

Laporan Praktek Topik Khusus II

Menjalankan CRUD API dengan Golang, MySQL, dan Redis



SEMESTER VI

DISUSUN OLEH :

MUHAMMAD ABEL AL-FAHREZI (2211083034)

**PROGRAM STUDI TEKNOLOGI REKAYASA PERANGKAT LUNAK
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI PADANG
2025**

A. Landasan Teori

1. Golang (Go Language)

Golang, yang dikembangkan oleh Google, merupakan bahasa pemrograman sumber terbuka yang dirancang untuk efisiensi, kesederhanaan, dan keandalan. Dengan dukungan bawaan untuk konkurensi melalui goroutine dan channel, Golang sangat cocok untuk pengembangan aplikasi web, sistem terdistribusi, dan aplikasi jaringan. Fitur garbage collection otomatis mengurangi beban pengelolaan memori, sementara sifat statis diketiknya membantu mendeteksi kesalahan pada waktu kompilasi. Kinerja tinggi Golang menjadikannya pilihan populer untuk membangun API dan layanan web, dengan framework seperti Gin dan Echo yang mempermudah pengembangan aplikasi web.

2. API

API (Application Programming Interface) merupakan fondasi utama dalam pengembangan aplikasi web modern. API memungkinkan aplikasi yang berbeda untuk berkomunikasi dan bertukar data. Dalam konteks ini, RESTful API menjadi pilihan populer karena arsitekturnya yang berbasis pada prinsip-prinsip HTTP, memungkinkan operasi CRUD (Create, Read, Update, Delete) dilakukan dengan mudah dan efisien. Operasi CRUD sendiri merupakan inti dari manipulasi data, di mana 'Create' digunakan untuk membuat data baru, 'Read' untuk membaca data, 'Update' untuk memperbarui data, dan 'Delete' untuk menghapus data.

3. MySQL

MySQL, sebagai sistem manajemen basis data relasional (RDBMS), berperan sebagai penyimpanan data persisten. MySQL digunakan untuk menyimpan dan mengelola data yang akan diakses dan dimanipulasi oleh API. Keandalannya dan dukungannya yang luas menjadikannya pilihan yang tepat untuk aplikasi yang membutuhkan penyimpanan data yang kuat.

4. Redis

Redis, sebagai penyimpanan data dalam memori (in-memory data store), digunakan untuk meningkatkan performa API. Redis berfungsi sebagai cache untuk menyimpan data yang sering diakses, sehingga mengurangi beban pada MySQL dan mempercepat respons API. Selain itu, Redis juga dapat digunakan untuk manajemen sesi, menyimpan data sesi pengguna untuk meningkatkan pengalaman pengguna.

