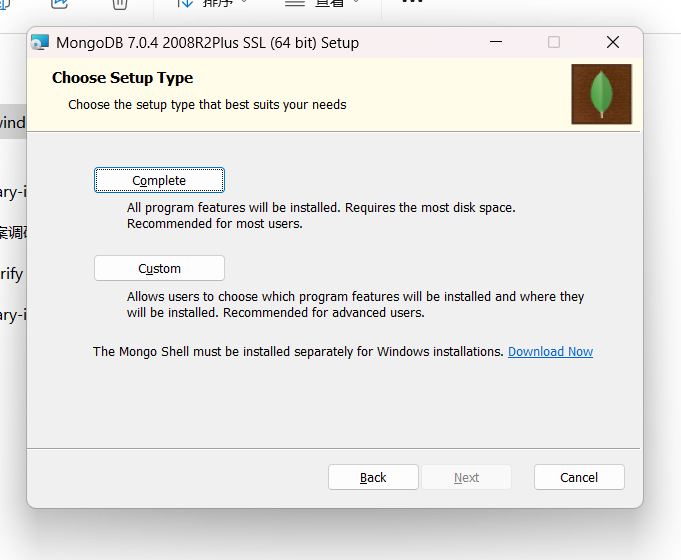
# 安装MongoDB

* 打开这个网站 <https://www.mongodb.com/try/download/community> 下载APP
* 点击下载好的APP 然后 next 在
* 在这个步骤时

在安装MongoDB的过程中，你通常会遇到“Complete”和“Custom”安装选项：

- \*\*Complete（完全安装）\*\*：这个选项将安装MongoDB的所有功能和组件，包括数据库服务器、Mongo Shell（命令行工具），以及其他相关工具。对于大多数用户来说，这是推荐的选项，因为它包含了你可能需要的所有功能。

- \*\*Custom（自定义安装）\*\*：如果你需要自定义安装，比如改变安装路径或者只选择安装某些特定的组件，那么可以选择这个选项。这适用于那些熟悉MongoDB并且知道他们具体需要哪些组件的高级用户。

根据截图中的提示，如果你不需要特定的自定义设置，可以选择“Complete”来进行全面安装。这样做可以确保你的开发环境中包含了所有可能需要的MongoDB工具和服务。

* 在上边这个步骤时图形用户界面, 应用程序

  描述已自动生成

在这个安装阶段，你需要配置MongoDB服务的一些选项。这包括如何运行服务、服务的名称以及数据和日志文件的存储位置。以下是你可以设置的一些选项：

1. \*\*安装MongoDB作为服务\*\*：确保选中了“Install MongoDB as a Service”，这样MongoDB将作为一个Windows服务安装，能够在Windows启动时自动运行。

2. \*\*服务运行方式\*\*：

- \*\*作为网络服务用户运行\*\*：选择“Run service as Network Service user”将使用Windows的网络服务账户来运行MongoDB服务。这通常是最简单的配置方式，不需要额外的用户管理。

- \*\*作为本地或域用户运行\*\*：如果选择“Run service as a local or domain user”，则需要提供特定的用户账户信息。这是一个更高级的选项，通常用于企业环境，需要你输入账户的域、名称和密码。

3. \*\*服务名称\*\*：

- 你可以保留默认的服务名称“MongoDB”，或者如果你希望自定义服务名称，也可以在此更改。

4. \*\*数据目录\*\*：

- 数据目录是MongoDB存储其数据文件的地方。点击“Browse”按钮选择一个你希望存放数据的路径。确认这个位置有足够的空间存储你的数据库。

5. \*\*日志目录\*\*：

- 日志目录是MongoDB将其日志文件写入的地方。同样，点击“Browse”按钮选择一个路径用于存储日志文件。

一旦你设置好这些选项，就可以点击“Next”继续安装过程。如果你更改了数据或日志的目录，请确保所选目录存在并且MongoDB服务具有读写这些目录的权限。如果目录不存在，安装程序通常会为你创建它们

* 这个步骤时图形用户界面, 文本, 应用程序, 电子邮件

  描述已自动生成

在这个安装步骤中，你被问到是否要安装MongoDB Compass。MongoDB Compass是MongoDB的官方图形用户界面（GUI），它可以帮助你更直观地管理和查询MongoDB数据库。

