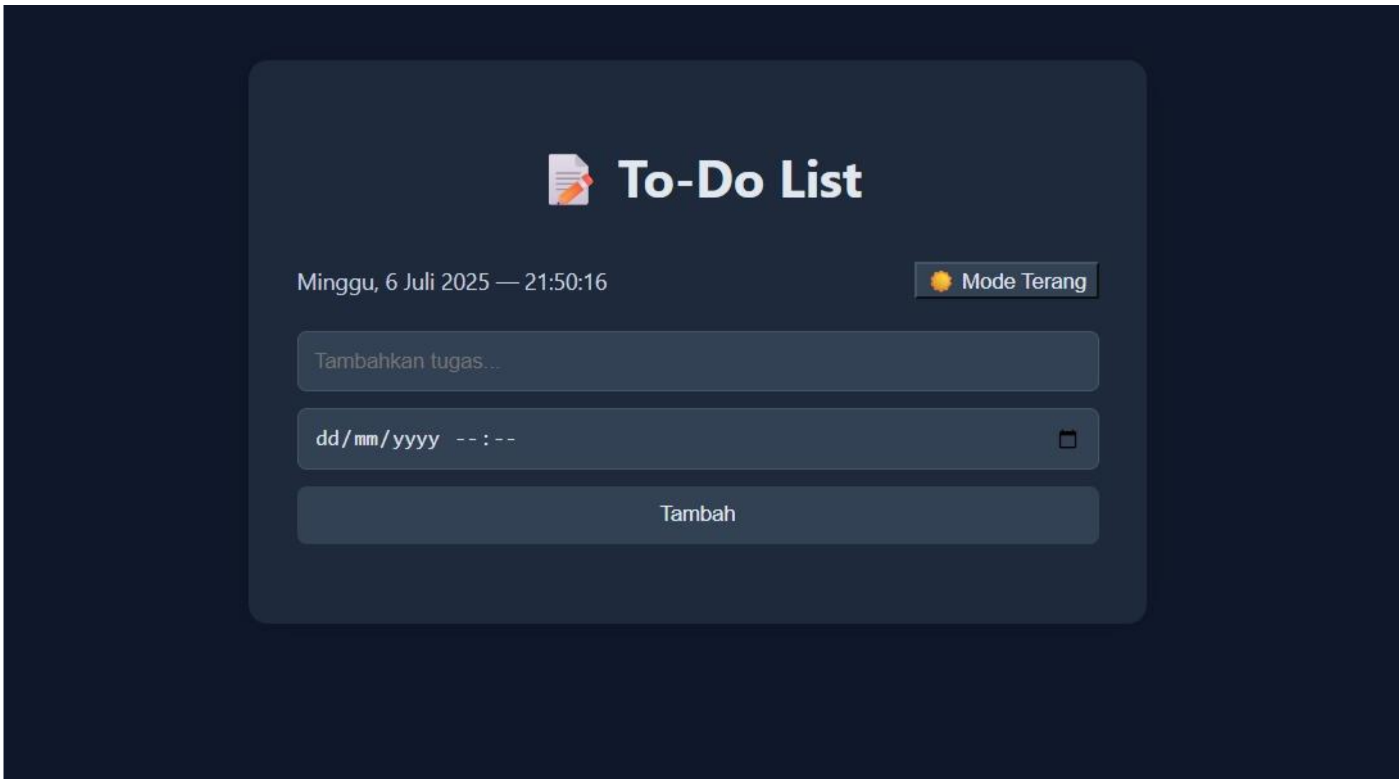
Mid Exam

Semester 20243

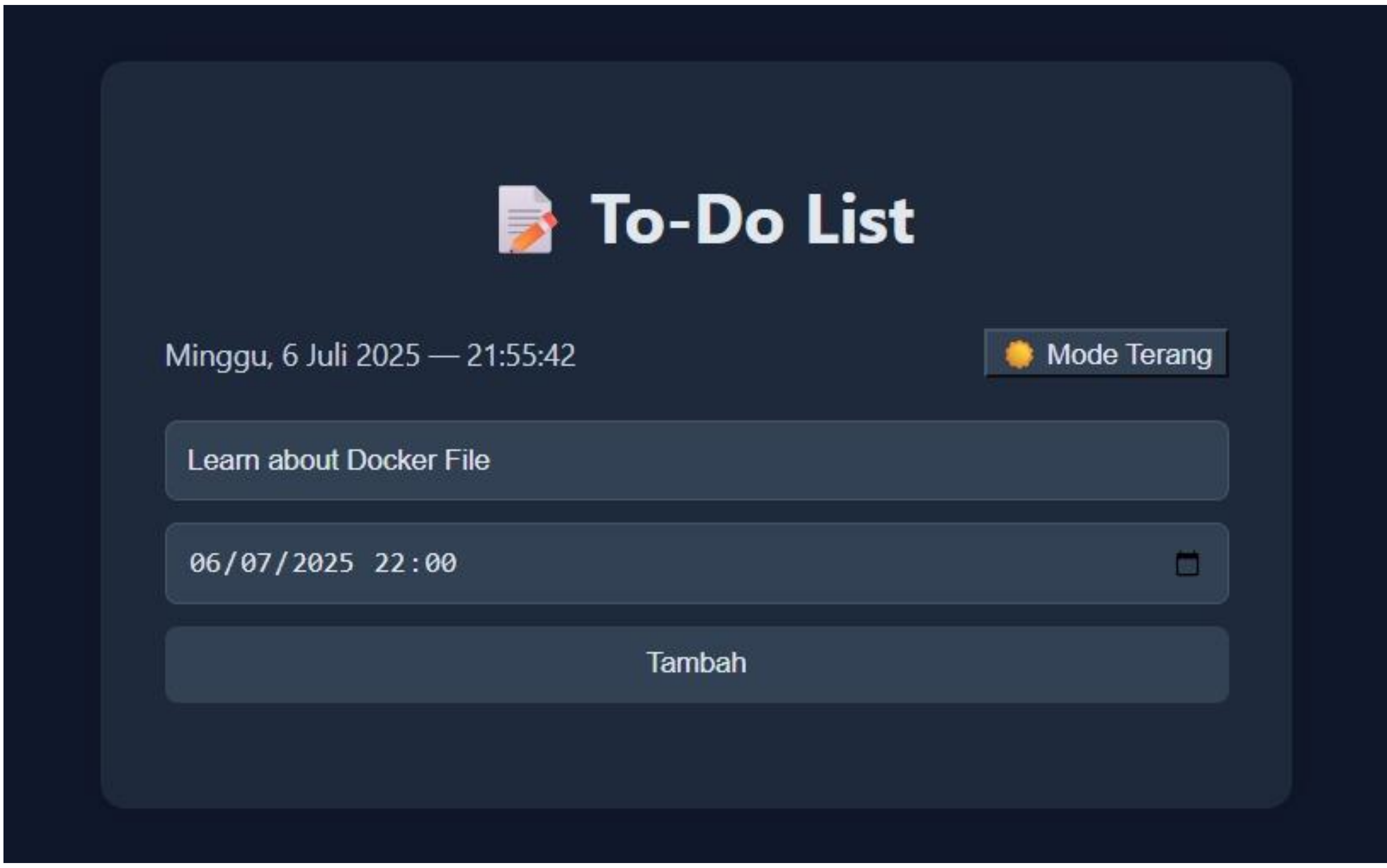
Subject	: **Distributed and Parallel System**
Study Program	: Informatics
Student Name	: 1. M. Rizki Kurniawan 2. Dean Adwitiya Pandana
Student ID	: 1. 001202300037 2. 001202300108

