

# Household Marketplace — Ready-to-use web app

Below are the three full files (copy-paste ready). Use these on GitHub Pages or open locally.

## index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width,initial-scale=1" />
  <title>Household Marketplace</title>
  <link rel="stylesheet" href="style.css" />
</head>
<body>
  <header class="topbar">
    <h1>Household Marketplace</h1>
    <div class="controls">
      <input id="search" type="search" placeholder="Search items..." />
      <button id="openFormBtn" class="btn">Add Item</button>
    </div>
  </header>

  <main class="container">
    <section class="form-panel hidden" id="formPanel">
      <form id="itemForm">
        <h2 id="formTitle">Add Item</h2>
        <label>Image (one only)</label>
        <input id="imageInput" type="file" accept="image/*" />
        <div id="previewWrap" class="preview-wrap hidden"><img id="imgPreview"
alt="preview"/></div>

        <label>Name</label>
        <input id="nameInput" type="text" required maxlength="80" />

        <label>Description</label>
        <textarea id="descInput" rows="3" maxlength="600"></textarea>

        <div class="row">
          <div class="col">
            <label>Price</label>
            <input id="priceInput" type="text" placeholder="e.g. 1200 LKR or
$10" />
```

```

        </div>
        <div class="col">
            <label>Condition (optional)</label>
            <input id="conditionInput" type="text" placeholder="e.g. Good" />
        </div>
    </div>

    <label>Location (optional)</label>
    <input id="locationInput" type="text" />

    <label>Contact info</label>
    <input id="contactInput" type="text" required placeholder="phone or
email" />

    <div class="form-actions">
        <button type="submit" class="btn primary">Save</button>
        <button type="button" id="cancelBtn" class="btn">Cancel</button>
    </div>
</form>
</section>

<section class="list-panel">
    <div id="list" class="grid"></div>
</section>
</main>

<template id="cardTpl">
    <article class="card">
        <div class="card-media"><img src="" alt="item image"/></div>
        <div class="card-body">
            <h3 class="item-name"></h3>
            <p class="item-desc"></p>
            <p class="meta"><span class="price"></span> <span class="condition"></
span></p>
            <p class="meta small"><span class="location"></span> • <span
class="datetime"></span></p>
            <p class="contact small"></p>
        </div>
        <div class="card-actions">
            <button class="btn markSold">Mark Sold</button>
            <button class="btn edit">Edit</button>
            <button class="btn danger delete">Delete</button>
        </div>
    </article>
</template>

<script src="app.js"></script>

```

```
</body>
</html>
```

## style.css

```
:root{
  --bg:#f7f9fb;
  --card:#ffffff;
  --muted:#6b7280;
  --accent:#2563eb;
  --danger:#ef4444;
  --glass: rgba(255,255,255,0.6);
}
*{box-sizing:border-box}
html,body{height:100%}
body{
  margin:0;font-family:Inter, system-ui, -apple-system, 'Segoe UI', Roboto,
  'Helvetica Neue', Arial;
  background:linear-gradient(180deg,var(--bg),#ffffff);color:#0f172a;
}
.topbar{display:flex;align-items:center;justify-content:space-between;padding:
14px 20px;background:var(--card);box-shadow:0 1px 4px rgba(2,6,23,
0.06);position:sticky;top:0;z-index:10}
.topbar h1{font-size:18px;margin:0}
.controls{display:flex;gap:10px;align-items:center}
input[type=search]{padding:8px 12px;border-radius:10px;border:1px solid
#e6eef8;min-width:180px}
.btn{background:transparent;border:1px solid #cbd5e1;padding:8px 10px;border-
radius:8px;cursor:pointer}
.btn:hover{filter:brightness(0.98)}
.btn.primary{background:var(--accent);color:white;border-color:transparent}
.btn.danger{background:var(--danger);color:white;border-color:transparent}
.container{display:flex;gap:20px;padding:18px;max-width:1100px;margin:24px auto}
.form-panel{width:360px;background:var(--card);padding:14px;border-radius:
12px;box-shadow:0 6px 18px rgba(2,6,23,0.06)}
.form-panel.hidden{display:none}
.list-panel{flex:1}
label{display:block;margin-top:8px;font-size:13px;color:var(--muted)}
input[type=text],input[type=file],textarea{width:100%;padding:8px;border-radius:
8px;border:1px solid #e6eef8;margin-top:6px}
.row{display:flex;gap:10px}
.col{flex:1}
.form-actions{display:flex;gap:8px;margin-top:12px}
.preview-wrap{margin-top:8px}
```

