```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>SupportAI - Intelligent Customer Support Chatbot</title>
  <style>
    :root {
      --primary-color: #4a6bff;
      --secondary-color: #f8f9fa;
      --accent-color: #ff6b6b;
      --text-color: #333;
      --light-text: #777;
      --chatbot-bg: #ffffff;
      --user-bubble: #4a6bff;
      --bot-bubble: #f0f0f0;
    }

    * {
      margin: 0;
      padding: 0;
      box-sizing: border-box;
      font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    }

    body {
      background-color: var(--secondary-color);
    color: var(--text-color);
    line-height: 1.6;
}

header {
    background-color: white;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
    position: fixed;
    width: 100%;
    top: 0;
    z-index: 1000;
}

.navbar {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 1rem 5%;
    max-width: 1400px;
    margin: 0 auto;
}

.logo {
    display: flex;
    align-items: center;
}
```

```
.logo img {
    height: 40px;
    margin-right: 10px;
}

.logo h1 {
    font-size: 1.5rem;
    color: var(--primary-color);
}

.nav-links {
    display: flex;
    list-style: none;
}

.nav-links li {
    margin-left: 2rem;
}

.nav-links a {
    text-decoration: none;
    color: var(--text-color);
    font-weight: 500;
    transition: color 0.3s;
}

.nav-links a:hover {
    color: var(--primary-color);
}

.cta-button {
    background-color: var(--primary-color);
    color: white;
    border: none;
    padding: 0.6rem 1.2rem;
    border-radius: 5px;
    font-weight: 600;
    cursor: pointer;
    transition: background-color 0.3s;
}

.cta-button:hover {
    background-color: #3a5bef;
}

.hero {
    padding: 8rem 5% 4rem;
    text-align: center;
    background: linear-gradient(135deg, #f5f7ff 0%, #ffffff 100%);
}

.hero h2 {
    font-size: 2.5rem;
    margin-bottom: 1rem;
```

```
    color: var(--primary-color);
}

.hero p {
    font-size: 1.2rem;
    color: var(--light-text);
    max-width: 700px;
    margin: 0 auto 2rem;
}

.features {
    padding: 4rem 5%;
    max-width: 1200px;
    margin: 0 auto;
}

.section-title {
    text-align: center;
    margin-bottom: 3rem;
}

.section-title h3 {
    font-size: 2rem;
    color: var(--primary-color);
    margin-bottom: 1rem;
}
```

```
.features-grid {
   display: grid;
   grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
   gap: 2rem;
}

.feature-card {
   background-color: white;
   border-radius: 10px;
   padding: 2rem;
   box-shadow: 0 5px 15px rgba(0, 0, 0, 0.05);
   transition: transform 0.3s, box-shadow 0.3s;
}

.feature-card:hover {
   transform: translateY(-5px);
   box-shadow: 0 10px 25px rgba(0, 0, 0, 0.1);
}

.feature-icon {
   font-size: 2.5rem;
   color: var(--primary-color);
   margin-bottom: 1.5rem;
}

.feature-card h4 {
   font-size: 1.3rem;
```

```
  margin-bottom: 1rem;
}

.demo {
  padding: 4rem 5%;
  background-color: white;
}

.demo-container {
  max-width: 1000px;
  margin: 0 auto;
  display: flex;
  flex-direction: column;
  align-items: center;
}

.chatbot-container {
  width: 100%;
  max-width: 500px;
  height: 500px;
  border-radius: 15px;
  box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);
  overflow: hidden;
  display: flex;
  flex-direction: column;
  background-color: var(--chatbot-bg);
  margin-top: 2rem;
```

```
}

.chatbot-header {
    background-color: var(--primary-color);
    color: white;
    padding: 1rem;
    text-align: center;
}

.chatbot-messages {
    flex: 1;
    padding: 1rem;
    overflow-y: auto;
    display: flex;
    flex-direction: column;
}

.message {
    max-width: 80%;
    margin-bottom: 1rem;
    padding: 0.8rem 1rem;
    border-radius: 15px;
    line-height: 1.4;
    position: relative;
}

.user-message {
    align-self: flex-end;
    background-color: var(--user-bubble);
    color: white;
    border-bottom-right-radius: 5px;
}

.bot-message {
    align-self: flex-start;
    background-color: var(--bot-bubble);
    border-bottom-left-radius: 5px;
}

.chatbot-input {
    display: flex;
    padding: 1rem;
    border-top: 1px solid #eee;
}

.chatbot-input input {
    flex: 1;
    padding: 0.8rem;
    border: 1px solid #ddd;
    border-radius: 25px;
    outline: none;
    font-size: 1rem;
}
```

```
