# 

# **Smart Todo**

### AI-Powered Context-Aware Task Management System

Smart Todo is an intelligent task management platform that leverages AI to auto-create tasks based on contextual data such as messages, notes, or emails. It also supports manual task creation, real-time context entry tracking, and dynamic task categorization based on urgency and content.

## 

## Table of Contents

1. [Overview](#overview)
2. [Task APIs](#X766782e246d91d454ec3fb1e3970c16bb6366fd)
   * [GET /tasks/](#X057303cd603c81e453d014784b8f5da16ffbb42)
   * [POST /tasks/create/](#X6ecb6a9e2bac8ac034182e14cd484255be025ec)
   * [PATCH /tasks//](#X081813e7f346f2c5d934b07d32a053ef6a21b5e)
3. [Category API](#X693599987573139d7faf03362356aa9aca17d64)
   * [GET /categories/](#X8fceb358cf60f89b5dfd3fad09cff17c2cacd81)
4. [Context APIs](#X791e83616b0ddb95f4306fb706f32ad3254f65e)
   * [GET /context/](#X7731ba65e955a817e0e5c871de97b3c95452995)
   * [POST /context/add/](#Xcb817dcf27f15d39826d47856126651f2649c44)
5. [AI Task Auto-Creation API](#X4164088ed51cf06c6b5b6e33fd2291d6be2f8e6)
   * [POST /tasks/auto-create/](#X10617199dea6d52a9de0aa513206bae0862de51)
6. [Notes](#notes)

## API Documentation

### Overview

This document provides detailed API documentation for the Smart Todo List application, including endpoints for tasks, categories, context, and AI-powered task creation.

All APIs return data in **JSON format** and are prefixed with:

/api/

### 1. Task APIs

#### 1.1 GET /tasks/

**Description:** Retrieve a list of all tasks, ordered by priority and deadline.

**Response Format:**

[  
 {  
 "id": 1,  
 "title": "Frontend Task",  
 "description": "Fix layout issues",  
 "category": "Frontend",  
 "priority": "High",  
 "deadline": "2025-07-10"  
 },  
 ...  
]

#### 1.2 POST /tasks/create/

**Description:** Manually create a new task.

**Request Body:**

{  
 "title": "Backend Task",  
 "description": "Set up API endpoints",  
 "category": "Backend",  
 "priority": "Medium",  
 "deadline": "2025-07-11"  
}

**Response:** 201 Created

{  
 "id": 2,  
 "title": "Backend Task",  
 "description": "Set up API endpoints",  
 "category": "Backend",  
 "priority": "Medium",  
 "deadline": "2025-07-11"  
}

#### 1.3 PATCH /tasks/<id>/

**Description:** Retrieve or update an existing task using its ID.

**Example Request:**

{  
 "priority": "Low"  
}

**Response:** 200 OK

### 2. Category API

#### 2.1 GET /categories/

**Description:** Retrieve a list of all available task categories.

**Response Format:**

[  
 { "id": 1, "name": "Frontend" },  
 { "id": 2, "name": "Backend" }  
]

### 3. Context APIs

#### 3.1 GET /context/

**Description:** Retrieve all saved context entries, ordered by timestamp.

**Response Format:**

[  
 { "id": 1, "content": "Fix UI layout", "timestamp": "2025-07-05T13:40:00Z" }  
]

#### 3.2 POST /context/add/

**Description:** Create a new context entry.

**Request Body:**

{  
 "content": "Fix UI layout"  
}

**Response:** 201 Created

{  
 "id": 1,  
 "content": "Fix UI layout",  
 "timestamp": "2025-07-05T13:40:00Z"  
}

### 4. AI Task Auto-Creation API

#### 4.1 POST /tasks/auto-create/

**Description:** Automatically generate a task based on the provided context using AI.

**Request Body:**

{  
 "context": [  
 { "content": "Check the image size." }  
 ]  
}

**AI-Generated Task Response:** 201 Created

{  
 "id": 3,  
 "title": "Frontend Task",  
 "description": "Auto-created task (Context: Check the image size.)",  
 "category": "Frontend",  
 "priority": "Medium",  
 "deadline": "2025-07-09"  
}

### 

### Notes

* All endpoints return validation errors with 400 Bad Request if the input is invalid.
* AI-generated tasks are identified by the description prefix: **“Auto-created task”**.
* CORS headers must be configured properly when calling APIs from a frontend.