ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ КРАСНОДАРСКОГО КРАЯ "КРАСНОДАРСКИЙ КОЛЛЕДЖ ЭЛЕКТРОННОГО ПРИБОРОСТРОЕНИЯ"

Заявляемая номинация: Инженерно-математическая

Тема: Электронный журнал

Автор: Андреев Данил Олегович

Научный руководитель: Сторчак Евгений Евгеньевич

Место выполнения проектной работы: ГБПОУ КК ККЭП

2019

Содержание

Введение

Главной целью проекта являлась разработка электронный журнал, который в последствии может быть в ведён в образовательный процесс. Данный проект сможет сократить количество рутинных действий, которые требуется повторять многократно.

Возможность перевести всё на ”цифру” является очень весомым преимуществом, ибо благодаря этому станет возможно не только сократить количество времени, которое уходит на выполнение рутинных действий, но и возможность получать больший функционал. Например, получать глубокую аналитику по определённым студентам в реальном времени, в хоте чего станет возможно прогнозировать и делать прогнозы по поводу успеваемости обучающихся. Кроме того, при проектирование электронного журнала производиться подробный анализ предметной области, благодаря чему можно спроектировать систему максимально удобную для всех её пользователей. Каждый, начиная со студента и заканчивая администрацией учебного заведения сможет получить удобный инструмент для контроля. В будущем возможно внедрение дополнительных систем, который смогут дополнить функционал журнала.

Используемые технологии

Java

В качестве языка программирования был выбран Java. Java представляет собой язык программирования и платформу вычислений, которая была впервые выпущена Sun Microsystems в 1995 г. Существует множество приложений и веб-сайтов, которые не работают при отсутствии установленной Java, и с каждым днем число таких веб-сайтов и приложений увеличивается. Java отличается быстротой, высоким уровнем защиты и надежностью. От портативных компьютеров до центров данных, от игровых консолей до суперкомпьютеров, используемых для научных разработок, от сотовых телефонов до сети Интернет — Java повсюду!

Основной задачей разработчиков Java было создание переносимых приложений. JVM играет центральную роль в переносимости — она обеспечивает должный уровень абстракции между скомпилированной программой и базовой аппаратной платформой и операционной системой. Несмотря на этот дополнительный «слой», скорость работы приложений необычайно высока, потому что байт-код, который выполняет JVM, и она сама отлично оптимизированы. (Рисунок №1)

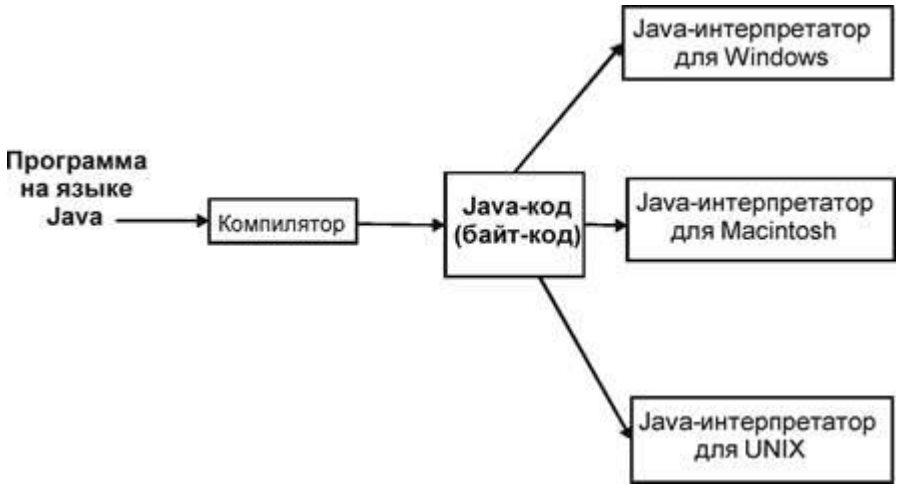

Рисунок 1- Принцип работы Java

В основе технологии Java лежит клиент-серверная модель, а Java-программа состоит из нескольких блоков, каждый из которых выполняет определенную часть общей задачи. На стороне клиента присутствуют только те блоки, которые необходимы в данный момент. Причем наиболее часто используемые блоки хранятся в кэше на жестком диске или в оперативной памяти компьютера пользователя. Поскольку блок загружается с сервера, то и управлять такой системой можно с сервера, т. е. централизованно. Это также гарантирует, что пользователь всегда будет использовать самую последнюю версию программы.

