ВЕБ-РАЗРАБОТКА И СИСТЕМЫ ЭЛЕКТРОННОГО УЧЁТА ПРИЁМА ВРАЧЕЙ: ОБЗОР ТЕХНОЛОГИЙ И ПОДХОДОВ

Атаханов Мухаммаджон Фахриддин угли

Ургенчский государственный университет имени Абу Райхана Беруни
E-mail: muxammad.a@urdu.u

K E Y W O R D S

веб-разработка, электронная запись на приём, медицинские информационные системы, REST API, HL7 FHIR, информационная безопасность, UX/UI, микросервисная архитектура

A B S T R A C T

В работе проведён систематический обзор технологий и подходов к разработке веб-систем электронного учёта приёма врачей. Методологическую основу составили сравнительный анализ и контент-анализ 10 литературных и нормативных источников 2015–2024 гг. Выявлены четыре категории систем и их сравнительные характеристики, проанализированы доминирующие архитектурные паттерны (REST, микросервисы, событийно-ориентированный подход), технологические стеки и требования к информационной безопасности. Сформулированы функциональные и нефункциональные требования к системам. Установлено, что для малых учреждений оптимальна монолитная архитектура, для региональных платформ — микросервисная. Научная новизна заключается в системной классификации существующих решений и разработке комплекса практических рекомендаций с учётом нормативно-правовых требований Республики Узбекистан

Введение

Цифровизация здравоохранения является одним из ключевых направлений развития информационного общества. По данным Всемирной организации здравоохранения, к 2025 году более 70% медицинских учреждений в развитых странах перешли на электронные системы учёта пациентов, что позволило сократить административные издержки на 25–40%. В Республике Узбекистан данное направление закреплено нормативно-правовыми актами государственного уровня **[10]**. Согласно Стратегии «Узбекистан — 2030», цифровизация системы здравоохранения определена как приоритетная задача, а развитие электронных государственных услуг в сфере медицины включено в перечень ключевых индикаторов.

В условиях роста численности населения Узбекистана — по данным Государственного комитета по статистике, в 2024 году численность населения превысила 38 млн человек — традиционные способы ведения записи пациентов становятся неэффективными. Бумажные журналы учёта, телефонные очереди и ручная обработка данных приводят к значительным потерям времени как для пациентов, так и для медицинского персонала **[5]**. Исследования показывают, что средний пациент тратит от 30 до 90 минут ожидания в очереди регистратуры при наличии предварительной записи по телефону, тогда как веб-системы сокращают это время до 5–10 минут.

Системы электронного учёта приёма врачей (СЭУПВ) — программные решения, автоматизирующие запись на приём, управление расписанием, ведение медицинской документации и взаимодействие между пациентами и врачами **[6]**. Веб-технологии обеспечивают доступность таких систем с любого устройства без необходимости установки специального программного обеспечения, кроссплатформенность и централизованное хранение данных **[4]**. Несмотря на наличие

(cc) BY

ряда готовых решений, разработчики и заказчики сталкиваются с дефицитом систематизированных рекомендаций по выбору технологического стека, архитектурных паттернов и мер обеспечения безопасности применительно к задачам учёта приёма врачей **[1, 6]**.

Теоретической основой исследования служат концепция информационных систем здравоохранения (Health Information Systems), модель жизненного цикла разработки программного обеспечения (SDLC) и принципы пациентоориентированного проектирования (Patient-Centered Design). В сфере медицинских информационных систем особое значение приобретают требования к интероперабельности — способности систем обмениваться данными без потери их семантической точности, — что непосредственно определяет выбор архитектуры и стандартов.

Цель исследования — провести систематический обзор технологий и подходов к разработке веб-систем электронного учёта приёма врачей, выявить их преимущества и ограничения, сформулировать практические рекомендации. В рамках исследования поставлены следующие вопросы: (RQ1) — какие типы СЭУПВ существуют и в чём их принципиальные отличия? (RQ2) — какие архитектурные паттерны и технологические стеки применяются при разработке СЭУПВ? (RQ3) — каковы требования к информационной безопасности и правовому соответствию? (RQ4) — каков необходимый набор функциональных и нефункциональных требований к подобным системам?

