**Лабораторная №1**

Разработать простое приложения с рендерингом на сервере. Например, список задач со статусом их выполнения, фильтрацией по статусу и выставлением ожидаемой даты завершения, а так же возможностью прикреплять файлы к каждой задаче. Сервер должен отдавать клиенту готовую разметку, отправка данных серверу должна осуществляться через отправку форм.

**Лабораторная №2**

Простое приложение, как в лабораторной работе 1, но с другой архитектурой. На сервере должен быть реализован REST API, на клиентк - Single Page Application. Обмен данных должен осуществляться путем отправки/принятия http запросов с данными в формате JSON или файлов в формате multipart/form-data. Обновление данных на клиенте не должно приводить к перегрузке страницы. Серверный REST API должен поддерживать ожидаемую семантику: правильно использовать http методы (GET для чтения данных, POST/PUT для изменения, DELETE для удаления и т.п.) и возвращать правильные коды ответов (200 в случае успешного чтения/изменения данных, 404 если ресурс не найдет и т.п.).

**Лабораторная №3**

Добавить к приложению из лабораторной №2 аутентификацию на базе JWT токенов. Токен должен передаваться через httponly cookie на клиент и так же отправляться на сервер. При попытке прочитать/изменить данные на сервере без валидного токена, должен клиенту должен возвращаться 401 код. При получении кода 401 клиент должен потребовать от пользователя ввода логина/пароля. Для формирования jwt токена можно использовать только пакеты jsonwebtoken и bcrypt. Логику аутентификации нужно описать в виде отдельного middleware той библиотеки, на которой написан сервер (например, Express).

**Лаборатонрая №4**

Как лабораторная 3, но заменить REST API на обмен данных через Web Sockets. Можно использовать библиотеку SockeIO.

**Лабораторная №5**

Как лабораторные 3 и 4, но на сервере сделать API на GraphQL.