**Клиент — сервер** — вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами.

Система обмена книгами будет разработана на основе трёхуровневой архитектуры: клиент, сервер приложения, база данных – и будет представлять собой web-приложение.



**Сервер** — программный компонент вычислительной системы, выполняющий сервисные (обслуживающие) функции по запросу клиента, предоставляя ему доступ к определённым ресурсам или услугам.

В качестве языка программирования был выбран Java. Java обладает набором библиотек и инструментов, которые способствуют качественной и быстрой разработке web-приложений, а также предоставляет кроссплатформенную свободу и запускается практически на всех устройствах. Для реализации серверной части приложения планируется использование технологии Java Enterprise Edition, которая обладает набором спецификаций и методов для языка Java, описывающих архитектуру серверной части, а также обеспечивающих масштабируемость приложения и гибкость при создании.

В качестве сервера приложений системы выбран GlassFish.

Для сборки проекта будем использовать фреймворк Maven, который позволяет управлять необходимыми зависимостями и версиями.

Сервер включает в себя два компонента: базу данных и компонент управления системой.

При создании баз данных планируется использование MongoDB. **MongoDB** — документно-ориентированная система управления базами данных с открытым исходным кодом, не требующая описания схемы таблиц. Считается одним из классических примеров NoSQL-систем, использует JSON-подобные документы и схему базы данных.

База данных «Библиотека» будет содержать книги, участвующие в работе системы. Каждая книга характеризуется следующим набором характеристик:

* Автор.
* Название.
* Ключевые слова, при помощи которых книгу можно будет найти при поиске.
* BCID (Bookcrossing ID) – уникальный идентификатор книги.
* Владелец книги. Это поле заполняется только в том случае, если читатель, добавивший книгу, хочет, чтобы она к нему вернулась, иначе владельцем книги является библиотека.
* Статус. Книга имеет два состояния: свободна, занята.
* Жанр. Указывается для того, чтобы читатель мог просматривать книги выбранного жанра в библиотеке.
* Читатель. Человек, у которого книга находится в данный момент.
* Доступный способ обмена.

Для работы с системой пользователь должен будет пройти процедуру аутентификации, в связи с этим возникает необходимость в создании базы данных зарегистрированных пользователей. Она будет содержать:

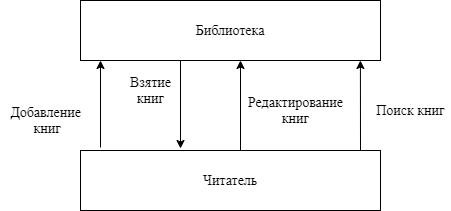
* имя;
* логин;
* пароль;

Компонент управления системой будет включать в себя реализацию следующих функций:

* регистрация;
* аутентификация;
* добавление книг;
* поиск книг;
* взятие книг;
* редактирование книг, которые находятся у пользователя в наличии (на книжной полке);
* просмотр библиотеки;
* просмотр доски заказов.

При входе в систему пользователю будет показан список недавно добавленных в библиотеку книг. С этого места будет возможен переход в библиотеку, в личный кабинет, в область поиска книг, на доску заказов. Из личного кабинета пользователь может перейти на свою книжную полку, чтобы изменить информацию о книгах в ней. Если при поиске книга не была найдена, она будет помещена на доску заказов.

При работе с библиотекой читатель может выполнять следующие действия:



1. Добавлять книгу.
2. Искать книгу.



Если искомая книга отсутствует в библиотеке, читателю будет предложено оставить свой запрос на доске заказов. При добавлении книги на доску заказов необходимо указать следующие характеристики:

* Автор.
* Название.
* Ключевые слова.

1. Взять книгу.
2. Редактировать книгу. Под редактированием книги подразумевается изменения способа обмена. Читатель может редактировать только ту книгу, что находится на его книжной полке.

Для тестирования системы воспользуемся облачной платформой Heroku. Данная платформа предоставляет бесплатный хостинг.

**Клиент** — это аппаратный или программный компонент вычислительной системы, посылающий запросы серверу.

Клиентская часть, разрабатываемой системы, будет представлена jsp-файлами, основная часть содержимого которых будет написана при помощи HTML и CSS. **Java Server Pages (JSP)** - это одна из технологий J2EE, которая представляет собой расширение технологии сервлетов для упрощения работы с web-содержимым. Страница **JSP** позволяет легко разделить web-содержимое на статическую и динамическую часть, допускающую многократное использование ранее определенных компонентов.

**MVC model**

Шаблон проектирования MVC предполагает разделение данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: Модель, Представление и Контроллер – таким образом, что модификация каждого компонента может осуществляться независимо.