readme.md for @in.work/in.work

A modern Kanban project management system with bidirectional external sync capabilities. Built as a Jira alternative focused on simplicity, real-time collaboration, and seamless integration with external systems like Gitea, GitLab, and email.

Features

Core Kanban

- **Boards & Columns** - Flexible Kanban boards with customizable columns and WIP limits
- **Cards** - Rich work items with descriptions, priorities, labels, due dates, and time tracking
- **Drag & Drop** - Intuitive card movement between columns
- **Card Links** - Link cards with relationships (blocks, relates to, duplicates, parent/child)

Collaboration

- **Real-time Updates** - WebSocket-powered live updates across all connected clients
- **Presence Indicators** - See who's viewing the same board
- **Typing Indicators** - Know when teammates are commenting
- **Card History** - Full timeline of all changes and movements

External Sync

- **Gitea Integration** - Bidirectional sync with Gitea issues
- **Mirror to External** - Create cards locally, mirror them to external repos
- **Webhook Support** - Real-time updates from external systems
- **Privacy Filter** - AI-powered filtering of sensitive data during sync

Access Control

- **Organizations** - Multi-tenant with organization-level isolation
- **Projects** - Group related boards within organizations
- **Role-Based Access** - Hierarchical permissions (owner, admin, manager, contributor, viewer)
- **Board Sharing** - Share boards across organizations or via public links
- **OAuth/OIDC** - External authentication providers (Gitea, GitLab, etc.)

API & Automation

- **REST API** - Comprehensive API for all operations
- **API Tokens** - Scoped tokens for automation (`inw_` prefix)
- **Audit Logging** - Full audit trail for compliance

Tech Stack

- **Backend**: Deno + TypeScript
- **Database**: MongoDB via @push.rocks/smartdata
- **Frontend**: Angular 19 with signals
- **Real-time**: Native WebSocket
- **AI**: @push.rocks/smartai (configurable per instance)

Quick Start

Prerequisites

- Deno 2.x
- MongoDB 6.x+
- Node.js 20+ (for UI development)

Development Setup

1. Clone the repository:

```
git clone https://github.com/your-org/in.work.git
cd in.work
```

2. Create environment config:

```
mkdir -p .nogit
cat > .nogit/env.json << 'EOF'
{
  "MONGO_URL": "mongodb://localhost:27017",
  "MONGO_DB": "inwork",
  "JWT_SECRET": "your-secret-key-change-in-production"
}
EOF
```

3. Start the development server:

```
deno task dev
```

4. Access the application:

- Web UI: http://localhost:3000
- API Health: http://localhost:3000/api/v1/health
- WebSocket: ws://localhost:3000/ws/realtime

Default Admin

On first run, a platform admin is created:

- Email: `admin@in.work`
- Password: `admin123`

Change this immediately in production!

API Overview

Authentication

```
# Login
curl -X POST http://localhost:3000/api/v1/auth/login \
  -H "Content-Type: application/json" \
  -d '{"email":"admin@in.work","password":"admin123"}'

# Use token in subsequent requests
```

```
curl http://localhost:3000/api/v1/organizations \
  -H "Authorization: Bearer <access_token>"
```

Key Endpoints

Method	Endpoint	Description
POST	`/api/v1/auth/login`	Login
POST	`/api/v1/auth/register`	Register
GET	`/api/v1/organizations`	List organizations
POST	`/api/v1/organizations`	Create organization
GET	`/api/v1/projects`	List projects
GET	`/api/v1/boards/:id`	Get board
GET	`/api/v1/boards/:id/cards`	Get board cards
POST	`/api/v1/boards/:id/cards`	Create card
POST	`/api/v1/cards/:id/move`	Move card
POST	`/api/v1/connections`	Create external connection
POST	`/api/v1/connections/:id/sync`	Trigger sync

Real-time WebSocket

Connect to receive live updates:

```
const ws = new WebSocket('ws://localhost:3000/ws/realtime?token=' + accessToken);

ws.onmessage = (event) => {
  const data = JSON.parse(event.data);
  console.log('Event:', data.type, data.data);
};

// Subscribe to a board
ws.send(JSON.stringify({
  action: 'subscribe',
  roomId: 'Board:xxx',
  roomType: 'board'
```

```
}));
```

Event Types

- `card:created`, `card:updated`, `card:moved`, `card:deleted`
- `column:created`, `column:updated`, `column:deleted`
- `presence:join`, `presence:leave`, `presence:update`
- `typing:start`, `typing:stop`

Project Structure

```
in.work/
├── ts/
│   ├── index.ts              # Entry point
│   ├── cli.ts                # CLI commands
│   ├── inwork.ts             # Main application class
│   ├── plugins.ts            # Dependency imports
│   ├── interfaces/           # TypeScript interfaces
│   ├── models/               # SmartData entities
│   ├── services/             # Business logic
│   ├── api/                  # REST API router
│   └── sync/                 # External sync engine
├── ui/                       # Angular 19 frontend
├── test/                     # Test files
├── .nogit/                   # Local config (gitignored)
├── deno.json                 # Deno configuration
└── readme.md
```

Configuration

Environment Variables

Variable	Description	Default
`MONGO_URL`	MongoDB connection URL	`mongodb://localhost:27017`

Variable	Description	Default
`MONGO_DB`	Database name	`inwork`
`JWT_SECRET`	Secret for JWT signing	Auto-generated (dev)
`PORT`	Server port	`3000`
`SMTP_HOST`	SMTP server for emails	-
`SMTP_USER`	SMTP username	-
`SMTP_PASS`	SMTP password	-

External Sync

Gitea Setup

1. Create OAuth application in Gitea
2. Add connection via API:

```
curl -X POST http://localhost:3000/api/v1/connections \
  -H "Authorization: Bearer <token>" \
  -H "Content-Type: application/json" \
  -d '{
    "name": "My Gitea",
    "systemType": "gitea",
    "config": {
      "baseUrl": "https://gitea.example.com",
      "token": "<gitea-token>"
    },
    "syncSettings": {
      "syncDirection": "bidirectional",
      "autoSync": true,
      "syncIntervalMinutes": 15
    }
  }'
```

3. Link a board to a repository and sync will begin automatically

License

MIT

Revision #3
Created 2026-03-28 11:09:38 UTC by foss.global Team
Updated 2026-03-28 12:16:13 UTC by foss.global Team