ПЕТРОЗАВОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
КАФЕДРА ИНФОРМАТИКИ И МАТЕМАТИЧЕСКОГО ОБЕСПЕЧЕНИЯ

Отчет по дисциплине «Верификации ПО»

Выполнила:
студентка 2 курса магистратуры группа 22608 Д.С. Шорец
Преподаватель:
доцент, к.ф-м.н. К. А. Кулаков

Петрозаводск
2016

Содержание

1 Объект тестирования

1.1 Описание объекта тестирования

Объектом тестирования является веб-приложение, предназначенное для публикации, редактирования, просмотра текстовых статей, сгруппированных по различным категориям.

Веб-приложение состоит из трех компонентов:

- клиентской часть веб-приложения;
- серверная часть веб-приложения;
- база данных.

Графически схему их взаимодействия компонентов можно представить так, см. рисунок 1.

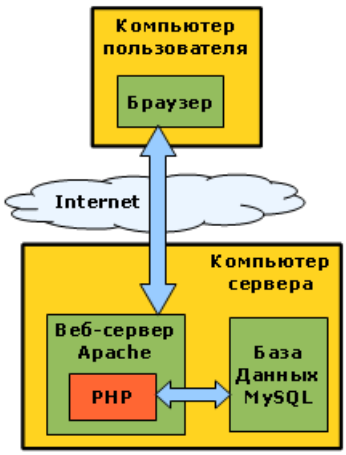

Рисунок 1 Графическая схема взаимодействия компонентов веб-приложения

Для реализации приложения использовались следующие технологии: PHP, CSS, MySQL, JavaScript, html.

1.2 Функциональность объекта тестирования

Пользователей приложением можно разделить на два типа:

- не авторизированные пользователи – обычные пользователи
- авторизированные пользователи – администраторы

Для каждого вида пользователя предусмотрен свой функционал приложения.

Веб-приложение включает в себя 8 веб-страниц:

Страницы доступные для обычных пользователей:

- **Homepage** – главная страница приложения. Функции доступные на этой странице:

- просмотр последних 5 добавленных статей, отсортированных в порядке убывания по дате добавления;
- переход к архиву всех статей приложения;
- переход к архиву статей определенной категории;
- переход к просмотру статьи;
- переход к форме авторизации пользователя

- **Archive** – страница предоставляющая доступ к архиву статей. Функции доступные на этой странице:
  - просмотр архива всех статей приложения;
  - просмотр архива статей определенной категории;
  - переход к просмотру статьи;
  - возврат к главной странице приложения.

- **viewArtical** – страница просмотра статьи. Функции доступные на этой странице:
  - просмотр статьи;
  - переход к архиву статей определенной категории;
  - возврат к главной странице приложения.

- **loginForm** – страница авторизации пользователя. Функции доступные на этой странице:
  - авторизация пользователя;
  - возврат к главной странице приложения.

Страницы доступные для администраторов:

- **listArticles –** страница просмотра всех статей. Функции доступные на этой странице:
  - просмотр всех статей, находящихся в приложение;
  - переход к просмотру статьи;
  - переход к редактированию статьи;
  - переход к созданию новой статьи;
  - удаление статьи;
  - переход к просмотру списка категорий статьи;
  - выход из системы.

- **editArticle** – страница предназначена для редактирования/создания статей. Функции доступные на этой странице:
  - создавать новую статью;

- редактировать существующую статью;
- подтверждение создания/редактирования статьи;
- отмена создания/редактирования статьи;
- выход из системы;
- возврат к странице просмотра всех статей.

- **listCategories** – страница просмотра категорий. Функции доступные на этой странице:
  - просмотр списка всех категорий;
  - удаление категории;
  - переход к созданию категории;
  - переход к редактированию категории;
  - выход из системы;
  - возврат к странице просмотра всех статей.

- **editCategories** – страница предназначена для редактирования/ создания категорий. Функции доступные на этой странице:
  - создание новой категории;
  - редактирование существующей категории;
  - подтверждение создания/редактирования категории;
  - отмена создания/редактирования категории;
  - выход из системы;
  - возврат к странице просмотра всех статей.

Общий список основных функций приложения:

1. просмотр списка статей;
2. переход между страницами;
3. просмотр архива всех статей;
4. просмотр архива статей определенной категории;
5. авторизация пользователя;
6. выход из системы;
7. просмотр статьи;
8. создание новой статьи;
9. редактирование существующих статей;
10. удаление существующих статей;
11. просмотр списка категорий;
12. создание новой категории;
13. редактирование существующих категорий;
14. удаление существующих категорий;

15. отмена создания новой статьи/категории;
16. отмена редактирования новой статьи/категории;
17. подтверждение создания статьи/категории;
18. подтверждение редактирования статьи/категории;

В аттестационном тестирование принимают участие все перечисленные функции.

2 Стратегия тестирования

2.1 Структура объекта тестирования

Обща структура приложения представлена на рисунке «Общая структура веб-приложения», см. рисунок 2.

Рисунок 2 Общая структура веб-приложения

2.2 Описание модулей системы

config.php - файла конфигурации, в нем хранятся различные установки нашего приложения

article.php – файл содержит класс для обработки статей article

Основные функции и методы класса:

- ***__construct()*** - специальный метод, который автоматически вызывается системой PHP каждый раз, когда создается новый объект Article. Конструктор получает необязательный массив $data, в котором содержатся данные для свойств нового объекта. Затем присваивает данные свойствам в теле конструктора. Метод фильтрует данные, прежде чем присвоить их свойствам

- ***getById()*** - метод, реализующий доступ к базе данных MySQL. Он принимает в качестве аргумента ID статьи ($id) и возвращает запись с указанным ID из таблицы articles, сохраняя данные в новом объекте Article.

- ***getList() -*** метод, реализующий доступ к базе данных MySQL. Он возвращает несколько статей сразу или возвращает статьи только из указанной категории.

  getList() принимает 3 аргумента:

  - ***$numRows -*** максимальное количество получаемых статей;
  - ***$order -*** порядок сортировки получаемых статей;
  - ***$categoryId –*** категория статьи.

- ***insert() –*** метод добавления новой статьи в таблицу articles, используя значения из текущего объекта Article:

  Алгоритм работы метода:

  - Сначала метод проверяет, что объект не имеет установленного свойства $id. Если у объекта есть ID, то, вероятно, статья уже имеется в базе данных и ее добавлять не нужно;
  - Затем метод выполняет запрос SQL INSERT для вставки записи в таблицу articles, используя указатели места замещения для передачи значений свойств в базу данных;
  - После выполнения запроса, метод возвращает ID новой статьи с помощью функции PDO lastInsertId() и сохраняет значение в свойстве $id. В таблице articles для поля id свойство auto_increment, поэтому MySQL генерирует уникальное значение ID для каждой новой записи.

- ***update () -*** метод похож на метод insert(),за исключением того, что здесь происходит обновление записи в базе данных вместо создания новой записи.

  Алгоритм работы метода:

  - Проверяет наличие ID у объекта, так как обновить можно только запись с известным ID;
  - Затем используем выражение SQL UPDATE для обновления полей записи.

- ***delete () –*** метод удаления статьи из таблицы articles

  Алгоритм работы метода:

  - Проверяет наличие ID у объекта, так как удалить можно только запись с известным ID;
  - Затем используем выражение SQL DELETE для удаления из таблицы articles статьи.

index.php – файл содержит скрипт клиентской части, т.е. выводит страницы в браузере пользователя.

Основные функции файла:

- ***require()*** – функция, которая подключает файл config.php, в случае отсутствия подключаемого файла функция сгенерирует ошибку;

- ***archive() -*** функция выводит список всех статей в базе данных принадлежавших к одной категории . Для этого используется метод getList() класса Article ***;***

- ***viewArticle() -*** функция выводит страницу одной статьи. Она получает ID статьи для вывода из параметра URL articleId, затем вызывает метод класса Article getById() для получения объекта статьи. Если нет параметра articleId или статья не может быть найдена, то функция просто выводит главную страницу;

- **homepage() -** функция выводит главную страницу сайта, на которой содержится список из нескольких статей, количество которых указано в параметре конфигурации HOMEPAGE_NUM_ARTICLES (по умолчанию 5).

admin.php – файл содержит скрипт серверной части.

Основные функции и методы файла:

- **session_start() -** функция запускает новую сессию пользователя, которая позволяет контролировать регистрацию пользователя в системе. Если сессия для пользователя уже имеется, то PHP автоматически возобновит ее и будет использовать;

- **login() -** функция вызывается, когда нужно произвести регистрацию пользователя в системе;

- **logout() -** функция вызывается, когда пользователь выходит из системы;

- **newArticle() -** функция позволяет пользователю создавать новую статью. Функция создает новый объект Article, сохраняет данные формы в объекте с помощью вызова функции storeFormValues(), вставляет статью в базу данных с помощью функции insert() и перенаправляет обратно на список статей, выводя сообщение об успешном завершении операции;

- **editArticle() –** функция позволяет пользователю редактировать статью. Функция получает существующую статью с помощью getById(), записывает новые значения в объекте Article, затем сохраняет измененный объект с помощью функции update() (если статья не найдена в базе данных, функция выведет сообщение об ошибке);

- **deleteArticle() -** функция позволяет пользователю удалить статью. Функция сначала получает статью (если статьи нет в базе данных, то выводится сообщение об ошибке), а затем вызывает метод delete() для удаления данных из базы. После завершения операции функция перенаправляет пользователя на страницу со списком статей и выводит сообщение о удалении;

- **listArticles() –** функция выводит список статей в CMS. Используется метод getList() класса Article для получения всех статей. Затем применяем шаблон listArticles.php для вывода списка.

