Software Design Specifications

Video Analytics System

Version: 1.3

Project Code	https://github.com/k232003-Talal/Video_Platform_Analytics_System
Supervisor	-
Co Supervisor	-
Project Team	Syed Umer Taiyab Daniyal Ahmed Talal Ali
Submission Date	8/May/2025

Document History

[Revision history will be maintained to keep a track of changes done by anyone in the document.]

Version	Name of Person	Date	Description of change
1.0	Daniyal Ahmed	5-5-2025	Document Created
1.1	Umer Taiyab	6-5-2025	Added Data Dictionary
1.2	Talal Ali	7-5-2025	Added Sequence and State Diagrams
1.3	Umer Taiyab	7-5-2025	Added References

Distribution List

[Following table will contain list of people whom the document will be distributed after every sign-off]

Name	Role
Talal Ali	Team Lead
Daniyal Ahmed	Frontend Engineer
Umer	Backend Engineer

Document Sign-Off

[Following table will contain sign-off details of document. Once the document is prepared and revised, this should be signed-off by the sign-off authority.
Any subsequent changes in the document after the first sign-off should again get a formal sign-off by the authorities.]

Version	Sign-off Authority	Project Role	Signature	Sign-off Date

[Not Applicable]

Document Information

Category	Information
Customer	Video Hosting Sites
Project	Video Platform Analytics System
Document	Software Design Specification
Document Version	1.3
Status	Completed
Author(s)	Umer Taiyab , Talal Ali, Daniyal Ahmed
Approver(s)	Ms. Sobia Iftikhar
Issue Date	5-5-2025
Document Location	-
Distribution	Advisor Project Coordinator’s Office (through Advisor)

Definition of Terms, Acronyms and Abbreviations

[This section should provide the definitions of all terms, acronyms, and abbreviations required to interpret the terms used in the document properly]

Term	Description
ASP	Active Server Pages
DD	Design Specification

Table of Contents

1 Introduction

1.1 Purpose of Document

This document outlines the design specifications for the Video Platform Analytics System. This document is intended for the development team, who will use it as a blueprint for implementation, and the course instructor, who will evaluate the design approach. The design methodology employed for this project is **top-down design**.

1.2 Intended Audience

The primary audience for this Software Design Specification is the development team responsible for building the Video Platform Analytics System. They will use this document to understand the system's architecture, components, and functionalities, guiding their implementation efforts. A secondary audience is the course instructor, who will review this document to assess the design choices and overall approach taken for the project.

1.3 Document Convention

This document adheres to the following formatting conventions:

- The main body text will be in **Arial**, with a font size of **12 points**.
- Main headings (e.g., 1 Introduction) will be in **Arial**, with a font size of **16 points**.
- Subheadings (e.g., 1.1 Purpose of Document) will be in **Arial**, with a font size of **14 points**.

1.4 Project Overview

The Video Platform Analytics System will enable authenticated users (video content creators) to log in and access a comprehensive view of their channel's and individual videos' analytical data. This data will be presented through a combination of plain text displays for key metrics and graphical representations to illustrate performance trends over time. The system will feature a layered architecture, separating concerns such as user interface, application logic, and data access (although the data will be local for this project).

1.5 Scope

The Video Platform Analytics System will include the following features:

- **Login/Signup:** Functionality for users to create accounts and securely log in to the system.
- **Channel Dashboard:** A central page displaying overall channel statistics, including total views, subscriber count, and revenue.
- **Content Page:** A section listing all the user's videos, with options to sort them based on various criteria.
- **Video Statistics:** Detailed metrics for individual videos, including view counts and potentially other relevant data.

- **Graphical Views:** Visual representations (graphs) of daily statistics for both the overall channel and individual videos.

The system will **not** include the following features:

- **Connection to an online database:** All data will be stored and accessed locally within the application.
- **Cross-platform compatibility:** The application will be designed and built exclusively for the Windows operating system.
- **Online Authentication:** There will be no integration of online authentication, such as google authentication.

2 Design Considerations

2.1 Assumptions and Dependencies

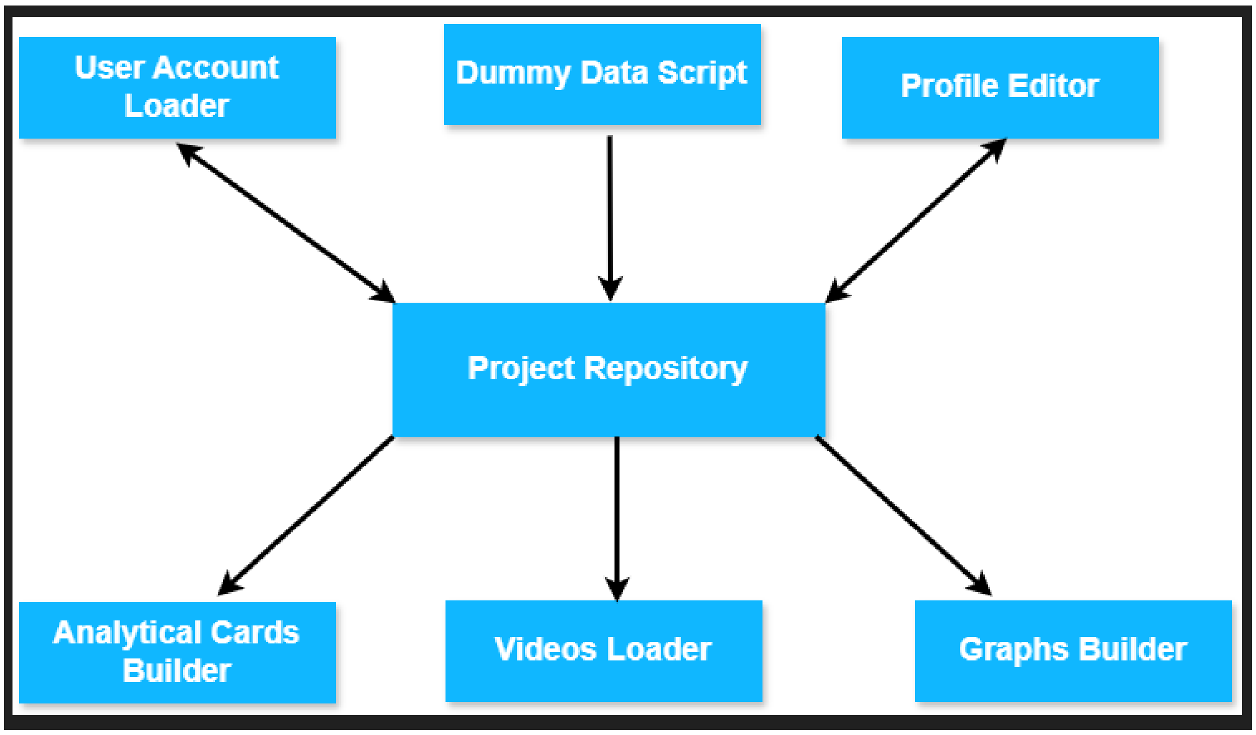
- **Local Data Storage (SQLite):** The design assumes that all analytical data for channels and videos will be stored and accessed using a local SQLite database. The database schema and data structures will need to be carefully designed to efficiently store and retrieve the required metrics.
- **Flutter Framework:** The system's user interface and potentially some of the application logic will be developed using the Flutter framework. The design will need to adhere to Flutter's architectural patterns and best practices.
- **SQLite Integration with Flutter:** A dependency exists on the successful integration of the SQLite database with the Flutter application. The design will need to incorporate appropriate Flutter packages or plugins for database interaction.
- **Pre-existing User Accounts:** It is assumed that the system will handle the creation and authentication of user accounts internally within the SQLite database. Integration with external authentication providers is not within the scope.
- **Basic Graphing within Flutter:** We assume the availability and integration of a suitable charting library within the Flutter ecosystem that can generate line graphs based on the daily statistics retrieved from the SQLite database. The selection and implementation of this library will be a design task.

