ROUTERMASTER: A Full-Stack Route and Logistics Management System using Java Spring Boot

Ayush Verma[1], Aishwaryjeet Shrivastava[2], Md. Abid[3], Krishnakant Markam[4], Dr. Kavita Chourasia[5], Ms. Kamini Maheshwari[6], Dr. Divakar Singh[7]

[1,2,3,4]CSE Student, BUIT Bhopal, India

[5]Co-Guide, BUIT Bhopal, India

[6]Guide, BUIT Bhopal, India

[7]CSE HOD, BUIT Bhopal, India

DOI: https://dx.doi.org/10.51584/IJRIAS.2026.110200110

Received: 22 February 2026; Accepted: 28 February 2026; Published: 17 March 2026

ABSTRACT

This paper presents *RouterMaster*, a full-stack route and logistics management system developed to streamline trans- portation processes including route creation, vehicle assignment, and delivery monitoring. Traditional route management sys- tems rely heavily on manual coordination, leading to delays, inefficiency, and poor tracking. RouterMaster leverages Java Spring Boot for backend development, MySQL for relational data storage, and a modern web frontend to automate logistics workflows. The system provides secure user authentication, CRUD operations for routes and vehicles, driver assignment management, and a centralized dashboard accessible through REST APIs. The application was tested using Postman and validated for correctness, data integrity, and API compliance. RouterMaster demonstrates how modern enterprise Java tech- nologies can replace manual logistics operations with scalable and efficient digital solutions.

Index Terms: Spring Boot, Java, Route Management, Logistics Automation, RESTful APIs, MySQL, Hibernate, MVC Architec- ture

INTRODUCTION

The logistics and transportation sector has long relied on manual methods such as spreadsheets and paper-based records for route planning and vehicle management. These approaches are inefficient, error-prone, and difficult to scale as organiza- tions grow. The increasing demand for real-time visibility and process automation has made digital transformation essential in logistics operations.

Problem Statement: Existing manual route management systems lack centralized access, real-time update capabilities, and structured tracking, resulting in miscommunication, de- lays, and operational inefficiencies.

Objective: To develop a RESTful web application that enables organizations to digitally manage routes, vehicles, and drivers through a secure and centralized platform, eliminat- ing the dependency on manual record-keeping. RouterMaster addresses these challenges by offering an integrated platform where administrators can add and update routes, assign drivers and vehicles, and monitor operations through a unified dash- board built using Java Spring Boot and MySQL.

LITERATURE REVIEW

Several logistics and fleet management systems have been developed commercially and academically. Enterprise solu- tions such as Oracle Transportation Management and SAP TM offer comprehensive

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February 2026

functionality but are proprietary, expen- sive, and overly complex for small and medium enterprises. Open-source alternatives often lack customization support or developer-friendly architecture.

Research in logistics automation indicates that digitizing route planning processes can reduce operational time by up to 40% and significantly minimize human error [1]. Modern backend frameworks like Spring Boot provide a lightweight yet powerful platform for building scalable REST APIs, while Hibernate ORM simplifies database interaction through object-relational mapping [2]. RouterMaster builds upon these findings to deliver a modular, open-source, and developer- extensible solution tailored for route and logistics management.

METHODOLOGY

Tech Stack:

- Spring Boot (Backend Framework)
- Spring Security (Authentication & Authorization)
- Hibernate ORM / Spring Data JPA (Database Layer)
- MySQL (Relational Database)
- Maven (Dependency Management)
- Postman (API Testing)
- Git & GitHub (Version Control)

Architecture Overview:

- Model-View-Controller (MVC) layered architecture
- Entity-based modeling for Route, Vehicle, Driver, and User
- REST API design with standardized HTTP endpoints
- Centralized exception handling via @ControllerAdvice

