itids

In-Train Display Information System

Project referred to:

TRANSLINK

Mykhailo Halushko

Kwantlen Polytechnic University

February 10, 2025

Document Version: 0.1.0

Project version: 0.1.0

Introduction

Project Overview

The **In-Train Information Display System** (ITIDS) is a real-time transit information system software designed to enhance passenger experience within **Metro Vancouver's SkyTrain network**, operated by **TransLink**. The system provides essential travel details such as **upcoming station names, estimated arrival times, transfer connections, and route status updates** through a digital display inside the train.

SkyTrain, one of the longest **fully automated rapid transit systems** in North America, operates across multiple lines, serving thousands of commuters daily. ITIDS aims to **modernize in-train communication** by offering clear, accurate, and timely transit information, reducing passenger confusion and improving accessibility.

Purpose and Objectives

The primary goal of ITIDS is to streamline **passenger information delivery** by integrating with the existing SkyTrain guideway system and **providing real-time updates**. The system is designed to:

- **Enhance passenger experience** by displaying upcoming station details and real-time travel updates.
- **Support multi-modal connectivity**, helping passengers identify nearby bus, SeaBus, and West Coast Express transfers.
- **Improve accessibility** by presenting clear, multilingual, and visually accessible information.
- **Reduce reliance on audio announcements**, offering a complementary visual navigation aid.
- **Optimize system adaptability**, ensuring easy integration with future SkyTrain expansions.

Scope of the Project

ITIDS is developed as a modular and scalable system for easy implementation across SkyTrain's Expo, Millennium, and Canada Lines. The system consists of:

- **Database Management** – Storing station details, train routes, and connection points.
- **User Interface Design** – A digital display with real-time dynamic updates.
- **Backend Processing** – Handling data from guideways and station connections to keep information accurate.
- **Adaptive Functionality** – Supporting different train models and configurations within the SkyTrain network.

Target Audience

This documentation is intended for:

- **Developers and Engineers** – To understand system architecture, database design, and integration methods.
- **TransLink and SkyTrain Operations** – To evaluate feasibility and potential deployment within existing infrastructure.
- **Passengers and Transit Users** – To benefit from enhanced real-time navigation and improved in-train communication.

Challenges Addressed by ITIDS

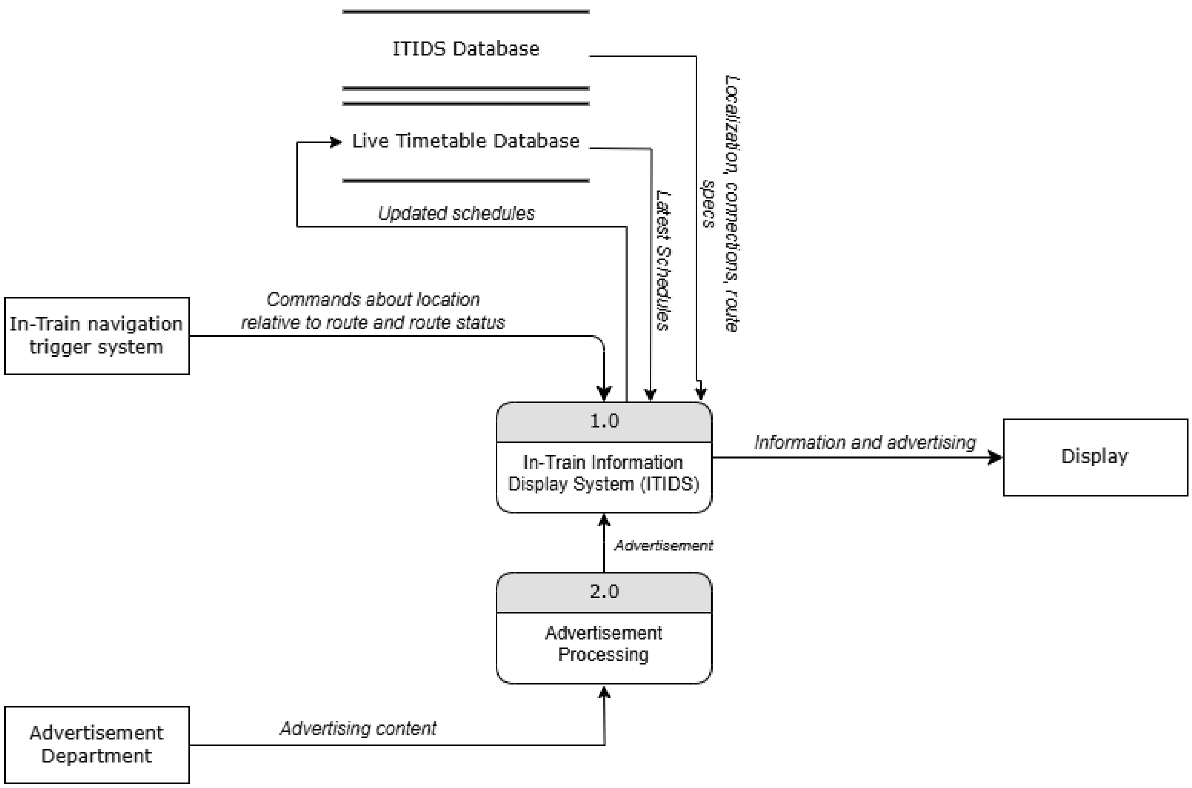
Challenges in the Current System:

- **Limited Visual Information** – SkyTrain currently relies on **audio announcements**, which may be unclear due to **background noise** or **language barriers**.
- **No Real-Time Transfer Updates** – Passengers do not always receive **live updates** about connecting **buses, SeaBus, or delays**.
- **Inconsistent Navigation** – Different **train models and lines** have varying levels of onboard passenger information.
- **No Live Data Integration** – The system does not currently **adapt to real-time delays or operational changes**.

System Architecture

Context Data Flow Diagram

This diagram **outlines the ITIDS system at a macro level**, showing how different components interact **without detailing internal processes.**



Component	**Description**
ITIDS Database & Live Timetable Database	Stores all **station, route, connections, and schedule data**.
In-Train Navigation Trigger System	Detects train movement and triggers updates.
ITIDS (Main System Process 1.0)	The core processing unit that gathers and manages real-time station updates.
Advertisement Processing (Process 2.0)	Manages and displays advertisements inside the train.
Display (Sink)	The final presentation layer for passengers.

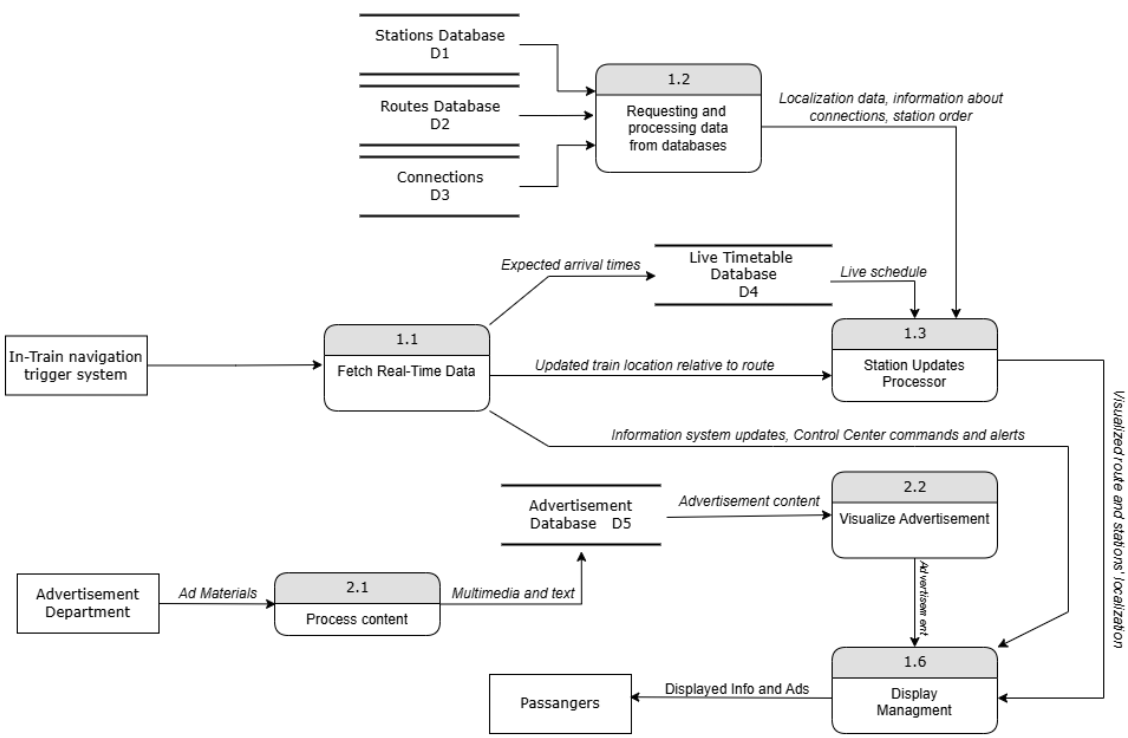
Level-1 Data Flow Diagram

This DFD Level 1 expands on the ITIDS system, showing specific sub-processes that handle:

- **Data retrieval from databases**
- **Real-time station updates**
- **Advertisement management**
- **Information processing and display**

Flow description:

- The **In-Train Navigation Trigger System** provides **real-time location data** → ITIDS fetches relevant **station, route, and timetable details** from the databases.
