ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

Факултет Компютърни Системи и Технологии
Катедра Програмиране и Компютърни Технологии

Маг. инж. Александра Йоанис Бриасули

СЪВРЕМЕННИ ТЕХНОЛОГИИ – ПРОЕКТИРАНЕ И РАЗРАБОТВАНЕ НА МОДЕЛ ЗА ОБЛАЧНИ ИЗЧИСЛЕНИЯ ЗА БАЗИ ОТ ДАННИ В АДМИНИСТРАЦИЯТА

А В Т О Р Е Ф Е Р А Т

на дисертация за придобиване на образователна и научна степен

"ДОКТОР"

Област: 5. Технически науки

Професионално направление: 5.3. Комуникационна и компютърна техника

Научна специалност: Автоматизирани системи за обработка на информация и управление

Научн ръководители: доц. д-р Даниела Минковска

доц. д-р Людмила Стоянова

СОФИЯ, 2023

Дисертационният труд е обсъден и насочен за защита от Катедрения съвет на катедра „Програмиране и компютърни технологии “ към Факултет Компютърни системи и технологии на ТУ-София на редовно заседание, проведено на 11.12.2023 г.

Публичната защита на дисертационния труд ще се състои на 11.03.2024 г. от 13:00 часа в Конферентната зала на БИЦ на Технически университет – София на открито заседание на научното жури, определено със заповед № ОЖ-5.3 - 71/12.12.2023 г. на Ректора на ТУ-София в състав:

1. Проф. д-р Румен Трифонов – председател
2. Доц. д-р Веска Ганчева – научен секретар
3. Проф. д.т.н. Стойчо Стойчев
4. Проф. д-р Александър Бекярски
5. Доц. д-р Йорданка Анастасова

Рецензенти:
1. Проф. д.т.н. Стойчо Стойчев
2. Доц. д-р Йорданка Анастасова

Материалите по защитата са на разположение на интересуващите се в канцеларията на Факултет Компютърни системи и технологии на ТУ-София, блок № 1, кабинет 1443А.

Дисертантът е редовен докторант към катедра „Програмиране и компютърни технологии“ на факултет Компютърни системи и технологии. Изследванията по дисертационната разработка са направени от автора, като някои от тях са подкрепени от научноизследователски проекти.

Автор: маг. инж. Александра Бриасули
Заглавие: Съвременни технологии – Проектиране и разработване на модел за облачни изчисления за бази от данни в администрацията
Тираж: 25 броя
Отпечатано в ИПК на Технически университет – София

Въведение

Настоящата криза направи приемането на облачните изчисления (Cloud Computing) още по-важни аспекти за частния и публичния сектор в страната. Рентабилният и новаторски характер на облачните изчисления може да намали разходите, свързани с хардуер и софтуер, като същевременно поддържа качеството на услугата. Това е особено ценно както за частния, така и за публичния сектор в Гърция, особено за новите предприятия и бизнес. Тенденцията към децентрализация и ефективно използване на ИТ ресурсите се дължи на преминаването към облачни услуги, като се решават проблеми като повишено търсене на съхранение и изчислителна мощност, неефективно използване на ресурсите на центъра за данни и нарастващи разходи за поддръжка. Гръцката ИКТ екосистема се счита за благоприятна за приемане на облачни изчисления, но времето зависи от техническите фактори и външните условия като икономическия климат.

Основният фокус на дисертационния труд, представен тук, е проектирането и разработването на облачен модел на бази данни за управление, който ще формира основата за облачна платформа за конфедерация. Базата данни има за цел да управлява модел на изчислителен облак и включва различни потребителски роли, проследяване на ресурси, модификации, клиентски заявки и транзакции. Също така включва система за рейтинг, за да стимулира потребителите, които допринасят за конфедерацията, като същевременно обезсърчава експлоатацията. Освен това дисертационният труд ще изследва нерелационни бази данни за цялостна система за управление на данни, като се стреми да стандартизира ресурсите въз основа на специфични характеристики за подобрена гъвкавост на платформата. По-широките цели на дисертационния труд включват оценка на значението на своевременното приемане на облачните изчисления за конкурентоспособността на гръцката икономика и оценка на степента на интеграция на гръцките предприятия и бизнес, като се използват първични изследвания, базирани на въпросник на хартиен носител в подкрепа на тези анализи.

Първа глава: Преглед на технологиите

1. Навигиране в революцията на данните: релационни срещу нерелационни (NoSQL) модели на бази данни

Бързият напредък на технологиите през последните години прави невъобразимите преди възможности, сега достъпни и рентабилни за потребителите. Използването на тези технологии осигурява за бизнеса намаляване на оперативните разходи, свързани с управлението на хардуерни и софтуерни ресурси.

Базата данни е структурирана колекция от данни, съхранявани на компютър, предназначени за лесен достъп и управление. Тя е организирана по начин, който улеснява актуализациите и модификациите от администратор на база данни. Освен това базите данни се поддържат от системи за управление на бази данни (СУБД) (Database Management Systems (DBMS), които са отговорни за организацията и приоритизирането на данните. Тези СУБД системи играят решаваща роля за бързото извличане и актуализиране на данни, осигурявайки ефективно управление на данните в базата данни.

Релационната база данни, изобретена през 1970 г. от Edgar Codd, е структурирана колекция от данни, организирани във взаимосвързани таблици, използвайки езика за структурирани заявки (SQL) за взаимодействие с потребителя. Въпреки че е широко възприет в приложения от тип управление, релационният модел е изправен пред предизвикателства по отношение на скалируемостта, високите разходи за настройка и ограничената адаптивност за модерни приложения с високо търсене. В отговор NoSQL системите се появяват в началото на 21-ви век, движени от компании Web 2.0 като Google и Amazon. NoSQL или нерелационните бази данни се отклоняват от традиционните бази данни, базирани на SQL, като предлагат различни методи за управление на данни. Тези системи, категоризирани от CAP теоремата, оспорват традиционните свойства на ACID, което води до BASE подход. NoSQL системите се справят с недостатъците на релационните бази данни, като предоставят разпределени, мащабируеми и ефективни решения, подходящи за паралелна обработка. Големите уебсайтове

преминават към технологията NoSQL, сигнализирайки за признаване на развиващите се нужди от управление на данни и търсенето на по-ефективни методи отвъд ограниченията на релационния модел.