2.2 Risks and Volatile Areas

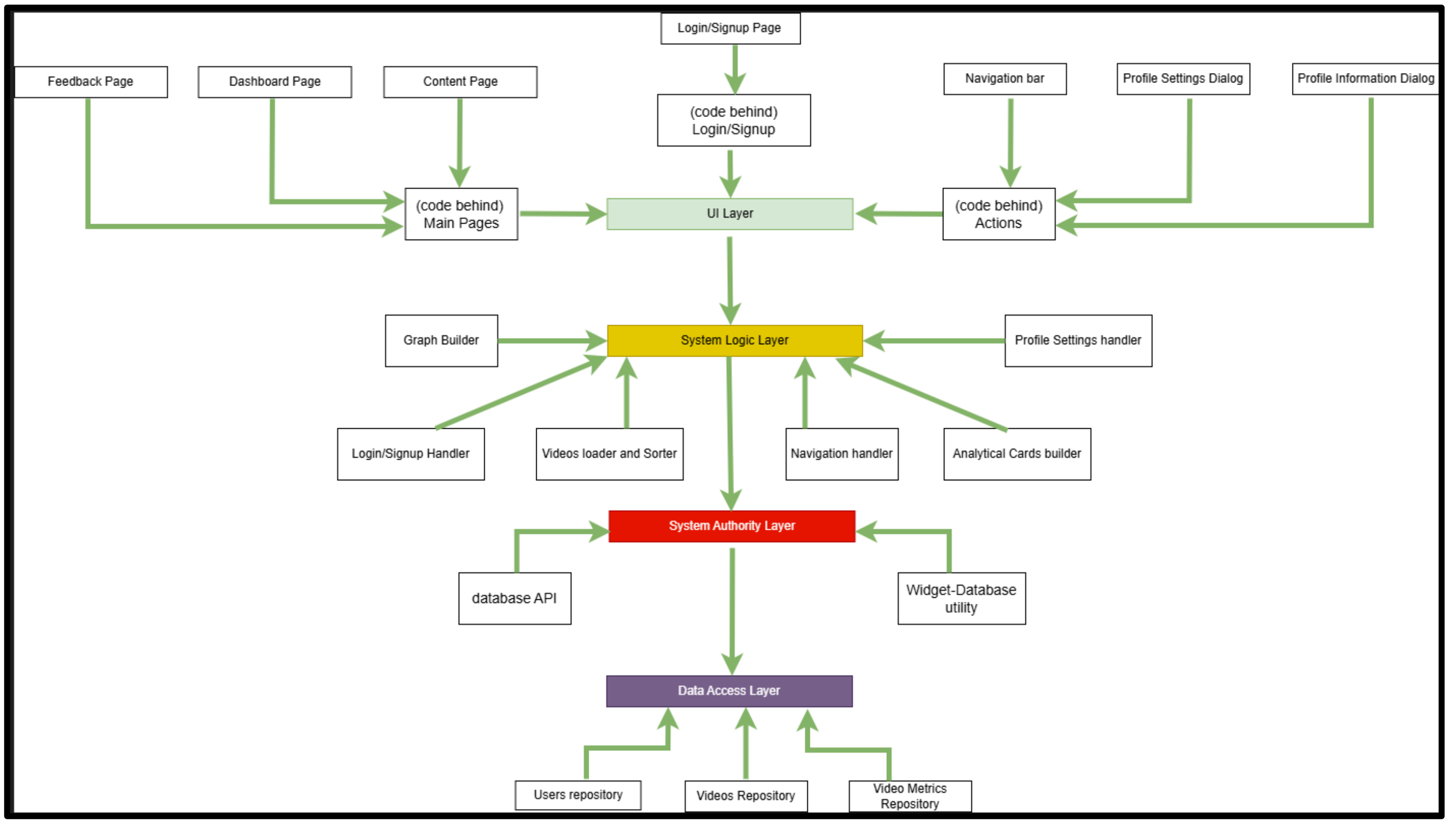
- **Performance with Large Datasets (SQLite):** A significant technical challenge is the potential for performance degradation when querying and processing large amounts of analytical data stored in the local SQLite database, especially when generating graphs based on historical daily data. The design will need to consider efficient database querying strategies, indexing, and potentially data aggregation techniques to ensure acceptable performance. The way Flutter interacts with SQLite for large datasets will also be a critical design consideration.
- **Complexity of Graphing Requirements (Flutter):** If the requirements for the graphs become more complex within the Flutter environment (e.g., requiring multiple data series, interactive elements, or specific customization options), the initially chosen Flutter charting library might need to be re-evaluated, potentially leading to design modifications and increased development effort. We will initially aim for simple, clear line graphs provided by a readily available and well-supported Flutter package to mitigate this risk.
- **User Interface Responsiveness (Flutter/SQLite):** Ensuring a responsive user interface in Flutter while querying and displaying data from the local SQLite database, especially for potentially large datasets and complex graphs, is a potential risk. The design will need to consider asynchronous operations and efficient data handling to prevent UI freezes and maintain a smooth user experience.

3 System Architecture

3.1 System Level Architecture



3.2 Software Architecture



4 Design Strategy

The Video Platform Analytics System employs several key design strategies to ensure a robust, maintainable, and scalable application:

Architectural Patterns:

1. **Model-View-Controller (MVC) Pattern**: We will implement the MVC pattern to separate the application's concerns:
    - **Model**: Data layer handling database operations with SQLite for storing user information, video statistics, and analytics data
    - **View**: Flutter UI components displaying analytics dashboards, graphs, and user interfaces
    - **Controller**: Business logic layer managing data flow between the model and view
2. **Repository Pattern**: For database access, implementing repositories to abstract data sources and provide clean APIs for the application layers to consume analytics data.

Future System Extension Considerations:

1. **Modular Design**: The system will be organized into distinct modules (authentication, dashboard, analytics, content management) to allow independent development and easier future enhancements.
2. **API Abstraction**: Although currently using local SQLite storage, the data access layer will be designed with interfaces that could accommodate remote API integration in future versions.

User Interface Strategy:

1. **Material Design Principles**: Following Flutter's Material Design guidelines to ensure a consistent and intuitive user experience
2. **Responsive Layout**: Implementing responsive design patterns to accommodate various screen sizes within the Windows environment.
3. **Progressive Disclosure**: Complex analytics information will be presented using a progressive disclosure approach, showing summary data first with options to drill down for detailed information.

Concurrency Strategy:

1. **Asynchronous Operations**: Using Flutter's asynchronous programming model to ensure UI responsiveness when performing database operations.