- **listCategories() -** функция выводит список всех категорий для администратора;

- **newCategory() -** функция для добавления новых категорий в базу данных;

- **editCategory() -** функция редактирует существующую категорию в базе данных, позволяя пользователю менять название категории и/или описание.она действует по такому же сценарию, что и editArticle();

- **deleteCategory() -** функция позволяет удалять категорию из базы данных. Она вызывается, когда пользователь нажимает ссылку Delete This Category (Удалить

категорию) на странице редактирования категорий. Сначала функция получает категорию, заданную параметром запроса categoryId (если категории нет, то выводится сообщение об ошибке). Затем производится проверка наличия статьей в категории. Если статьи есть, то выводится сообщение об ошибке и происходит выход из функции. Если статей в категории нет, то функция удаляет категорию и перенаправляет на страницу со списком категорий, выводя сообщение об удалении категории.

header.php – файл содержит шаблон для заголовка страниц приложения.

footer.php – файл содержит шаблон для подвала страниц приложения.

homepage.php – файл содержит шаблон для вывода заголовка статей на главной странице приложения. Скрипт проходит циклом по массиву объектов Article, который хранится в $results['articles'], и выводит для каждой статьи дату публикации, название и резюме. Название ссылается на '.' (index.php), передавая параметр action=viewArticle вместе с ID статьи в URL. Таким образом, пользователь может прочитать статью, нажав на ссылку названия. Также шаблон включает ссылку на архив статей ("./?action=archive").

archive.php – файл содержит шаблон для вывода архива всех статей. Код практически идентичен homepage.php. Шаблон включает ссылку на главную страницу "Return to Homepage". Данный шаблон имеет достаточно простой код. Здесь выводится для выбранной статьи название, резюме, содержание и дата публикации. Кроме того, на странице размещается ссылка для возвращения на главную.

viewArticle.php - файл содержит шаблон для вывода одной статьи. Данный шаблон имеет достаточно простой код. Здесь выводится для выбранной статьи название, резюме, содержание и дата публикации. Кроме того, на странице размещается ссылка для возвращения на главную страницу.

loginForm.php - файл содержит форму регистрации администратора.

listArticles.php - файл содержит шаблон для вывода списка статей для редактирования администратором.

listCategories.php - файл содержит шаблон для вывода списка категорий в базе данных.

editCategory.php - файл содержит форму редактирования, позволяя администратору добавлять категорию или редактировать существующую.

editArticle.php - файл содержит форму редактирования, которая используется как для создания новых статей, так и для редактирования существующих. Она отправляет к admin.php?action=newArticle, или к admin.php?action=editArticle, в зависимости от значения переменной $results['formAction']. Форма также имеет область для сообщений

об ошибках, а также поля для названия статьи, резюме, содержания и даты публикации. Здесь также имеются 2 кнопки для сохранения и удаления изменений и ссылка, позволяющая администратору удалить только что отредактированную статью.

Category.php - файл содержит класс для обработки категорий статей category.

Основные функции и методы класса:

- ***__construct()*** - специальный метод, который создает новый объект Category;
- ***storeFormValues()*** – метод для сохранения данных из формы редактирования, методы для получения одной категории по ID и списка категорий, а также методов для вставки, обновления и удаления категории в базе данных;
- ***getById()*** - метод возвращает объект Category, соответствующий заданному ID
- ***getList() –*** метод возвращает все (или диапазон) объектов Category из базы данных
- ***insert() –*** метод вставляем текущий объект Category в базу данных и устанавливаем его свойство ID.
- ***update () -*** метод обновляет текущий объект Category в базе данных.
- ***delete () –*** метод удаляет текущий объект Category из базы данных

2.3 Описание стратегии модульного тестирования

Объектами модульного тестирования выбраны:

- метод getById класс Article;
- метод getList класс Article;
- функция require из скрипта клиентской части (файл index.php);
- метода getById класс Category;
- метода getList класс Category;
- функция login скрип серверной части приложения (admin.php)

Для проведения модульного тестирования необходимо определить все возможные входные данные, соответствующие им ожидаемые результаты.

Для совершения процедуры тестирования необходимы:

- Selenium WebDriver;
- PHPUnit;

- Браузер Mozzila Firefox.

2.4 Описание стратегии интеграционного тестирования

Для проведения интеграционного тестирования применяется стратегия восходящего тестирования. Объектами интеграционного тестирования выбраны:

- функции getlist класс article + archive из скрипта клиентской части (файл index.php) + модуль archive.php
- функции getlist класс article + getbyid класс category + archive из скрипта клиентской части (файл index.php) + модуль archive.php

- функции getbyid класс article + viewarticle из скрипта клиентской части (файл index.php)+ модуль viewarticle.php

- функции getbyid класс article + getbyid класс category + viewarticle из скрипта клиентской части (файл index.php) + модуль viewarticle.php

- функции getlist класс article + homepage из скрипта клиентской части (файл index.php) + модуль homepage.php

- функции getlist класс article + getbyid класс category+ homepage из скрипта клиентской части (файл index.php) + модуль homepage.php

- функции insert класс article + newarticle из скрипта северной части (файл admin.php) + модуль editarticle.php

- функции insert класс article + getbyid класс category +newarticle из скрипта северной части (файл admin.php) + модуль editarticle.php

- функции update класс article + editarticle из скрипта северной части (файл admin.php) + модуль editarticle.php

- функции update класс article + getbyid класс category + editarticle из скрипта северной части (файл admin.php) + модуль editarticle.php

- функции delete класс article + deletearticle из скрипта северной части (файл admin.php) + listarticles.php

- функции delete класс article + deletearticle из скрипта северной части (файл admin.php) + listarticles.php

- функции getlist класс article + listarticles из скрипта северной части (файл admin.php) + модуль listarticles.php
- функции getlist класс article + getbyid класс category+ listarticles из скрипта северной части (файл admin.php) + модуль listarticles.php

- функции getlist класс category + listcategories + модуль listcategories.php

- функции getlist класс category + newcategory+ модуль editcategories.php

- функции getlist класс category + editcategory+ модуль editcategories.php

Для проведения интеграционного тестирования необходимо определить все возможные входные данные, соответствующие им ожидаемые результаты.

Для совершения процедуры тестирования необходимы:
- Selenium WebDriver;
- PHPUnit;
- Браузер Mozzila Firefox.

2.5 Описание стратегии аттестационного тестирования

Аттестационное тестирование проводится для всей системы, что подразумевает выполнение действий в пользовательском интерфейсе.

Для совершения процедуры тестирования необходимы:

- Web браузер.

2.6 Описание стратегии тестирования юзабилити

Тестирование юзабилити проводится для всей системы в целом, что подразумевает выполнение действий в пользовательском интерфейсе. Тестирование юзабилити проводится с целью выяснения, насколько, разработанное приложение понятно и удобно для конечных пользователей.

Предполагается, что конечные пользователи имеют следующие характеристики:

- муж. /жен. от 17 до 56 лет
- опыт владения компьютером – слабое или среднее;
- образование не имеет значение.

Для совершения процедуры тестирования необходимы:

- web браузер;
- несколько добровольцев, схожих по характеристикам с предполагаемыми пользователями приложения.

2.7 Критерии прохождения тестов (применим для всех видов тестов)

Любой тест считается успешно пройденным, если ожидаемый и фактический результаты совпадают. Если тест завершается неудачей, то перед принятием решения целесообразно проверить правильность самого теста.

Если тест завершился неудачей и тест реализован правильно, то производится заключение о найденной ошибке.

2.8 Критерий приостановления тестирования (применим для всех видов тестов)

Тестирование должно быть приостановлено, если количество не пройдённых тестов превысит 40% от общего количества.

2.9 Критерий возобновления работы (применим для всех видов тестов)

Необходимо заново начать тестирование при получении уведомления, что найденные при тестировании ошибки исправлены.

3 Детальный план тестов

Корректное значений переменных, участвующих в тестирование

№	Наименование переменной	Корректное значение переменной
1	ID	числовое значение от 1 до 1000
2	numRows	числовое значение от 1 до 20
3	publicationDate	дата в формате DD.MM.YY
4	categoryId	числовое значение от 1 до 1000
5	username	от 5 до 18 символов, допускаются символы латинского алфавита, цифры
6	password	от 5 до 18 символов, обязательно должны присутствовать символы латинского алфавита и цифры
7	title	от 1 до 50 символов допускаются символов латинского алфавита, кириллица, цифры, знаки препинания
8	name	от 1 до 50 символов допускаются символов латинского алфавита, кириллица, цифры, знаки препинания
9	summary	от 1 до 200 символов допускаются символов латинского алфавита, кириллица, цифры, знаки препинания
10	contect	от 1 до 2000 символов допускаются символов латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «»
11	description	от 1 до 200 символов допускаются символов латинского алфавита, кириллица, цифры, знаки препинания

3.1 Модульное тестирование

1. Тестирование метода **getById** класс Article.

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
1	Передаем в качестве аргумента ID существующей записи	int ID существующей статьи	общий	Возвращается объект Article соответствующий заданному ID
2	Передаем в качестве аргумента ID пустое значение	null	негативный	false
3	Передаем в качестве аргумента ID не существующей записи	int ID не существующей статьи	негативный	false
4	Передаем в качестве аргумента ID не корректное значение	int ID не корректное значение	общий	false

2. Тестирование метода **getList** класс Article.

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
5	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке убывания	$numRows = all $order = `publicationDate DESC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке убывания

6	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке возрастания	$numRows = all $order = publicationDate ASC $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке возрастания
7	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке возрастания	$numRows = all $order = title ASC $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке возрастания
8	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке убывания	$numRows = all $order = title DESC $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке убывания
9	Выводим все статьи из базы данных, отсортированных по названию категории в порядке убывания	$numRows = all $order = name DESC $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по названию категории в порядке убывания
10	Выводим все статьи из базы данных, отсортированных по названию категории в порядке возрастания	$numRows = all $order = name ASK $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по названию категории в порядке возрастания
11	Выводим определенное количество статей из базы данных, отсортированных по дате публикации в порядке возрастания	$numRows = i i=1….∞ $order = publicationDate ASC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по дате публикации в порядке возрастания
12	Выводим определенное количество статей из базы данных, отсортированных по дате публикации в порядке убывания	$numRows = i i=1….∞ $order = publicationDate DESC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по дате публикации в порядке убывания
13	Выводим определенное количество статей из базы данных, отсортированных по заголовку статьи в порядке убывания	$numRows = i i=1….∞ $order = title DESC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по заголовку статьи в порядке убывания
14	Выводим определенное количество статей из базы данных, отсортированных по заголовку статьи в порядке возрастания	$numRows = i i=1….∞ $order = title ASC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по заголовку статьи в порядке возрастания
15	Выводим определенное количество статей из базы данных, отсортированных по названию категории в порядке возрастания	$numRows = i i=1….∞ $order = name ASC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по названию категории в порядке возрастания

16	Выводим определенное количество статей из базы данных, отсортированных по названию категории в порядке убывания	$numRows = i i=1….∞ $order = name DESC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по названию категории в порядке убывания
17	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания	$numRows = all $order = publictionDate DESC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания
18	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания	$numRows = all $order = publictionDate ASC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания
19	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания	$numRows = all $order = title ASC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания
20	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания	$numRows = all $order = title DESC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания
21	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по названию категории в порядке возрастания	$numRows = all $order = name ASC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания
22	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по названию категории в порядке убывания	$numRows = all $order = name DESC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания
23	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания	$numRows = i i=1….∞ $order = publictionDate DESC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания
24	Выводим определенное количество статей из	$numRows = i i=1….∞	общий	Возвращается заданное количество объектов

	базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания	$order = publicationDate ASC $categoryId = существующая категория ID		Article, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания
25	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания	$numRows = i i=1….∞ $order = title DESC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания
26	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания	$numRows = i i=1….∞ $order = title ASC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания
27	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по названию категории в порядке возрастания	$numRows = i i=1….∞ $order = name ASC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, относящихся к определенной категории и отсортированных по названию категории в порядке возрастания
28	Передаем в качестве параметра $numRows пустое значение	$numRows = null	негативный	false
29	Передаем в качестве параметра $numRows недопустимое значение	$numRows = недопустимое значение	негативный	false

3. Тестирование функции **require** из скрипта клиентской части (файл index.php).