.chatbot-input button {
   background-color: var(--primary-color);
   color: white;
   border: none;
   border-radius: 50%;
   width: 45px;
   height: 45px;
   margin-left: 0.5rem;
   cursor: pointer;
   transition: background-color 0.3s;
}

.chatbot-input button:hover {
   background-color: #3a5bef;
}

.typing-indicator {
   display: flex;
   padding: 0.8rem 1rem;
}

.typing-dot {
   width: 8px;
   height: 8px;
   background-color: #999;
   border-radius: 50%;
   margin: 0 2px;
```

```
  animation: typingAnimation 1.4s infinite ease-in-out;
}

.typing-dot:nth-child(1) {
  animation-delay: 0s;
}

.typing-dot:nth-child(2) {
  animation-delay: 0.2s;
}

.typing-dot:nth-child(3) {
  animation-delay: 0.4s;
}

@keyframes typingAnimation {
  0%, 60%, 100% { transform: translateY(0); }
  30% { transform: translateY(-5px); }
}

.suggested-actions {
  margin-top: 1rem;
  display: flex;
  flex-direction: column;
  gap: 0.5rem;
}
```

```
.suggested-action {
    background-color: white;
    border: 1px solid #ddd;
    border-radius: 8px;
    padding: 0.6rem;
    cursor: pointer;
    transition: all 0.2s;
    text-align: left;
}

.suggested-action:hover {
    background-color: var(--primary-color);
    color: white;
    border-color: var(--primary-color);
}

footer {
    background-color: #2a2a2a;
    color: white;
    padding: 3rem 5%;
    text-align: center;
}

.footer-content {
    max-width: 1200px;
    margin: 0 auto;
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: 2rem;
    text-align: left;
}

.footer-column h4 {
    margin-bottom: 1.5rem;
    font-size: 1.2rem;
}

.footer-column ul {
    list-style: none;
}

.footer-column ul li {
    margin-bottom: 0.8rem;
}

.footer-column ul li a {
    color: #bbb;
    text-decoration: none;
    transition: color 0.3s;
}

.footer-column ul li a:hover {
    color: white;
}
```

```
.copyright {
    margin-top: 3rem;
    padding-top: 1.5rem;
    border-top: 1px solid #444;
    color: #bbb;
    font-size: 0.9rem;
}

@media (max-width: 768px) {
    .navbar {
        flex-direction: column;
        padding: 1rem;
    }

    .nav-links {
        margin-top: 1rem;
    }

    .nav-links li {
        margin-left: 1rem;
        margin-right: 1rem;
    }

    .hero {
        padding-top: 6rem;
    }
```

```html
            .hero h2 {
                font-size: 2rem;
            }
        }
    </style>
</head>
<body>
    <header>
        <nav class="navbar">
            <div class="logo">
                <img src="https://via.placeholder.com/40" alt="SupportAI Logo">
                <h1>SupportAI</h1>
            </div>
            <ul class="nav-links">
                <li><a href="#features">Features</a></li>
                <li><a href="#demo">Demo</a></li>
                <li><a href="#pricing">Pricing</a></li>
                <li><a href="#contact">Contact</a></li>
            </ul>
            <button class="cta-button">Get Started</button>
        </nav>
    </header>

    <section class="hero">
        <h2>Revolutionizing Customer Support with AI</h2>
        <p>Our intelligent chatbot provides instant, accurate, and personalized assistance to your customers 24/7, reducing response times and increasing satisfaction.</p>
        <button class="cta-button">Try It Free</button>
    </section>

    <section class="features" id="features">
        <div class="section-title">
            <h3>Powerful Features</h3>
            <p>Everything you need to transform your customer support</p>
        </div>
        <div class="features-grid">
            <div class="feature-card">
                <div class="feature-icon">⚡</div>
                <h4>Instant Responses</h4>
                <p>Customers get answers immediately, without waiting in queues or during off-hours.</p>
            </div>
            <div class="feature-card">
                <div class="feature-icon">🧠</div>
                <h4>Machine Learning</h4>
                <p>Our AI learns from every interaction to provide better answers over time.</p>
            </div>
            <div class="feature-card">
                <div class="feature-icon">🔄</div>
                <h4>Seamless Handoff</h4>
                <p>When needed, smoothly transfer conversations to human agents with full context.</p>
