

## Career Compass MVP: 3-Day Development Plan

This plan breaks the Minimum Viable Product (MVP) into three focused, 5-hour sessions, ensuring the core value proposition (logging activities and generating a branded weekly post via Gemini) is fully functional by the end of Day 3.

### Day 1: Foundation & Daily Logging (Approx. 5 Hours)

**Goal:** Establish the app structure, complete Firebase authentication, and finalize the data persistence for daily logs.

Task	Detail	Estimated Time	Status Check
1.0 Setup & Firebase Init	Verify basic HTML/Tailwind structure. Initialize Firebase app, Firestore, and Auth instances.	1.0h	Firebase services initialized.
1.1 Authentication	Implement <code>onAuthStateChanged</code> . Handle <code>signInWithCustomToken</code> (if available) or <code>signInAnonymously</code> . Capture <code>userId</code> .	1.5h	<code>userId</code> displayed in UI. Logging button enabled.
1.2 Daily Log Form	Wire up the HTML form inputs ( <code>date</code> , <code>title</code> , <code>description</code> , <code>imageUrl</code> ) to capture data via event listeners.	1.0h	Form data is readable in JavaScript.
1.3 Firestore Saving	Implement the <code>handleSubmitLog</code> function, using <code>addDoc</code> to save the logged activity to the Firestore collection ( <code>/artifacts/{appId}/users/{userId}/daily_logs</code> ).	1.5h	New entry successfully appears in Firestore. Form resets on submission.
Day 1 Total		5.0 Hours	Daily logs are saved persistently.

### Day 2: Data Aggregation & Real-Time UI (Approx. 5 Hours)

**Goal:** Implement real-time data retrieval, filtering, and rendering of the weekly activities.

Task	Detail	Estimated Time	Status Check
------	--------	----------------	--------------

2.1 Real-time Listener	Implement <code>setupLogListener</code> using <code>onSnapshot</code> to fetch and update the global <code>logs</code> array in real-time. Use <code>orderBy('timestamp', 'desc')</code> .	1.5h	UI updates instantly when a new log is submitted.
2.2 Week Calculation Logic	Implement <code>getWeekStart</code> and <code>getWeekEnd</code> functions to accurately calculate the current week's boundary dates (Sunday to Saturday).	1.0h	Week dates displayed correctly in the aggregation header.
2.3 Weekly Filtering & Render	Implement <code>renderWeeklyActivities</code> to filter the <code>logs</code> array based on the current week's boundaries. Dynamically render the logs in the "Weekly Aggregation" list.	1.5h	Only current week's logs appear in the list.
2.4 Generator Button Logic	Update the text on the "Generate Post" button to display the count of activities ( <code>0 Activities</code> ). Enable/disable the button based on activity count.	1.0h	Button accurately reflects the number of logs.
Day 2 Total		5.0 Hours	Weekly logs are aggregated and displayed accurately.

Day 3: AI Branding & Polish (Approx. 5 Hours)

**Goal:** Integrate the Gemini API, complete the core post-generation feature, and finalize the MVP with necessary polish.

Task	Detail	Estimated Time	Status Check
3.1 AI Prompt Construction	Finalize the prompt structure (system and user query) within the <code>handleGeneratePost</code> function, ensuring it dynamically includes all aggregated <code>weeklyActivities</code> .	1.0h	Prompt string is correctly formatted with all log data.
3.2 Gemini API Call	Implement the <code>withExponentialBackoff</code> wrapper and the core <code>fetch</code> logic for the Gemini API call ( <code>gemini-2.5-flash-preview-09-2025</code> ).	1.5h	Successful Markdown post received from the API and logged to the console.
3.3 Output Display & Loading	Update the UI to display the generated post ( <code>generatedPostElement</code> ). Implement loading state ( <code>isGenerating</code> variable and spinner icon) and success/error messages.	1.5h	Post appears in the output box upon completion; loading spinner works.

**3.4 Copy-to-Clipboard & Polish**

Implement the `handleCopyToClipboard` function. Perform final visual checks, fix any remaining style issues, and confirm all Lucide icons render correctly.

1.0h

Post copies to clipboard; MVP is aesthetically complete and functional.