2. Светът на големите данни (Big Data)

Концепцията за "Големи данни" се появи като трансформираща сила в съвременното общество, характеризиращо се с бази данни, които надминават възможностите на конвенционалния софтуер за бази данни. „Големите данни“ надхвърлят конкретен обем данни и обхващат бази данни, вариращи от терабайти до петабайти. Разпространението на свързани с интернет устройства и технологии, стимулира генерирането на огромни обеми данни, оказвайки влияние върху науката, бизнеса и индустрията. Докато „Големите данни“ поставят уникални предизвикателства при извличането, записването и анализа на данни, изследователите развиват аналитични умения за работа с големи набори от данни. Въпреки че тези анализи се различават значително от традиционните статистически подходи поради хетерогенността на данните, те предлагат ценни прозрения чрез свързване на различни източници на данни и компенсиране на загубена или противоречива информация.

Големите данни направиха революция в области като астрономия, биология, образование и здравеопазване, което доведе до търсене на професионалисти с аналитични умения.

Сложният процес на тълкуване на резултатите включва изследване на предположения и потенциални източници на грешки, а валидирането на резултатите е от решаващо значение поради различни източници на грешки.

Машинното обучение играе ключова роля в анализа на големи данни, особено когато размерите на данните продължават да растат експоненциално, с изследователи, които разработват алгоритми, които могат да обработват огромни количества данни ефективно.

Трите ключови характеристики на Големите данни са: бързонарастващ обем данни, необходимостта от анализ в реално време и включването на различни формати. Въпреки предимствата си, Големите данни представят няколко предизвикателства, включително рискове за сигурността на данните и дискриминация. Усилията за справяне с тези предизвикателства включват премахване на недостига на умения, използване на анализ на големи данни за откриване на заплахи за сигурността, прилагане на законодателство като Общия регламент за защита на личните данните (GDPR), изграждане на прозрачност и доверие и прилагане на закони за защита на потребителите и равни регулации и права за възможностите.

3. Облачни изчисления: архитектури, модели на услуги и тяхното въздействие върху съвременните организации

Облачните изчисления (Cloud Computing) е трансформираща сила, предлагаща достъп при поискване до изчислителни ресурси, съхранение на данни и мощност за обработка чрез интернет. Характеризиран с характеристики като самообслужване при поискване и бърза еластичност, предлага модели на внедряване (публични, частни, общностни, хибридни), които отговарят на специфичните организационни нужди. Сервизните модели (SaaS, PaaS, IaaS) предлагат различни функционалности, докато архитектурата се състои от хардуер, инфраструктура, платформа и слоеве на приложения. Облачните изчисления улесняват оптималното използване на ресурсите, мащабируемостта и ефективното управление на данни, оказвайки значително влияние върху съвременните организации.

Предимства и недостатъци на облачните имплементации

<u>Предимства на архитектурата за внедряване на облак:</u>

Облачните имплементации предлагат намаляване на разходите и оптимално използване на ресурсите, елиминирайки необходимостта от закупуване и поддръжка на хардуер и софтуер, предоставяне на ресурси при поискване, и предлагане на ефективно използване на хранилището чрез виртуализация. Облачните изчисления осигуряват достъп от всяко място с интернет връзка, а тяхната гъвкавост и сигурност, включително мащабируеми ресурси и стабилни мерки, подобряват достъпността, като същевременно защитават потребителските данни и дейности.

Недостатъци на архитектурата за внедряване на облак:

Облачните изчисления представляват рискове за сигурността, включително уязвимост към злонамерен софтуер и пробиви на данни, с ограничен потребителски контрол върху мерките за сигурност. Загрижеността за поверителността зависи от доверието в доставчика на облак, адресирано чрез споразумения и системи за сигурност. Мрежовите зависимости от интернет свързаността правят облачните услуги податливи на прекъсвания. Предизвикателства при наблюдението на местоположението и достъпа до данни по време на възникване на проблеми с доставчици или законови промени, заедно с ограничения потребителски контрол върху инфраструктурата. Сложностите при блокиране на доставчика правят смяната на доставчика предизвикателство поради проблеми с оперативната съвместимост. Стандартизацията играе решаваща роля, осигурявайки оперативна съвместимост, улеснявайки преходите между доставчиците и подобрявайки договорните разпоредби в споразуменията за ниво на обслужване, за да смекчат загубата на данни и проблемите със съвместимостта.

3.2 Навигация в облака: Архитектури за IoT и обработка на големи данни

Гореспоменатите технологии и класификации, които в момента се изпълняват, се прилагат към бази данни за облачни изчисления NoSQL.

Националният институт за стандарти и технологии (NIST) определя ключови функционални характеристики на облачните изчисления, включително самообслужване при поискване, широк мрежов достъп, изобилие от ресурси, бърза еластичност и калибрирана услуга. Допълнителни функции и концепции, свързани с облачните изчисления, включват ескалация, еластичност, виртуализация, надеждност и многократно подновяване.

Тези характеристики и функции определят основната функционалност и предимствата на облачните изчисления, като предлагат лесно управление на ресурсите, мащабируемост, виртуализация и надеждност.

Специфичните услуги в облачните изчисления за администрацията включват:

1.**Интернет на нещата (Internet of Things (IoT)** - IoT и големи архитектури за обработка на данни се отнасят до мрежи от взаимосвързани електронни устройства със сензори и интернет свързаност, използвани в различни приложения като преносими устройства, интелигентни домове, селско стопанство, здравеопазване и индустрия. Предизвикателствата на IoT включват управление на данни, сигурност и съвместимост на устройства. Дисертационният труд има за цел да изследва архитектурите на облачните изчисления за ефективна обработка и съхранение на IoT данни.

2. Архитектури за обработка на облачни изчисления - Терминът облачни изчисления се отнася до способността за предоставяне на изчислителни ресурси от разстояние, чрез интернет за автоматизиране на процеси, извършване на изчисления, обработка и съхраняване на данни.

3. Големи архитектури за обработка на данни - Големите облачни архитектури за обработка на данни се състоят от източници на данни, пакетна обработка, приемане на съобщения в реално време, обработка на потоци, съхранение на аналитични данни, анализ и докладване и оркестрация. Нуждата от мащабируеми, бързи и устойчиви на грешки архитектури за обработка и съхранение на големи обеми от данни нараства, като IoT и облачните изчисления обработват обработката на данни в тристепенни IoT архитектури. Технологии за обработка на данни, включително λ (Lambda) и Kappa архитектури, са предназначени за групова обработка, обработка в реално време, прогнозен анализ и машинно обучение.

