Peter the Great St. Petersburg Polytechnic University
Higher School of Intelligent Systems and Supercomputer Technologies

WEB APPLICATION PROGRAMMING

«Books Library Database System»

Project executed — Tatiana Berestova
By students gr.3530903/80301 — Maria Kurashkina

Saint Petersburg
2021

CONTENTS

1. Motivation

This web-application was created as a task for the course Web Application Programming to apply practically the knowledge brought in the course. The task required a relational database in 3rd normal form. Such sort of database is nicely suitable for collections, storages, libraries, etc. We decided to describe a book library in such a way, adding the information about them, also about their authors and publishing houses.

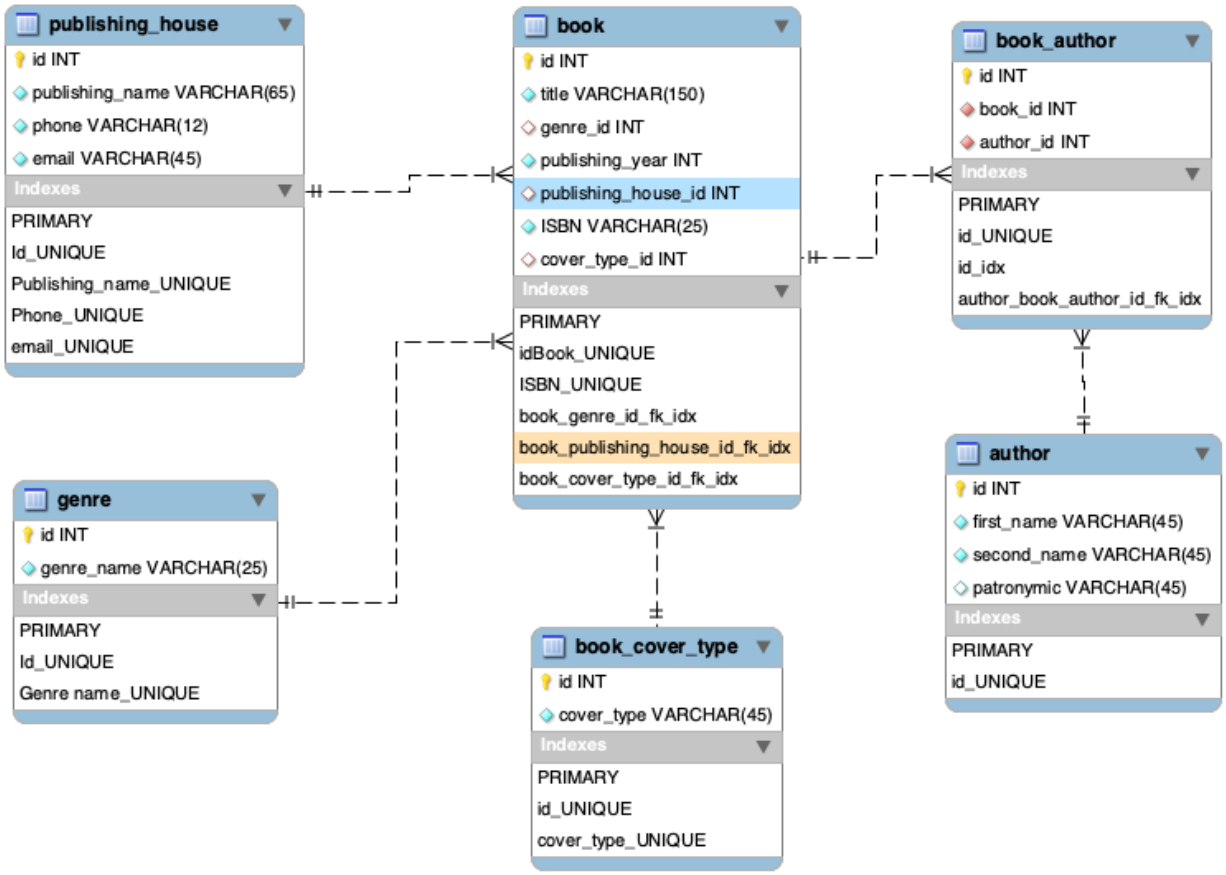
2. Installation

To make use of our books library one needs to install the open source Java servlet container and server called Apache Tomcat (https://tomcat.apache.org/), mySQL Database management system (https://www.mysql.com/), package for Java programming Java Development Kit (JDK) (https://www.oracle.com/java/technologies/downloads/). With the help of this tool one can run a MySQL and Apache Tomcat server which are necessary for the client/server communication and the database queries.

After installing Tomcat one needs to copy out *gui* and *interface* folder into a superfolder «BooksLibrary». This folder has to be copied into the folder «apache-tomcat/webapps». The database has to be imported (for example, using MySQLWorkbench) to the mysql server with the file named «database_books_dump.sql». After this is done one can access the library via the address http://localhost:8080/BooksLibrary/gui/ in the browser.

3. Database

3.1. Schema



3.2. Relation

1) book
   id: primary key
   title: the title of the book
   genre_id: foreign key for the *genre* table
   publishing_year: year when book was published
   publishing_house_id: foreign key for the *publishing_house* table
   ISBN: international standard book number
   cover_type_id: foreign key for the *book_cover_type* table

2) genre
   id: primary key
   genre_name: genre of the book (novel, horror, fantasy, etc)

3) publishing_house
   id: primary key
   publishing_name: name of the publishing house
   email: email of the publishing house

phone: phone number of the publishing house

4) book_cover_type
   id: primary key
   cover_type: cover type of the book (hardcover or paperback)

5) author
   id: primary key
   first_name: first name of the author
   second_name: second name of the author
   patronymic: patronymic of the author

6) book_author
   id: primary key
   book_id: foreign key for the *book* table
   author_id: foreign key for the *author* table

4. Implementation

4.1. Server-Side

On the server side we created twelve different .jsp files which are responsible for the data flow between client and server. In those files the requests to the database are processed in different ways. The files can handle GET and POST requests and have different modes of operating like ‘show’, ‘new’ and ‘delete’. For each of those modes a SQL command is being built together according to the received parameters and then sent to the database. This is then returned to the javascript (client side) in a json format.

