Nama : **Yati Sumiati**
NIM : **22060025**
Prodi : **Teknik Informatika / Reg. Pagi**
Matkul : **Aplikasi Berbasis Platform**
Subject : **Persamaan dan perbedaan Ionic Framework, Quasar dan Framework7**

Framework hybrid	Persamaan	Perbedaan
Ionic Framework	• Multi_platform : Ketiganya dapat digunakan untuk membangunaplikasi yang berjalan di berbagai platform, termasuk IOS, Android, dan web. • Komponen UI : Ketiganya menyediakan berbagai komponen UI bawaan yang dapat anda gunakan untuk membangun aplikasi dengan cepat dan mudah. • Komunitas : Ketiganya memiliki komunitas pengembang yang aktif dan banyak sumber daya tersedia untuk membantu memulainya.	• Didukung oleh Capacitor : Memungkinkan membangun aplikasi native dengan kinerja tinggi. • Komunitas besar : Memiliki komunitas pengembang yang besar dan banyak pligin dan pustaka. • Lebih Kompleks : Memiliki kurva belajar yang lebih curam disbanding dengan Framework7.
Quasar Framework		• Berdasarkan Vue.js : Dibangun di atas Vue.js, yang merupakan kerangka kerja JavaScript popular. • Desain responsif: Menyediakan banyak fitur untuk membantu Anda membuat aplikasi responsif yang terlihat bagus di berbagai perangkat.

		Lebih baru: Quasar Framework adalah kerangka kerja yang lebih baru dibandingkan dengan Ionic Framework dan Framework7.
Framework7		Ringan dan cepat: kerangka kerja yang ringan dan cepat, yang membuatnya ideal
		untuk aplikasi yang membutuhkan kinerja tinggi. Mudah digunakan: mudah dipelajari dan digunakan, bahkan untuk pengembang pemula. Fitur yang lebih sedikit: Framework7 memiliki fitur yang lebih sedikit dibandingkan dengan Ionic Framework dan Quasar.

Hasil Output Pengerjaan Typescript Asyncronus

1. Typescript Asyncronus

C: > Users > hp > typescript-file-reader > src > example.txt

```
echo "Ini adalah contoh isi file yang akan dibaca secara asinkron oleh program TypeScript."
```

C: > Users > hp > typescript-file-reader > {} package.json > {} scripts

```
{
  "name": "typescript-file-reader",
  "version": "1.0.0",
  "main": "index.js",
  Debug
  "scripts": {
    "build": "tsc && cp src/example.txt dist/",
    "start": "node dist/index.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "devDependencies": {
    "@types/node": "^20.12.12",
    "typescript": "^5.4.5"
  }
}
```

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 > ...

```json
{
  "name": "readfile-api",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "start": "ts-node src/server.ts"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "@types/express": "^4.17.21",
    "@types/node": "^20.12.12",
    "body-parser": "^1.20.2",
    "express": "^4.19.2",
    "ts-node": "^10.9.2",
    "typescript": "^5.4.5"
  },
  "description": ""
```

C: > Users > hp > readfile-api > src > server.ts > ...

```ts
import express from 'express';
import bodyParser from 'body-parser';
import fs from 'fs/promises';
import path from 'path';

const app = express();
const port = 3000;

app.use(bodyParser.json());

// Endpoint untuk membaca dan menampilkan file secara asinkron
app.get('/readfile', async (req, res) => {
    try {
        const filePath = path.join(__dirname, 'data.txt'); // Pastikan f
        const data = await fs.readFile(filePath, 'utf-8');
        res.send(data);
    } catch (error) {
        res.status(500).send({ error: 'Error reading file' });
    }
});

app.listen(port, () => {
    console.log(`Server is running on http://localhost:${port}`);
});
```

C: > Users > hp > readfile-api > src > data.txt

```
Hallo nama saya Yati Sumiati
```

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
```