**1 ОБЗОР ЛИТЕРАТУРЫ**

**1.1 Обзор аналогов**

В данном подразделе рассмотрим существующие аналоги реализуемого программного обеспечения.

**1.1.1 SpaceNews**

SpaceNews [1] (см. рисунок 1.1) – это источник новостей и аналитических материалов о компаниях, агентствах, технологиях и тенденциях, формирующих мировую космическую отрасль.

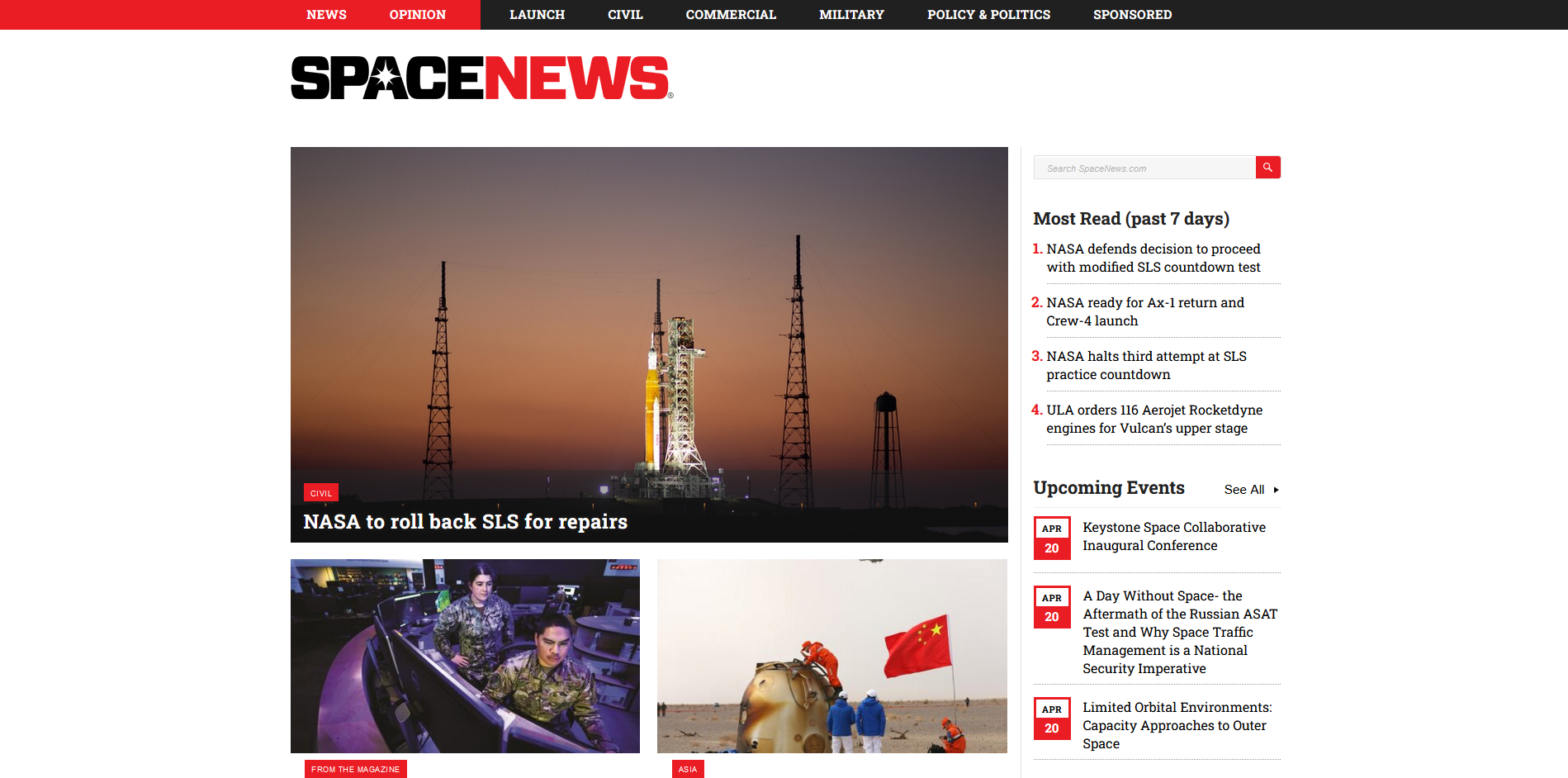


Рисунок 1.1 – Главная страница SpaceNews [1]

На протяжении более трех десятилетий SpaceNews остается единственным изданием, к которому первыми обращаются космические профессионалы во всем мире. Будь то последние тенденции в военных космических возможностях, разработках в области спутниковой связи или текущем состоянии бюджета, можно рассчитывать на SpaceNews. Этот сайт обеспечивает всестороннее освещение космической отрасли и критическую точку зрения, на которую пользователь может положиться, благодаря новостям, комментариям и анализу. Можно выделить явные достоинства и недостатки.