```

.preview-wrap img{max-width:100%;height:auto;border-radius:8px;border:1px solid
#e6eef8}
.grid{display:grid;grid-template-columns:repeat(auto-fill,minmax(240px,
1fr));gap:16px}
.card{background:var(--card);border-radius:
12px;overflow:hidden;display:flex;flex-direction:column;box-shadow:0 8px 18px
rgba(2,6,23,0.06)}
.card-media img{width:100%;height:180px;object-fit:cover}
.card-body{padding:12px}
.item-name{margin:0 0 6px 0;font-size:16px}
.item-desc{margin:0 0 10px 0;color:var(--muted);font-size:14px}
.meta{color:var(--muted);font-size:13px;margin:0}
.small{font-size:12px;color:var(--muted)}
.card-actions{display:flex;gap:8px;padding:10px;border-top:1px solid #f1f5f9}
.card-actions .btn{flex:1;padding:8px;font-size:13px}
.card .sold-overlay{position:absolute;top:0;left:0;right:0;bottom:
0;background:rgba(0,0,0,0.45);color:white;display:flex;align-
items:center;justify-content:center;font-weight:700;font-size:20px}
@media(max-width:880px){.container{flex-direction:column;padding:12px}.form-
panel{width:100%}}

```

## app.js

```

// app.js - full functionality
const STORAGE_KEY = 'market_items';
const MAX_IMAGE_BYTES = 200 * 1024; // 200KB
const MAX_IMAGE_WIDTH = 600;

// UI refs
const itemForm = document.getElementById('itemForm');
const imageInput = document.getElementById('imageInput');
const imgPreview = document.getElementById('imgPreview');
const previewWrap = document.getElementById('previewWrap');
const nameInput = document.getElementById('nameInput');
const descInput = document.getElementById('descInput');
const priceInput = document.getElementById('priceInput');
const contactInput = document.getElementById('contactInput');
const conditionInput = document.getElementById('conditionInput');
const locationInput = document.getElementById('locationInput');
const listEl = document.getElementById('list');
const search = document.getElementById('search');
const openFormBtn = document.getElementById('openFormBtn');
const formPanel = document.getElementById('formPanel');
const cancelBtn = document.getElementById('cancelBtn');

```

```

const formTitle = document.getElementById('formTitle');
const cardTpl = document.getElementById('cardTpl');

let items = [];
let editId = null; // currently editing id
let currentImageBase64 = '';

// Utility: load & save
function loadItems(){
  try{
    const raw = localStorage.getItem(STORAGE_KEY) || '[]';
    return JSON.parse(raw);
  }catch(e){
    console.error('Failed to parse items', e);
    return [];
  }
}

function saveItems(arr){
  localStorage.setItem(STORAGE_KEY, JSON.stringify(arr));
}

// On first load, seed demo items
function seedDemo(){
  const existing = loadItems();
  if(existing.length) return;
  const now = Date.now();
  const demo = [
    {id:'demo-1', name:'Stainless Steel Kettle',
description:'1.5L kettle, works fine, small dent', price:'$8',
contact:'demo@seller.com', condition:'Good', location:'Colombo', datetime:
now-86400000*3, image: '', sold:false},
    {id:'demo-2', name:'Wooden Chair', description:'Solid wood, some
scratches', price:'$15', contact:'chair@example.com', condition:'Fair',
location:'Kandy', datetime: now-86400000*5, image:'', sold:false},
    {id:'demo-3', name:'Set of Plates', description:'12 ceramic plates, like
new', price:'$12', contact:'plates@example.com', condition:'Like new',
location:'Galle', datetime: now-86400000*1, image:'', sold:false},
    {id:'demo-4', name:'LED Lamp', description:'Desk lamp, adjustable',
price:'$6', contact:'lamp@example.com', condition:'Good', location:'Colombo',
datetime: now-86400000*6, image:'', sold:false},
    {id:'demo-5', name:'Microwave (small)', description:'Compact microwave,
700W', price:'$45', contact:'microwave@example.com', condition:'Good',
location:'Jaffna', datetime: now-86400000*12, image:'', sold:false}
  ];
  saveItems(demo);
}