To_Do_Lists

- start menu



- adding to do list



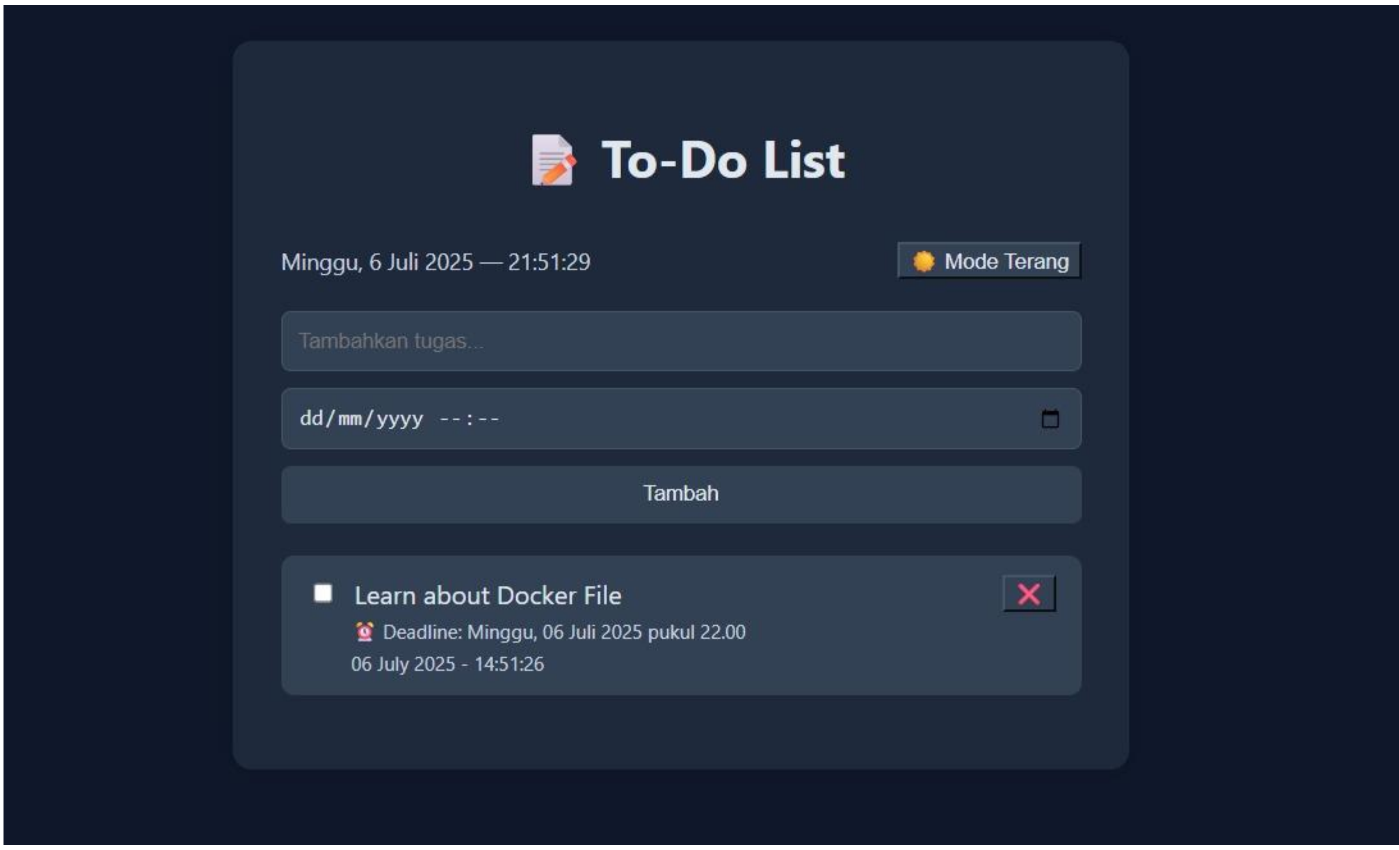
📝 To-Do List

Minggu, 6 Juli 2025 — 21:55:42

Learn about Docker File

06/07/2025 22:00

- result after adding

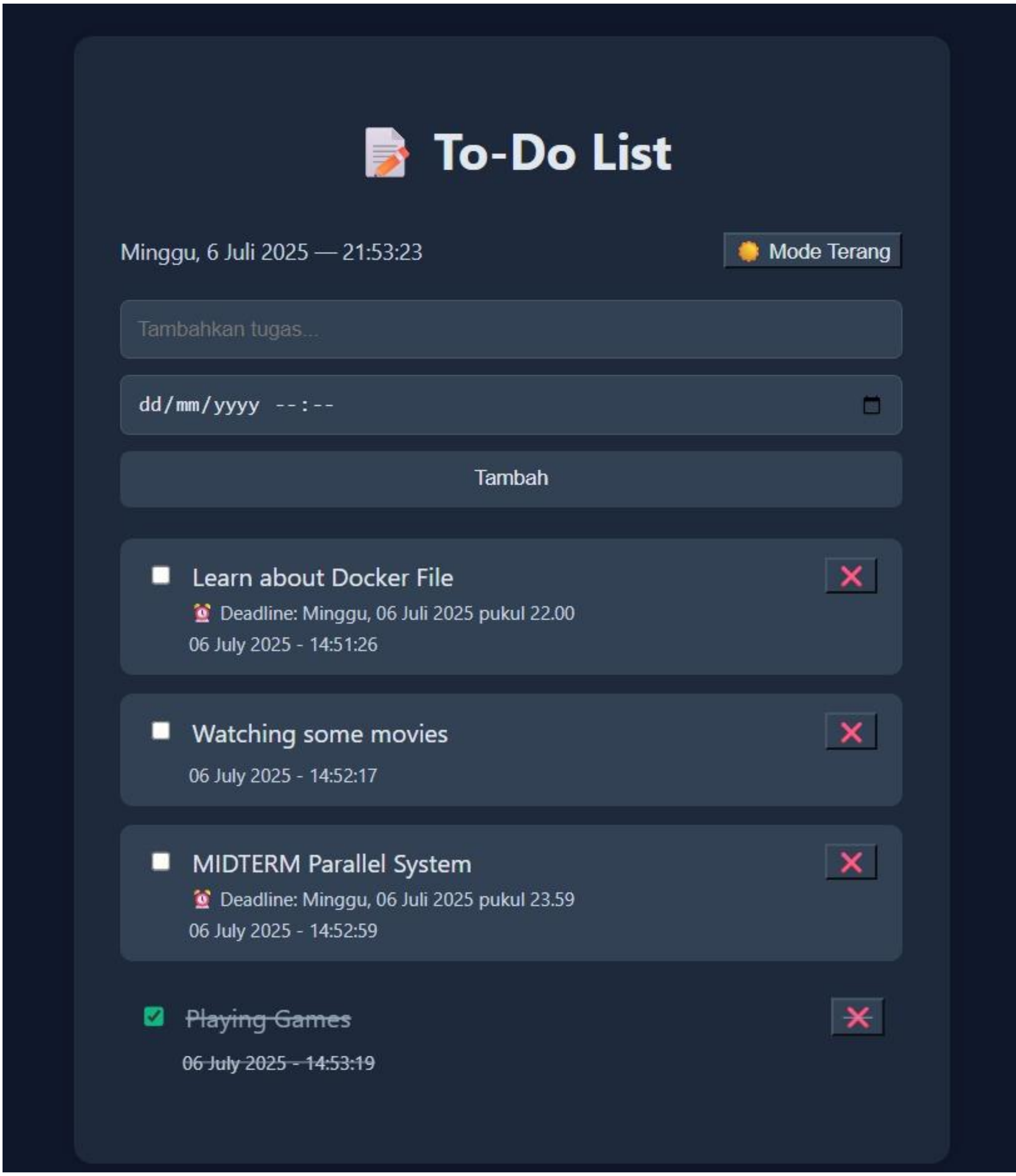


📝 To-Do List

Minggu, 6 Juli 2025 — 21:51:29

☐ Learn about Docker File ❌
⏰ Deadline: Minggu, 06 Juli 2025 pukul 22.00
06 July 2025 - 14:51:26

- adding more to do lists / adding more to do lists



📝 To-Do List

Minggu, 6 Juli 2025 — 21:53:23

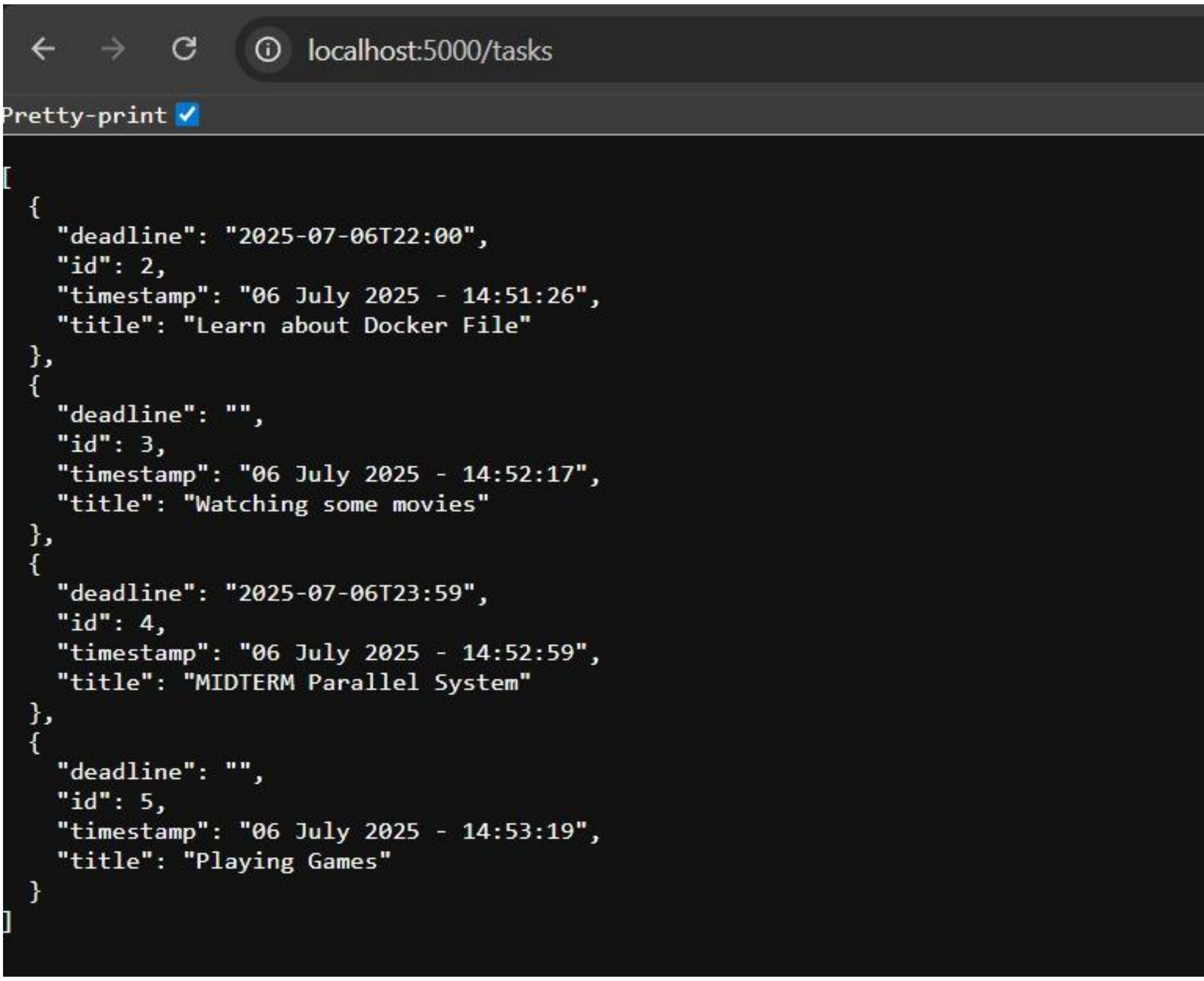
Learn about Docker File
⏰ Deadline: Minggu, 06 Juli 2025 pukul 22.00
06 July 2025 - 14:51:26

Watching some movies
06 July 2025 - 14:52:17

MIDTERM Parallel System
⏰ Deadline: Minggu, 06 Juli 2025 pukul 23.59
06 July 2025 - 14:52:59