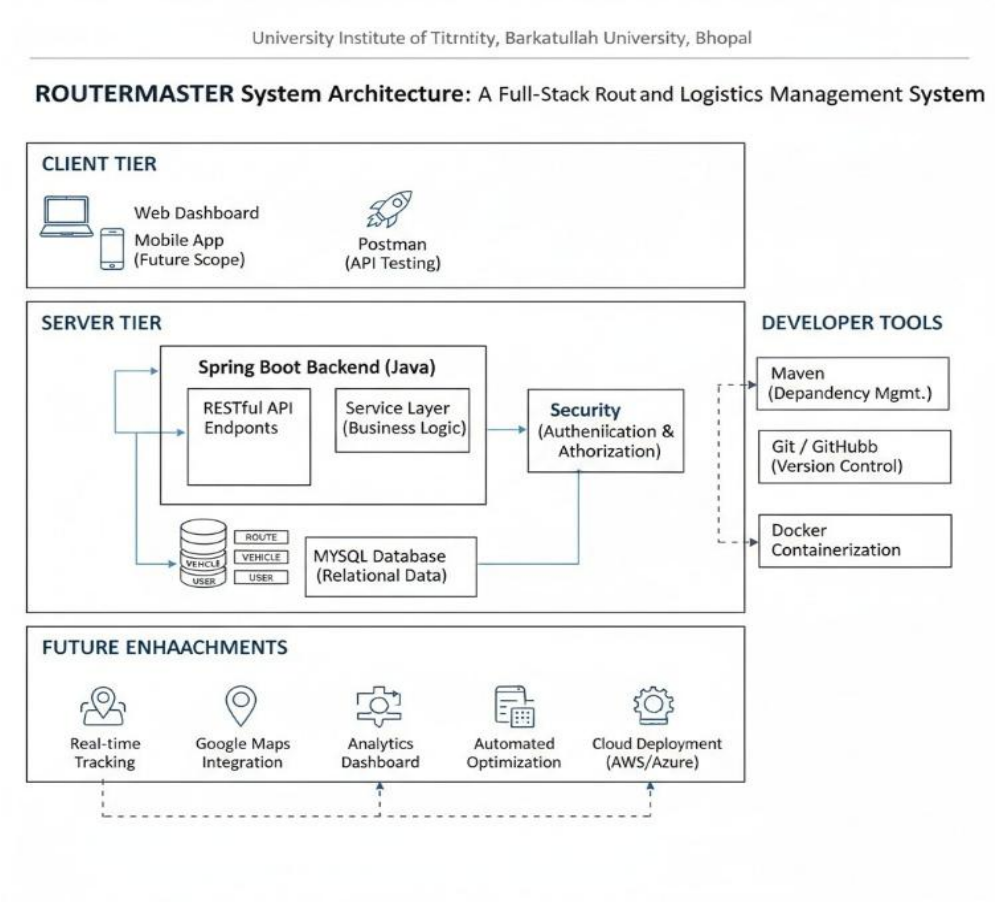

Fig. 1. ROUTERMASTER System Architecture: A Full-Stack Route and Logistics Management System

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February 2026

Development Workflow:

1) Database schema design and normalization
2) Entity mapping using Hibernate annotations
3) Service layer implementation for business logic
4) Controller layer implementation for REST endpoints
5) API testing and validation using Postman
6) Integration testing and debugging

SYSTEM DESIGN & ER MODEL

Entities:

- **User**: Stores authentication credentials and role informa- tion
- **Route**: Stores source, destination, distance, and status
- **Vehicle**: Stores vehicle number, type, and capacity
- **Driver**: Stores driver name, contact details, and assign- ments

Entity Relationships:

- One User can manage many Routes (one-to-many)
- One Driver can be assigned to multiple Routes
- One Vehicle can participate in multiple Route assign- ments
- Route acts as the central entity linking Vehicles and Drivers

The system follows a normalized relational schema ensuring data integrity and minimal redundancy across all logistics entities.

