<TollBalance>

Software Requirements Specification

Version <Version 2.0>

Revision History

Date	Version	Description	Author
<MM/DD/YYYY>	<version>	<description>	<author>
11/23/2024	1.0	First Composition of SRS Document	Ioannis Dimoulas Enrica Iliana Maggiori Antigoni Maria Karanika Emmanouil Pantelakis
02/12/2025	2.0	Final Composition of SRS Document	Ioannis Dimoulas Enrica Iliana Maggiori Antigoni Maria Karanika Emmanouil Pantelakis

Table of Contents

Introduction

This SRS document will provide a clear and structured description of the system's requirements, use cases, and roles, ensuring alignment between stakeholders and developers. It will serve as a reference for the development, testing, and validation phases of the project.

The document includes the following sections:

- **Overall Description**: A high-level overview of the system, its purpose, scope, objectives, constraints, and primary stakeholders.
- **Functional Requirements**: A detailed description of the system's specific functionalities and behaviors, focusing on use cases and workflows.
- **Technical Requirements**: Specifications for the technologies, frameworks, standards, and performance expectations the system must meet.
- **User Requirements**: Details on the roles, access levels, and tasks expected to be performed by end users.
- **Hardware Requirements**: Minimum and recommended hardware specifications to ensure reliable system operation.
- **Deployment Requirements**: Conditions, dependencies, and strategies for deploying the system in the production environment.

The main focuses of this SRS document are:

1. **Requirement Specification**: Clearly defining the functional and non-functional requirements of the system to ensure all stakeholders share a unified understanding of the deliverables.
2. **Role and Access Definition**: Outlining the roles, and their specific access to system functionalities for secure and efficient usage.
3. **System Functionality**: Clearly describing the core features and capabilities the system must provide to fulfill users needs. It outlines the key actions users can perform, the processes supported by the system, and the expected behaviors to ensure efficient operation.
4. **Technical Standards**: Providing a detailed description of the technologies, performance criteria, and security standards required for development and operation.
5. **Hardware and Deployment Considerations:** Specifying the required hardware resources and deployment environment to support reliable and efficient system operations.

Purpose

The purpose of this Software Requirements Specification (SRS) document is to clearly define the functional and non-functional requirements for the development of TollBalance, a platform designed to provide debts management and traffic statistics. This document serves as a comprehensive guide for stakeholders, ensuring a shared understanding of the system's capabilities, requirements, and expected behaviors throughout its development and implementation.

Scope

The TollBalance platform is designed to provide comprehensive solutions for managing debts and analyzing traffic and debt related data. The software will serve as a tool for companies to track and settle its debts, track amounts owed, and gain insights into traffic patterns, tolls usage, and revenue distribution by various charts and plots that will be offered. It will also provide historical data and analytics for better financial decision-making. The system will be accessible only to authorized company employees.

Definition, Acronyms and Abbreviations

This section provides definitions, acronyms, and abbreviations for terms used throughout this SRS to ensure clarity and prevent misunderstandings.

Term	Definition
API (Application Programming Interface)	A set of protocols and tools for building software applications.

Overall Description

Problem Statement

The problem the "Toll Balance" platform will address involves managing the interoperability of toll collection systems across various highways, which use different automatic tolling systems each managed by a different operator. These systems allow vehicles to pass through toll booths using any compatible transponder from other highways. This interoperability generates financial obligations between toll operators that must be accurately recorded and assigned to the respective operators.

Background

The problem arises due to the need for interoperability between multiple toll systems operated by different companies managing various highways. Each highway has its own toll collection system, but with the increasing mobility of vehicles across different regions, it became necessary to allow vehicles with toll transponders from one highway to pass through toll booths on other highways. This interconnection enables a seamless experience for drivers, allowing them to travel across different toll networks without needing separate transponders or payment systems for each highway.

The absence of a centralized system for managing and reconciling these debts, along with the complexity of tracking toll usage across various networks, creates challenges in accurate billing, reporting, and debt settlement. Moreover, the data generated from these toll transactions holds valuable insights that, if properly analyzed, can benefit operators, regulators, and other stakeholders in making data-driven decisions about traffic flow, infrastructure investment, and service improvements.

