PERSONALISED STUDY PLANNER

By: Rumaysa Abdul Bari
Date: 14-08-2025

Table of Content

Introduction

Effective time management is one of the most significant challenges faced by students during exam preparation. With multiple subjects to study, overlapping deadlines, and limited days before exams, students often struggle to allocate their study time efficiently. This can lead to last-minute cramming, unbalanced preparation, and increased stress levels, which ultimately affect performance.

The Personalized Study Planner is designed to solve this problem by providing an automated, organized, and easy-to-use solution. It is a simple yet powerful web application that enables students to create a well-structured study plan in just a few minutes. By entering the number of subjects, their names, and respective exam dates, the application calculates the number of days left for each subject and smartly distributes the total available study time. This ensures that students can focus proportionately on each subject according to the urgency and time remaining.

By combining automation with personalization, the Personalized Study Planner not only helps students stay on track but also reduces decision fatigue. This approach ensures better organization, balanced preparation, and reduced exam stress, ultimately contributing to improved academic performance. In the long run, tools like this can promote better study habits and time management skills that benefit students beyond just exam periods.

Requirements

1. Physical Requirements

- Processor: Intel Core i3 or above / AMD equivalent
- RAM: Minimum 4 GB (8 GB recommended)
- Storage: At least 500 MB free space for Python, libraries, and project files
- Display: Minimum 1280×720 resolution
- Internet Connection: Required for installing dependencies and running Streamlit

2. Software Requirements

- Operating System: Windows 10/11, macOS, or Linux
- Programming Language: Python 3.8 or higher
- Required Python Libraries:
  - streamlit – For building the interactive web app
  - datetime – For working with dates and deadlines
  - pandas – For managing tabular data (optional for export features)

Construction

Step 1: Install Python (3.8+) from python.org and check:

python --version

pip --version

Step 2: Create a project folder named **study-planner** and open it in your code editor.

Step 3: Inside the folder, create:

- **app.py** (main application file)
- **requirements.txt** (library list)

Step 4: Create and activate a virtual environment:

```
python -m venv venv

venv\Scripts\activate     # Windows

source venv/bin/activate  # macOS/Linux
```

Step 5: Add required libraries to requirements.txt:

```
streamlit

pandas
```

Step 6: Install dependencies:

pip install -r requirements.txt

Step 7: Write the app code in **app.py** (title, subject inputs, date inputs, study plan generation).

Step 8: Add improvements

- Export plan to CSV
- Customize daily study hours
- Color-coded deadlines

Step 9: Run the app locally:

streamlit run app.py

How to run

1. The user inputs the number of subjects.
2. The app dynamically asks for each subject's name and exam date.
3. It calculates the number of days remaining until each exam.
4. Based on total available time, it allocates recommended daily study hours per subject.
5. The plan is displayed in a simple and readable format.

```
C:\WINDOWS\system32\cmd.

Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rumay>cd "C:\Users\rumay\OneDrive\Desktop\study planner"

C:\Users\rumay\OneDrive\Desktop\study planner>python -m venv venv

C:\Users\rumay\OneDrive\Desktop\study planner>venv\Scripts\activate

(venv) C:\Users\rumay\OneDrive\Desktop\study planner>pip install -r requirements.txt
Collecting streamlit (from -r requirements.txt (line 1))
  Using cached streamlit-1.48.1-py3-none-any.whl.metadata (9.5 kB)
Collecting pandas (from -r requirements.txt (line 2))
  Using cached pandas-2.3.1-cp313-cp313-win_amd64.whl.metadata (19 kB)
Collecting numpy (from -r requirements.txt (line 3))
  Using cached numpy-2.3.2-cp313-cp313-win_amd64.whl.metadata (60 kB)
Collecting matplotlib (from -r requirements.txt (line 4))
  Using cached matplotlib-3.10.5-cp313-cp313-win_amd64.whl.metadata (11 kB)
Collecting altair!=5.4.0,!=5.4.1,<6,>=4.0 (from streamlit->-r requirements.txt (line 1))
  Using cached altair-5.5.0-py3-none-any.whl.metadata (11 kB)
Collecting blinker<2,>=1.5.0 (from streamlit->-r requirements.txt (line 1))
  Using cached blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)
Collecting cachetools<7,>=4.0 (from streamlit->-r requirements.txt (line 1))
  Using cached cachetools-6.1.0-py3-none-any.whl.metadata (5.4 kB)
Collecting click<9,>=7.0 (from streamlit->-r requirements.txt (line 1))
  Using cached click-8.2.1-py3-none-any.whl.metadata (2.5 kB)
Collecting packaging<26,>=20 (from streamlit->-r requirements.txt (line 1))
  Using cached packaging-25.0-py3-none-any.whl.metadata (3.3 kB)
Collecting pillow<12,>=7.1.0 (from streamlit->-r requirements.txt (line 1))
  Using cached pillow-11.3.0-cp313-cp313-win_amd64.whl.metadata (9.2 kB)
Collecting protobuf<7,>=3.20 (from streamlit->-r requirements.txt (line 1))
  Using cached protobuf-6.31.1-cp310-abi3-win_amd64.whl.metadata (593 bytes)
Collecting pyarrow>=7.0 (from streamlit->-r requirements.txt (line 1))
  Using cached pyarrow-21.0.0-cp313-cp313-win_amd64.whl.metadata (3.4 kB)
Collecting requests<3,>=2.27 (from streamlit->-r requirements.txt (line 1))
  Using cached requests-2.32.4-py3-none-any.whl.metadata (4.9 kB)
Collecting tenacity<10,>=8.1.0 (from streamlit->-r requirements.txt (line 1))
  Using cached tenacity-9.1.2-py3-none-any.whl.metadata (1.2 kB)
Collecting toml<2,>=0.10.1 (from streamlit->-r requirements.txt (line 1))
  Using cached toml-0.10.2-py2.py3-none-any.whl.metadata (7.1 kB)
Collecting typing-extensions<5,>=4.4.0 (from streamlit->-r requirements.txt (line 1))
```

```
(venv) C:\Users\rumay\OneDrive\Desktop\study planner>streamlit run app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.1.7:8501
```

