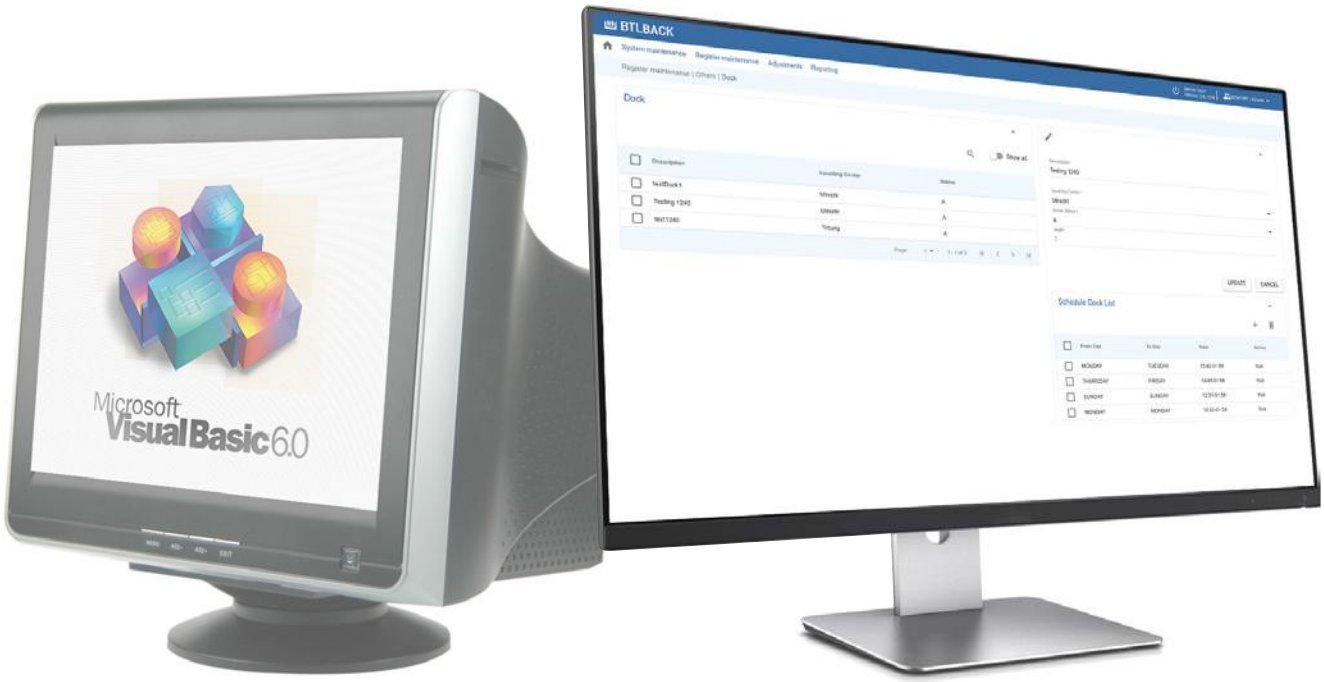
Web-based app for return of empty bottles

The customer is an independent business partner for companies operating in the online media field. It conducts project management on an outsourcing basis: from development to support, from security advice to hosting. The customer's team consists of 15 highly skilled professionals with at least 18 years of experience in online business.

This project is basically about the rebuild of an existing VB6 application in modern Java code. The application allows users to monitor or slightly adjust the process of counting and paying for the empty bottles (articles) that are coming to SRN.