Методологическую основу составили: сравнительный анализ существующих систем по ключевым критериям **[1, 4, 6]**, контент-анализ технической документации, нормативных актов и описаний систем **[5, 6]**, а также нормативный анализ — сопоставление технических решений с требованиями законодательства и отраслевых стандартов **[7, 8, 9]**. Поиск источников проводился в базах данных Scopus, IEEE Xplore, Google Scholar, eLibrary.ru за период 2015–2024 гг. Критерии включения: рецензируемые статьи и технические отчёты, официальные стандарты и нормативные акты. Критерии исключения: материалы до 2015 года, дублирующие и нерелевантные источники. Объект исследования — технологии разработки веб-систем электронного учёта приёма врачей. Предмет — архитектурные паттерны, технологические стеки и требования к безопасности. Ограничение работы состоит в опоре преимущественно на вторичные данные.

Результаты

Классификация существующих систем (RQ1)

На основе обзора литературы **[1, 6]** выявлены четыре категории СЭУПВ. Каждая категория ориентирована на определённый тип пользователей и отличается принципиальными характеристиками с точки зрения стоимости, гибкости, безопасности и интеграционных возможностей:

- Государственные платформы — интегрированные решения национального масштаба (ЕГИСЗ в России, E-Health на Украине, MedElement в Казахстане). Финансируются из государственного бюджета, обязательны для государственных учреждений. Обеспечивают высокий уровень безопасности и интеграцию с ГИС, однако отличаются низкой гибкостью и длительными циклами обновления;
- Коммерческие SaaS-решения — платформы по модели подписки (DocDoc, Prodoctorov, MEDODS, iDoctor). Быстрое развёртывание без капитальных затрат, ответственность за безопасность лежит на вендоре. Ограничения: зависимость от поставщика, недостаточная кастомизация для специфических процессов учреждения;
- Корпоративные решения — системы, разрабатываемые под конкретные учреждения или сети клиник. Обеспечивают максимальную гибкость и соответствие уникальным бизнес-процессам, однако требуют значительных первоначальных инвестиций и наличия собственной IT-команды;
- Открытые проекты (Open Source) — решения с открытым исходным кодом (OpenMRS, OpenEMR, GNU Health, Bahmni). Нулевая стоимость лицензии,

(cc) BY

возможность полной адаптации, однако требуют существенных затрат на внедрение, настройку и сопровождение.

Сравнительный анализ по семи ключевым критериям **[1, 4, 6]** представлен в таблице 1. Расширенная версия таблицы включает дополнительные критерии — время внедрения и тип поддержки — которые имеют практическое значение при выборе решения.

Таблица 1.

Сравнительный анализ типов систем электронного учёта приёма врачей [1, 4, 6]

Критерий	Гос. платформы	SaaS	Корпоративные	Open Source
Стоимость	Бесплатно	Подписка	Высокая	Бесплатно
Гибкость	Низкая	Средняя	Высокая	Высокая
Безопасность	Высокая	Средняя	Высокая	Средняя
Интеграция с ГИС	Встроена	Частично	Настраиваемая	Ручная
Масштабируемость	Высокая	Высокая	Средняя	Средняя
Время внедрения	Долгое	Быстрое	Долгое	Среднее
Поддержка вендора	Гос. орг.	Коммерческая	Внутренняя	Сообщество

Архитектурные паттерны (RQ2)

Анализ существующих решений выявил три доминирующих архитектурных паттерна. Первый — **клиент-серверная монолитная архитектура** — остаётся наиболее распространённым для систем малого и среднего масштаба **[6]**. Всё приложение развёртывается как единый артефакт, что упрощает разработку и отладку. Наибольшее распространение получил архитектурный стиль REST (Representational State Transfer): взаимодействие клиента и сервера через HTTP-запросы обеспечивает слабую связность компонентов, простоту масштабирования и независимость от технологического стека. Альтернативой является GraphQL — позволяет клиенту декларативно запрашивать только необходимые данные, что критично для мобильных клиентов с ограниченной пропускной способностью канала **[4]**.

Второй паттерн — **микросервисная архитектура** — применяется для крупных региональных и федеральных платформ **[1, 4]**. Система разбивается на независимые сервисы с чёткими зонами ответственности: сервис управления записями, сервис расписания, сервис уведомлений, сервис аутентификации, сервис медицинской документации. Каждый микросервис разрабатывается, тестируется и масштабируется независимо. Оркестрация обеспечивается Docker и Kubernetes; межсервисное взаимодействие реализуется через Apache Kafka или RabbitMQ в асинхронном режиме, либо gRPC — в синхронном. API Gateway выступает единой точкой входа для клиентских запросов.

