МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра математической кибернетики и компьютерных наук

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ «УПРАВЛЕНИЕ СИСТЕМАМИ ОХРАННО-ПОЖАРНОЙ СИГНАЛИЗАЦИИ FIRESECNT»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Бей Елизаветы Евгеньевны

Научный руководитель
зав.каф.техн.пр.,
к. ф.-м. н., доцент _____ И. А. Батраева

Заведующий кафедрой
к. ф.-м. н., доцент _____ С. В. Миронов

Саратов 2025

ВВЕДЕНИЕ

Мобильные приложения становятся неотъемлемой частью современной среды, предлагая пользователям удобный и быстрый способ работы с различными сервисами на мобильных устройствах. Они дают дополнительные возможности для взаимодействия и коммуникации и служат еще одним инструментом для доступа к данным и сервисам. В этой связи, мобильные приложения выступают важным дополнением к основным веб-клиентам, повышая уровень удобства и доступности для пользователей.

ООО «Р-СОФТ» — это аккредитованная IT-компания, которая занимается разработкой программного обеспечения и оказанием услуг в сфере информационных технологий. Она была основана в 2022 году и с тех пор зарекомендовала себя как надежный поставщик решений в области системной интеграции, разработки приложений, а также аутсорсинга IT-услуг. Программное обеспечение FireSecNT является одним из крупных проектов «Р-СОФТ», обеспечивающим управление системами охранно-пожарной сигнализации.

Компания «Р-СОФТ» инициировала разработку мобильного приложения на `React Native` в дополнение к существующему веб-клиенту FireSecNT с целью оптимизации процесса инсталляции оборудования и настройки системы безопасности, повышения удобства использования и обеспечения кросс-платформенной доступности.

Целью работы является создание мобильного приложения «Управление системами охранно-пожарной сигнализации FireSecNT» с помощью фреймворка `React Native` для компании ООО «Р-СОФТ».

Поставленная цель определила следующие ключевые задачи:

1) провести анализ предметной области и требований;
2) разработать модульную и легко масштабируемую архитектуру приложения;
3) обеспечить сетевое взаимодействие с сервером FireSecNT по `API`;
4) спроектировать интуитивно понятный и адаптированный пользовательский интерфейс;
5) реализовать базовый функционал бизнес-логики приложения;
6) проанализировать результаты разработки.

Структура и объём работы. Бакалаврская работа состоит из введения, 2 разделов, заключения, списка использованных источников и 5 приложений.

Общий объем работы — 57 страниц, из них 40 страниц — основное содержание, включая 23 рисунка и 1 таблицу, CD-носитель в качестве приложения, список использованных источников информации — 20 наименований.

Первый раздел «Анализ предметной области и обзор стека технологий» посвящен описанию основных понятий, необходимых для понимания функционала приложения, а также теоретической информации об используемых технологиях, включая выбор программных библиотек и фреймворков.

Для создания конфигурации системы охранно-пожарной сигнализации FireSecNT необходимо разработать **проект**, который определяет множества таких объектов, как адресные устройства, зоны, сценарии, планы помещений и т.д., а также их настройки и взаимосвязи. В ПО FireSecNT предусмотрены два режима работы проекта: «В разработке» и «Активный проект». В режиме «В разработке» настраиваются основные компоненты проекта, одним из которых является **дерево устройств** — иерархическая схема, отображающая структуру подключенного оборудования и адресных устройств на защищаемом объекте. **Адресные устройства** — это аппаратные компоненты, исполнительные устройства или датчики, подключенные к приемно-контрольному прибору. **Приемно-контрольный прибор** принимает сингналы по адресной линии связи (АЛС) от подключенных к нему адресных устройств и управляет ими. Таким образом, устройства регистрируют различные факторы, указывающие на возникновение пожара и передают информацию на приемно-контрольные приборы, которые в свою очередь отправляют данные на сервер FireSecNT для дальнейшей обработки.

Событие — это контролируемое изменение состояния системы, сопровождаемое световой индикацией, звуковой сигнализацией и отображаемой на экране прибора и компьютера информацией.

Журнал событий — это список последних событий, содержащий полные данные о каждом инциденте, включая дату и время, тип события и вовлеченные устройства.

