React-компонент – это простая функция JavaScript, которая возвращает элементы React [3]. Как уже отмечалось, создание React было вызвано проблемой с поддержанием и улучшением больших приложений, где компоненты не имели такой простой привязки. React с легкостью решает эту проблему. У компонентов разделена логика выполнения, и сложный компонент можно создать, используя более простые структуры.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Начало работы с React 2022 [Электронный ресурс]. – Режим доступа: https://developer.mozilla.org/ru/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_getting_started. – Дата доступа: 30.09.2022.

2. Stack Overflow Developer Survey 2022 [Электронный ресурс]. – Режим доступа: https://survey.stackoverflow.co/2022/#most-loved-dreaded-and-wanted-language-love-dread. – Дата доступа: 30.09.2022.

3. Словарь терминов React – React [Electronic resource]. – Access mode: https://ru.reactjs.org/docs/glossary.html. – Date of access: 31.09.2022.

А. А. КОЗИНСКИЙ
Брест, БрГТУ

ИСПОЛЬЗОВАНИЕ СЕТЕЙ ОБНАРУЖЕНИЯ ОБЪЕКТОВ НА ОСНОВЕ СЕМЕЙСТВА YOLO

В 2022 г. нами выполнена разработка метода статистической оценки интенсивности движения на городских магистралях. Краткая формулировка постановки задачи состояла в следующем: сформировать датасет с помощью веб-камеры для наблюдения за объектами; средствами нейронной сети выполнить распознавание движущихся объектов (легковых автомобилей, микроавтобусов, грузовых автомобилей, пешеходов, мотоциклистов и велосипедистов); разработать приложение для автоматизации оценки статистической загрузки перекрестков в режиме онлайн трансляции в глобальной сети Интернет. Пример перекрестка с установленной камерой приведен в [1]. Указанный перекресток «похож» по расположению на перекресток, находящийся на улице Колесника в г. Бресте. Свойство («похожести») может быть использовано для обучения и тестирования нейронной сети в случае, когда целевой перекресток недоступен.

Для выбора возможного подхода к детектированию объектов выполнен анализ готовых моделей распознавания. Среди возможных

решений нами рассматривались следующие модели и технологии: OpenCV, Deep Learning, YOLO, Single Shot Detectors (SSDs), Faster R-CNN, Mask R-CNN, HOG + Linear SVM, Haar cascades. Для дальнейшего использования определена модель YOLOv5 семейства YOLO [2]. Она обладает таким важным качеством, как официальная, поддерживаемая модель с открытым исходным кодом, размещенным на витрине Torch Hub, реализованным в PyTorch. Это позволило нам использовать среду Colaboratory для реализации детектирования объектов и получить оценку статистической загрузки перекрестка в зоне наблюдения произвольной онлайн-камеры. Метод, реализованный нами, может быть использован для оценки загруженности магистралей всего города при условии увеличения вычислительных мощностей.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Веб-камера на перекрестке Ленина – Андропова, г. Петрозаводск (Российская Федерация) [Электронный ресурс]. – Режим доступа: https://www.geocam.ru/online/lenina-andropova. Дата доступа: 26.09.2022.

2. Object Detection [Electronic resource]. – Access mode: https://pyimagesearch.com/category/object-detection/. – Date of access: 26.09.2022.

А. П. КОНДРАТЮК
Брест, БрГУ имени А. С. Пушкина

ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЙ ENTITY FRAMEWORK CORE

Существует много примеров разного уровня по юнит-тестированию, показывающих, как просто можно проверять логику приложений с помощью юнит-тестов. Однако не все так просто бывает при тестировании приложений, в которых центральную роль играет база данных. Такая ситуация часто встречается при создании веб-приложений. При модульном тестировании приложений часто возникает проблема тестирования базы данных [1] .

Так как тестирование может повлиять на архитектуру кода, рекомендуется планировать его заранее и обеспечить хорошее покрытие тестами по мере развития приложения.

Когда создаются тесты для приложения EF Core, необходимо решить, будут ли они затрагивать систему рабочих баз данных, как само