**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

**АРХАНГЕЛЬСКИЙ КОЛЛЕДЖ ТЕЛЕКОММУНИКАЦИЙ**

**ИМ. Б.Л. РОЗИНГА (ФИЛИАЛ) СПбГУТ**

**(АКТ (ф) СПбГУТ)**

СОГЛАСОВАНО

Рук. предприятия

\_\_\_\_\_\_\_\_\_\_\_ В.С. Федосеев

(Подпись) (И.О. Фамилия)

«10» мая 2025г.

**ТЕХНИЧЕСКИЙ ОТЧЁТ**

**по ПМ.11, ПМ.01**

|  |
| --- |
| ООО «КАКТУС» |
| Информационные системы и программирование |
| 09.02.07. 25ТО01. 024 ПЗ |
| (Обозначение документа) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИСПП-35 | |  | 10.05.2025 | А.Е. Туйкова |
|  | (Группа) | | (Подпись) | (Дата) | (И.О. Фамилия) |
| Рук. практики от предприятия | | |  | 10.05.2025 | В.Б. Елсуков |
|  | |  | (Подпись) | (Дата) | (И.О. Фамилия) |

Архангельск 2025

Содержание

[Перечень сокращений и обозначений 3](#_Toc198241867)

[Введение 4](#_Toc198241868)

[1 Охрана труда и техника безопасности при работе на ПК 6](#_Toc198241869)

[1.1 Общие требования безопасности 6](#_Toc198241870)

[1.2 Требования безопасности перед началом работы 6](#_Toc198241871)

[1.3 Требования безопасности во время работы 6](#_Toc198241872)

[1.4 Требования безопасности по окончанию работ 7](#_Toc198241873)

[1.5 Требования безопасности при аварийных ситуациях 7](#_Toc198241874)

[2 Выполнение работ по ПМ.11 8](#_Toc198241875)

[2.1 Проектирование базы данных 8](#_Toc198241876)

[2.2 Разработка базы данных и объектов базы данных 10](#_Toc198241877)

[2.3 Администрирование и защита базы данных 14](#_Toc198241878)

[3 Выполнение работ по ПМ.01 18](#_Toc198241879)

[3.1 Проектирование программного обеспечения 18](#_Toc198241880)

[3.2 Назначение и область применения 18](#_Toc198241881)

[3.3 Постановка задачи 18](#_Toc198241882)

[3.4 Описание алгоритма функционирования системы 19](#_Toc198241883)

[3.5 Разработка программных модулей 19](#_Toc198241884)

[3.6 Разработка мобильного приложения 20](#_Toc198241885)

[3.7 Отладка и тестирование программных модулей 20](#_Toc198241886)

[3.8 Оптимизация и рефакторинг программного кода 21](#_Toc198241887)

[Заключение 22](#_Toc198241888)

[Список использованных источников 23](#_Toc198241889)

Перечень сокращений и обозначений

В настоящем техническом отчете применяются следующие сокращения и обозначения:

БД – база данных

ПК – персональный компьютер

ПО – программное обеспечение

СУБД – система управления базами данных

API – интерфейс прикладного программирования

CRUD – создание, чтение, обновление, удаление

DDL – группа команд для создания, изменения и удаления объектов базы данных

ERD – диаграмма «сущность-связь»

ER-модель – модель «сущность-связь»

FK – внешний ключ

MVVM – Model-View-ViewModel

PK – первичный ключ

SQL – структурированный язык запросов

UML – унифицированный язык моделирования

Введение

Местом производственной практики является ООО «Кактус». Предприятие разрабатывает продукт, который оптимизирует работу светофоров, динамически меняя время их работы для уменьшения пробок на перекрестках.

Цели производственной практики:

* получение практического опыта по выполнению работ по ПМ.11 «Разработка, администрирование и защита баз данных» и развитие общих и профессиональных компетенций;
* получение практического опыта по выполнению работ по ПМ.01 «Разработка модулей программного обеспечения для компьютерных систем» и развитие общих и профессиональных компетенций.

Задачами производственной практики являются:

* осуществление сбора, обработки и анализа информации для проектирования БД;
* проектирование БД;
* разработка объектов БД;
* реализация БД в конкретной СУБД;
* администрирование БД;
* защита информации в БД с использованием технологии защиты информации;
* формирование алгоритма работы программных модулей;
* разработка программных модулей;
* выполнение отладки программных модулей с использованием специализированных программных средств;
* выполнение тестирования программных модулей;
* осуществление рефакторинга и оптимизации программного кода;
* разработка модуля программного обеспечения для мобильных платформ.

