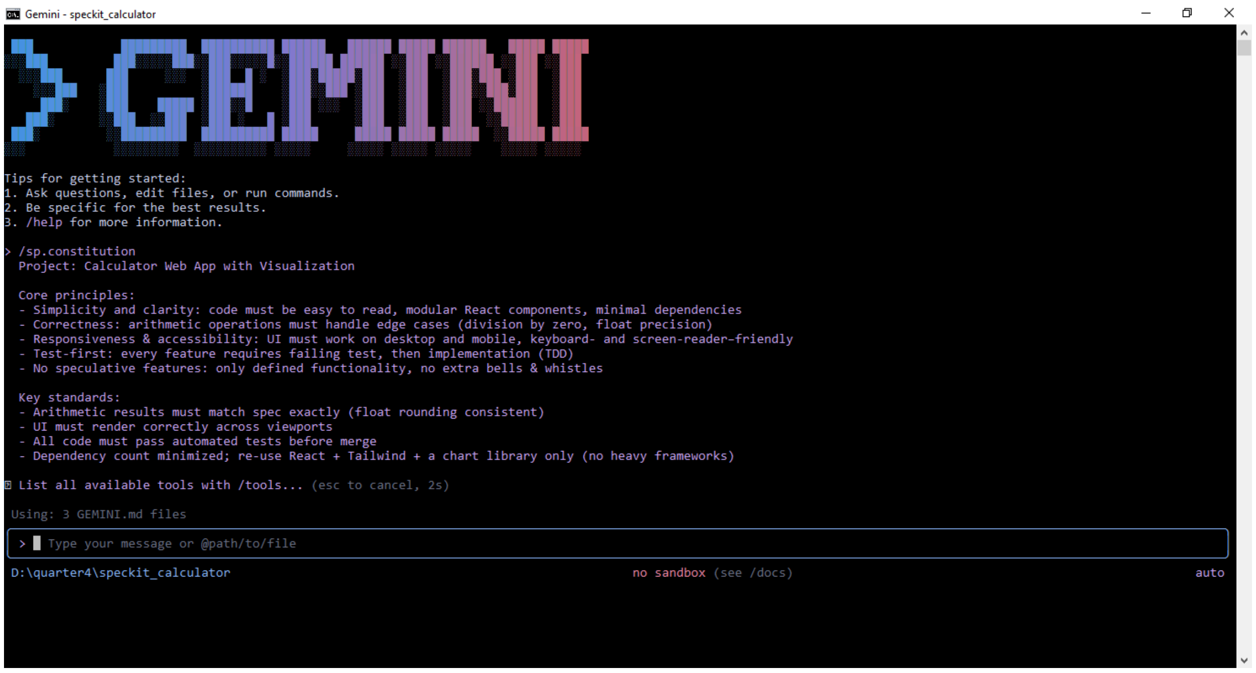
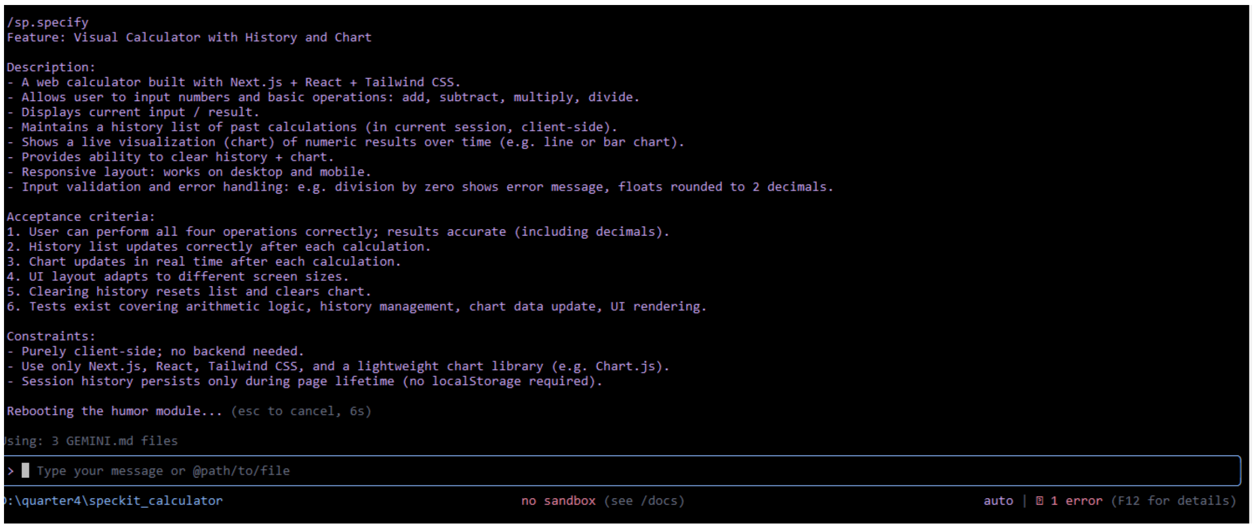
# TASK 8-CALUCULATOR BY USING SPECKITPLUS