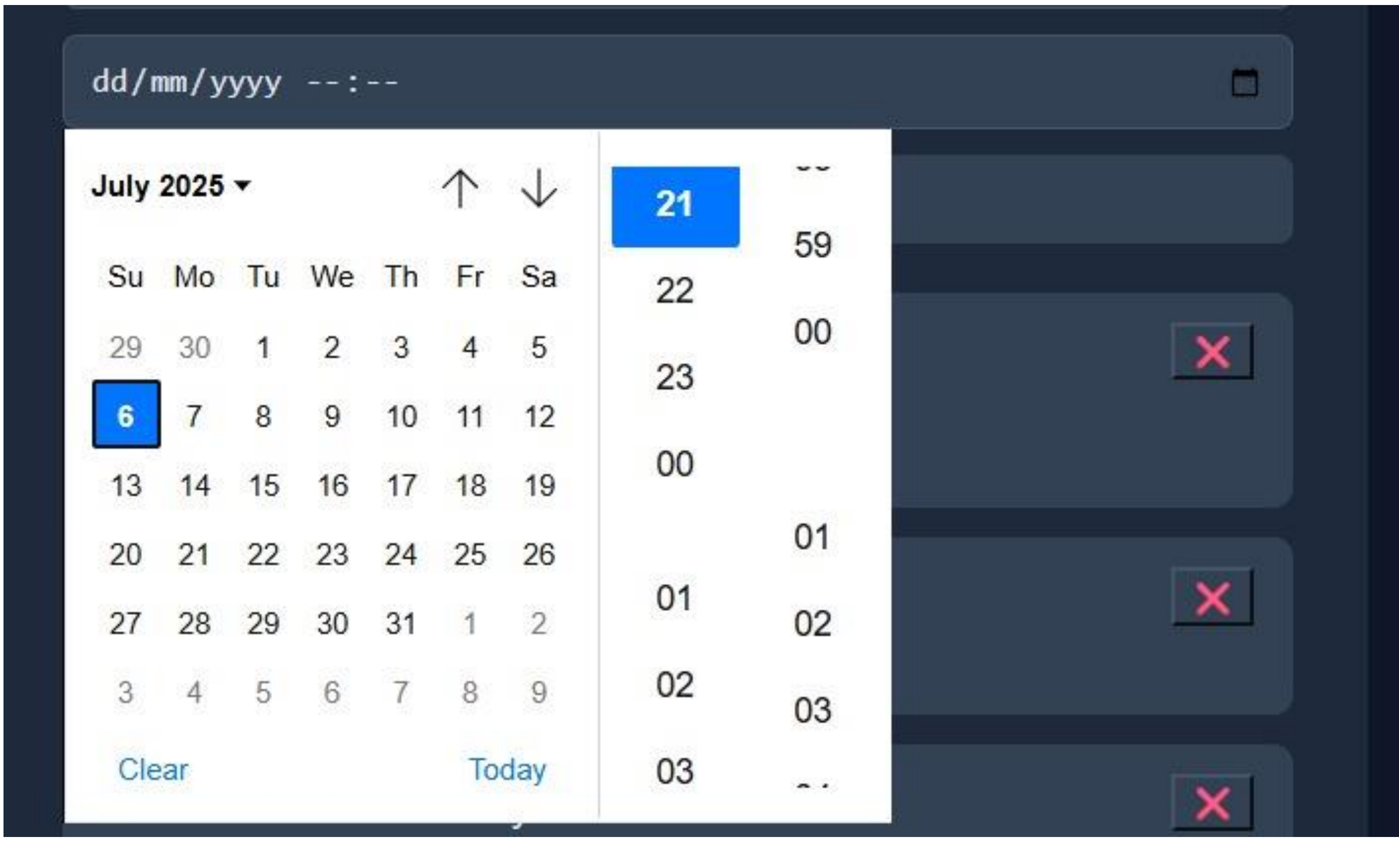
✓ ~~Playing Games~~
~~06 July 2025 - 14:53:19~~

- database using flask (python)

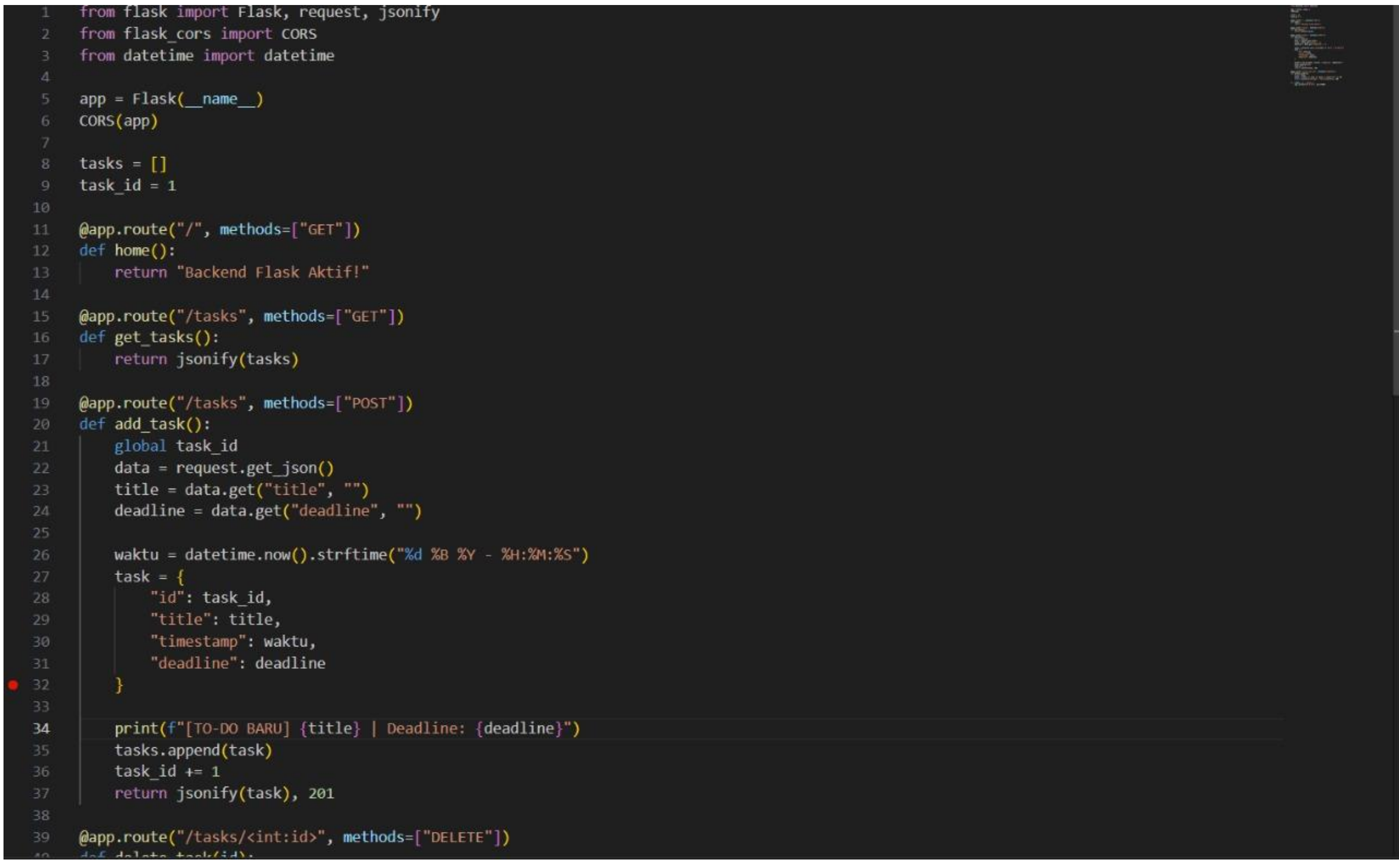


```
[
  {
    "deadline": "2025-07-06T22:00",
    "id": 2,
    "timestamp": "06 July 2025 - 14:51:26",
    "title": "Learn about Docker File"
  },
  {
    "deadline": "",
    "id": 3,
    "timestamp": "06 July 2025 - 14:52:17",
    "title": "Watching some movies"
  },
  {
    "deadline": "2025-07-06T23:59",
    "id": 4,
    "timestamp": "06 July 2025 - 14:52:59",
    "title": "MIDTERM Parallel System"
  },
  {
    "deadline": "",
    "id": 5,
    "timestamp": "06 July 2025 - 14:53:19",
    "title": "Playing Games"
  }
]
```

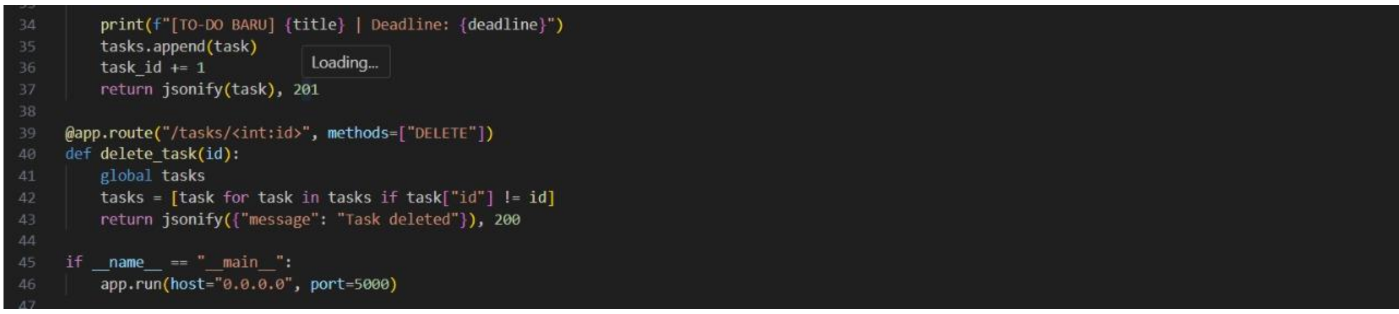
- add deadlines in real time using calendar and clock (if you don't want it, you can leave it blank)



- backend kode sumber menggunakan flask

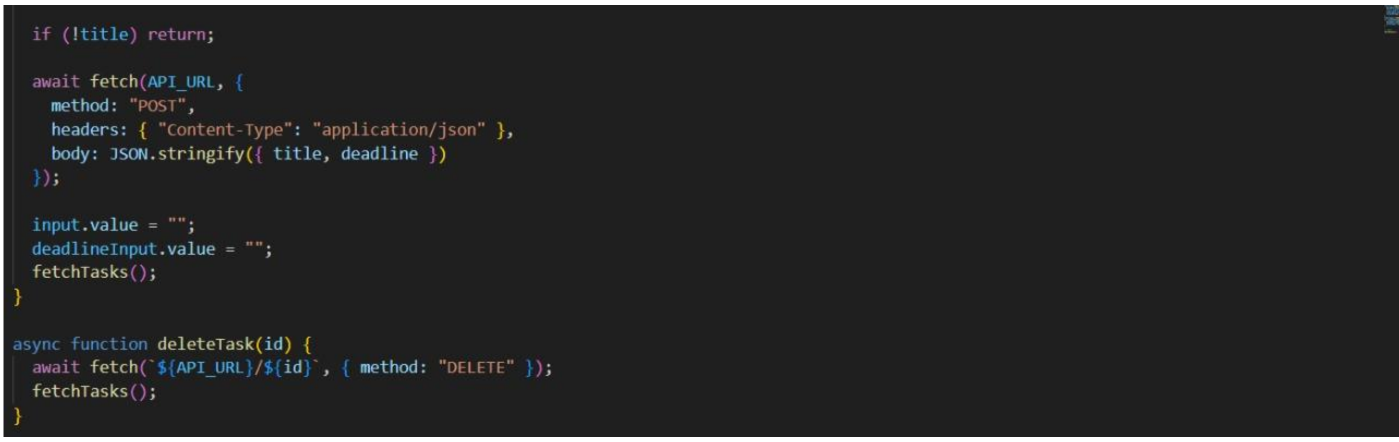


```python
from flask import Flask, request, jsonify
from flask_cors import CORS
from datetime import datetime

app = Flask(__name__)
CORS(app)

tasks = []
task_id = 1

@app.route("/", methods=["GET"])
def home():
    return "Backend Flask Aktif!"

@app.route("/tasks", methods=["GET"])
def get_tasks():
    return jsonify(tasks)

@app.route("/tasks", methods=["POST"])
def add_task():
    global task_id
    data = request.get_json()
    title = data.get("title", "")
    deadline = data.get("deadline", "")

    waktu = datetime.now().strftime("%d %B %Y - %H:%M:%S")
    task = {
        "id": task_id,
        "title": title,
        "timestamp": waktu,
        "deadline": deadline
    }

    print(f"[TO-DO BARU] {title} | Deadline: {deadline}")
    tasks.append(task)
    task_id += 1
    return jsonify(task), 201

@app.route("/tasks/<int:id>", methods=["DELETE"])
```



