Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Петрозаводский государственный университет»
Институт математики и информационных технологий
Кафедра информатики и математического
обеспечения

Ключников Максим Сергеевич
22608

"Тестирование web-приложения"

Отчет по дисциплине

Верификация ПО

Преподаватель:

Кулаков Кирилл
Александрович

Петрозаводск — 2018

Содержание

План тестирования

Описание объекта тестирования

Объектом тестирования является web-приложение, предназначенное для создания skill-билдов (наборов навыков) для онлайн игры TESO и возможности поделиться ссылкой на созданной билд. Серверная часть приложения разработана на языке PHP 7.2 с использованием фреймворка Yii2. Клиентская часть приложения - при помощи фреймворка Twitter Bootstrap и фреймворка Jquery.

Функциональность объекта тестирования

Пользователей приложения можно разделить на три роли.

1. Неавторизованные пользователи - гости
2. Авторизованные пользователи - пользователи
3. Авторизованные пользователи - администраторы

Для каждой роли предусмотрен свой функционал приложения.

Список функций приложения:

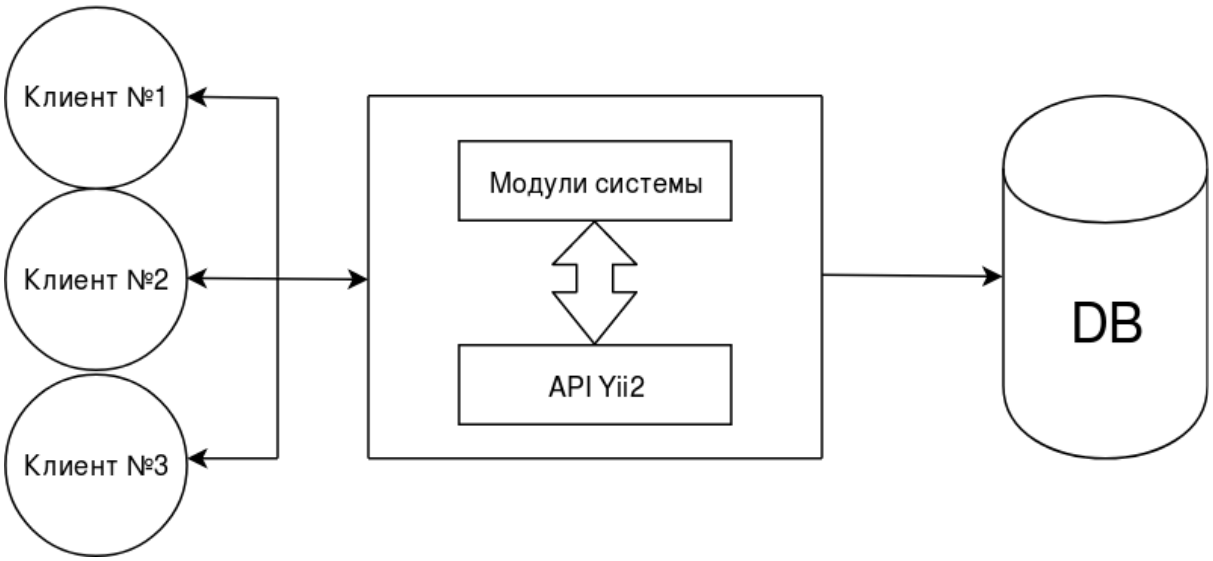
1. Регистрация
2. Авторизация
3. Смена роли пользователя
4. Смена пароля пользователя
5. CRUD функционал для сущности “ветка навыков”
6. CRUD функционал для сущности “навык”
7. CRUD функционал для сущности “набор навыков”

Под CRUD функционалом понимается функционал создания (Create), просмотра (Read), редактирования (Update) и удаления (Delete) сущности.

В аттестационном тестировании принимают участие все вышеперечисленные функции.

Стратегия тестирования

Структура объекта тестирования



Система разделена на следующие модули.

1. User - модуль для работы с пользовательскими аккаунтами
2. Admin - модуль для администрирования системы
3. Builder - модуль для работы с билдами
4. Skill - модуль для работы с навыками
5. Branch - модуль для работы с ветками

Тестирование будет проводиться для всех вышеописанных модулей системы, кроме Admin. Методы, которые будут подвергнуты тестированию, указаны ниже:

Модуль	**Тестируемый метод**	**Вид тестирования**
1. User	setRole(string $role)	Блочное
	validatePassword(string $password)	Блочное
	resetPassword(string $password)	Блочное
	login()	Интеграционное
2. Admin	-	-
3. Builder	createSkillBarData()	Блочное
	validateStats()	Блочное
	validateSkills(array $ids)	Интеграционное
	createBuild()	Интеграционное
4. Skill	findById(int $id)	Интеграционное
5. Branch	findById(int $id)	Интеграционное

Основные положения процедуры проведения тестирования

1. Тест считается успешно пройденным, если ожидаемый и фактический результат теста совпадают.
2. Этап тестирования считается завершенным в том случае, когда все тесты текущего этапа успешно пройдены.
3. Переход к следующему этапу тестирования может быть совершен только в том случае, если предыдущий этап тестирования завершен.
4. Процедура проведения тестирования может быть остановлена только в случае ошибки, блокирующей выполнение тестов.
5. Возобновление процедуры проведения тестирования происходит после исправления блокирующей ошибки, которая помешала предыдущему выполнению.

Инструменты тестирования

Для проведения блочного и интеграционного тестирования будут использованы следующие инструменты.

1. PHP Unit
2. Codeception

Блочное тестирование

Первый вид тестирования, которому будет подвержена система, будет блочное тестирование. Данный вид тестирования будет применен к некоторым функциям из модулей builder, user, skill, branch. Список функций для тестирования:

1. Метод setRole(string $role) модели user модуля user.

Проверка установки новой роли для пользователя.

2. Метод validatePassword(string $password) модели user модуля user.

Тест проверяет работу функции валидации пароля.

3. Метод createSkillBarData() класса BuilderService модуля builder.

Тест проверяет работоспособность метода, который создает json строку из атрибутов формы - положительных чисел.

4. Метод validateStats() класса BuilderForm модуля builder.

Проверка данных формы: свойства класса BuildForm $magicka, $strength и $stamina должны быть не более 64.

5. Метод resetPassword(string $password) класса User модуля user.

Тест проверяет работу функции смены пароля.

Интеграционное тестирование

Второй этап - интеграционное тестирование. Интеграционный тест проверяет работу взаимодействия модулей системы. Интеграционному тестированию будут подвергнуты методы, которые в процессе своего выполнения обращаются к методам из других модулей системы, которые в свою очередь уже протестированы на этапе блочного тестирования. Тестироваться будут следующие методы:

1. Метод validateSkills(array $ids) класса BuilderService модуля builder.

Метод проверяет навыки. Возвращает TRUE, если навыки в массиве принадлежат к веткам не более, чем одного класса. Возвращает FALSE, если в массиве есть хотя два ID навыков из веток, принадлежащих разным классам. Взаимодействует с методом findById(int $id) класса BranchRepository модуля branch и методом findById(int $id) класса SkillRepository модуля skill. Метод проходит по массиву входных данных, по ID получает из БД модель Skill. По branch_id получает из БД модель Branch и определяет принадлежность навыка к классу.

2. Метод createBuild() класса BuilderService модуля builder.

Тест проверяет работу создания билда. После заполнения и валидации формы вызывается метод createBuild(). Суть работы метода состоит в том, чтобы проверить полученные данные и, в случае корректности, сохранить билд. Взаимодействует с модулями build, skill. Также взаимодействует с модулем user, проверяя является ли гостем текущий пользователь. Если текущий пользователь авторизован, то создается дополнительная запись в БД в таблице “многие-ко-многим” user_build.

3. Метод login() класса LoginForm модуля user.

Авторизация пользователя в системе. Метод включает в себя валидацию формы входа и работу с БД. Метод взаимодействует с модулем Yii2 для работы с БД.

4. Метод findById(int $id) класса BranchRepository модуля branch.

Тест проверяет поиск записи в таблице Branch по ID. Метод взаимодействует с модулем Yii2 для работы с БД.

5. Метод findById(int $id) класса SkillRepository модуля branch.

Тест проверяет поиск записи в таблице Skill по ID. Метод взаимодействует с модулем Yii2 для работы с БД.

Нагрузочное тестирование

Для нагрузочного тестирования будем использовать утилиту Apache Benc. С количеством запросов страницы = 10000 и количеством конкурентных запросов = 50, с протоколом = TLS1.2, что дает максимальную нагрузку на сайт.

Аттестационное тестирование

Аттестационное тестирование проводится для всей системы, что подразумевает выполнение действий в пользовательском интерфейсе. Будут проверяться все функциональные требования. Для этого нам понадобится WEB-браузер.

План блочного тестирования

Метод resetPassword(string $password) класса User модуля user.

Тест №	Входные данные	Тип теста	Ожидаемый результат
Б1.1	Строка с паролем	Позитивный	Смена пароля пользователя
Б1.2	NULL	Негативный	Ошибка: resetPassword(string $id) expects parameter 1 to be string, null given in

Метод validateStats() класса BuildForm модуля builder.

Тест №	Входные данные	Тип теста	Ожидаемый результат
Б2.1	Сумма значений меньше 64	Позитивный	TRUE
Б2.2	Сумма значений больше 64	Негативный	FALSE
Б2.3	Один из атрибутов равен NULL	Негативный	FALSE
Б2.4	Один из атрибутов равен отрицательному числу	Негативный	FALSE

Метод createSkillBarData() класса BuilderService модуля builder.

Тест №	Входные данные	Тип теста	Ожидаемый результат

Б3.1	Атрибуты - массив положительных чисел	Позитивный	TRUE
Б3.2	Один или несколько чисел в массиве отрицательны	Негативный	FALSE
Б3.3	Массив состоит из количества символов больше шести (максимальное количестве символов в баре)	Негативный	FALSE

Метод setRole(string $role) модели user модуля User.

Тест №	Входные данные	Тип теста	Ожидаемый результат
Б4.1	‘admin’	Позитивный	TRUE
Б4.2	NULL	Негативный	Ошибка: setRole(string $role) expects parameter 1 to be string, null given in
Б4.3	Строка с несуществующей ролью	Негативный	FALSE

Метод validatePassword(string $password) модели user модуля User.

Тест №	Входные данные	Тип теста	Ожидаемый результат
Б5.1	Строка с паролем	Позитивный	TRUE
Б5.2	NULL	Негативный	Ошибка: validatePassword(string $password) expects parameter 1 to be string, null given in

План интеграционного тестирования

Метод findById(int $id) класса BranchRepository модуля branch.

