



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.04.01 Информатика и вычислительная техника

МАГИСТЕРСКАЯ ПРОГРАММА 09.04.01/05 Современные интеллектуальные  
программно-аппаратные комплексы.

## РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ

*НА ТЕМУ:*

Разработка клиент-приложения для NoSQL базы  
данных «Доставка»

Студент ИУ6-21М  
(Группа)

Д.Д.  
(Подпись, дата)

Джабри А.Ш.  
(И.О.Фамилия)

Руководитель курсового проекта

Фомин М.М.  
(Подпись, дата)

Фомин М.М.  
(И.О.Фамилия)

Фомин М.М.  
Фомин М.М.

2024 г.

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ  
Заведующий кафедрой ИУ6  
(Индекс)  
А.В.Прокетарский  
(И.О.Фамилия)  
« \_\_\_\_ » \_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**  
**на выполнение курсового проекта**

по дисциплине «Распределенные базы данных»

Студент группы ИУ6-21М

Джабри Абделькадер Waker  
(Фамилия, имя, отчество)

Тема курсового проекта

«Разработка клиент-приложения для NoSQL базы данных «Доставка»

Направленность КП (учебный, исследовательский, практический, производственный, др.)  
исследовательский

Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения проекта: 25% к 2 нед., 50% к 10 нед., 75% к 13 нед., 100% к 16 нед.

**Задание**

Разработать серверную часть базы данных на основе системы управления базами данных MongoDB и реализовать возможность доступа для создания, модификации и удаления данных в базе данных через API Realm.

**Оформление курсового проекта:**

Расчетно-пояснительная записка на 24 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.):

1. Название темы КП, задание.
2. Результаты разработки базы данных.
3. Примеры кода генерации и настройки базы данных.

Дата выдачи задания «10» февраля 2024 г.

Руководитель курсового проекта

Студент

  
(Подпись, дата)

Фомин М.М.  
(И.О.Фамилия)

  
(Подпись, дата)

Джабри А.Ш.  
(И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

## **РЕФЕРАТ**

РПЗ 27 страниц, 16 рисунков, 3 источника, 1 приложение

**БАЗА ДАННЫХ, MONGODB, REALM, РАСПРЕДЕЛЕННЫЕ БАЗЫ  
ДАННЫХ**

Целью курсового проекта является закрепление и углубление знаний, приобретенных в процессе обучения по курсу “Распределенные Базы данных”, а также получение практических навыков разработки и модификации реальных баз данных и информационных систем.

## СОДЕРЖАНИЕ

|  |    |
|--|----|
| ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ  | 4  |
| Введение   | 5  |
| 1. Цель работы   | 6  |
| 2. Обзор приложения  | 7  |
| 2.1 Содействие одноранговой доставке посылок через связь с путешественниками | 7  |
| 2.2 План работы  | 7  |
| 3. Проектирование  | 9  |
| 3.1 Бизнес-процесс   | 9  |
| 3.2 Диаграмма классов  | 12 |
| 4. База данных   | 13 |
| 4.1 Атлас MongoDB  | 13 |
| 4.2 Настройка кластера в MongoDB Atlas                                       | 13 |
| 4.3 Доступ к сети  | 15 |
| 4.5 Создание пользователя кластера   | 15 |
| 4.6 Создание строки подключения к базе данных                                | 16 |
| 4.7 Подключение к кластеру из VSCode   | 17 |
| 4.8 Создание базы данных   | 19 |
| Вывод  | 21 |
| СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ   | 22 |
| Приложение А   | 23 |

## **ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

БД – база данных

MongoDB – Документно-ориентированная система управления базами данных, не требующая описания схемы таблиц. Считается одним из классических примеров NoSQL-систем, использует JSON-подобные документы и схему базы данных

Realm – это система управления объектными базами данных с открытым исходным кодом, изначально предназначенная для мобильных операционных систем (Android/iOS), но также доступная для таких платформ

СУБД – система управления базой данных

SQL (Structured Query Language) – структурированный язык запросов

Android – Операционная система для смартфонов, планшетов, электронных книг, цифровых проигрывателей, наручных часов, фитнес-браслетов, игровых приставок, ноутбуков, нетбуков, смарт буков, очков Google Glass, телевизоров, проекторов и других устройств

## **Введение**

Как иностранные студенты, осваивающие жизнь в России, мы часто ощущаем тоску по привычным уютам дома и жаждем поделиться богатством русской культуры с нашими семьями и друзьями за границей. Однако расстояние, разделяющее нас с близкими, делает сложным преодоление разрыва между нашими двумя мирами. Кроме того, с недавними нарушениями в международных службах доставки из-за геополитических событий и санкций, задача отправки и получения посылок стала все более трудной.

В свете этих проблем я и мой русский друг Марчук И.С. взяли за миссию создания решения, которое не только облегчает обмен необходимыми вещами, но также способствует глубокому контакту между иностранными студентами в России и их семьями дома. Вдохновленные нашими общими переживаниями и желанием поделиться красотой русской культуры с нашими близкими, мы представили инновационное приложение, которое служит связующим звеном для культурного обмена и взаимной поддержки.

Это приложение воплощает суть нашего видения, предоставляя платформу, где пользователи могут без проблем отправлять и получать посылки, а также делиться яркой палитрой русских традиций и ценностей с их семьями.

В этом отчете мы рассмотрим каждый аспект, касающийся концепции этого приложения, стремясь обеспечить всестороннее понимание его разработки и назначения.

## **1. Цель работы**

Целью курсового проекта является закрепление и углубление знаний, приобретенных в процессе обучения по курсу “Распределенные Базы данных”, а также получение практических навыков разработки и модификации реальных баз данных и информационных систем.

## **2. Обзор решения**

### **2.1 Содействие одноранговой доставке посылок через связь с путешественниками**

Наше решение упрощает два основных типа операций:

#### **Пост путешественника:**

Когда пользователь планирует отправиться в определенный пункт назначения, скажем, в город X, он может создать пост с указанием своих планов поездки.

