**Приложение А**

Диаграмма вариантов использования

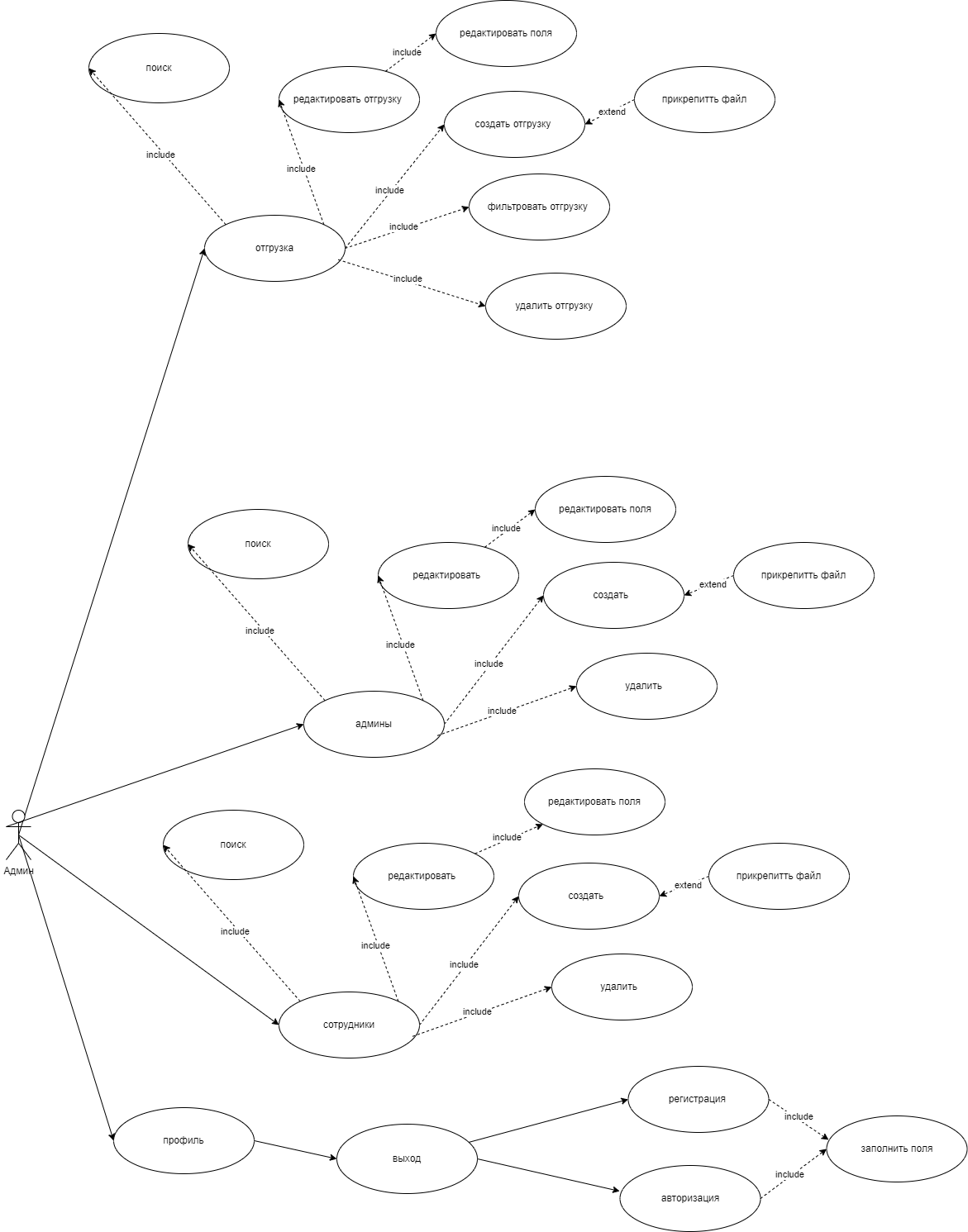


Рисунок А.1 — Диаграма вариантов использования

**Приложение Б**

Тесты на использование

Таблица Б.1 — Тесты на использование

Вот пример тест-кейсов в виде таблицы, с учетом предоставленных данных:

| Идентификатор | Приоритет | Заглавие и шаги выполнения | Исходная информация | Ожидаемый результат |
| --- | --- | --- | --- | --- |
| TC001 | Высокий | Проверка отображения отгрузок на главной странице1. Открыть главную страницу.2. Проверить отображение списка отгрузок.3. Убедиться, что отгрузки отсортированы по дате в порядке убывания. | Пользователь авторизован, существует хотя бы одна отгрузка в базе данных. | Список отгрузок отображается и отсортирован по дате в порядке убывания. |
| TC002 | Высокий | Проверка перенаправления на страницу регистрации для неавторизованных пользователей1. Открыть главную страницу.2. Проверить, что неавторизованный пользователь перенаправляется на страницу регистрации. | Пользователь не авторизован. | Пользователь перенаправляется на страницу регистрации. |
| TC003 | Высокий | Создание новой отгрузки1. Перейти на страницу создания отгрузки.2. Заполнить форму (имя отгрузки, водитель, транспортное средство, точка отправления, точка назначения).3. Нажать кнопку "Создать".4. Проверить, что новая отгрузка отображается в списке отгрузок. | Пользователь авторизован, существует хотя бы один водитель в базе данных. | Новая отгрузка успешно создается и отображается в списке отгрузок. |
| TC004 | Высокий | Обновление существующей отгрузки1. Перейти на страницу обновления отгрузки.2. Изменить данные отгрузки (имя отгрузки, водитель, транспортное средство, точка отправления, точка назначения).3. Нажать кнопку "Обновить".4. | Пользователь авторизован, существует хотя бы одна отгрузка в базе данных. | Данные отгрузки успешно обновлены. |

