TransitNet Optimizer

By

Muhammad Bilal

Department of Computer Science
Quaid-i-Azam University
Islamabad, Pakistan
2025

TransitNet Optimizer

By

Muhammad Bilal

A PROJECT REPORT SUBMITTED IN THE PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF SCIENCE

IN

COMPUTER SCIENCE

Supervised By

Dr. Rabeeh Ayaz Abbasi

Department of Computer Science
Quaid-i-Azam University
Islamabad, Pakistan
2025

Acknowledgments

Assalam o Alaikum,

First, I want to thank Allah Almighty with all my heart for guiding me during my time at Quaid-e-Azam University. I am very grateful to my mentor, Dr. Rabeeh Ayaz Abbasi, for his constant help and valuable advice. I also want to specially thank Dr. Onaiza Maqbool, Dr. Ayyaz Hussain, Dr. Akmal Saeed Khattak, Dr. Khalid Saleem, Dr. Ghazanfar Farooq Siddique, Dr. Muhammad Shuaib Karim, Dr. Muazzam Ali Khan Khattak, Dr. Mudassar Azam Sindhu, Dr. Umer Rashid, Ms. Ifrah Farukh Khan, Ms. Memoona Afsheen, and Sir Adeel ur Rehman Zaffar for their incredible support during my journey in the Computer Science department.

I would also like to sincerely thank Sir Irfan ul Haq for his continuous support and guidance during the challenges I faced in this project. His advice played a key role in helping me achieve success.

Throughout this amazing journey, I have not only learned a lot about the industry but also applied my skills and knowledge to the best of my ability. I am deeply thankful for the endless support of my family and friends, especially Muhammad Ahmad and Ahsan Bashir, who helped me reach this important milestone.

Abstract

Public transport systems in many cities, including Islamabad, are not working well because they use old and outdated planning methods. Right now, transport routes are planned using manual surveys, which take a lot of time, are often inaccurate, and don’t match how people actually travel. Because of this, public transport networks are not efficient, leading to poor service, longer travel times, and higher costs.

This project aims to create a smart Public Transport Route Optimization System that uses data to improve route planning and make the system more efficient. By analyzing real-time and historical travel data, the system will study how people move around the city and suggest better routes that improve access and reduce traffic jams.

This system will help transport authorities, commuters, and city planners by providing a smarter and more efficient way to optimize public transport routes. By fixing inefficiencies and improving services, this project will make public transport more reliable and user-friendly. It will also help reduce fuel consumption, lower environmental impact, and improve urban mobility. In the future, the system could be upgraded to include real-time traffic updates and integrate different types of transport for an even better solution.

Table of Contents

List of Abbreviations

Term	Description
PT	Public Transport
GCF	Grant Challenge Fund
HEC	Higher Education Commision
MAT	Mass Transit Authorities
CDA	Capital Development Authority

Chapter 1

Introduction

1.1 Background and Context

Public transportation engineers traditionally rely on manual surveys to define Public Transport (PT) routes. This process is time-consuming, tedious, costly, and prone to errors. This project aims to address these challenges by acquiring mobility patterns of people from cellular networks and leveraging this data to define new PT routes and optimize existing ones.

The TransitNet Optimizer is designed as an interactive, web-based solution to enhance public transport networks by utilizing aggregated mobility patterns derived from cellular data. The system will feature PT network optimization algorithms, an interactive data manipulation interface with efficient backend management, and multi-layer visualizations for effective result interpretation.

This solution will provide PT management authorities with a computational tool to make informed decisions about PT service planning and operations, leading to increased efficiency and improved route planning.

1.2 Motivation

I am currently working as a Backend Developer on the project Optimum Use of Existing Resources: A Prototype Model of Road Safety (GCF-744), funded by the World Bank under HEC's Grant Challenge Fund. Through this experience, I have observed that the existing public transport network in Islamabad does not align with the actual mobility patterns of its residents. This misalignment highlights a critical need for route optimization.

I believe that introducing a data-driven approach, like the TransitNet Optimizer, can transform the current manual and error-prone practices of route planning into an efficient, automated process. By leveraging cellular mobility data and advanced algorithms, this solution has the potential to minimize resource wastage, reduce traffic congestion, and improve overall public transportation services.

Currently, Mass Transit Authorities (MAT) and the Capital Development Authority (CDA) of Islamabad lack a technical solution to address this issue. By utilizing the TransitNet Optimizer, we can create a public transport network that is tailored to the mobility patterns of the population, ensuring more efficient and user-friendly transportation services.

1.3 Problem Statement

Public transport systems in many cities, particularly in Islamabad, rely on outdated planning methods. Currently, transport routes are planned using manual surveys, which take a lot of time and are often inaccurate. Moreover, MAT lack a technical tool to design routes based on the actual mobility patterns of people. This gap hinders the optimization of public transport networks, leading to inefficiencies and poor service delivery.

1.4 Scope

1.4.1 Functionalities

Data Upload & Management

- **Interactive Upload Interface:** Users can easily upload public transport network data and inter-region mobility percentages. The interface is user-friendly and supports both full file uploads and partial input. The backend is updated only if new changes are detected in the uploaded data, ensuring data integrity and system efficiency.
- **Admin Dashboard:** An admin panel allows users to perform CRUD (Create, Read, Update, Delete) operations on public transport data, empowering users to manage and update their transport network data effectively.

Network Visualization

- **Multi-layer Map View:** The platform generates a dynamic network graph displayed on a map with multiple visualization layers, including:
  - **Public Transport Stops:** View and analyze the distribution and accessibility of transport stops across the network.
  - **Mobility Patterns:** Visualize inter-region mobility patterns to identify areas with high or low traffic flow.
  - **Overlay View:** An interactive overlay that combines public trans-

port stops with mobility patterns, offering insights into the network's performance and highlighting potential areas for optimization.

- **Interactive Filters:** Users can apply multiple filters to each visualization layer, enabling more detailed analysis and allowing for targeted insights.

Route Optimization

- **Optimization Algorithm:** The system includes a robust optimization algorithm that analyzes the current public transport network and recommends improvements. This feature optimizes routes to minimize travel times, reduce network diameter, and improve overall service efficiency.
- **Optimized Network Visualization:** The optimized network is displayed with detailed statistics, such as network diameter, coverage, and other relevant metrics. This allows transit authorities to make informed decisions based on quantitative data.

1.4.2 Stakeholders

The primary users of TransitNet Optimizer are mass transit authorities and urban planners. The platform is designed to assist them in visualizing their current public transport network, understanding mobility patterns, and implementing optimizations to enhance network performance. Ultimately, this will lead to improved public transport services, benefiting the community.

1.5 Existing Systems

Public transport optimization has been a subject of increasing attention in urban mobility research, driven by the complexities of modern transportation systems. Previous efforts have largely focused on integrating advanced data sources such as mobile phone data and GPS information to enhance network planning and optimization. For instance, the analysis of mobile phone data has gained traction for identifying urban mobility patterns. Studies like Yang et al. Yang et al. (2019) explored cellular data from millions of travelers to create models that predict travel patterns, while Çolak et al. Çolak et al. (2018) leveraged mobile phone data to infer transportation modes, offering cost-effective alternatives to traditional surveys. These efforts have paved the way for dynamic transportation planning based on real-time mobility data.

Additionally, multi-objective optimization approaches have been developed to balance cost, efficiency, and environmental impact in public transport systems. Xiao et al. Xiao et al. (2024) applied GPS data and traffic reports to optimize routes while maintaining sustainable practices. Graph-based techniques, including algorithms like Dijkstra's and Kruskal's, have also been widely employed to model transportation networks and improve route planning, minimizing costs and maximizing network efficiency Likaj (2013). These methods are particularly useful in handling the dynamic nature of real-time transportation data.

Despite these advancements, challenges remain in integrating large-scale mobility data with existing public transport networks, particularly when it comes to optimizing inter-region connections and designing efficient routes. The current systems often rely on traditional data collection methods, which can be inaccurate and labor-intensive. This project aims to bridge these gaps by integrating mobility patterns and optimizing routes through a computational framework that processes public transport data and inter-region travel patterns efficiently.

Chapter 2

Requirements and Analysis

The Requirements and Analysis chapter defines the system's needs and examines how these requirements will shape its development. This chapter ensures that the system meets user expectations and performs efficiently in real-world scenarios. It provides a structured approach to understanding user needs, technical constraints, and system functionality. By analyzing these aspects, the chapter helps in identifying potential challenges and ensuring a smooth transition from planning to implementation.

2.1 Functional Requirements

1. **Data Upload & Management**
    - A simple and user-friendly interface to upload public transport data and mobility percentages.
    - Support for both full and partial data uploads.
    - The backend should update only when changes are detected to maintain data accuracy.

- An admin dashboard to perform CRUD (Create, Read, Update, Delete) operations on transport data.

2. **Network Visualization**

    - A multi-layer map showing transport stops, mobility patterns, and an overlay view.
    - Interactive filters to allow users to analyze details more effectively.

3. **Route Optimization**

    - An optimization algorithm to improve routes, reduce travel time, and make services more efficient.
    - Display the optimized network with key metrics to help users make informed decisions.

2.2 Non-Functional Requirements

Performance : Fast response times: visualizations and optimizations should load within 10 seconds.

Usability : The interface should be easy to use and navigate, even for non-technical users.

Scalability : The system should handle larger datasets and support future upgrades.

Security : Secure login and data encryption to protect sensitive information.

Reliability : The system should have 99.9% uptime and include regular data backups.

Compatibility : The system should work on both web and mobile platforms.

2.3 Use Cases

The following use cases describe the key functionalities and interactions that the TransitNet Optimizer provides PT Engineers , Management Authorities and Policy Makers. These use cases ensure an efficient, user-centric approach to public transport network optimization and planning.

2.3.1 Sign-Up

UC-1	User Authentication - Sign-Up
Actors	• Public Transportation Engineers • PT Management Authorities • Policy Makers
Goal	To allow new users to create an account with the appropriate user type.
Preconditions	• The system is accessible via a web interface.
Steps	1. The user navigates to the sign-up page. 2. The system prompts the user to enter details such as name, email, password, and user type (e.g., Engineer, Policy Maker). 3. The system validates the inputs for accuracy and completeness. 4. The user submits the information. 5. The system creates an account and sends a confirmation email to the user. 6. The user verifies the account through the email link.

Alternative Flows	• **Invalid Input:** If any required field is empty or incorrect, the system prompts the user to correct the input and resubmit. • **Email Already Registered:** If the email is already in use, the system notifies the user and suggests password recovery. • **Verification Failure:** If the user fails to verify the email within a set timeframe, the system provides an option to resend the verification link.
Postconditions	A new user account is created and verified with the appropriate user type.

2.3.2 Login

UC-2	User Authentication - Login
Actors	• Public Transportation Engineers • PT Management Authorities • Policy Makers
Goal	To securely authenticate users for accessing the system.
Preconditions	• The user has an existing account.
Steps	1. The user navigates to the login page. 2. The system prompts the user to enter their email and password. 3. The user selects their user type (e.g., Engineer, Policy Maker) if applicable. 4. The system validates the credentials and user type. 5. Upon successful validation, the user is granted access to the system. 6. If the credentials are invalid, the system prompts the user to retry or recover their password.

Alternative Flows	• **Incorrect Credentials:** If the user enters incorrect email/password, the system displays an error message and allows three retries before temporarily locking the account. • **Forgot Password:** If the user selects "Forgot Password," the system initiates a password recovery process via email verification. • **Account Locked:** After multiple failed attempts, the system temporarily locks the account and notifies the user to reset the password or contact support.
Postconditions	The user is securely authenticated and gains access to the system.

2.3.3 Route Creation

UC-3	Defining New Public Transport Routes
Actors	• Public Transportation Engineers • PT Management Authorities
Goal	To define new PT routes based on aggregated mobility patterns derived from cellular data.
Preconditions	• Mobility data has been collected and preprocessed. • The user has access to the TransitNet Optimizer system.
Steps	1. The user logs into the TransitNet Optimizer system. 2. The system displays a map interface with aggregated mobility data overlaid. 3. The user selects an area of interest to analyze movement patterns. 4. The system suggests potential routes based on detected mobility clusters.

Steps	1. The user evaluates and modifies the suggested routes using interactive tools. 2. The finalized routes are saved to the system for further analysis or export.
Alternative Flows	• **No Mobility Data Available:** If the system lacks mobility data, the user is notified and prompted to upload or request data. • **Invalid Route Suggestions:** If the suggested routes are not optimal, the user can manually adjust routes using editing tools. • **Data Loading Issues:** If map data fails to load, the system provides troubleshooting steps or allows the user to retry.
Postconditions	New PT routes are defined and stored in the system.

2.3.4 Route Optimization

UC-4	Optimizing Existing Public Transport Routes
Actors	• Public Transportation Engineers • PT Management Authorities
Goal	To optimize current PT routes to improve efficiency and reduce operational costs.
Preconditions	• Existing PT route data is uploaded to the system. • Mobility data is available for analysis.
Steps	1. The engineer uploads the current PT route data into the system. 2. The system overlays the routes on the mobility pattern visualization. 3. The engineer identifies inefficiencies, such as redundant stops or routes with low utilization. 4. The system suggests optimized routes based on travel demand and mobility patterns. 5. The engineer reviews and refines the suggested optimizations. 6. The updated routes are saved and exported for implementation.

Alternative Flows	• **Missing Data:** If route or mobility data is missing, the system notifies the user and provides an option to upload or retrieve necessary data. • **Ineffective Optimization:** If the suggested optimizations do not meet requirements, the engineer can manually adjust or request an alternative optimization. • **System Errors:** If the system encounters processing errors, it logs the issue and suggests troubleshooting or retrying later.
Postconditions	Optimized PT routes are stored and available for implementation.

2.3.5 Data Manipulation

UC-5	Interactive Data Manipulation
Actors	• Public Transportation Engineers • PT Management Authorities
Goal	To enable users to manipulate and analyze mobility data interactively for customized insights.
Preconditions	• Mobility data is accessible in the system.
Steps	1. The engineer accesses the data manipulation interface. 2. The system provides tools to filter, sort, and visualize data by criteria such as time, location, or travel frequency. 3. The engineer performs data manipulations to uncover specific patterns or trends. 4. The results are displayed in real-time using multi-layer visualizations. 5. The engineer saves the analyzed data for use in route planning.

Alternative Flows	• **Missing Data:** If the required mobility data is unavailable, the system notifies the user and suggests alternative data sources or retrieval options. • **Invalid Filters:** If the applied filters return no meaningful results, the system alerts the user and offers recommendations for refining the query. • **System Lag:** If real-time visualization is delayed, the system displays a progress indicator and allows the user to adjust the processing resolution.
Postconditions	Customized mobility insights are available for PT planning.

2.3.6 Layered Visualization

UC-6	Multi-Layer Visualization of Results
Actors	• Public Transportation Engineers • PT Management Authorities
Goal	To provide comprehensive visual interpretations of mobility patterns and optimization results.
Preconditions	• Mobility analysis or route optimization results are ready for visualization.
Steps	1. The user selects the visualization module from the system interface. 2. The system displays multiple visualization layers, such as heatmaps, cluster diagrams, and optimized route overlays. 3. The user toggles layers on and off to focus on specific data insights. 4. The user customizes visualization parameters for better clarity (e.g., adjusting zoom levels or color schemes). 5. The user exports the visualizations for reporting or presentation.

