PRD

## **1. PRD Information**

* **Feature Name**: Journey Logs
* **Owner**: [Product Manager]
* **Contributors**: [Architects], [Tech Leads & Developers], [Designers]
* **Approvers:** [Product Leads], [Engineering Leads], [Design Leads]
* **Version History**:

| **Version** | **Creation Date** | **Owner** | **Contributors** | **Changelog** | **Signed off by** |
| --- | --- | --- | --- | --- | --- |
| v1 | 15th May, 2025 | [Person] | [Person] | First draft of PRD | [Person] - [Date] |
|  |  |  | [Person] |  | [Person] - [Date] |
|  |  |  | [Person] |  | [Person] - [Date] |

## **2. Overview**

### **2.1 Objective**

The objective of the Journey Logs feature is to provide users with a comprehensive view of customer progression through marketing journeys, enabling efficient monitoring, troubleshooting, and optimization of customer journeys. This feature will allow marketing managers, analysts, and customer support representatives to access and analyze log data for specific journeys, helping them identify issues and make data-driven decisions.

### **2.2 Background**

Journey Logs are designed to enhance the journey management experience by providing visibility into how users interact with and progress through marketing journeys. This visibility allows teams to:

* **Monitor**: Track customer progression through journeys in real-time
* **Troubleshoot**: Quickly identify and resolve issues for specific customers
* **Analyze**: Understand journey performance and identify optimization opportunities
* **Support**: Provide better customer support by having visibility into customer journey status

## **3. Scope**

### **3.1 In-Scope**

* View of journey logs with event date time, user ID, block name, and status
* Real-time log streaming for active journeys
* Navigation between journey builder and logs view
* Filtering logs by user ID
* Pagination controls with customizable rows per page
* Status visualization with color-coding for different statuses
* Data retention disclaimer for 3-month time scope
* Engagement Block status should reflect Message delivery status

### **3.2 Out of Scope**

* Exporting log data to CSV (future enhancement)
* Filtering by additional criteria such as status and block name (future enhancement)
* Date range selection for filtering logs (future enhancement)
* Custom log fields
* Anomaly detection for identifying unusual patterns in journey logs (future enhancement)

## **4. Product Requirements**

### **4.1 Epics**

**Epic 1**: Core Functionality

* **Description**: Enable users to view and navigate journey logs with essential information such as event date time, user ID, block name, and status, with real-time streaming capabilities for active journeys.

**Epic 2**: Filtering and Search

* **Description**: Provide functionality to filter journey logs by User ID and understand data limitations.

**Epic 3**: Pagination and Display Options

* **Description**: Allow users to customize their view with pagination controls and rows per page options.

**Epic 4**: Status Visualization

* **Description**: Implement clear visual indicators for different log statuses to improve usability.

### **4.2 User Stories**

**Epic 1**: Core Functionality

* **User Story 1**: As a marketing manager, I want to view logs for a specific journey, so that I can monitor user progression through the journey and identify any issues.
  + **Acceptance Criteria**:
    - Journey logs page displays the journey name in the title
    - Logs show Event Date Time, User ID, Block Name, and Status
    - Logs are sorted by Event Date Time in descending order (newest first)
    - Each status has a distinct color coding for quick visual identification
    - Logs for active journeys appear in real-time without requiring page refresh
* **User Story 2**: As a marketing manager, I want to easily navigate between the journey builder and logs view, so that I can analyze logs and make journey adjustments efficiently.
  + **Acceptance Criteria**:
    - "View Logs" option is available in the journey's context menu
    - Back button in logs view returns to the journey builder
    - Journey name is preserved when navigating between views

**Epic 2**: Filtering and Search

* **User Story 3**: As a customer support representative, I want to filter journey logs by User ID, so that I can quickly find and troubleshoot issues for specific customers.
  + **Acceptance Criteria**:
    - Search field allows filtering by User ID
    - Filtering updates results in real-time
    - Clear indication of active filters is displayed
    - "No results" message appears when no logs match filter criteria
    - Filter can be easily cleared
* **User Story 4**: As a data analyst, I want to understand the time scope of available log data, so that I can make informed decisions about data analysis and reporting.
  + **Acceptance Criteria**:
    - Clear disclaimer indicates logs are only available for the last 3 months
    - Information about accessing archived logs is provided
    - Disclaimer is prominently displayed but doesn't disrupt the user experience

