Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий – РТФ

Школа бакалавриата

ОТЧЕТ

По проекту
«Разработка модуля для расчета нагрузок (электрика, вода, тепло) на сайте концептуального проектирования и квартирографии»

по дисциплине «Проектный практикум»

Заказчик: Фамилия И.О. Чекалова М.Р.

Куратор: Фамилия И.О.
ученая степень, ученое звание, должность Кузнецов Д.С.

Студенты команды Jaysoft

Фамилия И.О. Ларин Т.Е.

Фамилия И.О. Брайденбихер В.Н.

Фамилия И.О. Мелешкин В.Е.

Фамилия И.О. Султанов М.Е.

Фамилия И.О. Татаринов К.И.

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Проект предполагает создание модуля, который на основе сгенерированных типов квартир и их площади рассчитывает нагрузки на инженерные сети (электрика, вода, тепло) и оценивает финансовые трудозатраты на их разработку.

Уровни результата (минимальный / базовый / оптимальный):

а) минимальный: реализация базового функционала расчета нагрузок на электрику, воду и тепло;

б) базовый: полноценный модуль с интеграцией в сайт концептуального проектирования;

в) оптимальный: готовый модуль с возможностью оценки финансовых трудозатрат на инженерные системы.

1 Цель проекта и задачи проекта

Создание модуля, который на основе сгенерированных типов квартир и их площади рассчитывает нагрузки на инженерные сети (электрика, вода, тепло) для оценки стоимости разработки инженерных систем на этапе концепции.

Задачи проекта:

1) ознакомиться со всеми алгоритмами расчетов;
2) создать тех. часть алгоритма по расчету воды;
3) создать тех. часть алгоритма по расчету тепла;
4) создать тех. часть алгоритма по расчету электроснабжения;
5) создать единый модуль со всеми тех. частями каждого алгоритмами.

2 Актуально и важность проекта

Из огромного объема подобных модулей или сервисов для нашего заказчика нет самого нужного для их задач, им нужен модуль или сервис, который по их алгоритмам будет делать нужные расчеты нагрузок, все существующие аналоги проводят расчеты не по нужным для нашего заказчика алгоритмам, поэтому мы создаем для них собственный модуль по их алгоритмам и прочим требованиям, для интеграции с их веб-сайтом.

3 Описание области применения продукта

Наш продукт будет принадлежат компании нашего заказчика и как итог наш созданный модуль будет проводить расчеты нагрузок на тепло, электроснабжение и воду для квартир, модуль будет помогать компании в их сфере деятельности.

4 Ожидаемые результаты

Модуль, интегрированный в сайт концептуального проектирования, для расчёта нагрузок на инженерные сети и оценки финансовых трудозатрат.

Обязательны след. расчёты:

– расчёт воды,

– расчёт тепла,

– расчёт электроснабжения.

ОСНОВНАЯ ЧАСТЬ

Успех проекта стал возможен благодаря слаженной работе команды. Четкое распределение задач, оперативное взаимодействие и профессиональный подход на каждом этапе позволили достичь высоких результатов в установленные сроки.

1 Информация о работе каждого участника

Для наглядного отображения вклада каждого члена команды в реализацию проекта были составлены следующие таблицы (таблица 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15). В них детально указаны ключевые задачи, выполненные участниками в рамках каждой контрольной точки, а также обобщенный вклад на протяжении всего жизненного цикла проекта.

Таблица 1 – Ларин Т.Е. в 1 КТ

Выполненная задача (1 КТ)	Затраченное время
Создание всех нужных пространств для работы (доска Kaiten, командное хранилище и т.д.)	1 час
Организация встреч с заказчиком для обсуждения плана действий по проекту	15 мин
Проведение анализа конкурентов (результат итоговый документ)	1 час
Подготовка всех нужных материалов и запись видео для защиты 1 КТ	3 часа