Ламбда архитектурата (λ architecture) - управлява ефективно големи данни с пакетен слой, скоростен слой и обслужващ слой. Подходящ е за системи, изискващи точност при обработката на данни. Главната функция на архитектурата λ се характеризира с отношението: Заявка = λ (Пълни данни) = λ (Данни за предаване на живо) * λ (Съхранени данни)

Капа архитектурата (Kappa architecture) - опростява обработката на големи данни, като се фокусира върху обработката в реално време и обработката на неструктурирани данни, като се състои от слой за скорост и слой за обслужване. Главната функция на архитектурата k се характеризира с връзката: Заявка = k (Пълни данни) = k (Данни за предаване на живо.

MapReduce е мощен инструмент за анализ на големи данни, улеснява обработката на разпределени данни чрез функции за картографиране и редуциране, паралелизирайки

ефективно изчисленията между компютърни клъстери. MapReduce може да се справи с широк набор от изчислителни проблеми.

Популярни облачни доставчици и услуги

Няколко популярни комерсиални доставчици на облачни изчисления и техните услуги включват Amazon EC2, част от AWS, предоставяща мащабируеми изчислителни ресурси, AMI, автоматично мащабиране и балансиране на натоварването. Microsoft Azure поддържа както PaaS, така и IaaS, хоства компоненти на приложения в моментни снимки на виртуална машина и предоставя възможности за балансиране на натоварването. Google App Engine (GAE): Услугата за хостинг на уеб приложения, работеща на ниво PaaS, се фокусира върху мащабируемостта и абстрахира основния хардуер и операционна система. Flexiant Cloud Provider: IaaS доставчик с Flexiant Cloud Orchestrator и Flexiant Concerto, предлагащ решения за управление на облак и разработка на приложения, поддържащи различни хипервайзори и облачни доставчици.

Типове NoSQL, транзакции и облачна интеграция

Тези категории предлагат разнообразни решения за различни нужди за управление на данни, от просто съхранение на ключ/стойност до сложни връзки, базирани на графики. NoSQL базите данни се избират въз основа на системните изисквания.

Бази данни в администрацията:

- Въведение в теоремата CAP теоремата и характеристиките на ACID.
- NoSQL базите данни предлагат подобрено управление на данни за съвременните нужди.
- Шардингът подобрява сигурността на данните.

NoSQL базите данни се категоризират основно в следните четири типа:

Базиран на двойка ключ-стойност: Подходящ за голям обем данни, използва двойки ключ/стойност (напр. Redis, Dynamo), базиран на колони (напр. Cassandra, HBase), ориентиран към документи (напр. MongoDB, CouchDB) и базиран на графики (напр. Neo4J, Infinite Graph). По отношение на обработката на транзакциите, релационните бази данни използват ACID транзакции за целостта на данните, докато NoSQL базите данни се придържат към принципите BASE (основна наличност, меко състояние, евентуална последователност), поради теоремата на CAP, което налага компромис между последователност, наличност и толерантност към дялове в разпределени хранилища за данни.

Разделяне на архитектури Master-Slave и Peer-to-Peer:

Разделянето на архитектури Master-slave и peer-to-peer се отнася до начина, по който компонентите на системите работят заедно. Master-slave включва централизиран контрол, рискувайки потенциално слабо място, докато peer-to-peer предоставя равни възли, увеличавайки достъпността, но изисквайки сложни механизми за съгласуваност на данните.

NoSQL бази данни в облачните изчисления:

Революцията в облачното съхранение довежда до появата на NoSQL бази данни, които се използват за широкомащабно съхранение на данни като: Marklogic, MongoDB, Apache CouchDB, Google App Engine, IBM Cloudant, Oracle Cloud, Bigtable, Azure CosmosDB, HBase и др.

3.4 Предимства и недостатъци на административните технологии

Административните технологии в съвременното управление на данни носят уникални предимства и недостатъци. Базите данни, базирани на двойки ключ-стойност, предлагат мащабируемост в разпределени системи, но им липсват последователност на транзакциите и възможности за филтриране. Базите данни, ориентирани към документи, се отличават с поддръжка на заявки и мащабируемост, но им липсват ACID транзакции и се сблъскват с предизвикателства при ad-hoc заявки. Базираните на колони бази данни са ефективни от пространството, но са по-малко подходящи за ad hoc докладване. Графично базираните бази данни се справят добре с връзките M:N, но се затрудняват с агрегираните заявки. Ламбда архитектурата балансира скоростта и надеждността на данните, докато Капа архитектурата опростява повторната обработка, но компрометира точността на данните. Изборът на правилната архитектура зависи от конкретните случаи на употреба и изисквания, като се вземат предвид техните силни и слаби страни.

4. Изчерпателен преглед на ключовите технологии

Този изчерпателен преглед подчертава ключови технологии в модерното уеб развитие, управление на данни и комуникация. Python, широко използван език за програмиране на високо ниво, е известен със своята четливост и поддържа бързо развитие за различни мащаби. Flask, уеб рамка на Python, опростява разработката на уеб приложения с олекотен подход. REST API, базиран на Representational State Transfer, позволява оперативно съвместими уеб приложения. MongoDB, NoSQL база данни, съхраняват данни в гъвкави документи във формат JSON, предлагайки мащабируемост. JSON, текстово базиран формат на данни, улеснява асинхронната комуникация, а HTTP, основата на уеб комуникацията с данни, дефинира методи като GET, POST, PUT и DELETE в модел клиент-сървър.

Втора глава: Разработка и анализ на модел за управление на интелигентна облачна конфедерация

1. Въведение

В областта на вземането на решения, базирани на данни, стабилните и адаптивни бази данни са от решаващо значение за ефективната администрация. Този дисертационен труд изследва персонализиран модел на база данни, обръщайки внимание на неговата необходимост, основни компоненти и приложения. Традиционните методи за управление на данни стават непрактични, тъй като организациите се разширяват, подчертавайки необходимостта от ефективно съхранение, извличане и анализ на данни в администрацията. Мащабируемите бази данни осигуряват последователна производителност с нарастващи данни, като същевременно налагат правила за цялост и сигурност за чувствителна информация. Добре структурираните бази данни позволяват автоматизация и ефективност чрез процеси като отчитане и анализи. Основните компоненти включват щателно изработен дизайн на схема, интуитивен потребителски интерфейс, стабилни протоколи за сигурност и възможности за интеграция с други административни инструменти и системи.

2. Облачен компютърен модел за организационни административни бази данни

Проектирането и разработването на модел за облачни изчисления за бази данни, насочени към административни цели, изисква щателно планиране и разбиране на работните процеси на административните данни. Ето целта на този обхват.

2.1. Разбиране на организационните цели

Разбирането на изискванията е първата критична стъпка в дизайна на базата данни. От съществено значение е да се гарантира, че системата поддържа нуждите от данни, потребителите и очаквания растеж.