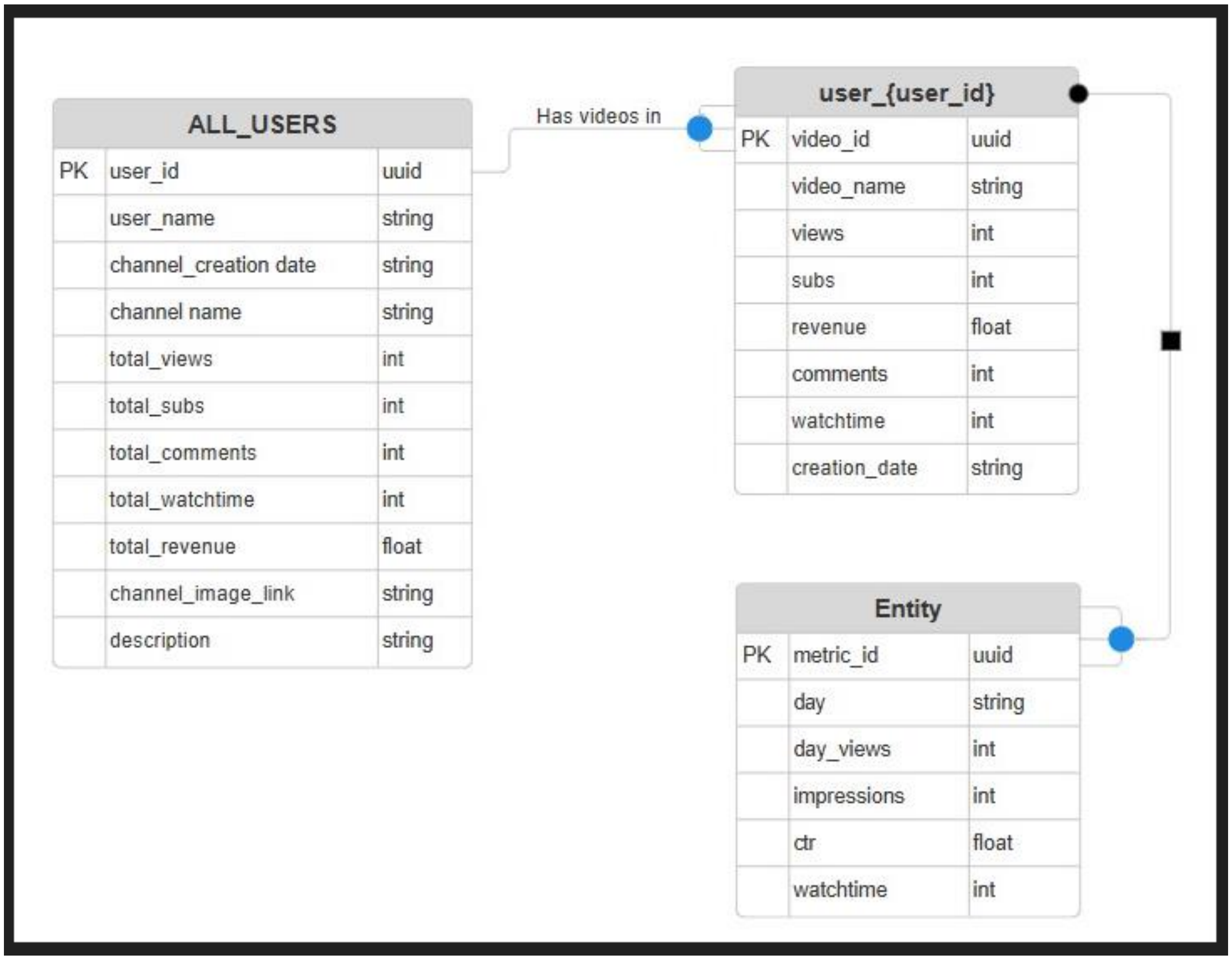
Trade-offs and Decisions:

1. **Local Storage vs. Cloud Backend**: We've chosen local SQLite storage for simplicity and to meet project constraints, accepting limitations in data sharing and real-time updates.
2. **Flutter vs. Native Windows Development**: We've selected Flutter for cross-platform potential despite some performance overhead, providing a foundation for possible future expansion to other platforms.

5 Detailed System Design

5.1 Database Design

5.1.1 ER Diagram



ALL_USERS

PK	user_id	uuid
	user_name	string
	channel_creation date	string
	channel name	string
	total_views	int
	total_subs	int
	total_comments	int
	total_watchtime	int
	total_revenue	float
	channel_image_link	string
	description	string

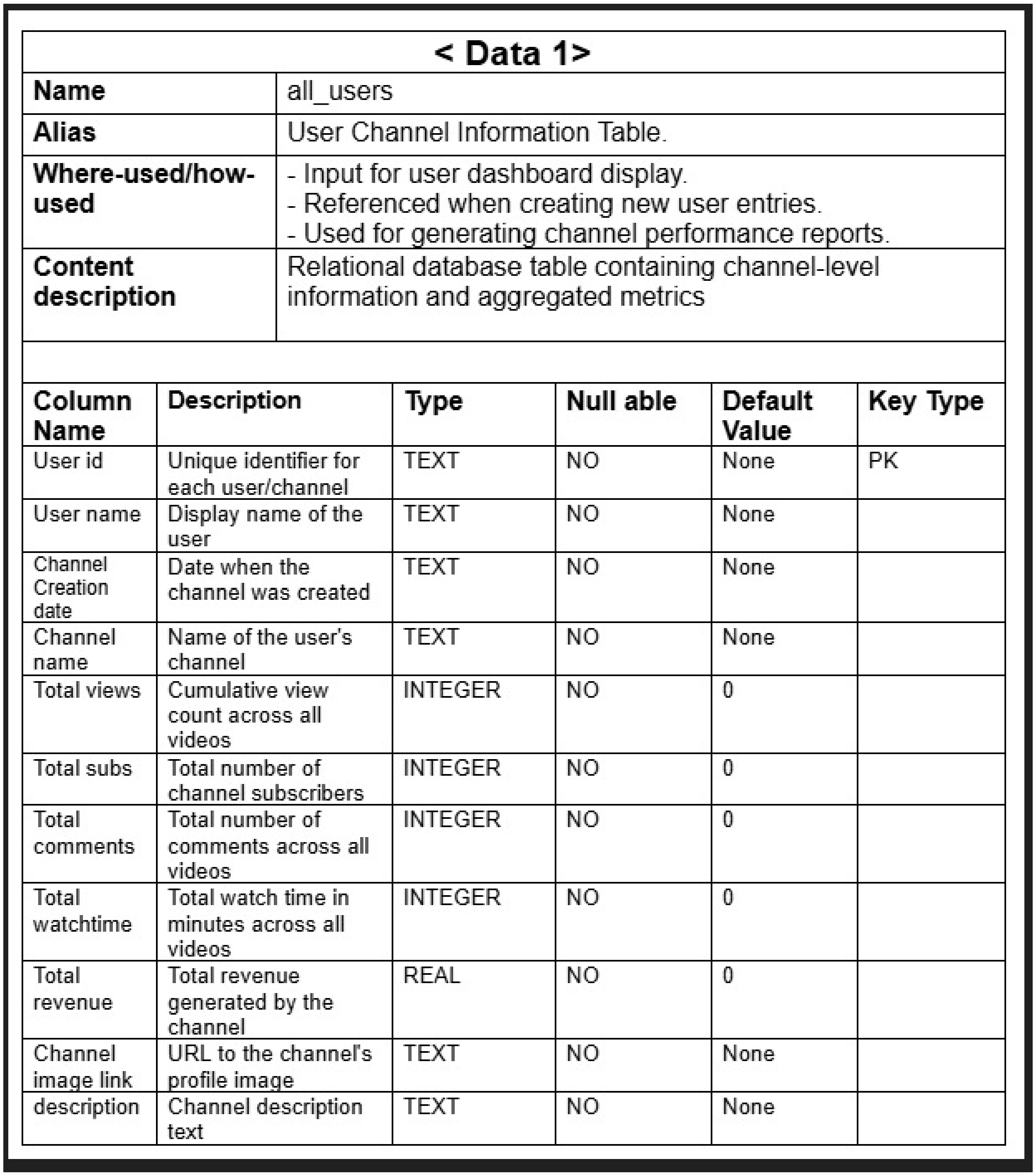
user_{user_id}

PK	video_id	uuid
	video_name	string
	views	int
	subs	int
	revenue	float
	comments	int
	watchtime	int
	creation_date	string

Entity

PK	metric_id	uuid
	day	string
	day_views	int
	impressions	int
	ctr	float
	watchtime	int

