**INFORME DE ASEGURAMIENTO DE LA CALIDAD**

**Nombre del aplicativo:** Future Cash (FC)  
**Versión:** 1.0.0  
**Fecha de elaboración:** 31/03/2025  
**Desarrollador(es) / Equipo de QA:** Cristhian Alejandro Barrientos Moya  
**Empresa / Proyecto:** Future Cash Digital

**ÍNDICE**

1. Introducción
2. Información General
3. Estrategia de Pruebas
4. Casos de Prueba
5. Resultados y Evaluación
6. Conclusiones
7. Anexos (si aplica)

**1. INTRODUCCIÓN**

**Objetivo del Documento**

Este informe documenta el proceso de aseguramiento de la calidad del aplicativo **Future Cash (FC)**, una plataforma web diseñada para ofrecer presupuestos inteligentes y gestión financiera. Se describen los criterios, metodologías y herramientas utilizadas para garantizar un sistema funcional, seguro y centrado en la experiencia del usuario. Se realizan pruebas exhaustivas de funcionalidad, seguridad y rendimiento.

**Alcance**

El informe evalúa los siguientes módulos y funcionalidades clave:

* Registro y autenticación de usuarios (login y signup).
* Dashboard interactivo con navegación lateral (sidenav).
* Panel de usuario con configuración personalizable.
* APIs backend para login, signup y obtención de datos del usuario.
* Medidas de seguridad contra ataques como inyecciones SQL, XSS y DDoS.

**Ambientes evaluados:**

* **Desarrollo:** localhost.
* **Staging:** [A definir].
* **Producción:** [A definir].

**Descripción General del Sistema**

**Future Cash** es una aplicación web que permite a los usuarios gestionar sus finanzas personales mediante presupuestos inteligentes. Ofrece registro e inicio de sesión, un dashboard dinámico, gráficas de tendencias financieras y recomendaciones personalizadas. El sistema usa **Node.js y Express.js** en el backend, **MySQL** como base de datos y un frontend basado en **HTML5, CSS3 y JavaScript**, con animaciones gestionadas por **GSAP** y notificaciones mediante **SweetAlert2**.

**2. INFORMACIÓN GENERAL**

**Plataforma y Tecnologías**

* **Plataforma:** Web.
* **Frontend:** HTML5, CSS3, JavaScript (GSAP, SweetAlert2, Chart.js).
* **Backend:** Node.js, Express.js.
* **Base de Datos:** MySQL.
* **Otros:** Font Awesome (íconos), JWT para autenticación.

**Ambientes de Prueba**

* **Desarrollo:** localhost:1000.
* **Staging:** [A definir].
* **Producción:** [A definir].

**Documentación Técnica y de Usuario**

* Diagramas de arquitectura del sistema (adjuntos en anexos).
* Manual de usuario (en desarrollo).
* Documentación de APIs con Postman.

**3. ESTRATEGIA DE PRUEBAS**

**Tipos de Pruebas Realizadas**

* **Pruebas Funcionales:** Validación del correcto funcionamiento del registro, login, navegación y visualización de datos.
* **Pruebas de Seguridad:** Detección de vulnerabilidades como inyecciones SQL, XSS y ataques de fuerza bruta.
* **Pruebas de Rendimiento:** Evaluación del comportamiento del sistema bajo carga y estrés.
* **Pruebas de Usabilidad:** Evaluación de la experiencia del usuario.
* **Pruebas de Regresión:** Asegurar que nuevas implementaciones no afecten funcionalidades existentes.

**Criterios de Aceptación**

* **100%** de aprobación en pruebas funcionales críticas.
* **Ausencia de vulnerabilidades de seguridad de alto riesgo** (OWASP Top 10).
* **Tiempo de respuesta < 2 segundos** con 100 usuarios concurrentes.
* **Cumplimiento de estándares de accesibilidad (WCAG 2.1 nivel AA).**

**Herramientas Utilizadas**

* **Pruebas Funcionales:** Selenium WebDriver.
* **Pruebas de Rendimiento:** JMeter.
* **Análisis de Código:** Fortify.
* **Pruebas de Seguridad:** OWASP ZAP, Burp Suite, SQLMap.
* **Monitoreo:** ELK Stack (Elasticsearch, Logstash, Kibana).

**4. CASOS DE PRUEBA**

**Ejemplo de Casos de Prueba**

**TC001 - Registro de Usuario**

* **Precondiciones:** El usuario no debe estar registrado.
* **Pasos:** Acceder a la página de registro, completar los datos y enviar el formulario.
* **Resultado Esperado:** Registro exitoso y redirección al dashboard.

**TC002 - Inicio de Sesión**

* **Precondiciones:** Usuario registrado y activo.
* **Pasos:** Ingresar credenciales y hacer clic en "Ingresar".
* **Resultado Esperado:** Acceso exitoso al sistema.

**TC003 - Prueba de Seguridad (SQL Injection)**

* **Precondiciones:** Acceso a la página de login.
* **Pasos:** Ingresar ' OR 1=1 -- como contraseña e intentar acceder.
* **Resultado Esperado:** Acceso denegado.

**5. RESULTADOS Y EVALUACIÓN**

**Resumen de Ejecución**

* **Total de Pruebas Ejecutadas:** [A completar].
* **Aprobadas:** [A completar].
* **Fallidas:** [A completar].

**Errores y Defectos**

* **Defecto 001:** Inyección SQL detectada en login.
* **Defecto 002:** Tiempo de respuesta superior a 2s bajo carga.

**6. CONCLUSIONES**

**Hallazgos y Recomendaciones**

* Mejorar validaciones de entrada en el backend.
* Optimizar consultas SQL para mejorar tiempos de respuesta.
* Implementar autenticación de sesiones con JWT más segura.

**7. ANEXOS**

* Diagramas de arquitectura.
* Reportes de seguridad y rendimiento.
* Logs de pruebas automatizadas.

**NOTA:** Este informe es una plantilla profesional basada en el proyecto **Future Cash**. Los campos entre corchetes [ ] deben completarse tras ejecutar las pruebas.