УДК 004.4
DOI 10.17513/snt.40061

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ПРОВЕРКИ КОДА ДЛЯ ПОДГОТОВКИ СПЕЦИАЛИСТОВ В СФЕРЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

[1]Нажимова Н.А., [2]Нажимов А.В.

[1]ФГБОУ ВО «Нижегородский технический университет имени Р.Е. Алексеева», филиал, Дзержинск, e-mail: adilia@list.ru;
[2]ФГАОУ ВО «Национальный исследовательский Нижегородский государственный университет имени Н.И. Лобачевского», филиал, Дзержинск, e-mail: avbix@mail.ru

В статье рассматривается система автоматической проверки кода студентов IT-специальностей, основанная на использовании системы контроля версий Git и облачного сервиса репозиториев GitHub. Цель исследования заключается в разработке автоматизированной системы проверки кода студентов IT-специальностей. Описаны ключевые преимущества данной системы, такие как автоматизация процесса проверки, повышение объективности оценок, масштабируемость, оперативность обратной связи и поддержка дистанционного обучения. Автоматизация позволяет преподавателям освободиться от рутинной работы, сосредоточиться на более значимых аспектах образовательного процесса, включая индивидуальную поддержку студентов и развитие учебных программ. Инструменты GitHub Actions интегрированы для выполнения тестирования кода. Это позволяет создать эффективную и надежную инфраструктуру для проверки заданий. Подчеркнута важность использования современных инструментов в образовательном процессе, что способствует подготовке студентов к реальной профессиональной деятельности. В статье также обсуждаются направления дальнейшей работы, включая расширение функциональности системы, разработку методов персонализированной обратной связи, применение машинного обучения для анализа кода и интеграцию с учебными платформами. Эти улучшения могут повысить качество обучения, сделать процесс проверки более гибким и адаптивным, а также обеспечить студентов необходимыми навыками для работы в условиях современной разработки программного обеспечения. Система автоматической проверки кода на базе Git и GitHub представляет собой важный шаг в сторону инноваций в области образования и подготовки высококвалифицированных специалистов в сфере IT.

Ключевые слова: автоматическая проверка кода, тестирование, Git, GitHub, GitHub Actions, IT-специальности, образование

AUTOMATED CODE REVIEW SYSTEM FOR TRAINING SPECIALISTS IN THE FIELD OF INFORMATION TECHNOLOGY

[1]Nazhimova N.A., [2]Nazhimov A.V.

[1]Nizhny Novgorod state technical University named after R.E. Alekseev, branch, Dzerzhinsk, e-mail: adilia@list.ru;
[2]National Research Lobachevsky State University of Nizhny Novgorod, branch, Dzerzhinsk, e-mail: avbix@mail.ru

The article discusses an automated code review system for IT students, based on the use of the Git version control system and the GitHub cloud repository service. The aim of the study is to develop an automated code review system for IT students. The key advantages of this system are described, such as automation of the review process, increased objectivity of assessments, scalability, timely feedback, and support for distance learning. Automation allows educators to be freed from routine tasks, enabling them to focus on more significant aspects of the educational process, including individual student support and curriculum development. GitHub Actions tools are integrated to perform code testing. This creates an efficient and reliable infrastructure for assignment evaluation. The importance of using modern tools in the educational process is emphasized, as it helps prepare students for real-world professional activities. The article also discusses directions for future work, including expanding system functionality, developing methods for personalized feedback, applying machine learning for code analysis, and integrating with educational platforms. These improvements can enhance the quality of education, make the review process more flexible and adaptive, and equip students with the necessary skills for working in modern software development environments. The automated code review system based on Git and GitHub represents an important step towards innovation in education and the training of highly qualified IT specialists.

Keywords: automated code review, testing, Git, GitHub, GitHub Actions, IT specialties, education

В настоящее время профессии из сферы информационных технологий становятся все более востребованными и динамично развивающимися, требуя от высших учебных заведений и преподавателей IT-специальностей более эффективных методов обучения и оценки знаний [1]. При этом значительная часть процесса обучения сопряжена с написанием кода программ на различных языках программирования, набор которых с каждым днем становится шире. В этом контексте автоматизация

процесса проверки заданий по написанию кода становится необходимой и актуальной задачей.

