PLANORAMA

Sprint 3 Planning Document

Team 22: Sruthi Bhamidipati, Yana Sheoran, Vihaan Khajanchi, Jack Kenny

Sprint Overview

This sprint introduces a comprehensive set of new features focused on intelligent planning, collaborative task management, and personal productivity tracking. A major advancement is the smart subtask suggestion system, which offers actionable subtasks for common task types using a curated phrase bank, and applies basic NLP techniques (like sentence segmentation) as a fallback when no direct match is found. The system provides users with a prompt to review, accept, reject, or modify these suggestions, making it easier to break down broad goals into manageable actions from the start.

Time tracking and progress evaluation also take center stage this sprint. Users can now enter estimated times for tasks and compare them to actual time logs, with visual indicators showing whether tasks were completed under, over, or on time. This supports better planning and time awareness. Two new analytical features—weekly summaries and a productivity tracker—help users reflect on their most and least productive days, as well as spot long-term trends across multiple weeks. To encourage sustainable work habits, a break reminder system has also been introduced to notify users after extended periods of uninterrupted focus.

This sprint also delivers a full rollout of team collaboration features. Team members can now add, assign, claim, and complete tasks, with real-time visibility across the team. Additional functionality includes per-user display names, task commenting, and support for marking and viewing completed team tasks. In parallel, the sprint closes out several previously incomplete features from previous sprints: undo for task deletions, automatic archiving of completed tasks, note-taking with formatting, and default task templates. Together, these updates result in a more intelligent, collaborative, and organized experience for users managing both personal and shared workloads.

Scrum Master: Sruthi Bhamidipati

Meeting Plan: T/Th 3:30-4:00 pm

Risks and Challenges:

- Integration Complexity: Coordinating new condition-based features like subtask suggestions, break timers, and task rollovers across backend logic and frontend behavior may lead to edge cases and require careful planning to ensure smooth user interaction.
- NLP Limitations: The smart subtask system relies on simple phrase matching and sentence splitting, which may fail to provide useful suggestions for less structured, vague, or uncommon task descriptions.
- UI/UX Overload: Introducing multiple interactive elements—such as suggestion prompts, comments, reminders, and team controls—risks overwhelming users if not thoughtfully designed and prioritized within the interface.
- Session Consistency and Team Sync: Without real-time updates, team features like claiming, assigning, and commenting may appear inconsistent across users' views unless session data is refreshed or periodically synced.
- Feature Interdependencies: New features like completion tracking, archival, time summaries, and reminders often rely on shared task data, increasing the chance of bugs or regressions if interactions aren't thoroughly tested across all use cases.

Current Sprint Detail

User Story #1

As a user, I would like to receive smart suggestions for breaking up long tasks into smaller subtasks.

#	Description	Estimated Time	Owner
1	Create a suggestion bank that maps common task phrases (e.g., “write essay”, “plan trip”) to curated subtasks.	1.5 hrs	Sruthi
2	Implement keyword/fuzzy matching logic to detect when a task matches an entry in the suggestion bank (e.g., lowercasing, stopword filtering, partial token overlap).	2 hrs	Sruthi
3	Use lightweight NLP (e.g., with spaCy or nltk) to segment long task descriptions into sentence-like chunks or clauses as a fallback when no match is found	2.5 hrs	Sruthi
4	Design a frontend suggestion prompt that shows generated subtasks and allows users to accept, reject, or edit the list.	1.5 hrs	Sruthi
5	Implement editable subtasks in the UI and convert accepted suggestions into actual subtasks tied to the main task.	2 hrs	Sruthi
6	Write unit and integration tests for suggestion matching, fallback parsing, and frontend accept/edit/reject flow.	2 hrs	Sruthi

Acceptance Criteria:

- Given the user creates a task that matches a known phrase (e.g., “Write essay”), when the system processes the task, then a suggestion prompt should appear showing context-appropriate subtasks.
- Given the task doesn’t match the suggestion bank but has a long description, when the system processes it, then subtasks should be suggested by splitting the description into smaller steps or actions.
- Given the user sees the suggestion prompt, when they click “Accept”, then the subtasks should be added to the task and saved.
- Given the user sees the suggestion prompt, when they click “Edit”, then they should be able to modify or remove the subtasks before confirming them.
- Given the user sees the suggestion prompt, when they click “Reject”, then the subtasks should not be added and the prompt should be dismissed.
- Given the user revisits a task with accepted suggestions, when the task loads, then the subtasks should appear and behave like regular subtasks (editable, completable, etc.).

User Story #2

As a user, I would like to receive a weekly summary of my most and least productive days so that I can adjust my workflow.

#	Description	Estimated Time	Owner
1	Define productivity metric (e.g., number of completed tasks per day, optionally weighted by priority/status if needed).	1 hr	Sruthi

2	Implement backend logic to aggregate completed task data for the current calendar week (Sunday to Saturday).	2 hrs	Sruthi
3	Compute most and least productive days based on task completion count.	1 hr	Sruthi
4	Design frontend component to display a weekly breakdown (bar graph or list view with 7 days, highlight top/bottom).	2 hrs	Sruthi
5	Add a “Weekly Summary” page with the ability to view the current and previous week(s).	2 hrs	Sruthi
6	Add date filtering logic to view previous full weeks (e.g., “Week of Mar 24 – Mar 30”).	1.5 hrs	Sruthi
7	Write unit and integration tests for aggregation logic, date filtering, and summary rendering.	1.5 hrs	Sruthi

Acceptance Criteria:

- Given the user has completed tasks and/or logged time during a calendar week (Sunday to Saturday), when the summary loads, then it should display task count and time logged for each day of that week.
- Given both task counts and time logs are present, when the summary calculates productivity, then the most and least productive days should be highlighted based on a combined productivity score.
- Given a user has completed tasks but no time logs (or vice versa), when the summary loads, then it should still include available data for each metric.
- Given the user views the current week before Saturday, when the summary is displayed, then it should show partial data for the days completed thus far.
- Given the user clicks or hovers over a specific day in the summary, then they should see a detailed breakdown of tasks completed, time logged, and the calculated productivity score.
- Given the user selects a past calendar week from a dropdown or filter, when the summary loads, then the data should reflect that specific week’s activity.

User Story #3

As a user, I would like to receive notifications so that I can be reminded of important tasks and stay productive.

#	Description	Estimated Time	Owner
1	Define rules for triggering notifications (e.g., tasks due today, overdue, or marked high priority) and ensure logic runs at the beginning of each session.	1 hr	Sruthi
2	Build backend logic or middleware that compiles notification data on session load (or user login) based on task metadata.	2.5 hrs	Sruthi
3	Design a notification UI (e.g., bell icon or dropdown panel) with badge and expandable list of notifications.	1.5 hrs	Sruthi
4	Implement frontend logic to show notifications on session start unless the user has opted out in settings.	2 hrs	Sruthi

5	Add per-notification and “dismiss all” functionality that hides notifications for the current session only.	1 hr	Sruthi
6	Create settings option to enable/disable task notifications.	1 hr	Sruthi
7	Write tests for task filtering, session-based dismissal, and settings toggle logic.	2 hrs	Sruthi

Acceptance Criteria:

- Given the user starts a new session and notifications are enabled, when tasks are due today, overdue, or marked high priority, then the notification panel should appear with those tasks listed.
- Given the user dismisses a notification during a session, when they continue using the app, then the dismissed notification should not reappear until the next session if still valid.
- Given the user clicks “Dismiss All,” when the panel updates, then all notifications should be hidden for the remainder of the session.
- Given the user disables notifications in their settings, when they start a new session, then no task notifications should be shown.
- Given notifications are enabled and a task becomes due while the user is already in the session, when the app detects the change, then a new notification should appear dynamically.
- Given the user clicks on a task in the notification panel, when the task table is shown, then a temporary visual highlight (e.g., border) should appear around the corresponding task row, and the highlight should disappear when the user clicks anywhere else on the page.