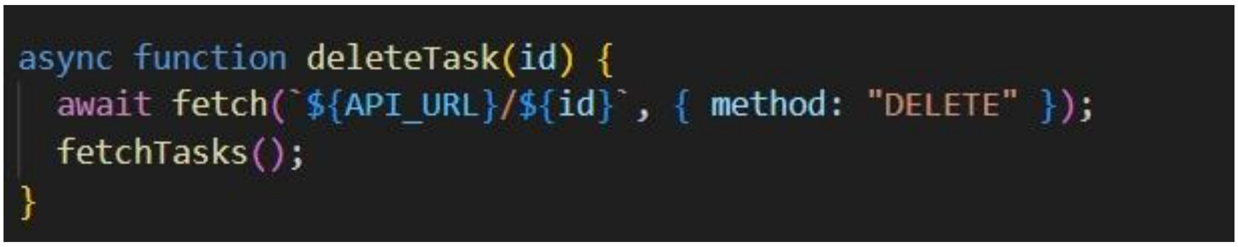
```
    print(f"[TO-DO BARU] {title} | Deadline: {deadline}")
    tasks.append(task)
    task_id += 1
    return jsonify(task), 201

@app.route("/tasks/<int:id>", methods=["DELETE"])
def delete_task(id):
    global tasks
    tasks = [task for task in tasks if task["id"] != id]
    return jsonify({"message": "Task deleted"}), 200

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

- POST /tasks → send new tasks to the backend



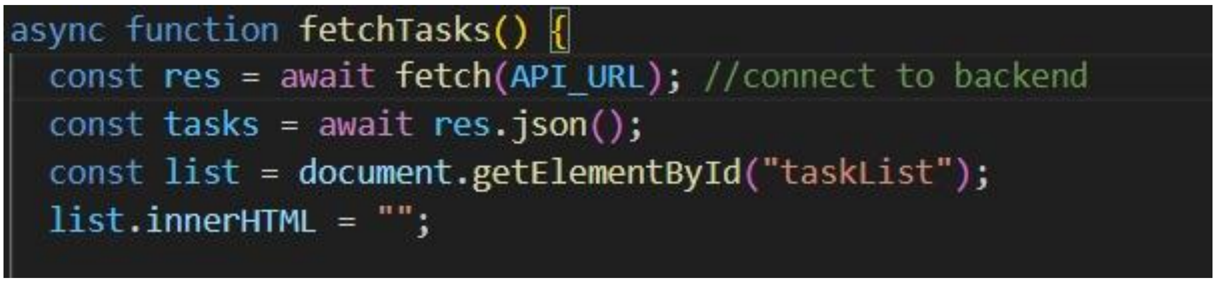
```
  if (!title) return;

  await fetch(API_URL, {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify({ title, deadline })
  });

  input.value = "";
  deadlineInput.value = "";
  fetchTasks();
}

async function deleteTask(id) {
  await fetch(`${API_URL}/${id}`, { method: "DELETE" });
  fetchTasks();
}
```

- DELETE /tasks/<id> → remove tasks from the backend



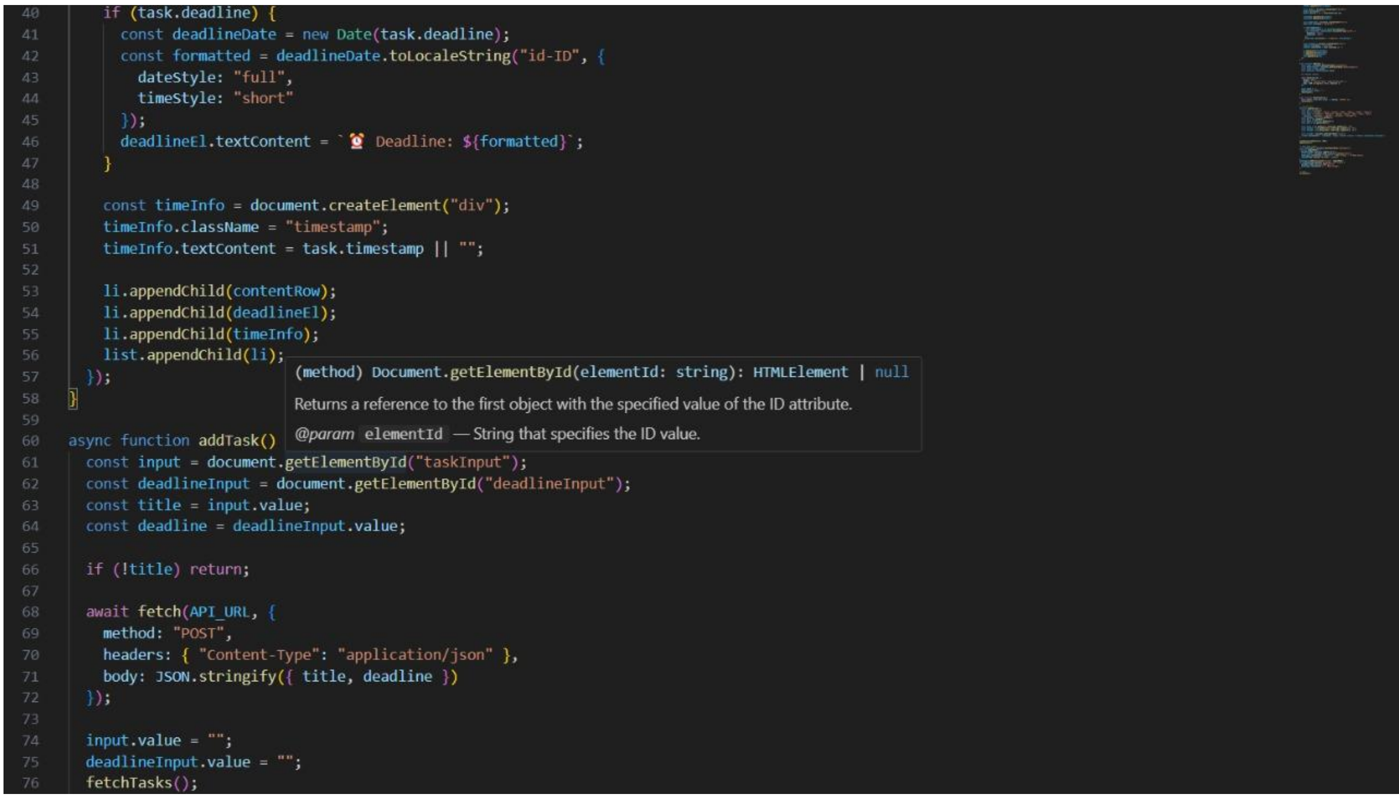
```
async function deleteTask(id) {
  await fetch(`${API_URL}/${id}`, { method: "DELETE" });
  fetchTasks();
}
```

- connects to the backend



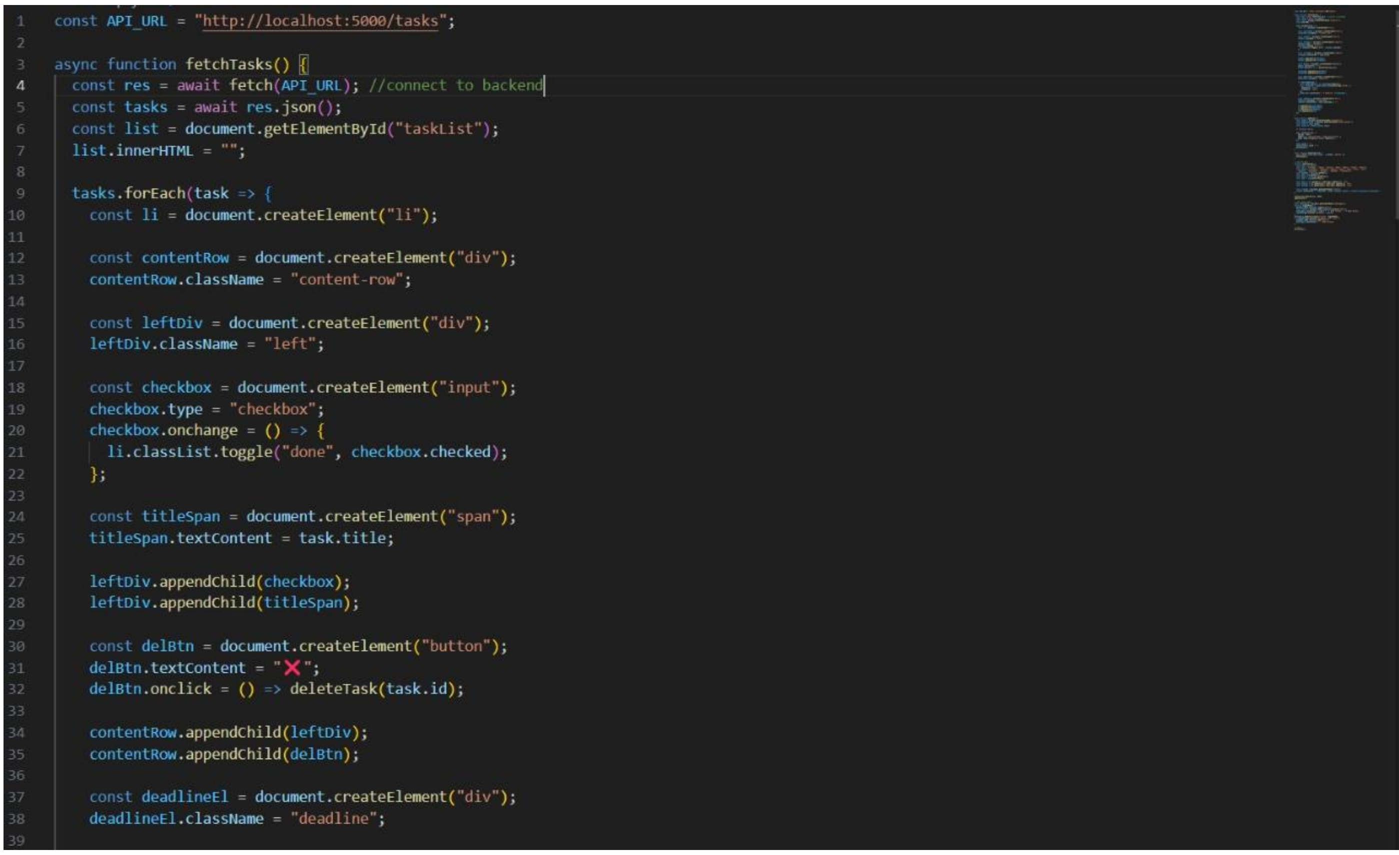
```
async function fetchTasks() {
  const res = await fetch(API_URL); //connect to backend
  const tasks = await res.json();
  const list = document.getElementById("taskList");
  list.innerHTML = "";
```

- complete script.js source code

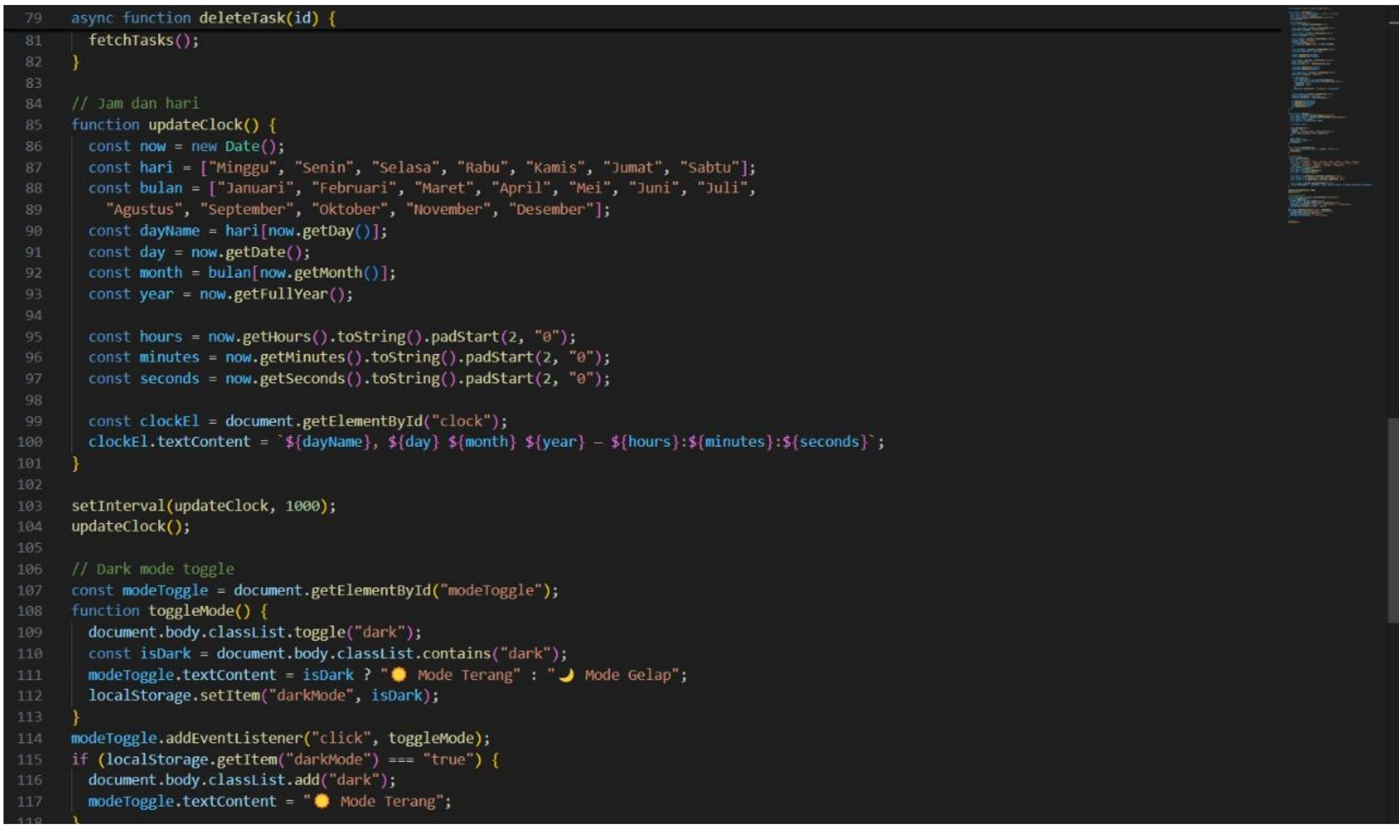


