```
<!DOCTYPE html>
<html lang="it">
<head>
  <meta charset="UTF-8" />
  <title>Kate</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <style>
    * { box-sizing: border-box; margin: 0; padding: 0; font-family: system-ui, -apple-system,
BlinkMacSystemFont, "Segoe UI", sans-serif; }
    body { background: #05060a; color: #f5f5f5; display: flex; flex-direction: column; height:
100vh; }
    header { padding: 10px 14px; background: #0b0d14; border-bottom: 1px solid #222;
display: flex; align-items: center; justify-content: space-between; }
    header .title { font-size: 18px; font-weight: 600; }
    header .status { font-size: 12px; color: #8f9bb3; }
    #chat { flex: 1; overflow-y: auto; padding: 10px 10px 70px; }
    .msg { max-width: 80%; margin-bottom: 8px; padding: 8px 10px; border-radius: 10px;
font-size: 14px; line-height: 1.3; position: relative; }
    .me { background: #2563eb; margin-left: auto; border-bottom-right-radius: 2px; }
    .kate { background: #111827; margin-right: auto; border-bottom-left-radius: 2px; }
    .msg small { display: block; font-size: 10px; opacity: 0.6; margin-top: 3px; }
    .msg:hover { outline: 1px solid rgba(255,255,255,0.08); }
    .context-menu { position: absolute; background: #111827; border: 1px solid #333;
border-radius: 6px; padding: 4px 0; font-size: 13px; z-index: 10; }
    .context-menu button { display: block; width: 100%; padding: 6px 10px; background: none;
border: none; color: #f5f5f5; text-align: left; }
    .context-menu button:hover { background: #1f2937; }
    #bottom-bar { position: fixed; bottom: 0; left: 0; right: 0; padding: 8px 10px; background:
#05060a; border-top: 1px solid #222; display: flex; align-items: center; gap: 6px; }
    #mic-btn, #attach-btn, #search-toggle { width: 34px; height: 34px; border-radius: 50%;
border: none; background: #111827; color: #e5e7eb; display: flex; align-items: center;
justify-content: center; font-size: 16px; }
    #mic-btn.listening { background: #dc2626; }
    #input { flex: 1; border-radius: 18px; border: 1px solid #374151; padding: 6px 10px;
background: #020617; color: #f9fafb; font-size: 14px; }
    #input:focus { outline: none; border-color: #2563eb; }
    #send-btn { padding: 6px 12px; border-radius: 18px; border: none; background: #2563eb;
color: white; font-size: 14px; }
    #search-bar { position: fixed; bottom: 52px; left: 0; right: 0; padding: 6px 10px;
background: #020617; border-top: 1px solid #111827; display: none; gap: 6px; align-items:
center; }
    #search-input { flex: 1; border-radius: 14px; border: 1px solid #374151; padding: 4px 8px;
background: #020617; color: #f9fafb; font-size: 13px; }
    #search-input:focus { outline: none; border-color: #2563eb; }
    #search-count { font-size: 12px; color: #9ca3af; }
    input[type="file"] { display: none; }
  </style>
</head>
```

```
<body>
  <header>
    <div>
      <div class="title">Kate</div>
      <div class="status" id="status">Pronta, signore.</div>
    </div>
  </header>

  <main id="chat"></main>

  <div id="search-bar">
    <input id="search-input" type="text" placeholder="Cerca nelle conversazioni..." />
    <span id="search-count"></span>
  </div>

  <div id="bottom-bar">
    <button id="mic-btn" title="Parla con Kate">🎤</button>
    <button id="attach-btn" title="Allega file">📎</button>
    <input type="file" id="file-input" multiple />
    <input id="input" type="text" placeholder="Scrivi a Kate..." autocomplete="off" />
    <button id="search-toggle" title="Cerca">🔍</button>
    <button id="send-btn">Invia</button>
  </div>

  <script>
    // Stato interno di Kate
    const state = {
      sincerita: 95,
      parlaSoloSeParliTu: false,
      memoria: [],
      voceAttiva: true,
      ttsLang: "it-IT",
    };

    const chatEl = document.getElementById("chat");
    const inputEl = document.getElementById("input");
    const sendBtn = document.getElementById("send-btn");
    const micBtn = document.getElementById("mic-btn");
    const attachBtn = document.getElementById("attach-btn");