**Epic 3**: Pagination and Display Options

* **User Story 5**: As a marketing analyst, I want to adjust the number of logs displayed per page, so that I can view more or fewer logs based on my current task and screen size.
  + **Acceptance Criteria**:
    - Dropdown allows selection of 10, 25, 50, or 100 rows per page
    - Page resets to first page when rows per page changes
    - Current selection is clearly indicated
    - Total number of entries and current range is displayed
* **User Story 6**: As a journey manager, I want to easily navigate through multiple pages of logs, so that I can review all relevant log entries efficiently.
  + **Acceptance Criteria**:
    - Pagination controls allow navigation to first, previous, specific, next, and last pages
    - Current page is visually highlighted
    - Pagination controls are disabled when not applicable (e.g., previous button on first page)
    - Page numbers adjust dynamically based on current page and total pages

**Epic 4**: Status Visualization

* **User Story 7**: As a marketing operations specialist, I want to quickly identify the status of each log entry, so that I can focus on specific outcomes (completed, failed, waiting, or skipped).
  + **Acceptance Criteria**:
    - Completed status is displayed with green background
    - Failed status is displayed with red background
    - Waiting status is displayed with blue background
    - Skipped status is displayed with gray background
    - Status badges are consistently styled and easily distinguishable
* **User Story 8**: As a marketing operations specialist, I want to view the message status of each engagement log entry, so that I can understand if the message was sent or not (sent, delivered, undelivered, or skipped).

## **5. UI/UX Design**

* **To be updated by the Design team**
* [**https://v0-journey-live-more.vercel.app/**](https://v0-journey-live-more.vercel.app/)

**Click on three ellipses on the top right and then click on View Logs in the above link ^**

* Key components to include:
  + Journey logs table layout
  + Status badge designs and color scheme
  + Pagination controls
  + Filter interface
  + Navigation elements between journey builder and logs
  + Real-time log streaming indicators and controls
  + Auto-scroll functionality for real-time log viewing

## **6. Technical Requirements**

### **6.1 Fair Usage**

* Implement rate limiting for log retrieval to prevent excessive load
* Limit the maximum number of rows that can be retrieved at once (maximum 100 per page)
* Apply caching strategies to improve performance for frequently accessed log data
* Implement WebSocket or Server-Sent Events (SSE) for efficient real-time log streaming
* Optimize database queries to handle continuous real-time log requests

### **6.2 Security & Compliance**

* **Permissions**: Ensure only authorized users can view journey logs
* **API Access**: Restrict API access for log data to specific user roles
* **Risk**: Journey logs may contain sensitive customer information
  + **Mitigation**: Apply appropriate data masking for sensitive fields
* **Risk**: High log volume could impact system performance
  + **Mitigation**: Implement pagination, filtering, and indexing strategies
* **Risk**: Real-time streaming could impact server performance
  + **Mitigation**: Implement throttling and use efficient WebSocket connections

## **7. Dependencies**

* **Journey Builder Integration**: Dependency on the **Journey Builder Team** to implement the "View Logs" option in journey context menu
* **Data Retention Policy**: Dependency on the **Data Engineering Team** to implement the 3-month data retention policy and archiving
* **User Authentication and Permissions**: Dependency on the **Security Team** to ensure proper access controls
* **Real-time Infrastructure**: Dependency on the **Infrastructure Team** to support WebSocket or SSE capabilities for real-time streaming
* **Event Processing**: Dependency on the **Events Platform Team** to ensure events can be processed and displayed in real-time

## **8. Release Plan**

### **8.1 Timelines**

| **Phase** | **Epics** | **Release Date** |
| --- | --- | --- |
| Phase 1 | Epic 1 (Core Functionality with real-time streaming) | 15th July 2025 |
|  | Epic 4 (Status Visualization) |  |
| Phase 2 | Epic 2 (Filtering and Search) | 1st August 2025 |
| Phase 3 | Epic 3 (Pagination and Display Options) | 15th August 2025 |

### **8.2 Release Tasks**

**Phase 1**

* Release:
  + Beta release
  + Hypercare period
  + Final release
* Documentation:
  + Product documentation draft
  + Release notes draft
  + API doc confluence link
  + Status
* Academy:
  + Product training format and date
  + Quizzes
  + Status
* Product Marketing:
  + GTM kit draft
  + Status
* Pricing:
  + Pricing details
  + Status
* Customer onboarding:
  + Customer onboarded
  + Feedback/tickets/challenges
  + Status

## **9. Metrics**

Metrics that need to be tracked:

* Adoption metrics:
  + Number of users accessing journey logs by role (marketing managers, analysts, support)
  + Percentage of journeys where logs are viewed
  + Frequency of log access per journey
  + Number of users utilizing real-time streaming feature
* Usage and Usability metrics:
  + Average time spent on journey logs page
  + Most commonly used filters
  + Pagination pattern usage
  + Number of User ID searches performed
  + Frequency of pausing/resuming real-time updates
  + Number of tickets created around Journey Logs
    - As designed
    - Tasks
    - Enhancements
* Performance and Errors metrics:
  + Journey logs page loading time
  + Log retrieval response time
  + Filter application response time
  + Real-time streaming latency
  + WebSocket connection stability
  + Number of "no results" occurrences
  + Number of bug tickets
  + Server load during peak usage of real-time streaming
* Documentation metrics:
  + Top doc bot queries related to journey logs
  + Percent queries answered
* Academy metrics:
  + Journey logs course completions
* Marketing metrics:
  + Feature usage in demos
  + Customer feedback on real-time streaming value
  + Customer feedback on feature value overall

## **10. Appendix**

### **10.1 Research**

* **User Interviews**: Summary of user feedback regarding journey visibility needs
* **Competitive Analysis**: Overview of similar logging features in competitor marketing automation platforms

### **10.2 Key Decisions**

* Decision to limit log data to 3 months for performance reasons
* Selection of the four primary status types (completed, failed, waiting, skipped)
* Choice to prioritize User ID filtering in initial release
* Decision to implement real-time log streaming for active journeys to provide immediate visibility into journey progression
* Selection of WebSocket/SSE technology for real-time communication

### **10.3 Open Questions**

* Should journey logs be exportable in the initial release?
* What additional filters should be prioritized for future releases?
* Should users be able to configure custom alerts based on log patterns?
* What is the maximum number of concurrent real-time streaming sessions the system should support?
* How should the system handle performance degradation during high-volume real-time streaming?
* What types of anomalies should be detected in a future anomaly detection feature?
* Should anomaly detection be rule-based or use machine learning approaches?

Tech doc

* [Objective](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4812669730/Journey+Logs#Objective)
* [Scope](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4812669730/Journey+Logs#Scope)
  + [Recently Entered Customer List](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4812669730/Journey+Logs#Recently-Entered-Customer-List)
  + [User Log Details:-](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4812669730/Journey+Logs#User-Log-Details%3A-)
  + [Customet Not Part Of Version:-](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4812669730/Journey+Logs#Customet-Not-Part-Of-Version%3A-)
  + [Customer journey in multiple versions](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4812669730/Journey+Logs#Customer-journey-in-multiple-versions)
  + [Searching a Customer](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4812669730/Journey+Logs#Searching--a-Customer)

**Author** : @Tofan Kumar

**Reviewers**: @Prashu Chaudhary @Prasenjit Saha

**PRD**:-

**JIRA**:- [CAP-153083: Engage BE - [Journey logs] - [Journey Testing & Debugging] Closed](https://capillarytech.atlassian.net/browse/CAP-153083)

**TestCase:-**

## **Objective**

The objective of the Journey Logs feature is to provide users with a comprehensive view of customer progression through marketing journeys, enabling efficient monitoring, troubleshooting, and optimization of customer journeys. This feature will allow marketing managers, analysts, and customer support representatives to access and analyze log data for specific journeys, helping them identify issues and make data-driven decisions.

## **Scope**

### **Recently Entered Customer List**

The UI should display a paginated list of customers who have entered a specific journey meta, sorted in descending order by entry time (most recent entries displayed first).

**Solution:-**

* Develop a new backend API in Adiona with pagination support
* API will query the journeys collection filtered by journeyMeta and organization ID
* Results sorted by \_id in descending order
* Return user IDs for UI consumption
* UI will use returned user IDs to fetch customer details (first name, last name) for display

**API Contract :-**

endpoint -

/v1/journeys/journeyMetaId/{journeyMetaId}/version/{journeyVersion} ?pageNumber=0&limit=100

method - GET

pagination support - YES

queryParam - pageNumber, limit

sample response -

{

"data": {

"totalEntryCount": 100,

"users": [

{

"userId": 1234,

"enteredAtInMs": 1752228044234

}

]

},

"errors": null

}

### **User Log Details:-**

Users should be able to view detailed journey logs for individual customers by clicking on a customer entry.

