**MANUAL TÉCNICO DE FUTURE CASH (FC)**

**Versión del documento:** 1.0  
**Fecha de elaboración:** 31/03/2025  
**Autor del Proyecto:** Cristhian Alejandro Barrientos Moya **Desarrollador:** Cristhian Alejandro Barrientos Moya  
**Empresa / Proyecto:** Future Cash Digital

**ÍNDICE**

1. Introducción  
   1.1 Propósito del Documento  
   1.2 Alcance del Sistema  
   1.3 Definiciones y Acrónimos
2. Descripción General del Sistema  
   2.1 Contexto y Objetivos  
   2.2 Funcionalidades Clave  
   2.3 Usuarios y Roles
3. Metodología de Programación  
   3.1 Enfoque de Desarrollo  
   3.2 Herramientas y Tecnologías  
   3.3 Control de Versiones y Colaboración
4. Arquitectura del Sistema  
   4.1 Visión General de la Arquitectura  
   4.2 Componentes del Frontend  
   4.3 Componentes del Backend  
   4.4 Base de Datos  
   4.5 Integraciones y APIs
5. Detalles Técnicos  
   5.1 Flujo de Datos y Procesos  
   5.2 Seguridad  
   5.3 Rendimiento y Escalabilidad
6. Instalación y Despliegue  
   6.1 Requisitos del Sistema  
   6.2 Pasos de Instalación  
   6.3 Configuración Inicial
7. Mantenimiento y Soporte  
   7.1 Monitoreo y Logs  
   7.2 Actualizaciones y Parches  
   7.3 Soporte Técnico
8. Conclusiones
9. Anexos  
   9.1 Diagramas de Arquitectura  
   9.2 Glosario  
   9.3 Referencias

**1. INTRODUCCIÓN**

**1.1 Propósito del Documento**

Este manual técnico describe de manera detallada la arquitectura, diseño y funcionamiento del sistema **Future Cash (FC)**, una plataforma web diseñada para la gestión inteligente de presupuestos y finanzas personales. Está dirigido a desarrolladores, arquitectos de software y administradores de sistemas que necesiten comprender, implementar o mantener el sistema.

**1.2 Alcance del Sistema**

**Future Cash** es una herramienta que permite a los usuarios:

* Registrarse e iniciar sesión de manera segura.
* Acceder a un dashboard interactivo con navegación lateral.
* Visualizar gráficos de tendencias financieras.
* Recibir recomendaciones personalizadas basadas en su comportamiento financiero.
* Actualizar su información personal desde un panel de usuario.

El sistema está diseñado para ser intuitivo, seguro y escalable, con planes futuros para “integrar servicios financieros externos”.

**1.3 Definiciones y Acrónimos**

* **FC:** Future Cash.
* **API:** Application Programming Interface (Interfaz de Programación de Aplicaciones).
* **UI:** User Interface (Interfaz de Usuario).
* **UX:** User Experience (Experiencia de Usuario).
* **SQL:** Structured Query Language (Lenguaje de Consulta Estructurado).
* **HTTPS:** HyperText Transfer Protocol Secure (Protocolo Seguro de Transferencia de Hipertexto).

**2. DESCRIPCIÓN GENERAL DEL SISTEMA**

**2.1 Contexto y Objetivos**

**Future Cash** fue creado para brindar a los usuarios una solución práctica y eficiente para gestionar sus finanzas personales. Su objetivo principal es mejorar el bienestar económico mediante la generación de presupuestos inteligentes y análisis financieros personalizados, todo desde una interfaz web accesible y fácil de usar.

**2.2 Funcionalidades Clave**

* **Registro y Autenticación:** Formularios seguros para la creación de cuentas (signup) y el inicio de sesión (login).
* **Dashboard Interactivo:** Navegación lateral (sidenav) con acceso a secciones principales.
* **Panel de Usuario:** Gestión de datos personales y configuraciones.
* **Visualización de Datos:** Gráficos interactivos para tendencias financieras.
* **APIs de Gestión:** Endpoints para la interacción con los datos del usuario.

**2.3 Usuarios y Roles**

* **Usuario Registrado:** Accede a todas las funcionalidades financieras y personaliza su alcance y privilegios.
* **Administrador:** (En desarrollo) Gestiona usuarios y configuraciones globales del sistema.

**3. METODOLOGÍA DE PROGRAMACIÓN**

**3.1 Enfoque de Desarrollo**

El desarrollo de **Future Cash** se llevó a cabo utilizando una metodología ágil, con ciclos de desarrollo (sprints) de dos semanas. Se priorizó un diseño "Mobile-First" para garantizar que la plataforma sea completamente responsiva en dispositivos móviles y de escritorio, integración que actualmente se encuentra en proceso según sea la demanda de nuevos objetos añadidos.

**3.2 Herramientas y Tecnologías**

* **Frontend:** HTML5, CSS3, JavaScript (con librerías como GSAP para animaciones, SweetAlert2 para notificaciones, Font Awesome para los iconos y Chart.js para gráficos estáticos sin cambios, con proyecciones a uso de analíticas para la creación de los gráficos dinámicos).
* **Backend:** Node.js con el framework Express.js para la creación de APIs RESTful.
* **Base de Datos:** MySQL como motor principal para el almacenamiento de datos.
* **Control de Versiones:** Git, gestionado a través de GitHub.
* **Entorno de Desarrollo:** Visual Studio Code con extensiones para debugging y linting.

**3.3 Control de Versiones y Colaboración**

El código fuente se almacena en un repositorio GitHub con tres ramas principales:

* dev: Desarrollo activo.
* main: Producción.  
  Se utiliza la metodología GitFlow, con revisiones de código mediante Pull Requests para garantizar la calidad.

