# Schedule Diagram – Sprint 3

Group Name: Chroma Notes

Group Members:

Giuseppe D’Orazio

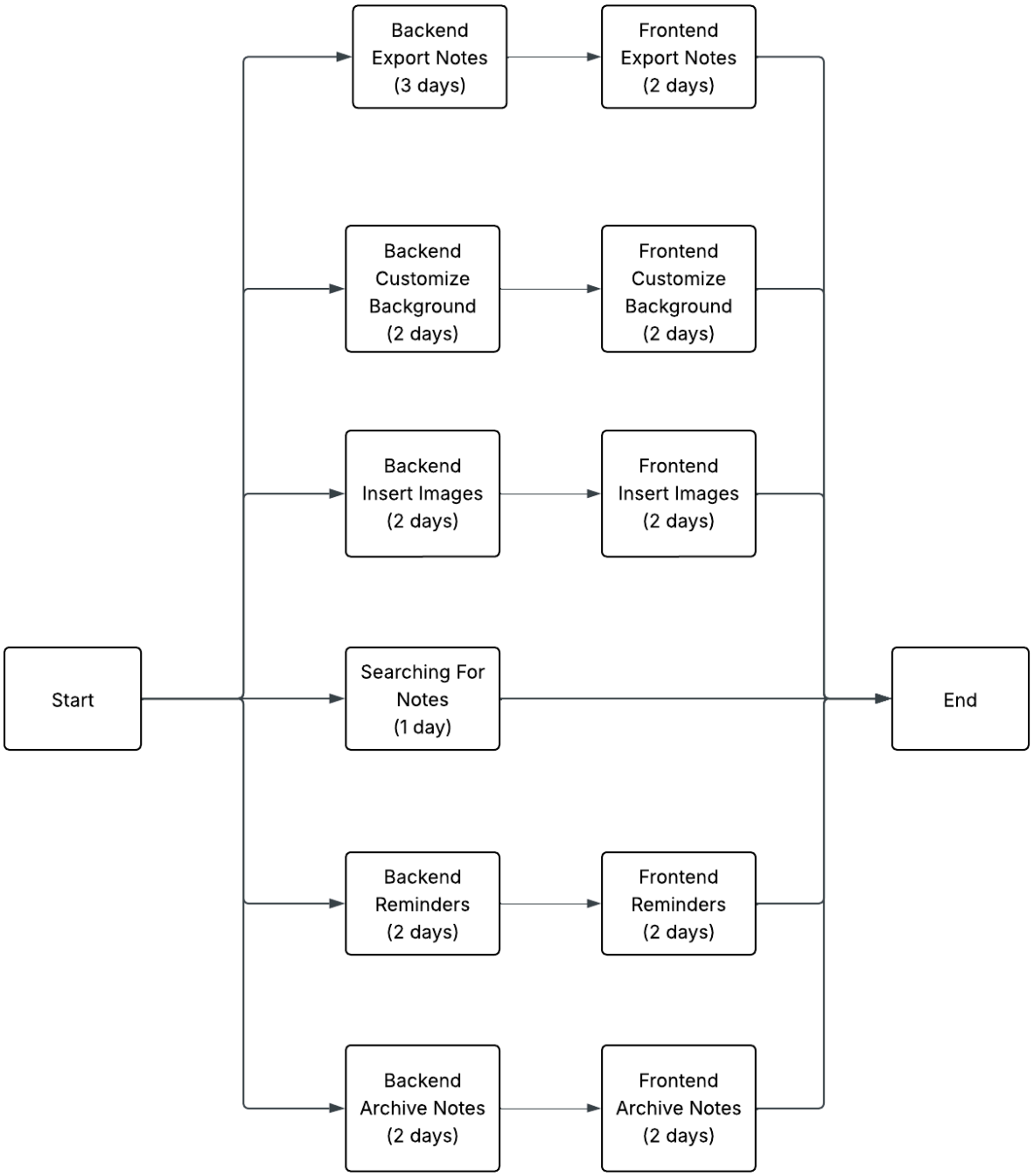
Ravneet Deol

Victoria Bolognese

Araz Manouchian

Date: November 30, 2025

Network Diagram:



Task Dependences:

The network diagram illustrates all task dependencies for Sprint 3. Some tasks could be done in parallel, but several required the backend to begin before the frontend could.

Backend → Frontend Dependences

1. Export Notes
    - Backend Export Notes → Frontend Export notes
    - 3 days backend, following by 2 days frontend
2. Customize Background
    - Backend Customize Background → Frontend Customize Background
    - 2 days backend, followed by 2 days frontend
3. Insert Images
    - Backend Insert Images → Frontend Insert Images
    - 2 days backend, followed by 2 days frontend
4. Reminders
    - Backend Reminders → Frontend Reminders
    - 2 days backend, followed by 2 days frontend
5. Archive Notes
    - Backend Archive Notes → Frontend Archive Notes
    - 2 days backend, followed by 2 days frontend

Independent Task

1. Searching For Notes
    - This task was independent of all others
    - Duration: 1 day

Critical Path Analysis:

The critical path represents the longest sequence of dependent tasks that determines the minimum time needed to complete the sprint's work.

The longest dependency chain is:

1. Backend Export Notes (3 days)
2. Frontend Export Notes (2 days)

Total duration of this chain: 5 days

This is longer than all other chains:

- Customize Background: 2 days + 2 days = 4
- Insert Images: 2 days + 2 days = 4
- Reminders = 2 days + 2 days = 4
- Archive Notes = 2 days + 2 days = 4
- Searching For Notes: 1 day, independent

Conclusion:
The Export Notes user story forms the critical path.
Any delay in either Backend or Frontend would delay the sprint, because it is the longest chain of dependent tasks, but the other user stories are not dependent on this one, so it wouldn't be a huge delay.

Ensuring the Sprint Stayed on Schedule:

To keep the sprint on schedule, our team followed several coordination and planning strategies

- **Backend-First Prioritization**
  - Backend tasks were prioritized whenever possible, providing stable, completed endpoints for the frontend team to build upon efficiently.
- **Adaptive Workflow During Backend Delays**
  - When backend work was delayed, the frontend team proactively prepared UI components and layouts. Once the backend tasks were completed, the frontend seamlessly integrated the APIs. This approach kept the sprint on schedule and ensured both teams remained productive.
- **Workload Balancing**
  - Tasks were distributed fairly among team members to prevent overloading any individual and to allow parallel progress on backend tasks.
- **Task Independence**
  - Independent tasks, such as searching for notes, were scheduled during lighter workload periods or while awaiting dependencies. This optimized productivity without blocking other tasks.
- **Consistent Team Communication**
  - Open communication was maintained throughout the sprint. Team members reported early completions or blockers promptly, allowing tasks to be reprioritized as needed to keep the sprint on track.