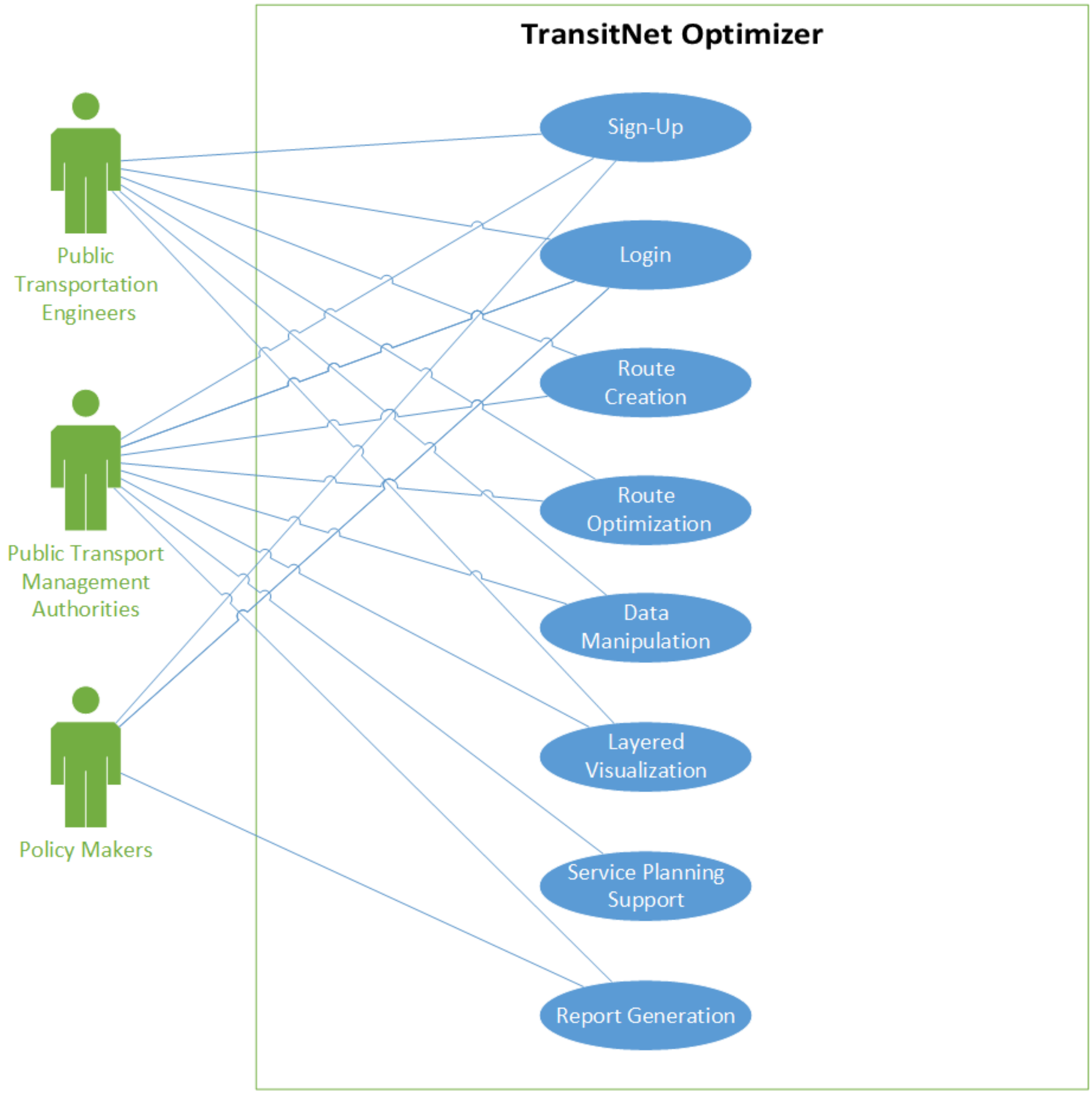
Alternative Flows	• **Missing Data Layers:** If certain data layers are unavailable, the system notifies the user with a warning and suggests alternative visualizations. • **Layer Selection Error:** If the user selects incompatible layers (e.g., combining layers with mismatched data types), the system prompts the user to correct the selection. • **Export Failure:** If the export process fails (e.g., due to file corruption), the system alerts the user and provides options for retrying or exporting in another format.
Postconditions	Multi-layer visualizations are generated and shared for effective decision-making.

2.3.7 Service Planning Support

UC-7	Decision Support for PT Service Planning
Actors	• PT Management Authorities
Goal	To assist authorities in making data-driven decisions for public transport service planning.
Preconditions	• All analysis and optimization processes are complete.
Steps	1. The authorities log into the system and access decision support tools. 2. The system presents key metrics and visualizations derived from mobility analysis and route optimization. 3. The authorities simulate different scenarios (e.g., adding or removing routes, adjusting schedules). 4. The system provides predictive insights on the impact of proposed changes. 5. The authorities finalize and export their decisions for implementation.

Alternative Flows	• **Scenario Simulation Error:** If an error occurs during the simulation of proposed changes, the system alerts the authorities and provides an option to retry or adjust the input parameters. • **Inaccurate Predictions:** If the predictive insights are inaccurate (e.g., due to insufficient data), the system recommends refining the input data or adjusting model parameters. • **Export Failure:** If the export of the final decision plan fails, the system alerts the authorities and offers options to retry or export the plan in an alternative format.
Postconditions	A data-driven decision plan is ready for implementation.

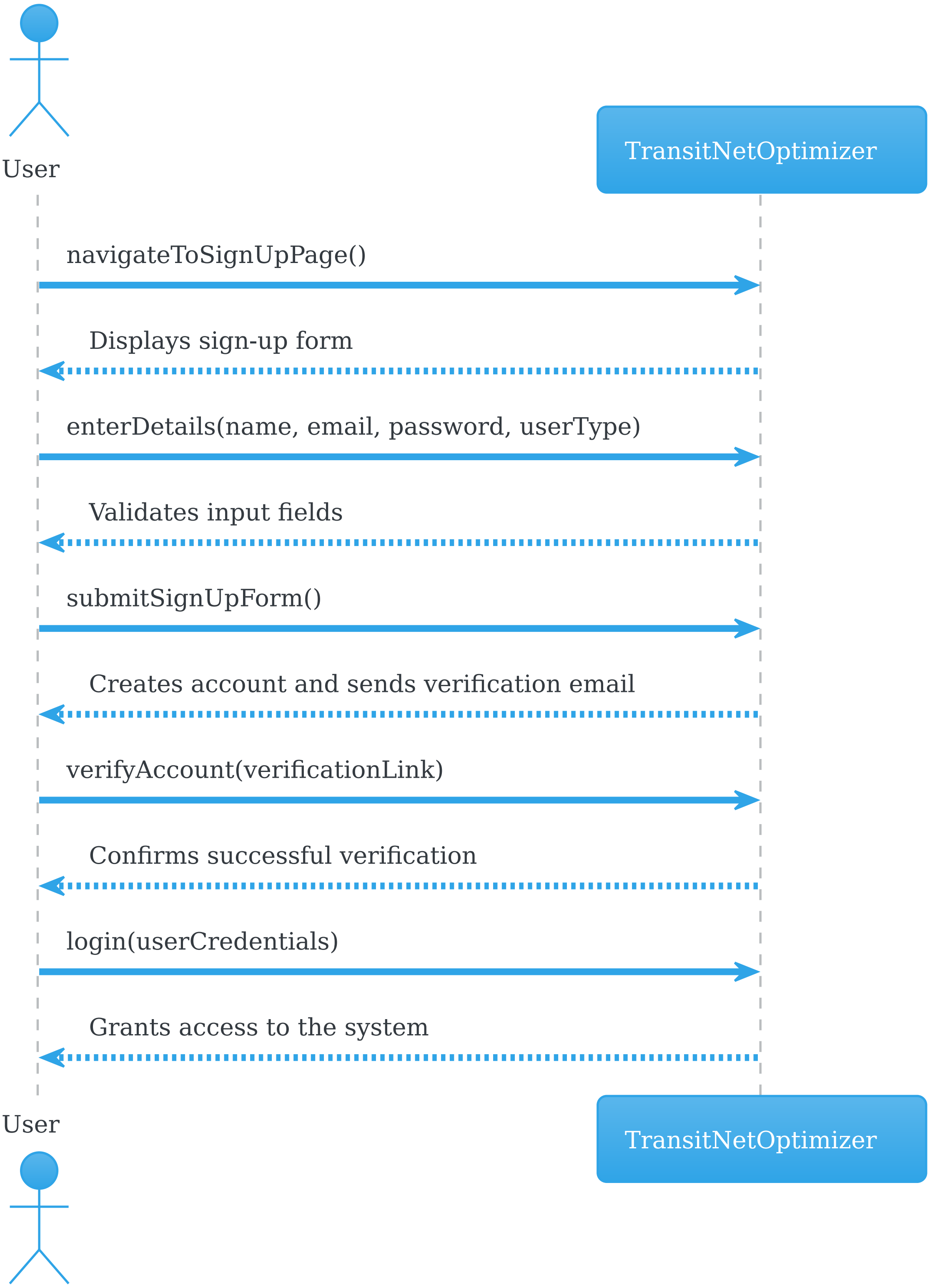
2.4 Use Case Diagram



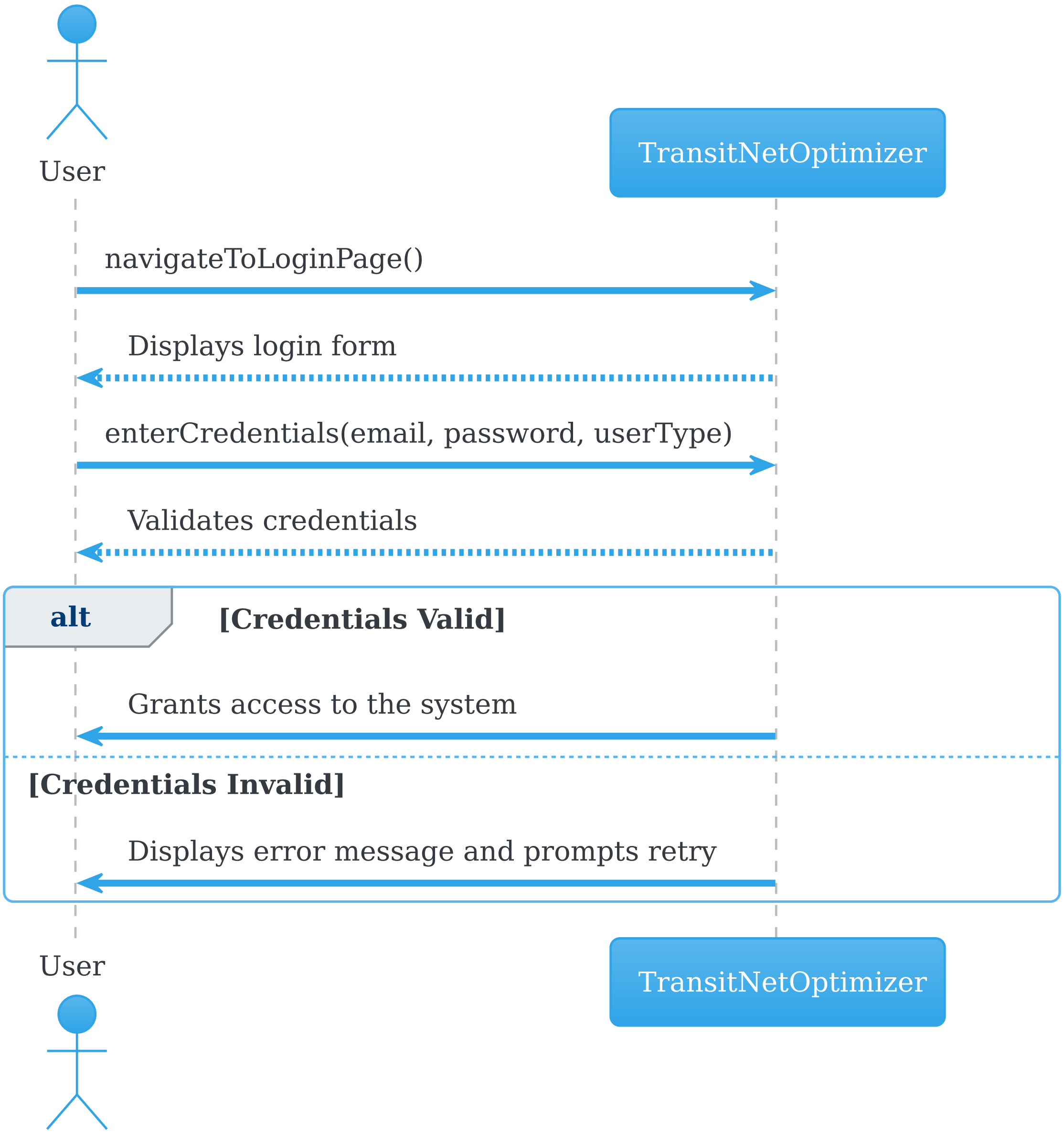
2.5 System Sequence Diagrams

System Sequence Diagrams illustrate the sequence of interactions between the system and external actors (such as users or other systems). They show how the system responds to a specific event or request over time. SSDs help to define the flow of data and the order of operations needed for a particular function in the system.