Основной компонент этой технологии - виртуальный Java-процессор, который представляет собой среду для исполнения Java-команд, или так называемых байт-кодов. Любая Java-программа должна соответствовать спецификации виртуального Java-процессора, которая полностью определяет систему команд Java, типы данных, обрабатываемых Java-процессором, и его регистры.

Кроме виртуального процессора, технология Java включает в себя (в качестве необязательного элемента) объектно-ориентированный язык программирования, построенный на основе языка C++, к которому добавили

новые механизмы для обеспечения безопасности и распределенных вычислений.

Особенностью Java являются аплеты. Аплет - это небольшая программка, в которой должно быть определено несколько обязательных функций. Аплет загружается по сети и может выполняться на Web-браузере, который поддерживает язык Java. Именно эта часть Java-технологии предназначена для использования во всемирной сети Internet, и поэтому защита должна распространяться как на сам аплет, так и на клиента сети, который использует этот аплет.

Существует множество областей применения Java, от сайтов электронной коммерции до Android приложений, от научных до финансовых приложений, таких как трейдинговые системы, от игр, типа Minecraft, до настольных программных средств, таких как Eclipse, Netbeans и IntelliJ, от open source фреймворков до J2ME приложений и т.д.

Java очень обширно применяется в финансовой сфере. Многие мировые инвестиционные банки, типа Goldman Sachs, Citigroup, Barclays, Standard Charted и другие используют Java для написания фронт-энд и бэк-энд офисных электронных систем, систем регулирования и конфирмации, проектов обработки данных и некоторых других. Преимущественно Java используется при написании серверных приложений, в большинстве своём без какого-либо пользовательского интерфейса, которые получают данные с одного сервера, обрабатывают их и отправляют дальше. Java Swing был также популярен для создания «толстоклиентных» интерфейсоф, но сейчас C# быстро захватывает рынок в этой области, а Swing уже выдыхается.

Также Java широко используется в электронной коммерции и в области вэб-приложений. Огромное количество RESTful сервисов было создано с использованием Spring MVC, Struts 2.0 и похожих фреймворков. Даже простейшие приложения, основанные на Servlet, JSP и Struts, достаточно популярны в различных государственных проектах. Многие вэб-приложения

государственных, оздоровительных, страховых, образовательных, оборонительных и некоторых других отделений написаны на Java.

Spring Boot

Авторы Spring решили предоставить разработчикам некоторые утилиты, которые автоматизируют процедуру настройки и ускоряют процесс создания и развертывания Spring-приложений, под общим названием Spring Boot.

Spring Boot — это полезный проект, целью которого является упрощение создания приложений на основе Spring. Он позволяет наиболее простым способом создать web-приложение, требуя от разработчиков минимум усилий по его настройке и написанию кода.

Spring Boot обладает большим функционалом, но его наиболее значимыми особенностями являются: управление зависимостями, автоматическая конфигурация и встроенные контейнеры сервлетов.

Чтобы ускорить процесс управления зависимостями, Spring Boot неявно упаковывает необходимые сторонние зависимости для каждого типа приложения на основе Spring и предоставляет их разработчику посредством так называемых starter-пакетов (spring-boot-starter-web, spring-boot-starter-data-jpa и т.д.)

Starter-пакеты представляют собой набор удобных дескрипторов зависимостей, которые можно включить в свое приложение. Это позволит получить универсальное решение для всех, связанных со Spring технологий, избавляя программиста от лишнего поиска примеров кода и загрузки из них требуемых дескрипторов зависимостей (пример таких дескрипторов и стартовых пакетов будет показан ниже)

Например, если вы хотите начать использовать Spring Data JPA для доступа к базе данных, просто включите в свой проект зависимость spring-boot-

starter-data-jpa и все будет готово (вам не придется искать совместимые драйверы баз данных и библиотеки Hibernate)

React Js

React часто упоминают в одном ряду с другими javascript фреймворками, но споры «React vs Angular» не имеют смысла, потому что это не сопоставимые вещи. Angular — это полноценный фреймворк (включающий и уровень представления). React — нет. Вот почему React вызывает столько непонимания в развивающемся мире полноценных фреймворков — это только представление.

React дает вам язык шаблонов и некоторые callback-функции для отрисовки HTML. Весь результат работы React — это HTML. Ваши связки HTML/JavaScript, называемые компонентами, занимаются тем, что хранят свое внутреннее состояние в памяти (например: какая закладка выбрана), но в итоге вам просто выплевывается HTML. Вы всегда можете сказать, как ваш компонент будет отрисован, глядя на исходный код. Это может быть важным преимуществом, хотя это ничем не отличается от шаблонов Angular.