Достоинства:

* удобный и простой интерфейс, сайт легкий для восприятия;
* широкая функциональность;
* адаптация для мобильных устройств.

Недостатки:

* существует платная подписка;
* отсутствие русской локализации.

**1.1.2 SpaceFlight Now**

SpaceFlight Now [2] (см. рисунок 1.2) – веб-приложение компаньон, работающее на SpaceX API и Reddit. Предназначено для отслеживания информации о запусках SpaceX для космических энтузиастов.

Является онлайн-порталом космических новостей, на котором публикуются текущие новости, текущие миссии, предстоящие запуски и архивные статьи. Spaceflight Now обеспечивает непревзойденное ежедневное освещение космической программы. С момента своего запуска в декабре 1999 года он стал одним из самых надежных источников точных и последних новостей для космического сообщества. Существенным недостатком данного сайта является отсутствие оптимизации для использования в мобильной версии, что ухудшает удобство просмотра сайта.

Исходя из предложенных ранее фактов, резюмируем достоинства и недостатки данного ресурса.

Достоинства:

- не требует регистрации;

- условно бесплатный;

- простой интерфейс, сайт легкий для восприятия.

Недостатки:

- отсутствие адаптации для мобильных устройств;

- ограниченная функциональность.

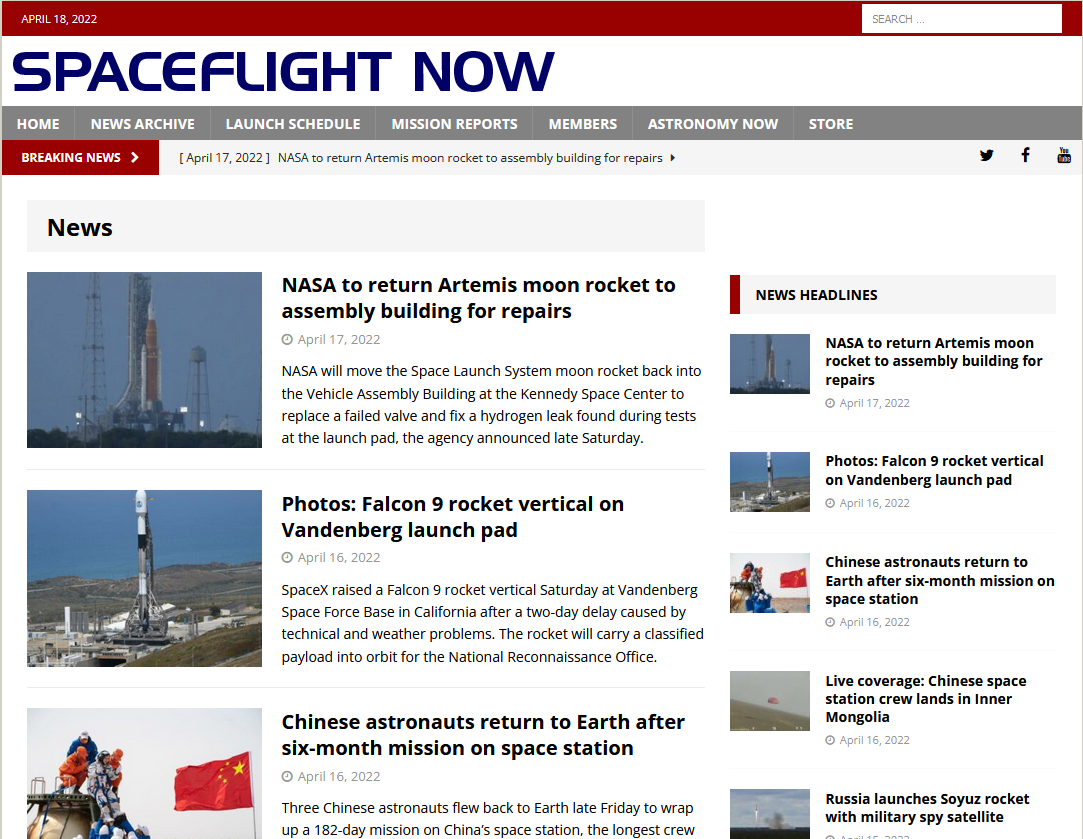


Рисунок 1.2 – Главная страница SpaceFlight Now [2]

**1.2 Обзор технологий**

Разрабатываемая в рамках данного дипломного проекта программная система является веб-приложением. Под веб-приложением понимается клиент-серверное приложение, в котором роль клиента играет браузер, а сервером является веб-сервер, который может хранится в облачном хранилище. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети. Благодаря такому подходу клиент может не зависеть от конкретной операционной системы. Из этого следует, что веб-приложения являются кроссплатформенными сервисами.

