Navigating Emerging Technologies for Business Reinvention

Catalyst
by Zoho

Introduction

In today’s digital age, IT is no longer viewed solely as a cost center, but rather as a powerful engine driving innovation and growth. Companies that strategically leverage modern tech are outpacing competitors, creating groundbreaking products and services, and optimizing operations for maximum efficiency.

However, the tech landscape is constantly evolving. To remain ahead of the curve, businesses must embrace a culture of continuous learning and exploration. This requires not just keeping up with the latest trends, but actively experimenting and integrating new solutions into your existing infrastructure

This e-book is designed to help you navigate the exciting world of emerging technologies. We will cover a wide range of topics and trends that are transforming the way businesses operate, including artificial intelligence, event-driven architectures (EDA), energy-efficient apps, and cross-platform development frameworks.

Artificial intelligence and machine learning

Artificial intelligence and machine learning are no longer the exclusive domain of big tech companies. The emergence of advanced AutoML tools and innovations in the generative AI space have significantly lowered the barrier to entry. If you haven't yet harnessed the power of AI to optimize internal processes and deliver more value to your customers, the risk of falling behind has never been greater.

The benefits of AI for businesses are vast and multifaceted.
Here are just a few examples:

Automation: AI automates repetitive tasks, which frees up human capital for higher-level strategic thinking and creative endeavors. For example, you can use Zia services by Catalyst to automate tasks like text analytics and image moderation.

Data-driven decisions: AI analyzes massive data sets to uncover hidden patterns and trends. This enables businesses to make data-driven decisions that deliver better outcomes.

Improved customer experience: AI personalizes customer interactions, streamlines customer service processes, and provides real-time support. This leads to increased satisfaction and loyalty.

Risk Management: AI-powered algorithms identify potential risks and vulnerabilities in real time, enabling businesses to proactively address threats and avoid disasters. For example, you can use Catalyst QuickML to build a fraud detection model that protects your organization from fraudulent activities.

The power of generative AI

Generative AI (Gen AI) holds immense potential for businesses looking to innovate and disrupt their industries. It allows computers to not just analyze data, but also create entirely new content and ideas autonomously.

The beauty of Gen AI lies in its ease of integration and rapid time to value. Take Large Language Models (LLMs) for example. With minimal training data specific to your industry, LLMs can be fine-tuned to generate content, translate languages, write different kinds of creative text formats, and answer your questions in an informative and human-like way.

Let's explore how LLMs and generative AI can be used across industries to create significant value:

Cybersecurity

- Train LLMs on vast amounts of network data to identify threats and suspicious patterns in real time. This can significantly reduce the time it takes to detect and respond to cyberattacks.

- Create personalized cybersecurity training content tailored to different employee roles and risk profiles.

- Develop AI-powered systems that can automatically analyze security incidents, trigger appropriate countermeasures, and even generate reports.

Healthcare

- Use Gen AI to process vast sets of scientific data and accelerate the discovery of new drugs and treatment options.

- Scrutinize a patient's medical history, genetic data, and current health status to generate personalized treatment plans.

- Automate the generation of routine medical reports by training LLMs on existing data, specific to your institution.

Content creation

- Overcome writer's block with AI-powered brainstorming tools that suggest relevant topics and content ideas based on your target audience and industry trends.
- Generate targeted marketing copy and product descriptions tailored to specific customer segments. LLMs can personalize content based on demographics, interests, and past purchase behavior.

Code generation

- Use Gen-AI powered coding assistants to generate code for repetitive tasks, predict code vulnerabilities, and get algorithmic advice.
- Generate comprehensive test cases automatically, based on code functionality.

Customer support

- Develop intelligent chatbots powered by LLMs that can answer customer questions 24/7, troubleshoot basic issues, and even escalate complex problems to human agents.
- Examine customer feedback and social media conversations using LLMs to understand customer sentiment towards your brand and products.

The potential applications of Gen AI are constantly expanding, and businesses that invest in this revolutionary tech are poised to gain a significant competitive edge.

Event-driven architecture (EDA)

Event-driven architecture (EDA) is a design pattern in which application logic is driven by events. It is fundamentally different from traditional request-response architectures where components talk to each other via direct method calls or request APIs.

In EDA, applications publish events whenever something significant happens in the system, such as when a new user signs up, a transfer is initiated, or an account is deleted. These events are then broadcasted to interested parties (other applications or services) that have subscribed to receive them.

This decoupled approach offers a plethora of advantages over traditional architectures:

Loose coupling: Services are loosely coupled in an EDA, meaning they don't need to be aware of each other's implementation details or internal state. This simplifies development, maintenance, testing, and deployment.

Scalability: Applications can be scaled independently without impacting others, and event processing can be distributed for optimal performance.

Responsiveness: Event-driven apps react to events in near real time, which leads to faster response times and improved user experiences. This is particularly important for applications that require low-latency interactions, such as web apps.

Extensibility: Since events serve as a common communication mechanism, organizations can easily integrate third-party systems and services into their ecosystems.

Improved developer productivity: EDA promotes a focus on business events rather than complex communication protocols. This further simplifies development and allows developers to concentrate on core business logic.

If you are looking for a ready-to-use platform to build event-driven applications, check out the Event Listeners component of Catalyst Cloud Scale by Zoho. It's an event bus service that listens to predefined events and automatically executes associated business logic functions.

GraphQL implementation

GraphQL is a query language for APIs. Unlike traditional REST APIs, where clients have limited control over the data they receive, and often need to make multiple requests to fetch related data, GraphQL allows clients to specify exactly the data they need, in a single request.

GraphQL APIs are defined by a schema that outlines the types of data available and the relationships between them. The clients leverage this schema to fetch data from the server.

For example, suppose we have a GraphQL schema for a social media platform with two types: User and Post. The User type has fields like id, name, and email, while the Post type has fields like id, title, content, and author (which is a User type).

With GraphQL, the client can fetch the required data in a single query, like this:

```
query{
user(id: "123") {
   Name
   Email
   posts{
   Title
   Content
}
}
}
```

This query fetches the name and email of the user with the ID "123" along with the titles and content of all their posts.

Why use GraphQL?

Here are some benefits of using GraphQL over REST APIs:

Increased efficiency: Fetching only the required data reduces network traffic and improves overall application performance.

Query complex data models: You can request data from related models in a single query (like in the above example), simplifying complex data retrieval.

Easier development: GraphQL APIs offer a single endpoint for all data retrieval. This simplifies development efforts by reducing the need to manage multiple endpoints or send multiple requests.

Versionless APIs: GraphQL's flexible schema makes it easier to evolve APIs over time without breaking existing clients or introducing backward compatibility issues.

How to get started with GraphQL

Follow these steps to implement a GraphQL API:

- **Describe your data model** using a schema definition language (SDL) that specifies data types, relationships, and access permissions.
- Choose a **GraphQL server implementation** that fits your tech stack, and set up a server to handle incoming GraphQL queries. Popular options include: Apollo Server, GraphQL Yoga, and Express GraphQL.
- **Implement functions** (resolvers) that handle incoming queries, fetch data from your backend databases or services, and return the requested data according to your schema.
- E**xpose your GraphQL endpoint** so that clients can send queries and mutations to the API.
- Use **GraphQL client libraries** to send queries and handle responses within your frontend application.

Node.js and Python trends

Node.js and Python are two top choices when it comes to backend development. In the next few sections, we will explore some of the current trends in Node.js and Python, and provide guidance on selecting the most suitable programming language or framework for your project.

Node.js trends

Node.js is a JavaScript runtime environment that's known for its event-driven architecture and non-blocking I/O model. Some emerging trends in the Node.js ecosystem include the following:

- To facilitate real-time communication, an increasing number of Node.js applications have started to use WebSockets.

- There's a growing focus on TypeScript as a preferred language for Node.js development. TypeScript offers static typing, which improves code maintainability and reduces runtime errors.

- GraphQL is increasingly being used with Node.js to build efficient and flexible APIs.

- Popular Node.js frameworks like TensorFlow.js and PyTorch.js are making it easier to integrate ML capabilities into web applications.

Python trends

Python is a simple yet versatile high-level programming language. Here are some recent trends in the world of Python:

- Many cutting-edge libraries for data science and generative AI continue to be released in Python, including OpenAI's Python library.

- In the past, Python was primarily used for implementing data science and machine learning features. However, it is now increasingly being considered as a primary language for a wide range of business use cases.

- Latest versions include various changes, to the language itself and the standard library, that directly improve performance. As Python continues to get used for resource-intensive AI and ML tasks, these performance enhancements are likely to continue in future releases

A checklist for choosing the right programming language/framework

- Start by considering the primary function of your application. Does it require real-time features or data streaming? Node.js might be better. Or is it a data-heavy application focused on machine learning? Python might be a better fit.

- What programming languages/framework are your developers most familiar with? Choose a language/framework your team is experienced in to reduce learning curves.

- If rapid development is a priority, consider the ease of use and existing libraries offered by the reviewed languages/frameworks. For example, Python has arguably the largest number of machine learning libraries available

- Consider the performance characteristics of each language/framework. For example, if you want the fastest possible performance, you may want to go with C/C++ or Rust. Alternatively, if you want to balance performance with ease of development, then you can consider either Python or Node.js.