- The **processed station updates** are sent to **Display Management (1.6)** for visualization inside the train.
- The **Advertisement Processing System (2.0)** retrieves **advertisement content** and integrates it into the display.
- **Passengers** see updated **station details and advertisements** on the **train's display system**.



Key Components & Their Functions

Component	Description
Stations Database (D1)	Stores **station details**, such as names, zones, and exit directions.
Routes Database (D2)	Stores **train route information**, including station order and direction.
Connections Database (D3)	Stores **connections** with IDs, logos, type, etc.
Live Timetable Database (D4)	Maintains **real-time train schedules and delays**.
Advertisement Database (D5)	Stores **advertising content for display inside the train**.

Major Processes & Data Flow

ITIDS Core Processes (System 1.0)

Process	Function
1.1 Fetch Real-Time Data	Collects train position updates from the **In-Train Navigation Trigger System*.**
1.2 Request & Process Data	Fetches relevant station and timetable data from **D1, D2, and D3 databases**.
1.3 Station Updates Processor	Analyzes train location and updates **next station information** in real time.
1.6 Display Management	Sends processed data to the **Display** for passengers.

Advertisement Processing (System 2.0)

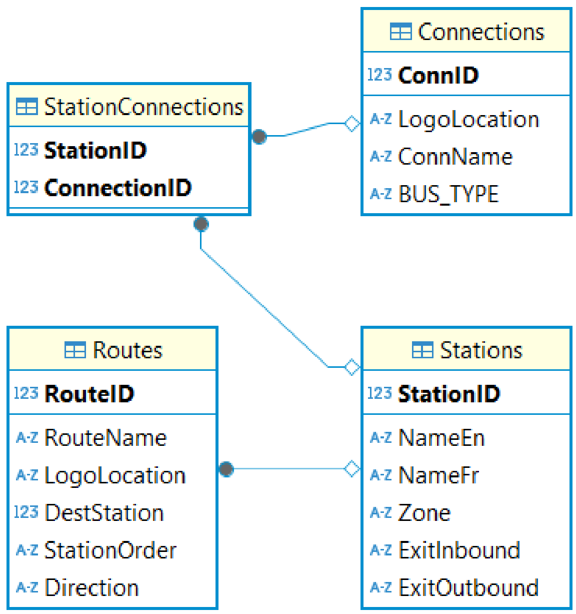
Process	Function
2.1 Process Content	Retrieves ad content from **Advertisement Database (D5)** and formats it.
2.2 Visualize Advertisement	Sends advertisement content to the **Display**.

