*В данном документе приведены отрывки из работ, выполненных в рамках студенческой программы. Полные отчеты находятся по ссылке:* *https://github.com/realtuatara/portfolio*

**Решение задач интеллектуального анализа данных (ИАД): кластеризация объектов средствами интегрированной системы Statistica, языка R**

## 1. Цели

- изучить алгоритмы и методы кластерного анализа данных на примере решения конкретной задачи ИАД;
- исследовать эффективность использования различных алгоритмов и методов кластерного анализа данных для решения прикладной задачи;
- ознакомиться и получить практические навыки работы с модулями интегрированной статистической системы Statistica, языка R, реализующими решение задачи кластеризации объектов.

## 2. Постановка задачи ИАД

Изучаются показатели работы программистов крупной организации. Необходимо выделить группы (классы) программистов в соответствии с рядом устойчивых признаков. Разделение программистов на группы проводится с целью установления и обоснования тарифных разрядов и размера заработной платы для каждой выделенной группы. Также требуется построить правило отнесения программиста к одной из выделенных групп (классов).

Исходные данные для проведения статистического анализа представлены в табл. 1. Рассматриваются следующие показатели (признаки) для каждого программиста:

- возраст;
- время написания первой тестовой программы, в час.;
- время написания второй тестовой программы, в час.;
- стаж работы по специальности в данной организации;
- образование (непрофильное – 0; профильное - 1);

наличие сертификатов о повышении квалификации в области программирования и программного обеспечения (0 –сертификаты отсутствуют; 1- сертификаты 1-го уровня; 2 - сертификаты 2-го уровня, 3- сертификаты 3-го уровня).

Анализ полученных результатов:

