Task API

A simple RESTful API for managing tasks, built with PHP, PDO, and SQLite/MySQL. Supports creating, reading, updating, and deleting (CRUD) task records.

Project Structure

```
task-api/
├── public/             # Public web root for PHP built-in server
│   ├── api.php         # Main API entry point (handles routing and JSON 
responses)
│   └── db.php          # Database connection class (SQLite or MySQL)
├── restful_api.sql     # Database schema file for MySQL/SQLite
└── README.md           # This documentation
```

Getting Started

Requirements

- PHP 7.4+ (with PDO SQLite or PDO MySQL extension)
- Composer (optional for future enhancements)

Run Locally

1. Clone the repository:

```
git clone https://github.com/thooyamani/task-api.git
cd task-api
```

1. Start the PHP built-in server:

```
php -S localhost:8000 -t public
```

1. Access the API at:

```
http://localhost:8000
```

1. Initialize the database:
2. For SQLite, the table is auto-created on first run.

3. For MySQL, import the schema from `restful_api.sql`:

```
mysql -u your_username -p your_database < restful_api.sql
```

Database Schema (restful_api.sql)

```
CREATE TABLE task (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    description TEXT NOT NULL,
    status VARCHAR(50) DEFAULT 'pending'
);
```

API Endpoints

Get All Tasks

Request:

```
GET /tasks
```

Response:

```
[
  { "id": 1, "title": "Task 1", "description": "Description 1", "status":
"pending" },
  ...
]
```

Get Single Task by ID

Request:

```
GET /tasks/{id}
```

Example:

```
GET /tasks/1
```

Response:

```
{ "id": 1, "title": "Task 1", "description": "Description 1", "status":
"pending" }
```

Create New Task

Request:

```
POST /tasks
Content-Type: application/json

{
  "title": "New Task",
  "description": "This is a new task"
}
```

Response:

```
{ "message": "Task created", "id": 5 }
```

Update Existing Task

Request:

```
POST /tasks/{id}
Content-Type: application/json

{
  "title": "Updated Task",
  "status": "completed"
}
```

Example:

```
POST /tasks/1
```

Response:

```
{ "message": "Task updated" }
```

Delete a Task

Request:

```
DELETE /tasks/{id}
```

Example:

```
DELETE /tasks/1
```

Response:

```
{ "message": "Task deleted" }
```

Design Decisions

- **Routing:** Simple PATH_INFO based routing using native PHP ($_SERVER).
- **Database:**
- Defaults to SQLite.
- Can be switched to MySQL by modifying public/db.php.
- **PDO Prepared Statements:** Protects against SQL injection.
- **Built-in Server:** Use php -S to run without Apache/Nginx.

Security

- Uses prepared statements.
- Validates task ID (is_numeric).
- Returns proper HTTP status codes.

Future Improvements

- Add PUT and PATCH methods for better RESTful semantics.
- Add user authentication (JWT/OAuth).
- Add Redis/File caching for GET requests.
- Add OpenAPI/Swagger documentation.
- Add unit tests.

Author

Thooyamani https://github.com/thooyamani