APPENDIX A – SOURCE CODE

SQL QUERY

```
CREATE DATABASE IF NOT EXISTS restaurant_booking;
USE restaurant_booking;
CREATE TABLE IF NOT EXISTS users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(150) NOT NULL,
    email VARCHAR(150) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    phone VARCHAR(30),
    role ENUM('admin','customer') DEFAULT 'customer',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
CREATE TABLE IF NOT EXISTS bookings (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    booking_date DATE,
    booking_time TIME,
    guests INT DEFAULT 2,
    occasion VARCHAR(100),
    preference VARCHAR(100),
    theme VARCHAR(100),
    section VARCHAR(100),
    status ENUM('Pending','Confirmed','Cancelled','Completed','Paid') DEFAULT
'Pending',
    total_amount DECIMAL(10,2) DEFAULT 0.00,
    payment_status ENUM('Unpaid','Paid','Refunded') DEFAULT 'Unpaid',
    cancellation_fee DECIMAL(10,2) DEFAULT 0.00,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE SET NULL
);
```

```
CREATE TABLE IF NOT EXISTS feedback (
   id INT AUTO_INCREMENT PRIMARY KEY,
   booking_id INT,
   user_id INT,
   rating TINYINT,
   comment TEXT,
   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   FOREIGN KEY (booking_id) REFERENCES bookings(id) ON DELETE
CASCADE,
   FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE SET NULL
);
CREATE TABLE IF NOT EXISTS transactions (
   id INT AUTO_INCREMENT PRIMARY KEY,
   booking_id INT,
   amount DECIMAL(10,2),
   method VARCHAR(50),
   status ENUM('Success','Failed','Refunded') DEFAULT 'Success',
   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   FOREIGN KEY (booking_id) REFERENCES bookings(id) ON DELETE
CASCADE
);
INSERT IGNORE INTO users (name, email, password, phone, role)
VALUES ('Admin', 'admin@admin.com', 'admin123', '9999999999', 'admin');
```

DB_CONFIG.PY

```
import os
import mysql.connector
def get_db_connection():
   return mysql.connector.connect(
      host=os.getenv('DB_HOST', 'localhost'),
      user=os.getenv('DB_USER', 'root'),
      password=os.getenv('DB_PASS', 'root@123'),
```

```
        database=os.getenv('DB_NAME', 'restaurant_booking'),
        autocommit=True
    )
```

