<!DOCTYPE html>

<html lang="ja">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Bar Keike</title>

    <script type="module">

        // Firebaseの必要な機能をインポートします

        import { initializeApp } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-app.js";

        import { getAuth, signInAnonymously, signInWithCustomToken, onAuthStateChanged } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-auth.js";

        import { getFirestore, collection, addDoc, updateDoc, onSnapshot, doc } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-firestore.js";

        // ====================================================================================

        // !!! ここにあなたのFirebaseプロジェクトの設定情報を貼り付けてください !!!

        // Firebaseコンソールでウェブアプリを登録した際に表示された「firebaseConfig」オブジェクトを、

        // この「{ ... }」の中に、そっくりそのまま貼り付けてください。

        // 例: const firebaseConfig = { apiKey: "AIzaSy...", authDomain: "...", projectId: "...", ... };

        // ====================================================================================

        const firebaseConfig = {

apiKey: "AIzaSyCQC87L1mJF5m28NWYB11n07s9fQa0R04k",

authDomain: "bar-keike-d217b.firebaseapp.com",

projectId: "bar-keike-d217b",

storageBucket: "bar-keike-d217b.firebasestorage.app",

messagingSenderId: "213459603645",

appId: "1:213459603645:web:978ceca8b56c10e17167d4",

measurementId: "G-PG2Z1JZEYF"

};

        // ====================================================================================

        // !!! ここにあなたのFirebaseプロジェクトのIDを文字列として入力してください !!!

        // Firebaseコンソールで確認したprojectIdと同じものにしてください。

        // 例: const myAppId = "your-project-id";

        // ====================================================================================

        const myAppId = " bar-keike-d217b ";

        // アプリ全体で使うFirebase関連の変数を準備します

        window.firebaseApp = null;

        window.db = null;

        window.auth = null;

        window.userId = null; // 現在のユーザーID

        window.ordersCollectionRef = null; // 注文データを保存する場所

        window.orders = []; // データベースから読み込んだ注文を一時的に保存するリスト

        // カスタムメッセージボックスを表示する関数

        function showMessageBox(message) {

            const messageBoxText = document.getElementById('messageBoxText');

            const messageBoxOverlay = document.getElementById('messageBoxOverlay');

            messageBoxText.textContent = message;

            messageBoxOverlay.classList.add('visible'); // visibleクラスを追加して表示

        }

        // カスタムメッセージボックスを非表示にする関数

        function hideMessageBox() {

            const messageBoxOverlay = document.getElementById('messageBoxOverlay');

            messageBoxOverlay.classList.remove('visible'); // visibleクラスを削除して非表示

        }

        // Firebaseを初期化して、ユーザーを認証する関数

        async function initializeFirebase() {

            try {

                // Firebaseの設定が正しく入力されているか確認

                if (!firebaseConfig.apiKey || firebaseConfig.projectId === "YOUR\_FIREBASE\_PROJECT\_ID\_HERE") {

                    console.error("Firebaseの設定が正しくありません。プレースホルダーを置き換えてください。");

                    showMessageBox("エラー: Firebase設定が正しくありません。コード内の指示に従って設定してください。");

                    return; // 設定が不完全ならここで処理を中断

                }

                // Firebaseアプリを初期化

                window.firebaseApp = initializeApp(firebaseConfig);

                window.db = getFirestore(window.firebaseApp); // Firestoreデータベースのインスタンスを取得

                window.auth = getAuth(window.firebaseApp); // 認証サービスのインスタンスを取得

                // Firestoreの便利な関数をwindowオブジェクトに登録（コードの他の場所からアクセスしやすくするため）

                window.doc = doc;

                window.addDoc = addDoc;

                window.updateDoc = updateDoc;

                // ユーザーの認証状態を監視する（ログインしているか、匿名ユーザーかなど）

                onAuthStateChanged(window.auth, async (user) => {

                    if (user) {

                        // ユーザーがログインしていれば、そのIDを取得して表示

                        window.userId = user.uid;

                        document.getElementById('userIdDisplay').textContent = `ユーザーID: ${window.userId}`;

                        // 認証が完了したら、Firestoreから注文データをリアルタイムで取得する設定を開始

                        setupFirestoreListener();

                    } else {

                        // ユーザーがログインしていなければ、匿名でログインを試みる

                        try {

                            await signInAnonymously(window.auth);

                        } catch (anonError) {

                            console.error("匿名認証に失敗しました:", anonError);

                            showMessageBox(`エラー: 認証に失敗しました。${anonError.message}`);

                        }

                    }

                });

            } catch (error) {

                console.error("Firebaseの初期化または認証中にエラーが発生しました:", error);

                showMessageBox(`エラー: Firebaseの初期化に失敗しました。${error.message}`);

            }

        }

        // Firestoreから注文データをリアルタイムで取得するリスナーを設定する関数

        function setupFirestoreListener() {

            // 注文データを保存するコレクションの場所を指定（myAppIdを使用）

            window.ordersCollectionRef = collection(window.db, `artifacts/${myAppId}/public/data/orders`);

            // リアルタイムリスナーを設定：データベースの変更を常に監視し、変更があれば自動的にデータを更新

            onSnapshot(window.ordersCollectionRef, (snapshot) => {

                const fetchedOrders = [];

                snapshot.forEach(doc => {

                    const data = doc.data();

                    // FirestoreのタイムスタンプオブジェクトをJavaScriptのDateオブジェクトに変換

                    if (data.timestamp && typeof data.timestamp.toDate === 'function') {

                        data.timestamp = data.timestamp.toDate();

                    }

                    fetchedOrders.push({ id: doc.id, ...data }); // ドキュメントIDも一緒に保存

                });

                // ローカルの注文リストを最新のデータで更新

                window.orders = fetchedOrders;

                // もし現在キッチン画面が表示されていれば、表示を更新

                if (currentMode === 'kitchen') {

                    updateKitchenDisplay();

                }

            }, (error) => {

                console.error("Firestoreからのデータ取得中にエラーが発生しました:", error);

                showMessageBox(`エラー: 注文データの取得に失敗しました。${error.message}`);

            });

        }

        // ページが完全に読み込まれたらFirebaseの初期化を開始

        window.addEventListener('load', initializeFirebase);

    </script>

    <style>

        /\* 全体のリセットと基本スタイル \*/

        \* {

            margin: 0;

            padding: 0;

            box-sizing: border-box;