B. Tools

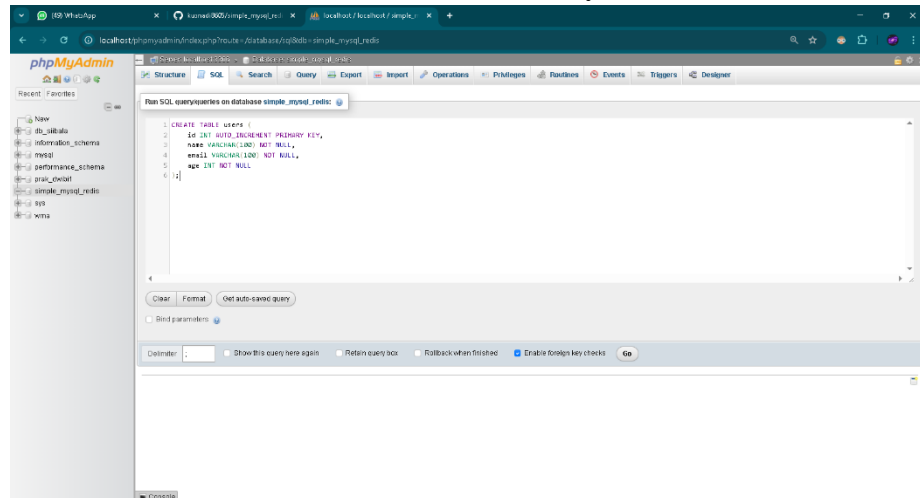
- MySQL
- Golang
- Visual Studio Code

- Postman
- Redis

C. Langkah Kerja

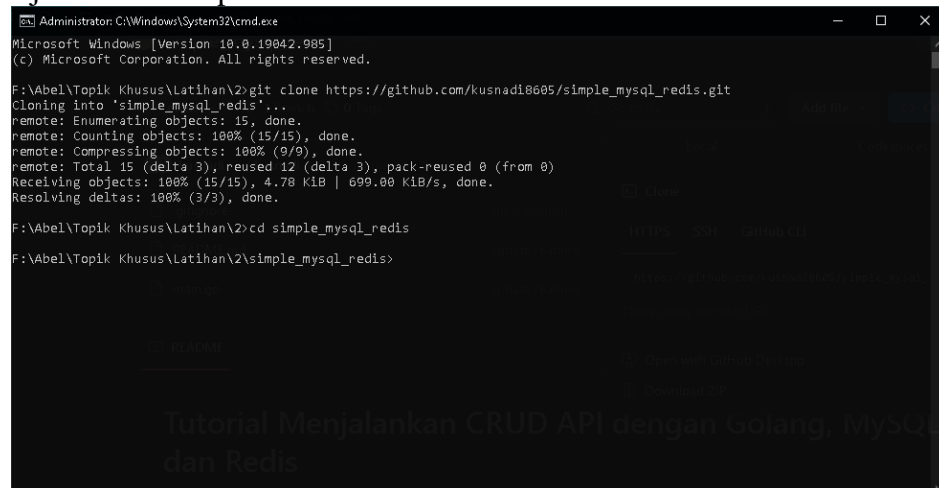
1. Setup Database

- Buat database dan create table users didalamnya



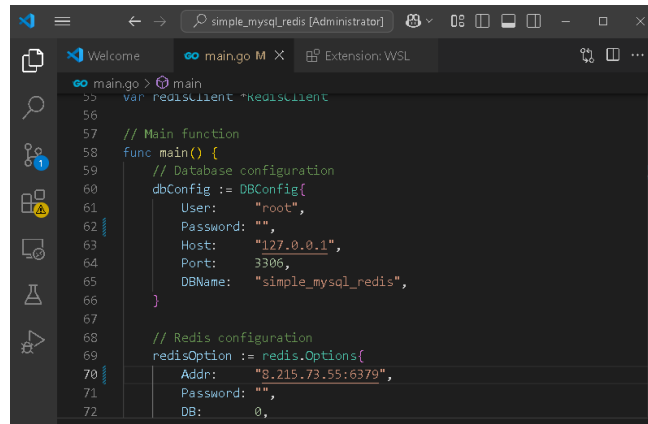
2. Clone Repository

- Setelah melakukan setup database lalu lakukan clone repository yang akan dijadikan bahan praktek



3. Konfigurasi Environment

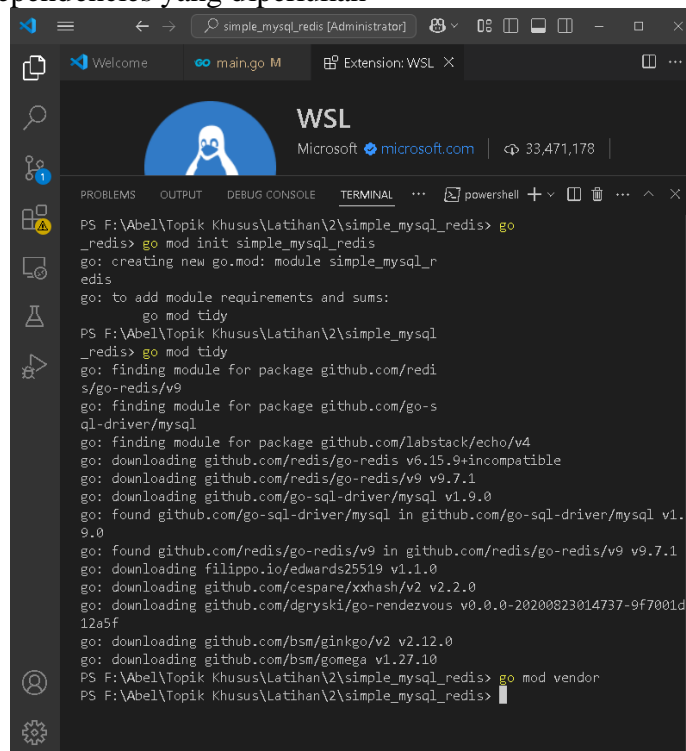
- Sesuaikan konfigurasi DB dan redi
- Ubah file main.go pada bagian dbConfig untuk menghubungkan nya dengan redis.



```
55 var redisClient *redis.Client
56
57 // Main function
58 func main() {
59     // Database configuration
60     dbConfig := DBConfig{
61         User: "root",
62         Password: "",
63         Host: "127.0.0.1",
64         Port: 3306,
65         DBName: "simple_mysql_redis",
66     }
67
68     // Redis configuration
69     redisOption := redis.Options{
70         Addr: "8.215.73.55:6379",
71         Password: "",
72         DB: 0,
```