Request	Response
http://localhost:8080/BooksLibrary/interface/allBooks.jsp	[{"CoverType":"paperback","Year":"2017","Isbn":" 978-5-9925-1254-0","Authors":"Mikhail Yuryevich Lermontov","Title":"A Hero of Our Time","PublishingHouse":"AST","ID":"38","Genre":"novel"},{"CoverType":"paperback","Year":"2015","Isbn":"978-5-17-093121-7","Authors":"Alexander Sergeevich Pushkin","Title":"Eugene

	Onegin","PublishingHouse":"AST","ID":"2","Genre":"novel"},{"CoverType":"hardcover","Year":"2019","Isbn":"978-5-17-088216-8","Authors":"Ray Bradbury","Title":"Fahrenheit 451","PublishingHouse":"AST","ID":"1","Genre":"dystopian novel"},{"CoverType":"paperback","Year":"2020","Isbn":"978-5-389-06475-1","Authors":"William Shakespeare","Title":"Hamlet","PublishingHouse":"Azbooka","ID":"3","Genre":"tragedy"},{"CoverType":"paperback","Year":"2020","Isbn":"978-5-17-121600-9","Authors":"William Shakespeare","Title":"Romeo and Juliet","PublishingHouse":"AST","ID":"24","Genre":"tragedy"},{"CoverType":"paperback","Year":"2001","Isbn":"0-13-022616-5","Authors":"Xuedong Huang, Alex Acero, Hsiao-Wuen Hon","Title":"Spoken language processing: a guide to theory, algorithm, and system development","PublishingHouse":"PrenticeoHall PTR","ID":"37","Genre":"scientific journalistic"},{"CoverType":"paperback","Year":"2018","Isbn":"978-5-9925-1341-7","Authors":"Alexander Sergeevich Pushkin","Title":"The Bronze Horseman","PublishingHouse":"Litera","ID":"26","Genre":"poem"},{"CoverType":"paperback","Year":"2021","Isbn":"978-1-52-937931-0","Authors":"Stephen King","Title":"The Mist","PublishingHouse":"HODDER & STOUGHTON","ID":"36","Genre":"horror"},{"CoverType":"paperback","Year":"2021","Isbn":"978-5-17-105782-4","Authors":"Alexander Sergeevich Pushkin","Title":"The Queen of Spades","PublishingHouse":"AST","ID":"5","Genre":"novella"}]
http://localhost:8080/BooksLibrary/interface/deleteAuthor.jsp?authorID=24	–
http://localhost:8080/BooksLibrary/interface/addAuthor.jsp?FN	–

=Nikolai&SN=Gogol&P=Vasilyevich	

4.2. Client-Side

The client side of the application consists of three files. First index.html which is responsible for the basic structure of the page, then style.css which handles all the styling and then application.js processes all the dynamic changes on the page and sends/receives the Ajax to/from the server.