Таблица 2 – Ларин Т.Е. во 2 КТ

Выполненная задача (2 КТ)	Затраченное время
Выполнение обязанностей тимлида (проверка работ участников команды, установление дедлайнов и т.д.)	6-7 часов в период окончания 1 КТ до момента составления этой таблицы
Организация встреч с заказчиком для обсуждения плана действий по проекту	20 мин
Составление аналитических требований к проекту (бизнес-требования, пользовательские требования и т.д.) с учетом конфиденциальности проекта	2 часа

Продолжение таблицы 2

Составление и визуальное представление бизнес-процессов IDEF0 (1 часть)	30 минут
Составление таблицы работ всех участников команды	1.5 часа
Подготовка всех нужных материалов и запись видео для защиты КТ2	2 часа

Таблица 3 – Ларин Т.Е. в 3 КТ

Выполненная задача (3 КТ)	Затраченное время
Установление дедлайнов для участников команды и проверка прогресса по проекту	6 часов за все 3 КТ
Организация встреч с заказчиком для обсуждения плана действий по проекту	30 мин
Составление итогового отчета по проекту	4 часа
Подготовка всех нужных материалов и запись видео для защиты 3 КТ	6 часов

Таблица 4 – Мелешкин В.Е. в 1 КТ

Выполненная задача (1 КТ)	Затраченное время
Создание презентации для второй контрольной точки	1 час
Создание логотипа команды и помощь в аналитике	2 часа

Таблица 5 – Мелешкин В.Е. во 2 КТ

Выполненная задача (2 КТ)	Затраченное время
Создание презентации для второй контрольной точки	1 час

Продолжение таблицы 5

Составление бизнес-процессов и отрисовка схем к ним (1 часть)	30 минут
Коммуникация с целью получения необходимых для презентации материалов	1 час

Таблица 6 – Мелешкин В.Е. в 3 КТ

Выполненная задача (3 КТ)	Затраченное время
Создание презентации для третьей контрольной точки	1,5 часа
Оформление отчёта команды	4 часа

Таблица 7 – Брайденбихер В.И. в 1 КТ

Выполненная задача (1 КТ)	Затраченное время
Согласование распределения работ между backend разработчиками по разрабатываемому сервису	1 час
Изучение существующей платформы заказчика по расчету нагрузок	2 часа

Таблица 8 – Брайденбихер В.И. во 2 КТ

Выполненная задача (2 КТ)	Затраченное время
Изучение ТЗ заказчика — необходимого функционала модуля для расчета нагрузок воды	1-1.5 час
Реализация расширяемого модуля для расчета нагрузок воды на здание (водоснабжение, водоотвод)	6 часов
Написание Unit-тестов и тестирование созданного модуля	2 часа

Таблица 9 – Брайденбихер В.И в 3 КТ

Выполненная задача (3 КТ)	Затраченное время
Изучение ТЗ заказчика — необходимого функционала модуля для расчета нагрузок воды	1-1.5 часа
Реализация Domain-модуля расчета теплонагрузки на здание	~6 часов
Написание Unit-тестов Domain-модуля расчетов теплонагрузки в здании	2 часа
Написание репозитория получения удельной тепловой нагрузки для здания	~1.5 часа
Написание Unit-тестов репозитория получения удельной тепловой нагрузки	1 час
Реализация Domain-модуля расчета электронагрузки в здании	~8 часов
Написание Unit-тестов и Domain-модуля расчетов электронагрузки на здание	2 часа

Таблица 10 – Татаринов К.И. в 1 КТ

Выполненная задача (1 КТ)	Затраченное время
Согласование распределения работ между backend разработчиками по разрабатываемому сервису	1 час
Изучение существующей платформы заказчика по расчету нагрузок	2 часа

Таблица 11 – Татаринов К.И. во 2 КТ

Выполненная задача (2 КТ)	Затраченное время
Изучение ТЗ заказчика — необходимого функционала модуля для расчета нагрузок воды	1 час

Продолжение таблицы 11

Согласование распределения работ между backend разработчиками по разрабатываемому сервису	1 час
Разработка REST API с Swagger и кастомным обработчиком ошибок, а также настройка логирования	6 часов

