Sprint 2 Plan - Planetarium

Scrumbags

Sprint Completion 11/7/2023, Revision 3, 11/4/2023

Goal

Create the basic and foundational functionality and UI for the application. Establish infrastructure and get familiar with Flutter and Firebase.

Task Listing

Check here for the most up to date view.

As a user, I want to be able to…:

1. See my inputted information when I reopen the app so that I do not need to keep it open - 13pt - 6hrs
    a. Support Google SSO - 2hr
    b. Handle Google SSO errors and integrate Google username into database - 2hr
    c. Create methods to read user data from the database into memory upon app launch - 2hr
2. Create tasks so I can add tasks to my schedule - 2pt - 1hr
    a. Create methods that save a task object to the database - 1hr
3. Click on tasks to see more information about them - 5 pt - 3hr
    a. Create a new Widget on the UI to accept new tasks - 1hr
    b. Display tasks in daily any view - 2hr
    c. Display tasks in weekly view – 0hr
    d. Display tasks in monthly view - 0hr
    e. Link the UI Widget to the new method - 0hr
4. Change properties of a task so that I can mark them as completed, move to another date, etc - 2pt - 1hr
    a. Add getters and setters to Task class to update their attributes - 1hr
5. Task gestures so that I can move them around the timeline as necessary - 5pt - 2hr
    a. Add UI for swipe left on a task Widget - 2hr
    b. Add UI for swipe right on a task Widget - 0hr

    c. Add UI for double tap on a task Widget - 0hr
    d. Add UI for holding down on a task Widget - 0hr
    e. Add UI for moving tasks to different times on the timeline - 0hr
6. Create events so that I can add new events to my schedule - 1pt - 1hr
    a. Create methods that save an event object to the database - 1hr
7. Be able to see/Click on events so that I can see more information about them - 8pt - 2hr
    a. Create a new Widget on the UI to create new events - 0hr
    b. Display events in daily view - 2hr
    c. Display events in weekly view - 0hr
    d. Display events in monthly view - 0hr
    e. Link the UI Widget to the new method - 0hr
8. Change properties of an event so that I can change details about location/time/etc - 8pt - 4hr
    a. Add getters and setters to Event class to update their attributes - 1hr
    b. Create recurrence functions to set an event as recurring - 2hr
    c. Support infinite recurrence - 1hr
9. Click on properties of an event so that I can see more detail - 2pt - 1hr
    a. Add UI for seeing event and moving back to different times on the timeline - 1hr
10. Switch between time windows so that I can plan for the short, medium and long term as needed - 8pt-2hr
    a. build the screen for the daily/weekly view - 1hr
    b. Build the screen for the monthly view - 0hr
    c. Add UI buttons and swipe gestures to change time windows - 1hr

Definitions of Done

Task is done when:

- Code pushed into PR and all requested team members have reviewed it (minimum 1)
- Pull request was accepted into main branch
- Documentation describing basic functionality (not too in depth)
- Code compiles and runs without errors

General user story acceptance criteria:

- All tasks are done
- All tests pass
- All acceptance criteria are met

1. See my inputted information when I reopen the app so that I do not need to keep it open - 13pt
    a. Support Google SSO

   b. Create methods to read user data from the database into memory upon app launch
   c. *Acceptance Criteria*
      - ☐ Ensure the user can log in and their data gets successfully saved
      - ☐ Once the user closes the app and reopens it, they can see all the data they had before in the app
2. Create tasks so I can add tasks to my schedule - 2pt
   a. Create methods that save a task object to the database
   b. *Acceptance Criteria*
      - ☐ Successfully add a task to the database
      - ☐ All the data in the database matches with what was written by the user
3. Click on tasks to see more information about them - 8pt
   a. Create a new Widget on the UI to accept new tasks
   b. Display tasks in daily view
   c. Display tasks in weekly view
   d. Display tasks in monthly view
   e. Link the UI Widget to the new method
   f. *Acceptance Criteria*
      - ☐ Be able to see names of tasks in all views and be able to add tasks
      - ☐ In any view, be able to select a task to expand it into a larger view
      - ☐ be able to see more information about it in this view
4. Change properties of a task so that I can mark them as completed, move to another date, etc - 2pt
   a. Add getters and setters to Task class to update their attributes
   b. *Acceptance Criteria*
      - ☐ Successfully mark a task as complete or move it to another day
      - ☐ Changes made are able to be reflected in the database
5. Click on properties of a task so that I can change them as necessary - 5pt
   a. Add UI for swipe left on a task Widget
   b. Add UI for swipe right on a task Widget
   c. Add UI for double tap on a task Widget
   d. Add UI for holding down on a task Widget
   e. Add UI for moving tasks to different times on the timeline
   f. *Acceptance Criteria*
      - ☐ Successfully use gestures to modify/move a task
      - ☐ Be able to see the task is on a new day when the gesture is done
      - ☐ be able to see the task is completed when the gesture is done
6. Create events so that I can add new events to my schedule - 1pt
   a. Create methods that save an event object to the database
   b. *Acceptance Criteria*

