Литература

1. Саргсян, С. В. Оценка состояния изоляции обмотки двигателя при воздействии влаги / С. В. Саргсян // Вестник НПУА. Электротехника, Энергетика. – 2019. – № 2. – С. 52–59.

2. Диагностирование параметров качества изоляции обмоток трансформаторов при ее увлажнении / И. Л. Громыко, Д. В. Мирош, В. Н. Галушко, И. С. Евдасев // Вестник Белорусского государственного университета транспорта: наука и транспорт. – 2022. – № 2(45). – С. 15–19.

3. Стенд для послеремонтных испытаний асинхронных двигателей напряжением до 1 000 В / О. В. Владимиров, И. В. Ившин, М. Ф. Низамиев [и др.] // Известия высших учебных заведений. Проблемы энергетики. – 2019. – Т. 21, № 3–4. – С. 58–66.

А. Д. Михальков
(ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **В. Н. Кулинченко**, ст. преподаватель

ПРЕИМУЩЕСТВА И ПРИНЦИПЫ РЕАЛИЗАЦИИ ПАТТЕРНА MVC В SPRING FRAMEWORK

MVC – это не просто паттерн проектирования веб-приложений, он представляет собой комплекс из нескольких меньших шаблонов, что делает его мощным и гибким инструментом. Он разделяет веб-приложение на три основных компонента: модель, представление и контроллер.

Модель представляет собой объектную модель определенной области, содержащую данные и методы для работы с ними.

Представление отвечает за визуализацию данных и их представление пользователю.

Контроллер является связующим звеном между пользователем и системой.

Преимущества паттерна MVC:

– разделение ответственностей – MVC четко разделяет бизнес-логику, визуальное представление и обработку запросов, что улучшает структуру и расширяемость приложения;

– унификация кода – компоненты MVC становятся автономными, что облегчает их повторное использование в других частях приложения;

– читаемость и понятность кода – организация кода в MVC делает его более читаемым и понятным для других разработчиков;

– гибкость и масштабируемость – MVC обеспечивает высокую гибкость и масштабируемость приложения, что упрощает добавление новых функций и поддержку на различных платформах.

Реализация проекта на Spring включает следующие шаги:

1. Подготовка окружения – в начале проекта необходимо подготовить окружение для разработки, включая установку необходимых инструментов, таких как JDK (Java Development Kit) и среду разработки (IDE).

2. Установка Spring Framework – Spring Framework является одним из наиболее популярных инструментов для создания веб-приложений на Java. Его можно установить с помощью системы управления зависимостями, такой как Maven или Gradle.

3. Настройка конфигурации Spring – для настройки конфигурации Spring необходимо определить бины (компоненты Spring) и их взаимосвязи. Это можно сделать с помощью XML-конфигурации или аннотаций Java.

4. Настройка представлений – необходимо создать HTML-шаблоны или другие представления, которые будут отображать данные пользователю. Рекомендуется использовать технологии шаблонизации, такие как Thymeleaf или JSP, для вставки данных из модели в представление.

5. Создание классов модели, представления и контроллера – модель представляет собой объектную модель предметной области, представление отвечает за визуализацию данных, а контроллер обрабатывает запросы пользователя и взаимодействует с моделью и представлением.

6. Настройка маршрутизации запросов – маршрутизация запросов осуществляется с помощью конфигурации URL-шаблонов и их соотнесения с методами контроллеров.

7. Запуск и тестирование приложения – после завершения разработки приложения необходимо запустить его на локальном сервере и протестировать функциональность.

В. А. Мотолько
(БрГУ имени А. С. Пушкина, Брест)
Науч. рук. **Е. В. Зубей**, канд. физ.-мат. наук, доцент

ПРЕДОТВРАЩЕНИЕ КОНФЛИКТОВ ПРИ СОСТАВЛЕНИИ РАСПИСАНИЯ ЗАНЯТИЙ

Составление расписания учебных занятий на факультете – это одна из наиболее сложных и ответственных задач, с которой сталкиваются администраторы образовательных учреждений. Ручное составление расписания требует не только тщательного планирования, но и учета множества факторов, что делает этот процесс сложным и времязатратным. Поэтому базовым функционалом приложения для составления расписаний является механизм предотвращения конфликтов.

Конфликты в расписании могут произойти по различным причинам, включая:

1. Расписание преподавателя. Пересечение расписаний преподавателя могут создавать проблемы, когда один и тот же преподаватель должен проводить занятия в разных группа или разных курсах в одно и то же время.

2. Пересечение аудиторий. Данная проблема состоит в том, что несколько занятий должны проводиться в одной аудитории в одно и то же время.

3. Пересечение временных слотов. В заведениях с большим числом учащихся на факультете, студенты могут учиться в несколько смен. Это заставляет составителя рассматривать две предыдущие причины в нескольких сменах.

Для решения этих проблем существует механизм предотвращения конфликтов, суть которого в том, чтобы не дать составителю поставить занятие, приводящие к конфликту.

Помимо предотвращения, механизм также должен иметь и визуальную составляющую. Так, строки расписания, которые недоступны из-за того, что преподаватель уже занят, должны быть окрашены в какой-либо цвет, например, красный. Если строка недоступна из-за занятости аудитории, то её окрашивают в другой цвет, например, оранжевый. Это позволит составителю понять, что он может поставить занятие (дисциплина, преподаватель, аудитория) в эту строку, если изменит выбранную аудиторию. Если расписание преподавателя достигло максимальной недельной нагрузки, следует раскрасить все расписание в ещё один цвет.

Реализовать данных механизм достаточно просто. Имея входные данные в виде тройки: дисциплина, преподаватель, делаем следующие шаги:

1. Проверка текущей нагрузки преподавателя. Она может храниться в отдельной переменной или каждый раз рассчитываться, исходя из всего расписания (не является оптимальный вариантом). Если нагрузка равна максимальной, то вызываем функцию, закрашивающую строку расписания в любой цвет, для каждой строки, и прекращаем алгоритм. Если в результате работы программы нагрузка оказывается больше возможной недельной, стоит пересмотреть алгоритм её изменения.

2. Проверка занятости аудитории. Используя метод расписания, получаем массив ячеек в строке на всех курсах, смена которых совпадает со сменой активного курса.