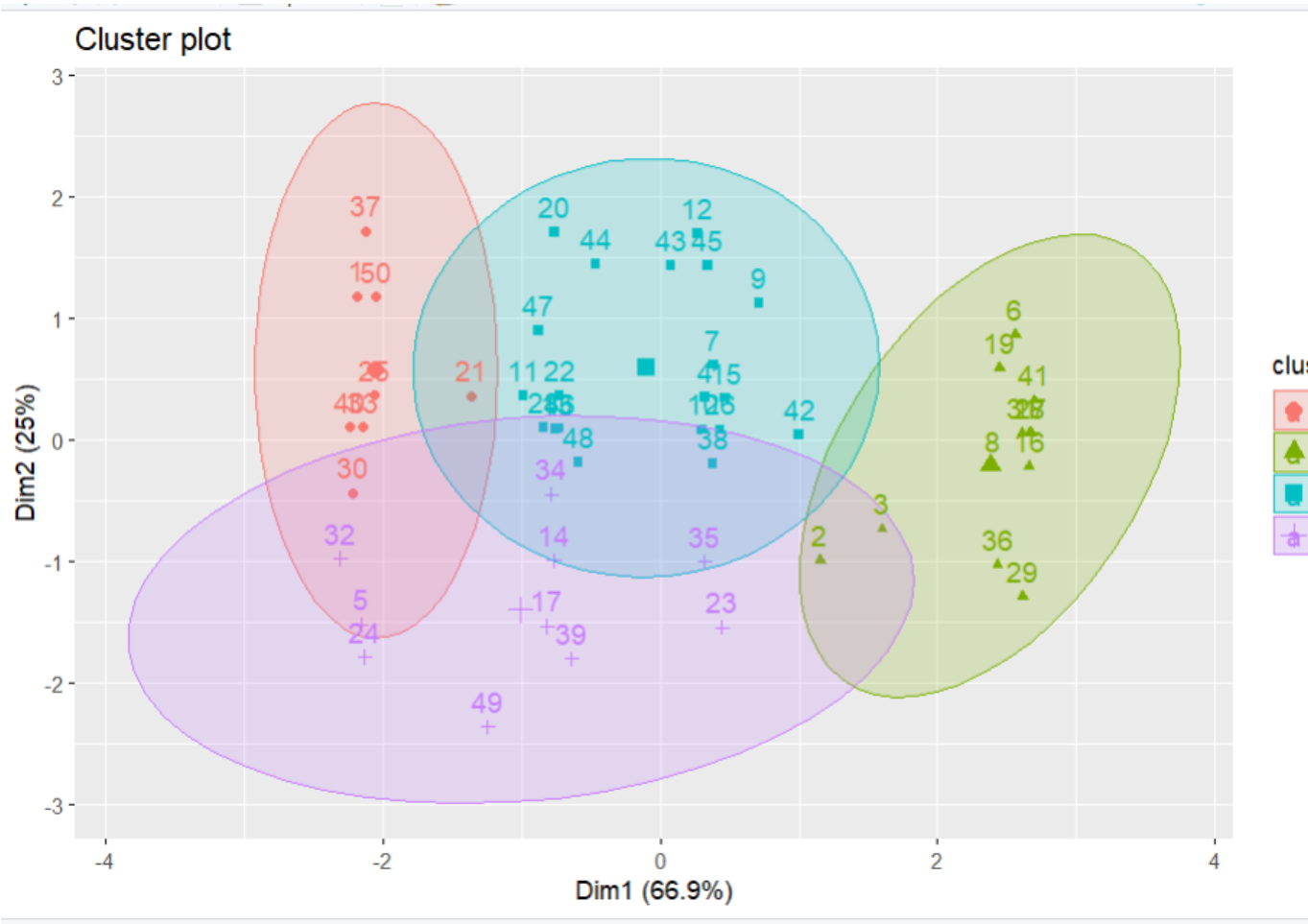


*Рис. 1 Оптимальное кол-во кластеров - 4*

В ходе выполнения работы были выявлены 4 кластера в данных варианта:

1. Красный – старшие программисты с малым стажем и наибольшим временем выполнения тестовых заданий. Большинство не имеют сертификатов о повышении квалификации.
2. Зеленый – старшие программисты с большим стажем и меньшим временем выполнения тестовых заданий. Большинство имеют сертификаты о повышении квалификации 3 уровня (максимальный).
3. Голубой – старшие программисты с средним стажем и средним временем выполнения тестовых заданий. Большинство имеют сертификаты 2 уровня.
4. Фиолетовый – молодые программисты (<30) с малым стажем работы и большим временем выполнения заданий. Несмотря на слабые показатели при тестах, большинство опережают программистов красного кластера.

## Разработка базы данных «Бытовая техника»

**1. Задание:**
Сформировать несколько таблиц. Предусмотреть: ввод данных, редактирование, просмотр данных. Обязательные требования к базе данных: наличие таблиц-справочников и таблиц, использующих справочники; предусмотреть следующие роли: оператор базы данных; пользователь базы данных; администратор БД.

