TASKIFY

Users App: Authentication, Registration, and Role-Based Access

1. User Model (`SignupUser`)

- **Fields:**
  - `first_name`, `last_name`, `username`, `email`, `password`
  - `role`: Choices are `employee`, `teamlead`, `project_manager`, `manager`, `admin`
  - `status`: Boolean, whether the user is active/enabled
  - `is_verified`: Boolean, whether the user has completed email/OTP verification
- **Role System:**
  - Each user has a `role` field, which is used throughout the app for access control and permissions.
  - Roles are used in session (`request.session['user_role']`) to control access to views and features.

---

2. User Verification (`UserVerification`)

- **Purpose:** Handles OTP/verification code for new registrations and password resets.
- **Fields:**
  - `user` (OneToOne to SignupUser)
  - `verification_code` (6-digit OTP)
  - `created_at`, `attempts`
- **Methods:**

- `generate_verification_code()`: Generates and saves a new OTP.
- `is_code_expired()`: Checks if OTP is expired (2 minutes).
- `is_code_valid(entered_code)`: Validates entered OTP, tracks attempts.
- `is_max_attempts_reached()`: Max 3 attempts allowed.

---

3. Authentication & Registration Views

Registration (`Register`)

- Handles new user registration.
- Checks for unique email and username.
- Passwords are hashed.
- Default role is `employee`.
- Sends OTP to email for verification.
- Sets `is_verified = False` until OTP is confirmed.

OTP Verification (`VerifyOTP`)

- Handles both registration and password reset OTP flows.
- Validates OTP, tracks attempts, and expires after 2 minutes.
- On success, sets `is_verified = True` and activates the user.

Admin User Registration (`Register_UserbyAdmin`)

- Allows admins/managers to create users of any role.
- Optionally assigns a department.
- Checks for unique email/username.
- Sets role as selected by admin.

Login (`Login`)

- Authenticates user by username and password.
- Checks if user is active (`status`).
- On success, sets session variables: `user_id`, `username`, `user_role`.
- Redirects based on role:
  - `admin/manager` → admin dashboard
  - others → employee dashboard

Logout (`Logout`)

- Clears the session and redirects to login.

Password Reset (`ForgetPassword`, `ConfirmNewPassword`)

- Sends OTP to email for password reset.
- On OTP verification, allows user to set a new password.

---

4. Role-Based Access Control

- **Session-Based:** User role is stored in session (`request.session['user_role']`).
- **View Restrictions:** Many views check for role before allowing access (e.g., only `admin/manager` can register users, promote users, or toggle status).
- **Promotion/Demotion (`PromoteUser`):**
  - Only `admin/manager` can promote/demote users.
  - Cannot demote an admin.
  - Updates the `role` field.
- **Enable/Disable Users (`ToggleUserStatus`):**
  - Only `admin/manager` can enable/disable users.

- Cannot disable an admin.
- Updates the `status` field.

---

5. User Endpoints (`users/urls.py`)

URL Pattern	View Function	Purpose
`register/`	Register	User registration
`verify-otp/`	VerifyOTP	OTP verification
`register-user-by-admin/`	Register_UserbyAdmin	Admin/manager user creation
`promote-user/<user_id>/<new_role>/`	PromoteUser	Promote/demote user (admin/manager)
`toggle-status/<user_id>/`	ToggleUserStatus	Enable/disable user (admin/manager)
`forgot-password/`	ForgetPassword	Start password reset (send OTP)
`confirm-newpass/`	ConfirmNewPassword	Set new password after OTP
`logout/`	Logout	Log out
(root, i.e. `''`)	Login	User login

---

6. Security & Best Practices

- **Password Hashing:** Uses Django's `make_password` and `check_password` for secure password storage and verification.
- **Session Management:** All authentication and role checks are session-based.
- **Email Verification:** All new users and password resets require OTP verification via email.
- **Account Status:** Users can be enabled/disabled by admin/manager.
- **Role Hierarchy:** Only privileged roles can manage other users' roles/status.

---

7. Integration with Other Apps

- **Department Assignment:** Users can be assigned to departments (via `UserDepartment` in the dashboard app).
- **Role-Based Dashboard:** After login, users are redirected to the appropriate dashboard based on their role.
- **Notifications, Tasks, Projects:** User model is referenced throughout the dashboard app for task assignment, notifications, and more.