Предприятием предоставлено рабочее место с персональным компьютером и необходимым аппаратным и программным обеспечением:

* процессор: Intel Xeon E5-2640 2.40 GHz;
* системная плата: HUANANZHI X99-8M-F;
* видеокарта: NVIDIA GeForce GTX 1070;
* оперативная память – 16 ГБ;
* операционная система: Microsoft Windows 10 Pro;
* прикладное ПО: Microsoft Office, Visual Studio, Microsoft SQL Server Management Studio 20, draw.io, Яндекс Браузер, Android Studio.

# Охрана труда и техника безопасности при работе на ПК

## Общие требования безопасности

Настоящая инструкция определяет основные требования охраны труда и техники безопасности при работе за ПК на предприятии.

К самостоятельной работе допускаются:

* лица старше 18 лет, прошедшие медицинский осмотр и не имеющие противопоказаний к работе по данной специальности;
* прошедшие первичный инструктаж на рабочем месте.

## Требования безопасности перед началом работы

Перед началом работ обязательно:

* проветрить рабочее помещение;
* визуально осмотреть рабочее место, убедиться в отсутствии видимых повреждений оборудования и проводов;
* проверить правильность подключения оборудования к электросети;
* отрегулировать высоту и наклон рабочего кресла, а также положение монитора для обеспечения удобной и эргономичной позы.

## Требования безопасности во время работы

Во время работы сотрудник обязан:

* содержать в порядке и чистоте свое рабочее место;
* соблюдать оптимальное расстояние от экрана монитора до глаз;
* регулярно делать перерывы в работе;
* следить за вентиляцией и охлаждением оборудования;
* в случае плохого самочувствия во время работы сообщить об этом руководителю и обратиться за медицинской помощью.

## Требования безопасности по окончанию работ

По окончании работ сотрудник обязан:

* корректно завершить работу всех программ и ПК;
* привести в порядок рабочее место;
* убедиться, что все электрооборудование выключено;
* сообщить о неисправностях и других замечаниях при работе с ПК непосредственному руководителю или лицам, осуществляющим техническое обслуживание оборудования.

## Требования безопасности при аварийных ситуациях

В случае аварийных ситуаций необходимо:

* при возгорании электропроводки, оборудования отключить электропитание и принять меры по тушению пожара с помощью имеющихся первичных средств пожаротушения, сообщить о происшедшем непосредственному руководителю;
* в случае внезапного ухудшения здоровья прекратить работу, выключить оборудование, сообщить об этом руководителю и при необходимости обратиться к врачу;
* при повреждении оборудования, кабелей, проводов, неисправности заземления, появлении запаха гари, возникновении постороннего шума и других неисправностях немедленно отключить электропитание оборудования и сообщить о случившемся непосредственному руководителю и лицу, осуществляющему техническое обслуживание оборудования.

# Выполнение работ по ПМ.11

## Проектирование базы данных

ООО «Кактус» требуется вести учет и систематизацию данных о перекрестках, светофорах и событиях. Для хранения и обработки данных необходимо использовать БД.

В БД следует хранить информацию о регулируемых перекрестках, которые обслуживает предприятие. Каждый перекресток имеет уникальный номер, широту и долготу (уникальные), количество полос, наличие поворотных полос и дату последнего обновления (необязательно).

В списке событий содержится тип событий, время начала и конца события, описание события (необязательно), уровень влияния на трафик.

Требуется хранить информацию о светофорах. У каждого светофора есть уникальный номер, тип светофора, состояние и дата установки (необязательно).

На рисунке 1 показана концептуальная модель предметной области в виде ERD, созданная с помощью средств проектирования Draw.io.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1 – Концептуальная модель

CASE-средство проектирования БД Microsoft SQL Server, выбранная для хранения данных, является реляционной, поэтому требуется преобразовать разработанную концептуальную модель в логическую с учетом правил преобразования ER-модели в реляционную модель данных:

* связь между сущностями Перекресток и Светофор 1:М и обязательна для обеих сущностей, поэтому формируется два отношения с внешним ключом в отношении Светофор;
* связь между сущностями Перекресток и События М:М и необязательна для обеих сущностей, поэтому формируется три отношения с внешними ключами в отношении ПерекрестокСобытие.

На рисунке 2 показана логическая модель предметной области, полученная путем преобразования концептуальной модели и созданная с помощью средства проектирования Draw.io.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2 – Логическая модель

На рисунке 3 показана физическая модель предметной области, разработанная на основе логической модели для СУБД SQL Server 2022.

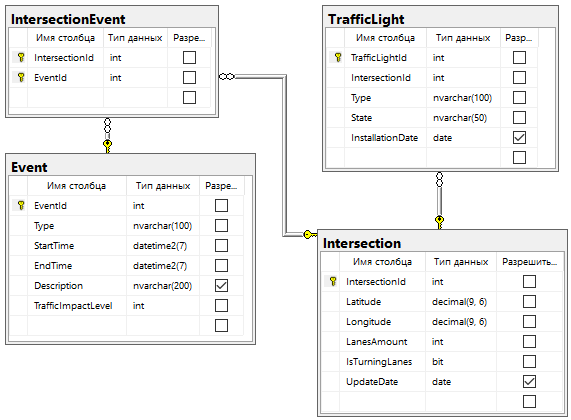


Рисунок 3 – Физическая модель

## Разработка базы данных и объектов базы данных

В таблице 1 в виде словаря данных представлено описание созданных таблиц и ограничений целостности БД.