5. Files

5.1. SQL dump of the database

```
CREATE DATABASE  IF NOT EXISTS `books` /*!40100 DEFAULT CHARACTER SET utf8 */ /*!80016 DEFAULT ENCRYPTION='N' */;
USE `books`;
-- MySQL dump 10.13  Distrib 8.0.26, for Win64 (x86_64)
--
-- Host: 127.0.0.1    Database: books
-- ------------------------------------------------------
-- Server version 8.0.26

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!50503 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `author`
--

DROP TABLE IF EXISTS `author`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
```

```
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `author` (
  `id` int NOT NULL AUTO_INCREMENT,
  `first_name` varchar(45) NOT NULL,
  `second_name` varchar(45) NOT NULL,
  `patronymic` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `id_UNIQUE` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=24 DEFAULT CHARSET=utf8mb3;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `author`
--

LOCK TABLES `author` WRITE;
/*!40000 ALTER TABLE `author` DISABLE KEYS */;
INSERT INTO `author` VALUES
(1,'Ray','Bradbury',''),(2,'Alexander','Pushkin','Sergeevich'),(4,'Mikhail','Bulgakov','Afanasevich'),(12,'Nikolai','Gogol','V
asilyevich'),(13,'Mikhail','Lermontov','Yuryevich'),(14,'Stephen','King',''),(15,'Xuedong','Huang',''),(16,'Alex','Acero',''),(1
7,'Hsiao-Wuen','Hon',''),(23,'William','Shakespeare','');
/*!40000 ALTER TABLE `author` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `book`
--

DROP TABLE IF EXISTS `book`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `book` (
  `id` int NOT NULL AUTO_INCREMENT,
  `title` varchar(150) NOT NULL,
  `genre_id` int DEFAULT NULL,
  `publishing_year` int NOT NULL,
  `publishing_house_id` int DEFAULT NULL,
  `ISBN` varchar(25) NOT NULL,
  `cover_type_id` int DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `idBook_UNIQUE` (`id`),
```

```
  UNIQUE KEY `ISBN_UNIQUE` (`ISBN`),
  KEY `book_genre_id_fk_idx` (`genre_id`),
  KEY `book_publishing_house_id_fk_idx` (`publishing_house_id`),
  KEY `book_cover_type_id_fk_idx` (`cover_type_id`),
  CONSTRAINT `book_cover_type_fk` FOREIGN KEY (`cover_type_id`) REFERENCES `book_cover_type` (`id`) ON
DELETE SET NULL,
  CONSTRAINT `book_genre_id_fk` FOREIGN KEY (`genre_id`) REFERENCES `genre` (`id`) ON DELETE SET
NULL,
  CONSTRAINT `book_publishing_house_id_fk` FOREIGN KEY (`publishing_house_id`) REFERENCES
`publishing_house` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=46 DEFAULT CHARSET=utf8mb3;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `book`
--

LOCK TABLES `book` WRITE;
/*!40000 ALTER TABLE `book` DISABLE KEYS */;
INSERT INTO `book` VALUES (1,'Fahrenheit 451',7,2019,4,'978-5-17-088216-8',1),(2,'Eugene
Onegin',3,2015,4,'978-5-17-093121-7',2),(3,'Hamlet',8,2020,5,'978-5-389-06475-1',2),(5,'The Queen of
Spades',6,2021,4,'978-5-17-105782-4',2),(24,'Romeo and Juliet',8,2020,4,'978-5-17-121600-9',2),(26,'The Bronze
Horseman',10,2018,2,'978-5-9925-1341-7',2),(36,'The Mist',9,2021,9,'978-1-52-937931-0',2),(37,'Spoken language
processing: a guide to theory, algorithm, and system development',13,2001,10,'0-13-022616-5',2),(38,'A Hero of Our
Time',3,2017,4,' 978-5-9925-1254-0',2);
/*!40000 ALTER TABLE `book` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `book_author`
--

DROP TABLE IF EXISTS `book_author`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `book_author` (
  `id` int NOT NULL AUTO_INCREMENT,
  `book_id` int NOT NULL,
  `author_id` int NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `id_UNIQUE` (`id`),
```

```
  KEY `id_idx` (`book_id`),
  KEY `author_book_author_id_fk_idx` (`author_id`),
  CONSTRAINT `author_book_author_id_fk` FOREIGN KEY (`author_id`) REFERENCES `author` (`id`) ON DELETE
CASCADE,
  CONSTRAINT `author_book_book_id_fk` FOREIGN KEY (`book_id`) REFERENCES `book` (`id`) ON DELETE
CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=53 DEFAULT CHARSET=utf8mb3;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `book_author`
--

LOCK TABLES `book_author` WRITE;
/*!40000 ALTER TABLE `book_author` DISABLE KEYS */;
INSERT INTO `book_author` VALUES
(1,1,1),(3,2,2),(4,5,2),(32,26,2),(34,36,14),(35,37,15),(36,37,16),(37,37,17),(40,38,13),(51,24,23),(52,3,23);
/*!40000 ALTER TABLE `book_author` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `book_cover_type`
--

DROP TABLE IF EXISTS `book_cover_type`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `book_cover_type` (
  `id` int NOT NULL AUTO_INCREMENT,
  `cover_type` varchar(45) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `id_UNIQUE` (`id`),
  UNIQUE KEY `cover_type_UNIQUE` (`cover_type`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb3;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `book_cover_type`
--

LOCK TABLES `book_cover_type` WRITE;
```

```
/*!40000 ALTER TABLE `book_cover_type` DISABLE KEYS */;
INSERT INTO `book_cover_type` VALUES (1,'hardcover'),(2,'paperback');
/*!40000 ALTER TABLE `book_cover_type` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `genre`
--

DROP TABLE IF EXISTS `genre`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `genre` (
  `id` int NOT NULL AUTO_INCREMENT,
  `genre_name` varchar(25) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `Id_UNIQUE` (`id`),
  UNIQUE KEY `Genre name_UNIQUE` (`genre_name`)
) ENGINE=InnoDB AUTO_INCREMENT=15 DEFAULT CHARSET=utf8mb3;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `genre`
--

LOCK TABLES `genre` WRITE;
/*!40000 ALTER TABLE `genre` DISABLE KEYS */;
INSERT INTO `genre` VALUES (4,'adventure'),(2,'detective'),(7,'dystopian
novel'),(5,'fantastic'),(1,'fantasy'),(9,'horror'),(11,'love story'),(3,'novel'),(6,'novella'),(10,'poem'),(13,'scientific
journalistic'),(8,'tragedy');
/*!40000 ALTER TABLE `genre` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `publishing_house`
--

DROP TABLE IF EXISTS `publishing_house`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `publishing_house` (
```

```
  `id` int NOT NULL AUTO_INCREMENT,
  `publishing_name` varchar(65) NOT NULL,
  `phone` varchar(12) NOT NULL,
  `email` varchar(45) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `Id_UNIQUE` (`id`),
  UNIQUE KEY `Publishing_name_UNIQUE` (`publishing_name`),
  UNIQUE KEY `Phone_UNIQUE` (`phone`),
  UNIQUE KEY `email_UNIQUE` (`email`)
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=utf8mb3;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `publishing_house`
--

LOCK TABLES `publishing_house` WRITE;
/*!40000 ALTER TABLE `publishing_house` DISABLE KEYS */;
INSERT INTO `publishing_house` VALUES
(1,'ROSMAN','84959337070','rosman@rosman.ru'),(2,'Litera','88124413648','sales@litera.spb.ru'),(3,'Drofa','8495795
0550','drofa@drofa.msk.ru'),(4,'AST','84952321625','zakaz@ast.ru'),(5,'Azbooka','88123270455','main@azbooka.spb
.ru'),(7,'Eksmo','84954116886','info@eksmo.ru'),(9,'HODDER &
STOUGHTON','442031226000','enquiries@hachette.co.uk'),(10,'PrenticeoHall
PTR','8003823419','corpsales@prenhall.com');
/*!40000 ALTER TABLE `publishing_house` ENABLE KEYS */;
UNLOCK TABLES;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2021-12-10 22:13:02
```

5.2. Server-Side

addAuthor.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
    pageEncoding="UTF-8"
    import="java.sql.*"
    import="org.json.*"%>
<%
try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}
        try {
            Connection connection_;
            String dbms = "jdbc:mysql://localhost:3306/books";
            connection_ = DriverManager.getConnection(dbms, "root", "root");
            Statement statement = connection_.createStatement();
            String FirstName = request.getParameter("FN");
            String SecondName = request.getParameter("SN");
            String Patronymic = request.getParameter("P");
            String sql = "INSERT INTO author(first_name, second_name, patronymic) "
            + "VALUES ('" + FirstName + "', '" + SecondName + "', '" + Patronymic + "')";
            statement.executeUpdate(sql);
        }
        catch(SQLException e) {
            e.printStackTrace();
            }
%>
```

addBook.jsp

