Security Audit Report

Nutrition Tracker Application

Audit Date: December 22, 2024
Audit Type: White-Box Security Audit
Framework: FastAPI + Jinja2 + SQLAlchemy

Executive Summary

This security audit identified 16 significant vulnerabilities across the Nutrition Tracker application. The most critical findings relate to broken session management, missing CSRF protection, file upload vulnerabilities, and insecure access control patterns. The application uses bcrypt for password hashing (which is good), but has several other security weaknesses that require immediate attention.

Vulnerability Summary

ID	Type	Severity	Location	Route
		CRITICAL		
		CRITICAL		
3	Missing CSRF Protection	HIGH	admin_food_form.html:10	/admin/foods/*
4	Missing CSRF Protection	HIGH	admin_user_form.html:10,39	/admin/users/*
5	File Upload - No Validation	HIGH	home_router.py:435-436	/home/profile/upload_avatar
6	File Upload - No Validation	HIGH	camera_router.py:56-59	/camera/result
7	Hardcoded Secret Key	HIGH	main.py:39	Configuration
8	Weak Password Policy	MEDIUM	auth_service.py:176-177	/home/profile/change_password
9	XSS - DOM Manipulation	MEDIUM	your_meal.html:209	/camera/result
10	XSS - Reflected	MEDIUM	your_meal.html:222	/camera/result
11	Information Disclosure	MEDIUM	auth_router.py:47-50	/account/login
12	Potential IDOR	MEDIUM	home_router.py:310-315	/home/meals/detail
13	Missing Cookie Security	MEDIUM	auth_router.py:44-45	/account/login
14	Supabase Keys Exposure	MEDIUM	auth_router.py:206-209	/api/supabase/config
15	Mass Assignment Risk	LOW	admin_service.py:51-52	/admin/users/update
16	Debug Information	LOW	camera_router.py:73-76	/camera/result

Critical Issues - Detailed Analysis

1. Broken Session Management (CRITICAL)

Location: auth_router.py:44-45 and deps.py:22-23

The session token is a static string "user-is-logged-in" and user_id is stored directly in an unsigned cookie. This means ANY ATTACKER can impersonate any user by simply setting:
- Cookie: session_token=user-is-logged-in
- Cookie: user_id=1 (or any other user ID)

Impact: Complete authentication bypass. Attacker can access any user account.

Remediation:
- Use cryptographically secure session tokens (JWT or signed cookies)
- Implement server-side session storage with secure random token IDs
- Never trust user-controlled data for authentication decisions

2. Missing CSRF Protection (HIGH)

Location: All HTML forms in templates/

No CSRF tokens found in any forms. All state-changing operations (POST/PUT/DELETE) are vulnerable.
Affected routes include:
- /admin/foods/create, /admin/foods/update, /admin/foods/delete
- /admin/users/update, /admin/users/reset-password
- /home/diary/add, /home/meals/create, /home/meals/update
- /home/profile/change_password, /home/profile/upload_avatar

Impact: Attacker can craft malicious pages that submit forms on behalf of authenticated users.

Remediation:
- Implement CSRF tokens using FastAPI-CSRF or similar
- Add CSRF token to all forms

3. File Upload Vulnerabilities (HIGH)

Location: home_router.py:435-436 and camera_router.py:56-59

File uploads accept any file type without server-side validation.
Missing checks:
- File extension validation
- MIME type verification
- Magic number (file signature) check
- File size limits

- Content scanning

Impact: Potential malware upload, storage abuse, or attacks via malicious files.

Remediation:
- Validate file extension against whitelist (.jpg, .png, .gif)
- Check MIME type
- Verify magic bytes
- Set file size limits

4. Hardcoded Secret Key Fallback (HIGH)

Location: main.py:39

Default secret key "your-secret-key-here-change-in-production" is used if SECRET_KEY environment variable is missing. This predictable value breaks session security.

Remediation:
- Remove fallback value
- Fail fast if SECRET_KEY is not set
- Use os.getenv("SECRET_KEY") and raise error if None

Medium Severity Issues

5. XSS Vulnerabilities

Location: your_meal.html:209 and :222

Food search results are rendered using innerHTML with unescaped data. If food names contain malicious scripts, they will execute.

Remediation: Use textContent instead of innerHTML, or properly escape HTML entities.

6. Weak Password Policy

Location: auth_service.py:176-177

Minimum password length is only 6 characters with no complexity requirements.

Remediation: Enforce minimum 8-12 characters with complexity requirements.

7. Missing Cookie Security Flags

Location: auth_router.py:44-45

Cookies are set without HttpOnly, Secure, or SameSite attributes.

Remediation: Add httponly=True, secure=True, samesite="Lax" to all sensitive cookies.

Positive Findings

Password Hashing

The application uses bcrypt for password hashing, which is a strong algorithm.
Location: auth_service.py:13-14

SQL Injection Prevention

The application uses SQLAlchemy ORM with parameterized queries. No SQL injection vulnerabilities were found.

Priority Remediation Order

1. CRITICAL - Fix Session Management (IDs 1-2)
   This is a complete authentication bypass. Implement proper JWT or signed session tokens.

2. HIGH - Add CSRF Protection (IDs 3-4)
   Add CSRF middleware and tokens to all forms.

3. HIGH - Secure File Uploads (IDs 5-6)
   Add file type validation before upload.

4. HIGH - Remove Hardcoded Secret (ID 7)
   Fail application startup if SECRET_KEY not set.

5. MEDIUM - Fix XSS Issues (IDs 9-10)
   Escape all user input in JavaScript templates.

6. MEDIUM - Secure Cookie Configuration (ID 13)
   Add HttpOnly, Secure, SameSite flags.

7. MEDIUM - Strengthen Password Policy (ID 8)
   Increase minimum length and add complexity requirements.

Conclusion

The Nutrition Tracker application has critical security vulnerabilities that must be addressed before production deployment. The session management system is fundamentally broken and allows complete authentication bypass. Combined with missing CSRF protection, an attacker could:

1. Forge any user session
2. Access admin panel by guessing/enumerating admin user IDs
3. Modify user data, food entries, and passwords
4. Upload malicious files

Recommended Action: Address the Critical and High severity issues immediately before any production use.

Report generated as part of Information Security course audit assignment.

December 22, 2024