- ☐ Be able to create an event and have it persist in the database successfully
- ☐ All the data in the database matches with what was written by the user

7. Click on events so that I can see more information about them - 8pt
    a. Create a new Widget on the UI to create new events
    b. Display events in daily view
    c. Display events in weekly view
    d. Display events in monthly view
    e. Link the UI Widget to the new method
    f. Acceptance Criteria
        - ☐ Be able to see events in all views
        - ☐ Be able to add an event to the planner in all views by bringing up a larger view
        - ☐ Be able to select an event in any view and expand it into a larger view
8. Change properties of an event so that I can change details about location/time/etc - 8pt
    a. Add getters and setters to Event class to update their attributes
    b. Create recurrence functions to set an event as recurring
    c. Support infinite recurrence
    d. Acceptance Criteria
        - ☐ Change an event's location, time, and anything mutable successfully
        - ☐ Recurring tasks are stored in the database
9. Click on properties of an event so that I can see more detail - 2pt
    a. Add UI for moving events to different times on the timeline
    b. Acceptance Criteria
        - ☐ In the larger view of an event, be able to see and edit all fields of the event
        - ☐ Open an event and make sure all the detail is visible
        - ☐ Changes are reflected in the database
10. Switch between time windows so that I can plan for the short, medium and long term as needed - 5pt
    a. build the screen for the daily view
    b. build the screen for the monthly view
    c. add UI buttons and swipe gestures
    d. Acceptance Criteria
        - ☐ Able to move from daily view to monthly view with UI and gestures
        - ☐ All screens display the correct corresponding tasks/events for each time window

