Study Planner Application – Implementation Guide

This document explains **what was implemented** and **which file is responsible for which requirement**.

---

1. Project Structure Overview

```
src/
│
├── components/
│   ├── Navbar.jsx        # Top navigation (Planner | Progress | Dashboard)
│   ├── SessionForm.jsx   # Create new study sessions
│   ├── SessionRow.jsx    # Inline edit + complete session row
│
├── pages/
│   ├── Planner.jsx       # Study plan management page
│   ├── Progress.jsx      # Progress table + charts page
│   ├── Dashboard.jsx     # Visual analytics overview
│
├── context/
│   └── StudyContext.jsx  # Global state management (React Context)
│
├── App.js                # Routing configuration
├── index.js              # App bootstrap + Context provider
└── index.css             # Global styles
```

---

2. Requirement-wise Implementation

Requirement 1: Study Plan Management

Create, View, Edit study sessions

- **SessionForm.jsx**
- Form to create study sessions
- Fields: subject, topic, date, hours
- Uses Context method `addSession`
- **SessionRow.jsx**

- Displays sessions in a table
- Inline editing using Material-UI TextField
- Save/Edit functionality using icons
- **Planner.jsx**
- Combines SessionForm and session list table

Requirement 2: Progress Tracking

Mark sessions as completed & track hours per subject

- **StudyContext.jsx**
- Maintains `completed` flag per session
- `toggleComplete()` updates completion state
- **Progress.jsx**
- Aggregates hours per subject
- Displays total hours in a table

Requirement 3: Visualization

Pie Chart & Line Chart

- **Progress.jsx**
- Pie Chart: Time spent per subject
- Line Chart: Study hours over time
- Uses `recharts` with `ResponsiveContainer`
- **Dashboard.jsx**
- High-level visual overview (can include KPIs + charts)

Requirement 4: Navigation

Planner | Progress | Dashboard routing

- **App.js**
- Uses React Router
- Defines routes for all pages
- **Navbar.jsx**
- Material-UI AppBar
- Navigation links

Requirement 5: Data Handling

React Context with hardcoded values

- **StudyContext.jsx**
- Uses `createContext` and `useState`
- Hardcoded initial sessions
- Centralized state (single source of truth)
- **index.js**
- Wraps App with `StudyProvider`

Requirement 6: User Experience (Responsive UI)

Material-UI based responsive design

- **Material-UI components** used across app
- Container, Box, Table, AppBar, Buttons
- **Responsive behavior**
- `Container` adapts to screen size
- `Box` with flexWrap for mobile layouts
- Charts wrapped in `ResponsiveContainer`

---

3. Key Design Decisions

- React Context chosen over Redux (lighter & sufficient)
- Material-UI chosen for professional & responsive UI
- Inline editing for better UX
- Charts + tables combined for clarity

---

4. How to Explain This Project (Interview Tip)

> "This is a React-based Study Planner app using Context API for state management and Material-UI for responsive design. It allows users to plan study sessions, track progress by subject, and visualize progress using charts. All data is centrally managed via Context with hardcoded initial values."

---

5. Conclusion

This application fulfills **all functional and UI requirements** mentioned in the problem statement and follows **production-ready React practices**.

End of documentation.