APP.PY

```
from flask import Flask, render_template, request, redirect, url_for, session, flash
from db_config import get_db_connection
from functools import wraps
from datetime import datetime
app = Flask(__name__)
app.secret_key = "replace_with_a_secret_key"
def login_required(role=None):
    def decorator(fn):
        @wraps(fn)
        def wrapper(*args, **kwargs):
            if 'user' not in session:
                flash('Please log in first.', 'warning')
                return redirect(url_for('login'))
            if role and session['user']['role'] != role:
                flash('Access denied.', 'danger')
                return redirect(url_for('index'))
            return fn(*args, **kwargs)
        return wrapper
    return decorator
# ---- Home / Landing ----
@app.route('/')
def index():
    return render_template('index.html')
# ---- Register ----
@app.route('/register', methods=['GET','POST'])
def register():
    if request.method == 'POST':
```

```
        name = request.form.get('name','').strip()
        email = request.form.get('email','').strip().lower()
        password = request.form.get('password','').strip()
        phone = request.form.get('phone','').strip()
        if not (name and email and password):
            flash('Please fill all required fields.', 'warning')
            return render_template('register.html')
        conn = get_db_connection()
        cur = conn.cursor()
        try:
            cur.execute("INSERT INTO users (name,email,password,phone,role)
VALUES (%s,%s,%s,%s,%s)",
                        (name, email, password, phone, 'customer'))
            flash('Registration successful. Please login.', 'success')
            return redirect(url_for('login'))
        except Exception as e:
            conn.rollback()
            flash('Email already registered or error occurred.', 'danger')
        finally:
            cur.close(); conn.close()
    return render_template('register.html')
# ---- Login ----
@app.route('/login', methods=['GET','POST'])
def login():
    if request.method == 'POST':
        email = request.form.get('email','').strip().lower()
        password = request.form.get('password','').strip()
        conn = get_db_connection()
        cur = conn.cursor(dictionary=True)
        cur.execute("SELECT * FROM users WHERE email=%s AND password=%s",
(email, password))
```

```
        user = cur.fetchone()
        cur.close(); conn.close()
        if user:
            session['user'] = {
                'id': user['id'],
                'name': user['name'],
                'email': user['email'],
                'role': user['role']
            }
            flash('Logged in successfully.', 'success')
            if user['role'] == 'admin':
                return redirect(url_for('admin_dashboard'))
            return redirect(url_for('booking'))
        flash('Invalid credentials.', 'danger')
    return render_template('login.html')
# ---- Logout ----
@app.route('/logout')
def logout():
    session.pop('user', None)
    flash('Logged out.', 'info')
    return redirect(url_for('index'))
# ---- Booking page ----
@app.route('/booking', methods=['GET','POST'])
@login_required(role='customer')
def booking():
    if request.method == 'POST':
        user_id = session['user']['id']
        booking_date = request.form.get('date')
        booking_time = request.form.get('time')
        guests = int(request.form.get('guests', 2))
        occasion = request.form.get('occasion','Other')
```

```
        preference = request.form.get('preference','Window Side')
        theme = request.form.get('theme','Family')
        section = request.form.get('section','AC')
        amount = float(request.form.get('amount', 500.00))
        # insert booking
        conn = get_db_connection()
        cur = conn.cursor()
        cur.execute("""
            INSERT INTO bookings (user_id, booking_date, booking_time, guests,
occasion, preference, theme, section, status, total_amount, payment_status)
            VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)
        """, (user_id, booking_date, booking_time, guests, occasion, preference, theme,
section, 'Pending', amount, 'Unpaid'))
        booking_id = cur.lastrowid
        cur.close(); conn.close()
        # store booking id in session for payment flow
        session['pending_booking_id'] = booking_id
        flash('Booking created. Proceed to payment to confirm.', 'info')
        return redirect(url_for('payment', booking_id=booking_id))
    # GET -> render
    return render_template('booking.html')
# ---- Payment simulation ----
@app.route('/payment/<int:booking_id>', methods=['GET','POST'])
@login_required(role='customer')
def payment(booking_id):
    # verify booking belongs to user and is unpaid/pending
    conn = get_db_connection()
    cur = conn.cursor(dictionary=True)
    cur.execute("SELECT * FROM bookings WHERE id=%s AND user_id=%s",
(booking_id, session['user']['id']))
    booking = cur.fetchone()
```

```
    cur.close(); conn.close()
    if not booking:
        flash('Invalid booking or permission denied.', 'danger')
        return redirect(url_for('booking'))
    if request.method == 'POST':
        # read mock card info (we do not validate security here)
        card_name = request.form.get('card_name','').strip()
        card_number = request.form.get('card_number','').strip()
        expiry = request.form.get('expiry','').strip()
        cvv = request.form.get('cvv','').strip()