**2. Графическое представление связей между таблицами:**

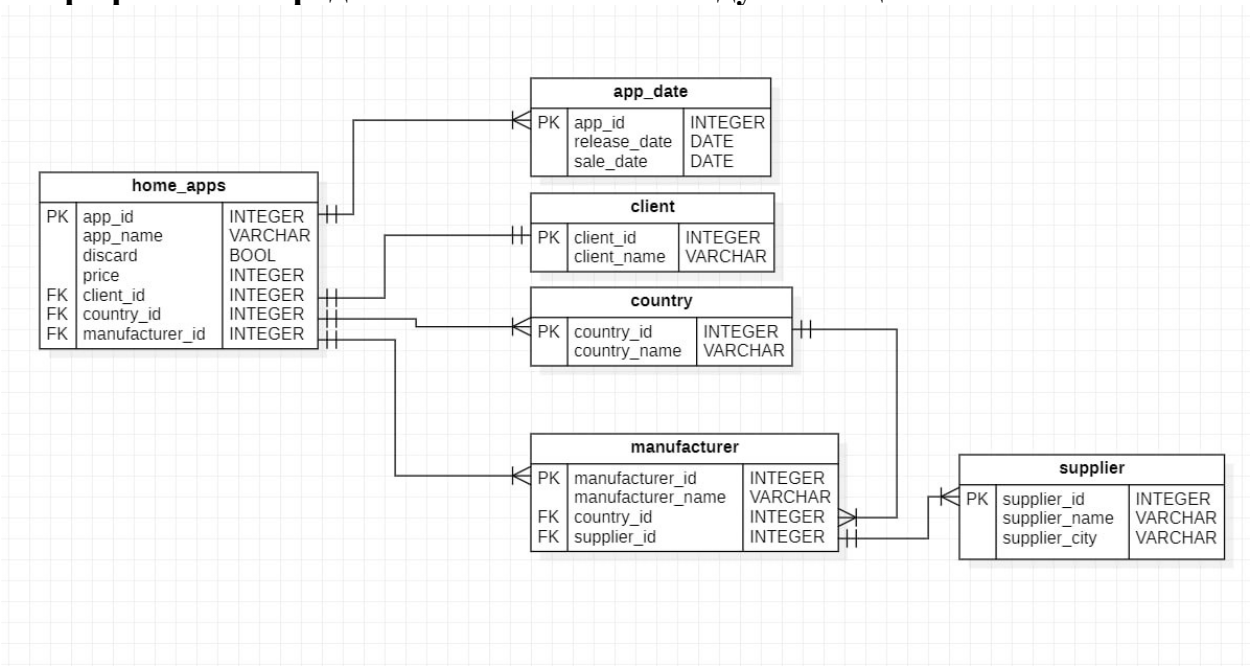


*Рисунок 2. Графическое представление связей между таблицами*

## Анализ результатов и выводы

Разработана база данных в СУБД PostgreSQL. В базе данных присутствуют таблицы -справочники и таблицы, использующие справочники. В пояснительной записке представлена структура таблиц и связи между ними. Написаны функции для реализации запросов с использованием процедурного языка plpgsql, приведены примеры результатов выполнения функций. Для реализации курсовой работы были применены знания и навыки разработки БД, выполнения запросов, написания функций, полученные в ходе выполнения лабораторных работ по курсу и изучения документации PostgreSQL.

**Изучение принципов работы сверточных нейронных сетей для задачи классификации изображений с использованием библиотеки Tensorflow**.

**Ход работы:**

Зафиксируем количество образцов и меток классов.

```
Тип данных структуры train_images  <class 'numpy.ndarray'>
Тип данных структуры train_labels  <class 'numpy.ndarray'>

Состав:
  - обучающего   набора изображений: (60000, 28, 28)
  - проверочного набора изображений: (10000, 28, 28)
  - обучающего   набора меток: (60000,)
  - проверочного набора меток: (10000,)
```

Рис.1 Количество образцов и меток классов

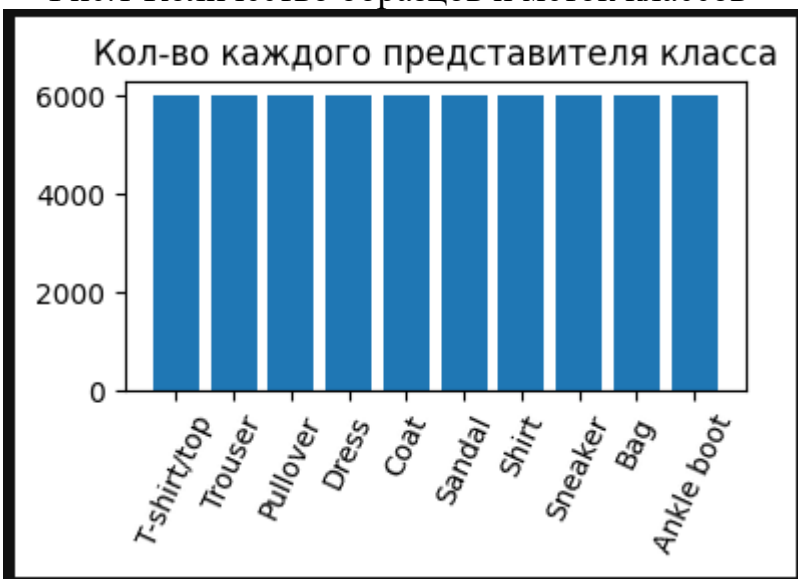


Рис.2 Количество образцов и меток классов

- Примеры графических изображений выбранных образцов.

Класс: bag

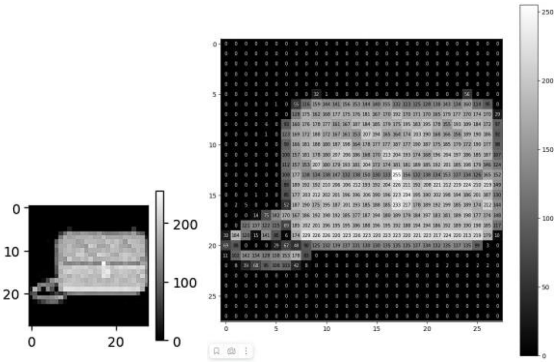

Рис 3. bag

Класс Dress:

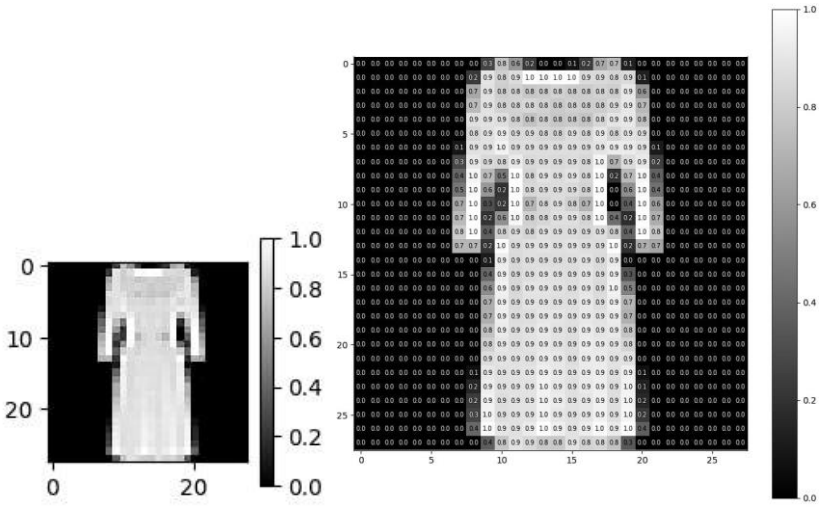

Рис 4. T-shirt

## Опыт №1

```
# 2.1) Построение модели нейронной сети многослойного(Sequential) типа ----------------------------
model = tf.keras.Sequential()

# ДОБАВЛЕНИЕ НОВЫХ СЛОЁВ --------------------------
model.add( layers.Input( shape=(28,28,1) ))  # Входной слой, обеспечивает приём всех пикселей исходного изображения  shape=(height, width, colorChannels)

#========================== Блок выделения признаков ======================
model.add( layers.Conv2D      ( filters= 2, kernel_size=( 3, 3, ),  strides     = 1 ,   activation=None,  ))
model.add( layers.AveragePooling2D( pool_size=( 4, 3, ),  strides     = 3 ,   ))
##... дополнительные пары свёрточный + объединяющий
##model.add( layers.Conv2D (...))
##model.add( layers.>>>Вид_объеденяющего_слоя<<< (...))

#========================== Блок классификации ========================
model.add( layers.Flatten() )                          # Слой для преобразования изображений в одномерный вектор
model.add( layers.Dense(units= 64, activation= activations.sigmoid, ))  # Скрытый полносвязный слой
##... дополнительные скрытые слои классификатора

#========================== Выходной слой вычисления y_i ======================
model.add( layers.Dense(units= 10, activation= activations.softmax  )) # Выходной слой. Для классификации activation= softmax.
```

Рис 5. Архитектура сверточной сети НС с описанием последовательности и параметров выбранных слоев

```
Вывод короткой информации о структуре НС
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_8 (Conv2D)           (None, 26, 26, 2)         20

 average_pooling2d_5 (Averag  (None, 8, 8, 2)          0
 ePooling2D)

 flatten_4 (Flatten)         (None, 128)               0

 dense_8 (Dense)             (None, 64)                8256

 dense_9 (Dense)             (None, 10)                650

=================================================================
Total params: 8,926
Trainable params: 8,926
Non-trainable params: 0
_________________________________________________________________
```