Разрабатывая публичный сайт или приложение, и производя рендеринг всего на клиенте, то это неправильный путь. Клиентский рендеринг — это причина, почему SoundCloud работает медленно, и почему Stack Overflow (используя только серверный рендеринг) работает так быстро. Используя React можете рендерить React на сервере, и это будет правильный подход.

Используемые принципы и подходы

MVC. Модель. Вид. Контроллер.

Основные задачи MVC:

1. Экранировать модель – то есть, сделать так, чтобы модель не “знала” ничего, о технической реализации приложения, о пользовательском интерфейсе, сетевых протоколах, работе с базой и даже архитектуре системы. Модель должна отображать предметную область и быть вне технологий.

2. Отделить слой представления – значит, иметь возможность создать несколько представлений для одной и той же модели, которые могут существовать параллельно, например, для разных пользовательских устройств, браузеров, мобильных платформ и оконных приложений. Еще это дает возможность распараллелить работу, вводя разделение труда с GUI программистами и даже дизайнерами, которые могут делать шаблоны отображения, но в код лезть не должны.

Идея, которая лежит в основе конструкционного шаблона MVC, очень проста: нужно чётко разделять ответственность за различное функционирование в наших приложениях.(Рисунок №2)

Рисунок 2-MVC

Приложение разделяется на три основных компонента, каждый из которых отвечает за различные задачи.

Контроллер управляет запросами пользователя (получаемые в виде запросов HTTP GET или POST, когда пользователь нажимает на элементы интерфейса для выполнения различных действий). Его основная функция — вызывать и координировать действие необходимых ресурсов и объектов, нужных для выполнения действий, задаваемых пользователем. Обычно контроллер вызывает соответствующую модель для задачи и выбирает подходящий вид.

Модель - это данные и правила, которые используются для работы с данными, которые представляют концепцию управления приложением. В любом приложении вся структура моделируется как данные, которые обрабатываются определённым образом. Что такое пользователь для приложения — сообщение или книга? Только данные, которые должны быть обработаны в соответствии с правилами (дата не может указывать в будущее, е-

mail должен быть в определённом формате, имя не может быть длиннее X символов, и так далее).

Вид обеспечивает различные способы представления данных, которые получены из модели. Он может быть шаблоном, который заполняется данными. Может быть несколько различных видов, и контроллер выбирает, какой подходит наилучшим образом для текущей ситуации. Веб приложение обычно состоит из набора контроллеров, моделей и видов. Контроллер может быть устроен как основной, который получает все запросы и вызывает другие контроллеры для выполнения действий в зависимости от ситуации.

ООП. Объектно-ориентированное программирование

ООП — это и ОО программирование и проектирование. Одно без другого бессмысленно чуть более чем полностью. Создано ООП для проектирования/программирования программных продуктов. Не для моделирования процессов. Не для проектирования протоколов, а именно для программных продуктов, для их реализации. Для упрощения системы, которая будет реализовывать протокол или бизнес-процесс или что-то еще.

Когда вы начинаете использовать ООП, первое что вы должны сделать — это начать использовать объектное мышление. Я уже когда-то говорил что это самая большая проблема ООП, научиться мыслить объектно очень сложно. И очень важно учиться это делать как можно раньше (GoF с аналогиями типа мост, конструктор, фасад очень в этом помогут). Используя объектное мышление, вы легко сможете проектировать сложные системы оперируя объектами и взаимодействием между ними. Т.е. ООП без объектного мышления не позволит вам начать использовать всю силу и мощь ООП.

Программные средства

Библиотека OpenCV

OpenCV является наиболее известной библиотекой компьютерного зрения. За долгое время своего существования она приобрела обширную аудиторию пользователей и стала, де-факто, стандартом в области компьютерного зрения. Множество алгоритмов, работающих «из коробки», открытость исходного кода, замечательная поддержка, большое сообщество пользователей и разработчиков, возможность пользоваться библиотекой на языках C, C++, Python (а также Matlab, C#, Java) под различными операционными системами — это далеко не полный список того, что позволяет OpenCV оставаться востребованной. Но OpenCV не стоит на месте — постоянно добавляется функционал.

Эта библиотека представляет собой набор модулей, каждый из которых связан с определенной областью компьютерного зрения. Существует стандартный набор модулей — так сказать, «must have» для любой задачи компьютерного зрения.