**Solution:**

* Leverage existing API endpoint: /journeyMetaId/{journeyMetaId}/version/{journeyVersion}/userId/{userId}
* This endpoint currently returns block execution details and entry timings

**Additional Enhancements:**

* Integrate message status retrieval for engagement blocks in the existing API
* Create new Thrift method in Veneno-Listener service
* Method will query inboxes table to fetch message status for specific user

message status details can be found under **additionalDetails** field incase of engagement blocks.

this field would contain both message status and message description. these are the possible values .

"addtionalDetails": {  
 "messageStatus": "SENT",  
 "messageLabel": "Sent to gateway, but no response has been received. Its an intermediate status"  
 }

{

"success": true,

"status": 200,

"result": {

"data": [

{

"orgId": 50146,

"orgOUId": -1,

"journeyMetaId": "688a471959c9e12d8984f0b6",

"journeyVersion": 0,

"journeyId": "688a472e59c9e12d8984f104",

"levelId": "c21537e6-8a7b-41d0-a8f4-79f2c424adfa",

"userId": 438447887,

"status": "COMPLETED",

"migratedFromPreviousVersion": false,

"currentBlockId": "end\_1753892633",

"currentBlockState": "COMPLETED",

"lastIssuedTokenNo": 1,

"lastConsumedTokenNo": 1,

"enteredAtInMs": 1753892637774,

"lastUpdatedTimeInMs": 1753892655010,

"migrationHistory": {

"contextType": "MIGRATE\_JOURNEY",

"movedFrom": {

"journeyMetaId": "688a471959c9e12d8984f0b6",

"journeyVersion": 0

},

"movedTo": {

"journeyMetaId": "688a471959c9e12d8984f0b6",

"journeyVersion": 1

}

},

"journeyBlockHistory": [

{

"blockExecDetailId": "688a472e59c9e12d8984f105",

"blockId": "entry\_1753892633",

"blockType": "ENTRY",

"currentState": "COMPLETED",

"execEffect": {

"effectType": "ENTRY",

"eventId": "c21537e6-8a7b-41d0-a8f4-79f2c424adfa",

"eventType": "transactionAdded",

"eventDate": 1753892637774,

"path": {

"pathName": "sp1",

"pathBlockId": "email\_1753892633"

}

},

"enteredAtInMs": 1753892654519,

"lastUpdatedTimeInMs": 1753892654692,

"blockExecLogs": [

]

},

{

"blockExecDetailId": "688a472e59c9e12d8984f10a",

"blockId": "email\_1753892633",

"blockType": "EMAIL\_ENGAGEMENT",

"currentState": "COMPLETED",

"execEffect": {

"effectType": "EMAIL\_ENGAGEMENT",

"eventId": "1a1a2793-e0bb-44b7-a0be-81c1c4f5fd13",

"eventType": "EXECUTE",

"eventDate": 1753892654696,

"communicationRequestId": "ee0837af-d060-455e-aac2-8829dcf9947d",

"errorMessage": null,

"errorStack": null

},

"enteredAtInMs": 1753892654695,

"lastUpdatedTimeInMs": 1753892654917,

"messageStatus": "SENT",

"additionalDetails": {

"messageStatus": "SENT",

"messageLabel": "Sent to gateway, but no response has been received. Its an intermediate status"

},

"blockExecLogs": [

]

},

{

"blockExecDetailId": "688a472e59c9e12d8984f10f",

"blockId": "end\_1753892633",

"blockType": "END",

"currentState": "COMPLETED",

"execEffect": {

"effectType": "END",

"eventId": "6bed98ed-339c-4a01-ba5d-b04485703b8d",

"eventType": "EXECUTE",

"eventDate": 1753892654926

},

"enteredAtInMs": 1753892654925,

"lastUpdatedTimeInMs": 1753892655007,

"blockExecLogs": [

]

}

]

}

],

"errors": null

}  
}

### **Customet Not Part Of Version:-**

customer entered different version and not available in the selected version. we can use the journey history fetch api using the journeymeta id. it will return the block execution details for each version of the journey meta

### **Customer journey in multiple versions**

for journeys created through migration (migration from one version to another ) there is a new field named as **migrationHistory** added in the api response which would store the migration history. context type would be **MIGRATE\_JOURNEY** in this case.