```javascript
    if (task.deadline) {
      const deadlineDate = new Date(task.deadline);
      const formatted = deadlineDate.toLocaleString("id-ID", {
        dateStyle: "full",
        timeStyle: "short"
      });
      deadlineEl.textContent = `⏰ Deadline: ${formatted}`;
    }

    const timeInfo = document.createElement("div");
    timeInfo.className = "timestamp";
    timeInfo.textContent = task.timestamp || "";

    li.appendChild(contentRow);
    li.appendChild(deadlineEl);
    li.appendChild(timeInfo);
    list.appendChild(li);
  });
}

async function addTask() {
  const input = document.getElementById("taskInput");
  const deadlineInput = document.getElementById("deadlineInput");
  const title = input.value;
  const deadline = deadlineInput.value;

  if (!title) return;

  await fetch(API_URL, {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify({ title, deadline })
  });

  input.value = "";
  deadlineInput.value = "";
  fetchTasks();
```

```
(method) Document.getElementById(elementId: string): HTMLElement | null
Returns a reference to the first object with the specified value of the ID attribute.
@param elementId — String that specifies the ID value.
```

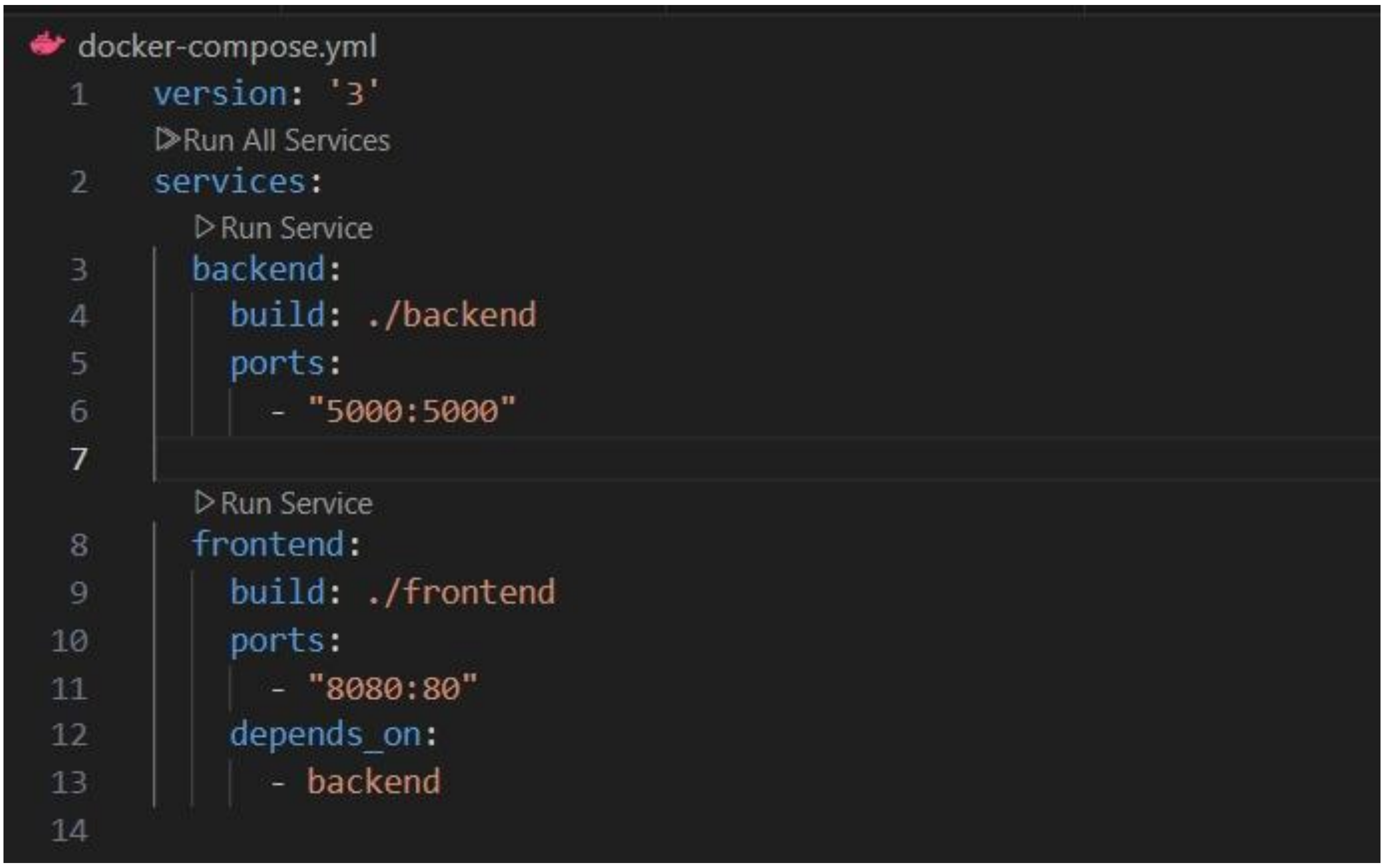