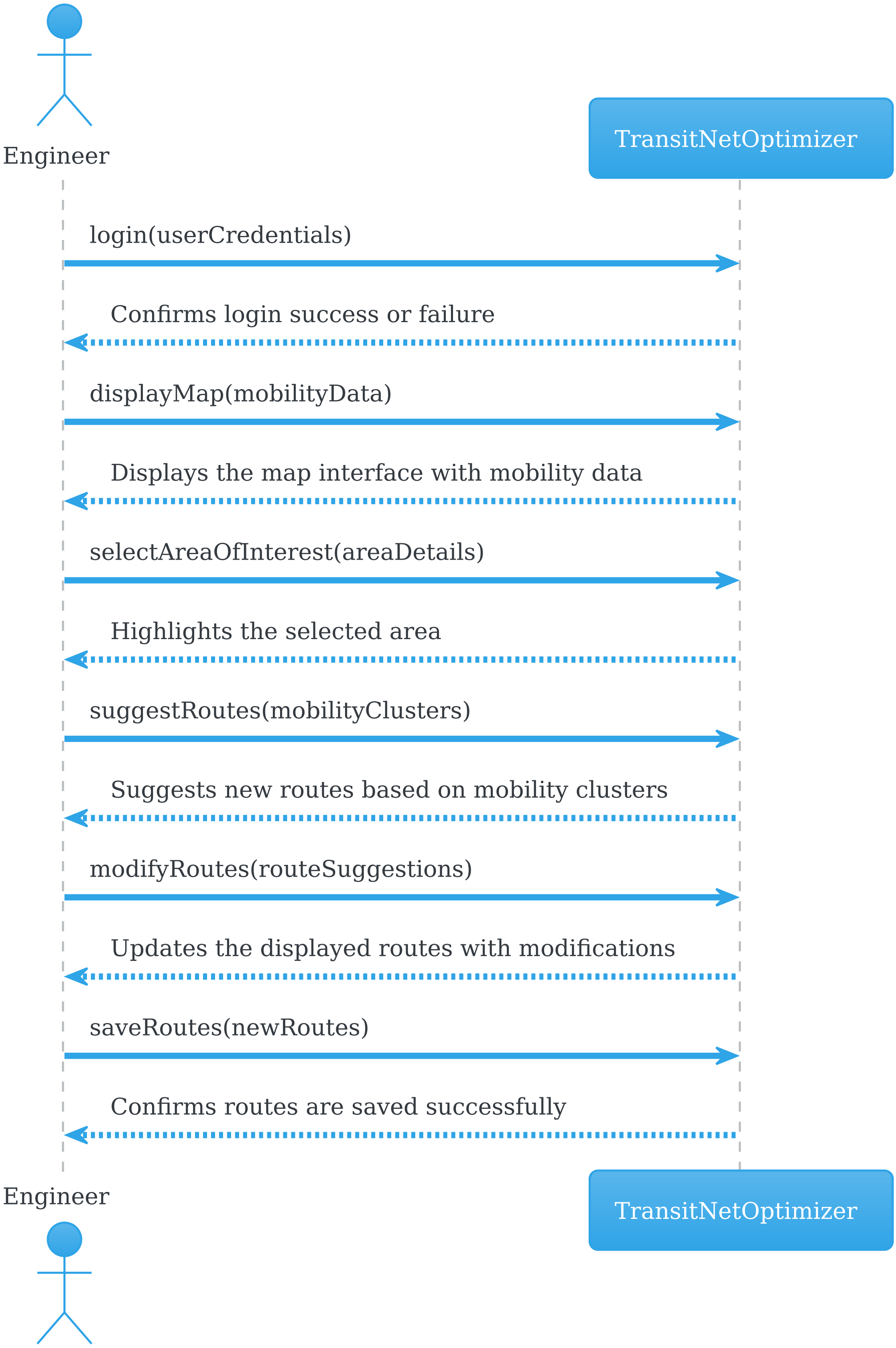
2.5.1 User Authentication - Sign-Up


User
TransitNetOptimizer
navigateToSignUpPage()
Displays sign-up form
enterDetails(name, email, password, userType)
Validates input fields
submitSignUpForm()
Creates account and sends verification email
verifyAccount(verificationLink)
Confirms successful verification
login(userCredentials)
Grants access to the system
User
TransitNetOptimizer

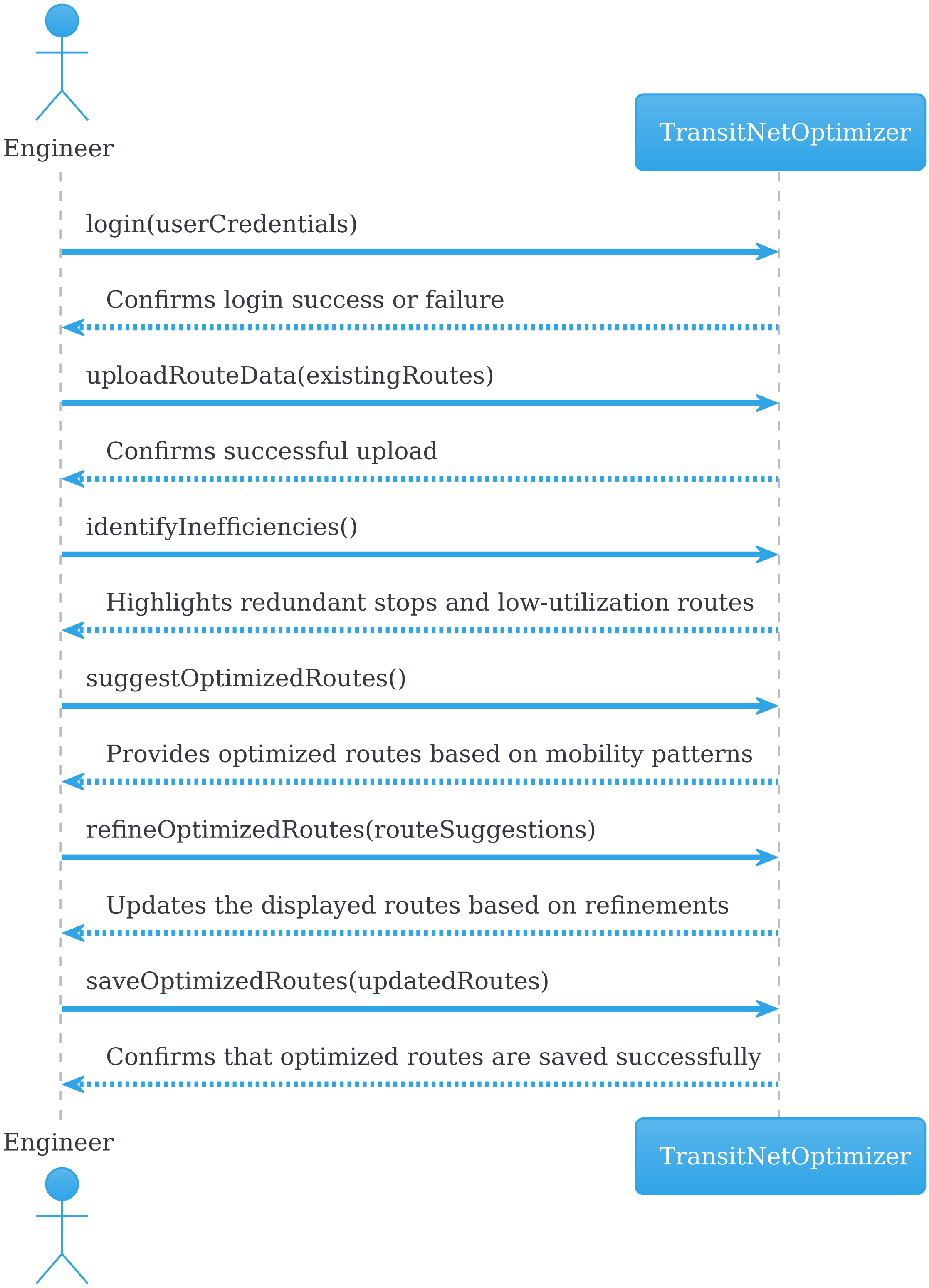
2.5.2 User Authentication - Login



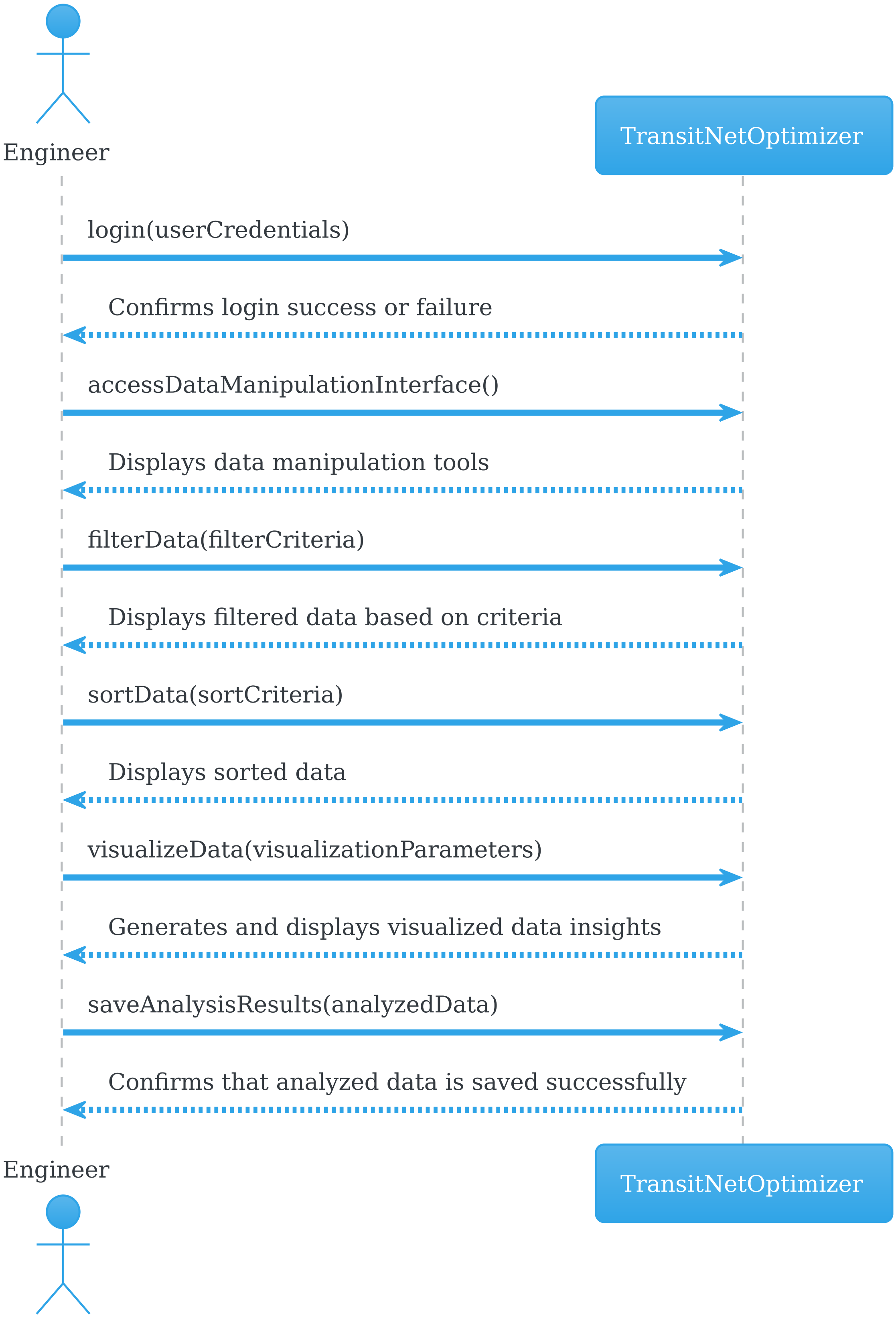
2.5.3 Route Creation


Engineer
TransitNetOptimizer
login(userCredentials)
Confirms login success or failure
displayMap(mobilityData)
Displays the map interface with mobility data
selectAreaOfInterest(areaDetails)
Highlights the selected area
suggestRoutes(mobilityClusters)
Suggests new routes based on mobility clusters
modifyRoutes(routeSuggestions)
Updates the displayed routes with modifications
saveRoutes(newRoutes)
Confirms routes are saved successfully
Engineer
TransitNetOptimizer

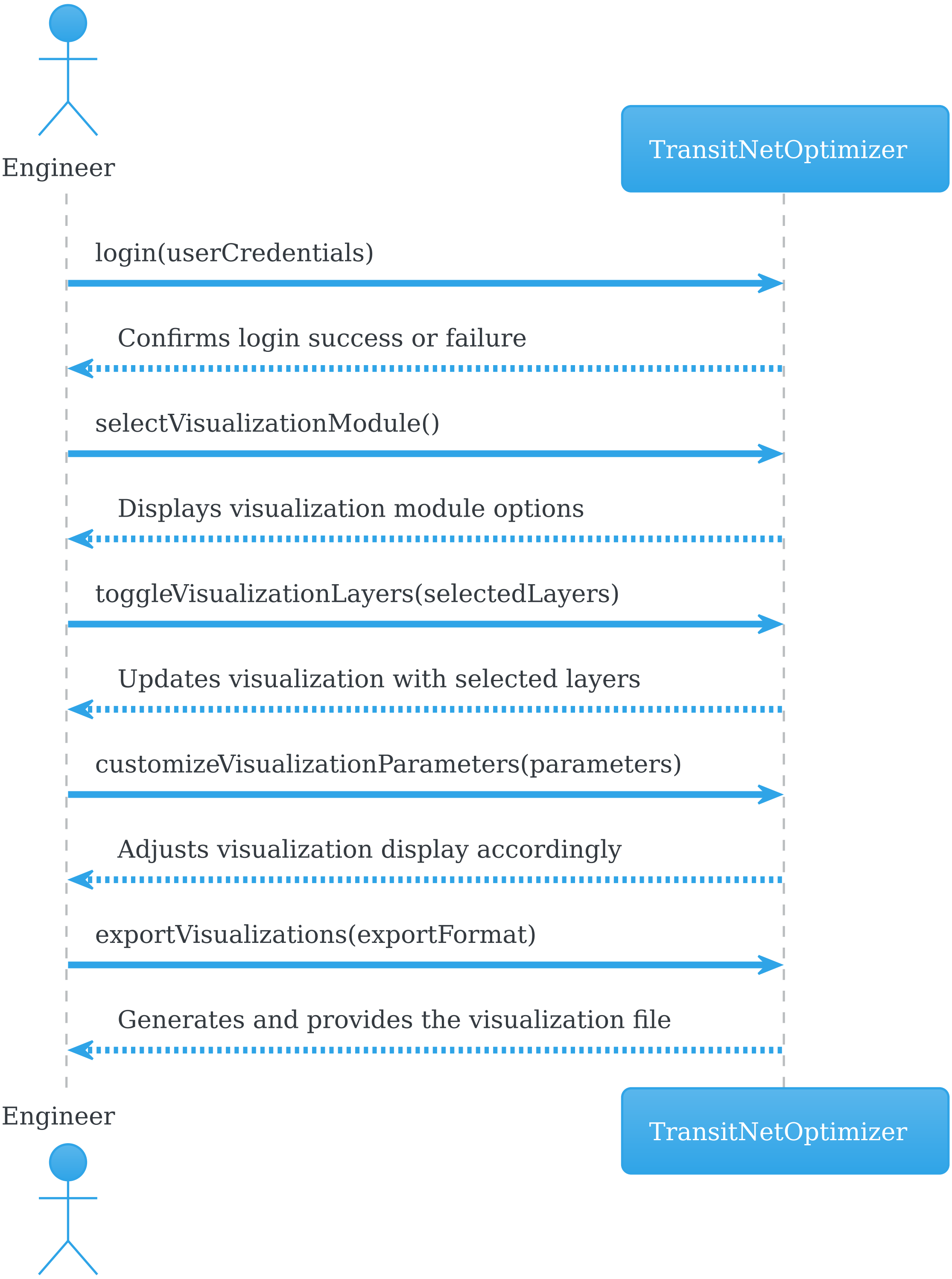
2.5.4 Route Optimization


Engineer
TransitNetOptimizer
login(userCredentials)
Confirms login success or failure
uploadRouteData(existingRoutes)
Confirms successful upload
identifyInefficiencies()
Highlights redundant stops and low-utilization routes
suggestOptimizedRoutes()
Provides optimized routes based on mobility patterns
refineOptimizedRoutes(routeSuggestions)
Updates the displayed routes based on refinements
saveOptimizedRoutes(updatedRoutes)
Confirms that optimized routes are saved successfully
Engineer
TransitNetOptimizer

