1. Ответить на вопросы (письменно):

# Как организовать клиент-серверное взаимодействие?

Для этого есть следующие варианты: встроенные пакеты java.io с использованием ServerSocket и Socket(на клиенте) или пакетом java.nio, который использует ServerSocketChannel и SocketChannel(на клиенте).

Первый вариант простой, но из-за лока потока, не подходит для передачи больших файлов (т.к. на время их передачи ничего нельзя сделать)

Второй гибкий, и с помощью буферов дает возможность не лочится на передаче файлов.

# Как и в каком виде передавать файлы?

В байт коде, переводим в байты, передаем и на клиенте собираем обратно.

# Как пересылать большие файлы?

Разбиваем на байты, кладем в буфер и передаем, на другом конце также используем буфер. В котором проверяем, что все что пришло, что отправлялось. И потом собираем в файл обратно.

# Как пересылать служебные команды?

В java.io нужно выкручиваться. Один из возможных вариантов, разбивать отправку файла в несколько последовательных посылок, в начале послания передавать команду начинающуюся со спец символа. Например, /filename, /filetype, /filecontent и далее через пробел посылать название или содержание файла. На втором конце ждать посылку и читать первые символы, чтобы определить, что делать с байтами.

В java.nio.file через Files.newOutputStream(), у которого есть параметр в котором можно передать команду

# Что хранить в базе данных?

В базе можно хранить, данные о пользователе и его файлах.

# Как передавать структуру каталогов/файлов?

В java.io.File через class FileInputStream в параметре имени файла, если нужен путь к файлу, так-же его туда.

В java.nio.file через class Files и его методы.

# Какую библиотеку использовать для сетевого взаимодействия: java.io, java.nio, Netty?

Netty, т.к. это NIO клиент-серверный фреймворк, он уже использует java.nio, который уже в свою очередь превосходит по готовому функционалу java.io.