```
<%@page contentType="application/json; charset=UTF-8" pageEncoding="UTF-8"%>
<%@page import="java.sql.*"%>
<%@page import="org.json.*"%>
<%

    try{
        Class.forName("com.mysql.jdbc.Driver");
    }
    catch(ClassNotFoundException exc){exc.printStackTrace();}

    try{
        Connection connection_;
```

```
        String dbms = "jdbc:mysql://localhost:3306/books";
        connection_ = DriverManager.getConnection(dbms, "root", "root");
        Statement statement = connection_.createStatement();
        String title_book = request.getParameter("title");
        String genre_id = request.getParameter("genre");
        String year = request.getParameter("year");
        String house_id = request.getParameter("house");
        String ISBN = request.getParameter("ISBN");
        String cover_id = request.getParameter("cover");
        int count = Integer.parseInt(request.getParameter("count"));
        String add_book_req = "INSERT INTO book(title, genre_id, publishing_year, publishing_house_id, ISBN,
cover_type_id) VALUES('" + title_book + "', " + genre_id + ", " + year + ", " + house_id + ", '" + ISBN + "', " + cover_id +
")";
        statement.executeUpdate(add_book_req);
        String get_id_req = "SELECT id FROM book WHERE ISBN = '" + ISBN + "'";
        ResultSet res_get_id_req = statement.executeQuery(get_id_req);
        int book_id = res_get_id_req.next()?res_get_id_req.getInt("id"):1;
        int i=0;
        while (i < count){
            String str = "author" + (i+1);
            String author_id = request.getParameter(str);
            String book_author_req = "INSERT INTO book_author(book_id, author_id) VALUES(" + book_id + ", " +
author_id + ")";
            statement.executeUpdate(book_author_req);
            i++;
        }
    } catch(SQLException e){
        e.printStackTrace();
}
%>
```

addGenre.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
    pageEncoding="UTF-8"
    import="java.sql.*"
    import="org.json.*"%>
<%
try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
```

```
    e.printStackTrace();
}
        try {
            Connection connection_;
            String dbms = "jdbc:mysql://localhost:3306/books";
            connection_ = DriverManager.getConnection(dbms, "root", "root");
            Statement statement = connection_.createStatement();
            String Genre = request.getParameter("genre");
            String sql = "INSERT INTO genre(genre_name) "
            + "VALUES ('" + Genre + "')";
            statement.executeUpdate(sql);
        }
        catch(SQLException e) {
            e.printStackTrace();
            }
        %>
```

addPublishingHouse.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
    pageEncoding="UTF-8"
    import="java.sql.*"
    import="org.json.*"%>
<%
try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}
        try {
            Connection connection_;
            String dbms = "jdbc:mysql://localhost:3306/books";
            connection_ = DriverManager.getConnection(dbms, "root", "root");
            Statement statement = connection_.createStatement();
            String Name = request.getParameter("name");
            String Email = request.getParameter("email");
            String Phone = request.getParameter("phone");
            String sql = "INSERT INTO publishing_house(publishing_name, phone, email) "
            + "VALUES ('" + Name + "', '" + Phone + "', '" + Email +"')";
            statement.executeUpdate(sql);
        }
```

```
        catch(SQLException e) {
            e.printStackTrace();
            }
        %>
```

allAuthors.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
    pageEncoding="UTF-8"
    import="java.sql.*"
    import="org.json.*"%>
<%
response.setHeader("Cache-Control", "no-cache");


try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}
        try {
            Connection connection_;
            String dbms = "jdbc:mysql://localhost:3306/books";
            connection_ = DriverManager.getConnection(dbms, "root", "root");
            Statement statement = connection_.createStatement();
            String sql = "SELECT author.id 'id', CONCAT(first_name, ' ', patronymic, ' ', second_name) author,
author.second_name 'second_name', GROUP_CONCAT(book.title SEPARATOR ', ') 'books' "
            + "FROM book JOIN book_author "
            + "ON book.id=book_author.book_id "
            + "RIGHT JOIN author ON book_author.author_id=author.id "
            + "GROUP BY author.id "
            + "ORDER BY(author.second_name)";
            ResultSet resTeams = statement.executeQuery(sql);
            JSONArray authors = new JSONArray();
            while(resTeams.next()) {
                JSONObject obj = new JSONObject();
                obj.put("ID", resTeams.getString("id"));
                obj.put("Author", resTeams.getString("author"));
                obj.put("Books", resTeams.getString("books"));
                authors.put(obj);
                }
            out.println(authors);
```

```
        }
        catch(SQLException e) {
            e.printStackTrace();
            }
        %>
```

allBooks.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
    pageEncoding="UTF-8"
    import="java.sql.*"
    import="org.json.*"%>
<%
response.setHeader("Cache-Control", "no-cache");


try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}
        try {
            Connection connection_;
            String dbms = "jdbc:mysql://localhost:3306/books";
            connection_ = DriverManager.getConnection(dbms, "root", "root");
            Statement statement = connection_.createStatement();
            String sql = "SELECT book.id 'id', book.title 'title', book.publishing_year 'year', genre.genre_name 'genre',
GROUP_CONCAT(concat_ws(' ', author.first_name, author.patronymic, author.second_name) SEPARATOR ', ')
'authors', book_cover_type.cover_type 'cover_type', publishing_house.publishing_name 'publishing_house',
book.ISBN 'ISBN' "
            + "FROM book JOIN genre "
            + "ON book.genre_id=genre.id "
            + "JOIN book_author ON book.id=book_author.book_id "
            + "JOIN author ON book_author.author_id=author.id "
            + "JOIN book_cover_type ON book.cover_type_id=book_cover_type.id "
            + "JOIN publishing_house ON book.publishing_house_id=publishing_house.id "
            + "GROUP BY book.id "
            + "ORDER BY(book.title)";
            ResultSet resTeams = statement.executeQuery(sql);
            JSONArray books = new JSONArray();
            while(resTeams.next()) {
                JSONObject obj = new JSONObject();
```