Objectives

The "Toll Balance" platform aims to achieve the following key objectives:

Accurate Debt Tracking and Management:

- Enable seamless tracking of financial obligations between toll operators when vehicles cross toll booths from different highways.
- Automatically calculate debts owed between operators, by processing the tolls' crossing data provided by the operators, and correctly assigned to the respective operators.

Debt Settlement Process:

- Facilitate the settlement of debts between toll operators.
- Provide transparency and accountability in the settlement process, allowing operators to review and confirm financial exchanges.

Comprehensive Traffic Data Analytics:

- Collect and analyze data related to toll transactions, vehicle movements, and traffic patterns across highways.
- Generate insightful charts on traffic variation, toll booth usage, and revenue distribution, supporting better decision-making for highway operators and stakeholders.

Data Privacy and Security:

- Ensure that all financial and traffic-related data is securely stored and processed to protect sensitive information from unauthorized access or breaches.
- Implement authentication mechanisms to safeguard the privacy and integrity of user and transaction data.

System Scalability:

- Design the system to be scalable, capable of handling increasing volumes of data and transactions as the system grows and is adopted by more toll operators.

Constraints

1. **Interoperability Limitations:**

   The system depends on consistent and accurate data integration from different toll operators. Discrepancies in data formats or delays in data sharing could impact functionality.

2. **Delayed Data Updates:**

   The system does not provide real-time data but relies on periodic updates from toll operators. Any delays in data submission could result in outdated information being displayed.

3. **Data Dependency:**

   The accuracy and reliability of the system's analytics and debt calculations rely heavily on the quality of data provided by toll operators. Missing or incorrect data could affect outcomes.

4. **No Direct Payment Support:**

   The software does not support direct payments or financial transactions within the system. Debt settlements must be handled externally by the involved operators.

5. **Scalability Limitations:**

   Although designed to scale, the system's performance might degrade if the volume of transactions or users exceeds the capacity of the underlying infrastructure.

6. **Infrastructure Dependency:**

   The system requires reliable internet connectivity and server infrastructure. Any disruptions to these could impact its availability and performance.

Functional Requirements

The system provides the following features:

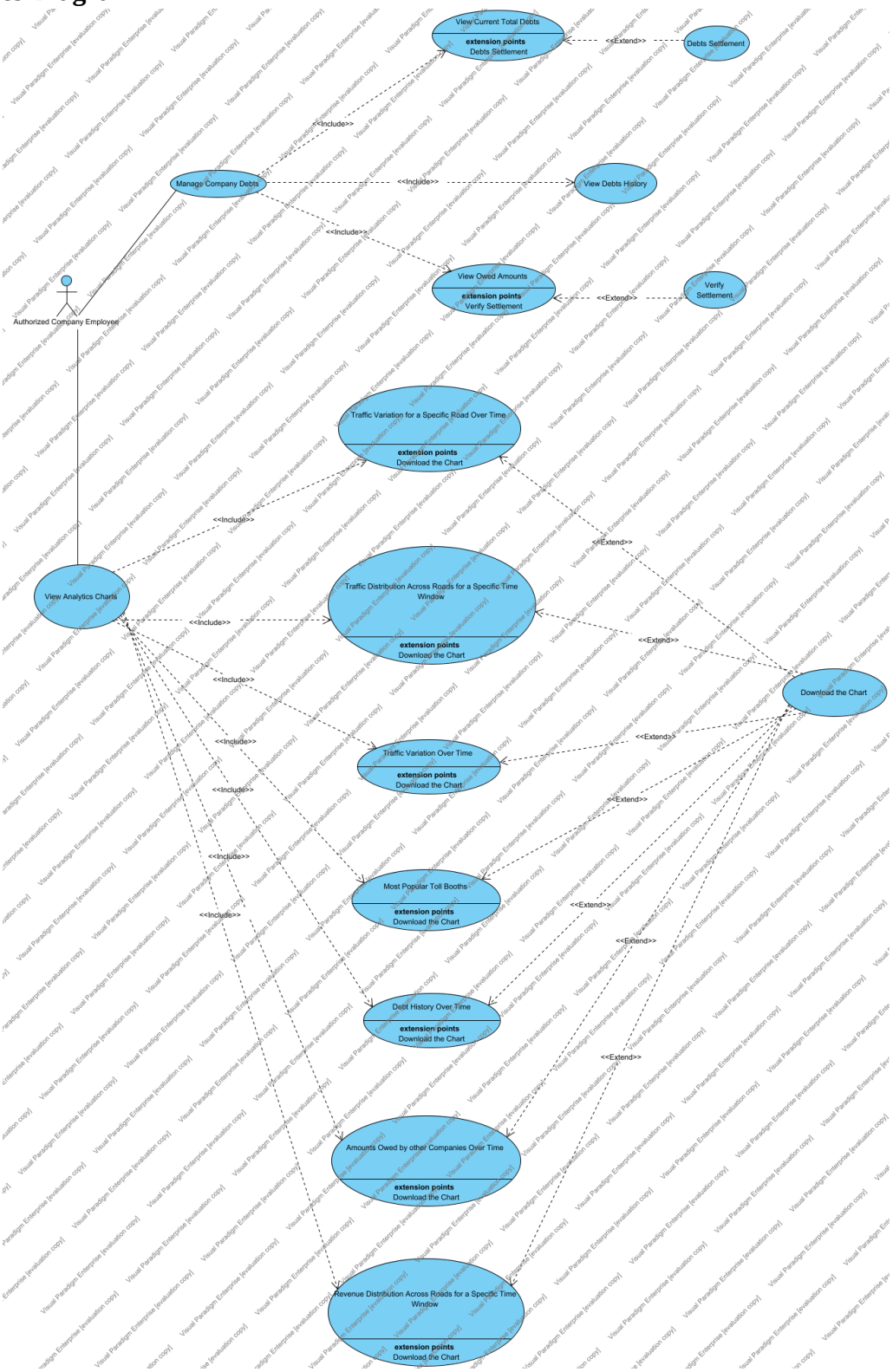
Debt Management

The system offers comprehensive debt management capabilities, enabling companies to efficiently track and manage their financial obligations. It displays current debts owed to other companies as well as amounts owed by others, providing a clear view of financial standings. Additionally, the system maintains a detailed historical record of debts over time, allowing users to analyze trends and changes. Debt settlement is streamlined through functionalities that enable authorized users to verify, review, and manage outstanding obligations, ensuring transparency and accountability in financial transactions.

Data Analytics

In the area of data analytics, the system provides a variety of diagrams to visualize traffic and revenue data. Users can view traffic variations on specific roads over time, assess traffic distribution across multiple roads for a given time frame, and analyze revenue distribution among toll roads. The system identifies the most popular toll booths based on vehicle crossings and presents historical data on debts and amounts owed by other companies. These diagrams are designed to provide stakeholders with valuable insights and can be downloaded for further review or reporting purposes.

Use Cases Diagram



Use Case	Description
Amounts Owed by other Companies Over Time	This use case displays a historical trend of the amounts owed to the company by other companies over a selected time period. Companies can analyze changes in outstanding balances, identify repayment patterns, and track financial relationships with other companies.
Debt History Over Time	This use case presents a historical trend of company debts, showing how debts owed have evolved over time.
Debts Settlement	This use case facilitates the process of settling debts owed by the company. The authorized employee can select specific debts and settle them. The debt status will be updated once payments are processed by the other companies.
Download the Chart	This use cases allows the authorized company employee to download any generated chart in png format for offline use or reporting purposes.
Manage Company Debts	This use case allows the company's authorized employee to access and interact with the company's debt information. It provides options to view current company's debts, amounts owed by other companies, and debt history. Debt settlement processes can also be initialized and payments by the other companies can be verified.
Most Popular Toll Booths	This use case highlights the most frequently used toll booths based on the number of vehicle crossings during a specific period. The chart can be filtered to focus on particular time ranges.
Revenue Distribution Across Roads for a Specific Time Window	This use case illustrates how revenue is distributed across different roads during a specified time window. Companies can identify roads generating the most or least revenue within the chosen period.
Traffic Distribution Across Roads for a Specific Time Window	This use case provides a chart showing the proportion of traffic distributed across different roads within a defined time window. It helps in identifying high-traffic and low-traffic roads during the specified period.
Traffic Variation for a Specific Road Over Time	This use case displays the traffic trend for a specific road over a selected time period. Users can analyze how traffic volumes fluctuate on a particular road over time.
Traffic Variation Over Time	This use case shows an overall trend of traffic variation across all monitored roads over a given time period. Companies can analyze traffic patterns and detect seasonal or time-based fluctuations.
Verify Settlement	This use case enables the authorized company employee to confirm and validate the successful settlement of debts. The verification will update the system to reflect the cleared status of debts.
View Analytics Charts	This use case allows the authorized company employee to access a variety of analytics charts that provide insights into traffic, tolls usage, revenue distribution, and debt trends.
View Current Total Debts	This use case displays a summary of all debts currently owed by the company to other companies, including details such as debts amounts with its associated companies and the ability to settle them as well.
View Debts History	This use cases provides access to a monthly historical record of all debts for a specific company for the last year.
View Owed Amounts	This use case shows a list of amounts owed to the company by other companies. Users can review details such as the owing company, outstanding amounts, and payment statuses, with the ability to verify a pending payment.

