Project Design Phase - Part 2

The requirements(Customer Journey Maps):

Customer journey maps are used to map the relationship between a customer and an organization over time and across all channels on which they interact with the business. Design teams use customer journey maps to see how customer experiences meet customers' expectations and find areas where they need to improve designs.

Customer journey maps are research-based tools which design teams use to reveal typical customer experiences over time and visualize the many dimensions and factors involved. These enable brands to learn more about target users. Team members examine tasks and questions (e.g., what-ifs) regarding how a design meets or fails to meet customers' needs over time when they encounter a product or service.

A timescale – a defined journey period (e.g., 1 week) including selected areas from awareness to conversion and beyond.

Scenarios – the context and sequence of events in which a user/customer must achieve a goal (e.g., a user wants to buy a ticket on the phone), from first actions (recognition of a problem) to last actions (e.g., subscription renewal).

Touchpoints – *what* customers *do* while interacting and *how* they do it.

Channels – *where* they perform actions (e.g., Facebook).

Thoughts and feelings –what the customer thinks and feels at each touchpoint.

User Journey Maps - Example:

Requirement Analysis:

Requirement analysis is an element of project management that helps ensure clarity, completeness, and relevance. The goal is to define expectations for a project. The requirements may apply to software development, process improvement, or a new technology purchase.

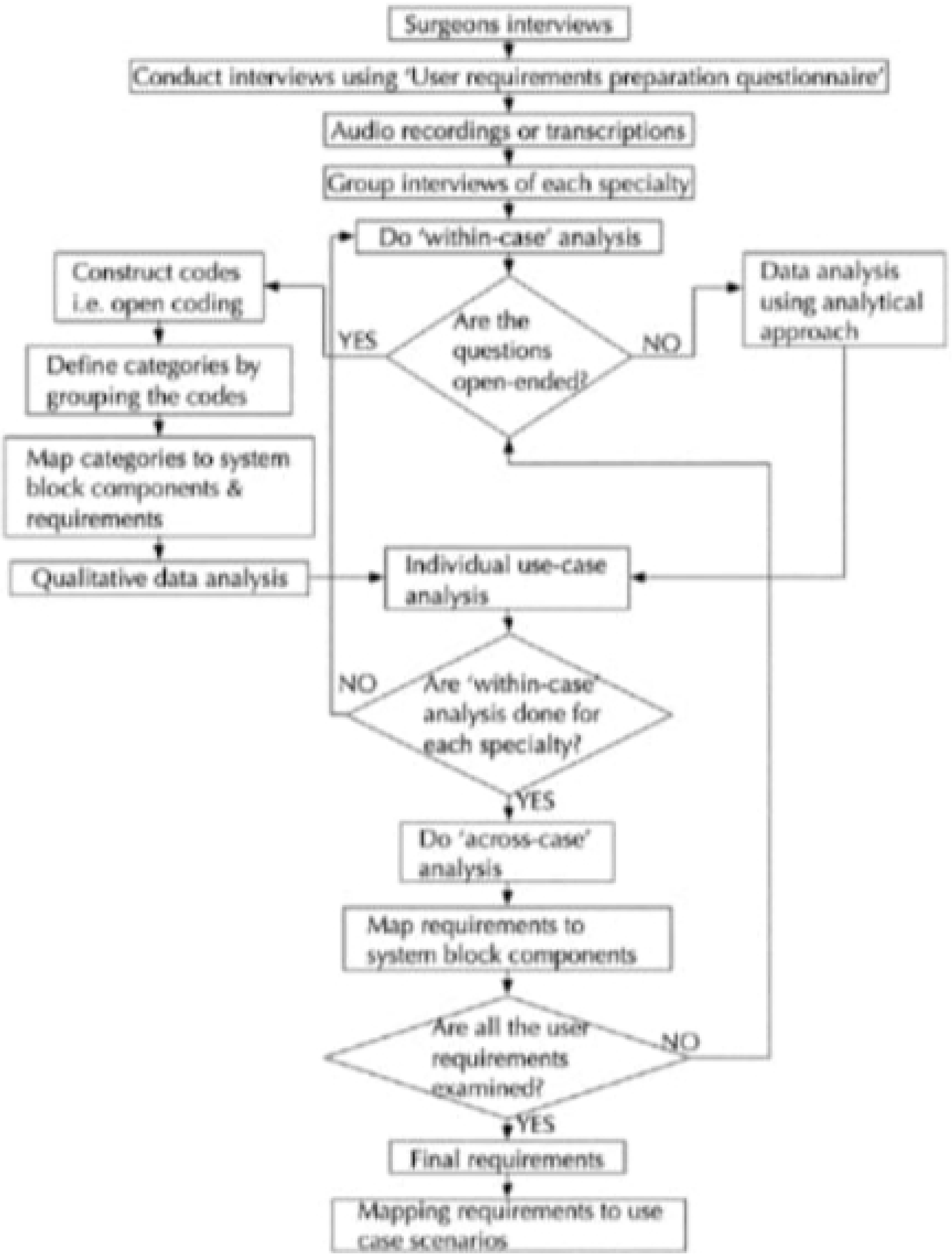
The purpose of requirements analysis is to collect the information necessary to design a product to meet the needs of stakeholders. Once the requirements analysis is complete, it will align with tasks that result in a suitable product.