** - or equivalent name of existing trigger system*

Entity-Relationship Diagram

Overview

The Entity-Relationship (E-R) diagram represents the core database structure of the In-Train Information Display System (ITIDS). It defines how different entities interact to provide **real-time station, route, and connection data to passengers**. The system consists of **four primary tables:**



- **Stations** – Stores information about SkyTrain stations, including their names and exit directions.
- **Routes** – Defines routes, their destination stations, and stop order.
- **Connections** – Contains various transit connections available at stations, such as buses and other train lines.
- **StationConnections** – A junction table that maps stations to their available connections, allowing for efficient lookup of transit transfer options.

**RouteColor attribute has been added to Routes and E-R will be updated in the next version of the document*

Stations

Attribute	Description
StationID	Unique **identifier for each station.**
NameEn / NameFr	**Station name in English and French** (for multilingual support).
Zone	Fare **zone** category (e.g., 1, 2, 3).
ExitInbound / ExitOutbound	Defines which side the train doors will open for passengers, **depending on route direction** (Inbound/Outbound).

Routes

The **Routes** table defines SkyTrain routes, their stop order, and destination stations.

Attribute	**Description**
RouteID	Unique identifier for route.
RouteName	The route's official name (e.g., "Expo Line").
LogoLocation	Path to the logo image representing the route.
DestStation	The **destination station** for the route.
StationOrder	Defines the sequence of stops on the route.
Direction	Indicates whether the train is **Inbound (toward Vancouver downtown)** or **Outbound (away from Vancouver downtown)**.
RouteColor	**Hex color** of the route, used in CSS for progress bar and station markers color.

Connections

The **Connections** table stores **public transit connections** on the stations (e.g., buses, SeaBus, or other lines).

Attribute	**Description**
ConnID	Unique identifier for each transit connection.
LogoLocation	Path to the **logo** representing the connection type.
ConnName	Name of the transit service (e.g., "99 B-Line Bus").
BUS_TYPE	Used **only for bus-type** connection **for visual difference** between **Rapid Buses, Night buses and Ordinary one**. (For non-bus connections this attribute is NULL)

StationConnections (Junction Table)

Attribute	**Description**
StationID	References the **Stations** table.
ConnectionID	References the **Connections** table.

Feasibility Analysis

Technical Analysis

Technology Stack

To ensure efficiency, scalability, and ease of integration, the following technologies are used in ITIDS:

- **Python (Flask for database operations):**
  The backend is built using Python with the Flask framework to handle database interactions. Flask provides a lightweight, yet powerful way to fetch and process real-time station data. It serves as the core of the ITIDS backend, facilitating API requests and data retrieval.

- **SQLite (Database Management):**
  SQLite is used to store station names, route information, and connection details. This database system is chosen for its **lightweight nature and simplicity**, making it an ideal choice for development and testing. While SQLite is effective for small- to medium-scale applications, a more robust database such as **PostgreSQL or MySQL** may be considered when transitioning ITIDS into a commercial system.

- **HTML & CSS (Physical Design & UI Structure):**
  The frontend interface is structured using HTML and CSS. The design prioritizes **readability, accessibility, and responsiveness**, ensuring that passengers can quickly and easily interpret station updates. The physical design also accommodates **multilingual support** for inclusivity.

- **JavaScript (Animations & Real-Time Updates):**
  JavaScript is used to handle **dynamic UI updates and animations**. For instance, it enables smooth transitions when **switching between stations**, ensuring a seamless user experience. It also synchronizes **real-time station data with the train's movement**, displaying updated station details as the train progresses along its route.

System Behavior and Real-Time Functionality

The ITIDS system is **designed to automatically switch station displays based on real-time train movement**. **The process follows these steps:**

1. **Train Location Detection:**
   - ITIDS receives real-time location triggers from the In-Train Navigation System***.**
   - This system determines when the train has departed from one station and is approaching the next.

