CRANCSTORM

Software Design Document

Version [3.0]

February 6, 2026

Team Members:

Braydon Lamoureux

Kristopher Tomas

Noelia Canela

Ethan Ferguson

Project Sponsor: Dr. Steven Gerhke

Faculty Mentor: Dr. Ana Paula Chavez

Team Mentor: Bailey Hall

Table of Contents

1. Introduction and Design Context

1.1 Project Context

Urban cycling has grown significantly as cities promote sustainable transportation and reduce vehicle congestion. However, cyclists face increased safety risks due to traffic intensity, inconsistent bike lane infrastructure, and limited access to real-time routing tools tailored specifically to bicycles.

Traditional navigation systems prioritize shortest-distance or fastest-time routes primarily optimized for motor vehicles. These routes may direct cyclists through high-traffic corridors or unsafe intersections.

The CRANCSTORM project addresses this gap by developing a Bicycle Route Navigation System that prioritizes traffic intensity, safety, and rider efficiency.

1.2 Sponsor Context

This project is sponsored by Dr. Steven Gehrke, whose research focuses on urban mobility, traffic systems, and transportation safety. The system aligns with broader transportation research goals aimed at improving cyclist infrastructure and safety outcomes.

The value of this system lies in:

- Supporting safer commuter cycling
- Enabling traffic-aware route decisions
- Demonstrating scalable traffic-based routing logic

1.3 Problem Statement

Current navigation systems:

- Do not prioritize bicycle-specific safety constraints
- Lack real-time traffic weighting for cyclist routing
- Provide limited visibility into traffic intensity metrics

Cyclists must manually interpret traffic conditions, increasing risk and reducing efficiency.

1.4 Solution Vision

CRANCSTORM will deliver a web-based navigation system that:

- Accepts user start and destination inputs
- Computes optimized routes using traffic-weighted pathfinding
- Displays routes visually on a map
- Supports multi-user access
- Maintains up-to-date traffic and route data

The system prioritizes:

- Safety (traffic-aware weighting)
- Performance (fast route generation)
- Scalability (multi-user web access)
- Maintainability (modular service architecture)

1.5 Functional and Non-Functional Requirements

Functional Requirements

- FR1: The system shall compute traffic-weighted routes.
- FR2: The system shall allow authenticated user sessions.
- FR3: The system shall visualize routes on an interactive map.
- FR4: The system shall cache previously computed routes.

Non-Functional Requirements

- NFR1: Route generation shall complete within 2 seconds.
- NFR2: System shall support 100 concurrent users.
- NFR3: Passwords shall be securely hashed.
- NFR4: System uptime target: 99%.

---

2. Implementation Overview

2.1 Product Overview

CRANCSTORM is a web-based client-server application that integrates mapping visualization, route computation, and traffic analysis services.

The product consists of:

- A browser-based user interface
- RESTful backend API services
- Route and traffic data storage
- Caching mechanisms for performance

2.2 Technology Stack and Rationale

Frontend

- React.js with TypeScript
    - Component-based UI development
    - Strong typing for maintainability
    - Efficient state management
- Leaflet or Mapbox
    - Interactive mapping
    - Route visualization capabilities

Backend

- Node.js with Express
    - Lightweight REST API framework
    - High concurrency support

    - Fast JSON-based communication

Database

- MySQL
    - Structured storage of road network and user data
    - ACID compliance for consistency
- Redis
    - In-memory caching for frequently requested routes

Authentication

- Session-based login with hashed passwords (bcrypt)
- Role-based access control for user isolation

Hosting

- Local development server
- Designed for future cloud deployment (AWS, Azure, etc.)