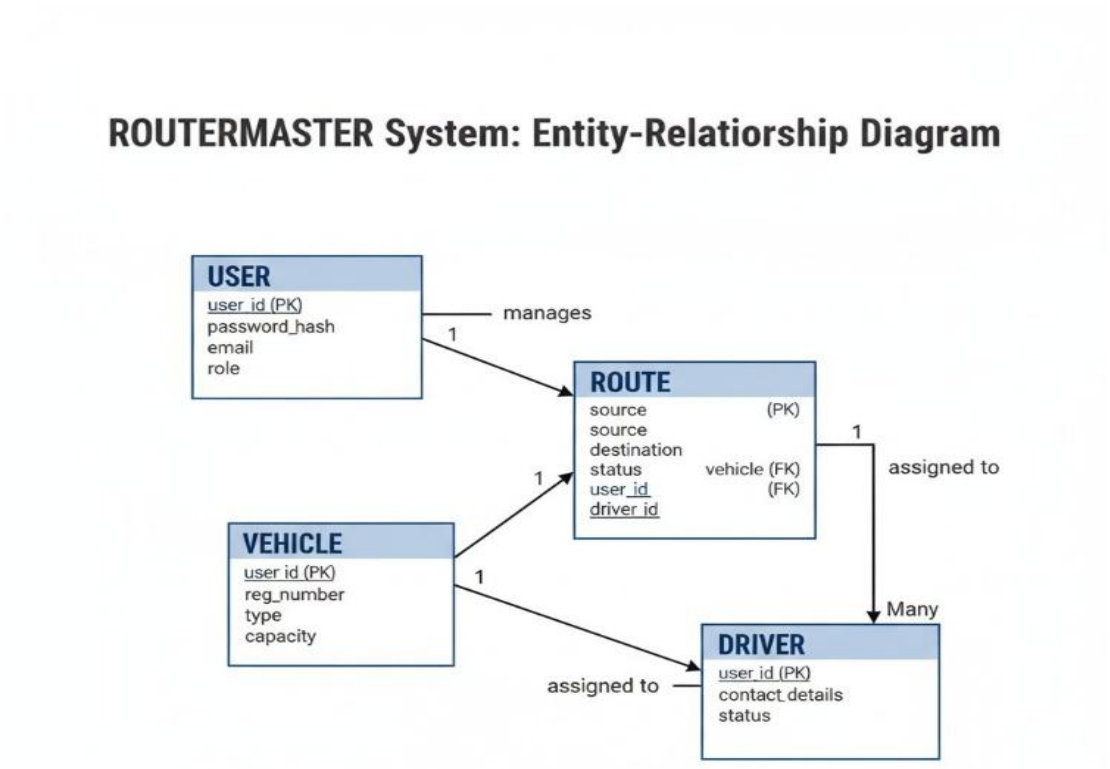

Fig. 2. ROUTERMASTER System: Entity-Relationship Diagram

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February 2026

IMPLEMENTATION DETAILS

RouterMaster is structured using the standard Controller- Service-Repository pattern in Spring Boot. Each module is implemented as an independent Spring component to main- tain clean separation of concerns. The RouteController handles incoming HTTP requests and delegates business logic to the RouteService. The service layer validates input data and enforces business rules such as preventing duplicate route entries. Persistence is managed through Spring Data JPA repositories that interface with the MySQL database. Spring Security is configured to protect all API endpoints, requir- ing authentication before access. Passwords are stored using BCrypt hashing to ensure security. Role-based access control differentiates between admin and standard user permissions.

