МИНИСТЕРСТВО ОБРАЗОВАНИЯ, КУЛЬТУРЫ И ИССЕЛЕДОВАНИЯ

РЕСПУБЛИКИ МОЛДОВА

БЕЛЬЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. «АЛЕКУ РУССО»

ФАКУЛЬТЕТ ТОЧНЫХ НАУК, ЭКОНОМИКИ И ОКРУЖАЮЩЕЙ СРЕДЫ

КАФЕДРЫ МАТЕМАТИКИ И ИНФОРМАТИКИ

**ИСПОЛЬЗОВАНИЕ API GOOGLE SERVICES В ГИБРИДНЫХ МОБИЛЬНЫХ ПРИЛОЖЕНИЯХ JS НА ОСНОВЕ VUEJS**

**ЛИЦЕНЗИОННАЯ РАБОТА**

**Автор:**

Студентка группы IS31Z

**Станислава ВОВОД**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

**Научный руководитель:**

**Корина НЕГАРА**

др., конф., унив.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

**Дмитрий СТОЯН**

асист. унив.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

**БЭЛЦЬ, 2021**

MINISTERUL EDUCAŢIEI AL REPUBLICII MOLDOVA

UNIVERSITATEA DE STAT „ALECU RUSSO” DIN BĂLŢI

FACULTATEA DE ŞTIINŢE REALE, ECONOMICE ȘI ALE MEDIULUI

CATEDRA DE MATEMATICĂ ȘI INFORMATICĂ

**UTILIZAREA API GOOGLE SERVICES ÎN APLICAȚII MOBILE HIBRIDE JS PE BAZA VUEJS**

**TEZĂ DE LICENTA**

**Autor:**

Studenta al grupei IS31Z

**Stanislava VOVOD**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnatura)

**Conducător științific:**

**Corina NEGARA**

dr., conf., univ.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnatura)

**Dumitru STOIAN**

asist. univ.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnatura)

**BĂLȚI, 2021**

Controlată:

Data „\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_ 2021

Conducător ştiinţific: dr., conf. univ., Corina NEGARA

asist. univ., Dumitru STOIAN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Aprobată

şi recomandată pentru susţinere

la şedinţa Catedrei de ştiinţe economice

Proces-verbal nr. \_\_\_\_\_\_ din „\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_202\_\_

Şeful catedrei dr., conf. univ. Corina NEGARA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnatura)

MINISTRY OF EDUCATION OF THE REPUBLIC OF MOLDOVA

"ALECU RUSSO" STATE UNIVERSITY OF BALTI

FACULTY OF REAL, ECONOMIC AND ENVIRONMENTAL SCIENCES

DEPARTMENT OF MATHEMATICS AND COMPUTING

**USING API GOOGLE SERVICES IN HYBRID MOBILE APPLICATIONS JS ON VUEJS**

**LICENSE THESIS**

**Author:**

Student of group IS31Z

**Stanislava VOVOD**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(signature)

**Scientific leader:**

**Corina NEGARA**

dr., conf., univ.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(signature)

**Dumitru STOIAN**

asist. univ.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(signature)

**BĂLȚI, 2021**

*ANNOTATION*

Licensed work. " Using Google Services in hybrid mobile applications JS on Vue.js", Vovod Stanislava. Specialty: 0444.1-Computer science (exact sciences). Balti, 2021. The structure of the work. The licensed work contains an introduction, two chapters, conclusions, a list of references from 20 sources, 44 pages of the main text, 10 figures.

Keywords: web applications, mobile applications, web developing, hybrid mobile applications, google services.

The purpose of the work is development of a demo hybrid mobile application using Vue.js framework and embed Google Services in it.

Aprobat:

Șeful catedrei de matematică și informatică

dr., conf. univ. C.Negara

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_202\_\_

**Graficul calendaristic de executare a tezei de licență**

Tema tezei de licența: UTILIZAREA GOOGLE SERVICES ÎN APLICAȚII MOBILE HIBRIDE JS PE BAZA VUEJS

confirmată prin ordinul rectorului USARB nr.\_\_\_ din „\_\_\_\_\_\_\_\_\_\_\_\_\_”

Termenul limită de prezentare a tezei de licență la Catedra de matematică și informatică „\_\_\_\_\_\_\_\_\_\_\_”.

**Etapele executării tezei de licență:**

|  |  |  |
| --- | --- | --- |
| Etapele | Termenul de realizare | Viza de executare |
| 1. Stabilirea temei; fixarea obiectivelor; selectarea surselor de informare. | 11.09.20 | realizat |
| 1. Investigația cadrului teoretic al cercetării; expunerea cadrului teoretic al cercetării. | 03.10.20 | realizat |
| 1. Întocmirea problemei cercetării; stabilirea tipului de cercetare. | 21.10.20 | realizat |
| 1. Specificarea unităților studiate. | 10.11.20 | realizat |
| 1. Alegerea metodelor de cercetare. | 18.11.20 | realizat |
| 1. Culegerea datelor; selectarea modalităților de prelucrare a datelor; stocarea datelor; analiza datelor | 21.01.21 | realizat |
| 1. Elaborarea concluziilor și a recomandărilor practice. | 15.03.21 | realizat |
| 1. Susținerea preventivă a tezei. | 30.04.21 | realizat |

Student (ă) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura)

Conducători științifici \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(*semnătura*)

СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 2](#_Toc41253564)

[ВВЕДЕНИЕ 3](#_Toc41253565)

[1. ИСПОЛЬЗОВАНИЕ ГУГЛ СЕРВИСОВ В СОВРЕМЕННЫХ ИНФОРМАЦИОННЫХ СИСТЕМАХ 5](#_Toc41253566)

[1.1. Один из многофункциональных сервисов гугл - Firebase 6](#_Toc41253567)

[1.1.1. Аутентификация в Firebase 6](#_Toc41253568)

[1.1.2. Firebase Хостинг 7](#_Toc41253569)

[1.1.3. Cloud Storage 8](#_Toc41253570)

[1.1.4. База данных реального времени Firebase 9](#_Toc41253571)

[1.2. Google OAuth 10](#_Toc41253572)

[1.2.1. Абстрактное описание протокола 11](#_Toc41253573)

[1.2.2. Регистрация приложения 12](#_Toc41253574)