ID тестового примера	**Описание входных данных**	**Входные данные**	**Тип теста**	**Ожидаемые результаты**
30	Подключаем существующий файл config.php	нет	общий	true
31	Подключаем не существующий файл config.php	нет	негативный	false

4. Тестирование метода **getById** класс Category.

ID тестового	**Описание входных данных**	**Входные данные**	**Тип теста**	**Ожидаемые результаты**

примера				
32	Передаем в качестве аргумента ID существующей записи	int ID существующей категории	общий	Возвращается объект Category соответствующий заданному ID
33	Передаем в качестве аргумента ID пустое значение	null	негативный	false
34	Передаем в качестве аргумента ID не существующей записи	int ID не существующей категории	негативный	false
35	Передаем в качестве аргумента ID не корректное значение	int ID не корректное значение	общий	false

5. Тестирование метода **getList** класс Category.

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
36	Выводим все категории из базы данных, отсортированных по наименованию категории в порядке убывания	$numRows = all $order = `name DESC`	общий	Возвращается весь список объектов Category отсортированных по дате публикации в порядке убывания
37	Выводим все категории из базы данных, отсортированных по наименованию категории в порядке убывания	$numRows = all $order = `name ASC`	общий	Возвращается весь список объектов Category отсортированных по дате публикации в порядке возрастания
38	Выводим определенное количество категорий из базы данных, отсортированных по наименованию категории в порядке возрастания	$numRows = i i=1….∞ $order = `name ASC`	общий	Возвращается заданное количество объектов Category отсортированных по дате публикации в порядке возрастания.
39	Выводим определенное количество категорий из базы данных, отсортированных по наименованию категории в порядке убывания	$numRows = i i=1….∞ $order = `name DESC`	общий	Возвращается заданное количество объектов Category отсортированных по дате публикации в порядке убывания.
40	Передаем в качестве параметра $numRows пустое значение	$numRows = null	негативный	false
41	Передаем в качестве параметра $numRows недопустимое значение	$numRows = недопустимое значение	негативный	false

6. Тестирование функции **login** скрип серверной части приложения

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
42	Авторизация пользователя с существующей парой логин/пароль	$username = username $password =password	общий	Успешный вход в систему
43	Авторизация пользователя с неправильным паролем	$username = username $password = password1	негативный	Сообщение об ошибке
44	Авторизация пользователя с неправильным паролем	$username = username1 $password = password	негативный	Сообщение об ошибке
45	Авторизация пользователя с неправильной парой логин/пароль	$username = username1 $password = password1	негативный	Сообщение об ошибке
46	Авторизация пользователя с пустым значением логина	$username = null $password = password1	негативный	Сообщение об ошибке
47	Авторизация пользователя с пустым значением пароля	$username = username $password = null	негативный	Сообщение об ошибке
48	Аавторизация пользователя с пустыми значениями пары логин/пароль	$username = null $password = null	негативный	Сообщение об ошибке

3.2 Интеграционное тестирование

1. Функции **getList** класс Article + **archive** из скрипта клиентской части (файл index.php) + модуль **archive.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
49	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке убывания на странице archive.php	$numRows = all $order = `publicationDate DESC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке убывания на странице archive.php
50	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке возрастания на странице archive.php	$numRows = all $order = `publicationDate ASC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке возрастания на станице archive.php

51	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке возрастания на станице archive.php	$numRows = all $order = `title ASC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке возрастания на станице archive.php
52	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке убывания на станице archive.php	$numRows = all $order = `title DESC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке убывания на станице archive.php
53	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания на станице archive.php	$numRows = all $order = publictionDate DESC $categoryId = not null	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания на станице archive.php
54	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания на станице archive.php	$numRows = all $order = publictionDate ASC $categoryId =not null	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания на станице archive.php
55	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания на станице archive.php	$numRows = all $order = title ASC $categoryId = not null	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания на станице archive.php
56	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания на станице archive.php	$numRows = all $order = title DESC $categoryId = not null	общий	Возвращается весь список объектов Article, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания на станице archive.php
57	Передаем в качестве параметра $numRows пустое значение, отсортированных по дате публикации в порядке убывания на станице archive.php	$numRows = null	негативный	false
58	Передаем в качестве параметра $numRows недопустимое значение	$numRows = недопустимое значение	негативный	false

2. Функции **getList** класс Article + **getById** класс Category + **archive** из скрипта клиентской части (файл index.php) + модуль **archive.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
59	Выводим полный архив статьей для определенной категории из базы данных, отсортированных по дате публикации в порядке убывания на странице archive.php	$numRows = all $order = `publicationDate DESC` $categoryId = существующая категория ID	общий	Возвращается полный архив статьей для определенной категории из базы данных, отсортированных по дате публикации в порядке убывания на странице archive.php
60	Выводим полный архив статьей для определенной категории из базы данных, отсортированных по дате публикации в порядке возрастания на странице archive.php	$numRows = all $order = publicationDate ASC $categoryId = существующая категория ID	общий	Возвращается полный архив статьей для определенной категории из базы данных, отсортированных по дате публикации в порядке возрастания на странице archive.php
61	Выводим полный архив статьей для определенной категории из базы данных, отсортированных по заголовку статьи в порядке возрастания на странице archive.php	$numRows = all $order = `title ASC` $categoryId = существующая категория ID	общий	Возвращается полный архив статьей для определенной категории из базы данных, отсортированных по заголовку статьи в порядке возрастания на странице archive.php
62	Выводим полный архив статьей для определенной категории из базы данных, отсортированных по заголовку статьи в порядке убывания на странице archive.php	$numRows = all $order = `title DESC` $categoryId = существующая категория ID	общий	Возвращается полный архив статьей для определенной категории из базы данных, отсортированных по заголовку статьи в порядке убывания на странице archive.php
63	Выводим полный архив статьей для определенной категории из базы данных, отсортированных по наименованию категории в порядке возрастания на странице archive.php	$numRows = all $order = `name ASC` $categoryId = существующая категория ID	общий	Возвращается полный архив статьей для определенной категории из базы данных, отсортированных по наименованию категории в порядке возрастания на странице archive.php
64	Выводим полный архив статьей для определенной категории из базы данных, отсортированных по наименованию категории в порядке убывания на странице archive.php	$numRows = all $order = `name DESC` $categoryId = существующая категория ID	общий	Возвращается полный архив статьей для определенной категории из базы данных, отсортированных по наименованию категории в порядке убывания на странице archive.php
65	Передаем в качестве параметра $numRows пустое значение	$numRows = null	негативный	false

66	Передаем в качестве параметра $numRows недопустимое значение	$numRows = недопустимое значение	негативный	false

3. Функции **getById** класс Article + **viewArticle** из скрипта клиентской части (файл index.php)+ модуль **viewArticle.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
67	Передаем в качестве аргумента ID существующей записи	int ID существующей статьи	общий	Возвращается объект Article соответствующий заданному ID
68	Передаем в качестве аргумента ID пустое значение	int ID = null	общий	возвращается главная страница
69	Передаем в качестве аргумента ID не существующей записи	int ID не существующей статьи	общий	возвращается главная страница
70	Передаем в качестве аргумента ID не корректное значение	int ID не корректное значение	общий	возвращается главная страница

4. Функции **getById** класс Article + **getById** класс Category + **viewArticle** из скрипта клиентской части (файл index.php) + модуль **viewArticle.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
71	Передаем в качестве аргумента ID существующей записи, ID категории пустое значение	int ID существующей статьи $categoryId = null	общий	Возвращается объект Article соответствующий заданному ID
72	Передаем в качестве аргумента ID пустое значение, ID категории пустое значение	int ID = null $categoryId = null	общий	возвращается главная страница
73	Передаем в качестве аргумента ID не существующей записи.	int ID не существующей статьи	общий	возвращается главная страница
74	Передаем в качестве аргумента ID не корректное значение.	int ID не корректное значение	общий	возвращается главная страница
75	Передаем в качестве аргумента ID существующей записи, ID категории существующее значение	int ID существующей статьи $categoryId существующее значение	общий	Возвращается объект Article соответствующий заданному ID, с параметром name из класса Category и ссылкой архив категории
76	Передаем в качестве аргумента ID существующей	int ID существующей статьи	негативный	false

	записи, ID категории не существующее значение	$categoryId не существующее значение		

5. Функции **getList** класс Article + **homepage** из скрипта клиентской части (файл index.php) + модуль **homepage.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
77	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке убывания на станице homepage.php	$numRows = all $order = publicationDate DESC $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке убывания на станице homepage.php
78	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке возрастания на станице homepage.php	$numRows = all $order = publicationDate ASC $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке возрастания на станице homepage.php
79	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке возрастания на станице homepage.php	$numRows = all $order = title ASC $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке возрастания на станице homepage.php
80	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке убывания на станице homepage.php	$numRows = all $order = title DESC $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке убывания на станице homepage.php
81	Выводим определенное количество статей из базы данных, отсортированных по дате публикации в порядке возрастания на станице homepage.php	$numRows = i i=1….∞ $order = publicationDate ASC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по дате публикации в порядке возрастания на станице homepage.php
82	Выводим определенное количество статей из базы данных, отсортированных по дате публикации в порядке убывания на станице homepage.php	$numRows = i i=1….∞ $order = publicationDate DESC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по дате публикации в порядке убывания на станице homepage.php
83	Выводим определенное количество статей из базы данных, отсортированных по заголовку статьи в порядке убывания на станице homepage.php	$numRows = i i=1….∞ $order = title DESC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортированных по заголовку статьи в порядке убывания на станице homepage.php
84	Выводим определенное количество статей из	$numRows = i i=1….∞ $order = title ASC $categoryId = null	общий	Возвращается заданное количество объектов Article отсортиро-

