

Table of contents

TaskLynx / Nest Hotel – Docs	2
API – Index	4
Trabajo	8
Trabajador	38
Trabajador - Views	67
Tutorial	77
Business – Index	87
Mobile – Index	88
Nest Hotel – Index	89
Despliegue de Gestión Hotelera.....	90
Despliegue de Hotel Nest.....	95

TaskLynx / Nest Hotel – Docs



TaskLynx-logo-oval.svg

Introduction

TaskLynx is an ecosystem composed of TaskLynx Api, TaskLynx Business and TaskLynx Mobile. TaskLynx Api (<https://github.com/DanielAlmazan/TaskLynx-SpringBoot>) is a REST API that allows communication between TaskLynx Business (<https://github.com/DanielAlmazan/TaskLynx-JavaFX>) and TaskLynx Mobile (<https://github.com/DanielAlmazan/TaskLynx-Mobile>).

Nest Hotel (<https://github.com/DanielAlmazan/hotel-nest>) is a REST API that allows communication with TaskLynx Business (<https://github.com/DanielAlmazan/TaskLynx-JavaFX>) in order to manage the cleaning of hotel rooms.

TaskLynx Mobile (<https://github.com/DanielAlmazan/TaskLynx-Mobile>) is a mobile application that allows hotel staff to manage the cleaning of hotel rooms. It communicates with TaskLynx Api (<https://github.com/DanielAlmazan/TaskLynx-SpringBoot>) to get the tasks to be done. "Authentication" is done by using the endpoint `/api/trabajadores/{idTrabajador}/{contraseña}`. Didn't use JWT because it was not required.

TaskLynx Business (<https://github.com/DanielAlmazan/TaskLynx-JavaFX>) is a desktop application that allows hotel managers to manage the cleaning of hotel rooms and other

tasks. It communicates with TaskLynx Api (<https://github.com/DanielAlmazan/TaskLynx-SpringBoot>) to get the tasks to be done and with Nest Hotel (<https://github.com/DanielAlmazan/hotel-nest>) to manage the rooms.

API – Index



TaskLynx-API-oval-logo.svg

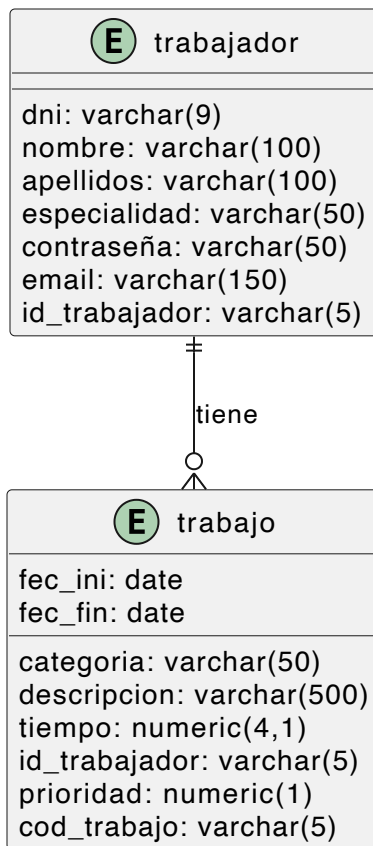
Introduction

TaskLynx API (<https://github.com/DanielAlmazan/TaskLynx-SpringBoot>) is a REST API that allows communication between TaskLynx Business (<https://github.com/DanielAlmazan/TaskLynx-JavaFX>) and TaskLynx Mobile (<https://github.com/DanielAlmazan/TaskLynx-Mobile>).

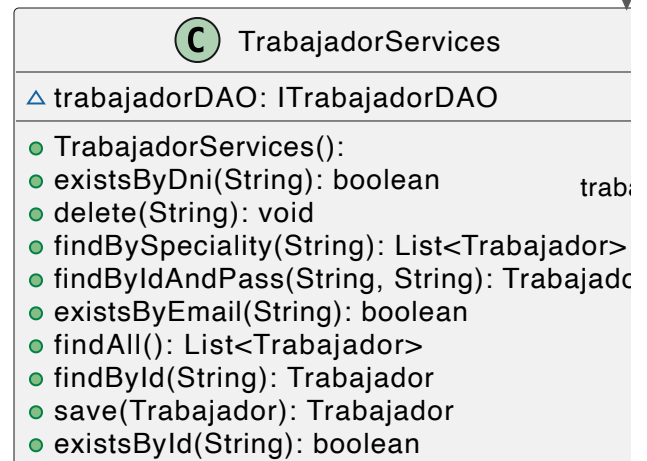
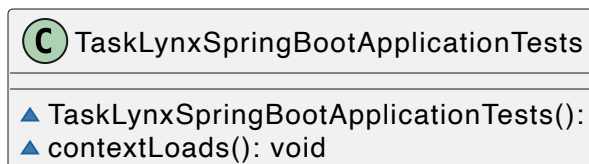
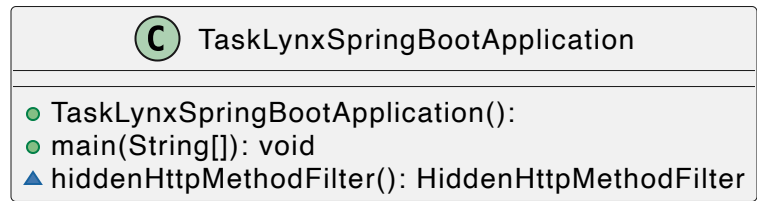
Technologies

- Java 21
- Spring Boot 2.5.4
- Apache Maven 3.9.6
- PostgreSQL

Entity-Relationship Diagram

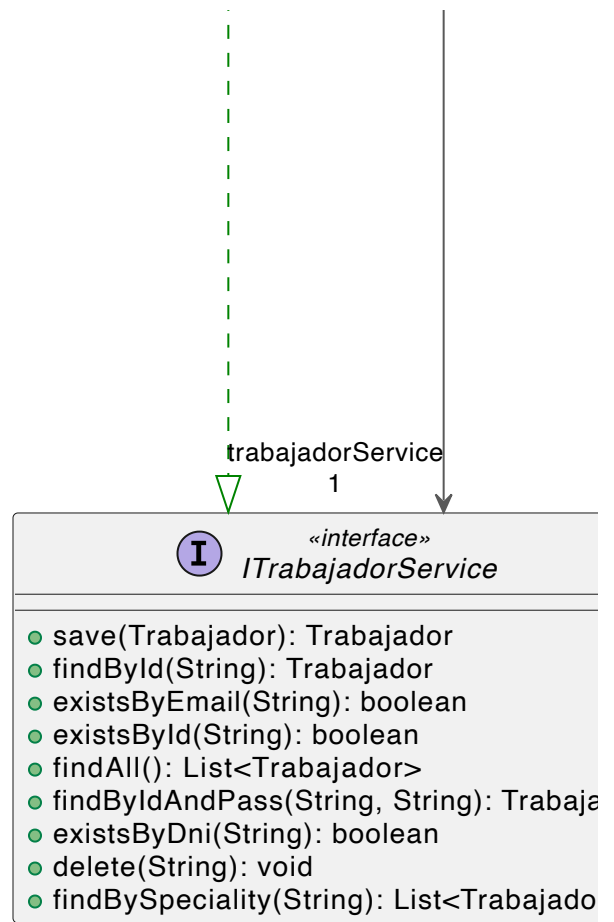


Class Diagram



trab:

trab:



Authors

TRABAJADOR CRUD:

Aitor Moreno Iborra (<https://github.com/LtVish>)


TRABAJO CRUD:

Miguel Collado (<https://github.com/MiguelColl>)

API Development:


Daniel Enrique Almazán Sellés (<https://github.com/DanielAlmazan>)

Trabajo

 Trabajo
<ul style="list-style-type: none">❑ categoria: String❑ idTrabajador: Trabajador❑ fecIni: LocalDate❑ fecFin: LocalDate❑ descripcion: String❑ codTrabajo: String❑ tiempo: BigDecimal❑ prioridad: BigDecimal
<ul style="list-style-type: none">● Trabajo():● getDescripcion(): String● getTiempo(): BigDecimal● getIdTrabajador(): Trabajador● setFecFin(LocalDate): void● getCodTrabajo(): String● setDescripcion(String): void● toString(): String● setFecIni(LocalDate): void● setCategoria(String): void● setCodTrabajo(String): void● setIdTrabajador(Trabajador): void● getFecIni(): LocalDate● setPrioridad(BigDecimal): void● setTiempo(BigDecimal): void● getPrioridad(): BigDecimal● getCategoria(): String● getFecFin(): LocalDate

BASE: /api/trabajos

GET BASE

 Returns all the tasks in the database. If there are no tasks, it will return an empty list.

Successful response

```
{
  "result": [
    {
```




```
"codTrabajo": "J001",
"categoria": "Mecánica",
"descripcion": "Reparación de tuberías",
"fecIni": "2024-01-01",
"fecFin": null,
"tiempo": null,
"idTrabajador": null,
"prioridad": 2
},
{
  "codTrabajo": "J002",
  "categoria": "Fontanería",
  "descripcion": "Reparación de tuberías",
  "fecIni": "2022-01-02",
  "fecFin": null,
  "tiempo": 5.0,
  "idTrabajador": {
    "idTrabajador": "T002",
    "dni": "23456789B",
    "nombre": "Ana",
    "apellidos": "Gómez",
    "especialidad": "Fontanería",
    "contraseña": "contraseña2",
    "email": "ana.gomez@example.com"
  },
  "prioridad": 2
},
{
  "codTrabajo": "J003",
  "categoria": "Electricidad",
  "descripcion": "Instalación de luces",
  "fecIni": "2022-01-03",
  "fecFin": null,
  "tiempo": 8.0,
  "idTrabajador": {
    "idTrabajador": "T003",
    "dni": "34567890C",
    "nombre": "Carlos",
```

```
        "apellidos": "Martínez",
        "especialidad": "Electricidad",
        "contraseña": "contraseña3",
        "email": "carlos.martinez@example.com"
    },
    "prioridad": 3
}
],
"error": false
}
```


Empty list response

```
{
  "result": [],
  "error": false
}
```

GET BASE/:id

 Returns a task by its ID. If the task does not exist, it will return an error message.


Successful response

 /api/trabajos/J001

```
{
  "result": {
    "codTrabajo": "J001",
    "categoria": "Mecánica",
    "descripcion": "Reparación de tuberías",
    "fecIni": "2024-01-01",
    "fecFin": null,
  }
}
```


```
    "tiempo": null,  
    "idTrabajador": null,  
    "prioridad": 2  
  },  
  "error": false  
}
```

Non-existent task response

 /api/trabajos/J025

```
{  
  "errorMessage": "El trabajo con ID 'J025' no existe en la  
base de datos",  
  "error": true  
}
```

POST BASE

 Create a new task in the database and returns the task created. If there are any errors, it will return an error message with the list of errors. If the worker already exists, it will return an error message.

All fields are required except:

- fecFin: task completion date
- tiempo: duration of the task
- idTrabajador: worker assigned

Successful body

```
{
  "codTrabajo": "J007",
  "categoria": "Pintura",
  "descripcion": "Pintar las paredes",
  "fecIni": "2024-01-09",
  "prioridad": 4
}
```

Response

```
{
  "result": {
    "codTrabajo": "J007",
    "categoria": "Pintura",
    "descripcion": "Pintar las paredes",
    "fecIni": "2024-01-09",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": null,
    "prioridad": 4
  },
  "error": false
}
```

Other successful body

```
{
  "codTrabajo": "J008",
  "categoria": "Pintura",
  "descripcion": "Pintar las paredes",
  "fecIni": "2024-01-09",
  "fecFin": "2024-01-11",
  "tiempo": 8.5,
}
```

```
"idTrabajador": {
  "idTrabajador": "T001",
  "dni": "12345678A",
  "nombre": "Pepe",
  "apellidos": "Gomez",
  "especialidad": "Fontanería",
  "contraseña": "123",
  "email": "wefwefesadf.com"
},
"prioridad": 4
}
```

Response

```
{
  "result": {
    "codTrabajo": "J008",
    "categoria": "Pintura",
    "descripcion": "Pintar las paredes",
    "fecIni": "2024-01-09",
    "fecFin": "2024-01-11",
    "tiempo": 8.5,
    "idTrabajador": {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Pepe",
      "apellidos": "García",
      "especialidad": "Fontaneria",
      "contraseña": "123",
      "email": "wefwefewf@ergerg.com"
    },
    "prioridad": 4
  },
  "error": false
}
```

Error body

```
{
  "codTrabajo": "J009",
  "fecIni": "2024-01-09",
  "prioridad": 4
}
```

Response

```
{
  "errorsList": [
    "La categoría no puede estar vacía",
    "La descripción no puede estar vacía"
  ],
  "error": true
}
```

Other error body

```
{
  "codTrabajo": "J008",
  "categoria": "Pintura",
  "descripcion": "Pintar las paredes",
  "fecIni": "2024-01-09",
  "prioridad": 4
}
```

Response

```
{
  "errorsList": [
```

```
    "El trabajo con id 'J008' ya existe en la base de datos"
  ],
  "error": true
}
```

PUT BASE/:id

⚠ Update a task by its ID and returns the task updated. If the worker does not exist, it will return an error message. If any of the fields are empty or invalid, it will return an error message with the list of errors.

