**1. Backend - Node.js (Express + MongoDB)**

This backend provides **user authentication, account management, deposits, withdrawals, and balance**

**Setup**

Run the following commands to set up your backend:

mkdir asset-x-bank-backend

cd asset-x-bank-backend

npm init -y

npm install express mongoose dotenv jsonwebtoken bcrypt cors

**Backend Code:**

Create a file server.js

**require("dotenv").config();**

**const express = require("express");**

**const mongoose = require("mongoose");**

**const bcrypt = require("bcrypt");**

**const jwt = require("jsonwebtoken");**

**const cors = require("cors");**

**const app = express();**

**app.use(express.json());**

**app.use(cors());**

**mongoose.connect(process.env.MONGO\_URI, { useNewUrlParser: true, useUnifiedTopology: true });**

**// User Schema**

**const UserSchema = new mongoose.Schema({**

**username: String,**

**password: String,**

**balance: { type: Number, default: 0 }**

**});**

**const User = mongoose.model("User", UserSchema);**

**// Middleware for authentication**

**const authMiddleware = async (req, res, next) => {**

**const token = req.header("Authorization");**

**if (!token) return res.status(401).json({ error: "Unauthorized" });**

**try {**

**const decoded = jwt.verify(token, process.env.JWT\_SECRET);**

**req.user = decoded;**

**next();**

**} catch (err) {**

**res.status(400).json({ error: "Invalid token" });**

**}**

**};**

**// \*\*Routes\*\***

**// User Signup**

**app.post("/signup", async (req, res) => {**

**const { username, password } = req.body;**

**const hashedPassword = await bcrypt.hash(password, 10);**

**const user = new User({ username, password: hashedPassword });**

**await user.save();**

**res.json({ message: "User created successfully" });**

**});**

**// User Login**

**app.post("/login", async (req, res) => {**

**const { username, password } = req.body;**

**const user = await User.findOne({ username });**

**if (!user || !(await bcrypt.compare(password, user.password))) {**

**return res.status(400).json({ error: "Invalid credentials" });**

**}**

**const token = jwt.sign({ userId: user.\_id }, process.env.JWT\_SECRET, { expiresIn: "1h" });**

**res.json({ token });**

**});**

**// Get Balance**

**app.get("/balance", authMiddleware, async (req, res) => {**

**const user = await User.findById(req.user.userId);**

**res.json({ balance: user.balance });**

**});**

**// Deposit Money**

**app.post("/deposit", authMiddleware, async (req, res) => {**

**const { amount } = req.body;**

**if (amount <= 0) return res.status(400).json({ error: "Invalid amount" });**

**const user = await User.findById(req.user.userId);**

**user.balance += amount;**

**await user.save();**

**res.json({ message: "Deposit successful", balance: user.balance });**

**});**

**// Withdraw Money**

**app.post("/withdraw", authMiddleware, async (req, res) => {**

**const { amount } = req.body;**

**const user = await User.findById(req.user.userId);**

**if (amount <= 0 || amount > user.balance) {**

**return res.status(400).json({ error: "Invalid or insufficient funds" });**

**}**

**user.balance -= amount;**

**await user.save();**

**res.json({ message: "Withdrawal successful", balance: user.balance });**

**});**

**app.listen(5000, () => console.log("Server running on port 5000"));**

**Environment Variables (.env)**

Create a .env file:

**MONGO\_URI=mongodb://localhost:27017/assetx-bank**

**JWT\_SECRET=your\_secret\_key**

**2. Frontend - React**

This frontend allows users to **sign up, log in, check balance, deposit, and withdraw money**.

**Setup**

Run the following commands:

**npx create-react-app asset-x-bank-frontend**

**cd asset-x-bank-frontend**

**npm install axios react-router-dom**

**Frontend Code**

Create a services/api.js file:

**import axios from "axios";**

**const API\_URL = "http://localhost:5000";**

**export const signup = async (username, password) => {**

**return axios.post(`${API\_URL}/signup`, { username, password });**

**};**

**export const login = async (username, password) => {**

**const response = await axios.post(`${API\_URL}/login`, { username, password });**

**localStorage.setItem("token", response.data.token);**

**return response.data;**

**};**

**export const getBalance = async () => {**

**const token = localStorage.getItem("token");**

**return axios.get(`${API\_URL}/balance`, { headers: { Authorization: token } });**

**};**

**export const deposit = async (amount) => {**

**const token = localStorage.getItem("token");**

**return axios.post(`${API\_URL}/deposit`, { amount }, { headers: { Authorization: token } });**

**};**

**export const withdraw = async (amount) => {**

**const token = localStorage.getItem("token");**

**return axios.post(`${API\_URL}/withdraw`, { amount }, { headers: { Authorization: token } });**

**};**

**React Components**

Modify App.js:

**import React, { useState } from "react";**

**import { signup, login, getBalance, deposit, withdraw } from "./services/api";**

**function App() {**

**const [username, setUsername] = useState("");**

**const [password, setPassword] = useState("");**

**const [balance, setBalance] = useState(null);**

**const [amount, setAmount] = useState(0);**

**const [message, setMessage] = useState("");**

**const handleSignup = async () => {**

**await signup(username, password);**

**setMessage("Signup successful!");**

**};**

**const handleLogin = async () => {**

**await login(username, password);**

**setMessage("Login successful!");**

**};**

**const fetchBalance = async () => {**

**const res = await getBalance();**

**setBalance(res.data.balance);**

**};**

**const handleDeposit = async () => {**

**await deposit(amount);**

**fetchBalance();**

**};**

**const handleWithdraw = async () => {**

**await withdraw(amount);**

**fetchBalance();**

**};**

**return (**

**<div>**

**<h1>Asset-x Banking</h1>**

**<input type="text" placeholder="Username" onChange={(e) => setUsername(e.target.value)} />**

**<input type="password" placeholder="Password" onChange={(e) => setPassword(e.target.value)} />**

**<button onClick={handleSignup}>Signup</button>**

**<button onClick={handleLogin}>Login</button>**

**<p>{message}</p>**

**<h2>Account Actions</h2>**

**<button onClick={fetchBalance}>Check Balance</button>**

**{balance !== null && <p>Balance: ${balance}</p>}**

**<input type="number" placeholder="Amount" onChange={(e) => setAmount(Number(e.target.value))} />**

**<button onClick={handleDeposit}>Deposit</button>**

**<button onClick={handleWithdraw}>Withdraw</button>**

**</div>**

**);**

**}**

**export default App;**