	базы данных, отсортированных по заголовку статьи в порядке возрастания на станице homepage.php			ванных по заголовку статьи в порядке возрастания на станице homepage.php
85	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания на станице homepage.php	$numRows = all $order = publictionDate DESC $categoryId =not null	общий	Возвращается весь список объектов Article, и отсортированных по дате публикации в порядке убывания на станице homepage.php
86	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания на станице homepage.php	$numRows = all $order = publictionDate ASC $categoryId = not null	общий	Возвращается весь список объектов Article, отсортированных по дате публикации в порядке возрастания на станице homepage.php
87	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания на станице homepage.php	$numRows = all $order = title ASC $categoryId = not null	общий	Возвращается весь список объектов Article, отсортированных по заголовку статьи в порядке возрастания на станице homepage.php
88	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания на станице homepage.php	$numRows = all $order = title DESC $categoryId = not null	общий	Возвращается весь список объектов Article, отсортированных по заголовку статьи в порядке убывания на станице homepage.php
89	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания на станице homepage.php	$numRows = i i=1….∞ $order = `publictionDate` DESC $categoryId = not null	общий	Возвращается заданное количество объектов Article, отсортированных по дате публикации в порядке убывания на станице homepage.php
90	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания на станице homepage.php	$numRows = i i=1….∞ $order = `publictionDate` ASC $categoryId = not null	общий	Возвращается заданное количество объектов Article, отсортированных по дате публикации в порядке возрастания на станице homepage.php
91	Выводим определенное количество статей из	$numRows = i i=1….∞ $order = `title` DESC $categoryId = not null	общий	Возвращается заданное количество объектов Article, отсортиро-

	базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания на станице homepage.php			ванных по заголовку статьи в порядке убывания на станице homepage.php
92	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания на станице homepage.php	$numRows = i i=1….∞ $order = `title` ASC $categoryId = not null	общий	Возвращается заданное количество объектов Article, по заголовку статьи в порядке возрастания на станице homepage.php
93	Передаем в качестве параметра $numRows пустое значение, отсортированных по дате публикации в порядке убывания на станице homepage.php	$numRows = null	негативный	false
94	Передаем в качестве параметра $numRows недопустимое значение	$numRows = недопустимое значение	негативный	false

6. Функции **getList** класс Article + **getById** класс Category+ **homepage** из скрипта клиентской части (файл index.php) + модуль **homepage.php**

ID тестового примера	**Описание входных данных**	**Входные данные**	**Тип теста**	**Ожидаемые результаты**
95	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания на станице homepage.php	$numRows = all $order = publictionDate DESC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, отсортированных по дате публикации в порядке убывания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
96	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания на станице homepage.php	$numRows = all $order = publictionDate ASC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, отсортированных по дате публикации в порядке возрастания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
97	Выводим все статьи из базы данных, относящихся к определенной	$numRows = all $order = title ASC	общий	Возвращается весь список объектов Article, отсортированных по заголовку статьи в порядке воз-

	категории и отсортированных по заголовку статьи в порядке возрастания на станице homepage.php	$categoryId = существующая категория ID		растания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
98	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания на станице homepage.php	$numRows = all $order = title DESC $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article, отсортированных по заголовку статьи в порядке убывания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории.
99	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания на станице homepage.php	$numRows = i i=1….∞ $order = `publicationDate` DESC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, отсортированных по дате публикации в порядке убывания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
100	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания на станице homepage.php	$numRows = i i=1….∞ $order = `publicationDate` ASC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, отсортированных по дате публикации в порядке возрастания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
101	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания на станице homepage.php	$numRows = i i=1….∞ $order = `title` DESC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, отсортированных по заголовку статьи в порядке убывания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
102	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания на станице homepage.php	$numRows = i i=1….∞ $order = `title` ASC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, отсортированных по заголовку статьи в порядке возрастания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
103	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по наименованию категории в порядке	$numRows = i i=1….∞ $order = `name` DESC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, отсортированных по наименованию категории в порядке убывания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории

	убывания на станице homepage.php			
104	Выводим определенное количество статей из базы данных, относящихся к определенной категории и отсортированных по наименованию категории в порядке возрастания на станице homepage.php	$numRows = i i=1….∞ $order = `name` ASC $categoryId = существующая категория ID	общий	Возвращается заданное количество объектов Article, отсортированных по наименованию категории в порядке возрастания на станице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
105	Выводим полный архив статьей для определенной категории из базы данных, отсортированных по наименованию категории в порядке возрастания на странице homepage.php	$numRows = all $order = `name ASC` $categoryId = существующая категория ID	общий	Возвращается полный архив статьей из базы данных, отсортированных по наименованию категории в порядке возрастания на странице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
106	Выводим полный архив статьей для определенной категории из базы данных, отсортированных по наименованию категории в порядке убывания на странице homepage.php	$numRows = all $order = `name DESC` $categoryId = существующая категория ID	общий	Возвращается полный архив статьей отсортированных по наименованию категории в порядке убывания на странице homepage.php. У каждой статьи указывается имя категории и внизу под статьей появится ссылка на архив категории
107	Выводим архив статей с недопустимым значением категории ID	$categoryId = недопустимое значение категория ID	негативный	false

7. Функции **getList** класс Article + **listArticles** из скрипта северной части (файл admin.php) + модуль **listArticles.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
108	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке убывания на станице listArticles.php	$numRows = all $order = `publicationDate DESC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке убывания на станице listArticles.php
109	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке возрастания на станице listArticles.php	$numRows = all $order = `publicationDate ASC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке возрастания на станице listArticles.php

110	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке возрастания на станице listArticles.php	$numRows = all $order = `title ASC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке возрастания на станице listArticles.php
111	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке убывания на станице listArticles.php	$numRows = all $order = `title DESC` $categoryId = null	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке убывания на станице listArticles.php
112	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке убывания на станице listArticles.php	$numRows = all $order = publictionDate DESC $categoryId = not null	общий	Возвращается весь список объектов Article, отсортированных по дате публикации в порядке убывания на станице listArticles.php
113	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по дате публикации в порядке возрастания на станице listArticles.php	$numRows = all $order = publictionDate ASC $categoryId = not null	общий	Возвращается весь список объектов Article, отсортированных по дате публикации в порядке возрастания на станице listArticles.php
114	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке возрастания на станице listArticles.php	$numRows = all $order = title ASC $categoryId = not null	общий	Возвращается весь список объектов Article, отсортированных по заголовку статьи в порядке возрастания на станице listArticles.php
115	Выводим все статьи из базы данных, относящихся к определенной категории и отсортированных по заголовку статьи в порядке убывания на станице listArticles.php	$numRows = all $order = title DESC $categoryId = not null	общий	Возвращается весь список объектов Article, отсортированных по заголовку статьи в порядке убывания на станице listArticles.php
116	Передаем в качестве параметра $numRows пустое значение, отсортированных по дате публикации в порядке убывания на станице listArticles.php	$numRows = null	негативный	false
117	Передаем в качестве параметра $numRows недопустимое значение	$numRows = недопустимое значение	негативный	false

8. Функции **getList** класс Article + **getById** класс Category+ **listArticles** из скрипта северной части (файл admin.php) + модуль **listArticles.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
118	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке убывания на станице listArticles.php	$numRows = all $order = `publicationDate DESC` $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке убывания на станице listArticles.php. У каждой статьи указывается имя ее категории. Заголовок статьи станет ссылкой для перехода на страницу редактирования статьи.
119	Выводим все статьи из базы данных, отсортированных по дате публикации в порядке возрастания на станице listArticles.php	$numRows = all $order = `publicationDate ASC` $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article отсортированных по дате публикации в порядке возрастания на станице listArticles.php. У каждой статьи указывается имя ее категории. Заголовок статьи станет ссылкой для перехода на страницу редактирования статьи.
120	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке возрастания на станице listArticles.php	$numRows = all $order = `title ASC` $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке возрастания на станице listArticles.php. У каждой статьи указывается имя ее категории. Заголовок статьи станет ссылкой для перехода на страницу редактирования статьи.
121	Выводим все статьи из базы данных, отсортированных по заголовку статьи в порядке убывания на станице listArticles.php	$numRows = all $order = `title DESC` $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article отсортированных по заголовку статьи в порядке убывания на станице listArticles.php. У каждой статьи указывается имя ее категории. Заголовок статьи станет ссылкой для перехода на страницу редактирования статьи.
122	Выводим все статьи из базы данных, отсортированных по названию категории в порядке возрастания на станице listArticles.php	$numRows = all $order = `name ASC` $categoryId = существующая категория ID	общий	Возвращается весь список объектов Article отсортированных по названию категории в порядке возрастания на станице listArticles.php. У каждой статьи указывается имя ее категории. Заголовок статьи станет ссылкой для перехода на страницу редактирования статьи.
123	Выводим все статьи из базы данных, отсортированных по названию ка-	$numRows = all $order = `name DESC`	общий	Возвращается весь список объектов Article отсортированных по названию категории статьи в порядке убывания на станице

	тегории в порядке убывания на станице listArticles.php	$categoryId = существующая категория ID		listArticles.php. У каждой статьи указывается имя ее категории. Заголовок статьи станет ссылкой для перехода на страницу редактирования статьи.
124	Передаем в качестве параметра $numRows пустое значение, отсортированных по дате публикации в порядке убывания на станице listArticles.php	$numRows = null	негативный	false
125	Передаем в качестве параметра $numRows недопустимое значение	$numRows = недопустимое значение	негативный	false

9. Функции **insert** класс Article + **newArticle** из скрипта северной части (файл admin.php) + модуль **editArticle.php**

ID тестового примера	**Описание входных данных**	**Входные данные**	**Тип теста**	**Ожидаемые результаты**
126	Добавление новой статьи с корректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	общий	Вывод сообщения об успешном завершении операции. Добавление нового объекта в Article
127	Добавление новой статьи с некорректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
128	Добавление новой статьи	$id = null	негативный	Вывод сообщения об ошибке.