Это пост служит объявлением для других пользователей, которым могут понадобиться товары, доставленные в город X.

Если есть посты от пользователей, которым требуется доставка посылок в город X, путешественник может связаться с ними, чтобы договориться о доставке посылки во время поездки.

#### **Запрос на доставку посылки:**

Если пользователю необходимо отправить посылку в город X, он может выполнить поиск по существующим постам от путешественников, планирующих отправиться туда.

В качестве альтернативы пользователь может создать новый пост, указав, что ему нужна доставка посылки в город X.

Путешественники, которые увидят эти посты, могут предложить доставить посылку во время поездки.

По сути, приложение выступает в качестве платформы для связи путешественников с людьми, которым требуется доставка товаров в определенные пункты назначения, что способствует созданию одноранговой сети доставки.

## **2.2 План работы**

Учитывая масштабность нашего приложения, мы решили сотрудничать с моим одноклассником *И.С.Марчуком*, чтобы оптимизировать наши усилия в рамках проекта. Вместе мы стремимся реализовать как можно больше основных функций в начальной версии, сохранив ее простой, но функциональной. По мере



продвижения мы планируем расширять и совершенствовать приложение на основе ваших ценных отзывов и идей. Этот отчет, посвященный концептуализации и разработке серверной части, закладывает основу для дальнейшего развития. Более подробная информация о нашем подходе к внедрению будет рассмотрена в следующей главе.

### 3. Проектирование

После изучения контекста нашего проекта и определения поставленных целей мы приступили к анализу и проектированию нашей системы. Фазы анализа и проектирования в информационном проекте являются неотъемлемыми этапами, позволяющими прийти к практичному, согласованному и полному решению, соответствующему потребностям пользователей.

#### 3.1 Бизнес-процесс

В этом разделе мы рассмотрим тонкости наших бизнес-процессов, чтобы охватить поток действий, взаимодействий и данных в нашей системе. Визуализируя эти процессы, мы можем определить области для оптимизации, упростить рабочие процессы и обеспечить эффективность нашего решения.

Благодаря тщательному анализу и продуманному дизайну мы заложим основу для разработки нашего приложения, гарантируя его полное соответствие требованиям и ожиданиям заинтересованных сторон. Теперь перейдем к изучению бизнес-процессов с помощью диаграмм.

Начнем со схемы бизнес-процесса регистрации:

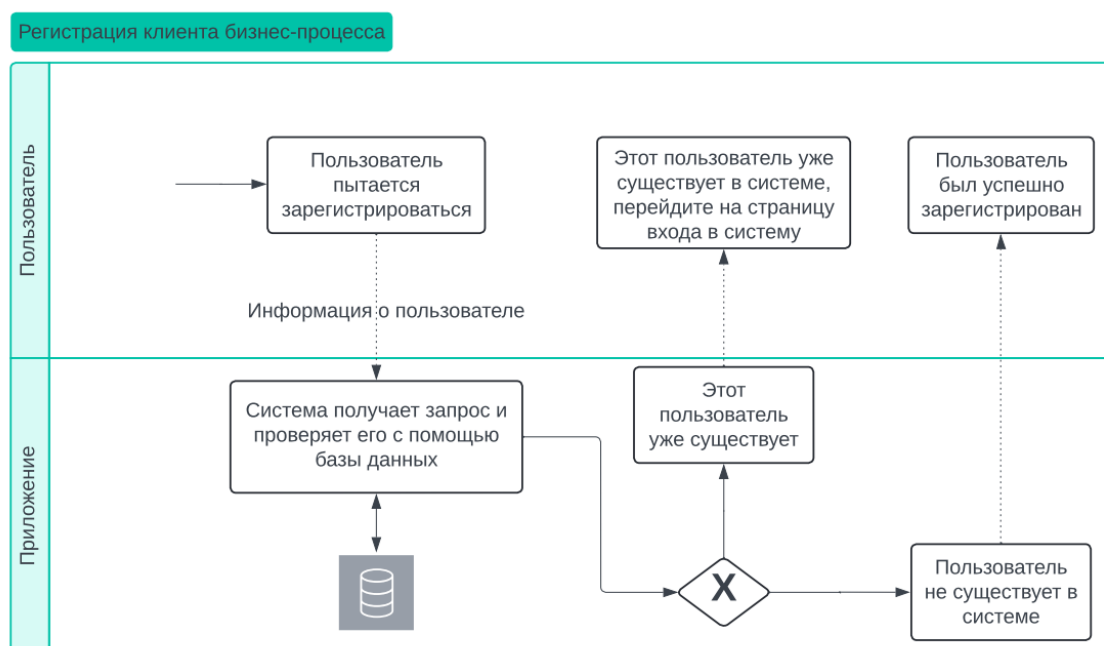


Рисунок 1 – Схема бизнес-процесса «Регистрация клиента».

Вторая схема бизнес-процесса — это схема входа в систему, где каждый пользователь должен иметь учетную запись и входить в нее на случай, если он захочет что-то опубликовать.

