**Основные положения**

Т.к. действия должны происходить поверх карты, то подразумевается, что все общение с сервером будет происходить без перезагрузок. Значит AJAX запросы…

Изначально на клиент передаются только координаты объектов (кол-во зависит от текущего масштаба карты - скорее всего, в запросе будет передан “разброс” координат, в котором нужно найти объекты)

Далее, в зависимости от действий пользователя, сервер выдает необходимые данные.

Необходима обработка ошибок. Если на стороне сервера возникла какая-то ошибка или неверные/некорректные данные, то возвращаем на клиент сообщение с текстом ошибки и соответствующим HTTP кодом.

**Добавление**

Стандартное добавление объекта…

Обязательные поля: координаты и фото.

Остальное - на усмотрение добавляющего.

Нужна возможность “пакетного” добавления. Т.е. сразу много фоток и файл со списком координат. Сопоставление по индексу (1-я фотка = 1-я координата).

Файлы предполагаются большого размера, так-что желательно реализовать “красивую” загрузку. С индикатором, возможностью докачки и т.п.

**Оптимизация изображений**

При загрузке изображения сразу создавать несколько его “вариантов”:

* Миниатюра (чистим метаданные, меняем размер, значительно сжимаем)
* Оптимизированное (чистим метаданные, незначительно сжимаем)
* Оригинал (вообще не трогаем)

Данные варианты хранить в отдельной папке для каждого объекта. Назвать папку можно по ID в БД, чтобы не хранить лишний url.

Сами файлы называть одинаково в каждой папке, чтобы была простая логика url’а:

* min.jpg
* optimal.jpg
* original.raw

**Редактирование**

Необходима возможность редактирования всех полей объекта (описание, координаты, название и т.п.)

**Удаление**

Удаляем объект из БД, все его лайки и папку с фотками.

Также удаляем его из всех избранных.

**Пагинация**

Т.к. все происходит поверх карты, то наилучшим решением будет стандартная “бесконечная” подгрузка (как в ВК). Данные будут выдаваться “порциями”, при достижении конца.

Чтобы на клиенте не мучались с “встройкой” новых данных, лучше на сервере использовать какой-нибудь шаблонизатор.

Перед выдачей очередной “порции” данных, мы рендерим HTML-блок и отправляем его.

На клиенте будет, в таком случае, обычный .innerHTML += Полученные данные.

Естественно, отсекать нужную порцию данных нужно ещё при запросе к бд (чтобы не гонять лишние данные). Кол-во объектов в одной “порции” лучше хранить в отдельном конфиг-файле, чтобы можно было быстро изменить.

**Фильтрация**

Есть два “блока” фильтров: место и тип

При фильтрации формируется новый запрос к бд с условием, на основе выбранных фильтров.

В данном запросе “блоки” объединяются через лог. ***И***, а элементы внутри этих блоков - через лог. ***ИЛИ***.

Например:

(Волжский ***ИЛИ*** Волгоград) ***И*** (Садик ***ИЛИ***  Магазин ***ИЛИ*** Кинотеатр)

**Пользовательская активность**

Тут много зависит от того, будут ли регистрироваться пользователи или нет…

**Лайки**

Отдельная таблица/коллекция, в которой храним все лайки, вида “объект - пользователь”. Т.е. кто и что лайкнул. (Если это nosql, то можно и в документе самого объекта хранить массив с лайками)

Потом просто считаем кол-во строк/записей для необходимого объекта.

Защита от повторного лайка, в случае с зарегистрированными пользователями предельно проста: мы просто проверяем, нет ли лайк от этого пользователя для данного объекта. Если еще нет, то добавляем.

Для незарегистрированного сложнее…

Это, скорее всего, нужно будет делать на стороне клиента (хранить лайкнутые объекты в куках или локальном хранилище)

**Избранное**

С зарегистрированными все снова предельно просто: храним в БД ID выбранных записей. Создаём отдельную таблицу/коллекцию вида “объект - пользователь”. Т.е. что и кто сохранил. Тут вариант с хранением массива “в пользователе” не подойдёт, т.к. при удалении объекта, нужно будет удалять его из избранного у всех пользователей. В варианте с отдельной таблицей, это сделать намного проще.

С “гостями” сложнее. Тут снова придется разбираться на стороне клиента (хранить список избранных записей в куках или локальном хранилище)