	с некорректными данными	$publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = с несуществующим значением в Categories		В таблице Article все должно оставаться без изменений
129	Добавление новой статьи с некорректными данными	$id = null $publicationDate = не корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
130	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = не корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
131	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = не корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
132	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = не корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

133	Добавление новой статьи с некорректными данными	$id = null $publicationDate = не корректное значение $title = не корректное значение $summary = не корректное значение $content = не корректное значение $categoriaId = не корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
134	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = не корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
135	Добавление новой статьи с некорректными данными	$id = null $publicationDate = null $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
136	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = null $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
137	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = null $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
138	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение	негативный	Вывод сообщения об ошибке.

		$title = корректное значение $summary = корректное значение $content = null $categoriaId = корректное значение		В таблице Article все должно оставаться без изменений
139	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = null	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
140	Добавление новой статьи с некорректными данными	$id = null $publicationDate = null $title = null $summary = null $content = null $categoriaId = null	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

10. Функции **insert** класс Article + **getById** класс Category +**newArticle** из скрипта серверной части (файл admin.php) + модуль **editArticle.php**

ID тестового примера	**Описание входных данных**	**Входные данные**	**Тип теста**	**Ожидаемые результаты**
141	Добавление новой статьи с корректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	общий	Вывод сообщения об успешном завершении операции. Добавление нового объекта в Article
142	Добавление новой статьи с некорректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$content = корректное значение $name = корректное значение		
143	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = с несуществующим значением в Categories	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
144	Добавление новой статьи с некорректными данными	$id = null $publicationDate = не корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
145	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = не корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
146	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = не корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
147	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$content = не корректное значение $name = корректное значение		
148	Добавление новой статьи с некорректными данными	$id = null $publicationDate = не корректное значение $title = не корректное значение $summary = не корректное значение $content = не корректное значение $name = не корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
149	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = не корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
150	Добавление новой статьи с некорректными данными	$id = null $publicationDate = null $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
151	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = null $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
152	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = null $content = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$name = корректное значение		
153	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = null $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
154	Добавление новой статьи с некорректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = null	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
155	Добавление новой статьи с некорректными данными	$id = null $publicationDate = null $title = null $summary = null $content = null $name = null	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

11. Функции **update** класс Article + **editArticle** из скрипта северной части (файл admin.php) + модуль **editArticle.php**

ID тестового примера	**Описание входных данных**	**Входные данные**	**Тип теста**	**Ожидаемые результаты**
156	Редактирование статьи с корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	общий	Вывод сообщения об успешном завершении операции. Изменение объекта в Article
157	Редактирование статьи с не корректными данными	$id = null	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение		
158	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = с несуществующим значением в Categories	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
159	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = не корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
160	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = не корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

161	Редактирование статьи С не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = не корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
162	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = не корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
163	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = не корректное значение $title = не корректное значение $summary = не корректное значение $content = не корректное значение $categoriaId = не корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
164	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$categoriaId = не корректное значение		
165	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = null $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
166	Редактирование статьи С не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = null $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
167	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = null $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
168	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = null $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

169	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = null	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

12. Функции **update** класс Article + **getById** класс Category + **editArticle** из скрипта серверной части (файл admin.php) + модуль **editArticle.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
170	Редактирование статьи с корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	общий	Вывод сообщения об успешном завершении операции. Изменение объекта в Article
171	Редактирование статьи с не корректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
172	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$title = корректное значение $summary = корректное значение $content = корректное значение $name = с несуществующим значением в Categories		
173	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = не корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
174	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = не корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
175	Редактирование статьи С не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = не корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
176	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = не корректное значение $name = корректное значение		
177	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = не корректное значение $title = не корректное значение $summary = не корректное значение $content = не корректное значение $name = не корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
178	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = не корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
179	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = null $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
180	Редактирование статьи С не корректными данными	$id = ID существующего объекта из Article	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$publicationDate = корректное значение $title = null $summary = корректное значение $content = корректное значение $name = корректное значение		
181	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = null $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
182	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = null $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
183	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = null	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений
184	Редактирование статьи с не корректными данными	$id = ID не существующего объекта из Article $publicationDate = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$title = корректное значение $summary = корректное значение $content = корректное значение $name = null		
185	Редактирование статьи с не корректными данными	$id = ID существующего объекта из Article $publicationDate = null $title = null $summary = null $content = null $name = null	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

13. Функции **delete** класс Article + **deleteArticle** из скрипта северной части (файл admin.php) + **listArticles.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
186	Удаление статьи с корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	общий	Вывод сообщения об успешном завершении операции. Удаление объекта в Article
187	Удаление статьи с не корректными данными	$id = ID не существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$categoriaId = корректное значение		
188	Удаление статьи с не корректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $categoriaId = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

14. Функции **delete** класс Article + **getById** класс Category + **deleteArticle** из скрипта северной части (файл admin.php) + **listArticles.php**

ID тестового примера	**Описание входных данных**	**Входные данные**	**Тип теста**	**Ожидаемые результаты**
189	Удаление статьи с корректными данными	$id = ID существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	общий	Вывод сообщения об успешном завершении операции. Удаление объекта в Article
190	Удаление статьи с не корректными данными	$id = ID не существующего объекта из Article $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

		$name = корректное значение		
191	Удаление статьи с не корректными данными	$id = null $publicationDate = корректное значение $title = корректное значение $summary = корректное значение $content = корректное значение $name = корректное значение	негативный	Вывод сообщения об ошибке. В таблице Article все должно оставаться без изменений

15. Функции **getList** класс Category + **listCategories** + модуль **listCategories.php**

ID тестового примера	**Описание входных данных**	**Входные данные**	**Тип теста**	**Ожидаемые результаты**
192	Выводим все категории из базы данных, отсортированных по наименованию категории в порядке убывания	$numRows = all $order = `name DESC`	общий	Возвращается весь список объектов Category отсортированных по дате публикации в порядке убывания. С ссылкой на модуль `EditCategories.php` и с функцией `DeleteCategories`
193	Выводим все категории из базы данных, отсортированных по наименованию категории в порядке убывания	$numRows = all $order = `name ASC`	общий	Возвращается весь список объектов Category отсортированных по дате публикации в порядке возрастания. С ссылкой на модуль `EditCategories.php` и с функцией `DeleteCategories`
194	Передаем в качестве параметра $numRows пустое значение	$numRows = null	негативный	false
195	Передаем в качестве параметра $numRows недопустимое значение	$numRows = недопустимое значение	негативный	false

16. Функции **getList** класс Category + **newCategory**+ модуль **EditCategories.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
196	Создание новой категории с корректными данными	$categoryId = null $name = корректные данные $description = корректные данные	общий	Вывод сообщения об успешном завершении операции. Создание объекта в Categorys
197	Создание новой категории с не корректными данными	$categoryId = ID существующей категории $name = корректные данные $description = корректные данные	негативный	Вывод сообщения об ошибке. В Categorys все остается без изменения
198	Создание новой категории с не корректными данными	$categoryId = null $name =не корректные данные $description = корректные данные	негативный	Вывод сообщения об ошибке. В Categorys все остается без изменения
199	Создание новой категории с не корректными данными	$categoryId = null $name =корректные данные $description = не корректные данные	негативный	Вывод сообщения об ошибке. В Categorys все остается без изменения

17. Функции **getList** класс Category + **EditCategory**+ модуль **EditCategories.php**

ID тестового примера	Описание входных данных	Входные данные	Тип теста	Ожидаемые результаты
200	Редактирование категории с корректными данными	$categoryId = ID существующей категории $name = корректные данные $description = корректные данные	общий	Вывод сообщения об успешном завершении операции. Редактирование объекта в Categorys
201	Редактирование категории с не корректными данными	$categoryId = null $name = корректные данные $description = корректные данные	негативный	Вывод сообщения об ошибке. В Categorys все остается без изменения
202	Редактирование категории с не корректными данными	$categoryId = ID существующей категории $name = не корректные данные $description = корректные данные	негативный	Вывод сообщения об ошибке. В Categorys все остается без изменения
203	Редактирование категории с не корректными данными	$categoryId = ID существующей категории	негативный	Вывод сообщения об ошибке. В Categorys все остается без изменения

		$name =корректные данные $description = не корректные данные		

3.3 Аттестационное тестирование

№	204
Тестируемая функция	Авторизация пользователя
Описание	На странице авторизации пользователя вводится правильную пару логин/пароль
Ожидаемый результат	Появляется сообщение об успешной авторизации. Пользователь перенаправляется на страницу listArticle.

№	205
Тестируемая функция	Авторизация пользователя
Описание	На странице авторизации пользователя вводится неправильная пара логин/пароль
Ожидаемый результат	Появляется сообщение об ошибке авторизации

№	206
Тестируемая функция	Авторизация пользователя
Описание	На странице авторизации пользователь пытается авторизоваться с пустыми значениями пары логин/пароль
Ожидаемый результат	Появляется сообщение об ошибки авторизации

№	207
Тестируемая функция	Выход из системы
Описание	Пользователь нажимает кнопку выход из системы
Ожидаемый результат	Появляется сообщение об успешном выходе из системы. Пользователь отправляется на страницу Homepage.

№	208
Тестируемая функция	Просмотр страницы Homepage
Описание	Обычный пользователь включил приложение
Ожидаемый результат	Пользователь видит страницу Homepage приложения. На ней представлены 5 последних добавленных статей, отсортированных по дате публикации. Пользователь видит только полный заголовок статьи и ее короткое резюме, наименование ее категории. Пользователю доступны ссылки перехода на другие страницы: • заголовок статьи – переход на страницу ViewArticle; • ссылка на архив – переход на страницу Archive, на странице присутствует список всех статей приложения; • наименование категории – переход на страницу Archive, на странице присутствует список всех статей приложения данной категории; • кнопка авторизации – переход на страницу loginForm.

№	209
Тестируемая функция	Просмотр страницы Homepage
Описание	1 Переход на страницу Homepage через кнопку перехода
Ожидаемый результат	Пользователь видит страницу Homepage приложения. На ней представлены 5 последних добавленных статей, отсортированных по дате публикации. Пользователь видит только полный заголовок статьи и ее короткое резюме, наименование ее категории. Пользователю доступны ссылки перехода на другие страницы: • заголовок статьи – переход на страницу ViewArticle; • ссылка на архив – переход на страницу Archive, на странице присутствует список всех статей приложения; • наименование категории – переход на страницу Archive, на странице присутствует список всех статей приложения данной категории; кнопка авторизации – переход на страницу loginForm.