2. **Database Query Execution:**
   - Based on the location data, the backend (Flask) queries the SQLite database to fetch:
     - The **next station name.**
     - **Transfer options** (buses, SkyTrain lines, etc.).
     - **Expected arrival** options of transfer options.

3. **Frontend Display Update (JS & CSS Animations):**
   - The JavaScript frontend receives the updated station data and triggers **smooth screen transitions**.
   - CSS animations adjust the **progress bar**, visually indicating the train's movement.
   - Any **live service updates or delays** from the SkyTrain system are displayed dynamically.

4. **Passenger Interaction & Multimodal Integration:**
   - The system provides **clear transfer options** to buses, SeaBus, and West Coast Express.
   - Future implementations **may include** touchscreen interfaces for additional passenger interaction.

Scalability and Future Considerations

While the current **SQLite-based system** is suitable for early-stage development and deployment, a **more scalable database (PostgreSQL or MySQL)** will be needed when transitioning to a **full-scale commercial solution**. Additional considerations for long-term development include:

- **Integration with real-time transit APIs** to fetch **live delays and operational changes**.
- **Cloud database storage** to allow better data accessibility and synchronization across train units.
- **Potential AI-based prediction models** for **train arrival times and crowd estimation**.

** - or equivalent name of existing trigger system*

Economic Analysis

Introduction

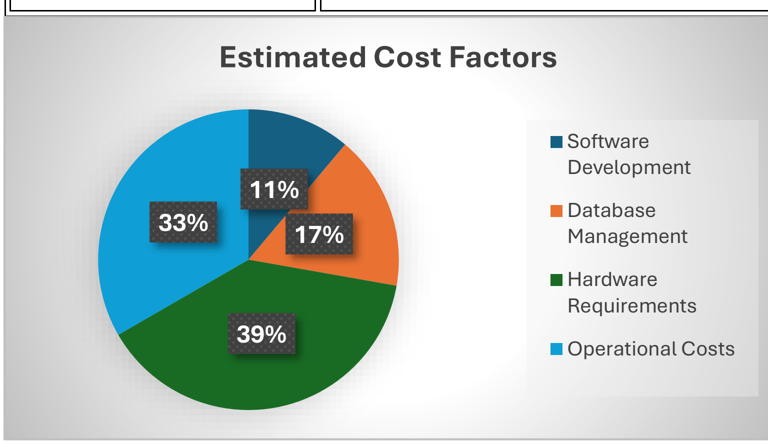
The economic feasibility of ITIDS is analyzed **from a theoretical standpoint**, as this project is currently **a non-commercial initiative**. However, in a real-world implementation, the following key cost factors would be considered:

- **Development Costs** (software, database management, UI/UX).
- **Hardware & Deployment** (if specialized displays are needed).
- **Maintenance & Updates** (keeping data accurate and up to date).

This section provides an overview of the potential costs and benefits **if the project were to be scaled into a real-world transit system**.

Potential Cost Factors

Category	**Details**	**Estimated Impact**
Software Development	Backend (Flask, SQLite), Frontend (JS, HTML, CSS)	**Low** (Free/Open source)
Database Management	SQLite (local) vs. Cloud-based (PostgreSQL)	**Medium** (If cloud-hosted)
Hardware Requirements	Train displays, onboard processing units	**High** (If new screens are needed)
Operational Costs	API integration, real-time updates, maintenance	**Medium**
Revenue Potential	Digital ads displayed within trains	**High** (if monetized)



Note: **Since ITIDS is a prototype, no actual budget is allocated.** However, these factors would influence cost **if the system were implemented commercially**.