Автоматизация процесса проверки заданий по программированию имеет ряд значимых преимуществ. Прежде всего, она позволяет освободить время преподавателя от монотонной и рутинной работы, связанной с проверкой большого объема кода. Это время можно использовать для более важных задач, таких как разработка учебных материалов, проведение индивидуальных консультаций со студентами и повышение качества обучения в целом. Кроме того, автоматизированные системы проверки кода (АСПК) позволяют улучшить качество обратной связи для студентов. Быстрая и точная оценка и комментирование кода помогают студентам лучше понять свои ошибки и улучшить свои навыки программирования. АСПК также обеспечивают необходимую объективность. Оценки основываются на заданных критериях и алгоритмах, что исключает субъективные суждения и обеспечивает равные условия для всех студентов. АСПК могут легко масштабироваться для обработки большого объема работ, что особенно важно при проведении массовых офлайн- или онлайн-курсов. При этом время, необходимое для проверки одного задания, существенно сокращается. В условиях дистанционного обучения АСПК становится особенно ценной, поскольку облегчает оценку работ студентов и обеспечивает непрерывное обучение без необходимости присутствия в аудитории. В целом использование современных технологий в обучении, включая АСПК, демонстрирует передовой подход учебного заведения и его готовность использовать инновации для обеспечения высокого качества образования, что соответствует Указу Президента РФ от 21.07.2020 № 474 «О национальных целях развития Российской Федерации на период до 2030 года» [2].

Цель исследования заключается в разработке автоматизированной системы проверки кода студентов IT-специальностей.

Материалы и методы исследования

Рассмотрим различные подходы и инструменты, которые могут быть использованы для реализации АСПК, включая использование тестовых фреймворков, статический анализатор кода, различные плагины и расширения для интегрированных сред разработки (IDE), интеграция с системами управления версиями.

Использование тестовых фреймворков: тестовые фреймворки, такие как JUnit для Java, pytest для Python или Jasmine для JavaScript, позволяют написать автоматизированные тесты для разных аспектов кода, включая его функциональность, производительность и безопасность.

Статические анализаторы кода: инструменты статического анализа кода, например, ESLint для JavaScript или pylint для Python, автоматически проверяют код на соответствие стандартам кодирования, выявляют потенциальные ошибки и предупреждают о неправильном использовании языковых конструкций [3].

Плагины и расширения для IDE: многие современные интегрированные среды разработки (IDE), такие как Visual Studio Code, IntelliJ IDEA или PyCharm, предлагают плагины и расширения для автоматической проверки кода на лету, а также для выполнения автоматизированных тестов и анализа качества кода [4].

Все рассмотренные выше варианты реализации АСПК вполне пригодны для проверки кода в контексте разработки программного обеспечения, однако их сложно применить для проверки кода студентов, выполняющих определенное задание. Во-первых, все описанные системы напрямую не поддерживают обмен кодом и возможность его обсуждения со студентом. Во-вторых, использование конкретного инструментария накладывает ограничение на использование различных языков программирования в рамках одной АСПК. В-третьих, многие из описанных решений так или иначе являются коммерческими продуктами западных разработчиков, что в условиях санкций может накладывать определенные ограничения на их использование. Описанные недостатки можно преодолеть благодаря использованию в качестве основы для АСПК систем управления версиями. Использование систем управления версиями, таких как Git, в сочетании с автоматизированными средствами непрерывной интеграции и непрерывной доставки (CI/CD), позволяет автоматически проверять код, оставлять обратную связь и реализовать возможность проверки кода на разных языках программирования в рамках одной АСПК.

В работе предложено использовать систему контроля версий Git в качестве основы для создания АСПК. Следует отметить, что знание студентами основ системы контроля версий Git является необходимым условием для успешного применения АСПК в процессе обучения. Однако, как показывает опыт, подобные знания легко прививаются в рамках 5-часового лабораторного практикума и в свою очередь являются весьма полезными в контексте будущего трудоустройства по ряду причин, описанных ниже.

