Next JS Notes

Step 1:

Basic - Set up the project.

```
npm i daisyui@latest prisma @prisma/client next-auth @auth/prisma-adapter prettier eslint-config-prettier prettier-plugin-tailwindcss
```

Install Tailwind CSS Intellisense package - Suggestions need to be done

Daisy UI Setup -:

```
@plugin "daisyui";
```

Install Prettier and make it the default code formatter.
extend eslint config to include prettier.

prettier.config.js

```
module.exports = {
  plugins: ["prettier-plugin-tailwindcss"],
};
```

Install eslint and prisma extensions in VS Code.

Install zod too for future validations

```
npm i zod
```

Step 2 :

- Mongodb Project
- Create a Project.
- Add a clustor.
- Generate Password in databse access - PE4owLTNZ42VaWLb
- Check Network Access for ip whitelisting
- Connect to compass
- Add a sample data to make sure everything is working

  aGbNUjShoPR3ZRhG

Step 3:
Introduction to Prisma

https://www.prisma.io/docs/orm/overview/introduction

Prisma Setup:

```
npx prisma init
```

schema.prisma file change

```
// This is your Prisma schema file,
// learn more about it in the docs: https://pris.ly/d/prisma-schema

// Looking for ways to speed up your queries, or scale easily with your serverl
ess or edge functions?
// Try Prisma Accelerate: https://pris.ly/cli/accelerate-init

generator client {
  provider = "prisma-client-js"
```

```
  output   = "../app/generated/prisma"
}

datasource db {
  provider = "mongodb"
  url      = env("DATABASE_URL")
}
```

creating schema from existing databse using prisma built in function

```
npx prisma db pull
```

example schema

```
model Products {
  id          String   @id @default(auto()) @map("_id") @db.ObjectId
  description String
  image       String
  name        String
  price       Int
  createdAt   DateTime @default(now())
  updatedAt   DateTime @default(now())

  @@map("products")
}
```

in order to push the changes in schema to reflect in db
used to make database in sync with the Prisma schema.

```
npx prisma db push
```

Need to create a client to connect with the db

```
npx prisma generate
```

create a prisma instance at lib→db→prisma.ts

```
import { PrismaClient } from "@prisma/client";

const globalForPrisma = globalThis as unknown as { prisma: PrismaClient };

export const prisma = globalForPrisma.prisma || new PrismaClient();

if (process.env.NODE_ENV !== "production") globalForPrisma.prisma = prism
a;
```

Step 4:
configure Husky

```
npm install --save-dev husky lint-staged
```

```
npx husky init
```

precommit file

```
#!/usr/bin/env sh
set -e

npx lint-staged
npm run -s typecheck
```

package.json file chanage

```
"lint-staged": {
   "*.{js,jsx,ts,tsx}": [
     "eslint --fix --max-warnings=0",
     "prettier --write"
   ]
 }


 "typecheck": "tsc -p tsconfig.json --noEmit"
```