Requirements

Requirements Diagram

Requirement	Description
Data Accuracy and Consistency	Ensure that the data within the system is accurate and consistent.
Debt Calculating	At every data import, the debts between companies will be updated.
Debt Settlement	After a company settles its debts, it will have to register the settlement in the system.
Debt Settlement Verification	Each company will be able to verify that a debt owed by another company has been paid off.
Download Chart	A specific chart can be downloaded as an image file.
Importing Toll Passes	Each company will be able to submit a csv file containing their toll passes data for a specific amount of time.
Performance	The system should be able to respond rapidly and efficiently to multiple user requests.
Reliability	The system should consistently operate reliably without frequent failures.
Scalability	The system should be able to accommodate an increasing amount of data and user traffic.
Security and Privacy	Implement strong security measures to ensure user privacy, as well as secure data transmission and storage.
User Authentication	The companies and the supervising authority will need to enter their credentials to be authenticated and gain access to the system.
User-friendly Interface	The system interface design should be easy to use, intuitive and visually

Design	appealing in order to provide a positive user experience.
View Debt History Chart	Each company will be able to view its monthly debt history as a graph relative to time.
View Debts History List	Companies will be able to view their monthly debt history for the last year.
View Most Popular Tolls Chart	Each company can view chart of the most popular tolls that belong to it based on the number of vehicle crossings during a specific period.
View Owed Amounts History Chart	Each company can view the history of the amounts owed to it by the other companies as a graph relative to time.
View Owed Amounts List	Each company will be able to view the amounts owed by the other companies.
View Roads' Revenues Chart	Companies can view comparative charts about their roads' revenues.
View Total Current Debt	Each company will be able to view the sum of its current debts to other companies.
View Traffic Distribution Chart	Each company can view pie charts showing the traffic distribution across the roads that belong to it for a selected time window.
View Traffic Variation per Road Plot	Each company can view plots about the traffic variation of a road that belongs to it for a selected time window.
View Traffic Variation Plot	Each company can view plots of the total traffic variation on roads belonging to it for a selected time window.

User Requirements

Actor	Description
Authorized Company Employee	An employee designated by the company to access and manage financial and analytical data within the system. This role includes responsibilities such as viewing and managing debts, analyzing traffic and revenue statistics, and downloading charts. Access is restricted to ensure confidentiality and proper handling of sensitive financial information.

Technical Requirements

Performance

The system must process and display analytics and debt information efficiently, ensuring that data retrieval and chart generation do not exceed a delay of 2 seconds under normal operating conditions.

Scalability

The system should be able to accommodate an increasing number of users and transactions, with support for additional toll operators and growing datasets without degradation in performance.

Security

Robust security mechanisms must be in place to protect sensitive financial and traffic data. This includes encrypted data storage, secure communication channels (e.g., HTTPS), and role-based access control to ensure only authorized users access specific features.

Maintainability

The system must be designed for ease of maintenance, with modular components that can be updated or replaced independently. Clear documentation should accompany the software to simplify debugging and enhancements.

Usability
The interface must be intuitive and user-friendly, allowing employees to navigate and utilize system features with minimal training. Charts and analytics should be easily interpretable, with export options for convenience.

Localization Support
The system supports two languages: Greek and English. Users can toggle between these languages to ensure accessibility for stakeholders in both language groups. Regional formats such as date, time, and currency will also be adjusted accordingly.