Long-Term Viability & ROI *(If Commercialized)*

If ITIDS were to be implemented in **Metro Vancouver's SkyTrain system**, or any other system, economic feasibility would depend on:

- **Reduction in printed schedules → Saves operational costs.**
- **Advertising integration → Generates revenue for transit authorities.**
- **Passengers experience improvements → Justifies investment.**

Conclusion

Since this is a **non-commercial project**, no real budget constraints exist. However, in a real-world implementation, **the cost of hardware, software, and operational maintenance** would be the primary financial considerations. **A long-term revenue model (e.g., ads, cost savings) could offset these expenses, making ITIDS a viable transit technology investment.**

Operational Feasibility

Operational feasibility assesses whether ITIDS (In-Train Information Display System) can function smoothly within the existing **SkyTrain infrastructure** while meeting the needs of passengers and transit operators. This section evaluates **ease of integration, user adoption, maintenance requirements, and long-term sustainability**.

Ease of Integration with SkyTrain Infrastructure

- **Existing Infrastructure Limitations:**
  - SkyTrain cars **do not have built-in displays**, except for the upcoming **Mark V trains**.
  - ITIDS requires **separate screen installation** for older train models (Mark I-IV).
  - Possible implementation on **platform-level screens** if onboard displays are unavailable.
- **Technical Adaptability:**
  - ITIDS can be integrated with **SkyTrain's navigation triggers** to determine station arrival.
  - Uses **lightweight database processing** (SQLite) to minimize resource consumption *(Applies only for current version of the project)*.
  - Compatible with **potential cloud-based updates** when TransLink's real-time API is accessible.
- **Challenges & Considerations:**
  - Retrofitting older train models may require **additional hardware investment**.
  - Different train types may need **UI scaling adjustments** for screen compatibility.
- **Verdict:** Feasible but requires additional hardware installation for most SkyTrain models.

Passenger Usability & Adoption

- **User-Friendly Design:**
    - Clear, **high-contrast text** ensures readability for passengers.
    - **Multilingual** support (English and French) improves accessibility.
    - Consistent UI elements ensure predictable **navigation experience.**
- **Challenges & Considerations:**
    - **Passengers are accustomed to audio announcements;** visual integration may require an adaptation period.
    - **Syncing visual updates with existing voice announcements** to avoid confusion.
- **Verdict:** Highly feasible but requires adaptation for current SkyTrain passengers.

System Maintenance & Long-Term Sustainability

- **Software Maintenance:**
    - Uses **Flask (Python) for backend processing** with minimal maintenance requirements.
    - SQLite provides a **lightweight database solution** until scalability demands a **PostgreSQL transition**.
    - Routine updates will be needed if **real-time TransLink data integration** is implemented.
- **Hardware Maintenance:**
    - If onboard displays are installed, **periodic hardware servicing** will be required.
    - Display malfunctions could require **physical technician intervention**, increasing operational costs.
- **Challenges & Considerations:**
    - Requires **long-term support for software updates and data accuracy**.
    - Future versions may need **server-based architecture** for cloud-hosted operations.
- **Verdict:** Feasible, with low software maintenance but potential hardware upkeep costs.

Operational Workflow (How ITIDS Functions in Real-Time)

1. **Train detects movement using navigation triggers**, activating station update requests.
2. **System queries the database (SQLite/PostgreSQL in the future)** to retrieve:
    - Next station name.
    - Transfer connections (bus, SeaBus, SkyTrain lines).
    - Exit direction (left/right).
3. **JavaScript frontend updates the display**, ensuring smooth UI transitions.
4. **If real-time timetable data is available**, ITIDS can display live delay information.