            </div>
        <div class="feature-card">
            <div class="feature-icon">🌐</div>
            <h4>Multilingual Support</h4>
            <p>Communicate with customers in their preferred language automatically.</p>
        </div>
        <div class="feature-card">
            <div class="feature-icon">📊</div>
            <h4>Analytics Dashboard</h4>
            <p>Gain insights into customer queries, satisfaction, and chatbot performance.</p>
        </div>
        <div class="feature-card">
            <div class="feature-icon">🔌</div>
            <h4>Easy Integration</h4>
            <p>Connect with your existing CRM, helpdesk, and e-commerce platforms.</p>
        </div>
    </div>
</section>

<section class="demo" id="demo">
    <div class="demo-container">
        <div class="section-title">
            <h3>Try Our Chatbot</h3>
            <p>Experience the future of customer support firsthand</p>
        </div>
        <div class="chatbot-container">
            <div class="chatbot-header">
                <h4>SupportAI Assistant</h4>
```

```html
                </div>
                <div class="chatbot-messages" id="chatbot-messages">
                    <!-- Messages will appear here dynamically -->
                </div>
                <div class="chatbot-input">
                    <input type="text" id="user-input" placeholder="Type your question here..." onkeypress="handleKeyPress(event)">
                    <button onclick="sendMessage()">→</button>
                </div>
            </div>
        </div>
    </section>

    <footer id="contact">
        <div class="footer-content">
            <div class="footer-column">
                <h4>SupportAI</h4>
                <p>Transforming customer support through artificial intelligence and automation.</p>
            </div>
            <div class="footer-column">
                <h4>Product</h4>
                <ul>
                    <li><a href="#features">Features</a></li>
                    <li><a href="#demo">Demo</a></li>
                    <li><a href="#pricing">Pricing</a></li>
                    <li><a href="#">Integrations</a></li>
                </ul>
        </div>
        <div class="footer-column">
            <h4>Resources</h4>
            <ul>
                <li><a href="#">Documentation</a></li>
                <li><a href="#">API Reference</a></li>
                <li><a href="#">Blog</a></li>
                <li><a href="#">Case Studies</a></li>
            </ul>
        </div>
        <div class="footer-column">
            <h4>Company</h4>
            <ul>
                <li><a href="#">About Us</a></li>
                <li><a href="#">Careers</a></li>
                <li><a href="#">Contact</a></li>
                <li><a href="#">Privacy Policy</a></li>
            </ul>
        </div>
    </div>
    <div class="copyright">
        &copy; 2023 SupportAI. All rights reserved.
    </div>
</footer>

<script>
    // Enhanced Chatbot functionality with context awareness
```

```
const chatbot = {
  context: {},
  conversationHistory: [],

  responses: {
    greetings: [
      "Hello! I'm your AI support assistant. How can I help you today?",
      "Hi there! What can I do for you today?",
      "Welcome! How may I assist you?"
    ],
    account: {
      login: [
        "For login issues, please try resetting your password first. Would you like me to send a password reset link?",
        "Login problems can often be solved by clearing your browser cache. Have you tried that?",
        "I can help with login issues. Are you receiving any specific error message?"
      ],
      password: [
        "I can help you reset your password. Please confirm the email associated with your account.",
        "Password resets are quick. Would you like me to initiate the process now?",
        "For security, we'll need to verify your identity before resetting your password."
      ],
      general: [
        "I see you're having account issues. Could you be more specific about what's not working?",
        "Account problems can be frustrating. Let me help - what exactly is the issue?",
        "I can assist with various account issues. What do you need help with?"
