NYU

NYC TRANSIT HUB

END TERM EVALUATION

PRESENTED BY RUSHBAH B ,
ARANB D, UTKARSH M ,
SANYUKTA T, ARYAN A
29/04/2025

PROJECT OVERVIEW

- **Project Name:** NYC Transit Hub
- **Objective:** Develop a web application to offer real-time updates, schedules, and transit information for New York City's public transportation system. The platform will integrate with the MTA API to provide accurate and up-to-date transit data, ensuring commuters have access to essential travel information.
- **API Used:** https://api.mta.info

KEY FEATURES

1. Real-Time Train Updates: Live updates for subway lines.
2. Route Details: View route ID, direction, headsign, next stop, and arrival time.
3. Dynamic Frontend: Responsive UI for easy access to live train data.
4. Future Scope: User authentication, favorites, multilingual support, and push notifications.

NYU

TECH STACK USED

Backend:

- FastAPI for server-side development
- nyct_gtfs Python library to fetch live MTA GTFS feeds
- Uvicorn server for development

Frontend:

- React.js for dynamic, responsive web app
- API calls to FastAPI backend

Development Setup:

- Local development using `npm start` for frontend and `uvicorn main:app --reload` for backend

Future Enhancements (Planned for Production):

- Redis caching to optimize API usage

IMPLEMENTATION TIMELINE

WEEK 1

- Team formation (4-6 students)
- Exploring MTA API

WEEK 2 & 3

- Set up FastAPI backend and React frontend environment
- Backend live transit update integration

WEEK 4

- Finalize backend functionality
- Initiate frontend integration

NYU

PROGRESS ACHIEVED

- Successfully developed backend API /trains endpoint.
- Integrated live MTA subway data fetching using nyct_gtfs.
- Frontend fetches and displays real-time transit data dynamically.
- FastAPI backend deployed locally with CORS enabled for frontend communication.

NYU

SYSTEM DESIGN

MTA APIs

- Provides real-time transit data.
- Scheduler makes automated API calls at specific intervals to fetch up transit information.

Scheduler

- Responsible for periodically fetching new transit data from the MTA A
- Updates both the **MySQL database** and the **Redis cache** with the lat information.

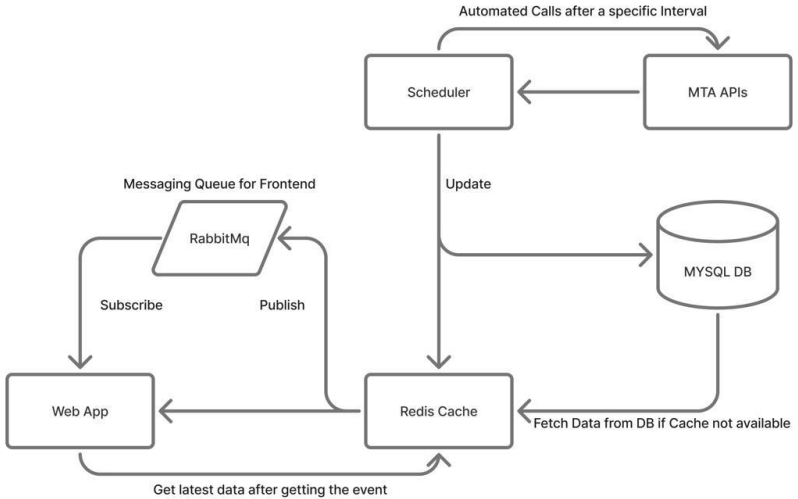
MySQL Database

- Stores historical and persistent transit data.
- If data is not available in the cache, it is fetched from MySQL.

Redis Cache

- Acts as a temporary storage for frequently accessed data to **reduce A calls and database queries**.
- If requested data is in Redis, it is returned immediately; otherwise, da fetched from MySQL and stored in the cache for future use.

Helps in **avoiding race conditions** and **optimizing performance**.



NYU

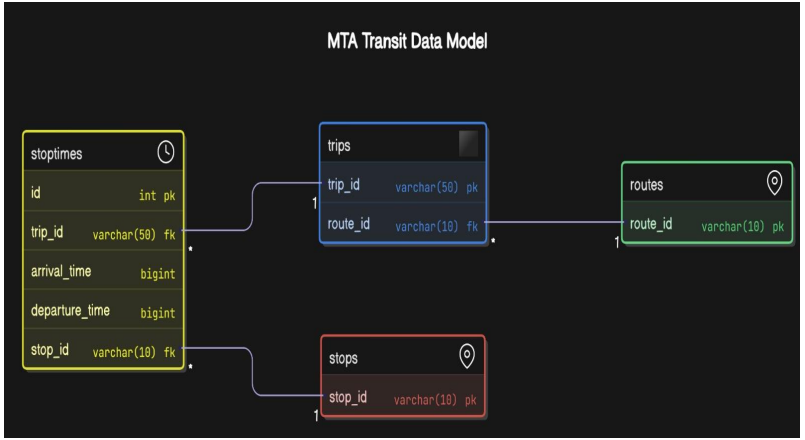
ENTITY RELATION DIAGRAM

Relationships:

1- routes to trips: A route can have multiple trips. This is a one-to-many relationship, indicated by the single line on the routes side. The route_id in the trips table is a foreign key referencing the route_id in the routes table.

2- trips to stoptimes: A trip can have multiple entries in stoptimes, recording the arrival and departure times at different stops. This is also a one-to-many relationship. The trip_id in the stoptimes table is a foreign key referencing the trip_id in the trips table.

3- stops to stoptimes: A stop can appear in multiple stoptimes records for different trips. This is a one-to-many relationship. The stop_id in the stoptimes table is a foreign key referencing the stop_id in the stops table.



NYU

FRONTEND AND FUNCTIONALITY

Frontend Built Using React.js:

- **Displays upcoming train schedules fetched live from the backend.**

- **Information displayed per trip:**
  - Trip ID
  - Route ID (subway line)
  - Direction (N/S or E/W)
  - Headsign (Train destination)
  - Next stop and expected arrival time

How It Works:

- User opens the website.
- Frontend sends a request to /trains API endpoint.
- Real-time transit information is displayed instantly.

Upcoming Features:

- Search trains by route.
- Save favorite subway lines.
- User login for personalized experiences.
- Multilingual support.

NYU

NYC Subway Tracker

Apr 29, 2025 | 12:47:18 PM

Search stations...

All Lines

1 Av

L — **8 Av** Northbound
IN_TRANSIT_TO 2 minutes
2 minutes

L — **Canarsie-Rockaway Pkwy** Southbound
STOPPED_AT a few seconds
a few seconds

103 St

1 — **South Ferry** Southbound
STOPPED_AT a few seconds
a few seconds

6 — **Pelham Bay Park** Northbound
STOPPED_AT a few seconds
a few seconds

B — **Bedford Park Blvd** Northbound
STOPPED_AT a few seconds
a few seconds

NEXT STEPS

Complete frontend enhancement for favorites and search.

Implement user authentication and favorite routes to personalize user experience.

Optimize API calls and improve data handling for better performance and reduced latency.

Implement caching (Redis) to store frequently accessed transit data, reducing database queries and preventing race conditions**.**

NYU

CONCLUSION

- Backend API integration completed successfully.
- Frontend dynamic fetching and real-time updates operational.
- Focus ahead: advanced user features and production-level deployment optimizations.
- Open to feedback and improvements.

NYU


“
DON’T BE
SOMEONE’S
SUBWAY
STORY
NYU

THANK YOU

NYU