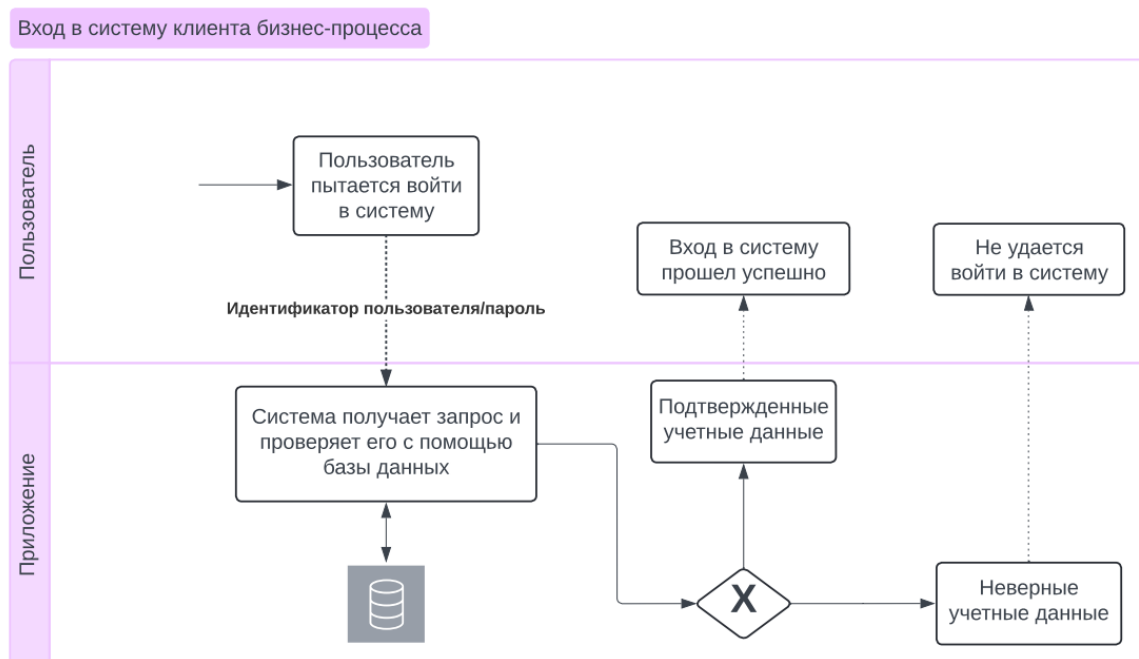


Рисунок 2 – Схема бизнес-процесса «Войдите в систему».

Третья схема бизнес-процесса выглядит следующим образом: когда пользователь (путешественник) собирается куда-то в случае командировки и у него есть место чтобы взять с собой посылку, или когда пользователю нужно что-то куда-то отправить и он ищет кого-то, кто туда направляется.



Рисунок 3 – Схема бизнес-процесса «новый пост».

Четвертая схема бизнес-процесса — это когда пользователь ищет конкретную запись, и в основном это происходит путем фильтрации адресов прибытия и отбытия

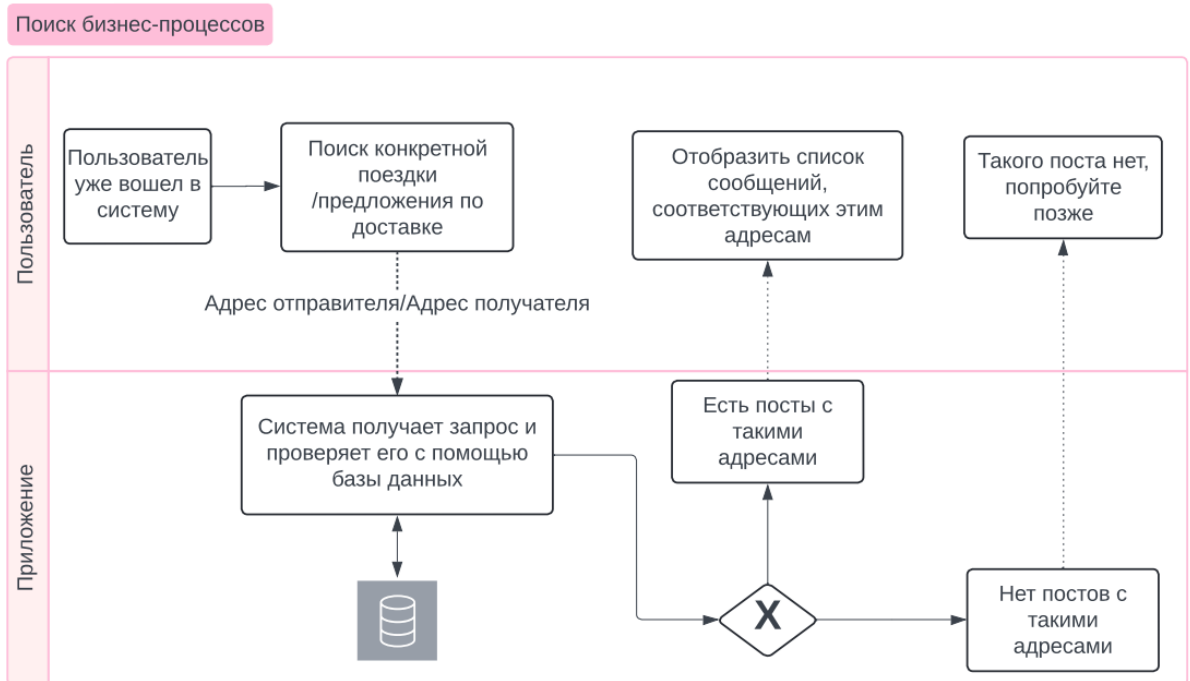


Рисунок 4 – Схема бизнес-процесса «Поиск».

Пятый и последний бизнес-процесс — это редактирование существующей записи.



Рисунок 5 – Схема бизнес-процесса «Редактировать пост».

### 3.2 Диаграмма классов

Диаграмма классов — это схема, используемая для выражения статической структуры системы в терминах классов и отношений между этими классами, класс характеризуется:

- Названием класса;
- Атрибутом;
- Методом.

