# 4. Реализация и тестирование системы

4.1. Реализация компонентов системы

В реализации компонентов системы имеется сервис Data Management, который предоставляет функциональность для работы с информацией о дронах и камерах. Этот сервис состоит из следующих файлов:

uav/mod.rs – Основной файл, определяющий структуру Uav и методы для создания экземпляров дронов с заданными параметрами. Пример структуры Uav представлен в листинге 1.

Листинг 1 – Структура Uav

#[derive(Debug, Deserialize, Serialize)]

pub struct Uav {

id: u64, // uav id

pub name: String, // uav name

pub max\_payload\_mass: u64, // maximum payload in grams

pub flight\_duration: u64, // average flight duration in seconds

pub takeoff\_speed: f64, // average takeoff speed in meters per second

pub flight\_speed: f64, // average flight speed in meters per second

pub min\_altitude: f64, // minimum safe flight altitude in meters

pub max\_altitude: f64, // maximum safe flight altitude in meters

}

uav/uav\_handle.rs – Файл содержит функции для работы с дронами через Tauri, такие как создание, обновление и удаление дронов, а также получение списка всех дронов. Пример функции для создания нового дрона представлен в листинге 2.

Листинг 2 – функции для создания нового дрона

#[tauri::command]

pub fn new\_uav(uav: Uav) -> String {

    let conn = Connection::open("mydatabase.db").expect("Cant open base");

    println!("Received new UAV: {:?}", uav);

    match uav\_sql::insert(&uav, &conn) {

        Ok(\_) => "Ok».to\_string(),

        Err(e) => e.to\_string(),

    }

}

uav/uav\_sql.rs – Файл содержит функции для работы с базой данных, включая создание таблицы для хранения информации о дронах, а также вставка, обновление, удаление и получение списка всех дронов. Пример функции для создания таблицы в базе данных представлен в листинге 3.

Листинг 3 – функции для создания таблицы uav

pub fn create\_table(conn: &Connection) -> Result<usize> {

    conn.execute(

       "CREATE TABLE IF NOT EXISTS uav (

                uav\_id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,

                uav\_name TEXT NOT NULL,

                uav\_max\_payload\_mass INTEGER NOT NULL CHECK (uav\_max\_payload\_mass >= 0),

                uav\_flight\_duration INTEGER NOT NULL CHECK (uav\_flight\_duration >= 0),

                uav\_takeoff\_speed REAL NOT NULL CHECK (uav\_takeoff\_speed >= 0),

                uav\_flight\_speed REAL NOT NULL CHECK (uav\_flight\_speed >= 0),

                uav\_min\_altitude REAL DEFAULT 0 NOT NULL CHECK (uav\_min\_altitude >= 0),

                uav\_max\_altitude REAL DEFAULT 0 NOT NULL CHECK (uav\_max\_altitude >= 0)

                )",

        (), // empty list of parameters.

    )

}

camera/mod.rs – Основной модуль, который содержит структуру и методы для работы с камерой. Структура Camera содержит различные свойства камеры, включая ее идентификатор, имя, массу, угол обзора по оси X и разрешение. Здесь также определены метод для создания новых камер. Пример структуры Camera представлен в листинге 4.

Листинг 4 – структура Camera

#[derive(Debug, Deserialize, Serialize)]

pub struct Camera {

    id: u64,               // id

    pub name: String,      // name

    pub mass: u64,         // mass in grams

    pub fov\_x: f64,        // x-axis viewing angle in degrees

    pub resolution\_x: u16, // camera resolution x

    pub resolution\_y: u16, // camera resolution y

}

camera/camera\_handle.rs – Этот модуль предоставляет функции для обработки запросов к базе данных. Здесь определены функции для создания новой камеры, обновления существующей камеры, удаления камеры и получения всех камер из базы данных. Каждая из этих функций открывает соединение с базой данных и вызывает соответствующую функцию из модуля camera\_sql. Пример функции для обновления данных о камере представлен в листинге 5.