[1.2.3. Идентификатор клиента и секрет клиента 12](#_Toc41253575)

[1.2.4. Итог по первой главе 12](#_Toc41253576)

[2. ГИБРИДНЫЕ ПРИЛОЖЕНИЯ ИСПОЛЬЗУЯ VUEJS И VUETIFY 13](#_Toc41253577)

[2.1. VueJS при создании пользовательских интерфейсов………………………...14](#_Toc41253578)

[2.2. Vuetify - Framework для создания гибридных мобильных приложений 15](#_Toc41253579)

[3. ВНЕДРЕНИЕ ГУГЛ-СЕРВИСОВ В СИСТЕМЫ, ИСПОЛЬЗУЮЩИЕ VUEJS 17](#_Toc41253580)

[3.1. Описание структуры приложения 17](#_Toc41253581)

[3.2. Установка и настройка среды разработки 17](#_Toc41253582)

[3.2.1. Node Package Manager (NPM) 17](#_Toc41253583)

[3.2.2. Инструменты командной строки (CLI) 18](#_Toc41253584)

[3.2.3. Использование Cordova для определения гибридной трансляции 18](#_Toc41253585)

[3.2.3. Создание приложения 18](#_Toc41253586)

[3.3. Создание проекта в Firebase 19](#_Toc41253587)

[3.4. Интеграция Firebase 19](#_Toc41253588)

[Заключение 24](#_Toc41253589)

[Библиография 25](#_Toc41253590)

# ВВЕДЕНИЕ

На рынке существуют тысячи стартапов, пытающихся продвинуть свои идеи, а все больше владельцев бизнеса стараются автоматизировать наиболее монотонные процессы внутри компаний, продвигать свою продукцию или же установить более тесный контакт с аудиторией онлайн. Однако, все возрастающая популярность приложений привела к появлению к их различным вариациям: на рынке есть нативные, гибридные и веб приложения.

Судя по динамике развития мобильных проектов, мировые тенденции подтверждаются на практике - увеличивается спрос на мобильную разработку, растут скорости создания приложений, повышается интерес к несложным решениям с доступной стоимостью разработки. Всё это открывает возможности для совершенствования гибридных приложений, которые в отдельных случаях могут проявлять себя не хуже нативных. Отдавая предпочтение тем или другим, важно отслеживать состояние рынка и правильно расставлять приоритеты по скорости запуска проекта, стоимости и производительности.

*Цель*данной работы состоит в исследовании гибридных web технологий и их использовании для создания приложений при помощи языка программирования JavaScript в фреймворке Vue.js.

Для выполнения поставленной цели были спланированы следующие *задачи*:

* Изучение основ гибридных web технологий;
* Исследование возможности использования гибридных web технологий в качестве замены нативных технологий;
* Анализ работы современных облачных технологий для интеграции в приложения;
* Разработка мобильного приложения используя гибридные web технологии.

*Ценность работы:*данная работа полезна своей теоретической частью, так как было произведено исследование основных технологий, используемых для создания гибридных web приложений, практическая часть будет полезна всем желающим ознакомиться с процессом создания гибридного мобильного приложения используя Javascript и данный набор технологий.

*Актуальность:* на сегодняшний день мобильные устройства все больше завоевывают долю рынка потребления интернет контента, гибридные мобильные приложения - это набор технологий, которые призваны сочетать в себе основные преимущества мобильной платформы будучи обычным web приложением.

Основываясь на поставленных целях, работа была разделена на 3 главы:

В первой главе было проведено исследование технологий в составе гибридных мобильных приложений таких как Firebase API, Google OAuth и другие.

Во второй главе было проведено описание гибридного web приложений, и таких технологий так Vue, Vuetify и Cordova для создания мобильного веб приложения.

В третьей главе описывается процесс создания гибридного web приложения используя данный набор технологий:

В ходе данной работы будет разработано приложение, которое включает в себя набор функций по типу популярного приложения для создания и размещения семейных покупок. Это будет web приложение, включающее в себе множество функций нативного приложения, так как оно будет построено на основе гибридных web технологий;

Работа состоит из 27 страниц, содержит 6 изображений, 1 таблицу и 16 библиографических источников.

# ИСПОЛЬЗОВАНИЕ ГУГЛ СЕРВИСОВ В СОВРЕМЕННЫХ ИНФОРМАЦИОННЫХ СИСТЕМАХ

Разработка серверной части - один из самых трудных этапов создания приложения. Зачастую, при планировании разработки проекта недооценивается необходимый объем ресурсов и время создания бэкенда (Рис.1.1.). Еще одна проблема - ограниченность доступных команде ресурсов. Чаще всего разрабатывать бэкенд приходится с помощью тех инструментов и технологий, которыми владеют члены команды. Процесс получается длительным, а само приложение — достаточно сложным и дорогим с точки зрения сопровождения. А длительная разработка бэкенда на даже относительно простых проектах ведет к увеличению расходов и другим рискам: расход денег без визуально видимого результата.