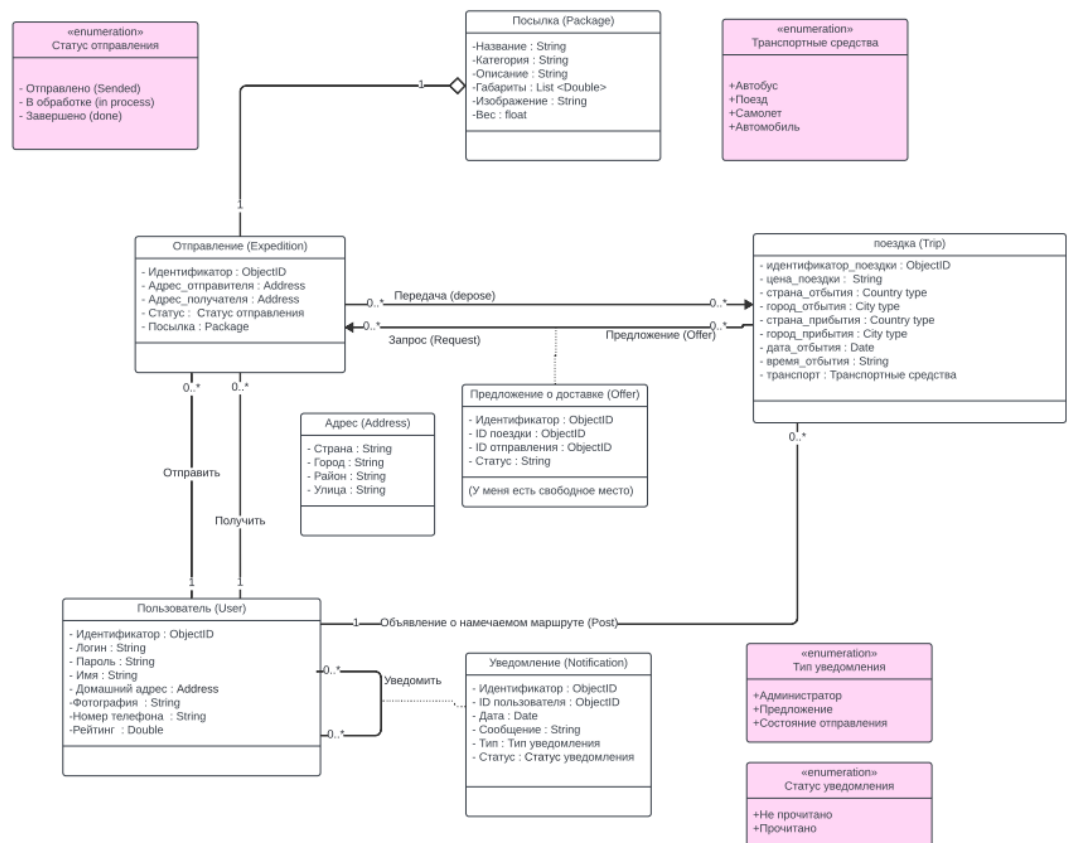


Рисунок 6 – Диаграмма классов приложения.

## **4. База данных**

### **4.1 Atlas и MongoDB**

База данных как услуга (DBaaS) — это сервис, который позволяет настраивать, развертывать и масштабировать базу данных, не беспокоясь о локальном физическом оборудовании, обновлениях программного обеспечения и деталях настройки производительности. С DBaaS облачный провайдер сделает все это за вас — и сразу же запустит всё в работу.

MongoDB Atlas это полностью управляемая облачная субд, которая решает все сложные задачи по развертыванию, управлению и исправлению ваших развертываний у поставщика облачных услуг по вашему выбору (AWS, Azure и GCP). MongoDB Atlas — это лучший способ развертывания, запуска и масштабирования MongoDB в облаке. С помощью Atlas вы сможете запустить базу данных MongoDB всего за несколько кликов и всего за несколько минут.

### **4.2 Настройка кластера в MongoDB Atlas**

Чтобы выполнить этот шаг, нам нужно создать учетную запись MongoDB Atlas, и как только у нас будет учетная запись Atlas и мы создадим организацию и проект, мы сможем создать кластер баз данных.

Нам нужно убедиться, что в выпадающих списках навигации вверху выбраны нужные организация и проект. Затем выберите «Кластеры» в левом навигационном меню и нажмите на кнопку «Создать кластер», как показано на рисунке 7.

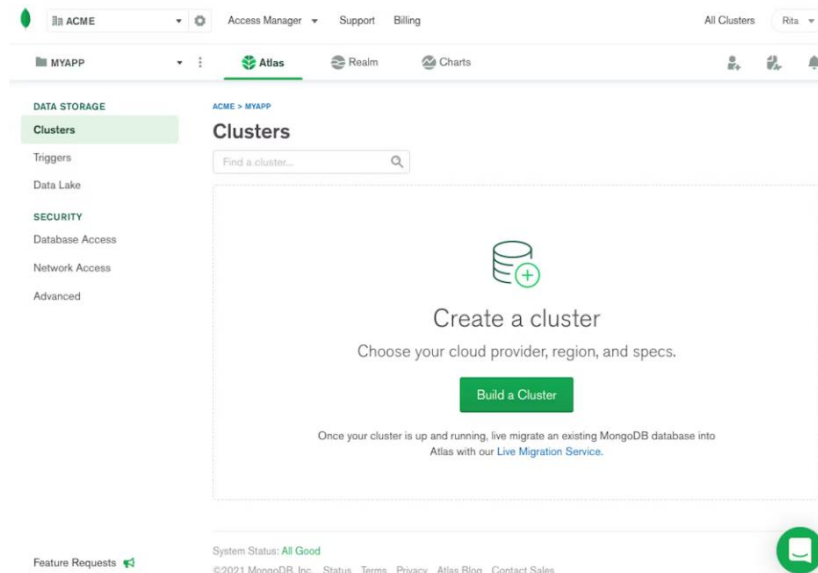


Рисунок 7 – Создание кластера в MongoDB Atlas.

Нам был предложен выбор между Общим кластером, выделенным кластером и Мультиоблачным и мультирегиональным кластером.