```javascript
const API_URL = "http://localhost:5000/tasks";

async function fetchTasks() {
  const res = await fetch(API_URL); //connect to backend
  const tasks = await res.json();
  const list = document.getElementById("taskList");
  list.innerHTML = "";

  tasks.forEach(task => {
    const li = document.createElement("li");

    const contentRow = document.createElement("div");
    contentRow.className = "content-row";

    const leftDiv = document.createElement("div");
    leftDiv.className = "left";

    const checkbox = document.createElement("input");
    checkbox.type = "checkbox";
    checkbox.onchange = () => {
      li.classList.toggle("done", checkbox.checked);
    };

    const titleSpan = document.createElement("span");
    titleSpan.textContent = task.title;

    leftDiv.appendChild(checkbox);
    leftDiv.appendChild(titleSpan);

    const delBtn = document.createElement("button");
    delBtn.textContent = "❌";
    delBtn.onclick = () => deleteTask(task.id);

    contentRow.appendChild(leftDiv);
    contentRow.appendChild(delBtn);

    const deadlineEl = document.createElement("div");
    deadlineEl.className = "deadline";

```



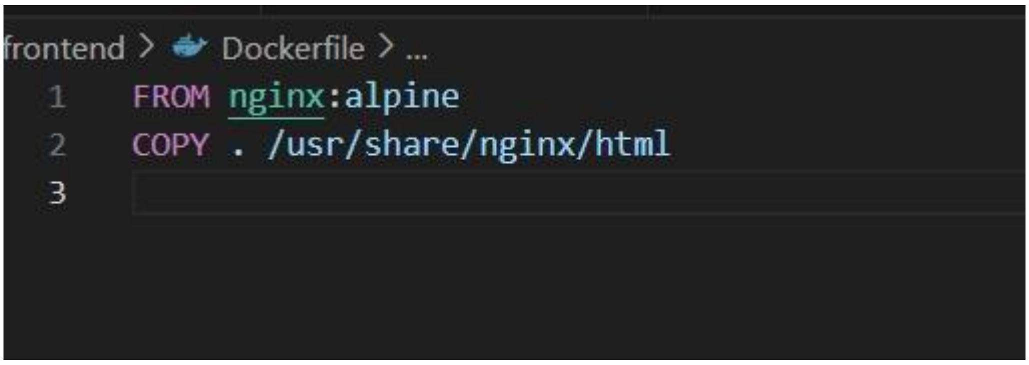
```javascript
async function deleteTask(id) {
  fetchTasks();
}

// Jam dan hari
function updateClock() {
  const now = new Date();
  const hari = ["Minggu", "Senin", "Selasa", "Rabu", "Kamis", "Jumat", "Sabtu"];
  const bulan = ["Januari", "Februari", "Maret", "April", "Mei", "Juni", "Juli",
    "Agustus", "September", "Oktober", "November", "Desember"];
  const dayName = hari[now.getDay()];
  const day = now.getDate();
  const month = bulan[now.getMonth()];
  const year = now.getFullYear();

  const hours = now.getHours().toString().padStart(2, "0");
  const minutes = now.getMinutes().toString().padStart(2, "0");
  const seconds = now.getSeconds().toString().padStart(2, "0");

  const clockEl = document.getElementById("clock");
  clockEl.textContent = `${dayName}, ${day} ${month} ${year} – ${hours}:${minutes}:${seconds}`;
}

setInterval(updateClock, 1000);
updateClock();

// Dark mode toggle
const modeToggle = document.getElementById("modeToggle");
function toggleMode() {
  document.body.classList.toggle("dark");
  const isDark = document.body.classList.contains("dark");
  modeToggle.textContent = isDark ? "🌞 Mode Terang" : "🌙 Mode Gelap";
  localStorage.setItem("darkMode", isDark);
}
modeToggle.addEventListener("click", toggleMode);
if (localStorage.getItem("darkMode") === "true") {
  document.body.classList.add("dark");
  modeToggle.textContent = "🌞 Mode Terang";
```

- Docker-compose.yml



```yaml
version: '3'
services:
  backend:
    build: ./backend
    ports:
      - "5000:5000"

  frontend:
    build: ./frontend
    ports:
      - "8080:80"
    depends_on:
      - backend

```

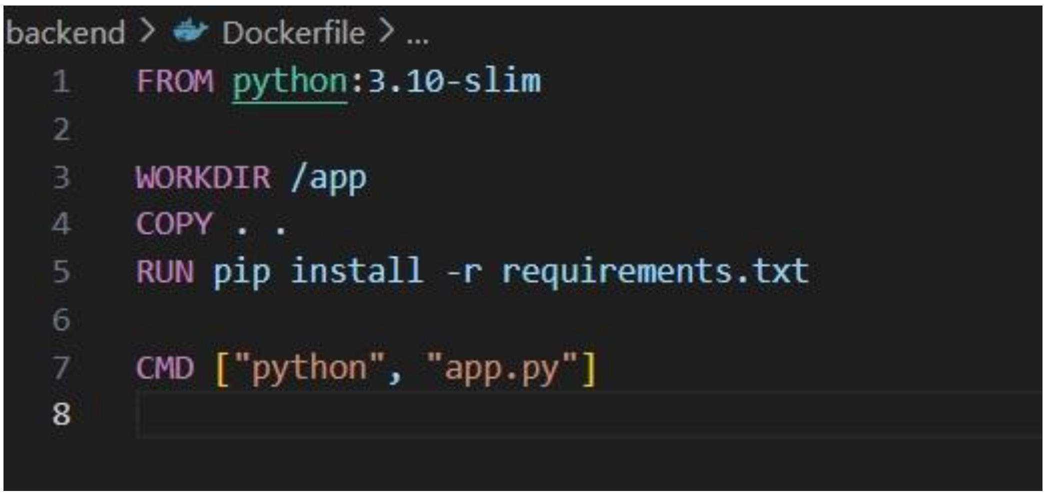
- Dockerfile on the frontend



frontend > Dockerfile > ...

```
FROM nginx:alpine
COPY . /usr/share/nginx/html
```

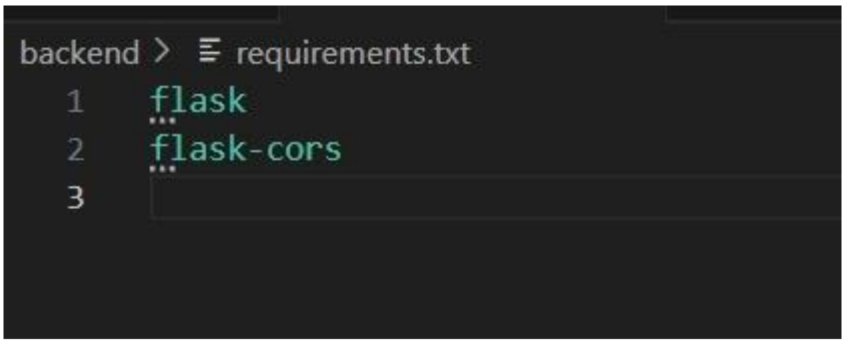
- Dockerfile on the backend



backend > Dockerfile > ...

