Study Tracker — Smart Learning Companion

Project Distinctiveness

This Study Tracker web application stands out for its practical value, user-focused design, and integration of multiple advanced Django features to create a seamless experience for students who want to take control of their academic goals.

What makes this project distinctive is how it moves beyond a basic CRUD (Create, Read, Update, Delete) system to provide:

Key Distinctive Features

1. Full User Authentication System

- Built using Django's secure authentication system.

- Includes registration, login, logout, and session expiration on browser close.

- Styled, modern login/register forms with clear instructions and password validation feedback.

2. Custom Goal and Log Tracking

- Users can create specific study goals, set target hours, deadlines, and even a reminder time.

- A study log system allows users to record daily study progress, linking directly to goals.

3. Visual Feedback via Graphs

- Interactive charts display progress toward each goal, offering a clear view of performance.

- Uses AJAX to fetch dynamic data and update charts without page reloads.

4. Automated Email Reminders

- Sends email reminders to the user if the current time matches the reminder time set for any goal.

- Utilizes Django's SMTP email backend and time-based trigger logic.

5. Dark/Light Theme Toggle

- Provides users the option to switch between light and dark modes.

- The site dynamically adapts its background, form, and text colors based on the theme.

6. Styled UI with Instructional Feedback

- Modern, responsive design using custom CSS.

- Features include:

- Instruction boxes

- Help texts for fields (e.g., password tips)

- Centralized, clean form layouts

Technical Distinctiveness

- Utilizes Django models, forms, views, template inheritance, and static file handling.

- Integrates AJAX for charts, dynamic email logic, and contextual rendering (e.g., user-based filtering).

- Implements CSRF protection, session handling, and custom redirect logic for improved security and UX.

Why This Project Matters

Unlike generic task managers or to-do lists, this Study Tracker is tailored for students, helping them:

- Define meaningful study goals

- Stay accountable with progress logs

- Visualize efforts via charts

- Receive reminders at the right time

CONTAINED IN EACH FILE

Tracker

This is the main app that runs the

Study tracker features.

models.py

Defines two main data types:

Goal: A study target with a title,

description, and deadline.

StudyLog: A record of time spent studying,

linked to a specific goal.

forms.py

Builds the forms users see when they add a goal or log study hours.

These forms are linked to the models above.

views.py

Contains the app’s logic. It decides what data to show on each page, and

what to do when a user submits a form.

urls.py

Maps the app’s URLs like dashboard or add_goal to the correct view

functions.

admin.py

Registers the models with Django’s admin panel, so the developer can manage data from a secure backend interface.

Templates

Contains the actual web pages shown to the user. Uses Django's templating language.

base.html

A layout that every other page is built on. It holds shared elements like the navigation bar.

dashboard.html

The user's homepage after logging in. Shows current goals and study logs, and displays a chart of study activity.

add_goal.html

A form that allows users to create new goals (e.g., “Revise Biology for 3 hours this week”).

add_log.html

A form that lets users track how much time they studied and what they covered

registration/login.html & register.html

Login and sign-up pages that manage user accounts securely.

STATIC

Holds visual and interactive files used by

the app.

styles.css

Styles all the pages — controls the layout, colors, and fonts.

chart.js

Draws interactive charts (like study time graphs) on the dashboard using

data from the server.

theme.js

Adds a light/dark mode toggle so users can personalize the website's

appearance.

Images

(dark-bg.jpg, light-bg.jpg, etc.)

Background images

DESIGN APPROACH

We used a user-centered design approach. Studytracker, was aimed at helping students manage study time and stay organized.Here's how we applied the UCD approach:

Identified user needs:

We talked to a few classmates and found common issues like forgetting deadlines, poor time tracking, and lack of study motivation.

Planned features based on feedback:

We chose to include:

- Study session logging
- Subject-based progress tracking
- Daily reminders for tasks

Created early designs:

We made designs showing key screens (home dashboard, log study, calendar view).

User testing:

Eight classmates tried the prototype and gave feedback. Based on this, we:

- Made the interface simpler
- Added color coding for different subjects
- Reduced clicks to log a session
- Added light and dark modes

Why we chose UCD:

It kept our design focused on real user problems, gave us a clear process to follow, and improved the usefulness of the final system.

Mobile responsiveness

1. **Added a Media Query**: Introduced a media query to target screens with a width of 768px or less, ensuring styles adapt specifically to smaller devices like tablets and mobile phones.

2. **Button Adjustments:** Reduced button padding and font size to make buttons more compact and proportional on smaller screens, improving usability and layout.

3. **Logout Button Font Size:** Adjusted the font size of the logout button to ensure it remains readable and easy to tap on smaller screens, enhancing usability.

HOW TO RUN STUDY TRACKER

1. Clone the Repository

Clone the repository to your local machine:

```
git clone <https://github.com/KelvinAbidha/IAP_Assignment>
cd <Studytracker >
```

2. Create a Virtual Environment

```
python -m venv venv
venv\Scripts\activate
```

3. Install Dependencies

```
pip install django
```

4. Apply Database Migrations

Run migrations to set up the database schema:

```
python manage.py makemigrations
python manage.py migrate
```

5. Run the Development Server

Start the Django development server:

```
python manage.py runserver
```

Open your browser and go to http://127.0.0.1:8000 to view the application.