Тест №	Входные данные	Тип теста	Ожидаемый результат
И1.1	ID из таблицы branch	Позитивный	Функция возвращает объект модели Branch
И1.2	NULL	Негативный	NULL
И1.3	Число, не равное, ни одному	Позитивный	NULL

	из ID таблицы branch		

Метод findById(int $id) класса SkillRepository модуля branch.

Тест №	Входные данные	Тип теста	Ожидаемый результат
И2.1	ID из таблицы skill	Позитивный	Функция возвращает объект модели Skill
И2.2	NULL	Негативный	NULL
И2.3	Число, не равное, ни одному из ID таблицы skill	Позитивный	NULL

Метод validateSkills(array $ids) класса BuilderService модуля builder.

Тест №	Входные данные	Тип теста	Ожидаемый результат
И3.1	Массив с существующими ID из таблицы skill	Позитивный	TRUE
И3.2	Массив с несуществующими ID из таблицы skill	Негативный	FALSE
И3.3	Пустой массив	Негативный	FALSE
И3.4	NULL	Негативный	Ошибка: isSkillAvailable(int $id, string $class) expects parameter 1 to be array, null given in

Метод createBuild() класса BuilderService модуля builder.

Тест №	Входные данные	Тип теста	Ожидаемый результат
И4.1	Данные формы указаны верно	Позитивный	TRUE
И4.2	Данные формы указаны неверно	Негативный	FALSE

Метод login() класса LoginForm модуля user.

Тест №	Входные данные	Тип теста	Ожидаемый результат
И5.1	Пользователь является гостем, данные формы логина соответствуют правилам валидации	Позитивный	TRUE
И5.2	Пользователь является гостем, данные формы логина НЕ соответствуют правилам валидации	Негативный	FALSE
И5.3	Пользователь уже авторизован, данные формы логина соответствуют правилам валидации	Позитивный	TRUE
И5.4	Пользователь уже авторизован, данные формы логина НЕ соответствуют правилам валидации	Негативный	FALSE

План нагрузочного тестирования

Выполняем команду:
ab -c 50 -n 10000 -f TLS1.2 -H "Accept-Encoding: gzip,deflate" <URL САЙТА>,
где ключи:
-c - количество конкурентных запросов.
-n - количество запросов страницы
-f - Указать протокол. (SSL3, TLS1, TLS1.1, TLS1.2 или ALL)
-H - добавить произвольную строку заголовка

План аттестационного тестирования

Проверка авторизации пользователя в системе + Выход из системы.

Тест №	Входные данные	Тип теста	Ожидаемый результат
А1.1	На странице авторизации пользователя вводится правильная пара логин/пароль и нажимается кнопка “Войти”	Позитивный	Появляется сообщение об успешной авторизации. Пользователь перенаправляется на страницу ESO Builder

A1.2	На странице авторизации пользователя вводится неправильная пара логин/пароль и нажимается кнопка “Войти”	Негативный	Появляется сообщение об ошибке авторизации
A1.3	На странице авторизации пользователя вводятся пустые значения логин/пароль и нажимается кнопка “Войти”	Негативный	Появляется сообщение об ошибке валидации
A1.4	На странице нажимается кнопка “Выйти”	Позитивный	Пользователь выходит из системы и перенаправляется на главную страницу

Проверка регистрации пользователя в системе.

Тест №	Входные данные	Тип теста	Ожидаемый результат
A2.1	На странице регистрации пользователя вводится корректная пара логин/пароль и нажимается кнопка “Зарегистрироваться”	Позитивный	Появляется сообщение об успешной регистрации. Пользователь авторизуется в системе и перенаправляется на главную страницу
A2.2	На странице регистрации пользователя вводится пара логин/пароль, которая уже существует и нажимается кнопка “Зарегистрироваться”	Негативный	Появляется сообщение об ошибке валидации
A2.3	На странице регистрации пользователя вводятся пустые значения логин/пароль и нажимается кнопка “Зарегистрироваться”	Негативный	Появляется сообщение об ошибке валидации

Проверка работы на главной странице системы (Просмотр страницы ESO Builder и создание билда).

Тест №	Входные данные	Тип теста	Ожидаемый результат
А3.1	Обычный пользователь перешел на страницу ESO Builder	Позитивный	Пользователь видит страницу ESO Builder приложения. Пользователь видит только кнопку “Создать билд” Пользователю доступны ссылки перехода на другие страницы: •Создать билд – создать новый билд, переход на страницу создания билда
А3.2	На странице ESO Builder нажимается кнопка “Создать билд”	Позитивный	Пользователь переходит на страницу создания билда

Аттестационное тестирование функций администратора (Ветки).

Тест №	Входные данные	Тип теста	Ожидаемый результат
А4.1	Администратор перешел на страницу Ветки	Позитивный	Администратор видит

			страницу Ветки приложения. На ней представлены 10 последних добавленных веток. Пользователь видит только название веток. Пользователю доступны ссылки перехода на другие страницы: •Плюс – создать новую ветку, переход на страницу создания новой ветки; •Карандаш – редактирование ветки из списка, переход на страницу редактирования ветки; •Корзина - удаление ветки из списка; •Поисковая строка - поиск ветки по названию; •Навыки - переход на страницу Навыки; •Ветки - обновление страницы

			Ветки.
А4.2	На странице Ветки в поисковом поле вводится значение, которое есть в списке	Позитивный	Отображение искомой строки
А4.3	На странице Ветки в поисковом поле вводится значение, которого нет в списке	Негативный	Появляется сообщение об ошибке
А4.4	На странице Ветки на выбранной строке нажимается кнопка “Корзина”	Позитивный	Выбранная ветка удаляется, список обновляется
А4.5	На странице Ветки нажимается кнопка “Плюс”	Позитивный	Переход на страницу создания новой ветки
А4.6	На странице Ветки на выбранной строке нажимается кнопка “Карандаш”	Позитивный	Переход на страницу редактирования выбранной ветки
А4.7	На странице редактирования ветки правильно редактируются поля “Название” и “Родительская ветка”	Позитивный	Изменения сохраняются, список на странице Ветки обновляется
А4.8	На странице редактирования ветки неправильно редактируются поля “Название” и “Родительская ветка”	Негативный	Появляется сообщение об ошибке
А4.9	На странице создания ветки правильно заполняются формы “Название” и “Родительская ветка” и нажимается кнопка “Создать ветку”	Позитивный	Ветка успешно добавляется в список на странице Ветки
А4.10	На странице создания ветки	Негативный	Появляется

	неправильно заполняются поля “Название” и “Родительская ветка” и нажимается кнопка “Создать ветку”		сообщение об ошибке

Аттестационное тестирование функций администратора (Навыки).

Тест №	Входные данные	Тип теста	Ожидаемый результат
А5.1	Обычный пользователь перешел на страницу Навыки	Позитивный	Пользователь видит страницу Навыки приложения. На ней представлены 10 последних добавленных навыков. Пользователь видит название и описание каждого навыка. Пользователю доступны ссылки перехода на другие страницы: •Плюс – создать новый навык, переход на страницу создания нового навыка; •Карандаш – редактирование определенного навыка, переход на страницу

			редактирования наыка; •Корзина - удаление навыка из списка; •Поисковая строка - поиск навыка по названию; •Навыки - обновление страницы Навыки; •Ветки - переход на страницу Ветки.
A5.2	На странице Навыки в поисковом поле вводится значение, которое есть в списке	Позитивный	Отображение искомой строки
A5.3	На странице Навыки в поисковом поле вводится значение, которого нет в списке	Негативный	Появляется сообщение об ошибке
A5.4	На странице Навыки на выбранной строке нажимается кнопка “Корзина”	Позитивный	Выбранный навык удаляется, список обновляется
A5.5	На странице Навыки нажимается кнопка “Плюс”	Позитивный	Переход на страницу создания нового навыка
A5.6	На странице Навыки на выбранной строке нажимается кнопка “Карандаш”	Позитивный	Переход на страницу редактирования выбранного навыка
A5.7	На странице редактирования навыка правильно редактируются поля “Название”, “Тип”, “Родительский навык”, “Ветка”, “Время произнесения”,	Позитивный	Изменения сохраняются, список на странице

	“Цель”, “Стоимость”, “Дальность применения”, “Описание”, “Длительность”, “Морф эффект”, “Требуется очков в ветке для разблокировки”		Навыки обновляется
А5.8	На странице редактирования ветки неправильно редактируется хотя бы одно из полей “Название”, “Тип”, “Родительский навык”, “Ветка”, “Время произнесения”, “Цель”, “Стоимость”, “Дальность применения”, “Описание”, “Длительность”, “Морф эффект”, “Требуется очков в ветке для разблокировки”	Негативный	Появляется сообщение об ошибке
А5.9	На странице создания навыка правильно заполняются поля “Название”, “Тип”, “Родительский навык”, “Ветка”, “Время произнесения”, “Цель”, “Стоимость”, “Дальность применения”, “Описание”, “Длительность”, “Морф эффект”, “Требуется очков в ветке для разблокировки” и нажимается кнопка “Создать навык”	Позитивный	Навык успешно добавляется в список на странице Навыки
А5.10	На странице создания навыка неправильно заполняется хотя бы одно поле “Название”, “Тип”, “Родительский навык”, “Ветка”, “Время произнесения”, “Цель”, “Стоимость”, “Дальность применения”, “Описание”, “Длительность”, “Морф эффект”, “Требуется очков в ветке для разблокировки” и нажимается кнопка “Создать навык”	Негативный	Появляется сообщение об ошибке

Аттестационное тестирование функций администратора (смена роли пользователя).

Тест №	Входные данные	Тип теста	Ожидаемый результат
А6.1	Администратор в консоли приложения набирает команду ./yii admin/role с аргументами командной строки <ID пользователя> и <роль>	Позитивный	Роль указанного пользователя меняется

A6.2	Администратор в консоли приложения набирает команду ./yii admin/role с некорректными аргументами командной строки (неправильный ID пользователя)	Негативный	Вывод сообщения ‘user not found’
A6.3	Администратор в консоли приложения набирает команду ./yii admin/role с некорректными аргументами командной строки (несуществующая в системе роль)	Негативный	Ошибка ‘role not found’
A6.4	Администратор в консоли приложения набирает команду ./yii admin/role, не указывая аргументы командной строки	Негативный	Error: Отсутствуют обязательные аргументы: userId, role

Аттестационное тестирование функций администратора (смена пароля пользователя).