В качестве способа взаимодействия сайта и веб-приложения с сервером используется архитектура REST API.

API — это интерфейс прикладного программирования. Это набор правил, которые позволяют программам общаться друг с другом. Разработчик создает API на сервере и позволяет клиенту общаться с ним.

REST определяет, как выглядит API. Это набор правил, которым следуют разработчики при создании своего API. Одно из этих правил гласит, что вы должны иметь возможность получать фрагмент данных (называемый ресурсом) при ссылке на определенный URL-адрес.

Каждый URL-адрес называется запросом, а данные, отправленные вам, называются ответом.

Данные языки программирования чаще всего используются для разработки веб-сервера: C#, Ruby, Java, JavaScript, Python.

**1.2.1 Архитектура клиент-сервер**

Архитектура клиент-сервер — это вычислительная модель, в которой сервер размещает, доставляет и управляет большей частью ресурсов и услуг, потребляемых клиентом. В этом типе архитектуры один или несколько клиентских компьютеров подключены к центральному серверу по сети или через Интернет [3] (см. рисунок 1.3). Эта система разделяет вычислительные ресурсы. Архитектура клиент-сервер также известна как модель сетевых вычислений или сеть клиент-сервер, поскольку все запросы и услуги доставляются по сети.

Архитектура клиент-сервер — это архитектура компьютерной сети, в которой множество клиентов запрашивают и получают услуги от централизованного сервера.

Клиентские компьютеры предоставляют интерфейс, позволяющий пользователю компьютера запрашивать услуги сервера и отображать результаты, возвращаемые сервером.

Серверы ждут поступления запросов от клиентов и затем отвечают на них.

Многие клиенты могут одновременно получать доступ к информации сервера, и в то же время клиентский компьютер может выполнять другие задачи.



Рисунок 1.3 – Архитектура клиент-сервер [3]

Основные характеристики клиент-серверной архитектуры:

* горизонтальная масштабируемость и вертикальная масштабируемость;
* клиентское или серверное приложение напрямую взаимодействует с протоколом транспортного уровня для установления связи и отправки или получения информации;
* затем транспортный протокол использует протоколы нижнего уровня для отправки или получения отдельных сообщений. Таким образом, компьютеру необходим полный стек протоколов для запуска клиента или сервера;
* клиентские и серверные машины нуждаются в разном количестве аппаратных и программных ресурсов;
* клиентские и серверные машины могут принадлежать разным поставщикам;
* один компьютер серверного класса может одновременно предлагать несколько услуг; для каждой службы требуется отдельная серверная программа.

Преимущества:

* общие ресурсы среди разных платформ;
* совместная работа с данными;
* возможности обработки данных, несмотря на местоположение;

простота обслуживания;

* улучшенный обмен данными;
* интеграция услуг;
* безопасность.

Недостатки:

* перегруженные серверы: при частых одновременных клиентских запросах серверы сильно перегружаются;

влияние централизованной архитектуры: поскольку она централизована, в случае сбоя критического сервера запросы клиентов не выполняются.

**1.2.2 Шаблон проектирования REST API**

REST API [4] (также известный как RESTful API) — это интерфейс прикладного программирования, который соответствует ограничениям архитектурного стиля REST и позволяет взаимодействовать с веб-службами RESTful. REST означает передачу репрезентативного состояния и был создан компьютерным ученым Роем Филдингом.

API — это набор определений и протоколов для создания и интеграции прикладного программного обеспечения. Иногда его называют договором между поставщиком информации и пользователем информации, устанавливающим контент, требуемый от потребителя, и контент, требуемый производителем.

Другими словами, API помогает пользователю передать то, что он хочет, этой системе, чтобы она могла понять и выполнить запрос.

API выполняет роль посредника между пользователями или клиентами и ресурсами или веб-сервисами, которые они хотят получить. Это также способ для организации обмена ресурсами и информацией, который помогает сохранить при этом безопасность, контроль и аутентификацию, определяя, кто и к чему получает доступ.

Еще одно преимущество API заключается в том, что вам не нужно знать особенности кэширования - как извлекается ваш ресурс или откуда он берется.

