А.И. Шадоба (УО «ГГУ имени Ф. Скорины», Гомель)
Науч. рук. **Н.А. Шаповалова**, старший преподаватель

АВТОМАТИЗАЦИЯ УЧЕТА КВАРТПЛАТЫ ДЛЯ УО «ГГУ ИМЕНИ Ф. СКОРИНЫ»

В основе данного проекта лежит задача автоматизация учета квартплаты. В разработке использовалась последняя версия программы «1С: Предприятие 8.3», что позволило, быстро и качественно используя новейшие возможности среды, разработать и при необходимости дорабатывать расчеты.

Необходимо было реализовать расчет по всем услугам, механизм перерасчета, а также отчеты, которые демонстрируют сумму начислений по квартире. Согласно заданию, для каждой квартиры создается справочник «Лицевой счет», в который помещаются действующие услуги, список жильцов и др. Затем в документе «Начисление услуг», также для каждой квартиры, заполняются необходимые данные, а именно: плательщик, услуга, тариф, количество (в зависимости от услуги это может быть, как количество человек, так и расход воды, электроэнергии и т.п.), сумма и перерасчет, если тариф изменялся, и документ за предыдущий месяц необходимо пересчитывать. Заполнение данных документа происходит с помощью кнопки Автозаполнение, которая была доработана по полям: количество, тариф, сумма и перерасчет.

А. И. Шадоба (УО «ГГУ имени Ф. Скорины», Гомель)
Науч. рук. **Н.А. Шаповалова**, старший преподаватель

ОСНОВНЫЕ МЕХАНИЗМЫ ДЛЯ АВТОМАТИЗАЦИЯ УЧЕТА КВАРТПЛАТЫ ДЛЯ УО «ГГУ ИМЕНИ Ф. СКОРИНЫ»

Суть автоматизации учета квартплаты состоит в том, чтобы система автоматически рассчитывала сумму для каждой услуги и заполняла необходимые данные в документе «Начисление услуг». Основная часть данной задачи была реализована вручную с использованием встроенного языка. Для каждой услуги создавалась отдельная функция.

При изменении какой-либо части, используемой при расчете и влияющей на сумму, в нашем случае это тариф, необходимо чтобы система автоматические отслеживала данные действия и пересчитывала записи, которые стали неактуальными. Для этих целей подходит механизм перерасчета. Перерасчет в системе представляет собой отдельную физическую таблицу, в которой хранится информация о пересчитываемом объекте.

Пользователь при необходимости может пересчитывать сумму с учетом нового тарифа с помощью кнопки Перерасчет в документе

Начисление услуг. Система автоматические проверит, если записи в таблице перерасчета, и при их наличии произведёт расчет, используя те же функции, что и при обычном расчете, затем поместит результат в поле Перерасчет в документе Начисление услуг.

Для создания отчета использовалась система компоновки данных. Данная система позволяет быстро и без ручного написания кода создать отчет, настроить его отображение, что значительно облегчает работу и уменьшает временные затраты.

А.Г. Шведов (УО «ГГУ имени Ф. Скорины», Гомель)
Науч. рук. **В.Д. Левчук,** канд. техн. наук, доцент

РЕАЛИЗАЦИЯ ПРОЕКТА СОЦИАЛЬНО СЕТИ НА МИКРОСЕРВИСНОЙ АРХИТЕКТУРЕ

При создании микросервисного приложения существует проблема взаимодействия микросервисов друг с другом. Так как каждый сервис использует различные технологии, то есть необходимость в разработке вспомогательного модуля, который отвечает за взаимодействие между сервисами. Для обеспечения совместной работы проекта микросервисов следует использовать набор основных паттернов и практик Микросервисной архитектуры. Многие из которых реализованы в Spring Cloud – что является расширением возможности Spring Boot в различные стороны.

Для конфингурации сервисов используется Spring Cloud Config. Это позволяет горизонтально масштабировать хранилище конфигураций для многофункциональнойсистемы. В качестве источника данных на данный момент поддерживаются Git и простые файлы, хранящиеся локально. В проекте Spring Cloud Config отдает файлы, соответствующие имени запрашивающего Spring приложения (для изменения можно брать настройки Spring profile из определенной ветки системы контроля версий).

Для автоматического определения сетевого адреса для допустимых частей приложения, которые могут динамически изменяться по причинам масштабирования, падений и обновлений используется Service discovery. В проекте используется реализация Eureka. В проете это реализованно с помощью добавления в каждом сервисе соответствующих анотаций, а также встроенного микросервиса eureka.

Для балансировки обращений к серверам используется Ribbon. Это client-side балансировщик.

```
@Bean
@LoadBalanced
RestTemplate restTemplate() {
 return new RestTemplate();
}
```