Таблица 1 – Словарь данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ключ** | **Поле** | **Тип данных** | **Обязательное** | **Примечание** |
| Intersection | | | | |
| PK | IntersectionId | int | + | Автоинкрементный |

*Продолжение таблицы 1*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ключ** | **Поле** | **Тип данных** | **Обязательное** | **Примечание** |
|  | Longitude | decimal(9,6) | + | Совместно уникально с Latitude |
|  | LanesAmount | int | + |  |
|  | UpdateDate | date | - |  |
|  | IsTurningLanes | bit | + |  |
|  | Latitude | decimal(9,6) | + | Совместно уникально с Longitude |

Для создания таблиц в БД и ограничений целостности требуется выполнить DDL команды. Код создания таблицы Перекрестки представлен листингом 1.

Листинг 1 – SQL-запрос для создания таблицы Intersection

-- Создание таблицы Intersection с полями и первичным ключом

CREATE TABLE Intersection (

IntersectionId INT PRIMARY KEY IDENTITY(1,1),

Latitude DECIMAL(9,6) NOT NULL,

Longitude DECIMAL(9,6) NOT NULL,

LanesAmount INT NOT NULL,

IsTurningLanes BIT NOT NULL,

UpdateDate DATE NULL

);

Для решения поставленных задач для таблиц БД требуется создать уникальные ограничения в соответствии со словарем данных, используя команды DDL. Код создания уникальных ограничений для таблиц представлен листингом 2.

Листинг 2 – SQL-запрос для создания уникального ограничения

-- Создание уникального ограничение на Latitude и Longitude

CREATE UNIQUE INDEX UQ\_Intersection\_Latitude\_Longitude

ON Intersection (Latitude, Longitude)

Таблицы необходимо заполнить данными. Код заполнения таблицы Intersection представлен в листинге 3.

Листинг 3 – SQL-запрос для заполнения таблицы Intersection данными

-- Добавление данных в таблицу Перекрестки

INSERT INTO Intersection (Latitude, Longitude, LanesAmount, IsTurningLanes, UpdateDate)

VALUES

(55.755826, 37.617300, 4, 1, '2025-04-01'),

(40.730610, -73.935242, 6, 0, '2025-03-15'),

(51.507351, -0.127758, 3, 1, '2025-02-28');

Для решения поставленных задач в БД требуется создать следующие представления:

* представление EventTrafficImpactView, которое показывает информацию о событиях и их уровне воздействия на трафик, с возможностью фильтрации за последние 30 дней (код создания представления показан в листинге 4);
* представление TrafficLightSummaryView, предоставляющее сводную информацию о светофорах и связанных с ними перекрестках.

Листинг 4 – Код создания представления EventTrafficImpactView

--Информацию о событиях, которые произошли за последние 30 дней и могут повлиять на трафик

CREATE VIEW EventTrafficImpactView AS

SELECT

EventId,

Type AS EventType,

StartTime,

EndTime,

Description,

TrafficImpactLevel,

DATEDIFF(minute, StartTime, EndTime) AS EventDurationMinutes

FROM

Event

WHERE

EndTime >= GETDATE() - 30;

Для решения поставленных задач в БД требуется создать следующие подпрограммы:

* триггер tr\_AddedEvent, который записывает время добавления нового события в лог-таблицу EventLog (код создания триггера представлен в листинге 5);
* хранимую процедуру AddIntersection, которая добавляет данные о новом перекрестке на основе обязательных данных для заполнения (код создания хранимой процедуры показан в листинге 6).

Листинг 5 – Код создания триггера tr\_AddedEvent

--Триггер, добавляющий время создания события в лог-таблицу

CREATE TRIGGER tr\_AddedEvent

ON Event

AFTER INSERT

AS

BEGIN

INSERT INTO EventLog (EventId, EventType, LogMessage, LogTimestamp)

SELECT

i.EventId,

i.Type,

'Новое событие добавлено',

GETDATE()

FROM inserted AS i;

END;

Листинг 6 – Код создания хранимой процедуры AddIntersection

CREATE PROCEDURE [dbo].[AddIntersection]

@Latitude DECIMAL(9,6),

@Longitude DECIMAL(9,6),

@LanesAmount INT,

@TurningLanesPresent BIT,

@LastUpdateDate DATE

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO Intersection (Latitude, Longitude, LanesAmount, IsTurningLanes, UpdateDate)