```
            obj.put("ID", resTeams.getString("id"));
            obj.put("Title", resTeams.getString("title"));
            obj.put("Year", resTeams.getString("year"));
            obj.put("Genre", resTeams.getString("genre"));
            obj.put("Authors", resTeams.getString("authors"));
            obj.put("CoverType", resTeams.getString("cover_type"));
            obj.put("PublishingHouse", resTeams.getString("publishing_house"));
            obj.put("Isbn", resTeams.getString("ISBN"));
            books.put(obj);
            }
         out.println(books);
      }
      catch(SQLException e) {
         e.printStackTrace();
         }
      %>
```

allPublHouses.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
   pageEncoding="UTF-8"
   import="java.sql.*"
   import="org.json.*"%>
<%
response.setHeader("Cache-Control", "no-cache");


try {
   Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
   e.printStackTrace();
}
      try {
         Connection connection_;
         String dbms = "jdbc:mysql://localhost:3306/books";
         connection_ = DriverManager.getConnection(dbms, "root", "root");
         Statement statement = connection_.createStatement();
         String sql = "SELECT publishing_house.id 'id', publishing_house.publishing_name 'name',
publishing_house.phone 'phone', publishing_house.email 'email', GROUP_CONCAT(book.title SEPARATOR ', ')
'books' "
         + "FROM book RIGHT JOIN publishing_house "
         + "ON book.publishing_house_id=publishing_house.id "
```

```
        + "GROUP BY publishing_house.id "
        + "ORDER BY(publishing_house.publishing_name)";
        ResultSet resTeams = statement.executeQuery(sql);
        JSONArray publHouses = new JSONArray();
        while(resTeams.next()) {
            JSONObject obj = new JSONObject();
            obj.put("ID", resTeams.getString("id"));
            obj.put("Name", resTeams.getString("name"));
            obj.put("Email", resTeams.getString("email"));
            obj.put("Phone", resTeams.getString("phone"));
            obj.put("Books", resTeams.getString("books"));
            publHouses.put(obj);
            }
        out.println(publHouses);
    }
    catch(SQLException e) {
        e.printStackTrace();
        }
    %>
```

book_info.jsp

```
<%@page contentType="application/json; charset=UTF-8" pageEncoding="UTF-8"%>
<%@page import="java.sql.*"%>
<%@page import="org.json.*"%>
<%

  try{
      Class.forName("com.mysql.jdbc.Driver");
  }
  catch(ClassNotFoundException exc){exc.printStackTrace();}

  try{
      Connection connection_;
      String dbms = "jdbc:mysql://localhost:3306/books";
      connection_ = DriverManager.getConnection(dbms, "root", "root");
      Statement statement = connection_.createStatement();

      String genres_req = "SELECT id, genre_name "
                  + "FROM genre";
      ResultSet res_genres_req = statement.executeQuery(genres_req);
```

```
JSONObject result = new JSONObject();
JSONArray genres = new JSONArray();
while(res_genres_req.next())
{
   JSONObject obj = new JSONObject();
   obj.put("genre_id", res_genres_req.getString("id"));
   obj.put("genre_name", res_genres_req.getString("genre_name"));
   genres.put(obj);
}
result.put("genre", genres);

String publ_house_req = "SELECT id, publishing_name "
                + "FROM publishing_house";
ResultSet res_publ_house_req = statement.executeQuery(publ_house_req);
JSONArray publ_houses = new JSONArray();
while(res_publ_house_req.next())
{
   JSONObject obj = new JSONObject();
   obj.put("publ_house_id", res_publ_house_req.getString("id"));
   obj.put("publ_house_name", res_publ_house_req.getString("publishing_name"));
   publ_houses.put(obj);
}
result.put("publishing_house", publ_houses);

String cover_type_req = "SELECT id, cover_type "
                + "FROM book_cover_type";
ResultSet res_cover_type_req = statement.executeQuery(cover_type_req);
JSONArray cover = new JSONArray();
while(res_cover_type_req.next())
{
   JSONObject obj = new JSONObject();
   obj.put("cover_id", res_cover_type_req.getString("id"));
   obj.put("cover_type", res_cover_type_req.getString("cover_type"));
   cover.put(obj);
}
result.put("cover", cover);

String authors_req = "SELECT id, CONCAT(first_name, ' ', patronymic, ' ', second_name) author FROM author";
ResultSet res_authors_req = statement.executeQuery(authors_req);
JSONArray author = new JSONArray();
while(res_authors_req.next())
```

```
        {
        JSONObject obj = new JSONObject();
        obj.put("author_id", res_authors_req.getString("id"));
        obj.put("author_name", res_authors_req.getString("author"));
        author.put(obj);
        }
        result.put("author", author);
        out.print(result);
    } finally {}
%>
```

deleteAuthor.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
    pageEncoding="UTF-8"
    import="java.sql.*"
    import="org.json.*"%>
<%
try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}
        try {
            Connection connection_;
            String dbms = "jdbc:mysql://localhost:3306/books";
            connection_ = DriverManager.getConnection(dbms, "root", "root");
            Statement statement = connection_.createStatement();
            String AuthorId = request.getParameter("authorID");
            String sql = "DELETE FROM author WHERE id =" + AuthorId;
            statement.executeUpdate(sql);
        }
        catch(SQLException e) {
            e.printStackTrace();
            }
        %>
```

deleteBook.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
    pageEncoding="UTF-8"
    import="java.sql.*"
```

```
    import="org.json.*"%>
<%
try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}
        try {
            Connection connection_;
            String dbms = "jdbc:mysql://localhost:3306/books";
            connection_ = DriverManager.getConnection(dbms, "root", "root");
            Statement statement = connection_.createStatement();
            String BookId = request.getParameter("bookID");
            String sql = "DELETE FROM book WHERE id =" + BookId;
            statement.executeUpdate(sql);
        }
        catch(SQLException e) {
            e.printStackTrace();
            }
        %>
```

