CODING for E-MAIL REMAINDER SYSTEM

BACKEND (SERVER.js)

```
// server.js
app.get('/api/reminders', (req, res) => {
db.all(`SELECT * FROM reminders ORDER BY id DESC LIMIT 100`, [], (err, rows)
=> {
if (err) return res.status(500).json({ error: err.message });
res.json(rows);
});
});

// API: delete reminder
app.delete('/api/reminders/:id', (req, res) => {
const id = req.params.id;
db.run(`DELETE FROM reminders WHERE id = ?`, [id], function (err) {
if (err) return res.status(500).json({ error: err.message });
res.json({ deleted: this.changes });
});
});

// Function to send due reminders
function sendDueReminders() {
const nowIso = new Date().toISOString();
db.all(`SELECT * FROM reminders WHERE sent = 0 AND send_at <= ?`, [nowIso],
(err, rows) => {
if (err) return console.error('DB fetch error:', err);
if (!rows || rows.length === 0) return; // nothing to send

rows.forEach(row => {
const mail = {
from: process.env.FROM_EMAIL || 'naa-noreply@example.com',
to: row.email,
subject: row.subject + ' — Naan Muudhalvan Reminder',
text: row.message + '\n\n(Generated by Naan Muudhalvan reminder system)'
};

transporter.sendMail(mail, (err, info) => {
if (err) {
console.error('Send mail error for id', row.id, err);
} else {
console.log('Email sent for id', row.id, '=>', info && info.response ? info.response : info);
db.run(`UPDATE reminders SET sent = 1 WHERE id = ?`, [row.id]);
}});
});
});
}

// Schedule the job every minute (for demo). In production choose sensible schedule.
cron.schedule('* * * * *', () => {
console.log('Cron: checking for due reminders at', new Date().toISOString());
sendDueReminders();
});

const PORT = process.env.PORT || 3000;
app.listen(PORT, () => console.log(`Server listening on port ${PORT}`));
```

FRONT END (PUBLIC,INDEX.html)

```
<!doctype html>
<input id="email" type="email" required />
<label>Subject</label>
<input id="subject" required />
<label>Message</label>
<textarea id="message" rows="4" required></textarea>
<label>Send at (ISO Datetime, e.g. 2025-11-13T14:00:00Z)</label>
<input id="send_at" placeholder="YYYY-MM-DDTHH:MM:SSZ" required />
<button type="submit">Create reminder</button>
</form>

<h3>Existing reminders</h3>
<div id="list"></div>

<script>
const form = document.getElementById('remForm');
const list = document.getElementById('list');

async function fetchList(){
const r = await fetch('/api/reminders');
const data = await r.json();
list.innerHTML = data.map(rem => `
<div class="card">
<strong>#${rem.id}</strong> — <em>${rem.email}</em> — <small>send_at:
${rem.send_at}</small>
<p>${rem.message}</p>
<p>Sent: ${rem.sent ? 'Yes' : 'No'}</p>
<button onclick="del(${rem.id})">Delete</button>
</div>
`).join('') || '<p>No reminders</p>';
}

async function del(id){
await fetch('/api/reminders/' + id, { method: 'DELETE' });
fetchList();
}

form.onsubmit = async (e) => {
e.preventDefault();
const payload = {
email: document.getElementById('email').value,
subject: document.getElementById('subject').value,
message: document.getElementById('message').value,
send_at: document.getElementById('send_at').value
};
const r = await fetch('/api/reminders', {
method: 'POST', headers: {'Content-Type':'application/json'}, body:
JSON.stringify(payload)
});
if (r.ok) {
form.reset();
fetchList();
alert('Reminder created!');
} else {
const err = await r.json();
alert('Error: ' + (err.error || 'unknown'));
}
};

// initial load
fetchList();
</script>
</body>
</html>
```