Тест №	Входные данные	Тип теста	Ожидаемый результат
A7.1	Администратор в консоли приложения набирает команду ./yii admin/password с аргументами командной строки <ID пользователя> и <пароль>	Позитивный	Смена пароля пользователя
A7.2	Администратор в консоли приложения набирает команду ./yii admin/password с некорректными аргументами командной строки (неправильный ID пользователя)	Негативный	Вывод сообщения ‘user not found’
A7.3	Администратор в консоли приложения набирает команду ./yii admin/role, не указывая аргументы командной строки	Негативный	Error: Отсутствуют обязательные аргументы: userId, password

Реализация тестирования

Реализация блочного и интеграционного тестирования

После первого запуска блочных и интеграционных тестов были получены следующие результаты:

✔ BuildFormTest: Validate stats if stats greater (0.02s)
✔ BuildFormTest: Validate stats if stats less (0.00s)
✔ BuildFormTest: Validate stats with null (0.00s)
✔ BuildFormTest: Validate stats with negative number (0.00s)
✔ BuilderServiceTest: Create skill bar data correct (0.00s)
✔ BuilderServiceTest: Create skill bar data negative (0.00s)
✔ BuilderServiceTest: Create skill bar data incorrect (0.00s)
✔ BuilderServiceValidateSkillsTest: Validate correct skills (0.02s)
✔ BuilderServiceValidateSkillsTest: Validate incorrect skills (0.02s)
✔ BuilderServiceValidateSkillsTest: Validate empty skills (0.00s)
E BuilderServiceValidateSkillsTest: Validate null skills (0.00s)
✔ BuilderServiceValidateSkillsTest: Create build correct (0.00s)
✔ BuilderServiceValidateSkillsTest: Create build incorrect (0.00s)
✔ LoginFormTest: Login no user (0.01s)
✔ LoginFormTest: Login wrong password (0.51s)
✔ LoginFormTest: Login correct (0.57s)
✔ RepositoryTest: Find skill correct (0.03s)
✔ RepositoryTest: Find skill null (0.02s)
✔ RepositoryTest: Find not existing skill (0.02s)
✔ RepositoryTest: Find branch correct (0.02s)
✔ RepositoryTest: Find branch null (0.02s)
✔ RepositoryTest: Find not existing branch (0.02s)
✔ ResetPasswordTest: Reset password correct (0.50s)
✖ ResetPasswordTest: Reset password null (0.48s)
✔ SetRoleTest: Set role correct (0.03s)
E SetRoleTest: Set role wrong role (0.01s)
E SetRoleTest: Set role no role (0.00s)
E ValidatePasswordTest: Validate password null (0.00s)

ERRORS!
Tests: 29, Assertions: 30, Errors: 4, Failures: 1.

Используя информацию, полученную в ходе тестирования, стало понятно, что тесты *testValidateNullSkills, testSetRoleNoRole, testValidatePasswordNull* некорректны в своем применении, так как в реализации этих тестов мы пытаемся присвоить заранее неверное значение по типу в качестве аргумента. Если такое значение будет передано в эту функцию в ходе выполнения кода, то это не является ошибкой работы данной функции. *testSetRoleWrongRole* дал нам информацию о том, что при попытке указать несуществующую роль в системе, программа удаляет текущую роль пользователя. Ошибку необходимо исправить. *testResetPasswordNull* дал нам информацию о том, что в методе передаваемое значение не проверяется на NULL. Есть два пути решения: либо указать строгую типизацию параметров в методе и отказаться от теста, либо добавить проверку на NULL.

Результат тестирования после исправления ошибок

✔ BuildFormTest: Validate stats if stats greater (0.02s)
✔ BuildFormTest: Validate stats if stats less (0.00s)
✔ BuildFormTest: Validate stats with null (0.00s)
✔ BuildFormTest: Validate stats with negative number (0.00s)
✔ BuilderServiceTest: Create skill bar data correct (0.00s)
✔ BuilderServiceTest: Create skill bar data negative (0.00s)
✔ BuilderServiceTest: Create skill bar data incorrect (0.00s)
✔ BuilderServiceValidateSkillsTest: Validate correct skills (0.02s)
✔ BuilderServiceValidateSkillsTest: Validate incorrect skills (0.02s)
✔ BuilderServiceValidateSkillsTest: Validate empty skills (0.00s)
✔ BuilderServiceValidateSkillsTest: Validate null skills (0.00s)
✔ BuilderServiceValidateSkillsTest: Create build correct (0.00s)
✔ BuilderServiceValidateSkillsTest: Create build incorrect (0.00s)
✔ LoginFormTest: Login no user (0.02s)
✔ LoginFormTest: Login wrong password (0.50s)
✔ LoginFormTest: Login correct (0.58s)
✔ RepositoryTest: Find skill correct (0.03s)
✔ RepositoryTest: Find skill null (0.02s)
✔ RepositoryTest: Find not existing skill (0.01s)
✔ RepositoryTest: Find branch correct (0.02s)
✔ RepositoryTest: Find branch null (0.02s)
✔ RepositoryTest: Find not existing branch (0.02s)
✔ ResetPasswordTest: Reset password correct (0.50s)
✔ ResetPasswordTest: Reset password null (0.48s)
✔ SetRoleTest: Set role correct (0.03s)
✔ SetRoleTest: Set role wrong role (0.00s)

✔ ValidatePasswordTest: Validate password correct (0.54s)
✔ SetRoleTest: Set role no role (0.01s)
✔ ValidatePasswordTest: Validate password null (0.02s)
OK (29 tests, 34 assertions)

Пример реализации блочного тестирования

На примере теста SetRoleTest.php для метода SetRole()

```php
<?php

namespace common\tests\unit\models;

use Codeception\Test\Unit;
use common\modules\user\models\User;
use dektrium\rbac\components\DbManager;
use dektrium\rbac\components\ManagerInterface;
use Mockery\Mock;

/**
* Class SetRoleTest
*/
class SetRoleTest extends Unit
{
  /**
   * Тест установки роли с правильной ролью
   */
  public function testSetRoleCorrect()
  {
      // получение мок объекта (как и во всех блочных тестах)
      $user = $this->getUserMock();
      // мок еще одного объекта, который используется внутри функции setRole()
      \Yii::$container->set(ManagerInterface::class, \Mockery::mock(DbManager::class)->makePartial());
      // проверка работы функции
```

```
        verify($user->setRole(User::ROLE_ADMIN))->true();
    }

    /**
     * Тест установки роли с неверной ролью
     */
    public function testSetRoleWrongRole()
    {
        // получение мок объекта (как и во всех блочных тестах)
        $user = $this->getUserMock();
        // мок еще одного объекта, который используется внутри функции setRole()
        \Yii::$container->set(ManagerInterface::class, \Mockery::mock(DbManager::class)->makePartial());
        verify($user->setRole('not_existing_role'))->false();
    }

    /**
     * Тест установки роли с передачей null вместо роли
     */
    public function testSetRoleNoRole()
    {
        // получение мок объекта (как и во всех блочных тестах)
        $user = $this->getUserMock();
        // мок еще одного объекта, который используется внутри функции setRole()
        \Yii::$container->set(ManagerInterface::class, \Mockery::mock(DbManager::class)->makePartial());
        // проверка работы функции
        verify($user->setRole(null))->true();
    }

    /**
     * @return Mock|User
     */
    protected function getUserMock()
    {
```

```
        // мок объекта User
        // мок - создание объекта с такими же свойствами, полями и функциями, но
       // без реализации этих функций
        // для того, чтобы получить объект, который не взаимодействует с БД (здесь ActiveRecord)
        $user = \Mockery::mock(User::class)->makePartial();
        // Разрешаем метод save (всегда будет отдавать true). ничего не сохраняем в БД,
симуляция выполнения функции save
        $user->allows()->save(\Mockery::any())->andReturn(true);
        // симуляция метода attributes() (в классе ActiveRecord пытается получить из БД список
атрибутов модели)
        $user->allows()->attributes()->andReturn([
            'id',
        ]);
        //присвоение ID "виртуальному" пользователю (его нет в БД)
        $user->id = 1;
        /**
         * модель (ActiveRecord) - это строка из БД в виде класса PHP, то есть, если у нас есть
строка в таблице user с email = some@mail.ru то и в модели User будет такое свойство email с
таким же значением
         */
        return $user;
    }
}
```

Пример реализации интеграционного тестирования

На примере теста LoginFormTest.php

```
<?php
        namespace common\tests\unit\models;

        use Yii;
        use frontend\modules\user\forms\LoginForm;
        use common\fixtures\UserFixture;
        use Codeception\Test\Unit;
```

```php
/**
* Login form test
*/
class LoginFormTest extends Unit
{
  /**
   * Fixture создает в тестовой базе (после выполнения тестов, данные сразу удаляются) записи с данными указанными в файле по пути dataFile
   * создает запись в таблице User
   * это интеграционный тест, потому что взаимодействует с БД (помимо других модулей приложения)
   * @return array
   */
  public function _fixtures()
  {
     return [
        'user' => [
           'class' => UserFixture::class,
           'dataFile' => codecept_data_dir() . 'user.php'
        ]
     ];
  }

  /**
   * Тест на логин пользователя с указанием несуществующих данных
   */
  public function testLoginNoUser()
  {
     // создание модели формы авторизации
     $model = new LoginForm([
        'username' => 'not_existing_username',
        'password' => 'not_existing_password',
     ]);

     // попытка логина и проверка как отработала функция
```

```
        expect('model should not login user', $model->login())->false();
        // Проверка на то, является ли текущий юзер гостем или нет (предыдущая функция пытается логинить пользователя)
        expect('user should not be logged in', Yii::$app->user->isGuest)->true();
    }

    /**
     * Тест на логин пользователя с указанием верного логина, но неправильного пароля
     */
    public function testLoginWrongPassword()
    {
        // создание модели формы авторизации
        $model = new LoginForm([
            'username' => 'user',
            'password' => 'wrong_password',
        ]);

        // попытка логина и проверка как отработала функция
        expect('model should not login user', $model->login())->false();
        // проверка, есть ли ошибки в пароле
        expect('error message should be set', $model->errors)->hasKey('password');
        // Проверка на то, является ли текущий юзер гостем или нет
        expect('user should not be logged in', Yii::$app->user->isGuest)->true();
    }

    /**
     * Тест на логин пользователя с указанием правильных данных
     */
    public function testLoginCorrect()
    {
        // создание модели формы авторизации
        $model = new LoginForm([
            'username' => 'user',
            'password' => 'password_0',
        ]);
```

```
        // попытка логина и проверка как отработала функция
        expect('model should login user', $model->login())->true();
        // проверка, есть ли ошибки в пароле
        expect('error message should not be set', $model->errors)->hasntKey('password');
        // Проверка на то, является ли текущий юзер гостем или нет (предыдущая функция пытается логинить пользователя)
        expect('user should be logged in', Yii::$app->user->isGuest)->false();
    }
}
```

Пример заглушки

```
/**
     * @return Mock|User
     */
    protected function getUserMock()
    {
        // заглушка для объекта User
        $user = \Mockery::mock(User::class)->makePartial();
        // позволяем вызывать метод save(), который будет всегда возвращать true
        $user->allows()->save(\Mockery::any())->andReturn(true);
        // позволяем вызывать метод updateAttributes(), который будет всегда возвращать true
        $user->allows()->updateAttributes(\Mockery::any())->andReturn(true);
        // позволяем вызывать метод attributes(), который будет всегда возвращать массив [id, passwod_hash]
        $user->allows()->attributes()->andReturn([
            'id',
            'password_hash',
        ]);
        $user->id = 1;

        return $user;
    }
```

Реализация нагрузочного тестирования

В Apache Bench выполняем команду
ab -c 50 -n 10000 -f TLS1.2 -H "Accept-Encoding: gzip,deflate" <URL САЙТА>

Получаем данные.