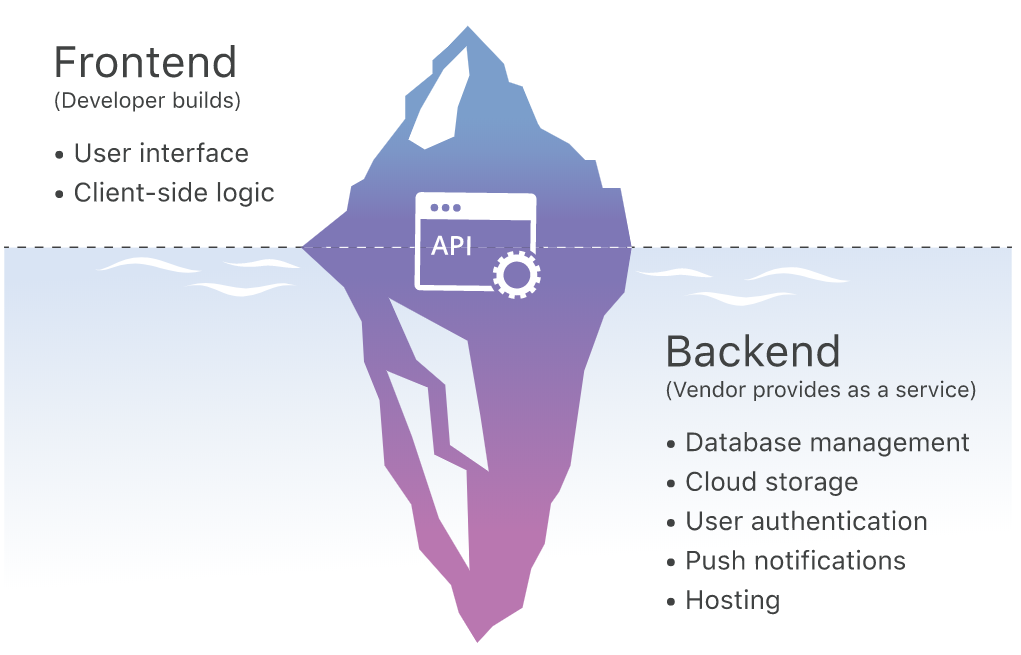


Рис.1.1. Принцип работы современного приложения.

Решить эту проблему возможно с помощью **Backend as a Service (BaaS**) - готовых серверных сервисов, собрав которые вместе, получается необходимый универсальный бэкенд для любого проекта. BaaS является хорошим решением, избавляющем от существенных затрат, а именно написания и поддержки серверного бэкенда. BaaS дает возможность реализовать простой сервис силами одного мобильного разработчика [1]. Такое решение будет особо уместно для небольших стартапов, которые сначала хотят проверить бизнес идею на её работоспособность, не планируя очень сложной логики и взаимодействия со сторонними сервисами — хотя с BaaS такое тоже возможно реализовать.

Преимущества бэкенда на BaaS, в основном, заключаются в том что, используя BaaS, разработчики способны быстро построить необходимый бэкенд и платформу для обработки данных, поступающих из мобильных приложений.

BaaS позволяет избегать разработчикам необходимости иметь дело с:

* физическим сервером приложения;
* базой данных;
* клиент-серверной библиотекой;
* написанием админки;
* дизайном своего API;
* хостингом.

BaaS-функции включают в себя облачное хранилище, push-уведомления, управление пользователями и файлами, службы определения местоположения и др [2]. Все эти сервисы имеют собственный API, чтобы легко встраиваться в приложения.

* 1. Один из многофункциональных сервисов гугл - Firebase

Firebase помогает быстро создавать качественные приложения, увеличивать аудиторию вовлеченных пользователей и повышать доходы. Платформа содержит множество полезных функций для приложения, в том числе серверный код для мобильных сервисов, статистику, а также инструменты для монетизации и расширения аудитории.

* + 1. Аутентификация в Firebase

Большинство приложений должны идентифицировать личность пользователя. Знание личности пользователя позволяет приложению безопасно сохранять пользовательские данные в облаке и обеспечивать одинаковую персонализированную работу на всех устройствах пользователя.

Аутентификация Firebase предоставляет бэкэнд-сервисы, простые в использовании SDK и готовые библиотеки пользовательского интерфейса для аутентификации пользователей в приложении. Он поддерживает аутентификацию с использованием паролей, телефонных номеров, популярных провайдеров федеративной идентификации, таких как Google, Facebook и Twitter, и многого другого. Аутентификация Firebase тесно интегрируется с другими сервисами Firebase и использует отраслевые стандарты, такие как OAuth 2.0 и OpenID Connect, поэтому ее можно легко интегрировать с пользовательским бэкэндом.

В приложение с использованием Firebase возможно войти либо с помощью FirebaseUI в качестве полного решения для проверки подлинности, либо с помощью пакета Firebase Authentication SDK для ручной интеграции одного или нескольких методов входа в приложение (Рис.1.2.).

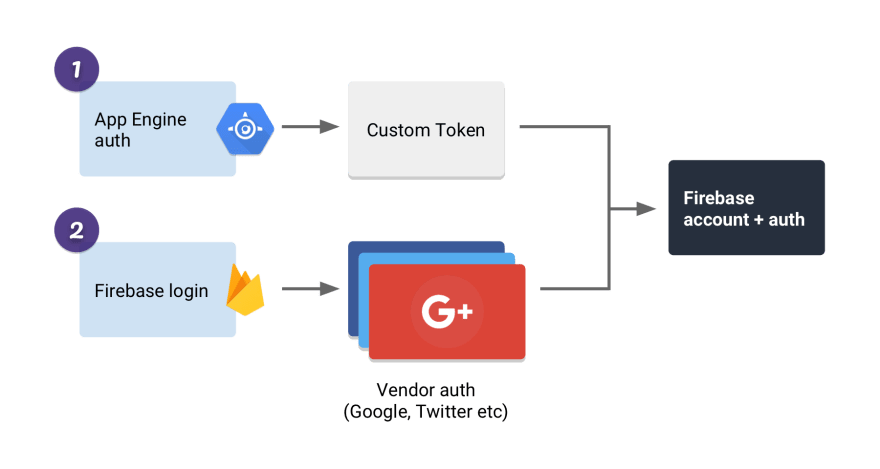


Рис.1.2. Принцип работы аутентификации*.*

Firebase предоставляет следующие возможности по аутентификации:

* **Аутентификация по электронной почте и паролю;**
* **Интеграция с федеративным поставщиком удостоверений;**
* **Проверка подлинности номера телефона;**
* **Кастомная аутентификация системной интеграции;**
* **Анонимная авторизация.**

Также стоит отметить, что по умолчанию аутентифицированные пользователи могут читать и записывать данные в базу данных Firebase Realtime и облачное хранилище [3].

* + 1. Firebase Хостинг

Firebase Hosting - это хостинг для веб-сайтов промышленного уровня для разработчиков. С помощью одной команды появляется возможность быстро развернуть веб-приложения и предоставлять как статический, так и динамический контент в глобальную CDN (сеть доставки контента). Также можно соединить Firebase Hosting с Cloud Functions или Cloud Run для создания и размещения микросервисов в Firebase.

Firebase Hosting создан для современного веб-разработчика. Веб-сайты и приложения стали более мощными, чем когда-либо, благодаря появлению интерфейсных JavaScript-фреймворков, таких как Angular и статических генераторов, таких как Jekyll. Он предоставляет инфраструктуру, функции и инструменты, адаптированные для развертывания веб-сайтов, приложений и управления ими не зависимо от того, простая ли это целевая страница приложения либо сложное Progressive Web App (PWA).