Таблица 12 – Татаринов К.И. в 3 КТ

Выполненная задача (3 КТ)	Затраченное время
Изучение ТЗ заказчика — необходимого функционала модуля для расчета нагрузок воды	2 часа
Реализация Domain-модуля расчета теплонагрузки на здание	~6 часов
Написание Unit-тестов Domain-модуля расчетов теплонагрузки в здании	2 часа
Написание репозитория получения удельной тепловой нагрузки для здания	~1.5 часа
Написание Unit-тестов репозитория получения удельной тепловой нагрузки	1 час
Реализация Domain-модуля расчета электронагрузки в здании	~8 часов
Написание Unit-тестов и Domain-модуля расчетов электронагрузки на здание	2 часа

Таблица 13 – Султанов М.И. в 1 КТ

Выполненная задача (1 КТ)	Затраченное время
Согласование распределения работ между backend разработчиками по разрабатываемому сервису	1 час
Изучение существующей платформы заказчика по расчету нагрузок	2 часа

Таблица 14 – Султанов М.И. во 2 КТ

Выполненная задача (2 КТ)	Затраченное время
Интеграция с доменным слоем	3 часа
Согласование распределения работ между backend разработчиками по разрабатываемому сервису	1 час
Организовано медиаторное взаимодействие между слоями, реализованы CQRS-обработчики, настроена валидация входящих DTO с помощью FluentValidation	5 часов

Таблица 15 – Султанов М.И. в 3 КТ

Выполненная задача (3 КТ)	Затраченное время
Изучение ТЗ заказчика — необходимого функционала модуля для расчета нагрузок воды	2-3 часа
Реализация Domain модуля расчета теплонагрузки на здание	~6 часов
Написание Unit-тестов Domain модуля расчетов теплонагрузки в здании	2 часа
Написание репозитория получения удельной тепловой нагрузки для здания	~1.5 часа
Написание Unit-тестов репозитория получения удельной тепловой нагрузки	1 час
Реализация Domain-модуля расчета электронагрузки в здании	~8 часов
Написание Unit-тестов и Domain-модуля расчетов электронагрузки на здание	2 часа

2 Разбор требований заказчиков

2.1 Пользовательские требования

2.1.1 Интерфейсные требования

Интеграция функционала в веб-интерфейс с целью использования облегчения ввода данных и просмотра результатов расчётов.

2.2 Бизнес-требования

2.2.1 Цели проекта

Автоматизация расчёта нагрузок для ускорения проектирования, снижение количества ошибок при проведении расчётов вручную, а также обеспечение конфиденциальности данных.

2.2.2 Ограничения

Данные, используемые в проекте, являются конфиденциальными и не должны передаваться третьим лицам. Доступ к модулю осуществляется только для авторизированных лиц.

2.2.3 Критерии успеха

Сокращение времени расчёта нагрузок на 50% в сравнении с расчётами, проведёнными вручную. Соответствие расчётов нормативным требованиям заказчика.

2.3 Функциональные требования

2.3.1 Функциональные ожидания

Система, разработанная в процессе работы над проектом, должна отвечать следующим требованиям:

– система должна принимать входные данные;

– система должна автоматически определять количество типов квартир и секций на основе входных данных;

– система должна выполнять расчеты по водоснабжению, теплоснабжению и электроснабжению.

2.3.2 Входные данные

Параметры участка (площадь, этажность, тип застройки). Количество и типы квартир (1-к, 2-к, 3-к и т.д.). Количество лифтов, окон, инженерных систем.

2.3.3 Расчётные модули

Водоснабжение: расчёт расхода холодной и горячей воды, расчёт водоотведения, учёт нормативов потребления. Теплоснабжение: расчёт теплопотерь здания, определение мощности отопления и вентиляции. Электроснабжение: определение типов потребителей и их характеристик, определение установленных мощностей.

2.3.4 Формирование отчётности

Генерация сводных таблиц по нагрузкам, визуализация данных (графики, диаграммы).

