PLANIFICACIÓN:BANK MANAGER

Proyecto Final III BIM

H.E - Software Development

Centro Educativo Técnico Laboral KINAL

🏫 **IN6BM – 2025**

🪪 **Anthony Josue Escobar Ponce - 202022G**

🪪 **Aníbal Guillermo Herrera Ortiz – 2020324**

Sistema de gestión bancario

"El objetivo principal es crear y desarrollar una aplicación web de sistema bancario moderno, funcional y seguro. Lograr que permita la administración de usuarios y cuentas bancarias a través de un panel administrativo y de trabajador, además de una interfaz de cliente. Este sistema debe cumplir con el hecho de realizar gestiones seguras y dinámicas de usuarios, cuentas, movimientos financieros, y servicios exclusivos. contemplando autenticación por tokens, gestión de movimientos y productos, y soporte para divisas externas."

PLANIFICACION SCRUM

TEAM:

Ambos integrantes formaremos parte del equipo de desarrollo, colaborando activamente en la construcción de la aplicación.

ROLES PRINCIPALES:

Anthony Escobar:

Product Owner, guiando la visión del producto y priorizando el backlog.

Scrum Master, quien facilitará las ceremonias SCRUM y ayudará a resolver obstáculos.

Aníbal Herrera:

Desarrollador Principal, Desarrollador principal de Frontend de nuestra API web y developer encargado de estructuración final.

SPRINTS:

SEMANA 1

Definición de tecnologías, módulos, atributos, funcionalidades a crear y utilizar para el desarrollo de nuestra aplicación web, investigando y conociendo otras API's web de sistema bancario para utilizar como ideas base; establecer estilos, fonts, vistas del usuario por rol desde el frontend y estructuración del proyecto primera versión.

SEMANA 2

Creación y desarrollo de Api Backend, definiendo e implementando la creación de nuestra construcción con módulos, controladores, middlewares, helpers, y routes definidos.

SEMANA 3

Consultar API's web de divisas para utilizar, Y consultoría de ideas para implementación de la misma, para utilización en cuentas y transacciones.

SEMANA 4

Desarrollo final del backend, construcción de modelaje de transacciones movimientos con cuentas relacionadas y sus usuarios, estableciendo posible rol nuevo para manejo general simple como depósitos.

SEMANA 5

Desarrollo de Frontend, estableciendo dashboards de cliente y administrador, con vistas principales sobre gestion de usuarios y productos.

SEMANA 5

Desarrollo final de Frontend, con dashboards finalizados e implementación de necesidades según cada panel de módulos de cuentas movimientos y usuarios.

Revisión y entrega de avances cada 4 dias para corrección de errores y validación de requerimientos a cumplir a continuación de manera previa a cada SPRINT

PLANIFICACION DETALLADA

SEMANA 1-2:

Tecnologías Requeridas

1. Backend: Nodejs, Express ,JavaScript
2. Frontend: React,Vite, Boostrap,CSS
3. Base de Datos: MongoDB
4. Autenticación: JWT para la gestión de sesiones de usuario
5. Librerías: PDFkit
6. Herramientas de Gestión: GitHub

Dependencias:

BACKEND:

- express
- helmet
- cors
- morgan
- dotenv
- jsonwebtoken
- mongoose
- argon2
- express-validator
- swagger-jsdoc
- swagger-ui-express
- express-rate-limit

FRONTEND:

- react-router-dom
- react-hot-toast
- axios
- prop-types

DOCUMENTACIÓN DE LA API (SWAGGER)

Definir endpoints básicos:

- /bankManagement/auth: Registro y login de usuarios.
- /bankbankManagementAPI/user: Gestión de usuarios (clientes, administradores).
- /bankManagement/account: Consultas de saldo, movimientos y operaciones.
- /bankManagement/movement: Depósitos, transferencias, historial.
- /bankManagement/product: Productos bancarios contratables por los clientes.
- /bankManagement/favorite: Alias para cuentas frecuentes.
- /bankManagement/currency: Conversión de saldo a divisas mediante API externa.

Revisiones Técnicas (Cada 3 días)

- **Día 3:** Revisión de modelos y relaciones (Usuario, Cuenta, Movimiento, Producto).
- **Día 6:** Revisión de configuración inicial (Express, conexión MongoDB, estructura de rutas).

SEMANA 2-3:

Desarrollo:

Configuración Inicial del Proyecto:

Crear repositorio en GitHub con estructura MERN (client para React, server para Node.js).

Instalar dependencias básicas: Express, Mongoose, JWT,

CORS. Crear proyecto Node.js con Express.

Conectar a MongoDB y definir esquemas con Mongoose.

Estructurar carpetas: models, controllers, routes, middlewares,helpers,config.

Implementación de Modelos en MongoDB ATLAS:

1. Módulo de autenticación (JWT)
   - Creación de login protegido para el administrador inicial (ADMINB).
   - Emisión de tokens JWT tras el login exitoso.
   - Middleware para protección de rutas según el tipo de usuario.

