Экземпляр класса Book (Книга) содержит поля String : название, автор, издательство ; Int : Id(уникальный идентификатор), кол-во страниц, общее кол-во экз. в библиотеке, кол-во в наличии (кроме тех, что выданы читателям), список (int) Id (внешних ключей) «активных», т.е. невозвращенных заказов(выдач) данной книги.

Экземпляр класса Listener (Читатель) содержит поля String : Имя, фамилия, адрес, телефон; Boolean : пол; список (int) Id (внешних ключей) «активных», т.е. невозвращенных заказов(выдач) данного читателя ; Int : Id(уникальный идентификатор.

Экземпляр класса Order(выдача, заказ) содержит поля ; Int : Id(уникальный идентификатор, String : комментарий при выдаче заказа, комментарий при возвращении (если оно состоялось), внешние ключи Id книги и Id Читателя.

Экземпляр класса Library (Библиотека) содержит поля список<Book> книг, список<Listener> читателей, список<Order> «активных»(невозвращенных) заказов-выдач, список <Order> «закрытых» (возвращенных) заказов-выдач, int : max Id по всем спискам, id(уникальный идентификатор , String : название, путь к файлу, лог внесенных изменений : выдачи, возвраты, добавление/удаление/редактирование книг/читателей.

Экземпляр класса ListLibrary(список библиотек) содержит список <Note> названий, id(уникальных идентификаторов, путей к файлам с данными конкретных библиотек, int : max id списка доступных библиотек.

Пункты меню организованы наследниками интерфейса Command с методами : сообщить название и выполнить. Пункты добавляются или исчезают из меню в зависимости от состояния объекта Library. Т.е. если, например, заводится новая библиотека, где пока не добавлены книги/читатели – будут доступны только пункты – добавить. Если существует хотя-бы один читатель/книга – появятся пункты : удалить/редактировать/дать подр.информацию/вывести список – сортируется по фамилиям+именам читателей или названиям книг в соответствующих «под-меню». Если существует и книга и читатель в основном меню библиотеки появится пункт – выдать книгу. Если есть хотя-бы одна выдача – появится пункт «вернуть книгу».

При запуске программы можно воспользоваться существующим шаблоном HandCreate (заполнены две библиотеки) для тестирования или прочитать из файла (пока там сохранен тот-же шаблон).

Добавить читателя/книгу, если такие уже есть в реестре/каталоге данной библиотеки не получится. В случае с книгой программа предложит объединить, т.е. увеличить кол-во экз. (общее по данной книге и в наличии(не выданное)). Удалить из каталога книгу/читателя не получится если есть выданные и невозвращенные экземпляры книги или у данного читателя есть что-то невозвращенное. Программа сообщит Id незакрытых заказов по книге/читателю. При редактировании читателя/книги в полях, доступных для редактирования можно оставлять прежние значения, нажав ENTER при пустой строке(пропустить). В остальных случаях пустую строку ввести не получится. Пользовательский ввод обрабатывается на предмет корректных Id при выборе и существующих пунктов меню. Выдать книгу читателю не получится если она уже у него есть в списке невозвращенных (активных) заказов или закончилась в наличии(все экз. на руках других читателей). Удалить (утилизировать) 1 экз. книги не получится, если все экземпляры на руках. При возвращении книги, заказ перемещается из списка «активных» заказов в список «закрытых заказов»

При получении подр. информации о книге/читателе, выводится еще и список завершенных (ранее возвращенных) заказов с комментариями при выдаче и возврате.

Логирование изменений при редактировании осуществляются детализировано – по измененным полям. При выходе из данной библиотеки, выводится лог всех совершенных изменений и запрос на сохранение в файл.

При удалении библиотеки – файл с данными тоже удаляется.

Программа написана в соответствии с MVP, т.е. в классе GUI ViewModels логики нет, только printLn и scanner.nextLine, для последующего «прикручивания» swing.

Метод «слияния» библиотек пока не написан – не было времени обдумать очередность замены внешних и уникальных ключей по всем спискам и проработать вопрос дублирования.