

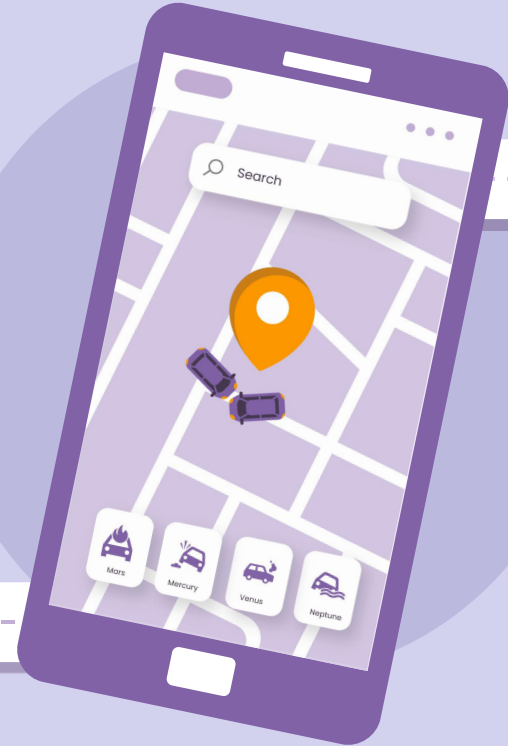
Mobile Computing Final Project

AYAN KUMAR SINGH

KANISHK KUMAR MEENA

RAHUL JHA

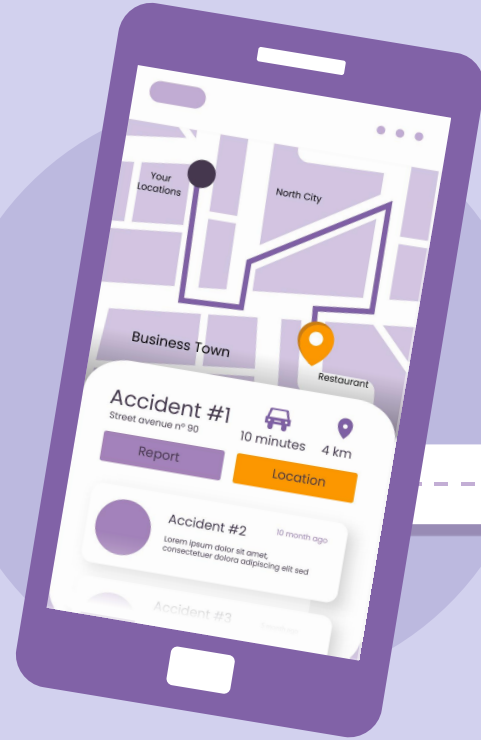
VANSH AGARWAL



TransitDelhi

A comprehensive Delhi transit application

- All-in-one solution for navigating Delhi's public transportation
- Modern UI built with Jetpack Compose
- Multiple integrated features in a single app



Project Overview



Key Features

- Metro Information System
- Real-Time Transit Tracking with Geocoding
- Bus Map System
- Fuel Station Finder
- Advanced Theme System
- Material Design 3 Interface



Advanced Implementation

- Accessibility enhancements
- Sensing capabilities
- Native API integration
- Unique UI elements



Basic Implementation

Core Functionality

- Multiple screens using Jetpack Compose
- Network connectivity with real-time data
- Local data storage and caching
- Background services for data updates
- Comprehensive error handling

Architecture

- MVVM pattern
- Repository pattern for data management
- Dependency injection with Hilt
- DataStore for preference persistence

Real-Time Transit Tracking



Features

- Live vehicle position tracking
- Address-to-coordinate conversion
- Reverse geocoding for readable locations
- Auto-refreshing data
- Distance calculations from user location
- Interactive map visualization

Implementation

- Location services integration
- Geocoding API integration for Delhi region
- Real-time data fetching
- Map visualization with vehicle markers
- Distance calculations using Android Location APIs



Real-Time Transit Tracking

Key Implementation Files

- RealTimeTransitScreen.kt: List view of vehicles
- LiveBusMapScreen.kt: Map visualization
- GeocodingService.kt: Location data transformation
- TransitViewModel.kt: Data management
- TransitRepository.kt: API integration

User Experience

- Toggle between list and map views
- Search for vehicles near specific addresses
- Automatic and manual refresh options
- Vehicle details at a glance
- Distance-based information



Bus Map System



Features

- Real-time bus location tracking
- Interactive map interface
- Color-blind friendly visualization
- Auto-refresh capability
- Bus filtering by route

Implementation

- Google Maps integration
- Room database for local caching
- Real-time API integration
- Accessibility features for color blind users



Metro Information System



Features

- Station search with real-time filtering
- Optimal route finding algorithm
- Visual journey representation
- Time, distance and fare estimation

Implementation

- Custom data structure for metro graph
- Dijkstra's algorithm for path finding
- Custom UI components for route visualization
- DMRC GTFS data integration

Metro Information System



Key Implementation Files

- MetroScreen.kt: Main UI implementation
- MetroViewModel.kt: Business logic and path finding
- MetroRepository.kt: Data management
- app/src/main/assets/lines/: Metro line data
- app/src/main/assets/DMRC_GTFS/: Official transit data

