🧾Backend Fejlesztői Dokumentáció

# 1. Projekt Áttekintés

Ez a Next.js alapú backend egy személyes pénzügyi nyilvántartó rendszer backend oldala. Lehetővé teszi a felhasználók számára kiadásaik és bevételeik nyilvántartását, kategorizálását, valamint összesítések készítését.

# 2. Technológiai Stack

|  |  |
| --- | --- |
| Technológia | Leírás |
| Next.js | Full-stack React keretrendszer (API route-okat használunk backendként) |
| Prisma ORM | Típusbiztos adatmodell |
| MongoDB | NoSQL adatbázis a tranzakciók tárolására |
| Swagger (OpenAPI) | Automatikus API dokumentáció |
| TypeScript | Típusbiztos, könnyebben karbantartható kód |

# 3. Telepítés (Local Dev)

1. Repo klónozása és csomagok telepítése

git clone <https://github.com/Dani5822/bank-backend.git>

cd bank-backend

npm install

2. `.env` fájl beállítása

Test user:

DATABASE\_URL=”mongodb+srv://test:test@user.0z2pk.mongodb.net/test”  
JWT\_SECRET="trzfugioplmnfxbcvbjnluzktjtuziozupoéjkbjhvg"

3. Prisma generálás

npx prisma generate

4. Fejlesztői szerver indítása

npm run start:dev

# 4. Adatmodell (Prisma)

enum Currency {

  HUF

  USD

  EUR

}

enum ExpenseCategory {

  Shopping

  Rent

  Transport

  Transfer

  Transaction

  Other

}

enum IncomeCategory {

  Salary

  Transfer

  Transaction

  Other

}

enum Metric {

  Day

  Week

  Month

  Year

}

model User {

  id        String    @id @default(auto()) @map("\_id") @db.ObjectId

  firstName String

  lastName  String

  role      UserType  @default(User)

  email     String    @unique

  password  String

  Expenses  Expense[]

  expenseId String[]  @db.ObjectId

  Incomes   Income[]

  incomeId  String[]  @db.ObjectId

  createdAt DateTime  @default(now())

  updatedAt DateTime  @updatedAt

  Accounts  Account[] @relation(fields: [accountId], references: [id])

  accountId String[]  @db.ObjectId

}

model Income {

  id           String         @id @default(auto()) @map("\_id") @db.ObjectId

  total        Float

  category     IncomeCategory @default(Other)

  description  String?

  User         User           @relation(fields: [userId], references: [id])

  userId       String         @db.ObjectId

  Account      Account        @relation(fields: [accountId], references: [id])

  accountId    String         @db.ObjectId

  createdAt    DateTime       @default(now())

  updatedAt    DateTime       @updatedAt

}

model Expense {

  id           String          @id @default(auto()) @map("\_id") @db.ObjectId

  total        Float

  category     ExpenseCategory @default(Other)

  description  String?

  User         User            @relation(fields: [userId], references: [id])

  userId       String          @db.ObjectId

  createdAt    DateTime        @default(now())

  updatedAt    DateTime        @updatedAt

  Account      Account         @relation(fields: [accountId], references: [id])

  accountId    String          @db.ObjectId

  RepeatableTransaction RepeatableTransaction? @relation(fields: [repeatableTransactionId], references: [id])

  repeatableTransactionId String? @db.ObjectId

}

model Account {

  id        String    @id @default(auto()) @map("\_id") @db.ObjectId

  Users     User[]    @relation(fields: [userId], references: [id])

  userId    String[]  @db.ObjectId

  total     Float

  ownerName String

  ownerId String

  Expenses  Expense[]

  Incomes   Income[]

  createdAt DateTime  @default(now())

  updatedAt DateTime  @updatedAt

  currency  Currency  @default(HUF)

  RepeatableTransaction RepeatableTransaction[]

}

model RepeatableTransaction {

  id           String   @id @default(auto()) @map("\_id") @db.ObjectId

  name         String?

  repeatStart  DateTime @default(now())

  repeatAmount Int      @default(1)

  repeatMetric Metric   @default(Day)

  repeatEnd    DateTime @default(now())

  lastChange   DateTime @default(now())

  Account      Account  @relation(fields: [accountId], references: [id])

  accountId    String   @db.ObjectId

  createdAt    DateTime @default(now())

  total        Float

  category     ExpenseCategory @default(Other)

  description  String?

  Expenses Expense[]

# 5. Swagger Dokumentáció

Swagger UI megtekintés:  
1. Indísd el a szervert (npm run start:dev)  
2. Add meg az URL-t a keresőben: http://localhost:3000/docs