МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК

Кафедра программного обеспечения

РЕКОМЕНДОВАНО К ЗАЩИТЕ В ГЭК

Заведующий кафедрой

к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.С. Воробьева

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 года

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

бакалавра

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ИНФОРМАЦИОННОЙ ПОДДЕРЖКИ ВНЕУЧЕБНЫХ МЕРОПРИЯТИЙ В ВУЗЕ

02.03.03 Математическое обеспечение и администрирование

информационных систем

Профиль «Технологии программирования»

Выполнила работу Синицина

студентка 4 курса Анна

очной формы обучения Вячеславовна

Руководитель Ступников

к.ф-м.н., доцент Андрей

Анатольевич

Тюмень 2021

**ОГЛАВЛЕНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc74045050)

[**ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ** 6](#_Toc74045051)

[**1.1.** **ТЕНДЕНЦИЯ РАЗВИТИЯ В ОБЛАСТИ ВНЕУЧЕБНЫХ СТУДЕНЧЕСКИХ МЕРОПРИЯТИЙ** 6](#_Toc74045052)

[**1.2.** **ПРОБЛЕМА ИНФОРМИРОВАНИЯ СТУДЕНТОВ** 7](#_Toc74045053)

[**1.3.** **ОБЩАЯ ПОСТАНОВКА ЗАДАЧИ РАЗРАБОТКИ РЕКОМЕНДАТЕЛЬНОЙ СИСТЕМЫ** 8](#_Toc74045054)

[**1.4.** **ПОДХОДЫ ПОСТРОЕНИЯ РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ** 10](#_Toc74045055)

[**1.4.1.** **ФИЛЬТРАЦИЯ ПО СОДЕРЖАНИЮ** 10](#_Toc74045056)

[**1.4.2.** **КОЛЛАБОРАТИВНАЯ ФИЛЬТРАЦИЯ** 11](#_Toc74045057)

[**ГЛАВА 2. ПОСТАНОВКА ЗАДАЧИ** 14](#_Toc74045058)

[**2.1.** **ФОРМАЛЬНАЯ ПОСТАНОВКА ЗАДАЧИ РЕКОМЕНДАЦИИ МЕРОПРИЯТИЙ** 14](#_Toc74045059)

[**2.2.** **ВЫБОР ПОДХОДА И МЕТОДА** 14](#_Toc74045060)

[**ГЛАВА 3. РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ** 16](#_Toc74045061)

[**3.1.** **АРХИТЕКТУРА ВЕБ-ПРИЛОЖЕНИЯ** 16](#_Toc74045062)

[**3.2.** **СТРУКТУРА БАЗЫ ДАННЫХ** 17](#_Toc74045063)

[**3.3.** **РЕАЛИЗАЦИЯ РЕКОМЕНДАТЕЛЬНОЙ СИСТЕМЫ** 19](#_Toc74045064)

[**3.4.** **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ** 20](#_Toc74045065)

[**ЗАКЛЮЧЕНИЕ** 22](#_Toc74045066)

[**СПИСОК ЛИТЕРАТУРЫ** 24](#_Toc74045067)

# **ВВЕДЕНИЕ**

В настоящее время студенческая жизнь предполагает участие в различных внеучебных мероприятиях, особенно на первых курсах обучения, когда всем важно влиться в коллектив и обрести новые знакомства. Сейчас в вузах появляется множество студенческих объединений, которые инициируют и организуют мероприятия для учащихся, будь то интеллектуальные соревнования, музыкальные конкурсы, концерты, спортивные игры и многое другое. Следовательно, организаторам важно иметь каналы связи с потенциальными и фактическими участниками, а им – получать актуальную информацию не только о проводимых мероприятиях, но и знать историю посещенных.

В современной взаимосвязи организаторов университетских мероприятий и участников существует проблема обмена информацией на расстоянии. Организаторы вынуждены самостоятельно искать различные способы для обеспечения полного цикла проведения мероприятия. Ознакомительную информацию они публикуют в социальных сетях, регистрация проводится с помощью сервисов, подобных Google Forms, связь с организаторами (при необходимости задать тот или иной вопрос) производится через мессенджеры. Все данные обрабатываются людьми вручную.

Вследствие подобной разобщенности отсутствует возможность иметь базу данных всех участников, историю проведенных мероприятий и их результаты. К тому же информированными остаются только те, кто уже знают о том или ином студенческом объединении, а привлечение новых участников остаётся трудной задачей. Всё это, очевидно, не лучшее решение данной проблемы.