        }

        body {

            font-family: 'Hiragino Sans', 'Yu Gothic', sans-serif; /\* 日本語フォントを優先 \*/

            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%); /\* グラデーション背景 \*/

            min-height: 100vh; /\* 画面全体を覆う \*/

            color: #333; /\* 基本の文字色 \*/

        }

        /\* コンテナのスタイル \*/

        .container {

            max-width: 1200px; /\* 最大幅 \*/

            margin: 0 auto; /\* 中央寄せ \*/

            padding: 20px; /\* 内側の余白 \*/

        }

        /\* ヘッダーのスタイル \*/

        .header {

            text-align: center; /\* 中央寄せ \*/

            color: white; /\* 文字色 \*/

            margin-bottom: 30px; /\* 下部の余白 \*/

        }

        .header h1 {

            font-size: 2.5rem; /\* フォントサイズ \*/

            margin-bottom: 10px; /\* 下部の余白 \*/

            text-shadow: 2px 2px 4px rgba(0,0,0,0.3); /\* テキストシャドウ \*/

        }

        #userIdDisplay {

            font-size: 0.9rem;

            color: rgba(255, 255, 255, 0.8);

            margin-top: 5px;

        }

        /\* モード切り替えボタンのスタイル \*/

        .mode-toggle {

            display: flex; /\* Flexboxで配置 \*/

            justify-content: center; /\* 中央寄せ \*/

            gap: 20px; /\* ボタン間のスペース \*/

            margin-bottom: 30px; /\* 下部の余白 \*/

        }

        .mode-btn {

            padding: 12px 24px; /\* 内側の余白 \*/

            border: none; /\* ボーダーなし \*/

            border-radius: 25px; /\* 角丸 \*/

            font-size: 1.1rem; /\* フォントサイズ \*/

            cursor: pointer; /\* カーソルをポインターに \*/

            transition: all 0.3s ease; /\* ホバー時のアニメーション \*/

            font-weight: bold; /\* 太字 \*/

        }

        .mode-btn.active {

            background: #fff; /\* アクティブ時の背景色 \*/

            color: #667eea; /\* アクティブ時の文字色 \*/

            box-shadow: 0 4px 15px rgba(0,0,0,0.2); /\* シャドウ \*/

        }

        .mode-btn:not(.active) {

            background: rgba(255,255,255,0.2); /\* 非アクティブ時の背景色（透明度あり） \*/

            color: white; /\* 文字色 \*/

            border: 2px solid rgba(255,255,255,0.3); /\* ボーダー \*/

        }

        .mode-btn:hover {

            transform: translateY(-2px); /\* ホバー時に少し上に移動 \*/

        }

        /\* 各セクションのスタイル \*/

        .section {

            display: none; /\* デフォルトでは非表示 \*/

            background: rgba(255,255,255,0.95); /\* 背景色（透明度あり） \*/

            backdrop-filter: blur(10px); /\* 背景のぼかし効果 \*/

            border-radius: 20px; /\* 角丸 \*/

            padding: 30px; /\* 内側の余白 \*/

            box-shadow: 0 10px 30px rgba(0,0,0,0.2); /\* シャドウ \*/

        }

        .section.active {

            display: block; /\* アクティブ時に表示 \*/

        }

        /\* メニューカテゴリのスタイル \*/

        .menu-category {

            margin-bottom: 30px; /\* 下部の余白 \*/

        }

        .category-title {

            font-size: 1.5rem; /\* フォントサイズ \*/

            color: #667eea; /\* 文字色 \*/

            margin-bottom: 15px; /\* 下部の余白 \*/

            padding-bottom: 10px; /\* 下部のパディング \*/

            border-bottom: 2px solid #667eea; /\* 下線 \*/

            display: flex; /\* Flexboxで配置 \*/

            align-items: center; /\* 垂直方向中央寄せ \*/

            gap: 10px; /\* アイコンとテキストの間隔 \*/

        }

        .gem-icon {

            width: 24px; /\* 幅 \*/

            height: 24px; /\* 高さ \*/

            display: inline-block; /\* インラインブロック要素 \*/

        }

        /\* カクテルグリッドのスタイル \*/

        .cocktail-grid {

            display: grid; /\* グリッドレイアウト \*/

            grid-template-columns: repeat(auto-fit, minmax(300px, 1fr)); /\* レスポンシブなカラム設定 \*/