Рис 6. Краткая информация о структуре НС

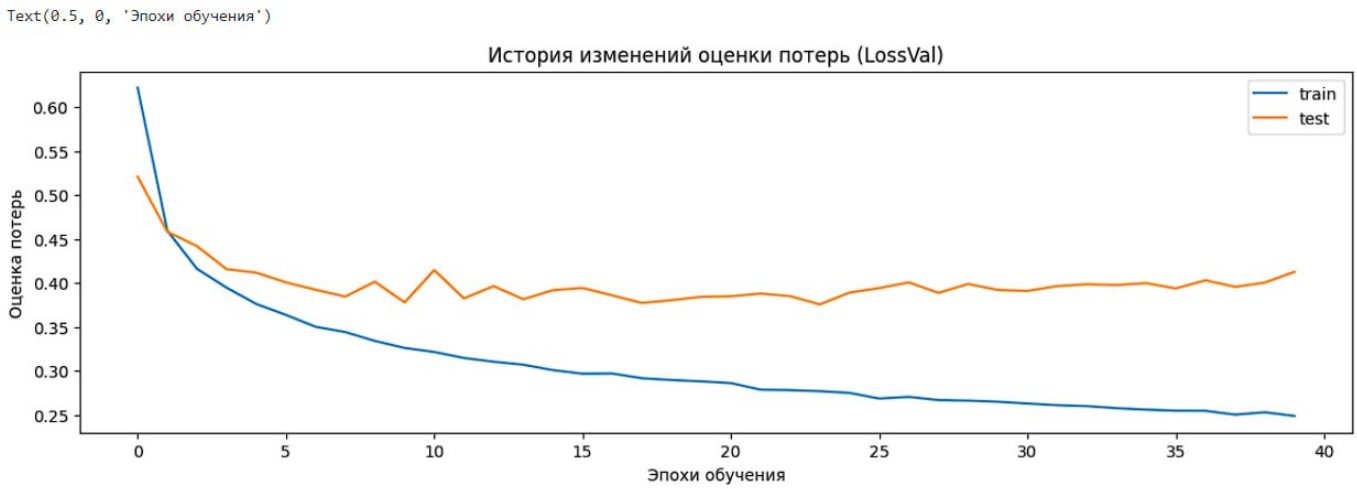


Рис 7. Графики истории изменения функции потерь по эпохам

**Вывод:**

В ходе выполнения эксперимента изучены основы работы сверточных нейронных сетей (Convolutional Neural Networks, CNN) в контексте классификации изображений. В качестве инструмента для реализации и обучения CNN была применена библиотека TensorFlow. Обучение модели проводилось на наборе данных, содержащем изображения разных предметов гардероба.

Было проведено 4 опыта, после которых была получена точность обучения 90%, что является приемлемым значением.

## Аннотация к ВКРб

Выпускная квалификационная работа бакалавра выполнена студентом Великжаниным С.И.

Руководитель проекта: к.ф.-м.н. Борин В.М.

Тема: “Приложение для учета инвентаря бара в отеле “Grand Autograph”.

Пояснительная записка содержит листов – 63, рисунков – 58.

Ключевые слова: WEB, приложение, MVT, Django, Python, SQLite, Bootstrap, сервер.

Цель выпускной квалификационной работы – разработка приложения для учета инвентаря бара в отеле “Grand Autograph”. Приложение решает проблему работников бара в управлении логистикой и администрированием рабочих процессов. В частности, отслеживание наличия продукции в баре, составление заявок на пополнение инвентаря, ведение учета продаж и координирование персонала.

Достигнуты поставленные цели:

- Изучены рабочие процессы в баре отеля “Grand Autograph”;
- Разработана структура базы данных;
- Разработана клиентская часть приложения;
- Разработана серверная часть приложения;
- Протестирована работа приложения.

## Архитектура приложения

Для реализации приложения использован клиент-серверный архитектурный паттерн **MVT (Model – View – Template),** который позволяет создать расширяемую информационную системы с большим количеством компонентов. Такой вариант реализации дает возможность разделить приложение на клиентскую и серверную части. Разделение упрощает процесс разработки проекта, потому что изменения в программном коде интерфейса пользователя не касаются программного кода логики приложения, и наоборот – изменения в логике никак не влияют на внешний вид.

Данные приложения и пользователей хранятся на сервере в базе данных в защищенном виде, что повышает уровень безопасности приложения. На рисунке 8 приведена схема работы архитектурного паттерна MVT.