Benchmarking eso.test (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests

Server Software: nginx/1.4.6
Server Hostname: eso.test
Server Port: 80

Document Path: /
Document Length: 15827 bytes

Concurrency Level: 50
Time taken for tests: 216.100 seconds
Complete requests: 10000
Failed requests: 0
Total transferred: 163730000 bytes
HTML transferred: 158270000 bytes
Requests per second: 46.27 [#/sec] (mean)
Time per request: 1080.498 [ms] (mean)
Time per request: 21.610 [ms] (mean, across all concurrent requests)
Transfer rate: 739.90 [Kbytes/sec] received

Connection Times (ms)

	min	mean[+/-sd]	median	max
Connect:	0	0 0.0	0	1
Processing:	74	1078 387.5	996	4174
Waiting:	31	999 367.2	923	4062

Total: 75 1078 387.5 996 4174

Percentage of the requests served within a certain time (ms)
50% 996
66% 1045
75% 1084
80% 1115
90% 1287
95% 1466
98% 2072
99% 3748
100% 4174 (longest request)

Согласно полученным данным, наш сайт может обработать 46 одновременных пользователей в секунду.
Среднее время на выполнение группы параллельных запросов - 1080.498 миллисекунд
Среднее время на выполнение одного запроса - 21.610 ms

Стоит понимать, что это очень грубые и неточные данные, так как тестирование проводилось на локальном сервере и было обусловлено лишь работой сервера и производительностью интерпретатора (учитывая, конечно же, скрипты расположенные на главной странице сайта)

Реализация аттестационного тестирования

Аттестационное тестирование функций пользователя

Тест №	А1.1-П
Тестируемая функция	Вход в систему (авторизация)
Описание	На странице авторизации пользователя вводится правильная пара логин /пароль и нажимается кнопка “Войти”
Входные значения	логин=”**user1**”, пароль=”**password1**”
Ожидаемый результат	Появляется сообщение об успешной авторизации. Пользователь перенаправляется на главную страницу
Фактический результат	Появляется сообщение об успешной авторизации. Пользователь перенаправляется

	на главную страницу
Результат теста	Тест пройден без ошибок

Тест №	A1.2-H
Тестируемая функция	Вход в систему (авторизация)
Описание	На странице авторизации пользователя вводится неправильная пара логин/пароль и нажимается кнопка “Войти”
Входные значения	логин=”**pfRjsdf55**” , пароль=”**yrt8etUo**”
Ожидаемый результат	Появляется сообщение об ошибке валидации
Фактический результат	Появляется сообщение об ошибке валидации
Результат теста	Тест пройден без ошибок

Тест №	A1.3-H
Тестируемая функция	Вход в систему (авторизация)
Описание	На странице авторизации пользователя вводятся пустые значения логин/пароль и нажимается кнопка “Войти”
Входные данные	логин=”” , пароль=””
Ожидаемый результат	Появляется сообщение об ошибке валидации
Фактический результат	Появляется сообщение об ошибке валидации
Результат теста	Тест пройден без ошибок

Тест №	A1.4-H
Тестируемая функция	Выход из системы
Описание	На странице нажимается кнопка “Выйти”
Входные данные	нажимается кнопка “Выйти”
Ожидаемый результат	Пользователь выходит из системы и перенаправляется на главную страницу

Фактический результат	Пользователь выходит из системы и перенаправляется на главную страницу
Результат теста	Тест пройден без ошибок

Тест №	А2.1-П
Тестируемая функция	Регистрация
Описание	На странице регистрации пользователя вводится пара логин/пароль и нажимается кнопка “Зарегистрироваться”
Входные данные	логин=”**user2**”, пароль=”**password2**”
Ожидаемый результат	Появляется сообщение об успешной регистрации. Пользователь авторизуется в системе и перенаправляется на главную страницу
Фактический результат	Появляется сообщение об успешной регистрации. Пользователь авторизуется в системе и перенаправляется на главную страницу
Результат теста	Тест пройден без ошибок

Тест №	А2.2-Н
Тестируемая функция	Регистрация
Описание	На странице регистрации пользователя вводится пара логин/пароль, которая уже существует и нажимается кнопка “Зарегистрироваться”
Входные данные	логин=”**user1**”, пароль=”**password2**”
Ожидаемый результат	Появляется сообщение об ошибке валидации
Фактический результат	Появляется сообщение об ошибке валидации
Результат теста	Тест пройден без ошибок

Тест №	А2.3-Н

Тестируемая функция	Регистрация
Описание	На странице регистрации пользователя вводятся пустые значения логин/пароль и нажимается кнопка “Зарегистрироваться”
Входные данные	логин=””, пароль=””
Ожидаемый результат	Появляется сообщение об ошибке валидации
Фактический результат	Появляется сообщение об ошибке валидации
Результат теста	Тест пройден без ошибок

Тест №	А3.1-П
Тестируемая функция	Просмотр главной страницы
Описание	Авторизованный пользователь перешел на главную страницу
Входные данные	переход на главную страницу
Ожидаемый результат	Пользователь видит главную страницу приложения. Пользователь видит только кнопку “Создать билд” Пользователю доступны ссылки перехода на другие страницы: •Создать билд – создать новый билд, переход на страницу создания билда
Фактический результат	Пользователь видит главную страницу приложения. Пользователь видит только кнопку “Создать билд” Пользователю доступны ссылки перехода на другие страницы: •Создать билд – создать новый билд, переход на страницу создания билда
Результат теста	Тест пройден без ошибок

Тест №	А3.2-П
Тестируемая функция	Создать билд
Описание	На главной странице нажимается кнопка “Создать билд”

Входные данные	нажимается кнопка “Создать билд”
Ожидаемый результат	Пользователь переходит на страницу создания билда
Фактический результат	Пользователь переходит на страницу создания билда
Результат теста	Тест пройден без ошибок

Аттестационное тестирование функций администратора

Тест №	А4.1-П
Тестируемая функция	Просмотр страницы Ветки
Описание	Администратор перешел на страницу Ветки
Входные данные	переход на страницу Ветки
Ожидаемый результат	Администратор видит страницу Ветки приложения. На ней представлена таблица веток. Пользователь видит только название веток. Пользователю доступны ссылки перехода на другие страницы: •Плюс – создать новую ветку, переход на страницу создания ветки; •Карандаш – редактирование ветки из списка, переход на страницу редактирования ветки; •Корзина - удаление ветки из списка; •Поисковая строка - поиск ветки по названию; •Навыки - переход на страницу Навыки; •Ветки - обновление страницы Ветки.
Фактический результат	Администратор видит страницу Ветки приложения. На ней представлена таблица веток. Пользователь видит только название веток. Пользователю доступны ссылки перехода на другие страницы: •Плюс – создать новую ветку, переход на страницу создания ветки; •Карандаш – редактирование ветки из списка, переход на страницу редактирования ветки; •Корзина - удаление ветки из списка; •Поисковая строка - поиск ветки по названию; •Навыки - переход на страницу Навыки; •Ветки - обновление страницы Ветки.
Результат теста	Тест пройден без ошибок

Тест №	А4.2-П
Тестируемая функция	Поиск на странице Ветки
Описание	На странице Ветки в поисковом поле вводится значение, которое есть в списке
Входные данные	вводится “**Тестовая ветка**”
Ожидаемый результат	Отображение искомой строки “Тестовая ветка”
Фактический результат	Отображение искомой строки “Тестовая ветка”
Результат теста	Тест пройден без ошибок

Тест №	А4.3-Н
Тестируемая функция	Поиск на странице Ветки
Описание	На странице Ветки в поисковом поле вводится значение, которого нет в списке
Входные данные	вводится “**DFRygds7634**”
Ожидаемый результат	Таблица пуста
Фактический результат	Таблица пуста
Результат теста	Тест пройден без ошибок

Тест №	А4.4-П
Тестируемая функция	Удаление на странице Ветки
Описание	На странице Ветки на выбранной строке нажимается кнопка “Корзина”
Входные данные	на строке “Тестовая ветка” нажимается кнопка “Корзина”
Ожидаемый результат	“Тестовая ветка” удаляется, список обновляется
Фактический результат	“Тестовая ветка” удаляется, список обновляется
Результат теста	Тест пройден без ошибок

Тест №	А4.5-П
Тестируемая функция	Добавление на странице Ветки
Описание	На странице Ветки нажимается кнопка “Плюс”
Входные данные	нажимается кнопка “Плюс”
Ожидаемый результат	Переход на страницу создания новой ветки
Фактический результат	Переход на страницу создания новой ветки
Результат теста	Тест пройден без ошибок

Тест №	А4.6-П
Тестируемая функция	Редактирование на странице Ветки
Описание	На странице Ветки на выбранной строке нажимается кнопка “Карандаш”
Входные данные	на “Тестовой ветке” нажимается кнопка “Карандаш”
Ожидаемый результат	Переход на страницу редактирования выбранной ветки
Фактический результат	Переход на страницу редактирования выбранной ветки
Результат теста	Тест пройден без ошибок

Тест №	А4.7-П
Тестируемая функция	Редактирование ветки
Описание	На странице редактирования ветки правильно редактируются поля “Название” и “Родительская ветка”
Входные данные	Название=”**Нетестовая ветка**” , Родительская ветка=”**Родительская ветка**”
Ожидаемый результат	Изменения сохраняются, список на странице Ветки обновляется
Фактический результат	Изменения сохраняются, список на странице

	Ветки обновляется
Результат теста	Тест пройден без ошибок

Тест №	А4.8-Н
Тестируемая функция	Редактирование ветки
Описание	На странице редактирования ветки неправильно редактируются поля “Название” и “Родительская ветка”
Входные данные	Название=”**123456**” , Родительская ветка=””
Ожидаемый результат	Появляется сообщение об ошибке валидации
Фактический результат	Появляется сообщение об ошибке валидации
Результат теста	Тест пройден без ошибок