            gap: 20px; /\* グリッド間のスペース \*/

        }

        /\* カクテルカードのスタイル \*/

        .cocktail-card {

            background: white; /\* 背景色 \*/

            border-radius: 15px; /\* 角丸 \*/

            padding: 20px; /\* 内側の余白 \*/

            box-shadow: 0 5px 15px rgba(0,0,0,0.1); /\* シャドウ \*/

            transition: all 0.3s ease; /\* ホバー時のアニメーション \*/

            cursor: pointer; /\* カーソルをポインターに \*/

            border: 2px solid transparent; /\* デフォルトのボーダー \*/

        }

        .cocktail-card:hover {

            transform: translateY(-5px); /\* ホバー時に少し上に移動 \*/

            box-shadow: 0 10px 25px rgba(0,0,0,0.15); /\* シャドウ \*/

        }

        .cocktail-card.selected {

            border-color: #667eea; /\* 選択時のボーダー色 \*/

            background: #f8f9ff; /\* 選択時の背景色 \*/

        }

        .cocktail-name {

            font-size: 1.2rem; /\* フォントサイズ \*/

            font-weight: bold; /\* 太字 \*/

            color: #333; /\* 文字色 \*/

            margin-bottom: 8px; /\* 下部の余白 \*/

        }

        .cocktail-subtitle {

            font-size: 0.9rem; /\* フォントサイズ \*/

            color: #666; /\* 文字色 \*/

            font-style: italic; /\* 斜体 \*/

            margin-bottom: 10px; /\* 下部の余白 \*/

        }

        .cocktail-ingredients {

            font-size: 0.9rem; /\* フォントサイズ \*/

            color: #555; /\* 文字色 \*/

            margin-bottom: 10px; /\* 下部の余白 \*/

        }

        .cocktail-description {

            font-size: 0.85rem; /\* フォントサイズ \*/

            color: #777; /\* 文字色 \*/

            line-height: 1.4; /\* 行の高さ \*/

        }

        /\* 注文サマリーのスタイル \*/

        .order-summary {

            position: fixed; /\* 固定配置 \*/

            bottom: 20px; /\* 下からの距離 \*/

            right: 20px; /\* 右からの距離 \*/

            background: #667eea; /\* 背景色 \*/

            color: white; /\* 文字色 \*/

            padding: 20px; /\* 内側の余白 \*/

            border-radius: 15px; /\* 角丸 \*/

            box-shadow: 0 10px 30px rgba(0,0,0,0.3); /\* シャドウ \*/

            min-width: 250px; /\* 最小幅 \*/

            max-height: 400px; /\* 最大高さ \*/

            overflow-y: auto; /\* 縦スクロールを可能に \*/

            z-index: 1000; /\* 最前面に表示 \*/

        }

        .customer-name-input {

            width: 100%; /\* 幅を100%に \*/

            padding: 10px; /\* 内側の余白 \*/

            margin-bottom: 15px; /\* 下部の余白 \*/

            border: none; /\* ボーダーなし \*/

            border-radius: 8px; /\* 角丸 \*/

            font-size: 1rem; /\* フォントサイズ \*/

        }

        .customer-name-input::placeholder {

            color: #999; /\* プレースホルダーの色 \*/

        }

        .order-count {

            font-size: 1.1rem; /\* フォントサイズ \*/

            font-weight: bold; /\* 太字 \*/

            margin-bottom: 15px; /\* 下部の余白 \*/

        }

        .order-item {

            display: flex; /\* Flexboxで配置 \*/

            justify-content: space-between; /\* 要素を両端に寄せる \*/

            align-items: center; /\* 垂直方向中央寄せ \*/

            margin-bottom: 10px; /\* 下部の余白 \*/

            padding: 8px; /\* 内側の余白 \*/

            background: rgba(255,255,255,0.1); /\* 背景色（透明度あり） \*/

            border-radius: 8px; /\* 角丸 \*/

        }

        .order-item-name {

            flex: 1; /\* 残りのスペースを埋める \*/

            font-size: 0.9rem; /\* フォントサイズ \*/

        }

        .order-controls {

            display: flex; /\* Flexboxで配置 \*/

            gap: 5px; /\* ボタン間のスペース \*/

            align-items: center; /\* 垂直方向中央寄せ \*/

        }

        .qty-btn {

            background: rgba(255,255,255,0.2); /\* 背景色（透明度あり） \*/

            border: none; /\* ボーダーなし \*/

            color: white; /\* 文字色 \*/

            width: 25px; /\* 幅 \*/

            height: 25px; /\* 高さ \*/

            border-radius: 50%; /\* 円形 \*/

            cursor: pointer; /\* カーソルをポインターに \*/

            font-size: 0.9rem; /\* フォントサイズ \*/

        }

        .qty-btn:hover {

            background: rgba(255,255,255,0.3); /\* ホバー時の背景色 \*/

        }

        .qty-display {

            color: white; /\* 文字色 \*/

            font-weight: bold; /\* 太字 \*/

            min-width: 20px; /\* 最小幅 \*/

            text-align: center; /\* 中央寄せ \*/

        }

        .send-order-btn {

            width: 100%; /\* 幅を100%に \*/

            padding: 12px; /\* 内側の余白 \*/

            margin-top: 15px; /\* 上部の余白 \*/

            background: #fff; /\* 背景色 \*/

            color: #667eea; /\* 文字色 \*/

            border: none; /\* ボーダーなし \*/

            border-radius: 8px; /\* 角丸 \*/

            font-weight: bold; /\* 太字 \*/

            cursor: pointer; /\* カーソルをポインターに \*/

            font-size: 1rem; /\* フォントサイズ \*/

        }

        .send-order-btn:hover {

            background: #f0f0f0; /\* ホバー時の背景色 \*/

        }

        .send-order-btn:disabled {

            background: #ccc; /\* 無効時の背景色 \*/

            cursor: not-allowed; /\* カーソルを禁止マークに \*/

        }

        /\* キッチン画面の注文グリッド \*/

        .kitchen-orders {

            display: grid; /\* グリッドレイアウト \*/

            grid-template-columns: repeat(auto-fit, minmax(350px, 1fr)); /\* レスポンシブなカラム設定 \*/

            gap: 20px; /\* グリッド間のスペース \*/

        }

        /\* キッチン注文カードのスタイル \*/

        .order-card {

            background: white; /\* 背景色 \*/

            border-radius: 15px; /\* 角丸 \*/

            padding: 20px; /\* 内側の余白 \*/

            box-shadow: 0 5px 15px rgba(0,0,0,0.1); /\* シャドウ \*/

            border-left: 5px solid #667eea; /\* 左ボーダー \*/

        }

        .order-header {

            display: flex; /\* Flexboxで配置 \*/

            justify-content: space-between; /\* 要素を両端に寄せる \*/

            align-items: center; /\* 垂直方向中央寄せ \*/

            margin-bottom: 15px; /\* 下部の余白 \*/

        }

        .order-user-id { /\* 注文カード内のユーザーID表示用 \*/

            font-size: 0.75rem;

            color: #888;

            margin-top: 3px;

