Comparison of Monolithic and Microservices Architecture for a URL Shortener Service

---

1. Introduction

In this document, we will explore two architectural approaches—monolithic and microservices—for building a URL shortener application. This application provides users with a service to shorten URLs, redirect users to the original URLs, and track click counts.

---

2. Overview of the Application Requirements

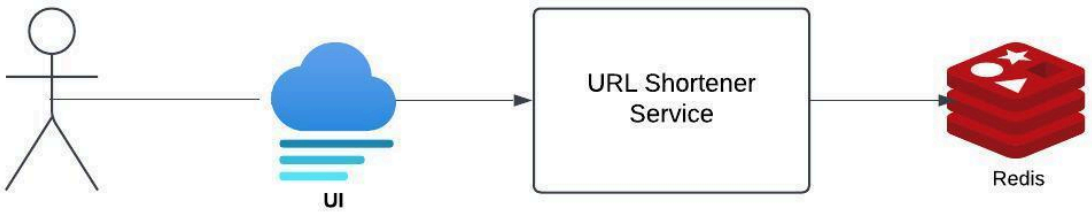
The URL shortener application consists of three primary functionalities:

- **URL Shortening**: Allows users to generate a shortened version of a long URL.
- **Redirection**: Redirects users from the shortened URL to the original URL.
- **Click Tracking**: Tracks the number of times each shortened URL is accessed.

To implement this functionality, we have considered two architectures: a monolithic approach and a microservices approach.

---

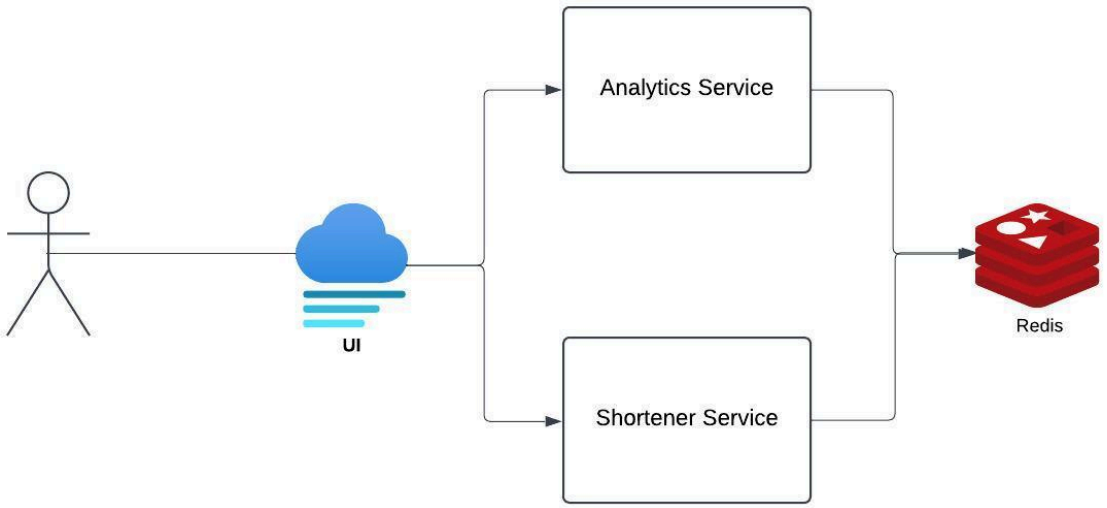
3. Monolithic Architecture



In the monolithic architecture, the entire application functionality is bundled into a single codebase and deployment unit. This structure includes:

- **URL Shortener Service:** A single service that handles all aspects of the application: URL shortening, redirection, and click tracking.
- Redis acts as a centralised data store to store URL mappings and click counts.

4. Microservices Architecture



The microservices architecture decomposes the monolithic application into smaller, independent services. Each service is designed to handle a specific functionality within the application:

- **Shortener Service**: Responsible for generating shortened URLs and storing them in Redis.
- **Analytics Service**: Manages redirection to the original URL and tracks the number of clicks.

In this architecture, Redis serves as a shared data store, with both services reading from and writing to it as needed.