Листинг 5 – функция для обновления данных о камере

#[tauri::command]

pub fn update\_camera(camera: Camera) -> String {

let conn = Connection::open("mydatabase.db").expect("Cant open base");

println!("Received updated camera: {:?}", camera);

match camera\_sql::update(&camera, &conn) {

Ok(\_) =>"Ok».to\_string(),

Err(e) => e.to\_string(),

}

}

camera/camera\_sql.rs – Этот модуль содержит функции для взаимодействия с базой данных SQLite. Здесь определены функции для создания таблицы в базе данных, вставки новой камеры в базу данных, обновления существующей камеры, удаления камеры из базы данных и получения всех камер из базы данных. Все эти функции работают непосредственно с базой данных, выполняя соответствующие SQL-запросы. Пример функции обновления данных о камере представлен в листинге 6.

Листинг 6 – функция для обновления данных о камере в базе

pub fn update(camera: &Camera, conn: &Connection) -> Result<usize> {

    conn.execute(

        "

                UPDATE camera SET

                    camera\_name = ?1,

                    camera\_mass = ?2,

                    camera\_fov\_x = ?3,

                    camera\_resolution\_x = ?4,

                    camera\_resolution\_y = ?5

                WHERE camera\_id = ?6",

        (

            &camera.name,

            &camera.mass,

            &camera.fov\_x,

            &camera.resolution\_x,

            &camera.resolution\_y,

            &camera.id,

        ),

    )

}

Сервис Algorithms, описанный файлом algorithms.rs, представляет собой ключевую составляющую системы и выполняет алгоритмические функции. Он включает в себя следующие функции:

discretize\_area: Эта функция принимает вектор кортежей, представляющих собой координаты x и y полигона, и размеры фотографии (ширину и высоту). Функция начинает с инициализации минимальных и максимальных значений x и y. Далее, функция проходит по всей области с шагом равным ширине и высоте фотографии и проверяет находится ли какой-либо из углов прямоугольника внутри полигона, основываясь на лучевом методе, в случае успеха, добавляет точку в результирующий вектор. После того как все области были проверены, функция возвращает вектор кортежей, представляющих собой координаты центров прямоугольников, которые составляют полигон. Реализация представлена в листинге 1 во вложении А.

nearest\_neighbor: Эта функция принимает вектор точек и начальную точку, а затем использует алгоритм ближайшего соседа для определения оптимального пути через все точки. Функция инициализирует набор оставшихся точек и результатирующий вектор с начальной точкой. Функция затем входит в цикл, продолжающийся, пока все точки не будут обработаны. В каждой итерации цикла функция находит ближайшую точку к текущей точке, удаляет ее из оставшихся точек и добавляет ее в результатирующий вектор [12]. Реализация представлена в листинге 2 во вложении А.

euclidean\_distance: Это вспомогательная функция, которая принимает две точки и возвращает евклидово расстояние между ними. Расстояние вычисляется как квадратный корень из суммы квадратов разностей координат x и y. Реализация представлена в листинге 7.

Листинг 7 – функция расчета расстояния между двумя точками

// Helper function to calculate the Euclidean distance between two points

pub fn euclidean\_distance(a: &(f64, f64), b: &(f64, f64)) -> f64 {

let (x1, y1) = \*a;

let (x2, y2) = \*b;

((x2 - x1).powi(2) + (y2 - y1).powi(2)).sqrt()

}

brute\_force: Эта функция использует алгоритм полного перебора для поиска оптимального пути через набор точек. Функция начинает с создания отдельного потока для каждой точки, каждый из которых запускает функцию brute\_force\_helper. Эта вспомогательная функция рекурсивно исследует все возможные пути через оставшиеся точки, обновляя лучший путь при обнаружении более короткого пути. Реализация представлена в листинге 3 во вложении А.

calculate\_distance: Эта функция вычисляет общую длину пути, проходящего через заданный набор точек Реализация представлена в листинге 8.

