Out-of-the-box performance optimization

When it's time to develop, OutSystems guides developers to ensure every application has no performance bottlenecks with design-time validation. As it generates the source code for applications, OutSystems optimizes code performance at all application layers. From the number of connections used to access the database to the size of the pages, it's able to ensure optimal functionality for the intended application.

Design-time validation

The OutSystems visual editor includes performance validation tests. This way, developers know about performance bottlenecks even before deploying the application.

This is delivered by the TrueChange engine, which reports potential errors, scalability and performance bottlenecks, and dependency inconsistency.

The issues that are reported include:

- Holding large amounts of data in memory
- Inefficient modeling of a database table
- Inefficient database accesses
- Logic that processes more data than necessary
Code containment and optimization

All the code generated by OutSystems is optimized to ensure that the application being developed behaves correctly and avoids scalability and reliability issues. To reduce bandwidth bottlenecks, it uses compression to optimize page size. Automatic AJAX queuing ensures end-users won’t harshly impact the server with multiple simultaneous AJAX requests. Smart prevention of duplicate AJAX requests keeps identical requests from being sent to the server. And, partial screen rendering using AJAX further reduces traffic and improves the user experience.

Database fetch reduction ensures that only data that is effectively used in the application is fetched from the database regardless of what the developer has coded. Smart database-to memory-algorithms assure that minimal memory is used by only loading datasets into memory as a last resort. Database connection pooling prevents excessive use of the operating system, the network and database resources and smart connection release reduces the consumption of resources by the operating system, network and database.

Smart commit, rollback, and release of transactions ensures that no database update is left forgotten and that no database update is done in case of error. Automated indexing of primary and foreign keys optimizes the most common database queries. Database timeout containment eliminates the risk of heavy queries affecting the performance of the application and database environment.

Adapter timeout containment eliminates the risk of custom coded components and integration affecting the performance of applications. Asynchronous bulk logging ensures that no logging activity affects end-users and business events and built-in log rotation ensures that database logs do not grow indefinitely.

Scheduler throttling keeps even heavy loads of badly programmed scheduled activities from taking over all of the processing power. Transactional deployment ensures that applications are deployed correctly and that incompatible versions of an application’s components are not running at the same time.

Multi-tenant data isolation and containment allows teams to operate multiple applications in perfect data isolation.