Contents

1. Introduction

This sample document outlines the threat model for the Brew & Bloom Coffee Co. ordering web application. It systematically identifies security objectives, critical assets, the application's profile and decomposition, and potential threats and vulnerabilities.

2. Security Objectives

- **Confidentiality**: Safeguard customer account details, including personal information and payment data, from unauthorized access or disclosure.
- **Integrity**: Prevent unauthorized modification of critical application data, including product catalog information (especially prices), customer orders, and user account details.
- **Availability**: Ensure the Brew & Bloom Coffee Co. Ordering Application maintains an availability of 99.99% for both customers placing orders and staff managing operations.

3. Assets

3.1. Customer Personally Identifiable Information (PII)

3.1.1. Names
3.1.2. Contact details (email, phone, delivery address)
3.1.3. Order history

3.2. Payment Information

3.2.1. Tokenized payment data
3.2.2. Transaction records
3.2.3. Other associated financial details

3.3. Order Data

3.3.1. Details of placed orders (including items, quantities, customizations, prices, status, and fulfillment information)

3.4. Menu Data

3.4.1. Information about available coffee items, pastries, beverages, descriptions, prices, and availability

3.5. Staff Credentials

3.5.1. Authentication details (usernames, passwords, session tokens) for Baristas, Managers, and Administrators

3.6. Application Code and Configuration

3.6.1. Source code
3.6.2. Server configurations
3.6.3. Environment variables
3.6.4. API keys for third-party services

3.7. Database Integrity and Availability

3.7.1. The underlying data store, ensuring accuracy, consistency, and continuous access to all stored information

3.8. Brew & Bloom Coffee Co. Revenue

3.8.1. Protection against financial losses due to fraudulent orders, unauthorized refunds, or system downtime

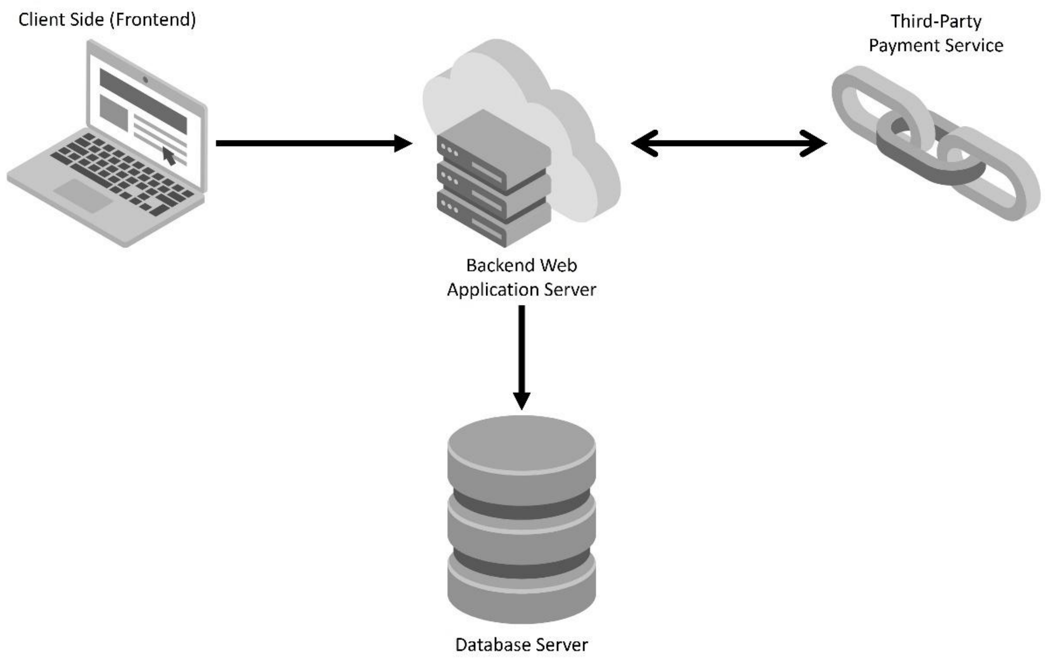
3.9. System Uptime and Performance

3.9.1. The continuous operation and responsiveness of the web application for a seamless user experience

4. Application Profile

4.1. Architecture Design

The Brew & Bloom Coffee Co. ordering web application is a digital platform designed to enhance the customer experience and streamline internal operations of the coffee shop.