如果你是MongoDB的新用户，我建议安装MongoDB Compass，因为它会使得学习和操作MongoDB变得更简单。它提供了一个用户友好的界面，可以让你执行查询、查看数据结构、分析性能问题，以及更直观地查看数据库中的文档和集合。

如果你打算使用MongoDB Compass，只需选中“Install MongoDB Compass”复选框，然后点击“Next”继续安装过程。这将自动下载并安装最新版本的MongoDB Compass。

如果你决定不通过安装程序安装Compass，你可以随时从MongoDB官方网站手动下载并安装它。

* 安装成功后 这个界面

图形用户界面, 文本, 应用程序

描述已自动生成

MongoDB Compass 提供了多种语言选项，包括中文。你可以按照以下步骤来将界面设置为中文：

1. 打开 MongoDB Compass。

2. 在顶部菜单栏中，找到并点击 “Edit” 菜单。

3. 在下拉菜单中，选择 “Preferences” 或可能的类似选项。

4. 在 “Preferences” 界面中，查找语言或地区设置。

5. 从语言选项中选择中文（可能显示为 “Chinese” 或 “中文”）。

6. 确认你的选择，并根据需要重新启动 MongoDB Compass 以应用更改。

如果找不到直接的语言设置选项，可能是因为当前版本的 MongoDB Compass 不支持在应用内直接切换语言。在这种情况下，Compass 通常会使用操作系统的默认语言设置。

如果在 MongoDB Compass 中找不到中文设置选项，并且你的操作系统已经设置为中文，Compass 应该会自动显示为中文界面。如果操作系统是英文的，而你希望使用中文界面，可能需要检查 MongoDB Compass 的支持文档或在线资源，看看是否有其他用户找到了解决方案。有时候，这些设置可能会因为软件更新而发生变化。

# 运行mongodb

命令: net start MongoDB或者 mongod

* 这个是安装好后 默认的相关信息

# 文本 描述已自动生成

# 要创建一个表

* 图形用户界面, 文本, 应用程序

  描述已自动生成

在MongoDB Compass的“Create Database”窗口中，你需要填写两个字段来创建新的数据库和集合：

1. \*\*Database Name\*\*：在这里输入你想要创建的数据库的名称。例如，如果你正在创建一个用于用户信息的数据库，你可能会命名为 `UserDB` 或 `user\_accounts`。

2. \*\*Collection Name\*\*：在这里输入你想在新数据库中创建的第一个集合的名称。集合在MongoDB中类似于传统关系型数据库中的表。因为你想要创建一个用户登录表，你可以将这个集合命名为 `users` 或 `user\_logins`。

填写这些信息后，你可以点击 \*\*"Create Database"\*\* 按钮来创建数据库和集合。

这里是一个如何填写的例子：

- Database Name: `UserDB`

- Collection Name: `users`

这将创建一个新的数据库 `UserDB` 和一个新的集合 `users`。在这个集合中，你可以存储用户信息，如用户名、密码（最好是哈希过的）、电子邮件地址等。

完成这些步骤后，你可以在新创建的集合中添加和管理文档，每个文档都代表一个用户记录。

图形用户界面, 文本

描述已自动生成

从您提供的截图来看，您已经成功创建了一个名为 `users\_logins` 的集合，它位于 `user\_accounts` 数据库中。目前这个集合中没有数据。