Логика работы системы реализуется через **сценарии** — последовательности автоматических действий, которые выполняются системой в ответ на определенные события. В зависимости от назначения выделяются три типа сценариев: управляющие, исполнительные и специальные.

После завершения настройки проекта и физического монтажа оборудова-

ния инсталляторами согласно аппаратной конфигурации, проект переводится в режим «Активный проект». В данном режиме осуществляется последняя синхронизация, а именно запись созданной конфигурации в базу данных приборов, подключенных к компьютеру. Затем после успешной процедуры записи система считается полностью настроенной, переходит в режим эксплуатации и решает следующие основные задачи:

1) мониторинг состояния адресной системы охранно-пожарной сигнализации;
2) оповещение о тревогах и неисправностях;
3) регистрация и анализ происходящих событий.

React Native — это фреймворк с открытым исходным кодом для создания нативных мобильных приложений. React Native предоставляет возможность создания кроссплатформенных мобильных приложений с помощью технологии React и языка JavaScript для операционных систем iOS и Android с использованием одной кодовой базы, что существенно сокращает время и ресурсы, затрачиваемые на разработку. В основе React Native лежит библиотека React, реализующая проектирование пользовательских интерфейсов через компоненты. В составе фреймворка React Native предусмотрена обширная библиотека базовых компонентов, облегчающих разработку основных элементов интерфейса. Кроме того, фреймворк предоставляет разработчикам возможности создания собственных компонентов.

В контексте React Native пропсы — это объект, который содержит данные или методы, передаваемые компоненту. С помощью пропсов реализуется механизм передачи данных от родительского компонента к дочернему, что обеспечивает возможность повторного использования.

Состояние — это объект, предназначенный для хранения данных, оказывающих непосредственное влияние на визуальное представление компонента. При изменении состояния React Native автоматически инициирует процесс перерисовки компонента, что приводит к обновлению его отображения в пользовательском интерфейсе.

Жизненный цикл компонентов в React Native представляет собой строго определенную последовательность фаз, происходящих на протяжении всего времени существования компонента. Функции, вызываемые при смене фазы жизненного цикла, позволяют разработчику управлять поведением компонента

и точно контролировать процессы инициализации, обновления и очистки ресурсов. Жизненный цикл компонентов можно условно разделить на следующие основные фазы:

1) во время фазы монтирования происходит создание экземпляра компонента и его встраивание в виртуальный DOM;
2) фаза обновления активируется при изменении внутреннего состояния или получении новых пропсов;
3) фаза размонтирования предшествует удалению компонента из дерева элементов.

Context предоставляет механизм для передачи данных компонентам без необходимости явной передачи пропсов через каждый уровень иерархии. Такое решение особенно полезно для данных, которые должны быть доступны многим компонентам, например, информация о текущем пользователе, тема оформления, язык приложения.

Redux — это библиотека для управления состоянием приложений, предоставляющая централизованное хранилище состояния, которое помогает избежать многоуровневой передачи пропсов и делает управление состоянием более предсказуемым и структурированным. Все состояние приложения state хранится в виде дерева объектов внутри единого хранилища store. Для модификации состояния используются действия actions, которые представляют собой объекты, описывающие события в приложении. Логика обработки этих событий определяется в редьюсерах reducers — чистых функциях, которые принимают предыдущее состояние и действие и возвращают новое состояние. Для извлечения данных из Redux хранилища используются селекторы, которые представляют собой функции, обеспечивающие оптимизированный доступ к состоянию.

В современной разработке React Native приложений Redux Toolkit предлагает эффективный подход к организации состояния через концепцию слайсов. Слайсы представляют собой логически завершенные модули состояния, каждый из которых отвечает за определенную область приложения. Центральным механизмом для работы с асинхронной логикой в Redux Toolkit является функция createAsyncThunk, которая представляет способ обработки асинхронных операций, обеспечивающий автоматическое обновление состояния при переходе между стадиями запроса.

`AsyncStorage` предоставляет собой асинхронное, нешифрованное хранилище пар ключ-значение, позволяющее сохранять небольшие объемы данных локально на устройстве между сессиями приложения. Такое решение подходит для хранения настроек, пользовательских предпочтений или токенов аутентификации.