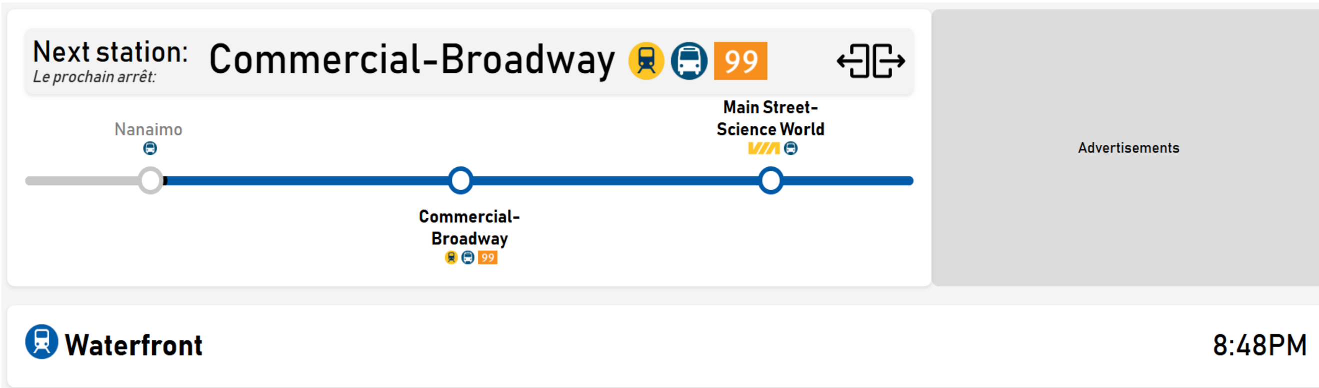
- **Verdict:** Fully functional as a local system; future enhancements can introduce **real-time updates**.

Conclusion

- ITIDS is **operationally feasible** but requires **hardware installations** for older SkyTrain models that do not have built-in displays.
- **Software integration is lightweight**, requiring minimal computing resources.
- **Passenger adaptation** is expected but manageable through **clear visual design and multilingual support**.
- **Long-term maintenance** is primarily software-based, though hardware upkeep may be required if displays are installed.
- **Final Verdict:** ITIDS can be integrated into SkyTrain operations, but its success depends on **hardware availability and long-term scalability planning**.

