МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «СТАВРОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»

Факультет цифровых технологий

Кафедра информационных систем

Методические указания
по выполнению и защите курсового проекта по дисциплине «Web-программирование»
для обучающихся очной и заочной форм обучения направления подготовки 09.03.02 «Информационные системы и технологии»

Ставрополь 2026

Содержание

1. Цели и задачи работы

Целью курсового проекта по дисциплине «Web-программирование» является систематизация и практическое применение знаний о клиентских и серверных Web-технологиях при разработке законченного программного продукта, решающего конкретную прикладную задачу.

Курсовой проект должен показать способность обучающегося самостоятельно уточнять требования, проектировать пользовательский интерфейс и архитектуру приложения, реализовывать программные модули, организовывать работу с данными, выполнять тестирование и представлять результат в форме, пригодной для воспроизведения и оценки.

Основные цели выполнения проекта:

- закрепление навыков семантической разметки HTML, адаптивного оформления CSS и программирования на JavaScript;
- освоение полного цикла разработки Web-приложения: от анализа требований до публикации и сопровождения;
- формирование навыков обоснованного выбора библиотек, инструментов сборки, способов хранения и передачи данных;
- применение требований к доступности, безопасности, производительности и качеству пользовательского интерфейса;
- развитие навыков технической документации, контроля версий, тестирования и публичной защиты программного продукта.

При выполнении курсового проекта обучающийся решает следующие задачи:

- анализирует предметную область, целевую аудиторию и существующие аналоги;
- формулирует функциональные и нефункциональные требования, пользовательские сценарии и критерии приёмки;
- проектирует структуру данных, навигацию, интерфейс и взаимодействие компонентов;

- реализует адаптивное Web-приложение с обработкой пользовательского ввода и динамическим изменением интерфейса;
- организует получение, сохранение и обработку данных с использованием Web API, локального хранилища или серверной части в соответствии с заданием;
- проводит функциональное тестирование, проверку граничных случаев, доступности, безопасности и производительности;
- подготавливает исходный код, пояснительную записку, руководство пользователя и материалы для демонстрации.

Результатом является работоспособное Web-приложение и комплект материалов, позволяющий преподавателю развернуть проект, проверить заявленные функции и оценить самостоятельность принятых решений.

2. Рекомендуемые темы курсовых проектов

Тема выбирается из перечня, предложенного кафедрой, либо формулируется обучающимся и утверждается ведущим преподавателем. В формулировке темы должны быть обозначены назначение приложения, основные пользователи и ключевой результат разработки. Конкретный набор технологий и обязательных функций фиксируется в задании.

1. Разработка адаптивного Web-приложения для планирования учебных задач и контроля сроков.
2. Разработка личного кабинета обучающегося с визуализацией учебного прогресса.
3. Разработка Web-сервиса бронирования аудиторий, оборудования или консультаций.
4. Разработка информационной системы учёта заявок службы технической поддержки.
5. Разработка каталога образовательных ресурсов с поиском, фильтрацией и избранным.
6. Разработка Web-приложения для проведения опросов и анализа результатов.
7. Разработка панели мониторинга показателей на основе открытых данных.
8. Разработка клиентского приложения для работы с открытым REST API.
9. Разработка интерактивной карты объектов университета или городской инфраструктуры.
10. Разработка Web-приложения для учёта личных финансов и визуализации расходов.
11. Разработка прототипа интернет-магазина с каталогом, корзиной и оформлением заказа.
12. Разработка системы регистрации участников и расписания мероприятий.
13. Разработка Web-приложения для ведения базы знаний кафедры.

14. Разработка электронного портфолио с управлением проектами и достижениями.
15. Разработка PWA для работы с данными в условиях нестабильного подключения.
16. Разработка доступного Web-интерфейса для пользователей с особыми потребностями.
17. Разработка сервиса визуального сравнения наборов данных или вариантов решений.
18. Разработка Web-приложения для мониторинга агротехнологических показателей.
19. Разработка интерфейса управления устройствами или датчиками на основе имитационного API.
20. Разработка Web-приложения по теме, предложенной обучающимся, при наличии согласованного технического задания.

Не допускается выполнение проекта в виде набора статических страниц без программной логики. Минимальная функциональность должна включать навигацию, обработку ввода, динамическое состояние, валидацию данных, информирование пользователя об ошибках и не менее одного механизма сохранения или получения данных.