№	210
Тестируемая функция	Просмотр страницы viewArticle
Описание	1 Переход на страницу viewArticle по ссылке «имя статьи»
Ожидаемый результат	Пользователь видит страницу viewArticle на которой должна отображаться

	• заголовок статьи; • краткое резюме статьи; • текст статьи; • дата публикации; • имя категории к которой принадлежит статья; • кнопка возврата на страницу Homepage.

№	211
Тестируемая функция	Просмотр страницы Archive
Описание	1 Переход на страницу Archive по ссылке «Название категории»
Ожидаемый результат	Пользователь видит страницу Archive на которой должна отображаться • название категории; • описание категории; • список всех статей, принадлежащих категории (у каждой статьи должно иметься название и краткое резюме статьи, дата публикации); • кнопка возврата на страницу Homepage

№	212
Тестируемая функция	Просмотр страницы Archive
Описание	1 Переход на страницу Archive по кнопке перехода «ViewArchive»
Ожидаемый результат	Пользователь видит страницу Archive на которой должна отображаться список статей с • названием статьи; • краткий обзор статьи; • дата публикации; • название категории к которой принадлежит статья; • кнопка возврата на страницу Homepage

№	213
Тестируемая функция	Просмотр страницы listArticle
Описание	1 Переход на страницу listArticle после авторизации пользователя
Ожидаемый результат	Пользователь видит страницу listCategory на которой должна отображаться список всех статей с • названием статьи; • краткий обзор статьи; • дата публикации; • название категории к которой принадлежит статья; • кнопка выхода; • кнопка для удаления для каждой статьи; • кнопка создания новой статьи.

№	214
Тестируемая функция	Удаление статьи
Описание	1 Авторизированный пользователь на странице listArticle, нажимает кнопку «Удалить» у первой статьи на странице
Ожидаемый результат	• появляется сообщение об удачном удаление статьи; • статья удаляется из списка на странице listArticle; • статья удаляется из списка на странице Archive; • статья удаляется из БД.

№	215
Тестируемая функция	Просмотр страницы editArticle
Описание	1 Переход на страницу editArticle через кнопку создание новой статьи
Ожидаемый результат	Пользователь видит страницу editArticle на которой должна отображаться следующие элементы • поле для ввода названия статьи; • поле для ввода краткого обзор статьи; • список выбора категорий; • поле для ввода текста статьи; • кнопка выхода; • кнопка сохранения; • кнопка отмены.

№	216
Тестируемая функция	Создание новой статьи
Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 3. заполнение поля «Краткий обзор статьи». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. заполнение поля «Ввод текста статьи». Вводим от 1 до 2000 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 5. выбор категории; 6. нажимаем кнопку «сохранить»;

Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение статьи. 2. статья добавляется в список на странице listarticle; 3. статья добавляется в список на странице archive; 4. статья становится доступна на странице ViewArchive; 5. статья добавляется в бд.

№	217
Тестируемая функция	Создание новой статьи
Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 51 до 100 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 3. заполнение поля «Краткий обзор статьи». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. заполнение поля «Ввод текста статьи». Вводим от 1 до 2000 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 5. выбор категории; 6. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке; 2. вывод сообщения об успешном сохранение статьи; 3. статья не добавляется в список на странице listarticle; 4. статья не добавляется в список на странице archive; 5. статья не добавляется в бд.

№	218
Тестируемая функция	Создание новой статьи

Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 1 до 50 символов, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «»; 3. заполнение поля «Краткий обзор статьи». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. заполнение поля «Ввод текста статьи». Вводим от 1 до 2000 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 5. выбор категории; 6. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке; 2. вывод сообщения об успешном сохранение статьи; 3. статья не добавляется в список на странице listarticle; 4. статья не добавляется в список на странице archive; 5. статья не добавляется в бд.

№	219
Тестируемая функция	Создание новой статьи
Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 3. заполнение поля «Краткий обзор статьи». Вводим от 201 до 2000 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. заполнение поля «Ввод текста статьи». Вводим от 1 до 2000 символов, допускаются символов латинского алфавита, кириллица,

	цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 5. выбор категории; 6. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке; 2. вывод сообщения об успешном сохранение статьи; 3. статья не добавляется в список на странице listarticle; 4. статья не добавляется в список на странице archive; 5. статья не добавляется в бд.

№	220
Тестируемая функция	Создание новой статьи
Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 3. заполнение поля «Краткий обзор статьи». Вводим от 1 до 200 символов, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 4. заполнение поля «Ввод текста статьи». Вводим от 1 до 2000 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 5. выбор категории; 6. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке; 2. вывод сообщения об успешном сохранение статьи; 3. статья не добавляется в список на странице listarticle; 4. статья не добавляется в список на странице archive; 5. статья не добавляется в бд.

№	221
Тестируемая функция	Создание новой статьи
Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 3. заполнение поля «Краткий обзор статьи». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. заполнение поля «Ввод текста статьи». Вводим от 2001 до 3000 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 5. выбор категории; 6. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке; 2. вывод сообщения об успешном сохранение статьи; 3. статья не добавляется в список на странице listarticle; 4. статья не добавляется в список на странице archive; 5. статья не добавляется в бд.

№	222
Тестируемая функция	Создание новой статьи
Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 3. заполнение поля «Краткий обзор статьи». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания;

	4. заполнение поля «Ввод текста статьи». Вводим от 1 до 2000 символов, символов китайского алфавита 5. выбор категории; 6. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке; 2. вывод сообщения об успешном сохранение статьи; 3. статья не добавляется в список на странице listarticle; 4. статья не добавляется в список на странице archive; 5. статья не добавляется в бд.

№	223
Тестируемая функция	Создание новой статьи
Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 3. заполнение поля «Краткий обзор статьи». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. заполнение поля «Ввод текста статьи». Вводим от 1 до 2000 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 5. не выбираем категорию; 6. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение с просьбой выбрать категорию статьи;

№	224
Тестируемая функция	Создание новой статьи

Описание	1. Переход на страницу editArticle через кнопку создание новой статьи; 2. заполнение поля «Название статьи». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 3. заполнение поля «Краткий обзор статьи». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. заполнение поля «Ввод текста статьи». Вводим от 1 до 2000 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «» 5. выбор категории; 6. нажимаем кнопку «отменить»;
Ожидаемый результат	1. пользователь остается на странице editArticle; 2. все поля очищаются от введенных данных.

№	225
Тестируемая функция	Просмотр страницы editArticle
Описание	1 Переход на страницу editArticle по ссылке «имя статьи»
Ожидаемый результат	Пользователь видит страницу editArticle на которой должна отображаться следующие элементы • название статьи; • краткий обзор статьи; • категория статьи; • текст статьи; • кнопка выхода; • кнопка сохранения; • кнопка отмены.

№	228
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Название статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По

	окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 50 символов; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение статьи. 2. статья изменяется в списке на странице listarticle; 3. статья изменяется в списке на странице archive; 4. статья изменяется на странице ViewArchive; 5. статья изменяется в бд.

№	226
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Краткий обзор статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 200 символов; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение статьи. 2. статья изменяется в списке на странице listarticle; 3. статья изменяется в списке на странице archive; 4. статья изменяется на странице ViewArchive; 5. статья изменяется в бд.

№	227
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Ввод текста статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «». По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 2000 символов; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;

Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение статьи. 2. статья изменяется в списке на странице listarticle; 3. статья изменяется в списке на странице archive; 4. статья изменяется на странице ViewArchive; 5. статья изменяется в бд.

№	228
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Изменение категории статьи; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение статьи. 2. статья изменяется в списке на странице listarticle; 3. статья изменяется в списке на странице archive; 4. статья изменяется на странице ViewArchive; 5. статья изменяется в бд.

№	229
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Название статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 50 символов; 3. Редактирование поля «Краткий обзор статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 200 символов; 4. Редактирование поля «Ввод текста статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [],

	<>, ’’, «». По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 2000 символов; 5. Изменение категории статьи; 6. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение статьи. 2. статья изменяется в списке на странице listarticle; 3. статья изменяется в списке на странице archive; 4. статья изменяется на странице ViewArchive; 5. статья изменяется в бд.

№	230
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Название статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 51 символа; 3. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке 2. изменения в статье не происходят

№	231
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Название статьи». Вводим специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «». По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 50 символов; 3. нажимаем кнопку «сохранить»;
Ожидаемый результат	3. пользователю выводится сообщение об ошибке 4. изменения в статье не происходят

№	232
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Краткий обзор статьи». Вводим специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «». По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 200 символов; 3. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке 2. изменения в статье не происходят

№	233
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Краткий обзор статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 201 символа; 3. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке 2. изменения в статье не происходят

№	234
Тестируемая функция	Редактирование статьи
Описание	1. Переход на страницу editArticle по ссылке «имя статьи»; 2. Редактирование поля «Ввод текста статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «». По окончанию редактирования количество символов в поле должно быть не меньше 2001 символа; 3. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке

	2. изменения в статье не происходят

№	235
Тестируемая функция	Редактирование статьи
Описание	7. Переход на страницу editArticle по ссылке «имя статьи»; 8. Редактирование поля «Название статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 50 символов; 9. Редактирование поля «Краткий обзор статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 200 символов; 10. Редактирование поля «Ввод текста статьи». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания, специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «». По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 2000 символов; 11. Изменение категории статьи; 12. нажимаем кнопку «отменить»;
Ожидаемый результат	1. пользователь остается на странице editArticle; 2. во всех измененных строках появляется исходный текст.

№	236
Тестируемая функция	Просмотр страницы listCategory
Описание	1 Переход на страницу listCategory по ссылке «имя категории»
Ожидаемый результат	Пользователь видит страницу listCategory на которой должна отображаться список всех категорий приложений, у каждой категории должны выводиться: • наименование; • описание; • кнопка удаления категории; • кнопка для создания новой категории; • кнопка выхода; • кнопка перехода на страницу listArticale.

№	237
Тестируемая функция	Удаление категории
Описание	1 пользователь на странице listCategory, нажимает кнопку «Удалить» у категории которой не принадлежит ни одна статья
Ожидаемый результат	• появляется сообщение об удачном удаление категории; • категория удаляется из БД.