2.3 Project Constraints

- Development timeline limited to a 16-week academic semester
- Limited access to live municipal traffic feeds
- Local development environment during initial implementation
- Scalability limited to moderate concurrent usage during initial deployment

2.4 External Dependencies

- Mapping APIs (Leaflet / Mapbox)
- Potential third-party traffic data providers

- Browser compatibility
- Node.js runtime environment

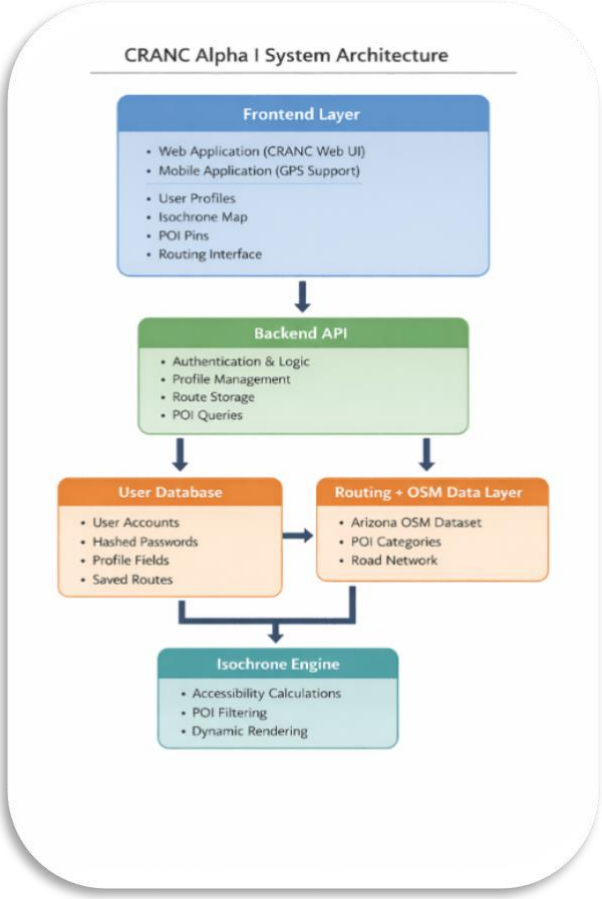
---

3. Architectural Overview and Decisions

3.1 Architectural Style

CRANCSTORM follows a Layered Client-Server Architecture with service modularization.

High-Level Architecture Components:



Client Layer

- React Web Interface
- Map Visualization Component
- Authentication UI

Application Layer (Server/API)

- User Service
- Route Service
- Traffic Service
- Authentication Middleware

Data Layer

- MySQL Database
- Redis Cache

3.2 Request Flow Example

1. User submits start and end location.
2. Client sends POST request to /routes.
3. Route Service:

- Retrieves road graph from database.
- Queries Traffic Service for intensity data.
- Executes traffic-weighted shortest path algorithm.

4. Result is cached in Redis.
5. Route returned to client and rendered on map.

3.3 Key Architectural Decisions

Decision 1: Service Separation

Rationale: Improves maintainability and modular testing.
Trade-off: Slightly increased complexity in communication.

Decision 2: MySQL Over NoSQL

Rationale: Road network data is highly relational.
Trade-off: Less flexible schema evolution.

Decision 3: Redis Caching

Rationale: Route calculations are computationally expensive.
Trade-off: Cache invalidation complexity.

---

4. Component-Level Design

4.1 Web Client

Responsibilities:

- Authentication
- Route input
- Map visualization

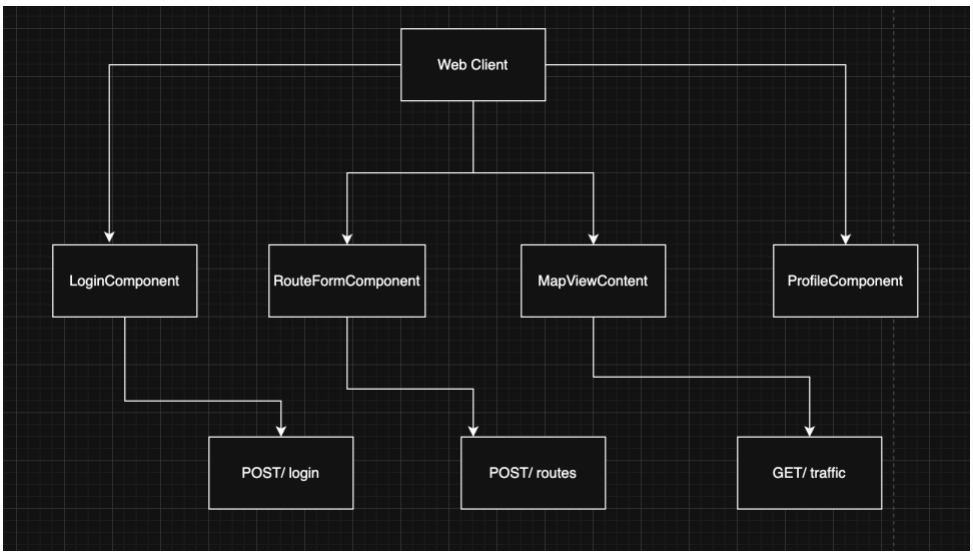
- Display traffic overlays

Key Components:

- LoginComponent
- RouteFormComponent
- MapViewComponent
- ProfileComponent

Public Interface:

- POST /login
- POST /routes
- GET /traffic



4.2 Route Service

Responsibilities:

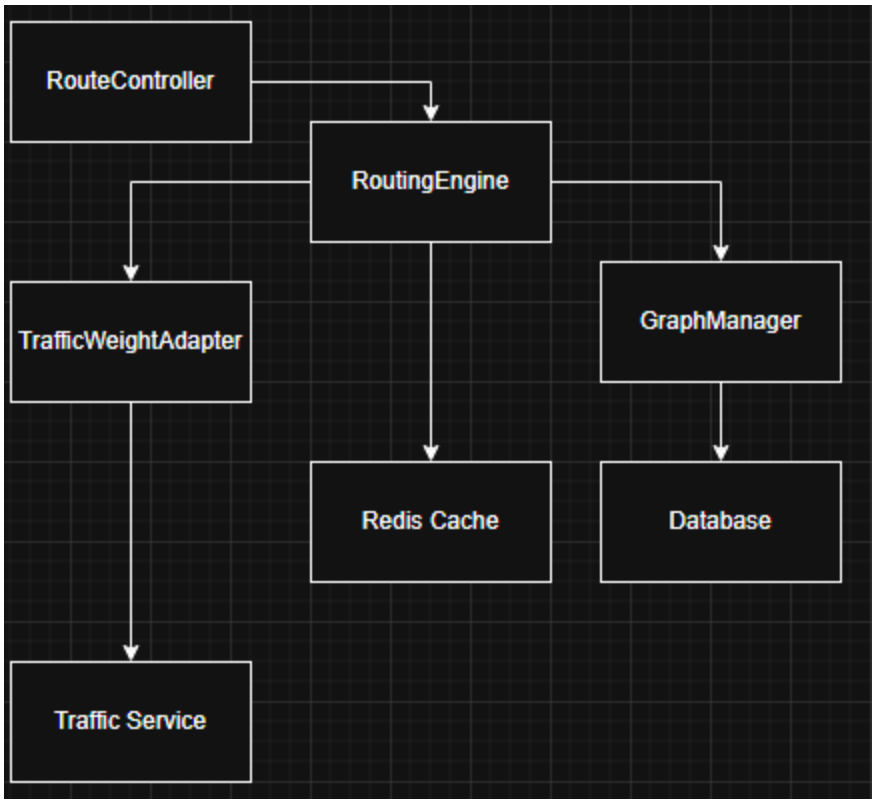
- Graph representation of road network
- Traffic-weighted route calculation
- Route result formatting

Algorithm:

Uses Dijkstra's algorithm with edge weights adjusted by traffic intensity.

