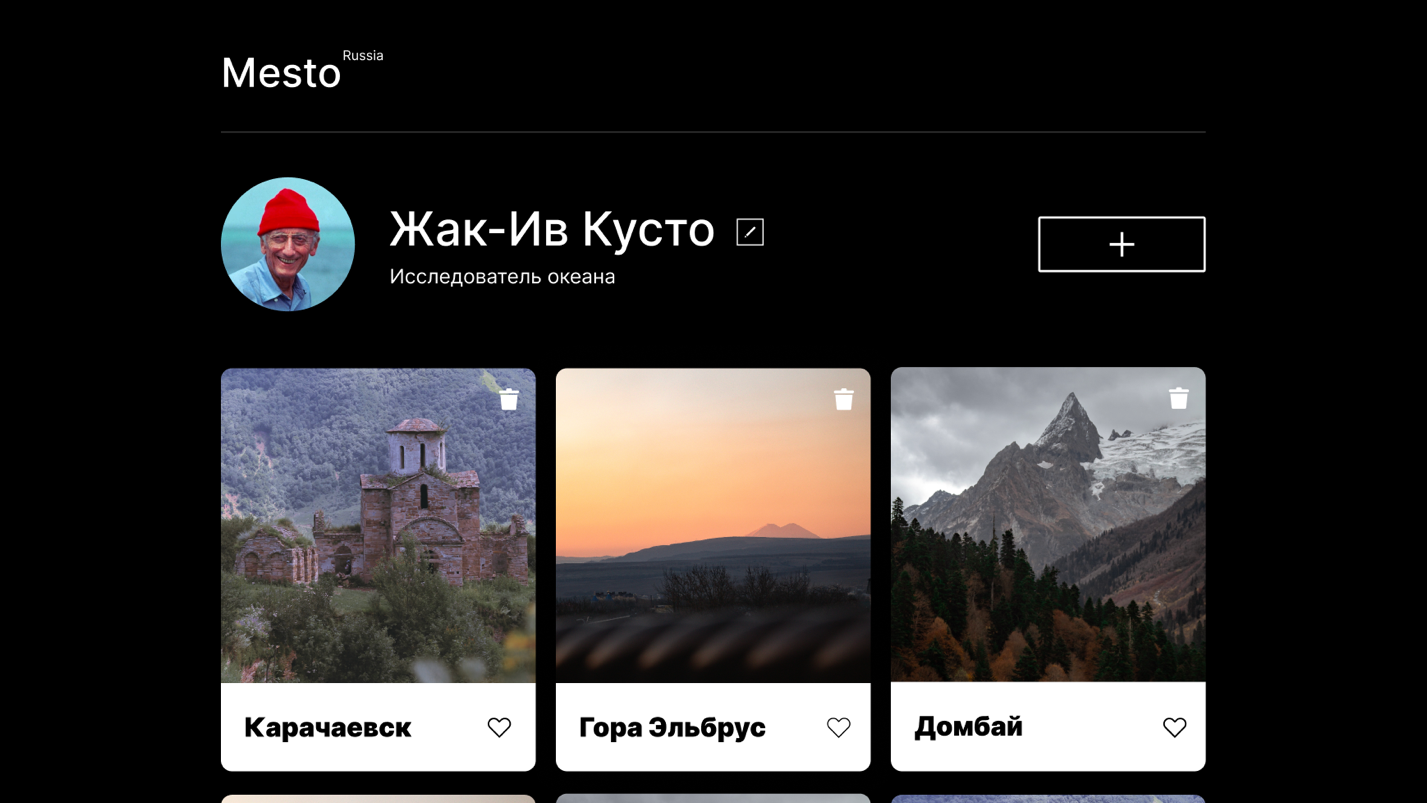
**Проектная работа 7**

Код в проекте Mesto разрастается — нужно его организовать.

Проектная работа этого спринта целиком нацелена на рефакторинг. Используйте свои знания о классах, чтобы переписать некоторые части кода.



*Функциональность готова, принимайтесь за рефакторинг*

**Создание классов Card и FormValidator**

Создайте класс Card, который создаёт карточку с текстом и ссылкой на изображение:

* принимает в конструктор её данные и селектор её template-элемента;
* содержит приватные методы, которые работают с разметкой, устанавливают слушателей событий;
* содержит приватные методы для каждого обработчика;
* содержит один публичный метод, который возвращает полностью работоспособный и наполненный данными элемент карточки.

Для каждой карточки создайте экземпляр класса Card.

Создайте класс FormValidator, который настраивает валидацию полей формы:

* принимает в конструктор объект настроек с селекторами и классами формы;
* принимает вторым параметром элемент той формы, которая валидируется;
* имеет приватные методы, которые обрабатывают форму: проверяют валидность поля, изменяют состояние кнопки сабмита, устанавливают все обработчики;
* имеет публичный метод enableValidation, который включает валидацию формы.

Для каждой проверяемой формы создайте экземпляр класса FormValidator.

**Требования к коду**

Эта самостоятельная работа отличается от предыдущих. До сих пор от вас требовалось реализовать какую-то функциональность. Вы думали, как заставить кнопки работать или сделать так, чтобы попап появлялся и исчезал. Сейчас весь функционал уже готов. Ваша задача — начать работу по организации кода.

Первое требование — добавить классы Card и FormValidator в код. Каждый из них выполняет строго одну задачу. Всё, что относится к решению этой задачи, находится внутри класса.

Второе требование — разбить JavaScript на модули. В проекте должно быть три js-файла:

* Card.js с кодом класса Card,
* FormValidator.js с кодом класса FormValidator,
* index.js со всем остальным кодом.

Классы Card и FormValidator экспортируются из соответствующих файлов, импортируются в index.js и используются в нём.

Отдельные js-файлы подключены в index.html как модули.

**Установка локального сервера**

В теме «Модули в JS» мы упомянули, что при подключении js-файлов как модулей вы столкнётесь с ошибками, и код не будет работать. Чтобы это исправить, нужно установить локальный сервер.

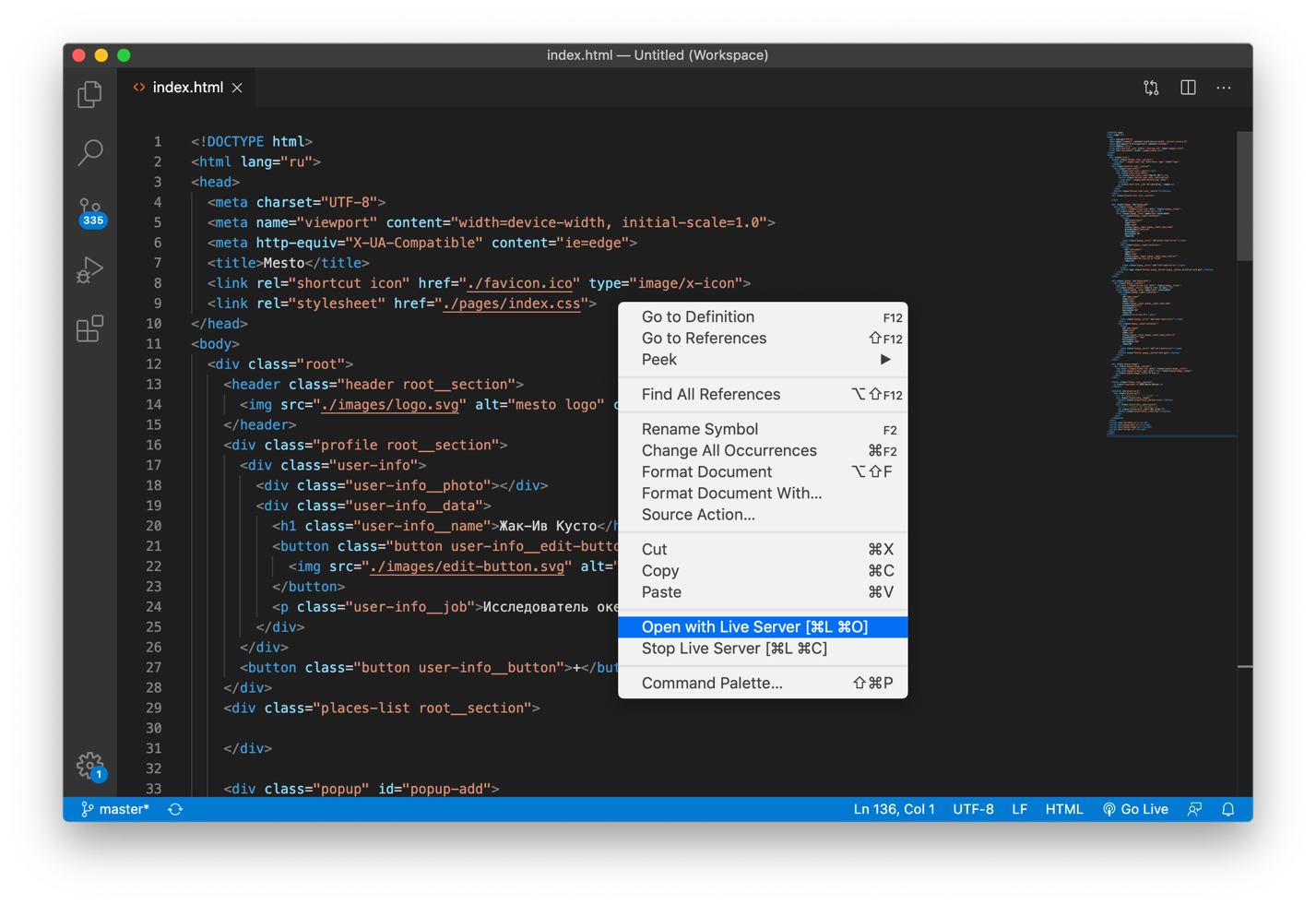
Самый простой способ — установить расширение в текстовый редактор. В VSCode это Live Server.

Чтобы добавить Live Server в VSCode, нажмите Ctrl + Shift + X на Windows или Linux и напишите в поисковой строке имя расширения — Live Server. Когда оно появится в результатах поиска, кликните на него и нажмите Install:



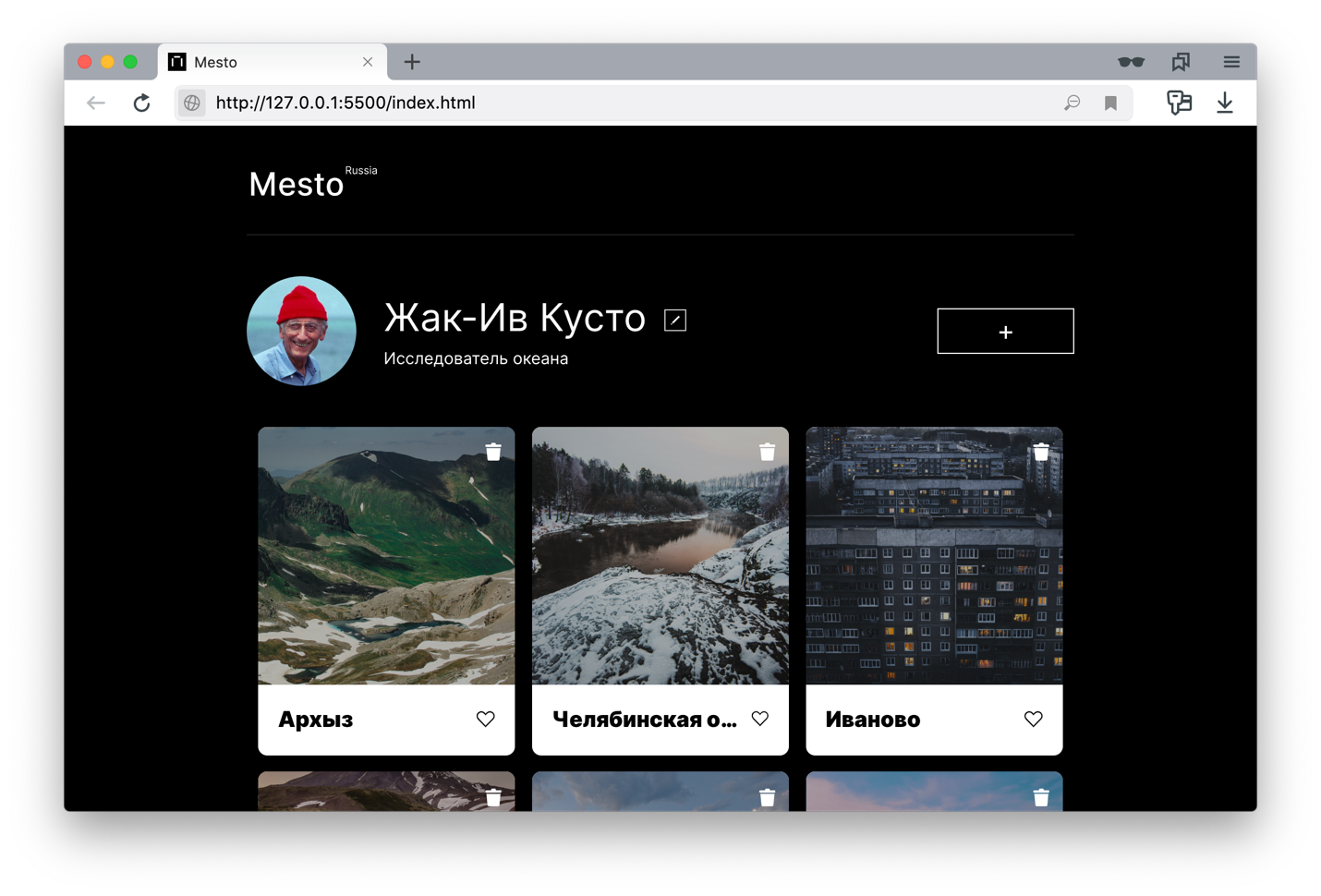
*Установка расширений для разных редакторов аналогична*

После установки расширения перезагрузите VSCode. Откройте index.html проекта, кликните правой кнопкой по окну с кодом, нажмите «Open With Live Server»:



*Теперь всё заработает*

Локальный сервер запущен и файлы раздаются с него.



*Обратите внимание на url — это адрес сервера*

Дополнительный бонус — если изменить содержимое любого файла в проекте, сервер перезапустится, а содержимое окна браузера обновится без перезагрузки страницы.

В будущем вы узнаете больше о том, как работает сервер, и даже создадите свой собственный. А пока просто пользуйтесь расширением и сконцентрируйтесь на проектных задачах.