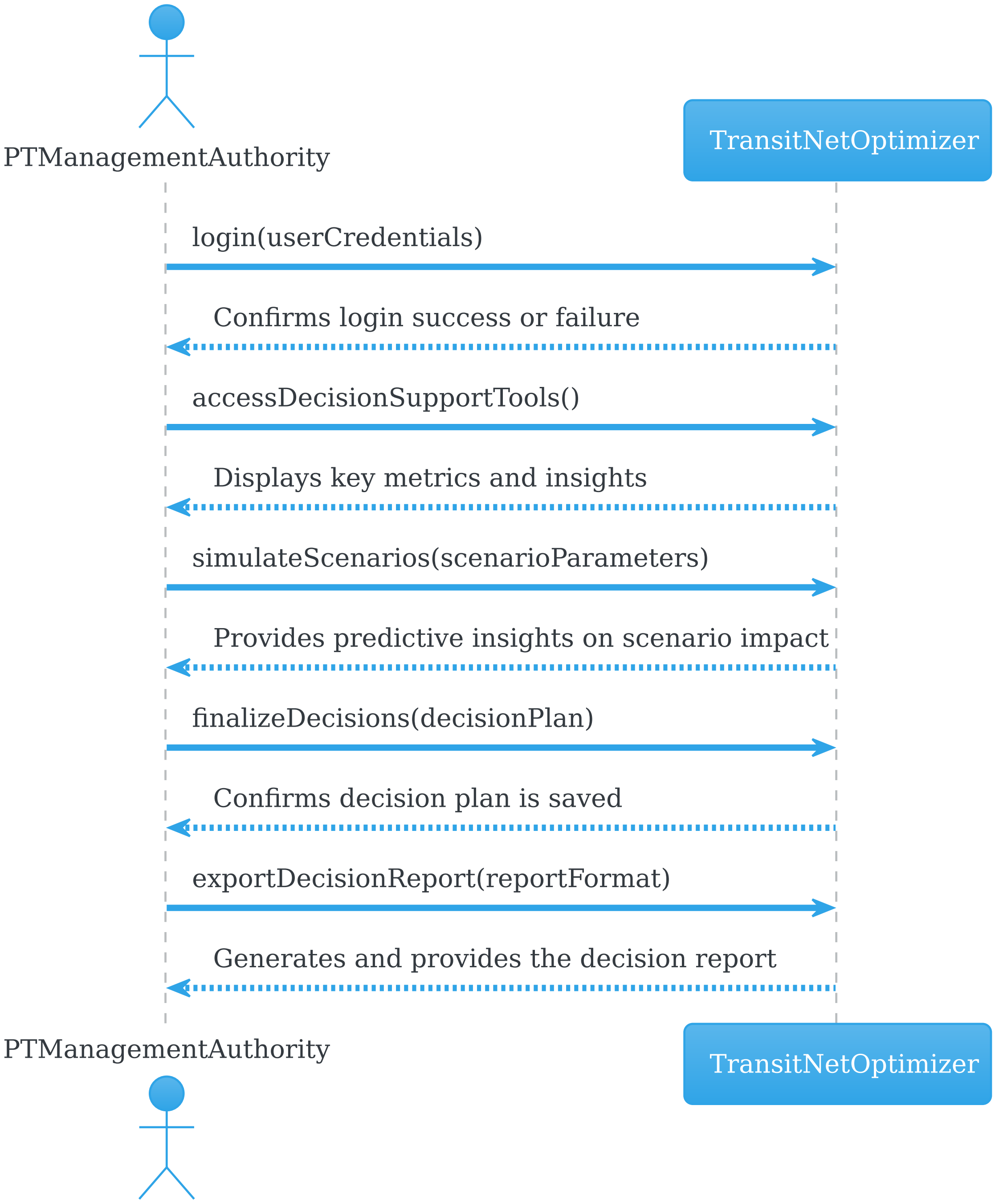
2.5.5 Data Manipulation


Engineer
TransitNetOptimizer
login(userCredentials)
Confirms login success or failure
accessDataManipulationInterface()
Displays data manipulation tools
filterData(filterCriteria)
Displays filtered data based on criteria
sortData(sortCriteria)
Displays sorted data
visualizeData(visualizationParameters)
Generates and displays visualized data insights
saveAnalysisResults(analyzedData)
Confirms that analyzed data is saved successfully
Engineer
TransitNetOptimizer

2.5.6 Layered Visualization

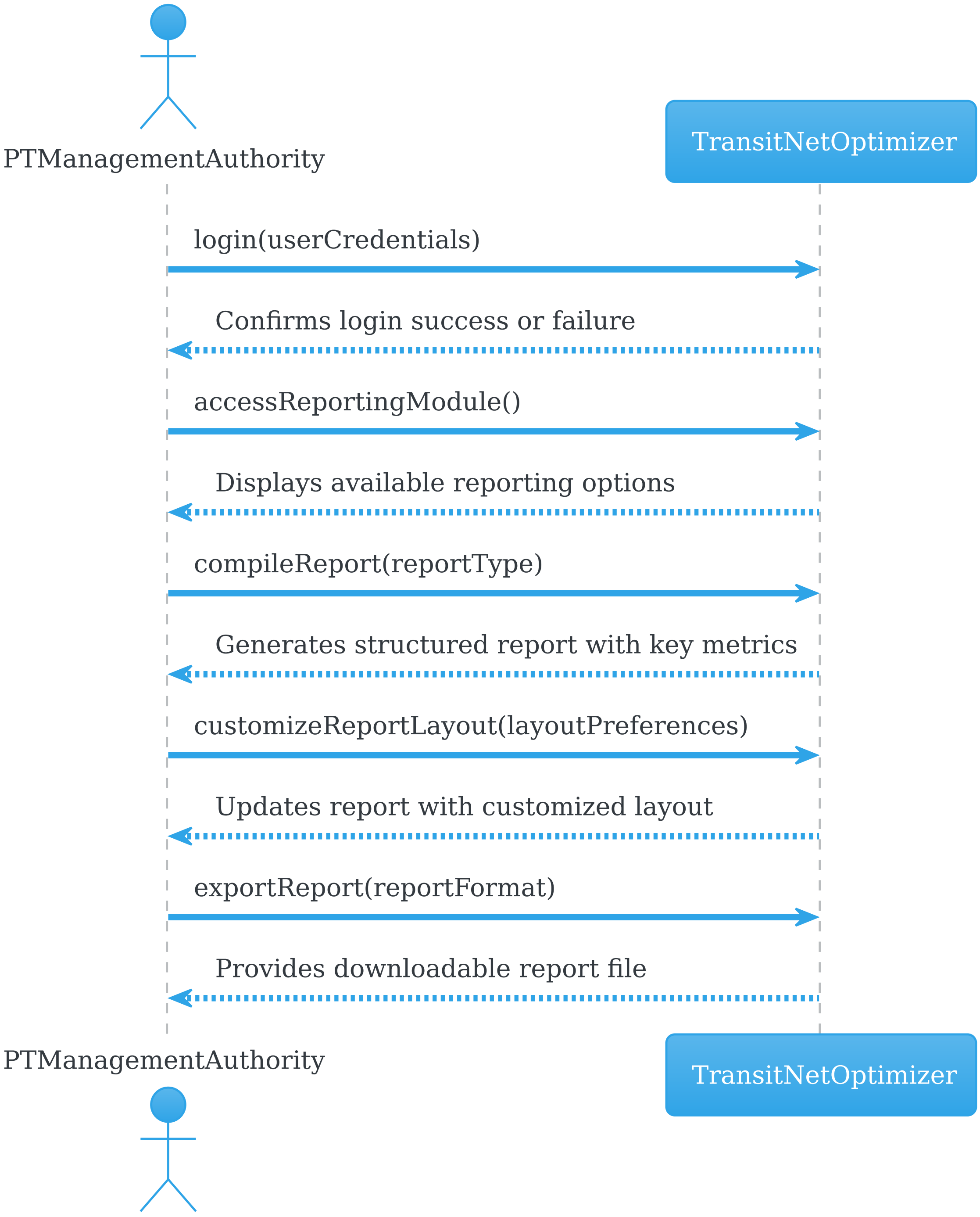

Engineer
TransitNetOptimizer
login(userCredentials)
Confirms login success or failure
selectVisualizationModule()
Displays visualization module options
toggleVisualizationLayers(selectedLayers)
Updates visualization with selected layers
customizeVisualizationParameters(parameters)
Adjusts visualization display accordingly
exportVisualizations(exportFormat)
Generates and provides the visualization file
Engineer
TransitNetOptimizer

2.5.7 Service Planning Support

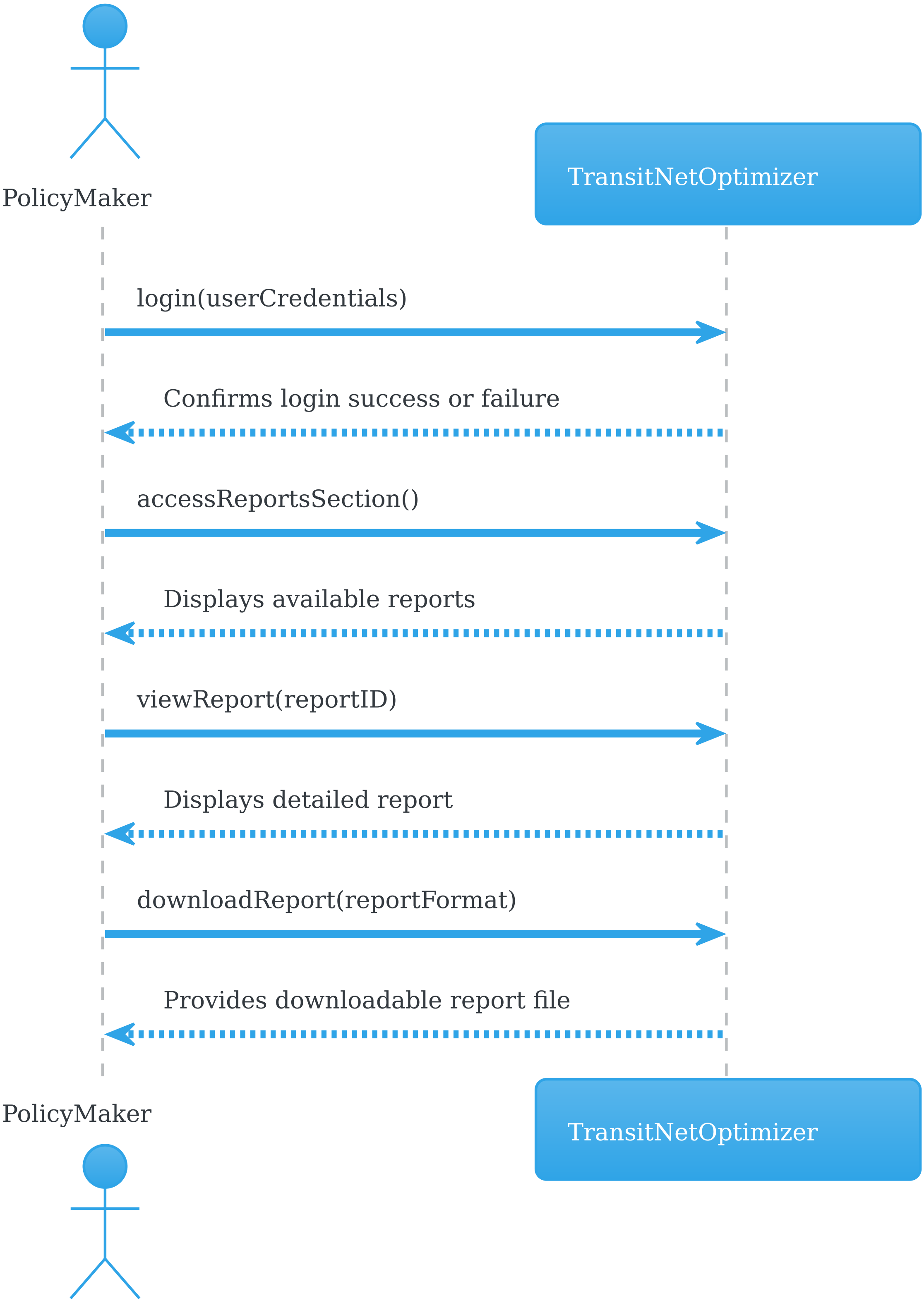

PTManagementAuthority
TransitNetOptimizer
login(userCredentials)
Confirms login success or failure
accessDecisionSupportTools()
Displays key metrics and insights
simulateScenarios(scenarioParameters)
Provides predictive insights on scenario impact
finalizeDecisions(decisionPlan)
Confirms decision plan is saved
exportDecisionReport(reportFormat)
Generates and provides the decision report
PTManagementAuthority
TransitNetOptimizer

2.5.8 Report Generation

PT Management Authorities

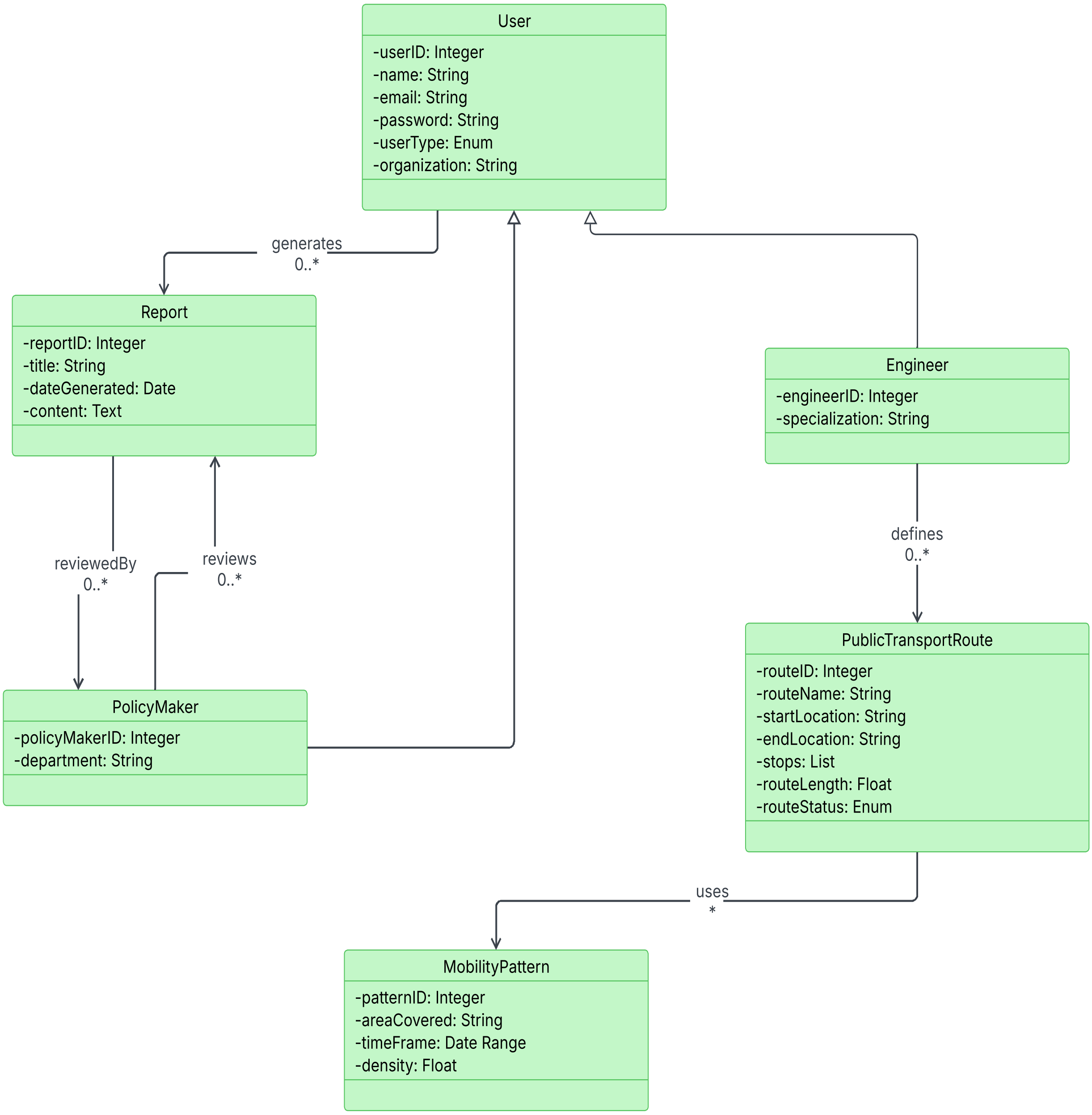

PTManagementAuthority
TransitNetOptimizer
login(userCredentials)
Confirms login success or failure
accessReportingModule()
Displays available reporting options
compileReport(reportType)
Generates structured report with key metrics
customizeReportLayout(layoutPreferences)
Updates report with customized layout
exportReport(reportFormat)
Provides downloadable report file
PTManagementAuthority
TransitNetOptimizer

Policy Makers



2.6 Domain Model

The Domain Model visually represents the core concepts of the system and their relationships. It focuses on the problem domain (not the system's technology or implementation) and helps in understanding the key entities and their interactions in the real-world context the system addresses. It serves as a bridge between the problem space and the solution space, ensuring the system design aligns with user needs.