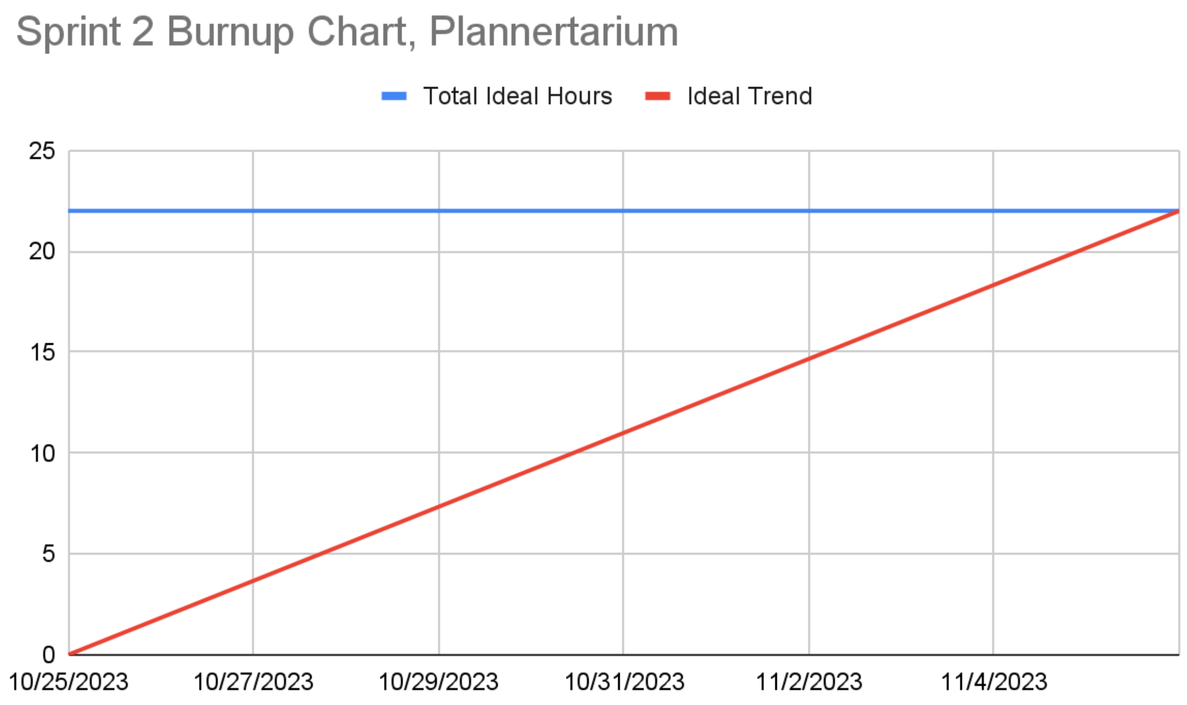
Team Roles

Steven Xue - Backend developer, *Scrum master*
Liam Xu - Frontend developer
Andrew Hu - Frontend developer, Backend developer, *Product Owner*
Andrew Yegiayan - Backend developer
Cheng Wai Chong - Frontend developer

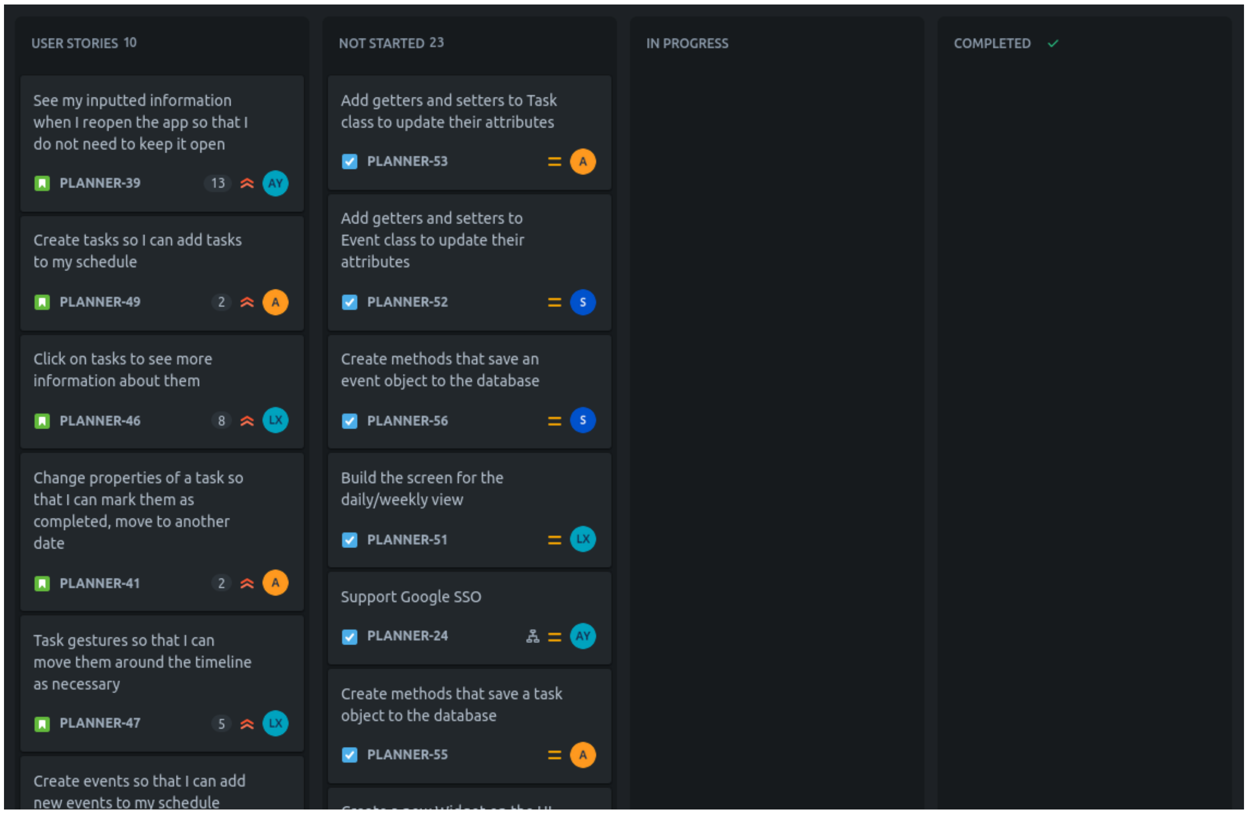
Initial Task Assignment

Cheng Wai Chong - US3, Create a new Widget on the UI to accept new tasks
Liam Xu - US6, build the screen for the daily view
Andrew Yegiayan - US1, Add user authentication to track user accounts
Steven Xue - US8, Add getters and setters to Event class to update their attributes
Andrew Hu - US4, Add getters and setters to Task class to update their attributes

