实验三：UML 类图

实验内容：

1.分析系统业务流程和系统功能定义系统的初步静态结构,找出重要的类和对象。

2.使用 StarUML 建模工具绘制出类图.并描述和建立类间的关系。（包扩关联关系、包含关系、扩展关系以及泛化关系。）

3.提交实验报告。

实验步骤：

1. 分析系统业务流程和系统功能定义系统的初步静态结构,找出重要的类和对象。

类图是系统中描述静态结构的一种建模工具。它主要关注系统中的类、它们之间的关系 (如关联、依赖、聚合等），以及类的内部结构，包括属性和操作。

概述： 类图描述了系统的静态结构，包括类和它们之间的关系。这包括类的属性和操作，以及与其他元素的关联。

分析系统业务流程和功能：定义系统的初步静态结构，列出所有可能的类和对象，并找出与系统相关的重要类和对象。

建模分析步骤：

① 寻找名词（候选对象）： 从需求中找出所有名词，它们可能是潜在的类或对象。

② 合并含义相同的名词： 整合具有相似含义的名词，排除不相关的名词，并寻找隐含的名词。

③ 去掉只能作为类属性的名词：排除那些只适合作为类属性而不是独立类的名词。

④ 确定候选类： 剩下的名词即为候选类。

⑤ 确定类的属性： 根据需求、常识、问题域等确定每个类的属性。

⑥ 补充动态属性： 添加类的动态属性，如状态、对象间联系（聚合、关联）等。

⑦ 寻找类的操作： 从需求中的动词、功能或系统责任中找出类的操作。

⑧ 补充操作： 从状态转换、流程跟踪、系统管理等方面补充类的操作。

⑨ 合并、筛选操作： 对寻找的操作进行合并和筛选，确保它们是类的有效操作。

⑩ 分配职责： 将操作在类之间进行合理分配，形成每个类的候选操作。

⑪ 补充类的分析文档： 对每个类进行详细文档补充，为进一步的设计提供基础。

这些步骤有助于在类图中建立系统的静态结构，使得系统的类和对象得以清晰定义和组织。这提供了一个有力的基础，支持进一步的系统设计和开发。

UML 中类图实例

该系统类图按照 Controller，Model，Entity 的划分体现了一种 MVC（模型-视图-控制器）架构的设计理念，通过明确定义各模块的职责，提高了系统的可维护性、可扩展性和复用性。Controller 作为控制器负责用户输入的处理和应用逻辑的协调，有效分离了用户界面和业务逻辑，使得系统更易于理解和修改。Model 作为模型管理数据和业务逻辑，独立于用户界面和控制流程，保证了数据的一致性和可维护性。Entity 作为实体类代表数据对象，通过数据映射实现对数据源的操作，提供了数据的封装和抽象，增强了系统的灵活性。这种模块划分促进了系统的可维护性、可扩展性和测试性，使得开发人员能够更有效地构建、维护和扩展应用程序。

如表 1，系统实体类包括音乐、歌曲集合、歌单、专辑、用户、普通用户、歌手。其中，普通用户和歌手是用户的泛化，歌单和专辑是歌曲集合的泛化。实体类之间通过关联、依赖、聚合等关系建立联系，形成系统的静态结构。如表 2，系统 Controller 类包括用户公共模块控制器类、歌手专有模块控制器类、普通用户专有模块控制器类、搜索模块控制器类、播放器模块控制器类。这些控制器类负责处理不同模块的功能，确保系统在用户交互和业务逻辑方面运行顺畅。同时，如表 3，系统业务类涵盖音乐业务类、歌曲集合业务类、歌单业务类、专辑业务类、用户业务类、普通用户业务类、歌手业务类。这些业务类分别处理系统核心业务，包括

音乐信息管理、歌曲集合编辑、用户信息维护等，为系统提供全面而高效的功能支持。整体而言，这些类的建模是通过分析系统业务流程和功能，找出系统中的重要类和对象，并通过类图形式清晰地呈现出系统的静态结构和业务逻辑。

表 1：系统实体类的整体说明

序号	类名称	解释
1	音乐	代表一首具体的音乐作品，包含音乐的各种属性和信息。
2	歌曲集合	表示包含多首歌曲的集合，可以包括歌单、专辑等。
3	歌单	是歌曲集合的一种泛化，表示包含一组音乐的列表，由普通用户创建和分享。
4	专辑	也是歌曲集合的一种泛化，表示由歌手或音乐制作团队发布的一组音乐作品集。
5	用户	表示系统中的用户，包括普通用户和歌手。
6	普通用户	是用户的一种具体类型，代表系统中普通的音乐爱好者，可以创建歌单、收藏音乐等。
7	歌手	是用户的一种具体类型，代表具有音乐创作和表演能力的个体，可以发布音乐作品。