4. Install Dependencies

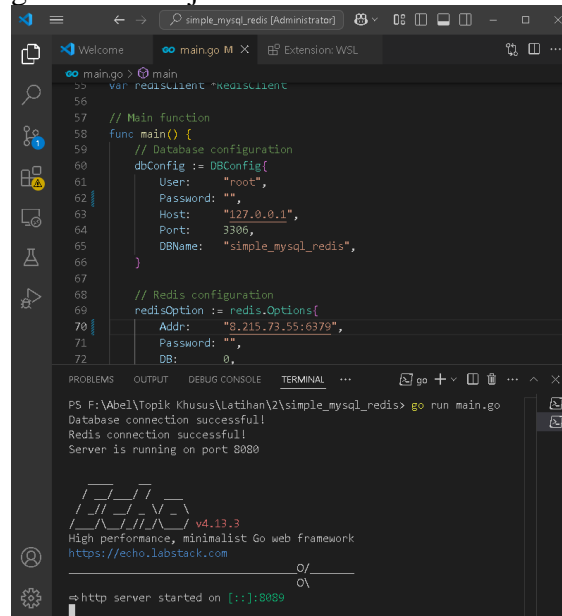
- Install dependencies yang diperlukan



```
PS F:\Abel\Topik Khusus\Latihan\2\simple_mysql_redis> go
_redis> go mod init simple_mysql_redis
go: creating new go.mod: module simple_mysql_r
edis
go: to add module requirements and sums:
go mod tidy
PS F:\Abel\Topik Khusus\Latihan\2\simple_mysql
_redis> go mod tidy
go: finding module for package github.com/redi
s/go-redis/v9
go: finding module for package github.com/go-s
ql-driver/mysql
go: finding module for package github.com/labstack/echo/v4
go: downloading github.com/redis/go-redis v6.15.9+incompatible
go: downloading github.com/redis/go-redis/v9 v9.7.1
go: downloading github.com/go-sql-driver/mysql v1.9.0
go: found github.com/go-sql-driver/mysql in github.com/go-sql-driver/mysql v1.
9.0
go: found github.com/redis/go-redis/v9 in github.com/redis/go-redis/v9 v9.7.1
go: downloading filippo.io/edwards25519 v1.1.0
go: downloading github.com/cespare/xxhash/v2 v2.2.0
go: downloading github.com/dgryski/go-rendezvous v0.0.0-20200823014737-9f7001d
12a5f
go: downloading github.com/bsm/ginkgo/v2 v2.12.0
go: downloading github.com/bsm/gomega v1.27.10
PS F:\Abel\Topik Khusus\Latihan\2\simple_mysql_redis> go mod vendor
PS F:\Abel\Topik Khusus\Latihan\2\simple_mysql_redis>
```

5. Jalankan Server

- Run file main.go untuk menjalankan server.



```
main.go
56 var redisClient *redis.Client
57
58 // Main function
59 func main() {
60     // Database configuration
61     dbConfig := DBConfig{
62         User: "root",
63         Password: "",
64         Host: "127.0.0.1",
65         Port: 3306,
66         DBName: "simple_mysql_redis",
67     }
68
69     // Redis configuration
70     redisOption := redis.Options{
71         Addr: "0.215.73.55:6379",
72         Password: "",
73         DB: 0,
74     }
75
76     // Connect to database
77     db, err := sql.Open("mysql", dbConfig.User+":"+dbConfig.Password+"@("+dbConfig.Host+":"+dbConfig.Port+")/"+dbConfig.DBName)
78     if err != nil {
79         log.Fatalf("Error connecting to database: %v", err)
80     }
81
82     // Connect to Redis
83     rdb := redis.NewClient(&redisOption)
84     if err := rdb.Ping().Err(); err != nil {
85         log.Fatalf("Error connecting to Redis: %v", err)
86     }
87
88     // Create a new user
89     createNewUser(db, rdb)
90
91     // Start the server
92     http.ListenAndServe(":8080", nil)
93 }
```

PS F:\Vabel\Topik Khusus\Latihan\2\simple_mysql_redis> go run main.go

Database connection successful!