Второй раздел «Разработка программной части» посвящен постановке задачи, описанию функционала, проектированию архитектуры приложения, реализации основных компонентов пользовательского интерфейса.

Комплект программного обеспечения FireSecNT предназначен для настройки логики работы и начального конфигурирования системы охранно-пожарной сигнализации Рубеж™, записи конфигурации в приемно-контрольные приборы, мониторинга и управления оборудованием защищаемого объекта в режиме реального времени. Мобильное приложение «Управление системами охранно-пожарной сигнализации FireSecNT» (далее Приложение) функционирует как компонент программного обеспечения FireSecNT, который расширяет функциональность, предоставляя удаленный доступ к ключевым функциям системы с мобильных устройств, позволяя таким образом фиксировать состояние оборудования и управлять им в режиме реального времени.

Основной функционал Приложения включает:

- обеспечение удаленного мониторинга и управления системами безопасности через мобильный интерфейс;
- визуализация дерева устройств;
- отображение параметров и статусов всех устройств;
- просмотр и фильтрация событий в системе;
- управление адресными приборами и устройствами пожарной и охранной сигнализации, системами пожаротушения;
- управление сценариями и индикация их текущего состояния;
- добавление записей и комментариев в блокнот для каждого устройства.

Архитектура Приложения построена на основе страниц, каждая из которых отвечает за определенный аспект взаимодействия с системой FireSecNT.

При запуске приложения отображается страница «Настройки сервера», где выполняется конфигурация первичного подключения. Далее пользователь переходит на страницу «Аутентификация», где реализован безопасный механизм ввода учетных данных. Затем система осуществляет проверку состояния

активного проекта через API сервера FireSecNT. При обнаружении активного проекта пользователь направляется на страницу «Активный проект», где отображаются его основные параметры. В противном случае система перенаправляет пользователя на страницу «Нет активного проекта», которая информирует о необходимости активации проекта через веб-интерфейс.

Для работы с подключенным оборудованием реализована страница «Дерево устройств», визуализирующая иерархическую структуру подключенных устройств в виде интерактивного дерева с индикацией статусов. Каждый элемент дерева является точкой входа на страницу «Параметры устройства», где представлен расширенный функционал для управления устройствами, просмотра детализированных параметров и создания и отображения заметок в блокноте.

Функционал страницы «Сценарии» предоставляет возможности для управления сценариями. Для сценариев с повышенными требованиями безопасности реализована страница «Подтверждение авторизации», которая осуществляет проверку пароля.

Страница «Журнал событий» позволяет просмотреть список последних событий в системе и выполнить фильтрацию по типу события.

На странице «Настройки пользователя» реализован механизм выхода из системы, а также отображение текущих настроек сервера, наличия лицензии FireSecNT и других параметров.

Структура навигации построена по модульному принципу с четким разделением на два основных состояния: до и после аутентификации пользователя. Условная маршрутизация в Приложении реализована с помощью контекстов. До прохождения процедуры аутентификации пользователь взаимодействует со страницами «Настройки сервера» и «Аутентификация», объединенных в навигацию AuthStack.

После успешной аутентификации инициируется перенаправление на главный навигационный модуль HomeDrawer, представляющий собой боковое выдвижное меню, которое содержит два основных раздела: модуль «Активный проект» и «Настройки».

Центральный модуль «Активный проект» в состоянии отсутствия активного проекта отображает страницу «Нет активного проекта», в противном случае активным становится навигационный модуль ActiveProjectBottomTabs, предоставляющий собой меню снизу экрана и содержащий следующие основ-

ные разделы:

— страница «Активный проект»;
— модуль DeviceStack;
— модуль ScenariosStack;
— страница «Журнал событий».

Навигация в модуле DeviceStack реализована с использованием стековой модели: при выборе устройства на странице «Дерево устройств» происходит переход к странице «Параметры устройства». Аналогично, ScenariosStack использует стековую навигацию для перехода между страницами «Сценарии» и «Подтверждение авторизации».

В основе системы управления состоянием лежит хранилище Redux, объединяющее несколько специализированных слайсов в единую предсказуемую модель данных. Слайсы получают действия от различных частей Приложения, производят их обработку и изменяют состояние хранилища. Компоненты Приложения получают доступ к данным через селекторы, что обеспечивает оптимальную производительность.