Одним из решений поставленной проблемы является использование рекомендательной системы в информационном веб-приложении для связи организаторов мероприятий с фактическими и потенциальными участниками. Рекомендательная система подбирает и предлагает пользователю релевантный контент, основываясь на своих знаниях о пользователе, контенте и взаимодействии пользователя и контента. [ССЫЛКА, Фальк, с. 36]. Есть несколько подходов для реализации рекомендательной системы. Одним из подходов, основанных на персонализированных рекомендациях, является совместная или коллаборативная фильтрация (collaborative filtering). Для построения прогнозов она использует известные предпочтения группы пользователей для прогнозирования неизвестных предпочтений другого пользователя. Допущение этого подхода в контексте имеющейся проблемы основывается на следующем: пользователи, которые посетили какие-либо мероприятия в прошлом, будут склонны посетить похожие мероприятия в будущем.

Возможным вариантом решения может стать веб-сервис, который позволит агрегировать весь вышеописанный функционал для удобства всех студентов – как и участников, так и организаторов. В единую базу данных будет вноситься информация о пользователях, студенческих сообществах и мероприятиях. Организаторы смогут создавать свои страницы и размещать информацию о проводимых встречах. Все пользователи будут иметь возможность не только оставаться информированными о них, но также регистрироваться и наблюдать историю своей внеучебной студенческой активности. Организаторам к тому же данный сервис поможет найти новую аудиторию благодаря рекомендательной системе.

Цель работы: разработка веб-приложения для информационной поддержки внеучебных студенческих мероприятий в вузе, обеспечивающего систему рекомендаций.

Задачи работы:

1. Изучить подходы и методы разработки рекомендательных систем
2. Спроектировать архитектуру веб-приложения
3. Разработать базу данных
4. Заполнить базу данных реально существующими мероприятиями
5. Разработать программный продукт:

* Разработать модуль аутентификации и авторизации пользователей
* Разработать модули для сбора информации о мероприятиях и их участниках
* Разработать модули для поиска информации об организаторах, мероприятиях и их участниках
* Разработать рекомендательную систему для отображения пользователю потенциально подходящих мероприятий

# **ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

* 1. **ТЕНДЕНЦИЯ РАЗВИТИЯ В ОБЛАСТИ ВНЕУЧЕБНЫХ СТУДЕНЧЕСКИХ МЕРОПРИЯТИЙ**

Студенчество – активный период становления человека как члена общества, личности в целом, проявления самых разнообразных интересов: занятие спортом, художественных, технических и научных достижений. [sotsiokulturnaya-deyatelnost-] В том возрасте, на который обычно приходится время учебы в вузе, личность развивает свою индивидуальность и задействует все составляющие своей самоидентификации. [notv\_2013\_32.pdf]

В процессе социализации студентов в вузах, особенно первокурсников, сложен. Учеба на первом курсе является стрессом: появляется новое окружение, с которым студенту требуется находить контакт; более сложные предметы; непривычный стиль обучения и многое другое.

Для решения этой проблемы внутри вуза формируются студенческие объединения по тем или иным интересам. Одни студенты сами или при поддержке образовательного заведения организовывают и проводят мероприятия для других студентов, своих единомышленников. Подобная практика помогает студентам создавать коммуникативные связи не только со своими одногруппниками, но и с людьми с других учебных направлений, что значительно расширяет круг общения человека и помогает быстрее адаптироваться в незнакомой обстановке.

Сейчас студенческие сообщества сильно развиты в учебных заведениях по всей России. В одном только Тюменском государственном университете их насчитывается больше тридцати: от танцевальных и театральных студий до интеллектуальных клубов и научных объединений. Мероприятия могут представлять из себя обычные встречи или же соревнования по тем или иным дисциплинам. Некоторые из объединений проводят всего несколько мероприятий в год, другие – могут организовывать до десятка встреч в месяц.

С развитием информационных технологий изменился и процесс подготовки мероприятий для студентов. Теперь нет необходимости каждый раз развешивать объявления по всему учебному кампусу – достаточно сделать пост в социальной сети, чтобы проинформировать студентов. Во время карантина 2020-2021 годов еще раз подтвердилась значимость технологий во взаимодействии организаторов и участников студенческих мероприятий – все встречи стали проводиться удаленно с помощью Zoom, Microsoft Teams и других сервисов, предоставляющих возможность создавать онлайн-конференции для множества человек.