Successful body

⚠ /api/trabajos/J007


We add an assigned worker

```
{
  "codTrabajo": "J007",
  "categoria": "Pintura",
  "descripcion": "Pintar las paredes",
  "fecIni": "2024-01-09",
  "prioridad": 4,
  "idTrabajador": {
    "idTrabajador": "T001",
    "dni": "12345678A",
    "nombre": "Pepe",
    "apellidos": "García",
    "especialidad": "Fontanería",
    "contraseña": "123",
    "email": "wefwefewf@ergerg.com"
  }
}
```

Response

```
{
  "result": {
    "codTrabajo": "J007",
    "categoria": "Pintura",
    "descripcion": "Pintar las paredes",
    "fecIni": "2024-01-09",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Pepe",
      "apellidos": "García",
      "especialidad": "Fontanería",
      "contraseña": "123",
      "email": "wefwefewf@ergerg.com"
    },
    "prioridad": 4
  },
  "error": false
}
```

Error body

 /api/trabajos/J020

```
{
  "codTrabajo": "J020",
  "categoria": "Pintura",
  "descripcion": "Pintar las paredes",
  "fecIni": "2024-01-09",
```




```
"prioridad": 4
}
```


Response

```
{
  "errorsList": [
    "No se pudo editar, el trabajo con id 'J020' no existe en
    la base de datos"
  ],
  "error": true
}
```

DELETE BASE/:id


 Delete a task by its ID. Returns a response without errors. If the task does not exist, it will return an error message.

Successful response

 /api/trabajos/J008

```
{
  "error": false
}
```

Non-existent task response

 /api/trabajos/J020

```
{
  "errorMessage": "El trabajo con id 'J020' no existe en la
base de datos",
  "error": true
}
```

GET BASE/pendientes

⚠ Returns all the tasks that are pending. If there are no pending tasks, it will return an empty list.

200 – Results

```
{
  "result": [
    {
      "codTrabajo": "J001",
      "categoria": "Carpintería",
      "descripcion": "Construcción de armario",
      "fecIni": "2022-01-01",
      "fecFin": null,
      "tiempo": 10.0,
      "idTrabajador": {
        "idTrabajador": "T001",
        "dni": "12345678A",
        "nombre": "Juan",
        "apellidos": "Pérez",
        "especialidad": "Carpintería",
        "contraseña": "contraseña1",
        "email": "juan.perez@example.com"
      },
      "prioridad": 1
    },
    {

```

```
"codTrabajo": "J002",
"categoria": "Fontanería",
"descripcion": "Reparación de tuberías",
"fecIni": "2022-01-02",
"fecFin": null,
"tiempo": 5.0,
"idTrabajador": {
  "idTrabajador": "T002",
  "dni": "23456789B",
  "nombre": "Ana",
  "apellidos": "Gómez",
  "especialidad": "Fontanería",
  "contraseña": "contraseña2",
  "email": "ana.gomez@example.com"
},
"prioridad": 2
},
{
  "codTrabajo": "J003",
  "categoria": "Electricidad",
  "descripcion": "Instalación de luces",
  "fecIni": "2022-01-03",
  "fecFin": null,
  "tiempo": 8.0,
  "idTrabajador": {
    "idTrabajador": "T003",
    "dni": "34567890C",
    "nombre": "Carlos",
    "apellidos": "Martínez",
    "especialidad": "Electricidad",
    "contraseña": "contraseña3",
    "email": "carlos.martinez@example.com"
  },
  "prioridad": 3
},
{
  "codTrabajo": "J800",
  "categoria": "Espacio-Tiempo",
```

```

    "descripcion": "Visitar a Sarah Connor",
    "fecIni": "2029-10-20",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": null,
    "prioridad": 1
  },
  {
    "codTrabajo": "J004",
    "categoria": "Electricidad",
    "descripcion": "Desinstalación de luces",
    "fecIni": "2022-01-03",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T003",
      "dni": "34567890C",
      "nombre": "Carlos",
      "apellidos": "Martínez",
      "especialidad": "Electricidad",
      "contraseña": "contraseña3",
      "email": "carlos.martinez@example.com"
    },
    "prioridad": 4
  }
],
"error": false
}

```

200 – Empty

```

{
  "result": [],
  "error": false
}

```

GET BASE/sinTrabajador

⚠ Returns all the tasks that are pending and have no worker assigned. If there are no pending tasks without a worker assigned, it will return an empty list.

200 – Results

```
{
  "result": [
    {
      "codTrabajo": "J800",
      "categoria": "Espacio-Tiempo",
      "descripcion": "Visitar a Sarah Connor",
      "fecIni": "2029-10-20",
      "fecFin": null,
      "tiempo": null,
      "idTrabajador": null,
      "prioridad": 1
    },
    {
      "cod_trabajo": "J008",
      "categoria": "Pintura",
      "descripcion": "Pintar las paredes",
      "fec_ini": "2024-01-09",
      "fec_fin": null,
      "tiempo": null,
      "id_trabajador": null,
      "prioridad": 4
    }
  ],
  "error": false
}
```

200 – Empty

```
{
  "result": [],
  "error": false
}
```

GET BASE/completados

⚠ Returns all the tasks that are completed. If there are no completed tasks, it will return an empty list.

200 – Results


```
{
  "result": [
    {
      "codTrabajo": "J799",
      "categoria": "Espacio-Tiempo",
      "descripcion": "Conseguir ropa",
      "fecIni": "2029-10-20",
      "fecFin": "1984-10-20",
      "tiempo": -999.9,
      "idTrabajador": {
        "idTrabajador": "T800",
        "dni": "80080080T",
        "nombre": "Arnaldo",
        "apellidos": "Charcheneguer",
        "especialidad": "Espacio-Tiempo",
        "contraseña": "Dame tu ropa",
        "email": "sayonara@volvere.com"
      },
      "prioridad": 2
    }
  ],
}
```


```
"error": false
}
```

200 – Empty

```
{
  "result": [],
  "error": false
}
```

GET BASE/:id/trabajador

 This endpoint is not necessary since the worker is already included in the task object, but it doesn't hurt anyone.

 Returns the worker assigned to a task by its ID. If the task doesn't have a worker assigned, it will return an error message. If the task does not exist, it will return an error message.

200 – Result

```
{
  "result": {
    "idTrabajador": "T800",
    "dni": "80080080T",
    "nombre": "Arnaldo",
    "apellidos": "Charcheneguer",
    "especialidad": "Espacio-Tiempo",
    "contraseña": "Dame tu ropa",
    "email": "sayonara@volvere.com"
  },
}
```

```
"error": false
}
```

404 – Not Found

```
{
  "errorMessage": "El trabajo con ID 'J800' no tiene un
trabajador asignado",
  "error": false
}
```

PUT BASE/asignar/:codTrabajo/:idTrabajador

⚠ Assigns a worker to a task by their IDs. If the task does not exist, it will return an error message. If the worker does not exist, it will return an error message. If the task already has a worker assigned, it will return an error message.

200 – Success

⚠ /api/trabajos/asignar/J800/T800


```
{
  "errorMessage": {
    "codTrabajo": "J800",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Visitar a Sarah Connor",
    "fecIni": "2029-10-20",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",

```




```
    "apellidos": "Charcheneguer",
    "especialidad": "Espacio-Tiempo",
    "contraseña": "Dame tu ropa",
    "email": "sayonara@volvere.com"
  },
  "prioridad": 1
},
"error": false
}
```

404 – Task Not Found

 /api/trabajos/asignar/J8000/T8000


```
{
  "errorMessage": "Trabajo no encontrado",
  "error": true
}
```

404 – Worker Not Found

 /api/trabajos/asignar/J800/T8000


```
{
  "errorMessage": "Trabajador no encontrado",
  "error": true
}
```

400 – Worker already assigned


 /api/trabajos/asignar/J001/T800

```
{
  "errorMessage": "El trabajo ya tiene un trabajador
asignado",
  "error": true
}
```


PUT BASE/finalizar/:id

 Accepts from 1 to 3 parameters:


- id (@PathVariable) (required): task ID

 Mandatory parameter.

- fec_fin (@RequestParam) (not required): task completion date
- tiempo (@RequestParam) (not required): duration of the task

 Marks a task as completed by setting the completion date and the duration of the task. Checks if the task exists. Checks if the task has a worker assigned. Checks that tiempo is not greater than 8 hours per day. Checks that fec_fin is not before fecIni. Checks that fec_fin is not after the current date. If fec_fin is not provided, it will set the current date. If tiempo is not provided, it will set the duration to 8 hours per day (not the best algorithm, as no one works 8 hours every day). If the task is already completed, just updates following the same rules mentioned above.

200 – No parameters


 /api/trabajos/finalizar/J8

```

{
  "result": {
    "codTrabajo": "J8",
    "categoria": "Carpintería",
    "descripcion": "Lijar maderos",
    "fecIni": "2024-01-09",
    "fecFin": "2024-05-15",
    "tiempo": 48,
    "idTrabajador": {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Juan",
      "apellidos": "Pérez",
      "especialidad": "Carpintería",
      "contraseña": "contraseña1",
      "email": "juan.perez@example.com"
    },
    "prioridad": 4
  },
  "error": false
}

```

200 – Same date as start date, no time

 /api/trabajos/finalizar/J8?fec_fin=2024-01-09


```

{
  "result": {
    "codTrabajo": "J8",
    "categoria": "Carpintería",
    "descripcion": "Lijar maderos",
    "fecIni": "2024-01-09",
    "fecFin": "2024-01-09",
    "tiempo": 8,
    "idTrabajador": {

```

```
    "idTrabajador": "T001",
    "dni": "12345678A",
    "nombre": "Juan",
    "apellidos": "Pérez",
    "especialidad": "Carpintería",
    "contraseña": "contraseña1",
    "email": "juan.perez@example.com"
  },
  "prioridad": 4
},
"error": false
}
```


200 – Next day, no time

 /api/trabajos/finalizar/J8?fec_fin=2024-01-10

```
{
  "result": {
    "codTrabajo": "J8",
    "categoria": "Carpintería",
    "descripcion": "Lijar maderos",
    "fecIni": "2024-01-09",
    "fecFin": "2024-01-10",
    "tiempo": 8,
    "idTrabajador": {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Juan",
      "apellidos": "Pérez",
      "especialidad": "Carpintería",
      "contraseña": "contraseña1",
      "email": "juan.perez@example.com"
    },
    "prioridad": 4
  }
}
```


```
},
"error": false
}
```

200 – No day, acceptable time

 /api/trabajos/finalizar/J8?tiempo=10


```
{
  "result": {
    "codTrabajo": "J8",
    "categoria": "Carpintería",
    "descripcion": "Lijar maderos",
    "fecIni": "2024-01-09",
    "fecFin": "2024-05-15",
    "tiempo": 10,
    "idTrabajador": {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Juan",
      "apellidos": "Pérez",
      "especialidad": "Carpintería",
      "contraseña": "contraseña1",
      "email": "juan.perez@example.com"
    },
    "prioridad": 4
  },
  "error": false
}
```

400 – Acceptable day, unacceptable time

 /api/trabajos/finalizar/J8?fec_fin=2024-01-10&tiempo=10

```
{
  "errorMessage": "El tiempo no puede ser mayor a 8 horas por
día trabajado",
  "error": true
}
```

404 – Not Found

 /api/trabajos/finalizar/J009


```
{
  "errorMessage": "Trabajo no encontrado",
  "error": true
}
```

400 – Worker not assigned

 /api/trabajos/finalizar/J008


```
{
  "errorMessage": "El trabajo no tiene un trabajador
asignado",
  "error": true
}
```

400 – Date before start date

 /api/trabajos/finalizar/J8?fec_fin=2024-01-04


```
{
  "errorMessage": "La fecha de finalización no puede ser menor
a la fecha de inicio",
  "error": true
}
```

400 – Date after current date

 /api/trabajos/finalizar/J8?fec_fin=2094-01-10


```
{
  "errorMessage": "La fecha de finalización no puede ser mayor
a la fecha actual",
  "error": true
}
```

400 – More than 8h/day

 /api/trabajos/finalizar/J8?fec_fin=2024-01-10&tiempo=200

```
{
  "errorMessage": "El tiempo no puede ser mayor a 8 horas por
día trabajado",
  "error": true
}
```

400 – Negative time

 /api/trabajos/finalizar/J8?fec_fin=2024-01-10&tiempo=-2

```
{
  "errorMessage": "El tiempo no puede ser menor a 0",
  "error": true
}
```

PUT BASE/editarFechaFin/:id

⚠ This endpoint is necessary, as simple PUT BASE /:id does not check anything, and PUT BASE /finalizar/:id will calculate tiempo when it's not provided.

⚠ Same as PUT BASE /finalizar/:id but only for the completion date.

200 – Acceptable date

⚠ /api/trabajos/editarFechaFin/J8?fec_fin=2024-01-10

```
{
  "result": {
    "codTrabajo": "J8",
    "categoria": "Carpintería",
    "descripcion": "Lijar maderos",
    "fecIni": "2024-01-09",
    "fecFin": "2024-01-10",
    "tiempo": 2.0,
    "idTrabajador": {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Juan",
      "apellidos": "Pérez",
      "especialidad": "Carpintería",
      "contraseña": "contraseña1",
      "email": "juan.perez@example.com"
    }
  },
}
```



```
    "prioridad": 4
  },
  "error": false
}
```

200 – No parameter

 /api/trabajos/editarFechaFin/J8

```
{
  "result": {
    "codTrabajo": "J8",
    "categoria": "Carpintería",
    "descripcion": "Lijar maderos",
    "fecIni": "2024-01-09",
    "fecFin": "2024-05-15",
    "tiempo": 2.0,
    "idTrabajador": {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Juan",
      "apellidos": "Pérez",
      "especialidad": "Carpintería",
      "contraseña": "contraseña1",
      "email": "juan.perez@example.com"
    },
    "prioridad": 4
  },
  "error": false
}
```

400 – Date before start date

⚠️ /api/trabajos/editarFechaFin/J8?fec_fin=2024-01-05

```
{
  "errorMessage": "La fecha de finalización no puede ser menor
a la fecha de inicio",
  "error": true
}
```

400 – Date after current date

⚠️ /api/trabajos/editarFechaFin/J8?fec_fin=2094-01-10


```
{
  "errorMessage": "La fecha de finalización no puede ser mayor
a la fecha actual",
  "error": true
}
```

PUT BASE/editarTiempo/:id

⚠️ This endpoint is necessary, as simple PUT BASE /:id does not check anything, and PUT BASE /finalizar/:id will assume today's date when fec_fin is not provided.