• Дефиниране на обхвата: Разпознаване на мисията, визията и основните административни задачи на организацията.

• Консултация със заинтересованите страни: необходимо е да се обсъдят нуждите и ограниченията с ИТ експерти, администратори, мениджъри и дори представителна извадка от редовните служители.

Цел: Преди да започнем значим проект, особено такъв, който включва потенциална миграция или промяна в начина на съхранение и достъп до данните, разбирането на основните цели на организацията е от първостепенно значение. Това фундаментално знание гарантира, че всички направени промени или въведени системи са в съответствие с мисията, стратегическите цели и ценностите на организацията.

Компоненти:

1. **Определение на обхвата**

- Обосновка: Помага за определяне на границите и мащаба на проекта за база данни. Какви данни ще трябва да обработва? Кои отдели или процеси ще поддържа?
- Предимство: Избягва забавянето на мисията, гарантира, че разпределението на ресурсите съответства на изискванията на проекта и предоставя ясна представа за въздействието на проекта върху по-широката организация.

2. **Консултация със заинтересованите страни::**

- Обосновка: Различните заинтересовани страни в една организация имат различни гледни точки и приоритети. Например финансовият директор може да е загрижен за разходите и възвръщаемостта на инвестициите, докато ръководителят на отдел може да даде приоритет на функционалността и лекотата на използване.
- Полза: Осигурява цялостно разбиране на нуждите и потенциалните предизвикателства. Води до по-холистична, добре оформена система, която отговаря на по-широк набор от организационни нужди.

Методологии за ефективно прилагане:

1. SWOT анализ: SWOT анализът (силни страни, слаби страни, възможности, заплахи) може да предложи прозрения за текущото състояние на организацията и бъдещите стремежи.
2. Фокус групи: Ангажирането с малки, разнообразни групи от организацията може да осигури нюансирано разбиране на по-широките организационни цели.
3. Картографиране на целите: Визуалните инструменти и рамки, като стратегически карти или логически модели, могат да помогнат за ясното артикулиране и свързване на всеобхватните цели с конкретните цели на проекта.

2.2. Избор на облачна среда

Изборът на публичен, частен, хибриден или мулти-облак трябва да съответства на инфраструктурната стратегия на организацията. Mell & Grance дефинира моделите за облачни изчисления, наблягайки на съображения като цена, контрол и сигурност.

- Публичен облак: избор на доставчици като AWS, Azure или Google Cloud за рентабилни, мащабируеми решения.
- Частен облак: използва се, когато сигурността на данните, суверенитетът и пълният контрол са от най-голям приоритет.
- Хибриден облак: смес, която се възползва максимално от възможностите на публичния и частния облак.

Цел: Изборът на правилната облачна среда е ключов за привеждане в съответствие с организационните нужди, бюджетни ограничения и дългосрочни изисквания за скалируемост.

Компоненти:

1. **Публичен облак:**

- Обосновка: Доставчиците на публичен облак предлагат споделени ресурси, което ги прави рентабилни и мащабируеми.
- Предимство: Те са подходящи за организации, които искат да избегнат високи първоначални разходи и изискват гъвкавост при мащабиране на ресурсите въз основа на търсенето.

2. **Частен облак**:

- Обосновка: Предлага специални ресурси и подобрена сигурност, което ги прави идеални за организации, които работят с чувствителни данни или имат строги изисквания за съответствие с нормативните изисквания.
- Предимство: Осигурява по-добър контрол върху заобикалящата среда, като гарантира, че инфраструктурата и ресурсите са в съответствие със специфичните изисквания.

3. **Хибриден облак:**

- Обосновка: Комбинира най-доброто както от публичните, така и от частните облаци, позволявайки на организациите да се възползват от рентабилността на публичния облак за определени задачи, като същевременно запазват чувствителните операции в частния облак.
- Предимство: Предлага балансиран подход към гъвкавост, цена и сигурност.

Методологии за ефективно прилагане:

1. Оценка на нуждите: Извършват се задълбочени вътрешни оценки, за да се определи коя облачна среда отговаря най-добре на нуждите на организацията.
2. Анализ на разходите и ползите: Претеглят се плюсовете и минусите, както финансово, така и оперативно, на всяка среда, за да се осигури дългосрочна жизнеспособност.
3. Пилотно тестване: Преди пълноценната миграция, се тества избраната облачна среда с подмножество от операции, за да се идентифицират потенциални предизвикателства.

2.3. Избор на решение за база данни

Cattell подчертава, че съвременните приложения често изискват комбинация от релационни и NoSQL бази данни, в зависимост от структурата на данните и случаите на използване.

• RDBMS: Подходящ за структурирани организационни данни като записи на служители, финансови данни и управление на проекти.

• NoSQL: Приема се за гъвкави структури от данни, като обратна връзка, регистрационни файлове и проекти за сътрудничество.

Цел: Изборът на подходящо решение за база данни гарантира, че организацията може ефективно да съхранява, извлича и управлява своите данни, като взема предвид както настоящите, така и потенциалните бъдещи изисквания.

Компоненти:

1. **Система за управление на релационни бази данни RDBMS (Relational Database Management System):**

- **Обосновка:** RDBMS са структурирани и използват схема за определяне на връзките между таблиците с данни. Те са особено ефективни, когато целостта на данните и релационните структури на данните са от първостепенно значение.
- **Предимство**: Те предоставят свойства на ACID (атомност, консистенция, изолация, издръжливост), като гарантират надеждност на данните и улесняват сложни заявки.

2. **NoSQL:**

- **Обосновка**: NoSQL базите данни са по-гъвкави по отношение на структурата на данните, позволявайки съхранението на различни набори от данни като документи, ключ-стойности или базирани на графики данни.
- **Предимство:** Тези бази данни обикновено са по-мащабируеми и могат да обработват големи обеми от бързо променящи се, нерелационни данни. Те са особено подходящи за приложения в реално време и решения за големи данни.

Методологии за ефективно прилагане:

1.Одит на данни: Преглед на естеството на данните, с които работи организацията. Структурирано ли е? Колко бързо расте? Какви запитвания се очакват?

2.Сравнителен анализ на производителността: Тестване на потенциални решения за бази данни при очаквани натоварвания, за да се прецени тяхната производителност.

3.Подготвени за бъдещето: трябва да се обмисли бъдещата посока на организацията. Ако организацията очаква бързо нарастване на неструктурираните данни, NoSQL може да бъде по-добро дългосрочно решение, дори ако текущите нужди се посрещат от RDBMS.