deletePublHouse.jsp

```
<%@ page language="java" contentType="application/json; charset=UTF-8"
    pageEncoding="UTF-8"
    import="java.sql.*"
    import="org.json.*"%>
<%
try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}
        try {
            Connection connection_;
            String dbms = "jdbc:mysql://localhost:3306/books";
            connection_ = DriverManager.getConnection(dbms, "root", "root");
            Statement statement = connection_.createStatement();
            String PublHouseId = request.getParameter("publHouseID");
            String sql = "DELETE FROM publishing_house WHERE id =" + PublHouseId;
            statement.executeUpdate(sql);
```

```
        }
        catch(SQLException e) {
            e.printStackTrace();
            }
%>
```

freeAuthor.jsp

```
<%@page contentType="application/json; charset=UTF-8" pageEncoding="UTF-8"%>
<%@page import="java.sql.*"%>
<%@page import="org.json.*"%>
<%

    try{
        Class.forName("com.mysql.jdbc.Driver");
    }
    catch(ClassNotFoundException exc){exc.printStackTrace();}

    try{
        Connection connection_;
        String dbms = "jdbc:mysql://localhost:3306/books";
        connection_ = DriverManager.getConnection(dbms, "root", "root");
        Statement statement = connection_.createStatement();
        int count = Integer.parseInt(request.getParameter("count"));
        int i=0;
        String result = "(";
        while (i < count){
            String str = "author" + (i+1);
            if (i != count-1){
                result += request.getParameter(str) + ", ";
            } else{
                result += request.getParameter(str);
            }
            i++;
        }
        result += ")";
        String authors_req = "SELECT id, CONCAT(first_name, ' ', patronymic, ' ', second_name) author FROM author
WHERE id NOT IN" + result;
        ResultSet res_authors_req = statement.executeQuery(authors_req);
        JSONObject res = new JSONObject();
        JSONArray author = new JSONArray();
```

```
        while(res_authors_req.next())
        {
        JSONObject obj = new JSONObject();
        obj.put("author_id", res_authors_req.getString("id"));
        obj.put("author_name", res_authors_req.getString("author"));
        author.put(obj);
        }
        res.put("author", author);
        out.print(res);
    } finally {}
%>
```

5.3. Client-Side

index.html

```
<!DOCTYPE html>
<html>

<head>
    <meta http-equiv="content-type" content="text/html;charset=UTF-8" />
    <title>Books</title>
    <link href="style.css" media="screen" rel="stylesheet" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script type="text/javascript" src="application.js"></script>
</head>

<body onload="Books()">
    <div id="menu">
        <div id="nav" class="over">
            <li align="center"><b><a href="javascript:Books()">Books Overview</a></b></li>
            <li align="center"><b><a href="javascript:Authors()">Authors Overview</a></b></li>
            <li align="center"><b><a href="javascript:PublHouses()">Publishing Houses Overview</a></b></li>
        </div>
        <div id="nav" class="add">
            <li align="center"><b><a href="javascript:addBook()">Add new Book</a></b></li>
            <li align="center"><b><a href="javascript:addAuthor()">Add new Author</a></b></li>
            <li align="center"><b><a href="javascript:addGenre()">Add new Genre</a></b></li>
            <li align="center"><b><a href="javascript:addPublHouse()">Add new Publishing House</a></b></li>
        </div>
    </div>
    <div id="container"></div>
```

```
</body>
```

style.css

```
#nav {
    margin: 0.97%;
}

#nav li {
    margin: auto;
    display: inline-block;
}

.over li {
    width: 20%;
}

.add li {
    width: 19%;
}

#nav li a {
    text-decoration: none;
    color: #101010;
}

#nav li a:hover {
    text-decoration: underline;
}

body {
    background: url("background.jpg") no-repeat center center fixed;
    background-size: cover;
    font-family: arial;
    background-color: #e0e0e0;
    font-size: 14px;
    color: #404040;
    padding: 20px;
    margin: 10px 0px;
    text-align: center;
}
```

```
#menu {
    display: flex;
    flex-direction: column;
    width: 70%;
    height: 80px;
    margin: 0.5% auto;
    font-size: 14px;
    background-color: #c6c5c5;
    border: 6px solid #000000;
    border-radius: 10px;
}

a {
    text-decoration: none;
    color: #404040;
}

a:hover {
    text-decoration: underline;
}

#container {
    text-align: left;
    vertical-align: middle;
    margin: 0px auto;
    padding: 10px;
    width: 70%;
    background-color: #ffffff;
    border: 12px solid #000000;
    border-radius: 10px;
}

table.books,
table.authors,
table.publhouses {
    width: 100%;
    border-spacing: 2px;
    table-layout: autho;
}
```

```
.books tr:nth-child(2n), .authors tr:nth-child(2n), .publhouses tr:nth-child(2n) {
    background-color: #f0f0f0
}
td,
th {
    padding: 3px;
}

.button_add {
    align-items: center;
    border-radius: 4px;
    border-width: 0;
    background-color: #c6c5c5;
    cursor: pointer;
    height: 30px;
    padding-left: 16px;
    padding-right: 16px;
    text-align: left;
    text-decoration: none;
    font-size: 14px;
}

#book_title {
    width: 300%;
}
```