        }

        .order-customer {

            font-size: 1.1rem; /\* フォントサイズ \*/

            font-weight: bold; /\* 太字 \*/

            color: #667eea; /\* 文字色 \*/

            margin-bottom: 5px; /\* 下部の余白 \*/

        }

        .order-time {

            font-size: 0.9rem; /\* フォントサイズ \*/

            color: #666; /\* 文字色 \*/

        }

        .order-status {

            padding: 5px 10px; /\* 内側の余白 \*/

            border-radius: 15px; /\* 角丸 \*/

            font-size: 0.8rem; /\* フォントサイズ \*/

            font-weight: bold; /\* 太字 \*/

        }

        .status-pending {

            background: #fff3cd; /\* 準備中の背景色 \*/

            color: #856404; /\* 準備中の文字色 \*/

        }

        .status-completed {

            background: #d4edda; /\* 完了の背景色 \*/

            color: #155724; /\* 完了の文字色 \*/

        }

        .cocktail-checklist {

            list-style: none; /\* リストのマークを削除 \*/

        }

        .cocktail-checklist li {

            display: flex; /\* Flexboxで配置 \*/

            align-items: center; /\* 垂直方向中央寄せ \*/

            gap: 10px; /\* 要素間のスペース \*/

            padding: 8px 0; /\* 上下のパディング \*/

            border-bottom: 1px solid #eee; /\* 下線 \*/

        }

        /\* キッチン表示で完了したアイテムの視覚的なスタイル \*/

        .cocktail-checklist .item-completed-visual {

            text-decoration: line-through; /\* 打ち消し線 \*/

            opacity: 0.6; /\* 透明度 \*/

        }

        .cocktail-checklist input[type="checkbox"] {

            width: 18px; /\* 幅 \*/

            height: 18px; /\* 高さ \*/

            cursor: pointer; /\* カーソルをポインターに \*/

        }

        .complete-order-btn {

            width: 100%; /\* 幅を100%に \*/

            padding: 10px; /\* 内側の余白 \*/

            margin-top: 15px; /\* 上部の余白 \*/

            background: #28a745; /\* 背景色 \*/

            color: white; /\* 文字色 \*/

            border: none; /\* ボーダーなし \*/

            border-radius: 8px; /\* 角丸 \*/

            cursor: pointer; /\* カーソルをポインターに \*/

            font-weight: bold; /\* 太字 \*/

        }

        .complete-order-btn:hover {

            background: #218838; /\* ホバー時の背景色 \*/

        }

        .complete-order-btn:disabled {

            background: #ccc; /\* 無効時の背景色 \*/

            cursor: not-allowed; /\* カーソルを禁止マークに \*/

        }

        /\* 特別マークのスタイル \*/

        .special-mark {

            font-size: 0.8rem; /\* フォントサイズ \*/

            background: #ff6b6b; /\* 背景色 \*/

            color: white; /\* 文字色 \*/

            padding: 2px 6px; /\* 内側の余白 \*/

            border-radius: 4px; /\* 角丸 \*/

            margin-left: 5px; /\* 左側の余白 \*/

        }

        /\* シェイクマークのスタイル \*/

        .shake-mark {

            font-size: 0.8rem; /\* フォントサイズ \*/

            background: #28a745; /\* 背景色 \*/

            color: white; /\* 文字色 \*/

            padding: 2px 6px; /\* 内側の余白 \*/

            border-radius: 4px; /\* 角丸 \*/

            margin-left: 5px; /\* 左側の余白 \*/

        }

        /\* 注文がない場合のメッセージ \*/

        .no-orders {

            text-align: center; /\* 中央寄せ \*/

            color: #666; /\* 文字色 \*/

            font-style: italic; /\* 斜体 \*/

            padding: 40px; /\* 内側の余白 \*/

        }

        /\* カスタムメッセージボックスのスタイル \*/

        .message-box-overlay {

            position: fixed;

            top: 0;

            left: 0;

            width: 100%;

            height: 100%;

            background-color: rgba(0, 0, 0, 0.5);

            display: flex;

            justify-content: center;

            align-items: center;

            z-index: 2000;

            visibility: hidden; /\* 初期状態は非表示 \*/

            opacity: 0; /\* 初期状態は透明 \*/

            transition: opacity 0.3s ease, visibility 0s 0.3s;

        }

        .message-box-overlay.visible {

            visibility: visible;

            opacity: 1;

            transition: opacity 0.3s ease;

        }

        .message-box {

            background-color: white;

            padding: 30px;

            border-radius: 15px;

            box-shadow: 0 5px 20px rgba(0, 0, 0, 0.3);

            text-align: center;

            max-width: 400px;

            width: 90%;

            color: #333;

        }

        .message-box p {

            font-size: 1.1rem;

            margin-bottom: 20px;

            word-wrap: break-word; /\* 長いメッセージがはみ出ないように \*/

        }

        .message-box button {

            background-color: #667eea;

            color: white;

            border: none;

            padding: 10px 20px;

            border-radius: 8px;

            cursor: pointer;

            font-size: 1rem;

            font-weight: bold;

            transition: background-color 0.3s ease;

        }

        .message-box button:hover {

            background-color: #5a6cdb;

        }

        /\* レスポンシブデザイン \*/

        @media (max-width: 768px) {

            .container {

                padding: 10px; /\* パディングを減らす \*/

            }

            .cocktail-grid {

                grid-template-columns: 1fr; /\* 1カラム表示 \*/

            }

            .order-summary {

                position: static; /\* 固定配置を解除 \*/

                margin-top: 20px; /\* 上部の余白 \*/

                width: 100%; /\* 幅を100%に \*/

                max-height: none; /\* 最大高さを解除 \*/

            }

            .kitchen-orders {

                grid-template-columns: 1fr; /\* 1カラム表示 \*/

            }

        }

    </style>

</head>

<body>

    <div class="container">

        <div class="header">

            <h1>✨Bar Keike✨</h1>

            <p>お好みのカクテルをお選びください</p>

            <div id="userIdDisplay" style="margin-top: 10px;">ユーザーID: ロード中...</div>

        </div>

        <div class="mode-toggle">

            <button class="mode-btn active" data-mode="customer">🍹 お客様</button>

            <button class="mode-btn" data-mode="kitchen">👨‍🍳 キッチン</button>

        </div>

        <div id="customer-section" class="section active">

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">💎</span>

                    宝石カクテル ★アルコール★

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="サファイア・ナイト">

                        <div class="cocktail-name">サファイア・ナイト</div>

                        <div class="cocktail-subtitle">「澄んだ夜空の星々」</div>

                        <div class="cocktail-ingredients">ディタ×グレープフルーツ×ブルー×トニック</div>

                        <div class="cocktail-description">ライチとグレープフルーツでさわやかなカクテル。最初の一杯に。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ルビー・キス">

                        <div class="cocktail-name">ルビー・キス <span class="special-mark">特別</span><span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-subtitle">「甘酸っぱくも情熱的に」</div>

                        <div class="cocktail-ingredients">アマレット×レモン×グレナデン＋ソーダ</div>

                        <div class="cocktail-description">あんずのお酒アマレットの甘さとレモンの酸味が絶妙な一杯。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="トパーズ・サンセット">

                        <div class="cocktail-name">トパーズ・サンセット</div>

                        <div class="cocktail-subtitle">「甘く美しい夕焼け」</div>

                        <div class="cocktail-ingredients">ピーチ×オレンジ×パッション×グレナデン</div>

                        <div class="cocktail-description">ファジーネーブルをベースに、飲みやすく楽しい一杯。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="パール・ヌーン">

                        <div class="cocktail-name">パール・ヌーン</div>

                        <div class="cocktail-subtitle">「花の香りの昼下がり」</div>

                        <div class="cocktail-ingredients">サンジェルマン×アイスティー＋レモンスライス</div>

                        <div class="cocktail-description">お花の香りのお酒をアイスティーと。ケーキと合わせればそこはパリ。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="アメジスト・ドリーム">

                        <div class="cocktail-name">アメジスト・ドリーム</div>

                        <div class="cocktail-subtitle">「幻想的な夢の世界」</div>

                        <div class="cocktail-ingredients">ウォッカ×ブルー×グレナデン＋ジンジャー</div>

                        <div class="cocktail-description">モスコミュールをベースにきれいな紫色を。甘くごくごく飲める一杯。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="翡翠～ヒスイ～">