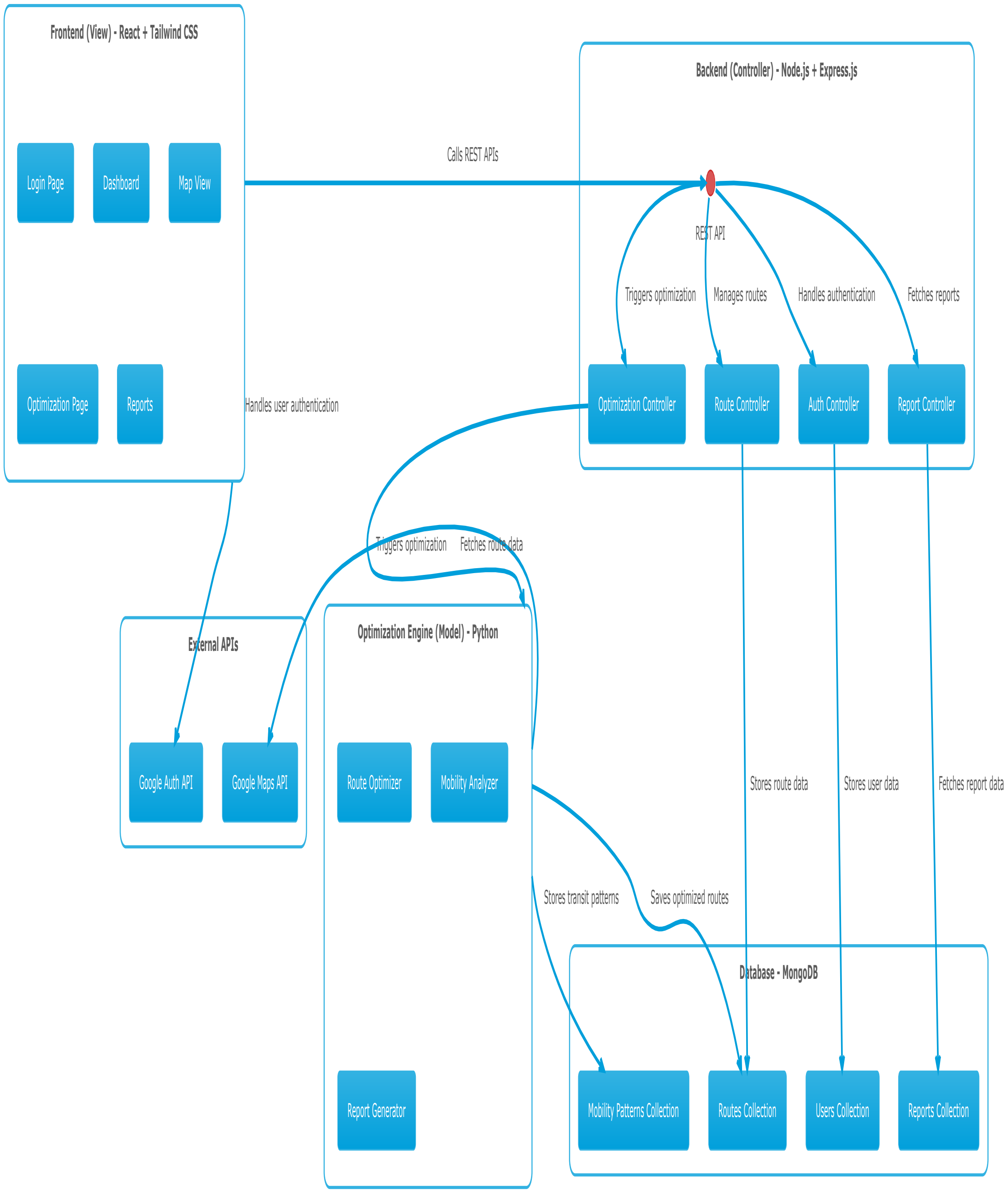
Chapter 3

System Design

The System Design chapter outlines the architectural and component-level design of the system being developed. This chapter describes how the system will be structured and how its various components will interact to meet the requirements identified earlier. It provides a detailed view of the system's functionality, using various diagrams and models to illustrate the design concepts.

3.1 Architectural Diagram

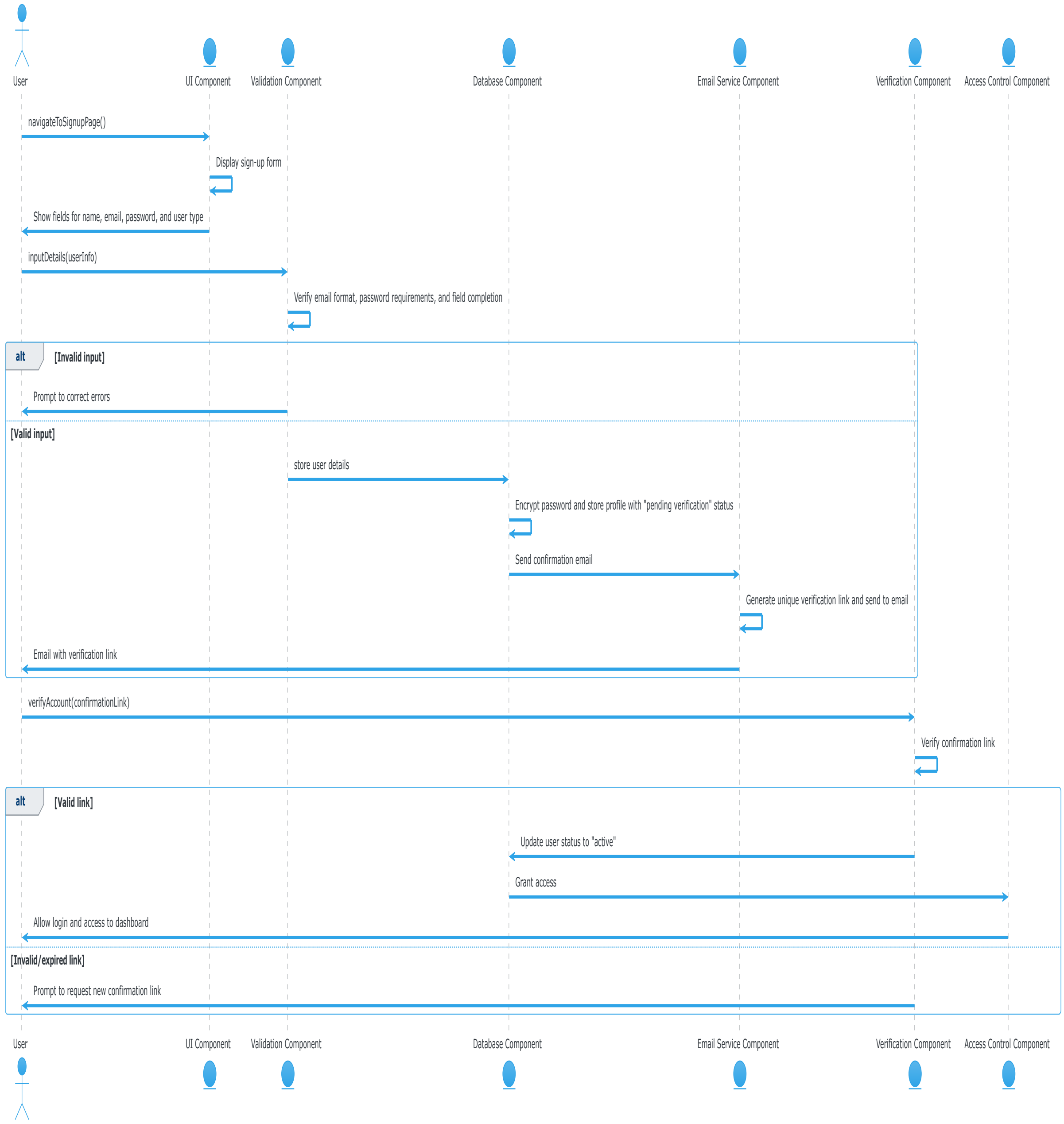
The Architectural Diagram gives a clear, overall view of the system's structure. It highlights key parts like the user interface, backend, database, and external services. The diagram shows how data moves through the system and how different modules work together. This helps in understanding the system's scalability, performance, and how it integrates with other services.



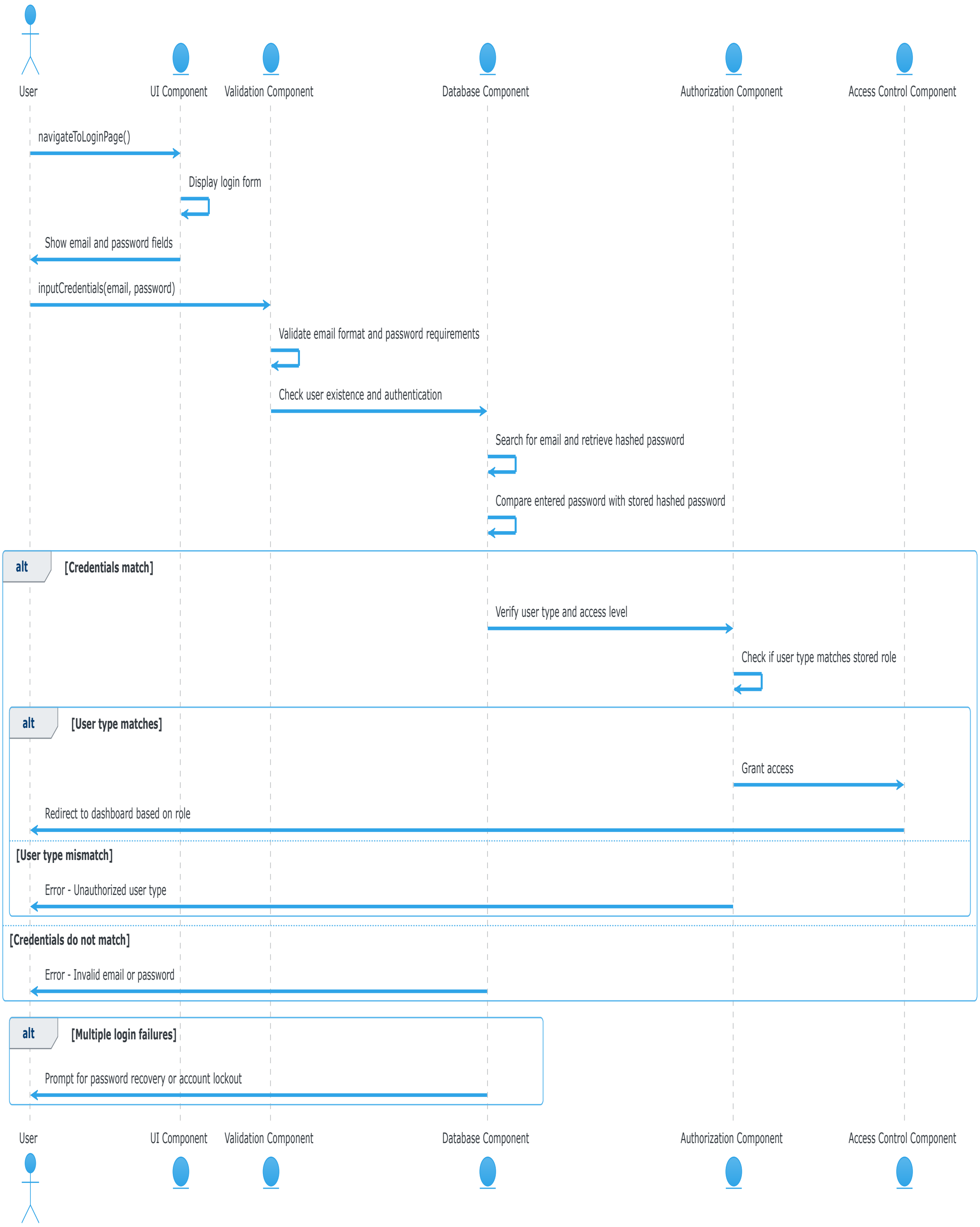
3.2 Sequence Diagrams

Sequence Diagrams are used to visualize the flow of messages between objects in the system during a specific process or scenario. They help to capture the dynamic behavior of the system, showing how objects interact and the sequence of events in response to user inputs or system triggers. These diagrams are essential in identifying object dependencies and the order of operations.