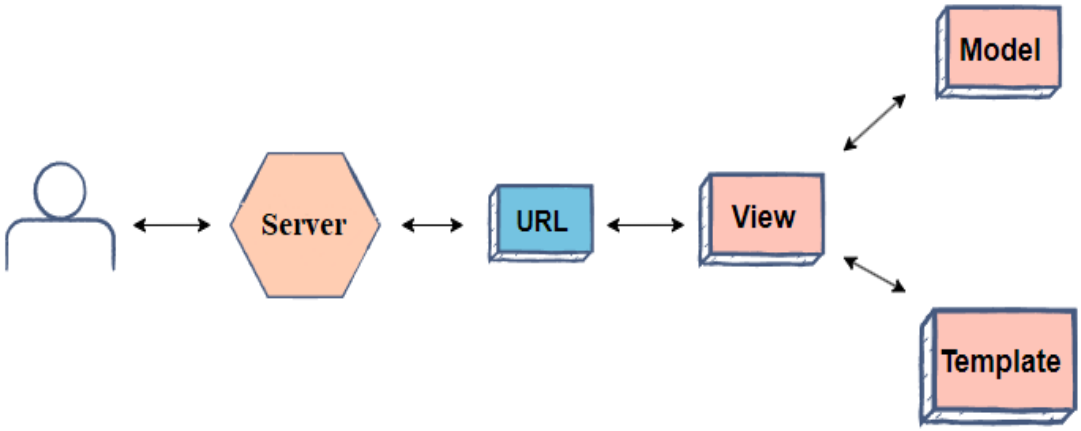


Рисунок 8 – Схема работы архитектурного паттерна MVT.

Для проектирования приложения взята клиент-серверная архитектура, которая подразумевает разделение на клиентскую и серверную части. Такой вариант реализации означает, что вся логика приложения находится на сервере, а пользователь взаимодействует только с интерфейсом клиентской части.

Интерфейс клиентской части приложения реализован с помощью веб-технологий, что облегчает процесс разработки и дает возможность разворачивать приложение как на локальном сервере, так и на удаленном, предоставляя доступ из любой точки мира через сеть Интернет. Для структурирования и представления содержимого интерфейса выбран язык разметки гипертекста **HTML** и каскадные таблицы стилей **CSS**. Данные технологии поддерживаются большинством современных браузеров, поэтому пользователю не придется устанавливать сторонние приложения, кроме самого браузера.

Для взаимодействия с клиентской частью приложения и реализации динамичного функционала выбран язык **JavaScript**, потому что он может работать с содержимым HTML-страницы. Для упрощения процесса

разработки был выбран фреймворк Bootstrap, который предоставляет готовые классы и компоненты.

Для серверной части выбран языка программирования **Python**, который имеет простой синтаксис и большой выбор библиотек для веб-разработки. Одна из основных библиотек приложения – высокоуровневый фреймворк Django, который включает в себя настройку веб-сервера, механизмы для авторизации пользователей, административный интерфейс. Для хранения данных приложения используется БД **SQLite**, потому что она имеет реляционную структуру хранения данных, встраивается напрямую в приложение и использует минимальные ресурсы системы.

В качестве среды разработки выбрана PyCharm, которая имеет глубокую интеграцию с фреймворком **Django** в виде инструментов анализа, автодополнения и рефакторинга программного кода в реальном времени, что позволяет предотвратить ошибки и повышает скорость написания кода.