⚠️ Same as PUT BASE /finalizar/:id but only for the duration of the task. Also checks that fec_fin is not null

200 – Acceptable time

 /api/trabajos/editarTiempo/J8?tiempo=4

```
{
  "result": {
    "codTrabajo": "J8",
    "categoria": "Carpintería",
    "descripcion": "Lijar maderos",
    "fecIni": "2024-01-09",
    "fecFin": "2024-01-10",
    "tiempo": 4.0,
    "idTrabajador": {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Juan",
      "apellidos": "Pérez",
      "especialidad": "Carpintería",
      "contraseña": "contraseña1",
      "email": "juan.perez@example.com"
    },
    "prioridad": 4
  },
  "error": false
}
```


200 – No parameter

 /api/trabajos/editarTiempo/J8

```
{
  "result": {
    "codTrabajo": "J8",
    "categoria": "Carpintería",
    "descripcion": "Lijar maderos",
```


```
"fecIni": "2024-01-09",
"fecFin": "2024-05-15",
"tiempo": 48.0,
"idTrabajador": {
  "idTrabajador": "T001",
  "dni": "12345678A",
  "nombre": "Juan",
  "apellidos": "Pérez",
  "especialidad": "Carpintería",
  "contraseña": "contraseña1",
  "email": "juan.perez@example.com"
},
"prioridad": 4
},
"error": false
}
```

400 – Date before start date

 /api/trabajos/editarTiempo/J8?fec_fin=2024-01-05

```
{
  "errorMessage": "La fecha de finalización no puede ser menor
a la fecha de inicio",
  "error": true
}
```


400 – Date after current date

 /api/trabajos/editarTiempo/J8?fec_fin=2094-01-10

```
{
  "errorMessage": "La fecha de finalización no puede ser mayor
```

```
a la fecha actual",  
  "error": true  
}
```

400 – Null date


 /api/trabajos/editarFechaFin/J8?tiempo=3

Assuming fec_fin is null

```
{  
  "errorMessage": "El trabajo no tiene una fecha de  
finalización asignada",  
  "error": true  
}
```


Trabajador

Model

 Trabajador
<ul style="list-style-type: none">❑ contraseña: String❑ trabajos: Set<Trabajo>❑ idTrabajador: String❑ especialidad: String❑ nombre: String❑ email: String❑ dni: String❑ apellidos: String
<ul style="list-style-type: none">● Trabajador(String, String, String, String, String, String, String):● Trabajador():● Trabajador(String, String, String, String, String, String, String, String):● setDni(String): void● setEspecialidad(String): void● setTrabajos(Set<Trabajo>): void● toString(): String● setApellidos(String): void● getEspecialidad(): String● setContraseña(String): void● getApellidos(): String● getIdTrabajador(): String● getEmail(): String● setNombre(String): void● setEmail(String): void● getNombre(): String● getTrabajos(): Set<Trabajo>● getDni(): String● setIdTrabajador(String): void● getContraseña(): String

BASE: /api/trabajadores

GET BASE

 Returns all the workers in the database. If there are no workers, it will return an empty list.


Successful response

```
{
  "result": [
    {
      "idTrabajador": "T003",
      "dni": "34567890C",
      "nombre": "Carlos",
      "apellidos": "Martínez",
      "especialidad": "Electricidad",
      "contraseña": "contraseña3",
      "email": "carlos.martinez@example.com"
    },
    {
      "idTrabajador": "T001",
      "dni": "12345678A",
      "nombre": "Juan",
      "apellidos": "Pérez",
      "especialidad": "Carpintería",
      "contraseña": "contraseña1",
      "email": "juan.perez@example.com"
    }
  ],
  "error": false
}
```


Empty list response

```
{
  "result": [],
  "error": false
}
```

GET BASE/:id


 Returns a worker by its ID. If the worker does not exist, it will return an error message.

Successful response

 /api/trabajadores/T003

```
{
  "result": {
    "idTrabajador": "T003",
    "dni": "34567890C",
    "nombre": "Carlos",
    "apellidos": "Martínez",
    "especialidad": "Electricidad",
    "contraseña": "contraseña3",
    "email": "carlos.martinez@example.com"
  },
  "error": false
}
```

Non-existent worker response

 /api/trabajadores/asdtg

```
{
  "errorMessage": "El trabajador con ID: 'asdtg' no existe en la base de datos",
  "error": true
}
```

POST BASE



Create a new worker in the database. Returns the worker created. If there are any errors, it will return an error message with the list of errors. If the worker already exists, it will return an error message.

All the fields are required. ID, DNI and email are unique fields.

Successful body

```
POST /api/trabajadores
Content-Type: application/json
{
  "idTrabajador": "T001",
  "dni": "12345678A",
  "nombre": "Pepe",
  "apellidos": "García",
  "especialidad": "Fontanería",
  "contraseña": "123",
  "email": "wefwefewf@ergerg.com"
}
```

Response

```
{
  "result": {
    "idTrabajador": "T001",
    "dni": "12345678A",
    "nombre": "Pepe",
    "apellidos": "García",
    "especialidad": "Fontanería",
    "contraseña": "123",
    "email": "wefwefewf@ergerg.com"
  },
}
```

```
"error": false
}
```

Error body

```
POST /api/trabajadores
Content-Type: application/json
{
  "idTrabajador": "T001",
  "dni": "",
  "nombre": "Pepe",
  "apellidos": "García",
  "especialidad": "Fontanería",
  "contraseña": "123",
  "email": "holaHolita"
}
```

Error response

```
{
  "errorsList": [
    "El campo DNI no puede estar vacío",
    "El email no es válido"
  ],
  "error": true
}
```

Existent worker body


```
POST /api/trabajadores
Content-Type: application/json
{
  "idTrabajador": "T001",
```

```
"dni": "12312312T",
"nombre": "Pepe",
"apellidos": "García",
"especialidad": "Fontanería",
"contraseña": "123",
"email": "pepe@garcia.com"
}
```


Existent worker response

```
{
  "errorsList": [
    "Ya existe un trabajador con el ID: T001"
  ],
  "error": true
}
```

PUT BASE/:id

 Update a worker by its ID. Returns the worker updated. If the worker does not exist, it will return an error message. If any of the fields are empty or invalid, it will return an error message with the list of errors.

Successful body

 /api/trabajadores/T001


```
PUT /api/trabajadores
Content-Type: application/json
{
  "idTrabajador": "T001",
  "dni": "12345678A",
  "nombre": "Pepe",
```

```
"apellidos": "Gomez",
"especialidad": "Fontanería",
"contraseña": "123",
"email": "wefwefewf@ergerg.com"
}
```

Response

```
{
  "result": {
    "idTrabajador": "T001",
    "dni": "12345678A",
    "nombre": "Pepe",
    "apellidos": "Gomez",
    "especialidad": "Fontanería",
    "contraseña": "123",
    "email": "wefwefewf@ergerg.com"
  },
  "error": false
}
```

Error body

 /api/trabajadores/T001

```
PUT /api/trabajadores
Content-Type: application/json
{
  "idTrabajador": "T001",
  "dni": "",
  "nombre": "Pepe",
  "apellidos": "García",
  "especialidad": "Fontanería",
  "contraseña": "123",


```

```
"email": "holaHolita"
}
```

Error response

```
{
  "errorsList": [
    "El campo DNI no puede estar vacío",
    "El email no es válido"
  ],
  "error": true
}
```

Non-existent worker body

 /api/trabajadores/T999


```
PUT /api/trabajadores
Content-Type: application/json
{
  "idTrabajador": "T999",
  "dni": "12344321T",
  "nombre": "José",
  "apellidos": "Jimenez",
  "especialidad": "Ebanistería",
  "contraseña": "123",
  "email": "jose@jimenez.com"
}
```

Non-existent worker response


```
{
  "errorsList": [
```

```
    "No se pudo editar, el trabajador con ID 'T999' no existe  
    en la base de datos"  
  ],  
  "error": true  
}
```

DELETE BASE/:id


 Delete a worker by its ID. Returns a response without errors. If the worker does not exist, it will return an error message.

Successful response

 /api/trabajos/T001


```
{  
  "error": false  
}
```

Non-existent worker response


 /api/trabajos/T999

```
{  
  "errorMessage": "El trabajador con ID 'T999' no existe en la  
base de datos",  
  "error": true  
}
```

GET BASE/:id/:contraseña


 Returns the pending tasks of a worker by its ID and password. If the worker does not exist, it will return an error message. If the credentials are incorrect, it will return an error message. If the worker has no pending tasks, it will return an empty list.

200 – empty

 /api/trabajadores/T800/Dame%20tu%20ropa

```
{
  "result": [],
  "error": false
}
```

200 – results

 /api/trabajadores/T003/contrase%C3%B1a3

```
{
  "result": [
    {
      "codTrabajo": "J003",
      "categoria": "Electricidad",
      "descripcion": "Instalación de luces",
      "fecIni": "2022-01-03",
      "fecFin": null,
      "tiempo": 8.0,
      "idTrabajador": {
        "idTrabajador": "T003",
        "dni": "34567890C",
        "nombre": "Carlos",
        "apellidos": "Martínez",
        "especialidad": "Electricidad",

```

```

        "contraseña": "contraseña3",
        "email": "carlos.martinez@example.com"
    },
    "prioridad": 3
},
{
    "codTrabajo": "J004",
    "categoria": "Electricidad",
    "descripcion": "Desinstalación de luces",
    "fecIni": "2022-01-03",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
        "idTrabajador": "T003",
        "dni": "34567890C",
        "nombre": "Carlos",
        "apellidos": "Martínez",
        "especialidad": "Electricidad",
        "contraseña": "contraseña3",
        "email": "carlos.martinez@example.com"
    },
    "prioridad": 4
}
],
"error": false
}

```

404 – Not Found

 /api/trabajadores/T009/Dame_tu_SONRISA

```

{
    "errorMessage": "El trabajador con ID: 'T009' no existe en
la base de datos",

```



```
"error": true
}
```

404 – Not Found

⚠ /api/trabajadores/T800/Dame_tu_SONRISA

```
{
  "errorMessage": "Las credenciales son incorrectas",
  "error": true
}
```

GET BASE/especialidad/:especialidad

⚠ Returns all the workers with a specific specialty. If there are no workers with that specialty, it will return an empty list.

200 – results

⚠ /api/trabajadores/especialidad/espacio-tiempo

```
{
  "result": [
    {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    }
  ]
}
```

```
],  
  "error": false  
}
```

200 – empty

⚠ /api/trabajadores/especialidad/hghfgtgcghvuhbk

```
{  
  "result": [],  
  "error": false  
}
```

GET BASE/:id/trabajos

⚠ Returns a list of all the tasks of a worker by its ID. If the worker has no tasks, it will return an empty list. If the worker does not exist, it will return an error message.

200 – results

⚠ /api/trabajadores/T800/trabajos


```
{  
  "result": [  
    {  
      "codTrabajo": "J799",  
      "categoria": "Espacio-Tiempo",  
      "descripcion": "Conseguir ropa",  
      "fecIni": "2029-10-20",  
      "fecFin": "1984-10-20",  
      "tiempo": -999.9,  
    }  
  ]  
}
```

```

    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 2
  },
  {
    "codTrabajo": "J800",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Conseguir ropa",
    "fecIni": "2029-10-20",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 2
  }
],
"error": false
}


```

200 – empty

 /api/trabajadores/TDUDE/trabajos


```
{
  "result": [],
  "error": false
}
```

404 – ID Not Found

 /api/trabajadores/T900/trabajos

```
{
  "errorMessage": "El trabajador con ID: 'T9' no existe en la
base de datos",
  "error": true
}
```

GET BASE/:id/trabajos/pendientes

 Returns a list of all the pending tasks of a worker by its ID. If the worker has no pending tasks, it will return an empty list. If the worker does not exist, it will return an error message.

200 – Results

 /api/trabajadores/T800/trabajos/pendientes

```
{
  "result": [
    {
```


```

    "codTrabajo": "J800",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Visitar a Sarah Connor",
    "fecIni": "2029-10-20",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 1
  },
  {
    "codTrabajo": "J803",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Sonreír",
    "fecIni": "2029-10-20",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 4
  },
  {
    "codTrabajo": "J801",
    "categoria": "Espacio-Tiempo",

```

```
"descripcion": "Conseguir un amoto",
"fecIni": "2029-10-20",
"fecFin": null,
"tiempo": null,
"idTrabajador": {
  "idTrabajador": "T800",
  "dni": "80080080T",
  "nombre": "Arnaldo",
  "apellidos": "Charcheneguer",
  "especialidad": "Espacio-Tiempo",
  "contraseña": "Dame tu ropa",
  "email": "sayonara@volvere.com"
},
"prioridad": 2
},
],
"error": false
}
```

200 – Empty

 /api/trabajadores/T800/trabajos/pendientes

```
{
  "result": [],
  "error": false
}
```

404 – Worker not found

 /api/trabajadores/T099/trabajos/pendientes

```
{
  "errorMessage": "El trabajador con ID: 'T099' no existe en
la base de datos",
  "error": true
}
```

GET BASE/:id/trabajos/pendientes/prioridad

⚠ Returns an ascendant sorted list of all the pending tasks of a worker by its ID. If the worker has no pending tasks, it will return an empty list. If the worker does not exist, it will return an error message.