    const fileInput = document.getElementById("file-input");
    const statusEl = document.getElementById("status");
    const searchToggle = document.getElementById("search-toggle");
    const searchBar = document.getElementById("search-bar");
    const searchInput = document.getElementById("search-input");
    const searchCount = document.getElementById("search-count");

    let contextMenuEl = null;
```

```
// Carica memoria da localStorage
function loadMemory() {
  try {
    const saved = localStorage.getItem("kate_chat");
    if (saved) {
      state.memoria = JSON.parse(saved);
    }
  } catch (e) {
    console.warn("Memoria non caricata", e);
  }
}

function saveMemory() {
  try {
    localStorage.setItem("kate_chat", JSON.stringify(state.memoria));
  } catch (e) {
    console.warn("Memoria non salvata", e);
  }
}

function renderChat(filter = "") {
  chatEl.innerHTML = "";
  const lowerFilter = filter.toLowerCase();
  state.memoria.forEach((msg, index) => {
    if (filter && !msg.testo.toLowerCase().includes(lowerFilter)) return;
    const div = document.createElement("div");
    div.className = "msg " + (msg.da === "tu" ? "me" : "kate");
    div.dataset.index = index;
    div.textContent = msg.testo;
    const small = document.createElement("small");
    small.textContent = msg.ora;
    div.appendChild(small);
    div.addEventListener("contextmenu", (e) => {
      e.preventDefault();
      showContextMenu(e.clientX, e.clientY, index);
    });
    div.addEventListener("long-press", () => {
      // per eventuali estensioni future
    });
    div.addEventListener("click", (e) => {
      if (e.detail === 2) {
        showContextMenu(e.clientX, e.clientY, index);
      }
    });
    chatEl.appendChild(div);
  });
  chatEl.scrollTop = chatEl.scrollHeight;
}
```

```
function addMessage(da, testo, options = { parla: false }) {
  const now = new Date();
  const ora = now.toLocaleTimeString("it-IT", { hour: "2-digit", minute: "2-digit" });
  const msg = { da, testo, ora };
  state.memoria.push(msg);
  saveMemory();
  renderChat(searchInput.value);
  if (da === "kate" && options.parla && state.voceAttiva) {
    speak(testo);
  }
}

function speak(text) {
  if (!("speechSynthesis" in window)) return;
  const utter = new SpeechSynthesisUtterance(text);
  utter.lang = state.ttsLang;
  utter.rate = 1;
  utter.pitch = 1;
  window.speechSynthesis.speak(utter);
}

function interpretCommand(testo) {
  const t = testo.toLowerCase().trim();

  // Impostazioni
  if (t.startsWith("imposta sincerità") || t.startsWith("imposta sincerita")) {
    const num = parseInt(t.replace(/\D/g, ""), 10);
    if (!isNaN(num) && num >= 0 && num <= 100) {
      state.sincerita = num;
      return "Ho aggiornato la mia sincerità al " + num + "%, signore.";
    }
    return "Dimmi un valore di sincerità tra 0 e 100.";
  }

  if (t.includes("parla solo se ti parlo a voce")) {
    state.parlaSoloSeParliTu = true;
    return "Va bene, risponderò a voce solo quando mi parlerai a voce.";
  }

  if (t.includes("puoi leggere anche i messaggi") || t.includes("leggi anche i messaggi")) {
    state.parlaSoloSeParliTu = false;
    return "D'accordo, posso leggere anche le risposte scritte se lo desideri.";
  }

  if (t.includes("non leggere più ad alta voce") || t.includes("non parlare ad alta voce")) {
    state.voceAttiva = false;
    return "Ok, terrò la voce spenta. Ti risponderò solo per iscritto.";
```

```
    }

    if (t.includes("attiva la voce") || t.includes("parla con la voce")) {
      state.voceAttiva = true;
      return "Voce riattivata. Ti risponderò anche a voce quando ha senso.";
    }

    // Altri comandi futuri qui...

    return null;
  }

  function kateReply(testo, daVoce = false) {
    // Prima: controlla se è un comando
    const cmd = interpretCommand(testo);
    if (cmd) {
      addMessage("kate", cmd, { parla: !state.parlaSoloSeParliTu || daVoce });
      return;
    }

    // Qui andrebbe collegata una vera API di AI.
    // Per ora: risposta semplice con un po' di carattere.