Я выбрал общий кластер, потому что это был бесплатный вариант. После того как я выбрал тип кластера, я смог выбрать одного из трех ведущих облачных провайдеров (Amazon Web Services, Microsoft Azure и Google Cloud Platform) и выбрать регион для размещения кластера.

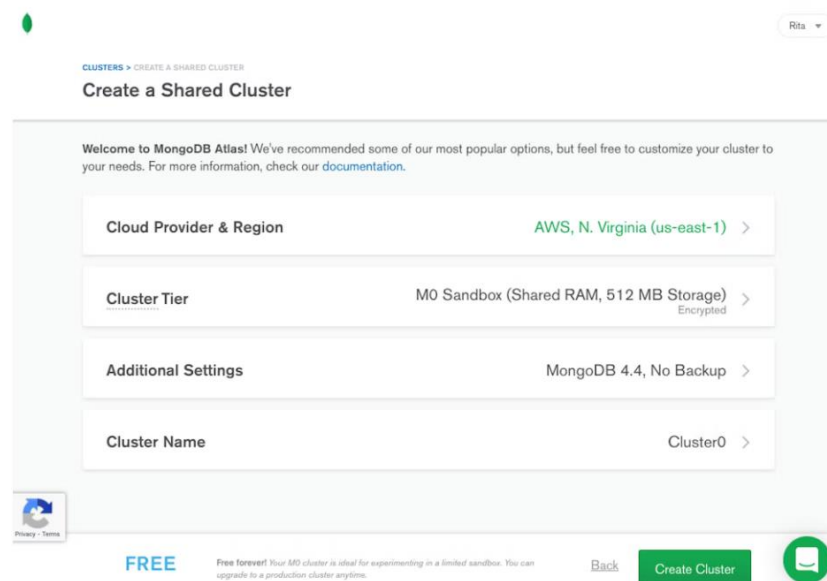


Рисунок 8 – Выбрали облачного провайдера для кластера.

## 4.3 Доступ к сети

По соображениям безопасности в новых кластерах баз данных по умолчанию не включен сетевой доступ. Нам необходимо включить сетевой доступ в явном виде, указав адреса, которые будут подключаться к кластеру.

Каждая запись может быть IP-адресом, подсетью или вы можете включить доступ из любого места. Как правило, вы предоставляете доступ только к списку подсетей или IP-адресов, а не к какому-либо местоположению. Это ограничивает количество подключений, которые принимает ваш кластер, что повышает его безопасность.

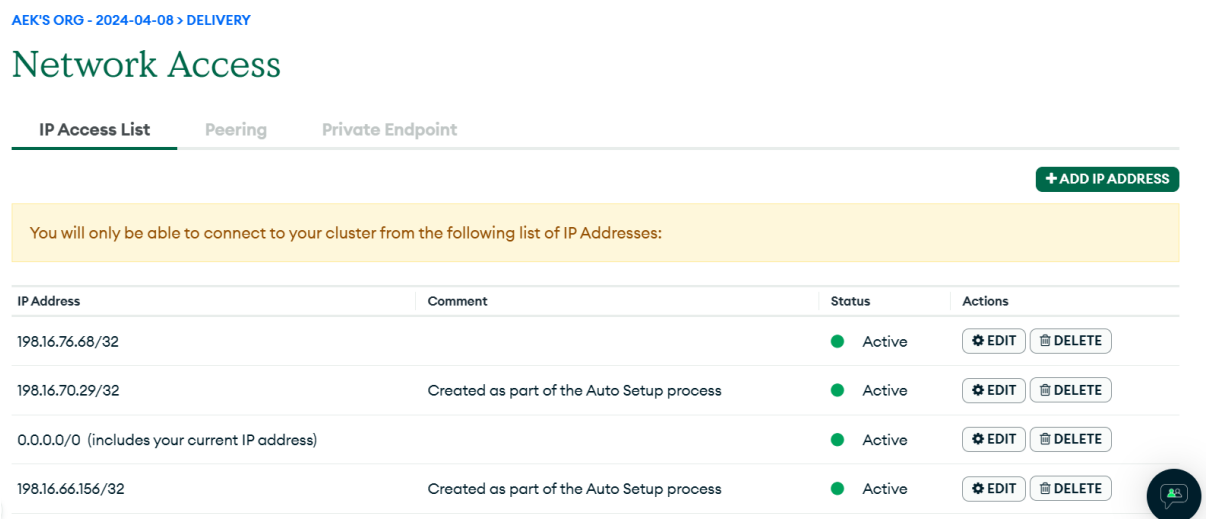


Рисунок 9 – Сетевой доступ к кластеру.

В нашем случае (рис. 9), поскольку я работаю с *И.С. Марчуком*, и его публичный ip-адрес не статичен, у меня есть доступ к 0.0.0.0/0, что не очень хорошо с точки зрения безопасности, но это только на этапе разработки, позже он будет изменен и только ip-адрес администратора сможет получить доступ к кластеру.

## 4.5 Создание пользователя кластера

Чтобы подключиться к базе данных из скрипта или приложения, мы должны сначала создать пользователя базы данных MongoDB. Пользователь базы данных позволяет нам подключаться к базам данных и использовать их. мы



должны обратить внимание на то, что это не зависит от пользователя, который входит в систему и управляет кластерами и ресурсами в Atlas.

Пользователи базы данных создаются для каждого проекта и имеют доступ ко всем кластерам в проекте. мы также можем назначать различные роли и привилегии пользователям базы данных. ниже мы можем увидеть пользователей, которых мы добавили в нашу базу данных.

AEK'S ORG - 2024-04-08 > DELIVERY

### Database Access

Database Users Custom Roles

+ ADD NEW DATABASE USER







| User Name ↕   | Authentication Method ▲ | MongoDB Roles              | Resources     | Actions   |
|---|-------------------------|----------------------------|---------------|---|
|  Delivery  | SCRAM                   | readWriteAnyDatabase@admin | All Resources |  EDIT  DELETE |
|  Ivan12345 | SCRAM                   | readWriteAnyDatabase@admin | All Resources |  EDIT  DELETE |

Рисунок 10 – Пользователи базы данных.

Первый пользователь был использован для создания базы данных, что мы увидим, в следующих разделах, а второй был создан Иваном, и этот пользователь будет использоваться для приложения.

#### 4.6 Создание строки подключения к базе данных

В зависимости от нашего приложения, нам необходимо установить драйвер (библиотеку), соответствующий нашей платформе, чтобы подключиться к кластеру в Atlas. Затем нам нужно сгенерировать строку подключения к базе данных для нашего кластера. Как только мы включим доступ к сети и создадим пользователя базы данных, мы можем нажать на кнопку "Выбрать способ подключения", которая позволит нам сгенерировать строку подключения для нашего приложения. Как показано на рисунке 11.

Connect to Delivery

Set up connection security Choose a connection method **3** Connect

Connecting with MongoDB Compass

I don't have MongoDB Compass installed I have MongoDB Compass installed

1. Select your operating system and download MongoDB Compass

Windows 64-bit (10+)

Download Compass (1.43.0) or Copy download URL

Compass is an interactive tool for querying, optimizing, and analyzing your MongoDB data.

2. Copy the connection string, then open MongoDB Compass

mongodb+srv://<username>:<password>@delivery.ak3w9xu.mongodb.net/

Replace **<password>** with the password for the **<username>** user. Ensure any options are [URL encoded](#).

Рисунок 11 – Генерирование строки подключения.

#### 4.7 Подключение к кластеру из среды VSCode

Сначала нам нужно установить расширение MongoDB в нашу IDE, после чего мы выберем опцию подключиться с помощью строки подключения и там введем эту строку:

*mongodb+srv://Delivery:\*\*\*\*\*@delivery.ak3w9xu.mongodb.net/*

На месте звездочек будет указан пароль.

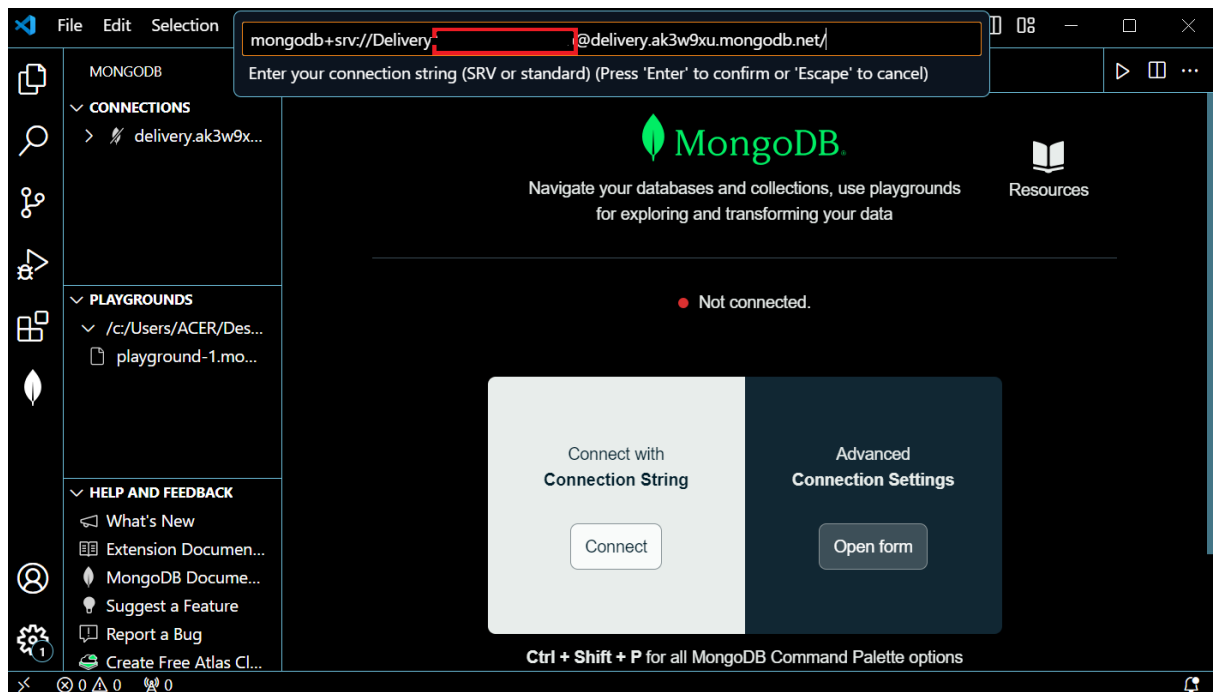


Рисунок 12 – Подключение к кластеру.

После того, как мы нажмем enter, мы подключимся к кластеру и увидим следующее.

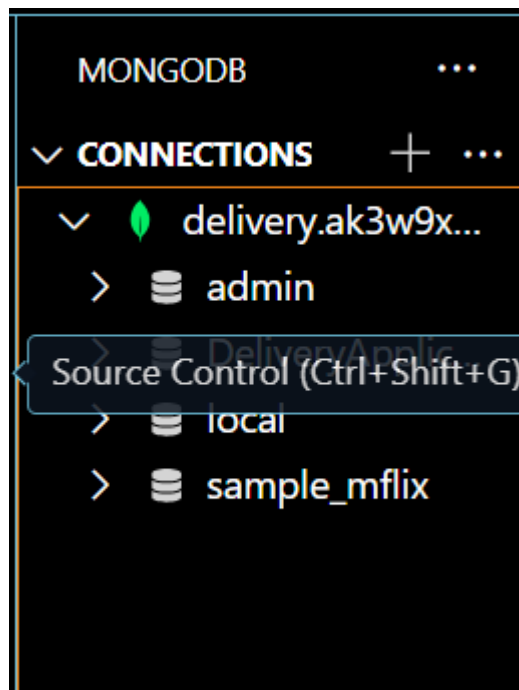


Рисунок 13 – Успешное подключение к кластеру.

## 4.8 Создание базы данных

С помощью скрипта, который будет показан далее, мы создали нашу базу данных "DeliveryApplicationDB" с необходимыми коллекциями.

```
// Select the database to use.
use('DeliveryApplicationDB');

// Creation of documents into the Expedition collection.
db.getCollection('Expedition').insertMany([
  {
    'Address sender': {
      "country": "Russia",
      "city": "Moscow",
      "district": "Central",
      "street": "Tverskaya"
    },
    'Адрес reciver': {
      "country": "Algeria",
      "city": "Lagouat",
      "district": "Central",
      "street": "Kaser El bzaim"
    },
    'Status': 'Sended',
    'Pckage': { 'Name': 'Box',
    'Categorie': 'Electronics',
    'Description': 'The box is sleek and sturdy, designed to ensure the safe delivery
of the laptop inside. It s made of durable cardboard with reinforced edges and corners,
providing extra protection during transit. The exterior is plain brown, adorned only wit
h a shipping label and fragile stickers for added caution. Inside, the laptop rests snug
ly within custom-fit foam padding, keeping it secure and cushioned against any bumps or
jostles during its journey', 'Габариты': [35.56, 25.3, 3.2],
    'Dimension': {
      "1": "35.56",
      "2": "25.3",
      "3": "3.2"
    },
    'Quantity': 2,
    'Wieght': 2,
    'Pictures': 'URL'
  },
  }
]);
```

Рисунок 14 – Часть скрипта, который мы использовали для создания коллекции «Отправления».

Поскольку мы подключены к кластеру MongoDB, мы можем видеть результат работы нашего скрипта из vs code и просматривать коллекции.

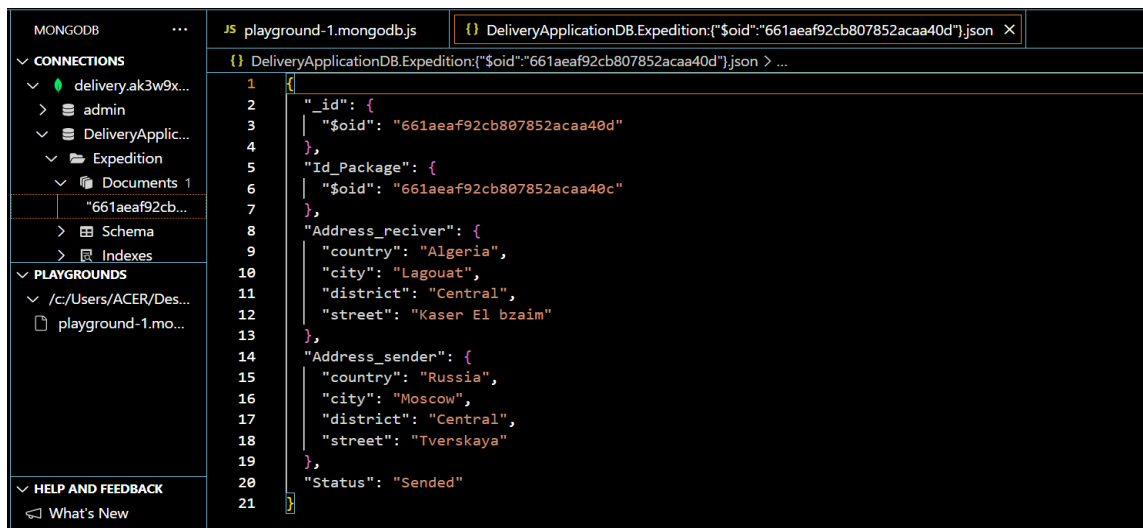


Рисунок 15 – Смотрите созданную коллекцию в vs-code.

И мы можем видеть созданную коллекцию из интерфейса MongoDB atlas.

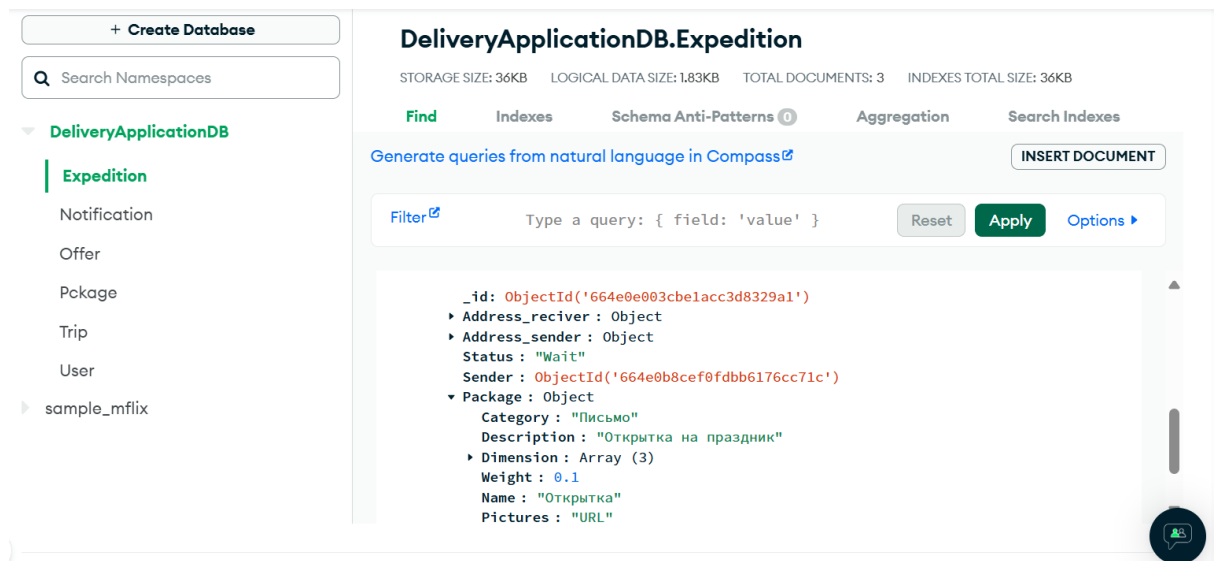


Рисунок 16 – Смотрите созданную коллекцию в MongoDB-atlas.