2. Módulo de usuarios
   - Creación de usuarios clientes y administradores.
   - Validaciones como DPI único, ingresos ≥ Q100, generación automática de número de cuenta.
   - Campos esenciales: nombre, nickname, DPI, celular, dirección, correo, nombre de trabajo, ingresos mensuales, tipo de cuenta.

Módulo de productos/servicios bancarios
   - Registro y gestión de productos exclusivos para clientes.
   - Asociar productos con usuarios.

4. Módulo de movimientos financieros
   - Depósitos (modificables en cantidad, reversibles antes de 1 minuto).
   - Transferencias entre cuentas (máximo Q2,000 por transacción, máximo Q10,000 diarios).
   - Registro detallado en historial de cuenta.
   - Cálculo de saldo automático en base a movimientos.

5. Módulo de visualización administrativa
   - Vista de usuarios (excepto otros administradores).
   - Últimos 5 movimientos por usuario.
   - Ordenar cuentas por cantidad de movimientos.
   - Visualización de saldo de cada cliente.

6. Módulo de clientes (interfaz)
   - Visualizar y editar datos personales (excepto DPI y contraseña).
   - Consultar saldo e historial.
   - Gestionar favoritos (cuentas con alias).
   - Realizar transferencias desde favoritos.

7. Módulo de divisas
   - Integrar API externa (IBM/Google) para convertir saldo actual a distintas monedas.
   -

Tareas Principales:

1. **Autenticación y Autorización (JWT):**
   - Middleware para validación de roles (validateJWT, hasRole).
   - Controladores: auth, users, accounts, movements, products.
2. **Modelos principales:**
   - User: Validación de DPI único, ingresos ≥ Q100, rol por defecto CLIENT.
   - Account: Generación automática de número de cuenta, saldo.
   - Movement: Depósitos, transferencias, historial.
   - Product: Registro de productos contratables por usuarios.
3. **Controllers:**
   - Controlador por modelo (usuarios, cuentas, movimientos, productos, favoritos, conversión).
   - Controlador para consumo de API externa de divisas.
4. **Restricciones de negocio implementadas:**
   - Transferencias ≤ Q2000, máximo diario Q10,000.
   - Depósitos editables hasta 1 minuto.
   - Rechazo de transferencias con saldo insuficiente.
   - Alias con cuentas favoritas y acceso directo.

Revisiones Técnicas:

- **Día 3 (Semana 2):** Validar autenticación, creación de cuentas y middleware por rol.
- **Día 6 (Semana 2):** Verificación de transferencias, límites de montos y reversión de depósitos.

SEMANA 4-5:

Tareas Principales:

- **Vistas principales:**
  - Login/Register.
  - Dashboard de usuario (saldo, movimientos).
  - Panel de administrador (gestión de usuarios/cuentas).
  - Formularios para transferencias y contratación de productos.

1. **Autenticación en React:**
   - LoginForm con manejo de JWT y roles.
   - Persistencia del usuario en localStorage.
2. **Panel de Cliente:**
   - Visualizar saldo, movimientos e historial.
   - Realizar transferencias y depósitos.
   - Agregar cuentas favoritas y enviar por alias.
3. **Panel Administrativo:**
   - Crear y gestionar usuarios.
   - Ver historial completo por cuenta.
   - Revisar cantidad de movimientos y saldo por cliente.
   - Registrar productos bancarios disponibles.
4. **Integración de API de Divisas:**
   - Consultar el equivalente del saldo actual en USD/EUR.

Revisiones Técnicas:

- **Día 3 (Semana 3):** Verificar integración del token, rutas protegidas, lógica por rol.
- **Día 6 (Semana 3):** Pruebas de funcionalidades en dashboards (cliente y admin), consulta de divisas

SEMANA 5-6:

Pruebas y Despliegue

- **Testing:**
  - Postman (backend).
  - Pruebas de usabilidad (frontend).
- **Despliegue:**
  - Configurar dominios y variables de entorno.
  - Asegurar HTTPS y CORS.

Herramientas Adicionales

- **Librerías:**
  - API de divisas (Exchange).
- **Gestión:**
  - Reuniones diarias vía WhatsApp.
  - Revisiones técnicas cada 3 días.

Criterios de Aceptación

- Backend probado al 100% (Postman).
- Frontend responsive y sin errores en consola.
- Despliegue funcional en Vercel/Firebase.

Ceremonias SCRUM

- **Daily Stand-ups:** Reuniones diarias 8:00 PM por WhatsApp para revisar avances.
- **Sprint Planning:** Lunes de cada semana.
- **Sprint Review:** Viernes (demostración funcional).
- **Retrospectiva:** Viernes post-review para mejoras y ajustes.

Criterios de Aceptación

- Backend probado con Postman al 80% (login, depósitos, transferencias, consultas, favoritos).
- Lógica de negocios implementada correctamente (restricciones, validaciones, saldos).
- Frontend funcional sin errores en consola.
- Paneles separados para cliente y administrador.
- Conversión de saldo funcionando con API externa.
- Despliegue exitoso en Firebase (frontend) y Vercel (backend).