2.4 Нефункциональные требования

2.4.1 Производительность

Время расчета нагрузок не должно превышать 5 минут для типового проекта. Система должна обрабатывать до 1000 квартир в одном расчёте.

2.4.2 Безопасность

Все данные должны храниться в зашифрованном виде, доступ к ним осуществляется только с помощью авторизации (логин, пароль и 2FA). Также используется обязательный запрет на копирование.

2.4.3 Надёжность

Обеспечение корректности расчётов (погрешность менее 2%). Резервное копирование данных раз в сутки.

2.4.4 Интеграция

API для передачи данных в power.inpad.store, а также поддержка импорта из CAD-систем.

2.5 Системные требования

2.5.1 Техническая платформа

Веб-приложение с backend на C#, frontend не подлежит реализации исходя из задач проекта (он заведомо реализован заказчиком самостоятельно).

2.5.2 Развёртывание

Облачный хостинг (AWS, Azure) или локальный сервер, поддержка Docker для изоляции среды.

2.5.3 Масштабируемость

Возможность увеличения вычислительных мощностей при росте нагрузки.

3 Анализ конкурентов

3.1 PlanPro

В ходе проекта были проанализированы конкуренты, данные конкуренты не являются прямыми, а косвенными, т. к. текущий проект — это отдельный модуль для заказчика, а конкуренты находятся в открытом доступе и могут быть куплены или бесплатно использованы.

PlanPro [1] — это многофункциональный онлайн-сервис, созданный для проектирования с модулем расчета нагрузок, интегрированным в основной функционал. Данный модуль ориентирован на профессиональных архитекторов и дизайнеров, однако нельзя точно сказать, по каким алгоритмам данный сервис проводит расчёты.

Стек разработки: веб-фреймворк React, backend на Node.js с использованием базы данных PostgreSQL. Расчеты выполняются с использованием Python.

Тарифы: подписка по уровням доступа с разным объемом ресурсов и функционала.

База данных: PostgreSQL для хранения данных проекта, внутренняя база для определенных норм и расчетов.

Плюсы: присутствует многофункциональность, высокая точность расчетов, имеется возможность интеграции с другими инструментами.

Минусы: высокая стоимость, сложный интерфейс для новичков (не интуитивно понятный при первой работе).

3.2 EasyCale

EasyCale [2] — Специализированный сервис для расчета нагрузок, простой и интуитивно понятный интерфейс, однако нельзя точно сказать по

какому алгоритму проводятся расчеты, предположительно не то, что нужны заказчику.

Стек разработки: веб-фреймворк Angular, backend на Python (Django), база данных MySQL.

Тарифы: платные расчеты поштучно или пакетные предложения.

База данных: MySQL для хранения данных пользователей и результатов расчетов.

Плюсы: очень простой в использовании, невысокая стоимость для небольших проектов.

Минусы: ограниченный функционал, абсолютно не подходит для больших и сложных проектов.

3.3 QuickSketch

QuickSketch [3] — сервис для быстрого эскизного проектирования с упрощенным модулем расчета нагрузок (не является специализированным в плане расчетов).

Стек разработки: веб-фреймворк Vue.js, backend на PHP, база данных SQLite. Упрощенные алгоритмы расчета (абсолютно не подходят для нашего проекта).

Тарифы: Freemium модель, базовый функционал бесплатный, расширенный — платный.

База данных: SQLite, используется для упрощения хранения данных.

Плюсы: быстрота работы, доступность сервиса.

Минусы: очень низкая точность расчетов, а также ограниченный функционал.

4 Архитектура программного продукта

На проекте данного семестра задачей было разработать только микросервис расчета различного рода нагрузок и параметров, приходящихся на здание. Выбранная архитектура разработки микросервиса: чистая архитектура (Clean Architecture) с разделением на:

– domain layer (ядро), куда входят сущности (HeatLoadCalculation, ElectricLoadCalculation), а также бизнес-правила и интерфейсы репозиториев;

– application layer, куда входят CQRS (Meditation-паттерн), валидация (FluentValidation), а также DTO для межслойного взаимодействия;

– infrastructure layer, то есть репозитории (доступ к данным по пятидневке и тепловой нагрузке);

– api layer, куда входят Rest Endpoints (ASP.NET Core), Swagger-документация, а также глобальная обработка ошибок.