Тест №	А4.9-П
Тестируемая функция	Создание ветки
Описание	На странице создания ветки правильно заполняются формы “Название” и “Родительская ветка” и нажимается кнопка “Создать ветку”
Входные данные	Название=”**Дочерняя ветка**” , Родительская ветка=”**Тестовая ветка**”
Ожидаемый результат	“Дочерняя ветка” успешно добавляется в список на странице Ветки
Фактический результат	“Дочерняя ветка” успешно добавляется в список на странице Ветки
Результат теста	Тест пройден без ошибок

Тест №	А4.10-Н
Тестируемая функция	Создание ветки
Описание	На странице создания ветки неправильно

	заполняются поля “Название” и “Родительская ветка” и нажимается кнопка “Создать ветку”
Входные данные	Название=”**123456**” , Родительская ветка=” ”
Ожидаемый результат	Появляется сообщение об ошибке валидации
Фактический результат	Появляется сообщение об ошибке валидации
Результат теста	Тест пройден без ошибок

Тест №	А5.1-П
Тестируемая функция	Просмотр страницы Навыки
Описание	Администратор перешел на страницу Навыки
Входные данные	переход на страницу Навыки
Ожидаемый результат	Администратор видит страницу Навыки приложения. На ней представлена таблица навыков. Пользователь видит название и описание каждого навыка. Пользователю доступны ссылки перехода на другие страницы: •Плюс – создать новый навык, переход на страницу создания навыка; •Карандаш – редактирование определенного навыка, переход на страницу редактирования навыка; •Корзина - удаление навыка из списка; •Поисковая строка - поиск навыка по названию; •Навыки - обновление страницы Навыки; •Ветки - переход на страницу Ветки.
Фактический результат	Администратор видит страницу Навыки приложения. На ней представлена таблица навыков. Пользователь видит название и описание каждого навыка. Пользователю доступны ссылки перехода на другие страницы: •Плюс – создать новый навык, переход на страницу создания навыка; •Карандаш – редактирование определенного навыка, переход на страницу редактирования навыка; •Корзина - удаление навыка из списка; •Поисковая строка - поиск навыка по названию; •Навыки - обновление страницы Навыки;

	•Ветки - переход на страницу Ветки.
Результат теста	Тест пройден без ошибок

Тест №	A5.2-П
Тестируемая функция	Поиск на странице Навыки
Описание	На странице Навыки в поисковом поле вводится значение, которое есть в списке
Входные данные	Навык = “**Тестовый навык**”
Ожидаемый результат	Отображение искомой строки “Тестовый навык”
Фактический результат	Отображение искомой строки “Тестовый навык”
Результат теста	Тест пройден без ошибок

Тест №	A5.3-Н
Тестируемая функция	Поиск на странице Навыки
Описание	На странице Навыки в поисковом поле вводится значение, которого нет в списке
Входные данные	Навык = “**апывап453ав**”
Ожидаемый результат	Таблица пуста
Фактический результат	Таблица пуста
Результат теста	Тест пройден без ошибок

Тест №	A5.4-П
Тестируемая функция	Удаление на странице Навыки
Описание	На странице Навыки на выбранной строке нажимается кнопка “Корзина”
Входные данные	на строке “Тестовый навык” нажимается кнопка “Корзина”
Ожидаемый результат	“Тестовый навык” удаляется, список обновляется

Фактический результат	“Тестовый навык” удаляется, список обновляется
Результат теста	Тест пройден без ошибок

Тест №	А5.5-П
Тестируемая функция	Добавление на странице Навыки
Описание	На странице Навыки нажимается кнопка “Плюс”
Входные данные	нажимается кнопка “Плюс”
Ожидаемый результат	Переход на страницу создания нового навыка
Фактический результат	Переход на страницу создания нового навыка
Результат теста	Тест пройден без ошибок

Тест №	А5.6-П
Тестируемая функция	Редактирование на странице Навыки
Описание	На странице Навыки на выбранной строке нажимается кнопка “Карандаш”
Входные данные	На странице Навыки на строке “Тестовый навык” нажимается кнопка “Карандаш”
Ожидаемый результат	Переход на страницу редактирования “Тестовый навык”
Фактический результат	Переход на страницу редактирования “Тестовый навык”
Результат теста	Тест пройден без ошибок

Тест №	А5.7-П
Тестируемая функция	Редактирование навыка
Описание	На странице редактирования навыка правильно редактируются поля “Название” (строка, обязательное значение), “Тип” (строка, необязательное значение), “Родительский навык” (выбор из списка),

	“Ветка” (выбор из списка), “Время произнесения” (число с плавающей точкой, необязательное значение), “Цель” (строка, необязательное значение), “Стоимость” (целое число, необязательное значение), “Дальность применения” (целое число, необязательное значение), “Описание” (строка, необязательное значение), “Длительность” (целое число, необязательное значение), “Морф эффект” (строка, необязательное значение), “Требуется очков в ветке для разблокировки” (целое число, необязательное значение)
Входные данные	Название=”**Нетестовый навык**” Тип=”**тип1**” Родительский навык=”**Родительский навык**” Ветка=”**Тестовая ветка**” Время произнесения=”**1**” Цель=”**На врага**” Стоимость=”**5000**” Дальность применения=”**20**” Описание=”**Накладывает проклятие..**” Длительность=”**5**” Морф эффект=”” Требуется очков в ветке для разблокировки=”**5**”
Ожидаемый результат	Изменения сохраняются, список на странице Навыки обновляется
Фактический результат	Изменения сохраняются, список на странице Навыки обновляется
Результат теста	Тест пройден без ошибок

Тест №	A5.8-H
Тестируемая функция	Редактирование навыка
Описание	На странице редактирования ветки неправильно редактируется хотя бы одно из полей “Название”, “Тип”, “Родительский навык”, “Ветка”, “Время произнесения”, “Цель”, “Стоимость”, “Дальность применения”, “Описание”, “Длительность”, “Морф эффект”, “Требуется очков в ветке для разблокировки”
Входные данные	Название=”**123456**” Тип=””

	Родительский навык=”” Ветка=”**Тестовая ветка**” Время произнесения=”**22:00**” Цель=”” Стоимость=”**5 000 000 000 000 000**” Дальность применения=”**5 000 000 000 000 000**” Описание=”**Увеличение скорости**” Длительность=”-**5**” Морф эффект=”” Требуется очков в ветке для разблокировки=””
Ожидаемый результат	Появляется сообщение об ошибке валидации
Фактический результат	Появляется сообщение об ошибке валидации
Результат теста	Тест пройден без ошибок

Тест №	А5.9-П
Тестируемая функция	Создание навыка
Описание	На странице создания навыка правильно заполняются поля “Название”, “Тип”, “Родительский навык”, “Ветка”, “Время произнесения”, “Цель”, “Стоимость”, “Дальность применения”, “Описание”, “Длительность”, “Морф эффект”, “Требуется очков в ветке для разблокировки” и нажимается кнопка “Создать навык”
Входные данные	Название=”**Нетестовый навык**” Тип=”**тип1**” Родительский навык=”**Родительский навык**” Ветка=”**Тестовая ветка**” Время произнесения=”**1**” Цель=”**На врага**” Стоимость=”**5000**” Дальность применения=”**20**” Описание=”**Накладывает проклятие..**” Длительность=”**5**” Морф эффект=”” Требуется очков в ветке для разблокировки=”**5**”
Ожидаемый результат	“Новый навык” успешно добавляется в список на странице Навыки
Фактический результат	“Новый навык” успешно добавляется в список на странице Навыки

Результат теста	Тест пройден без ошибок

Тест №	А5.10-Н
Тестируемая функция	Создание навыка
Описание	На странице создания навыка неправильно заполняется хотя бы одно поле “Название”, “Тип”, “Родительский навык”, “Ветка”, “Время произнесения”, “Цель”, “Стоимость”, “Дальность применения”, “Описание”, “Длительность”, “Морф эффект”, “Требуется очков в ветке для разблокировки” и нажимается кнопка “Создать навык”
Входные данные	Название=”**123456**” Тип=”” Родительский навык=”” Ветка=”**Тестовая ветка**” Время произнесения=”**22:00**” Цель=”” Стоимость=”**5 000 000 000 000 000**” Дальность применения=”**5 000 000 000 000 000**” Описание=”**Увеличение скорости**” Длительность=”-**5**” Морф эффект=”” Требуется очков в ветке для разблокировки=””
Ожидаемый результат	Появляется сообщение об ошибке валидации
Фактический результат	Появляется сообщение об ошибке валидации
Результат теста	Тест пройден без ошибок

Тест №	А6.1-П
Тестируемая функция	Смена роли пользователя
Описание	Администратор в консоли приложения набирает команду ./yii admin/role с аргументами командной строки <ID пользователя> и <роль>
Входные данные	ID пользователя= “1” роль = “admin”
Ожидаемый результат	Команда завершает свою работу. Роль

	пользователя с ID 1 меняется на роль admin (проверяется в БД в таблице auth_assignment)
Фактический результат	Команда завершает свою работу. Роль пользователя с ID 1 меняется на роль admin (проверяется в БД в таблице auth_assignment)
Результат теста	Тест пройден без ошибок

Тест №	A6.2-H
Тестируемая функция	Смена роли пользователя
Описание	Администратор в консоли приложения набирает команду ./yii admin/role с некорректными аргументами командной строки (неправильный ID пользователя)
Входные данные	ID пользователя= “**dfgsh334wgvW3TWBW**” роль = “admin”
Ожидаемый результат	Вывод сообщения ‘user not found’
Фактический результат	Вывод сообщения ‘user not found’
Результат теста	Тест пройден без ошибок

Тест №	A6.3-H
Тестируемая функция	Смена роли пользователя
Описание	Администратор в консоли приложения набирает команду ./yii admin/role с некорректными аргументами командной строки (несуществующая в системе роль)
Входные данные	ID пользователя= “1” роль = “not_existing_role”
Ожидаемый результат	Ошибка ‘role not found’
Фактический результат	Команда завершает свою работу. В таблице auth_assignment пропадает запись о пользователе с ID 1
Результат теста	ОШИБКА. Поведение функции привело к тому, что пользователь теряет роль в системе (см.