User Story #4

As a user, I would like to set an estimated time and an actual time for each of my tasks, and see the difference between the two.

#	Description	Estimated Time	Owner
1	Add an "Estimated Time" field to the task model and task creation/edit UI.	2.5 Hrs	Yana
2	Use existing time log data to calculate total "Actual Time" per task.	1 Hr	Yana
3	Display estimated vs. actual time and the variance in task details view.	2 Hr	Yana
4	Color-code the time difference: green if under, red if over, gray if equal.	1.5 Hrs	Yana
5	Add optional filters/sorting: tasks over time, under time, or on time.	2 Hrs	Yana
6	Write tests for estimated time entry, difference calculation, and UI display.	1 Hr	Yana

Acceptance Criteria:

- Given the user is creating or editing a task, when they enter an estimated time, then the estimated time should be saved and displayed in the task details.
- Given the user has logged time on a task, when they view the task, then the actual time spent should be displayed alongside the estimated time.

- Given both estimated time and actual time are available for a task, when the task is viewed, then the difference between the two should be displayed.
- Given there is a time difference between the estimated and actual times, when the task is displayed, then the difference should be color-coded (green if under, red if over, gray if equal).
- Given the user is viewing their task list, when they use sorting or filtering options, then they should be able to view tasks sorted by time performance (e.g., tasks completed under time, over time, or on time).
- Given the user enters an invalid value (e.g., non-numeric characters) in the "Estimated Time" field, when they try to save the task, then an error message should appear indicating that only valid numbers are allowed.
- Given the user tries to input an estimated time less than zero (negative value), when they save the task, then an error message should appear, stating that estimated time cannot be negative.
- Given the user attempts to save a task with an empty estimated time field, when they try to save the task, then an error message should appear asking them to provide an estimated time.

User Story #5

As a user, I would like to compare my productivity across different weeks/months to track my long-term progress.

#	Description	Estimated Time	Owner
1	Modify the database schema to include total logged time per user for weekly tracking.	1 Hr	Yana
2	Implement logic to aggregate and compare logged time over weeks.	2 Hrs	Yana
3	Design a UI component for the Productivity Tracker section, displaying weekly comparisons.	2 Hrs	Yana
4	Show weekly trends in productivity, highlighting improvements or decreases in logged time	2 Hrs	Yana
5	Allow users to click and view previous weekly productivity trends in the Productivity Tracker section.	2 Hrs	Yana
6	Test the functionality, ensuring that logged time is tracked correctly for long-term progress.	1 Hr	Yana

Acceptance Criteria:

- Given the user logs time for tasks during a week, when the user clicks on the Productivity Tracker section, then the system should display the total time spent on tasks for the current week.
- Given the user compares two different weeks, when they view the Productivity Tracker section, then the system should display the difference in logged time between the two weeks in an easy-to-read format.
- Given the user clicks on the Productivity Tracker section, when they view their weekly productivity, then the UI should display the total logged time for each week in a visually clear format (e.g., bar chart or progress bar).
- Given the user views their weekly productivity trends, when there is an increase in logged time compared to the previous week, then the system should highlight this with a green indicator (e.g., “+10% this week”).
- Given the user views their weekly productivity trends, when there is a decrease in logged time compared to the previous week, then the system should highlight this with a red indicator (e.g., “-5% this week”).
- Given the user wants to view past weekly productivity trends, when they access the history in the Productivity Tracker section, then the system should display logged time for each previous week, showing trends over time.

- Given the user clicks on the Productivity Tracker section, when there are no tasks logged for the current week or a past week, then the system should display a message: “No task data available for this week. Please make sure tasks are logged.”