VALUES (@Latitude, @Longitude, @LanesAmount, @TurningLanesPresent, @LastUpdateDate);

SELECT SCOPE\_IDENTITY() AS IntersectionId;

END

## Администрирование и защита базы данных

Необходимо установить Microsoft SQL Server Management Studio 20 и SQL Server 2022 на предприятии. Для установки SQL Server 2022 необходимо выполнить следующие шаги:

* скачать exe-файл SQL Server 2022 Developer Edition с официального сайта;
* открыть exe-файл SQL Server 2022 и выбрать базовый тип установки;
* дождаться полной установки. На рисунке 4 представлено завершение установки SQL Server.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4 – SQL Server 2022. Вид окна завершения установки

Для установки Microsoft SQL Server Management Studio 20 необходимо выполнить следующие шаги:

* открыть exe-файл Microsoft SQL Server Management Studio 20;
* выбрать путь для установки;
* начать установку
* дождаться полной установки.

На рисунке 5 представлено завершение установки Microsoft SQL Server Management Studio 20. По окончании установки необходимо запустить приложение и проверить корректность его работы.

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 5 – SQL Server Management Studio 20. Вид окна завершения установки

Для защиты информации в БД требуется создать отдельные имена для входа и пользователей для аналитика данных и разработчика, который будет отслеживать таблицы для обнаружения проблем и ошибок при записи данных.

У аналитика данных должна быть возможность просмотра всех таблиц в БД, а у разработчика возможность чтения и записи. Листингом 7 представлен набор команд для создания пользователей, имен входов и разграничение прав.

Листинг 7 – Код создания пользователей и имен входов

--Создание имен входов и пользователей

CREATE LOGIN DataAnalyst WITH PASSWORD = 'AnalystCactus';

CREATE USER DataAnalyst FOR LOGIN DataAnalyst;

CREATE LOGIN DeveloperUser WITH PASSWORD = 'DeveloperCactus';

CREATE USER DeveloperUser FOR LOGIN DeveloperUser;

--Разграничение прав доступа для пользователей

GRANT SELECT ON SCHEMA::dbo TO DataAnalyst;

GRANT SELECT, INSERT, UPDATE, DELETE ON SCHEMA::dbo TO DeveloperUser;

Во избежание потери данных необходимо периодически выполнять полное резервное копирование БД. Резервные копии позволяют восстановить данные после сбоя и других непредвиденных проблем. Для выполнения резервного копирования БД требуется выполнить SQL-скрипт, представленный листингом 8.

Листинг 8 – Код создания резервной копии

--Создание резервной копии по указанному пути

BACKUP DATABASE [pr] TO DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup\pr.bak' WITH NOFORMAT, NOINIT, NAME = N'pr-Полная База данных Резервное копирование', SKIP, NOREWIND, NOUNLOAD, STATS = 10

GO

Для восстановления данных из созданной резервной копии требуется выполнить SQL-скрипт, представленный листингом 9.

Листинг 9 – Код восстановления резервной копии

--Восстановление БД по указанной резервной копии

BACKUP LOG [pr] TO DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup\pr\_LogBackup\_2025-05-04\_14-57-11.bak' WITH NOFORMAT, NOINIT, NAME = N'pr\_LogBackup\_2025-05-04\_14-57-11', NOSKIP, NOREWIND, NOUNLOAD, NORECOVERY , STATS = 5

RESTORE DATABASE [pr] FROM DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup\pr.bak' WITH FILE = 1, NOUNLOAD, STATS = 5

GO

# Выполнение работ по ПМ.01

## Назначение и область применения

ООО «Кактус» осуществляет контроль за дорожным движением. Необходимо уведомлять водителей о возможных задержках движения на перекрестках.

Предприятием поставлена задача разработать мобильное приложение для информирования водителей о времени ожидания на светофоре. Система использует БД для определения событий на перекрестках, которые усложняют движение. Наиболее загруженные участки движения помечаются цветовым индикатором.

Требуется разработать однопользовательское Android-приложение для отслеживания трафика на перекрестках. Для получения данных из БД в приложение необходимо разработать API.

Основные функции приложения:

* возможность просмотра событий и светофоров, связанных с перекрестком;
* соответствующая цветовая индикация загруженности перекрестка;
* возможность добавления нового события.

Действия, доступные пользователю приложения, отображены на диаграмме прецедентов, представленной на рисунке 6.