* 1. **ПРОБЛЕМА ИНФОРМИРОВАНИЯ СТУДЕНТОВ**

Как уже было упомянуто в предыдущей части, сегодня используются самые различные инструменты для информирования студентов об актуальных мероприятиях, проводимых в вузе, и их дополнительной информационной поддержке.

Обычно информирование студентов о проводимых мероприятиях реализуется посредством социальных сетей, в большинстве случаев такое объявление представляет собой пост в сообществе данного студенческого объединения в социальной сети Вконтакте.

Для регистрации на мероприятие используют сервисы, подобные Google Forms, Яндекс.Формы – создание небольших анкет, включающих основную информацию об человеке, чтобы можно было заранее узнать приблизительное количество участников и, исходя из этого, сделать выбор о месте проведения или подготовиться к следующим встречам.

Взаимодействие организаторов с участниками по поводу тех или иных вопросов происходит или с помощью социальной сети, или посредством мессенджеров (WhatsApp, Telegram, Viber и др.).

К тому же в зависимости от специфики мероприятий некоторые студенческие объединения для решения своих задач пользуются дополнительными сервисами. Так Киберспортивная лига ТюмГУ, которая занимается организацией соревнований внутри вуза по компьютерному спорту, нуждается в портале для формирования и отображения турнирных таблиц.

Значимую проблему представляет собой именно информирование студентов о проводимых встречах, потому что это тот этап, который проходят абсолютно все организаторы, к какой бы сфере ни относилось их сообщество. Сейчас разные студенческие объединения используют разные сервисы для информирования участников.

Организаторы сталкиваются с проблемой неструктурированности данных об участниках, потому что сервисы по типу Google Forms дают им информацию о регистрации на одно конкретное мероприятие. Если они захотят иметь свою базу участников, то могут сделать это только посредством ручных сбора и обработки информации о каждом мероприятии.

Другой задачей для них является привлечение потенциальных участников к посещению мероприятия и нахождение новых единомышленников. Размещение информации в небольшом сообществе социальной сети приносит не так много пользы, такую запись смогут увидеть только те студенты, которые уже знают про данное объединение и интересуются его активностью. К тому же сами студенты тоже могут не знать о том, какие объединения существуют и какое мероприятие посетить.

Эта проблема может быть решена с помощью реализации веб-приложения, содержащего рекомендательную систему, которая позволит для каждого пользователя составить список мероприятий, которые им могут быть интересны.

* 1. **ОБЩАЯ ПОСТАНОВКА ЗАДАЧИ РАЗРАБОТКИ РЕКОМЕНДАТЕЛЬНОЙ СИСТЕМЫ**

Рекомендательные системы – это новое направление информационных технологий, связанное с выявлением предпочтений пользователей. [статья razrabotka-rekomendatelnoy-…] Это персонализированная технология фильтрации информации, используемая для прогнозирования, понравится ли конкретному пользователю определенный элемент (задача прогнозирования), или же для идентификации набора из элементов, интересных определенному пользователю (задача рекомендации топ-).

Математическая постановка задачи может быть представлена следующим образом:

– множество пользователей,

– множество элементов.

– матрица рейтингов размера на , где на месте будет стоять некоторое число, если пользователь оценил элемент , и пусто в противном случае,

– требуемое количество рекомендаций, которые необходимо получить от разрабатываемой системы.

Для пользователя необходимо найти -мерный вектор , где элементы не оценены им и в то же время удовлетворяют предпочтениям пользователя , то есть в том случае, когда прогнозы имеют наибольшее значение. [Погорельская]

Выделяются два типа рекомендательных систем: персонализированные и неперсонализированные.

Неперсонализированная рекомендательная система создает рекомендации на основе общего рейтинга объекта (выше – элемента). Данная рекомендация не зависит от пользователя, который в ней нуждается. Результат системы – одна и та же рекомендация на один и тот же запрос. [ibs-28-p20.pdf] Главным из преимуществ этого типа является простота реализации, пользователь минимально взаимодействует с системой. Но в такой рекомендации не учитываются предпочтения отдельного пользователя, следовательно, она с малой вероятностью понравится большой части пользователей.