Git – это распределенная система контроля версий (DVCS), которая позволяет разработчикам отслеживать изменения в коде, работать совместно над проектами и управлять различными версиями кода [5]. В отличие от централизованных систем контроля версий, Git позволяет каждому разработчику иметь полную копию репозитория, что увеличивает надежность и гибкость работы. Git упрощает совместную работу над проектами, особенно в больших командах. Возможность создавать ветки и выполнять слияние изменений позволяет разработчикам работать над разными частями проекта одновременно, не мешая друг другу. Это важное умение для студентов, так как большинство проектов в индустрии разрабатываются коллективно. Git сохраняет историю всех изменений, сделанных в проекте, что позволяет разработчикам отслеживать и анализировать каждое из них в отдельности. Это полезно для выявления ошибок и анализа их причин. Студенты могут экспериментировать с новыми идеями без риска повредить основной код, что способствует более глубокому пониманию принципов разработки программного обеспечения. Git интегрируется с множеством современных инструментов и платформ, таких как GitHub, GitLab, Bitbucket и др. [6]. Знание Git предоставляет студентам доступ к этим инструментам, что позволяет им участвовать в открытых проектах open-source сообщества. Знание Git является необходимым требованием для большинства позиций в сфере разработки программного обеспечения. Это важный навык, который значительно повышает конкурентоспособность будущих специалистов на рынке труда.

Алгоритм работы АСПК на базе системы контроля версий Git и облачного сервиса GitHub в нотации BPMN 2.0 показан на рис. 1.

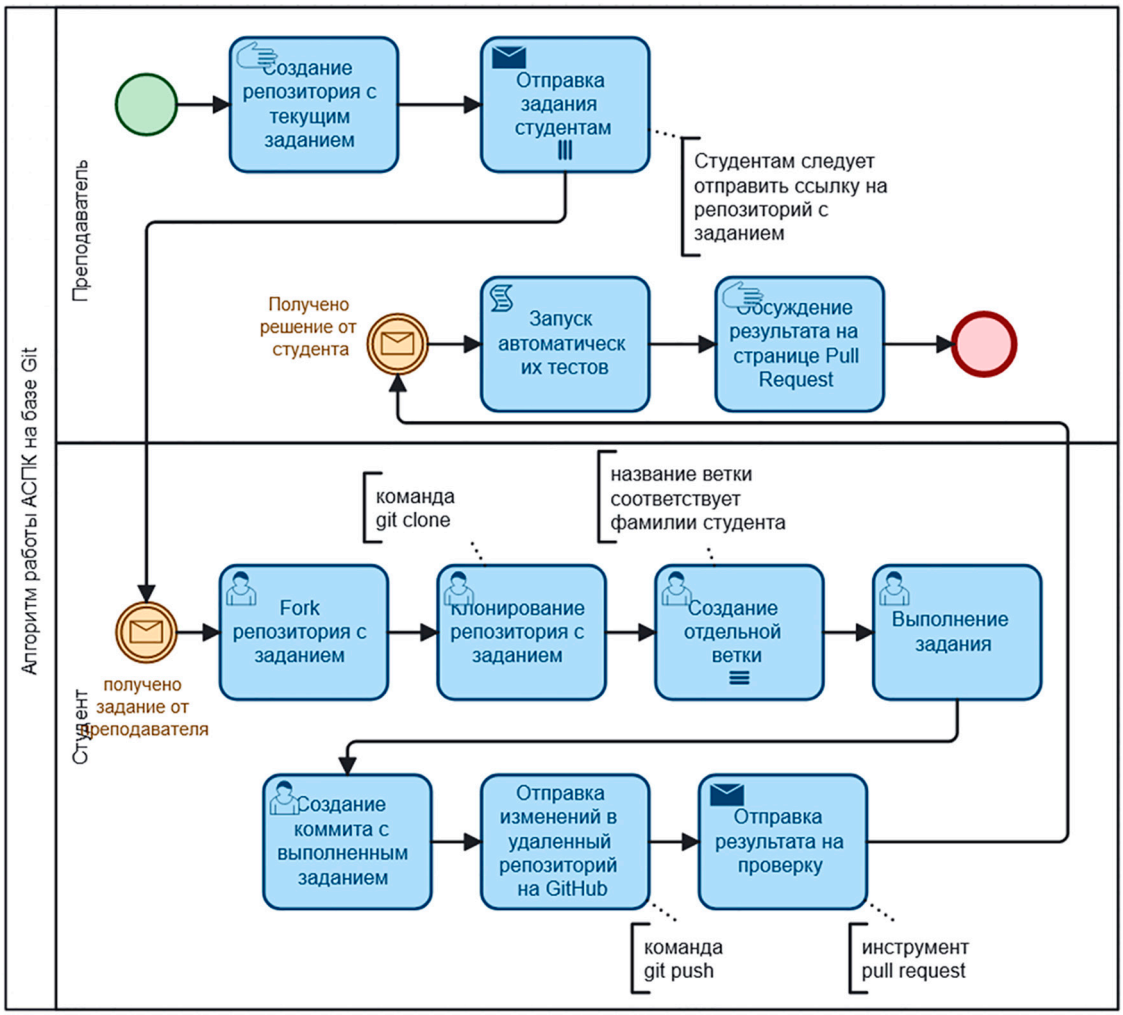

Рис. 1. Алгоритм работы АСПК на базе системы контроля версий Git и облачного сервиса GitHub в нотации BPMN 2.0