```
FROM python:3.10-slim

WORKDIR /app
COPY . .
RUN pip install -r requirements.txt

CMD ["python", "app.py"]
```

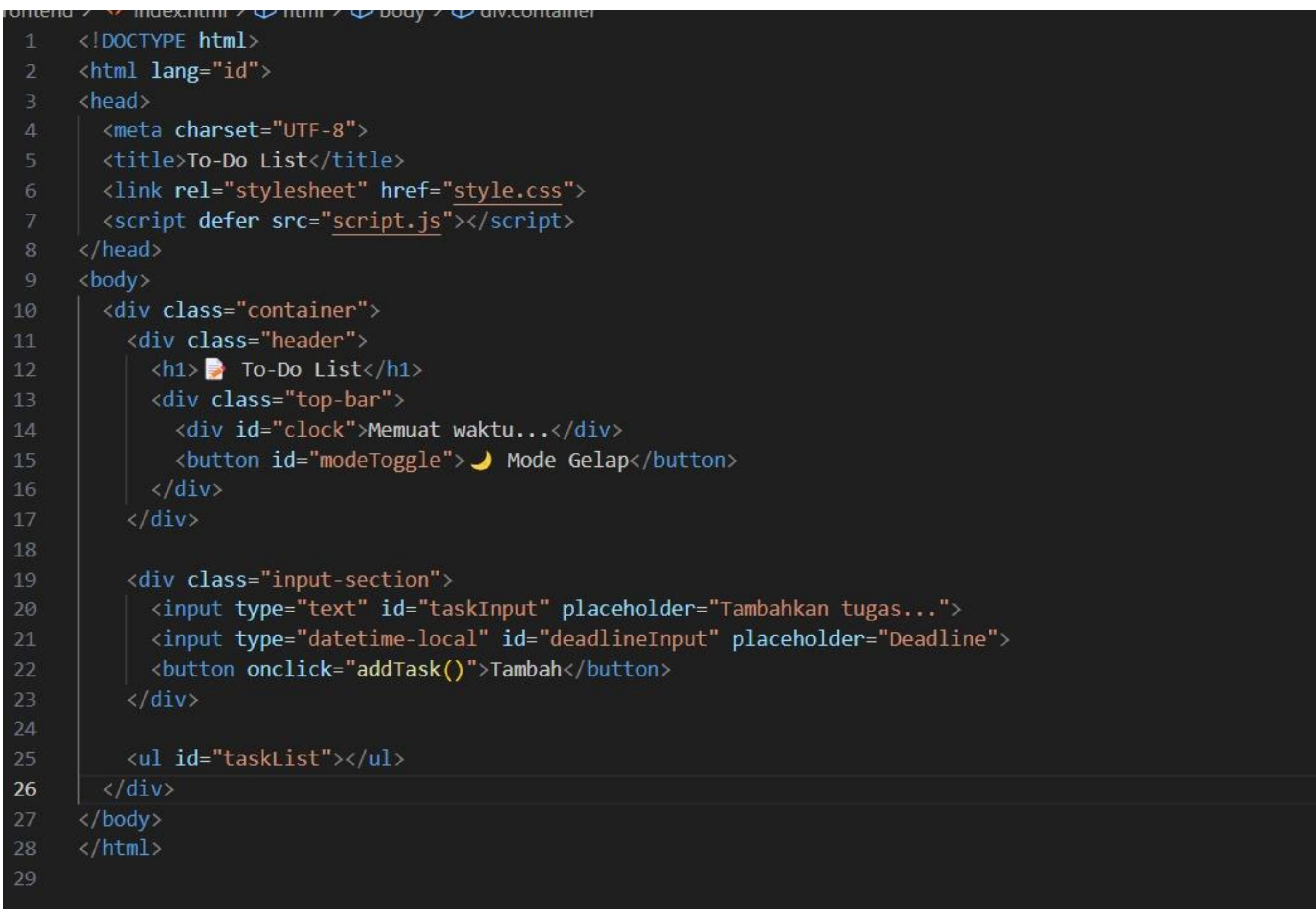
- requirements.txt



backend > requirements.txt

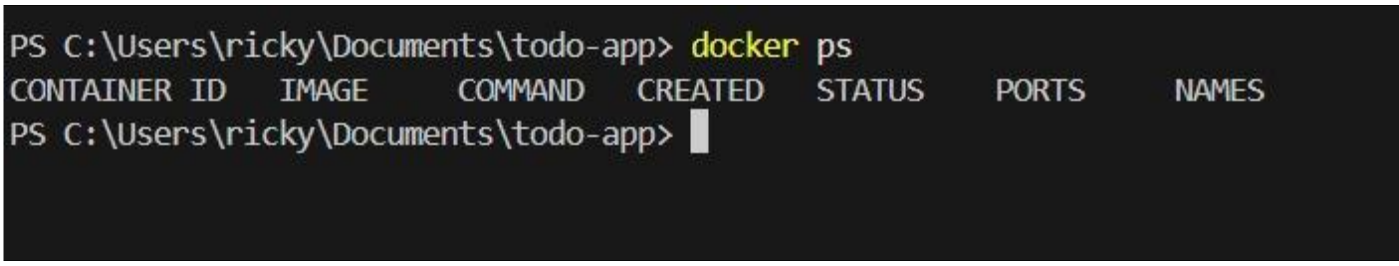
```
flask
flask-cors
```

- full source code index.html



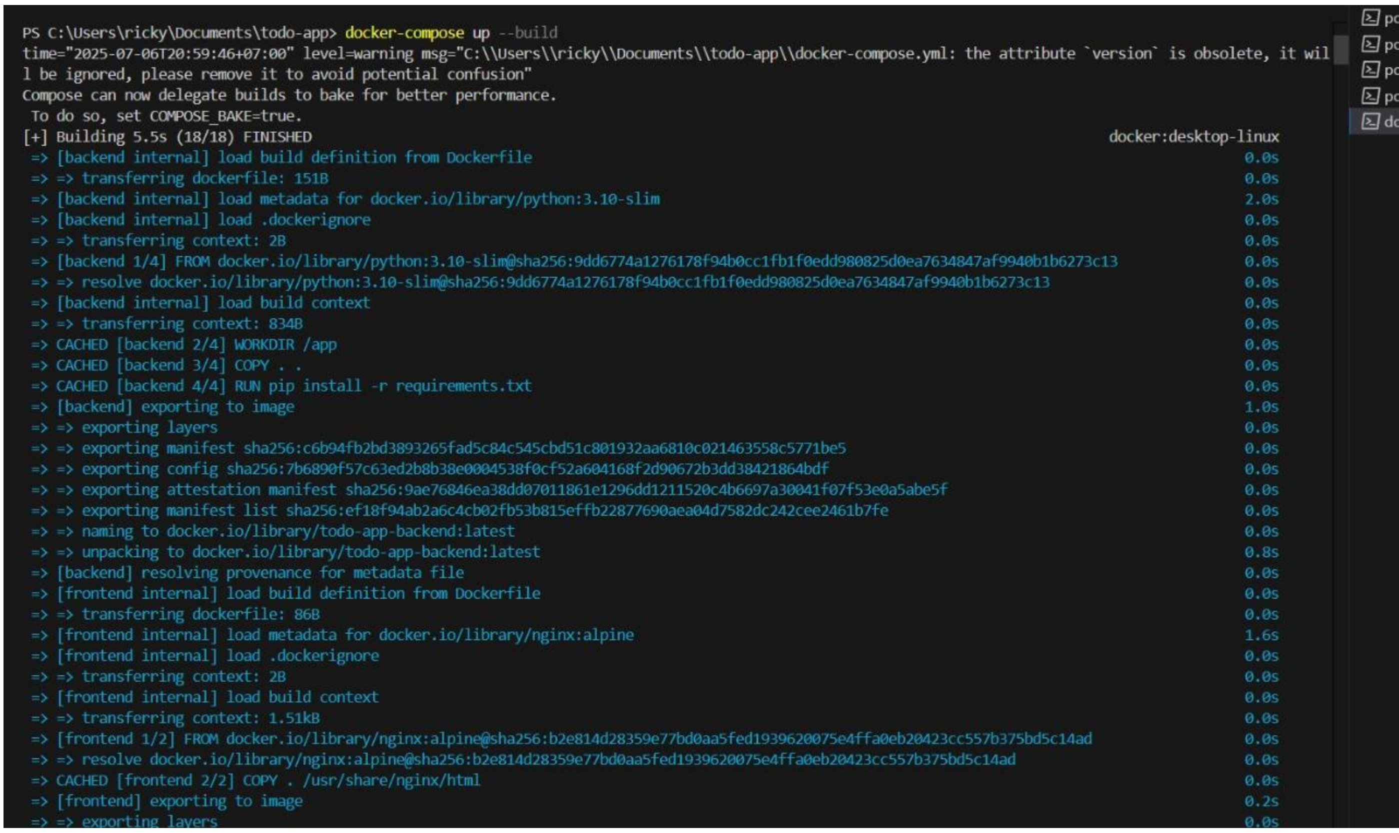
```html
<!DOCTYPE html>
<html lang="id">
<head>
  <meta charset="UTF-8">
  <title>To-Do List</title>
  <link rel="stylesheet" href="style.css">
  <script defer src="script.js"></script>
</head>
<body>
  <div class="container">
    <div class="header">
      <h1>📝 To-Do List</h1>
      <div class="top-bar">
        <div id="clock">Memuat waktu...</div>
        <button id="modeToggle">🌙 Mode Gelap</button>
      </div>
    </div>

    <div class="input-section">
      <input type="text" id="taskInput" placeholder="Tambahkan tugas...">
      <input type="datetime-local" id="deadlineInput" placeholder="Deadline">
      <button onclick="addTask()">Tambah</button>
    </div>

    <ul id="taskList"></ul>
  </div>
</body>
</html>

```

- docker ps



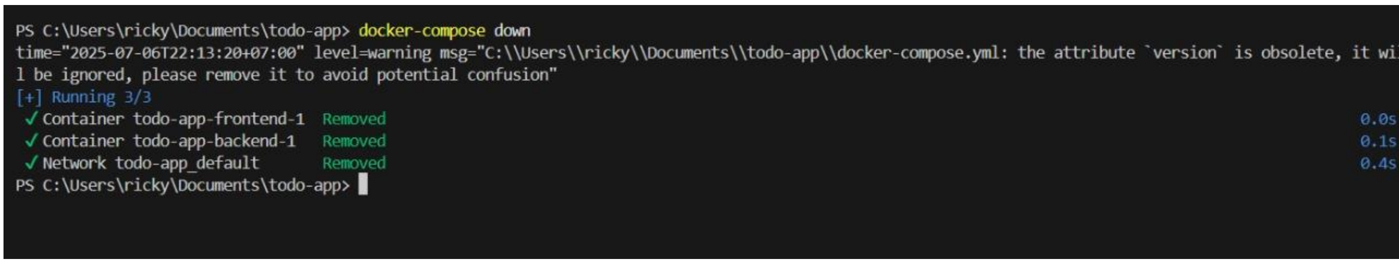
```
PS C:\Users\ricky\Documents\todo-app> docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
PS C:\Users\ricky\Documents\todo-app> 
```

- docker-compose up

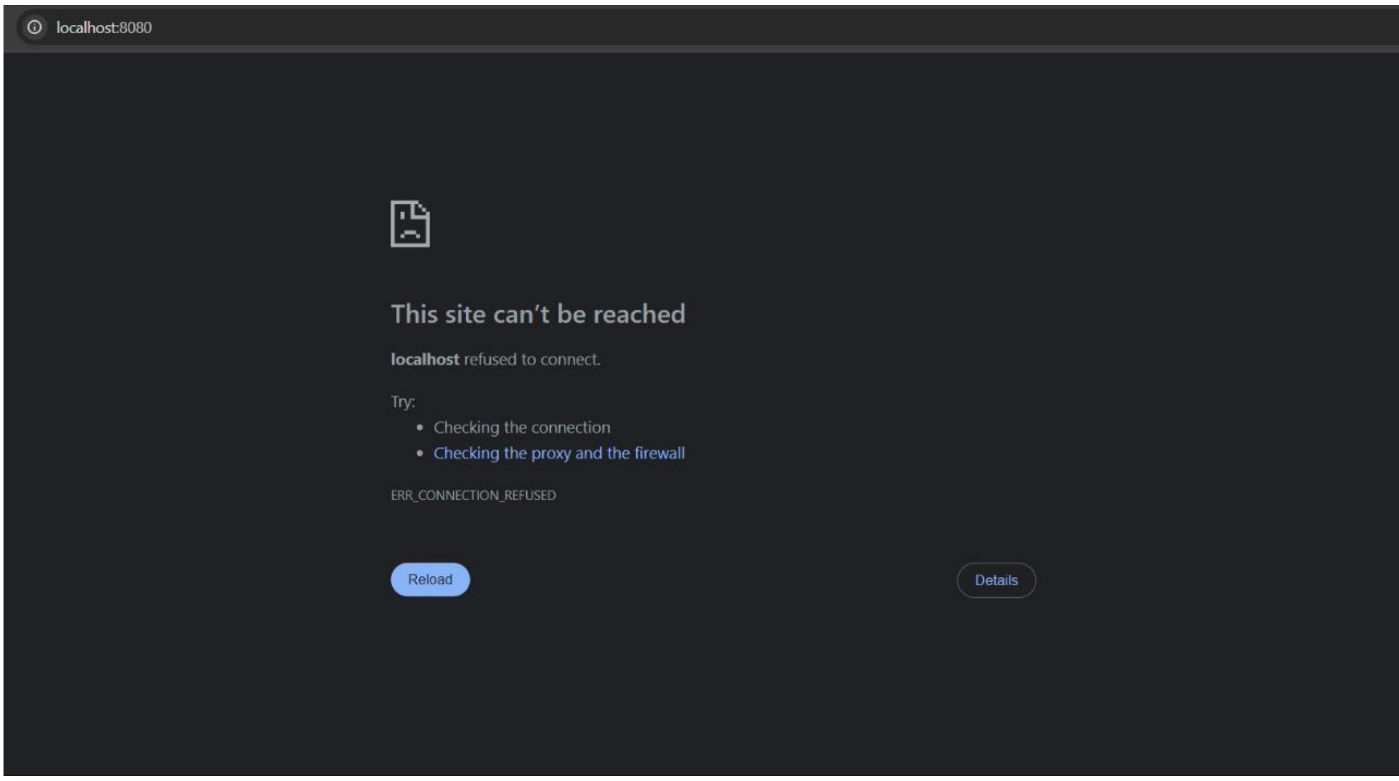