Redis connection successful!

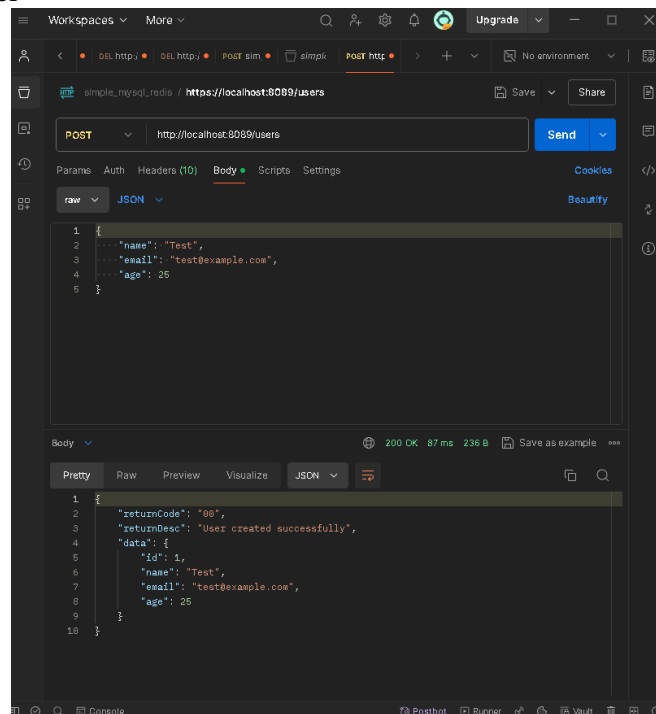
Server is running on port 8080

High performance, minimalist Go web framework
<https://echo.labstack.com>

http server started on [::]:8080

6. API Endpoint menggunakan Postman

- Create user



Workspaces More

simple_mysql_redis / https://localhost:8080/users

POST http://localhost:8080/users

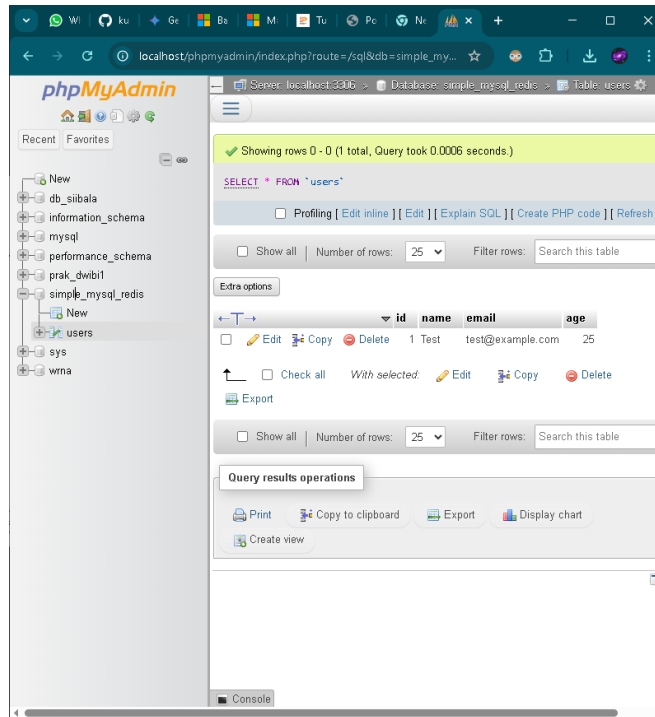
Body

```
1 {
2   "name": "Test",
3   "email": "test@example.com",
4   "age": 25
5 }
```