// Render

```

```

function renderItem(filter='') {
  listEl.innerHTML='';
  const f = filter.trim().toLowerCase();
  items.sort((a,b)=>b.datetime - a.datetime);
  items.forEach(item=>{
    if(f){
      const hay = (item.name + ' ' + item.description).toLowerCase();
      if(!hay.includes(f)) return;
    }
    const node = cardTpl.content.cloneNode(true);
    const card = node.querySelector('.card');
    const imgEl = node.querySelector('.card-media img');
    const nameEl = node.querySelector('.item-name');
    const descEl = node.querySelector('.item-desc');
    const priceEl = node.querySelector('.price');
    const condEl = node.querySelector('.condition');
    const locEl = node.querySelector('.location');
    const dtEl = node.querySelector('.datetime');
    const contactEl = node.querySelector('.contact');

    imgEl.src = item.image || placeholderDataURL(item.name);
    imgEl.alt = item.name;
    nameEl.textContent = item.name;
    descEl.textContent = item.description || '';
    priceEl.textContent = item.price || '';
    condEl.textContent = item.condition ? ' • ' + item.condition : '';
    locEl.textContent = item.location || '';
    dtEl.textContent = new Date(item.datetime).toLocaleString();
    contactEl.textContent = 'Contact: ' + (item.contact || '-');

    // sold overlay
    if(item.sold){
      const ov = document.createElement('div');
      ov.className = 'sold-overlay';
      ov.textContent = 'SOLD';
      card.style.position = 'relative';
      card.appendChild(ov);
    }

    // actions
    const editBtn = node.querySelector('.edit');
    const delBtn = node.querySelector('.delete');
    const soldBtn = node.querySelector('.markSold');

    editBtn.addEventListener('click', ()=>openEdit(item.id));
    delBtn.addEventListener('click', ()=>{ if(confirm('Delete item?'))
deleteItem(item.id); });
    soldBtn.addEventListener('click', ()=>toggleSold(item.id));

```

```

        listEl.appendChild(node);
    });
}

function placeholderDataURL(text){
    // simple placeholder image as dataURL (generated with canvas)
    const c = document.createElement('canvas');
    c.width = 800; c.height = 450;
    const ctx = c.getContext('2d');
    ctx.fillStyle = '#e2e8f0';
    ctx.fillRect(0,0,c.width,c.height);
    ctx.fillStyle = '#94a3b8';
    ctx.font = '40px sans-serif';
    ctx.textAlign = 'center';
    ctx.fillText(text.slice(0,20), c.width/2, c.height/2);
    return c.toDataURL('image/png');
}

// Add / Edit / Delete
async function handleFormSubmit(e){
    e.preventDefault();
    const name = nameInput.value.trim();
    if(!name){ alert('Name required'); return; }
    const obj = {
        id: editId || ('id-' + Date.now()),
        name,
        description: descInput.value.trim(),
        price: priceInput.value.trim(),
        contact: contactInput.value.trim(),
        condition: conditionInput.value.trim(),
        location: locationInput.value.trim(),
        datetime: Date.now(),
        image: currentImageBase64 || '',
        sold: false
    };
    if(editId){
        items = items.map(it=>it.id===editId?{...it,...obj, datetime:
it.datetime}:it);
        editId = null;
    } else {
        items.push(obj);
    }
    saveItems(items);
    resetForm();
    hideForm();
    renderItems(search.value);
}

```