表 2：系统控制器类的整体说明

序号	类名称	解释
1	用户公共模块控制器类	处理用户登录、注册、个人信息查看和编辑等公共操作。
2	歌手专有模块控制器类	处理歌手信息查看、歌曲上传等操作。
3	普通用户专有模块控制器类	处理普通用户的歌曲收藏、评论等操作。
4	搜索模块控制器类	负责处理用户的搜索请求，返回相关搜索结果。
5	播放器模块控制器类	处理音乐播放和控制操作。

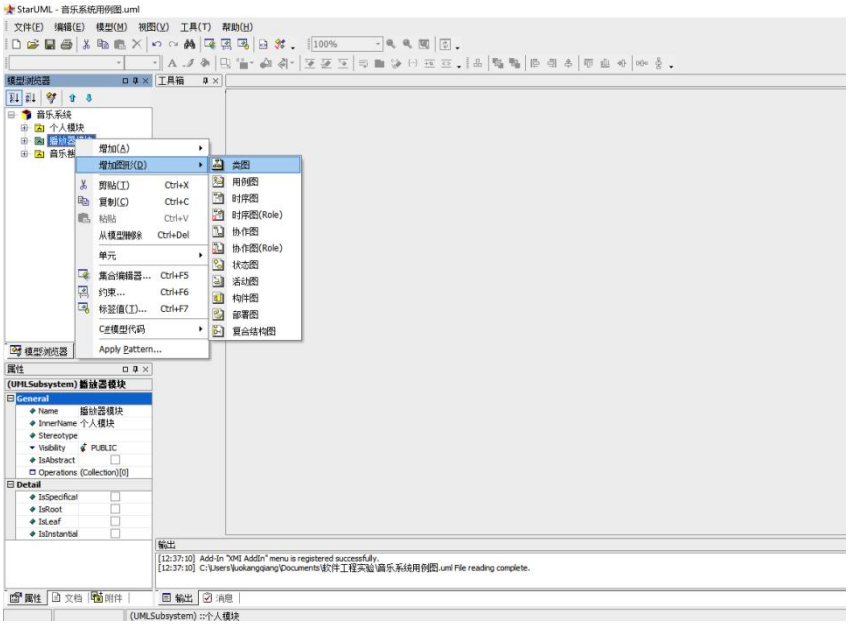
表 3：系统业务模型类的整体说明

序号	类名称	解释
1	音乐业务类	处理音乐相关的业务逻辑，包括音乐信息管理、播放统计等功能。
2	歌曲集合业务类	管理歌曲集合，包括创建、编辑、删除歌曲集合等操作。
3	歌单业务类	歌曲集合业务类的泛化，具有歌曲集合的基本功能，同时拓展了歌单独有的业务逻辑。
4	专辑业务类	歌曲集合业务类的泛化，具有歌曲集合的基本功能，同时拓展了专辑独有的业务逻辑。
5	用户业务类	管理用户信息，包括注册、登录、个人信息查看和编辑等功能。
6	普通用户业务类	用户业务类的泛化，具有用户的基本功能，同时拓展了普通用户独有的业务逻辑。
7	歌手业务类	用户业务类的泛化，具有用户的基本功能，同时拓展了歌手