Все данные об активном проекте и других сущностях хранятся в базе данных FireSecNT. Взаимодействие с сервером FireSecNT осуществляется через REST API. При загрузке Приложения пользователь вводит адрес сервера FireSecNT, используемый для дальнейшего сетевого обмена данными. Получение информации о проекте, устройствах, сценариях, событиях, а также отправка команд управления осуществляется в три этапа:

1) формирование и отправка запроса на сервер FireSecNT с необходимыми заголовками, указанием конечной точки и телом запроса;
2) получение ответа от сервера в JSON формате, его валидация и обработка;
3) обновление состояния хранилища Redux через действия и обработка возможных ошибок.

Для отображения уведомлений в Приложении был разработан компонент ToastView, который обрабатывает уведомления различных типов на основе данных, хранящихся в состоянии Redux. ToastView подписывается на состояние toast посредством хука useSelector, что позволяет отслеживать изменения параметров уведомлений. Используя хук useEffect, компонент отслеживает изменения состояния и при активации нового уведомления вызывает соответствующий метод отображения.

Компонент SeverAddressPage реализует функционал страницы «Настройки сервера». В разметке страницы присутствуют два поля ввода для указания IP-адреса сервера и порта, а также кнопка подтверждения. В случае обнаружения некорректного формата IP-адреса отображается уведомление об ошибке. При успешной проверке параметры подключения сохраняются в AsyncStorage для последующего использования и осуществляется переход на страницу «Аутентификация».

Компонент LoginPage, представляющий страницу «Аутентификация», содержит форму ввода учетных данных, состоящую из текстовых полей для логина и пароля, и кнопку подтверждения. При ее нажатии через dispatch отправляется действие loginUser, которое инициирует запрос к серверу. В случае неудачной аутентификации, система отображает соответствующее уведомление об ошибке аутентификации. Успешная авторизация приводит к перенаправлению в основной модуль HomeDrawer, обеспечивающий доступ к ключевым функциям системы.

Компонент NoActiveProjectPage реализует страницу «Нет активного проекта» и выполняет информационную функцию, отображая сообщение о необходимости активации проекта через веб-интерфейс FireSecNT. Компонент ActiveProjectPage представляет страницу «Активный проект», которая содержит основную информацию о проекте, а именно название, версию, дату и время создания, автора и описание.

Компонент SettingsPage был разработан для отображения страницы «Настройки», его разметка включает в себя отображение основных системных параметров, таких как язык интерфейса, версия, наличие лицензии, адрес сервера FireSecNT, порт, и кнопку для выхода из системы.

Страница «Дерево устройств» представлена с помощью компонента DeviceTreePage, который обеспечивает визуализацию иерархической структуры подключенных устройств системы. С помощью хука useSelector компонент получает состояние deviceTree, которое вычисляется после успешной загрузки активного проекта. Алгоритм построения дерева реализован в функции getDeviceTree, принимающей два параметра: полный список устройств devicesList и опциональный корневой элемент root. При первоначальном вызове без указания корневого элемента функция из списка устройств deviceList определяет корневое устройство как устройство с неопределенным идента-

фикатором родителя parentDeviceId. Для каждого узла дерева формируется объект, содержащий само устройство device и список дочерних элементов children, полученный рекурсивным применением getDeviceTree к каждому из устройств, имеющих указанное устройство в качестве родительского.

Компонент DeviceForm был разработан для представления одного устройства в дереве и отображения следующих элементов: интерактивной иконки управления видимостью дочерних элементов (при их наличии), типовой иконки устройства, статусного индикатора, названия и адреса устройства в АЛС. При нажатии на устройство, вызывается функция navigateToParams, которая устанавливает текущее устройство currentDevice и выполняет перенаправление на страницу «Параметры устройства».