                        <div class="cocktail-name">翡翠～ヒスイ～</div>

                        <div class="cocktail-subtitle">「きらめく川のせせらぎ」</div>

                        <div class="cocktail-ingredients">ジン×メロン×レモンジュース×ソーダ</div>

                        <div class="cocktail-description">メロンの甘さとジンのハーブの香りでさっぱりとした一杯。</div>

                    </div>

                </div>

            </div>

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">🥤</span>

                    ノンアルコール

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="サファイア・ブルー">

                        <div class="cocktail-name">サファイア・ブルー</div>

                        <div class="cocktail-ingredients">ブルー×レモンジュース×ソーダ</div>

                        <div class="cocktail-description">さっぱりとした青いレモンソーダ。まずはここから！</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="トパーズ・オレンジ">

                        <div class="cocktail-name">トパーズ・オレンジ</div>

                        <div class="cocktail-ingredients">パッションフルーツ×オレンジジュース×グレナデン</div>

                        <div class="cocktail-description">パッションフルーツで、いつもとは違ったトロピカルなオレンジジュースを。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="アメジスト・パープル">

                        <div class="cocktail-name">アメジスト・パープル</div>

                        <div class="cocktail-ingredients">ブルー×グレナデン＋ジンジャー</div>

                        <div class="cocktail-description">紫色のジンジャーエール。料理と合うこと間違いなし！</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="エメラルド・グリーン">

                        <div class="cocktail-name">エメラルド・グリーン</div>

                        <div class="cocktail-ingredients">グリーンアップル×レモン×ソーダ</div>

                        <div class="cocktail-description">青春の甘酸っぱさのさわやかなアップルソーダ。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="パール・ホワイト">

                        <div class="cocktail-name">パール・ホワイト</div>

                        <div class="cocktail-ingredients">卵×牛乳×砂糖</div>

                        <div class="cocktail-description">みんな大好きミルク・セーキ！お酒を飲む人も飲まない人も、デザートの一杯に。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="シンデレラ">

                        <div class="cocktail-name">シンデレラ <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">パイン×オレンジ×レモンジュース</div>

                        <div class="cocktail-description">ノンアルコールカクテルの定番。バランスの取れたフルーティーな味わい。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ブルーハワイ">

                        <div class="cocktail-name">ブルーハワイ <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">ブルー×パイン×レモンジュース</div>

                        <div class="cocktail-description">トロピカルな青いカクテル。夏気分を味わえます。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="プッシーキャット">

                        <div class="cocktail-name">プッシーキャット <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">オレンジ×パイン×グレープフルーツ×グレナデン</div>

                        <div class="cocktail-description">フルーツたっぷりの優しい味わい。猫のように上品で可愛らしい一杯。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="サマーディライト">

                        <div class="cocktail-name">サマーディライト <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">グレナデン×ライム＋ソーダ</div>

                        <div class="cocktail-description">夏の喜びを表現したさっぱりとしたカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="サラトガクーラー">

                        <div class="cocktail-name">サラトガクーラー</div>

                        <div class="cocktail-ingredients">ジンジャーエール×ライム</div>

                        <div class="cocktail-description">シンプルで爽快な炭酸系カクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="シャーリーテンプル">

                        <div class="cocktail-name">シャーリーテンプル</div>

                        <div class="cocktail-ingredients">グレナデン×ジンジャーエール</div>

                        <div class="cocktail-description">子役スターの名前を冠した甘くて美味しいカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ミモザ">

                        <div class="cocktail-name">ミモザ</div>

                        <div class="cocktail-ingredients">ノンアル白ワイン×オレンジ</div>

                        <div class="cocktail-description">ブランチにぴったりの上品なカクテル。</div>

                    </div>

                </div>

            </div>

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">🌿</span>

                    GIN（ハーブやスパイスでボタニカルな香りのお酒）

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="ジントニック">

                        <div class="cocktail-name">ジントニック</div>

                        <div class="cocktail-ingredients">ジン×トニック＋ライム</div>

                        <div class="cocktail-description">定番中の定番。爽やかで飲みやすいカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ジンバック">

                        <div class="cocktail-name">ジンバック</div>

                        <div class="cocktail-ingredients">ジン×ジンジャー</div>

                        <div class="cocktail-description">ジンジャーエールの辛さとジンの香りがマッチ。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="オレンジ・ブロッサム">

                        <div class="cocktail-name">オレンジ・ブロッサム</div>

                        <div class="cocktail-ingredients">ジン×オレンジ</div>

                        <div class="cocktail-description">オレンジの花の名前を持つフルーティーなカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ギムレット">

                        <div class="cocktail-name">ギムレット <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">ジン×ライム</div>

                        <div class="cocktail-description">シンプルながら奥深い、ジンとライムの完璧なバランス。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ジン・デイジー">

                        <div class="cocktail-name">ジン・デイジー <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">ジン×レモン×グレナデン</div>

                        <div class="cocktail-description">クラッシュアイスで仕上げる爽やかなジンカクテル。</div>

                    </div>

                </div>

            </div>

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">❄️</span>

                    VODKA（クセのないカクテルの定番）

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="モスコミュール">

                        <div class="cocktail-name">モスコミュール</div>

                        <div class="cocktail-ingredients">ウォッカ×ライム×ジンジャー</div>

                        <div class="cocktail-description">銅のマグカップで飲むのが伝統的。爽やかな定番カクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="スクリュードライバー">

                        <div class="cocktail-name">スクリュードライバー</div>

                        <div class="cocktail-ingredients">オレンジ×ウォッカ</div>

                        <div class="cocktail-description">シンプルで飲みやすい。朝食にも合うカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ソルティドッグ">

                        <div class="cocktail-name">ソルティドッグ</div>

                        <div class="cocktail-ingredients">ウォッカ×グレープフルーツ+塩</div>

                        <div class="cocktail-description">グラスの縁の塩がアクセント。さっぱりとした味わい。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="レディ・デイ">

                        <div class="cocktail-name">レディ・デイ <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">ウォッカ×グレープフルーツ×ライチ</div>

                        <div class="cocktail-description">女性に人気の華やかで上品なカクテル。</div>

                    </div>

                </div>

            </div>

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">🍑</span>

                    DITA（ライチがさわやかなお酒）

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="チャイナブルー">

                        <div class="cocktail-name">チャイナブルー</div>

                        <div class="cocktail-ingredients">ディタ×グレープフルーツ×ブルー×トニック</div>

                        <div class="cocktail-description">美しい青色のライチカクテル。見た目も味も楽しめます。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ディタ・エイジア">