2.4. Дизайн на база данни и мащабируемост

Hernandez подчертава, че една добре структурирана схема гарантира целостта и ефективността на данните, а обмислянето на стратегии за архивиране е също толкова важно за устойчивостта на данните.

- **Планиране на схема:** разработване на структура, която отразява организационната йерархия, роли и задачи.
- **Готовност за растеж:** проектиране на архитектурата, като се вземе предвид бъдещия растеж, обема на данните и променящите се административни нужди.

Цел: Ефективният дизайн на базата данни гарантира оптимална производителност, докато скалируемостта отговаря на нуждите от растеж и разширяване на съхранението на данни на организацията. Една добре проектирана база данни, която също може да се мащабира ефективно, гарантира, че изискванията за данни на организацията са изпълнени както сега, така и в бъдеще.

Компоненти:

1. **Схема за планиране:**

- Обосновка: Добре дефинираната схема е планът на базата данни, определящ как са организирани данните и как се обработват връзките между данните.
- Полза: Гарантира, че данните могат да бъдат достъпни и управлявани ефективно, като по този начин се подпомагат операциите на организацията и процесите на вземане на решения.

2. **Подготвеност за растеж:**

- Обосновка: Тъй като организациите се развиват, техните нужди за съхранение на данни се променят. База данни, която е подготвена за растеж, може да поеме нарастващите данни без значителни ревизии.
- Предимство: Намалява бъдещи разходи и смущения, свързани с разширяване на бази данни или миграции.

Методологии за ефективно прилагане:

1.Итеративен дизайн: започва се с основна схема и се прецизира с течение на времето въз основа на действителното използване и изисквания, вместо да се правят опити за проектиране на „перфектната“ схема от самото начало.

2.Планиране на капацитета: Редовно се прави оценка на обема и вида на данните, които се съхраняват, и бъдещ растеж се преценява въз основа на историческите тенденции и организационните прогнози.

3.Решения, базирани на облак: Облачните платформи често предоставят скалируемост като вградена функция, позволяваща на базите данни да се разширяват въз основа на търсенето без ръчна намеса.

2.5. Мерки за сигурност

Schneier подчертава важността на стабилните механизми за сигурност, включително криптиране, контрол на достъпа и непрекъснат мониторинг.

• **Криптиране от край до край:** трябва да съществува увереност, че данните остават защитени, както когато се съхраняват, така и когато се пренасят.

• **Удостоверяване и упълномощаване на потребителя:** внедряване на многофакторно удостоверяване и ролеви контрол на достъпа (RBAC) за чувствителни административни данни.

• **Редовни одити:** Периодично осъществяване на преглед на регистрационните файлове за достъп и дейностите, за да е сигурно, че няма аномалии или пробиви.

Цел: Гарантирането, че данните, съхранявани в облака, остават поверителни, запазват своята цялост и са достъпни, когато е необходимо, е от решаващо значение за организационното доверие и съответствие с нормативните изисквания. Мерките за сигурност са защитните действия и политики, които предпазват от заплахи, пробиви и потенциални загуби.

Компоненти:

1. **Криптиране:**
   • **Обосновка:** Шифроването на данни, както в покой, така и в транзит, гарантира, че дори ако възникне неоторизиран достъп, данните остават нечетими.
   • **Предимство:** Защитава чувствителните данни от пробиви и отговаря на много изисквания за съответствие.
2. **Управление на самоличността и достъпа (IAM):**
   • **Обосновка:** Определя кой има достъп до данните и какво може да прави с тях въз основа на роли и разрешения.
   • **Предимство:** Намалява риска от вътрешни пробиви и гарантира, че служителите имат достъп само до данни, необходими за техните роли.
3. **Редовни одити:**
   • **Обосновка:** Периодичните прегледи на протоколите за сигурност, регистрационните файлове за достъп и системните конфигурации могат да идентифицират уязвимости или неразрешени дейности.
   • **Полза:** Проактивно идентифицира и адресира потенциални слабости в системата.

Методология за ефективно прилагане:

1.Многофакторно удостоверяване (MFA): Включването на множество методи за проверка, като нещо, което човек знае (парола) и на нещо, което има (мобилно устройство), значително намалява шансовете за пробив.

2.Обучение на служителите: Редовните сесии за обучение гарантират, че всички членове на персонала са запознати с най-добрите практики и могат да идентифицират потенциални заплахи, като опити за фишинг.

3.Планове за архивиране и възстановяване след бедствие: трябва да съществува увереност, че дори в случай на загуба на данни, системите могат да бъдат възстановени бързо, за да се поддържа функционалността на организацията.

2.6. Интеграция и свързаност

Интеграцията осигурява цялостна функционалност. Rahm & Do обясняват значението на интегрирането на бази данни с други системи за по-ефективен поток от данни.

- Разработка на API: Създавайте интерфейси за програмиране на приложения за безпроблемна връзка с други организационни инструменти и платформи.
- Инструменти за мигриране на данни: Включете инструменти, които подпомагат прехода от по-стари системи или улесняват редовно архивиране и прехвърляне на данни.

Цел: Безпроблемната интеграция и ефективната свързаност гарантират, че облачните бази данни могат да комуникират с други организационни системи и платформи. Това не само поддържа последователност и достъпност на данните, но и подобрява цялостната оперативна ефективност.

Компоненти:

1. **API (интерфейси за програмиране на приложения):**
   - **Обосновка:** API позволяват различни софтуерни приложения да комуникират, улеснявайки обмена на данни в реално време и автоматизация.
   - **Предимство:** Позволява безпроблемен поток от данни между платформи, като гарантира последователност на системата и намалява грешките при ръчно въвеждане на данни.
2. **Мидълуер:**
   - **Обосновка:** Мидълуерът служи като мост между базата данни и приложенията, помагайки за стандартизиране на комуникационните протоколи и осигурявайки плавен трансфер на данни.
   - **Предимство:** Опростява интеграцията, особено в сложни ИТ среди с наследени системи.
3. **VPN (Виртуална частна мрежа):**
   - **Обосновка:** VPN мрежите създават защитен тунел за трансфер на данни между вътрешната мрежа на организацията и облачната база данни.
   - **Предимство:** Подобрява сигурността по време на обмен на данни и осигурява последователна свързаност, особено за отдалечен достъп.

Методология за ефективно прилагане:

1. Картографиране на данни: Ясно дефинирайте как полетата с данни в облачната база данни съответстват на полетата в други системи, като гарантирате последователен пренос на данни.

2. Непрекъснато наблюдение: Редовно наблюдавайте интеграциите за производителност и нива на грешки, като бързо адресирате всички аномалии.