## 1. Constitution



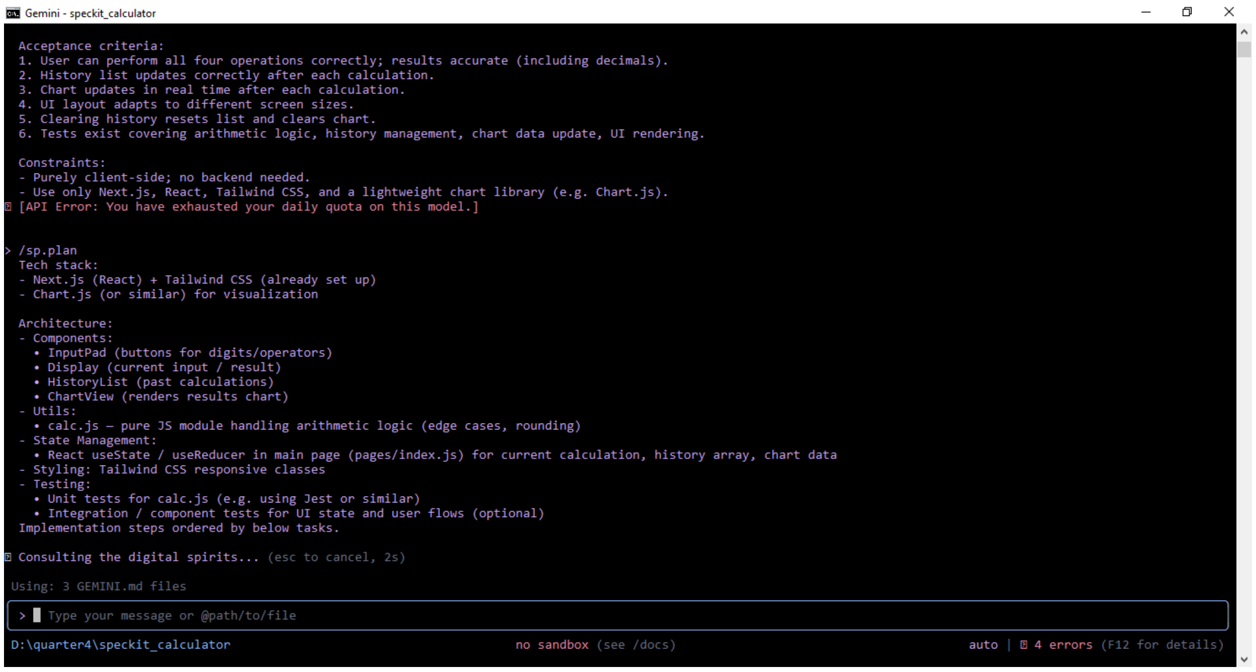
```
Gemini - speckit_calculator

Tips for getting started:
1. Ask questions, edit files, or run commands.
2. Be specific for the best results.
3. /help for more information.

> /sp.constitution
  Project: Calculator Web App with Visualization

  Core principles:
  - Simplicity and clarity: code must be easy to read, modular React components, minimal dependencies
  - Correctness: arithmetic operations must handle edge cases (division by zero, float precision)
  - Responsiveness & accessibility: UI must work on desktop and mobile, keyboard- and screen-reader-friendly
  - Test-first: every feature requires failing test, then implementation (TDD)
  - No speculative features: only defined functionality, no extra bells & whistles

  Key standards:
  - Arithmetic results must match spec exactly (float rounding consistent)
  - UI must render correctly across viewports
  - All code must pass automated tests before merge
  - Dependency count minimized; re-use React + Tailwind + a chart library only (no heavy frameworks)

  List all available tools with /tools... (esc to cancel, 2s)

 Using: 3 GEMINI.md files

 > Type your message or @path/to/file

 D:\quarter4\speckit_calculator                    no sandbox (see /docs)                    auto
```