                        <div class="cocktail-name">ディタ・エイジア</div>

                        <div class="cocktail-ingredients">ディタ×レモン×ウーロン茶</div>

                        <div class="cocktail-description">アジアンテイストのさっぱりカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="楊貴妃">

                        <div class="cocktail-name">楊貴妃 <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">サンジェルマン×グレープフルーツ×ディタ×ブルー×レモン</div>

                        <div class="cocktail-description">美しい絶世の美人のような、エレガントで複雑な味わい。</div>

                    </div>

                </div>

            </div>

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">🌰</span>

                    AMARETTO（あんずの甘いお酒）

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="アマレットジンジャー">

                        <div class="cocktail-name">アマレットジンジャー</div>

                        <div class="cocktail-ingredients">アマレット×ジンジャー</div>

                        <div class="cocktail-description">甘いアマレットとスパイシーなジンジャーの組み合わせ。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="アマレットコーク">

                        <div class="cocktail-name">アマレットコーク</div>

                        <div class="cocktail-ingredients">アマレット×コーラ</div>

                        <div class="cocktail-description">甘いアマレットとコーラの親しみやすい組み合わせ。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ボッチボール">

                        <div class="cocktail-name">ボッチボール</div>

                        <div class="cocktail-ingredients">アマレット×オレンジ×ソーダ</div>

                        <div class="cocktail-description">イタリアの球技の名前を持つ、バランスの良いカクテル。</div>

                    </div>

                </div>

            </div>

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">🍑</span>

                    PEACH（とにかく桃！）

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="ファジーネーブル">

                        <div class="cocktail-name">ファジーネーブル</div>

                        <div class="cocktail-ingredients">ピーチ×オレンジ</div>

                        <div class="cocktail-description">ピーチとオレンジの定番の組み合わせ。優しい甘さが人気。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="レゲエパンチ">

                        <div class="cocktail-name">レゲエパンチ</div>

                        <div class="cocktail-ingredients">ピーチ×烏龍茶</div>

                        <div class="cocktail-description">意外な組み合わせが美味しい、さっぱりとしたカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="ガルフストリーム">

                        <div class="cocktail-name">ガルフストリーム</div>

                        <div class="cocktail-ingredients">ウォッカ×ピーチ×グレフル×パイン×ブルー</div>

                        <div class="cocktail-description">南国の海流をイメージした、トロピカルなカクテル。</div>

                    </div>

                </div>

            </div>

             <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">🌸</span>

                    St-Germain（食べられる花の上品なリキュール）

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="サンジェルマントニック">

                        <div class="cocktail-name">サンジェルマントニック</div>

                        <div class="cocktail-ingredients">St-Germain×トニック＋オレンジの皮</div>

                        <div class="cocktail-description">エルダーフラワーの香りが上品な大人のカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="サンジェルマンスプリッツァ">

                        <div class="cocktail-name">サンジェルマンスプリッツァ</div>

                        <div class="cocktail-ingredients">St-Germain×スパークリングワイン×ソーダ＋オレンジの皮</div>

                        <div class="cocktail-description">華やかでお祝いにぴったりのスパークリングカクテル。</div>

                    </div>

                </div>

            </div>

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">🍈</span>

                    Melon（メロン！！）

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="ミッドナイトメロン">

                        <div class="cocktail-name">ミッドナイトメロン</div>

                        <div class="cocktail-ingredients">メロン×St-Germain×トニック＋オレンジの皮</div>

                        <div class="cocktail-description">夜更けに楽しむ大人のメロンカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="メロンミルク">

                        <div class="cocktail-name">メロンミルク</div>

                        <div class="cocktail-ingredients">メロン×牛乳</div>

                        <div class="cocktail-description">懐かしい味のクリーミーなメロンカクテル。</div>

                    </div>

                </div>

            </div>

            <div class="menu-category">

                <h2 class="category-title">

                    <span class="gem-icon">🌿</span>

                    CAMPARI（薬草を使った苦みのあるリキュール）

                </h2>

                <div class="cocktail-grid">

                    <div class="cocktail-card" data-cocktail-name="スプモーニ">

                        <div class="cocktail-name">スプモーニ</div>

                        <div class="cocktail-ingredients">カンパリ×グレフル×トニック</div>

                        <div class="cocktail-description">イタリアの定番カクテル。苦味と甘みのバランスが絶妙。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="サウス・ビーチ">

                        <div class="cocktail-name">サウス・ビーチ <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">カンパリ×アマレット×オレンジ×シロップ</div>

                        <div class="cocktail-description">南国ビーチをイメージした、複雑で深みのある味わい。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="カンパリシェカラート">

                        <div class="cocktail-name">カンパリシェカラート <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">カンパリ</div>

                        <div class="cocktail-description">カンパリそのものの味を楽しむストレートカクテル。</div>

                    </div>

                    <div class="cocktail-card" data-cocktail-name="カンパリクラシック">

                        <div class="cocktail-name">カンパリクラシック <span class="shake-mark">シェイク</span></div>

                        <div class="cocktail-ingredients">カンパリ×ソーダ</div>

                        <div class="cocktail-description">シェイクしてからソーダを加える、クラシックなスタイル。</div>

                    </div>

                </div>

            </div>

        </div>

        <div id="kitchen-section" class="section">

            <div id="kitchen-orders" class="kitchen-orders">

                <div class="no-orders">

                    現在、注文はありません

                </div>

            </div>

        </div>

        <div class="order-summary">

            <input type="text" class="customer-name-input" placeholder="お名前を入力してください" id="customerNameInput">

            <div class="order-count" id="orderCount">注文: 0品</div>

            <div id="orderItemsContainer"></div> <button class="send-order-btn" id="sendOrderBtn" disabled>注文を送信</button>