**4. ARQUITECTURA DEL SISTEMA**

**4.1 Visión General de la Arquitectura**

**Future Cash** adopta una arquitectura cliente-servidor. El frontend se ejecuta en el navegador del usuario, mientras que el backend opera como un servidor RESTful que proporciona datos a través de APIs JSON sobre HTTPS. Esta estructura modular facilita la escalabilidad y el mantenimiento.

**4.2 Componentes del Frontend**

* **Login y Registro:** Formularios con validación en tiempo real y ofuscación de contraseñas.
* **Dashboard:** Navegación lateral dinámica y carga de contenido mediante peticiones asíncronas.
* **Panel de Usuario:** Menú desplegable con información personal editable.

**4.3 Componentes del Backend**

* **APIs Principales:**
  + POST /api/usuarios/login: Autenticación de usuarios.
  + POST /api/usuarios/signup: Registro de nuevos usuarios.
  + GET /api/usuarios/info: Obtención de datos del usuario autenticado.
  + PUT /api/usuarios/updateInfo: Actualización de datos personales (en desarrollo).

**4.4 Base de Datos**

* **Esquema:**
  + Tabla personas: Almacena datos personales (nombre, email, etc.).
  + Tabla usuarios: Almacena credenciales y relación con personas mediante persona\_id.
* **Motor:** MySQL con soporte para transacciones y claves foráneas.

**4.5 Integraciones y APIs**

* Actualmente, el sistema opera de manera independiente.
* **Futuras Integraciones:** APIs externas para datos financieros en tiempo real (pendiente de implementación).

**5. DETALLES TÉCNICOS**

**5.1 Flujo de Datos y Procesos**

1. El usuario accede a la página de inicio (login.html) y completa el formulario de login o registro.
2. Tras la autenticación, se redirige al dashboard.html, que carga contenido dinámico desde /src/views/blog/ usando fetch.
3. En el panel de usuario, los datos se obtienen mediante una llamada a /api/usuarios/info.
4. Las actualizaciones se envían al backend vía PUT /api/usuarios/update.

**5.2 Seguridad**

* **Autenticación:** Contraseñas almacenadas como texto plano (actualmente se está buscando implementar hash con bcrypt u otro tipo de cifrado robusto, se encuentra en fase de investigación para tener conocimientos acerca del cifrado).
* **Protección:** Uso de consultas parametrizadas para evitar inyecciones SQL.
* **Pendiente:** Implementación de tokens CSRF y protección contra ataques XSS (actualmente se encuentra también en fase de implementación, recopilando información para su implementación de manera correcta).

**5.3 Rendimiento y Escalabilidad**

* **Optimización Actual:** Carga dinámica de recursos en el frontend.
* **Mejoras Futuras:**
  + Implementación de caché para datos estáticos y respuestas de API.
  + Balanceo de carga para manejar alta concurrencia.

**6. INSTALACIÓN Y DESPLIEGUE**

**6.1 Requisitos del Sistema**

* **Servidor:** Node.js v14 o superior, MySQL v8 o superior.
* **Cliente:** Navegador moderno (Chrome, Firefox, Safari).

**6.2 Pasos de Instalación**

1. Clonar el repositorio: git clone https://github.com/[tu-repositorio].git.
2. Instalar dependencias: npm install.
3. Configurar la conexión a la base de datos en server.js.
4. Iniciar el servidor: node server.js.

**6.3 Configuración Inicial**

* Crear la base de datos db\_web1 y las tablas personas y usuarios.
* Definir variables de entorno (e.g., puerto, credenciales de MySQL) en un archivo .env.

**7. MANTENIMIENTO Y SOPORTE**

**7.1 Monitoreo y Logs**

* **Logs:** Pendiente de implementación en /logs/app.log.
* **Monitoreo:** Recomendación de uso de herramientas como ELK Stack para análisis en tiempo real.

**7.2 Actualizaciones y Parches**

* Actualizaciones gestionadas mediante Git tags (e.g., v1.0.1).
* Parches de seguridad aplicados en ramas hotfix.

**7.3 Soporte Técnico**

* **Contacto:** [[soporte@futurecash.com](mailto:soporte@futurecash.com)].
* **Documentación:** Manual de usuario disponible en /docs.

**8. CONCLUSIONES**

**Future Cash** es una plataforma robusta y flexible, diseñada para ofrecer una experiencia de usuario óptima y un manejo eficiente de datos financieros. Su arquitectura modular y la metodología ágil empleada aseguran que el sistema pueda evolucionar con nuevas funcionalidades y mejoras. Para optimizarlo, se recomienda implementar cifrado de contraseñas, mecanismos de caché y estrategias de escalabilidad avanzadas.

**9. ANEXOS**

**9.1 Diagramas de Arquitectura**

* Diagrama de componentes del sistema (a incluir).
* Diagrama de secuencia para el proceso de login (a incluir).

**9.2 Glosario**

* **Sidenav:** Navegación lateral en la interfaz.
* **Dashboard:** Panel principal del usuario.

**9.3 Referencias**

* Documentación oficial de Express.js.
* Guías de seguridad OWASP.

**Resumen Ejecutivo**

El **Manual Técnico de Future Cash (FC)** ofrece una visión integral del sistema, abarcando su funcionalidad, alcance, metodología de programación y arquitectura. Desarrollado con un enfoque ágil, **FC** combina un frontend interactivo con un backend RESTful seguro, soportado por una base de datos MySQL. Este documento sirve como guía para comprender y mantener la plataforma, destacando su potencial de crecimiento y las mejoras sugeridas para seguridad y rendimiento.