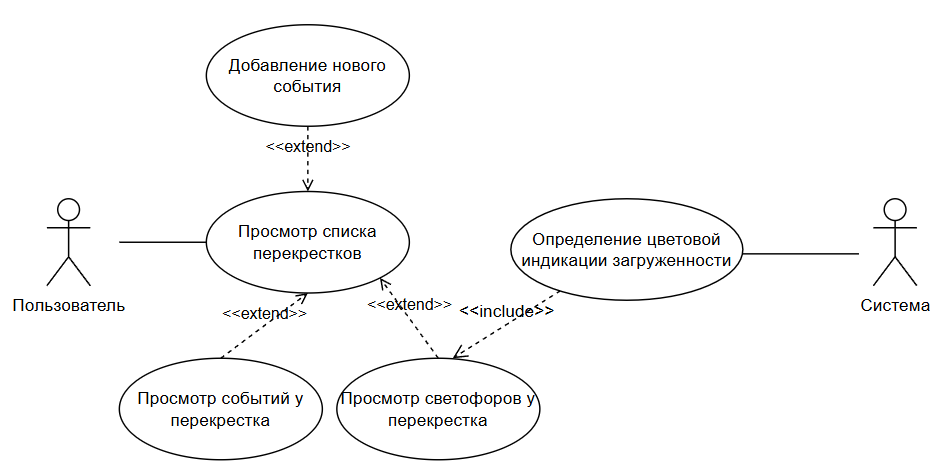


Рисунок 6 – Draw.io. Вид окна диаграммы прецедентов

## Разработка программных модулей

API требуется реализовать на языке С#, исходя из соображений безопасности кода, и с помощью технологии ASP.Net Core.

Для разработки API будет использоваться интегрированная среда разработки программ Microsoft Visual Studio Community 2022, так как это быстрая, производительная и простая в освоении IDE. Она позволяет работать с различными языками программирования, в том числе C#.

Для работы с данными модели Intersection был разработан контроллер IntersectionsController. Пример GET-метода получения данных о всех перекрестках представлен в листинге 10.

Листинг 10 – Код получения данных о перекрестках

//GET-метод получения данных о перекрестках

[HttpGet]

public async Task<ActionResult<IEnumerable<Intersection>>> GetIntersections()

{

try

{

var intersections = await \_context.Intersections.ToListAsync();

if (intersections == null || intersections.Count == 0)

return NotFound("Данные не найдены");

return intersections;

}

catch (Exception ex)

{

return StatusCode(500, "Ошибка сервера");

}

}

Для реализации возможности добавления нового события в БД был создан контроллер EventsController. Пример POST-метода представлен в листинге 11.

Листинг 11 – Код добавления нового события

// POST-метод создания нового события

[HttpPost]

public async Task<ActionResult<Event>> PostEvent(Event newEvent)

{

try

{

if (newEvent == null)

{

return BadRequest("Событие необходимо заполнить.");

}

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

\_context.Events.Add(newEvent);

await \_context.SaveChangesAsync();

return CreatedAtAction("GetEvent", new { id = newEvent.EventId }, newEvent);

}

catch (Exception ex)

{

return StatusCode(500, "Ошибка сервера.");

}

}

## Разработка мобильного приложения

Мобильное приложение разрабатывается под Android с применением языка Kotlin. Средой разработки выступает Android Studio, предоставляющая мощный инструментарий для создания, отладки и тестирования Android-приложений, а также полную поддержку Kotlin.

В архитектуре приложения используется модель MVVM. Модель представляет собой слой данных, содержащий классы, отвечающие за получение и хранение информации. ViewModel, в свою очередь, содержит логику обработки пользовательских действий, занимается подготовкой и трансформацией данных, а также обеспечивает связь между моделью и представлением.

На рисунке 7 представлено отображение одного из элементов списка Intersections.

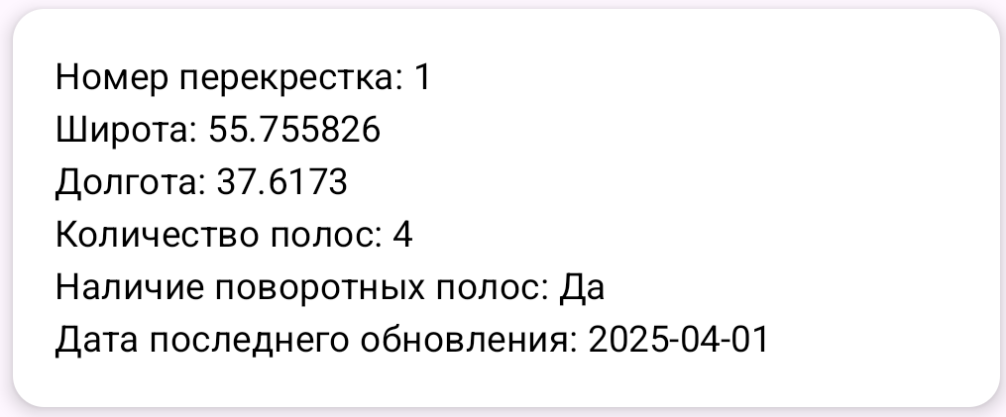


Рисунок 7 – TrafficTrackingApp. Вид элемента списка Intersections

При нажатии на перекресток раскрывается информация о светофорах и событиях, которые связаны с выбранным перекрестком. Пример отображения светофоров приведено на рисунке 8.