200 – Results

⚠ /api/trabajadores/T800/trabajos/pendientes/prioridad

```
{
  "result": [
    {
      "codTrabajo": "J800",
      "categoria": "Espacio-Tiempo",
      "descripcion": "Visitar a Sarah Connor",
      "fecIni": "2029-10-20",
      "fecFin": null,
      "tiempo": null,
      "idTrabajador": {
        "idTrabajador": "T800",
        "dni": "80080080T",
        "nombre": "Arnaldo",
        "apellidos": "Charcheneguer",
        "especialidad": "Espacio-Tiempo",
        "contraseña": "Dame tu ropa",
        "email": "sayonara@volvere.com"
      }
    },
  ],
}
```

```


    "prioridad": 1
  },
  {
    "codTrabajo": "J801",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Conseguir un amoto",
    "fecIni": "2029-10-20",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 2
  },
  {
    "codTrabajo": "J803",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Sonreír",
    "fecIni": "2029-10-20",
    "fecFin": null,
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 4
  }
}

```




```
],  
  "error": false  
}
```

200 – Empty

 /api/trabajadores/T800/trabajos/pendientes/prioridad


```
{  
  "result": [],  
  "error": false  
}
```

404 – Worker not found


 /api/trabajadores/T099/trabajos/pendientes/prioridad

```
{  
  "errorMessage": "El trabajador con ID: 'T099' no existe en  
la base de datos",  
  "error": true  
}
```

GET BASE/:id/trabajos/pendientes/:prioridad


 Returns a list of all the pending tasks of a worker by its ID and the specified priority. If the worker has no pending tasks, it will return an empty list. If the worker does not exist, it will return an error message.

200 – Results

 /api/trabajadores/T800/trabajos/pendientes/4

```
{
  "result": [
    {
      "codTrabajo": "J803",
      "categoria": "Espacio-Tiempo",
      "descripcion": "Sonreír",
      "fecIni": "2029-10-20",
      "fecFin": null,
      "tiempo": null,
      "idTrabajador": {
        "idTrabajador": "T800",
        "dni": "80080080T",
        "nombre": "Arnaldo",
        "apellidos": "Charcheneguer",
        "especialidad": "Espacio-Tiempo",
        "contraseña": "Dame tu ropa",
        "email": "sayonara@volvere.com"
      },
      "prioridad": 4
    }
  ],
  "error": false
}
```


200 – Empty

 /api/trabajadores/T800/trabajos/pendientes/3

```
{
  "result": [],
}
```


```
"error": false
}
```


404 – Worker not found

 /api/trabajadores/T099/trabajos/pendientes/3


```
{
  "errorMessage": "El trabajador con ID: 'T099' no existe en
la base de datos",
  "error": true
}
```

GET BASE/:id/trabajos/completados

 Accepts two optional query parameters: fechaIni and fechaFin

 Returns a list of all the completed tasks of a worker by its ID. If the worker has no completed tasks, it will return an empty list. If the worker does not exist, it will return an error message.

200 – No param | Results

 /api/trabajadores/T800/trabajos/completados

```
{
  "result": [
    {
      "codTrabajo": "J799",
      "categoria": "Espacio-Tiempo",
      "descripcion": "Conseguir ropa",

```

```

    "fecIni": "2029-10-20",
    "fecFin": "1984-10-20",
    "tiempo": -999.9,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 2
  },
  {
    "codTrabajo": "J803",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Sonreír",
    "fecIni": "2029-10-20",
    "fecFin": "1984-10-21",
    "tiempo": -999.9,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 4
  },
  {
    "codTrabajo": "J801",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Conseguir un amoto",
    "fecIni": "2029-10-20",
    "fecFin": "1984-10-20",

```

```

    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 2
  },
  {
    "codTrabajo": "J800",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Visitar a Sarah Connor",
    "fecIni": "2029-10-20",
    "fecFin": "1984-10-23",
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 1
  }
],
"error": false
}

```

200 — No param | Empty

⚠ /api/trabajadores/TDUDE/trabajos/completados

```
{
  "result": [],
  "error": false
}
```

200 — fechaIni | Results

⚠ /api/trabajadores/T800/trabajos/completados?fechaIni=1984-10-21

```
{
  "result": [
    {
      "codTrabajo": "J803",
      "categoria": "Espacio-Tiempo",
      "descripcion": "Sonreír",
      "fecIni": "2029-10-20",
      "fecFin": "1984-10-21",
      "tiempo": -999.9,
      "idTrabajador": {
        "idTrabajador": "T800",
        "dni": "80080080T",
        "nombre": "Arnaldo",
        "apellidos": "Charcheneguer",
        "especialidad": "Espacio-Tiempo",
        "contraseña": "Dame tu ropa",
        "email": "sayonara@volvere.com"
      },
      "prioridad": 4
    },
    {
      "codTrabajo": "J800",
```

```

    "categoria": "Espacio-Tiempo",
    "descripcion": "Visitar a Sarah Connor",
    "fecIni": "2029-10-20",
    "fecFin": "1984-10-23",
    "tiempo": null,
    "idTrabajador": {
      "idTrabajador": "T800",
      "dni": "80080080T",
      "nombre": "Arnaldo",
      "apellidos": "Charcheneguer",
      "especialidad": "Espacio-Tiempo",
      "contraseña": "Dame tu ropa",
      "email": "sayonara@volvere.com"
    },
    "prioridad": 1
  }
],
"error": false
}

```

200 — fechaFin | Results

 /api/trabajadores/T800/trabajos/completados?fechaFin=1984-10-21

```

{
  "result": [
    {
      "codTrabajo": "J799",
      "categoria": "Espacio-Tiempo",
      "descripcion": "Conseguir ropa",
      "fecIni": "2029-10-20",
      "fecFin": "1984-10-20",
      "tiempo": -999.9,
      "idTrabajador": {
        "idTrabajador": "T800",

```

```

        "dni": "80080080T",
        "nombre": "Arnaldo",
        "apellidos": "Charcheneguer",
        "especialidad": "Espacio-Tiempo",
        "contraseña": "Dame tu ropa",
        "email": "sayonara@volvere.com"
    },
    "prioridad": 2
},
{
    "codTrabajo": "J803",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Sonreír",
    "fecIni": "2029-10-20",
    "fecFin": "1984-10-21",
    "tiempo": -999.9,
    "idTrabajador": {
        "idTrabajador": "T800",
        "dni": "80080080T",
        "nombre": "Arnaldo",
        "apellidos": "Charcheneguer",
        "especialidad": "Espacio-Tiempo",
        "contraseña": "Dame tu ropa",
        "email": "sayonara@volvere.com"
    },
    "prioridad": 4
},
{
    "codTrabajo": "J801",
    "categoria": "Espacio-Tiempo",
    "descripcion": "Conseguir un amoto",
    "fecIni": "2029-10-20",
    "fecFin": "1984-10-20",
    "tiempo": null,
    "idTrabajador": {
        "idTrabajador": "T800",
        "dni": "80080080T",
        "nombre": "Arnaldo",

```




```

        "apellidos": "Charcheneguer",
        "especialidad": "Espacio-Tiempo",
        "contraseña": "Dame tu ropa",
        "email": "sayonara@volvere.com"
    },
    "prioridad": 2
}
],
"error": false
}

```

200 — fechaIni + fechaFin | Results

 /api/trabajadores/T800/trabajos/completados?fechaIni=1984-10-21&fechaFin=1984-10-22


```

{
  "result": [
    {
      "codTrabajo": "J803",
      "categoria": "Espacio-Tiempo",
      "descripcion": "Sonreír",
      "fecIni": "2029-10-20",
      "fecFin": "1984-10-21",
      "tiempo": -999.9,
      "idTrabajador": {
        "idTrabajador": "T800",
        "dni": "80080080T",
        "nombre": "Arnoldo",
        "apellidos": "Charcheneguer",
        "especialidad": "Espacio-Tiempo",
        "contraseña": "Dame tu ropa",
        "email": "sayonara@volvere.com"
      },
      "prioridad": 4
    }
  ]
}

```

```
    }  
  ],  
  "error": false  
}
```

404 — Worker not found

 /api/trabajadores/T099/trabajos/pendientes/3

```
{  
  "errorMessage": "El trabajador con ID: 'T099' no existe en  
la base de datos",  
  "error": true  
}
```

Trabajador - Views

GET /trabajadores



Returns a view with all the workers in the database

Trabajadores/indexTrabajadores (<http://localhost:8080/trabajadores>)

ID	Dni	Nombre	Apellidos	Especialidad	Email	Trabajos asociados	Opciones
T003	34567890C	Carlos	Martínez	Electricidad	carlos.martinez@example.com	J003	EDITAR ELIMINAR
T001	12345678A	Pepe	Gómez	Fontanería	pepe@gomez.com	J001 J002	EDITAR ELIMINAR
T002	12341234R	Ana	Gómez	Carpintería	ana@gomez.com	-	EDITAR ELIMINAR

[Volver](#)

trabajadores-table.png

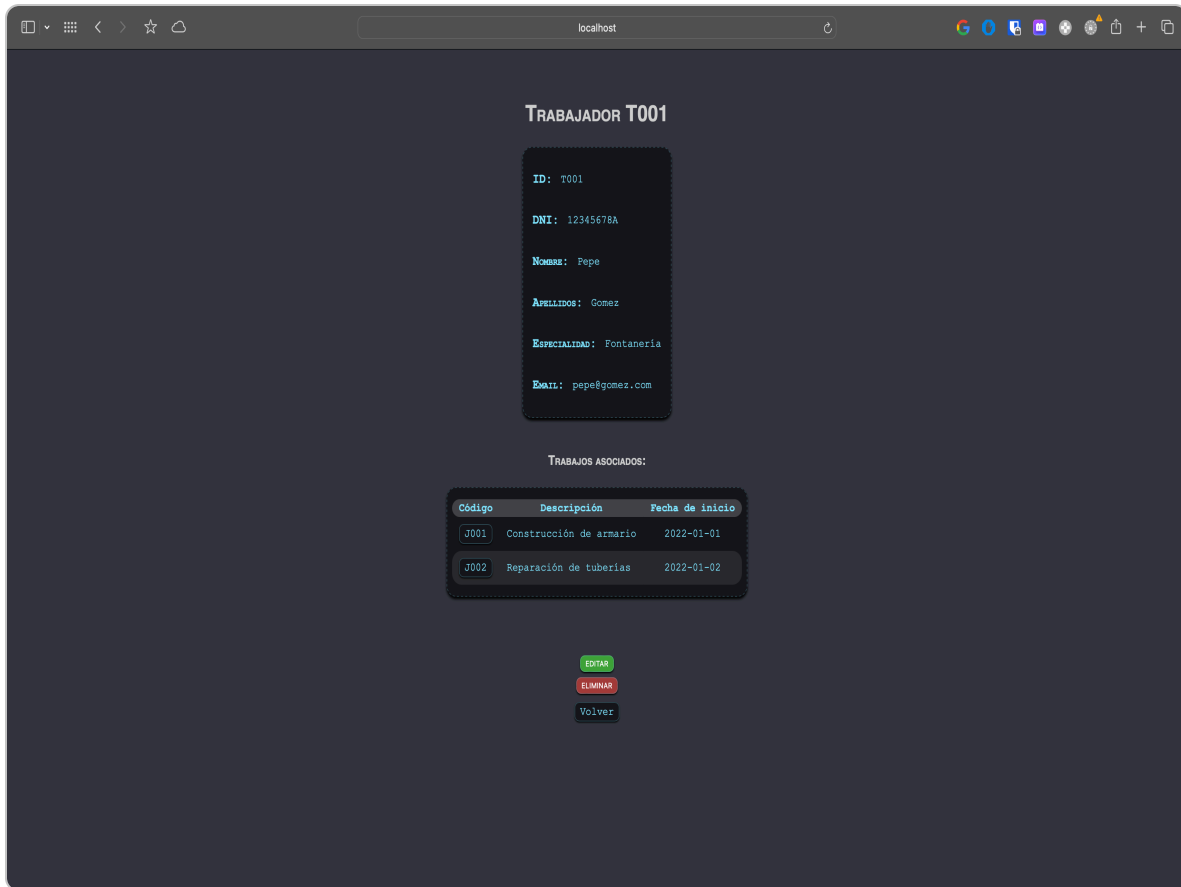
GET /trabajadores/:id



Returns a view with the detailed information of a specific worker

Trabajadores/trabajadoresDetalle

(<http://localhost:8080/trabajadores/T001>)



trabajadores-detail.png

GET /trabajadores/nuevo



Returns a view with a form to create a new worker

Trabajadores/trabajadoresForm (<http://localhost:8080/trabajadores/nuevo>)