Персонализированная рекомендательная система учитывает предпочтения каждого пользователя, поэтому работает гораздо эффективнее описанной выше неперсонализированной. Сперва рассчитывается схожесть между пользователями или элементами, а уже затем на основе этого строится рекомендация.

* 1. **ПОДХОДЫ ПОСТРОЕНИЯ РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ**

В настоящее время применяются различные подходы для построения персонализированных рекомендательных систем.[Погорельская] В данной работе описываются наиболее популярные из подходов: контент-ориентированный (content-based) метод и коллаборативная фильтрация (collaborative filtering).

* + 1. **ФИЛЬТРАЦИЯ ПО СОДЕРЖАНИЮ**

Суть контент-ориентированного подхода (CBF) заключается в том, чтобы пользователей сопоставить объекту или каким-либо другим вещам, которые им могли бы понравиться. Он сравнивает понравившиеся пользователю элементы с другими и рекомендует наиболее похожие.

Одним из преимуществ является то, что для формирования рекомендаций нет необходимости в большой базе пользователей. К тому же новые объекты, которые еще не будут иметь своего рейтинга, все равно могут быть рекомендованы пользователям.

С другой стороны, CBF имеет ряд недостатков. К ним относят проблему нового пользователя (холодного старта): когда в системе появляется новый пользователь, история его предпочтений пуста и нет объектов, на основе которых можно было бы формировать рекомендации. Вторая проблема состоит в предсказуемости: система будет предлагать только то, что пользователю нравилось ранее, он не получится нетипичных для себя рекомендаций.

Для построения подобных рекомендательных систем в основном применяется наивный байесовский классификатор, а также другие различные методы машинного обучения: нейронные сети, деревья решений или кластеризация.

* + 1. **КОЛЛАБОРАТИВНАЯ ФИЛЬТРАЦИЯ**

Коллаборативная фильтрация (CF) основана на предпочтениях группы пользователей. Суть подхода заключается в нахождении пользователей с общими интересами, а затем рекомендует пользователю те объекты, которые были интересны (высоко оценены) похожим на него пользователям.

Такой подход считается универсальным, потому что для его работы достаточно знать историю оценок данного пользователя и похожих на него.

Из этого следует, что такой подход тоже сталкивается с проблемой холодного старта: новому пользователю без истории предпочтений нельзя найти похожих пользователей, а следовательно – не получится сформировать хорошие рекомендации. К тому же система не сможет предложить пользователям те объекты, которые еще не были никем оценены. Другой проблемой является разреженность матрицы пользователей-рейтингов при большом числе объектов – это затрудняет поиск пользователей, которые оценили одни и те же объекты. Также система не способна делать рекомендации пользователям с уникальными вкусами (из-за отсутствия похожих).

Системы CF подразделяются на основанные на эвристических методах (на памяти, memory-based) и основанные на построении моделей предпочтения (model-based).

В подходе **на основе эвристических методов** прогноз делается при учете всех тех объектов, которые были оценены пользователем ранее. Для этого подхода запоминается матрица полезности, и рекомендации составляются путем запроса данного пользователя к остальной части матрицы полезности.

Такой подход в свою очередь подразделяется на коллаборативную фильтрацию пользователь-пользователь (user-based) и объект-объект (item-based).

В первом случае выбирается подмножество пользователей на основе их сходства, далее взвешенная комбинация рейтингов используется для прогноза рейтинга, который поставит данный пользователь.

В общем случае алгоритм будет выглядеть следующим образом:

1. Каждому пользователю присваивается мера сходства с данным пользователем . Для измерения сходства между пользователями могут использоваться коэффициент корреляции Пирсона, Евклидово расстояние или косинусная мера и т.д. Наиболее подходящей и доступной из них является косинусная мера:

где – рейтинг, присвоенный пользователем объекту , а – рейтинг, присвоенный пользователем объекту .

1. Выбирается число – количество похожих на пользователей.
2. Для каждого объекта рассчитывается предсказание о том, как целевой пользователь может оценить его:

где – прогноз рейтинга, поставленного пользователем объекту .

Метод item-based работает по подобному принципу, но в рамках него рассматривается схожесть не пользователей между собой, а объектов. Два элемента считаются похожими, если большинство пользователей, которые взаимодействовали с ними обоими, делали это одинаково. Этот метод называется «центрированным на элементах», поскольку он представляет элементы на основе взаимодействий, которые пользователи имели с ними, и оценивает расстояния между этими элементами.

