Employee Attendance Tracker Documentation & Backend API

A simple backend system for managing employee data, document uploads, attendance tracking, and email notifications using Node.js, Express, Sequelize, and MySQL.

Prerequisites

1. **Install MySQL** and create a database:

   ```
   CREATE DATABASE employee_tracker;
   ```

2. **Install dependencies**:

   ```
   npm install
   ```

3. **Start the project**:

   ```
   npm start
   ```

4. On **first run**, Sequelize will sync and auto-create tables. After that:

Comment out below mentioned line in `index.js` to avoid re-altering tables every time:

```
// await db.sequelize.sync({ alter: true });
```

Database Models Overview

- **Admin**: Stores token version for JWT validation.
- **Employee**: Basic profile, photo, hobbies, and status.
- **Document**: Uploaded files linked to employees.
- **Attendance**: Daily attendance entries per employee.

Authentication Strategy

- Admin login is based on **hardcoded credentials**:
  - Email: `admin@test.com`
  - Password: `admin123`
- No user table or bcrypt password hashing is used, because:
  - The task specifies hardcoded admin login.

    - No role-based auth or multiple user management is required.

For session validation, a `token_version` is stored in the `tbl_admin` table.
On each login, a new token version is generated, and added to the JWT.
When verifying a token, it's compared to the DB version if mismatched, token is invalidated.

Hobby Field Design

- `hobby` is currently stored as a `JSON` or comma-separated string in the Employee table.
- In production:
    - A separate **hobby master** and **employee_hobby mapping** table should be used for scalability and querying.
    - For this task, it's simplified as per the limited scope.

Email Notification

- SMTP-based email (via **Nodemailer**) is triggered **when attendance is marked**.
- Email includes:
    - Employee Name
    - Status (present, absent, half_day)
    - Date
- Email content is styled in HTML.
- SMTP credentials should be set via `.env`:

```
SMTP_HOST=smtp.example.com
SMTP_USER=your@email.com
SMTP_PASS=yourpassword
```

Dashboard Summary

- API endpoint returns a summary for **today**:
    - Total employees
    - Present count
    - Absent count
    - Half-day count

Folder Structure (Recommended)

```
├── controllers/
│   └── controller.js
├── models/
│   └── index.js, employee.js, attendance.js, etc.
│
```

```
├── routes/
│   └── employeeRoutes.js
├── validation/
│   └── validation.js
├── utils/
│   └── emailFun.js
├── uploads/
│   └── document
├── .env
├── index.js
├── package.json
```

Not Implemented (by design)

Feature	Reason
User login/authentication	Hardcoded admin login was specified
Token table	Token version is tracked in admin table instead
Bcrypt password encryption	Not required due to hardcoded login
Hobby normalization (master table)	Avoided due to task scope (can be added later if admin creates hobbies)
Email logging in DB	Optional – can be added via `email_log` model

Final Notes

- This project is **task-specific** for backend, not production-ready.
- Can be extended to multi-admin, role-based auth, and scalable relational design.
- You've handled file uploads, validations, JWT, and mailing excellent foundation.