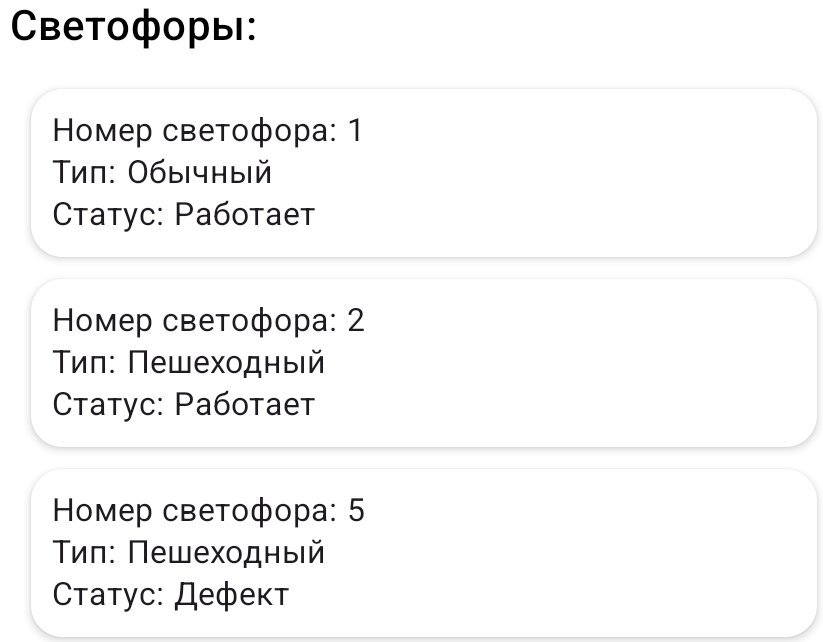


Рисунок 8 – TrafficTrackingApp. Вид отображения светофоров

На рисунке 9 представлено отображение событий, связанных с перекрестком, и индикация по влиянию на трафик.

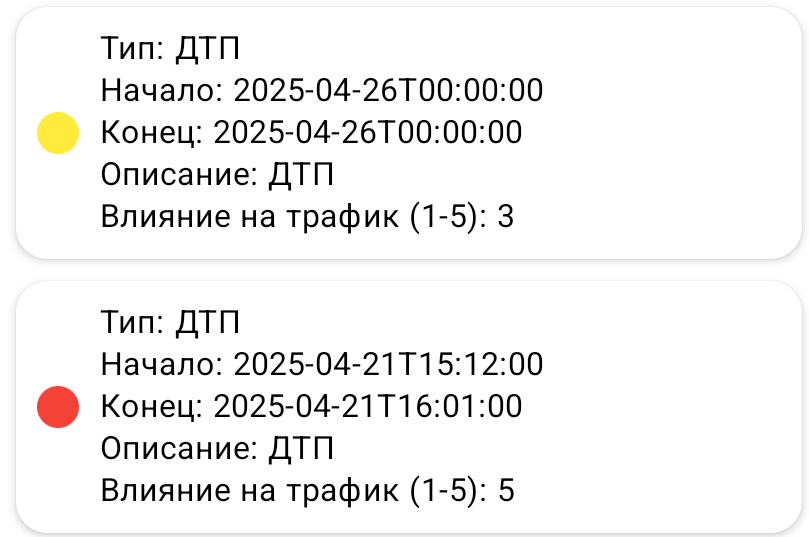


Рисунок 9 – TrafficTrackingApp. Вид отображения событий

В верхней панели на главном экране приложения при нажатии кнопки «+» откроется окно для добавления события. Пример отображения окна представлен на рисунке 10.

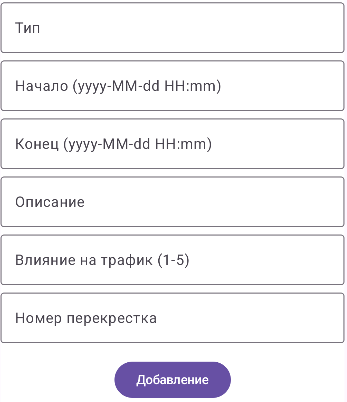


Рисунок 10 – TrafficTrackingApp. Вид окна добавления нового события

## Отладка и тестирование программных модулей

Для отладки приложения используется комплекс средств Visual Studio 2022, который позволяет произвести диагностику и локализацию ошибок, внести исправления в программу с целью их устранения.