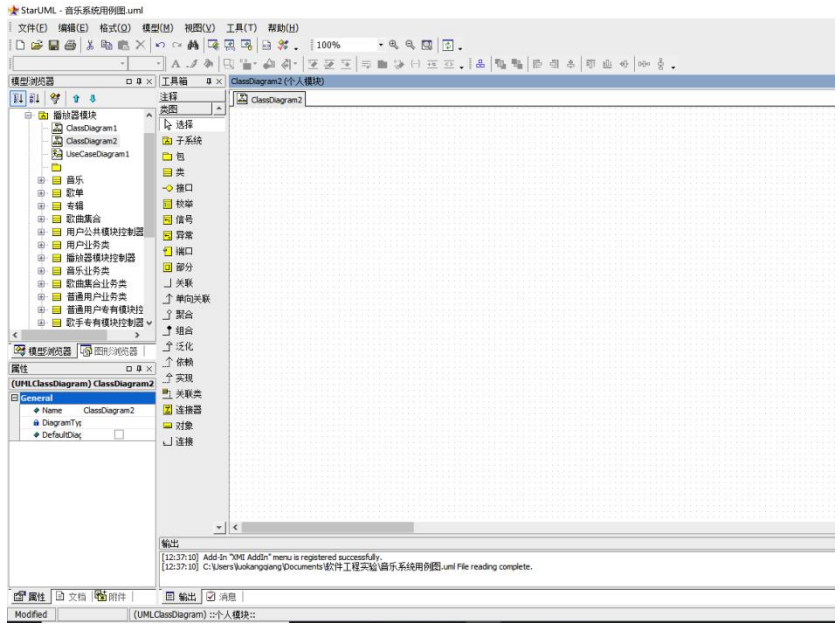
独有的业务逻辑。

2.使用 StarUML 建模工具绘制出类图.并描述和建立类间的关系。（包扩依赖关系、关联关系、组合关系、聚合关系以及泛化关系。）

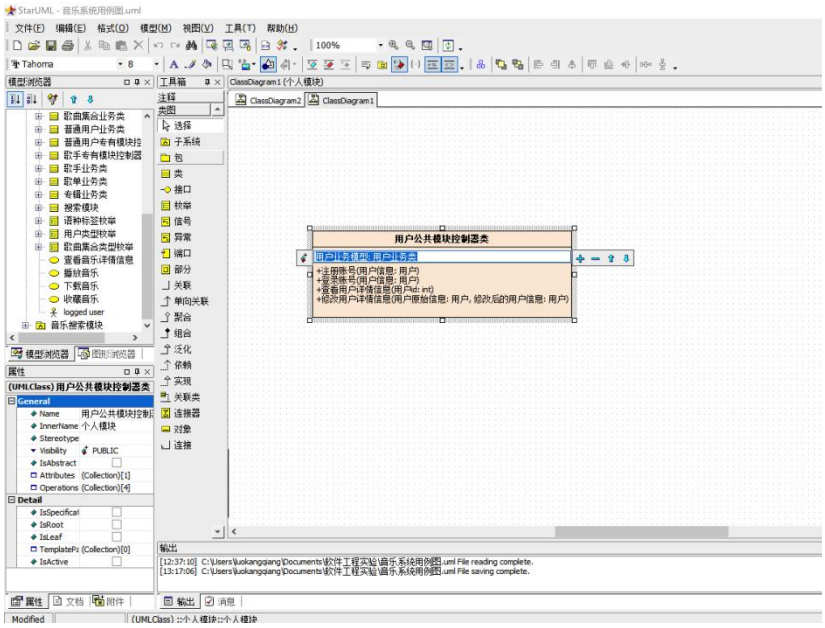
在如图的界面中选择类图。



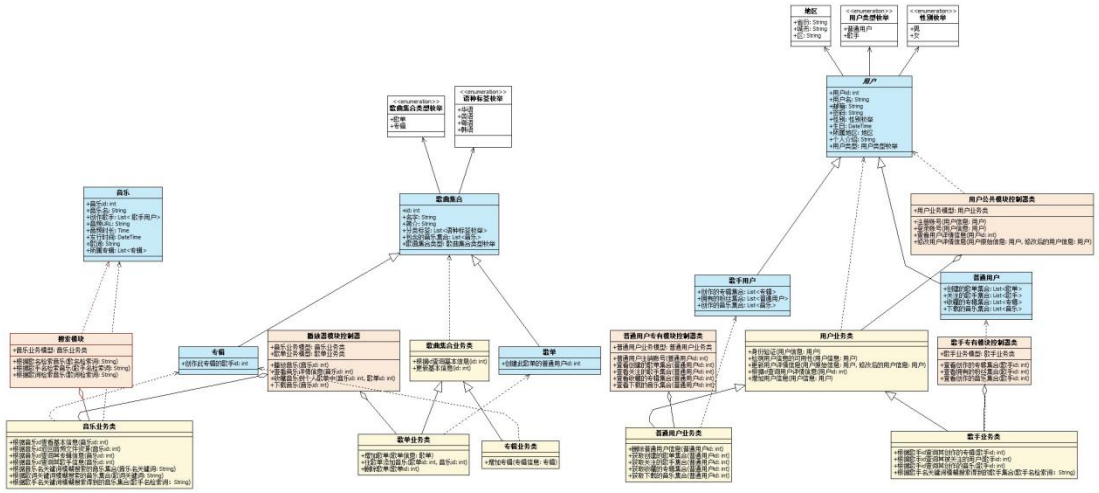
点击选中左侧的图标,并在工作窗口中合适位置点击放置并调节合适大小.



双击类图出现如下画面，可以修改和添加类属性（蓝色块）和类操作（橙色块）



3. 系统的总体类图及关系展示(包括依赖、关联、聚合、组合及泛化关系)



三、实验分析与讨论

在本次实验中，通过系统业务流程和功能的分析，成功识别了系统中的关键类和对象，并使用 StarUML 建模工具创建了相应的 UML 类图。该类图清晰地展示了系统的静态结构，包括实体类、控制器类和业务模型类之间的关系。实体类如音乐、歌曲集合、歌单等通过关联、泛化等关系建立了有效的联系，形成了系统的基本框架。控制器类负责处理不同模块的功能，而业务模型类则处理系统核心业务，共同构成了系统的关键组成部分。通过分析类图，我们可以清晰地理解系统中各类的职责和关系，为进一步的系统设计和开发奠定了基础。该类图的构建有助于提高系统的可维护性、可扩展性和复用性，为团队协作和系统维护提供了有力支持。整体而言，通过 UML 类图的建模过程，我们成功抽象和表达了系统的静态结构，为后续的系统实现提供了指导和参考。