3. Унифицирано управление на данни: Създайте централизирана рамка за управление, която определя стандарти за формати на данни, правила за валидиране и честота на синхронизиране между системите.

2.7 Архивиране, съхранение и възстановяване

Gray et al. посочва, че толерантността към грешки чрез репликация и географско съхранение е от съществено значение за критичните бази данни.

- **Планирани архиви:** Редовно трябва да се осъществява архивиране данните, за да се осигури минимални загуби в случай на повреди.
- **План за възстановяване след бедствие:** разполагайте със стабилен механизъм за възстановяване, пълен със срокове и отговорности.

Цел: Тази стъпка гарантира, че данните остават безопасни, достъпни и непокътнати дори при неочаквани събития като системни повреди, пробиви на данни или природни бедствия. Действа като предпазна мярка, осигуряваща непрекъснатост на операциите и минимизирайки потенциалната загуба на данни.

Компоненти:

1. **Архивиране:**
   - **Обосновка:** Редовното създаване на копия на базата данни гарантира, че данните могат да бъдат възстановени, ако основният източник на данни е компрометиран или изгубен.

• **Предимство:** Осигурява предпазна мрежа срещу неочаквана загуба на данни и предлага точка за възстановяване.

2. **Съхранение:**
   • **Обосновка:** Съхраняването на множество копия на данни на различни места или системи, означава, че ако една система се повреди, данните остават достъпни от друг източник.
   • **Предимство:** Осигурява наличност на данни и минимизира времето за престой, особено в случай на системни повреди.
3. **Възстановяване:**
   • **Обосновка:** Дефинира протоколите и процедурите за възстановяване на данни от резервни копия или излишни системи след повреда.
   • **Предимство:** Позволява на организацията бързо да се върне към работно състояние след прекъсване, минимизирайки оперативните и финансовите въздействия.

Методология за ефективно прилагане:

1. Инкрементални архиви: Вместо постоянно да се архивира цялата база данни, заснемането на промените, направени след последното архивиране, намалява нуждите от съхранение и ускорява процеса на архивиране.

2. Гео-съхранение: съхранение на излишните данни на географско отделни места, предпазвайки от регионални бедствия или прекъсвания.

3. Автоматизирани протоколи за възстановяване: употреба на автоматизация, за да се ускори процеса на възстановяване, като се гарантира, че системите могат да бъдат възстановени с минимална ръчна намеса и за възможно най-кратко време.

2.8. Потребителски интерфейс и достъпност

Дизайнът на интерфейса трябва да дава приоритет на използваемостта. Nielsen споменава, че добрият потребителски интерфейс следва принципите на яснота, обратна връзка и последователност [60].

- **Адаптивни табла за управление:** разработване на табла за управление, които предоставят важна информация с един поглед и могат да бъдат персонализирани за потребителски роли.
- **Мобилен достъп:** проектиране на интерфейси, достъпни на мобилни устройства, което позволява на администраторите да управляват задачи в движение.

Цел: Добре проектираният потребителски интерфейс (user interface - UI), комбиниран с ефективни функции за достъпност, гарантира, че потребителите могат да взаимодействат с базата данни ефективно, което прави въвеждането, извличането и анализа на данни, интуитивни и ясни. Тази стъпка е насочена и към по-широката цел за приобщаване, като гарантира, че потребителите с нарушения могат да имат достъп и да използват ефективно базата данни.

Компоненти:

1. **Потребителски интерфейс - User Interface (UI):**
- **Обосновка:** Удобният за потребителя интерфейс, подобрява потребителското изживяване, като намалява кривата на обучение и насърчава ефективното взаимодействие с базата данни.
- **Предимство:** Опростява задачите, намалява грешките и повишава удовлетвореността на потребителите.
2. **Потребителско изживяване - User Experience (UX):**
- **Обосновка:** Отвъд визуалния интерфейс, UX обхваща цялостното взаимодействие, включително скорост, надеждност и интуитивни работни процеси.
- **Предимство:** Гарантира, че потребителите могат да изпълняват задачите си ефективно.
3. **Функции за достъпност:**
- **Обосновка:** Гарантирането, че платформата може да се използва от хора с увреждания, като тези със зрителни или слухови нарушения, е ключов аспект на съвременния софтуерен дизайн.
- **Предимство:** Разширява потребителската база, отговаря на регулаторните стандарти и насърчава приобщаването.

Методология за ефективно прилагане:

1.Потребителско тестване: ангажираност на действителни потребители във фазата на тестване, за да се събере обратна връзка и да се разберат техните нужди и предизвикателства. Това може да насочи към усъвършенстването на интерфейса и функционалността.

2.Адаптивен дизайн: увереност, че интерфейсът е използваем и изглежда добре на редица устройства, от настолни компютри до мобилни устройства.

3.Включване на стандарти за достъпност: придържане към признати стандарти, като Указанията за достъпност на уеб съдържанието (WCAG), за да се гарантира, че платформата е достъпна за потребители с увреждания.

2.9Поддръжка и непрекъснато подобряване

Непрекъснатият мониторинг гарантира здравето на системата. Loukides посочва, че мониторингът дава представа за ефективността, потенциалните проблеми и потребителските дейности, като помага за проактивното управление.

- **Инструменти за наблюдение:** Използване на инструменти, които наблюдават здравето, производителността и моделите на използване на базата данни.
- **Цикъл за обратна връзка**: Редовно събиране на обратна връзка от потребителите, като се осъществяват итерации на модела при промяна на организационните нужди.

Цел: Поддръжката гарантира, че облачната база данни остава работеща, сигурна и ефективна. Непрекъснатото усъвършенстване, от друга страна, е свързано с усъвършенстване и подобряване на системата въз основа на променящи се изисквания, обратна връзка и технологичен напредък. Заедно тези елементи гарантират дълголетието на системата и постоянната уместност.

Компоненти:

1. **Редовни актуализации:**

- **Обосновка:** Софтуерните актуализации адресират известни уязвимости, коригират грешки и понякога въвеждат нови функции.
- **Предимство:** Подобрява сигурността, функционалността и цялостната стабилност на системата.

2. **Мониторинг:**

- **Обосновка:** Постоянно се наблюдава производителността на системата, потребителските взаимодействия и всички потенциални проблеми.
- **Предимство:** Предоставя информация в реално време, като позволява проактивно решаване на проблеми и осигурява оптимална производителност.

3. Верига за обратна връзка:

- **Обосновка:** Създат се канали, чрез които потребителите могат да предоставят обратна връзка, да съобщават за проблеми или да изискват функции.
- **Предимство:** Поддържа системата в съответствие с нуждите на потребителите, като повишава удовлетвореността на потребителите и полезността на системата.