#[tauri::command]

pub fn calculate\_distance(points: Vec<(f64, f64)>) -> f64 {

points

.iter()

.zip(points.iter().cycle().skip(1))

.map(|(a, b)| euclidean\_distance(a, b))

.sum()

}

4.2. Реализация интерфейса системы

Пользовательский интерфейс системы состоит из трех основных меню: центрального меню, левого меню и правого меню. Центральное меню отображает карту и маршрут дрона, а также содержит набор кнопок для управления.

Центральное меню предоставляет основные инструменты и функции для работы с картой, панель инструментов, отображение маршрута и выполнения других операций, связанных с управлением маршрутом дрона, оно представлено на рисунке 5.



Рисунок 5 – Центральное меню

Центральное меню состоит из элементов:

1. Карта и маршрут: В центре экрана отображается карта (загружаемая с OpenStreetMap [11]), на которой отображается маршрут дрона. Маршрут представлен в виде линии, соединяющей точки съемки, по которым дрон должен пролететь. Это визуальное представление планируемого маршрута.
2. Кнопки переключения меню: В верхней части интерфейса расположены кнопки, которые позволяют переключать отображение левого и правого меню. Пользователь может скрыть или показать эти меню по своему усмотрению для освобождения места на экране.
3. Панель инструментов: На панели инструментов расположены кнопки для выполнения различных действий:
   1. Отмена (Undo): Кнопка, позволяющая отменить последнее действие при построении полигона;
   2. Навигация (Navigation): Кнопка, позволяющая пользователю вернуться к режиму навигации после использования других инструментов. Это удобно, если пользователь не хочет случайно переместить точку полигона области съемки;
   3. Рисование (Draw): Кнопка, активирующая режим построения полигона. В этом режиме пользователь может указать область, которую дрон должен снять;
   4. Установка стартовой точки (Set Starting Point): Кнопка, позволяющая пользователю установить стартовую точку для маршрута дрона. После выбора этой опции пользователь может щелкнуть на карте, чтобы установить стартовую точку;
   5. Расчет (Calculate): Кнопка, которая выполняет валидацию введенных данных и отправляет их на обработку на сервере. Результаты расчета маршрута отображаются на карте.

Левое меню включает два блока: блок UAV (Беспилотный Летательный Аппарат) и блок Camera (Камера). Левое меню представлено на рисунке 6.



Рисунок 6 – Левое меню

Блок UAV включает в себя:

1. Выбор и отображение списка UAV: В верхней части блока расположен выпадающий список (select), который позволяет пользователю выбрать один из доступных UAV. Каждый элемент списка содержит имя UAV. При выборе UAV из списка его параметры отображаются в полях ниже.
2. Кнопка «Fetch»: Рядом с выпадающим списком находится кнопка «Fetch», которая позволяет пользователю обновить список UAV, загрузив его с базы.
3. Кнопка «UAV details»: Эта кнопка позволяет пользователю переключать отображение блока с параметрами UAV. При нажатии на кнопку блок с параметрами UAV может быть скрыт или показан на экране.
4. Внутри блока UAV параметры UAV отображаются в виде полей для ввода которые позволяют пользователю просматривать и редактировать параметры UAV. Некоторые из доступных параметров UAV включают:
   1. ID: Идентификатор UAV (только для чтения);
   2. Name: Имя UAV;
   3. Max Payload Mass: Максимальная масса груза, которую может нести UAV;
   4. Flight Duration: Длительность полета UAV;
   5. Takeoff speed: Скорость взлета UAV;
   6. Flight speed: Скорость полета UAV;
   7. Min altitude: Минимальная высота полета UAV;
   8. Max altitude: Максимальная высота полета UAV;
5. Под блоком с параметрами UAV располагается панель инструментов, которая содержит следующие кнопки:
   1. «Update»: Кнопка, которая позволяет пользователю обновить параметры выбранного UAV.
   2. «New»: Кнопка, которая позволяет пользователю создать новый UAV с указанными в полях параметрами.
   3. «Delete»: Кнопка, которая позволяет пользователю удалить выбранный UAV.
   4. «Undo»: Кнопка, которая позволяет пользователю отменить последнее изменение параметров UAV.