3. Требования к структуре работы

Курсовой проект включает программный продукт и пояснительную записку. Пояснительная записка должна содержать:

- титульный лист (Приложение 1);
- содержание;
- введение;
- основную часть;
- заключение;
- список использованных источников;
- приложения при необходимости.

3.1. Введение

Во введении обосновываются актуальность и практическая значимость темы, формулируются цель и задачи проекта, определяются объект и предмет разработки, целевые пользователи, применяемые методы и технологии. Завершается введение кратким описанием структуры работы. Рекомендуемый объём — 2-3 страницы.

3.2. Основная часть

Основная часть строится как последовательное обоснование и реализация проектного решения. Допускается уточнение наименований подразделов в соответствии с темой, однако следующая логика должна быть сохранена:

Раздел	Рекомендуемое содержание
1. Анализ задачи и требования	Предметная область и аналоги; пользователи и сценарии; функциональные и нефункциональные требования; критерии приёмки; обоснование технологического стека.
2. Проектирование и реализация	Архитектура и структура данных; навигация и прототип интерфейса; реализация компонентов и модулей; работа с API/хранилищем; обработка ошибок; ключевые фрагменты решений.
3. Тестирование и ввод в эксплуатацию	План и результаты тестирования; доступность, безопасность и производительность; порядок сборки и запуска; развёртывание; руководство пользователя; ограничения и направления развития.

Каждый раздел завершается краткими выводами. В тексте необходимо объяснять принятые решения, а не пересказывать листинги. Полные исходные

тексты размещаются в электронном приложении или репозитории; в пояснительную записку включаются только фрагменты, необходимые для объяснения алгоритма, структуры или нетривиального решения.

3.3. Заключение

В заключении кратко сопоставляются полученные результаты с целью, задачами и критериями приёмки, приводятся сведения о реализованных функциях, результатах тестирования, выявленных ограничениях и возможных направлениях развития. Рекомендуемый объём — 1-2 страницы.

3.4. Электронные материалы

К пояснительной записке прилагается архив проекта или ссылка на доступный преподавателю репозиторий. Комплект должен включать исходный код, файл README с инструкцией по установке и запуску, перечень зависимостей, тестовые данные, при необходимости конфигурационный пример без секретных ключей, а также ссылку на опубликованную версию проекта, если публикация предусмотрена заданием.

4. Требования к оформлению работы

Пояснительная записка оформляется на листах формата А4. Титульный лист содержит полное наименование учредителя и университета, факультета и кафедры, дисциплину, тему, сведения об исполнителе и преподавателе, место и год выполнения, а также поля для регистрации и итоговой оценки.

Основные параметры оформления:

- поля: левое — 30 мм, правое — 15 мм, верхнее и нижнее — 20 мм;
- основной шрифт — Times New Roman, 14 пт;
- межстрочный интервал — полуторный; абзацный отступ — 1,25 см;
- выравнивание основного текста — по ширине; автоматические переносы слов не используются;
- нумерация страниц — сквозная, внизу по центру; титульный лист учитывается, но номер на нём не печатается;
- каждый раздел первого уровня начинается с новой страницы; заголовки печатаются без точки в конце;
- рекомендуемый объём пояснительной записки — не менее 25 страниц без учёта приложений.

4.1. Таблицы, рисунки и листинги

Таблицы и рисунки размещаются после первого упоминания. Они нумеруются, имеют содержательные названия и сопровождаются пояснением результата. При переносе таблицы повторяется строка заголовков и используется обозначение «Продолжение таблицы ...». Диаграммы архитектуры, схемы навигации, модель данных, прототипы интерфейса и результаты измерений включаются тогда, когда они помогают проверить решение.

Фрагменты программного кода набираются моноширинным шрифтом 10-11 пт с одинарным интервалом. Каждый фрагмент должен быть кратким, читаемым и сопровождаться пояснением. Скриншоты кода вместо редактируемого текста не допускаются. Полные листинги передаются в электронном приложении.