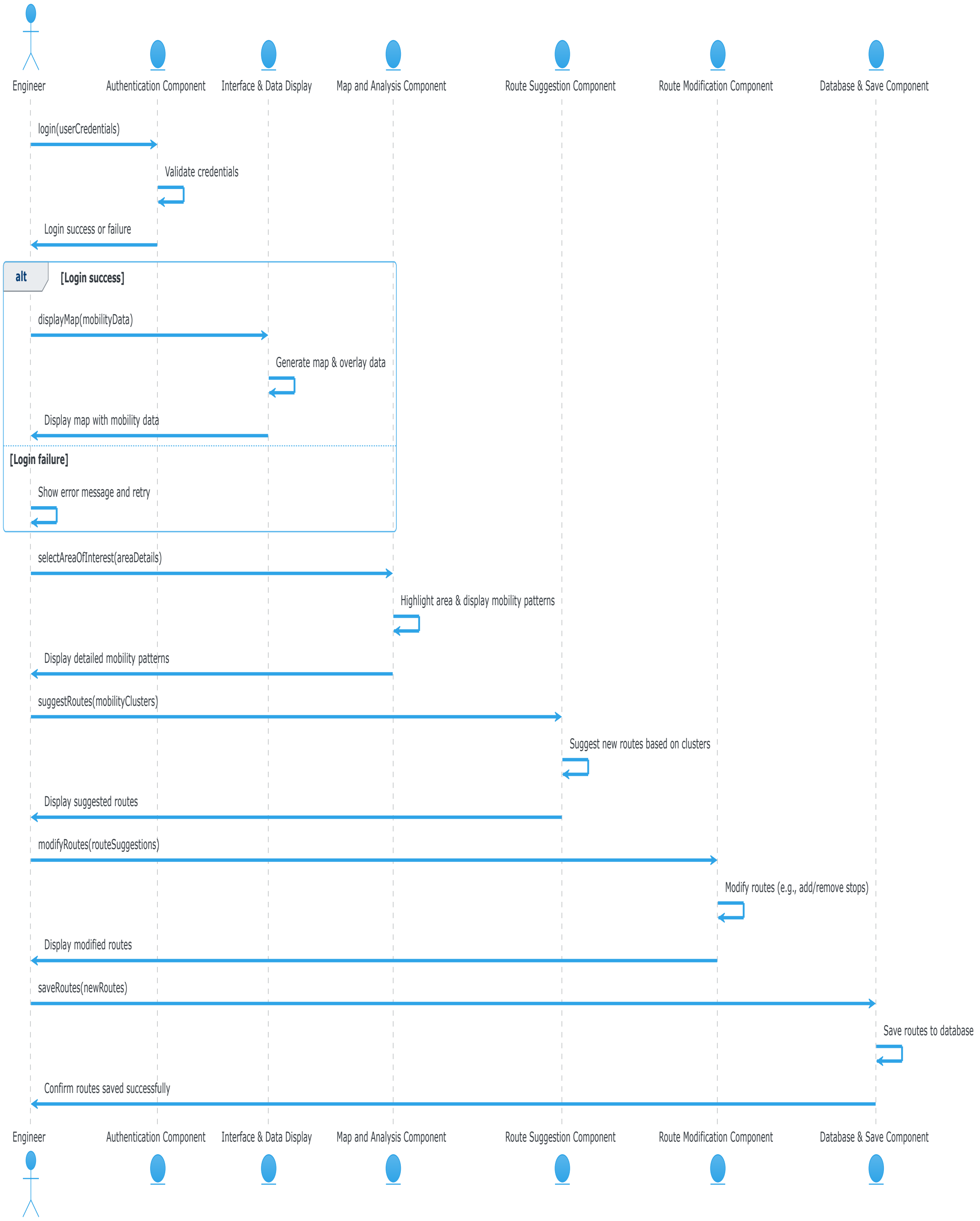
3.2.1 User Authentication - Sign-Up



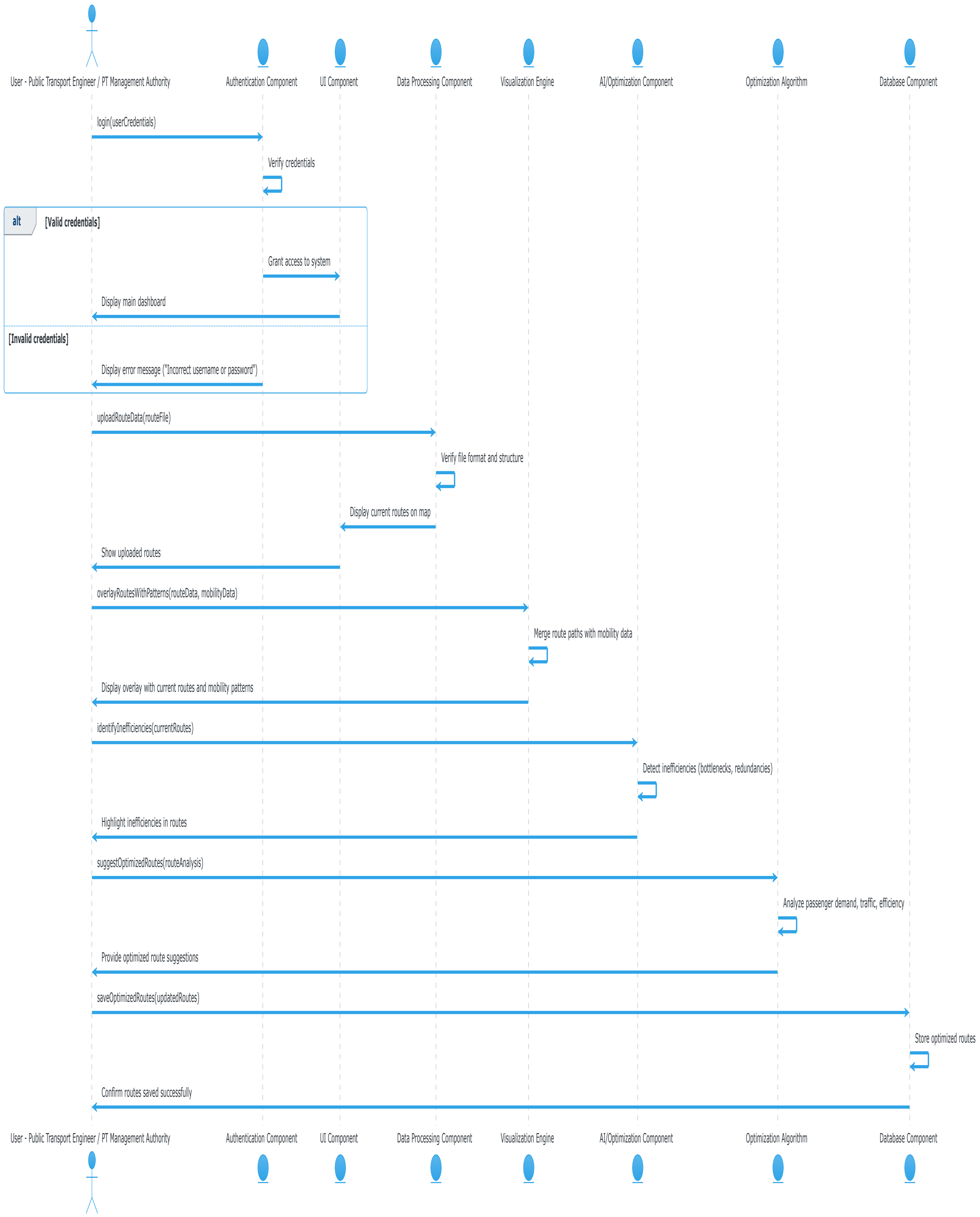
3.2.2 User Authentication - Login


User
UI Component
Validation Component
Database Component
Authorization Component
Access Control Component
navigateToLoginPage()
Display login form
Show email and password fields
inputCredentials(email, password)
Validate email format and password requirements
Check user existence and authentication
Search for email and retrieve hashed password
Compare entered password with stored hashed password
alt
[Credentials match]
Verify user type and access level
Check if user type matches stored role
alt
[User type matches]
Grant access
Redirect to dashboard based on role
[User type mismatch]
Error - Unauthorized user type
[Credentials do not match]
Error - Invalid email or password
alt
[Multiple login failures]
Prompt for password recovery or account lockout
User
UI Component
Validation Component
Database Component
Authorization Component
Access Control Component

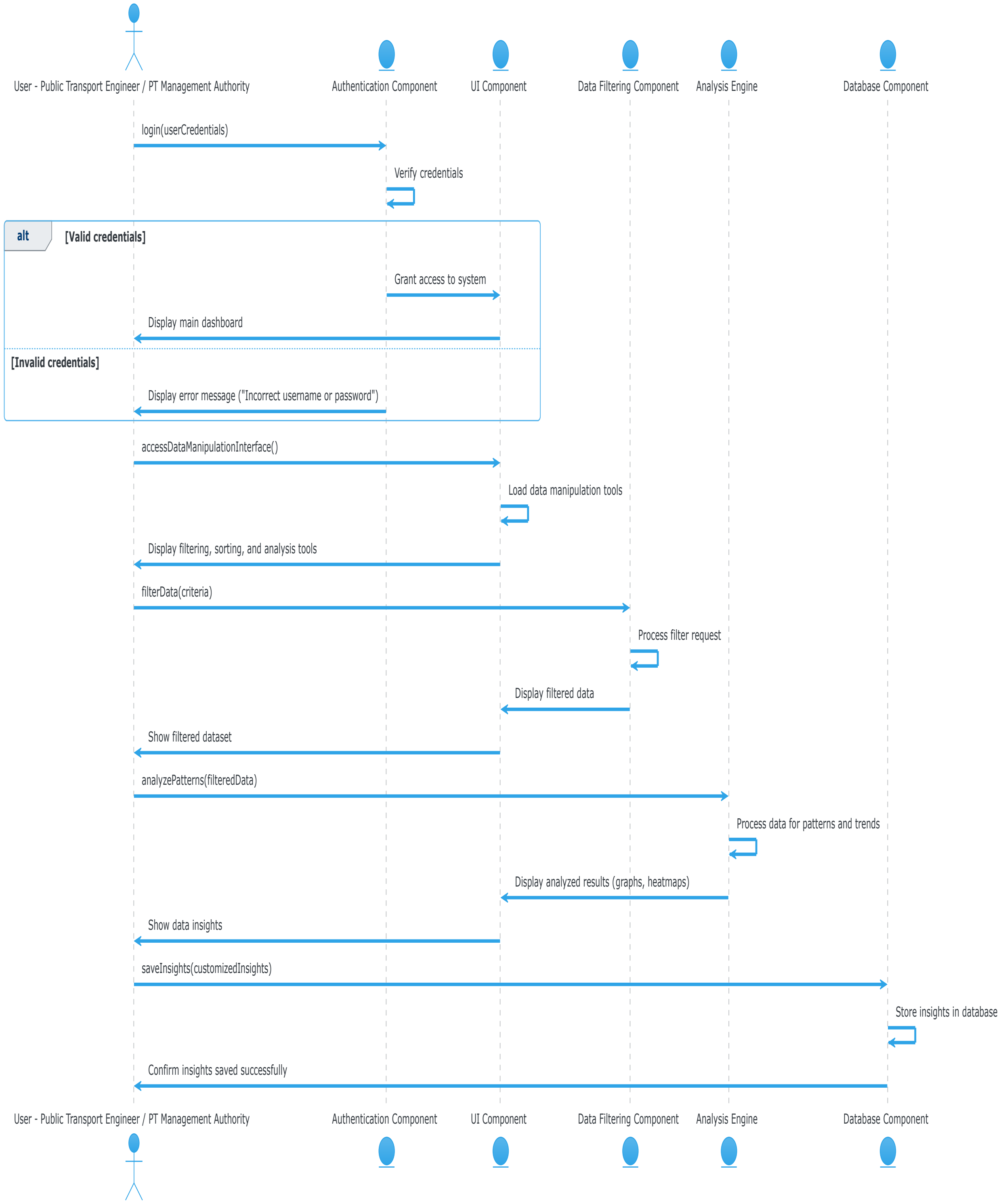
3.2.3 Route Creation


Engineer
Authentication Component
Interface & Data Display
Map and Analysis Component
Route Suggestion Component
Route Modification Component
Database & Save Component
login(userCredentials)
Validate credentials
Login success or failure
alt
[Login success]
displayMap(mobilityData)
Generate map & overlay data
Display map with mobility data
[Login failure]
Show error message and retry
selectAreaOfInterest(areaDetails)
Highlight area & display mobility patterns
Display detailed mobility patterns
suggestRoutes(mobilityClusters)
Suggest new routes based on clusters
Display suggested routes
modifyRoutes(routeSuggestions)
Modify routes (e.g., add/remove stops)
Display modified routes
saveRoutes(newRoutes)
Save routes to database
Confirm routes saved successfully
Engineer
Authentication Component
Interface & Data Display
Map and Analysis Component
Route Suggestion Component
Route Modification Component
Database & Save Component

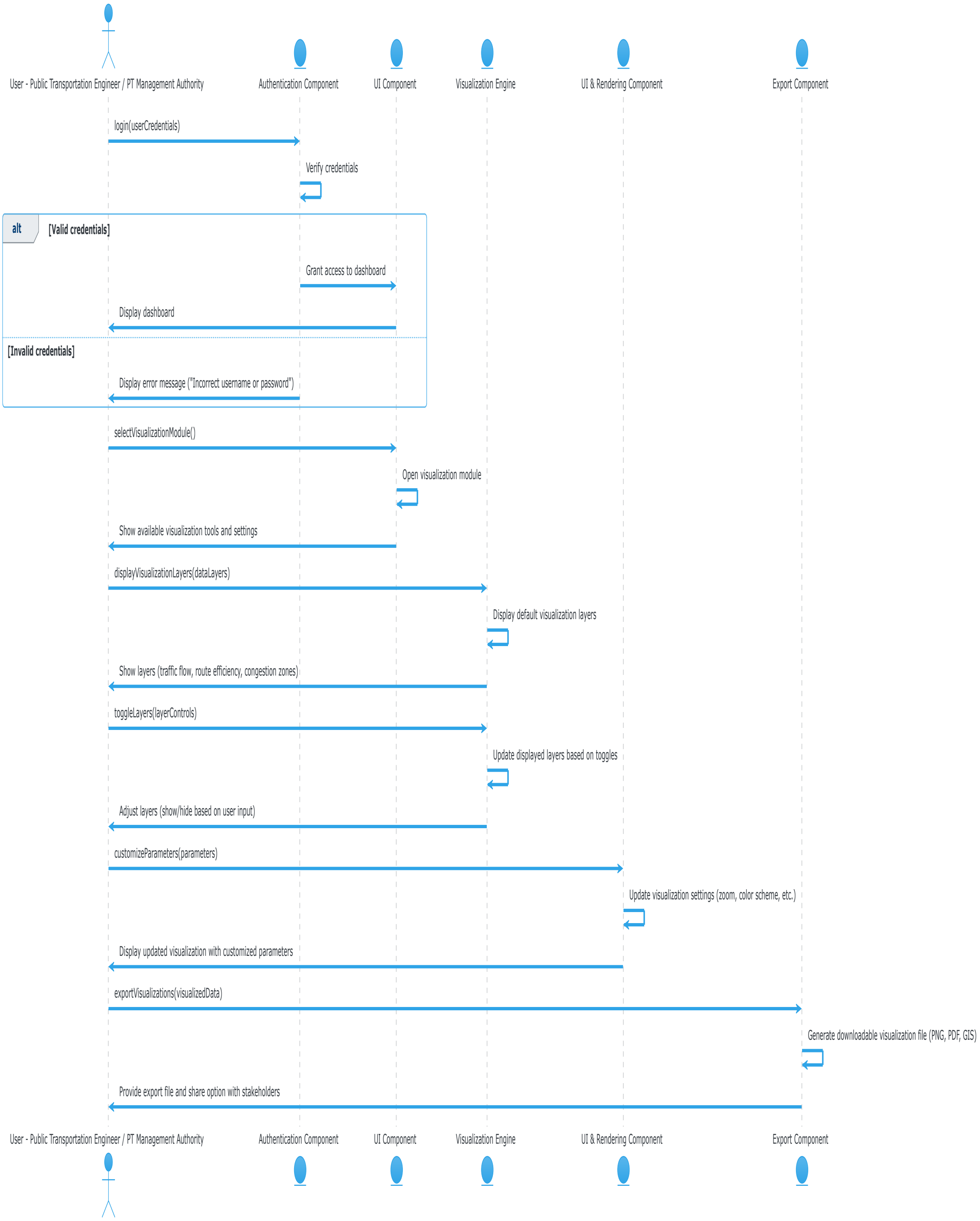
3.2.4 Route Optimization


User - Public Transport Engineer / PT Management Authority
Authentication Component
UI Component
Data Processing Component
Visualization Engine
AI/Optimization Component
Optimization Algorithm
Database Component
login(userCredentials)
Verify credentials
alt
[Valid credentials]
Grant access to system
Display main dashboard
[Invalid credentials]
Display error message ("Incorrect username or password")
uploadRouteData(routeFile)
Verify file format and structure
Display current routes on map
Show uploaded routes
overlayRoutesWithPatterns(routeData, mobilityData)
Merge route paths with mobility data
Display overlay with current routes and mobility patterns
identifyInefficiencies(currentRoutes)
Detect inefficiencies (bottlenecks, redundancies)
Highlight inefficiencies in routes
suggestOptimizedRoutes(routeAnalysis)
Analyze passenger demand, traffic, efficiency
Provide optimized route suggestions
saveOptimizedRoutes(updatedRoutes)
Store optimized routes
Confirm routes saved successfully
User - Public Transport Engineer / PT Management Authority
Authentication Component
UI Component
Data Processing Component
Visualization Engine
AI/Optimization Component
Optimization Algorithm
Database Component

