Angular + Node.js App

### 1. Backend: Node.js + Express + MySQL

#### Előkészületek

1. Hozz létre egy új mappát a backend számára, majd inicializáld a Node.js projektet:

```bash
mkdir backend
cd backend
npm init -y
```

2. Telepítsd a szükséges csomagokat:

```bash
npm install express mysql2 cors body-parser
```

#### MySQL adatbázis beállítása

1. Hozz létre egy MySQL adatbázist `angular_app` néven.
2. Hozz létre egy `users` táblát az alábbi oszlopokkal:

```sql
CREATE DATABASE angular_app;
USE angular_app;

CREATE TABLE users (
```

```
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    email VARCHAR(100)
);
```

### 3. Adatok hozzáadása teszteléshez:

```sql
INSERT INTO users (name, email) VALUES ('John Doe', 'john@example.com'), ('Jane Smith', 'jane@example.com');
```

#### REST API készítése Node.js-ben

**index.js**:

```javascript
const express = require('express');
const mysql = require('mysql2');
const cors = require('cors');
const bodyParser = require('body-parser');

const app = express();
app.use(cors());
app.use(bodyParser.json());

// MySQL kapcsolat beállítása
const db = mysql.createConnection({
```

```
  host: 'localhost',
  user: 'yourusername',
  password: 'yourpassword',
  database: 'angular_app'
});

db.connect(err => {
  if (err) throw err;
  console.log('MySQL kapcsolódva.');
});

// Felhasználók lekérdezése
app.get('/api/users', (req, res) => {
  db.query('SELECT * FROM users', (err, results) => {
    if (err) throw err;
    res.json(results);
  });
});

// Új felhasználó hozzáadása
app.post('/api/users', (req, res) => {
  const { name, email } = req.body;
  db.query('INSERT INTO users (name, email) VALUES (?, ?)', [name, email], (err,
results) => {
    if (err) throw err;
    res.json({ id: results.insertId, name, email });
  });
```

});

// Szerver indítása
const PORT = 3000;
app.listen(PORT, () => {
  console.log(`Server running on http://localhost:${PORT}`);
});
```

> **Megjegyzés**: A `yourusername` és `yourpassword` értékeket cseréld le a MySQL hozzáférési adataidra.

### 2. Frontend: Angular alkalmazás készítése

#### Új Angular projekt létrehozása

```bash
ng new angular-app
cd angular-app
```

#### HttpClient modul importálása

Az Angular HTTP kliens használatához importáld az `HttpClientModule`-t a `src/app/app.module.ts` fájlba:

```typescript
import { HttpClientModule } from '@angular/common/http';

```
@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    HttpClientModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

#### Szerviz készítése az API hívásokhoz

Készíts egy szervizt, amely az API hívásokat kezeli:

```bash
ng generate service user
```

**user.service.ts**:

```typescript
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';
```

```
@Injectable({
  providedIn: 'root'
})
export class UserService {
  private apiUrl = 'http://localhost:3000/api/users';

  constructor(private http: HttpClient) { }

  getUsers(): Observable<any> {
    return this.http.get<any>(this.apiUrl);
  }

  addUser(user: { name: string, email: string }): Observable<any> {
    return this.http.post<any>(this.apiUrl, user);
  }
}
```

#### Komponens készítése az adatok megjelenítéséhez

Készíts egy `UserList` komponenst a felhasználók megjelenítéséhez és új felhasználó hozzáadásához:

```bash
ng generate component user-list
```

**user-list.component.ts**:

```typescript
import { Component, OnInit } from '@angular/core';
import { UserService } from '../user.service';

@Component({
  selector: 'app-user-list',
  templateUrl: './user-list.component.html',
  styleUrls: ['./user-list.component.css']
})
export class UserListComponent implements OnInit {
  users: any[] = [];
  newUser = { name: '', email: '' };

  constructor(private userService: UserService) { }

  ngOnInit(): void {
    this.loadUsers();
  }

  loadUsers(): void {
    this.userService.getUsers().subscribe(data => {
      this.users = data;
    });
  }

  addUser(): void {
```

```
    this.userService.addUser(this.newUser).subscribe(user => {
      this.users.push(user);
      this.newUser = { name: '', email: '' };
    });
  }
}
```

**user-list.component.html**:
```html
<h2>Felhasználók</h2>
<ul>
  <li *ngFor="let user of users">
    {{ user.name }} - {{ user.email }}
  </li>
</ul>

<h3>Új felhasználó hozzáadása</h3>
<form (ngSubmit)="addUser()">
  <input [(ngModel)]="newUser.name" name="name" placeholder="Név" required>
  <input [(ngModel)]="newUser.email" name="email" placeholder="Email" required>
  <button type="submit">Hozzáadás</button>
</form>
```

#### AppComponent frissítése

Győződj meg róla, hogy az `AppComponent` tartalmazza a `UserListComponent`-et:

**app.component.html**:

```html
<app-user-list></app-user-list>
```

### Alkalmazás indítása

1. Indítsd el a backend szervert:
   ```bash
   node index.js
   ```

2. Indítsd el az Angular alkalmazást:
   ```bash
   ng serve
   ```

Most már elérheted az alkalmazást a `http://localhost:4200` címen, és láthatod az adatokat a MySQL adatbázisból. Az új felhasználók hozzáadásakor a backend frissíti az adatbázist.