Summary: The users app provides a robust authentication and user management system with role-based access control, OTP-based email verification, password reset, and admin/manager user management features. All sensitive operations are protected by session and role checks,
and the system is designed to integrate tightly with the dashboard and project/task manage ment features.

Dashboard App

1. Department

- **Purpose:** Represents a company department (e.g., HR, IT, Marketing).

- **Fields:**
  - `name`: Unique name of the department.
- **Usage:** Used to group users and projects.

---

2. UserDepartment

- **Purpose:** Links a user to a department.
- **Fields:**
  - `user`: ForeignKey to `SignupUser` (from users app).
  - `department`: ForeignKey to `Department`.
- **Usage:** Used for assigning users to departments, which is important for task/project assignment and access control.

---

3. Projects

- **Purpose:** Represents a project within the company.
- **Fields:**
  - `name`: Unique project name.
  - `description`: Project description.
  - `department`: ForeignKey to `Department` (which department owns the project).
  - `start_date`, `end_date`: Project timeline.
  - `status`: Project status (`pending`, `ongoing`, `completed`, `on_hold`, `cancelled`).
- **Usage:** Projects are containers for tasks and are department-specific.

---

4. Tasks

- **Purpose:** Represents a task assigned to a user within a project.
- **Fields:**
  - `project`: ForeignKey to `Projects`.
  - `assigned_to`: ForeignKey to `UserDepartment` (who is responsible).
  - `assigned_from`: ForeignKey to `SignupUser` (who assigned the task).
  - `task_name`, `task_description`: Task details.
  - `due_date`: When the task is due.
  - `expected_time`: How long the task is expected to take (as a duration).
  - `priority`: Task priority (`low`, `medium`, `high`).
  - `status`: Task status (`pending`, `in_progress`, `completed`, `on_hold`, `cancelled`, `not_assigned`).
  - `task_file`: FileField for file uploads related to the task.
  - `report`: Text field for a report or summary.
  - `submitted_on`: When the task was marked as completed.
- **Methods:**
  - `get_total_time_spent()`: Returns total time spent on the task (from activity logs).
  - `get_expected_time_display()`: Returns expected time as a human-readable string.
- **Usage:** Central to the system—tracks all work, assignments, and progress.

---

5. TaskActivityLog

- **Purpose:** Logs work sessions for tasks (start/stop times, duration).
- **Fields:**
  - `task`: ForeignKey to `Tasks`.

- `start_time`, `end_time`: When the session started/ended.
- `duration`: Duration of the session.

- **Methods:**
  - `calculate_duration()`: Calculates and stores the duration.
  - `start_work(cls, task)`: Ends any ongoing sessions and starts a new one.
  - `stop_work(cls, task)`: Stops the current session, adjusting for office hours if needed.
- **Usage:** Used for time tracking, statistics, and enforcing office hours.

---

6. OfficeHours

- **Purpose:** Defines working hours and breaks for each day of the week.
- **Fields:**
  - `day`: Day of the week.
  - `start_time`, `end_time`: Office start/end times.
  - `break_start_time`, `break_end_time`: Optional break times.
  - `is_working_day`: Whether this day is a working day.
  - `is_active`: Whether this office hour entry is active.
- **Methods:**
  - `is_within_office_hours(cls, datetime_obj)`: Checks if a datetime is within office hours.
- **Usage:** Used to enforce work schedules and for time tracking.

---

7. companyDetails

- **Purpose:** Stores company-wide information.

- **Fields:**
  - `company_logo`: Image/logo of the company.
  - `company_name`, `company_description`, `company_phone`, `company_email`, `company_website`, `company_address`, `company_founded_date`: Company info.
  - `company_social_media_links`: JSON field for social media links.
- **Methods:**
  - `get_instance()`: Ensures only one instance exists (singleton pattern).
- **Usage:** Used for displaying company info on dashboards and profiles.

---

8. employeeProfile

- **Purpose:** Stores additional information for each employee.
- **Fields:**
  - `user`: ForeignKey to `SignupUser`.
  - `profile_picture`: Employee's profile image.
  - `bio`, `contact_number`, `address`, `date_of_birth`: Personal info.
- **Usage:** Used for employee profile pages and personalization.

---

9. ChatMessage

- **Purpose:** Stores chat messages/comments for tasks.
- **Fields:**
  - `task`: ForeignKey to `Tasks`.
  - `user`: ForeignKey to `SignupUser` (who sent the message).
  - `message`: The message text.
  - `timestamp`: When the message was sent.

- **Usage:** Enables real-time chat and comments on tasks.

---

10. TaskReadStatus

- **Purpose:** Tracks when a user last read messages for a task (for unread indicators).
- **Fields:**
  - `user`: ForeignKey to `SignupUser`.
  - `task`: ForeignKey to `Tasks`.
  - `last_read_at`: Timestamp of last read.
- **Meta:**
  - `unique_together = ('user', 'task')`: Only one entry per user/task.
- **Usage:** Used to show unread message indicators in the UI.

---

11. Notification

- **Purpose:** Stores notifications for users (task updates, assignments, approvals, chat, etc.).
- **Fields:**
  - `user`: ForeignKey to `SignupUser`.
  - `message`: Notification text.
  - `timestamp`: When the notification was created.
  - `is_read`: Whether the notification has been read.
- **Methods:**
  - `get_relevant_notifications(cls, user)`: Returns unread notifications and recent read notifications (last 24 hours).
- **Usage:** Drives the notification system, badges, and real-time updates.

Summary Table

Model	Purpose/What it Stores
Department	Department name
UserDepartment	Links user to department
Projects	Project details, department, status, dates
Tasks	Task details, assignment, status, priority, file/report, expected time, time spent
TaskActivityLog	Work session logs for tasks (start/stop/duration)
OfficeHours	Working hours and breaks for each day
companyDetails	Company-wide info (logo, name, description, contact, social links)
employeeProfile	Employee personal info (profile picture, bio, contact, address, DOB)
ChatMessage	Task chat/comments (task, user, message, timestamp)
TaskReadStatus	Last read time for user/task (for unread indicators)
Notification	User notifications (message, timestamp, read/unread)

Dashboard App: Functionalities & Views by User Role

A. Admin/Manager Panel

Purpose: Admins and managers have full control over the system:
they manage departments, projects, employees, office hours, company details,
and can view company-wide statistics and notifications.

1. Department Management

- **AddDepartments**: Create new departments.
- **DeleteDepartment**: Remove a department.
- **AssignDepartment**: Assign users to departments.

2. Project Management

- **AddProjects**: Create new projects, assign to departments.
- **AllProjects**: View, update, and manage all projects.
- **UpdateProjectStatus**: Change the status of a project (pending, ongoing, completed, etc.).

3. Employee/User Management

- **Register_UserbyAdmin** (from users app): Add new users (any role),
  assign to departments.
- **ToggleUserStatus** (from users app): Enable/disable user accounts.
- **PromoteUser** (from users app): Change user roles (e.g., promote to teamlead).

4. Task Management

- **AssignedTasks**: View all tasks for ongoing projects.
- **DeleteTask**: Delete/reject tasks (with notification to the assigned user).

- **approve_task**: Approve tasks requested by employees.
- **GetTasks**: Get all tasks for a specific project (AJAX).

5. Office Hours & Settings

- **dashboard_settings**: View and edit office hours for each day.
- **add_office_hours / edit_office_hours / delete_office_hours**: CRUD for office hours.
- **start_tracking**: Start the office hours tracking system.

6. Company Profile

- **about_company**: View and update company details, logo, and social links.

7. Statistics & Analytics

- **admindashboard_stats**: View company-wide statistics, employee efficiency, hours worked, and task completion rates.

8. Management Commands

- **check_office_hours.py**: Run periodic checks on office hours (CLI).

Admin/Manager Panel: Typical Workflows

- Add/manage departments and assign users.
- Create/manage projects and assign to departments.
- Add new users, promote/demote, enable/disable accounts.
- Approve/reject employee task requests.
- Set and enforce office hours.
- View company statistics and employee performance.
- Manage company profile and branding.
- Receive and manage notifications for all system events.

Employee/Teamlead/Project Manager Panel

Purpose: Employees, teamleads, and project managers focus on their assigned work: viewing and managing their tasks, submitting reports/files, chatting, tracking time, and viewing their own statistics and notifications.

1. Dashboard & Task Management

- **new_employee_dashboard**: Main dashboard showing assigned tasks, projects, statistics, and notifications.
- **CreateTask**: Employees can request new tasks for themselves; teamleads/project managers can assign tasks to others.
- **UpdateTaskStatus**: Update the status of a task (e.g., mark as completed).
- **hold_task / start_working / stop_working**: Change task status to on hold, in progress, or stopped.
- **pollPendingTasks / updateOngoingTasks** (AJAX): Fetch and update task lists dynamically.

2. Task Approval (Teamlead/Project Manager)

- **approval_pending_tasks_json**: View and approve/reject tasks requested by employees in their department.
- **DeleteTaskTeamlead**: Reject/delete tasks as a teamlead/project manager.

3. File & Report Uploads

- **upload_task_file / upload_task_report**: Upload files and reports for tasks.
- **download_task_file**: Download task files.
- **get_task_file / get_task_report**: Fetch file/report info (AJAX).

4. Chat & Comments

- **get_task_comments**: Fetch chat messages for a task.

- **WebSocket consumers**: Real-time chat and unread indicators.

5. Notifications

- **allNotificaations**: View all notifications.
- **employee_notifications_json**: Fetch notifications for dropdowns.
- **mark_notifications_read / readAllNotifications**: Mark notifications as read.

6. Statistics & Analytics

- **employee_statistics**: View personal statistics (tasks done, time spent, on-time/late, performance score, charts).
- **employee_today_stats_json**: Fetch today's stats (AJAX).

7. Profile & Company Info

- **employee_profile**: View and update personal profile (picture, bio, contact, etc.).
- **employee_company_profile**: View company profile.

Employee/Teamlead/Project Manager Panel: Typical Workflows

- View and manage assigned tasks.
- Request new tasks or submit for approval.
- Start/stop/hold work on tasks and track time.
- Upload/download task files and submit reports.
- Chat in real-time with team members on tasks.
- View and mark notifications as read.
- View personal statistics and performance.
- Update personal profile and view company info.

Summary Table: Functionality by Role

Functionality	Admin/Manager Panel	Employee/Teamlead/PM Panel
Department Management	Yes	No
Project Management	Yes	View only (unless PM/lead)
Task Assignment/Approval	Yes	Request/submit, approve (lead/PM)
Task Status Update	Yes	Yes
File/Report Upload/Download	Yes	Yes
Office Hours Management	Yes	No
Time Tracking	View all	Track own
Statistics	Company/employee-wide	Personal
Notifications	Own	Own
Chat/Comments	All tasks	Own/assigned tasks
Company Profile Management	Yes	View only
Employee Profile Management	No	Yes (own)
User Management	Yes	No

How the Views Work Together

- **Admins/Managers**:
  - Set up the organization (departments, projects, office hours, company info).
  - Manage users and oversee all tasks and statistics.
  - Approve/reject tasks, monitor notifications, and ensure smooth operation.
- **Employees/Teamleads/Project Managers**:
  - Focus on their assigned work.
  - Request/submit tasks, upload files/reports, chat, and track their own performance.
  - Teamleads/PMs can approve/reject tasks for their department.

Security, Access Control, and Business Logic

---

1. Authentication & Session Management

- **All access to dashboard views requires authentication.**
  - Users must log in via the users app.
  - Upon login, the following are stored in the session:
    - `user_id`
    - `username`
    - `user_role` (e.g., employee, teamlead, project_manager, manager, admin)

    Session-based access:
  - Every sensitive view checks for the presence of `user_id` and the correct `user_role` in the session.
  - If not present or not authorized, the user is redirected to login or shown an error.

---

2. Role-Based Access Control (RBAC)

- **Roles:**
  - `employee`, `teamlead`, `project_manager`, `manager`, `admin`
- **Enforcement:**
  - Views check `request.session['user_role']` before allowing access or performing actions.
  - Example:
    Only `admin`/`manager` can access department/project/office hours management.
  - Teamleads/project managers can approve/reject tasks for their department,
    but not for others.
  - Employees can only see and act on their own tasks and profile.

What's possible:

- Admins/managers can manage everything.
- Teamleads/project managers can manage/approve tasks in their department.
- Employees can only interact with their own data.

What's not possible:

- Employees cannot access admin/manager views or data of other users.
- Teamleads/project managers cannot manage users or departments outside their scope.
- No user can escalate their own privileges via the UI.

---

3. Data Integrity & Ownership

- **Task/Project/Department Ownership:**
  - All queries for tasks, projects,
    and departments are filtered by the user's department or assignment.

- Employees cannot view or modify tasks not assigned to them.
- Teamleads/project managers can only approve/reject tasks in their department.

- **File/Report Uploads:**
  - Only the assigned user can upload files/reports for a task.
  - File uploads are stored in user-specific directories and linked to the correct task.
- **Chat/Comments:**
  - Only users assigned to a task (or with department access) can view or post comments.
  - Chat messages are linked to both the task and the user.

---

4. Notifications & Real-Time Features

- **Notification Delivery:**
  - Notifications are only created for relevant users (e.g., task assignee, teamlead, admin).
  - Real-time notifications use Django Channels and are sent only to the correct user's WebSocket group.
- **Unread Indicators:**
  - Unread status is tracked per user and per notification/message.
  - Only the intended recipient sees unread indicators.

---

5. Office Hours & Time Tracking Enforcement

- **Office Hours:**
  - Only admin/manager can define or edit office hours.
  - Time tracking for tasks is enforced based on office hours.
  - If a user tries to log time outside office hours, the system adjusts the end time to the last valid office hour.

- **Task Activity Logs:**
  - Each work session is logged with start/end time and duration.
  - Users cannot manually edit their time logs.

---

6. Business Logic Enforcement

- **Task Status Transitions:**
  - Employees can only move their tasks through allowed statuses (e.g., from pending to in progress, then to completed).
  - Teamleads/project managers/admins can approve/reject tasks, but cannot bypass required steps.
  - Once a task is completed, it cannot be edited except by an admin.
- **Task Assignment:**
  - Only authorized roles can assign tasks to others.
  - Employees can only request tasks for themselves.
- **Approvals:**
  - Task requests by employees must be approved by a teamlead/project manager/admin before work can begin.
- **Notifications:**
  - All major actions (assignment, approval, completion, rejection) trigger notifications to the relevant users.

---

7. File & Data Security

- **File Uploads:**
  - Only authenticated users can upload files.
  - Files are stored in protected directories and linked to the correct task/user.

- Download links are only available to authorized users.

- **Sensitive Data:**
  - No sensitive user data is exposed in templates or APIs.
  - All AJAX endpoints check session and role before returning data.

---

8. CSRF & Form Security

- **CSRF Protection:**
  - All forms and AJAX requests include CSRF tokens.
  - Django's built-in CSRF middleware is enabled.
- **Form Validation:**
  - All forms validate input on the server side.
  - Unique constraints (e.g., usernames, emails, department names) are enforced at the model and form level.

---

9. Auditability & Logging

- **Activity Logging:**
  - All task activity (start/stop work) is logged with timestamps.
  - Management commands and critical actions log to the server log for audit purposes.
- **Error Handling:**
  - All exceptions are caught and logged.
  - Users receive user-friendly error messages; sensitive details are not exposed.

---

10. What's Prevented by Design

- **Privilege Escalation:**
  - Users cannot change their own role or access data outside their scope.
- **Data Tampering:**
  - All updates are validated and checked for ownership/permissions.
- **Unauthorized File Access:**
  - Files are only accessible to users assigned to the relevant task.
- **Cross-Department Access:**
  - Teamleads/project managers cannot manage or view data for other departments.
- **Session Hijacking:**
  - Session-based access, logout flushes session, and CSRF protection is enforced.

---

11. Possible Security Enhancements

- **Rate Limiting:**
  - To prevent brute-force attacks on login or OTP endpoints.
- **Two-Factor Authentication:**
  - Could be added for admin/manager roles.
- **Audit Trail:**
  - More detailed logging of all admin actions for compliance.

---

12. Summary Table: Security & Logic

Area	What's Possible (Allowed)	What's Not Possible (Prevented)

Login/Session	Authenticated access, session-based	Unauthenticated access, session hijacking
Role Management	Admin/manager can promote/demote	Users cannot self-promote
Department Access	Only own department (unless admin)	Cross-department access
Task Management	Assign, approve, update own tasks	Edit others' tasks, bypass approval
File Upload/Access	Only for assigned tasks	Download/upload for others' tasks
Notifications	Only for relevant users	See others' notifications
Chat/Comments	Only for assigned/department tasks	See/post in others' tasks
Office Hours	Only admin/manager can edit	Employees cannot change office hours
Time Tracking	Only for own tasks, enforced by office	Falsify time logs, edit others' logs
CSRF/Form Security	All forms protected	CSRF attacks, form tampering
Error Handling	User-friendly errors, logs for admin	Sensitive info exposure

---

In summary: The dashboard app is designed with strong session-based authentication, strict role-based access control, and business logic that enforces data integrity and prevents privilege escalation or unauthorized access. All sensitive actions are protected by both backend checks and frontend UI restrictions, and the system is extensible for further security enhancements.