REST — это набор архитектурных ограничений, а не протокол или стандарт. Когда клиентский запрос выполняется через RESTful API, он передает представление о состоянии ресурса запрашивающей стороне или конечной точке. Эта информация или представление доставляется в одном из нескольких форматов через HTTP: JSON (обозначение объектов Javascript), HTML, XLT, Python, PHP или обычный текст. JSON является наиболее популярным форматом файлов для использования, потому что, несмотря на свое название, он не зависит от языка, а также удобен для чтения как людьми, так и машинами.

Чтобы API считался RESTful, он должен соответствовать следующим критериям:

* архитектура клиент-сервер, которая состоит из клиентов, серверов и ресурсов, с запросами, управляемыми через HTTP;
* связь клиент-сервер;
* кэшируемые данные, которые оптимизируют взаимодействие клиент-сервер.

В отличие от этого, REST — это набор рекомендаций, которые можно реализовать по мере необходимости, что делает REST API более быстрыми и легкими, с повышенной масштабируемостью - идеально подходит для мобильных веб-приложений.

**1.2.3 .NET**

.NET [5] — это бесплатная кроссплатформенная платформа для разработчиков с открытым исходным кодом для создания различных типов приложений. В .NET используются несколько языков, редакторов и библиотек для создания приложений для Интернета, мобильных устройств, компьютеров, игр и Интернета вещей.

Независимо от того, работаете ли вы на C#, F# или Visual Basic, ваш код будет выполняться в любой совместимой ОС. Различные реализации .NET справятся с тяжелой работой за вас:

.NET — это кроссплатформенная реализация .NET для веб-сайтов, серверов и консольных приложений в Windows, Linux и macOS.

.NET Framework поддерживает веб-сайты, службы, классические приложения и многое другое в Windows.

Xamarin/Mono — это реализация .NET для запуска приложений во всех основных мобильных операционных системах.

.NET Standard — это базовый набор API, общий для всех реализаций .NET. Каждая реализация также может предоставлять дополнительные API, специфичные для операционных систем, в которых она работает. Например, .NET Framework — это реализация .NET только для Windows, которая включает API для доступа к реестру Windows.

Чтобы расширить функциональность, Microsoft и другие компании поддерживают здоровую экосистему пакетов, построенную на .NET Standard. NuGet — это диспетчер пакетов, созданный специально для .NET и содержащий более 90 000 пакетов.

.NET имеет открытый исходный код и находится под управлением .NET Foundation. .NET Foundation — это независимая организация, способствующая открытой разработке и сотрудничеству в рамках экосистемы .NET.

**1.2.4 Angular**

Angular [6] - один из самых мощных интерфейсных фреймворков с открытым исходным кодом, построенный на TypeScript. Разработанный Google в 2010 году для создания динамичных и современных одностраничных приложений (SPA), он сначала начинался как среда JS, которая позже была преобразована в совершенно новую структуру TypeScript.

Angular — это современная структура и платформа MVVC. Он имеет огромное количество функций, включая двустороннюю привязку, внедрение зависимостей, RESTful API и обработку AJAX.

Angular включает в себя:

* компонентную среду для создания масштабируемых веб-приложений;
* коллекцию хорошо интегрированных библиотек, которые охватывают широкий спектр функций, включая маршрутизацию, управление формами, взаимодействие клиент-сервер и многое другое;
* набор инструментов разработчика, которые помогают разрабатывать, создавать, тестировать и обновлять код.

С Angular [7] можно масштабироваться от проектов с одним разработчиком до приложений корпоративного уровня.

Это всего лишь набор преимуществ, которые Angular предоставляет разработчикам. Angular - один из лучших вариантов для создания внешнего интерфейса, но он не единственный на рынке.

Хотя Angular полезен для создания небольших приложений, он предпочтительнее для больших приложений. Поддержка со стороны Google и его сообщества веб-разработчиков позволяет Angular быстро развиваться и быстро внедряться. Кроме того, его основной особенностью является использование языка Typescript, который совместим со всеми устройствами, операционными системами и браузерами.

**1.2.5 С#**

В качестве языка для написания веб-сервера данного проекта используется язык C# [8].

C# — это универсальный, современный и объектно-ориентированный язык программирования, произносимый как “C Sharp”. Он был разработан Microsoft во главе с Андерсом Хейлсбергом и его командой в рамках инициативы .NET и был одобрен Европейской ассоциацией производителей компьютеров (ECMA) и Международной организацией стандартов (ISO). C# является одним из языков для инфраструктуры общего языка. C# очень похож на Java синтаксически и прост для пользователей, которые знают C, C++ или Java.