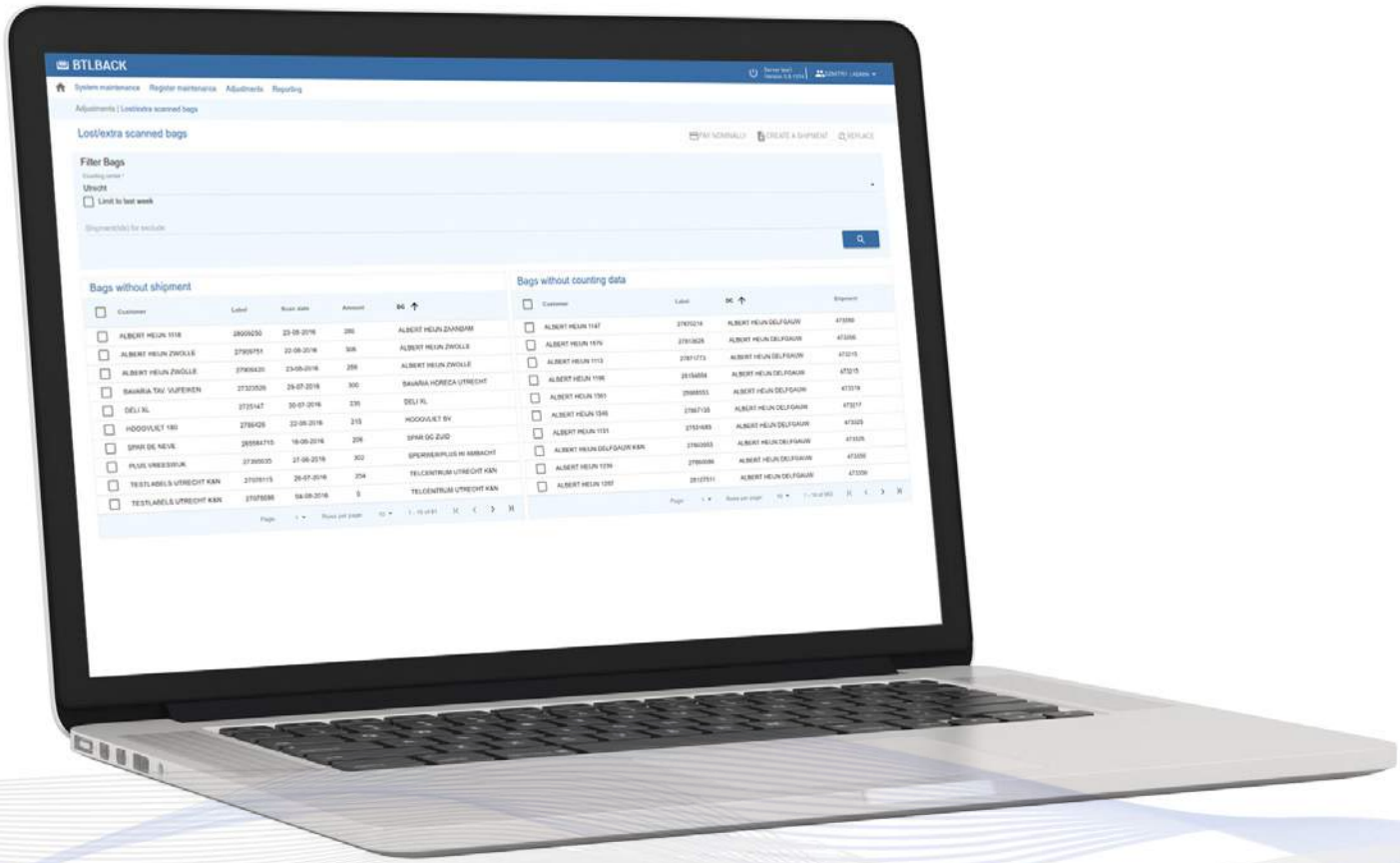
The application allows editing of the list of bottle types and collection points (customers/companies), it gives access to reports, and shows the current state of bags that are in progress or are missing somewhere, and gives details about tasks that are run to support the process



Challenge

Customers turned to us with the following problems:

- They needed a convenient way to build complex queries in the database.
- There were some performance problems with the client's site. The system was running slowly in the case of multiple requests.
- Customers needed to restrict access to the application using a White IP List where access would be allowed only to certain IP addresses.
- There was a need to synchronize data with other databases.
- It was necessary to construct graphs based on the processed data (e.g., "Number of bags per hour," "Number of shipments per hour," etc.).
- Customers needed improvement of TimeSlot overview and scheduling. It was necessary to develop an electronic schedule (organizer) enabling the reserving of time for unloading and servicing of trucks with bottles.
- They needed the ability to obtain additional user data from LDAP.