Продолжение таблицы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TC005 | Высокий | Удаление отгрузки1. Перейти на страницу удаления отгрузки.2. Нажать кнопку "Удалить".3. Проверить, что отгрузка удалена и больше не отображается в списке отгрузок. | Пользователь авторизован, существует хотя бы одна отгрузка в базе данных. | Отгрузка успешно удалена, не отображается в списке отгрузок. |
| TC006 | Средний | Поиск отгрузок1. Ввести критерии поиска (например, имя отгрузки, водитель, точка назначения) в поле поиска.2. Нажать кнопку "Поиск".3. Проверить, что отгрузки, соответствующие критериям поиска, отображаются. | Пользователь авторизован, существует несколько отгрузок в базе данных. | Отгрузки, соответствующие критериям поиска, отображаются. |

**Приложение В**

Листинг программы

from flask import Blueprint, render\_template, redirect, flash, url\_for, request, current\_app  
  
from ..functions import save\_picture  
from ..forms import RegistrationForm, LoginForm, ProfileForm  
from ..extensions import db, bcrypt  
from ..models.user import User  
from sqlalchemy import desc  
import os  
from flask\_login import login\_user, logout\_user, login\_required, current\_user  
  
user = Blueprint('user', \_\_name\_\_)  
  
@user.route('/user/register', methods=['POST', 'GET'])  
def register():  
 form = RegistrationForm()  
 if form.validate\_on\_submit():  
 hashed\_password = bcrypt.generate\_password\_hash(form.password.data).decode('utf-8')  
 avatar\_filename = save\_picture(form.avatar.data)  
 user = User(name=form.name.data, login=form.login.data, avatar=avatar\_filename, password=hashed\_password)  
 try:  
 db.session.add(user)  
 db.session.commit()  
 flash(f'user registered', 'success')  
 return redirect(url\_for('user.login'))  
 except Exception as e:  
 print(str(e))  
 flash(f'error user registered', 'danger')  
 return render\_template('user/register.html', form=form)  
  
  
@user.route('/user/login', methods=['POST', 'GET'])  
def login():  
 form = LoginForm()  
 if form.validate\_on\_submit():  
 user = User.query.filter\_by(login=form.login.data).first()  
 if user and bcrypt.check\_password\_hash(user.password, form.password.data):  
 login\_user(user, remember=form.remember.data)  
 next\_page = request.args.get('next')  
 flash(f'Вы успешно авторизованы', 'success')  
 return redirect(next\_page) if next\_page else redirect(url\_for('shipping.all'))  
 else:  
 flash(f'Ошибка входа, проверьте логин и пароль ', 'danger')  
 return render\_template('user/login.html', form=form)  
  
  
@user.route('/user/logout', methods=['POST', 'GET'])  
def logout():  
 logout\_user()  
 return redirect(url\_for('shipping.all'))  
  
  
@user.route('/user/profile')  
@login\_required # Требует входа в систему  
def profile():  
 return render\_template('post/profile.html', user=current\_user)  
  
  
@user.route('/user/all\_profile')  
@login\_required  
def all\_profile():  
 users = User.query.order\_by(desc(User.date)).all()  
 return render\_template('user/all\_profile.html', users=users)  
  
  
@user.route('/user/<int:id>/update\_user\_profile', methods=['POST', 'GET'])  
@login\_required  
def update\_user\_profile(id):  
 user = User.query.get(id)  
 if not user:  
 flash("Пользователь не найден", "danger")  
 return redirect('/')  
  
 form = ProfileForm()  
 if form.validate\_on\_submit():  
 user.name = form.name.data  
 user.login = form.login.data  
  
 # Проверяем загрузку нового аватара  
 if form.avatar.data:  
 user.avatar = save\_picture(form.avatar.data)  
  
 # ЛОГИРУЕМ ЗНАЧЕНИЕ ПЕРЕД СОХРАНЕНИЕМ  
 print(f"Аватар перед коммитом: {user.avatar}")  
  
 db.session.commit()  
 flash("Профиль успешно обновлён!", "success")  
 return redirect(url\_for('user.update\_user\_profile', id=user.id))  
  
 return render\_template('user/update\_user\_profile.html', user=user, form=form)  
  
  
@user.route('/user/<int:id>/delete\_user', methods=['POST', 'GET' ])  
@login\_required  
def delete\_user(id):  
 user = User.query.get(id)  
 try:  
 db.session.delete(user)  
 db.session.commit()  
 return redirect('/user/all\_profile')  
 except Exception as e:  
 print(str(e))  
 return str(e)  
  
@user.route('/search', methods=['GET'])  
def search():  
 search\_query = request.args.get('search\_query', '').strip()  
 # Логика поиска по пользователям  
 users = User.query.filter(User.name.ilike(f'%{search\_query}%')).all()  
 return render\_template('user/all\_profile.html', users=users)