В небольшой компании работает около 50 человек.

На этаже стоит новый Smart холодильник с напитками и едой.

В компании очень любят пить виноградную газировку.

Периодически кто-то выпивает эту виноградную газировку, а офис-менеджер не знает, что она закончилась.

Помимо этого, есть сотрудники, которые пьют только минералку и им нет смысла идти к холодильнику, если минералки там нет.

Энтузиасты из компании решили написать небольшую веб-утилититу для удаленного управления холодильником.

Цель системы: избавиться от максимального количества проблем связанных с контролем удаленного холодильника.

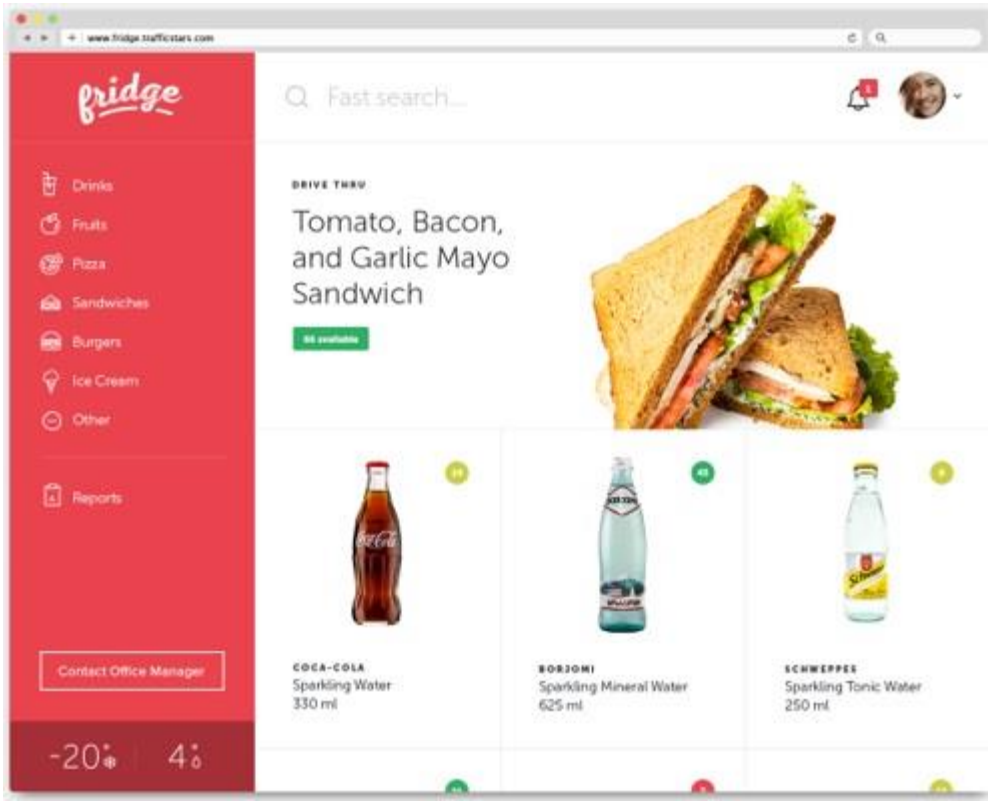
Ваша задача: разработать простой и удобный веб-интерфейс (не мобильное приложение) системы по удаленному управлению холодильником.

Здесь любой новичок мгновенно выдает себя тем, что начинает рисовать стандартный лонгрид, которых он так много видел в вебе, и которые, по его мнению, являются стандартным ответом на любые задачи бизнеса.

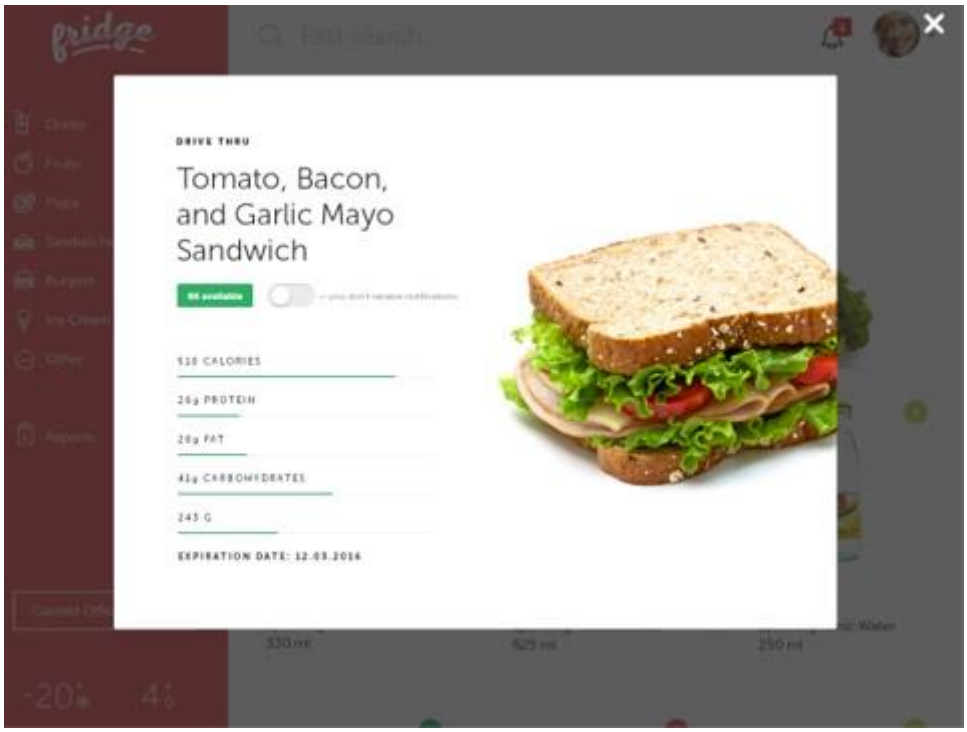
Также новичок обожает сам себе придумывать дополнительные требования, которых не существует в бизнес-задаче:

- отображать интерфейс нужно обязательно в окошке макбука, ведь в небольших компаниях никогда принципиально не используют Windows (сарказм),
- еще более обязательно потратить кучу личного времени — но создать цветное красивое навигационное меню и придумать логотип,
- обязательно упомянуть в интерфейсе про влажность и градусы, ведь эта информация будет так важна каждому...

Также любой неопытный новичок в мире не мыслит веб-интерфейс без личного кабинета с функцией установки аватара, без текстовых оповещений и строки поиска, а отсутствие ярких иллюстраций в любой микро-утилите и вовсе считается дурным тоном.



Зачем, почему, сколько времени займет программинг и в какой бюджет это выльется? — такие вопросы пока не заботят новичков, ведь им куда важнее явить миру свою запредельную креативность и придумать визуальную киллер-фичу со счетчиком калорий и протеинов в каждом куске еды:



Кто это все будет вводить в систему ? Сколько операторов потребуется для поддержания базы в актуальном состоянии? — такие вопросы также пока далеко за гранью понимания новичка. Он пока только рисует, а не делает общий бизнес.

Главнейшая и принципиальнейшая ошибка новичка — постоянная забота о том, как бы оставить свой след художника в мировом интерфейсном искусстве и всех удивить. Никогда, никогда так не делайте. Тянет просто порисовать для души, то порисуйте дома, но на работе вы — сфера услуг, которая оказывает бизнесу помощь по облегчению его нелегкой жизни.

Правильное решение данной задачи:

Куда логичнее было бы слегка проанализировать задачу перед началом выдачи очередного dribbble-шаблона и исходить из того, что интерфейс проектируется для небольшой группы людей с заведомо экспертным уровнем знакомства с информационными технологиями и интерфейсами.

Эти люди будут пользоваться этой системой регулярно и (при гарантии полного отсутствии альтернатив от конкурентов) — моментально обучатся любым интерфейсным решениям; а значит, экономия времени и места на рабочем экране вполне могут считаться единственными измеримыми факторами, по которым можно будет затем экономическую эффективность от внедрения в целом.

Далее, оценив аудиторию и условия, в которых сформировалась задача, наш учебник настоятельнейше рекомендует к каждому тестовому заданию, дизайн-макету или прототипу прикладывать Экономическое обоснование.

Подобные обоснования в обязательном порядке делали все рационализаторы и инноваторы еще во времена расцвета СССР, и это именно тот бесценный опыт, который ни в коем случае не стоит терять в наших реалиях 21 века. Особенно сейчас, когда некоторые ИТ-команды начали сорить ресурсами, абсолютно не задумываясь о бизнес-целях.

Конечно, для тестовой задачи не обязательно писать полноценный бизнес-план, вполне достаточно будет и такого обозначения наличия денежного обоснования для найденного решения:

> Подсчитано, что суммарное время походов всех 30 сотрудников до холодильника и обратно составляет 15 рабочих часов в месяц. Нами всего за 3 часа программинга будет создана утилита, при использовании которой в сумме весь отдел затратит в месяц всего 30 минут на походы к холодильнику. Таким образом, 14,5 часов рабочего времени ежемесячно будут сэкономлены с помощью нашего решения, ввиду чего предлагаем разово премировать команду разработки на сумму, эквивалентную стоимости этих 14,5 рабочих часов.

Помните нашу Главу 4: «Невозможно управлять тем, что не можешь измерить?» Так вот, добавление к картинке привычных взору любого опытного бизнесмена табличек из цифр и диаграмм сразу повышает интеллектуальную стоимость того же самого решения в разы.

Но за счет чего получится так сэкономить время всего коллектива, не перегружая его заполнением еще одного профиля, приватной перепиской в еще одном свеженьком чатике или уточнением текущей температуры в морозилке?

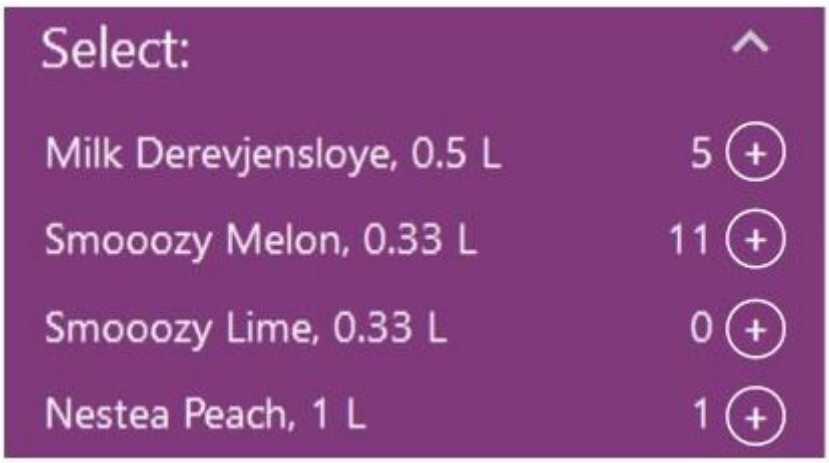
Для начала предположим, что место в холодильнике конечно, и в принципе будет невозможно заказывать случайное количество бутылок газировки, минералки или молочного коктейля с мороженым, так как их попросту некуда будет ставить, а

захламлять всю кухню упаковками — это анти-юзабилити. Именно поэтому, число напитков любого вида в нашей задаче всегда будет не более 30, причем, при достижении количества 5 или меньше — заказ на доставку отправляется из утилиты автоматически, не отвлекая офис-менеджера от более приоритетных задач.

Впрочем, перед надвигающимися пятничными посиделками, менеджеру можно будет дозаказывать напитки вручную даже при остатке 20 и более штук.

Обратите внимание: постоянная экономия рабочего времени и действий пользователя даже на манипуляции с самим интерфейсом — это и есть тот сложный «умный» алгоритм, вполне способный сам принимать стандартные решения и экономия пространства на самой кухне от лишних паков с напитками — который ведет к визуальному упрощению интерфейса. Так в каждой детали всюду виден ход мыслей опытного практика, внимательного к каждой детали реальности.

Далее, заказ же нового нестандартного напитка должен быть значительно ограничен, чтобы офис-менеджер не тратил рабочее время на поиски новых поставщиков для односолодового виски, брусничного морса, горячего кофе и т.д. Все доступные варианты для пользовательского выбора указываются в блоке Select ниже.



Еще одно принципиальное условие — возможность для пользователя посмотреть всю нужную информацию одним наведением без клика, вообще не разворачивая окно на полный экран, если не требуется взаимодействовать с утилитой, а только получить от нее информацию о наличии напитков. Это может быть как свернутое окно браузера, так и нативное приложение, например, для Win10.

Визуально, подобное решение же выглядит неброским и даже примитивным. Более того, оно не отвлекает, не развлекает, не переключает рабочий эмоциональный фон, не дарит игровые бонусы и не позволяет флиртовать с коллегой. Оно делает ровно то, что нужно и не пытается взять при этом «дизайнерского Оскара».

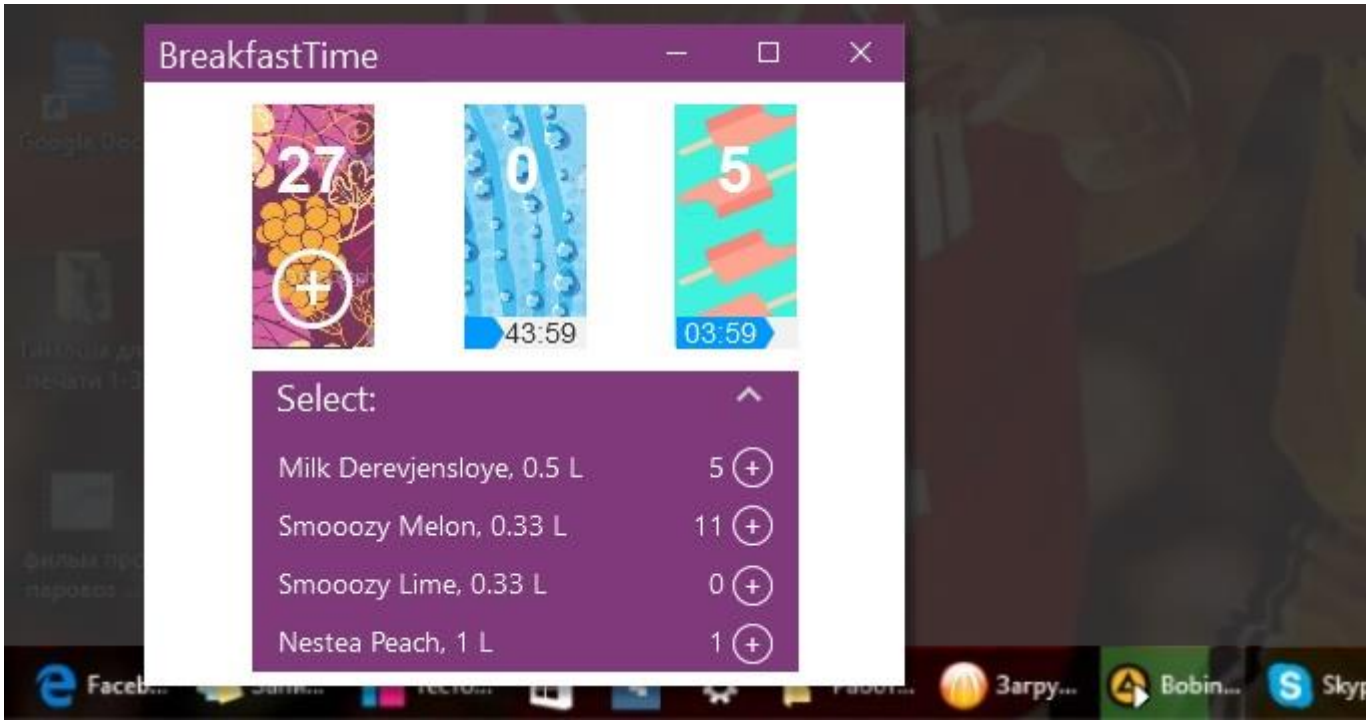
В общем виде, простой интерфейс есть предельно лаконичный интерфейс, где функционал «выведен на поверхность» только там, где больше ничего уже автоматизировать невозможно.

О 99 из каждых ста происходящих внутри системы расчетов и взаимодействий ваш конечный пользователь не должен даже подозревать. Прячьте все это как провода в короба.

Еще раз напомним, что ни в коем случае не нужно нарочито расширять функционал подобной утилиты до заказов комплексных обедов в компанию или до автоматического контроля сроков годности напитков в холодильнике. Если такое будет интересно заказчику — все придуманные далее фичи допродаются отдельным договором, за отдельную цену и сроки.

Не пихать все имеющиеся идеи сразу в одну корзину — это тоже показатель многолетнего сугубо практического опыта.

Ах да, сама утилита на рабочем столе всех тридцати сотрудников вымышленной компании должна выглядеть примерно так:



Пользователь просто наводит мышку и за секунду уже понимает, что за минералкой идти пока не стоит, и вместо получасовой болтовни на кухне со знакомой в ожидании подвоза, еще минут 40-50 можно будет спокойно сидеть и дописывать нужный код.

Лаконично как в Спарте, эффективно как в армии, но при этом конкретно и «чотко» как на деревенской дискотеке. А самое главное, что наш заказчик уже услышал обоснование в часах\рублях: что, где, как и за счет чего он реально сэкономит.

Классический trouble-shooting отличается от нашего анализа user experience тем, что здесь предложил бы совершенно другое решение: нанять официанта-разносчика напитков, или купить мини-холодильник в каждой комнате, или вообще запретить всем есть, пить и чихать в рабочее время. В любом случае, это будет уже работа не с пользователями и снижением числа их манипуляций, но с самой структурой компании: оптовыми закупками, кадровым расписанием, учетом рабочего времени и т.д. Совсем иной масштаб и, возможно, более эффективные решения. Но это уже выходит за рамки Учебника.