        # mark as paid and confirmed
        conn = get_db_connection()
        cur = conn.cursor()
        cur.execute("UPDATE bookings SET payment_status=%s, status=%s WHERE
id=%s", ('Paid', 'Confirmed', booking_id))
        cur.execute("INSERT INTO transactions (booking_id, amount, method, status)
VALUES (%s,%s,%s,%s)",
                    (booking_id, booking['total_amount'], 'Card(Mock)', 'Success'))
        cur.close(); conn.close()
        flash('Payment successful. Booking confirmed!', 'success')
        # clear pending booking from session
        session.pop('pending_booking_id', None)
        return redirect(url_for('history'))
    return render_template('payment.html', booking=booking)
# ---- Booking history ----
@app.route('/history')
@login_required()
def history():
    user = session['user']
    conn = get_db_connection()
    cur = conn.cursor(dictionary=True)
```

```
    if user['role'] == 'customer':
        cur.execute("SELECT * FROM bookings WHERE user_id=%s ORDER BY
created_at DESC", (user['id'],))
        bookings = cur.fetchall()
        cur.close(); conn.close()
        return render_template('history.html', bookings=bookings)
    else:
        # admins get all bookings
        cur.execute("SELECT b.*, u.name AS customer_name FROM bookings b LEFT
JOIN users u ON b.user_id=u.id ORDER BY b.created_at DESC")
        bookings = cur.fetchall()
        cur.close(); conn.close()
        return render_template('admin_view_bookings.html', bookings=bookings)
# ---- Cancel booking (customer) ----
@app.route('/cancel/<int:booking_id>', methods=['POST'])
@login_required(role='customer')
def cancel_booking(booking_id):
    # basic cancellation policy: fixed fee 100
    cancellation_fee = 100.00
    conn = get_db_connection()
    cur = conn.cursor()
    # ensure belongs to user and not already cancelled
    cur.execute("SELECT * FROM bookings WHERE id=%s AND user_id=%s",
(booking_id, session['user']['id']))
    b = cur.fetchone()
    if not b:
        cur.close(); conn.close()
        flash('Booking not found.', 'danger')
        return redirect(url_for('history'))
    cur.execute("UPDATE bookings SET status=%s, cancellation_fee=%s WHERE
id=%s", ('Cancelled', cancellation_fee, booking_id))
```

```
    cur.execute("INSERT INTO transactions (booking_id, amount, method, status)
VALUES (%s,%s,%s,%s)",
                (booking_id, -cancellation_fee, 'CancellationFee', 'Success'))
    cur.close(); conn.close()
    flash(f'Booking {booking_id} cancelled. ₹{cancellation_fee} cancellation fee
applied.', 'info')
    return redirect(url_for('history'))
# ---- Feedback (customer) ----
@app.route('/feedback/<int:booking_id>', methods=['GET','POST'])
@login_required(role='customer')
def feedback(booking_id):
    # ensure booking belongs to user
    conn = get_db_connection()
    cur = conn.cursor(dictionary=True)
    cur.execute("SELECT * FROM bookings WHERE id=%s AND user_id=%s",
(booking_id, session['user']['id']))
    booking = cur.fetchone()
    if not booking:
        cur.close(); conn.close()
        flash('Invalid booking for feedback.', 'danger')
        return redirect(url_for('history'))
    if request.method == 'POST':
        rating = int(request.form.get('rating', 5))
        comment = request.form.get('comment','').strip()
        cur = conn.cursor()
        cur.execute("INSERT INTO feedback (booking_id, user_id, rating, comment)
VALUES (%s,%s,%s,%s)",
                    (booking_id, session['user']['id'], rating, comment))
        cur.close(); conn.close()
        flash('Thank you for your feedback!', 'success')
        return redirect(url_for('history'))
```

```
    cur.close(); conn.close()
    return render_template('feedback.html', booking=booking)
# ---- Admin dashboard (simple) ----
@app.route('/admin')
@login_required(role='admin')
def admin_dashboard():
    conn = get_db_connection()
    cur = conn.cursor(dictionary=True)
    # totals
    cur.execute("SELECT COUNT(*) AS total_bookings FROM bookings")
    totals = cur.fetchone()
    # recent bookings
    cur.execute("SELECT b.*, u.name AS customer_name FROM bookings b LEFT
JOIN users u ON b.user_id=u.id ORDER BY created_at DESC LIMIT 10")
    recent = cur.fetchall()
    cur.close(); conn.close()
    return render_template('admin_dashboard.html', totals=totals, recent=recent)
# ---- Admin view feedback ----
@app.route('/admin/feedback')
@login_required(role='admin')
def admin_feedback():
    conn = get_db_connection()
    cur = conn.cursor(dictionary=True)
    cur.execute("""SELECT f.*, u.name AS customer_name, b.booking_date
                FROM feedback f
                LEFT JOIN users u ON f.user_id = u.id
                LEFT JOIN bookings b ON f.booking_id = b.id
                ORDER BY f.created_at DESC""")
    feedbacks = cur.fetchall()
    cur.close(); conn.close()
    return render_template('admin_feedback.html', feedbacks=feedbacks)
```

```
# ---- Admin update booking status ----
@app.route('/admin/update_status', methods=['POST'])
@login_required(role='admin')
def admin_update_status():
    booking_id = int(request.form.get('booking_id', 0))
    new_status = request.form.get('status')
    conn = get_db_connection()
    cur = conn.cursor()
    cur.execute("UPDATE bookings SET status=%s WHERE id=%s", (new_status,
booking_id))
    cur.close(); conn.close()
    flash('Booking status updated.', 'success')
    return redirect(url_for('admin_dashboard'))
if __name__ == '__main__':
    app.run(debug=True)
```

LOGIN.HTML

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Login</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
  <div class="header">
    <div class="brand">CozyBistro</div>
  </div>
  <div class="container">
    <div class="form-card">
      <h2>Login</h2>
      <form method="post">
```

```
      <label class="label">Email</label>
      <input class="input" name="email" type="email" required>
      <label class="label">Password</label>
      <input class="input" name="password" type="password" required>
      <div style="margin-top:12px;">
        <button class="btn" type="submit">Login</button>
        <a class="btn secondary" href="{{ url_for('register') }}" style="margin-
left:10px;">Register</a>
      </div>
    </form>
  </div>
 </div>
</body>
</html>
```

REGISTER.HTML

```
<!doctype html>
<html>
<head>
 <meta charset="utf-8">
 <title>Register</title>
 <link rel="stylesheet" href="{{ url_for('static', filename='style.css')
}}">
</head>
<body>
 <div class="header">
   <div class="brand">CozyBistro</div>
 </div>
 <div class="container">
   <div class="form-card">
```

```
    <h2>Create an account</h2>
    <form method="post">
      <label class="label">Full name</label>
      <input class="input" name="name" required>
      <label class="label">Email</label>
      <input class="input" name="email" type="email" required>
      <label class="label">Phone</label>
      <input class="input" name="phone" type="text">
      <label class="label">Password</label>
      <input class="input" name="password" type="password"
required>
      <div style="margin-top:12px;">
        <button class="btn" type="submit">Register</button>
        <a class="btn secondary" href="{{ url_for('login') }}"
style="margin-left:10px;">Already have account</a>
      </div>
    </form>
  </div>
 </div>
</body>
</html>
```

ADMIN_DASHBOARD.HTML

```
<!doctype html>
<html>
<head>
 <meta charset="utf-8">
 <title>Admin Dashboard</title>
 <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
```

```
</head>
<body>
  <div class="header"><div class="brand">CozyBistro</div>
    <div><a href="{{ url_for('admin_feedback') }}">Feedback</a> | <a href="{{
url_for('logout') }}">Logout</a></div>
  </div>
  <div class="container">
    <h2>Admin Dashboard</h2>
    <div class="card-row">
      <div class="card">
        <h3>Total Bookings</h3>
        <p style="font-size:28px; font-weight:700;">{{ totals.total_bookings
}}</p>
      </div>
    </div>
    <h3>Recent Bookings</h3>
    <table class="table">
<thead><tr><th>ID</th><th>Customer</th><th>Date</th><th>Time</th><th>
Guests</th><th>Occasion</th><th>Status</th><th>Action</th></tr></thead>
      <tbody>
        {% for b in recent %}
        <tr>
          <td>{{ b.id }}</td>
          <td>{{ b.customer_name or 'N/A' }}</td>
          <td>{{ b.booking_date }}</td>
          <td>{{ b.booking_time }}</td>
          <td>{{ b.guests }}</td>
          <td>{{ b.occasion }}</td>
          <td>{{ b.status }}</td>
          <td>
            <form method="post" action="{{ url_for('admin_update_status') }}">
```

```
            <input type="hidden" name="booking_id" value="{{ b.id }}">
            <select name="status">
              <option {% if b.status=='Pending' %}selected{% endif
%}>Pending</option>
              <option {% if b.status=='Confirmed' %}selected{% endif
%}>Confirmed</option>
              <option {% if b.status=='Paid' %}selected{% endif %}>Paid</option>
              <option {% if b.status=='Cancelled' %}selected{% endif
%}>Cancelled</option>
              <option {% if b.status=='Completed' %}selected{% endif
%}>Completed</option>
            </select>
            <button class="btn" type="submit">Update</button>
          </form>
        </td>
      </tr>
      {% endfor %}
    </tbody>
  </table>
 </div>
</body>
</html>
```

BOOKING.HTML

```
<!doctype html>
<html>
<head>
 <meta charset="utf-8">
 <title>Book a Table</title>
 <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
```

```
<div class="header">
  <div class="brand">CozyBistro</div>
  <div>
    <a href="{{ url_for('history') }}">My Bookings</a> |
    <a href="{{ url_for('logout') }}">Logout</a>
  </div>
</div>
<div class="container">
  <div class="form-card">
    <h2>Book Your Table</h2>
    <form method="post">
      <div class="form-row">
        <div style="flex:1;">
          <label class="label">Date</label>
          <input class="input" type="date" name="date" required>
        </div>
        <div style="width:160px;">
          <label class="label">Time</label>
          <input class="input" type="time" name="time" required>
        </div>
      </div>
      <label class="label">No. of Guests</label>
      <input class="input" type="number" name="guests" min="1" value="2">
      <label class="label">Occasion</label>
      <select class="input" name="occasion">
        <option>Birthday</option>
        <option>Anniversary</option>
        <option>Business Meeting</option>
        <option>Other</option>
      </select>
      <label class="label">Preferred Table</label>
```

```
        <select class="input" name="preference">
          <option>Window Side</option>
          <option>Outdoor</option>
          <option>AC Section</option>
          <option>Non-AC Section</option>
        </select>
        <label class="label">Theme / Ambience</label>
        <select class="input" name="theme">
          <option>Romantic</option>
          <option>Family</option>
          <option>Quiet Zone</option>
          <option>Party Area</option>
        </select>
        <label class="label">Section</label>
        <select class="input" name="section">
          <option>AC</option>
          <option>Non-AC</option>
        </select>
        <input type="hidden" name="amount" value="500">
        <div style="margin-top:12px;">
          <button class="btn" type="submit">Proceed to Payment (₹500)</button>
        </div>
      </form>
    </div>
  </div>
</body>
</html>
```

PAYMENT.HTML

```
<!doctype html>
<html>
```

```
<head>
  <meta charset="utf-8">
  <title>Payment</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
  <div class="header">
    <div class="brand">CozyBistro</div>
  </div>
  <div class="container">
    <div class="form-card">
      <h2>Secure Payment (Demo)</h2>
      <p>Booking ID: {{ booking.id }}   |   Amount: ₹{{
booking.total_amount }}</p>
      <form method="post">
        <label class="label">Card Holder Name</label>
        <input class="input" name="card_name" required>
        <label class="label">Card Number</label>
        <input class="input" name="card_number" maxlength="16" required
placeholder="XXXX XXXX XXXX XXXX">
        <div class="form-row">
          <div style="flex:1;">
            <label class="label">Expiry</label>
            <input class="input" type="month" name="expiry" required>
          </div>
          <div style="width:140px;">
            <label class="label">CVV</label>
            <input class="input" name="cvv" maxlength="4" required>
          </div>
        </div>
        <div style="margin-top:12px;">
```

```
        <button class="btn" type="submit">Pay Now</button>
      </div>
    </form>
  </div>
 </div>
</body>
</html>
```

FEEDBACK.HTML

```
<!doctype html>
<html>
<head>
 <meta charset="utf-8">
 <title>Feedback</title>
 <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
 <div class="header"><div class="brand">CozyBistro</div></div>
 <div class="container">
  <div class="form-card">
   <h2>Feedback for Booking #{{ booking.id }}</h2>
   <p>Date: {{ booking.booking_date }} | Time: {{ booking.booking_time }}</p>
   <form method="post">
    <label class="label">Rating</label>
    <select class="input" name="rating" required>
     <option value="5">5 - Excellent</option>
     <option value="4">4 - Very Good</option>
     <option value="3">3 - Good</option>
     <option value="2">2 - Poor</option>
     <option value="1">1 - Very Poor</option>
    </select>
    <label class="label">Comments</label>
```