Ключевые возможности:

* Передача контента через безопасное соединение;
* Статический и динамический контент + микросервисы;
* Быстрая доставка контента;
* Развертывание новых версий одной командой;
* Откат одним кликом.

Firebase Hosting обеспечивает быстрый и безопасный хостинг для веб-приложения, статического и динамического контента, микросервисов.

* + 1. Cloud Storage

Облачное хранилище для Firebase - это мощный, простой и экономически эффективный сервис хранения объектов, созданный для масштаба Google. Пакеты Firebase SDK для облачного хранилища повышают безопасность Google при загрузке и выгрузке файлов для приложений на основе Firebase, независимо от качества сети [4]. Помимо этого, доступны такие инструменты, как SDK для хранения изображений, аудио, видео или другого пользовательского контента. На сервере существует возможность использовать Google Cloud Storage , чтобы получить доступ к тем же файлам.

Ключевые возможности:

* Надежные операции;
* Высокий уровень безопасности;
* Высокая масштабируемость.

Разработчики используют Firebase SDK для облачного хранилища, чтобы загружать и скачивать файлы непосредственно с клиентов [5]. Если сетевое соединение плохое, клиент может повторить операцию прямо с того места, где он был прерван, что экономит время пользователей и пропускную способность.

* + 1. База данных реального времени Firebase

База данных Firebase Realtime представляет собой облачную базу данных [6]. Данные хранятся в формате JSON и синхронизируются в режиме реального времени с каждым подключенным клиентом. При создании кроссплатформенного приложения с помощью набора средств разработки SDK для iOS, Android и JavaScript, все клиенты совместно используют один экземпляр базы данных в реальном времени и автоматически получают обновления с самыми новыми данными.

База данных Firebase Realtime позволяет создавать многофункциональные приложения для совместной работы, обеспечивая безопасный доступ к базе данных непосредственно из кода на стороне клиента (Рис.1.3.).



Рис.1.3. Совместная работа устройств в реальном времени*.*

Данные сохраняются локально, и даже в автономном режиме события в реальном времени продолжают срабатывать, предоставляя конечному пользователю доступ к приложению. Когда устройство восстанавливает соединение, база данных реального времени синхронизирует локальные изменения данных с удаленными обновлениями, которые произошли, когда клиент находился в автономном режиме, автоматически объединяя любые несоответствия.

Ключевые возможности:

* Режим реального времени;
* Доступность в оффайн режиме;
* Доступно с клиентских устройств;
* Масштабирование по нескольким базам данных.

База данных реального времени является базой данных NoSQL и, как таковая, имеет различные оптимизации и функциональность по сравнению с реляционной базой данных.

* 1. Google OAuth

OAuth 2 представляет собой фреймворк для авторизации, позволяющий приложениям осуществлять ограниченный доступ к пользовательским аккаунтам на HTTP сервисах, например, на Facebook, GitHub и DigitalOcean. Он работает по принципу делегирования аутентификации пользователя сервису, на котором находится аккаунт пользователя, позволяя стороннему приложению получать доступ к аккаунту пользователя. OAuth 2 работает на web, desktop и мобильных приложениях [7].

OAuth выделяет четыре роли:

* Владелец ресурса;
* Клиент;
* Сервер ресурсов;
* Авторизационный сервер.

Владельцем ресурса является пользователь, который авторизует приложение для доступа к своему аккаунту. Доступ приложения к пользовательскому аккаунту ограничен “областью видимости” (scope) предоставленных прав авторизации (например, доступ на чтение или запись).

Сервер ресурсов непосредственно хранит защищённые данные аккаунтов пользователей, а авторизационный сервер проверяет подлинность информации, предоставленной пользователем, а затем создаёт авторизационные токены для приложения, с помощью которых приложение будет осуществлять доступ к пользовательским данным.

С точки зрения разработчика приложения API сервиса одновременно выполняет и роль сервера ресурсов и роль сервера авторизации. Далее мы будем считать эти две роли одной, и называть её Сервис или API.

Клиентом является приложение, которое хочет осуществить доступ к аккаунту пользователя. Перед осуществлением доступа приложение должно быть авторизовано пользователем, а авторизация должна быть одобрена со стороны API.

1.2.1. Абстрактное описание протокола

В данной диаграмме (Рис.1.4.) представлены роли, используемые в OAuth, и их взаимодействия друг с другом.



Рис.1.4. Диаграмма Взаимодействия ролей в OAuth.

Последовательность шагов на этой диаграмме:

1. Приложение запрашивает у пользователя авторизацию на доступ к серверу ресурсов;
2. Если пользователь авторизует запрос, приложение получает разрешение на авторизацию (authorization grant);
3. Приложение запрашивает авторизационный токен у сервера авторизации (API) путём предоставления информации о самом себе и разрешении на авторизацию от пользователя;
4. Если подлинность приложения подтверждена и разрешение на авторизацию действительно, сервер авторизации (API) создаёт токен доступа для приложения. Процесс авторизации завершён;
5. Приложение запрашивает ресурс у сервера ресурсов (API), предоставляя при этом токен доступа для аутентификации;
6. Если токен действителен, сервер ресурсов (API) предоставляет запрашиваемый ресурс приложению.

Фактический порядок шагов описанного процесса может отличаться в зависимости от используемого типа разрешения на авторизацию, но в целом процесс будет выглядеть описанным образом.

1.2.2. Регистрация приложения

Перед применением OAuth в приложении, необходимо зарегистрировать его на сервисе. Это делается путём регистрации в разделе “developer” или “API” сайта сервиса, где предоставляется следующая информация:

* Название приложения;
* Сайт приложения;
* Redirect URL или callback URL.

Redirect URL - это URL, на который сервис будет перенаправлять пользователя после авторизации (или отказа в авторизации) вашего приложения.

1.2.3. Идентификатор клиента и секрет клиента

После регистрации приложения сервис создаст учётные данные клиента - идентификатор клиента (client ID) и секрет клиента (client secret). Идентификатор клиента представляет собой публично доступную строку, которая используется API сервиса для идентификации приложения, а также используется для создания авторизационных URL для пользователей. Секрет клиента используется для аутентификации подлинности приложения для API сервиса, когда приложение запрашивает доступ к аккаунту пользователя. Секрет клиента должен быть известен только приложению и API.