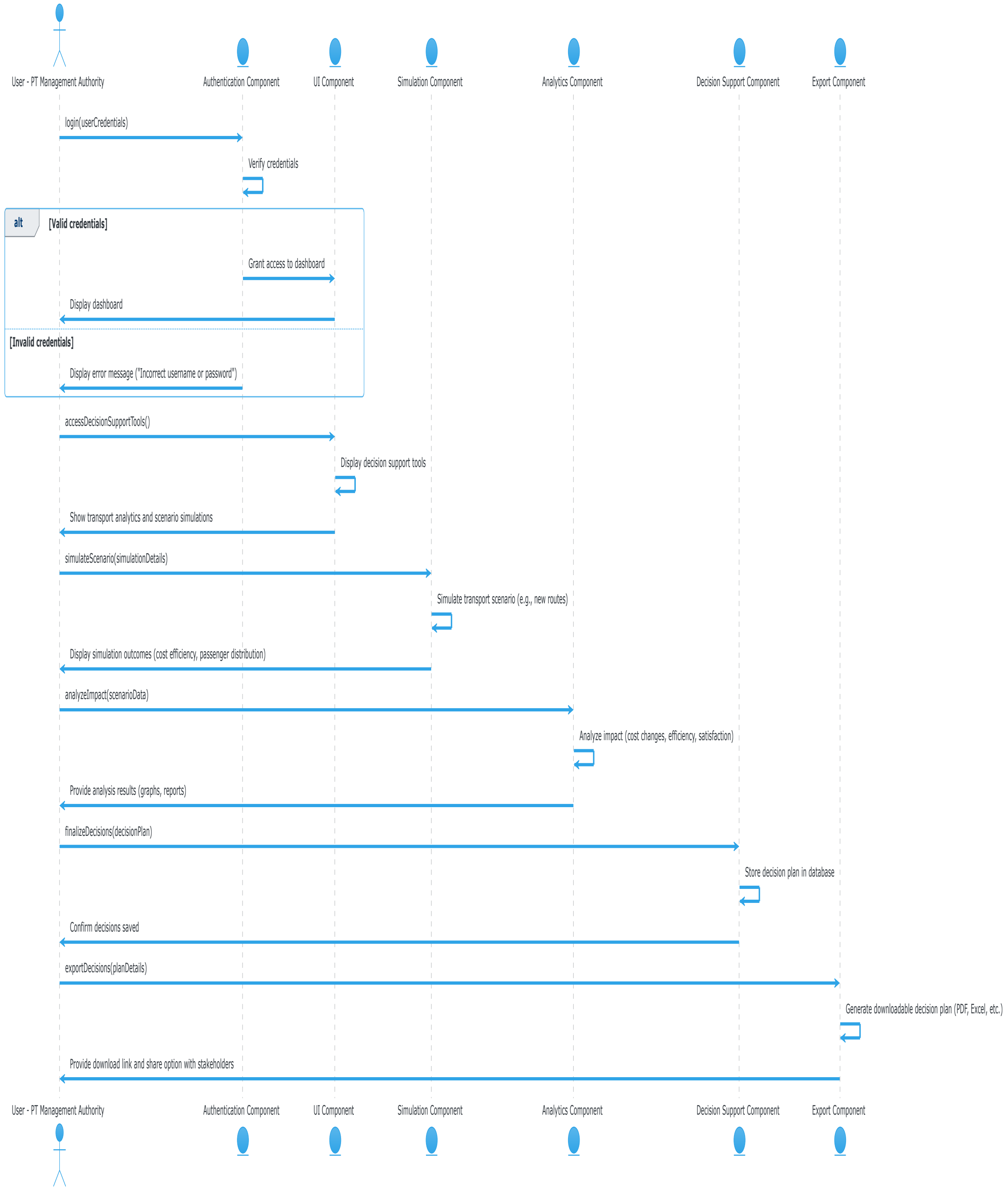
3.2.5 Data Manipulation


User - Public Transport Engineer / PT Management Authority
Authentication Component
UI Component
Data Filtering Component
Analysis Engine
Database Component
login(userCredentials)
Verify credentials
alt
[Valid credentials]
Grant access to system
Display main dashboard
[Invalid credentials]
Display error message ("Incorrect username or password")
accessDataManipulationInterface()
Load data manipulation tools
Display filtering, sorting, and analysis tools
filterData(criteria)
Process filter request
Display filtered data
Show filtered dataset
analyzePatterns(filteredData)
Process data for patterns and trends
Display analyzed results (graphs, heatmaps)
Show data insights
saveInsights(customizedInsights)
Store insights in database
Confirm insights saved successfully
User - Public Transport Engineer / PT Management Authority
Authentication Component
UI Component
Data Filtering Component
Analysis Engine
Database Component

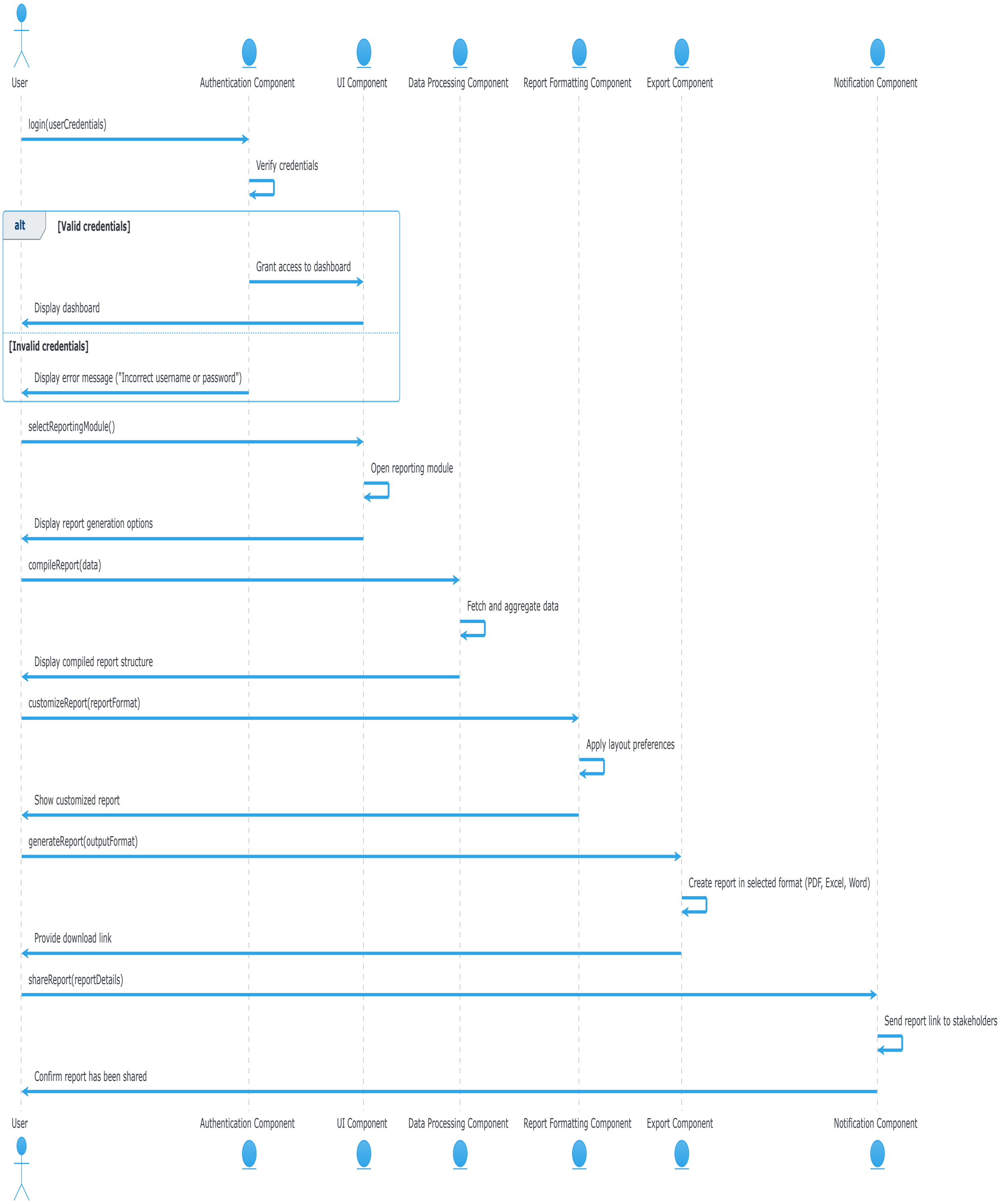
3.2.6 Layered Visualization


User - Public Transportation Engineer / PT Management Authority
Authentication Component
UI Component
Visualization Engine
UI & Rendering Component
Export Component
login(userCredentials)
Verify credentials
alt
[Valid credentials]
Grant access to dashboard
Display dashboard
[Invalid credentials]
Display error message ("Incorrect username or password")
selectVisualizationModule()
Open visualization module
Show available visualization tools and settings
displayVisualizationLayers(dataLayers)
Display default visualization layers
Show layers (traffic flow, route efficiency, congestion zones)
toggleLayers(layerControls)
Update displayed layers based on toggles
Adjust layers (show/hide based on user input)
customizeParameters(parameters)
Update visualization settings (zoom, color scheme, etc.)
Display updated visualization with customized parameters
exportVisualizations(visualizedData)
Generate downloadable visualization file (PNG, PDF, GIS)
Provide export file and share option with stakeholders
User - Public Transportation Engineer / PT Management Authority
Authentication Component
UI Component
Visualization Engine
UI & Rendering Component
Export Component

3.2.7 Service Planning Support

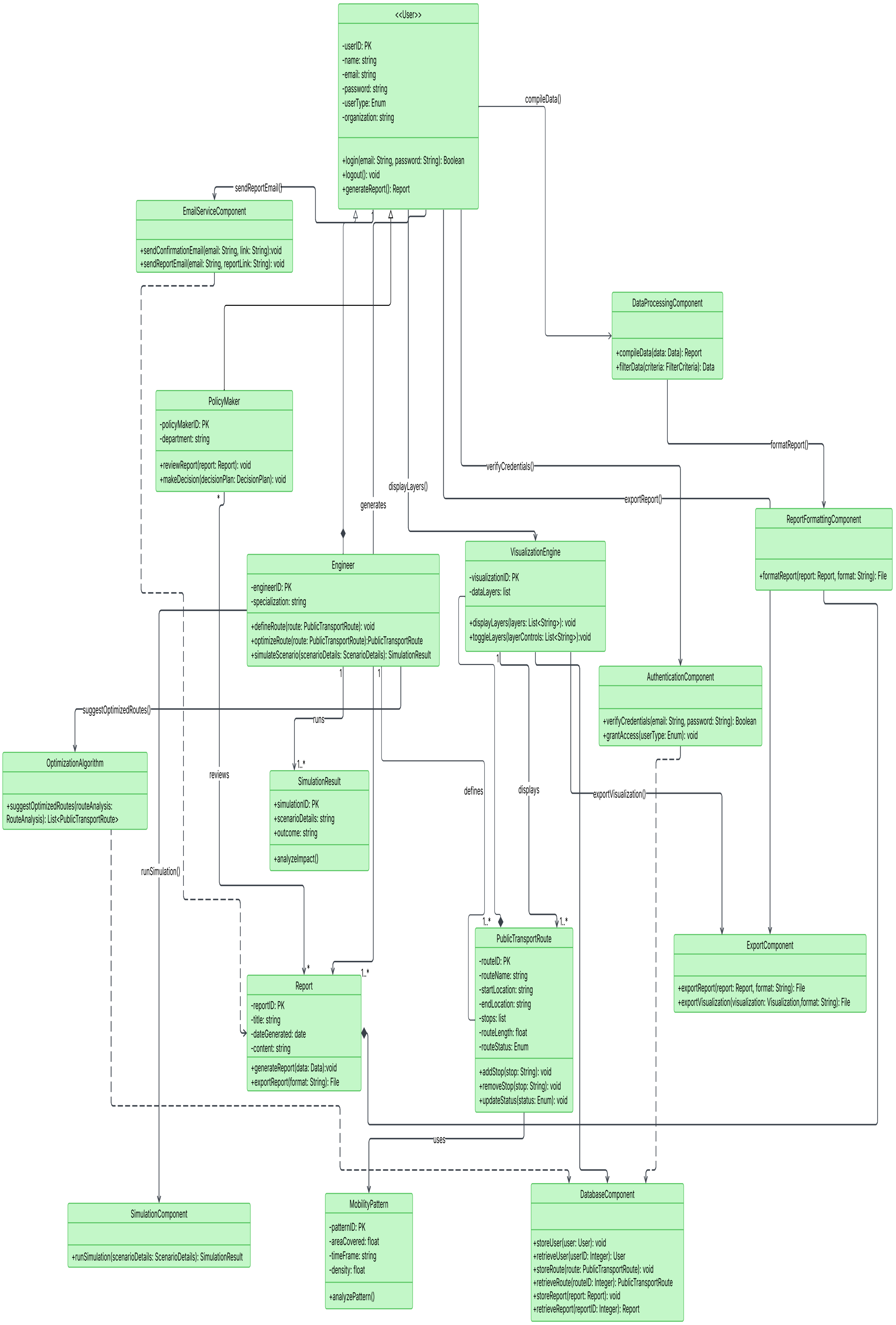

User - PT Management Authority
Authentication Component
UI Component
Simulation Component
Analytics Component
Decision Support Component
Export Component
login(userCredentials)
Verify credentials
alt
[Valid credentials]
Grant access to dashboard
Display dashboard
[Invalid credentials]
Display error message ("Incorrect username or password")
accessDecisionSupportTools()
Display decision support tools
Show transport analytics and scenario simulations
simulateScenario(simulationDetails)
Simulate transport scenario (e.g., new routes)
Display simulation outcomes (cost efficiency, passenger distribution)
analyzeImpact(scenarioData)
Analyze impact (cost changes, efficiency, satisfaction)
Provide analysis results (graphs, reports)
finalizeDecisions(decisionPlan)
Store decision plan in database
Confirm decisions saved
exportDecisions(planDetails)
Generate downloadable decision plan (PDF, Excel, etc.)
Provide download link and share option with stakeholders
User - PT Management Authority
Authentication Component
UI Component
Simulation Component
Analytics Component
Decision Support Component
Export Component

3.2.8 Report Generation


User
Authentication Component
UI Component
Data Processing Component
Report Formatting Component
Export Component
Notification Component
login(userCredentials)
Verify credentials
alt
[Valid credentials]
Grant access to dashboard
Display dashboard
[Invalid credentials]
Display error message ("Incorrect username or password")
selectReportingModule()
Open reporting module
Display report generation options
compileReport(data)
Fetch and aggregate data
Display compiled report structure
customizeReport(reportFormat)
Apply layout preferences
Show customized report
generateReport(outputFormat)
Create report in selected format (PDF, Excel, Word)
Provide download link
shareReport(reportDetails)
Send report link to stakeholders
Confirm report has been shared
User
Authentication Component
UI Component
Data Processing Component
Report Formatting Component
Export Component
Notification Component

3.3 Class Diagram

Class Diagrams describe the relationships between the different parts of your software system. They help you understand the "who’s who" and how they interact, much like a blueprint guides the construction of a building. Essentially, Class Diagrams provide a clear and organized way to design your software.