**Подход, основанный на модели предпочтений**, предполагает, что сходство между объектами и пользователями вызвано некоторой моделью скрытого взаимодействия. Одна такая предварительная модель разрабатывается на основе имеющихся данных. Когда появляется запрос от целевого пользователя, этот подход быстро в ответ дает его предпочтения.

Этот подход не является гибким, потому что построение подобной модели – ресурсоемкий процесс, в такую систему сложнее добавлять данные. Из-за отказа в использовании всего доступного набора данных прогнозы могут быть менее точны, чем при имплементации эвристической системы.

Существуют различные методы поиска скрытых признаков. Среди них есть матричная факторизация (MF, может представлять собой итерационный подход, основанный на градиентном спуске) и сингулярное разложение матриц (SVD). Если задача рекомендательной системы становится задачей без учителя, то применяется кластеризация.

В таблице 1 отображен сравнительный анализ основных подходов к построению рекомендательных систем.

Таблица 1

Подходы к построению рекомендательных систем

|  |  |  |
| --- | --- | --- |
| Рекомендательные системы | Преимущества | Недостатки |
| Неперсонализированные | Простота реализации. | Одна рекомендация для всех пользователей, предпочтения конкретного не учитываются. |
| Фильтрация по содержанию | Не нужна большая база пользователей.  Объекты без оценки все равно будут рекомендованы. | Нечего рекомендовать новому пользователю.  Рекомендации будут типовыми для пользователя. |
| Коллаборативная фильтрация | Универсальны.  Рекомендации не соответствуют строго предыдущим оценкам пользователя.  Нет необходимости в большом объеме информации о пользователях или объектах. | Проблема холодного старта.  Не рекомендует то, что еще никто не оценил. |

# **ГЛАВА 2. ПОСТАНОВКА ЗАДАЧИ**

* 1. **ФОРМАЛЬНАЯ ПОСТАНОВКА ЗАДАЧИ РЕКОМЕНДАЦИИ МЕРОПРИЯТИЙ**

Дано:

– множество мероприятий, где

– множество пользователей, где

Тогда имеет место матрица:

,

где – индикатор того, посетил ли пользователь мероприятие принимает значение 1, если посетил, иначе – 0)

Требуется:

Найти для пользователя -мерный вектор , где мероприятия не посещены им, актуальны (еще не завершены) и в то же время удовлетворяют предпочтениям пользователя , то есть в том случае, когда прогнозы имеют наибольшее значение.

* 1. **ВЫБОР ПОДХОДА И МЕТОДА**

Исходя из сравнительного анализа в таблице 1 и постановки задачи в предыдущем параграфе, можно сделать вывод о том, что лучше всего в контексте рекомендации мероприятий подходит коллаборативная фильтрация на основе эвристических методов.

В первую очередь она не ресурсоёмка, что позволит не перегрузить приложение, запросы будут выполняться быстро. Также в рамках данной работы можно будет ограничиться генерацией лишь самой основной информации о пользователях – того, какие мероприятия они посещали.

Фильтрация по содержанию не подходит, потому что при рекомендации мероприятий студентам мы не можем формировать прогнозы для них, основываясь лишь на их собственной истории посещения мероприятий – в таком случае будут рекомендовать все время одни и те же встречи.

В качестве метрики для нахождения ближайших соседей (то есть наиболее похожих пользователей) для целевого пользователя используется косинусная мера, описанная в предыдущей главе.

Проблема холодного старта, с которой можно столкнуться при выборе коллаборативной фильтрации в качестве подхода, может быть разрешима с помощью определения тех свойств пользователей, которые могли бы говорить об их схожести. В данном случае таким свойством был выбран институт пользователя. При регистрации каждый пользователь указывает институт, в котором обучается, после этого ему рекомендуются те мероприятия, которые наиболее популярны у других студентов этого института – а именно, чем больше студентов того же института посетило это мероприятие, тем выше оно в списке рекомендаций.

# **ГЛАВА 3. РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ**

* 1. **АРХИТЕКТУРА ВЕБ-ПРИЛОЖЕНИЯ**

Для поставленной задачи наилучшим решением является использование клиент-серверной архитектуры.

Общая архитектура реализуемого приложения отображена на рисунке 1.