В качестве внешних хранилищ данных используется база данных Postgres с PostGIS для хранения географической информации о городах, областях и температуры наиболее холодной пятидневки по городу из файлов стандартов, хранение географических данных обусловлено логикой определения наиболее холодной пятидневки — если города нет в таблице стандартов тогда определяется по ближайшему с выводом об этом информации пользователю.

Также для хранения некоторых данных простых небольших таблиц используются json файлы (к примеру, для удельной тепловой нагрузки на $м^2$ для многоэтажных зданий). Обусловленность такого выбора состоит в небольшом объеме хранимой информации и избыточности необходимости хранения таких данных в БД. Равно как и удобством в изменении данных, например при сопровождении системы если стандарты/требования с течением времени изменятся.

Более наглядно схема связей отображена на рисунке 1.

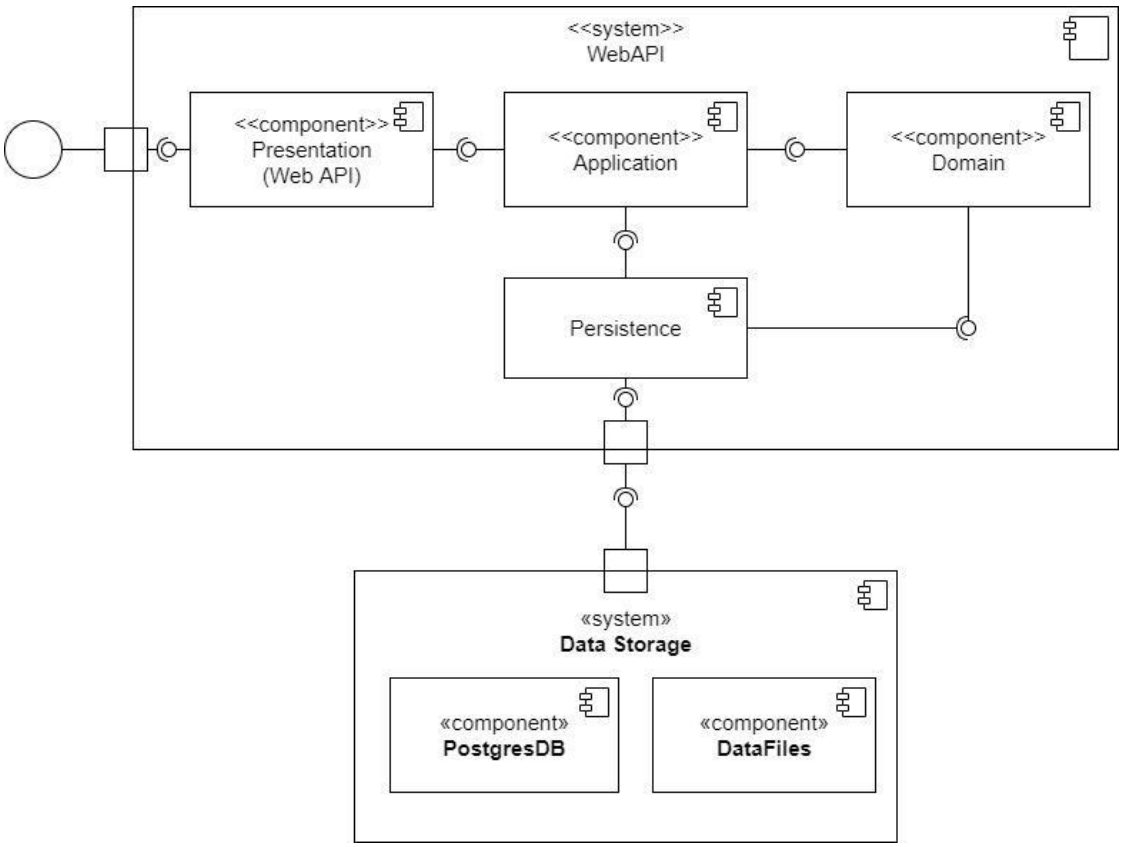

Рисунок 1 – Схема связей

5 Методологии разработки

5.1 Методология

Разработка итогового продукта проводилась по методологии Waterfall. Тесты же, в свою очередь, проводились при помощи unit-тестов. В последствии, по их результатам были исправлены ошибки, выявленные до сдачи продукта заказчику.