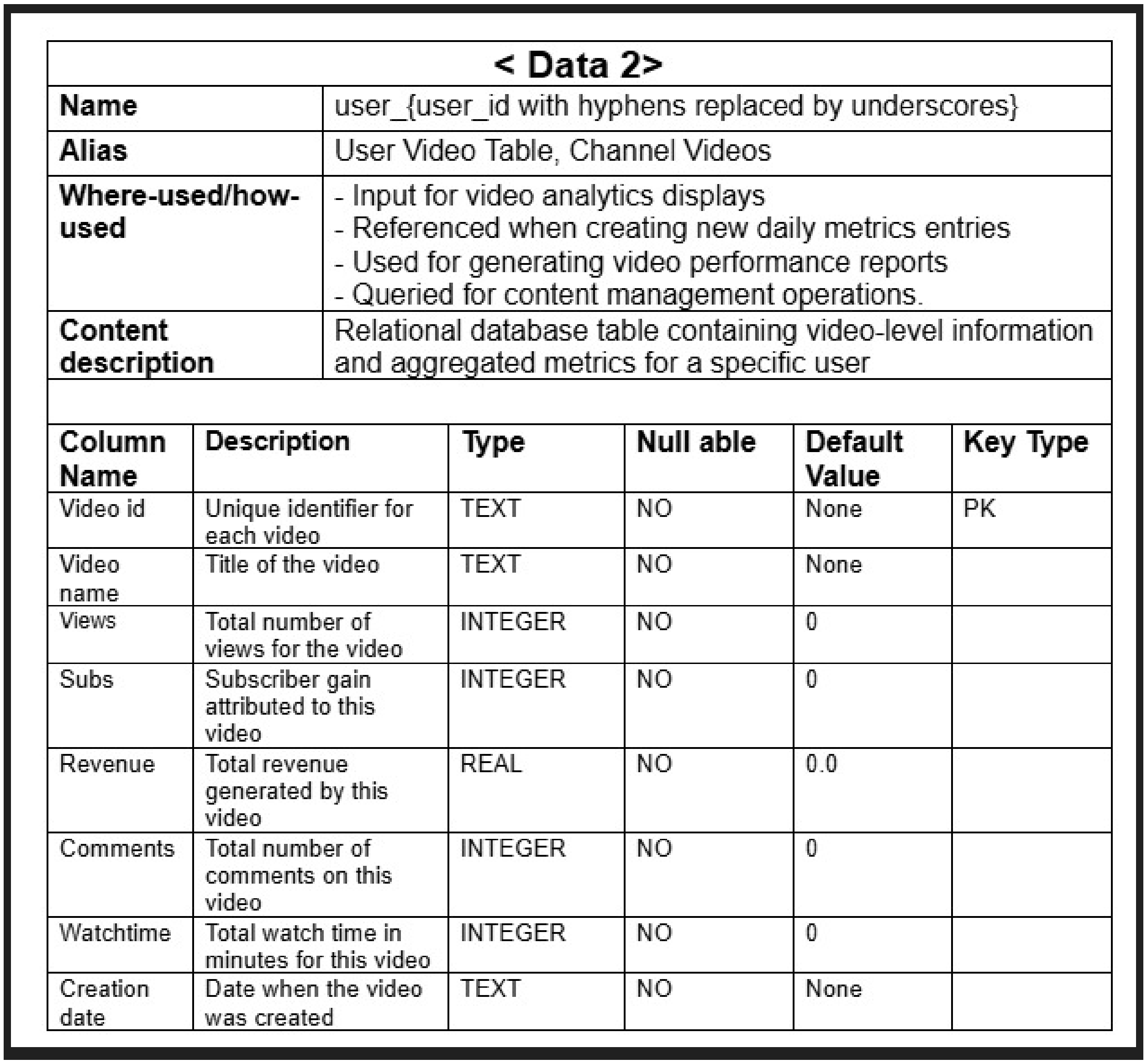
5.1.2 Data Dictionary



< Data 1>

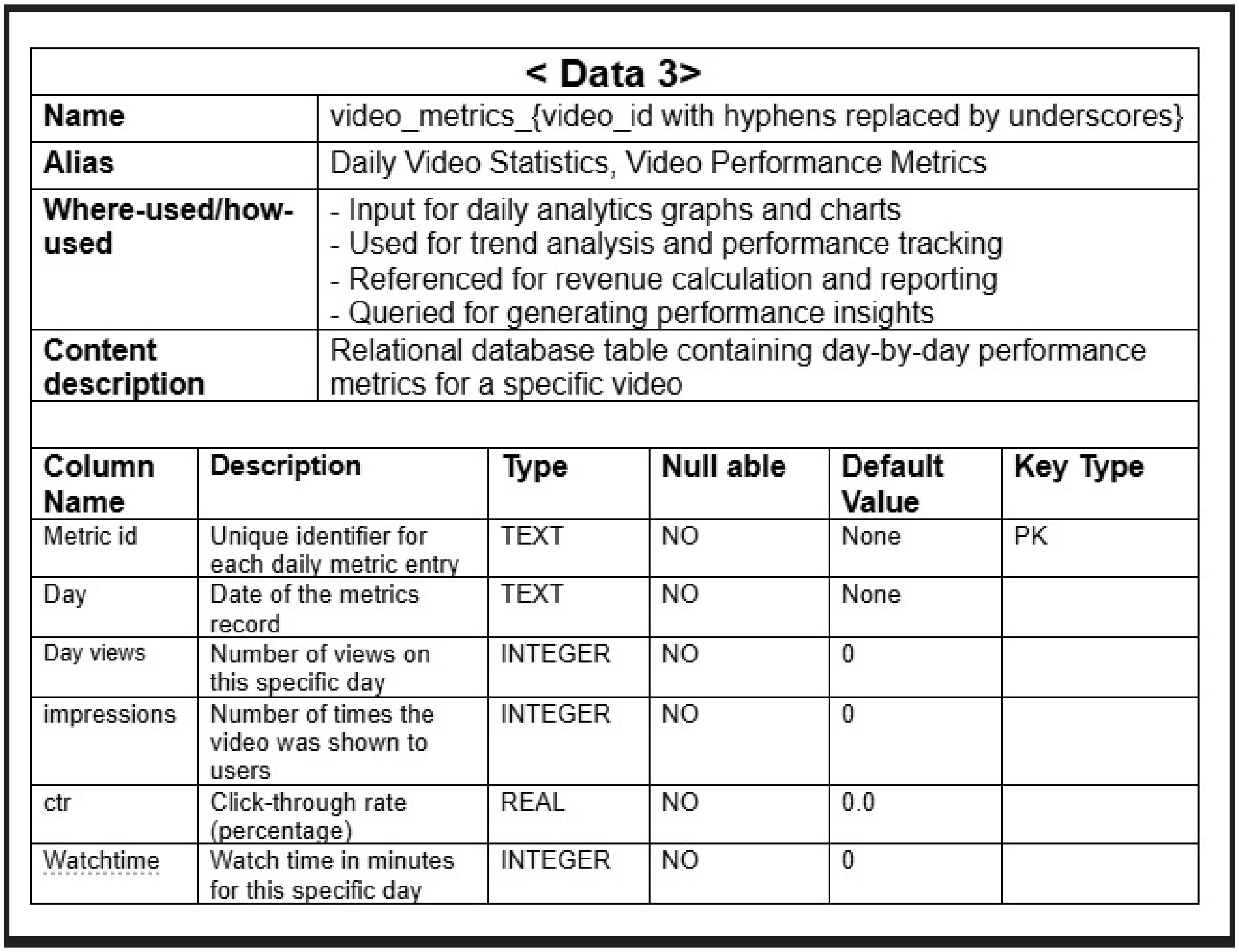
Name	all_users
Alias	User Channel Information Table.
Where-used/how-used	- Input for user dashboard display. - Referenced when creating new user entries. - Used for generating channel performance reports.
Content description	Relational database table containing channel-level information and aggregated metrics