Все они представлены в основном репозитории OpenCV. Также существует репозиторий с дополнительными модулями, реализующими экспериментальную или новую функциональность. Требования к экспериментальным модулям, по понятным причинам, мягче. И, как правило, когда какой-то из таких модулей становится достаточно развитым, сформировавшимся и востребованным, он может быть перенесен в основной репозиторий.

Язык программирования Python

По сравнению с компилирующим, или строго типизированными языками, такими как C, C++ или Java, Python во много раз повышает производительность труда разработчика. Объем программного кода на языке Python обычно составляет треть, или даже пятую часть эквивалентного программного кода на языке C++ или Java, что означает меньший объем ввода с клавиатуры, меньшее количество времени на отладку и меньший объем трудозатрат на сопровождение. Кроме того, программы на языке Python запускаются сразу же, минуя длительные этапы компиляции и связывания, необходимые в некоторых других языках программирования, что еще больше увеличивает производительность труда программиста.

Программный код на Python читается легче, что значит, многократное его использование и обслуживание выполняется гораздо проще, чем использование программного кода на других языках сценариев. Python содержит самые современные механизмы многократного использования программного кода, каким является ООП.

Python используют:

Компания Google использует Python в своей поисковой системе и оплачивает труд создателя Python — Гвидо ван Россума;

Такие компании, как Intel, Cisco, Hewlett-Packard, Seagate, Qualcomm и IBM, используют Python для тестирования аппаратного обеспечения;

Служба коллективного использования видеоматериалов YouTube в значительной степени реализована на Python;

NSA использует Python для шифрования и анализа разведданных;

Компании JPMorgan Chase, UBS, Getco и Citadel применяют Python для прогнозирования финансового рынка;

Популярная программа BitTorrent для обмена файлами в пиринговых сетях написана на языке Python;

Популярный веб-фреймворк App Engine от компании Google использует Python в качестве прикладного языка программирования;

NASA, Los Alamos, JPL и Fermilab используют Python для научных вычислений.

Реализация

При реализации проекта возможно использоваться либо готовое серверное оборудование, либо использовать так называемые публичные облака.(Рисунок №3)

Рисунок 3- Схема публичного облака

При реализации в публичных облаках, например Amazon AWS, возможно лишь оплачивать лишь за потребляемую вычислительную нагрузку, в случае с серверами, это не представляется возможным, так как там происходит оплата по временному фактору, то есть не важно какая нагрузка, цена будет фиксированная. В случае работы с публичными облаками возможно масштабировать инфраструктуру в зависимости от нагрузки, при этом сохранять максимальную отказоустойчивость и масштабируемость.

Раньше, чтобы развернуть какое-либо приложение, приходилось покупать и настраивать собственные физические серверы. Такой подход

обладал большим количеством недостатков, например, если для нормальной работы приложения ему достаточно «полтора сервера», платить все равно приходилось за два – расходы на содержание и обслуживание инфраструктуры оказывались неоправданно высокими.

Сегодня у нас есть такие сервисы, которые позволяют настроить виртуальный сервер и хранилище данных под собственные нужды. В этом случае расходы зависят от необходимого количества вычислительных (и других) ресурсов – вы платите только за то, что используете.

Обратившись к облачным вычислениям, возможно получить возможность настраивать инфраструктуру по своему усмотрению, затрачивая на это меньшее количество средств и усилий. Иными словами, эта модель направлена на повышение доступности вычислительных ресурсов и сочетает в себе пять характеристик: самообслуживание по требованию, широкая доступность через Интернет, объединение ресурсов в пул, способность к быстрой адаптации и измеримость. Способность к быстрой адаптации – это возможность быстро масштабировать ресурсы под требуемую нагрузку.

Облачные системы автоматически контролируют и оптимизируют использование ресурсов. Это осуществляется путем измерения различных параметров (размер хранилища данных, вычислительная мощность, пропускная способность). Таким образом, возможно получать полную информацию об объеме оказанных/потребленных услуг.

Заключение

В ходе выполнения проекта будет получено веб-приложение, написанное на языке программирования Java с использованием таких фреймворков Spring Boot, ReactJS и другие. Основным преимуществом моей системы является глубокий анализ предметной области с целью получить продукт, который будет удобен для всех групп пользователей. Для преподавателя это возможность уменьшить количество рутинных действий связанных с заполнением бумаг. Для студентов это возможность получать актуальные данные это об своей успеваемости.

Литература

1. https://www.java.com/ru/
2. https://tproger.ru/
3. http://src-code.net/
4. https://javarush.ru/
5. https://habr.com/ru/
6. https://ruseller.com/