编程规范

本次迭代本小组使用了react，spring boot，mySQL，webSocket等等框架，使用了Java，javascript，sql等编程语言，编程规范如下：

Java 编程规范：

1. 命名规范：

- 使用驼峰式命名法（camelCase）来命名变量、方法和类名。

- 类名使用名词或名词短语，并使用大写字母开头，如`UserController`。

- 方法名使用动词或动词短语，并使用小写字母开头，如`getUserById`。

- 变量名使用有意义的名词，并使用小写字母开头，如`userName`。

- 常量名使用全大写字母，使用下划线（\_）分隔单词，如`MAX\_RETRY\_COUNT`。

1. 代码格式化：

- 使用4个空格进行缩进。

- 使用适当的空格和空行来提高可读性，如在方法之间留一个空行。

- 使用大括号（{ }）在类、方法和控制流结构中创建块。

1. 注释规范：

- 使用注释来解释代码的目的、实现思路和非显而易见的部分。

- 使用Javadoc注释来描述公共接口、类和方法，并包含参数、返回值和可能的异常情况。

1. 异常处理：

- 避免捕获所有异常的通用异常块，而是针对特定的异常类型进行处理。

- 在异常处理中包含有意义的日志记录，以便于故障排除。

1. 单元测试：

- 使用适当的单元测试框架（如JUnit）编写测试用例，覆盖代码的各个方面和边界情况。

JavaScript 编程规范

1.代码风格

- 采用驼峰式命名法，变量名和方法名首字母小写，构造函数首字母大写。

- 使用分号作为语句结束符。

- 使用单引号或者反引号表示字符串，不要使用双引号。

- 缩进采用4个空格，不要使用Tab键。

- 使用ES6的箭头函数和let、const关键字。

2.变量和函数

- 使用const关键字定义不变的变量，使用let关键字定义可变的变量。

- 在函数声明和函数调用之间留有空格。

- 在定义函数时，不要使用匿名函数，尽可能使用命名函数。

- 在函数调用时，参数之间使用逗号分隔。

3.异常处理

- 不要在异步函数中使用try-catch语句，使用Promise.catch()方法捕获异常。

- 在catch块中，使用console.error()输出错误信息。

4.代码组织

- 将相关的变量和函数封装在一个对象中。

- 在对象字面量中，使用简洁的方法定义函数，而不是使用函数表达式。

- 将相关的模块放在一个独立的文件中，并使用模块导出和导入。

- 使用模块化的方式组织代码，避免全局变量的污染。

5.异步编程

- 避免在循环中使用异步操作，确保异步操作按顺序执行。

6. 格式化和注释

- 使用一致的缩进和换行风格，增加代码的可读性。

- 使用有意义的变量和函数命名，避免使用单个字符作为名称。

- 使用注释解释代码的功能、目的和注意事项。

7. 测试和调试

- 使用断言库或单元测试框架编写测试用例，确保代码的正确性。

- 使用调试工具（如浏览器的开发者工具）进行代码调试和性能分析。

MySQL 编程规范

1. 命名规范

- 数据库、表、列的命名应具有描述性，避免使用含糊不清或缩写的名称。

- 使用小写字母和下划线来命名数据库、表、列等对象，避免使用特殊字符或空格。

- 避免使用 MySQL 保留字作为对象名称，或者使用反引号（`）来转义保留字。

2. 数据类型和长度

- 根据数据的特性选择合适的数据类型，避免过度或不足的数据类型。

- 对于字符串类型，根据实际需要选择合适的长度，避免过长或过短的字段长度。

3. 主键和索引

- 每个表应该有一个主键，用于唯一标识每一行数据。

- 主键应该是简洁、稳定和不可更改的，避免使用易变的列作为主键。

- 根据查询的需求和频率创建适当的索引，避免过多或不必要的索引。

1. SQL 查询

- 缩进和格式化 SQL 查询，增加可读性。

- 避免在查询中使用 SELECT \*，而是明确列出所需的列。

- 在查询中使用 JOIN 操作来关联多个表，避免使用子查询嵌套过深。

1. SQL 语句

- 使用参数化查询（Prepared Statements）或存储过程来防止 SQL 注入攻击。

- 避免在 SQL 语句中直接拼接输入的变量，而是使用绑定参数的方式。

- 在 SQL 语句中使用事务来确保数据的一致性和完整性。

6. 注释和文档

- 使用注释解释 SQL 查询的目的、逻辑和关键步骤。

- 对于复杂的查询或存储过程，编写相应的文档，帮助他人理解和维护代码。