Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский политехнический университет»

Кафедра «Инфокогнитивные технологии»
Образовательная программа «Веб-технологии»

Лабораторная работа № 8
по дисциплине «Программная инженерия»

Выполнил студент
группы 181-321 Карпов Вадим
Проверил:
преподаватель Будылина Евгения Александровна

Москва
2020

Модульное тестирование

Для тестирования веб-приложения были выбраны следующие инструменты: Jest, vue-test-utils, vue-jest.

Модульное тестирование — фундаментальная часть разработки программного обеспечения. В модульных тестах выполняются небольшие фрагменты кода в изоляции для упрощения добавления новых функциональных возможностей и отслеживания ошибок.

Однофайловые компоненты Vue позволяют просто писать модульные тесты для компонентов в изоляции. Это помогает разрабатывать новую функциональность с уверенностью, что это не ломает работу существующей, и помогает другим разработчикам понять, как работает компонент.

Тесты на примере компонента формы входа

```
import { shallowMount, createLocalVue } from '@vue/test-utils'
import Vuex from 'vuex'
import LoginForm from '../components/LoginForm.vue'

const localVue = createLocalVue()

localVue.use(Vuex)

describe('Компонент Входа', () => {
  let actions
  let store
  let mutations

   beforeEach(() => {
    actions = {
      createUser: jest.fn(),
      signIn: jest.fn(),
      signOut: jest.fn()
    };
    mutations = {
      openLoginForm: jest.fn(),
      closeLoginForm: jest.fn(),
      updateUser: jest.fn(),
    }

    store = new Vuex.Store({
      state: {
        loginForm: false,
        user: null,
      },
      getters: {
        getLoginFormState: jest.fn(),
        getUser: jest.fn(),
        isAuthenticated: jest.fn()
      },
      actions,
      mutations
    })
  })
```

```
    test('Компонент является экземпляром Vue', () => {
      const wrapper = shallowMount(LoginForm,  { store, localVue })

      expect(wrapper.findComponent(LoginForm).vm).toBeTruthy()
    })

    test('Кнопка отправки по умолчанию в статичном состоянии', () => {
      const wrapper = shallowMount(LoginForm,  { store, localVue });
      const btn = wrapper.findComponent({ ref: 'sign' });

      expect(btn.props('loading')).toBe(false)
    })

    test('Компонент формы закрывается по клику', () => {
      const wrapper = shallowMount(LoginForm,  { store, localVue })
      const form = wrapper.find('#popUp');

      form.trigger('click')

      expect(mutations.closeLoginForm).toHaveBeenCalled()
    })

    test('Осуществляется переход на страницу регистрации', () => {
      const $route = {
        name: "Registration"
      };
      const push = jest.fn();
      const $router = {
        push: jest.fn()
      };
      const wrapper = shallowMount(LoginForm,  { store, localVue,  mocks: { $rout
e, $router } })
      const regBtn = wrapper.find('#reg');

      regBtn.trigger('click')

      expect(wrapper.vm.$route.name).toBe($route.name)
    })

    test('Компонент формы отправляет данные на сервер', () => {
      const wrapper = shallowMount(LoginForm,  { store, localVue })
      wrapper.setData({
        login: 'testName@mail.com',
        password: '123456789'
      })

      const signInButton = wrapper.findComponent({ ref: 'sign' });

      signInButton.trigger('click')
```

```
      expect(actions.signIn).toHaveBeenCalledWith(expect.anything(), expect.objec
tContaining({
        'email': 'testName@mail.com',
        'password': '123456789'
      }))
    })

})
```

Результаты тестирования системы