Types of Requirement Analysis:

- **Business Requirements:** The business requirements include high-level needs that will help achieve a company's objectives. They align with business objectives, mission, and goals. Specific features are not part of business requirements.
- **Customer Requirements:** These requirements include expectations of how a product will meet the needs of a specific user group.
- **Design Requirements:** Design requirements are a result of customer and product requirements. They convert the needs of the business and customers into design features.

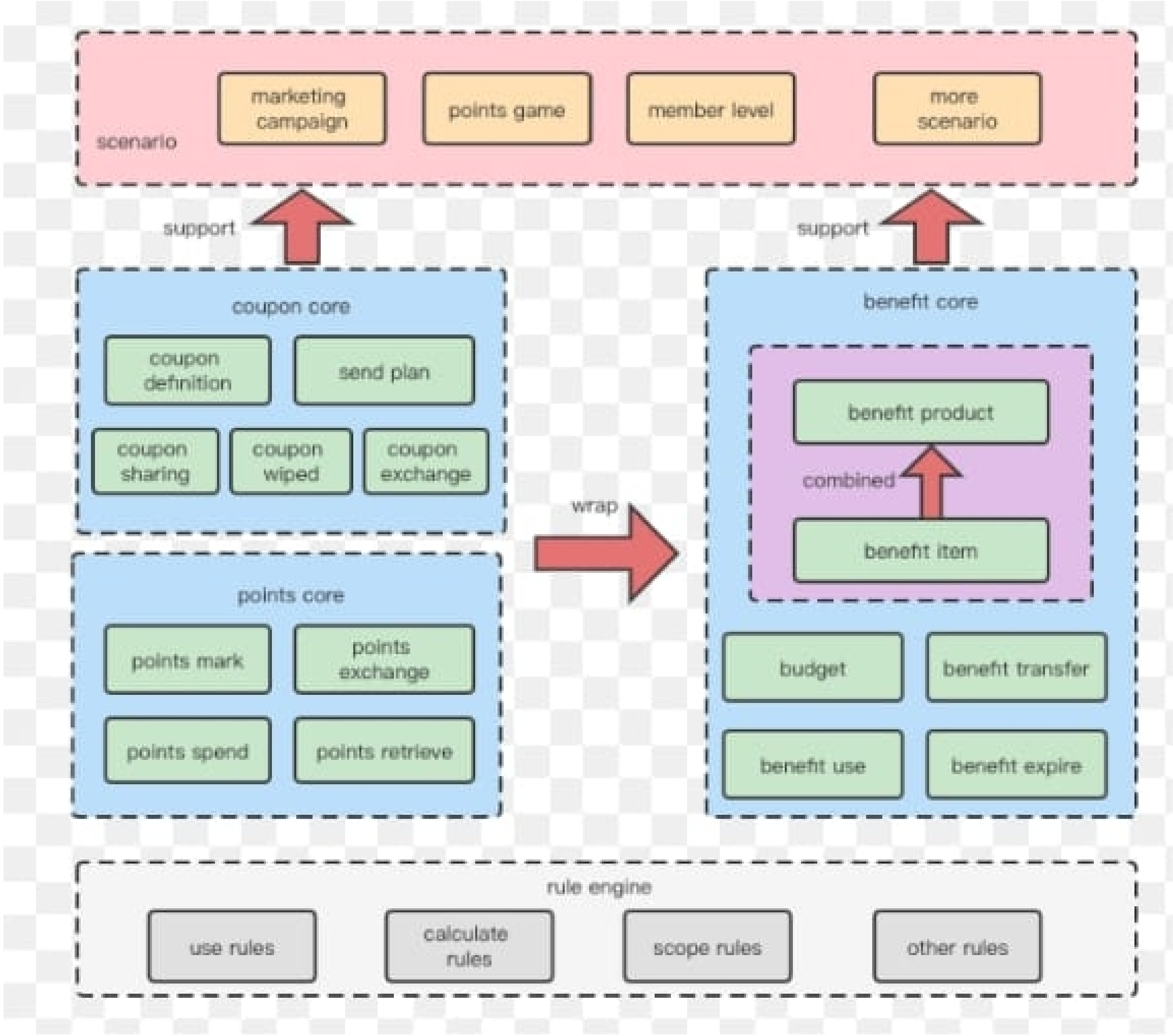
- **Functional Requirements:** A *functional requirement* is a subset of a product requirement. It describes the service that a software product must deliver, including the specific behaviors of features and functions.
- **Non-Functional Requirements:** A *non-functional requirement* is a general quality attribute of a solution, such as a performance measurement.
- **Product/Service Requirements:** A *product requirement document* (PRD) is an artifact that outlines the purpose, characteristics, functions, and features of a product, including software. The PRD is completed before development.