User Story #6

As a user, I would like to be given gentle reminders through pop-ups or alerts suggesting I take breaks after long periods of continuous work.

#	Description	Estimated Time	Owner
1	Implement logic to track continuous work time based on logged tasks.	3 Hrs	Yana
2	Set a predefined threshold for continuous work (e.g., 2 hours) and enable logic checking.	1 Hr	Yana
3	Design and implement a pop-up or alert to notify users to take breaks after reaching the continuous work threshold.	2 Hrs	Yana
4	Implement UI functionality to accept or snooze break reminders.	2 Hrs	Yana
5	Create a system to track whether the break reminder has been accepted or snoozed, and show the reminder accordingly.	2 Hrs	Yana
6	Test the break reminder functionality, ensuring reminders are shown correctly after continuous work periods and that dismissals/snoozes are handled properly and handle errors	2 Hrs	Yana

Acceptance Criteria:

- Given the user adds a time log to the tasks, when the user has been working continuously for the predefined threshold (e.g., 2 hours), then the system should show a pop-up or alert suggesting the user take a break.
- Given the user has not reached the predefined threshold for continuous work (e.g., 2 hours), when the system checks for a break reminder, then no pop-up or alert should appear.
- Given the user receives a break reminder pop-up, when the user clicks “Accept,” then the break reminder should go away.
- Given the user accepts a break reminder, when they resume work, then the continuous work timer should reset.
- Given the user receives a break reminder pop-up, when the user clicks “Snooze,” then the break reminder should be delayed and reappear after a set period (e.g., 5 minutes).
- Given the user accesses reminder settings, when they choose to customize the work threshold (e.g., 1.5 hrs or 3 hrs), then the system should trigger reminders based on the new custom value.
- When the user snoozes the reminder more than 3 times in a row, then the system should force the user to accept or dismiss the reminder—snoozing should be disabled after the limit is reached.
- Given the system enforces a mandatory break, when the user tries to log a new task, then a message should appear indicating that a break is in progress and task logging is temporarily disabled.

User Story #7
As a team member, I would like to be able to add tasks.

#	Description	Estimated Time	Owner
1	Create a simpler version of the main task page unique to each team	5 hrs	Vihaan
2	Have all members be able to add tasks	2 hrs	Vihaan
3	Tasks should have a section for assigning	1 hr	Vihaan
4	Adding tasks should require a name and date but the rest optional	1 hr	Vihaan
5	Add option to delete tasks	1 hr	Vihaan
6	Test and debug	1 hr	Vihaan

Acceptance Criteria:
- Given the team member clicks the button to add a task, they should see a popup asking for a name and date.
- Given the team member forgets either required option (name or date), then they will see a warning when submitting the task.
- Given the team member successfully adds a task, then it should show up on the task table for every team member.
- Given the team has no tasks, it should display a message saying there are no tasks.
- Given the team leader clicks delete on a task, it should be removed.

User Story #8
As a team member, I would like to be able to claim tasks.

#	Description	Estimated Time	Owner
1	Make each task created contain a button for claiming	1 hr	Vihaan
2	Users can add themselves to the assigned section and it will show for everyone	2 hr	Vihaan
3	Users can remove themselves	1 hr	Vihaan
4	Members cannot add or remove others	1 hr	Vihaan
5	Test and debug	1 hr	Vihaan

Acceptance Criteria:
- Given a team member clicks the claim button, then their name should show up in the assignee section.
- Given a task is assigned to a different user, then the current user should not have the option to remove them unless they are the owner.
- Given a team member clicks remove for themselves, their name should be removed.
- Given a task already has an assignee, then users should not be able to add themselves.

User Story #9
As a team leader, I would like to be able to assign tasks.