Интересно, что Хейлсберг-выдающийся инженер Microsoft, который создал другие продукты и языки, включая Borland Turbo C++ и Borland Delphi. С C# они сосредоточились на том, чтобы взять то, что было правильно в существующих языках, и добавить улучшения, чтобы сделать что-то лучше.

C# - мощный и гибкий язык программирования. Как и все языки программирования, он может быть использован для создания различных приложений. Язык не накладывает ограничений на то, что можно реализовать. C# используется для таких разнообразных проектов, как динамические веб-сайты, инструменты разработки и компиляторы.

C# был создан как язык объектно-ориентированного программирования (ООП). Другие языки программирования включают объектно-ориентированные функции, но очень немногие из них полностью объектно-ориентированы.

**1.2.6 TypeScript**

Для разработки пользовательского интерфейса используется язык программирования TypeScript [9].

TypeScript был создан Microsoft и был выпущен в 2012 году после двух лет разработки. Он был создан для обеспечения дополнительной статической проверки типов, что было бы особенно полезно при разработке крупномасштабных приложений. Одна из причин, по которой Microsoft разработала TypeScript, заключалась в том, что их внутренние команды испытывали проблемы с масштабированием JavaScript для собственных проектов Microsoft.

TypeScript является открытым исходным кодом, и код написан на самом TypeScript. TypeScript является обратно совместимым с JavaScript и компилируется в последний. Будучи надмножеством, TypeScript обладает всеми функциями JavaScript, а также некоторыми дополнительными функциями.

JavaScript динамически типизируется. Поэтому программы, написанные на JavaScript, не знают тип данных переменной до тех пор, пока этой переменной не будет присвоено значение во время выполнения. Переменная может быть переназначена или преобразована в значение другого типа без каких-либо проблем или предупреждений. Это может привести к ошибкам, которые часто упускаются из виду, особенно в больших приложениях.

TypeScript использует статическую типизацию. Переменным может быть присвоен тип, когда они объявлены. TypeScript проверяет типы во время компиляции и выдает ошибку, если переменной когда-либо присваивается значение другого типа.

TypeScript [10] поставляется с такими функциями, как улучшенные инструменты времени разработки, статический анализ кода, проверка типов во время компиляции и документация на уровне кода.

Все эти функции, которые поставляются с TypeScript, делают его идеальным языком программирования для создания крупномасштабных приложений JavaScript.

**1.2.7 База данных**

В качестве системы управления базами данных будет использоваться PostgreSQL.

PostgreSQL [11] — свободная объектно-реляционная система управления базами данных (СУБД). PostgreSQL является одной из наиболее популярных систем управления базами данных. Является мощной системой объектно-реляционных баз данных с открытым исходным кодом, активно разрабатывавшаяся более 30 лет и заслужившая прочную репутацию за надежность и производительность. Текущей версией является версия 14.2. Однако регулярно также выходят подверсии. PostgreSQL поддерживается для всех основных операционных систем – Windows, Linux, MacOS.

Основными достоинствами PostgreSQL являются:

* надежность;
* производительность (основывается на использовании индексов, интеллектуальном планировщике запросов, тонкой системы блокировок, системе управления буферами памяти и кэширования, превосходной масштабируемости при конкурентной работе);
* расширяемость (означает, что пользователь может настраивать систему путем определения новых функций, агрегатов, типов, языков, индексов и операторов);
* поддержка SQL;
* поддержка JSON;
* богатый набор типов данных.

**1.3 Постановка задачи**

Сегодня рынок программного обеспечения выставляет высокие требования ко всем разрабатываемым программным продуктам. Для современных программных средств важными требованиями являются масштабируемость, мультиплатформенность и переносимость. Использование вышеперечисленных технологий при разработке дипломного проекта позволяет сократить время на разработку, увеличить качество кода и за счет этого выполнить данные требования

Объектом исследования является процесс создания объектов и обработки данных, поступающих на сервер. Предметом исследования является создание программного комплекса, обеспечивающего полноценную работу веб-приложения.  
 Основными требованиями, которые были заложены в основу при разработке программного комплекса дипломного проекта стали: простота использования и расширяемость.

На основании вышесказанного, для разработки программного продукта были определены следующие задачи:

* разработка структуры БД;
* разработка серверной части;
* разработка клиентской части.

Веб-приложение будет представлять из себя сайт, который будет предоставлять следующие функции:

* регистрация пользователей в системе;
* авторизация пользователя в системе;
* создание личного диалога с пользователем;
* добавление комментариев и личной информации пользователя.