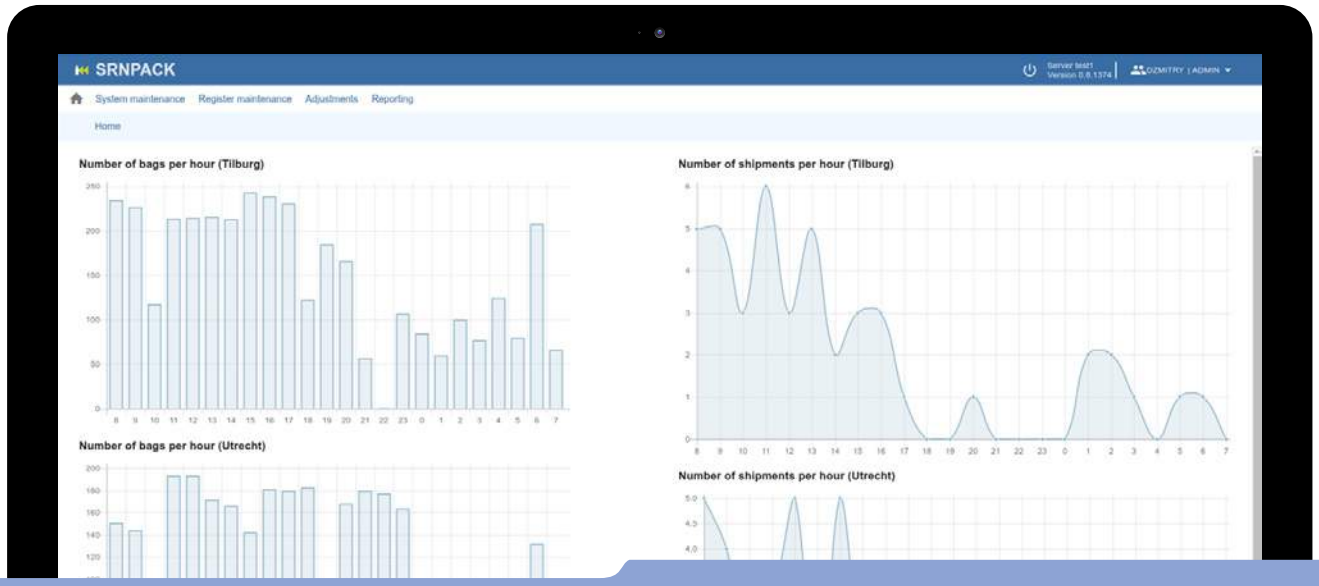
Solution

The problem was solved in the following way:

- To enable easy construction of complex queries in the database, we suggested and implemented the use of Querydsl.
- We optimized and used cache data to solve the performance problem and improve the operation speed when there are multiple requests.
- To restrict access to the application, we implemented Interceptor for requests intercepting and filtering (White IP List and Whitelist for paths).
- We implemented a file manager based on Qdrive.
- To construct graphs on the basis of processed data, we used an additional module of chart.js.
- We developed a data model and implemented basic logic for CRUD operations in accordance with business requirements (frontend and backend sides).
- We developed Time Slots Overview – an electronic schedule (organizer) to reserve the unloading time of the machine with bottles, featuring marking of fields "reserved time" / "available for reservation time" / "unavailable for reservation time" by different colors. The mechanism takes into account the overlap of reserved periods, and automatically suggests the nearest free time, if the time the customer has chosen for reservation is already taken. After reservation, the customer can view its details at any time.
- We created jobs or scheduling tasks for processing the scheduled data.
- We developed a variety of services for the system, for example, MailService – a shared service to send e-mails.
- We wrote unit tests.

Key Features

- Access by roles
- Imitation of logging in as a different user for the administrator
- History of the entities' changes
- Report generation
- Display of the data in graphs
- Receiving and sending data to external terminals and databases
- Sending e-mails
- Tasks that process data in manual or automated mode



Technologies Used

Java 8
Angular
Gradle
SpringBoot
MySQL
JasperReports