Column Name	Description	Type	Null able	Default Value	Key Type
User id	Unique identifier for each user/channel	TEXT	NO	None	PK
User name	Display name of the user	TEXT	NO	None	
Channel Creation date	Date when the channel was created	TEXT	NO	None	
Channel name	Name of the user's channel	TEXT	NO	None	
Total views	Cumulative view count across all videos	INTEGER	NO	0	
Total subs	Total number of channel subscribers	INTEGER	NO	0	
Total comments	Total number of comments across all videos	INTEGER	NO	0	
Total watchtime	Total watch time in minutes across all videos	INTEGER	NO	0	
Total revenue	Total revenue generated by the channel	REAL	NO	0	
Channel image link	URL to the channel's profile image	TEXT	NO	None	
description	Channel description text	TEXT	NO	None	



< Data 2>	
Name	user_{user_id with hyphens replaced by underscores}
Alias	User Video Table, Channel Videos
Where-used/how-used	- Input for video analytics displays - Referenced when creating new daily metrics entries - Used for generating video performance reports - Queried for content management operations.
Content description	Relational database table containing video-level information and aggregated metrics for a specific user

Column Name	**Description**	**Type**	**Null able**	**Default Value**	**Key Type**
Video id	Unique identifier for each video	TEXT	NO	None	PK
Video name	Title of the video	TEXT	NO	None	
Views	Total number of views for the video	INTEGER	NO	0	
Subs	Subscriber gain attributed to this video	INTEGER	NO	0	
Revenue	Total revenue generated by this video	REAL	NO	0.0	
Comments	Total number of comments on this video	INTEGER	NO	0	
Watchtime	Total watch time in minutes for this video	INTEGER	NO	0	
Creation date	Date when the video was created	TEXT	NO	None	



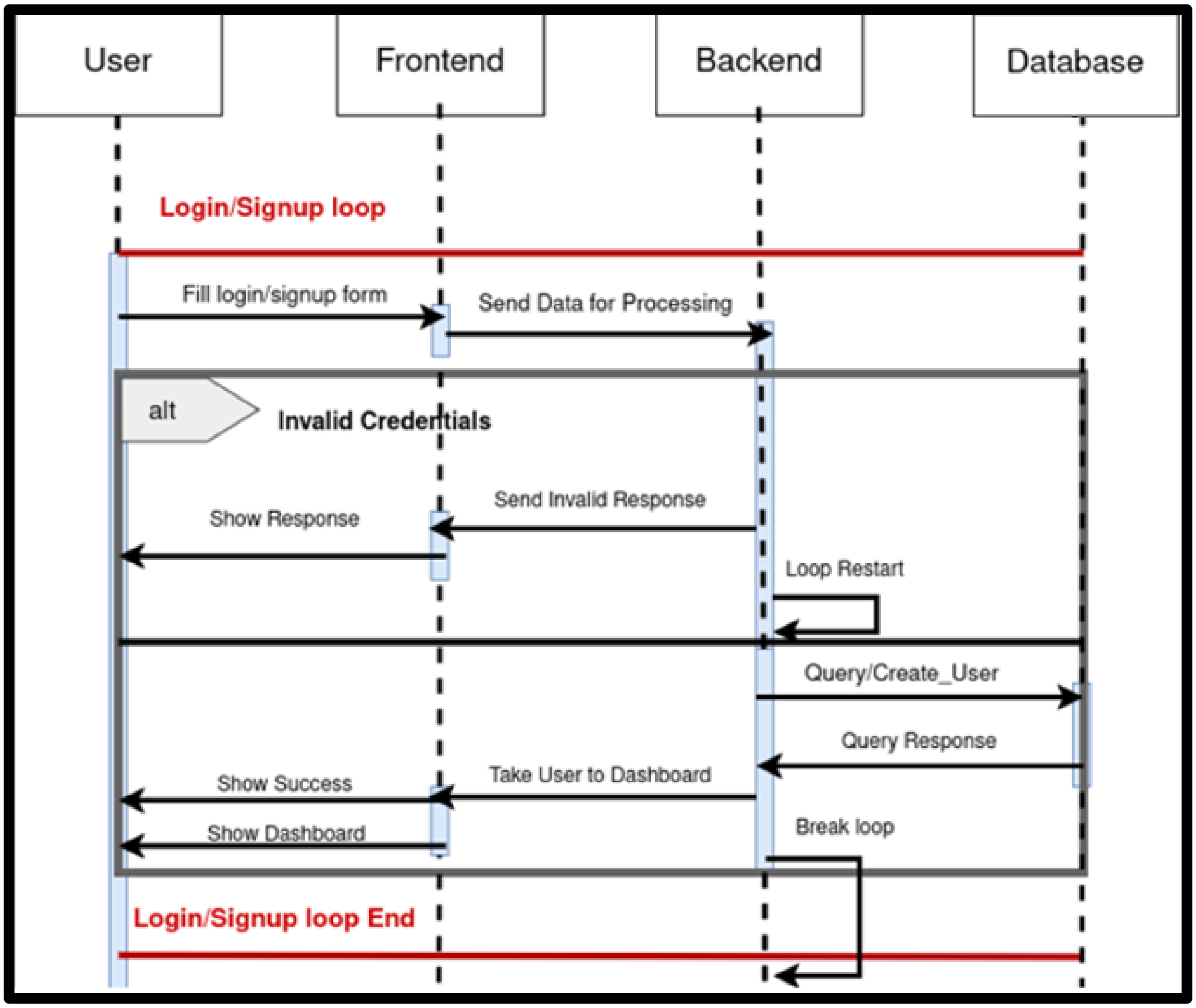
< Data 3>	
Name	video_metrics_{video_id with hyphens replaced by underscores}
Alias	Daily Video Statistics, Video Performance Metrics
Where-used/how-used	- Input for daily analytics graphs and charts - Used for trend analysis and performance tracking - Referenced for revenue calculation and reporting - Queried for generating performance insights
Content description	Relational database table containing day-by-day performance metrics for a specific video

Column Name	**Description**	**Type**	**Null able**	**Default Value**	**Key Type**
Metric id	Unique identifier for each daily metric entry	TEXT	NO	None	PK
Day	Date of the metrics record	TEXT	NO	None	
Day views	Number of views on this specific day	INTEGER	NO	0	
impressions	Number of times the video was shown to users	INTEGER	NO	0	
ctr	Click-through rate (percentage)	REAL	NO	0.0	
Watchtime	Watch time in minutes for this specific day	INTEGER	NO	0	