AÑADIR TRABAJADOR

Datos del trabajador

ID *

T002

DNI *

12341234R

Nombre

Ana

Apellidos

Gómez

Especialidad *

Carpintería

Contraseña *

Email

ana@gomez.com

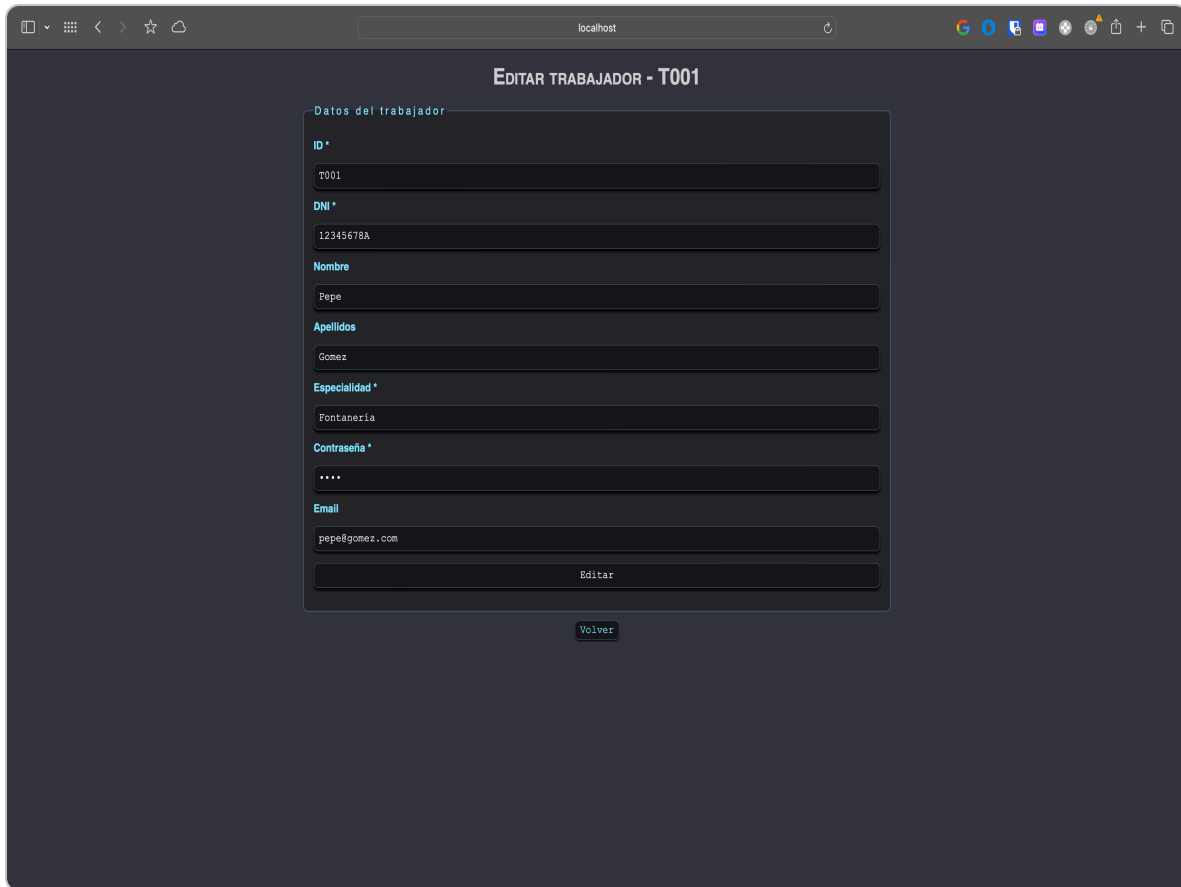
Añadir

Volver

trabajadores-add.png

GET /trabajadores/editar

- ⚠ Returns a view with the same form that 'trabajadores/nuevo', but modified to edit a worker Trabajadores/trabajadoresForm (<http://localhost:8080/trabajadores/editar?idTrabajador=T001>)



trabajadores-edit.png

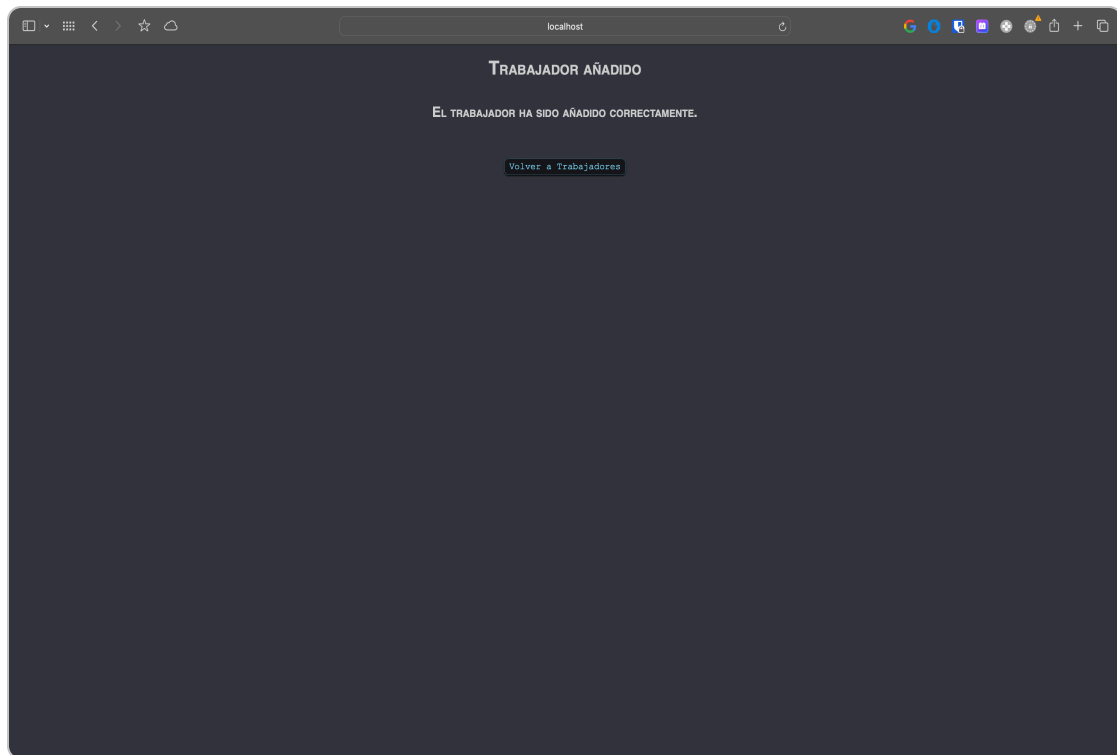
GET /trabajadores/processForm

- ⚠ This endpoint is not meant to be accessed directly, but to process the form in 'trabajadores/nuevo' and 'trabajadores/editar'. It checks if the method is POST or PUT and redirects to the corresponding endpoint.

POST /trabajadores/create

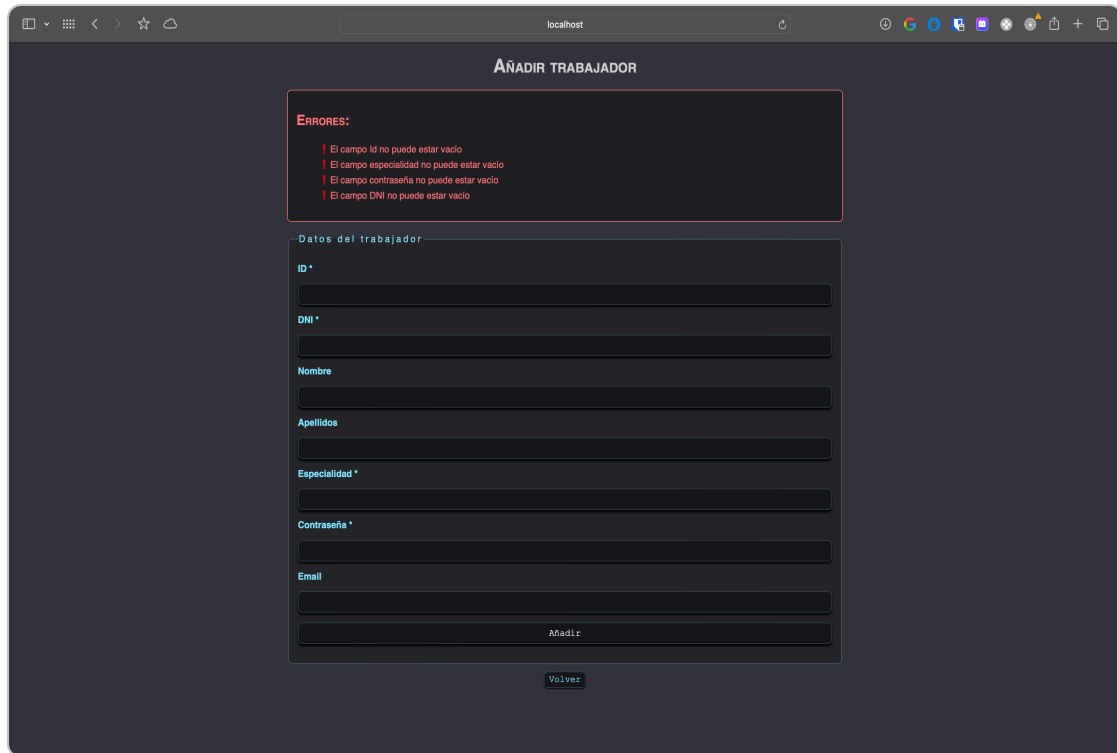
- ⚠ Creates a new worker in the database. If there is no error, redirects to the 'ready' view with a success message. If there is an error, redirects to the 'trabajadores/nuevo' view with an error message.

Successfully worker added



Add success screenshot

Error adding a worker

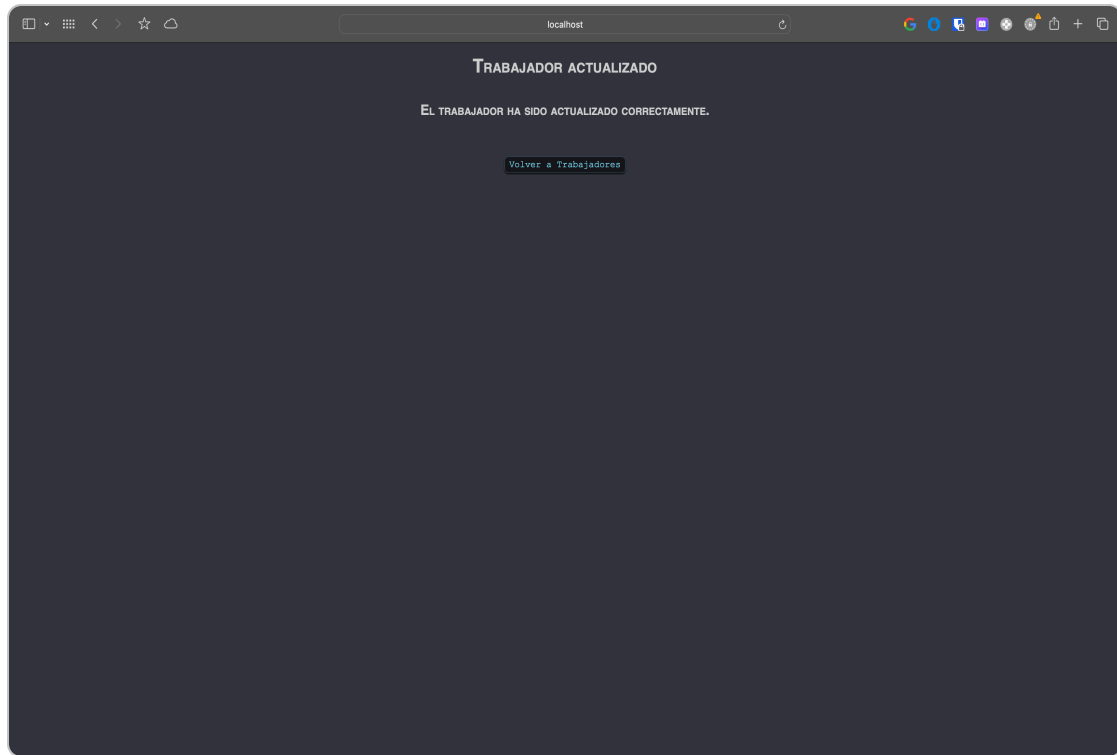


Add error screenshot

PUT /trabajadores/edit

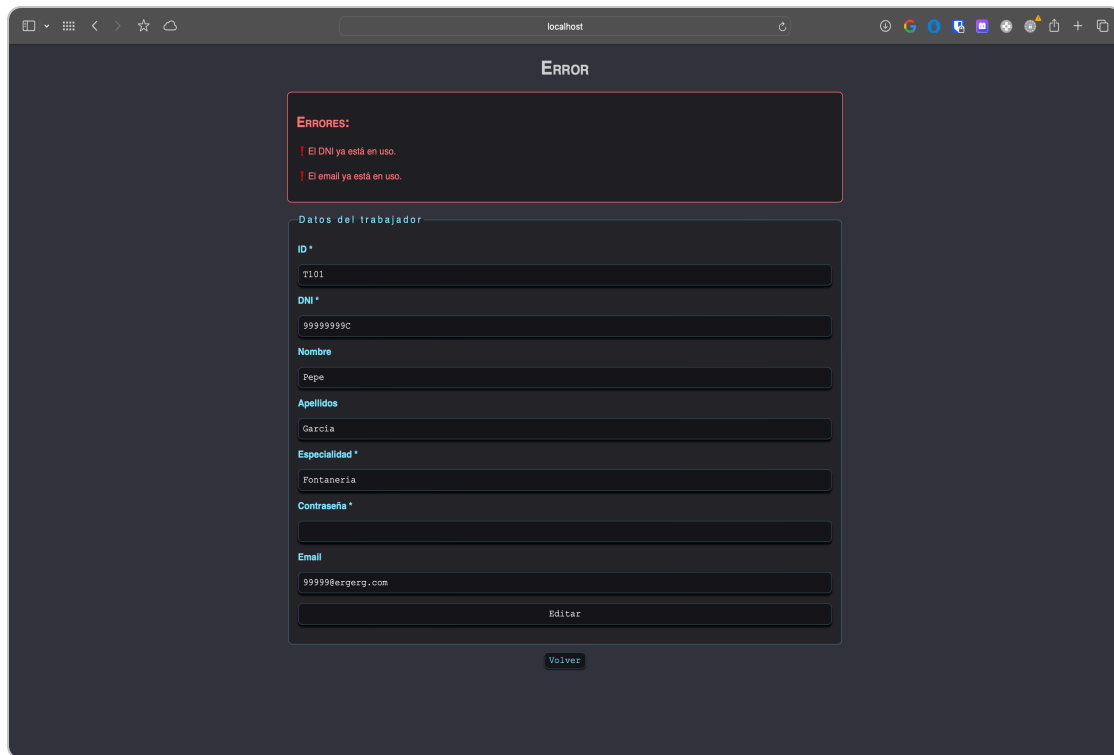
- ⚠ Updates the information of a worker in the database. If there is no error, redirects to the 'ready' view with a success message. If there is an error, redirects to the 'trabajadores/editar' view with an error message.