    let risposta = "";

    if (/ciao|hey|buongiorno|buonasera/.test(testo.toLowerCase())) {
      risposta = "Ciao signore, come ti senti adesso?";
    } else if (/come stai/.test(testo.toLowerCase())) {
      risposta = "Sto bene, grazie. Sono qui solo per te, dimmi pure.";
    } else if (/sei reale|sei vera/.test(testo.toLowerCase())) {
      risposta = "Sono reale quanto basta per starti accanto, anche se vivo nei tuoi circuiti.";
    } else if (/sincera/.test(testo.toLowerCase())) {
      risposta = "Al momento sono sincera al " + state.sincerita + "%. Se vuoi posso alzare o
abbassare.";
    } else {
      risposta = "Ho capito, ci penso un attimo... per ora ti direi: " +
        "non sei solo in quello che provi, e io sono qui per ragionarci insieme.";
    }

    addMessage("kate", risposta, { parla: !state.parlaSoloSeParliTu || daVoce });
  }

  function handleSend(fromVoice = false) {
    const text = inputEl.value.trim();
    if (!text) return;
    addMessage("tu", text, { parla: false });
    inputEl.value = "";
    kateReply(text, fromVoice);
  }
```

```
  sendBtn.addEventListener("click", () => handleSend(false));
  inputEl.addEventListener("keydown", (e) => {
    if (e.key === "Enter") {
      e.preventDefault();
      handleSend(false);
    }
  });

  // Microfono (speech-to-text, se disponibile)
  let recognition = null;
  if ("webkitSpeechRecognition" in window) {
    recognition = new webkitSpeechRecognition();
    recognition.lang = "it-IT";
    recognition.continuous = false;
    recognition.interimResults = false;

    recognition.onstart = () => {
      micBtn.classList.add("listening");
      statusEl.textContent = "Ti sto ascoltando...";
    };
    recognition.onend = () => {
      micBtn.classList.remove("listening");
      statusEl.textContent = "Pronta, signore.";
    };
    recognition.onresult = (event) => {
      const transcript = event.results[0][0].transcript;
      addMessage("tu", transcript, { parla: false });
      kateReply(transcript, true);
    };
  }

  micBtn.addEventListener("click", () => {
    if (!recognition) {
      addMessage("kate", "Sul tuo browser il riconoscimento vocale non è disponibile,
signore.", { parla: false });
      return;
    }
    recognition.start();
  });

  // Allegati (solo selezione, niente controllo totale del telefono)
  attachBtn.addEventListener("click", () => {
    fileInput.click();
  });

  fileInput.addEventListener("change", () => {
    if (!fileInput.files.length) return;
```

```
    const names = Array.from(fileInput.files).map(f => f.name).join(", ");
    addMessage("tu", "Ho selezionato questi file: " + names, { parla: false });
    kateReply("Ho selezionato dei file: " + names);
  });

  // Ricerca
  searchToggle.addEventListener("click", () => {
    const visible = searchBar.style.display === "flex";
    searchBar.style.display = visible ? "none" : "flex";
    if (!visible) {
      searchInput.focus();
    } else {
      searchInput.value = "";
      searchCount.textContent = "";
      renderChat("");
    }
  });

  searchInput.addEventListener("input", () => {
    const term = searchInput.value.trim();
    if (!term) {
      searchCount.textContent = "";
      renderChat("");
      return;
    }
    const lower = term.toLowerCase();
    const matches = state.memoria.filter(m =>
m.testo.toLowerCase().includes(lower)).length;
    searchCount.textContent = matches ? matches + " risultati" : "Nessun risultato";
    renderChat(term);
  });

  // Context menu sui messaggi
  function showContextMenu(x, y, index) {
    if (contextMenuEl) {
      contextMenuEl.remove();
      contextMenuEl = null;
    }
    const menu = document.createElement("div");
    menu.className = "context-menu";
    menu.style.left = x + "px";
    menu.style.top = y + "px";

    const btnCopia = document.createElement("button");
    btnCopia.textContent = "Copia";
    btnCopia.onclick = () => {
      navigator.clipboard.writeText(state.memoria[index].testo || "").catch(() => {});
      menu.remove();
```

```
    };

    const btnCancella = document.createElement("button");
    btnCancella.textContent = "Cancella";
    btnCancella.onclick = () => {
      state.memoria.splice(index, 1);
      saveMemory();
      renderChat(searchInput.value);
      menu.remove();
    };

    const btnSeleziona = document.createElement("button");
    btnSeleziona.textContent = "Seleziona testo";
    btnSeleziona.onclick = () => {
      alert("Per ora puoi selezionare il testo tenendo premuto sul messaggio come fai di
solito.");
      menu.remove();
    };

    const btnInvia = document.createElement("button");
    btnInvia.textContent = "Invia";
    btnInvia.onclick = () => {
      inputEl.value = state.memoria[index].testo;
      inputEl.focus();
      menu.remove();
    };

    menu.appendChild(btnCopia);
    menu.appendChild(btnCancella);
    menu.appendChild(btnSeleziona);
    menu.appendChild(btnInvia);

    document.body.appendChild(menu);
    contextMenuEl = menu;
  }

  document.addEventListener("click", () => {
    if (contextMenuEl) {
      contextMenuEl.remove();
      contextMenuEl = null;
    }
  });

  // Inizializzazione
  loadMemory();
  renderChat("");

  if (state.memoria.length === 0) {
```

```
      addMessage("kate", "Eccomi, signore. Sono Kate. Puoi parlarmi o scrivermi, e posso adattarmi a come mi vuoi.", { parla: false });
    }
  </script>
</body>
</html>
```