# Virtual logbook and Course hours tracking

## Introduction

The Virtual Logbook and Course Hours Tracking web application is designed to help users keep track of different topics taught during courses and the number of hours spent on these courses. The application is structured to support both regular users and class delegates/prefects. Here’s a detailed overview of the system's functionalities, architecture, and instructions on how to set up and run the application.

**Key Features**

* **User Roles**: The application supports two types of users:
  + **Regular Users**: These users belong to a particular class and can view logbook entries for their courses.
  + **Class Delegates/Prefects**: These users also belong to a particular class but have additional responsibilities, including filling out the logbook and tracking course hours.
* **Logbook Entries**: Class delegates/prefects can create logbook entries that include:
  + **Subject**: The course for which the entry is being made.
  + **Date**: The specific day the entry pertains to.
  + **Period**: The start and end times of the teaching session.
  + **Teacher**: The teacher who conducted the session.
  + **Text**: Detailed notes or descriptions of the session.
  + **Attachments**: One or more files (images, audio, etc.) related to the session.
* **Automatic Hour Logging**: The application automatically logs the number of hours a teacher has taught a particular course whenever a logbook entry is created.
* **Course Views**: Regular users can view a list of courses and see all logbook entries associated with a selected course.

## Technologies Used

- Django: Used as the backend framework for building the web application.

- React: Employed for developing the frontend user interface to enhance interactivity.

- PostgreSQL: Chosen as the database management system for storing data reliably and efficiently.

- React Query: Utilized for managing server-state and handling data fetching in the frontend.

- HTML/CSS: Used for structuring the user interface and styling components.

- Git/GitHub: Employed for version control and collaborative development.

## Features

- User Authentication: Allows users to register, login, and manage their accounts securely.

- Logbook Entry Creation: Enables teachers to create logbook entries with details such as start time, end time, course, and text.

- Course Management: Provides administrators with the ability to add, edit, and delete courses.

- Automatic Hour Tracking: Automatically calculates teaching hours based on logbook entries and updates course hour totals.

- File Upload: Allows users to upload files and associate them with logbook entries for documentation purposes.

## System Architecture

The system follows a client-server architecture, with the backend implemented using Django and the frontend developed using React. Data is stored in a PostgreSQL database, with Django ORM handling database interactions. The frontend communicates with the backend via RESTful APIs, facilitating seamless data exchange.

## Implementation Details

The implementation involved building Django models for users, courses, logbook entries, and file uploads. React components were created for the user interface, with React Query used for managing state and data fetching. The system's logic for calculating teaching hours was implemented in the Django backend, ensuring accuracy and consistency.

## User Interface

The user interface features an intuitive design with clear navigation and interactive elements. The delegate can easily log the teaching hours of teachers, while administrators have access to tools for managing courses and monitoring student activities. The interface is responsive, allowing users to access the system from various devices.

## 8. Future Enhancements

- \*\*Reporting Module:\*\* Implement a reporting module for generating detailed reports on teaching hours by course, teacher, etc.

- \*\*Notification System:\*\* Add a notification system to alert users of important events, such as upcoming deadlines or changes to logbook entries.

- \*\*Integration with Calendar:\*\* Integrate the system with calendar applications to synchronize teaching schedules and logbook entries.

## 9. Conclusion

The "Virtual Logbook and Course Hour Tracking" project has successfully addressed the need for an efficient and reliable system for logging teaching hours and tracking course hours. By leveraging modern technologies and best practices, the system provides a user-friendly platform that enhances productivity and accuracy in educational settings.

## 10. References

- Django Documentation: [https://docs.djangoproject.com/](https://docs.djangoproject.com/)

- React Documentation: [https://reactjs.org/docs/getting-started.html](https://reactjs.org/docs/getting-started.html)

- PostgreSQL Documentation: [https://www.postgresql.org/docs/](https://www.postgresql.org/docs/)

---

Feel free to customize the content as needed and add more details based on your project's specific requirements and achievements. If you have any questions or need further assistance, feel free to ask!

Certainly! Let's add a section on how to run both the React app and the Django app.

---

## 11. Running the Application

### 11.1. Running the Django Backend

1. \*\*Setup Virtual Environment:\*\*

- Navigate to the root directory of the Django project.

- Create a virtual environment using `virtualenv` or `venv`.

- Activate the virtual environment.

2. \*\*Install Dependencies:\*\*

```bash

pip install -r requirements.txt

```

3. \*\*Database Setup:\*\*

- Configure PostgreSQL database settings in `settings.py`.

- Run migrations to create database schema:

```bash

python manage.py makemigrations

python manage.py migrate

```

4. \*\*Run Development Server:\*\*

```bash

python manage.py runserver

```

The Django development server should now be running locally at `http://localhost:8000`.

### 11.2. Running the React Frontend

1. \*\*Navigate to Frontend Directory:\*\*

```bash

cd frontend

```

2. \*\*Install Dependencies:\*\*

```bash

npm install

```

3. \*\*Start Development Server:\*\*

```bash

npm start

```

The React development server should now be running locally at `http://localhost:3000`.

### 11.3. Accessing the Application

Once both the Django backend and React frontend servers are running, you can access the application by opening your web browser and navigating to `http://localhost:3000`. This will load the React frontend, which communicates with the Django backend API running at `http://localhost:8000`.

### 11.4. Production Deployment

For production deployment, you'll need to configure the Django backend and React frontend for deployment on a production server. This typically involves setting up a production database, configuring static files serving, and deploying the React app to a static file server or using a service like AWS Amplify, Netlify, or Vercel.

---

Feel free to adjust the instructions based on your specific deployment requirements and environment. If you need further assistance or have any questions, feel free to ask!