Successfully worker edited



Edit success screenshot

Error editing a worker



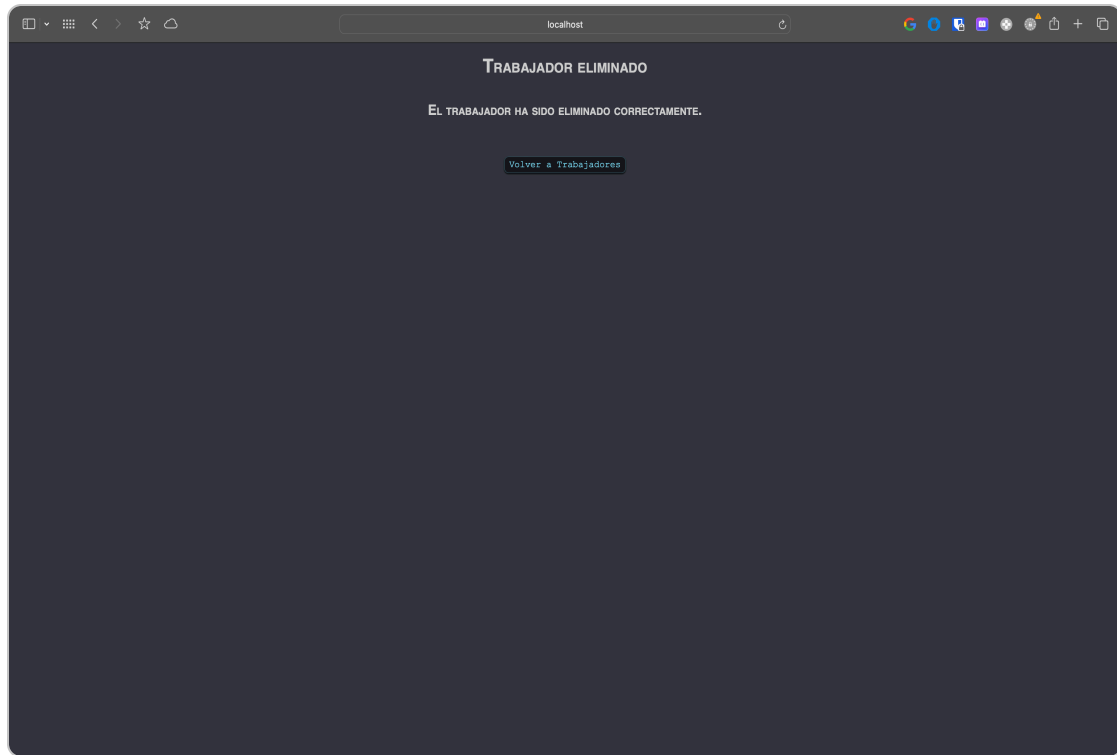
Edit error screenshot

DELETE /trabajadores/delete



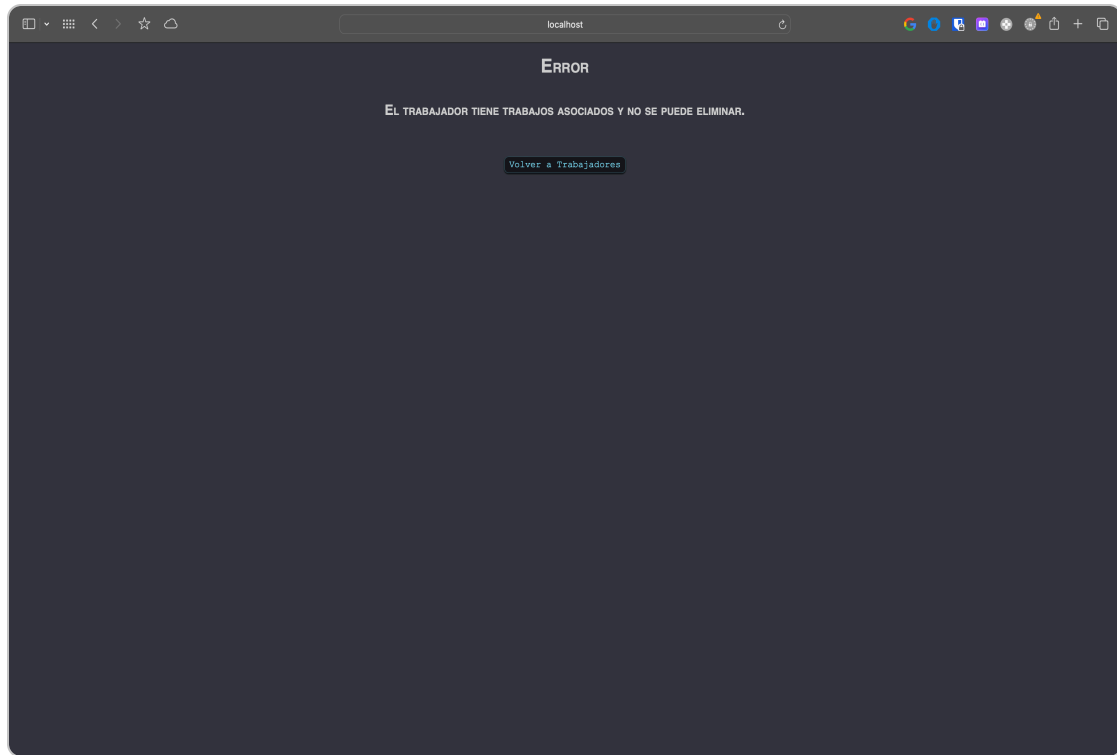
Deletes a worker from the database Redirects to the 'ready' view with a success message Workers with tasks assigned cannot be deleted

Successfully worker deleted



Delete success screenshot

Error deleting a worker



Delete error screenshot

Tutorial

How to enter the administration page

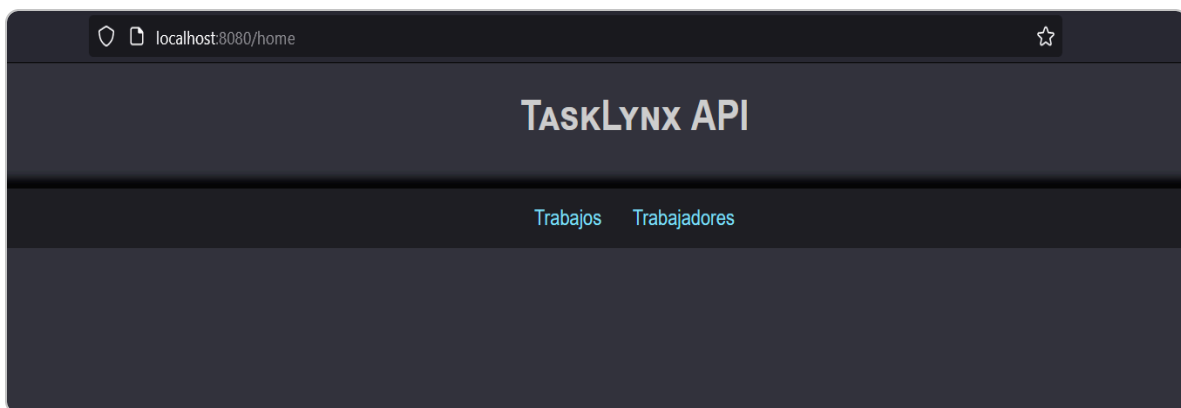
To enter the TaskLynx API administration page, you must access to the URL <http://localhost:8080> (<http://localhost:8080>).

It is also possible to access from:

- <http://localhost:8080/home> (<http://localhost:8080/home>)
- <http://localhost:8080/index> (<http://localhost:8080/index>)

No username or password is required.

The home page is the following:



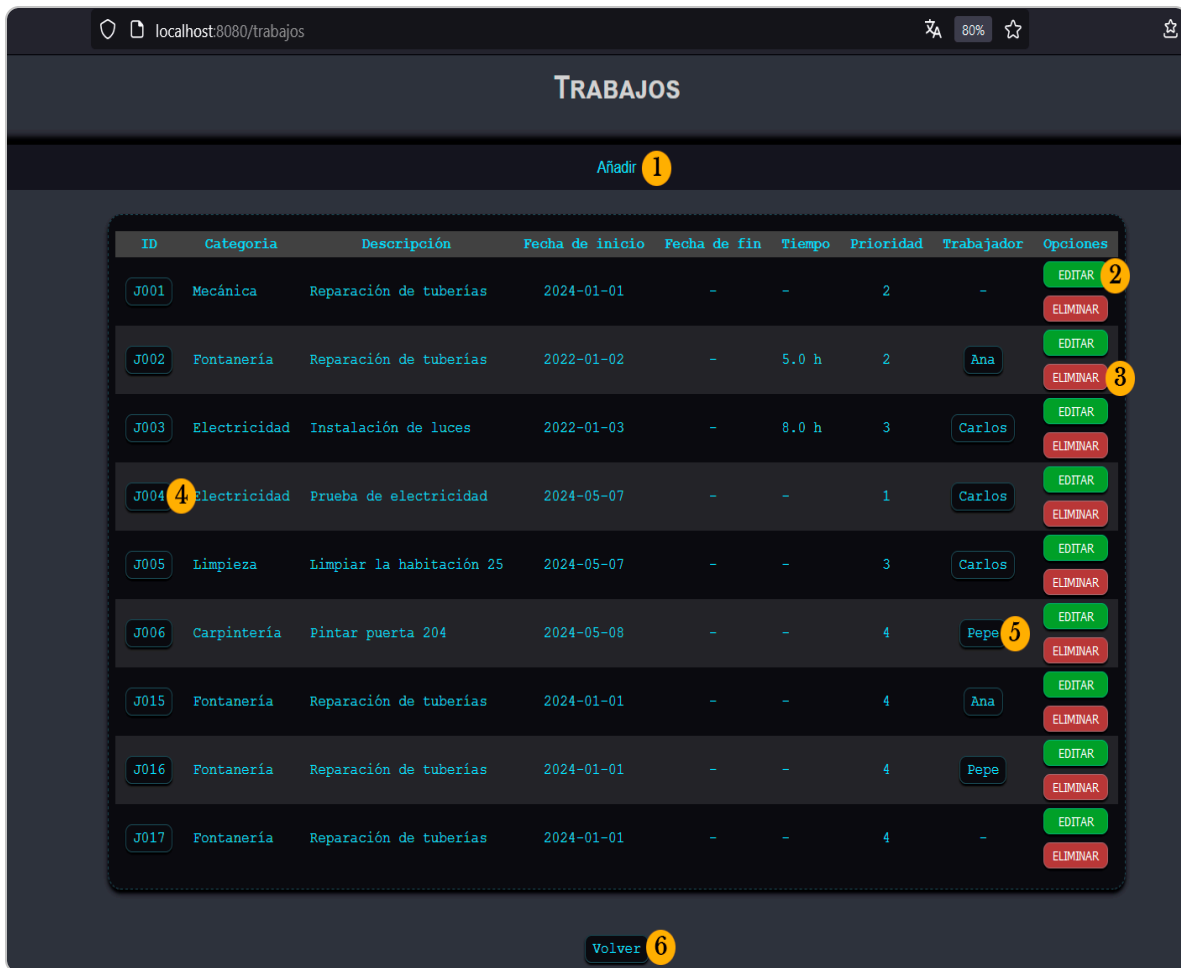
Home

Here you can navigate to the tasks and workers page.

Tasks

Index

The jobs page lists all the jobs that have been created in the application.



Tasks Index

From here, we can:

1. Create a task.
2. Edit a task.
3. Delete a task.
4. Go to the detailed view of a task by clicking its task code.
5. Go to a worker's detailed view by clicking on their name.
6. Go back to Home Page

Add

In this page, you can add a new task to the application.

localhost:8080/trabajos/nuevo

90%

NUEVO TRABAJO

Datos del trabajo

Código del trabajo *:

Categoría *:

Descripción *:

Fecha de inicio *:

dd / mm / aaaa

Trabajador asignado:

-- No asignado --

Prioridad *:

1

Crear trabajo

Volver

Task Add

You must add the information marked as *, the other information is optional. This is an example of a new task.

The screenshot shows a web browser window with the address bar displaying 'localhost:8080/trabajos/nuevo'. The page title is 'NUEVO TRABAJO'. The form is titled 'Datos del trabajo' and contains the following fields:

- Código del trabajo *:** J007
- Categoría *:** Limpieza
- Descripción *:** Limpiar la habitación 120
- Fecha de inicio *:** 13 / 05 / 2024
- Trabajador asignado:** Pepe García
- Prioridad *:** 2

At the bottom of the form is a button labeled 'Crear trabajo'. Below the form is a button labeled 'Volver'.

Task Add with Data

If the information is correct, you will see a page with a confirmation.

The screenshot shows a web browser window with the address bar displaying 'localhost:8080/trabajos/processForm'. The page title is 'TRABAJO AÑADIDO'. The main message is 'EL TRABAJO HA SIDO AÑADIDO CORRECTAMENTE.' Below this message is a button labeled 'Volver a Trabajos'.

Task Add Correct

If not, you will see the errors in the add page.

The screenshot shows a web browser window with the address bar displaying 'localhost:8080/trabajos/processForm'. The page title is 'NUEVO TRABAJO'. At the top right, there are icons for a shield, a document, a 70% zoom level, and a star. The main content area has a dark background. A red-bordered box at the top contains the heading 'ERRORES:' in red, followed by two error messages in red: '! La categoría no puede estar vacía' and '! La fecha de inicio no puede estar vacía'. Below this, a section titled 'Datos del trabajo' contains several form fields: 'Código del trabajo *:' with the value 'J007', 'Categoría *:' which is empty, 'Descripción *:' with the value 'Limpiar la habitación 120', 'Fecha de inicio *:' with a date picker showing 'dd/mm/aaaa', 'Trabajador asignado:' with the value 'Pepe García', and 'Prioridad *:' with the value '2'. At the bottom of this section is a 'Crear trabajo' button. Below the entire form section is a 'Volver' button.

Task Add Errors

Detail

This page show the details of a specific task. You can access this page by clicking a task code on the Tasks index page.