	отчет №6)

Тест №	А6.4-Н
Тестируемая функция	Смена роли пользователя
Описание	Администратор в консоли приложения набирает команду ./yii admin/role, не указывая аргументы командной строки
Входные данные	ID пользователя= “” роль = “”
Ожидаемый результат	Error: Отсутствуют обязательные аргументы: userId, role
Фактический результат	Error: Отсутствуют обязательные аргументы: userId, role
Результат теста	Тест пройден без ошибок

Тест №	А7.1-П
Тестируемая функция	Смена пароля пользователя
Описание	Администратор в консоли приложения набирает команду ./yii admin/password с аргументами командной строки <ID пользователя> и <пароль>
Входные данные	ID пользователя= “**1**” пароль = “**QwErTy**”
Ожидаемый результат	Смена пароля у пользователя с ID 1 на QwErTy
Фактический результат	Смена пароля у пользователя с ID 1 на QwErTy
Результат теста	Тест пройден без ошибок

Тест №	А7.2-Н
Тестируемая функция	Смена пароля пользователя
Описание	Администратор в консоли приложения набирает команду ./yii admin/password с некорректными аргументами командной строки

	(неправильный ID пользователя)
Входные данные	ID пользователя= “**fhy35herhb34**”
Ожидаемый результат	Вывод сообщения ‘user not found’
Фактический результат	Вывод сообщения ‘user not found’
Результат теста	Тест пройден без ошибок

Тест №	A7.3-H
Тестируемая функция	Смена пароля пользователя
Описание	Администратор в консоли приложения набирает команду ./yii admin/role, не указывая аргументы командной строки
Входные данные	ID пользователя= “” пароль = “”
Ожидаемый результат	Error: Отсутствуют обязательные аргументы: userId, password
Фактический результат	Error: Отсутствуют обязательные аргументы: userId, password
Результат теста	Тест пройден без ошибок

Журнал тестирования

Тест № + тип	Дата	Тестируемый модуль с описанием	Входные данные	Кол-во запусков	Найденные ошибки	Результат теста
Б4.1-П	23.11.18	testSetRoleCorrect - *метод setRole() Тест установки роли с правильной ролью*	‘admin’	5	0	Пройден успешно
Б4.3-Н	23.11.18	testSetRoleWrongRole - *метод setRole() Тест установки роли с неверной ролью*	‘not_exitsting_role’	5	1	Тест провален (см. отчет № 1)
Б4.2-Н	23.11.18	testSetRoleNoRole - *метод setRole() Тест установки роли с передачей null вместо роли*	NULL	5	1	Тест провален (см.

						отчет № 2)
Б5.1-П	23.11.18	testValidatePasswordCorrect - *метод validatePassword() Тест валидации пароля с паролем - строкой*	‘password’	5	0	Пройден успешно
Б5.2-Н	23.11.18	testValidatePasswordNull - *метод validatePassword() Тест валидации пароля с паролем равным NULL*	NULL	5	1	Тест провален (см. отчет № 3)
Б1.1-П	23.11.18	testResetPasswordCorrect - *метод resetPassword() Тест сброса пароля с новым паролем - строкой*	‘new_password’	5	0	Пройден успешно
Б1.2-Н	23.11.18	testResetPasswordNull - *метод resetPassword() Тест сброса пароля с новым паролем - NULL*	NULL	5	1	Тест провален (см. отчет № 4) Failures
Б2.2-Н	23.11.18	testValidateStatsIfStatsGreate - *метод validateStats() Проверка корректности работы функции при параметрах, преувеличивающих допустимые значения*	64, 64, 64	5	0	Пройден успешно
Б2.1-П	23.11.18	testValidateStatsIfStatsLess - *метод validateStats() Проверка корректности работы функции при параметрах, находящихся в области допустимых значений*	0, 0, 0	5	0	Пройден успешно
Б2.3-Н	23.11.18	testValidateStatsWithNull - *метод validateStats() Проверка корректности работы функции при параметрах, один из которых равен NULL*	NULL, 0 , 0	5	0	Пройден успешно
Б2.4-Н	23.11.18	testValidateStatsWithNegativeNumber - *метод validateStats() Проверка корректности работы функции при параметрах, один из которых равен отрицательному числу*	-1, 0, 0	5	0	Пройден успешно
Б3.1-П	23.11.18	testCreateSkillBarDataCorrect - *метод createSkillBarData() Проверка корректности работы функции при корректных параметрах*	[1, 2, 3, 4, 5, 6]	5	0	Пройден успешно

Б3.2-Н	23.11.18	testCreateSkillBarDataNegative - *метод createSkillBarData() Проверка корректности работы функции при некорректных параметрах (отрицательное число)*	[1, 2, 3, 4, 5, -6]	5	0	Пройден успешно
Б3.3-Н	23.11.18	testCreateSkillBarDataIncorrect - *метод createSkillBarData() Проверка корректности работы функции при некорректных параметрах (7 элементов в массиве)*	[1, 2, 3, 4, 5, 6, 7]	5	0	Пройден успешно
И2.1-П	24.11.18	testFindSkillCorrect - *метод findById() класса SkillRepository Тест поиска навыка по корректному ID*	1	5	0	Пройден успешно
И2.2-Н	24.11.18	testFindSkillNull - *метод findById() класса SkillRepository Тест поиска навыка по NULL*	NULL	5	0	Пройден успешно
И2.3-Н	24.11.18	testFindNotExistingSkill - *метод findById() класса SkillRepository Тест поиска навыка по некорректному ID*	6	5	0	Пройден успешно
И1.1-П	24.11.18	testFindBranchCorrect - *метод findById() класса BranchRepository Тест поиска ветки по корректному ID*	1	5	0	Пройден успешно
И1.2-Н	24.11.18	testFindBranchNull - *метод findById() класса BranchRepository Тест поиска ветки по NULL*	NULL	5	0	Пройден успешно
И1.3-Н	24.11.18	testFindNotExistingBranch - *метод findById() класса BranchRepository Тест поиска ветки по некорректному ID*	6	5	0	Пройден успешно
И5.1-Н	24.11.18	testLoginNoUser - *метод login() Тест на логин пользователя с указанием несуществующих данных*	Пользователь является гостем, данные формы логина соответствуют правилам валидации ‘username’=> ‘not_existing_username, ‘password’ => ‘not_existing_password’	5	0	Пройден успешно
И5.2-Н	24.11.18	testLoginWrongPassword - *метод login() Тест на логин пользователя с указанием*	‘username’=> ‘user ‘password’ => ‘wrong_password’	5	0	Пройден успешно

		верного логина, но неправильного пароля	’			
И5.3-П	24.11.18	testLoginCorrect() - *метод login Тест на логин пользователя с указанием правильных данных*	‘username’=> ‘user’, ‘password’ => ‘password_0’	5	0	Пройден успешно
И3.1-П	24.11.18	testValidateCorrectSkills - *метод validateSkills() Тест валидации навыков с навыками, которые имеются в БД*	[1, 2, 3, 4, 5]	5	0	Пройден успешно
И3.2-Н	24.11.18	testValidateIncorrectSkills - *метод validateSkills() Тест валидации навыков с навыками, которых нет в БД*	[1, 2, 3, 4, 5, 6]	5	0	Пройден успешно
И3.3-Н	24.11.18	testValidateEmptySkills - *метод validateSkills() Тест валидации навыков с пустым массивом навыков*	[]	5	0	Пройден успешно
И3.4-Н	24.11.18	testValidateNullSkills - *метод validateSkills() Тест валидации навыков с навыками, равными NULL*	NULL	5	1	Тест провален (см. отчет № 5)
И4.1-П	24.11.18	testCreateBuildCorrect - *метод createBuild() Тест создания билда с валидными данными*	skillBar => [1, 2, 3, 4, 5] magicka => 10 strength => 15 staming => 20	5	0	Пройден успешно
И.4.2-Н	24.11.18	testCreateBuildIncorrect - *метод createBuild() Тест создания билда с невалидными данными*	skillBar => [1, 2, 3, 4, 5, 6] magicka => 100 strength => 150 staming => 200	5	0	Пройден успешно

Журнал найденных ошибок

Номер отчета об ошибке	Дата составления отчета	Номер теста	Входные данные	Ожидаемый результат	Фактический результат
№ 1	23.11.18	Б4.3-Н	‘not_existing_role’	FALSE	**[yii\base\ErrorException] Trying to get property 'name' of non-object**
№ 2	23.11.18	Б4.2-Н	NULL	FALSE	**[TypeError] Argument 1 passed to Mockery_2_common_modules**

					_user_models_User::setRole() must be of the type string, null given, called in SetRoleTest.php on line 45
№ 3	23.11.18	Б5.2-Н	NULL	FALSE	***[TypeError] Argument 1 passed to Mockery_2_common_modules_user_models_User::validatePassword() must be of the type string, null given, called in ValidatePasswordTest.php on line 30***
№ 4	23.11.18	Б1.2-Н	NULL	FALSE	***Failed asserting that true is false.***
№ 5	24.11.18	И3.4-Н	NULL	FALSE	**[TypeError] Argument 1 passed to frontend\modules\builder\service\BuilderService::validateSkills() must be of the type array, null given, called in BuilderServiceValidateSkillsTest.php on line 70**
№ 6	23.11.18	А6.3-Н	ID пользователя= “1”, роль = “not_existing_role”	FALSE	**Команда завершает свою работу. В таблице auth_assignment пропадает запись о пользователе с ID 1.**

Отчеты об ошибках

Отчет №1

Краткое описание: *testValidateNullSkills. Функция проверяет, присутствуют ли числа из входного массива в таблице Skill в качестве ID. В качестве входного массива передается NULL*
Ожидаемый результат: FALSE
Фактический результат: **[TypeError] Argument 1 passed to frontend\modules\builder\service\BuilderService::validateSkills() must be of the type array, null given, called in BuilderServiceValidateSkillsTest.php on line 70**

Отчет №2

Краткое описание: *testSetRoleWrongRole. Функция пытается установить пользователю роль, которой нет в системе. В качестве входного параметра -* ’not_existing_role’
Ожидаемый результат: FALSE

Фактический результат: **[yii\base\ErrorException] Trying to get property 'name' of non-object**

Отчет №3

Краткое описание: *testSetRoleNoRole. Функция пытается установить пользователю роль, равную NULL. В качестве входного параметра* - NULL
Ожидаемый результат: FALSE
Фактический результат: **[TypeError] Argument 1 passed to Mockery_2_common_modules_user_models_User::setRole() must be of the type string, null given, called in SetRoleTest.php on line 45**

Отчет №4

Краткое описание: *testValidatePasswordNull. Функция валидации пароля. Входной параметр - NULL*
Ожидаемый результат: FALSE
Фактический результат: ***[TypeError] Argument 1 passed to Mockery_2_common_modules_user_models_User::validatePassword() must be of the type string, null given, called in ValidatePasswordTest.php on line 30***

Отчет №5

Краткое описание: *testResetPasswordNull.Функция сброса пароля. Входной параметр - NULL*
Ожидаемый результат: FALSE
Фактический результат: TRUE (***Failed asserting that true is false***)