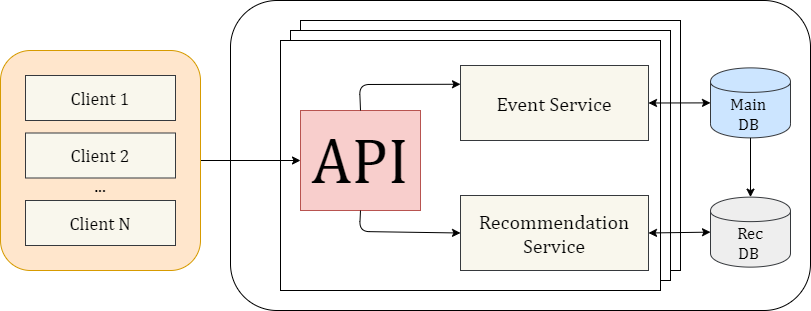


Рис. 1. Общая архитектура веб-приложения

На схеме изображены два основных модуля:

1. Event Service – модуль, отвечающий за сбор, обработку данных о пользователях, мероприятиях и студенческих сообществах.
2. Recommendation Service – модуль, отвечающий за формирования списка рекомендованных мероприятий для каждого пользователя.

Также тут отображены два базы данных: одна основная, содержащая информацию обо всех сущностях (пользователях, мероприятиях, студенческих объединениях), вторая – агрегирует только те данные, которые необходимы для создания рекомендаций, то есть данные о том, какой пользователь какие мероприятия посетил.

В качестве СУБД для них была выбрана реляционная PostgreSQL, причиной этому стал обширный список структур и типов данных, которые они поддерживает. Кроме числовых, с плавающей точкой, текстовых, булевых и других ожидаемых типов данных (а также множества их вариаций), PostgreSQL может похвастаться поддержкой XML, JSON, массивов, композитных типов и диапазонов. Поддержка JSON в контексте данной работы наиболее важна, чтобы в дальнейшем была возможность фиксировать результаты мероприятий – так с помощью JSON можно было бы хранить турнирные таблицы в одном поле записи таблицы.

Клиентской частью выступает веб-приложение, написанное с использованием фреймворка React JS, на котором созданы компоненты интерфейса для одностраничного приложения. Оно поддерживает регистрацию и авторизацию пользователей, а также реализовано разграничение прав. Кроме того, для создания форм и их валидации применяются возможности библиотеки Formik, а для

Пользователи с достаточным уровнем прав могут создавать и редактировать мероприятия и сообщества (создатели студенческих объединений).

Интерфейс должен отображать мероприятия, сгруппированные по студенческим сообществам, а также фильтровать их по следующим признакам:

1. По релевантности (пользовательские рекомендации)
2. По актуальности (активно мероприятия или завершено)
3. По виду мероприятия (встреча или соревнование, во втором случае добавляется фильтр по дисциплине)

Для подключения к базе и работы с данными используется Node.js, благодаря наличию большого числа библиотек для различных целей. Библиотека Express использовалась для маршрутизации запросов от клиента, а библиотека Passport как инструмент реализации аутентификации.

* 1. **СТРУКТУРА БАЗЫ ДАННЫХ**

PostgreSQL предоставляет различные методы аутентификации. Данная СУБД обеспечивает полную поддержку требований ACID и гарантирует изоляцию транзакций благодаря механизму многоверсионного управления одновременным доступом — транзакции на чтение никогда не блокируют транзакции на запись, и наоборот [Бартунов, с.22-24].

Диаграмма основной базы данных представлена на рисунке 2.

Основными сущностями являются: пользователи (users), студенческие объединения (communities) и мероприятия (events).

Можно заметить, что соревнования были выделены в отдельную таблицу. Это сделано для того, чтобы упростить дальнейшее масштабировании архитектуры, если будет необходимости добавить сущность «команда» для командных соревнований. Такую же роль выполняют все связи многие-ко-многим в таблицах users\_competitions, users\_events, users\_communities.