      ]
    },
    billing: {
      payment: [
        "For payment issues, please check if your card has expired or has sufficient funds.",
        "I can look up your payment status. May I have your account email?",
        "Payment problems can often be resolved by trying a different payment method."
      ],
      refund: [
        "Our refund policy allows returns within 30 days. Is your purchase within this period?",
        "I can initiate a refund request for you. Please provide your order number.",
        "Refunds typically take 5-7 business days to process. Would you like to proceed?"
      ],
      general: [
        "I can help with billing questions. What specifically do you need to know?",
        "Billing inquiries are common. Are you asking about payments, invoices, or something else?",
        "Let me assist with your billing question. Could you provide more details?"
      ]
    },
    product: {
      setup: [
        "Product setup is usually straightforward. Which step are you having trouble with?",
        "I can guide you through the setup process. Have you installed the software yet?",
                    "Setup issues are common. What exactly isn't working as expected?"
                ],
                feature: [
                    "I'd be happy to explain that feature. Which specific function are you asking about?",
                    "Our products have many features. Could you narrow down which one you need help with?",
                    "Feature questions are my specialty! What would you like to know?"
                ],
                general: [
                    "I can help with product support. Which product are you using?",
                    "Product issues can vary. Could you describe the problem in more detail?",
                    "Let me assist with your product question. What's not working properly?"
                ]
            },
            farewell: [
                "You're welcome! Don't hesitate to return if you have more questions.",
                "Glad I could help! Have a wonderful day!",
                "Happy to assist! Come back anytime you need support."
            ],
            default: [
                "I'm not sure I understand. Could you rephrase that?",
                "I want to make sure I help correctly. Could you provide more details?",
                "Let me try to understand better. What exactly do you need help with?"
            ]
        },

        getRandomResponse: function(category, subcategory) {
```

```
        const responses = subcategory
            ? this.responses[category][subcategory]
            : this.responses[category];
        return responses[Math.floor(Math.random() * responses.length)];
    },

    determineContext: function(message) {
        // Reset context if it's a new conversation
        if (this.conversationHistory.length === 0) {
            this.context = {};
        }

        message = message.toLowerCase();

        // Detect greeting
        if (/(hi|hello|hey)/.test(message) && this.conversationHistory.length === 0) {
            return { type: 'greeting' };
        }

        // Detect farewell
        if (/(thanks|thank you|bye|goodbye)/.test(message)) {
            return { type: 'farewell' };
        }

        // Check for account-related queries
        if (/(account|login|sign in|password|credentials)/.test(message)) {
            if (/(login|sign in)/.test(message)) {
```

```
        return { type: 'account', subType: 'login' };
    }
    if (/(password|credentials)/.test(message)) {
        return { type: 'account', subType: 'password' };
    }
    return { type: 'account', subType: 'general' };
}

// Check for billing-related queries
if (/(bill|payment|invoice|charge|refund|purchase)/.test(message)) {
    if (/(payment|charge)/.test(message)) {
        return { type: 'billing', subType: 'payment' };
    }
    if (/(refund|return)/.test(message)) {
        return { type: 'billing', subType: 'refund' };
    }
    return { type: 'billing', subType: 'general' };
}

// Check for product-related queries
if (/(product|feature|setup|install|trouble|problem)/.test(message)) {
    if (/(setup|install)/.test(message)) {
        return { type: 'product', subType: 'setup' };
    }
    if (/(feature|function)/.test(message)) {
        return { type: 'product', subType: 'feature' };
    }
```

```
        return { type: 'product', subType: 'general' };
    }

    // Use previous context if available
    if (this.context.type) {
        return this.context;
    }

    return { type: 'default' };
},

generateResponse: function(context) {
    this.context = context; // Store current context

    switch(context.type) {
        case 'greeting':
            return {
                text: this.getRandomResponse('greetings'),
                actions: [
                    "Account issues",
                    "Billing questions",
                    "Product support"
                ]
            };
        case 'account':
            return {
                text: this.getRandomResponse('account', context.subType),
    actions: context.subType === 'general' ?
      ["Login problems", "Password reset", "Account settings"] :
      ["Yes, please help", "No, that didn't work", "I need something else"]
  };
case 'billing':
  return {
    text: this.getRandomResponse('billing', context.subType),
    actions: context.subType === 'general' ?