Блок Camera аналогичен блоку UAV и включает в себя:

1. выпадающий список (select), который позволяет пользователю выбрать одну из доступных камер. Каждый элемент списка содержит имя камеры. При выборе камеры из списка ее параметры отображаются в полях ниже.
2. Кнопка «Fetch»: Рядом с выпадающим списком находится кнопка «Fetch», которая позволяет пользователю обновить список камер, загрузив его с базы.
3. Кнопка «Camera details»: Эта кнопка позволяет пользователю переключать отображение блока с параметрами камеры. При нажатии на кнопку блок с параметрами камеры может быть скрыт или показан на экране.
4. Внутри блока Camera параметры камеры отображаются в виде полей для ввода (input) и меток (label), которые позволяют пользователю просматривать и редактировать параметры камеры. Некоторые из доступных параметров камеры включают:
   1. ID: Идентификатор камеры (только для чтения).
   2. Name: Имя камеры.
   3. Mass (grams): Масса камеры в граммах.
   4. X-axis FOV (degrees): Угол обзора камеры по оси X в градусах.
   5. Resolution X: Разрешение камеры по оси X.
   6. Resolution Y: Разрешение камеры по оси Y.
5. Под блоком с параметрами камеры располагается панель инструментов, которая содержит следующие кнопки:
   1. «Update»: Кнопка, которая позволяет пользователю обновить параметры выбранной камеры.
   2. «New»: Кнопка, которая позволяет пользователю создать новую камеру с указанными в полях параметрами.
   3. «Delete»: Кнопка, которая позволяет пользователю удалить выбранную камеру.
   4. «Undo»: Кнопка, которая позволяет пользователю отменить последнее изменение параметров камеры.

Правое меню состоит из трех блоков: «Altitude Menu» (меню высоты), «Algorithm Menu» (меню алгоритма) и «Mission Parameters» (параметры миссии). Правое меню представлено на рисунке 8.



Рисунок 7 – Правое меню

Далее рассмотрен каждый из блоков подробнее:

1. Блок «Altitude Menu», здесь можно выбрать режим ввода высоты. Есть два варианта: «Manual altitude input» (ручной ввод высоты) и «Calculate using sm/px» (расчет высоты на основе sm/px):
   1. Если выбран режим ручного ввода, можно указать процент перекрытия и ввести высоту вручную.
   2. Если выбран режим расчета на основе sm/px, нужно указать процент перекрытия и ввести значение sm/px. После ввода всех параметров можно нажать кнопку «Calculate Altitude» (рассчитать высоту) для получения результата.
2. Блок «Algorithm Menu», здесь можно выбрать алгоритм для расчета маршрута. Есть два варианта: «Nearest Neighbor» (ближайший сосед) и «Brute Force» (полный перебор).
3. Блок «Mission Parameters», здесь отображаются параметры миссии, такие как длина маршрута, продолжительность миссии и количество фотографий. Также есть кнопка «Export to GeoJSON» (экспорт в формате GeoJSON), которая позволяет экспортировать координаты миссии в этот формат для дальнейшего использования.

Дополнительные иллюстрации, демонстрирующие пользовательский интерфейс, представлены в Приложении Б.

4.3. Тестирование системы

В ходе разработки и последующего программного обеспечения были разработаны и проведены комплексные тесты для проверки функциональности и стабильности работы системы. Проведенные тесты относятся к виду функционального тестирования, а также тестирования пользовательского интерфейса.

Тестирование проводилось в соответствии с предложенной методологией, при которой каждый из тестовых сценариев включал в себя конкретные шаги для воспроизведения действий пользователя, а также определенный ожидаемый результат. В процессе тестирования внимание было уделено как отдельным функциям, так и взаимодействию между различными компонентами системы.