№	238
Тестируемая функция	Удаление статьи
Описание	1 пользователь на странице listCategory, нажимает кнопку «Удалить» у категории которой принадлежит хотя бы одна статья
Ожидаемый результат	• появляется сообщение, предупреждающее о невозможности удалить категорию которой принадлежат статьи; • категория не удаляется из БД.

№	239
Тестируемая функция	Просмотр страницы EditCategory
Описание	1 Переход на страницу EditCategory по кнопке «создание новой категории»
Ожидаемый результат	Пользователь видит страницу EditCategory на которой должна отображаться следующие пустые поля, доступные для заполнения: • поле для ввода наименования категории; • поле для ввода описания категории; • кнопка «Сохранить»; • кнопка «Отменить»; • кнопка выхода; • кнопка перехода на страницу listArticale.

№	240
Тестируемая функция	Создание новой категории
Описание	1. Переход на страницу editCategory через кнопку создание новой статьи; 2. заполнение поля «Наименование категории». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры; 3. заполнение поля «Описание категории». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. нажимаем кнопку «сохранить»;

Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение категория. 2. Категория успешно выводится в списках категорий; 3. категория добавляется в бд.

№	241
Тестируемая функция	Создание новой категории
Описание	1. Переход на страницу editCategory через кнопку создание новой статьи; 2. заполнение поля «Наименование категории». Вводим от 51 символа, допускаются символов латинского алфавита, кириллица, цифры; 3. заполнение поля «Описание категории». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибки. 2. Категория не создается

№	242
Тестируемая функция	Создание новой категории
Описание	1. Переход на страницу editCategory через кнопку создание новой статьи; 2. заполнение поля «Наименование категории». Вводим от 1 до 50 символов, вводим специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «»; 3. заполнение поля «Описание категории». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение категория. 2. Категория успешно выводится в списках категорий; 3. категория добавляется в бд.

№	243
Тестируемая функция	Создание новой категории

Описание	1. Переход на страницу editCategory через кнопку создание новой статьи; 2. заполнение поля «Наименование категории». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры; 3. заполнение поля «Описание категории». Вводим от 201 символа, вводим символы латинского алфавита, кириллицы, цифры, знаки препинания; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение категория. 2. Категория успешно выводится в списках категорий; 3. категория добавляется в бд.

№	244
Тестируемая функция	Создание новой категории
Описание	5. Переход на страницу editCategory через кнопку создание новой статьи; 6. заполнение поля «Наименование категории». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры; 7. заполнение поля «Описание категории». Вводим от 1 до 200 символов, вводим специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «»; 8. нажимаем кнопку «сохранить»;
Ожидаемый результат	4. пользователю выводится сообщение об успешном сохранение категория. 5. Категория успешно выводится в списках категорий; 6. категория добавляется в бд.

№	245
Тестируемая функция	Создание новой категории
Описание	1. Переход на страницу editCategory через кнопку создание новой статьи; 2. заполнение поля «Наименование категории». Вводим от 1 до 50 символов, допускаются символов латинского алфавита, кириллица, цифры; 3. заполнение поля «Описание категории». Вводим от 1 до 200 символов, допускаются символов латинского алфавита, кириллица, цифры, знаки препинания; 4. нажимаем кнопку «отменить»;

Ожидаемый результат	1. пользователь остается на странице editCategory; 2. все поля очищаются от введенных данных.

№	246
Тестируемая функция	Просмотр страницы editCategory
Описание	1 Переход на страницу editCategory по ссылке «наименование категории»
Ожидаемый результат	Пользователь видит страницу editArticle на которой должна отображаться следующие элементы, доступные для редактирования: • название категории; • Описание категории; • кнопка выхода; • кнопка сохранения; • кнопка отмены.

№	247
Тестируемая функция	Редактирование категории
Описание	1. Переход на страницу editCategory по ссылке «имя категории»; 2. Редактирование поля «Название категории». Вводим символы латинского алфавита, кириллица, цифры. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 50 символов; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	6. пользователю выводится сообщение об успешном сохранение категории. 7. Изменение категории сохранится в бд.

№	248
Тестируемая функция	Редактирование категории
Описание	1. Переход на страницу editCategory по ссылке «имя категории»; 2. Редактирование поля «Описание категории». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 200 символов; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об успешном сохранение категории. 2. Изменение категории сохранится в бд.

№	249
Тестируемая функция	Редактирование категории
Описание	1. Переход на страницу editCategory по ссылке «имя категории»; 2. Редактирование поля «Название категории». Вводим символы латинского алфавита, кириллица, цифры. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 50 символов; 3. Редактирование поля «Описание категории». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 200 символов; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	8. пользователю выводится сообщение об успешном сохранение категории. 9. Изменение категории сохранится в бд.

№	250
Тестируемая функция	Редактирование категории
Описание	1. Переход на страницу editCategory по ссылке «имя категории»; 2. Редактирование поля «Название категории». Вводим специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «». По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 50 символов; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке. 2. Изменение категории не происходит.

№	251
Тестируемая функция	Редактирование категории
Описание	1. Переход на страницу editCategory по ссылке «имя категории»; 2. Редактирование поля «Название категории». Вводим символы латинского алфавита, кириллица, цифры. По окончанию редактирования количество символов в поле должно быть не меньше 51 символа; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;

Ожидаемый результат	1. пользователю выводится сообщение об ошибке. 2. Изменение категории не происходит.

№	252
Тестируемая функция	Редактирование категории
Описание	1. Переход на страницу editCategory по ссылке «имя категории»; 2. Редактирование поля «Описание категории». Вводим специальные символы: @, #, №, $, %, ^, &, *, {}, [], <>, ’’, «». По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 200 символов; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке. 2. Изменение категории не происходит.

№	253
Тестируемая функция	Редактирование категории
Описание	1. Переход на страницу editCategory по ссылке «имя категории»; 2. Редактирование поля «Описание категории». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 201 символа; 3. Остальные поля оставляем без изменений; 4. нажимаем кнопку «сохранить»;
Ожидаемый результат	1. пользователю выводится сообщение об ошибке. 2. Изменение категории не происходит.

№	254
Тестируемая функция	Редактирование категории
Описание	1. Переход на страницу editCategory по ссылке «имя категории»; 2. Редактирование поля «Название категории». Вводим символы латинского алфавита, кириллица, цифры. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 50 символов;

	3. Редактирование поля «Описание категории». Вводим символы латинского алфавита, кириллица, цифры, знаки препинания. По окончанию редактирования количество символов в поле должно быть не меньше 1 символа и не больше 200 символов; 4. нажимаем кнопку «отменить»;
Ожидаемый результат	1. пользователь остается на странице editCategory; 2. во всех измененных строках появляется исходный текст.

3.4 Тестирование юзабилити

№	255
Тестируется	интерфейс
Задание для пользователя	Пользователь, не знакомый с функциональностью приложения, без доступа к функциям администратора • открывает приложение в браузере; • самостоятельно изучает его; • описывает предназначения приложение; • описывает функциональные возможности, предоставляемые обычному пользователю.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	256
Тестируется	интерфейс
Задание для пользователя	Пользователь, не знакомый с функциональностью приложения предоставляемую администратору. • открывает приложение в браузере; • авторизуется; • самостоятельно изучает административную часть приложения; • описывает функциональные возможности, предоставляемые администратору.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	257
Тестируется	Функция просмотра всех статей, просмотр всех статей определенной категории, переход между страницами
Задание для пользователя	Пользователю, необходимо • просмотреть все статьи, которые находятся в приложение. • просмотреть все статьи, принадлежащие определенной категории;
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	258
Тестируется	функция просмотра всех статей, просмотр всех статей определенной категории, переход между страницами
Задание для пользователя	Пользователю, необходимо • просмотреть все статьи, принадлежащие определенной категории.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	259
Тестируется	просмотр всех статей определенной категории, переход между страницами
Задание для пользователя	Пользователю, необходимо • просмотреть три разных статьи.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	260
Тестируется	функция авторизации
Задание для пользователя	Пользователю, необходимо • попытаться авторизоваться с неправильной парой логин/пароль.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания;

	• сложность выполнения – легкая/средняя/сложная;

№	261
Тестируется	функция авторизации
Задание для пользователя	Пользователю, необходимо • попытаться авторизоваться с пустой парой логин/пароль.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	262
Тестируется	функция создания новой статьи
Задание для пользователя	Пользователю, необходимо • авторизоваться; • перейти на страницу создания новой статьи; • заполнить поля; • создать новую статью.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	263
Тестируется	Функция отмены создания новой статьи
Задание для пользователя	Пользователю, необходимо • авторизоваться; • перейти на страницу создания новой статьи; • заполнить поля; • отменить создание новой статьи.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	264
Тестируется	функция редактирования статьи
Задание для пользователя	Пользователю, необходимо

	• авторизоваться; • выбрать статью для редактирования; • отредактировать поля статьи; • сохранить изменения.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	265
Тестируется	функция отмена редактирования статьи
Задание для пользователя	Пользователю, необходимо • авторизоваться; • выбрать статью для редактирования; • отредактировать поля статьи; • отменить изменения.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	266
Тестируется	функция создания новой категории
Задание для пользователя	Пользователю, необходимо • авторизоваться; • перейти на страницу создания новой категории; • заполнить поля; • создать новую категорию.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	267
Тестируется	функция отмены создания новой категории
Задание для пользователя	Пользователю, необходимо • авторизоваться; • перейти на страницу создания новой категории;

	• заполнить поля; • отменить создание новой категории.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	268
Тестируется	функция редактирования категории
Задание для пользователя	Пользователю, необходимо • авторизоваться; • выбрать категорию для редактирования; • отредактировать поля статьи; • сохранить изменения.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	269
Тестируется	функция отмена редактирования категории
Задание для пользователя	Пользователю, необходимо • авторизоваться; • выбрать категорию для редактирования; • отредактировать поля категории; • отменить изменения.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

№	270
Тестируется	функция удаления категории
Задание для пользователя	Пользователю, необходимо • авторизоваться; • выбрать категорию для удаления; • удалить категорию.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания;

	• сложность выполнения – легкая/средняя/сложная;

№	271
Тестируется	функция удаления статьи
Задание для пользователя	Пользователю, необходимо • авторизоваться; • выбрать статью для удаления; • удалить статью.
Показатели для оценки тестирования	• ID пользователя; • статус выполнения – негативный/положительный; • время выполнения задания; • сложность выполнения – легкая/средняя/сложная;