Третий паттерн — **событийно-ориентированная архитектура (Event-Driven Architecture, EDA)** — применяется в системах с высокой нагрузкой и требованиями к реактивности **[4]**. При поступлении записи пациента генерируется событие, которое асинхронно обрабатывается подписанными сервисами: уведомительным, аналитическим, интеграционным. Такой подход обеспечивает горизонтальное масштабирование и отказоустойчивость, однако значительно усложняет отладку и трассировку транзакций.

Технологический стек (RQ2)

Клиентская часть реализуется на JavaScript-фреймворках: React.js (компонентная модель, Virtual DOM, развитая

(cc) BY

экосистема), Vue.js (низкий порог вхождения, двустороннее связывание), Angular (строгая типизация через TypeScript, встроенный DI) **[5]**. Для построения адаптивного интерфейса используются CSS-фреймворки: Tailwind CSS, Bootstrap, Material UI. Обязательным требованием является соответствие стандарту WCAG 2.1 уровня AA — система обслуживает разнородную аудиторию, включая пожилых пациентов и лиц с ограниченными возможностями **[5, 6]**. Управление состоянием приложения обеспечивается Redux (для React) или Pinia (для Vue).

Серверная часть реализуется на: Node.js с фреймворками Express или NestJS — высокая пропускная способность при обработке множества одновременных соединений; Python с Django или FastAPI — быстрая разработка, интеграция с ML-библиотеками; Java (Spring Boot) или C# (ASP.NET Core) — строгая типизация, зрелые экосистемы для корпоративных решений **[4, 6]**. Аутентификация реализуется по протоколу OAuth 2.0 с токенами JWT; авторизация — через RBAC-модель. DevOps-процессы выстраиваются на базе GitLab CI/CD или GitHub Actions, что обеспечивает автоматизированное тестирование и безопасное развёртывание.

Выбор СУБД определяется характером хранимых данных **[1]**: реляционные PostgreSQL и MySQL — для структурированных медицинских записей, обеспечивая транзакционность, целостность и поддержку сложных запросов; MongoDB — для хранения неструктурированных данных (JSON-результаты анализов, DICOM-изображения, произвольные медицинские формы); Redis — для кэширования часто запрашиваемых данных расписания и хранения пользовательских сеансов; Elasticsearch — для полнотекстового поиска по медицинским картам и журналам событий. Хранилище файлов (изображения, PDF-документы) реализуется через объектное хранилище S3-совместимого типа (MinIO или облачные решения).

Безопасность и правовое соответствие (RQ3)

Медицинские данные относятся к категории особо чувствительной персональной информации. Их обработка регулируется: GDPR (ЕС, 2018), HIPAA (США, 1996), Законом Республики Узбекистан «О персональных данных» № ЗРУ-547 **[9]** и Постановлением Президента РУз о развитии здравоохранения **[10]**. Нарушения требований влекут административную и уголовную ответственность, а также угрозу репутации учреждения. Штрафы за нарушение GDPR могут достигать 4% годового оборота организации, что наглядно демонстрирует серьёзность регуляторного риска **[1]**.

Обзор выявил следующий необходимый набор технических мер информационной безопасности:

1. шифрование данных при хранении с использованием алгоритма AES-256 и при передаче по протоколу TLS 1.3 **[3]**;
2. многофакторная аутентификация (MFA) для медицинского персонала, включая аппаратные токены для привилегированных пользователей **[2]**;
3. ролевая модель управления доступом (RBAC) с детализацией прав до уровня отдельных полей медицинской карты **[2]**;
4. ведение неизменяемого журнала аудита всех действий с медицинскими данными с хранением не менее 5 лет;
5. защита от угроз OWASP Top 10, включая SQL-инъекции, XSS, CSRF, небезопасную десериализацию **[8]**;
6. регулярное резервное копирование с тестированием восстановления (DR) и RTO не более 4 часов;
7. регулярное проведение пентестов и автоматизированное сканирование уязвимостей в CI/CD-пайплайне.

Стандарт HL7 FHIR (Fast Healthcare Interoperability Resources) версии 4.0 **[7]** де-факто является основой семантической совместимости при обмене медицинскими данными. FHIR определяет унифицированные форматы ресурсов (Patient, Appointment, Practitioner, Schedule, Slot) и RESTful API для их передачи. Интеграция со стандартом FHIR позволяет СЭУПВ взаимодействовать с лабораторными системами, аптечными сервисами, страховыми платформами и

(cc) BY

государственными регистрами без дополнительных адаптеров.