#	Description	Estimated Time	Owner
1	Add a dropdown or button for each task which contains all users in the team, only visible to owner	2 hr	Vihaan
2	Selecting a user should save them as the assignee	1 hr	Vihaan
3	The team leader can remove any assignee	1 hr	Vihaan
4	Test and debug	1 hr	Vihaan

Acceptance Criteria:

- Given the user is the owner of the team, they should have a different option than everyone else when assigning. They should have a list to choose any member.
- Given the team owner selects a member to assign, their name should show up for everyone as the assignee.
- Given the team owner clicks remove for any user, the user should be removed.
- Given a task already has an assignee, the team leader cannot add another unless they remove it first.

User Story #10

As a team member, I would like to mark team tasks as completed.

#	Description	Estimated Time	Owner
1	All members can mark a task as completed with a button	1 hr	Vihaan
2	It will move the task under a “completed” section	1 hr	Vihaan
3	The completed section can be hidden or expanded	1 hr	Vihaan
4	If the task had an assignee, it will say “completed by: [their name]”	2 hr	Vihaan
5	Can undo completion to bring it back to the normal table	2 hr	Vihaan
6	Test and debug	1 hr	Vihaan

Acceptance Criteria:

- Given the user clicks the button to mark as completed, the task should disappear from the main table and move to a “completed” section.
- Given the user clicks an expand button, the completed section should show its contents and clicking it again will close the section.
- Given the user was an assignee, then completing the task will show they were the one who completed it.
- Given a task does not have an assignee, then completing the task will credit the person who clicked complete.
- Given the user undoes a completion, the task should show back up in the team task table.

User Story #11

As a team member, I would like to leave comments on tasks so that I can communicate updates.

#	Description	Estimated Time	Owner
1	In each team’s task page, next to the task, there should be a comment button visible to the user.	1 hr	Jack

2	Comment UI	1 hr	Jack
3	Showing comment on other user's account	3 hrs	Jack
5	Delete comment button for comment creator	1 hr	Jack
6	Text to display who made the comment	1 hr	Jack
7	Comment buttons on comment UI	1 hr	Jack
8	Debug and test with unit tests.	1 hr	Jack

Acceptance Criteria:

- Given that the comment button is functional, when a team member clicks the comment button, the comment UI should appear on the screen.
- Given that the comment UI and buttons are functional, when a team member clicks the save or close button, they will either save the comment or close the UI.
- Given that the comments sync across team members' accounts, when a user makes a comment, another user in the same team will see the comment and see who it was made by.
- Given that the data persists, when a user reloads the page, the comments and data will persist across the session and still be there.

User Story #12

As a user, I would like to be able to roll over incomplete tasks to the following day, either automatically or manually.

#	Description	Estimated Time	Owner
1	Roll over button	1 hr	Jack
2	Automatic roll over button	1 hr	Jack
3	Automatic roll over time	1 hr	Jack
4	Automatic roll over functionality	3 hr	Jack
5	Visual indicator to mark when a deadline has passed	2 hr	Jack
6	Keep track of the amount of times a task has been rolled over	1 hr	Jack
7	Display amount of rollovers next to a task	1 hr	Jack
8	Debug and test with unit tests.	2 hr	Jack

Acceptance Criteria:

- Given that the system can track deadlines properly, when a deadline is crossed and the task is still not marked as complete, then a visual indicator will appear that asks if the user would like to roll over the task.
- Given that the visual indicator is working properly, when the indicator appears, a button will be presented to the user that they can click.
- Given that the roll over button works, when the user clicks the button, the deadline will be adjusted to be one day after the day the user clicked the button.

- Given the roll over display works, when a user rolls over a task, the display will show how many times total that the user has rolled over that specific task.
- Given that the automatic roll over feature is implemented, when a user clicks the automatic roll over, the task will be rolled over in the same way as the manual one but it will happen on a set day interval instead.
- Given that the automatic roll over time is implemented, when a user clicks the time, they will be given an interface to choose the time when the automatic roll over will roll tasks over to the next day.

User Story #13

As a user, I would like to be able to change my display name for different teams to best suit each team.