Flowcharts:


Surgeons interviews
Conduct interviews using 'User requirements preparation questionnaire'
Audio recordings or transcriptions
Group interviews of each specialty
Do 'within-case' analysis
Construct codes i.e. open coding
Define categories by grouping the codes
Map categories to system block components & requirements
Qualitative data analysis
YES
Are the questions open-ended?
NO
Data analysis using analytical approach
Individual use-case analysis
NO
Are 'within-case' analysis done for each specialty?
YES
Do 'across-case' analysis
Map requirements to system block components
Are all the user requirements examined?
NO
YES
Final requirements
Mapping requirements to use case scenarios

Technical Architecture:

Technical Architecture is the name of the total concept that is applied to the IT Infrastructure of an organization. IT Infrastructure is a coherent set of interconnected hardware and software, like networks, clouds, servers, clients, printers, tablet PC, smartphones.



Open Source Frameworks:

The Open Source Initiative has approved more than 80 licenses among hundreds of existing ones. These approved licenses can be divided into permissive and copyleft licenses. The former allows you to use the code for any purpose, at

your own risk, while acknowledging its author or contributor.

Third Party API's:

There are several types of payment APIs, which are often categorised based on their specific functionalities. Here are a few examples:

Transaction APIs

These are used to carry out payment transactions. They support functionalities such as initiating, refunding or cancelling payments.

Subscription APIs

These APIs are used for handling recurring payments, which are an important part of subscriptions. They allow businesses to bill customers automatically, at regular intervals.

Tokenisation APIs

These APIs allow for the secure storage of payment information. They replace sensitive payment data, such as credit card numbers, with a non-sensitive equivalent, known as a token.

Payout APIs

These APIs are used to send payments to individual accounts. Often, marketplaces or platforms use these APIs to send payments to sellers or service providers.

Pre-authorisation APIs

These APIs allow for the temporary holding of funds on a customer's card. This can be useful for situations in which a business needs to verify a customer's ability to pay before providing a service or shipping a product.

Data and reporting APIs

These APIs provide access to detailed data and analytics about payment transactions, which can be useful for understanding trends, monitoring performance and making informed business decisions.

Different payment service providers offer different sets of APIs with varying specific functionalities. Stripe, for

example, provides APIs that developers can integrate into their applications to handle payment transactions. We'll discuss Stripe's API capabilities further below.

Cloud Deployment:

Because it is a pay-per-use service, there is no substantial upfront fee, making it excellent for enterprises that require immediate access to resources.

No setup cost: The entire infrastructure is fully subsidized by the cloud service providers, thus there is no need to set up any hardware.

Infrastructure Management is not required: Using the public cloud does not necessitate infrastructure management.

No maintenance: The maintenance work is done by the service provider (not users).

Dynamic Scalability: To fulfill your company's needs, on-demand resources are accessible.