Auditing and Logging
The system must maintain detailed logs of all activities, including data updates, user actions, and system errors, for auditing purposes and compliance requirements.

Availability
The system should maintain 99.9% uptime, with scheduled maintenance windows communicated in advance. High availability is critical to ensure uninterrupted access to features and data.

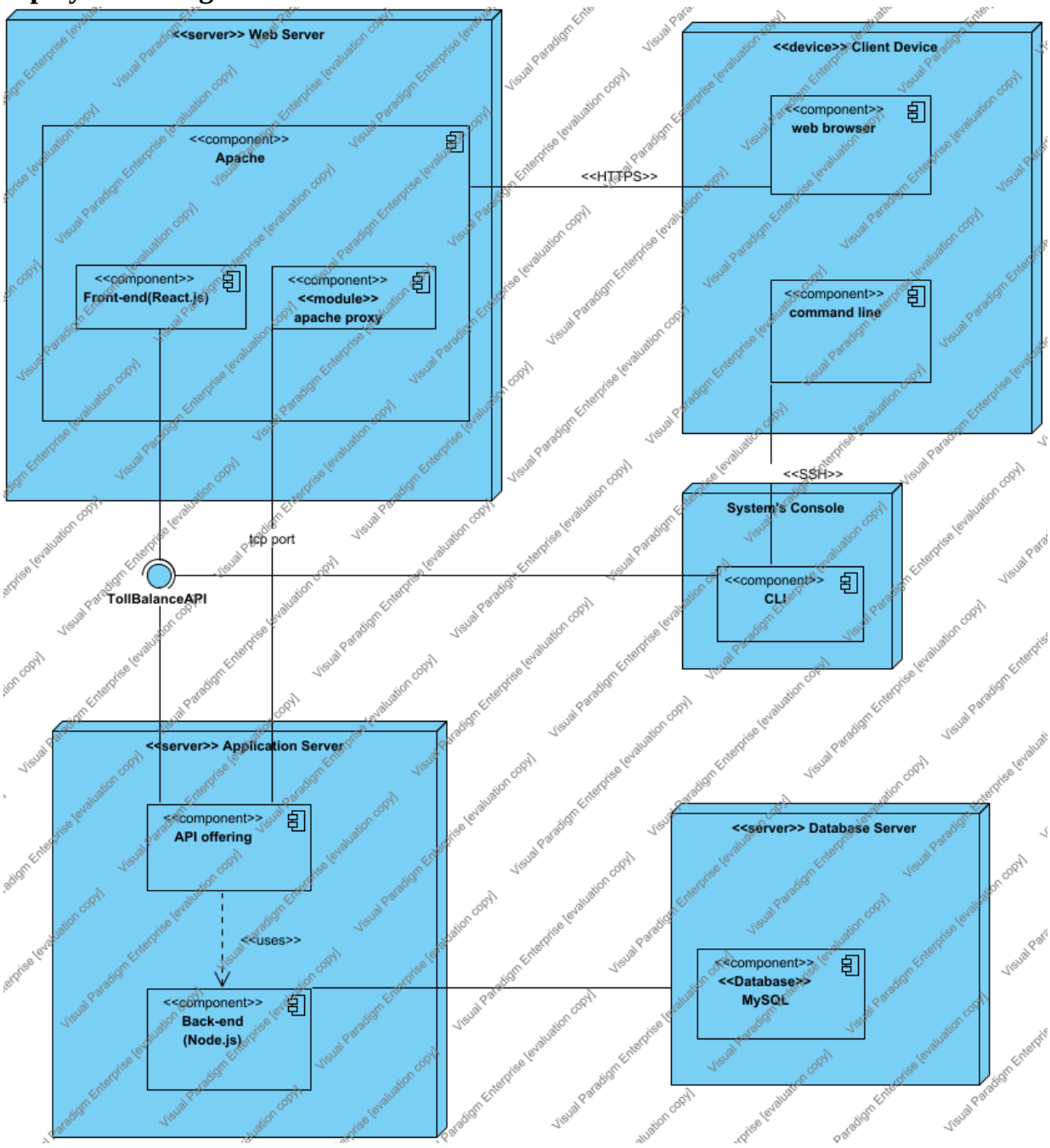
Hardware Requirements
Network
A stable internet connection with sufficient bandwidth to support data transfers between toll operators and users.