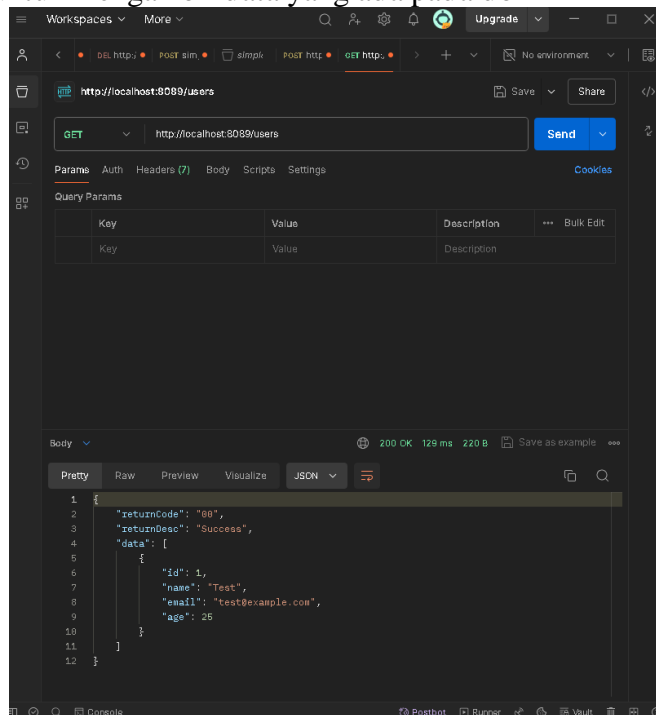
Body

```
1 {
2   "statusCode": 200,
3   "statusCode": "User created successfully",
4   "data": {
5     "id": 1,
6     "name": "Test",
7     "email": "test@example.com",
8     "age": 25
9   }
10 }
```

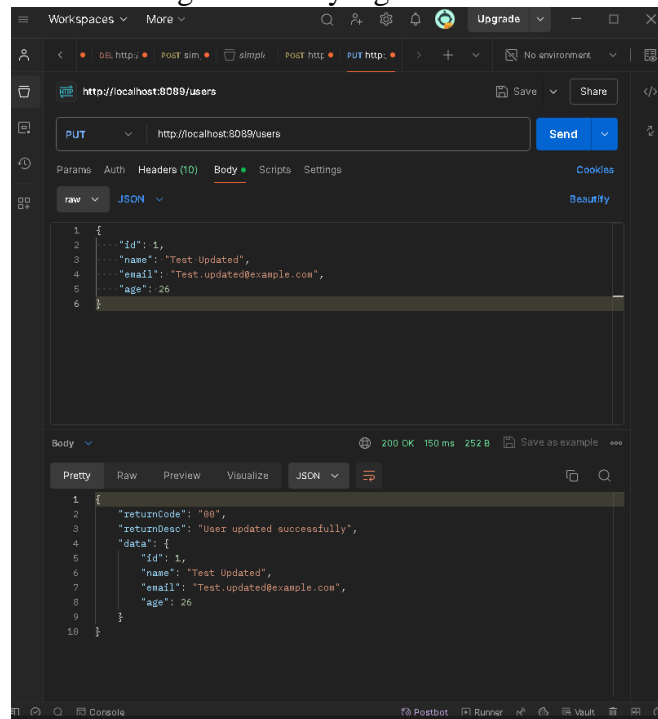
- Data berhasil ditambahkan ke db



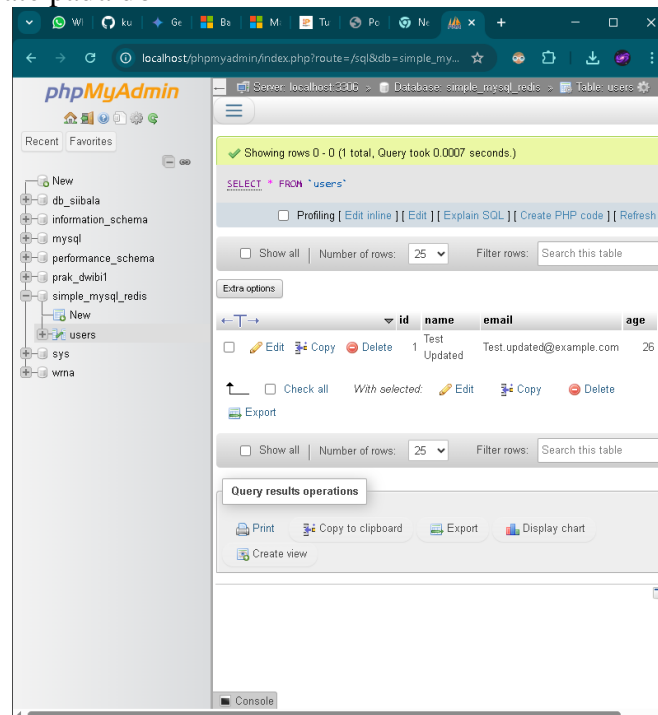
- Get user untuk mengambil data yang ada pada db