Disadvantages of the Public Cloud Model

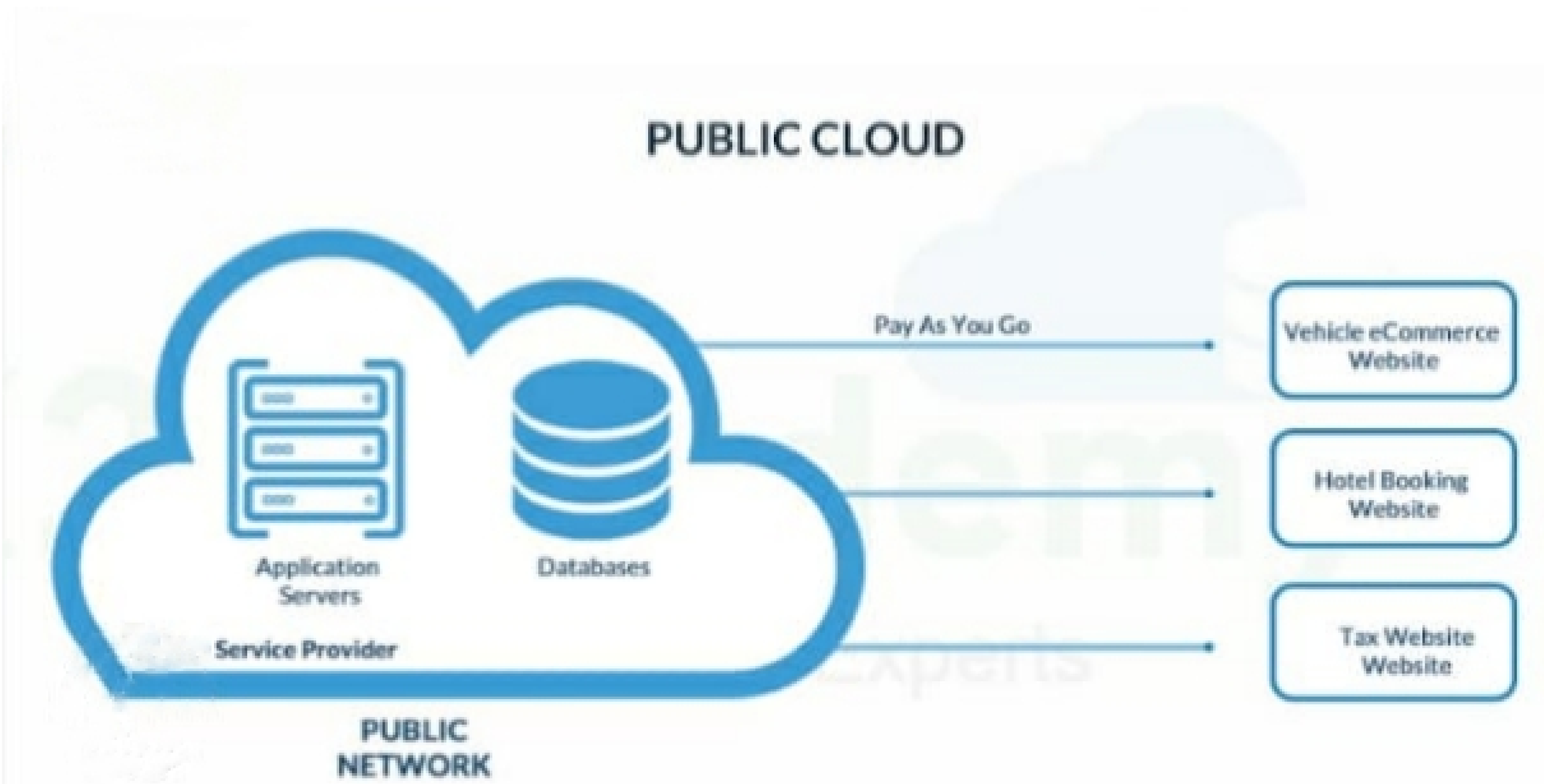
Less secure: Public cloud is less secure as resources are public so there is no guarantee of high-level security.

Low customization: It is accessed by many public so it can't be customized according to personal requirements.

Private Cloud

The private cloud deployment model is the exact opposite of the public cloud deployment model. It's a one-on-one environment for a single user (customer). There is no need to share

your hardware with anyone else. The distinction between private and public clouds is in how you handle all of the hardware. It is also called the “internal cloud” & it refers to the ability to access systems and services within a given border or organization. The cloud platform is implemented in a cloud-based secure environment that is protected by powerful firewalls and under the supervision of an organization’s IT department. The private cloud gives greater flexibility of control over cloud resources.


PUBLIC CLOUD
Pay As You Go
Vehicle eCommerce Website
Hotel Booking Website
Tax Website Website
Application Servers
Databases
Service Provider
PUBLIC NETWORK