Методология за ефективно прилагане:

1. Планирана поддръжка: Извършване на дейности по поддръжката по време на ненатоварени часове или периоди на по-слаба потребителска активност, за да се сведе до минимум прекъсванията.

2. Автоматизирани инструменти за наблюдение: Използване на инструменти, които могат автоматично да откриват проблеми с производителността, аномалии или заплахи за сигурността, като гарантират по-бързо време за реакция.

3. Итеративно развитие: Приема се итеративен подход, пускайки по-малки, чести актуализации, вместо големи, редки. Това позволява по-бързи корекции въз основа на обратна връзка и променящи се нужди.

2.10. Обучение и адаптиране

Провеждане на редовни обучения за администратори за нови функции или промени.

- **Ръководства за потребителя:** Предлагат изчерпателни ръководства и уроци за ефективно използване на системата от бази данни.
- **Обучителни сесии:** Организиране на периодично обучение, като се гарантира, че служителите се възползват максимално от предоставените инструменти.

Цел: Обучението и адаптирането са от съществено значение, за да се гарантира, че потребителите, особено тези, които са нови в облачната база данни, могат да използват платформата ефективно, ефикасно и безопасно. Тази стъпка е ключова за подпомагане на потребителите да преминат от по-стари системи или методологии към новата облачна база данни, осигурявайки непрекъснатост на работата и смекчавайки потенциални грешки.

Компоненти:

1. **Сесии за обучение на потребители:**

- **Обосновка:** Официалните сесии предоставят на потребителите знанията и уменията, от които се нуждаят, за да навигират и използват базата данни умело.
- **Полза:** Намалява кривата на обучение, потенциалните грешки и повишава производителността.

2. **Документация:**

• Обосновка: Подробни ръководства, често задавани въпроси и наръчници служат като справочници за потребителите, предлагайки инструкции стъпка по стъпка или решения на общи предизвикателства.

• Полза: Осигурява непрекъснат ресурс, към който потребителите да се позовават, когато са изправени пред несигурност.

3. **Бордови програми:**

- **Обосновка:** Създадени за нови потребители или екипи, тези програми ги запознават със системата, нейните функции и най-добрите практики.
- **Предимство:** Улеснява прехода и интегрирането на нови членове, като гарантира последователност в начина, по който се използва базата данни.

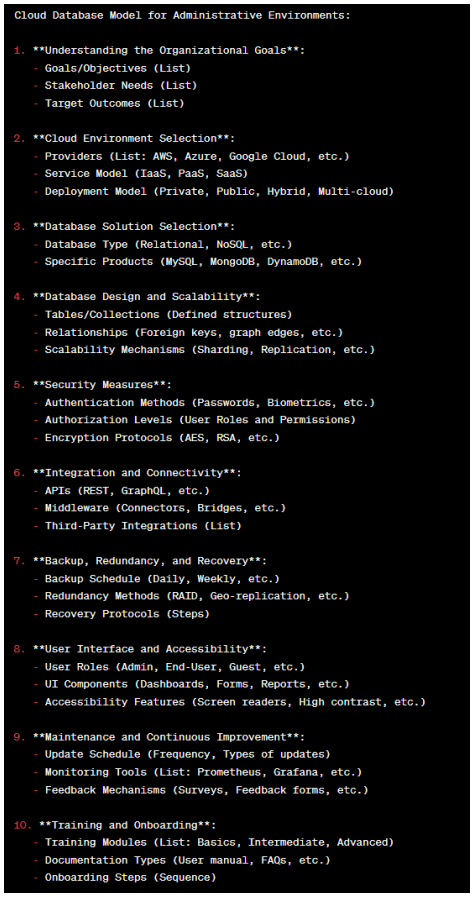
Методологии за ефективно прилагане:

1. Модулно обучение: Проектиране на обучителни сесии в модули, от основни до напреднали, позволявайки на потребителите да се включат на тяхното ниво на комфорт и умения.

2. Интерактивни уроци: Вместо просто традиционно обучение, са предложени интерактивни онлайн уроци, където потребителите могат да учат чрез правене, подобрявайки задържането и разбирането.

3. **Събиране на обратна връзка**: След обучението се събира обратна връзка, за да се разберат областите на подобрение, предизвикателствата, пред които са изправени потребителите, и да се прецизират бъдещите сесии за обучение.

Проектирането на схема за модел на облачна база данни, особено такъв, пригоден за административни среди, би включвало дефиниране на структурите от данни, връзките и ограниченията, върху които системата ще работи. Ето една основна схема от високо ниво, базирана на компонентите, които обсъждахме досега.



```
Cloud Database Model for Administrative Environments:

1. **Understanding the Organizational Goals**:
   - Goals/Objectives (List)
   - Stakeholder Needs (List)
   - Target Outcomes (List)

2. **Cloud Environment Selection**:
   - Providers (List: AWS, Azure, Google Cloud, etc.)
   - Service Model (IaaS, PaaS, SaaS)
   - Deployment Model (Private, Public, Hybrid, Multi-cloud)

3. **Database Solution Selection**:
   - Database Type (Relational, NoSQL, etc.)
   - Specific Products (MySQL, MongoDB, DynamoDB, etc.)

4. **Database Design and Scalability**:
   - Tables/Collections (Defined structures)
   - Relationships (Foreign keys, graph edges, etc.)
   - Scalability Mechanisms (Sharding, Replication, etc.)

5. **Security Measures**:
   - Authentication Methods (Passwords, Biometrics, etc.)
   - Authorization Levels (User Roles and Permissions)
   - Encryption Protocols (AES, RSA, etc.)

6. **Integration and Connectivity**:
   - APIs (REST, GraphQL, etc.)
   - Middleware (Connectors, Bridges, etc.)
   - Third-Party Integrations (List)

7. **Backup, Redundancy, and Recovery**:
   - Backup Schedule (Daily, Weekly, etc.)
   - Redundancy Methods (RAID, Geo-replication, etc.)
   - Recovery Protocols (Steps)

8. **User Interface and Accessibility**:
   - User Roles (Admin, End-User, Guest, etc.)
   - UI Components (Dashboards, Forms, Reports, etc.)
   - Accessibility Features (Screen readers, High contrast, etc.)

9. **Maintenance and Continuous Improvement**:
   - Update Schedule (Frequency, Types of updates)
   - Monitoring Tools (List: Prometheus, Grafana, etc.)
   - Feedback Mechanisms (Surveys, Feedback forms, etc.)

10. **Training and Onboarding**:
   - Training Modules (List: Basics, Intermediate, Advanced)
   - Documentation Types (User manual, FAQs, etc.)
   - Onboarding Steps (Sequence)
```

Фигура 1. облачен модел на база данни за административна среда

Тази схема служи като план на високо ниво за проектиране на цялостен модел на облачна база данни, пригоден за административни настройки. Всеки компонент се нуждае от по-подробни подкомпоненти, правила и структури при изграждането на действителна система.

