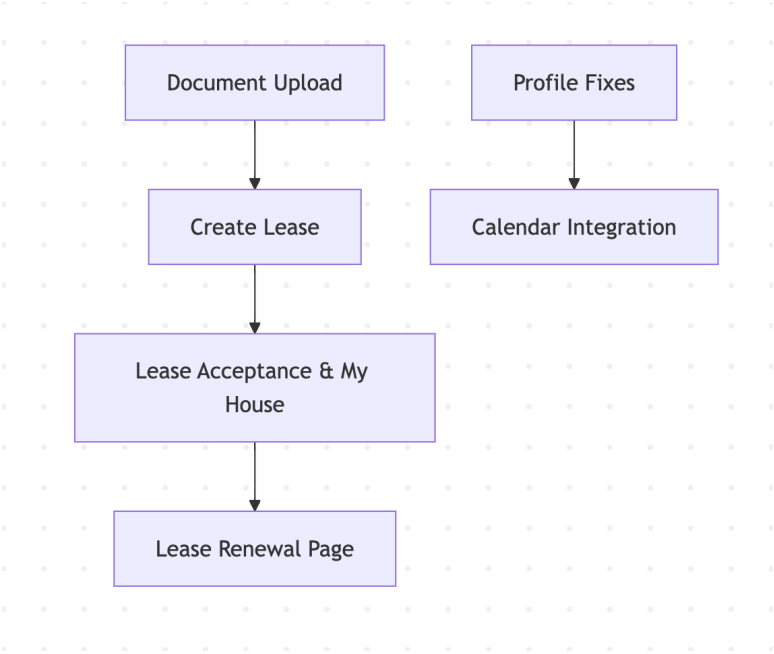
Schedule & Dependencies Analysis

Network Diagram



- **Calendar Integration** runs in parallel, but depends partially on **Profile Management**
- **Bug Fixes** (profile picture, account settings, payment time) support stability across all modules

Dependencies

- **Document Upload** is foundational for lease creation, especially for landlords preparing agreements.
- **Create Lease** must be implemented and accepted before **Lease Renewal** becomes relevant.
- **Lease Acceptance** updates the tenant dashboard ("My House") and requires backend agreement logic.
- **Calendar Feature** depends on profile data and user context for event tracking.
- **Profile Fixes** ensure users have valid states for accessing scheduling, documents, and account flows.

Critical Path

1. Document Upload
2. Create Lease
3. Lease Acceptance
4. Lease Renewal Page

Any delay in these features causes downstream features to be blocked or unusable.

Risk Areas

- **Late Execution**: Many tasks were completed near the end of the sprint, compressing QA and integration.
- **Scope Creep**: Bug fixes and small features were added mid-sprint, complicating original planning.
- **Frontend/Backend Sync**: Some delays occurred in aligning API responses (e.g., date formatting).
- **Calendar Component**: While standalone, its integration required clean profile data and UI space.
- **Jira Tracking**: Delayed updates made burndown tracking less useful in real-time.

What Went Wrong & Lessons Learned

What Went Wrong

- **Delayed Integration**: Feature implementation and PR reviews were clustered at the end of the sprint causing a disruption in the burndown.
- **Scope Drift**: Several "small" bug fixes (e.g., account settings, time formats) weren't originally planned but were pulled in.
- **Late Completion**: Testing time was limited due to late completion of tasks, risking potential regressions.

Lessons Learned

- **Enforce Mid-Sprint Review**: Hold team-wide checkpoints to align on integration and review progress.
- **Lock Scope After Planning**: Freeze scope unless critical. Add extras to the next sprint.
- **Live Jira Updates**: Team members should log progress daily to keep sprint tracking meaningful.
- **Parallelization**: Where possible, decouple features to allow parallel development and reduce bottlenecks.

- **Test Continuously**: Run integration checks early, even if features are unfinished, to avoid end-sprint pressure.