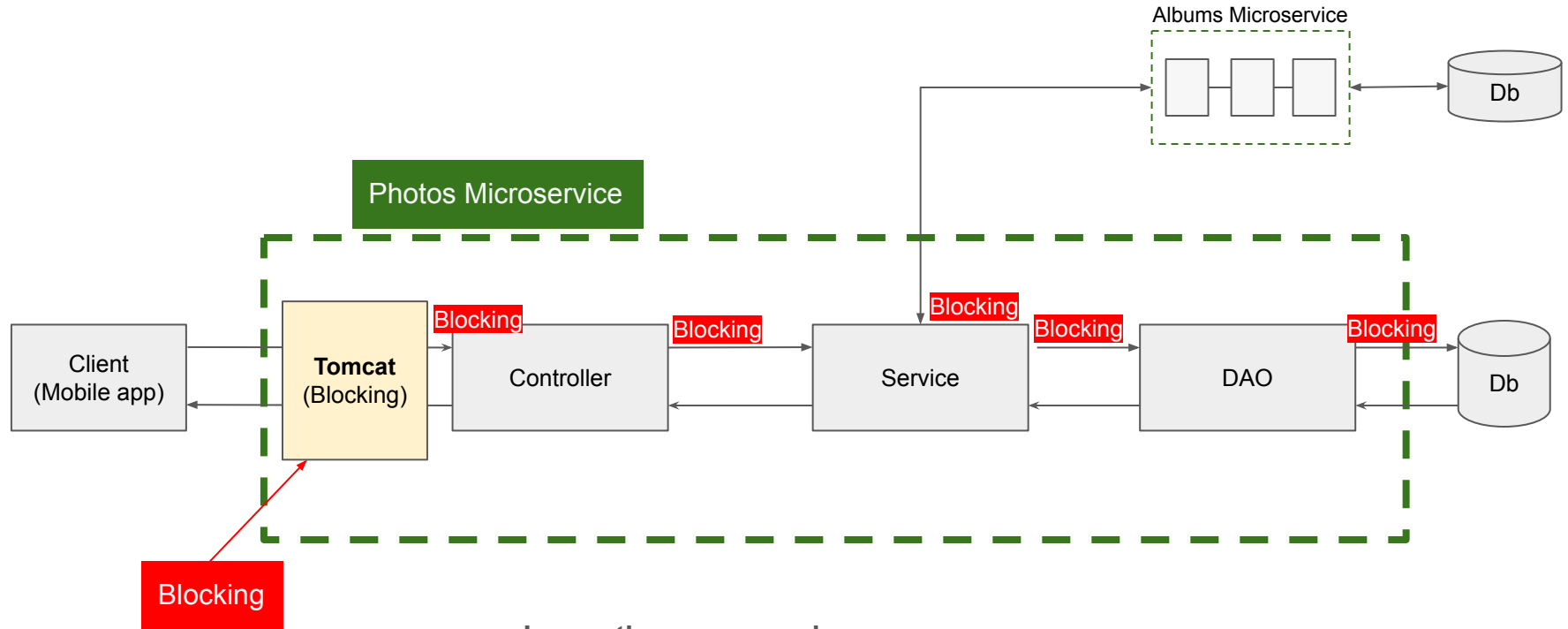


# Reactive Application

## Overview

# Traditional(Blocking) Spring MVC REST application



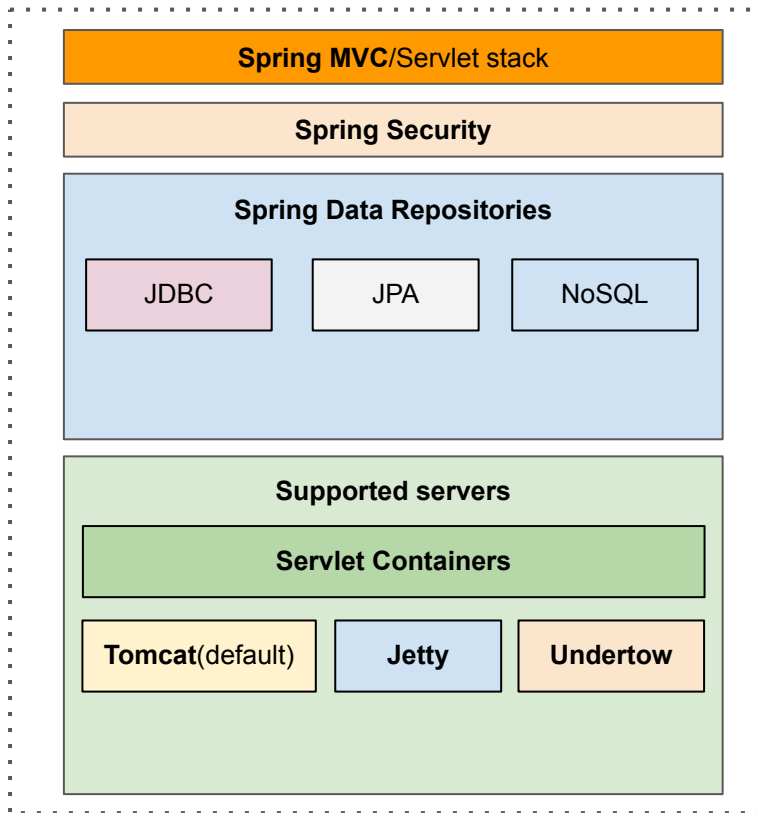
## Imperative programming:

- Sequential execution of code,
- Blocking I/O operations,
- Synchronous, step-by-step logic.

The diagram illustrates a non-blocking microservice architecture for a Photos application. A central dashed green box labeled "Photos Microservice" contains four components: "Netty Non Blocking" (yellow), "Controller", "Service", and "DAO" (all light purple). The flow is as follows: "Client (Mobile app)" (light purple) connects to "Netty Non Blocking". "Netty Non Blocking" connects to "Controller", which connects to "Service", which connects to "DAO". "DAO" connects to a "Db" (cylinder). External components include "Albums Microservice" (dashed green box with three light purple boxes) and its "Db" (cylinder). "Non-blocking" labels in green boxes point to the connections between "Netty Non Blocking" and "Controller", "Controller" and "Service", "Service" and "DAO", and "DAO" and its "Db". Another "Non-blocking" label points to the connection between "Service" and "Albums Microservice".

- Asynchronous data streams,
- Non-blocking I/O operations,
- Event-driven architecture.

## Traditional(Blocking) application



## Reactive(Non-blocking) application