```

function deleteItem(id){
  items = items.filter(i=>i.id!==id);
  saveItems(items);
  renderItems(search.value);
}

function toggleSold(id){
  items = items.map(i=> i.id===id?{...i, sold: !i.sold}:i);
  saveItems(items);
  renderItems(search.value);
}

function openEdit(id){
  const it = items.find(x=>x.id===id); if(!it) return;
  editId = id;
  formTitle.textContent = 'Edit Item';
  nameInput.value = it.name;
  descInput.value = it.description || '';
  priceInput.value = it.price || '';
  contactInput.value = it.contact || '';
  conditionInput.value = it.condition || '';
  locationInput.value = it.location || '';
  currentImageBase64 = it.image || '';
  if(currentImageBase64){
    imgPreview.src = currentImageBase64; previewWrap.classList.remove('hidden');
  } else { previewWrap.classList.add('hidden'); }
  showForm();
}

function resetForm(){
  itemForm.reset();
  previewWrap.classList.add('hidden');
  currentImageBase64 = '';
  editId = null;
  formTitle.textContent = 'Add Item';
}

function showForm(){ formPanel.classList.remove('hidden'); window.scrollTo({top:
0,behavior:'smooth'}); }
function hideForm(){ formPanel.classList.add('hidden'); resetForm(); }

// Image compression
function fileToImage(file){
  return new Promise((res,rej)=>{
    const reader = new FileReader();
    reader.onload = ()=>{
      const img = new Image();

```

```

        img.onload = ()=>res(img);
        img.onerror = rej;
        img.src = reader.result;
    };
    reader.onerror = rej;
    reader.readAsDataURL(file);
  });
}

async function compressImage(file){
  // returns base64 under MAX_IMAGE_BYTES if possible
  try{
    const img = await fileToImage(file);
    const canvas = document.createElement('canvas');
    const scale = Math.min(1, MAX_IMAGE_WIDTH / img.width);
    canvas.width = Math.round(img.width * scale);
    canvas.height = Math.round(img.height * scale);
    const ctx = canvas.getContext('2d');
    ctx.drawImage(img,0,0,canvas.width,canvas.height);

    // try quality steps
    let quality = 0.9;
    let dataUrl = canvas.toDataURL('image/jpeg', quality);
    while(dataUrl.length > MAX_IMAGE_BYTES && quality > 0.2){
      quality -= 0.1;
      dataUrl = canvas.toDataURL('image/jpeg', quality);
    }
    // as fallback, if still big, reduce canvas size until small enough
    let w = canvas.width;
    while(dataUrl.length > MAX_IMAGE_BYTES && w > 200){
      w = Math.round(w * 0.85);
      const h = Math.round((w / canvas.width) * canvas.height);
      const c2 = document.createElement('canvas');
      c2.width = w; c2.height = h;
      c2.getContext('2d').drawImage(img,0,0,w,h);
      dataUrl = c2.toDataURL('image/jpeg', quality);
    }
    return dataUrl;
  }catch(e){
    console.error('compressImage failed', e);
    return '';
  }
}

// handlers
imageInput.addEventListener('change', async (e)=>{
  const f = e.target.files && e.target.files[0];
  if(!f) return;

```

```

    const compressed = await compressImage(f);
    if(!compressed){ alert('Failed to process image'); return; }
    currentImageBase64 = compressed;
    imgPreview.src = compressed;
    previewWrap.classList.remove('hidden');
  });

  itemForm.addEventListener('submit', handleFormSubmit);
  openFormBtn.addEventListener('click', ()=>{ showForm(); });
  cancelBtn.addEventListener('click', ()=>{ hideForm(); });
  search.addEventListener('input', ()=> renderItems(search.value));

  // Edit by double-clicking a card (optional UX):
  listEl.addEventListener('dblclick',(e)=>{
    const card = e.target.closest('.card');
    if(!card) return;
    const idx = Array.from(listEl.children).indexOf(card);
    // since we render template clones, map index to sorted items
  });

  // init
  (function init(){
    seedDemo();
    items = loadItems();
    renderItems();
  })();

```

---

## Notes & usage

- The app stores everything in `localStorage` under the key `market_items`. If you want to reset, clear `localStorage` in the browser.
- Images are compressed using canvas to JPEG. The code aims to get images  $\leq 200\text{KB}$ ; large photos may still be downsized further.
- Demo items are seeded on first load (no images). You can edit/delete them as requested.

Copy these three files into your repo root and publish on GitHub Pages. They are ready to use.