4.2. Требования к программному продукту

- интерфейс корректно работает в актуальных версиях не менее двух браузерных движков или в средах, согласованных с преподавателем;
- страницы адаптируются как минимум к мобильной и настольной ширине экрана;
- используется семантическая разметка, обеспечивается клавиатурная навигация и понятные текстовые альтернативы для значимых изображений;
- пользовательский ввод валидируется; ошибки отображаются понятно и не приводят к потере данных;
- секреты, пароли и ключи API не включаются в исходный код и репозиторий; внешние данные обрабатываются с учётом возможных ошибок сети;
- проект запускается по инструкции из чистой копии архива или репозитория; зависимости и команды сборки зафиксированы;
- история изменений ведётся средствами системы контроля версий; итоговая версия имеет однозначную отметку или архив.

4.3. Заимствования и использование искусственного интеллекта

Курсовой проект проходит проверку на наличие некорректных заимствований в соответствии с локальными актами университета. Для программ бакалавриата степень оригинальности текста пояснительной записки должна быть не ниже 25%, если рабочей программой дисциплины или локальными актами не установлено более строгое требование.

Использование систем искусственного интеллекта допускается только в объёме, согласованном с ведущим преподавателем. Обучающийся обязан указать во введении, для каких задач использовалась такая система, проверить полученный результат, привести ссылки или примечания в соответствующих местах и сохранять способность объяснить и самостоятельно изменить весь представленный код и текст. Генерация материала не освобождает от ответственности за ошибки, лицензии, безопасность и достоверность.

5. Список рекомендованных основных и дополнительных источников литературы

При подготовке проекта рекомендуется использовать действующие стандарты и официальную документацию. Для быстро меняющихся Web-технологий приоритет имеет актуальная версия источника на дату выполнения работы. В списке использованных источников указываются дата обращения и конкретные документы, реально использованные в проекте.

1. ECMA International. ECMAScript® 2026 Language Specification (ECMA-262). https://tc39.es/ecma262/2026/
2. WHATWG. HTML Living Standard. https://html.spec.whatwg.org/
3. WHATWG. DOM Living Standard. https://dom.spec.whatwg.org/
4. WHATWG. Fetch Living Standard. https://fetch.spec.whatwg.org/
5. W3C. CSS Snapshot 2024. https://www.w3.org/TR/css-2024/
6. W3C. Web Content Accessibility Guidelines (WCAG) 2.2. https://www.w3.org/TR/WCAG22/
7. IETF. RFC 9110: HTTP Semantics. https://www.rfc-editor.org/rfc/rfc9110
8. OWASP Foundation. OWASP Top 10: Web Application Security Risks. https://owasp.org/Top10/
9. OWASP Foundation. Application Security Verification Standard. https://owasp.org/www-project-application-security-verification-standard/
10. MDN Web Docs. Learn Web Development. https://developer.mozilla.org/en-US/docs/Learn_web_development
11. Node.js. Documentation. https://nodejs.org/docs/latest/api/
12. npm. Documentation. https://docs.npmjs.com/
13. Git. Reference documentation. https://git-scm.com/docs
14. Chrome for Developers. DevTools documentation. https://developer.chrome.com/docs/devtools/
15. web.dev. Performance and quality guidance. https://web.dev/learn/performance/

Дополнительно используются учебники, научные статьи, документация выбранных библиотек и фреймворков, а также материалы по предметной области проекта. Рекомендуемый список — не менее 10 источников, преимущественно опубликованных или актуализированных за последние 5 лет, за исключением базовых стандартов и классических работ.

6. Требования к защите работы

Законченный курсовой проект представляется преподавателю в сроки, установленные кафедрой. Для допуска к защите необходимы оформленная пояснительная записка, программный продукт, доступные исходные материалы, рецензия преподавателя (Приложение 2) и соблюдение требований к оригинальности.

Защита проводится в форме публичного выступления продолжительностью 5-7 минут, демонстрации приложения и ответов на вопросы. Рекомендуемый объём презентации — 5-7 содержательных слайдов.

В докладе должны быть отражены:

- актуальность темы, цель и ключевые требования;
- архитектура, данные и обоснование выбранных технологий;
- реализованные пользовательские сценарии и наиболее значимые технические решения;
- результаты тестирования, доступности, безопасности и производительности;
- выводы, ограничения и направления развития.

На демонстрации необходимо показать запуск проекта, основной сценарий, обработку ошибочного или граничного ввода и сохранение/получение данных. Обучающийся должен уметь найти соответствующий фрагмент исходного кода, объяснить его назначение и внести небольшое изменение по запросу преподавателя.