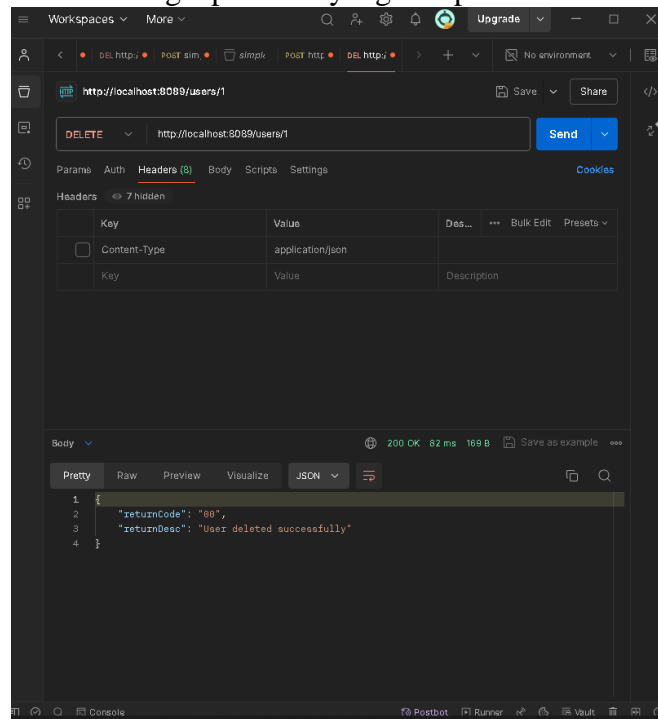
- Update user untuk mengubah data yang telah ada



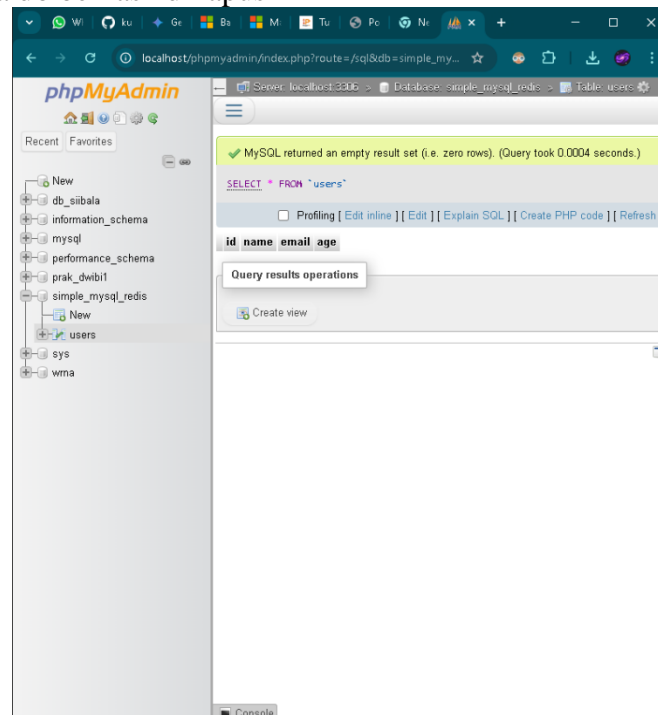
- Hasil update pada db



- Delete user untuk menghapus data yang ada pada db



- Data pada db berhasil dihapus



- Semua proses CRUD berhasil dilakukan.

D. Kesimpulan

Berdasarkan praktik yang telah dilakukan, berhasil menunjukkan integrasi yang efisien antara ketiga teknologi tersebut, di mana Golang memberikan performa tinggi untuk logika aplikasi, MySQL menyediakan penyimpanan data persisten yang andal, dan Redis meningkatkan kecepatan akses data melalui caching yang signifikan. Implementasi operasi CRUD membuktikan kemampuan teknologi-teknologi ini dalam manipulasi data yang efisien, memberikan contoh konkret tentang pengembangan aplikasi web modern yang skalabel dan berperforma tinggi, serta memperdalam pemahaman tentang konsep-konsep penting seperti API, RESTful API, dan caching, yang secara keseluruhan meningkatkan skalabilitas dan responsivitas aplikasi.