- Take into account any specific requirements or constraints of your project, such as integration with existing systems or compatibility with third-party services. For example, if most of your existing apps are written in JavaScript/TypeScript, then Node.js may be the better fit for you.

- Consider how easy or difficult it would be to deploy applications built with this language/framework on your chosen platform (cloud, server, etc.).

Flutter vs. React Native

Flutter and React Native are top choices for cross-platform mobile app development. Here's a comparison table that summarizes the differences between the two:

Aspect	Flutter	React Native
Development language	Dart (Google-developed, object-oriented, similar to Java/C#)	JavaScript (widely used, large community)
Learning curve	Steeper (new language, custom widgets)	Easier for developers familiar with JavaScript and web development
Performance	Excellent, native-like experience due to custom rendering engine	Generally good performance, but may face occasional performance issues due to JavaScript bridge
UI/UX design	Highly customizable with built-in, performant widgets (declarative approach)	Relies on native UI components, potentially requiring more native code
Community	Growing community with active development	Large and mature community with extensive third-party libraries
Platform support	Full support for Android and iOS with a single codebase	Full support for Android and iOS with a single codebase
Native access	Direct access to native APIs through platform channels	Access to native APIs through JavaScript bridge
Testing	Comprehensive widget testing framework, unit testing with popular tools	Standard JavaScript testing frameworks can be used
Cost	Free and open-source	Free and open-source

To make the right call between React Native and Flutter, carefully evaluate your specific needs and developer preferences. Here's a quick guide:

Choose Flutter if:

- You need exceptional performance and a highly customizable UI.
- Your team is comfortable learning a new language (Dart).
- You prioritize rapid development and hot reload features.

Choose React Native if:

- Your team has strong JavaScript expertise.
- You need faster development for simpler apps.
- A large, active community is important for support.

Full-stack development platforms, like Catalyst by Zoho, offer SDKs for languages like Flutter and React Native, making it even easier for you to get started.

Web automation with Headless Chrome

Headless Chrome is a way to run Chrome without the usual GUI. Within the headless environment, you can control Chrome programmatically through code, automating repetitive tasks and web interactions. For example, you can use simple commands to print the Document Object Model (DOM), generate PDFs of specific sections, or capture screenshots.

At the time of writing, Chrome offers two headless modes: old and new. In the old mode, separate code paths exist for headless and headful (normal) code implementations.

However, in the new mode, the Chrome team has unified the codebase for both implementations. For consistent testing, it's recommended to use the new mode.

Let's look at some benefits of Headless Chrome:

Automated testing: Run automated tests on your web applications to ensure functionality and identify regressions before they impact users.

Web scraping: Extract data from websites for various purposes like market research, competitor analysis, and data aggregation.

Automation: Automate mundane tasks to focus more on building immersive and well-tested web apps.

Programmatic access: The Chrome team has also released libraries, such as the **Puppeteer library for Node**, to interact with headless Chrome. This allows developers to programmatically navigate web pages, interact with elements, and test behavior.

If you are looking to manage a headless browser in a cloud setup, without having to manage the underlying infrastructure, check out the Catalyst SmartBrowz service.

Building energy-efficient mobile applications

In today's mobile-first world, users expect applications to be not just functional but also battery-friendly. A mobile app that drains a phone's battery quickly can lead to frustration and user abandonment. But more importantly, it can contribute to the larger environmental issue of electronic waste. Phones with depleted batteries often end up discarded, adding to the growing e-waste problem.

Here are some reasons why it's more crucial than ever to build Energy-efficient mobile applications:

Sustainability: Energy-efficient apps consume less power, leading to a smaller carbon footprint and a more sustainable mobile ecosystem.

User experience: Longer battery life translates to a more positive user experience. Users won't have to constantly scramble for a charger or worry about their phone dying at a crucial moment.

Brand reputation: Energy-conscious development practices contribute to a positive brand image, which resonates with environmentally conscious users.

Reduced development costs: Optimized apps require less processing power, potentially leading to lower server costs and improved scalability.

Strategies for building energy-efficient apps

Follow these strategies to prioritize the development of apps that naturally consume less energy:

- Write clean, efficient code that avoids unnecessary calculations and redundant operations. You can use profiling tools to identify performance bottlenecks in your code.

- Reduce CPU usage by optimizing screen updates. Lower the refresh rate when possible and avoid unnecessary UI redraws.

- Implement techniques like lazy loading, caching, and prefetching to minimize network requests and data transfer.

- Use background tasks judiciously and implement mechanisms to stop them when not needed

- Choose lightweight libraries and frameworks optimized for performance and energy efficiency, such as GraphQL over REST for efficient data fetching.

- Location services are a major battery drain. Request location updates only when necessary and switch to lower accuracy modes when possible.

What tools can you leverage to build energy-efficient apps?