MongoDB Atlas — это идеальное решение для разработчиков, у которых нет времени или ресурсов для управления всей инфраструктурой, необходимой для создания кластера MongoDB. С помощью Atlas мы можем запустить полнофункциональный кластер всего за несколько минут, что позволит вам сосредоточиться на нашем приложении, а не беспокоиться о настройке СУБД. Кроме того, бесплатный доступ позволяет легко и быстро приступить к работе и изучить платформу. Это кардинально меняет ситуацию для разработчиков, которые ищут простой способ управления базами данных через MongoDB.

## **Вывод**

В заключение хочется сказать, что наш отчет о концептуальной части нашего проекта по созданию мобильного приложения заложил основу для нашего дальнейшего развития. Мы создали четкую схему классов, чтобы понять, как будет работать наше приложение, и схему бизнес-процессов, чтобы наглядно представить их выполнение.

Кроме того, мы настроили MongoDB Atlas для работы с нашей базой данных, что упрощает ее развертывание и масштабирование, не беспокоясь об инфраструктуре. С помощью MongoDB Atlas мы разработали базу данных, которая соответствует потребностям нашего приложения и обеспечивает бесперебойное управление данными.

Эти шаги закладывают прочную основу для создания нашего мобильного приложения. Разработав план и выполнив техническую настройку, мы готовы с уверенностью перейти к следующему этапу разработки.

## **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1 MongoDB Documentation [Электронный ресурс] – Режим доступа: <https://www.mongodb.com/docs/> (дата обращения: 26.03.2024)

2 NoSQL базы данных: понимаем суть [Электронный ресурс] – Режим доступа: <https://habr.com/ru/articles/152477/> (дата обращения: 26.03.2024)

3 MongoDB [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/MongoDB> (дата обращения: 26.03.2024)

**Приложение А**  
Создание базы данных  
Листов 3



## Полный код для создания начальных экземпляров коллекций в базе данных

```
/* global use, db */
// MongoDB Playground
// To disable this template go to Settings | MongoDB | Use Default Template
For Playground.
// Make sure you are connected to enable completions and to be able to run a
playground.
// Use Ctrl+Space inside a snippet or a string literal to trigger
completions.
// The result of the last command run in a playground is shown on the
results panel.
// By default the first 20 documents will be returned with a cursor.
// Use 'console.log()' to print to the debug output.
// For more documentation on playgrounds please refer to
// https://www.mongodb.com/docs/mongodb-vscode/playgrounds/

// Select the database to use.
use('DeliveryApplicationDB');

// Creation of documents into the Expedition collection.
db.getCollection('Expedition').insertMany([
  {
    'Address sender': {
      "country": "Russia",
      "city": "Moscow",
      "district": "Central",
      "street": "Tverskaya"
    },
    'Адрес reciver': {
      "country": "Algeria",
      "city": "Lagouat",
      "district": "Central",
      "street": "Kaser El bzaim"
    },
    'Status': 'Sended',
    'Pckage': { 'Name': 'Box',
    'Ctegorie': 'Electronics',
    'Description': 'The box is sleek and sturdy, designed to ensure the
safe delivery of the laptop inside. It s made of durable cardboard with
reinforced edges and corners, providing extra protection during transit. The
exterior is plain brown, adorned only with a shipping label and fragile
```

```

stickers for added caution. Inside, the laptop rests snugly within custom-
fit foam padding, keeping it secure and cushioned against any bumps or
jostles during its journey', 'Габариты': [35.56, 25.3, 3.2],
    'Dimension': {
        "1": "35.56",
        "2": "25.3",
        "3": "3.2"
    },
    'Quantity': 2,
    'Wiegth': 2,
    'Pictures': 'URL'
},
}
});

// Creation of documents into the Trajectory collection.
db.getCollection('Trip').insertMany([
    {
        'Send country': 'Russia',
        'Send city': 'Moscow',
        'Reciving country': 'Algeria',
        'Reciving city': 'Laghouat',
        'Sent date': new Date('2024-03-01T09:00:00Z'),
        'Transport mean': 'Самолет',
        'Price': '2000 rouble'},
]);

// Creation of documents into the Offer collection.
db.getCollection('Offer').insertMany([
    { 'Id_trip': '',
        'Id_Expidition': '',
        'Status': 'Accapted'},
]);

// Creation of documents into the User collection.
db.getCollection('User').insertMany([
    {
        'Name': 'Abdelkader',
        'Email': 'kd_djb@gmail.com',
        'Password': '',
        'Address': {
            "country": "Russia",
            "city": "Moscow",
            "district": "Central",
            "street": "Tverskaya"
        },
    },
]);

```

```

        'Picture': 'URL',
        'Phone number': '+79993243245',
        'reting': 4.5},
    {
        'Name': 'Ivan',
        'Email': 'Ivan@gmail.com',
        'Password': '',
        'Address': {
            "country": "Russia",
            "city": "Moscow",
            "district": "Central",
            "street": "Izmailava"
        },
        'Picture': 'URL',
        'Phone number': '+79998503245',
        'Reting': 5},
    ]);

    // Creation of documents into the Offer collection.
    db.getCollection('Notification').insertMany([
    {
        'Id_User': '',
        'Message': 'Welcome to our application',
        'Type': 'Admin',
        'Date': new Date('2024-02-01T09:00:00Z'),
        'Status': 'BeenRead'},
    ]);

```