Преподаватель, использующий АСПК, начинает работу с создания репозитория с текущим заданием по написанию кода. Предполагается, что репозиторий создается на локальной машине педагога и далее отправляется на облачный сервис удаленных репозиториев GitHub. Здесь следует отметить, что использование GitHub в качестве основы для создания АСПК обусловлено высокой популярностью данного сервиса. Однако справедливо будет заметить, что и прочие подобные сервисы (GitLab, Bitbucket и др.), также обладают всем необходимым функционалом. Таким образом, на данном шаге алгоритма мы имеем репозиторий, который содержит файл с описанием задания. Затем ссылка на данный репозиторий отправляется студентам. Здесь может быть использован любой сервис обмена сообщениями либо система электронного обучения вуза. Получив сообщение, студент делает форк (fork) репозитория с заданием (рис. 2), то есть создает его копию в своем аккаунте облачного сервиса репозиториев, в нашем случае это GitHub. Затем студент клонирует упомянутый форк-репозиторий на свой локальный компьютер, использую команду *git clone <URL репозитория>*. В данном репозитории студент создает ветку, в качестве названия для которой выбирается фамилия студента. Команда для создания ветки *git branch <название ветки>*. Далее студент приступает к выполнению задания.

По завершении выполнения задания студент делает последний коммит командой *git commit* и отправляет изменения в свой удаленный репозиторий на облачном сервисе GitHub командой git push. Затем, используя инструмент Pull Request, студент отправляет задание на проверку преподавателю (рис. 2).

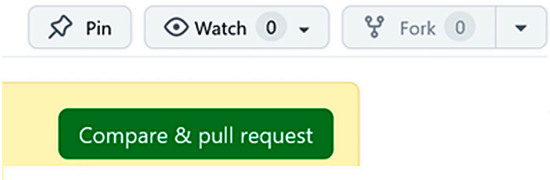

Рис. 2. Управляющие элементы на странице репозитория в GitHub для создания fork и pull request

При создании pull request преподаватель получает уведомление о новом решении на задание по написанию кода. Если в дальнейшем не требуются автоматические тесты, то на этом шаге можно остановиться, проверить код вручную и отправить комментарий непосредственно на странице pull request. В случае необходимости автоматической проверки кода в АСПК на базе Git и GitHub должны быть настроены инструменты GitHub Actions.

GitHub Actions – это встроенная в GitHub платформа автоматизации, предназначенная для упрощения выполнения различных задач в процессе разработки программного обеспечения. Этот набор инструментов позволяет автоматизировать такие процессы, как CI/CD, тестирование, развертывание и отправка уведомлений [7].

Как было отмечено выше, автоматическое тестирование кода можно настроить для абсолютного большинства современных языков программирования. Приведем в качестве примера алгоритм для настройки автоматической проверки кода на языке Python через GitHub Actions при создании pull request:

Создание файла конфигурации GitHub Actions:

1. Перейдите в репозиторий на GitHub.
2. В верхнем меню выберите вкладку Actions.
3. Нажмите на кнопку New workflow.
4. В появившемся окне выберите Set up a workflow yourself или один из предустановленных шаблонов, например Python application.

Создание основного файла конфигурации:

1. В каталоге репозитория должна быть создана директория .github/workflows.
2. В этой директории должен быть создан файл с расширением .yml, например python-app.yml (рис. 3).

Настройка кода проекта:

1. Убедитесь, что в вашем проекте есть файл requirements.txt для установки зависимостей, если они требуются.
2. Убедитесь, что у вас есть тесты в проекте. Например, создайте директорию tests и добавьте туда файл test_sample.py (рис. 4) с простым тестом.