* 1. Сервис Google Map

Google Maps – это комплекс приложений, созданных на базе бесплатного сервиса картографии и технологии, используемой Google. Данное приложение используется для поиска информации на карте с отметками достопримечательностей, организаций и т.д [8].

Оформление и возможности Google Maps

Сервис предлагает такие режимы отображения, как:

«Спутник» – показ спутниковых и аэрофотографий (сюда относятся «Имена объектов» – отображение сведений на схемах. До этого режим назывался «Гибрид»);

«Карта» – показ данных картографии;

«Ландшафт» – схемы, отображающие массивные элементы поверхности, например леса или холмы (ложный 3D-рельеф);

«Просмотр улиц» – режим, позволяющий ориентироваться по панорамным фотоснимкам на фоне городских улиц;

«Личные схемы» – персональный режим, благодаря которому можно нанести на карту требуемое содержимое.

Благодаря Google Maps пользователь может:

* просматривать интересные локации на карте;
* открывать файлы с расширением KML и KMZ, а также сохранять нужные метки в этих форматах;
* прокладывать маршрут;
* вносить персональный текст, картинки и видеоролики;
* отправлять созданную схему друзьям.

Некоторые режимы и функции:

Режим Google Transit дает подробные рекомендации по проезду между двумя указанными точками на доступном общественном транспорте и приблизительно подсчитывает, сколько это займет времени. Данная опция доступна для всей России.

Пользователь может отправить на почтовый ящик ссылку со своим местоположением либо разместить HTML-код карты Google на веб-ресурсе. Также в Google Maps имеется инструмент генерирования карты для последующей распечатки.

Встроены результаты поиска информации по организациям – поиск адресов компаний и их контактных данных в выбранном районе. К примеру, если написать запрос «суши в районе Тверская улица», на локальной системе координат обозначатся точки расположения ближайших заведений, их адреса, режимы работы. Также можно посмотреть дополнительные сведения, такие как формы оплаты, рейтинг, отзывы клиентов и др.

В картах доступна функция Map Maker (ранее это был отдельный сервис), позволяющая изменить содержание собственной информацией.

Значение Google Maps для SEO-продвижения

Сервис Google Maps полностью бесплатный. Но при этом существует ограничение: в локальном поиске отображается только 7 компаний, предоставляющих схожие товары/услуги.

Факторы ранжирования в «Картах Google»:

* полнота информации о компании, представленной при регистрации в «Google Мой Бизнес»;
* упоминание поисковых запросов и категории в имени компании;
* наличие веб-сайта, на котором прописан фактический адрес в тегах <adress> </adress>;
* положительные отзывы;
* интересная информация в заметках;
* подключение контекстной рекламы AdWords;
* присутствие на Google Earth.
  1. Итог по первой главе

Авторизация с помощью данного гугл сервиса дает большое количество преимуществ разработчикам, так же это простой способ аналога регистрации для пользователя, ведь им всего лишь нужно аутентифицироваться с помощью данных своего гугл аккаунта для первого и последующих входов в приложение, нет необходимости в заполнении какими-либо данными профиля, ведь все необходимое предоставляет гугл. Что касается разработчиков, такой подход авторизации снимает с них ответственность за хранимые данные в своей базе данных, таких как пароль или других данных [9]. Так же этот способ экономит много времени, которое могло быть потрачено на создание, построении защиты данных, которые должны храниться у разработчика, а также, устранять возможности обхода, обмана системы, кражи данных и так далее.

# ГИБРИДНЫЕ ПРИЛОЖЕНИЯ ИСПОЛЬЗУЯ VUEJS И VUETIFY

Гибридные приложения являются чем-то средним между нативными и веб-приложениями (Рис.2.1.). Фактически они создаются так, чтобы выглядеть и использоваться как нативные приложения. Их также устанавливают на телефон пользователя и их можно найти в магазинах приложений [9]. Различие заключается в том, что они обязательно должны размещаться в рамках нативного приложения и созданы, чтобы работать [10] через WebView, и таким образом они могут получать доступ к информации на устройстве пользователя для больших возможностей.

Дополнительные преимущества гибридных приложений:

1. Гибридные приложения обладают наибольшей функциональностью и персонализацией для пользователя;
2. Разработчики не ограничены одной платформой, вместо этого они могут создать гибридное приложение, которое будет работать с несколькими платформами (в случае работы как нативное приложение);
3. Гибриды - хорошая опция для разработчиков, которые создают визуально насыщенные приложения, например, игры (которые не будут хорошо работать в виде веб-приложений).



Рис.2.1. Гибридное приложение.

Выбор подходящей модели мобильного приложения - это очень важный этап в его разработке, на который влияют несколько факторов, таких как:

* техническая оценка разработчиков;
* потребность в доступе к информации на устройстве;
* влияние скорости интернета на приложение;
* одно- или многоплатформенное ли приложение.

Процесс разработки гибридного приложения [11] соответствует процессу разработки веб-сервиса с дополнительным этапом разработки интерфейса обложки мобильного приложения. В результате владелец проекта имеет и сайт веб-сервиса, и мобильное приложение.

2.1. VueJS при создании пользовательских интерфейсов

Vue - это прогрессивный фреймворк для создания пользовательских интерфейсов Web с использованием шаблона архитектуры MVVM (Model-View-ViewModel) [12]. В отличие от фреймворков-монолитов, Vue создан пригодным для постепенного внедрения. Его ядро в первую очередь решает задачи уровня представления (view), что упрощает интеграцию с другими библиотеками и существующими проектами. Он работает с уровнем представления. С другой стороны, Vue полностью подходит и для создания сложных одностраничных приложений (SPA, Single-Page Applications), если использовать его совместно с современными инструментами и дополнительными библиотеками. Двумя основными конкурентами Vue являются React и Angular. Вместе с ними Vue закрывает почти 100% рынка Web-разработки.

Поскольку Vue работает только на «уровне представления» и не используется для промежуточного программного обеспечения и бэкэнда, он может легко интегрироваться с другими проектами и библиотеками, в том числе с Vuetify [13]. Vue.js содержит широкую функциональность для уровня представлений и может использоваться для создания мощных одностраничных веб-приложений.

