Code

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

const app = express();

// Middleware for parsing JSON bodies
app.use(bodyParser.json());  // parse application/json&#8203;:contentReference[oaicite:10]{index=10}
```

```
// Create MySQL connection pool
const db = mysql.createPool({
 host: 'mysql1',       // MySQL service name in Docker (or "localhost" if running locally)
 user: 'root',         // MySQL username
 password: 'admin_xxx', // MySQL password
 database: 'db1'  // MySQL database name
});
```

```
db.getConnection((err, connection) => {
 if (err) {
   console.error('Error connecting to MySQL:', err);
   process.exit(1); // exit gracefully if DB connection fails
 }
```

```
const createTableQuery = `
   CREATE TABLE IF NOT EXISTS users (
     id INT AUTO_INCREMENT PRIMARY KEY,
     name VARCHAR(255),
     email VARCHAR(255)
   )`;
```

```
connection.query(createTableQuery, error => {
   connection.release(); // release the connection back to pool
   if (error) {
     console.error('Error creating users table:', error);
     // (We continue startup even if this fails, but in a real setup you might want to handle it)
   }
   // Start the server after ensuring DB setup
   app.listen(8080, () => {
     console.log('Server is running on port 8080');
   });
 });
});
```

```
app.get('/users', (req, res) => {
 db.query('SELECT * FROM users', (err, results) => {
   if (err) {
     console.error('DB error on SELECT:', err);
     return res.status(500).json({ error: 'Database error while fetching users' });
   }
   res.json(results);  // send array of users
 });
});
```

Move towards Promise

```
// Create MySQL connection pool and wrap it with Promise support
const pool = mysql.createPool({
 host: 'mysql1',         // MySQL service name in Docker (or "localhost" if running locally)
 user: 'root',           // MySQL username
 password: 'admin_xxx', // MySQL password
 database: 'db1'         // MySQL database name
});
const db = pool.promise();  // Use promise-based pool
const createTableQuery = `
 CREATE TABLE IF NOT EXISTS users (
   id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(255),
   email VARCHAR(255)
 )
`;
```

```
db.query(createTableQuery)
 .then(() => {
   console.log('Users table ensured.');
   // Start the server after successful DB setup
   app.listen(8080, () => {
     console.log('Server is running on port 8080');
   });
 })
 .catch(err => {
   console.error('Error setting up the database:', err);
   process.exit(1); // Exit gracefully if the DB setup fails
 });
```

```
app.get('/users', (req, res) => {
 db.query('SELECT * FROM users')
   .then(([rows]) => {
     res.json(rows);
   })
   .catch(err => {
     console.error('DB error on SELECT:', err);
     res.status(500).json({ error: 'Database error while fetching users' });
   });
});
```

Arguments in docker-compose.yml

```yaml
node1:
   build: .
   container_name: nodejs1
   ports:
     - "80:8080"
   volumes:
     - /Users/ralph/classes/353/w6:/usr/src/app
   environment:
     DB_HOST: mysql1         # points to the mysql1 service name
     DB_PORT: 3306           # default MySQL port
     DB_DATABASE: my_database
     DB_USER: user1
     DB_PASSWORD: user1_xxx
   depends_on:
     mysql1:
       condition: service_started
   stdin_open: true
   tty: true
```

```
const pool = mysql.createPool({
 host: process.env.DB_HOST || 'mysql1',              // From Docker Compose: DB_HOST
 port: process.env.DB_PORT || 3306,                    // From Docker Compose: DB_PORT
 user: process.env.DB_USER || 'user1',                 // From Docker Compose: DB_USER
 password: process.env.DB_PASSWORD || 'user1_xxx',     // From Docker Compose: DB_PASSWORD
 database: process.env.DB_DATABASE || 'my_database'    // From Docker Compose: DB_DATABASE
});
const db = pool.promise();
```

```yaml
node1:
   build: .
   container_name: nodejs1
   ports:
     - "80:8080"
   volumes:
     - /Users/ralph/classes/353/w6:/usr/src/app
   environment:
     DB_HOST: mysql1         # points to the mysql1 service name
     DB_PORT: 3306           # default MySQL port
     DB_DATABASE: my_database
     DB_USER: user1
     DB_PASSWORD: user1_xxx
   depends_on:
     mysql1:
       condition: service_started
   stdin_open: true
   tty: true
```