UI/UX Designs

Current design of the System



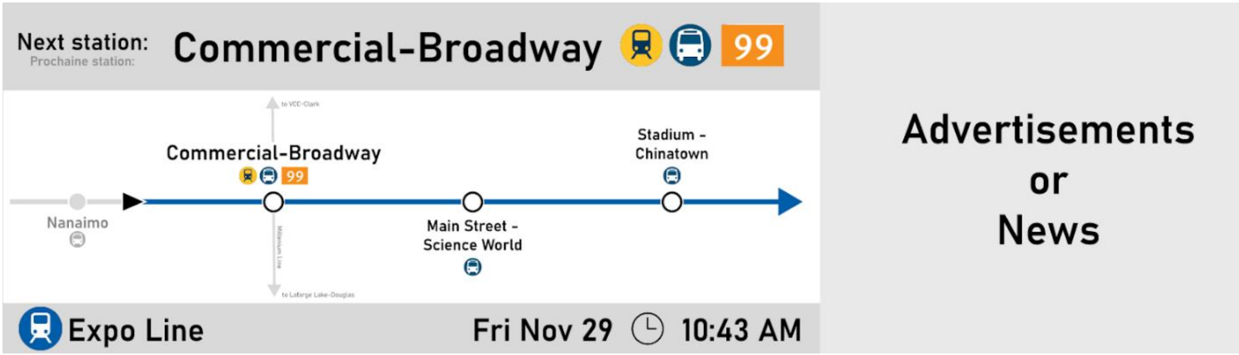
Content

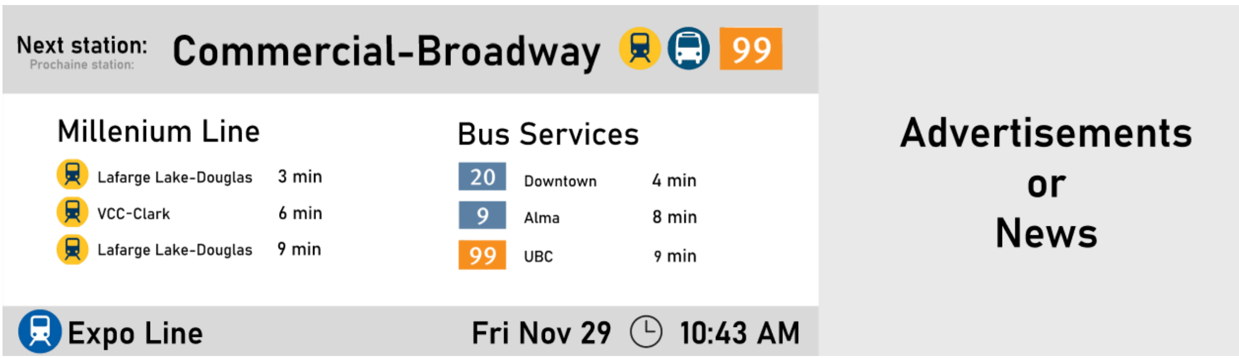
- Header
  - **Next station label**, written on English and French.
  - Next station name with **all possible connections** on it.
  - Exit directions.
- Route plan
  - Station names are placed on **checkerboarded order**, to provide **more space for long-named stations**.
  - **Connections** under each of the name for navigation purposes.
  - The progress bar has a **train marker for accessible progress tracking**.
  - The route plan contains **3 stations**: previous, next, and next-to-next for **easier navigation** if the passenger wants to find them on the transit map.
- Footer
  - Consist of **route logo and destination** on the left, and **time** on the right.
- Aside
  - Advertisement window.
- Colours:
  - Colour palette chosed for this **design follows all accesibility** regulations.
  - Each **route** has its **own colour** for progress bar and station markers.

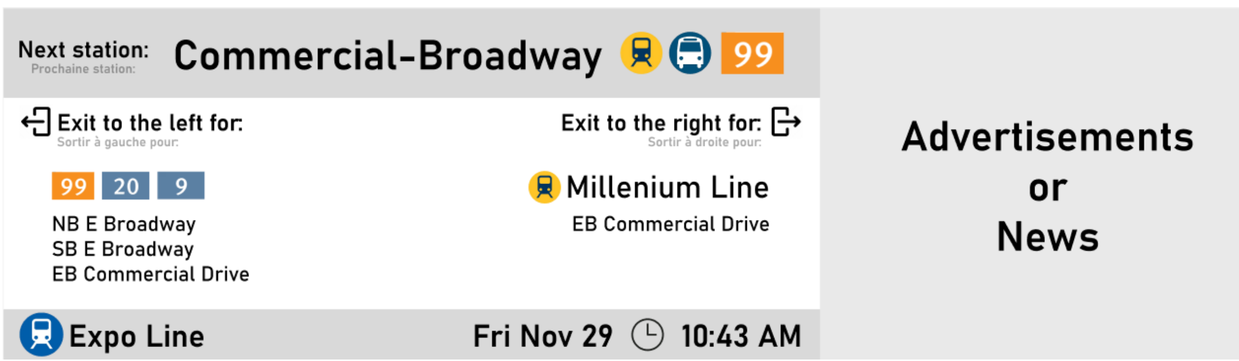
Known issues

- Too much **white space on the footer**, which can be filled with **date** and **current weather** conditions.
- **Bus connections for now are using images**, instead of text.
- Hard to implement **transition animations.**

Alternative design







Advantages:

- More displayed information.
- Realized more screens than in current interface.
- More detailed route section.

Known issues:

- Colour palette is hard to read for people with disabilities.
- Not implemented in Frontend (Made by using Figma).
- Outdated design.

Possible Improvements

Integration with Real-Time SkyTrain Timetable

- **Future Enhancement:** Implement a **Live Timetable Database** once access to **TransLink's scheduling system** is available.
- **Impact:** Allows ITIDS to display **dynamic arrival times, service alerts, and disruptions**.

AI-Powered Passenger Flow Analysis

- **Concept:** Utilize **computer vision or sensors** to estimate **passenger density** per train car.
- **Purpose:** Provide **real-time train crowding data**, helping passengers **choose less crowded cars**.

Mobile App Integration

- **Concept:** Extend ITIDS functionality to a **mobile app**.
- **Features:**
    - Personalized **real-time alerts** for upcoming transfers.
    - **Offline mode** with station details and transfer maps.