Функциональные и нефункциональные требования (RQ4)

На основе контент-анализа литературы **[1, 5, 6]** и существующих реализаций выявлены функциональные требования, сгруппированные по ролям:

- Пациент: регистрация и авторизация; поиск врача с фильтрацией по специальности, учреждению, времени и рейтингу; онлайн-запись с выбором слота; получение подтверждений и напоминаний по email/SMS/push; отмена и перенос записи; просмотр истории визитов;
- Врач: просмотр и управление расписанием; доступ к краткой информации о пациенте перед приёмом; внесение результатов приёма в электронную медицинскую карту; управление шаблонами назначений;
- Регистратор: запись пациента от имени учреждения; управление очерёдностью; обработка телефонных заявок с конвертацией в электронную запись;
- Администратор: управление структурой учреждения и справочниками; настройка расписания работы учреждения; формирование отчётности; управление правами доступа пользователей.

Наряду с функциональными, выявлены критически важные нефункциональные требования:

- Производительность: время отклика системы на запрос пользователя — не более 2 секунд при нагрузке до 1000 одновременных пользователей; время загрузки расписания врача — не более 1 секунды;
- Доступность: уровень доступности системы — не менее 99,5% (SLA), что допускает не более 44 часов простоя в год; проведение плановых работ — только в ночное время;
- Масштабируемость: горизонтальное масштабирование без изменения архитектуры при росте числа пользователей в 10 раз;
- Мобильность: адаптивный интерфейс, корректно отображающийся на экранах от 320px; Core Web Vitals — LCP < 2,5с, FID < 100мс, CLS < 0,1;
- Локализация: поддержка узбекского, русского и английского языков интерфейса; совместимость с кириллицей и латиницей.

Обсуждение

Результаты обзора показывают, что ни одна из рассмотренных категорий систем не является универсальной **[1, 6]**. Выбор решения определяется совокупностью факторов: масштабом учреждения, доступным бюджетом, требованиями к интеграции и регуляторным контекстом. Таблица 1 наглядно демонстрирует, что государственные платформы, несмотря на высокую безопасность и нулевую лицензионную стоимость, характеризуются низкой гибкостью и длительным временем внедрения, что делает их неприемлемыми для частных клиник с уникальными процессами.

С экономической точки зрения, SaaS-решения демонстрируют наименьшую совокупную стоимость владения (TCO) на горизонте 1–3 лет, однако при долгосрочном использовании (5+ лет) корпоративные решения могут оказаться выгоднее — при условии наличия компетентной IT-команды. Open Source-решения занимают промежуточную позицию: отсутствие лицензионных платежей компенсируется затратами на внедрение и кастомизацию **[4, 6]**. Для учреждений Узбекистана с ограниченным IT-бюджетом оптимальным может оказаться гибридный подход: Open Source-платформа (OpenEMR или OpenMRS) с коммерческой поддержкой от локального интегратора.

Для небольших медицинских учреждений (до 10 врачей, до 200 записей в день) оптимальна монолитная архитектура на Django/Laravel с PostgreSQL **[4, 6]** — низкие операционные затраты и минимальная инфраструктурная сложность при достаточной производительности. Для средних учреждений (10–50 врачей) рекомендована модульная монолитная архитектура с вертикальным масштабированием. Для региональных платформ (50+ учреждений, миллионы пользователей) предпочтительна полноценная микросервисная архитектура с оркестрацией

(cc) BY

Kubernetes **[1, 4]**. Frontend следует реализовывать на React.js или Vue.js с акцентом на мобильную адаптивность: по данным аналитических отчётов, в странах Центральной Азии более 65% интернет-трафика формируется мобильными устройствами.

Отдельного внимания заслуживает вопрос удобства использования (UX). Исследование **[5]** показывает, что недостаточная юзабилити медицинских информационных систем является одной из главных причин их отторжения медицинским персоналом. Врачи в среднем тратят до 2 часов рабочего времени в день на ввод данных в систему, что приводит к профессиональному выгоранию. Поэтому проектирование СЭУПВ должно начинаться с детального анализа пользовательских задач (Task Analysis) и прототипирования с участием конечных пользователей.

Критически важным аспектом является соответствие стандарту HL7 FHIR **[7]** и требованиям законодательства о защите персональных данных **[9]**. Пренебрежение этими аспектами на начальных этапах проектирования неизменно ведёт к затратным доработкам. Особую роль играет подготовка персонала: даже технически совершенная система не обеспечит ожидаемого эффекта без систематического обучения пользователей и управления изменениями в организации.