Функции Vue.js:

* Реактивные интерфейсы;
* Декларативный рендеринг;
* Связывание данных;
* Директивы (все директивы имеют префикс «V-». В директиву передается значение состояния, а в качестве аргументов используются html атрибуты или Vue JS события);
* Логика шаблонов;
* Компоненты;
* Обработка событий;
* Свойства;
* Переходы и анимация CSS;
* Фильтры.

Основная библиотека Vue.js 2 очень маленькая (всего 17 кБ). Это гарантирует, что нагрузка на проект, реализованный с помощью Vue.js, минимальна, а сайт будет быстро загружаться.

Vue подходит для небольших проектов, которым необходимо добавить немного реактивности, представить форму с помощью AJAX, отобразить значения при вводе данных пользователем, авторизацию или другие аналогичные задачи. Vue легко масштабируется и хорошо подходит для объемных проектов, поэтому его называют прогрессивным фреймворком [14].

Vue также отлично подходит для крупных одностраничных приложений благодаря своим основным компонентам, таким как Router и Vuex. С Vue можно как использовать общедоступные API для создания приложений, так и реализовывать выполняемые сервером приложения. Но Vue лучше всего подходит для разработки решений, которые используют внешние API для обработки данных.

С помощью Vue также можно создавать frontend блога на популярных CMS. Vue.js отлично подходит и для разработки динамических интерфейсов, которые адаптируются под пользователя.

2.2. Vuetify - Framework для создания гибридных мобильных приложений

Vuetify является библиотекой #1 для Vue.js и активно разрабатывается с 2016 года [15]. Цель проекта - предоставить пользователям все, что необходимо для создания многофункциональных, красивых и интересных веб-приложений, используя спецификацию Material Design. Это возможно благодаря постоянному циклу обновлений, долгосрочной поддержке предыдущих версий, отзывчивому участию сообщества, обширной экосистеме ресурсов и приверженности качественным компонентам (Tаб.1.).

Таб.1. Сравнение UI фреимворков

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Функции | Vuetify | Bootstrap | Buefy | Element UI | Quasar Framework |
| Full Accessibility and Section 508 support | + | + | + |  |  |
| Server Side Rendering support | + | + | + | + | + |
| Long-term Support | + |  |  |  |  |
| Release cadence\*\* | Weekly | Bi-Weekly | Bi-Monthly | Bi-Weekly | Bi-Weekly |
| Treeshaking | Automatic | Manual | Manual | Manual | Automatic |
| RTL support | + | + |  | + | + |
| Premium Themes | + | + |  |  |  |
| Business and Enterprise support | + |  |  |  |  |

Библиотека Vuetifyjs получила более 7 тысяч звёзд на GitHub. Она даёт разработчикам возможности по созданию пользовательских интерфейсов с использованием принципов Google Material Design. В её релизе V 1.0 Alpha можно найти более 80 компонентов, подходящих для повторного использования и спроектированных с применением семантических принципов, основанных на простых и чётких именах свойств, которые легко запоминать и использовать [16].

# 3. ВНЕДРЕНИЕ ГУГЛ-СЕРВИСОВ В СИСТЕМЫ, ИСПОЛЬЗУЮЩИЕ VUEJS

3.1. **Описание структуры приложения**

В ходе данной работы были рассмотрены преимущества гибридных web технологий, были изучены их основные элементы, а также, как работают ключевые технологии, входящие в данный набор, такие как Firebase, Google OAuth, Cordova, Vue, автономная работа приложений и другие.Приложение имеет следующие функциональные возможности:

* Авторизация пользователя - имея логин и пароль, пользователи, зарегистрированные в системе, имеют возможность войти в систему;
* Список растрат, сделанных членами семьи - пользователи имеют доступ к списку месячных растрат, сделанным ими, и / или тем, которые находятся на их уровне доступа;
* Модуль отчетов, в котором есть список отчетов, доступ к которым так же определяется в зависимости от уровня доступа.

Авторизация пользователей - это первый шаг, который должны сделать пользователи. Это делается путем ввода имени пользователя и пароля, полученных от системного администратора. Если логин и / или пароль были введены неправильно, пользователю показываются соответствующие сообщения, так что он понимает, какие данные были введены неправильно.

После авторизации, пользователь перенаправляется на свою главную страницу. Он может содержать различную информацию, такую как: карточка, содержащая информацию о совершенной покупке, ее цене и, по возможности, картинке; имени пользователя, изображении профиля, почты.

Пользователь, в зависимости от уровня доступа, имеет возможность просматривать, оценивать и добавлять собственные записи.

3.2. Установка и настройка среды разработки

3.2.1. Node Package Manager (NPM)

Рекомендуется использовать NPM при создании больших приложений на Vue. Эта опция прекрасно работает с такими инструментами сборки, как Webpack и Browserify. Vue также имеет совместимые инструменты для использования однофайловых компонентов. Команда установки: npm install vue устанавливает последнюю стабильную версию.

3.2.2. Инструменты командной строки (CLI)

Vue.js предоставляет инструкции (CLI) для быстрого создания инфраструктуры для амбициозных одностраничных приложений. Всего за несколько минут вы получите рабочую конфигурацию с перезагрузкой модулей, анализом ошибок при настройке производственной сборки.

CLI - это инструмент для тех, кто знаком с Node.js и соответствующими инструментами сборки. Команда установки: npm install vue-cli.

3.2.3. Использование Cordova для определения гибридной трансляции

Средство командной строки Cordova распространяется как пакет npm в готовом к использованию формате. Нет необходимости для его компиляции из исходного кода.

Чтобы установить инструмент командной строки cordova, необходимо выполнить следующие действия:

* Выполнить предыдущие 2 пункта;
* Установить модуль cordova с помощью инструмента npm Node.js.

Модуль cordova будет автоматически загружен npm при вводе команды npm install –g cordova.

3.2.3. Создание приложения

Разработка приложений в контексте рабочего пространства VueJs. Рабочая область содержит файлы для одного или нескольких проектов. Чтобы создать новое рабочее пространство и начальный проект для приложения, нужно выполнить команду CLI: vue create vue-app, vue-app в данном случае это название создаваемого проекта, оно может быть произвольным.

