Roteiro da aula 2

1. Alterar o arquivo schema. prisma, adicionando as tabelas:

model Day {

id String @id @default(uuid())

date DateTime

@@map("days")

@@unique([date])

}

model DayHabit {

id String @id @default(uuid())

day\_id String

habit\_id String

@@map("day\_habits")

@@unique([day\_id, habit\_id])

}

1. Alterar o arquivo schema. prisma, adicionando as tabelas:

model HabitWeekDay {

id String @id @default(uuid())

habit\_id String

week\_day Int

@@unique([habit\_id, week\_day])

@@map("habit\_week\_days")

}

1. npx prisma migrate dev
2. alterar o arquivo schema.prisma, adicionando os relacionamentos entre as tabelas:

model DayHabit {

id String @id @default(uuid())

day\_id String

habit\_id String

day Day @relation(fields: [day\_id], references: [id])

habit Habit @relation(fields: [habit\_id], references: [id])

@@unique([day\_id, habit\_id])

@@map("day\_habits")

}

model Day {

id String @id @default(uuid())

date DateTime

dayHabits DayHabit[]

@@unique([date])

@@map("days")

}

model HabitWeekDay {

id String @id @default(uuid())

habit\_id String

week\_day Int

habit Habit @relation(fields:[habit\_id], references: [id])

@@unique([habit\_id, week\_day])

@@map("habit\_week\_days")

}

model Habit {

id String @id @default(uuid())

title String

created\_at DateTime

dayHabits DayHabit[]

weekDays HabitWeekDay[]

@@map("habits")

}

1. npx prisma migrate dev
2. npm i -D prisma-erd-generator @mermaid-js/mermaid-cli
3. alterar o arquivo schema.prisma como:  
    generator erd{

provider = “prisma-erd-generator”

}

1. npx prisma generate
2. na pasta prisma, crie um arquivo seed.ts

import { PrismaClient } from '@prisma/client'

const prisma = new PrismaClient()

const firstHabitId = '0730ffac-d039-4194-9571-01aa2aa0efbd'

const firstHabitCreationDate = new Date('2022-12-31T03:00:00.000')

const secondHabitId = '00880d75-a933-4fef-94ab-e05744435297'

const secondHabitCreationDate = new Date('2023-01-03T03:00:00.000')

const thirdHabitId = 'fa1a1bcf-3d87-4626-8c0d-d7fd1255ac00'

const thirdHabitCreationDate = new Date('2023-01-08T03:00:00.000')

async function run() {

await prisma.habit.deleteMany()

await prisma.day.deleteMany()

/\*\*

\* Create habits

\*/

await Promise.all([

prisma.habit.create({

data: {

id: firstHabitId,

title: 'Beber 2L água',

created\_at: firstHabitCreationDate,

weekDays: {

create: [

{ week\_day: 1 },

{ week\_day: 2 },

{ week\_day: 3 },

]

}

}

}),

prisma.habit.create({

data: {

id: secondHabitId,

title: 'Exercitar',

created\_at: secondHabitCreationDate,

weekDays: {

create: [

{ week\_day: 3 },

{ week\_day: 4 },

{ week\_day: 5 },

]

}

}

}),

prisma.habit.create({

data: {

id: thirdHabitId,

title: 'Dormir 8h',

created\_at: thirdHabitCreationDate,

weekDays: {

create: [

{ week\_day: 1 },

{ week\_day: 2 },

{ week\_day: 3 },

{ week\_day: 4 },

{ week\_day: 5 },

]

}

}

})

])

await Promise.all([

/\*\*

\* Habits (Complete/Available): 1/1

\*/

prisma.day.create({

data: {

/\*\* Monday \*/

date: new Date('2023-01-02T03:00:00.000z'),

dayHabits: {

create: {

habit\_id: firstHabitId,

}

}

}

}),

/\*\*

\* Habits (Complete/Available): 1/1

\*/

prisma.day.create({

data: {

/\*\* Friday \*/

date: new Date('2023-01-06T03:00:00.000z'),

dayHabits: {

create: {

habit\_id: firstHabitId,

}

}

}

}),

/\*\*

\* Habits (Complete/Available): 2/2

\*/

prisma.day.create({

data: {

/\*\* Wednesday \*/

date: new Date('2023-01-04T03:00:00.000z'),

dayHabits: {

create: [

{ habit\_id: firstHabitId },

{ habit\_id: secondHabitId },

]

}

}

}),

])

}

run()

.then(async () => {

await prisma.$disconnect()

})

.catch(async (e) => {

console.error(e)

await prisma.$disconnect()

process.exit(1)

})

1. npx prisma db seed
2. na pasta src, crie uma pasta lib
3. crie o arquivo prisma.ts com o conteúdo

import { PrismaClient } from '@prisma/client'

export const prisma = new PrismaClient()

1. crie o arquivo routes.ts na pasta src

import {FastifyInstance} from 'fastify'

import { prisma } from "./lib/prisma"

export async function AppRoutes(app: FastifyInstance){

app.get('/hello2', async () => {

const habits = await prisma.habit.findMany({

where: {

title: {

startsWith: 'beber'

}

}

})

return habits

})

1. alterar o arquivo src/server.ts como:

import Fastify from 'fastify'

import cors from '@fastify/cors'

import { AppRoutes } from './routes'

const app = Fastify()

app.register(cors)

app.register(AppRoutes)

app.listen({

port: 3333,

})

.then( () => {

console.log('Http Server running')

})

1. npm install zod
2. npm install dayjs
3. alterar arquivo routes.ts, adicionando

app.post("/habits", async (request, response) => {

const createHabitBody = z.object({

title: z.string(),

weekDays: z.array(z.number().min(0).max(6)),

});

const { title, weekDays } = createHabitBody.parse(request.body);

const today = dayjs().startOf("day").toDate();

try {

const habit = await prisma.habit.create({

data: {

title,

created\_at: today,

weekDays: {

create: weekDays.map((weekDay) => {

return {

week\_day: weekDay,

};

}),

},

},

});

response.status(201).send({ habit });

} catch (error) {

console.error(error);

response.status(400).send({ error: "Error on create a new habit!" });

throw new Error("Error on create a new habit!");

}

});