      ["Payment issues", "Refund request", "Invoice question"] :
      ["Yes, that's correct", "No, different issue", "More options"]
  };
case 'product':
  return {
    text: this.getRandomResponse('product', context.subType),
    actions: context.subType === 'general' ?
      ["Setup help", "Feature question", "Troubleshooting"] :
      ["Yes, please explain", "No, different issue", "Show me documentation"]
  };
case 'farewell':
  return {
    text: this.getRandomResponse('farewell'),
    actions: []
  };
default:
  return {
    text: this.getRandomResponse('default'),
    actions: [
                        "Account help",
                        "Billing question",
                        "Product support"
                    ]
                };
        }
    },

    processMessage: function(message) {
        this.conversationHistory.push({ sender: 'user', text: message });
        const context = this.determineContext(message);
        return this.generateResponse(context);
    }
};

// DOM manipulation functions
function sendMessage() {
    const userInput = document.getElementById('user-input');
    const message = userInput.value.trim();

    if (message) {
        addMessage(message, 'user');
        userInput.value = ";

        showTypingIndicator();

        // Simulate processing delay
        setTimeout(() => {
            removeTypingIndicator();
            const response = chatbot.processMessage(message);
            addMessage(response.text, 'bot');

            if (response.actions && response.actions.length > 0) {
                addSuggestedActions(response.actions);
            }
        }, 1000 + Math.random() * 1500);
    }
}

function sendQuickReply(action) {
    addMessage(action, 'user');

    showTypingIndicator();

    setTimeout(() => {
        removeTypingIndicator();
        const response = chatbot.processMessage(action);
        addMessage(response.text, 'bot');

        if (response.actions && response.actions.length > 0) {
            addSuggestedActions(response.actions);
        }
    }, 800 + Math.random() * 1200);
}
```

```
function addMessage(text, sender) {
    const messagesContainer = document.getElementById('chatbot-messages');
    const messageElement = document.createElement('div');
    messageElement.className = `message ${sender}-message`;
    messageElement.textContent = text;
    messagesContainer.appendChild(messageElement);
    messagesContainer.scrollTop = messagesContainer.scrollHeight;
}

function showTypingIndicator() {
    const messagesContainer = document.getElementById('chatbot-messages');
    const typingElement = document.createElement('div');
    typingElement.className = 'message bot-message typing-indicator';
    typingElement.id = 'typing-indicator';

    for (let i = 0; i < 3; i++) {
        const dot = document.createElement('div');
        dot.className = 'typing-dot';
        typingElement.appendChild(dot);
    }

    messagesContainer.appendChild(typingElement);
    messagesContainer.scrollTop = messagesContainer.scrollHeight;
}

function removeTypingIndicator() {
    const typingElement = document.getElementById('typing-indicator');
    if (typingElement) {
        typingElement.remove();
    }
}

function addSuggestedActions(actions) {
    const messagesContainer = document.getElementById('chatbot-messages');
    const actionsContainer = document.createElement('div');
    actionsContainer.className = 'suggested-actions';

    actions.forEach(action => {
        const actionElement = document.createElement('div');
        actionElement.className = 'suggested-action';
        actionElement.textContent = action;
        actionElement.onclick = function() { sendQuickReply(action); };
        actionsContainer.appendChild(actionElement);
    });

    messagesContainer.appendChild(actionsContainer);
    messagesContainer.scrollTop = messagesContainer.scrollHeight;
}

function handleKeyPress(event) {
    if (event.key === 'Enter') {
        sendMessage();
    }
```

```
        }

        // Initialize conversation
        window.onload = function() {
            // Start with a greeting after a short delay
            setTimeout(() => {
                const initialResponse = chatbot.processMessage("hello");
                addMessage(initialResponse.text, 'bot');
                addSuggestedActions(initialResponse.actions);
            }, 1000);
        };

        // Smooth scrolling for anchor links
        document.querySelectorAll('a[href^="#"]').forEach(anchor => {
            anchor.addEventListener('click', function (e) {
                e.preventDefault();
                document.querySelector(this.getAttribute('href')).scrollIntoView({
                    behavior: 'smooth'
                });
            });
        });
    </script>
</body>
</html>
```