// ... (existing imports and data remain unchanged)

const knownUsers = ["alice", "bob", "charlie", "guru"]; const presetMessages = ["🔥 Great trade!", "💧 You earned this!", " Appreciate you!", "🐻Congrats!"]; const reactionEmojis = ["☁️", "🦬", "🦎", "🔥"];

export default function GuruDashboardMockup() { const [recipient, setRecipient] = useState(""); const [transferAmount, setTransferAmount] = useState(""); const [transferMessage, setTransferMessage] = useState(""); const [transferHistory, setTransferHistory] = useState([]); const [incomingTransfers, setIncomingTransfers] = useState([]);

const handleTransfer = () => { const amount = parseFloat(transferAmount); const validRecipient = knownUsers.includes(recipient.toLowerCase());

```
if (recipient && validRecipient && amount > 0) {
  const timestamp = new Date().toLocaleString();
  const newTransfer = {
    id: transferHistory.length + 1,
    to: recipient,
    from: "You",
    amount: amount.toFixed(2),
    date: timestamp,
    message: transferMessage,
    reaction: null
  };
  setTransferHistory([newTransfer, ...transferHistory]);
  setIncomingTransfers([newTransfer, ...incomingTransfers]);
  alert(`$${amount.toFixed(2)} sent to @${recipient}`);
  setRecipient("");
  setTransferAmount("");
  setTransferMessage("");
} else {
  alert("Invalid transfer amount or recipient.");
}
```

};

const handleReaction = (id, emoji) => { setIncomingTransfers(prev => prev.map(t => t.id === id ? { ...t, reaction: emoji } : t)); };

return (Send Funds (Beta) \<input type="text" placeholder="Recipient username" className="w-full p-2 border rounded mb-2" value={recipient} onChange={(e) => setRecipient(e.target.value)} /> \<input type="number" placeholder="Amount to send" className="w-full p-2 border rounded mb-2" value={transferAmount} onChange={(e) => setTransferAmount(e.target.value)} /> \<input type="text" placeholder="Add a note (optional)" className="w-full p-2 border rounded mb-2" value={transferMessage} onChange={(e) => setTransferMessage(e.target.value)} /> {presetMessages.map((msg, idx) => (\<button

key={idx} className="bg-gray-100 text-sm px-2 py-1 rounded hover\:bg-gray-200" onClick={() => setTransferMessage(msg)}>{msg}))} Send

```
  <div className="bg-white p-4 rounded-2xl shadow mb-4">
    <h2 className="text-md font-semibold mb-2">Social Feed</h2>
    <ul className="text-sm">
      {incomingTransfers.map((t) => (
        <li key={t.id} className="py-2 border-b">
          <strong>@{t.from}</strong> sent <strong>${t.amount}</strong> to
<strong>@{t.to}</strong> on {t.date}<br />
          <em className="text-gray-500">{t.message}</em>
          <div className="mt-1 flex gap-2 text-xl">
            {reactionEmojis.map((emoji, i) => (
              <button key={i} className="hover:scale-110 transition" onClick={()
=> handleReaction(t.id, emoji)}>{emoji}</button>
            ))}
          </div>
          {t.reaction && <p className="text-lg mt-1">Reaction: {t.reaction}</p>}
        </li>
      ))}
      {incomingTransfers.length === 0 && (
        <li className="text-gray-500">No recent transfers</li>
      )}
    </ul>
  </div>
</div>
```

); }