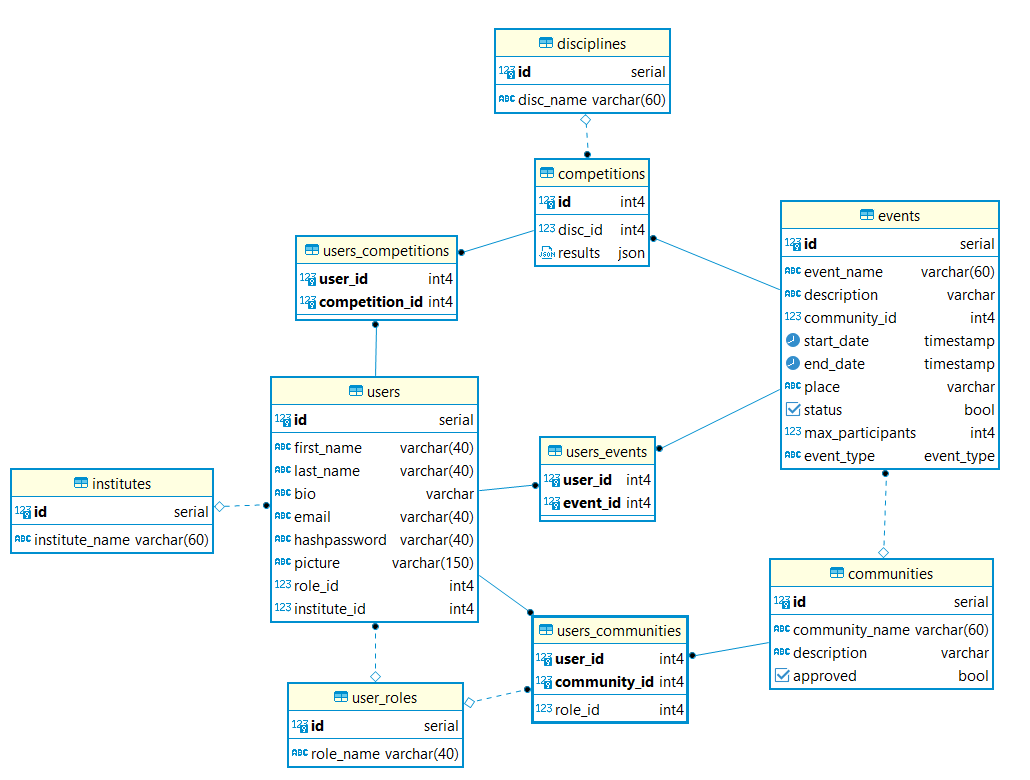


Рис. 2. Схема базы данных

Также каждому пользователю сопоставляется роль как в системе, так и в рамках сообщества – только владельцы (создатели) студенческих объединений могут иметь возможность создавать новые мероприятия и вносить изменения в старые.

База данных заполнена небольшим объемом данных – около 150 мероприятия – чтобы матрица полезности была менее разряжена.

Пример мероприятия вида «соревнование»:

* id: 56
* event\_name: "Своя игра на эко-тематику"
* description: "Второе мероприятие в рамках ВузЭкоФест – всем известная Своя игра, но только про эко-движение! В Zoom!"
* community\_id: 13
* start\_date: 2021-06-19 15:00:00
* end\_date: 2021-06-19 17:30:00
* place: "https://zoom.us/j/93494591217?pwd=SFdWRUIzN2l0YVlTUU"
* status: true
* max\_participants: null
* event\_type: competition
  1. **РЕАЛИЗАЦИЯ РЕКОМЕНДАТЕЛЬНОЙ СИСТЕМЫ**

В приложении 1 представлен программный код на языке Java Script для сортировки мероприятий, предлагаемых целевому пользователю, в соответствии с результатом коллаборативной фильтрации.

Входными данными для основного метода recommendate являются объект, ключами для полей которого являются идентификаторы всех пользователей (кроме целевого), а значением объект, ключами которого являются идентификаторы всех активных на данный момент мероприятий, причем в которых целевой пользователь еще не зарегистрирован, а значением – число 0 или 1 в зависимости от того, решил ли другой пользователь принять в нем участие. Именно эти мероприятия будут рекомендованы целевому пользователю, тут мы определяем порядок, в котором они будут отображены – от наиболее подходящего к наименее.

Вторым параметром передается результат метода knn – нахождения k-ближайших соседей. Там с помощью косинусной меры на основе посещения пользователями предыдущих мероприятий выбираются k таких, история участия которых была бы наиболее похожа на историю участия целевого пользователя в мероприятиях.

Функция cosine принимает на вход два объекта: вектор посещаемости мероприятий целевым пользователем и вектор посещаемости мероприятий другим пользователем (каждый из которых представлен в виде объекта той же структуры, описанной ранее). В ней с помощью косинусной меры из первой главы рассчитывается расстояние между векторами, результатом является число от 0 до 1, где 1 означает, что сравниваемые объекты идентичны.