要在此集合中添加数据，您可以按照以下步骤进行：

1. 点击 \*\*"ADD DATA"\*\* 按钮。这里您有两个选择：

- \*\*"Insert Document"\*\*：通过一个JSON格式的文档来手动插入单个文档。

- \*\*"Import File"\*\*：通过导入一个JSON或CSV文件来批量插入数据。

2. 如果您选择 \*\*"Insert Document"\*\*，会出现一个可以输入JSON数据的编辑器。例如，您可以输入以下内容创建一个用户登录记录：

```json

[

{

"\_id": "uniqueId1",

"username": "exampleUser1",

"password": "exampleHashedPassword1",

"email": "exampleUser1@email.com",

"last\_login": "2023-01-01T00:00:00Z"

},

{

"\_id": "uniqueId2",

"username": "exampleUser2",

"password": "exampleHashedPassword2",

"email": "exampleUser2@email.com",

"last\_login": "2023-01-02T00:00:00Z"

}

] ```

输入数据后，点击 \*\*"Insert"\*\* 按钮来添加文档到集合中。

3. 如果您选择从文件 \*\*"Import File"\*\*，则需要选择一个包含要导入的文档的JSON或CSV文件。这对于批量导入数据很有用。

请记住，密码字段应该存储密码的哈希值而不是明文密码，以确保安全。

一旦您添加了数据，这些文档就会出现在 `users\_logins` 集合中，并可以在MongoDB Compass中进行查看和进一步操作。如果您需要执行查询，可以使用顶部的查询栏来输入MongoDB查询语句。

# 关于nodejs项目的操作

* 先配置session中间件 再配置 passport中间件

这个错误表示Passport试图使用基于会话的认证，但是您的Express应用没有配置会话支持。要解决这个问题，您需要确保`express-session`中间件已经安装并且在您的Express应用中正确配置了。

这里是您可能需要执行的步骤：

1. \*\*安装express-session\*\*:

如果还没有安装`express-session`，您需要安装它。

```bash

npm install express-session

```

2. \*\*配置express-session中间件\*\*:

在您的主文件（`app.ts`或`server.ts`）中，您需要在使用任何Passport中间件之前配置express-session。

```typescript

import session from 'express-session';

import passport from 'passport';

const app = express();

// 设置express-session中间件

app.use(session({

secret: 'your\_secret\_key', // 替换为一个随机密钥

resave: false,

saveUninitialized: false,

cookie: {

maxAge: 1000 \* 60 \* 60 \* 24 // 设置cookie的过期时间，例如，1天

}

}));

// 初始化Passport

app.use(passport.initialize());

app.use(passport.session()); // 这里是passport使用会话

```

请注意，`secret`应该是一个只有您知道的字符串，用于签名会话ID cookie，这是保证会话安全的重要部分。`resave`和`saveUninitialized`的设置可能根据您的具体需求而有所不同。

3. \*\*确保顺序正确\*\*:

确保`app.use(session({...}))`在`passport.initialize()`和`passport.session()`调用之前。这是非常重要的，因为Passport依赖于express-session的功能。

4. \*\*重启服务器\*\*:

您修改配置后，需要重启服务器来使更新生效。

完成上述步骤后，您的登录应该不再报错。如果问题仍然存在，请再次检查您的代码，确认`express-session`的配置没有遗漏，并且所有相关中间件的顺序是正确的。

* Passport中间件
* 关于正确会话的设置

如果在`logoutUser`函数中`req.user`是`undefined`，但在`loginUser`函数中却存在，这通常表明在用户登录和注销之间的会话状态没有被正确维持。这可能是由几个不同的原因造成的：

1. \*\*会话配置问题\*\*：

确保您的应用正确配置了会话（`express-session`）。会话是用来在多个请求之间保持用户状态的，如果没有正确配置，那么用户在登录后的下一个请求（如注销）可能就不会包含他们的身份信息。

2. \*\*Cookie和前端行为\*\*：

如果您的前端和后端是分离的，确保登录后收到的cookie（通常包含会话ID）被正确保存并在随后的请求中发送。如果cookie没有被发送或被某种方式清除了（如用户手动清除或浏览器设置），会话信息就会丢失。

3. \*\*Passport中间件应用问题\*\*：

请确认您在所有需要用户身份信息的路由之前调用了`passport.initialize()`和`passport.session()`。这确保了Passport可以在每个请求中访问会话数据。

4. \*\*会话存储问题\*\*：

如果您的会话存储（如内存、数据库等）无法正确维持会话状态，也可能导致这个问题。例如，如果您使用的是默认的内存存储，那么在服务器重启时会话信息会丢失。

5. \*\*前端请求问题\*\*：

确保前端在发起请求时，例如注销请求，正确地携带了认证信息（如会话cookie）。

6. \*\*路由配置\*\*：

您的路由配置看起来没有问题。确保在`router.post('/logout', logoutUser);`之前已经设置了`passport.initialize()`和`passport.session()`。

为了进一步诊断问题，您可以检查以下几点：

- 当用户登录时，检查服务器是否发送了设置会话cookie的响应。

- 在客户端（浏览器）检查是否保存了这个cookie，并且确保后续请求（如注销）都包含了这个cookie。

- 检查会话存储是否正确配置，确保会话数据能够在请求之间持久化。

- 确保没有任何中间件或浏览器行为在登录和注销请求之间清除了会话数据。