The figure shown above illustrates the system architecture of the ordering application designed to provide a convenient online for customers to browse the menu, customize, and place orders for pickup or delivery, and to equip staff with tools for efficient order and menu management. The architecture is composed of four key components: the Client Side (Frontend), the Backend Web Application Server, the Database Server, and an integrated Third-Party Payment Service.

- Client Side (Frontend)
    - This layer represents the user interface built using HTML5, CSS3, and JavaScript, leveraging frameworks like React, Vue, or Angular
    - It supports customer-facing features such as:
        - User registration and login
        - Menu browsing with item details
        - Order customization and cart functionality

        - Secure order placement, order history, and status tracking
    - It also supports staff-facing features, including:
        - Secure logins for admins and staff
        - Real-time order visibility and updates
        - Menu item management (add, edit, delete)
        - Basic sales and operational reporting
- Backend Web Application Server
    - This component uses Node.js to implement RESTful APIs. It serves as the core logic layer, receiving requests from the frontend and interacting with the database and external services
    - Responsibilities include authenticating users, processing orders, managing business logic, and coordinating real-time updates
- Database Server
    - MongoDB is used for persistent data storage.
    - It holds structured information such as:
        - User profiles
        - Menu items
        - Order records
        - Staff roles and permissions
- Third-Party Payment Service
    - This module handles integration with an external payment gateway
    - It supports secure transactions via:
        - Credit/Debit Cards
        - E-wallets
        - Online Bank Transfers
    - The backend communicates securely with this service to process and verify customer payments

4.2. Roles and Permissions

The application supports distinct user roles, each with specific permissions tailored to their responsibilities. The following matrix details the access control for various functionalities across these roles:

Functionality / Role	**Customer**	**Barista**	**Manager**	**Administrator**
Account Management				
Register Own Account	C	-	-	-
Manage Own Profile	U	-	-	-
Login to Interface	X	X	X	X
Create Staff Account	-	-	-	C
Modify Staff Account	-	-	-	U
Delete Staff Account	-	-	-	D
Menu & Ordering				
Browse Menu	R	R	R	R
View Item Details	R	R	R	R
Add/Remove Cart Items	U	-	-	-
Customize Order	U	-	-	-
Place New Order	C	-	-	-
View Own Order History	R	-	-	-

Functionality / Role	Customer	Barista	Manager	Administrator
View Own Order Status	R			
View All Incoming Orders	-	R	R	R
Update Order Status	-	U	U	U
Cancel/Refund Order	-	-	U	U
Add New Menu Item	-	-	-	C
Modify Existing Menu Item	-	-	U	U
Delete Menu Item	-	-	-	D
Reporting & Monitoring				
Access Basic Sales Reports	-	-	R	R
Access System Reports	-	-	-	R
Monitor System Status	-	-	-	R
System Configuration				
Manage App Settings	-	-	-	U
Manage Third-Party Integrations	-	-	-	U

Legend:

- **C:** Create
- **R:** Read (View)
- **U:** Update (Modify)
- **D:** Delete
- **X:** Execute (e.g., log in, process payment)
- **-:** No Access

5. Application Decomposition

The Brew & Bloom Coffee Co. Ordering Application can be decomposed into the following key components and their interactions, forming the basis for identifying trust boundaries and data flows.

5.1. Trust Boundaries

- Client side is untrusted, therefore, all input originating from here must be validated on the server.
- The web application server trusts the client for basic UI interactions but not for data integrity or authorization decisions
- The MongoDB database server trusts calls from the web application server's identity
- Third-party services are external and partially trusted; communication channels are secured, but data received from them is validated.