Далее в методе recommendate эта мера используется в качестве веса, умножается на индикатор регистрации пользователя (0, 1) в активном мероприятии, в котором не участвует пользователь . Суммируется взвешенные индикаторы регистрации, затем сумма делится на число пользователей, зарегистрированных на активное мероприятие.

* 1. **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

Руководство пользователя приводится для обычного пользователя системы.

Для полноценной работы с веб-приложением требуется авторизация (рис. 3), несмотря на то, что информация о мероприятиях, студенческих объединениях и пользователях доступна всем.

Изображение выглядит как текст

Автоматически созданное описание

Рис. 3. Форма входа

На странице авторизации можно выполнить вход с помощью логина и пароля или перейти на страницу регистрации (рис.4), если учетной записи ещё нет.

Для регистрации пользователь должен указать свои имя и фамилию, уникальный Email, выбрать институт обучения, а также придумать пароль. Если указанный Email окажется неуникальным, то соответствующее предупреждение появится под полем ввода.

Изображение выглядит как текст

Автоматически созданное описание

Рис. 4. Форма регистрации

# **ЗАКЛЮЧЕНИЕ**

В ходе разработки веб-приложения были изучены и проанализированы подходы и методы построения рекомендательных систем, а в частности – контент-ориентированная и коллаборативная фильтрация, эвристические методы и основанные на модели.

Результатом выполнения работы является программный комплекс, состоящий из одностраничного веб-приложения и двух баз данных. Он позволяет пользователю-организатору информировать потенциальных участников об актуальных мероприятиях и также структурирует данные о студентах, а пользователю-участнику – следить за активностью студенческих объединений и получать рекомендации о том, какие внеучебные мероприятия из текущих могли бы понравиться именно ему.

Для достижения поставленной цели были решены следующие задачи:

1. Были изучены и проанализированы подходы и методы разработки рекомендательных систем.
2. Была разработана база данных и заполнена данными о реально проведенных внеучебных студенческих мероприятиях ТюмГУ.
3. Был разработан алгоритм генерации данных о посещении пользователями мероприятий.
4. Была реализована система рекомендаций внеучебных студенческих мероприятий для пользователей.
5. Было спроектировано и разработано веб-приложение.

В качестве дальнейшего развития этого проекта можно предложить добавление дополнительной сущности «команда» для командных соревновательных мероприятий (например, футбольные матчи, интеллектуальная игра «Что? Где? Когда?» или киберспортивный турнир по игре «Dota 2»). Также следует разработать функционал для фиксирования результатов проведенных мероприятий организаторами – что эта часть должна быть или универсальной для всех встреч и/или соревнований, или же наоборот реализация должна быть уникальной в зависимости от типа мероприятия. К тому же у пользователей может быть добавлена возможность оценивать посещенное мероприятие, это позволит не только создавать рейтинги, но также улучшить рекомендательную систему – в таком случае можно было бы в качестве веса использовать не индикатор «посетил/не посетил», а оценку, которая гораздо яснее отражает впечатления студента.

# **СПИСОК ЛИТЕРАТУРЫ**

**Приложение 1**

**Функции формирования рекомендации**

function knn(k, user, users, distance=cosine) {

  const result = {};

  for (u in users) {

    result[u] = distance(user, users[u]);

  }

  let sortedNeighbours = Object.entries(result).sort((a, b) => b[1] - a[1]);

  return Object.fromEntries(sortedNeighbours.slice(0, k));

}

function recommendate(usersActive, sims) {

    const events = usersActive[Object.keys(usersActive)[0]];

    const recomendations = {};

    for (e in events) {

        let top = 0;

        let bottom = 0;

        for (u in usersActive) {

            top += sims[u] \* usersActive[u][e];

            bottom += usersActive[u][e];

        }

        recomendations[e] = bottom < 0.0001 ? 0 : top/bottom;

    }

    return recomendations;

}

function cosine(a, b) {

    let top = 0;

    let bottomA = 0;

    let bottomB = 0;

    for (e in a) {

        top += a[e]\*b[e];

        bottomA += a[e]\*\*2;

        bottomB += b[e]\*\*2;

    }

    let bottom = Math.sqrt(bottomA)\*Math.sqrt(bottomB);

    return bottom < 0.0001 ? 0 : top/bottom;

}