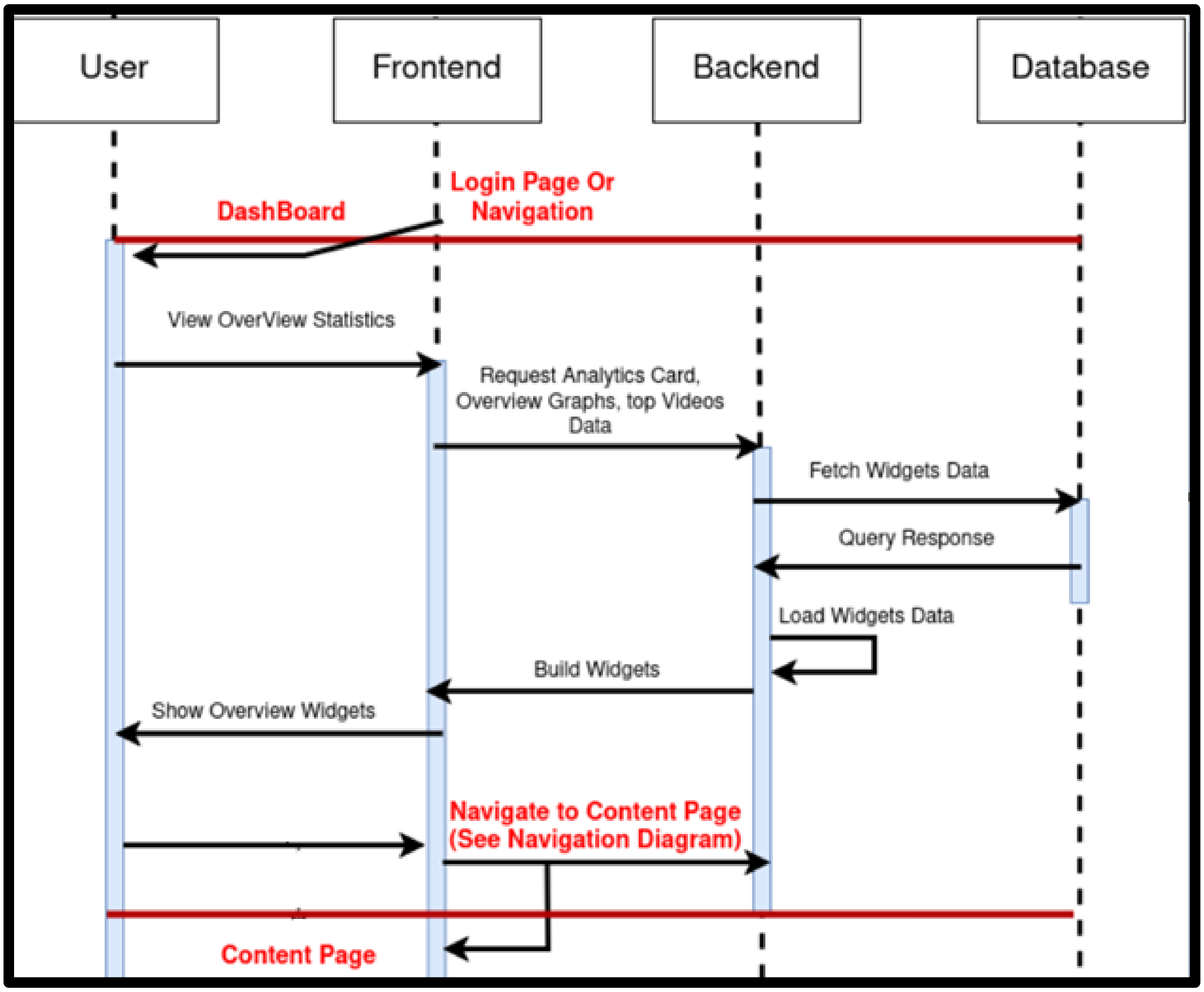
5.2 *Application Design*

5.2.1 Sequence Diagram

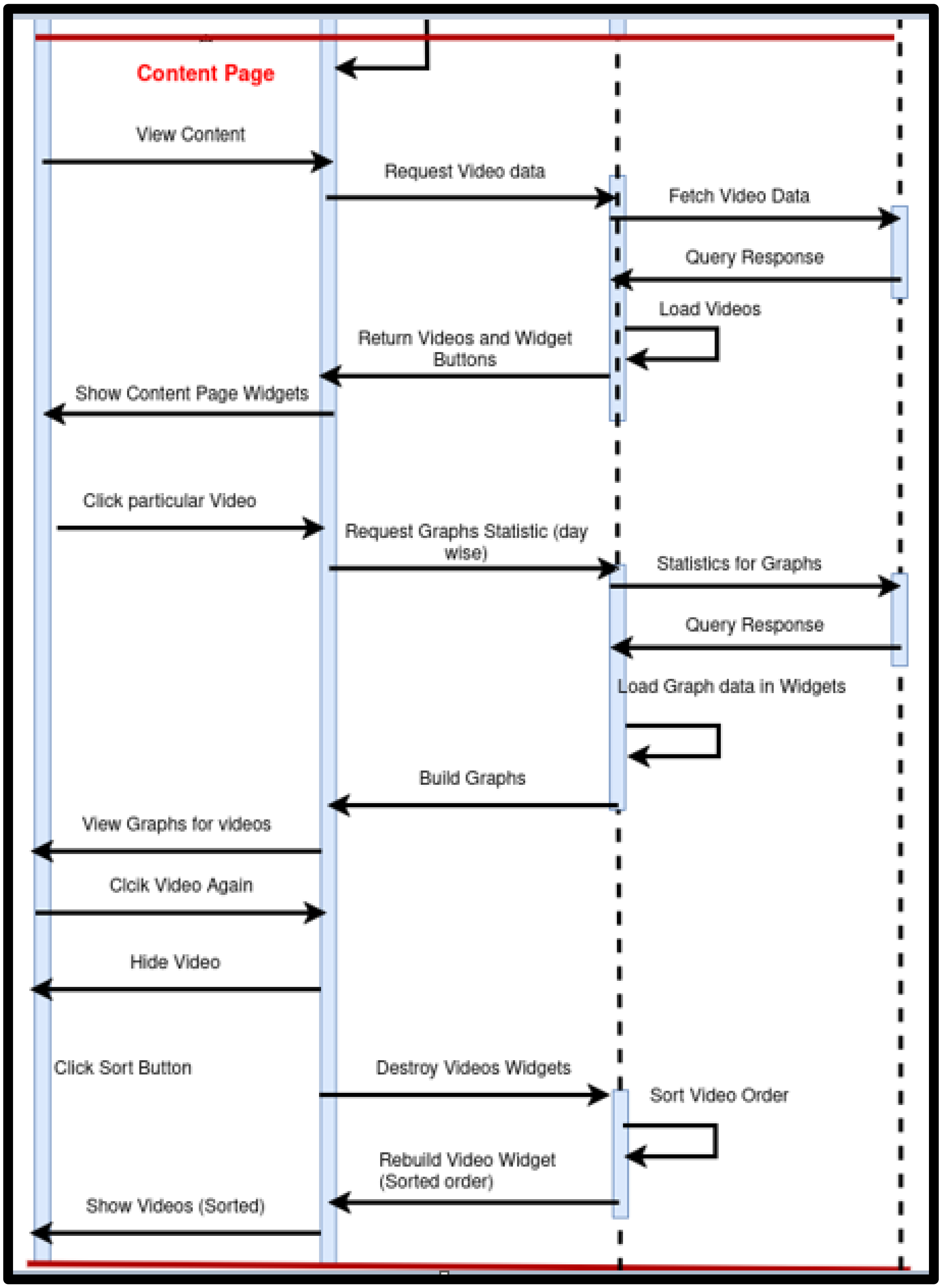
5.2.1.1 Login/Signup Sequence Diagram



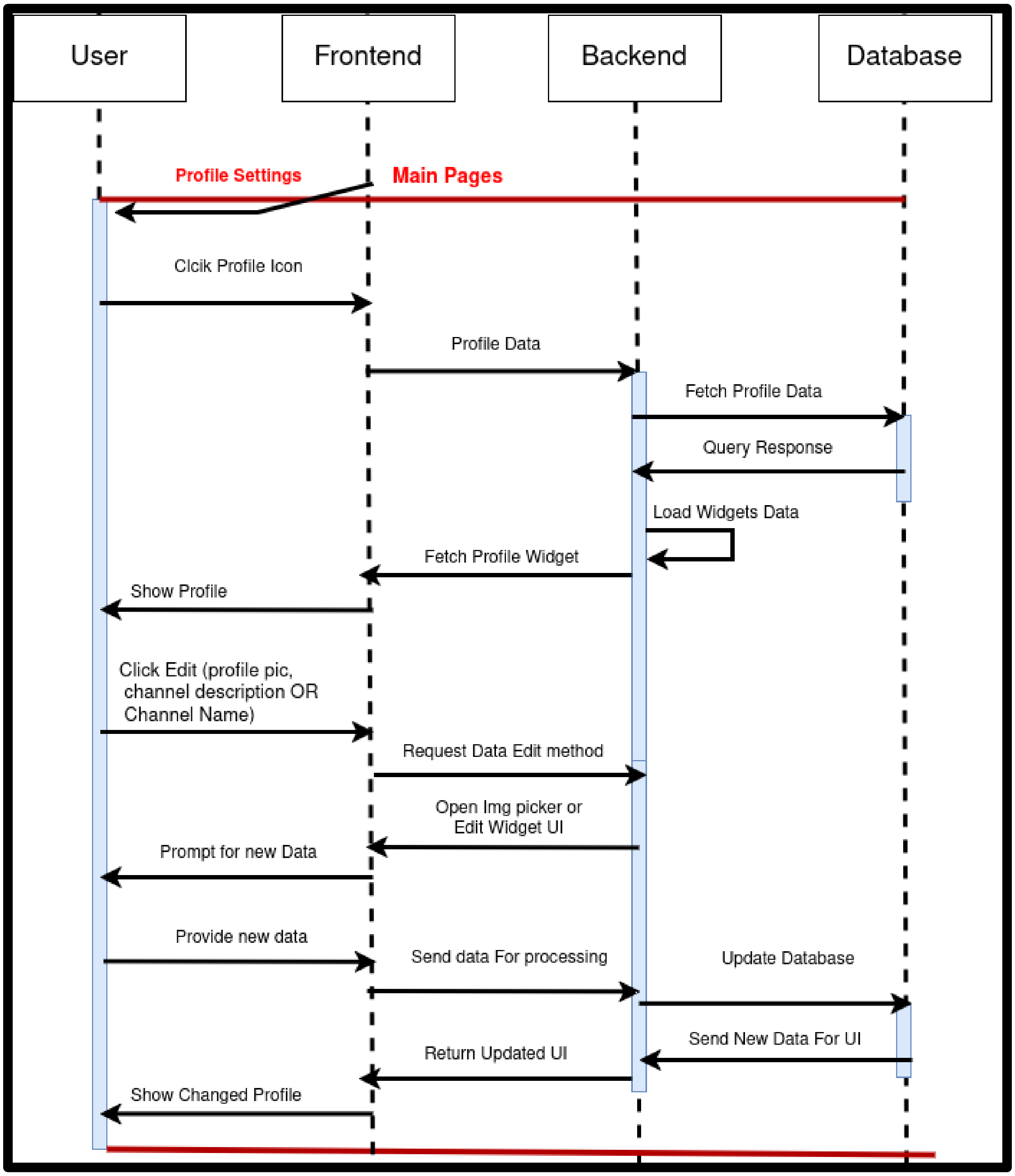
5.2.1.2 Dashboard Sequence Diagram



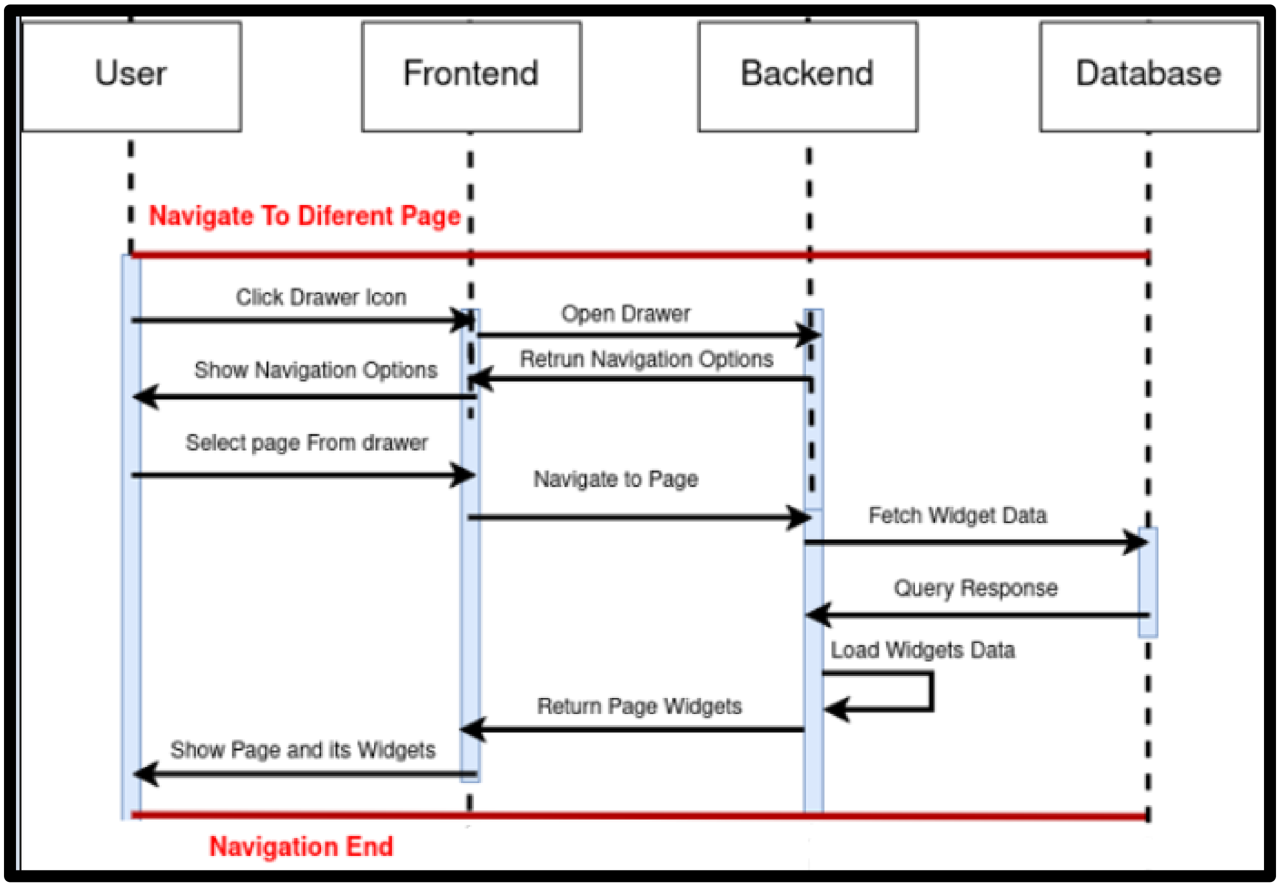
5.2.1.3 Content Page Sequence Diagram



5.2.1.4 Change Profile Sequence Diagram

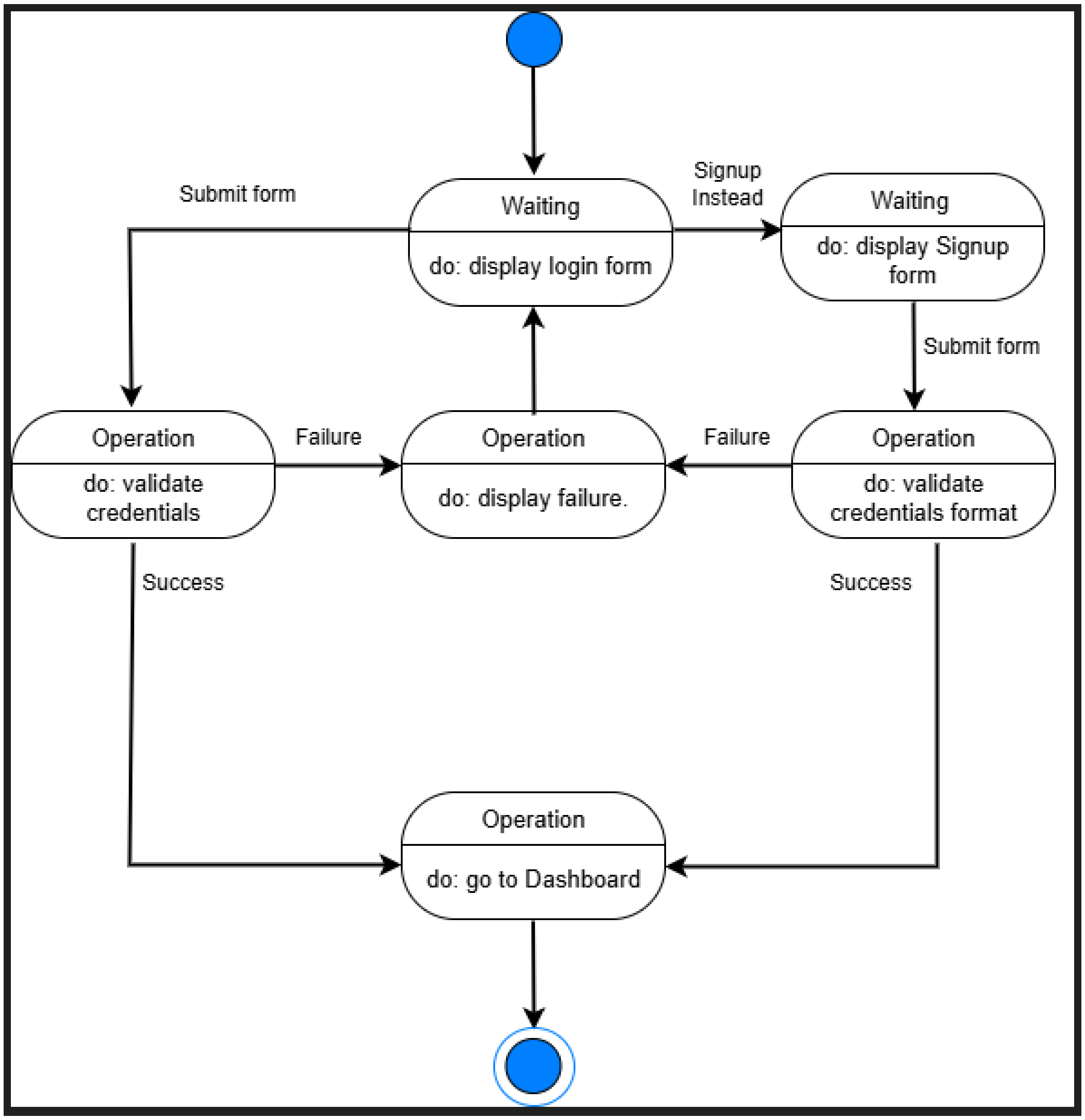


5.2.1.5 Navigation Drawer Sequence Diagram

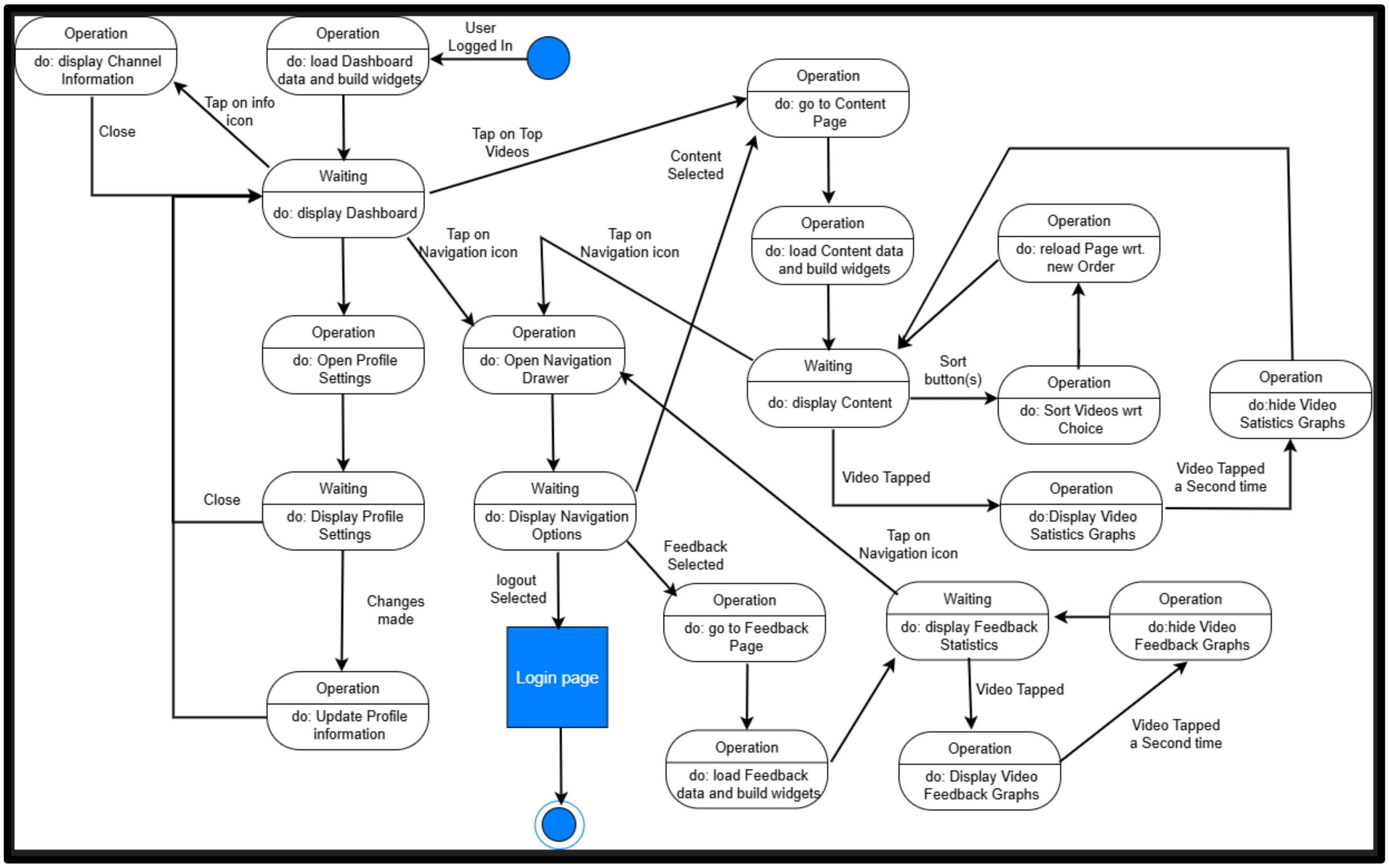


5.2.2 State Diagram

5.2.2.1 Login/Signup Page State Diagram



5.2.2.2 Main State Diagram


Operation
do: display Channel Information
Operation
do: load Dashboard data and build widgets
User Logged In
Operation
do: go to Content Page
Tap on info icon
Close
Tap on Top Videos
Content Selected