```
PS C:\Users\ricky\Documents\todo-app> docker-compose up --build
time="2025-07-06T20:59:46+07:00" level=warning msg="C:\\Users\\ricky\\Documents\\todo-app\\docker-compose.yml: the attribute `version` is obsolete, it wil
l be ignored, please remove it to avoid potential confusion"
Compose can now delegate builds to bake for better performance.
 To do so, set COMPOSE_BAKE=true.
[+] Building 5.5s (18/18) FINISHED                                                                                                docker:desktop-linux
 => [backend internal] load build definition from Dockerfile                                                                                     0.0s
 => => transferring dockerfile: 151B                                                                                                             0.0s
 => [backend internal] load metadata for docker.io/library/python:3.10-slim                                                                      2.0s
 => [backend internal] load .dockerignore                                                                                                        0.0s
 => => transferring context: 2B                                                                                                                  0.0s
 => [backend 1/4] FROM docker.io/library/python:3.10-slim@sha256:9dd6774a1276178f94b0cc1fb1f0edd980825d0ea7634847af9940b1b6273c13               0.0s
 => => resolve docker.io/library/python:3.10-slim@sha256:9dd6774a1276178f94b0cc1fb1f0edd980825d0ea7634847af9940b1b6273c13                        0.0s
 => [backend internal] load build context                                                                                                        0.0s
 => => transferring context: 834B                                                                                                                0.0s
 => CACHED [backend 2/4] WORKDIR /app                                                                                                            0.0s
 => CACHED [backend 3/4] COPY . .                                                                                                                0.0s
 => CACHED [backend 4/4] RUN pip install -r requirements.txt                                                                                     0.0s
 => [backend] exporting to image                                                                                                                 1.0s
 => => exporting layers                                                                                                                          0.0s
 => => exporting manifest sha256:c6b94fb2bd3893265fad5c84c545cbd51c801932aa6810c021463558c5771be5                                                0.0s
 => => exporting config sha256:7b6890f57c63ed2b8b38e0004538f0cf52a604168f2d90672b3dd38421864bdf                                                  0.0s
 => => exporting attestation manifest sha256:9ae76846ea38dd07011861e1296dd1211520c4b6697a30041f07f53e0a5abe5f                                    0.0s
 => => exporting manifest list sha256:ef18f94ab2a6c4cb02fb53b815effb22877690aea04d7582dc242cee2461b7fe                                           0.0s
 => => naming to docker.io/library/todo-app-backend:latest                                                                                       0.0s
 => => unpacking to docker.io/library/todo-app-backend:latest                                                                                    0.8s
 => [backend] resolving provenance for metadata file                                                                                             0.0s
 => [frontend internal] load build definition from Dockerfile                                                                                    0.0s
 => => transferring dockerfile: 86B                                                                                                              0.0s
 => [frontend internal] load metadata for docker.io/library/nginx:alpine                                                                         1.6s
 => [frontend internal] load .dockerignore                                                                                                       0.0s
 => => transferring context: 2B                                                                                                                  0.0s
 => [frontend internal] load build context                                                                                                       0.0s
 => => transferring context: 1.51kB                                                                                                              0.0s
 => [frontend 1/2] FROM docker.io/library/nginx:alpine@sha256:b2e814d28359e77bd0aa5fed1939620075e4ffa0eb20423cc557b375bd5c14ad                   0.0s
 => => resolve docker.io/library/nginx:alpine@sha256:b2e814d28359e77bd0aa5fed1939620075e4ffa0eb20423cc557b375bd5c14ad                            0.0s
 => CACHED [frontend 2/2] COPY . /usr/share/nginx/html                                                                                           0.0s
 => [frontend] exporting to image                                                                                                                0.2s
 => => exporting layers                                                                                                                          0.0s
```

- docker-compose down



```
PS C:\Users\ricky\Documents\todo-app> docker-compose down
time="2025-07-06T22:13:20+07:00" level=warning msg="C:\\Users\\ricky\\Documents\\todo-app\\docker-compose.yml: the attribute `version` is obsolete, it wi
l be ignored, please remove it to avoid potential confusion"
[+] Running 3/3
 ✓ Container todo-app-frontend-1  Removed                                                                                                           0.0s
 ✓ Container todo-app-backend-1   Removed                                                                                                           0.1s
 ✓ Network todo-app_default       Removed                                                                                                           0.4s
PS C:\Users\ricky\Documents\todo-app> █
```



localhost:8080

This site can't be reached

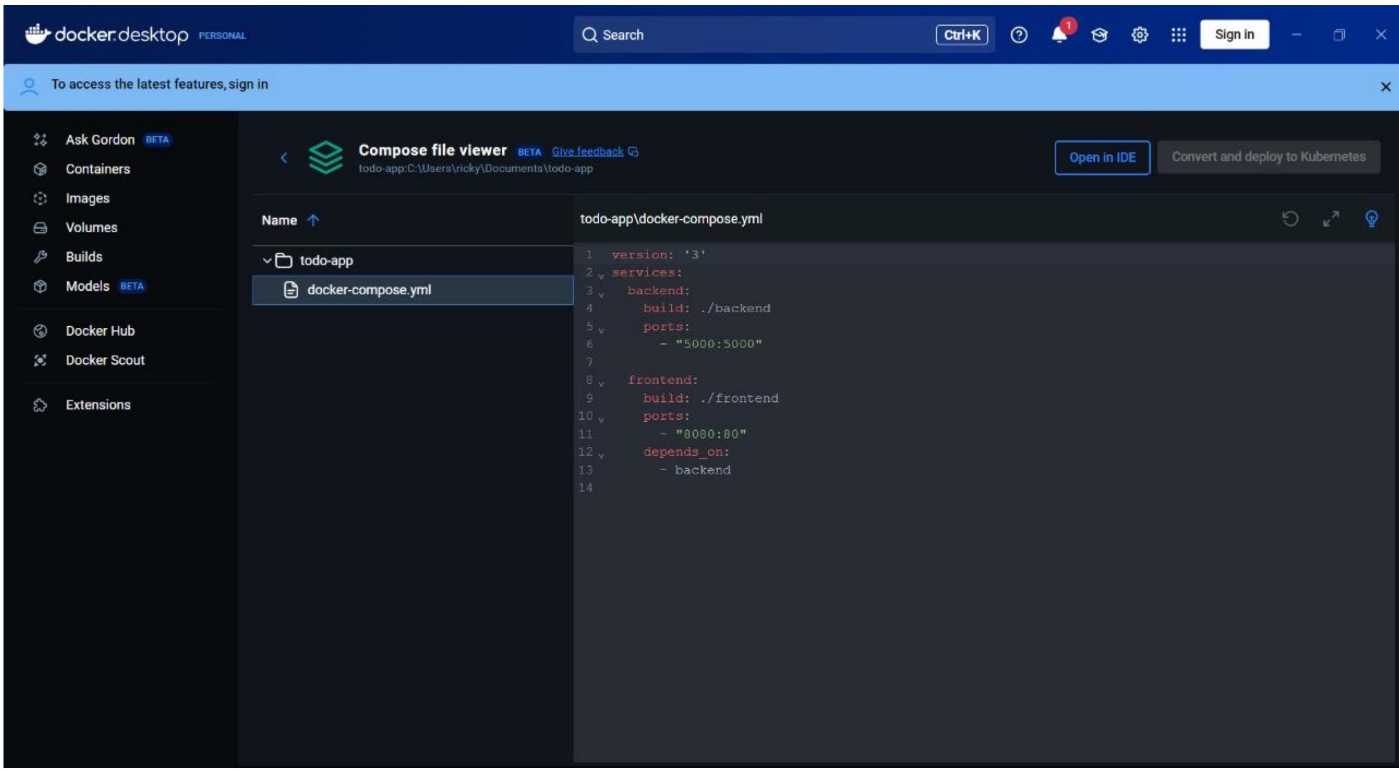
localhost refused to connect.

Try:

- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_REFUSED

- docker desktop


docker.desktop PERSONAL
Search
Ctrl+K
Sign in
To access the latest features, sign in
Ask Gordon BETA
Containers
Images
Volumes
Builds
Models BETA
Docker Hub
Docker Scout
Extensions
Compose file viewer BETA Give feedback
todo-app:C:\Users\ricky\Documents\todo-app
Open in IDE
Convert and deploy to Kubernetes
Name
todo-app
docker-compose.yml
todo-app\docker-compose.yml

```yaml
version: '3'
services:
  backend:
    build: ./backend
    ports:
      - "5000:5000"

  frontend:
    build: ./frontend
    ports:
      - "8080:80"
    depends_on:
      - backend
```

<u>Step-by-step Process of Deploying the Application Using Docker</u>

The To-Do List application deployment process uses Docker and Docker Compose to run two main services, namely the frontend (containing HTML, CSS, JavaScript) and backend (using Python Flask). The folder structure in this project consists of two main directories, namely frontend and backend, and one main file called docker-compose.yml. Each directory has its own Dockerfile that is used to build its image. The Dockerfile for the backend uses the base image python:3.10-slim, installs Flask and flask-cors, and executes app.py on port 5000. Meanwhile, the Dockerfile for the frontend uses the nginx:alpine image and copies all HTML, CSS, and JavaScript files to the nginx default directory.

The docker-compose.yml file serves to organize both services in one configuration file. In this file, the backend will be built from the backend directory and accessed via port 5000, while the frontend is built from the frontend directory and accessed via port 8080. The depends_on property is used to ensure that the frontend will only run once the backend is available.

To run this project, the main command used is docker-compose up --build. This command will build all images from their respective Dockerfiles and immediately run both containers. If there are no changes in the Dockerfile and just want to re-run the application, just use docker-compose up. The application can then be accessed via a browser, where the frontend is available at http://localhost:8080 and the backend can be checked via http://localhost:5000 or the endpoint http://localhost:5000/tasks.

Users can also monitor application processes and logs through the docker-compose logs backend or docker-compose logs frontend commands. To stop the entire container and remove the used network, the docker-compose down command is used. If needed, unused Docker images can also be removed by running docker rmi after viewing the image list with docker images.

Overall, the use of Docker and Docker Compose makes the application deployment process more structured, fast, and portable. Each service runs separately but is interconnected through Docker's internal network, so it does not require complex manual setup. This approach is very suitable for the development of small-medium scale projects such as this To-Do List application, because it provides consistency in the work environment and ease of service management.

GitHub Link : https://github.com/DeanAdwitiyap/To_Do_Lists