Следуя этим командам, все файлы, необходимые для начала создания приложения, а также основные компоненты App.Vue и main.js динамически создаются в выбранной папке (Рис.2.2). Далее нужно установить Vuetify. Для этого нужно ввести команду *vue add vuetify.* После установки Vuetify нужно установить cordova через команду *vue add cordova*.

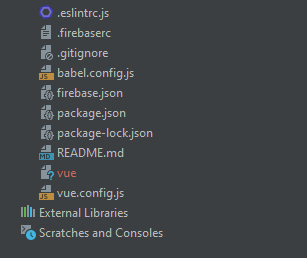
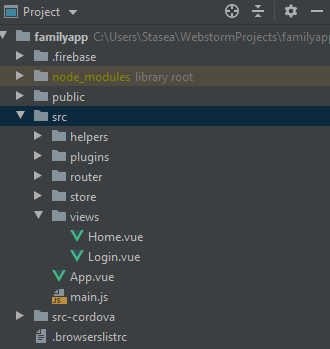


Рис.2.2. Стуктура vue vli проекта.

3.3. Создание проекта в Firebase

Если учетная запись Google уже существует, можно сразу войти в систему на странице https://firebase.google.com/. В противном случае необходимо зарегистрироваться по адресу http:// accounts.google.com. После входа можно создать свой проект. После нажатия кнопки Add project (Добавить проект) нужно ввести название проекта и выбрать страну. Нажать кнопку Create Project (Создать проект), чтобы перейти в консоль Firebase. Здесь конфигурируется база данных и аутентификация, а также генерируются ключи, которые нужны для начала работы. В разделе Database (База данных) есть два варианта: Realtime Database и Cloud Firestore. В проекте используется Realtime Database.

3.4. Интеграция Firebase

После настройки Firebase, следует его интеграция в приложение. Для этого файлы из библиотеки Firebase импортируются в файл проекта main.js.

import \* as firebase from "firebase/app";

import "firebase/analytics";

import "firebase/auth";

import "firebase/storage";

import "firebase/firestore";

Далее необходимо авторизовать приложение в Firebase. Для этого приложение инициализируется следующих свойств:

firebase.initializeApp({

apiKey: "AIzaSyDSwwHp51-LlwzIak0WXF8cScPMSS9VBwU",

authDomain: "family-d2146.firebaseapp.com",

databaseURL: "https://family-d2146.firebaseio.com",

projectId: "family-d2146",

storageBucket: "family-d2146.appspot.com",

messagingSenderId: "117478996669",

appId: "1:117478996669:web:e16a09e0f1bcf69ccfd5d4"

});

Для того, чтобы firebase был доступен глобально для всех компонентов приложения, необходимо написать следующее:

Vue.prototype.$firebase = firebase;

Далее происходит создание файла Login.vue, служащего для авторизации пользователя в приложении. Следом происходит создание интерфейса авторизации при загрузке страницы.

this.ui = new this.auth.AuthUI(this.$firebase.auth());

Далее был отрисован интерфейс в таг с id #firebaseui-auth-container.

this.ui.start("#firebaseui-auth-container")

<v-container class="auth-container" fluid>

<div id="firebaseui-auth-container"></div>

</v-container>

Вслед за этим новый объект интерфейса(инстанс) интерфейса передает callback, который вызывается при успешной регистрации.

signInSuccessWithAuthResult: authResult => {

const { user } = authResult;

this.$store.dispatch("setUserInfo", {

displayName: user.displayName,

email: user.email,

photoURL: user.photoURL

});

this.$router.push({ name: "Home" });

Код описанный выше запишет в глобальное хранилище store все данные пользователя для дальнейшего из использования в меню и при сохранении покупок приложения и перенаправит его на главную страницу.

Далее были описаны методы авторизации в приложении

signInOptions: [

this.$firebase.auth.GoogleAuthProvider.PROVIDER\_ID,

this.$firebase.auth.EmailAuthProvider.PROVIDER\_ID

]

Далее был написан метод для проверки авторизации при загрузке страницы и обновления данных в store, если они актуальны, в ином случае переадресации на страницу авторизации

this.$firebase.auth().onAuthStateChanged(user => {

if (user) {

this.$store.dispatch("setUserInfo", {

displayName: user.displayName,

email: user.email,

photoURL: user.photoURL

});

} else {

this.$router.push({ name: "Login" });

}

});

После проверки авторизации пользователь должен подписаться на обновления всех покупок и получить текущий список.

this.$firebase

.firestore()

collection("goods")

.orderBy("date", "desc")

onSnapshot(querySnapshot => {

this.items = [];

querySnapshot.forEach(doc => {

this.items.push(doc.data());

});

});

Затем из базы данных загружается firestore отсортированная коллекция goods и записывается в массив items для отрисовки на странице

<v-card class="mx-auto" :key="index" v-for="(item, index) of items">

<v-list-item>

<v-list-item-avatar color="grey">

<v-img :src="item.photoURL"></v-img>

</v-list-item-avatar>

<v-list-item-content>

<v-list-item-title class="headline">{{

item.displayName

}}</v-list-item-title>

<v-list-item-subtitle>{{ item.email }}</v-list-item-subtitle>

</v-list-item-content>

</v-list-item>

<v-img v-if="item.image" :src="item.image"></v-img>

<v-card-text>

{{ item.text }}

</v-card-text>

<v-card-actions>

<v-btn text color="deep-purple accent-4"> {{ item.price }} $ </v-btn>

<v-spacer></v-spacer>

<v-btn text color="deep-purple accent-4">

{{ getFormattedDate(item.date) }}

</v-btn>

</v-card-actions>

</v-card>

После отрисовки данных пользователь может просмотреть, оценить или добавить новые записи. Ниже приведен пример кода для отображения кнопки добавления

<v-btn @click="dialog = true" color="primary" bottom right>

<v-icon>mdi-plus</v-icon>

</v-btn>

При нажатии на кнопку пользователю будет предложено создать новую запись, написать название, цену и прикрепить картинку. После сохранения покупки данные будут отравлены на сервер.

if (this.image) {

const snapshot = await this.$firebase

.storage()

.ref()

.child(`images/${this.generateUuid()}`)

.put(this.image);

image = await snapshot.ref.getDownloadURL();

}

Код выше проверяет наличие картинки, и загружает ее на сервер. Для того что бы они не повторялись для каждой был сгенерирован уникальный номер. После загрузки картинки на сервер, в переменную записывается ее ссылка для сохранения базе данных. После получения ссылки, все данные сохраняются в базу данных, в коллекцию под названием goods. Ниже приведен пример кода:

try {

await this.$firebase

.firestore()

.collection("goods")

.add({

text: this.text,

price: this.price,

date: new Date(),

...this.user,

image

})

После добавления данных в базу диалоговое окно будет закрыто, а данные полей сброшены.

