**Two-Week (14-Day) Plan Recap**

**Week 1**

**Day 1: Project Setup ✅**

1. Initialize the repo (in your case, recreate the folder structure and Git).
2. Add a basic file structure (e.g., main.py, ui/, logic/, data/).
3. Commit your initial structure to Git (push if you want).

**Day 2: Input Topics Functionality**

1. Implement a function to **add a new topic/session** (e.g., “Binary Trees”).
2. Store these sessions in data.json.
3. Verify you can see the sessions after saving.

**Day 3: Basic GUI with Tkinter**

1. Create a **Tkinter window** (Tk()) in, say, ui/main\_window.py.
2. Add:
   * A text field/entry box for entering a new topic.
   * A button to add that topic (which calls your Day 2 function).
   * A list or text area to display all existing sessions.

**Day 4: Calculate Review Dates (Spaced Repetition Logic)**

1. In logic/scheduler.py, implement a function to set or update next\_review for each session based on your chosen intervals (e.g., Day 1, Day 3, Day 7...).
2. Whenever a user adds a topic, set its initial next\_review.

**Week 2**

**Day 5: Display Upcoming Reviews**

1. Enhance the UI to show sessions due **today** or within the next few days.
2. Use your spaced-repetition logic to filter or highlight those sessions.

**Day 6: Mark Topics as Reviewed**

1. Add a button in your UI to “Mark as Reviewed.”
2. Update the session’s next\_review based on the next spaced-rep interval.

**Day 7: Persistent Saving & Loading**

1. Ensure the app automatically **loads** from data.json on startup.
2. Make sure changes are **saved** to data.json whenever you add or review a session.

**Day 8: Polish the GUI**

1. Improve the layout (labels, frames, alignment).
2. Possibly set a custom window size or add styling so it’s user-friendly.

**Days 9–10: Testing & Debugging**

1. Run through edge cases (duplicate topics, blank input, etc.).
2. Fix any bugs in the logic or UI flow.

**Days 11–12: Documentation**

1. Update your README.md with:
   * Project overview, how to install/run.
   * Basic usage instructions (screenshots if you have them).
2. Add code comments/docstrings where needed.

**Days 13–14: Final Touches**

1. Refactor your code to maintain readability (split large functions, rename variables).
2. Optional final features (like a Pomodoro timer or notifications) if you have time.
3. Create a final commit or tag a release (e.g., v1.0)