Процесс тестирования осуществлялся с использованием различных входных данных, включая граничные значения и невалидные данные, чтобы оценить устойчивость и надежность системы в различных условиях, а также чтобы выявить и устранить возможные недостатки. Протоколы тестирования приведены в таблице 12. Результат выполнения теста 5 «Проверка параметров миссии» представлен на рисунке 8. Результат выполнения теста 6 «Визуализация маршрута»представлен на рисунке 9.

Таблица 12 – Протоколы тестирования системы

| **№** | **Название теста** | **Шаги** | **Ожидаемый результат** | **Тест пройден?** |
| --- | --- | --- | --- | --- |
| 1 | Загрузка дрона | 1. В левом меню выбирать идентификатор дрона.  2. Нажать кнопку «Fetch». | В полях деталей дрона отображается информация о выбранном дроне. | да |
| 2 | Редактирование дрона | 1. Выбирать дрон.  2. Включить режим редактирования.  3. Изменить параметры дрона.  4. Нажать кнопку «Update». | Информация о дроне в базе данных обновляется. | да |
| 3 | Расчет маршрута | 1. Ввети необходимые параметры, высоту полета, алгоритм, обозначить точку старта, обозначить зону съемки, выбрать дрон, выбрать камеру.  2. Нажать кнопку. «Calculate» | На карте отображается рассчитанный маршрут. | да |
| 4 | Некорректная попытка расчета маршрута | 1. Не ввести необходимые параметры, высоту полета, алгоритм, точку старта, зону съемки, дрон, камеру.  2. Нажать кнопку «Calculate». | Отображается ошибка о том, что не все параметры выбраны | да |
| 5 | Проверка параметров миссии | 1. Просмотрите отображаемые параметры миссии в правом меню. | Параметры миссии (длина маршрута, продолжительность миссии, количество фотографий) отображаются корректно. | да |
| 6 | Визуализация маршрута | 1. После расчета маршрута проверьте визуализацию маршрута на карте. | Маршрут корректно отображается на карте. | да |
| 7 | Экспорт в GeoJSON | 1. После расчета маршрута нажать кнопку «Export to GeoJSON». 2. Выбрать путь сохранения | Файл GeoJSON с информацией о маршруте сохраняется. | да |
| 8 | Загрузка камеры | 1. В левом меню выберать идентификатор камеры.  2. Нажать кнопку «Fetch». | В полях деталей камеры отображается информация о выбранной камере. | да |

Окончание таблицы 12

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Название теста** | **Шаги** | **Ожидаемый результат** | **Тест пройден?** |
| 9 | Редактирование камеры | 1. Выбрать камеру.  2. Включить режим редактирования.  3. Изменить параметры камеры.  4. Нажать кнопку «Update». | Информация о камере в базе данных обновляется. | да |
| 10 | Выбор алгоритма | 1. В правом меню выберите желаемый алгоритм. | Выбранный алгоритм устанавливается как текущий. | да |
| 11 | Ручной ввод высоты | 1. В правом меню выберите режим ручного ввода высоты.  2. Введите желаемую высоту. | Введенная высота устанавливается как текущая. | да |
| 12 | Расчет высоты | 1. В правом меню выберите режим расчета высоты.  2. Введите значение sm/px.  3. Нажмите кнопку «Calculate Altitude». | Высота автоматически рассчитывается и устанавливается как текущая. | да |
| 13 | Изменение параметра перекрытия | 1. В правом меню ввести значение в поле «Overlap (%)». | Значение перекрытия изменяется в соответствии с введенным значением. | да |
| 14 | Проверка работы с отсутствующими данными | 1. Нажать кнопку «Calculate», когда база данных пустая. | Система сообщает о том, что дрон не выбран. | да |

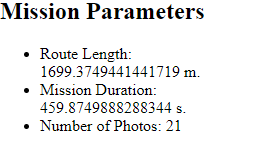


Рисунок 8 – Результат выполнения теста 5 «Проверка параметров миссии»



Рисунок 9 – Результат выполнения теста 6 «Визуализация маршрута»