# Заключение

В данной работе было произведено исследование гибридных web приложений. Был произведен анализ создания web приложений применяя соответствующий набор технологий.

В процессе выполнения работы были:

* Изучены основы гибридных web приложений;
* Исследованы возможности использования гибридных web технологий в качестве замены нативных технологий;
* Анализ работы современных облачных технологий для интеграции в приложения;
* Разработано приложение с использованием данного набора технологий.

В ходе данной работы были рассмотрены преимущества гибридных web технологий, были изучены их основные элементы, а также, как работают ключевые технологии, входящие в данный набор, такие как Firebase, Google OAuth, Cordova, Vue, автономная работа приложений и другие.

На основе всего вышесказанного, можно сформулировать вывод, что гибридные web технологии являются крайне полезными и инновационными инструментами для создания мобильных приложений. Гибридные web приложения включают в себя большинство преимуществ нативных приложений, при этом технически оставаясь - web приложением.

В заключении можно добавить, что в перспективе гибридные web технологии позволят убрать проблему кроссплатформенности, так как они позволяют вести разработку одновременно под все платформы сразу. Это приведет к более прогрессивному, прямолинейному и рентабельному пути разработки приложений, ведь теперь компаниям не нужно будет беспокоиться о разработке приложения сразу под несколько разных платформ, что выражается не только в количестве времени, потраченного на разработку, но и количества и специализации самих разработчиков нужных для разработки.

Данная работа предназначена для изучения гибридных web приложений, а также для ознакомления с методами и архитектурой web приложений на основе гибридных web технологии, также может быть полезна разработчикам, студентам, учителям и всем желающим ознакомиться с гибридными web приложениями.

# Библиография

1. *Поверхностный обзор javascript-enabled BaaS платформ*. Доступен: <https://habr.com/ru/post/150729/> [on-line].[цитирован 15.03.20];
2. *Firebase secures its real-time back-end service.* Доступен: <https://gigaom.com/2012/12/18/firebase-secures-its-real-time-back-end-service/>[on-line].[цитирован 20.03.20];
3. *Gerardus Blokdyk. Firebase The Ultimate Step-By-Step Guide. Emereo Pty Ltd*. ISBN 9780655321477;
4. *Используйте Firebase, единый кроссплатформенный SDK от Google, чтобы улучшить приложения.* Доступен: <https://developer.android.com/distribute/best-practices/develop/build-with-firebase?hl=ru> [on-line].[цитирован 27.03.20];
5. *Sourav BHOI. Internet of Things: an Application Based Approach Using Arduino Platform and Firebase. Independently Published, 2018* ISBN 1983041378, 9781983041372;
6. *Обзор облачных сервисов для разработки бэкенда мобильных приложений*. Доступен: <https://habr.com/ru/company/surfstudio/blog/463435/> [on-line].[цитирован 07.04.20];
7. *OAuth 2.0 простым и понятным языком*. Доступен: <https://habr.com/ru/company/mailru/blog/115163/> [on-line].[цитирован 10.04.20];
8. Описание сервиса Google Maps <https://wiki.rookee.ru/google-maps/> [on-line].[цитирован 10.04.21];
9. *Использование сервисов Firebase при разработке мобильных приложений*. Доступен: <https://cmsmagazine.ru/journal/cases-4264/> [on-line].[цитирован 16.04.20];
10. *Нативные, веб и гибридные приложения: что выбрать?* Доступен: <https://smartum.pro/ru/blog-ru/native-web-and-hybrid-apps/> [on-line].[цитирован 26.04.20];
11. *Нативные vs. гибридные приложения*. Доступен: <https://umbrellait.com/ru/blog/native-vs-hybrid-app/> [on-line].[цитирован 04.05.20];
12. *Разница между нативным и гибридным мобильным приложением*. Доступен: <https://wezom.com.ua/blog/chem-otlichajutsja-nativnoe-i-gibridnoe-mobilnye-prilozhenija> [on-line].[цитирован 10.05.20];
13. *Vue.js*. Доступен: <https://ru.vuejs.org/v2/guide/index.html> [on-line].[цитирован 10.05.20];
14. *Callum Macrae. Vue.js: Up and Running., O’Reilly, 2017., 219 с.,* [ISBN 9781491997246](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9781491997246);
15. *Alex Kyriakidis, Kostas Maniatis.*[*The Majesty of Vue.js*](https://books.google.ru/books?id=Xp7cDgAAQBAJ)*., Packt Publishing Ltd, 2016., 230 с.,* [ISBN 9781787125209](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9781787125209);
16. *VuetifyJS — строим свое приложение в стиле Material Design*. Доступен: <https://weatherless.ru/javascript/vuejs/vuetifyjs-stroim-svoe-prilozhenie-v-stile-material-design/> [on-line].[цитирован 20.05.20];
17. *Хэнчеп Эрик, Листуон Бенджамин Х88 Vue.js в действии., СПб.: Питер, 2019., 304 с.: ил., (Серия «Библиотека программиста»). Стр. 257*. ISBN 9785446110988

Declaralie privind asumarea raspunderii

Subsemnata \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, absolventa al Universitdtii de Stat “Alecu Russo” din Balti, Facultatea de \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ program de studii \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, declar pe propria raspundere ca teza de licenta/teza de master cu titlul \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, elaborata sub indrumarea dlui/dnei \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ este rezultatul propriilor cercetdri.

Declar ca nu am plagiat altd lucrare de licenta/master, monografii, articole etc., publicate sau postate pe Internet, toate sursele utilizate in tezi fiind mentionate in cuprinsul acesteia.

De asemenea, declar ca sunt de acord ca teza de licenta/teza de master sa fie verificata pentru confirmarea originalitilii.

Data Semnatura