Key UI Components

- Station search with autocomplete
- Interactive source/destination selection
- Detailed route display with interchanges
- Station-by-station journey breakdown

Fuel Station Finder

Features

- Location-based gas station discovery
- Distance-sorted station list
- Address search using geocoding
- Detailed fuel type and pricing information
- Direct calling functionality

Implementation

- Location permissions handling
- Geocoding integration for address search
- Distance calculations
- Phone intent integration
- Custom card UI for station details

Fuel Station Finder

Key Implementation Files

- FuelScreen.kt: Main UI implementation
- FuelStationViewModel.kt: Data management
- Location service integration
- Custom UI components

User Experience

- Search for stations near any address
- Permission request handling
- Proximity-sorted stations
- Detailed fuel information
- One-tap calling

Advanced Theme System

Features

- Multiple theme options
- Accessibility-focused themes
- Color-blind modes
- Theme persistence
- Material You integration

Implementation

- DataStore for preference storage
- Theme selector UI
- Dynamic color schemes
- Material 3 theming
- Accessibility adaptations

Advanced Theme System

Key Implementation Files

- ThemePreferences.kt: Theme data management
- ThemeViewModel.kt: Theme business logic
- ThemeSelector.kt: UI components
- Theme.kt: Theme implementations
- Color.kt: Color definitions

User Experience

- Dynamic theme changing
- Preview of theme appearance
- Accessibility enhancement options
- Theme persistence across sessions
- Intuitive selection interface

Navigation & Main Screens

Implementation

- Bottom navigation with Jetpack Compose
- Home screen with feature highlights
- Consistent design language
- Seamless navigation between features

Key Files

- MainActivity.kt: Navigation setup
- HomeScreen.kt: Entry point and feature access
- Navigation.kt: Route definitions
- Material Design 3 theming

Accessibility

Implementation

- Implementation
- High contrast UI components
- Scalable typography
- Content descriptions for screen readers
- Clear error messaging
- Permission explanations
- Color-blind friendly themes and visualizations

Files

- `ui/theme/Color.kt`: Accessible color scheme
- `ui/theme/Type.kt`: Typography definitions
- Screen content descriptions throughout codebase
- Permission request dialogs with clear explanations

Sensing Capabilities

Implementation

- Location services for positioning
- Geocoding for address/coordinate conversion
- Network state monitoring
- Device orientation adaptations
- Environmental context awareness

Key Components

- FusedLocationProvider integration
- GeocodingService with regional optimization
- NetworkCallback implementation
- Responsive layouts for orientation changes
- Context-aware feature adaptation based on time and location

Native Code/API Integration

Implementation

- C++ Location Library with JNI bindings
- NDK-based networking stack for efficient data transfer
- Native rendering for complex route visualizations
- Native data processing for GTFS datasets
- C++ math libraries for optimized calculations

Key Components

- JNI bridging between Java/Kotlin and native code
- Native memory management for large datasets
- Performance optimization for intensive calculations
- Multi-threading using native primitives
- Platform-specific optimizations

Unique UI Elements

Implementation

- Custom route visualization
- Interactive station selection
- Live vehicle cards
- Bus location markers with accessibility features
- Theme selector with previews
- Dynamic map interfaces

Design Philosophy

- Intuitive information hierarchy
- Visual clarity for complex data
- Interactive elements for better engagement
- Consistent design language
- Accessibility-first approach

Technical Architecture

MVVM Architecture

- View layer: Jetpack Compose screens
- ViewModel layer: State management and business logic
- Model layer: Repositories and data sources

Key Components

- StateFlow for reactive UI updates
- Coroutines for asynchronous operations
- Hilt for dependency injection
- Room for local data persistence
- DataStore for preferences
- Material 3 components for UI


Data Management



Local Storage

- Room database for bus and transit data
- DataStore for user preferences and theme settings
- Asset management for metro data

Network Integration

- Retrofit for API communication
 - Custom C++ networking stack
 - Real-time data synchronization
 - Offline caching mechanism
 - Geocoding service for address/coordinate conversion
- 



Challenges & Solutions

Challenges

- Complex metro route finding algorithm
- Real-time data synchronization
- Permissions management
- UI for complex data visualization
- Accessibility across diverse user needs

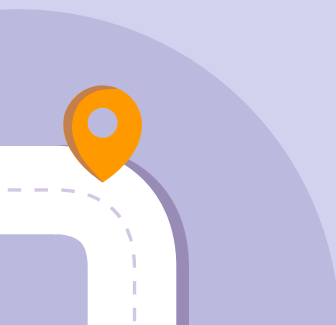

Solutions

- Custom graph implementation for metro network
- Kotlin Flow for reactive updates
- Comprehensive permission handling
- Custom composables for data visualization
- Multiple theme options and color-blind support

Future Enhancements



Planned Features

- Bus route planning
 - Real-time metro schedule information
 - Fare calculation for multi-modal journeys
 - User accounts for saving favorites
 - Offline mode for basic functionality
- 
- 

THANKS!