application.js

```
function Books() {
    var text = '<tr><th>Title</th><th>Publishing Year</th><th>Genre</th><th>Cover Type</th><th>Publishing
House</th><th>Authors</th><th>ISBN</th></tr>';
    var fields = ['Title', 'Year', 'Genre', 'CoverType', 'PublishingHouse', 'Authors', 'Isbn'];
    ShowForm('Books Library', 'books', text, '../interface/allBooks.jsp', 'Book', fields);
}

function ShowForm(header_text, table_class, table_header, page_jsp, delete_func_name, fields) {
    header = document.createElement('h1');
    header.innerHTML = header_text;
    table = document.createElement('table');
    table.setAttribute('class', table_class);
    table.innerHTML = table_header;
```

```
    var c = document.getElementById('container');
    c.innerHTML = '';
    c.appendChild(header);
    c.appendChild(table);
    var xhr = new XMLHttpRequest();
    xhr.open('GET', page_jsp);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            var resp = xhr.responseText;
            var myObject = eval('(' + resp + ')');
            myObject.forEach(value => {
                row = document.createElement('tr');
                row.innerHTML = AddRow(value, fields);
                edit = document.createElement('td');
                edit.innerHTML = '<a href="javascript:delete' + delete_func_name + '(' + value.ID + ')">Delete</a>';
                row.appendChild(edit);
                table.firstChild.appendChild(row);
            });
        }
    }
    xhr.send();
}

function AddRow(value, fields) {
    var res = "";
    for (let i = 0; i < fields.length; i++) {
        if (fields[i] == 'Books') {
            var books = value[fields[i]];
            if (books == undefined) {
                books = "";
            }
            res += '<th>' + books + '</th>';
        } else {
            res += '<th>' + value[fields[i]] + '</th>';
        }
    }
    return res;
}

function Authors() {
    var text = '<tr><th>Name</th><th>Books in Library</th></tr>';
```

```
    var fields = ['Author', 'Books'];
    ShowForm('Authors', 'authors', text, '../interface/allAuthors.jsp', 'Author', fields);
}

function PublHouses() {
    var text = '<tr><th>Publishing House Name</th><th>Phone Number</th><th>Email</th><th>Books in
Library</th></tr>';
    var fields = ['Name', 'Phone', 'Email', 'Books'];
    ShowForm('Publishing Houses', 'publhouses', text, '../interface/allPublHouses.jsp', 'PublHouse', fields);
}

function deleteBook(id) {
    Delete('book', 'bookID=', '../interface/deleteBook.jsp', Books, id);
}

function deleteAuthor(id) {
    Delete('author', 'authorID=', '../interface/deleteAuthor.jsp', Authors, id);
}

function deletePublHouse(id) {
    Delete('publishing house', 'publHouseID=', '../interface/deletePublHouse.jsp', PublHouses, id);
}

function Delete(confirm_text, param, jsp_page, nex_func, id) {
    if (confirm("Do you really want to delete this " + confirm_text + "?")) {
        var xhr = new XMLHttpRequest();
        var body = param + encodeURIComponent(id);
        xhr.open('POST', jsp_page);
        xhr.onreadystatechange = function() {
            if (xhr.readyState == 4) {
                nex_func();
            }
        }
        xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
        xhr.send(body);
    }
}

function addBook() {
    CreateForm('Book', 'add_book')
    CreateTextInput("Title", "book_title", "table_add_book");
```

```
    var xhr = new XMLHttpRequest();
    xhr.open('GET', '../interface/book_info.jsp');
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            var resp = xhr.responseText;
            var myObject = eval('(' + resp + ')');
            CreateSelectInput("genre", "Genre", myObject.genre, "genre_id", "genre_name", "table_add_book");
            CreateTextInput("Publishing Year", "publish_year", "table_add_book");
            CreateSelectInput("publish_house", "Publishing House", myObject.publishing_house, "publ_house_id",
"publ_house_name", "table_add_book");
            CreateTextInput("ISBN", "ISBN", "table_add_book");
            CreateSelectInput("cover", "Cover Type", myObject.cover, "cover_id", "cover_type", "table_add_book");
            CreateFormButton('button_add_book', 'add_book_button()', 'table_add_book');
            SelectorAdd("Author", myObject.author, "author_id", "author_name", "table_add_book", "+", 'newSelector()');

        }
    }
    xhr.send();
}

function SelectorAdd(header, val, id, name, table, add_remove, click) {
    var form = document.getElementById(table);
    var line = document.createElement('tr');
    var row1 = document.createElement('td');
    row1.innerHTML = header;
    line.appendChild(row1);
    var row2 = document.createElement('td');
    var elem = document.createElement('select');
    elem.setAttribute('class', 'authors');
    var res = "";
    val.forEach(value => {
        res += "<option value=" + value[id] + ">" + value[name] + "</option>";
    });
    var row3 = document.createElement('td');
    var button_add = document.createElement('input');
    button_add.setAttribute('type', 'button');
    button_add.setAttribute('value', add_remove);
    button_add.setAttribute('onClick', click);
    elem.innerHTML = res;
    row2.appendChild(elem);
    row3.appendChild(button_add);
```

```
    line.appendChild(row2);
    line.appendChild(row3);
    tr = document.getElementsByTagName('tr');
    tr1 = tr[tr.length - 1];
    tr1.before(line);

}

function newSelector() {
    var elem = document.getElementsByClassName('authors');
    var indx = elem.length;;
    var str = "";
    for (let i = 1; i < indx + 1; i++) {
        if (i != indx) {
            str += "author" + i + "=" + encodeURIComponent(elem[i - 1].value) + "&";
        } else {
            str += "count=" + indx + "&author" + i + "=" + encodeURIComponent(elem[i - 1].value);
        }
    }
    var xhr = new XMLHttpRequest();
    xhr.open('POST', '../interface/freeAuthor.jsp');
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            var resp = xhr.responseText;
            var myObject = eval('(' + resp + ')');
            SelectorAdd("Author", myObject.author, "author_id", "author_name", "table_add_book", "-",
'delSelector(this.parentNode.parentNode.rowIndex)');
            for (let i = 0; i < indx; i++) {
                elem[i].setAttribute("disabled", true);
            }