Task Detail

From here, we can:

1. Go to a worker's detailed view by clicking on their name.
2. Edit a task.
3. Delete a task.
4. Go back to Tasks Index

Edit

On this page, you can edit the details of a task. You can access this page by clicking the "EDITAR" button on Tasks index page.

localhost:8080/trabajos/editar?codTrabajo=J004

EDITAR TRABAJO - J004

Datos del trabajo

Código del trabajo *: J004

Categoría *: Electricidad

Descripción *: Prueba de electricidad

Fecha de inicio: Fecha actual 2024-05-07
dd/mm/aaaa

Fecha de fin: dd/mm/aaaa

Tiempo en horas:

Trabajador asignado: Carlos Martínez

Prioridad *: 1

Editar trabajo

Volver

Task Edit

When you modify the information that you need, you can confirm the changes with the button "Editar trabajo".

If the information is correct, you will see a page with a confirmation.

localhost:8080/trabajos/processForm

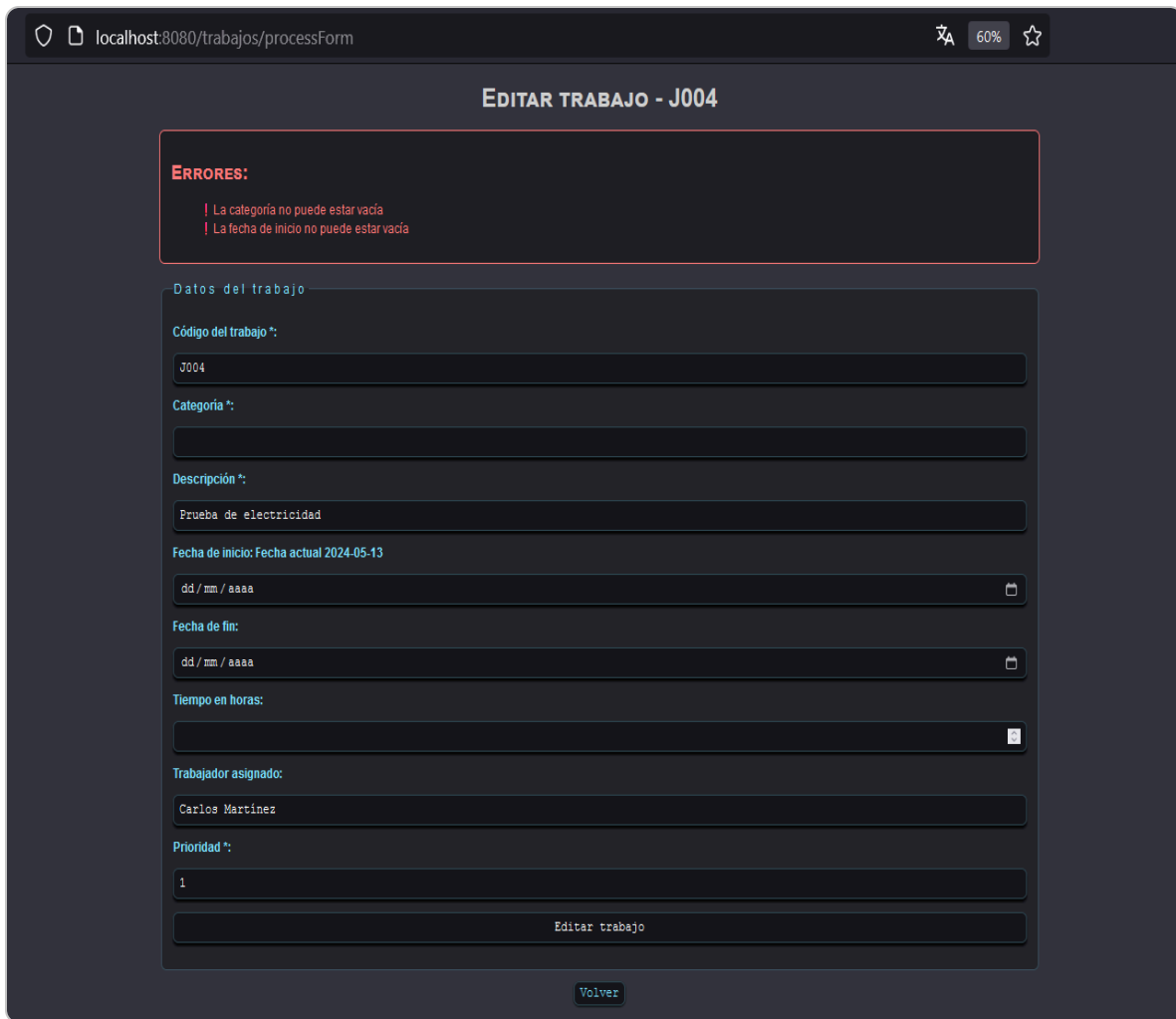
TRABAJO ACTUALIZADO

EL TRABAJO HA SIDO ACTUALIZADO CORRECTAMENTE.

Volver a Trabajos

Task Edit Correct

If not, you will see the errors in the edit page.



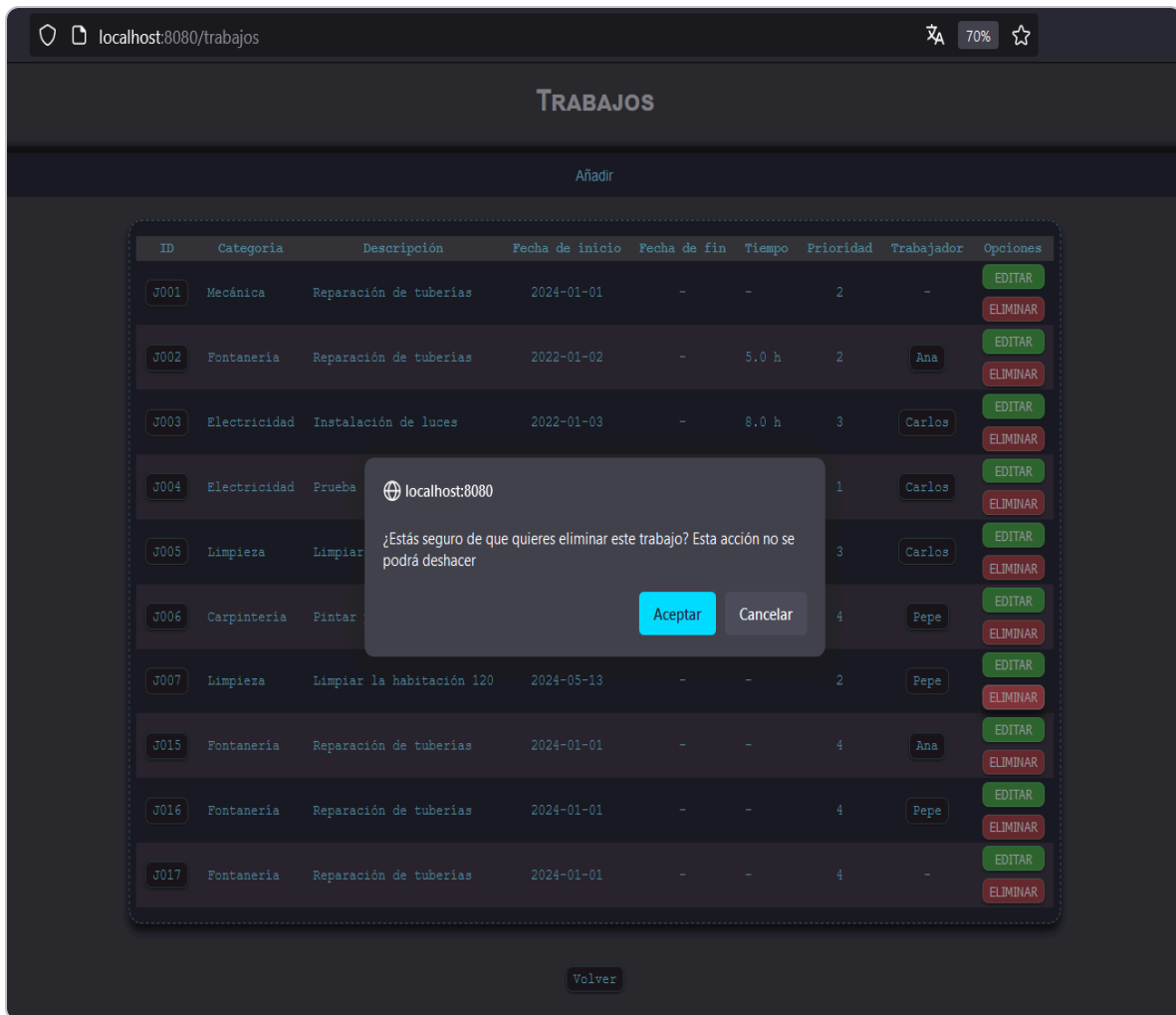
The screenshot shows a web browser window at localhost:8080/trabajos/processForm. The page title is "EDITAR TRABAJO - J004". A red-bordered box at the top contains the heading "ERRORES:" followed by two error messages: "La categoría no puede estar vacía" and "La fecha de inicio no puede estar vacía". Below this, the form is titled "Datos del trabajo" and contains several input fields: "Código del trabajo *" (filled with "J004"), "Categoría *" (empty), "Descripción *" (filled with "Prueba de electricidad"), "Fecha de inicio: Fecha actual 2024-05-13" (calendar icon), "Fecha de fin:" (calendar icon), "Tiempo en horas:" (empty), "Trabajador asignado:" (filled with "Carlos Martinez"), and "Prioridad *" (filled with "1"). At the bottom of the form is a button labeled "Editar trabajo". Below the form is a "Volver" button.

Task Edit Errors

Delete

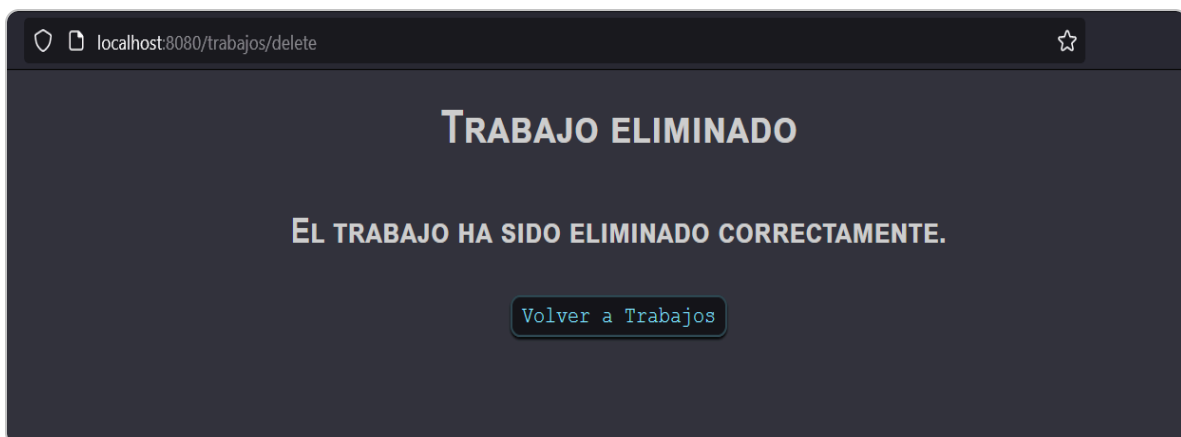
On this page, you can delete a task from the application. You can access this page by clicking the "ELIMINAR" button on Tasks index page.

You must confirm a window to process with task deletion.



Task Delete Confirm

Then, you will see a page with a confirmation.

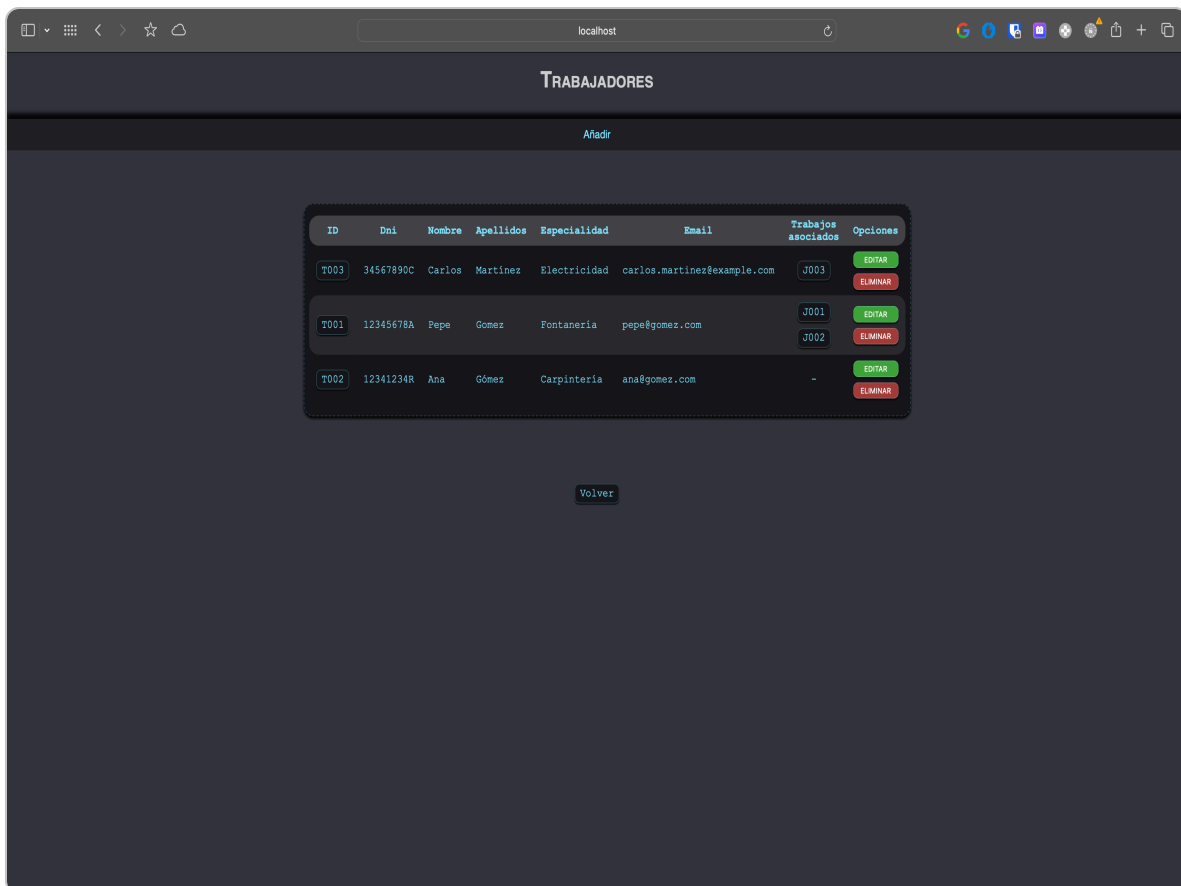


Task Delete Result

Workers

En la vista de trabajadores se listan todos los trabajadores presentes en la aplicación. Es una vista muy similar a la de trabajos, en la que podemos hacer las siguientes acciones:

1. Crear un trabajador.
2. Editar un trabajador.
3. Eliminar un trabajador.
4. Ir a la vista detalle de un trabajador haciendo click en su ID.
5. Ir a la vista detalle de un trabajo haciendo click en su código de trabajo.