5.2. Data Flows

1. Customer Places an Order
   - 1.1. Customer selects items, customizes them, and submits the order details to the Web Application Server.
   - 1.2. Web Application Server receives the order details and performs input validation and business logic checks (e.g., item availability, valid customizations).
   - 1.3. Web Application Server initiates a payment request, sending tokenized payment information to the Payment Gateway (a Third-Party Service).
   - 1.4. Payment Gateway processes the payment and sends a transaction response (success/failure) back to the Web Application Server.

    1.5. If payment is successful, the Web Application Server records the new order details and payment status in the MongoDB Database.
    1.6. MongoDB Database stores the new order record.
    1.7. Web Application Server sends an order confirmation and status update back to the Customer.
    1.8. Customer displays the order confirmation and updated order status.
2. Barista Updates Order Status
    2.1. Barista logs into the staff interface and requests to view incoming orders from the Web Application Server.
    2.2. Web Application Server authenticates and authorizes the Barista, then queries the MongoDB Database for relevant order data.
    2.3. MongoDB Database retrieves the pending order data and sends it to the Web Application Server.
    2.4. Web Application Server sends the order list to the Barista for display.
    2.5. Barista selects an order and updates its status (e.g., from "Received" to "Preparing").
    2.6. Barista sends the updated order status to the Web Application Server.
    2.7. Web Application Server validates the status update and updates the order record in the MongoDB Database.
3. Manager Accesses Basic Sales Reports
    3.1. Manager logs into the staff interface and requests to view basic sales reports from the Web Application Server.
    3.2. Web Application Server authenticates and authorizes the Manager, then queries the MongoDB Database for aggregated sales data.
    3.3. MongoDB Database retrieves the requested sales data and sends it back to the Web Application Server.
    3.4. Web Application Server processes the sales data (e.g., calculates totals, formats for display) and sends it as a response to the Manager.
    3.5. Manager displays the basic sales report on their web browser.
4. Administrator Adds New Menu Item
    4.1. Administrator logs into the staff interface and navigates to the menu management section.
    4.2. Administrator inputs details for a new menu item (name, description, price, etc.) and submits the information to the Web Application Server.
    4.3. Web Application Server receives the new menu item data, performs input validation, and authorizes the Administrator's request.
    4.4. Web Application Server sends the new menu item data to the MongoDB Database for creation.
    4.5. MongoDB Database creates the new menu item record.
    4.6. Web Application Server sends a confirmation of the successful menu item addition back to the Administrator.
    4.7. Administrator receives confirmation of the new menu item being added.

6. Threats

This section details potential threats to the Brew & Bloom Coffee Co. ordering application, categorized by the STRIDE model, and maps them to the components that the threats pose as a risk to.

