УДК 004.422.83

ОДИН ИЗ ПОДХОДОВ РАЗРАБОТКИ WEB-ПРИЛОЖЕНИЯ

[1]Линевич Д.О., [1]Скудняков Ю.А., [2]Гурский Н.Н.

[1]Белорусский государственный университет информатики и радиоэлектроники, г. Минск

[2]Белорусский национальный технический университет, г. Минск

Для разработки web-приложения в первую очередь необходимо сформулировать ее цель, определить архитектуру проекта, безопасность, масштабируемость, гибкость, скорость в работе с продуктом и доработок, высокую отказоустойчивость, быстродействие системы, простоту интеграций со сторонними приложениями и сервисами [1].

Процесс разработки современного сайта состоит из следующих основных этапов:

– техническое задание (ТЗ). Его разработку для web-специалистов выполняет, обычно, менеджер всего интернет-проекта, а работа с самим заказчиком начинается с заполнения брифа, где он излагает свои желания в отношении структуры сайта и его визуализации, уточняет ошибки и недоработки в случае их наличия в прошлой версии web-сайта, приводя свои примеры, как у его конкурентов. На основании брифа, менеджер создаёт ТЗ, учитывая при этом имеющиеся в наличии возможности дизайнерских и программных инструментов.

– вёрстка страниц и шаблонов в HTML. Утверждённый дизайн передаётся специалисту-верстальщику, «нарезающему» графическое изображение на отдельные картинки, из которых позже будет сложена HTML-страница. В ходе такой работы создаётся программный код, который возможно уже смотреть при помощи какого-либо браузера (интернет-обозревателя).

– программирование. После выполнения описанных выше этапов готовые файлы в формате HTML передаются для работы web-программисту. Разработка программного обеспечения интернет-сайта вполне может выполняться как «с самого нуля», так и на основании системы CMS, зачастую так называемого «cms-движка».В случае применения системы управления сайтом следует отметить, что она сама в некоторой степени представляет уже готовый сайт, включающий в себя заменяемые блоки. В этом случае программист выполняет функции «cms-специалиста», который должен заменить существующий стандартный шаблон на новый и оригинальный, разработанный на базе начального web-дизайна[2].

– тестирование – как заключительный этап web-разработки интернет-сайта. Сам такой процесс вполне может содержать в себе самые различные виды проверок, например, такие как: внешний вид страницы

сайта с увеличенными шрифтами, при различных размерах браузерного окна, или из-за отсутствия flash-плеера, и многое иное. Также используется и пользовательское тестирование, так называемое юзабилити.

Также следует отметить, что общий облик современного web-приложения, как правило, определяется полным разделением между клиентом и сервером. Каждая из сторон для другой является «черным ящиком». Сервер не содержит данные про внутреннее устройство клиентов, а клиенты – про устройство сервера. Кроме того, web-приложение может быть не единственным клиентом. API следует проектировать так, чтобы его можно было использовать без изменений на любой платформе –web, мобильные и десктопные приложения.

В качестве слоя API используется REST с рассылкой уведомлений через WebSocket. REST применяется почти повсеместно, мало в каких проектах найдутся причины для использования других подходов. WebSocket важны, потому что без них не удастся сделать «живой» интерфейс с мгновенным обновлением информации. Несмотря на высокий интерес сообщества к GraphQL, для большинства проектов плюсы этой технологии не перевесят минусов. Хорошей практикой является создавать API версионируемым и использовать автоматически сгенерированную документацию для его описания. Для небольших приложений возможна реализация бекенда в виде «монолита» – единого сервиса, в котором реализованы все необходимые фронтенду API. Для более сложных проектов в последнее время часто применяется микросервисный подход, когда та или иная функциональность фронтенда оформляется в виде независимого сервиса бекенда.

Такой подход повышает гибкость в разработке: нет нужды привязываться к одному языку программирования или к одному технологическому стеку; большие приложения смогут продолжать работать даже при отказе одного из сервисов; при возрастании нагрузки можно масштабировать только те сервисы, на которые данная нагрузка ложится в большей степени. Это позволяет экономнее расходовать ресурсы оборудования.

Если работа выполняется с большим количеством web -приложений, то имеет смысл создать собственный UI kit.

Литература

1. Беллиньясо, М. Разработка Web-приложений в среде ASP.NET 2.0: задача – проект – решение / М.Беллиньясо. – М.: «Диалектика», 2007. – С. 640. – ISBN 0-7645-8464-2.

2. Гото, К. Веб-редизайн / К. Гото, Э. Котлер// 2-е издание. – СПб.: «Символ-Плюс», 2006. – С. 416. – ISBN 5-93286-082-0.