Автоматическое выполнение GitHub Actions:

1. После создания pull request, GitHub Actions автоматически запустит конфигурацию, указанную в файле .github/workflows/python-app.yml.
2. На вкладке Actions в репозитории преподавателя можно будет наблюдать за ходом выполнения задачи, включая установку зависимостей, проверку стиля кода с помощью flake8 и выполнение тестов с помощью pytest.

Таким образом, GitHub Actions позволяет автоматизировать процесс проверки кода в рамках АСПК при создании pull request студентами. Это помогает обеспечить высокое качество кода и способствует выявлению ошибок.

```yaml
name: Python CI

on:
  pull_request:
    branches:
      - ivanov # Ветка соответствует фамилии студента
      - petrov
      - sidorov

jobs:
  test:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout repository
      uses: actions/checkout@v2

    - name: Set up Python
      uses: actions/setup-python@v2
      with:
        python-version: '3.8' # Можно изменить версию Python по необходимости

    - name: Install dependencies
      run: |
        python -m pip install --upgrade pip
        pip install flake8 pytest

    - name: Lint code
      run: |
        flake8 .

    - name: Run tests
      run: |
        pytest
```

Рис. 3. Код для файла python-app.yml

```python
# tests/test_sample.py
def test_example():
    assert 1 + 1 == 2
```

Рис. 4. Код для файла test_sample.py

Результаты исследования и их обсуждение

В данной статье была рассмотрена система автоматической проверки кода студентов IT-специальностей, основанная на использовании Git и GitHub. Подробно описаны преимущества этой системы, которые делают ее незаменимым инструментом в образовательной среде, позволяя преподавателям сосредоточиться на более важных аспектах обучения и профессионального развития студентов.

Следует также отметить направления дальнейшей работы, которые включают в себя: р*асширение функциональности*: интеграция дополнительных инструментов для проверки кода, таких как статические анализаторы, профайлеры производительности и инструменты для тестирования безопасности; *персонализация обратной связи*: разработка методов автоматической генерации персонализированных комментариев и рекомендаций для студентов на основе их кода и истории работы; *машинное обучение и искусственный интеллект*: использование методов машинного обучения для анализа кода студентов, предсказания возможных ошибок и предоставления рекомендаций по улучшению

кода; и*нтеграция с учебными платформами*: создание интеграций с популярными образовательными платформами и системами управления обучением, что позволит упростить процесс создания и управления учебными материалами.

Заключение

Возможность использования современных инструментов и платформ, таких как системы контроля версий, облачные сервисы удаленных репозиториев, скрипты для автоматической проверки кода на различных языках программирования, позволяют реализовать эффективную и надежную инфраструктуру для проверки заданий по созданию компьютерных программ различной сложности. Реализация таких систем особенно важна в условиях многочисленных групп и дистанционного обучения, которые соответствуют текущей ситуации в высшем техническом образовании в нашей стране.

Список литературы

1. Похорукова М.Ю. Особенности обучения программированию IT-специалистов на первом курсе // Современное педагогическое образование. 2023. № 10. С. 460–463.

2. Указ Президента РФ от 21.07.2020 № 474 «О национальных целях развития Российской Федерации на период до 2030 года» // СПС КонсультантПлюс. [Электронный ресурс]. URL: https://www.consultant.ru/document/cons_doc_LAW_357927/ (дата обращения: 03.04.2024).

3. Аристов М.С., Зотов Я.А., Яриков Д.В. Автоматизация тестирования программного обеспечения многопроцессорных систем реального времени // Знание. 2016. № 11–1. С. 97–105.

4. Thomas Clune, Michael Rilee, Damian Rouson. Testing as an Essential Process for Developing and Maintaining Scientific Software // 2nd Workshop on Sustainable Software for Science: Practices and Experiences (WSSSPE2). 2014. Vol. 9. P. 1–4. DOI:10.6084/m9.figshare.1112520.

5. Вьюшкова М.В., Чернова С.В. Принцип работы системы контроля версий GIT // Теория и практика современной науки. 2019. № 10 (52). С. 40–42.

6. Грузин Н.А., Голубничий А.А. Обзор и сравнение хостингов для git-репозиториев: Bitbucket, Github и Gitlab // E-Scio. 2021. № 1. С. 5–14.

7. Манаев Р.Г. Технология внедрения непрерывной интеграции в крупных высоконагруженных системах с минимизацией ошибок и временных потерь со стороны разработчиков // Инновации и инвестиции. 2020. № 12. С. 127–130.