Initial Burnup Chart



Initial Scrum Board



USER STORIES 10

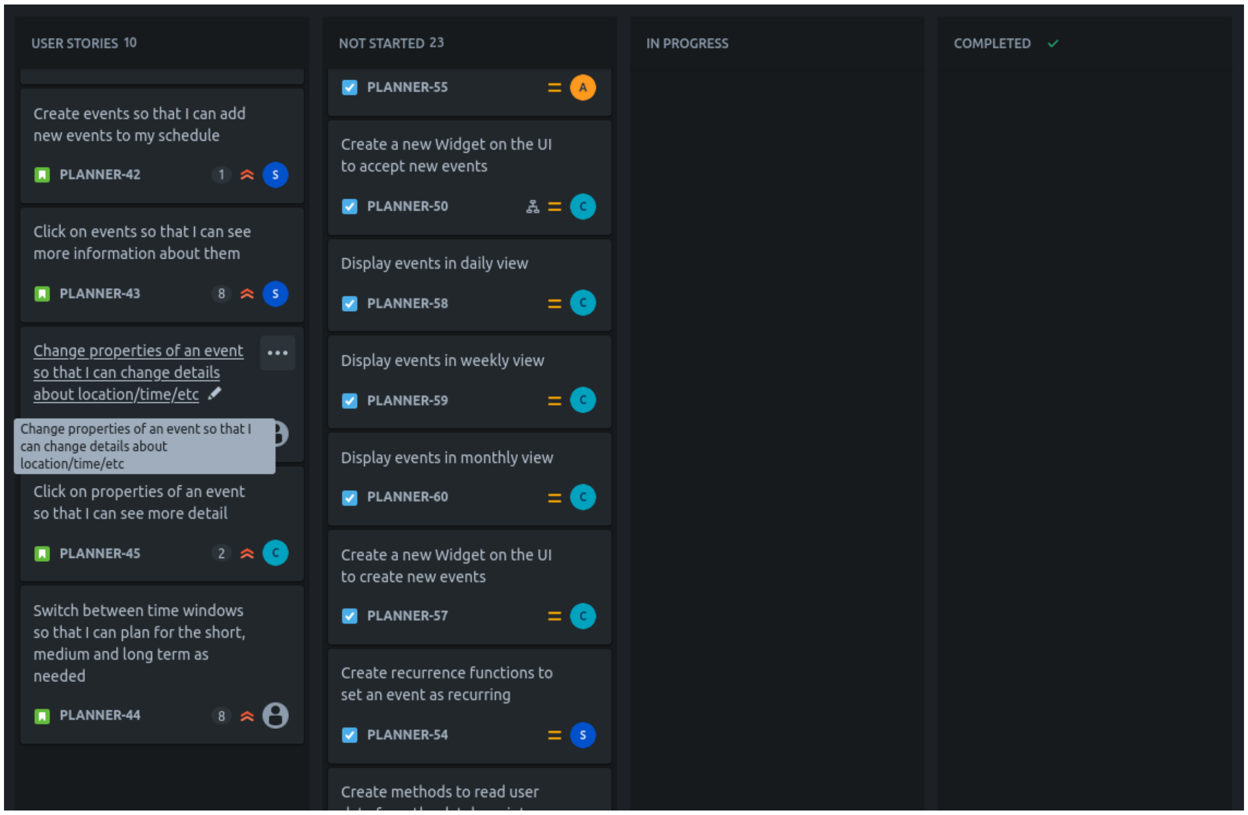
- See my inputted information when I reopen the app so that I do not need to keep it open — PLANNER-39 — 13 — AY
- Create tasks so I can add tasks to my schedule — PLANNER-49 — 2 — A
- Click on tasks to see more information about them — PLANNER-46 — 8 — LX
- Change properties of a task so that I can mark them as completed, move to another date — PLANNER-41 — 2 — A
- Task gestures so that I can move them around the timeline as necessary — PLANNER-47 — 5 — LX
- Create events so that I can add new events to my schedule

NOT STARTED 23

- Add getters and setters to Task class to update their attributes — PLANNER-53 — A
- Add getters and setters to Event class to update their attributes — PLANNER-52 — S
- Create methods that save an event object to the database — PLANNER-56 — S
- Build the screen for the daily/weekly view — PLANNER-51 — LX
- Support Google SSO — PLANNER-24 — AY
- Create methods that save a task object to the database — PLANNER-55 — A
- Create a new Widget on the UI

