**Лабораторная №1**

Разработать приложение с рендерингом на сервере – 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**

Разработать приложение, как в лабораторной работе №1, но с другой архитектурой. На сервере должен быть реализован стандарт REST API, на клиенте – 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 создания и 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 Сookie на клиент и так же отправляться на сервер. При попытке прочитать/изменить данные на сервере без валидного токена, клиенту должен возвращаться 401 код ошибки - Unauthorized. При получении кода 401 клиент должен потребовать от пользователя дополнительного ввода логина/пароля (всплывающая форма/окно).

**Лабораторная №4**

Разработать приложение, как в лабораторной работе №3, но заменить REST API на обмен данных через Web Sockets. Можно использовать библиотеку SockeIO.

**Лабораторная №5**

Разработать приложение, как в лабораторной работе №3-4, но на сервере сделать API на основе GraphQL.