7 Ways To Modernize Legacy Systems

Digital transformation over the last couple of years, especially since the COVID-19 pandemic, has made it crucial for businesses to come up with potent yet practical ways to upgrade existing legacy systems or go altogether for a new system that can help businesses change with the times.

“For many organizations, legacy systems are seen as holding back the business initiatives and business processes that rely on them,” says Stefan Van Der Zijden, VP Analyst, Gartner. “When a tipping point is reached, application leaders must look to application modernization to help remove the obstacles.”

If you too are looking for ways to modernize legacy applications/systems of your business, the best approach depends on the problem you're trying to solve. However, it is necessary to understand the risk-to-reward ratio before acting.

Firstly, we need to understand the issues, concerns or impediments that have been created by the legacy application as a result of its technology, architecture or functionality.

These issues can be pertaining to the business fit, business value and agility, risk or can pertain to the IT perspective involving the 3C's, namely, cost, complexity and compliance.

If the legacy application is not meeting the new requirements thrust upon by digital business, it may be a cost or risk liability. If the total cost of ownership is too high, the technology too complex, or security, compliance, support or scalability are being compromised, the applications need to be modernized to properly suit the business need and provide greater business value.

The best modernization approach is one that has a manifold impact from both a business and an IT perspective.

Once the problem is identified, evaluate your choices based on the Smart Modernization Options given below, ranked by ease of implementation (the easier one has the least risk and impact on the system and the business processes; the harder one has the most risk and impact).

7 Ways To Modernize Legacy Systems

1. **RePackage** - You can still make use of existing application features along with the newer system by repackaging its data and functions and making them available as services via an API.
2. **ReHost** - Relocate the application component to other infrastructure (physical, virtual or cloud) without modifying its code, features or functions.
3. **RePlatform** - Transfer to a new runtime platform, making slight yet necessary modifications to the code, but not the code structure, features or functions.
4. **ReFactor** - Re-engineer and optimize the existing code (although not its external behavior) to remove technical debt and improve inoperative attributes.
5. **ReArchitec**t - Change the code to move it to a new application architecture and exploit new and better capabilities.
6. **ReBuild** - Revamp or rewrite the application component from scratch while upholding its scope and specifications.
7. **Remove** - Drop the former application component altogether and replace it, keeping in mind new requirements and needs at the same time.