Стандартные средства отладчика Visual Studio 2022:

* для установки точки останова необходимо нажать клавишу F9, когда курсор находится на необходимой строке;
* для включения/отключения точек останова необходимо использовать комбинацию Ctrl+F9;
* для запуска отладки с остановками на точках останова необходимо нажать F5;
* для пошагового прохода без захода/с заходом в методы необходимо использовать F10 и F11 соответственно.

Процесс отладки представлен на рисунке 11.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11 – Visual Studio 2022. Расстановка точек остановки

В таблице 2 представлен набор тест-кейсов для тестирования приложения.

Таблица 2 – Тест-кейсы

|  |  |  |
| --- | --- | --- |
| **Действие** | **Ожидаемый результат** | **Фактический результат** |
| Нажатие на элемент перекрестка в списке | Открывается окно с детальной информацией о выбранном перекрестке | Совпадает с ожидаемым |
| В окне добавления нового события ввести в соответствующие поля ввода корректные данные: Тип – «ДТП», начало – «2025-04-13 16:00», | Новое событие добавляется в БД в таблицу Event и IntersectionEvent. Отображение нового события в приложении | Совпадает с ожидаемым |

*Продолжение таблицы 2*

|  |  |  |
| --- | --- | --- |
| **Действие** | **Ожидаемый результат** | **Фактический результат** |
| конец – «2025-04-13 17:00», влияние на трафик – 3, код перекрестка – 1. Нажать на кнопку «Добавление» |  | Совпадает с ожидаемым |
| В окне добавления нового события ввести в соответствующие поля ввода некорректные данные:  Тип – «ДТП», начало – «2025-04-13 16:00», конец – «2025-04-13 17:00», влияние на трафик – 7, код перекрестка – 11. Нажать на кнопку «Добавление» | Отображение окна с сообщением об ошибке некорректного ввода данных в поле ввода влияние на трафик | Совпадает с ожидаемым |

Для тестирования по сценарию необходимо создать UNIT-тест. Пример тестирования приведен в листинге 12.

Листинг 12 – Код UNIT-тестирования добавления нового события

//UNIT-тест добавления нового события

[Fact]

public async Task PostEvent\_WithValidData\_ReturnsCreatedAtAction()

{

var newEvent = new Event

{

Type = "Проишествие",

StartTime = DateTime.Now,

EndTime = DateTime.Now.AddHours(1),

TrafficImpactLevel = 3

};

int intersectionId = 1;

var intersection = new Intersection { IntersectionId = intersectionId };

\_mockContext.Setup(c => c.Intersections.FindAsync(intersectionId))

.ReturnsAsync(intersection);

\_mockContext.Setup(c => c.Events.Add(newEvent));

\_mockContext.Setup(c => c.SaveChangesAsync(It.IsAny<CancellationToken>()))

.ReturnsAsync(1);

var result = await \_controller.PostEvent(newEvent, intersectionId);

var actionResult = Assert.IsType<ActionResult<Event>>(result);

var createdAtActionResult = Assert.IsType<CreatedAtActionResult>(actionResult.Result);

Assert.Equal("GetEvent", createdAtActionResult.ActionName);

Assert.Equal(newEvent.EventId, createdAtActionResult.RouteValues["id"]);

Assert.Equal(newEvent, createdAtActionResult.Value);

Assert.Contains(intersection, newEvent.Intersections);

}

## Оптимизация и рефакторинг программного кода

Необходимо провести рефакторинг кода. В методе отображения данных представлен дублирующийся код. Требуется выделить общий компонент для карточек. Примеры по устранению дублирования и создания Composable-функции CommonCardItem представлены в листингах 13 и 14.

Листинг 13 – Код до рефакторинга

//Карточка для отображения информации о светофоре

Card(

modifier = Modifier

.fillMaxWidth()

.padding(vertical = 4.dp),

elevation = CardDefaults.cardElevation(defaultElevation = 2.dp),

colors = CardDefaults.cardColors(

containerColor = Color.White

)

) {

Column(

modifier = Modifier.padding(8.dp)

) {

Text(text = "Номер светофора: ${trafficLight.trafficLightId}", style = MaterialTheme.typography.bodySmall)

Text(text = "Тип: ${trafficLight.type}", style = MaterialTheme.typography.bodySmall)

Text(text = "Статус: ${trafficLight.state}", style = MaterialTheme.typography.bodySmall)

}

}

Листинг 14 – Код после рефакторинга

//Создание шаблона для карточки и его использование

@Composable  
fun CommonCardItem(  
 modifier: Modifier = Modifier,  
 backgroundColor: Color = Color.White,  
 contentColor: Color = MaterialTheme.colorScheme.onBackground,  
 elevation: Dp = 2.dp,  
 verticalPadding: Dp = 4.dp,  
 horizontalPadding: Dp = 0.dp,  
 contentPadding: Dp = 8.dp,  
 content: @Composable ColumnScope.() -> Unit  
) {  
 Card(  
 modifier = modifier  
 .fillMaxWidth()  
 .padding(vertical = verticalPadding, horizontal = horizontalPadding),  
 elevation = CardDefaults.cardElevation(defaultElevation = elevation),  
 colors = CardDefaults.cardColors(  
 containerColor = backgroundColor,  
 contentColor = contentColor  
 )  
 ) **{** Column(  
 modifier = Modifier.padding(contentPadding),  
 content = content  
 )  
 **}**}