<<User>>
-userID: PK
-name: string
-email: string
-password: string
-userType: Enum
-organization: string
+login(email: String, password: String): Boolean
+logout(): void
+generateReport(): Report
compileData()
sendReportEmail()
EmailServiceComponent
+sendConfirmationEmail(email: String, link: String):void
+sendReportEmail(email: String, reportLink: String): void
DataProcessingComponent
+compileData(data: Data): Report
+filterData(criteria: FilterCriteria): Data
PolicyMaker
-policyMakerID: PK
-department: string
+reviewReport(report: Report): void
+makeDecision(decisionPlan: DecisionPlan): void
formatReport()
verifyCredentials()
displayLayers()
exportReport()
generates
*
ReportFormattingComponent
+formatReport(report: Report, format: String): File
Engineer
-engineerID: PK
-specialization: string
+defineRoute(route: PublicTransportRoute): void
+optimizeRoute(route: PublicTransportRoute):PublicTransportRoute
+simulateScenario(scenarioDetails: ScenarioDetails): SimulationResult
VisualizationEngine
-visualizationID: PK
-dataLayers: list
+displayLayers(layers: List<String>): void
+toggleLayers(layerControls: List<String>):void
1
1
1
AuthenticationComponent
+verifyCredentials(email: String, password: String): Boolean
+grantAccess(userType: Enum): void
suggestOptimizedRoutes()
runs
1..*
OptimizationAlgorithm
+suggestOptimizedRoutes(routeAnalysis: RouteAnalysis): List<PublicTransportRoute>
reviews
SimulationResult
+simulationID: PK
+scenarioDetails: string
+outcome: string
+analyzeImpact()
defines
displays
exportVisualization()
runSimulation()
1..*
1..*
PublicTransportRoute
-routeID: PK
-routeName: string
-startLocation: string
-endLocation: string
-stops: list
-routeLength: float
-routeStatus: Enum
+addStop(stop: String): void
+removeStop(stop: String): void
+updateStatus(status: Enum): void
ExportComponent
+exportReport(report: Report, format: String): File
+exportVisualization(visualization: Visualization,format: String): File
*
1..*
Report
-reportID: PK
-title: string
-dateGenerated: date
-content: string
+generateReport(data: Data):void
+exportReport(format: String): File
uses
SimulationComponent
+runSimulation(scenarioDetails: ScenarioDetails): SimulationResult
MobilityPattern
-patternID: PK
-areaCovered: float
-timeFrame: string
-density: float
+analyzePattern()
DatabaseComponent
+storeUser(user: User): void
+retrieveUser(userID: Integer): User
+storeRoute(route: PublicTransportRoute): void
+retrieveRoute(routeID: Integer): PublicTransportRoute
+storeReport(report: Report): void
+retrieveReport(reportID: Integer): Report

Chapter 4

Implementation

4.1 System Overview

This system is designed to help engineers and policymakers optimize routes and analyze mobility patterns. It is built with modern technologies and follows a layered architecture to ensure scalability, efficiency, and ease of use.

4.2 Key Components

4.2.1 Frontend (The User Interface)

This is where users interact with the system. It is sleek, responsive, and easy to navigate, offering a seamless experience across devices. The design focuses on simplicity and intuitive layouts, ensuring users can quickly find what they need without confusion.

Features:

- **Login Page**: Secure login using Google accounts via Google Auth API.
- **Dashboard**: Displays reports, routes, and statistics in a clean, visual format.
- **Map View**: Integrates Google Maps API to visualize routes and optimized paths.
- **Optimization Page**: Engineers can input route details and request optimizations.
- **Reports Page**: Fetches and displays detailed reports generated by the backend.

Tech Stack: React.js, Tailwind CSS / Bootstrap

4.2.2 Backend (The Brain)

Handles all logic, communicates with the frontend, and coordinates with the optimization engine. It manages data processing, user requests, and ensures smooth integration with databases and external services. The backend ensures efficient operations and data flow, making sure the system functions seamlessly behind the scenes.

Key Functions:

- **Authentication**: Validates user logins via Google OAuth API.
- **Route Management**: Engineers can create and store new routes in the database.

- **Optimization Requests**: Triggers the Python optimization engine upon user request.

- **Report Generation**: Fetches data from the database and generates reports.

Tech Stack: Node.js, Express.js

4.2.3 Optimization Engine (The Problem Solver)

Analyzes data, optimizes routes, and provides actionable insights. By using advanced algorithms and machine learning techniques, it continuously improves route suggestions, ensuring the most efficient paths are selected. It adapts to real-time data, such as traffic and road conditions, to offer dynamic and precise optimization for better decision-making.

Key Scripts:

- **Route Optimizer**: Uses Google Maps API to determine the shortest, most efficient paths.

- **Mobility Analyzer**: Analyzes population density and transit demand for route improvement.

- **Report Generator**: Converts optimization results into understandable reports.

Tech Stack: Python, AI/ML Libraries (NumPy, SciPy, Pandas, TensorFlow)

4.2.4 Database (The Memory Bank)

Stores all necessary data for system functionality. It securely stores user information, historical data, route optimization results, and other critical records. The database ensures data integrity, easy retrieval, and smooth integration with the backend and optimization engine for seamless operation.
Stored Data:

- **Users**: Profiles, roles, and authentication details.
- **Routes**: Information on routes, stops, distances, and optimization status.
- **Reports**: Policy and engineering reports generated by the system.
- **Mobility Patterns**: Traffic and transit data for analysis.

Tech Stack: MongoDB (NoSQL)

4.3 How It All Works Together

1. **User Login**: A user logs in via Google Auth API on the React frontend.
2. **Authentication**: The backend validates the login and grants access.
3. **Route Creation**: An engineer adds a new route, stored in MongoDB.
4. **Optimization Request**: The backend triggers the Python optimization engine upon request.
5. **Data Analysis**: The Python engine analyzes mobility data, integrates Google Maps API insights, and optimizes routes.
6. **Results Storage**: The optimized route is stored in MongoDB.

7. **Frontend Update**: The frontend fetches the updated route and displays it using Google Maps.

8. **Report Generation**: Reports are generated and displayed for policymakers and engineers.

4.4 Tech Stack Summary

Layer	**Technologies**
Frontend (View)	React.js, Tailwind CSS, Bootstrap, Google Maps API
Backend (Controller)	Node.js, Express.js, REST APIs
Optimization Engine	Python, NumPy, SciPy, TensorFlow, Google Maps API
Database	MongoDB

Chapter 5

Testing

Testing is an essential step after software is developed. IBM describes software testing as "the process of evaluating and verifying that a software product or application does what it's supposed to do." Simply put, testing means running the software to find bugs or errors and ensure the software works well, is functional, and performs as expected.

The main goal of testing is to identify bugs and improve the software's quality and performance. While thorough testing can uncover many issues, it's important to note that no system can be completely error-free. Using various testing methods can help reduce the number of errors in the software.

5.1 Testing Technique

For this software, the testing method used is UAT (User Acceptance Testing). This testing is done after implementing a specific feature or module to check if it meets the required functionality outlined in the project's initial planning. UAT is carried out for each use case defined during the requirements analysis phase.

5.2 Test Cases

5.2.1 UAT Test Case: User Authentication - Sign-Up

Test Case ID	UAT-01
Test Title	User Sign-Up Functionality
Test Description	Verify that a new user can create an account and receive a confirmation email for verification.
Preconditions	The system is accessible via a web interface.
Test Steps	1. Navigate to the Sign-Up page. 2. Enter Name, Email, Password, and select User Type (e.g., Engineer). 3. Submit the details. 4. System validates inputs (correct email format, password strength, etc.). 5. Account is created and confirmation email is sent. 6. Click the verification link in the email to confirm the account.

Expected Results	• User account is created, and a confirmation email is sent. • User verifies account through email link.
Pass/Fail Criteria	• **Pass:** Account is created and verified successfully. • **Fail:** Account creation or verification fails.

5.2.2 UAT Test Case: User Authentication - Login

Test Case ID	UAT-02
Test Title	User Login Functionality
Test Description	Verify that users can log in with valid credentials and receive appropriate error messages for invalid credentials.
Preconditions	• User has an existing account. • User can access the login page.
Test Steps	1. Navigate to the Login page. 2. Enter valid Email and Password. 3. Select User Type (if applicable). 4. Click Login. 5. If credentials are valid, user is logged in and redirected to the dashboard. 6. If invalid, error message appears and user can retry or recover password. 7. Test Password Recovery (e.g., clicking "Forgot Password" link).

Expected Results	• Valid login: User is logged in and redirected. • Invalid login: Error message prompts user to retry or recover password.
Pass/Fail Criteria	• **Pass:** Successful login and error handling. • **Fail:** Login failure or missing error messages.

5.2.3 UAT Test Case: Defining New Public Transport Routes

Test Case ID	UAT-03
Test Title	Public Transport Route Definition
Test Description	Verify that users can define new public transport routes based on mobility patterns.
Preconditions	• Mobility data is collected and preprocessed. • User has access to TransitNet Optimizer.
Test Steps	1. Log into TransitNet Optimizer. 2. Access the map interface with mobility data. 3. Select an area of interest to analyze movement patterns. 4. System suggests potential routes based on mobility clusters. 5. Modify the suggested routes using interactive tools. 6. Save finalized routes to the system.
Expected Results	• System suggests routes based on mobility patterns. • User can modify and save new routes.

Pass/Fail Criteria	• **Pass:** User successfully defines and saves routes. • **Fail:** System fails to suggest, modify, or store routes.

5.2.4 UAT Test Case: Optimizing Existing Public Transport Routes

Test Case ID	UAT-04
Test Title	Public Transport Route Optimization
Test Description	Verify that users can optimize existing public transport routes based on mobility patterns and travel demand.
Preconditions	• Existing PT route data is uploaded. • Mobility data is available.
Test Steps	1. Upload current PT route data into the system. 2. System overlays routes on mobility visualization. 3. Identify inefficiencies (e.g., redundant stops, low utilization). 4. System suggests optimized routes based on demand. 5. Review and refine the suggested optimizations. 6. Save and export updated routes for implementation.

Expected Results	• System identifies inefficiencies and suggests optimized routes. • User can refine and save optimized routes.
Pass/Fail Criteria	• **Pass:** Routes are successfully optimized and saved. • **Fail:** System fails to analyze, suggest, or save optimized routes.

5.2.5 UAT Test Case: Interactive Data Manipulation

Test Case ID	UAT-05
Test Title	Interactive Mobility Data Analysis
Test Description	Verify that users can manipulate and analyze mobility data interactively to generate customized insights.
Preconditions	Mobility data is accessible in the system.
Test Steps	1. Access the data manipulation interface. 2. Use tools to filter, sort, and visualize data (e.g., time, location, travel frequency). 3. Perform data manipulations to identify patterns or trends. 4. View real-time multi-layer visualizations. 5. Save analyzed data for route planning.
Expected Results	• Users can interactively manipulate data. • System provides real-time visual insights. • Data is saved for future planning.

Pass/Fail Criteria	• **Pass:** Data manipulation, visualization, and saving work correctly. • **Fail:** System fails to process, visualize, or save manipulated data.

5.2.6 UAT Test Case: Multi-Layer Visualization of Results

Test Case ID	UAT-06
Test Title	Multi-Layer Data Visualization
Test Description	Verify that users can view and customize multi-layer visualizations of mobility patterns and optimization results.
Preconditions	Mobility analysis or route optimization results are available.
Test Steps	1. Select the visualization module. 2. View multiple visualization layers (e.g., heatmaps, clusters, route overlays). 3. Toggle layers on/off to focus on specific data. 4. Customize parameters (e.g., zoom, color schemes). 5. Export visualizations for reporting/presentation.
Expected Results	• System displays multi-layer visualizations. • Users can customize and export visual insights.

Pass/Fail Criteria	- **Pass:** Users can toggle, customize, and export visualizations. - **Fail:** System fails to display, customize, or export visual results.

5.2.7 UAT Test Case: Decision Support for PT Service Planning

Test Case ID	UAT-07
Test Title	Decision Support for Public Transport Planning
Test Description	Verify that authorities can use decision support tools to make data-driven PT planning decisions.
Preconditions	All mobility analysis and optimization processes are complete.
Test Steps	1. Log into the system and access decision support tools. 2. View key metrics and visualizations from mobility analysis. 3. Simulate scenarios (e.g., adding/removing routes, adjusting schedules). 4. System provides predictive insights on proposed changes. 5. Finalize and export decisions for implementation.
Expected Results	• System provides accurate metrics, simulations, and predictive insights. • Authorities can finalize and export a data-driven plan.

Pass/Fail Criteria	• **Pass:** System successfully supports data-driven decision-making. • **Fail:** System fails to provide insights, simulate scenarios, or export decisions.

5.2.8 UAT Test Case: Generating Reports for Stakeholders

Test Case ID	UAT-08
Test Title	Generating Reports for Public Transport Stakeholders
Test Description	Verify that users can generate, customize, and share detailed reports summarizing system data and optimizations.
Preconditions	System processes are complete, and data is ready for reporting.
Test Steps	1. Access the reporting module. 2. System compiles data into a structured report format. 3. Customize report layout (e.g., graphs, charts, key metrics). 4. Generate report in PDF or other formats. 5. Share report with stakeholders.
Expected Results	• Reports are generated, customized, and exported successfully. • Stakeholders receive detailed, well-structured reports.

Postconditions	Reports are available for stakeholder review and decision-making.
Pass/Fail Criteria	• **Pass:** System successfully generates and shares reports. • **Fail:** System fails to compile, customize, or export reports.

Chapter 6

Conclusion

This project focuses on developing a Public Transport Route Optimization System to tackle inefficiencies in Islamabad's public transport network. Currently, the system relies on outdated, manual methods for planning routes, which has led to poor network performance, delays, longer travel times, and ineffective service delivery. By integrating modern technologies like machine learning algorithms, GIS mapping, and real-time mobility data, the system optimizes transport routes to enhance service efficiency and reduce congestion.

The core features of the system, such as data upload and management, network visualization, and route optimization, are designed to provide a user-friendly and scalable solution for transport authorities. With detailed data insights, the system offers valuable recommendations that can improve the public transport network, making it more reliable and efficient for commuters.

This project not only addresses the technical aspects of route optimization but also highlights the importance of data-driven decision-making in urban mobility planning. The system includes a comprehensive admin dashboard, interactive visualizations, and real-time route optimization, enabling authorities to make informed decisions, cut down operational costs, and enhance public transport access.

In conclusion, the Public Transport Route Optimization System offers a complete solution to the challenges facing Islamabad's public transport network. It can serve as a model for other cities aiming to improve their transport systems. With future improvements like real-time traffic adaptation and multimodal transport integration, the system will continue to evolve to meet the growing needs of urban mobility. This project not only supports smarter urban planning but also aligns with sustainable transportation goals, benefiting both authorities and commuters alike.

Bibliography

Likaj, B. (2013). Graph-based optimization for transport logistics. *Graph Theory in Logistics*, 12:231–245.

Xiao, M., Chen, L., Feng, H., Peng, Z., and Long, Q. (2024). Smart city public transportation route planning based on multi-objective optimization: A review. *Archives of Computational Methods in Engineering*, 31(6):3351–3375.

Yang, F., Yao, Z., Ding, F., Tan, H., and Ran, B. (2019). Understanding urban mobility pattern with cellular phone data: A case study of residents and travelers in nanjing. *Sustainability*, 11(19):5502.

Çolak, S., Lima, A., et al. (2018). Inferring modes of transportation using mobile phone data. *EPJ Data Science*, 7:1–12.