Trello-like Task Management Documentation

* Function-based API Task.
* This Task management will handle all users' tasks on one Main Task Board and will be able to add/assign their team members to and create a Project task environment
* The Trello Task Management allows users to manage boards, tasks, and collaborate with team members.

Roles and Permissions

1. Regular User

**Role:** Basic user who can create and manage their boards and tasks.

**Permissions:**

* **Board Management**
  + Create, update, and delete their boards
  + Set board visibility (public/private)
  + View boards, they are members of
* **Task Management**
  + Create, update, and delete their tasks
  + Update task status (pending/doing/completed)
  + Assign tasks to board members
* **Profile Management**
  + Register and update their profile
  + Change password
  + View their profile information

**Restrictions:**

* Cannot modify other users' boards or tasks unless added as a member
* Cannot access admin features

2. Board Owner (Creator)

**Role:** User who created a board.

**Permissions:**

* All regular user permissions
* **Board Membership**
  + Add/remove members to/from their boards
  + Change board settings and visibility
* **Task Oversight**
  + View all tasks on their boards
  + Manage task assignments

**Restrictions:**

* Cannot modify boards they didn't create
* Cannot access admin features

3. Super Admin

**Role:** System administrator with full control.

**Permissions:**

* **User Management**
  + View all users
  + Delete user accounts
  + View all boards
  + Modify any board
  + Delete any board

Function List

Authentication & User Management

1. register\_user() - User registration
2. login\_user() - User authentication (JWT tokens)
3. forgot\_password\_sent\_email() - Initiate password reset
4. reset\_password() - Complete password reset
5. update\_profile() - Update user profile
6. delete\_profile() - Delete user account
7. view\_my\_profile() - View current user's profile
8. view\_all\_users() - Admin view of all users

Board Management

1. create\_board() - Create new board
2. update\_board() - Update board details
3. delete\_board() - Delete board
4. get\_my\_board() - View board with all tasks
5. search\_boards() - Search/filter boards
6. add\_member\_to\_board() - Add user to board
7. remove\_member\_from\_board() - Remove user from board
8. view\_board\_members() - List board members
9. clear\_board() - Remove all tasks from board

Task Management

1. create\_task() - Create new task card
2. update\_task() - Update task details
3. delete\_task() - Delete task
4. task\_get() - View task details
5. search\_tasks() - Search/filter tasks
6. mark\_task\_complete() - Update task status

Task Media Management

1. add\_media\_files() - Add images/attachments to task
2. delete\_task\_mediafile() - Remove specific media file
3. replace\_task\_mediafiles() - Replace all media files

Models Documentation

User

Extends Django's AbstractUser with additional fields:

* user\_id - Auto-increment primary key
* username - Unique username
* email - User email
* password - Hashed password
* full\_name - Optional full name
* profile\_image - Optional profile picture
* is\_admin, is\_superadmin - Permission flags
* Timestamps: created\_at, updated\_at

ForgotPasswordOTP

Stores OTPs for password reset:

* user - ForeignKey to User
* otp - 6-digit OTP code
* created\_at - Timestamp
* is\_used - Flag to prevent reuse

Board

Represents a task board:

* board\_id - Auto-increment primary key
* title - Board title
* description - Optional description
* visibility - "private" or "public"
* created\_by - Creator (User FK)
* members - Many-to-many relationship with Users
* Timestamps and updater information

TaskCard

Represents a task card:

* task\_id - Auto-increment primary key
* board - ForeignKey to Board
* Title, description, due date
* is\_completed - Status ("pending", "doing", "completed")
* Creator, updater, assignee information
* Timestamps

TaskImage

Stores images attached to tasks:

* task\_image\_id - Auto-increment primary key
* task\_card - ForeignKey to TaskCard
* task\_image - Image file
* Uploader and timestamp information

TaskAttachment

Stores file attachments for tasks:

* Similar structure to TaskImage but for generic files

API Endpoint Reference

Authentication

**Register User**

* **Endpoint:** POST /register\_user/
* **Description:** Creates a new user account
* **Request Body:**
* {
* "username": "string",
* "email": "string",
* "password": "string",
* "full\_name": "string"
* }

**Login User**

* **Endpoint:** POST /login\_user/
* **Description:** Authenticates user and returns JWT tokens
* **Request Body:**
* {
* "username": "string",
* "password": "string"
* }

**Forgot Password**

* **Endpoint:** POST /forgot\_password\_sent\_email/
* **Description:** Initiates password reset process
* **Request Body:**
* {
* "email": "string"
* }

Board Operations

**Create Board**

* **Endpoint:** POST /create\_board/
* **Description:** Creates a new board
* **Authentication:** Required
* **Request Body:**
* {
* "title": "string",
* "description": "string (optional)",
* "visibility": "private|public"
* }

**Update Board**

* **Endpoint:** PUT /update\_board/
* **Description:** Updates board details
* **Authentication:** Required (board owner)
* **Request Body:**
* {
* "board\_id": "int",
* "title": "string (optional)",
* "description": "string (optional)",
* "visibility": "private|public (optional)"
* }

Task Operations

**Create Task**

* **Endpoint:** POST /create\_task/
* **Description:** Creates a new task card
* **Authentication:** Required (board member)
* **Request Body:**
* {
* "board\_id": "int",
* "title": "string",
* "description": "string (optional)",
* "due\_date": "date (optional)",
* "assigned\_to": "int (user\_id, optional)"
* }

**Update Task Status**

* **Endpoint:** POST /mark\_task\_complete/
* **Description:** Updates task status
* **Authentication:** Required (board member)
* **Request Body:**
* {
* "task\_id": "int",
* "status": "pending|doing|completed"
* }