Incorporate these tools into your development workflows to build sustainable apps:

Battery Historian: An open-source tool that allows developers to analyze battery usage data from "bugreport" files and identify optimization avenues.

Android Energy Profiler: A native Android Studio tool for monitoring energy consumption metrics in real time and detecting any energy-intensive applications.

Xcode energy impact gauge: A built-in Xcode feature that can be used to inspect energy usage during app execution and identify energy-hungry code segments in iOS applications.

Power management APIs: Many operating systems, including Android and iOS, provide power management features to optimize resource utilization. For example, "App standby buckets" in Android restrict access to CPU, battery, and other resources.

Leveraging edge computing for real-time decision-making

If you are looking to process large volumes of data while minimizing network latency, edge computing can be a game changer for your organization. It fundamentally shifts the paradigm of data processing by bringing computational power closer to the source of data — i.e., the "edge" of the network.

This reduces, and sometimes removes entirely, the need to constantly transmit data back and forth to centralized servers. As a result, your applications are able to process data and make decisions in real time. Here are some best practices you can follow to set up and deploy resilient edge computing solutions:

1. Start by identifying use cases where real-time data processing and low-latency responses are critical. This will help you determine the processing power and storage capacity required for your edge services.

2. Choose hardware components purpose-built for edge computing tasks, such as edge servers, edge gateways, and edge accelerators. Also select software that is optimized for resource-constrained environments.

3. Since edge devices are often geographically dispersed, it’s important to consider security a fundamental design requirement. Implement encryption techniques, regular security updates, and secure authentication protocols to safeguard your data and systems.

4. Ensure reliable and consistent connectivity between edge devices and central servers for any data synchronization or management operations. Explore options like cellular networks, low-power wide-area networks (LPWAN), or satellite internet depending on the deployment location.

5. Leverage edge analytics tools and algorithms to analyze data and extract valuable insights at the edge. Libraries and platforms like TensorFlow Lite and EdgeX Foundry can be used for this purpose.

6. Design your edge computing architecture with scalability in mind. You should be able to easily add or remove edge devices as your needs evolve.

7. Ensure that you remain compliant with regulatory requirements and industry standards, especially when deploying edge computing solutions in regulated industries like healthcare, finance, or government.

Embracing the cloud for agility and scalability

Cloud platforms enable businesses to automate, adapt, innovate, and scale at the pace of modern business demands. Here are some undeniable benefits of going cloud-first:

Reduced time to market

Gone are the days of lengthy infrastructure setups and hardware procurement. With cloud-based services, you can provision resources instantly, integrate with pre-built solutions, and deploy applications faster than ever before. This leads to a reduced time to market, allowing your business to seize fleeting opportunities and capitalize on market trends.

Cost savings

Cloud computing typically operates on a pay-as-you-go model. You only pay for the resources you use, which eliminates the upfront costs associated with traditional on-premises infrastructure. This not only reduces your business's financial burden, but also eliminates the risk of over-provisioning hardware that sits idle.

On-demand scalability

Cloud infrastructure scales seamlessly, on demand. You can easily scale resources up or down in response to traffic surges or seasonal peaks. This ensures optimal performance and reduces maintenance overhead and costs associated with manual scaling efforts.

Enhanced security and compliance

All leading cloud providers invest heavily in robust security measures, compliance initiatives, and disaster recovery protocols. Your data is stored in geographically distributed data centers with advanced security features. Moreover, cloud services are designed for high availability, which minimizes downtime and ensures business continuity.

Global reach and collaboration

In the age of remote-first workforces, the cloud empowers businesses to operate on a global scale. Cloud-based applications and data can be securely accessed from anywhere with an internet connection. This fosters seamless collaboration across geographically dispersed teams.

Multi-cloud strategies

The cloud landscape is no longer a one-size-fits-all proposition. Businesses are free to adopt a multi-cloud strategy, where they leverage different cloud providers for specific needs. This offers greater flexibility, redundancy, and potentially better pricing compared to relying solely on a single vendor. And since most cloud vendors prioritize API-based interoperability, you can easily integrate services across multiple platforms.

Conclusion

In today's rapidly evolving technological landscape, staying ahead of the curve is an imperative. As a technology advocate, you can empower your business and customers by incorporating some of the technologies discussed here into your existing systems. For example, you could build a roadmap to leverage technologies like generative AI, event-driven architecture, or cloud-based solutions.

To accelerate your tech stack modernization journey, consider Catalyst. It’s a powerful pro-code, full-stack development platform that allows you to build, test, and deploy modern applications with unprecedented speed and agility. This translates to faster time to market, and the ability to deliver innovative solutions that keep your customers engaged.

Interested? Contact us for a free consultation.

Schedule a free consultation

Catalyst
by Zoho