Фигура 2 показва проста схема на база данни за базирана в облак административна система на общинска служба. Тази система ще управлява жителите, имотите и общинските услуги.

```sql
-- Residents Table
CREATE TABLE Residents (
    ResidentID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    DOB DATE,
    Email VARCHAR(100),
    PhoneNumber VARCHAR(15),
    Address VARCHAR(250)
);

-- Properties Table
CREATE TABLE Properties (
    PropertyID INT PRIMARY KEY AUTO_INCREMENT,
    Address VARCHAR(250) NOT NULL,
    OwnerID INT,
    PropertyValue DECIMAL(10,2),
    PropertyType VARCHAR(50),  -- Examples: "Residential", "Commercial", "Va
    FOREIGN KEY (OwnerID) REFERENCES Residents(ResidentID)
);

-- Municipal Services Table
CREATE TABLE Services (
    ServiceID INT PRIMARY KEY AUTO_INCREMENT,
    ServiceName VARCHAR(100),
    Cost DECIMAL(8,2),
    Description TEXT
);

-- Service Requests Table (to manage resident requests for municipal service
CREATE TABLE ServiceRequests (
    RequestID INT PRIMARY KEY AUTO_INCREMENT,
    ResidentID INT,
    ServiceID INT,
    RequestDate DATE,
    Status VARCHAR(50),  -- Examples: "Pending", "Completed", "Rejected"
    FOREIGN KEY (ResidentID) REFERENCES Residents(ResidentID),
    FOREIGN KEY (ServiceID) REFERENCES Services(ServiceID)
);
```

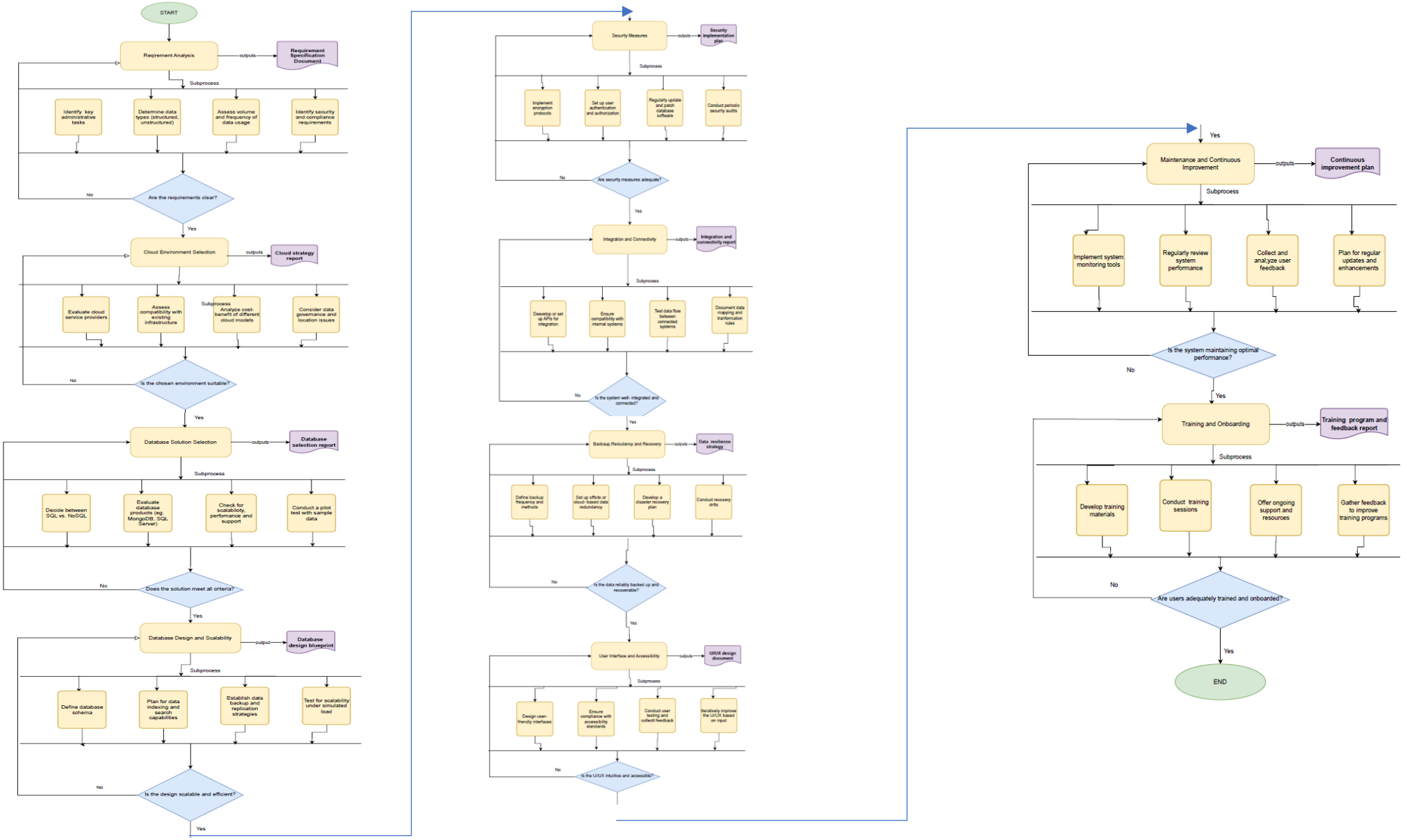
Фигура 2. схема на база данни за базирана в облак административна система

Разяснения:

- Таблицата “Жители” - съдържа информация за жителите, живеещи в общината.
- Таблицата “Имоти” - поддържа списък с имоти в рамките на общината. Има външен ключ, свързващ се с таблицата “Жители”, за да обозначи собствеността.
- Таблицата “Услуги” - каталогизира различните налични общински услуги, като събиране на отпадъци, водоснабдяване и др.
- Таблицата „Заявки за услуги“ - управлява заявките на жителите за специфични общински услуги. Жителите могат да правят заявки и статусът на тези заявки може да бъде проследен.

Това е основна схема и може да бъде разширена с повече подробности, исторически записи, различни видове услуги и други в зависимост от нуждите на общинската служба. