Technology Stack Summary

1. Core Development Technologies

Technology Category	Specific Selection	Version Requirement	Core Purpose
Programming Language	Java	JDK 11+	Develop core system logic, encapsulate TraaS interfaces, implement GUI, and write business logic
Traffic Simulation Engine	SUMO	1.18+	Provide underlying capabilities for road network modeling and traffic flow simulation, serving as the core for simulation data generation and operation
Communication Interface Library	TraaS	1.0	Encapsulate the TraCI protocol to realize bidirectional communication between Java and SUMO (data acquisition, control command issuance)
GUI Development Framework	Java Swing	Built-in (JDK-included)	Develop graphical user interfaces, including road network visualization, simulation control panels, data dashboards, and parameter configuration interfaces
Data Export Tools	Apache Commons CSV + iTextPDF	Commons CSV 1.9.0+, iTextPDF 7.1+	Export simulation data (vehicle speed, traffic flow, traffic light status, etc.) in CSV/PDF formats

2. Development & Collaboration Tools

Tool Category	Specific Selection	Core Purpose
Project Build Tool	Apache Maven	Dependency management, project compilation, packaging and building (generate runnable JAR files)
Version Control Tool	Git + GitHub/GitLab	Code version management, team collaborative development (branch management, code merging), and historical record tracing

Class Diagram Drawing Tool	PlantUML	Draw TraaS wrapper class diagrams and system architecture diagrams to visualize class structures and inter-class relationships
Documentation Tool	Markdown + LibreOffice/Word	Write technical documents, user guides, and project overviews, supporting bilingual format output
Testing & Debugging Tools	IntelliJ IDEA/Eclipse + SUMO GUI	Code development, debugging (breakpoint debugging, log output), and visual verification of simulation processes

3. Core Advantages of the Technology Stack

- Strong Compatibility: The combination of Java + SUMO + TraaS is cross-platform (Windows/Linux/macOS) and can run on mainstream operating systems without additional adaptation.
- High Development Efficiency: Leverages mature frameworks and toolkits (Swing, Maven, TraaS) to reduce repetitive development and focus on core business logic.
- Good Scalability: Modular design and standardized dependency management support subsequent integration of machine learning algorithms (e.g., congestion prediction) and GUI framework upgrades (e.g., JavaFX).
- Outstanding Usability: Avoids the operational complexity of SUMO's native tools, lowers the usage threshold through Java encapsulation, while retaining underlying simulation flexibility.
- Mature Ecosystem: All technical components have comprehensive documentation and community support, reducing costs for problem troubleshooting and function expansion.