Key Technical Decisions:

1. **UIKit with Storyboards**
   - Chose Storyboards for faster UI prototyping and visual layout management.
   - Maintained programmatic elements where needed (e.g., dynamic table cells).
2. **UserDefaults for Storage**
   - Selected for simplicity and zero setup—ideal for small-scale task data.
   - Used `Codable` to serialize/deserialize `Task` objects seamlessly.
3. **MVC Architecture**
   - Kept Storyboard-friendly while separating:
     - **Model**: `Task` struct + `TaskManager` (data handling).
     - **View**: Storyboard scenes + programmatic cell tweaks.
     - **Controller**: Mediates logic (e.g., swipe-to-delete animations).
4. **Hybrid Approach**
   - Mixed Storyboard segues (`Show`/`Present Modally`) with programmatic navigation where dynamic behavior was needed (e.g., filter transitions).

Challenges Faced:

1. **Storyboard Limitations**
   - Dynamic cell animations (e.g., delete effects) required programmatic overrides in `TaskTableViewCell`.
   - **Fix**: Used `UITableView`'s `commit editingStyle` with `UIView.animate`.
2. **UserDefaults Scalability**
   - Large task lists slowed down read/write operations.
   - **Mitigation**: Cached tasks in memory and

updated `UserDefaults` only on changes.

3 **State Sync Across Screens**
   - Changes in `AddEditTaskViewController` didn't immediately reflect in the main list.
   - **Fix**: Used unwind segues + `viewWillAppear` to reload data.

4 **Date Picker in Storyboard**
   - Toggling visibility caused layout issues.
   - **Fix**: Constrained picker to a container view and animated `isHidden` with `layoutIfNeeded()`.