Ограничение настоящего исследования состоит в том, что работа опирается преимущественно на вторичные данные и описания систем, представленные в публичных источниках. Это может создавать смещение в сторону широко известных решений. Эмпирическое сравнение производительности и юзабилити конкретных реализаций остаётся за рамками данного обзора и представляет перспективу для дальнейших исследований. Дополнительно следует отметить, что стремительное развитие технологий (в частности, появление LLM-ассистентов для медицинской документации) может изменить часть приведённых рекомендаций в краткосрочной перспективе.

Перспективными направлениями дальнейших исследований являются: применение алгоритмов машинного обучения для интеллектуального управления расписанием и прогнозирования нагрузки на врачей; разработка и стандартизация модулей телемедицинских консультаций с поддержкой видеосвязи; интеграция с носимыми медицинскими устройствами (IoT-сенсоры, умные часы, тонометры с Bluetooth); использование больших языковых моделей для автоматизации заполнения медицинских карт по итогам приёма.

Заключение

В данной работе проведён систематический обзор технологий и подходов к разработке веб-систем электронного учёта приёма врачей. Получены ответы на все поставленные исследовательские вопросы. По RQ1 выявлены четыре категории решений — государственные, SaaS, корпоративные и Open Source — и проведён их сравнительный анализ по семи критериям **[1, 6]**. По RQ2 описаны три доминирующих архитектурных паттерна (монолит с REST API, микросервисы, EDA) и сформированы рекомендации по технологическому стеку в зависимости от масштаба системы. По RQ3 систематизированы требования к информационной безопасности и правовому соответствию **[3, 7, 8, 9]**. По RQ4 сформулированы функциональные и нефункциональные требования к СЭУПВ с учётом потребностей всех ролей пользователей **[5, 6]**.

Ключевой практический вывод: оптимальный технологический выбор определяется масштабом, бюджетом и регуляторным контекстом учреждения. Не существует универсального решения — каждая категория СЭУПВ имеет свою область оптимального применения. При этом соблюдение стандарта HL7 FHIR **[7]** и защита персональных данных в соответствии с законодательством **[9]** остаются обязательными требованиями вне зависимости от выбранного технологического стека и масштаба системы.

(cc) BY

Научная новизна исследования заключается в системной классификации существующих СЭУПВ, расширенной по сравнению с имеющимися обзорами за счёт включения критериев времени внедрения и модели поддержки, а также в формулировке комплекса нефункциональных требований, ориентированных на условия Республики Узбекистан. Результаты работы могут быть использованы разработчиками и заказчиками систем электронной записи при принятии архитектурных и технологических решений, а также при подготовке технических заданий на разработку подобных систем **[4, 6]**.

Список литературы

1. Blobel B., Ruotsalainen P. Health Information Systems Interoperability, Safety and Security // Studies in Health Technology and Informatics. — 2018. — Vol. 249. — P. 3–20.
2. Esposito C. et al. Blockchain-Based Authentication and Authorization for Smart City Applications // Information Processing & Management. — 2021. — Vol. 58(2). — P. 1–18.
3. Garg S., Baliyan N. Comparative Analysis of Android and iOS from Security Viewpoint // Computer Science Review. — 2021. — Vol. 40. — P. 1–12.
4. Kuo M.H. Opportunities and Challenges of Cloud Computing to Improve Health Care Services // Journal of Medical Internet Research. — 2020. — Vol. 22(10). — e22.796.
5. Mearian L. EHR Usability Issues Are Driving Doctors out of Medicine. — Computerworld. — 2019. — URL: https://www.computerworld.com/article/ehr-usability
6. Nguyen L., Bellucci E., Nguyen L.T. Electronic Health Records Implementation: An Evaluation of Information System Impact and Contingency Factors // International Journal of Medical Informatics. — 2019. — Vol. 83(11). — P. 779–796.
7. HL7 International. HL7 FHIR Release 4.0. — 2019. — URL: https://hl7.org/fhir/
8. OWASP Foundation. OWASP Top Ten 2021. — 2021. — URL: https://owasp.org/www-project-top-ten/
9. Закон Республики Узбекистан «О персональных данных» № ЗРУ-547 от 02.07.2019 // Национальная база данных законодательства. — URL: https://lex.uz/ru/docs/4396428
10. Постановление Президента Республики Узбекистан «О мерах по ускоренному развитию системы здравоохранения» № ПП-3894 от 07.12.2018 // LEX.UZ. — URL: https://lex.uz/ru/docs/4072125

(cc) BY