7. Критерии оценки работы

Выполненный и защищённый курсовой проект оценивается по балльно-рейтинговой системе с учётом рабочей программы дисциплины. Максимальная сумма — 100 баллов.

Критерий	Максимальное значение, баллов	Набрано, баллов
Оформление пояснительной записки и комплекта материалов	10	
Содержание и качество программного продукта	60	
Защита курсового проекта	30	
ИТОГО	100	

Группа	Показатель	Баллы
Оформление	Соблюдение структуры, параметров оформления, качества таблиц/рисунков, ссылок и электронного комплекта	10
Содержание	Анализ задачи, требования, сценарии и обоснование технологий	10
Содержание	Полнота и корректность реализованных функций, обработка данных и ошибок	20
Содержание	Архитектура, модульность, читаемость кода и управление зависимостями	10
Содержание	Интерфейс, адаптивность, доступность и удобство использования	10
Содержание	Тестирование, безопасность, производительность и воспроизводимость запуска	10
Защита	Структура доклада и качество демонстрации	10
Защита	Понимание архитектуры, технологий и собственного кода	10
Защита	Полнота и обоснованность ответов на вопросы	10

Полный балл за содержание выставляется, если требования реализованы полностью, приложение устойчиво работает в заявленных сценариях, решения обоснованы, исходный код структурирован, а выводы подтверждены тестами. При частичном выполнении, существенных дефектах, отсутствии обязательных разделов или невозможности воспроизвести запуск балл снижается пропорционально выявленным недостаткам.

Перевод суммы баллов в пятибалльную систему:

- 89-100 баллов — «отлично»;

- 77-88 баллов — «хорошо»;
- 65-76 баллов — «удовлетворительно»;
- 0-64 балла — «неудовлетворительно».

При неудовлетворительной оценке обучающийся имеет право на повторную защиту после доработки в порядке, установленном локальными актами университета. Несвоевременная сдача или отсутствие защиты без уважительной причины образует академическую задолженность.

Приложение 1

МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«СТАВРОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»

Факультет цифровых технологий

Кафедра информационных систем

КУРСОВОЙ ПРОЕКТ

по дисциплине «Web-программирование»

Тема: «_____»

Выполнил(а):
студент(ка) ___ курса группы _____
Ф.И.О. _____
Направление: 09.03.02
Форма обучения: _____

Проверил(а):
учёная степень, должность _____
Ф.И.О. _____

Зарегистрирован
«____» _____ 20____ г.

Критерий	Макс., баллов	Набрано
Оформление	10	
Содержание и программный продукт	60	
Защита	30	
ИТОГО	100	

Оценка «_____» Дата _____ Подпись _____

Ставрополь, 20____

Приложение 2

Кафедра информационных систем

РЕЦЕНЗИЯ

на курсовой проект по дисциплине «Web-программирование»

Тема

_____

Обучающийся (Ф.И.О.)

_____

Курс _____ Группа _____ Форма обучения

_____

Преподаватель (Ф.И.О.)

_____

Выполнение общих требований

№	Требование	Результат
1	Структура и объём пояснительной записки соответствуют требованиям	Да / нет
2	Комплект исходных материалов доступен и воспроизводимо запускается	Да / нет
3	Степень оригинальности текста соответствует установленному порогу	Да / нет; ____ %
4	Сведения об использовании ИИ и внешних материалов раскрыты	Да / нет / не применимо

Критерии оценивания

Критерий	Макс.	Комментарий преподавателя	Балл
Оформление и комплектность	10		
Анализ задачи и требования	10		
Функциональность и корректность	20		
Архитектура и качество кода	10		
Интерфейс, адаптивность и доступность	10		
Тестирование, безопасность и производительность	10		
Доклад и демонстрация	10		
Понимание решений и кода	10		
Ответы на вопросы	10		

Критерий	Макс.	Комментарий преподавателя	Балл
ИТОГО	100		

Рекомендации:

_____

_____

____

Ведущий преподаватель _____ /

_____

(подпись) (Ф.И.О.)

Приложение 3

Пример содержания основной части курсового проекта

Тема примера: «Разработка Web-приложения для управления учебными задачами»

1. Анализ задачи и требования

Приложение предназначено для обучающихся, которым необходимо фиксировать задания, сроки, приоритеты и состояние выполнения. Основные сценарии: создание задачи, изменение статуса, фильтрация, поиск, сохранение данных и получение напоминания о приближении срока. В проекте должны быть описаны ограничения, критерии приёмки и ожидаемое поведение при ошибочном вводе.