Threat ID	STRIDE Category	Threat Title	Threat Description	Affected Components
T-001	Spoofing	Customer Account Spoofing	An attacker could impersonate a legitimate customer to place fraudulent orders or access their order history.	• Frontend • Web Server Backend • Database
T-002	Spoofing	Staff Account Spoofing	An attacker could impersonate staff to manipulate order statuses, access sales reports, or modify menu items and application configurations.	• Frontend • Web Server Backend • Database
T-003	Spoofing	Web Application Server Spoofing	An attacker might set up a fraudulent web server to phish user credentials or intercept sensitive data.	• Frontend • Web Server Backend
T-004	Tampering	Order Data Tampering	An attacker could modify order details (e.g., items, quantities, prices, status) in transit or at rest.	• Web Server Backend • Database
T-005	Tampering	Menu Data Tampering	An attacker could alter menu item data (e.g., prices or descriptions) before it is stored in the database.	• Web Server Backend • Database
T-006	Tampering	Sales Report Tampering	Unauthorized individuals could alter sales data in the MongoDB Database, leading to falsified reports.	• Web Server Backend • Database
T-007	Tampering	Application Code/Configuration Tampering	Malicious modification of the application's source code, server configurations, or environment variables could introduce backdoors or disable security features.	• Web Server Backend
T-008	Repudiation	Fraudulent Order Repudiation	A customer could deny placing an order if insufficient logging and non-repudiation controls are in place.	• Web Server Backend • Database
T-009	Repudiation	Staff Action Repudiation	Staff members (Barista, Manager, Administrator) could deny performing actions if audit logs are insufficient, not immutable, or can be tampered with.	• Web Server Backend • Database
T-010	Information Disclosure	Customer PII Disclosure	Unauthorized access to customer names, contact details, delivery addresses, and order history.	• Web Server Backend • Database
T-011	Information Disclosure	Staff Credential Disclosure	Exposure of usernames, passwords, or session tokens for staff members due to insecure storage or unencrypted transmission.	• Web Server Backend • Database
T-012	Information Disclosure	Application Code and Configuration Disclosure	Exposure of source code, server configurations, environment variables, or API keys.	• Web Server Backend
T-013	Information Disclosure	Sales Data Disclosure	Unauthorized access to sales reports or underlying aggregated sales data.	• Web Server Backend • Database
T-014	Denial of Service (DoS)	Web Application Server DoS	An attacker could flood the Web Application Server with excessive requests, making it unavailable.	• Web Server Backend
T-015	Denial of Service (DoS)	MongoDB Database DoS	Malicious or poorly optimized queries could overload the database, making it unresponsive.	• Database • Web Server Backend
T-016	Elevation of Privilege	Customer to Staff Privilege Escalation	A customer could exploit a vulnerability to gain unauthorized access to Barista, Manager, or Administrator functionalities.	• Web Server Backend
T-017	Elevation of Privilege	Barista/Manager to Administrator Privilege Escalation	A Barista or Manager could exploit a vulnerability to gain Administrator privileges.	• Web Server Backend
T-018	Elevation of Privilege	Horizontal Privilege Escalation	Gaining unauthorized access to another user's account within the same privilege level.	• Web Server Backend

7. Vulnerabilities

This section maps the identified threats to known Common Weakness Enumerations (CWEs).

Threat ID	CWE-20	CWE-22	CWE-77	CWE-78	CWE-79	CWE-89	CWE-94	CWE-200	CWE-269	CWE-287	CWE-306	CWE-352	CWE-400	CWE-434	CWE-502	CWE-798	CWE-862	CWE-863	CWE-918
T-001					✓	✓				✓	✓						✓		
T-002					✓	✓				✓	✓						✓		
T-003		✓	✓	✓	✓		✓												
T-004	✓				✓	✓			✓			✓					✓	✓	
T-005	✓					✓			✓					✓			✓	✓	
T-006	✓					✓			✓								✓	✓	
T-007		✓	✓	✓			✓							✓	✓				
T-008	✓					✓			✓								✓	✓	
T-009	✓					✓			✓								✓	✓	
T-010	✓	✓			✓	✓		✓	✓							✓	✓	✓	✓
T-011	✓	✓			✓	✓		✓	✓							✓	✓	✓	✓
T-012		✓	✓	✓				✓								✓			✓
T-013					✓	✓		✓	✓								✓	✓	✓
T-014	✓		✓	✓		✓	✓						✓	✓					
T-015						✓							✓						
T-016	✓				✓	✓			✓	✓	✓	✓			✓		✓	✓	
T-017	✓				✓	✓			✓	✓	✓	✓			✓		✓	✓	
T-018	✓				✓	✓			✓								✓	✓	

Links to the individual CWE pages:

- CWE-20: Improper Input Validation
- CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
- CWE-77: Improper Neutralization of Special Elements used in a Command ('Command Injection')
- CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
- CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
- CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
- CWE-94: Improper Control of Generation of Code ('Code Injection')
- CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
- CWE-269: Improper Privilege Management
- CWE-287: Improper Authentication
- CWE-306: Missing Authentication for Critical Function
- CWE-352: Cross-Site Request Forgery (CSRF)
- CWE-400: Uncontrolled Resource Consumption
- CWE-434: Unrestricted Upload of File with Dangerous Type
- CWE-502: Deserialization of Untrusted Data
- CWE-798: Use of Hard-coded Credentials
- CWE-862: Missing Authorization
- CWE-863: Incorrect Authorization
- CWE-918: Server-Side Request Forgery (SSRF)