**2** СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ

Изучив теоретические аспекты разрабатываемой системы и выработав список требований необходимых для разработки системы, разбиваем систему на функциональные блоки (модули). Это необходимо для обеспечения гибкой архитектуры. Такой подход позволяет изменять или заменять модули без изменения всей системы в целом.

В разрабатываемом веб-приложении можно выделить следующие блоки:

* блок пользовательского интерфейса;
* блок ядра клиентского приложения;
* блок для работы с клиентской базой данных;
* блок работы с клиентским приложением;
* блок регистрации;
* блок веб-сервиса;
* блок работы с реляционной базой данных;
* блок реляционной базы данных;
* блок ядра серверного приложения.

Структурная схема, иллюстрирующая перечисленные блоки и связи между ними приведена на чертеже ГУИР.400201.064 C1.

Каждый модуль выполняет свою задачу. Чтобы система работала каждый модуль взаимодействует с другими модулями путем обмена данными, используя различные форматы и протоколы.

Рассмотрим функциональные блоки веб-приложения.

*Блок пользовательского интерфейса* является клиентской частью веб-приложения. Данный блок представляет собой совокупность средств, при помощи которых пользователь взаимодействует с приложением через браузер. Для построения интерфейса используется технологии HTML, CSS, TypeScript. TypeScript — язык программирования, представленный Microsoft в 2012 году и позиционируемый как средство разработки веб-приложений, расширяющее возможности JavaScript. Строгость и большая функциональность языка позволяет разработать библиотеку компонент, с помощью которой будут строиться основные компоненты клиентского интерфейса [7].

*Блок для работы с клиентской базой данных* предоставляет доступ к локальному хранилищу браузера, где хранятся авторизационные данные пользователя, а также данные, которые по тем или иным причинам не были отправлены на сервер.

Интернет-хранилище – это программные методы и протоколы веб-приложения, используемые для хранения данных в веб-браузере. Интернет-хранилище представляет собой постоянное хранилище данных, похожее на куки, но со значительно расширенной ёмкостью и без хранения информации в заголовке запроса HTTP. Существуют два основных типа веб-хранилища: локальное хранилище и сессионное хранилище, ведущие себя аналогично постоянным и сессионным кукам соответственно.

Таким образом, данный блок представляет собой временное хранилище данных пользователя до ближайшего сеанса синхронизации с сервером.

*Блок ядра клиентского приложения* представляет собой центр управления всеми частями клиентской части приложения. Данный блок знает обо всех блоках на клиентской стороне, организует совместную работу всех блоков, а также организует работу с блоком для работы с клиентским приложением.

*Блок работы с клиентским приложением* представляет собой центр для связи клиентской части с серверной. Через него проходят все операции синхронизации и обмена данными.

*Блок регистрации* служит для регистрации новых пользователей приложения. Поскольку регистрация пользователей приложения содержит логику отличную от основной логики работы приложения, имеет смысл вынести его реализацию в отдельный блок.

*Блок веб-сервиса* представляет встроенное приложение с использованием фреймворка ExpressJS, который предназначен для удобного построения API веб-сервисов.

*Блок работы с реляционной базой данных* включает механизмы для взаимодействия с данными, используемыми приложением. Для ее реализации была выбрана база данных SQLite. SQLite – компактная встраиваемая реляционная база данных, запросы к которой можно осуществлять при помощи языка запросов SQL. База данных не поддерживает все особенности SQL и уступает в функциональности другим развитым СУБД, но вполне подходит для хранения и извлечения информации.

SQLite не использует парадигму клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а предоставляет библиотеку, с которой программа компонуется и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются функции (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется; ACID-функции достигаются в том числе за счёт создания файла журнала [8].

Несколько процессов или потоков могут одновременно без каких-либо проблем читать данные из одной базы. Запись в базу можно осуществить только в том случае, если никаких других запросов в данный момент не обслуживается; в противном случае попытка записи оканчивается неудачей, и в программу возвращается код ошибки. Другим вариантом развития событий является автоматическое повторение попыток записи в течение заданного интервала времени.

Благодаря архитектуре движка возможно использовать SQLite как на встраиваемых системах, так и на выделенных машинах с гигабайтными массивами данных.

*Блок ядра серверного приложения* представляет собой центр управления всеми частями серверной части приложения. Данный блок знает обо всех блоках на серверной стороне, организует совместную работу всех блоков.

Блок сервера работает на платформе NodeJS. NodeJS является платформой для создания современных веб-сервисов. Он позволяет писать асинхронный, неблокирующий код, используя стиль и шаблоны NodeJS. Позволяет использовать возможности баз данных, таких как Elasticsearch и Redis. Работать со многими протоколами, создавать веб-службы RESTful, клиенты и серверы сокетов TCP. В платформу встроена возможность проверять функциональность кода с помощью библиотеки Mocha и управлять его жизненным циклом с помощью менеджера пакетов NPM (Node Packet Manager). Используя ряд практических областей программирования, возможно использовать новейшие доступные функции ECMAScript, классы, и популярные модули, а также создавать богатые инструменты командной строки и веб-интерфейс с использованием современных технологий веб-разработки [9].