#	Description	Estimated Time	Owner
1	A change display name button in each team page	1 hr	Jack
2	Display name UI, buttons, and functionality	2 hrs	Jack
3	Names don't affect other names	1 hr	Jack
4	Names appear to other users in the team as the new name	3 hrs	Jack
5	Reset button and functionality	2 hr	Jack
6	Debug and test with unit tests.	1 hr	Jack

Acceptance Criteria:

- Given that the change display name button works, when the user presses a button, they will be presented with a UI for changing their name.
- Given that the display name UI is functional, when a user is on the UI, they will have the option to input a new name, save the name, and close out of the UI.
- Given that the display names are updated for the team members, when another team member is looking at the user within the team it will show up as the new display name.
- Given that the user is in multiple teams, when a user has changed their name to different things in different teams, it will show up as how the user set in those teams specifically.
- Given that the reset button is functional, when a user clicks the reset button, their display name for that team will be reset to the actual name of their account.

Sprint 2: Incomplete User Stories, Tasks, & Acceptance Criteria

Sprint 1, User Story #4

As a user, I would like to be able to delete tasks.

#	Description	Estimated Time	Owner
3	Implement undo functionality for accidental deletions.	2 Hrs	Jack
4	Debug and test task deletion and persistence with unit tests.	1 Hr	Jack

Acceptance Criteria:

- Given the user clicks "Undo" within a set time frame (e.g., 10 seconds), then the task should reappear in its original state in the task list.

- Given the undo time frame expires, when the user tries to restore the task, then they should no longer be able to undo the deletion.
- Given the user deletes multiple tasks, when they access the undo function, then only the last 10 deleted tasks should be recoverable.
- Given the user deletes more than 10 tasks, when they try to undo, then older deleted tasks (beyond the last 10) should no longer be available for recovery.

User Story #4

As a user, I would like to archive completed tasks so that I can keep my workspace clutter-free.

#	Description	Estimated Time	Owner
5	Implement an auto-archive function that moves completed tasks after a set time.	3 Hrs	Jack
9	Add time of day UI	1 Hr	Jack
10	Sync time of day with auto archival dropdown functionality	2 Hrs	Jack
11	Debug and test with unit tests.	2 Hrs	Jack

Acceptance Criteria:

- Given that automatic archival is implemented correctly, when a user is in the archived section, then there should be an auto archival in the top right corner that has a check box, period drop down, and time of day input.
- Given that automatic archival is enabled, when a task remains completed for a set period, then it should automatically move to the archive.
- Given that automatic archival is disabled, when a user is done with a task, then they will have to archive it themselves and it will remain on the main taskpage.
- Given that the time button works, when a user clicks the calendar, then they should be presented with a UI to enter a time.
- Given that the time UI is functional, when a user inputs their time, then the auto archival period will base when to do the next archival on the time the user input.
- Given that the time UI is functional and robust, when a user inputs a time that doesn't exist, then they will be shown an error saying that it is incorrect.

User Story #5

As a user, I would like to jot notes down about tasks that I have or have not completed so that I can reference thoughts for later.

#	Description	Estimated Time	Owner
3	Implement text editing interface within notes UI with basic formatting button options.	2 Hr	Jack
4	Add a delete notes button in the notes UI.	1 Hr	Jack
5	Make notes persist across multiple sessions and have them saved in the database.	2 Hrs	Jack
6	Debug and test notes features with unit tests.	2 Hrs	Jack

Acceptance Criteria:

- Given that the bullet points option is presented, when a user clicks the bullet points button, then a bullet point will appear that can be typed after, it will also create new bullet points when the user enters.
- Given that the bold option is presented, when a user clicks the bold button, then all text that the user enters within the note will be bolded until the button is unclicked.
- Given that the italics option is presented, when a user clicks the italics button, then all text that the user enters within the note will be italicized until the button is unclicked.
- Given that the delete note button is functional, when a user clicks the delete button, all the notes for that task should disappear and nothing should remain inside.
- Given that the notes persist, when a user refreshes or logs out, then when they are back in the main task page the notes will still be there.

