4) Flujo standard con aprobación

1. `POST /api/tasks/requests/create` con standard
2. Guardar `REQ_CREATE_STD` desde `response.request.id`
3. `GET /api/tasks/change-requests/mine?status=PENDING` con standard
4. `GET /api/tasks/change-requests/pending` con supervisor
5. `POST /api/tasks/change-requests/{{REQ_CREATE_STD}}/decision` con supervisor (`APPROVED`)
6. Guardar `TASK_ID_STD` desde `response.result.taskId`

### 5) Update / complete / delete por standard

1. `POST /api/tasks/tasks/{{TASK_ID_STD}}/requests/update` (standard)
2. Guardar `REQ_UPDATE_STD`
3. `POST /api/tasks/change-requests/{{REQ_UPDATE_STD}}/decision` (supervisor, `REJECTED`)

4. `GET /api/tasks/change-requests/mine?status=REJECTED` (standard)
5. `POST /api/tasks/tasks/{{TASK_ID_STD}}/requests/complete` (standard)
6. Guardar `REQ_COMPLETE_STD`
7. `POST /api/tasks/change-requests/{{REQ_COMPLETE_STD}}/decision` (supervisor, `APPROVED`)
8. `POST /api/tasks/tasks/{{TASK_ID_STD}}/requests/delete` (standard)
9. Guardar `REQ_DELETE_STD`
10. `POST /api/tasks/change-requests/{{REQ_DELETE_STD}}/decision` (supervisor, `APPROVED`)
11. `GET /api/tasks` y validar que la tarea eliminada no aparezca

## 6) Reportes

1. `GET /api/tasks/reports/snapshot` con supervisor
2. Validar campos: `total`, `completed`, `inProgress`, `pending`, `pendingApprovals`, `statusDistribution`, `priorityDistribution`, `history`

---

## Validaciones esperadas mínimas

- Sin token en rutas protegidas -> `401`
- Token inválido -> `401`
- Standard en endpoints de aprobación -> `403`
- `decision` inválida -> `400`
- `taskId` inválido -> `400`