Waiting
do: display Dashboard
Tap on Navigation icon
Tap on Navigation icon
Operation
do: load Content data and build widgets
Operation
do: reload Page wrt. new Order
Operation
do: Open Profile Settings
Operation
do: Open Navigation Drawer
Waiting
do: display Content
Sort button(s)
Operation
do: Sort Videos wrt Choice
Operation
do:hide Video Satistics Graphs
Video Tapped
Video Tapped a Second time
Close
Waiting
do: Display Profile Settings
Waiting
do: Display Navigation Options
Operation
do:Display Video Satistics Graphs
Feedback Selected
Tap on Navigation icon
logout Selected
Changes made
Operation
do: go to Feedback Page
Waiting
do: display Feedback Statistics
Operation
do:hide Video Feedback Graphs
Login page
Video Tapped
Operation
do: Update Profile information
Video Tapped a Second time
Operation
do: load Feedback data and build widgets
Operation
do: Display Video Feedback Graphs

6 References

1. **Flutter Documentation:**
   - Flutter SDK Documentation (2025). https://docs.flutter.dev/
   - Flutter State Management Guide (2024). https://docs.flutter.dev/data-and-backend/state-mgmt/intro
2. **SQLite Documentation:**
   - SQLite Documentation (2025). https://www.sqlite.org/docs.html
3. **Flutter-SQLite Integration:**
   - sqflite Package Documentation (2025). https://pub.dev/packages/sqflite
4. **UI/UX Design Resources:**
   - Flutter Material Components (2025). https://docs.flutter.dev/ui/widgets/material
5. **Similar Applications and Systems:**
   - YouTube Studio Analytics Interface (2024)
   - TikTok Creator Analytics Dashboard (2024)

7 Appendices

[Not Applicable]