        }
    }
    xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
    xhr.send(str);

}

function delSelector(idx) {
    var count = document.getElementsByClassName('authors').length;
    var elems = document.getElementsByTagName('tr');
    var authors = document.getElementsByClassName('authors');
```

```
    var count1 = elems.length;
    var table = document.getElementById('table_add_book');
    var sel = elems[idx];
    if (idx == count1 - 2) {
        id = count - 2;
        authors[id].disabled = false;
    }
    table.removeChild(sel);
}

function addAuthor() {
    CreateForm('Author', 'add_author');
    CreateTextInput("First Name", "first_name", "table_add_author", "30");
    CreateTextInput("Second Name", "second_name", "table_add_author", "40");
    CreateTextInput("Patronymic", "patr", "table_add_author", "30");
    CreateFormButton('button_add_author', 'add_author_button()', 'table_add_author');
}

function addGenre() {
    CreateForm('Genre', 'add_genre');
    CreateTextInput("Genre Name", "genre_name", "table_add_genre", "30");
    CreateFormButton('button_add_genre', 'add_genre_button()', 'table_add_genre');
}

function addPublHouse() {
    CreateForm('Publishing House', 'add_publ_house');
    CreateTextInput("Publishing Name", "publ_name", "table_add_publ_house", "40");
    CreateTextInput("Phone", "publ_phone", "table_add_publ_house", "25");
    CreateTextInput("Email", "publ_email", "table_add_publ_house", "30");
    CreateFormButton('button_add_publ_house', 'add_publ_house_button()', 'table_add_publ_house');
}

function add_book_button() {
    var authors = document.getElementsByClassName('authors');
    count = authors.length;
    res = "&count=" + count;
    for (let i = 0; i < count; i++) {
        id = i + 1;
        res += "&author" + id + "=" + encodeURIComponent(authors[i].value);
    }
    var xhr = new XMLHttpRequest();
```

```
    var body = "title=" + encodeURIComponent(document.getElementById("book_title").value) + "&genre=" +
encodeURIComponent(document.getElementById("genre").value) + "&year=" +
encodeURIComponent(document.getElementById("publish_year").value) +
        "&house=" + encodeURIComponent(document.getElementById("publish_house").value) + "&ISBN=" +
encodeURIComponent(document.getElementById("ISBN").value) + "&cover=" +
encodeURIComponent(document.getElementById("cover").value) + res;
    xhr.open('POST', '../interface/addBook.jsp');
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            Books();
        }
    }
    xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
    xhr.send(body);
}

function add_author_button() {
    var xhr = new XMLHttpRequest();
    var body = "FN=" + encodeURIComponent(document.getElementById("first_name").value) + "&SN=" +
encodeURIComponent(document.getElementById("second_name").value) + "&P=" +
encodeURIComponent(document.getElementById("patr").value);
    xhr.open('POST', '../interface/addAuthor.jsp');
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            Authors();
        }
    }
    xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
    xhr.send(body);
}

function add_genre_button() {
    var xhr = new XMLHttpRequest();
    var body = "genre=" + encodeURIComponent(document.getElementById("genre_name").value);
    xhr.open('POST', 'interface/addGenre.jsp');
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            alert("You have just added new genre");
        }
    }
    xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
```

```
    xhr.send(body);
}

function add_publ_house_button() {
    var xhr = new XMLHttpRequest();
    var body = "name=" + encodeURIComponent(document.getElementById("publ_name").value) + "&email=" +
encodeURIComponent(document.getElementById("publ_email").value) + "&phone=" +
encodeURIComponent(document.getElementById("publ_phone").value);
    xhr.open('POST', '../interface/addPublishingHouse.jsp');
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            PublHouses();
        }
    }
    xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
    xhr.send(body);
}

function CreateForm(header_name, form_id) {
    var main = document.getElementById("container");
    main.innerHTML = "";
    var header = document.createElement('h1');
    header.innerHTML = "Add " + header_name;
    main.appendChild(header);
    var form = document.createElement('form');
    table = document.createElement('table');
    form.setAttribute('method', 'POST');
    form.setAttribute('id', 'form_' + form_id);
    table.setAttribute('id', 'table_' + form_id);
    main.appendChild(form);
    form.appendChild(table);
}

function CreateTextInput(p_text, input_id, table, size) {
    var table = document.getElementById(table);
    var line = document.createElement('tr');
    var row1 = document.createElement('td');
    row1.innerHTML = p_text;
    line.appendChild(row1);
    var row2 = document.createElement('td');
    var input = document.createElement('input');
```

```
    input.setAttribute('id', input_id);
    input.setAttribute('type', 'text');
    input.setAttribute('size', size);
    row2.appendChild(input);
    line.appendChild(row2);
    table.appendChild(line);
}

function CreateSelectInput(select_id, p_text, val, id, name, table) {
    var form = document.getElementById(table);
    var line = document.createElement('tr');
    var row1 = document.createElement('td');
    row1.innerHTML = p_text;
    line.appendChild(row1);
    var row2 = document.createElement('td');
    var elem = document.createElement('select');
    elem.setAttribute('id', select_id);
    var res = "";
    val.forEach(value => {
        res += "<option value=" + value[id] + ">" + value[name] + "</option>";
    });
    elem.innerHTML = res;
    row2.appendChild(elem);
    line.appendChild(row2);
    form.appendChild(line);
}

function CreateFormButton(button_id, on_click, form_id) {
    var line = document.createElement('tr');
    var button_add = document.createElement('input');
    button_add.setAttribute('id', button_id);
    button_add.setAttribute('type', 'button');
    button_add.setAttribute('class', 'button_add');
    button_add.setAttribute('value', 'Add');
    button_add.setAttribute('onClick', on_click);
    line.appendChild(button_add);
    var form = document.getElementById(form_id);
    form.appendChild(line);
}
```