User Story #6

As a user, I would like to set default task templates so that I can quickly create repetitive tasks.

#	Description	Estimated Time	Owner
1	Make default task creation button	1 Hr	Jack
3	Create task template editing, creation, and deletion buttons.	1 Hr	Jack
4	Create and implement "favorite" feature for default task templates	2 Hrs	Jack
5	Implement "most recent" button	1 Hr	Jack
6	Ensure that the task templates persist across sessions and are saved in the database.	2 Hrs	Jack
7	Debug the task template with unit tests	1 Hr	Jack

Acceptance Criteria:

- Given that the task template UI is functional, when the user has more tasks than space in the UI, then they will be able to scroll to see the other default templates.
- Given that the create and edit buttons are functional, when a user clicks them, then it will be the same create and edit interface as the normal tasks.
- Given that the delete button is functional, when a user deletes a task, then it will be given a time based undo like for normal tasks but it will not store the last 10 deleted tasks.
- Given that the favorite feature is implemented, when a user sees a task, they will be given the option to click the favorite icon(a heart) that will put that template at the top of the list of templates.
- Given that the favorite icon is working, when a user clicks the icon, then they will see that there is a noticeable difference between the icon associated with a "favorited" template and a template that is not favorited.
- Given that the most recent button works, when a user has already used a default task template, then they will be presented with a button that will make the most recently used template when clicked.

Backlog

Functional

Task Management

1. ~~As a user, I would like a task page so I can view all my tasks.~~
2. ~~As a user, I would like to be able to create tasks.~~
3. ~~As a user, I would like to be able to edit tasks.~~
4. ~~As a user, I would like to be able to delete tasks.~~
5. ~~As a user, I would like to be able to create subtasks within a task.~~
6. ~~As a user, I would like to drag and drop tasks so that I can easily visualize my task order.~~
7. ~~As a user, I would like to categorize tasks by priority, tags, or deadlines so that I can make decisions on which order to take the tasks.~~
8. ~~As a user, I would like to color code tasks based on categories so that I can quickly identify them.~~
9. ~~As a user, I would like to set default task templates so that I can quickly create repetitive tasks.~~
10. ~~As a user, I would like to set different statuses for tasks (in-progress, blocked, waiting for review).~~
11. ~~As a user, I would like to be able to filter tasks by completion status.~~
12. ~~As a user, I would like to archive completed tasks so that I can keep my workspace clutter-free.~~
13. ~~As a user, I would like to be able to roll over incomplete tasks to the following day, either automatically or manually.~~
14. ~~As a user, I would like to be able to set task dependencies to organize connected tasks better.~~
15. ~~As a user, I would like to schedule tasks for a future start date so that I can plan ahead without cluttering my current task list.~~
16. ~~As a user, I would like to receive smart suggestions for breaking up long tasks into smaller subtasks.~~

Goal-Setting & Progress Tracking

17. ~~As a user, I would like to jot notes down about tasks that I have or have not completed so that I can reference thoughts for later.~~
18. As a user, I would like to track undesirable habits/tasks so that I can check progress on if I'm lowering how unproductive I am (if time allows).

Time-Tracking & Productivity Insights

19. ~~As a user, I would like to log time spent on tasks so that I can keep track of progress.~~
20. As a user, I would like to be able to visualize how my time is spent across different categories(if time allows).
21. ~~As a user, I would like to set an estimated time and an actual time for each of my tasks, and see the difference between the two.~~
22. ~~As a user, I would like to receive a weekly summary of my most and least productive days so that I can adjust my workflow.~~
23. ~~As a user, I would like to compare my productivity across different weeks/months to track my long-term progress.~~
24. As a user, I would like to check time spent on breaks so that I can maintain a healthy balance.
25. ~~As a user, I would like to be given gentle reminders through pop-ups or alerts suggesting I take breaks after long periods of continuous work.~~
26. As a user, I would like to receive productivity tips based on my task history (if time allows).