Interfaces:

- POST /routes
    - Input: {start, end}
    - Output: Route object with coordinates


RouteController
RoutingEngine
TrafficWeightAdapter
GraphManager
Redis Cache
Database
Traffic Service

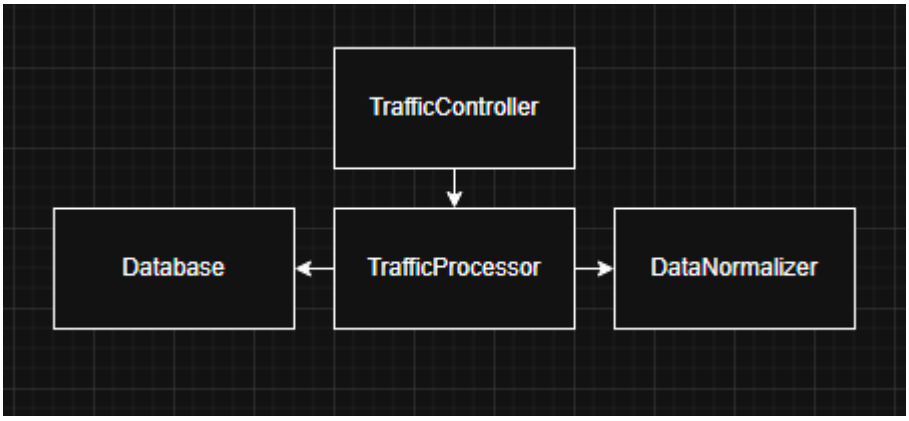
4.3 Traffic Service

Responsibilities:

- Collect traffic data
- Normalize traffic intensity
- Provide weighted factors for routing

Interfaces:

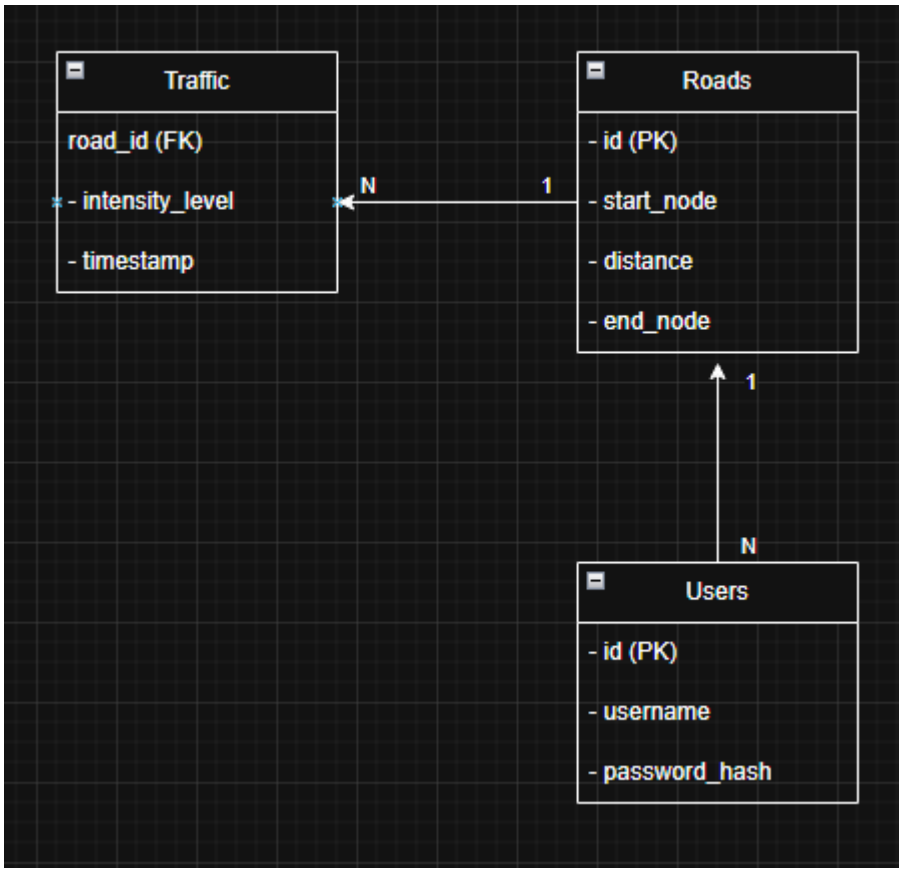
- GET /traffic?area=



4.4 Database Schema Overview

Tables:

- Users (id, username, password_hash)
- Roads (id, start_node, end_node, distance)
- Traffic (road_id, intensity_level, timestamp)



4.5 Cross-Cutting Concerns

Security

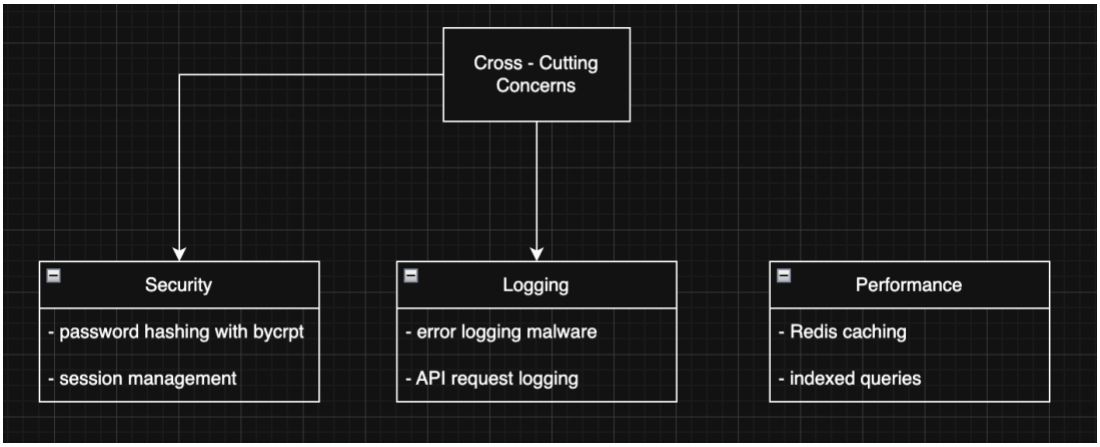
- Password hashing with bcrypt
- Session management
- Input validation

Logging

- Error logging middleware
- API request logging

Performance

- Redis caching
- Indexed database queries



5. Implementation Plan

5.1 Development Phases

Phase	Description	Duration
Phase 1	Environment setup & DB schema	1 week
Phase 2	Route & Traffic Services	2 weeks
Phase 3	Frontend UI Development	2 weeks
Phase 4	Integration Testing	1 week
Phase 5	Performance Optimization & Deployment	1 week

5.2 Execution Strategy

- Parallel frontend/backend development
- Weekly integration checkpoints
- Unit testing for services
- End-to-end testing before deployment

5.3 Risks and Mitigation

Risk: Route Calculation Performance

Mitigation: Caching and optimized graph structure.

Risk: Traffic Data Accuracy

Mitigation: Timestamp-based filtering and validation.

Risk: Session Security

Mitigation: Secure cookies and encryption.

---

6. Conclusion

CRANCSTORM addresses a critical gap in cyclist navigation by introducing traffic-aware route optimization tailored specifically for bicycle commuters. Unlike traditional navigation systems, this design incorporates real-time traffic intensity weighting to improve safety and efficiency.

The system architecture emphasizes modularity, performance, and scalability. Clear component ownership, defined interfaces, and structured implementation phases position the team for successful delivery.

By following this design, the team will produce a maintainable, extensible navigation platform that supports safer urban cycling and aligns with transportation research objectives.