При запуске выполнения команды управления над сценарием или исполнительным устройством может быть установлен обратный отсчет — задержка выполнения действия, настраиваемая при описании параметров. В системе предусмотрен механизм управления обратными отсчетами, например, остановить, выполнить немедленно, увеличить задержку, отменить действие или возобновить обратный отсчет. Компонент CountdownView предоставляет интерфейс для работы обратными отсчетами. Его отображение включает таймер с индикацией оставшегося времени, идентификацию целевого объекта (устройства или сценария), набор управляющих элементов и текущий статус. При нажатии на иконку крестика в правом верхнем углу можно «закрыть» обратный отсчет, что добавит его идентификатор в список скрытых обратных отсчетов.

Компонент DevicePage был разработан для реализации функционала страницы «Параметры устройств». Сверху экрана отображается текущее выбранное устройство, его статус, название и адрес, а ниже представлена система вкладок:

— вкладка «Управление» для управления устройством;
— вкладка «Параметры» для отображения параметров устройства;
— вкладка «Блокнот» для записей и комментариев.

Вкладка «Управление» реализована с помощью компонента ControlDevice, которая отображает список обратных отсчётов для устройств и список команд управления текущего устройства, сгруппированных по их типу, выполнить которые можно нажав на кнопку.

Компонент DeviceParams отображает вкладку «Параметры», которая со-

держит все характеристики устройства: полный адрес устройства, дополнительные статусы, тип, условия работы, зона привязки, список родительских устройств и многие другие.

Вкладка «Блокнот» представлена компонентом `Notebook`, который отображает список оставленных комментариев и их авторов с помощью компонента `NoteView`, а также текстовое поле и кнопку для создания комментария.

После отправки запроса управления устройством или добавления комментария появляется уведомление `Toast` с сообщением «В процессе». После получения ответа от сервера на экране появляется уведомление с сообщением о статусе операции.

Компонент `ScenariosPage` был спроектирован для реализации страницы «Сценарии». Он отображает список обратных отсчётов и управляемых сценариев, а также предоставляет функционал для управления ими. Компонент `ScenarioItem` отвечает за отображение информации об отдельном сценарии. Он включает в себя номер сценария, его название и иконку, отображающую статус сценария. При нажатии на `ScenarioItem` внизу экрана открывается меню `ScenaroisActionsMenu`, элементами которого являются команды управления сценарием. При нажатии на одну из них генерируется отправка действия `performScenarioAction`. После отправки запроса появляется уведомление с сообщением «В процессе». При получении ответа от сервера отображается уведомление о статусе выполнения операции.

Если в параметрах сценария указано, что он имеет повышенный уровень безопасности, то перед запуском команды управления происходит перенаправление на страницу «Подтверждение авторизации», где нужно дополнительно указать пароль.

Страница «Журнал событий» была реализована с помощью компонента `EventLogPage`, который обеспечивает отображение и фильтрацию системных событий через интерфейс с интерактивными элементами управления. Получение данных осуществляется через хук `useSelector`, извлекающий массив событий `events` из хранилища `Redux`. Для визуализации используется компонент `FlatList`, оптимизированный для работы с большими наборами данных. Система фильтрации построена на основе состояния `eventColors`, которе представляет собой список элементов, задающих параметры фильтров.

ЗАКЛЮЧЕНИЕ

В рамках дипломного проекта было разработано мобильное приложение «Управление системами охранно-пожарной сигнализации FireSecNT», представляющее собой решение для мониторинга и управления противо-пожарными системами. Во время реализации проекта был проведен глубокий анализ предметной области, а также исследованы современные подходы к проектированию мобильных приложений и особенности работы с системами безопасности и оборудованием.

В ходе дипломной работы была рассмотрена научно-техническая литература и документация по фреймворку `React Native` и другим библиотекам для мобильной разработки; был проведен комплексный анализ предметной области и требований; была спроектирована архитектура приложения; было реализовано взаимодействие с сервером FireSecNT; был разработан удобный и интуитивно понятный пользовательский интерфейс, а также реализован весь заявленный функционал. Приложение обеспечивает быстрый доступ к важнейшим функциям, отображает наглядное состояние системы и обеспечивает мониторинг и управление оборудованием защищаемого объекта в режиме реального времени.

Таким образом, дипломный проект достиг поставленной цели и выполнил все основные задачи. Полученные результаты подтверждают практическую значимость работы и открывают перспективы для дальнейшего развития приложения, включая расширение функционала и адаптацию к новым типам оборудования.