RESULTS & TESTING

A. API Testing with Postman

All REST endpoints were tested using Postman to verify correctness, response structure, and HTTP status codes:

- POST /api/auth/login — Authenticate user and return session
- GET /api/routes— Retrieve all route records
- POST /api/routes— Create a new route entry
- PUT /api/routes/{id} — Update an existing route
- DELETE /api/routes/{id} — Remove a route record
- GET /api/vehicles— Retrieve all vehicle records
- POST /api/drivers— Add a new driver

B. *Sample API Response*


```
[
  {
    "routeId": 1,
    "source": "Bhopal",
    "destination": "Indore","distance": 195.5,
    "status": "ACTIVE",
    "assignedDriver": "Rajesh Kumar","vehicle": "MP09−AB−1234"
  },
  {
    "routeId": 2,
    "source": "Indore",
```

```
    "destination": "Nagpur","distance": 310.0,
    "status": "PENDING",
    "assignedDriver": "Suresh Yadav","vehicle": "MP09−CD−5678"
  }
]
```


Testing Summary: All endpoints returned valid JSON responses with correct HTTP status codes (200 OK, 201 Created, 404 Not Found, 400 Bad Request). Invalid inputs were rejected with descriptive error messages. Data integrity tests confirmed that routes were stored accurately and that duplicate entries were prevented by the service layer.

DISCUSSION

CONCLUSION & FUTURE SCOPE

RouterMaster successfully demonstrates how logistics oper- ations can be modernized using a full-stack automation system built on Java Spring Boot and MySQL. The system provides secure authentication, complete route lifecycle management, vehicle and driver coordination, and reliable API communica- tion, effectively replacing inefficient manual processes.

Future Enhancements:

- Integration of JWT for stateless token-based authentica- tion
- Real-time GPS tracking for active route monitoring
- Google Maps API integration for visual route planning
- Analytics dashboard with route performance metrics
- Frontend development using React or Angular
- Mobile application for driver-side updates
- Cloud deployment via AWS or Azure
- Docker containerization for environment-independent de- ployment
- Automated route optimization using shortest-path algo- rithms

APPENDIX: SAMPLE CODE SNIPPETS

A. Spring Boot Route Controller

```
@RestController
@RequestMapping("/api/routes")
public class RouteController {
    @Autowired
    private RouteService routeService;
    @GetMapping("/")
    public List getAllRoutes() {
        return routeService.getAllRoutes();
```

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February 2026

RouterMaster confirms that a modular Spring Boot backend can effectively replace manual logistics management sys- tems. The layered architecture ensures maintainability, while the REST API design enables seamless frontend integration. Compared to heavyweight ERP systems, RouterMaster is lightweight, open-source, and easy to customize for specific business needs.

Challenges Faced:

- Designing a flexible schema to support varied route and vehicle relationships

```
}
@PostMapping("/")
public ResponseEntity addRoute(
        @RequestBody Route route) {
    return ResponseEntity.status(201)
        .body(routeService.addRoute(route));
}
```

- Implementing role-based access control without overcom- plicating authentication logic
- Maintaining modular code structure while managing inter-entity dependencies

Advantages of RouterMaster:

- Eliminates manual errors in route and driver management
- Centralized dashboard improves operational visibility
- Scalable architecture supports future feature additions
- REST API design enables integration with any frontend or mobile client
- Open-source and developer-friendly codebase

B. *Hibernate Entity: Route*

```
@Entity
@Table(name = "routes")
public class Route {
    @Id
    @GeneratedValue(
strategy = GenerationType.IDENTITY)
    private Long routeId;
    private String source;
```

```
    private String destination;
    private double distance;
    private String status;
    @ManyToOne
    @JoinColumn(name = "driver_id")
    private Driver assignedDriver;
    @ManyToOne
    @JoinColumn(name = "vehicle_id")
    private Vehicle vehicle;
}
```

REFERENCES

1. R. Kapoor and S. Mehta, *Digital Transformation in Logistics and Supply Chain Management*, Journal of Industrial Engineering, 2021.
2. C. Walls, *Spring Boot in Action*. Manning Publications, 2019.
3. C. Bauer and G. King, *Java Persistence with Hibernate*, 2nd ed. Manning Publications, 2015.
4. L. Richardson and M. Amundsen, *RESTful Web APIs*. O'Reilly Media, 2013.
5. P. DuBois, *MySQL*, 5th ed. Addison-Wesley Professional, 2013.
6. L. Spilca, *Spring Security in Action*. Manning Publications, 2020.