5.2 Планирование деятельности в ходе разработки

Брайденбихер В.Н. занимался Domain-логикой, тестированием Domain, а также написанием части репозиториев из слоя Infrastructure. Султанов М.И. занимался Application Layer, Infrastructure – CQRS, валидацией и обработкой исключений. Также частично занимался написанием репозиториев. Татаринов К.И. занимался Presentation Layer – написанием endpoints API, Swagger-документацией, интеграционным тестированием приложения.

ИНДИВИДУАЛЬНЫЙ ОТЧЕТ УЧАСТНИКОВ КОМАНДЫ

Индивидуальный отчёт тимлида и аналитика команды, Ларина Т.Е.: «В течение всего семестра я занимался установкой дедлайнов для участников команды, следил за их прогрессом по разработке, отмечал нарушения требований, если таковые имелись. В качестве аналитика вместе с другим аналитиком составил анализ конкурентов, бизнес-процессы, анализ требований заказчиков, после чего они были распределены на бизнес, функциональные и т. д. Весь семестр сопровождал всю команду связью с заказчиком, чтобы оперативно решать какие-либо вопросы или договариваться о встречах для решения вопросов. В доске Kaiten устанавливал карточки для участников команды, чтобы они следили за дедлайнами и выполняли поставленные для них задачи. В целом, за весь семестр мне понравилось работать с участниками команды и заказчиками, участники команды оперативно выполняли задачи и почти всегда без ошибок. Дедлайны соблюдали, разве что один раз забыли перенести вовремя карточку, из-за чего система в Kaiten выдала просрочку дедлайна, но задача была выполнена в срок. Заказчики постоянно были готовы отвечать на наши вопросы и помогать в решении наших проблем».

Индивидуальный отчёт дизайнера и аналитика команды, Мелешкина В.Е.: «В начале проекта, когда дизайн-составляющая оказалась минимальной, я сосредоточился на ключевых задачах: разработал логотип команды в первой контрольной точке и оформил все презентации для защит. Параллельно вместе со вторым аналитиком проводил анализ конкурентов и прорабатывал бизнес-процессы проекта. К финалу семестра занимался структурированием и оформлением итогового отчёта для сдачи. Работа в команде прошла продуктивно — заказчик оперативно реагировал на наши запросы, что помогло избежать бюрократических задержек. Несмотря на нулевую роль дизайна, проект дал ценный опыт в аналитике, командной координации и

презентационной работе. Особенно полезным было глубже понять, как устроены бизнес-процессы в реальных проектах».

Индивидуальный отчёт разработчика команды, Брайденбихера В.Н.: «В начале семестра совместно с командой backend-разработчиков было проведено детальное изучение технического задания и требований заказчика. В результате анализа мы приняли решение о реализации микросервиса с использованием принципов Clean Architecture, что обеспечило четкое разделение ответственности между слоями приложения и создало основу для построения гибкой, тестируемой системы. Основной зоной моей ответственности стала разработка доменного слоя (Domain Layer), включающего: базовые доменные модели данных, расчетные модули для трех видов нагрузок: водопотребления, тепловой и электрической, интерфейсы репозиториев для взаимодействия с уровнем данных. Разработанный доменный слой лег в основу следующих слоёв: слоя Persistence, где мной был реализован репозиторий для работы с данными по удельной тепловой нагрузке, а также слоя Application, отвечающего за координацию между компонентами, валидацию данных и управление бизнес-процессами. Особое внимание было уделено обеспечению надежности системы. Для этого мной была разработана система юнит-тестов, покрывающая ключевые сценарии работы расчетных алгоритмов, включая обработку пограничных случаев и исключительных ситуаций. На завершающем этапе выполнена работа по контейнеризации решения: создан Dockerfile с оптимальной конфигурацией, разработана docker-compose конфигурация для развертывания. Проделанная работа позволила создать стабильную, хорошо тестируемую архитектуру микросервиса, которая готова к дальнейшему развитию и интеграции с другими компонентами системы. Реализованное решение демонстрирует высокий уровень модульности и соответствует современным стандартам разработки».