Роль	Сценарий	Критерий приёмки
Пользователь	Создаёт задачу	Обязательные поля проверяются; новая задача появляется в списке без перезагрузки страницы
Пользователь	Фильтрует задачи	Фильтры по статусу и сроку можно комбинировать; состояние интерфейса остаётся понятным
Пользователь	Изменяет статус	Изменение сохраняется; после повторного открытия отображается актуальное состояние
Пользователь	Работает при ошибке сети	Интерфейс показывает причину и предлагает безопасно повторить операцию

2. Проектирование и реализация

Архитектура выбирается с учётом задания. Для клиент-серверного варианта приложение может состоять из браузерного интерфейса, модулей состояния и API-клиента, серверного API и хранилища данных. Для клиентского варианта серверное API заменяется локальным хранилищем или внешним сервисом. Границы модулей и формат данных должны быть описаны явно.

Обобщённая архитектура Web-проекта

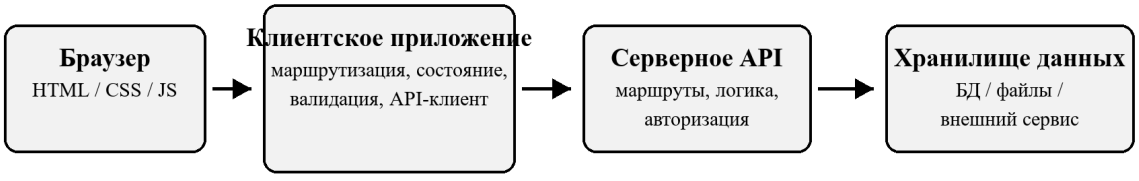

Рисунок 1 — Обобщённая архитектура Web-приложения

Метод	Маршрут	Назначение	Ожидаемый ответ
GET	/api/tasks	Получить список задач с параметрами фильтрации	200 и массив объектов
POST	/api/tasks	Создать задачу после валидации	201 и созданный объект
PATCH	/api/tasks/{id}	Изменить разрешённые поля	200 и обновлённый объект
DELETE	/api/tasks/{id}	Удалить задачу	204 без тела ответа

В пояснительной записке следует показать структуру каталогов, ключевые модели данных, правила валидации и обработку ошибок. Листинг включается только для нетривиального фрагмента, например функции нормализации ответа API или модуля управления состоянием:

```
async function loadTasks(filters) {
  const response = await fetch(buildUrl('/api/tasks', filters));
  if (!response.ok) throw new Error(`HTTP ${response.status}`);
  return response.json();
}
```

3. Тестирование и ввод в эксплуатацию

План тестирования строится по критериям приёмки и включает позитивные, негативные и граничные сценарии. Для каждого теста указываются начальные условия, действия, ожидаемый и фактический результат.

ID	Проверка	Ожидаемый результат	Факт
T-01	Создание задачи с корректными данными	Задача сохранена и отображается	
T-02	Пустое обязательное поле	Форма не отправлена, показана понятная ошибка	
T-03	Срок на границе текущей даты	Дата обработана по согласованному правилу	
T-04	Ошибка ответа API	Нет потери введённых данных; доступен повтор	
T-05	Управление только клавиатурой	Все функции доступны, фокус видим	
T-06	Ширина экрана 360 px	Нет горизонтальной прокрутки основного интерфейса	
T-07	Повторный запуск	Сохранённые данные восстановлены	

Перед защитой рекомендуется выполнить итоговую проверку:

- проект устанавливается и запускается по README;

- в репозитории отсутствуют секреты, временные файлы и каталоги зависимостей;
- основной пользовательский сценарий и обработка ошибок воспроизводятся;
- интерфейс проверен на мобильной и настольной ширине, клавиатурой и автоматизированными средствами браузера;
- результаты тестов, ограничения и известные дефекты отражены в записке;
- опубликованная версия соответствует версии исходного кода, представленной на защиту.

Приложение 3 является примером структуры и глубины раскрытия материала. Названия разделов, маршруты, модели и тесты должны быть заменены сведениями конкретного курсового проекта.