        </div>

    </div>

    <div id="messageBoxOverlay" class="message-box-overlay">

        <div class="message-box">

            <p id="messageBoxText"></p>

            <button onclick="hideMessageBox()">OK</button>

        </div>

    </div>

    <script>

        // グローバル変数

        let currentMode = 'customer'; // 現在の表示モード

        let selectedCocktails = {}; // 選択されたカクテルとその数量

        // let orders = []; // ローカルの注文リストはFirebaseから取得するため不要になります

        // let orderIdCounter = 1; // FirebaseがIDを自動生成するため不要になります

        // DOM要素のキャッシュ（頻繁にアクセスする要素は一度取得して変数に格納）

        const customerSection = document.getElementById('customer-section');

        const kitchenSection = document.getElementById('kitchen-section');

        const kitchenOrdersDiv = document.getElementById('kitchen-orders');

        const customerNameInput = document.getElementById('customerNameInput');

        const orderCountDiv = document.getElementById('orderCount');

        const orderItemsContainer = document.getElementById('orderItemsContainer');

        const sendOrderBtn = document.getElementById('sendOrderBtn');

        const modeToggleButtons = document.querySelectorAll('.mode-btn');

        const cocktailCards = document.querySelectorAll('.cocktail-card');

        const allSections = document.querySelectorAll('.section');

        const messageBoxOverlay = document.getElementById('messageBoxOverlay');

        const messageBoxText = document.getElementById('messageBoxText');

        /\*\*

         \* カスタムメッセージボックスを表示する関数

         \* @param {string} message - 表示したいメッセージ

         \*/

        function showMessageBox(message) {

            messageBoxText.textContent = message;

            messageBoxOverlay.classList.add('visible'); // visibleクラスを追加して表示

        }

        /\*\*

         \* カスタムメッセージボックスを非表示にする関数

         \*/

        function hideMessageBox() {

            messageBoxOverlay.classList.remove('visible'); // visibleクラスを削除して非表示

        }

        /\*\*

         \* 表示モードを切り替える関数

         \* @param {string} mode - 'customer' または 'kitchen'

         \* @param {HTMLElement} targetButton - クリックされたモードボタン要素

         \*/

        function switchMode(mode, targetButton) {

            currentMode = mode;

            // 全てのモードボタンのアクティブ状態を解除

            modeToggleButtons.forEach(btn => btn.classList.remove('active'));

            // クリックされたボタンをアクティブにする

            if (targetButton) {

                 targetButton.classList.add('active');

            }

            // 全てのセクションを非表示にする

            allSections.forEach(section => section.classList.remove('active'));

            // 選択されたモードのセクションを表示する

            if (mode === 'customer') {

                customerSection.classList.add('active');

            } else {

                kitchenSection.classList.add('active');

                // Firebaseからデータがロードされていれば表示更新

                if (window.orders) {

                    updateKitchenDisplay();

                } else {

                    // Firebaseの初期化が完了していない場合はメッセージを表示

                    if (!window.db) {

                        showMessageBox("データをロード中です。しばらくお待ちください。");

                    }

                }

            }

        }

        /\*\*

         \* カクテルを選択/選択解除（数量を増やす）する関数

         \* @param {string} name - カクテル名

         \*/

        function toggleCocktail(name) {

            // 既に選択されていれば数量を増やす、そうでなければ1に設定

            if (selectedCocktails[name]) {

                selectedCocktails[name]++;

            } else {

                selectedCocktails[name] = 1;

            }

            updateOrderSummary();

            updateCocktailCardSelection();

        }

        /\*\*

         \* カクテルカードの選択状態（selectedクラス）を更新する関数

         \*/

        function updateCocktailCardSelection() {

            cocktailCards.forEach(card => {

                const cocktailName = card.dataset.cocktailName;

                if (selectedCocktails[cocktailName]) {

                    card.classList.add('selected');

                } else {

                    card.classList.remove('selected');

                }

            });

        }

        /\*\*

         \* 注文サマリーの表示を更新する関数

         \*/

        function updateOrderSummary() {

            let totalItems = 0;

            let html = '';

            // selectedCocktailsオブジェクトをループして注文アイテムを生成

            for (const [name, count] of Object.entries(selectedCocktails)) {

                if (count > 0) {

                    totalItems += count;

                    html += `

                        <div class="order-item" data-item-name="${name}">

                            <div class="order-item-name">${name}</div>

                            <div class="order-controls">

                                <button class="qty-btn decrease-qty">-</button>

                                <span class="qty-display">${count}</span>

                                <button class="qty-btn increase-qty">+</button>

                            </div>

                        </div>

                    `;

                }

            }

            orderItemsContainer.innerHTML = html;

            orderCountDiv.textContent = `注文: ${totalItems}品`;

            const customerName = customerNameInput.value.trim();

            // Firebaseが初期化されていなければ送信ボタンを無効にする

            sendOrderBtn.disabled = totalItems === 0 || customerName === '' || !window.db;

        }

        // 注文サマリー内の数量変更ボタンに対するイベント委譲

        orderItemsContainer.addEventListener('click', function(event) {

            const target = event.target;

            // クリックされた要素が.qty-btnクラスを持っているかチェック

            if (target.classList.contains('qty-btn')) {

                const orderItemDiv = target.closest('.order-item');

                if (!orderItemDiv) return;

                const itemName = orderItemDiv.dataset.itemName;

                if (target.classList.contains('increase-qty')) {

                    changeQuantity(itemName, 1);

                } else if (target.classList.contains('decrease-qty')) {

                    changeQuantity(itemName, -1);

                }

            }

        });

        /\*\*

         \* 注文アイテムの数量を変更する関数

         \* @param {string} name - カクテル名

         \* @param {number} delta - 変更量（+1 または -1）

         \*/

        function changeQuantity(name, delta) {

            if (selectedCocktails[name]) {

                selectedCocktails[name] += delta;

                if (selectedCocktails[name] <= 0) {

                    delete selectedCocktails[name];

                }

            }

            updateOrderSummary();

            updateCocktailCardSelection();

        }

        /\*\*

         \* 注文を送信する関数

         \*/

        async function sendOrder() {

            // Firebaseが準備できていない場合はエラーメッセージを表示

            if (!window.db || !window.ordersCollectionRef || !window.userId) {

                showMessageBox("データベースが準備できていません。しばらくお待ちください。");

                return;

            }

            const customerName = customerNameInput.value.trim();

            if (!customerName || Object.keys(selectedCocktails).length === 0) {

                showMessageBox("お名前とカクテルを選択してください。");

                return;

            }

            // データベースに保存する注文データを作成

            const orderData = {

                customer: customerName,

                items: {...selectedCocktails},

                timestamp: new Date(),

                status: 'pending',

                completedItems: {},

                userId: window.userId

            };

            try {

                // Firestoreに注文を追加

                await window.addDoc(window.ordersCollectionRef, orderData);

                showMessageBox('注文を送信しました！');

            } catch (error) {

                console.error("Firestoreへの注文追加中にエラーが発生しました:", error);

                showMessageBox(`エラー: 注文の送信に失敗しました。${error.message}`);