"migrationHistory": {  
 "contextType": "MIGRATE\_JOURNEY",  
 "movedFrom": {  
 "journeyMetaId": "688a471959c9e12d8984f0b6",  
 "journeyVersion": 1  
 },  
 "movedTo": {  
 "journeyMetaId": "688a471959c9e12d8984f0b6",  
 "journeyVersion": 1  
 }  
 }

{

"success": true,

"status": 200,

"result": {

"data": [

{

"orgId": 50146,

"orgOUId": -1,

"journeyMetaId": "688a471959c9e12d8984f0b6",

"journeyVersion": 0,

"journeyId": "688a472e59c9e12d8984f104",

"levelId": "c21537e6-8a7b-41d0-a8f4-79f2c424adfa",

"userId": 438447887,

"status": "COMPLETED",

"migratedFromPreviousVersion": false,

"currentBlockId": "end\_1753892633",

"currentBlockState": "COMPLETED",

"lastIssuedTokenNo": 1,

"lastConsumedTokenNo": 1,

"enteredAtInMs": 1753892637774,

"lastUpdatedTimeInMs": 1753892655010,

"migrationHistory": {

"contextType": "MIGRATE\_JOURNEY",

"movedFrom": {

"journeyMetaId": "688a471959c9e12d8984f0b6",

"journeyVersion": 1

},

"movedTo": {

"journeyMetaId": "688a471959c9e12d8984f0b6",

"journeyVersion": 1

}

},

"journeyBlockHistory": [

{

"blockExecDetailId": "688a472e59c9e12d8984f105",

"blockId": "entry\_1753892633",

"blockType": "ENTRY",

"currentState": "COMPLETED",

"execEffect": {

"effectType": "ENTRY",

"eventId": "c21537e6-8a7b-41d0-a8f4-79f2c424adfa",

"eventType": "transactionAdded",

"eventDate": 1753892637774,

"path": {

"pathName": "sp1",

"pathBlockId": "email\_1753892633"

}

},

"enteredAtInMs": 1753892654519,

"lastUpdatedTimeInMs": 1753892654692,

"blockExecLogs": [

]

},

{

"blockExecDetailId": "688a472e59c9e12d8984f10a",

"blockId": "email\_1753892633",

"blockType": "EMAIL\_ENGAGEMENT",

"currentState": "COMPLETED",

"execEffect": {

"effectType": "EMAIL\_ENGAGEMENT",

"eventId": "1a1a2793-e0bb-44b7-a0be-81c1c4f5fd13",

"eventType": "EXECUTE",

"eventDate": 1753892654696,

"communicationRequestId": "ee0837af-d060-455e-aac2-8829dcf9947d",

"errorMessage": null,

"errorStack": null

},

"enteredAtInMs": 1753892654695,

"lastUpdatedTimeInMs": 1753892654917,

"messageStatus": "SENT",

"addtionalDetails": {

"messageStatus": "SENT",

"messageLabel": "Sent to gateway, but no response has been received. Its an intermediate status"

},

"blockExecLogs": [

]

},

{

"blockExecDetailId": "688a472e59c9e12d8984f10f",

"blockId": "end\_1753892633",

"blockType": "END",

"currentState": "COMPLETED",

"execEffect": {

"effectType": "END",

"eventId": "6bed98ed-339c-4a01-ba5d-b04485703b8d",

"eventType": "EXECUTE",

"eventDate": 1753892654926

},

"enteredAtInMs": 1753892654925,

"lastUpdatedTimeInMs": 1753892655007,

"blockExecLogs": [

]

}

]

}

],

"errors": null

}

}

### **Searching a Customer**

Implement search capability allowing users to find specific customers within journey logs.

* Develop new Adiona API endpoint
* API accepts user IDs and journeymeta info as input parameters
* Returns corresponding journey details

**API Contract :-**

endpoint -

/v1/journeys/journeyMetaId/{journeyMetaId}/version/{journeyVersion}/filter?userIds=1234,5678

method - GET

sample response -

{

"data": [

{

"userId": 1234,

"enteredAtInMs": 1752228044234

},

{

"userId": 5678,

"enteredAtInMs": 1752228044234

}

],

"errors": null

Doc

# Documentation

Doc link: <https://docs.capillarytech.com/docs/search-user-journey-history#/>

# **Search User Journey History**

The journey history feature provides a detailed and chronological record of all events as customers progress through a journey. Each log entry includes information such as the entry date and time for each journey block, and the corresponding status (for example, entered, wait completed, exited). This helps you monitor customer progression in real time, troubleshoot issues for individuals, and analyse the journey's performance.

You can use the search option and search for a customer who entered a journey using an identifier such as email, mobile number, customer ID and external ID.

You can also use the Entry date and time dropdown to check multiple logs for the customer if they have entered multiple times.