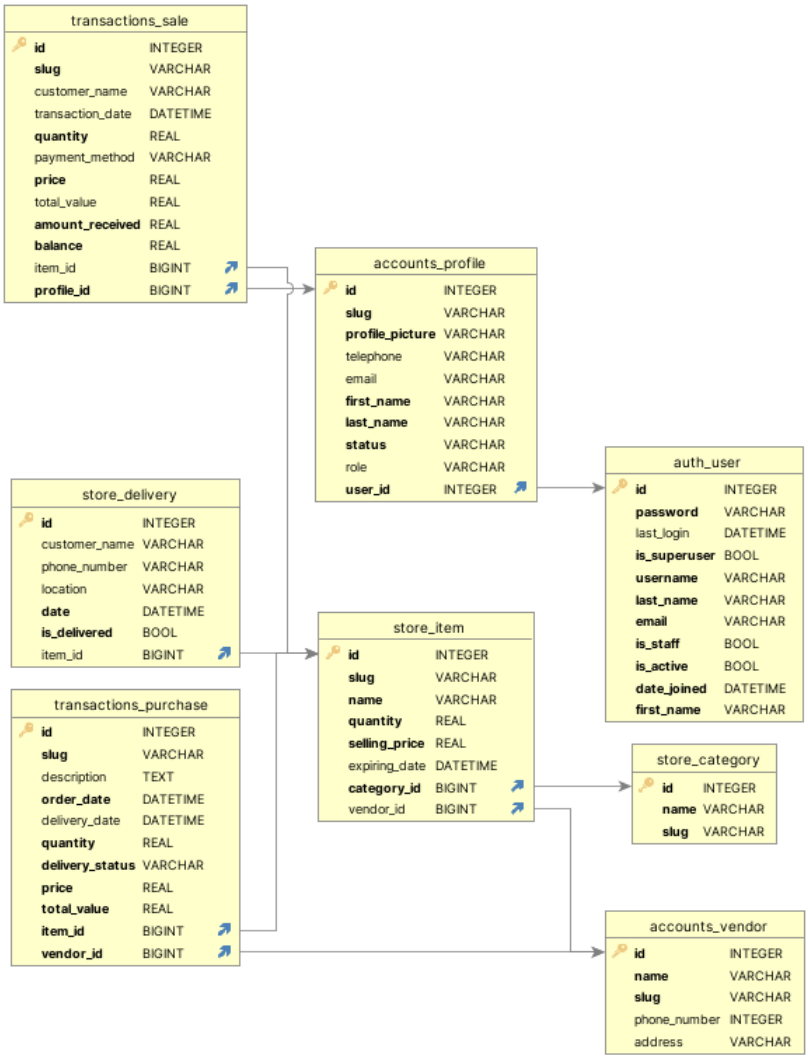


Рисунок 9 – Структура базы данных

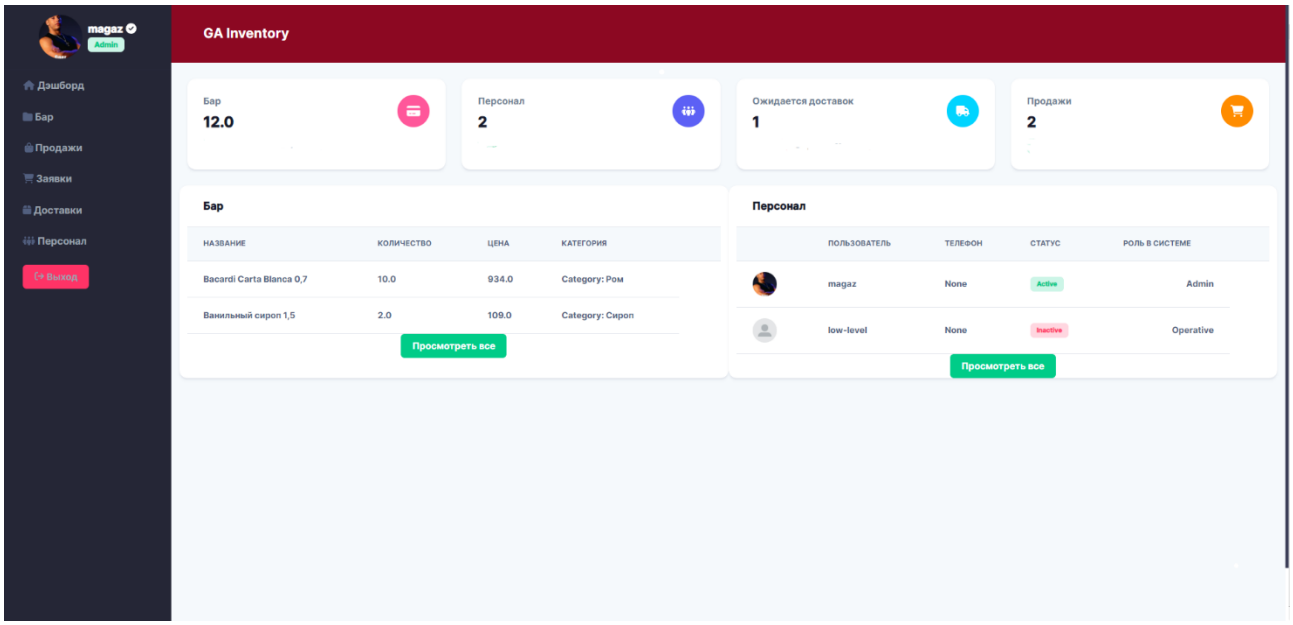


Рисунок 10 – Домашняя страница “Дэшборд”

В выпускной квалификационной работе бакалавра “Приложение для учета инвентаря бара в отеле “Grand Autograph” было разработано веб-приложение, которое решает проблему неэффективности рабочих процессов, связанных с инвентаризацией.

В частности, дает возможность отслеживать, управлять и редактировать информацию о продукции, находящейся на баре в реальном времени, управлять и отслеживать статус сотрудников, а также просматривать статистику доставок и продаж.