3.5 Пример реализации модульного тестирования

Пример реализации теста №46 (тестирование функции **login)**

```
require "json"
require "selenium-webdriver"
gem "test-unit"
require "test/unit"

class Test1 < Test::Unit::TestCase

  def setup
     @driver = Selenium::WebDriver.for :firefox
    @base_url = "http://kappa.cs.karelia.ru/
    @accept_next_alert = true
    @driver.manage.timeouts.implicit_wait = 30
    @verification_errors = []
  end

  def teardown
    @driver.quit
    assert_equal [], @verification_errors
  end

  def test_1
    @driver.get(@base_url + "/~shorec/forma55/cms/templates/admin/loginForm.php ")
    @driver.find_element(:id, "username").clear
    @driver.find_element(:id, "username").send_keys "username1"
    @driver.find_element(:id, "password").clear
    @driver.find_element(:id, "password").send_keys "password1"
    @driver.find_element(:css, "button[type=\"submit\"]").click
  end

  def element_present?(how, what)
    ${receiver}.find_element(how, what)
    true
  rescue Selenium::WebDriver::Error::NoSuchElementError
    false
  end

  def alert_present?()
    ${receiver}.switch_to.alert
    true
  rescue Selenium::WebDriver::Error::NoAlertPresentError
    false
  end
```

```
  def verify(&blk)
    yield
  rescue Test::Unit::AssertionFailedError => ex
    @verification_errors << ex
  end

  def close_alert_and_get_its_text(how, what)
    alert = ${receiver}.switch to().alert()
    alert_text = alert.text
    if (@accept_next_alert) then
      alert.accept()
    else
      alert.dismiss()
    end
    alert_text
  ensure
    @accept_next_alert = true
  end
end
```

3.6 Пример реализации интеграционного тестирования

Пример реализации теста №197 (тестирование функции **getList** класс Category + **new-Category**+ модуль **EditCategories.php)**

```
require "json"
require "selenium-webdriver"
gem "test-unit"
require "test/unit"

class Proba1 < Test::Unit::TestCase

  def setup
    @driver = Selenium::WebDriver.for :firefox
    @base_url = "http://kappa.cs.karelia.ru/"
    @accept_next_alert = true
    @driver.manage.timeouts.implicit_wait = 30
    @verification_errors = []
  end

  def teardown
    @driver.quit
    assert_equal [], @verification_errors
  end

  def test_proba1
    @driver.get(@base_url + "/~shorec/forma55/cms/templates/admin/EditCategory.php")
    @driver.find_element(:id, "name").clear
    @driver.find_element(:id, "name").send_keys "ggggg"
          @driver.find_element(:id, "description").clear
    @driver.find_element(:id, "description").send_keys "ggggg"
    @driver.find_element(:id, "btn-add").click
  end

  def element_present?(how, what)
    ${receiver}.find_element(how, what)
    true
  rescue Selenium::WebDriver::Error::NoSuchElementError
    false
  end

  def alert_present?()
    ${receiver}.switch_to.alert
    true
  rescue Selenium::WebDriver::Error::NoAlertPresentError
    false
  end

  def verify(&blk)
    yield
  rescue Test::Unit::AssertionFailedError => ex
    @verification_errors << ex
  end
```

```
  def close alert and get its text(how, what)
    alert = ${receiver}.switch_to().alert()
    alert_text = alert.text
    if (@accept_next_alert) then
      alert.accept()
    else
      alert.dismiss()
    end
    alert_text
  ensure
    @accept_next_alert = true
  end
end
```

3.7 Покрытие кода тестами

Количество строк кода тестируемого приложения = 1020;

Количество строк кода, покрытых тестами = 684;

Покрытие равно=(684/1020) *100%=67%

4 Результаты тестирования

4.1 Сводная таблица по тестированию

Модульные + интеграционные тесты

Номер тестов	Тестируемый модуль/модули	Найденные ошибки	Дата проведения
1-4	getById класс Article	0	27.12.15
5-29	getList класс Article	0	27.12.15
30-31	require из скрипта клиентской части (файл index.php)	0	27.12.15
32-35	getById класс Category	0	27.12.15
36-41	getList класс Category	0	27.12.15
42-48	login скрип серверной части приложения	0	27.12.15
49-58	getList класс Article + archive из скрипта клиентской части (файл index.php) + модуль archive.php	0	27.12.15
59-66	getList класс Article + getById класс Category + archive из скрипта клиентской части (файл index.php) + модуль archive.php	0	27.12.15
67-70	getById класс Article + viewArticle из скрипта клиент-ской части (файл index.php)+ модуль viewArticle.php	0	27.12.15
71-76	getById класс Article + getById класс Category + viewArticle из скрипта клиентской части (файл index.php) + модуль viewArticle.php	0	27.12.15
77-94	getList класс Article + homepage из скрипта клиентской части (файл index.php) + модуль homepage.php	1	27.12.15

95-107	getList класс Article + getById класс Category+ homepage из скрипта клиентской части (файл index.php) + модуль homepage.php	0	27.12.15
108-117	getList класс Article + listArticles из скрипта северной части (файл admin.php) + модуль listArticles.php	0	27.12.15
118-125	getList класс Article + getById класс Category+ listArticles из скрипта северной части (файл admin.php) + модуль listArticles.php	0	27.12.15
126 - 140	insert класс Article + newArticle из скрипта северной части (файл admin.php) + модуль editArticle.php	1	27.12.15
141 - 155	insert класс Article + getById класс Category +newArticle из скрипта северной части (файл admin.php) + модуль editArticle.php	0	28.12.15
156 - 169	update класс Article + editArticle из скрипта северной части (файл admin.php) + модуль editArticle.php	0	28.12.15
170 - 185	update класс Article + getById класс Category + editArticle из скрипта северной ча-сти (файл admin.php) + модуль editArticle.php	0	28.12.15
186 - 188	delete класс Article + deleteArticle из скрипта север-ной части (файл admin.php) + listArticles.php	0	28.12.15
189 - 191	delete класс Article + getById класс Category + deleteArticle из скрипта северной части (файл admin.php) + listArticles.php	0	28.12.15
192 - 195	getList класс Category + listCate-gories + модуль listCatego-ries.php	0	28.12.15
196 - 199	getList класс Category + newCat-egory+ модуль EditCatego-ries.php	0	28.12.15
200 - 203	getList класс Category + EditCate-gory+ модуль EditCategories.php	0	28.12.15

Аттестационные тесты

Номер тестов	Тестируемая функция	Найденные ошибки	Дата проведения
204-206	Авторизация пользователя	0	29.12.15

207	Выход из системы	0	29.12.15
208 -209	Просмотр списка статей на странице Homepage	0	29.12.15
210	Просмотр страницы viewArticle	0	29.12.15
211 - 212	Просмотр страницы Archive	0	29.12.15
213	Просмотр страницы listArticle	0	29.12.15
214	Удаление статьи	0	29.12.15
215	Просмотр страницы editArticle	0	29.12.15
216-224	Создание новой статьи	0	29.12.15
225	Просмотр страницы editArticle	0	29.12.15
226 - 235	Редактирование статьи	0	29.12.15
236	Просмотр страницы listCategory	0	29.12.15
237	Удаление категории	0	29.12.15
238	Удаление статьи	0	29.12.15
239 и 246	Просмотр страницы EditCate-gory	0	29.12.15
240 - 245	Создание новой категории	0	29.12.15
247 - 254	Редактирование категории	0	29.12.15

4.2 Найденные ошибки

Ошибки, найденные входе модульного и интеграционного тестирования:

Входе проведения тестирования была выявлена проблема с кодировкой и добавлением русскоязычного текста.

Для решения проблемы были выполнены следующие изменения кода:

- в файле classes/Article.php, в функции _construct исходные строки

```
if ( isset( $data['title'] ) ) $this->title = preg_replace ( "/[^\.\,\-\_\'\"\@\?\!\:\$ a-zA-Z0-9()]/", "", $data['title'] );
if ( isset( $data['summary'] ) ) $this->summary = preg_replace ( "/[^\.\,\-\_\'\"\@\?\!\:\$ a-zA-Z0-9()]/", "", $data['summary'] );
```

заменили на

```
if ( isset( $data['title'] ) ) $this->title = preg_replace ( "/[^\.\,\-\_\'\"\@\?\!\:\$ a-яА-ЯёЁa-zA-Z0-9()]/", "", $data['title'] );
if ( isset( $data['summary'] ) ) $this->summary = preg_replace ( "/[^\.\,\-\_\'\"\@\?\!\:\$ a-яА-ЯёЁa-zA-Z0-9()]/", "", $data['summary'] );
```

- в файле templates/include/header.php исходную строку

```
<html lang="en">
```

заменили на

```
<html>
```

и добавили строку

```
<meta charset="utf-8">
```

- в файле config.php исходные строки

```
define( "DB_DSN", "mysql:host=localhost;dbname=cms;" );
```

заменили на

```
define( "DB_DSN", "mysql:host=localhost;dbname=cms;charset=utf8" );
```

В ходе аттестационного тестирования ошибки не были обнаружены.

В ходе тестирования юзабилити были выявлены следующие недостатки приложения:

- удалить статью можно только на общей странице просмотра списка статей listArticle, при этом на странице где можно посмотреть конкретную статью, ее удалить нельзя;
- удалить категорию можно только на общей странице просмотра списка категорий listArticle, при этом на странице где можно посмотреть конкретную категорию, ее удалить нельзя;
- удалить категорию, можно только если к ней не привязана ни одна статья, однако пользователь нигде не может ознакомиться с тем сколько и какие статьи привязаны к данной категории;
- некорректно реализована функция удаления статьи и категории. Удаление статьи/категории происходит сразу же и без возвратно при нажатии на кнопку «Удалить». Что может привести к случайному нежелательно удалению статьи/категории;
- при заполнении полей во время редактирования/создания статьи или категории, информация о не корректном заполнение появляется только после нажатия кнопки «Сохранить», при этом в сообщение об ошибке не сообщается какое именно поле заполнено не корректно. Удобнее было бы проверка происходила сразу же у конкретного поля.

4.3 Оценка качества исследуемого объекта

В ходе модульного и интеграционного тестирования в приложение было выявлено 2 не критические ошибки, которые были устранены сразу же после нахождения. В ходе аттестационного тестирования не было выявлено ошибок. Следовательно, можно говорить о том, что с технической точки зрения приложение работоспособно, и выполняет все заявленные разработчиком функции без ошибок. Однако в ходе проведения тестирования на удобство использования приложения конкретными пользователями, были выявлены недостатки, которые требуют рассмотрения и устранения.