



**Nama** : MuhammadNaufal  
**NIM** : 22060024  
**Prodi** : Teknik Informatika / Reg. Pagi  
**Matkul** : Pengembangan Aplikasi Bergerak  
**Subject** : Persamaan dan perbedaan Ionic Framework, Quasar dan Framework7

Framework hybrid	Persamaan	Perbedaan
Ionic Framework	<ul style="list-style-type: none"> <li>❑ Multi_platform : Ketiganya dapat digunakan untuk membangun aplikasi yang berjalan di berbagai platform, termasuk IOS, Android, dan web.</li> <li>❑ Komponen UI : Ketiganya menyediakan berbagai komponen UI bawaan yang dapat anda gunakan untuk membangun aplikasi dengan cepat dan mudah.</li> </ul>	<ul style="list-style-type: none"> <li>❑ Didukung oleh Capacitor : Memungkinkan membangun aplikasi native dengan kinerja tinggi.</li> <li>❑ Komunitas besar : Memiliki komunitas pengembang yang besar dan banyak plugin dan pustaka.</li> <li>❑ Lebih Kompleks : Memiliki kurva belajar yang lebih curam disbanding dengan Framework7.</li> </ul>
Quasar Framework	<ul style="list-style-type: none"> <li>❑ Komunitas : Ketiganya memiliki komunitas pengembang yang aktif dan banyak sumber daya tersedia untuk membantu memulainya.</li> </ul>	<ul style="list-style-type: none"> <li>❑ Berdasarkan Vue.js : Dibangun di atas Vue.js, yang merupakan kerangka kerja JavaScript populer.</li> <li>❑ Desain responsif: Menyediakan banyak fitur untuk membantu Anda membuat aplikasi responsif yang terlihat bagus di berbagai perangkat.</li> <li>❑ Lebih baru: Quasar Framework adalah kerangka kerja yang lebih baru dibandingkan dengan Ionic Framework dan Framework7.</li> </ul>

Framework7		<ul style="list-style-type: none"> <li>□ Ringan dan cepat: kerangka kerja yang ringan dan cepat, yang membuatnya ideal</li> </ul>
		<p>untuk aplikasi yang membutuhkan kinerja tinggi.</p> <ul style="list-style-type: none"> <li>□ Mudah digunakan: mudah dipelajari dan digunakan, bahkan untuk pengembang pemula.</li> <li>□ Fitur yang lebih sedikit: Framework7 memiliki fitur yang lebih sedikit dibandingkan dengan Ionic Framework dan Quasar.</li> </ul>

## 1. Typescript Asinkronus

```

C: > Users > hp > typescript-file-reader > src > TS index.ts > readFileSync
1  import { promises as fs } from 'fs';
2  import { join } from 'path';
3
4  async function readFileAsync(filePath: string): Promise<void> {
5      try {
6          const data = await fs.readFile(filePath, 'utf-8');
7          console.log(data);
8      } catch (error) {
9          console.error('Error reading file:', error);
10     }
11 }
12
13 // Gunakan __dirname untuk mendapatkan direktori saat ini dan kembali satu level ke direktori src
14 const filePath = join(__dirname, '..', 'src', 'example.txt');
15 readFileAsync(filePath);
16
C: > Users > hp > typescript-file-reader > src > example.txt
1  echo "Ini adalah contoh isi file yang akan dibaca secara asinkron oleh program TypeScript."
2  |

```

```

C: > Users > hp > typescript-file-reader > {} package.json > {} scripts
1  {
2    "name": "typescript-file-reader",
3    "version": "1.0.0",
4    "main": "index.js",
5    "scripts": {
6      "build": "tsc && cp src/example.txt dist/",
7      "start": "node dist/index.js"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC",
12   "description": "",
13   "devDependencies": {
14     "@types/node": "^20.12.12",
15     "typescript": "^5.4.5"
16   }
17 }
18

```

output

```

C:\Users\hp\typescript-file-reader>npm start

> typescript-file-reader@1.0.0 start
> node dist/index.js

echo "Ini adalah contoh isi file yang akan dibaca secara asinkron oleh program TypeScript." >
C:\Users\hp\typescript-file-reader>

```

## 2. RESTful API

```

C: > Users > hp > readfile-api > {} package.json > ...
1  {
2    "name": "readfile-api",
3    "version": "1.0.0",
4    "main": "index.js",
5    "scripts": {
6      "start": "ts-node src/server.ts"
7    },
8    "keywords": [],
9    "author": "",
10   "license": "ISC",
11   "dependencies": {
12     "@types/express": "^4.17.21",
13     "@types/node": "^20.12.12",
14     "body-parser": "^1.20.2",
15     "express": "^4.19.2",
16     "ts-node": "^10.9.2",
17     "typescript": "^5.4.5"
18   },
19   "description": ""
}

C: > Users > hp > readfile-api > src > TS server.ts > ...
1  import express from 'express';
2  import bodyParser from 'body-parser';
3  import fs from 'fs/promises';
4  import path from 'path';
5
6  const app = express();
7  const port = 3000;
8
9  app.use(bodyParser.json());
10
11 // Endpoint untuk membaca dan menampilkan file secara asinkron
12 app.get('/readfile', async (req, res) => {
13   try {
14     const filePath = path.join(__dirname, 'data.txt'); // Pastikan file ada
15     const data = await fs.readFile(filePath, 'utf-8');
16     res.send(data);
17   } catch (error) {
18     res.status(500).send({ error: 'Error reading file' });
19   }
20 });
21
22 app.listen(port, () => {
23   console.log(`Server is running on http://localhost:${port}`);
24 });
25

```

```

C: > Users > hp > readfile-api > src > data.txt
1  Hallo nama saya Yati Sumiati
2

```

Output :

```
C:\Users\hp\readfile-api>npm start  
  
> readfile-api@1.0.0 start  
> ts-node src/server.ts  
  
Server is running on http://localhost:3000
```