## 2. Specification



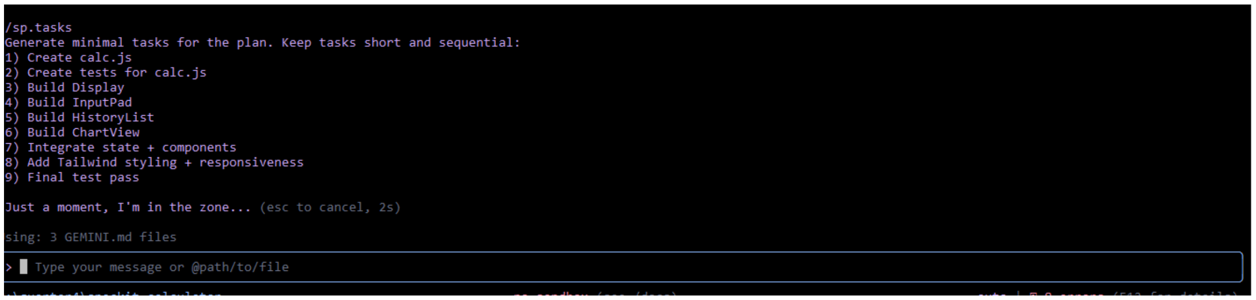
```
/sp.specify
Feature: Visual Calculator with History and Chart

Description:
- A web calculator built with Next.js + React + Tailwind CSS.
- Allows user to input numbers and basic operations: add, subtract, multiply, divide.
- Displays current input / result.
- Maintains a history list of past calculations (in current session, client-side).
- Shows a live visualization (chart) of numeric results over time (e.g. line or bar chart).
- Provides ability to clear history + chart.
- Responsive layout: works on desktop and mobile.
- Input validation and error handling: e.g. division by zero shows error message, floats rounded to 2 decimals.

Acceptance criteria:
1. User can perform all four operations correctly; results accurate (including decimals).
2. History list updates correctly after each calculation.
3. Chart updates in real time after each calculation.
4. UI layout adapts to different screen sizes.
5. Clearing history resets list and clears chart.
6. Tests exist covering arithmetic logic, history management, chart data update, UI rendering.

Constraints:
- Purely client-side; no backend needed.
- Use only Next.js, React, Tailwind CSS, and a lightweight chart library (e.g. Chart.js).
- Session history persists only during page lifetime (no localStorage required).

Rebooting the humor module... (esc to cancel, 6s)

Using: 3 GEMINI.md files

> Type your message or @path/to/file

:\quarter4\speckit_calculator                no sandbox (see /docs)                auto | 1 error (F12 for details)
```

## 3. Plan (high-level)



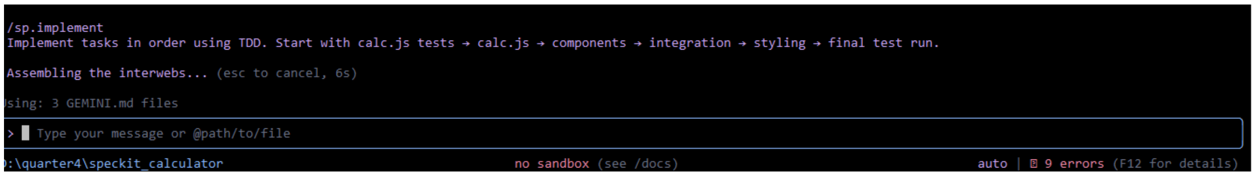
```
Gemini - speckit_calculator

 Acceptance criteria:
 1. User can perform all four operations correctly; results accurate (including decimals).
 2. History list updates correctly after each calculation.
 3. Chart updates in real time after each calculation.
 4. UI layout adapts to different screen sizes.
 5. Clearing history resets list and clears chart.
 6. Tests exist covering arithmetic logic, history management, chart data update, UI rendering.

 Constraints:
 - Purely client-side; no backend needed.
 - Use only Next.js, React, Tailwind CSS, and a lightweight chart library (e.g. Chart.js).
 [API Error: You have exhausted your daily quota on this model.]


> /sp.plan
 Tech stack:
 - Next.js (React) + Tailwind CSS (already set up)
 - Chart.js (or similar) for visualization

 Architecture:
 - Components:
   • InputPad (buttons for digits/operators)
   • Display (current input / result)
   • HistoryList (past calculations)
   • ChartView (renders results chart)
 - Utils:
   • calc.js — pure JS module handling arithmetic logic (edge cases, rounding)
 - State Management:
   • React useState / useReducer in main page (pages/index.js) for current calculation, history array, chart data
 - Styling: Tailwind CSS responsive classes
 - Testing:
   • Unit tests for calc.js (e.g. using Jest or similar)
   • Integration / component tests for UI state and user flows (optional)
 Implementation steps ordered by below tasks.

 Consulting the digital spirits... (esc to cancel, 2s)

 Using: 3 GEMINI.md files
 > Type your message or @path/to/file
 D:\quarter4\speckit_calculator        no sandbox (see /docs)        auto | 4 errors (F12 for details)
```