Client Computers
Minimum requirements:

- Processor: Dual-core 2.0 GHz or better
- RAM: 4 GB
- Storage: 500 MB free space
- Display: 1280x720 resolution or higher
- Browser: Latest versions of Chrome, Firefox, or Edge with JavaScript enabled.

Deployment Requirements

Deployment Diagram



Nodes

Name	Description
<<device>> Client Device	The client layer is represented by the user's device, which mainly uses a web browser to interact with the system. The web browser handles sending requests to the web server and displaying the content it receives. The

	system can also be accessed through a command-line interface (CLI), which communicates directly with the application server's API. This direct communication is important for administrative tasks that demand more control and accuracy than what is usually available through the web interface. The CLI also offers an alternative way to use the system's features.
<<server>> Application Server	The application server layer houses the system's back-end logic and is where the API is located. It serves as the core of the application's functionality, handling API requests, executing the required business logic, and performing data operations as needed. The back-end is built to manage complex tasks such as data validation, user authentication, and database interactions.
<<server>> Database Server	Data storage and management are managed by a separate database server running MySQL. This server is responsible for securely and efficiently storing the application data and works in conjunction with the application server's back-end. The back-end generates SQL queries, which are sent to the MySQL server for processing, and the server returns the requested data.
<<server>> Web Server	On the server side, the Apache web server serves as the gateway for all HTTP requests coming from the client's web browser. It is set up to directly deliver static content to the client, such as HTML, CSS, and client-side JavaScript, which make up the front-end of the application that users interact with. In addition to handling static content, the Apache server includes a proxy module that forwards requests for dynamic content to the application server. This module allows the web server to delegate tasks that require server-side processing, effectively functioning as a reverse proxy.
System's Console	The node system's console is the command-line environment where the system administrator and authorized users access the backend of the software. It features a robust CLI component that allows direct interaction with the system's databases and services, bypassing the need for a graphical user interface. The CLI is essential for carrying out tasks such as database maintenance, data entry, batch updates, and system diagnostics.

Components

Name	Description
<<Database>> MySQL	This database is essential to the system, containing information about toll stations, toll passes, operators and their debts. It is made up of several interconnected tables, each designed for a specific function. The database supports advanced queries, allowing for in-depth data retrieval to power various features, such as debt management and generating various diagrams.
<<module>> apache proxy	The Apache proxy acts as an intermediary between users and the web service, managing incoming requests and directing them to the correct backend processes. It can also offer additional features like SSL encryption, load balancing, and access control, improving both security and efficiency.
Apache	Apache is a key module that combines the Apache proxy and the front-end interface of the application. The front-end component, typically accessed through a web browser, is where users interact with the application. It is built using React.js. The front end is responsible for displaying

	information to users in an organized way, handling user inputs, and managing the state of the interface. Together, the Apache proxy and the front-end create the user-facing part of the application, bridging the user's requests with the server's responses.
API offering	The API Offering of our application is an integrated framework that enables client-server communication through HTTP requests and responses. It consists of a collection of defined routes (endpoints), each responsible for specific tasks. The server configuration ensures these routes are available over the network, while the application setup connects routing logic, middleware, and backend services to create a functional and scalable web service.
Back-end (Node.js)	This component forms the core of the server-side logic, responsible for initializing the server, setting up middleware, establishing database connections, and defining routes that correspond to various controller functions.
CLI	The CLI acts as a command-line interface that enables administrators and advanced users to interact with the backend system. It offers a range of commands for performing tasks such as user management, data retrieval, and content manipulation. Designed for efficiency, the CLI allows for scriptable and direct interaction with the system’s core functions, without requiring a graphical interface.
command line	The command line provides a text-based interface that allows administrators and users to run commands and interact with the application server remotely through Secure Shell (SSH). This method is essential for carrying out administrative tasks, managing application processes, and performing maintenance operations from any location with internet access. It offers a secure, encrypted connection to the server's CLI, enabling a powerful and direct way to control the application's core functions without the need for a graphical user interface.
Front-end (React.js)	This is the primary container that represents the entire React.js application. It houses all the other components related to the front-end of the application.
web browser	The web browser is the client application that allows users to access and interact with the front end of the web service. It displays the user interface, handles user inputs, and communicates with the server through the Apache proxy to make requests.