@Composable

fun TrafficLightItem(trafficLight: TrafficLight) {

CommonCardItem(

backgroundColor = Color.White,

contentColor = MaterialTheme.colorScheme.onBackground

) {

Text(text = "Номер светофора: ${trafficLight.trafficLightId}",

style = MaterialTheme.typography.bodySmall)

Text(text = "Тип: ${trafficLight.type}",

style = MaterialTheme.typography.bodySmall)

Text(text = "Статус: ${trafficLight.state}",

style = MaterialTheme.typography.bodySmall)

}

}

Для проведения оптимизации программного кода необходимо реализовать lazy-загрузку компонентов. Это позволит оптимизировать отображение больших список, показывая только видимые элементы. Снижается потребление памяти и повышается производительность интерфейса. Процесс оптимизации представлен в листинге 15.

Листинг 15 – Оптимизация компонентов

// Применение LazyColumn для отображения перекрестков

if (intersections.*isNotEmpty*()) {  
 LazyColumn(  
 modifier = Modifier.*fillMaxSize*(),  
 contentPadding = *PaddingValues*(16.*dp*)  
 ) {  
 *items*(intersections) { intersection ->  
 IntersectionItem(  
 intersection = intersection,  
 onIntersectionClick = {  
 viewModel.onIntersectionClicked(intersection)  
 }  
 )  
 if (selectedIntersection == intersection) {  
 IntersectionDetails(  
 trafficLights = trafficLights,  
 events = events,  
 onClose = { viewModel.clearSelectedIntersection() }  
 )  
 }  
 }  
 }  
}

Заключение

Для прохождения практики в ООО «Кактус» предприятием было предоставлено все необходимое ПО и оборудование.

Цели производственной практики достигнуты:

* получен практический опыт по выполнению работ по ПМ.11 «Разработка, администрирование и защита баз данных» и развиты общие и профессиональные компетенции;
* получен практический опыт по выполнению работ по ПМ.01 «Разработка модулей программного обеспечения для компьютерных систем» и развиты общие и профессиональные компетенции.

Выполненные задачи производственной практики:

* осуществлен сбор, обработка и анализ информации для проектирования БД;
* спроектирована БД;
* разработаны объекты БД;
* реализована БД в СУБД Microsoft SQL Server 2020;
* решены вопросы администрирования БД;
* реализованы технологии защиты информации в БД;
* сформирован алгоритм разработки;
* разработаны программные модули;
* выполнена отладка программных модулей с использованием программных средств Visual Studio;
* выполнено тестирование программных модулей;
* осуществлены рефакторинг и оптимизация программного кода;
* разработано мобильное приложение.

Список использованных источников

1. Волк В. К. Базы данных. Проектирование, программирование, управление и администрирование – 3-е изд. / В.К. Волк. – Санкт-Петербург : Лань, 2020. – 244 с. – URL: https://k0d.cc/storage/books/Databases/Базы%20данных%20Проектирование,%20программирование,%20управление%20и%20администрирование%20(Волк%202020).pdf (дата обращения: 13.04.2025). – Режим доступа: свободный. – Текст: электронный.
2. Куликов С. С. Тестирование программного обеспечения – 3-е изд., перераб. и доп. / С.С. Куликов. – Минск : Четыре четверти, 2020. – 294 с. – URL: https://ftp.zhirov.kz/books/IT/Other/Тестирование%20программного%20обеспечения%20%28Святослав%20Куликов%29.pdf (дата обращения: 24.04.2025). – Режим доступа: свободный. – Текст: электронный.
3. Фленов М. Е. Библия C#. – 6-е изд., перераб. и доп. / М.Е. Фленов. – Санкт-Петербург : БХВ-Петербург, 2024. – 512 с. – URL: https://ibooks.ru/bookshelf/396461/reading (дата обращения: 24.04.2025). – Режим доступа: для зарегистрир. пользователей. – Текст: электронный.
4. Android Developers. Официальная документация по разработке мобильных приложений на Android [Сайт]. – URL: https://developer.android.com (дата обращения: 20.04.2025). – Режим доступа: свободный. – Текст: электронный.
5. SQL Server. Техническая документация по SQL Server [Сайт]. – URL: https://learn.microsoft.com/ru-ru/sql/sql-server/?view=sql-server-ver16 (дата обращения: 08.04.2025). – Режим доступа: свободный. – Текст: электронный.