localhost:8501

Deploy

Personalized Study Planner

Enter number of subjects:

3

Enter name of subject 1

ds

Enter exam date for ds

2025/08/22

Enter name of subject 2

math

Enter exam date for math

2025/08/31

Enter name of subject 3

eng

Enter exam date for eng

2025/08/27

Generate Study Plan

27°C Haze | Search | ENG IN | 16:11 14-08-2025

localhost:8501

Deploy

Enter name of subject 2

math

Enter exam date for math

2025/08/31

Enter name of subject 3

eng

Enter exam date for eng

2025/08/27

Generate Study Plan

Your Day-by-Day Study Plan

	Date	Subject
0	2025-08-14	ds
1	2025-08-15	ds
2	2025-08-16	ds
3	2025-08-17	ds
4	2025-08-18	ds
5	2025-08-19	ds
6	2025-08-20	ds
7	2025-08-21	ds
8	2025-08-14	eng
9	2025-08-15	eng

27°C Haze | Search | ENG IN | 16:10 14-08-2025

Generate Study Plan

Syntax Code

```
import streamlit as st
import datetime
import pandas as pd

st.title("📚 Personalized Study Planner")

# Store data in session
if "subjects" not in st.session_state:
  st.session_state.subjects = []

# Step 1: Number of subjects
num_subjects = st.number_input("Enter number of subjects:", min_value=
step=1)

# Step 2: Collect subject names and dates
for i in range(num_subjects):
  if len(st.session_state.subjects) <= i:
    st.session_state.subjects.append({"name": "", "date": None})

        sub_name = st.text_input(f"Enter name of subject {i+1}
value=st.session_state.subjects[i]["name"], key=f"name_{i}")
  exam_date = st.date_input(
    f"Enter exam date for {sub_name or 'Subject ' + str(i+1)}",
    value=st.session_state.subjects[i]["date"] or datetime.date.today(),
    key=f"date_{i}"
```

```
)

  st.session_state.subjects[i]["name"] = sub_name
  st.session_state.subjects[i]["date"] = exam_date

# Step 3: Generate timetable
if st.button("Generate Study Plan"):
  today = datetime.date.today()
  plan = []

  # Create a list of days for each subject until exam
  for sub in sorted(st.session_state.subjects, key=lambda x: x["date"]):
    days_left = (sub["date"] - today).days
    for d in range(days_left):
      date = today + datetime.timedelta(days=d)
      plan.append({"Date": date.strftime("%Y-%m-%d"), "Subject":
sub["name"]})

  # Distribute subjects evenly
  df = pd.DataFrame(plan)
  st.subheader("📅 Your Day-by-Day Study Plan")
  st.dataframe(df)
```

Conclusion & Futurescope

The Personalized Study Planner is an effective and practical solution for students seeking to organize their exam preparation efficiently. By allowing users to input their subjects and exam dates, it automatically calculates the time available for each subject and distributes study hours in a balanced manner. This removes the need for tedious manual planning, reduces stress, and ensures that no subject is neglected. Its interactive interface, built using Python and Streamlit, makes it accessible to all students regardless of technical background, while its adaptability allows it to fit different academic schedules.

Looking ahead, the planner can be enhanced with advanced features such as exporting schedules to PDF or Excel, automated daily reminders via email or notifications, Google Calendar synchronization, and AI-powered prioritization of subjects based on difficulty or performance trends. Incorporating progress tracking and motivational insights could further encourage consistency in study habits. With these enhancements, the Personalized Study Planner has the potential to evolve from a simple planning tool into a comprehensive academic companion, helping students worldwide achieve better time management and improved exam results.

Thank you