## 4) /sp.tasks



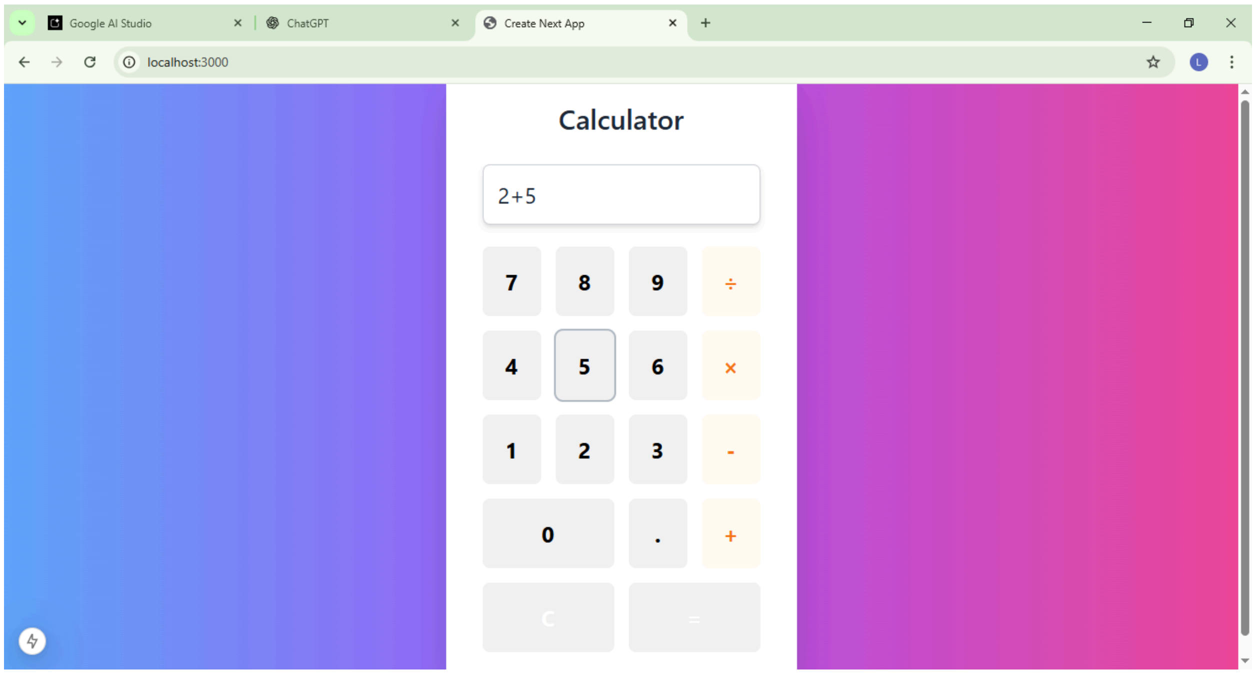
```
/sp.tasks
Generate minimal tasks for the plan. Keep tasks short and sequential:
1) Create calc.js
2) Create tests for calc.js
3) Build Display
4) Build InputPad
5) Build HistoryList
6) Build ChartView
7) Integrate state + components
8) Add Tailwind styling + responsiveness
9) Final test pass

Just a moment, I'm in the zone... (esc to cancel, 2s)

sing: 3 GEMINI.md files
> Type your message or @path/to/file
```

## 5) /sp.implement



```
/sp.implement
Implement tasks in order using TDD. Start with calc.js tests → calc.js → components → integration → styling → final test run.

Assembling the interwebs... (esc to cancel, 6s)

sing: 3 GEMINI.md files
> ▌ Type your message or @path/to/file
:\quarter4\speckit_calculator                     no sandbox (see /docs)                     auto | 9 errors (F12 for details)
```

## Final Caluclator


Google AI Studio
ChatGPT
Create Next App
localhost:3000
Calculator
2+5
7
8
9
÷
4
5
6
×
1
2
3
-
0
.
+
C
=