IN PROGRESS

COMPLETED ✓



USER STORIES 10

- Create events so that I can add new events to my schedule — PLANNER-42 — 1 — S
- Click on events so that I can see more information about them — PLANNER-43 — 8 — S
- Change properties of an event so that I can change details about location/time/etc

Change properties of an event so that I can change details about location/time/etc

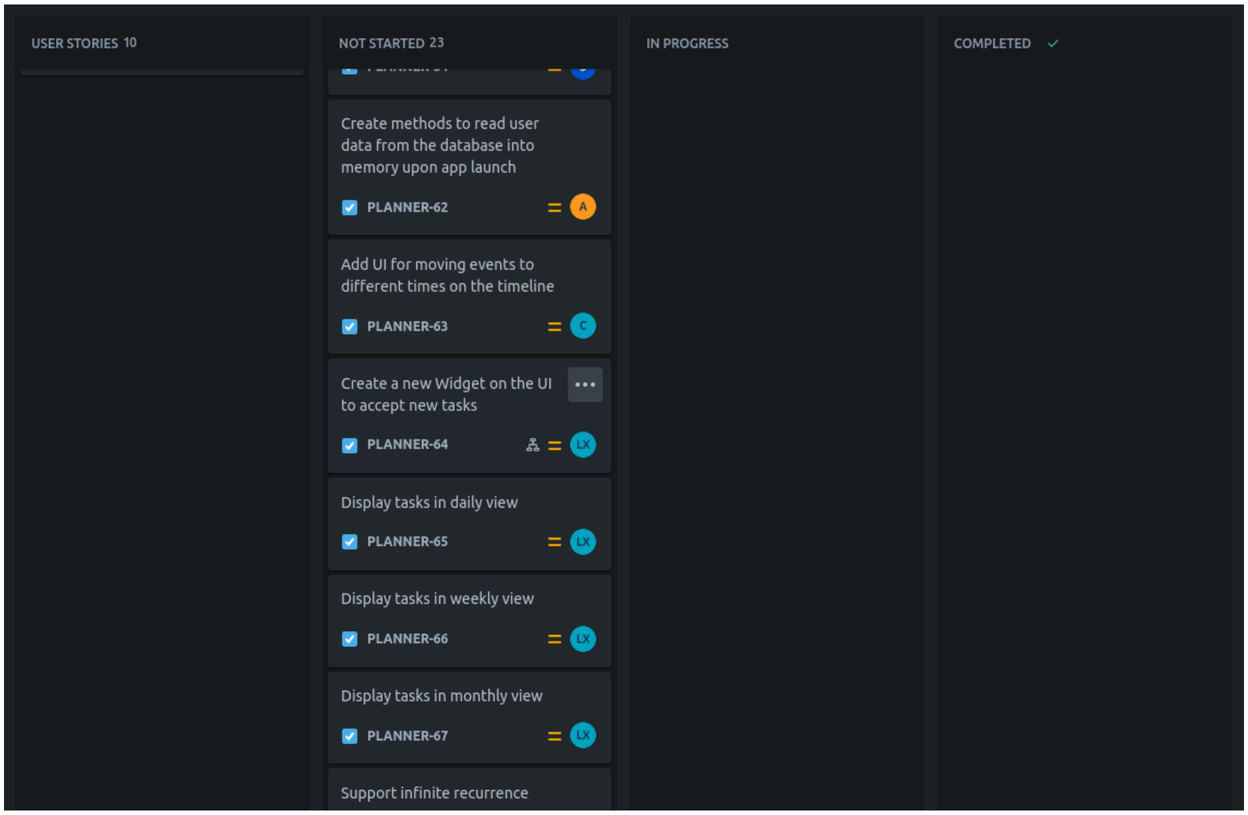
- Click on properties of an event so that I can see more detail — PLANNER-45 — 2 — C
- Switch between time windows so that I can plan for the short, medium and long term as needed — PLANNER-44 — 8

NOT STARTED 23

- PLANNER-55 — A
- Create a new Widget on the UI to accept new events — PLANNER-50 — C
- Display events in daily view — PLANNER-58 — C
- Display events in weekly view — PLANNER-59 — C
- Display events in monthly view — PLANNER-60 — C
- Create a new Widget on the UI to create new events — PLANNER-57 — C
- Create recurrence functions to set an event as recurring — PLANNER-54 — S
- Create methods to read user