Отчет №6

Краткое описание: Смена роли пользователя. Администратор в консоли приложения набирает команду ./yii admin/role с некорректными аргументами командной строки (ID пользователя= “1”, роль = “not_existing_role” - несуществующая в системе роль)
Ожидаемый результат: Ошибка ‘role not found’
Фактический результат: Команда завершает свою работу. В таблице auth_assignment пропадает запись о пользователе с ID 1. Поведение функции привело к тому, что пользователь теряет роль в системе

Интерфейс приложения

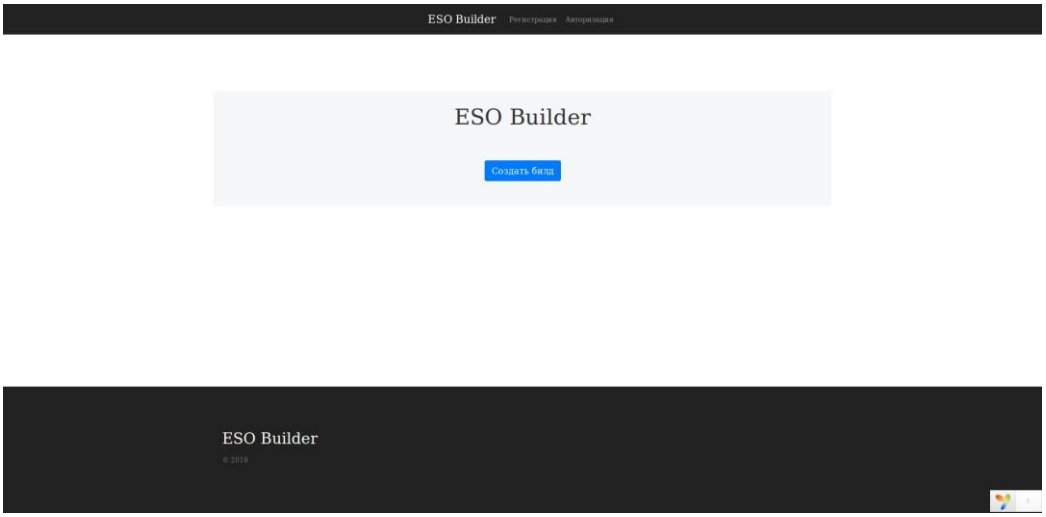


Рис. 1 Страница создания билда

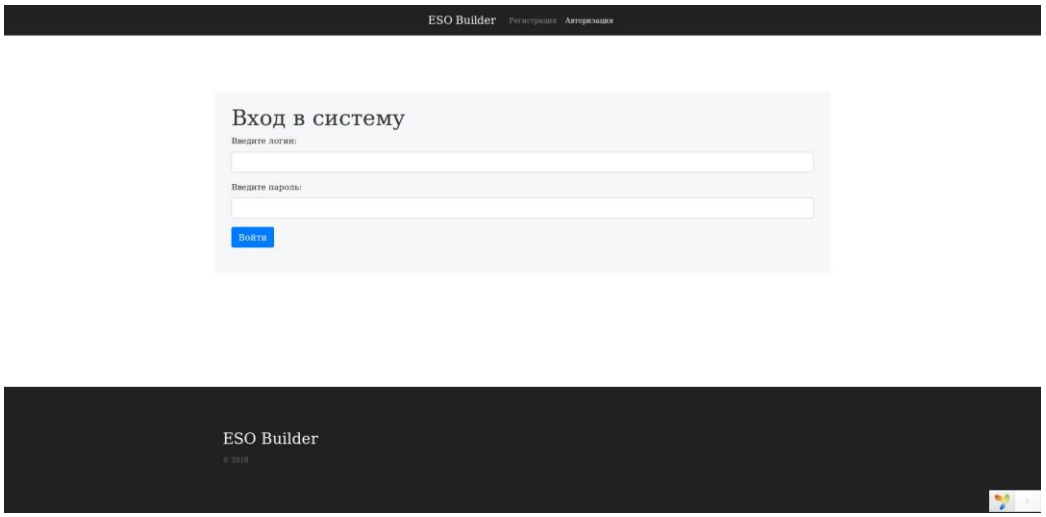


Рис. 2 Страница авторизации

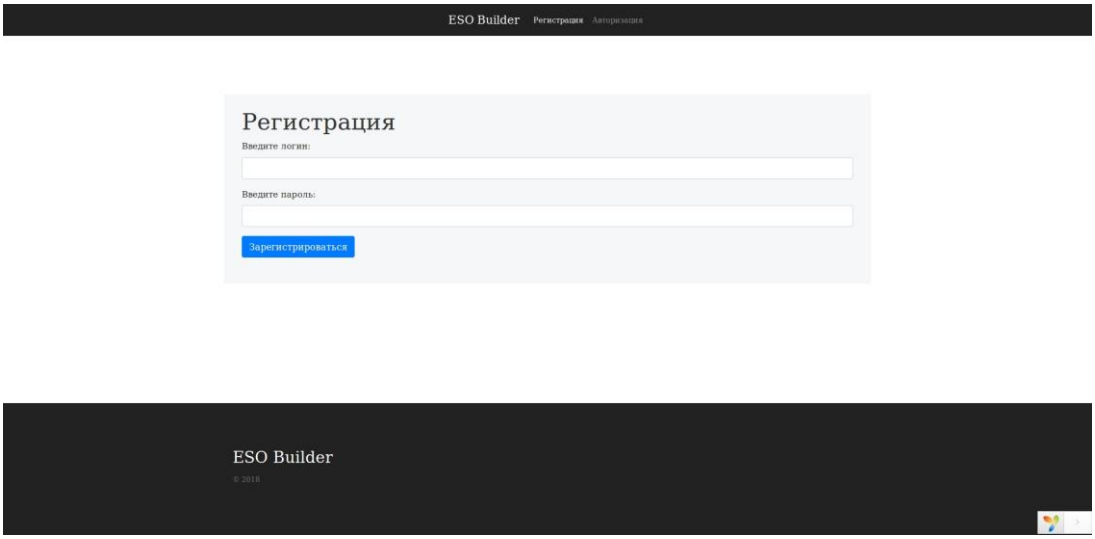


Рис. 3 Страница регистрации

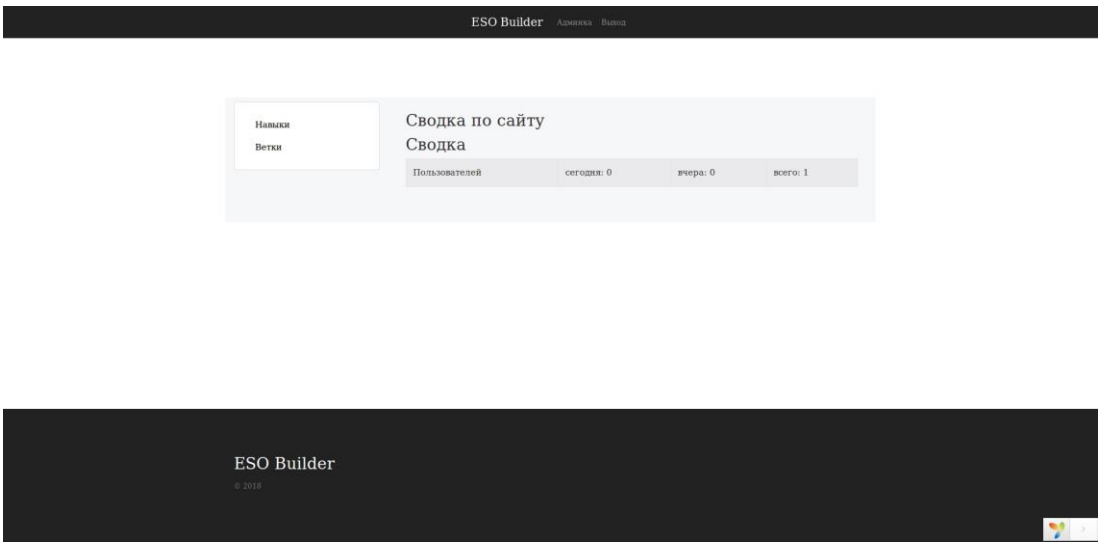


Рис. 4 Страница сводки по сайту

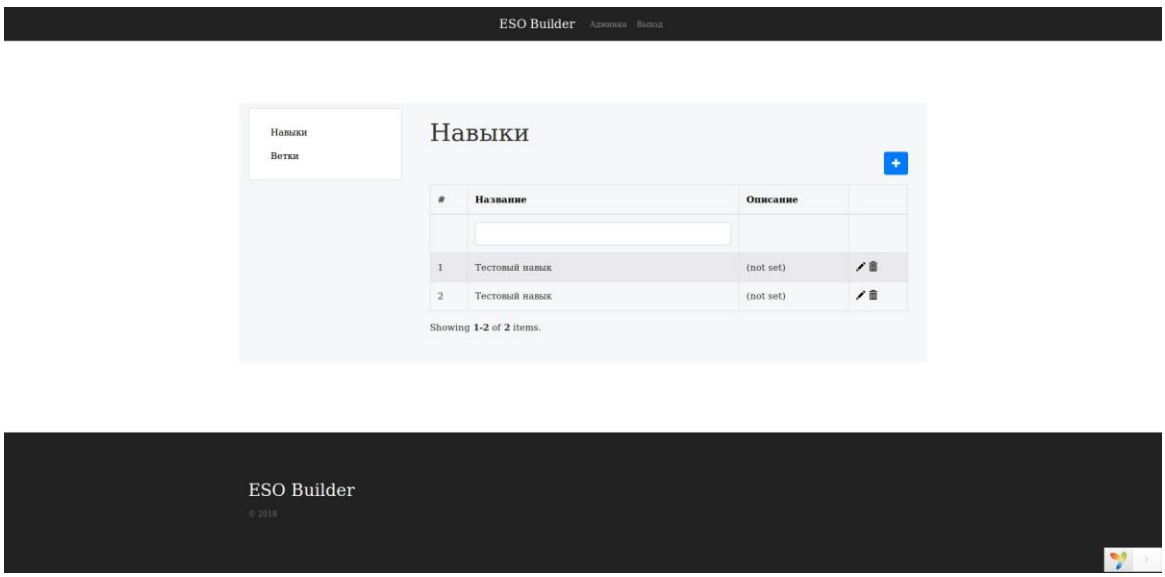


Рис. 5 Страница отображения списка навыков

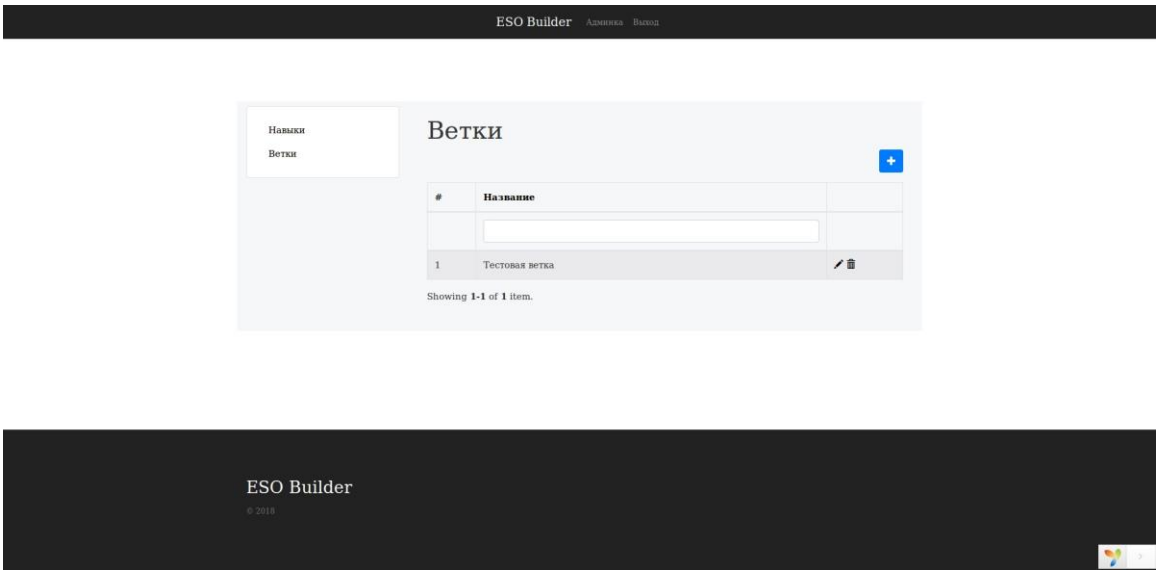


Рис. 6 Страница отображения списка веток

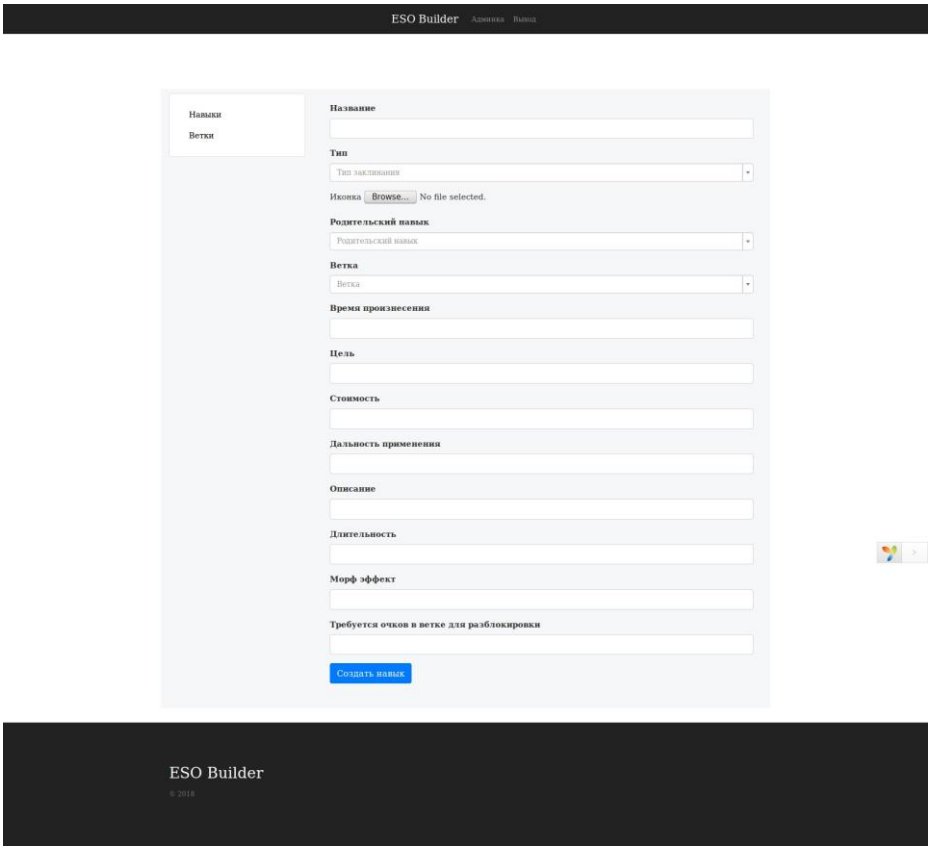


Рис. 7 Страница создания навыка

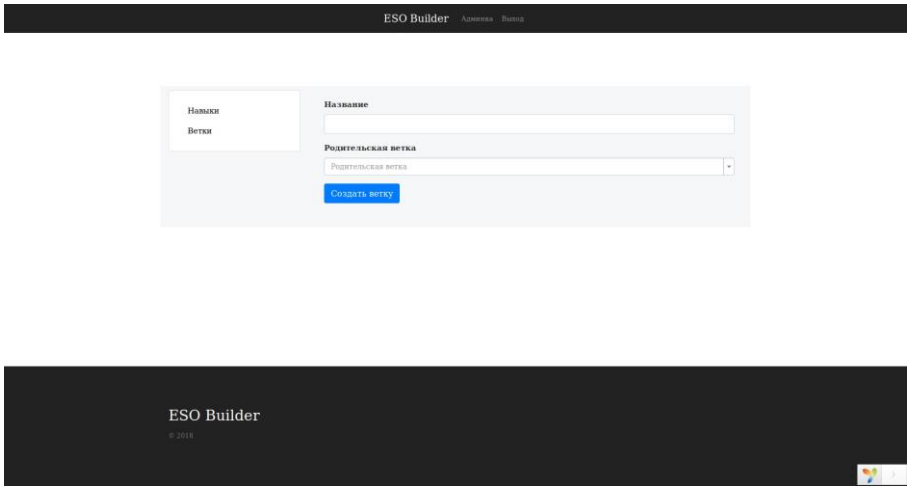


Рис. 8 Страница создания ветки

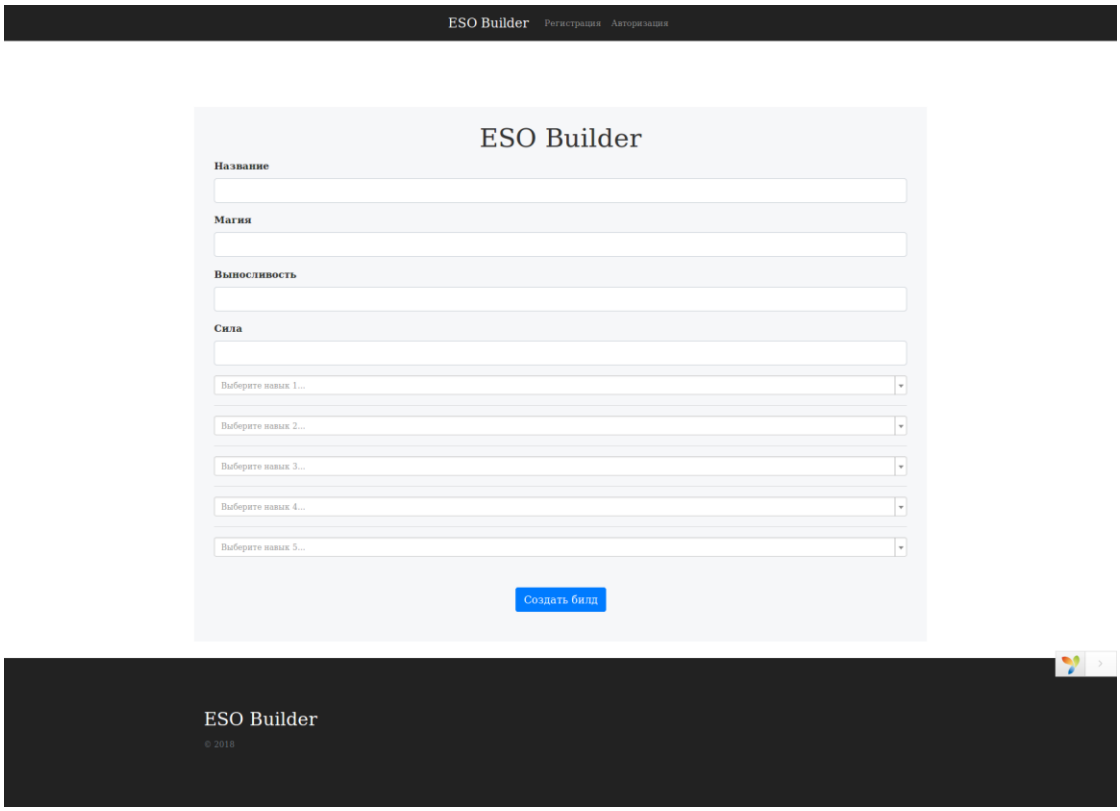


Рис. 9 Страница создания билда

Метрики тестирования

Всего запланировано тестов: 29
Выполнено: 29
Успешно: 23
Провалено: 6
Исправлено: 6

В качестве оценки тестового покрытия будем применять оценку покрытия требований. Ниже будут приведены функциональные требования к продукту. Будет указано, какие из них покрыты тестами, а какие нет. После списка будет расчитана оценка тестового покрытия.
Функциональные требования:

1. Регистрация