The screenshot shows a web browser window with the address bar set to 'localhost'. The page title is 'TRABAJADORES'. Below the title is a button labeled 'Añadir'. The main content area displays a table with the following data:

ID	Dni	Nombre	Apellidos	Especialidad	Email	Trabajos asociados	Opciones
T003	34567890C	Carlos	Martínez	Electricidad	carlos.martinez@example.com	J003	EDITAR ELIMINAR
T001	12345678A	Pepe	Gómez	Fontanería	pepe@gomez.com	J001 J002	EDITAR ELIMINAR
T002	12341234R	Ana	Gómez	Carpintería	ana@gomez.com	-	EDITAR ELIMINAR

Below the table is a button labeled 'Volver'.

trabajadores-table.png

Business – Index



TaskLynx-logo-business-oval.svg

Mobile – Index



TskLynx-mobile-oval-logo.svg

Nest Hotel – Index



Nest Hotel logo

Introduction

En este documento se detalla el despliegue de los proyectos Nest Hotel (<https://github.com/DanielAlmazan/hotel-nest>) mediante Docker y Gestión Hotelera mediante Apache2 + Passenger.

Este despliegue se ha realizado en un servidor VPS con Debian 11.

Despliegue de Gestión Hotelera



Apache 2 logo



Passenger logo

En nuestro VPS con Debian 11, instalamos Apache2

```
sudo apt-get install apache2
```

Configuración básica

Lo primero, por seguridad, nos hacemos una copia del fichero `apache2.conf`, aunque en principio no lo vamos a modificar.

```
sudo cp /etc/apache2/apache2.conf  
/etc/apache2/apache2.conf.original
```

```
sudo nano /etc/apache2/ports.conf
```

```
# If you just change the port or add more ports here, you will  
likely also  
# have to change the VirtualHost statement in  
# /etc/apache2/sites-enabled/000-default.conf  
  
Listen 80  
Listen 8000  
Listen 8080
```

```
Listen 8088
Listen 8888

<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Instalación de Passenger

Para instalar Passenger en el servidor (Debian, en este caso), ejecutamos los siguientes comandos:

```
sudo apt-get install -y dirmngr gnupg

sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80
--recv-keys 561F9B9CAC40B2F7

sudo apt-get install -y apt-transport-https ca-certificates

sudo sh -c 'echo deb https://oss-binaries.phusionpassenger.com/
apt/passenger bullseye main >
/etc/apt/sources.list.d/passenger.list'

sudo apt-get update

sudo apt-get install -y libapache2-mod-passenger
```

Habilitamos el módulo de Passenger

```
sudo a2enmod passenger
```

Seguidamente, reiniciamos Apache2

```
sudo systemctl restart apache2
```

Vamos a comprobar que el módulo de Passenger se ha instalado correctamente

```
sudo /usr/bin/passenger-config validate-install
```

```
[sudo] password for {user}:
```

```
What would you like to validate?
```

```
Use <space> to select.
```

```
If the menu doesn't display correctly, press '!'
```

- Passenger itself
- ● Apache

```
-----  
-----
```

```
Checking whether there are multiple Apache installations...
```

```
Only a single installation detected. This is good.
```

```
-----  
-----
```

```
* Checking whether this Passenger install is in PATH... ✓  
* Checking whether there are no other Passenger installations...  
✓  
* Checking whether Apache is installed... ✓  
* Checking whether the Passenger module is correctly configured  
in Apache... ✓
```

```
Everything looks good. :-)
```

i Es posible que, dependiendo de la configuración de vuestra consola, no se muestre correctamente el menú de opciones. En ese caso, pulsad ! para que se

muestre correctamente.

Creamos el fichero de configuración de nuestro sitio web

```
sudo nano /etc/apache2/sites-available/003-gestion-hotelera.conf
```

```
<VirtualHost *:8080>
    ServerAdmin {email}
    ServerName amorenoiborra.es
    DocumentRoot "/home/despliegue/Gestion-Hotelera"
    PassengerAppRoot "/home/despliegue/Gestion-Hotelera"
    PassengerAppType node
    PassengerStartupFile index.js
    <Directory "/home/despliegue/Gestion-Hotelera">
        Allow from all
        Options -MultiViews
        Require all granted
    </Directory>
</VirtualHost>
```

Indicamos que escuche en el puerto 8080, que es el que hemos configurado en `ports.conf`, que el directorio raíz de nuestro sitio web es `/home/despliegue/Gestion-Hotelera`, que el tipo de aplicación es `node`, que el fichero de inicio es `index.js` y que el directorio `/home/despliegue/Gestion-Hotelera` tiene permisos de lectura para todos los usuarios.

Habilitar el sitio

Creamos un enlace simbólico en `sites-enabled` para habilitar el sitio

```
sudo ln -s /etc/apache2/sites-available/003-gestion-hotelera.conf
\
/etc/apache2/sites-enabled/003-gestion-hotelera.conf
```

En caso de querer deshabilitar el sitio, bastaría con borrar el enlace simbólico.

Recargar Apache2

```
sudo systemctl reload apache2
```

Instalación de NodeJS

Ahora vamos a instalar NodeJS en nuestro servidor Debian 11

```
sudo apt-get install curl  
curl -sL https://deb.nodesource.com/setup_20.x | sudo -E bash -  
sudo apt-get install -y nodejs
```

Comprobamos la versión de NodeJS y de NPM

```
node -v
```

```
v20.0.0
```

```
npm -v
```

```
10.3.0
```

Despliegue de Hotel Nest



Docker logo

Instalación de Docker

Para instalar Docker en el servidor (Debian, en este caso), ejecutamos los siguientes comandos:

```
for pkg in docker.io docker-doc docker-compose podman-docker \
containerd runc; do sudo apt-get remove $pkg; done
```

Crear red de Docker

Para crear una red de Docker, ejecutamos el siguiente comando:

```
sudo docker network create node_mongo
```

Comprobamos que se ha creado correctamente con el siguiente comando:

```
sudo docker network ls
```

NETWORK ID	NAME	DRIVER	SCOPE
d167b3cb6ed3	bridge	bridge	local
d407dbcf94cc	host	host	local
885425ece5f9	node_mongo	bridge	local
fa4b3a84c89b	none	null	local

Crear contenedor de MongoDB

```
sudo docker run -d --name hotelDB --network node_mongo --restart \
unless-stopped -p 127.0.0.1:27017:27017 -v ~/data:/data/db mongo
```

Con este comando, creamos un contenedor de MongoDB llamado `hotelDB` que se conecta a la red `node_mongo`, se reinicia automáticamente si se detiene y se mapea el puerto 27017 del contenedor al puerto 27017 del servidor. Además, se monta el directorio `~/data` del servidor en el directorio `/data/db` del contenedor.

Crear el fichero Dockerfile

```
nano Dockerfile
```

```
# Imagen base
FROM node:latest

# Crear directorio de la aplicación
RUN mkdir -p /usr/src/app

# Establecer directorio de trabajo
WORKDIR /usr/src/app

# Copiar los ficheros package.json y package-lock.json
COPY package*.json ./
# Instalar dependencias
RUN npm install

# Copiar el resto de ficheros
```



```
COPY . .

# Exponer el puerto 3000
EXPOSE 3000

# Comando de inicio
CMD ["npm", "start"]
```

Crear el fichero .dockerignore

```
nano .dockerignore
```

```
node_modules
```

Preparar la aplicación para Docker

```
cat {path-to-project}/hotel-nest/src/app.module.ts
```

```
import { Module } from '@nestjs/common';
import { AppController } from './app.controller';
import { AppService } from './app.service';
import { LimpiezaModule } from './limpieza/limpieza.module';
import { AuthModule } from './auth/auth.module';
import { UsuarioModule } from './usuario/usuario.module';
import { MongooseModule } from '@nestjs/mongoose';
import { config } from 'dotenv';

config();

const dbHost = process.env.DEV_DB_HOST || 'hotelDB';
const dbPort = process.env.DEV_DB_HOST_PORT || '27017';
const devDbName = process.env.DB_NAME || 'hotel';

@Module({
  imports: [
```

```

    LimpiezaModule,

    mongooseModule: mongooseModule,
    mongooseModule.forRoot(`mongodb://${dbHost}:${dbPort}/${devDbName}
`),
    UsuarioModule,
    AuthModule,
    UsuarioModule,
  ],
  controllers: [AppController],
  providers: [AppService],
})
export class AppModule {}

```

⚠ En este caso, se han añadido variables de entorno para poder hacer configuraciones distintas en función del entorno en el que se ejecute la aplicación.

Construir la imagen de Docker

```
sudo docker build -t hotel-nest .
```

Comprobamos que se ha creado correctamente con el siguiente comando:

```
docker images
```

REPOSITORY	TAG	IMAGE	ID	CREATED	SIZE
hotel_nest	latest	76ccfe099c89	2	hours ago	1.33GB
mongo	Latest	ff65a94ec485	2	weeks ago	795MB

Ejecutamos el contenedor con el siguiente comando:

```

sudo docker run -d --name HotelNest --network node_mongo --restart
\
unless-stopped -p 3000:3000 hotel_nest

```

Compromosbar que se ha creado correctamente con el siguiente comando:

```
docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED
3bb36a602b9c	hotel_nest	"docker-entrypoint. S..."	2 hours ago
Up 2 hours	0.0.0.0:3000->3000/tcp, ::: 3000->3000/tcp		
HotelNest			
187d7e16ecbf	mongo	"docker-entrypoint. S..."	2 hours ago
Up 2 hours	127.0.0.1:27017->27017/tcp		
hotelDB			

Comprobar que la aplicación funciona

⚠ Para comprobar que la aplicación funciona correctamente podemos probar en un navegador:

- `http://amorenoiborra.es:3000/limpieza/sucias`
 - Este endpoint nos devuelve las habitaciones que no han sido limpiadas el día de hoy.

```
{
  "habitaciones": [
    {
      "_id": "1a1a1a1a1a1a1a1a1a1a1a1a",
      "numero": 1,
      "tipo": "doble",
      "descripcion": "Habitación doble, cama XL, terraza con vistas al mar",
      "ultimaLimpieza": "2023-09-20T11:24:00.000Z",
      "precio": 59.9,
      "incidencias": [
        {
          "fecha": "2023-09-20T11:24:00.000Z",
          "descripcion": "Incidente de mantenimiento en la terraza",
          "estado": "pendiente",
          "usuario": "usuario1"
        }
      ]
    }
  ]
}
```

```

        "descripcion": "No funciona el aire acondicionado",
        "inicio": "2023-09-19T18:12:54.000Z",
        "_id": "10011a1a1a1a1a1a1a1a1a1a1a"
    },
    {
        "descripcion": "No funciona el interruptor del aseo",
        "inicio": "2023-09-20T10:15:06.000Z",
        "_id": "10021a1a1a1a1a1a1a1a1a1a1a"
    }
],
"__v": 0
},
{
    "_id": "2b2b2b2b2b2b2b2b2b2b2b2b2b",
    "numero": 2,
    "tipo": "familiar",
    "descripcion": "Habitación familiar, cama XL y literas, aseo con bañera",
    "ultimaLimpieza": "2023-08-02T10:35:15.000Z",
    "precio": 65.45,
    "incidencias": [],
    "__v": 0
},
{
    "_id": "3c3c3c3c3c3c3c3c3c3c3c3c3c",
    "numero": 3,
    "tipo": "familiar",
    "descripcion": "Habitación familiar, cama XL y sofá cama, cocina con nevera",
    "precio": 69.15,
    "ultimaLimpieza": "2024-05-18T08:57:42.500Z",
    "incidencias": [],
    "__v": 0
},
{
    "_id": "4d4d4d4d4d4d4d4d4d4d4d4d4d",
    "numero": 4,
    "tipo": "suite",

```

```

    "descripcion": "Habitación con dos camas XL, terraza y
vistas al mar",
    "ultimaLimpieza": "2023-10-10T12:05:10.000Z",
    "precio": 110.2,
    "incidencias": [
      {
        "descripcion": "No funciona el jacuzzi",
        "inicio": "2023-10-08T19:24:43.000Z",
        "_id": "10014d4d4d4d4d4d4d4d4d4d4d4d4d4d"
      }
    ],
    "__v": 0
  }
]
}

```



- <http://amorenoiborra.es:3000/limpieza/limpias>
- Este endpoint nos devuelve las habitaciones que han sido limpiadas el día de hoy.

```

{
  "habitaciones": [
    {
      "_id": "5e5e5e5e5e5e5e5e5e5e5e5e5e5e5e5e",
      "numero": 5,
      "tipo": "individual",
      "descripcion": "Habitación simple, cama 150",
      "precio": 34.65,
      "ultimaLimpieza": "2024-05-28T08:57:42.501Z",
      "incidencias": [],
      "__v": 0
    }
  ]
}

```

```
} ]
```