Индивидуальный отчёт разработчика команды, Султанова М.И.: «В начале семестра я ознакомился с техническим заданием от заказчика, что

позволило понять основные требования и цели проекта. Совместно с командой разработчиков мы разработали архитектуру системы, определили ключевые компоненты и взаимодействия между ними. Также я освоил и изучил фреймворк ASP.NET, который является основой для разработки данного проекта, что обеспечило уверенное использование технологий и инструментов для реализации поставленных задач. Далее был реализован механизм медиаторного взаимодействия между слоями, что позволило централизовать обработку команд и запросов, повысить модульность и упростить масштабирование системы. После мной были созданы обработчики команд и запросов: CalculateWaterQuery/Handler — обработчик для расчета водопотребления, обработчики для вычислений теплонагрузки и электронагрузки, обработчик для получения наиболее холодной пятидневки по городу. Эти компоненты позволяют эффективно разделять операции чтения и записи, повышая производительность и масштабируемость системы. Также мной была произведена настройка валидации входящих данных, используя FluentValidation для проверки корректности входных DTO. Валидация была реализована согласно логике расчетов, что обеспечило надежность на этапе предварительной обработки данных. После был добавлен компонент PipelineBehavior, обеспечивающий автоматическую обработку исключений. Это повысило устойчивость системы и упростило управление ошибками, позволяя централизованно логировать и реагировать на исключения. В завершении работы мной была реализована логика преобразования DTO в доменные объекты и обратно, что поспособствовало разделению бизнес-логики и данных, а также обеспечило гибкость при изменениях в архитектуре. Также был создан репозиторий для получения температуры наиболее холодной пятидневки по городам Российской Федерации и проведено его тестирование. Моя работа за этот семестр позволила сформировать надежную основу для проекта, повысить его модульность, устойчивость и расширяемость. Я внес значительный вклад в разработку архитектуры и

реализацию ключевых компонентов, что обеспечит дальнейшее развитие системы и ее успешную эксплуатацию».

Индивидуальный отчёт разработчика команды, Татаринова К.И.: «В течение этого семестра мной была выполнена работа, направленная на создание и подготовку программного продукта к дальнейшему развитию и внедрению. Первым этапом я подробно изучил техническое задание от заказчика, что позволило понять ключевые требования и ожидаемые функциональные возможности системы. Ознакомление с ТЗ стало основой для дальнейшей разработки и проектирования архитектуры системы. Вместе с командой разработчиков была спроектирована архитектура программного продукта. Также была выбрана база данных - Postgres с расширением PostGIS. В процессе работы я освоил и активно использовал фреймворк ASP.NET Core, который является мощной и гибкой платформой для разработки веб-приложений и API. Благодаря его средствам мне удалось реализовать RESTful API. Он был задокументирован с использованием swagger, что обеспечило прозрачность и удобство взаимодействия с внешними системами и frontend. Также я реализовал кастомный обработчик исключений (middleware), обеспечивающий централизованное управление ошибками и их корректное отображение, настроил логирование с использованием Serilog, что позволило реализовать гибкое и расширяемое логирование процессов системы для последующего анализа и отладки, провел регистрацию и настройку дополнительных middleware и конфигураций, что повысило устойчивость и расширяемость системы. После всего вышеперечисленного я реализовал механизм преобразования DTO-запросов в команды и обратно, что обеспечивает корректное взаимодействие между уровнями системы и повышает удобство работы с данными. В завершении работы мной были написаны интеграционные тесты веб-приложения, что повысило стабильность и позволило своевременно выявить некоторые ошибки. В результате моей работы был сформирован удобный и структурированный API, который обладает четкой структурой ответов и ошибок, что обеспечивает его

готовность к интеграции с frontend-частью проекта и дальнейшему развитию, а также проверена работоспособность продукта на интеграционных тестах».