            }

            selectedCocktails = {};

            customerNameInput.value = '';

            updateOrderSummary();

            updateCocktailCardSelection();

        }

        /\*\*

         \* キッチン画面の注文表示を更新する関数

         \*/

        function updateKitchenDisplay() {

            if (!window.orders) return; // ordersがまだロードされていなければ何もしない

            // 未完了の注文を優先し、その後完了した注文を古い順にソート

            const sortedOrders = [...window.orders].sort((a, b) => {

                if (a.status === 'pending' && b.status === 'completed') return -1;

                if (a.status === 'completed' && b.status === 'pending') return 1;

                const timeA = a.timestamp instanceof Date ? a.timestamp.getTime() : 0;

                const timeB = b.timestamp instanceof Date ? b.timestamp.getTime() : 0;

                return timeA - timeB;

            });

            if (sortedOrders.length === 0) {

                kitchenOrdersDiv.innerHTML = '<div class="no-orders">現在、注文はありません</div>';

                return;

            }

            let html = '';

            sortedOrders.forEach(order => {

                const isOrderCompleted = order.status === 'completed';

                const allIndividualItemsChecked = Object.keys(order.items).every(itemName => (order.completedItems || {})[itemName]);

                html += `

                    <div class="order-card" data-order-id="${order.id}">

                        <div class="order-header">

                            <div>

                                <div class="order-customer">${order.customer}様</div>

                                <div class="order-time">${order.timestamp instanceof Date ? order.timestamp.toLocaleTimeString('ja-JP') : '時刻不明'}</div>

                                <div class="order-user-id">注文者ID: ${order.userId ? order.userId.slice(0, 8) + '...' : '不明'}</div>

                            </div>

                            <div class="order-status ${isOrderCompleted ? 'status-completed' : 'status-pending'}">

                                ${isOrderCompleted ? '完了' : '準備中'}

                            </div>

                        </div>

                        <ul class="cocktail-checklist">

                `;

                for (const [itemName, count] of Object.entries(order.items)) {

                    const isItemChecked = (order.completedItems || {})[itemName] || false;

                    const cocktailCardElement = findCocktailCardElementByName(itemName);

                    const hasShake = cocktailCardElement && cocktailCardElement.querySelector('.shake-mark');

                    const hasSpecial = cocktailCardElement && cocktailCardElement.querySelector('.special-mark');

                    html += `

                        <li>

                            <input type="checkbox"

                                   data-item-name="${itemName}"

                                   ${isItemChecked ? 'checked' : ''}

                                   ${isOrderCompleted ? 'disabled' : ''}>

                            <span class="${isItemChecked ? 'item-completed-visual' : ''}">

                                ${itemName} × ${count}

                                ${hasSpecial ? '<span class="special-mark">特別</span>' : ''}

                                ${hasShake ? '<span class="shake-mark">シェイク</span>' : ''}

                            </span>

                        </li>

                    `;

                }

                html += `

                        </ul>

                        <button class="complete-order-btn"

                                ${!allIndividualItemsChecked || isOrderCompleted ? 'disabled' : ''}>

                            ${isOrderCompleted ? '完了済み' : '注文完了'}

                        </button>

                    </div>

                `;

            });

            kitchenOrdersDiv.innerHTML = html;

        }

        // キッチン注文リスト内のインタラクション（チェックボックス、完了ボタン）に対するイベント委譲

        kitchenOrdersDiv.addEventListener('click', async function(event){

            const target = event.target;

            const orderCard = target.closest('.order-card');

            if (!orderCard) return;

            const orderId = orderCard.dataset.orderId;

            if (target.matches('input[type="checkbox"]')) {

                const itemName = target.dataset.itemName;

                await toggleItemCompletion(orderId, itemName, target.checked);

            } else if (target.classList.contains('complete-order-btn')) {

                await completeOrder(orderId);

            }

        });

        function findCocktailCardElementByName(cocktailName) {

            for (const card of cocktailCards) {

                if (card.dataset.cocktailName === cocktailName) {

                    return card;

                }

            }

            return null;

        }

        async function toggleItemCompletion(orderId, itemName, isChecked) {

            if (!window.db || !window.ordersCollectionRef) {

                showMessageBox("データベースが準備できていません。");

                return;

            }

            const order = window.orders.find(o => o.id === orderId);

            if (order && order.status !== 'completed') {

                const updatedCompletedItems = { ...(order.completedItems || {}) };

                updatedCompletedItems[itemName] = isChecked;

                try {

                    const orderDocRef = window.doc(window.db, `artifacts/${myAppId}/public/data/orders/${orderId}`);

                    await window.updateDoc(orderDocRef, { completedItems: updatedCompletedItems });

                } catch (error) {

                    console.error("Firestoreでのアイテム完了状態の更新中にエラーが発生しました:", error);

                    showMessageBox(`エラー: アイテム状態の更新に失敗しました。${error.message}`);

                }

            }

        }

        async function completeOrder(orderId) {

            if (!window.db || !window.ordersCollectionRef) {

                showMessageBox("データベースが準備できていません。");

                return;

            }

            const order = window.orders.find(o => o.id === orderId);

            if (order) {

                const allItemsTrulyChecked = Object.keys(order.items).every(itemName => (order.completedItems || {})[itemName]);

                if (allItemsTrulyChecked) {

                    try {

                        const orderDocRef = window.doc(window.db, `artifacts/${myAppId}/public/data/orders/${orderId}`);

                        await window.updateDoc(orderDocRef, { status: 'completed' });

                        showMessageBox('注文が完了しました！');

                    } catch (error) {

                        console.error("Firestoreでの注文完了状態の更新中にエラーが発生しました:", error);

                        showMessageBox(`エラー: 注文の完了に失敗しました。${error.message}`);

                    }

                } else {

                    showMessageBox('全ての品目がチェックされていません。');

                }

            }

        }

        modeToggleButtons.forEach(btn => {

            btn.addEventListener('click', (event) => {

                switchMode(event.currentTarget.dataset.mode, event.currentTarget);

            });

        });

        cocktailCards.forEach(card => {

            card.addEventListener('click', () => {

                toggleCocktail(card.dataset.cocktailName);

            });

        });

        customerNameInput.addEventListener('input', updateOrderSummary);

        sendOrderBtn.addEventListener('click', sendOrder);

        updateOrderSummary();

        const initialActiveButton = document.querySelector('.mode-btn.active');

        if (initialActiveButton) {

             switchMode(initialActiveButton.dataset.mode, initialActiveButton);

        }

    </script>

</body>

</html>