2. Авторизация (Login no user, Login wrong password, Login correct)
3. Сброс пароля (Reset password correct, Reset password null)
4. Смена роли через админку (Set role correct , Set role wrong role, Set role no role)
5. Смена пароля через админку (Validate password correct, Validate password null)
6. Создание билда (Validate stats if stats greater, Validate stats if stats less, Validate stats with null, Validate stats with negative number, Create skill bar data correct, Create skill bar data negative, Create skill bar data incorrect, Validate correct skills, Validate incorrect skills, Validate empty skills, Validate null skills, Create build correct, Create build incorrect, Find skill correct, Find skill null, Find not existing skill, Find branch correct, Find branch null, Find not existing branch)
7. Создание и редактирование навыка
8. Создание и редактирование ветки
9. Удаление навыка и удаление ветки

Расчет тестового покрытия относительно требований проводится по формуле:

Tcov = (Lcov/Ltotal) * 100%

где:

Tcov - тестовое покрытие

Lcov - количество требований, проверяемых тест кейсами

Ltotal - общее количество требований

Таким образом, тестовое покрытие составляет ~ 55%

Оценка качества исследуемого объекта

В ходе блочного и интеграционного тестирования в приложение было выявлено 5 не критических ошибок, которые были устранены сразу же после нахождения. В ходе аттестационного тестирования была выявлена 1 ошибка. Следовательно, можно говорить о том, что с технической точки зрения приложение работоспособно, и выполняет все заявленные разработчиком функции без ошибок.