ЗАКЛЮЧЕНИЕ

Итоговый продукт соответствует требованиям заказчика на 100%, backend был реализован по всем алгоритмам (водоснабжение, тепло, электроснабжение), а заказчики остались довольны, так как все их требования были выполнены, в частности о соблюдение конфиденциальности проекта, которое также не было нарушено в ходе разработке итогового продукта.

В ходе оценки качества программного продукта в результате тестирования были выявлены неточности в алгоритмах. Некоторые расчёты из алгоритмов в технической части выдавали неправильные результаты из-за лишних строчек кода, проблема была устранена и после работоспособность продукта была восстановлена. Продукт на данный момент полностью работоспособен.

Реализованный продукт возможно улучшить следующими деталями:

1) реализовать для всех модулей общий frontend;

2) реализовать возможность кастомизации алгоритмов без вмешательства в код продукта;

3) оптимизировать возможность использования продукта сразу на нескольких запросах (т.е. если вдруг делаются расчеты по сложному проекту, то пока они делаются иметь возможность запросить рассчитать более слабый проект).

В ходе разработке продукта, были сделаны выводы о том, что:

1) в коммуникации с заказчиком зачастую стоит быть настойчивее, так как это имеет большое влияние на скорость разработки и общий настрой команды;

2) проекты, не нуждающиеся в frontend-разработке, имеют гораздо более высокую скорость разработки backend в силу отсутствия большой части классических проблем разработки и лишней коммуникации;

3) конфиденциальные проекты — более длительные в разработке, так как перепроверка соблюдения всех правил передачи данных отнимает

большое количество времени и заставляет увеличивать количество коммуникаций с заказчиком.

В результате проектного практикума был разработан рабочий микросервис, предоставляющий API для выполнения расчетов. Кодовая база снабжена документацией через Swagger и обладает масштабируемой архитектурой, что упрощает дальнейшее развитие функциональности. Также подготовлен Docker-образ, готовый к развертыванию в производственной среде, что обеспечивает удобство deployment и эксплуатации сервиса.

Подводя итоги, проект успешен. Итоговый продукт готов и передан заказчикам. Конфиденциальность проектов — не очень хороший фактор, из-за которого могут возникать дополнительные проблемы, на решение которых необходимо тратить время всех участников команды. Для работы над подобными проектами обязательно заранее наладить всю коммуникацию с заказчиками и заранее подготовить все важные материалы, чтобы избежать дополнительных проблем

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Официальный сайт PlanPro [Электронный ресурс] / Эстония. – Таллинне, 2008. – URL: https://planpro.ee/en/ (дата обращения: 06.04.2025).

2. EasyCalc [Электронный ресурс] / Ease-Calc – URL: https://www.nexans.com.au/en/Market-Solutions-and-Services/Services-and-Solutions/Smart-Tools/EasyCalc.html (дата обращения: 06.04.2025).

3. QuickSketch [Электронный ресурс] / QuickSketch. – URL: https://www.nexans.com.au/en/Market-Solutions-and-Services/Services-and-Solutions/Smart-Tools/EasyCalc.html (дата обращения: 07.04.2025).

ПРИЛОЖЕНИЕ А

(справочное)

материалы участников команды

1. Таблица с выполненными задачами каждого из участников команды Jaysoft на платформе Google Docs — https://docs.google.com/spreadsheets/d/1mppnDaXI0vvdhCnzji8v8kS1r6uffnzDR6zTEzakx8Y/edit?gid=0#gid=0.

2. Схема бизнес-процессов — https://docs.google.com/document/d/1qh0-36gvdlT4m6K1QEL79aM2vyQMd61CvdD3K4CSfOI/edit?usp=drive_link.