IN PROGRESS

COMPLETED ✓



USER STORIES 10

NOT STARTED 23

IN PROGRESS

COMPLETED ✓

Create methods to read user data from the database into memory upon app launch

PLANNER-62 A

Add UI for moving events to different times on the timeline

PLANNER-63 C

Create a new Widget on the UI to accept new tasks

PLANNER-64 LX

Display tasks in daily view

PLANNER-65 LX

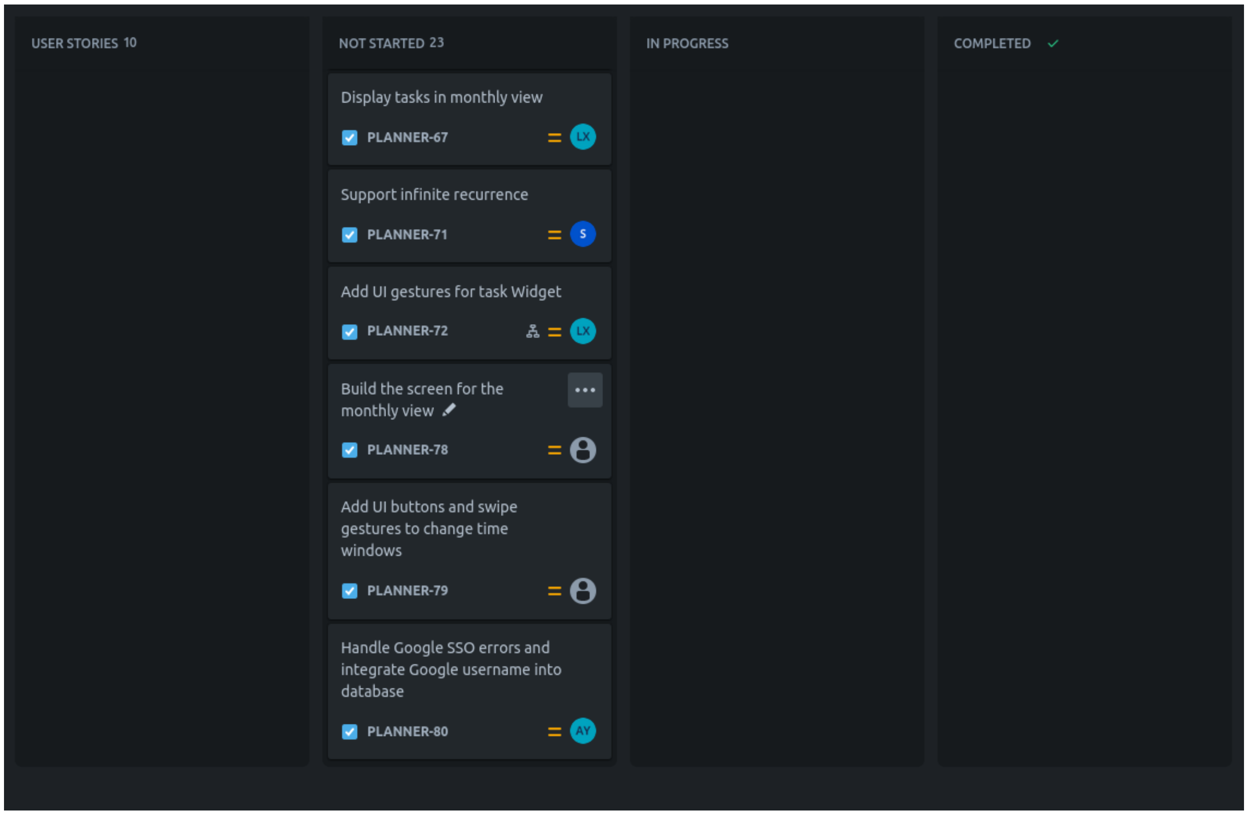
Display tasks in weekly view

PLANNER-66 LX

Display tasks in monthly view

PLANNER-67 LX

Support infinite recurrence



USER STORIES 10

NOT STARTED 23

IN PROGRESS

COMPLETED ✓

Display tasks in monthly view

PLANNER-67 LX

Support infinite recurrence

PLANNER-71 S

Add UI gestures for task Widget

PLANNER-72 LX

Build the screen for the monthly view

PLANNER-78

Add UI buttons and swipe gestures to change time windows

PLANNER-79

Handle Google SSO errors and integrate Google username into database

PLANNER-80 AY

Scrum Times

Monday 12:15pm
Wednesday 6pm
Friday 6pm