Gamification & Team Features

27. ~~As a user, I would like to see a streak tracker to visualize my consistent task completion.~~
28. ~~As a user, I'd like to be able to navigate to a profile tab where I can see relevant information about my account.~~
29. As a user, I would like to be able to share or display my streak or other object on my profile.
30. As a team member, I would like to view a productivity leaderboard to foster a sense of competition and teamwork.
31. As a team member, I would like to send and receive kudos for completed tasks to encourage engagement.
32. As a team leader, I would like to assign weighted points to tasks to reflect their complexity.
33. ~~As a user, I would like to be able to search for other users so that I can use other functionality like teams.~~
34. As a user, I would like to be able to be a contributor (leader or member) for multiple unique teams.

35. As a user, I would like to be able to access a unique team page for each of the teams that I am a part of to see team tasks and contributors.
36. ~~As a team leader, I would like to create a "team" so that I can manage group tasks.~~
37. ~~As a team leader, I would like to be able to add users to a team.~~
38. ~~As a team leader, I would like to be able to assign tasks.~~
39. As a team leader, I would like to set permissions for different team members so that I can control who edits tasks.
40. As a team leader, I would like to be able to view team members' activity.
41. As a team leader, I would like to track the time my members are spending on tasks.
42. ~~As a team member, I would like to be able to add tasks.~~
43. ~~As a team member, I would like to be able to claim tasks.~~
44. ~~As a team member, I would like to mark team tasks as completed.~~
45. ~~As a team member, I would like to leave comments on tasks so that I can communicate updates.~~
46. As a team member, I would like to tag people so that they can be notified when a task needs their attention.

User Preferences & Accessibility

47. ~~As a user, I would like to create an account with unique log in credentials.~~
48. ~~As a user, I would like to log in to my account when I return.~~
49. As a user, I'd like my data to be synced to my account so that signing in somewhere new shows me my tasks.
50. As a user, I would like to be able to use multiple devices so that I can be productive and check my tasks no matter my situation.
51. ~~As a user, I would like to access a settings option that allows me to customize my experience, including enabling dark mode or adjusting the app theme, so that I can work in my preferred visual setting.~~
52. ~~As a user, I would like to change the font size and spacing so that I can improve readability based on my preference.~~
53. As a user, I would like to choose what view I am in (calendar, timeline, etc.) so that it can be in my most preferred state.
54. As a user, I would like to set focus mode where notifications are muted for deep working sessions.
55. ~~As a user, I would like to receive notifications so that I can be reminded of important tasks and stay productive.~~
56. ~~As a user, I would like to be able to change my display name for different teams to best suit each team.~~
57. As a user, I would like to be able to change my profile picture so that I can customize my account.
58. ~~As a user, I would like to be able to delete my account if I no longer want to use the app.~~

Integrations & Data Export

59. As a user, I would like to integrate Planorama with third-party tools like Google Calendar or Slack for streamlined workflow (if time allows).
60. As a user, I would like to export my task data in CSV or PDF format for external analysis (if time allows).

Non-Functional

1. The platform must ensure tasks and insights load within 500ms for an optimal user experience while supporting up to 10,000 simultaneous users with minimal latency.
2. The architecture should be scalable with cloud-based solutions to accommodate a growing user base and new features.
3. A responsive design must ensure compatibility across desktop and mobile devices, while an intuitive and consistent user interface minimizes the learning curve.
4. To ensure data security, sensitive user information must be protected with end-to-end encryption, preventing unauthorized access. Additionally, all user actions should be authenticated using OAuth 2.0 standards.
5. The platform must maintain 99.9% uptime, ensuring high availability and reliability for users. Any downtime should be limited to scheduled maintenance periods, minimizing disruptions and optimizing system performance.