Todo Summary Assistant - Full Stack Internship Assessment

Objective

Build a full-stack application using React (frontend) and Node.js with Express (backend) that allows users to manage personal to-do items, generate summaries of completed tasks using OpenAI's API, and send the summary to a Slack channel. The system should be deployable and support environment configuration.

Deliverables

1. Source Code

- Frontend: React code to enter tasks, view list, check as done, and send to backend
- Backend: Express server to process completed tasks, send to OpenAI, post to Slack

2. Frontend Code (React)

App.js

```
import React, { useState } from 'react';
import axios from 'axios';
import './App.css';

function App() {
  const [task, setTask] = useState('');
  const [tasks, setTasks] = useState([]);
  const [summary, setSummary] = useState('');

  const handleAddTask = () => {
    if (task.trim()) {
      setTasks([...tasks, { text: task.trim(), done: false }]);
      setTask('');
    }
  };

  const toggleTask = (index) => {
    const updatedTasks = [...tasks];
    updatedTasks[index].done = !updatedTasks[index].done;
    setTasks(updatedTasks);
  };

  const handleSummarize = async () => {
    const completedTasks = tasks.filter(t => t.done).map(t => t.text);
    if (completedTasks.length === 0) {
      alert('Please complete at least one task.');
      return;
    }
```

```
    try {
      const response = await axios.post('http://localhost:5000/summarize', {
        completedTasks
      });
      setSummary(response.data.summary);
    } catch (error) {
      alert('Error generating summary.');
    }
  };

  return (
    <div className="App">
      <h1>Todo Summary Assistant</h1>
      <div className="input-section">
        <input
          type="text"
          placeholder="Enter your task"
          value={task}
          onChange={(e) => setTask(e.target.value)}
        />
        <button onClick={handleAddTask}>Add</button>
      </div>
      <ul className="task-list">
        {tasks.map((taskItem, index) => (
          <li key={index}>
            <input
              type="checkbox"
              checked={taskItem.done}
              onChange={() => toggleTask(index)}
            />
            <span className={taskItem.done ? 'completed' : ''}>
              {taskItem.text}
            </span>
          </li>
        ))}
      </ul>
      <button className="summarize-btn" onClick={handleSummarize}>
        Summarize Completed Tasks
      </button>
      {summary && (
        <div className="summary-box">
          <h3>Summary</h3>
          <p>{summary}</p>
        </div>
      )}
    </div>
  );
}

export default App;
```

3. Backend Code (Node.js + Express)

index.js

```
const express = require('express');
const cors = require('cors');
const bodyParser = require('body-parser');
const axios = require('axios');
const { Configuration, OpenAIApi } = require('openai');
require('dotenv').config();

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

const openai = new OpenAIApi(new Configuration({
  apiKey: process.env.OPENAI_API_KEY,
}));

app.post('/summarize', async (req, res) => {
  const { completedTasks } = req.body;

  if (!completedTasks || completedTasks.length === 0) {
    return res.status(400).json({ error: 'No completed tasks provided.' });
  }

  const prompt = `Summarize the following completed todo tasks:\n${completedTasks.join('\n')}`;

  try {
    const aiResponse = await openai.createChatCompletion({
      model: "gpt-3.5-turbo",
      messages: [{ role: "user", content: prompt }],
    });

    const summary = aiResponse.data.choices[0].message.content;

    await axios.post(process.env.SLACK_WEBHOOK_URL, {
      text: `Todo Summary:\n${summary}`
    });

    res.json({ summary });
  } catch (error) {
    console.error('Error:', error.message);
    res.status(500).json({ error: 'Summary generation failed' });
  }
});

const PORT = process.env.PORT || 5000;
app.listen(PORT, () => {
  console.log(`Server is running on port ${PORT}`);
});
```

4. .env Configuration

```
OPENAI_API_KEY=your_openai_key_here
SLACK_WEBHOOK_URL=your_slack_webhook_url
```

```
PORT=5000
```