```
      <textarea class="input" name="comment" rows="4" required></textarea>
      <div style="margin-top:12px;"><button class="btn" type="submit">Submit
Feedback</button></div>
    </form>
   </div>
  </div>
</body>
</html>
```

STYLES.CSS

```
/* static/style.css - Elegant indoor restaurant theme (light & warm) */
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600;700&displ
ay=swap');

:root{
  --bg1: #f6f2ef;
  --accent: #8b5e3c;    /* warm brown */
  --accent-2: #c69c6d;  /* light gold */
  --muted: #6b6b6b;
  --card: rgba(255,255,255,0.9);
}
*{box-sizing:border-box}
body{
  margin:0;
  font-family: 'Poppins', sans-serif;
  background: linear-gradient(135deg, #fefefe 0%, #f6f2ef 100%);
  color:#333;
  -webkit-font-smoothing:antialiased;
}
.header{
  background: linear-gradient(90deg, rgba(139,94,60,0.95), rgba(198,156,109,0.95));
```

```
  color: #fff;
  padding: 28px 12%;
  display:flex;
  justify-content:space-between;
  align-items:center;
}
.header .brand { font-size:24px; font-weight:700; }
.header a { color:#fff; text-decoration:none; margin-left:16px; font-weight:600; }
.container{ max-width:1100px; margin:28px auto; padding:0 20px; }
.banner{
  background-image: url('/static/images/bg.jpg');
  background-size:cover;
  background-position:center;
  height:340px;
  border-radius:16px;
  display:flex;
  align-items:center;
  color:#fff;
  padding:28px;
  margin-bottom:18px;
  box-shadow: 0 12px 30px rgba(0,0,0,0.12);
}
.banner .left{ max-width:60%; }
.banner h1{ font-size:34px; margin:0 0 8px; text-shadow: 0 4px 16px
rgba(0,0,0,0.25); }
.banner p{ margin:0 0 16px; opacity:0.95; }
.card-row{ display:flex; gap:18px; margin-bottom:18px; flex-wrap:wrap; }
.card{
  background:var(--card);
  border-radius:12px;
  padding:18px;
```

```
  flex:1;
  min-width:220px;
  box-shadow: 0 6px 18px rgba(0,0,0,0.06);
}
.card h3{ margin:4px 0 8px; color:var(--accent); }
.card p{ margin:0; color:var(--muted); font-size:14px; }
.form-card{
  background:var(--card);
  padding:20px;
  border-radius:12px;
  box-shadow: 0 8px 24px rgba(0,0,0,0.06);
  max-width:700px;
  margin: 0 auto 24px;
}
.form-row{ display:flex; gap:12px; flex-wrap:wrap; }
.input, select, textarea{
  width:100%;
  padding:10px 12px;
  border-radius:8px;
  border:1px solid #e6e0da;
  background:#fff;
  font-size:15px;
}
.label{ font-weight:600; font-size:14px; margin-bottom:6px; display:block;
color:#333; }
.btn{
  display:inline-block;
  background: linear-gradient(90deg,var(--accent),var(--accent-2));
  color:#fff;
  padding:10px 18px;
  border-radius:10px;
```

```
  border:none;
  cursor:pointer;
  font-weight:600;
}
.btn.secondary{ background:#fff; color:var(--accent); border:1px solid #e6d7c7; }
.table{
  width:100%;
  border-collapse:collapse;
  background:#fff;
  border-radius:8px;
  overflow:hidden;
  box-shadow: 0 8px 20px rgba(0,0,0,0.04);
}
.table th, .table td{
  padding:12px 14px;
  border-bottom:1px solid #f1e9e1;
  text-align:left;
}
.table th { background:#faf6f3; color:var(--muted); font-weight:600; }
.footer{ text-align:center; padding:18px 0; color:var(--muted); font-size:14px; margin-
top:24px; }
a.link { color:var(--accent); text-decoration:none; font-weight:600; }
@media (max-width:800px){
  .banner{ height:240px; padding:18px; }
  .banner .left{ max-width:100%; }
  .card-row{ flex-direction:column; }
}
```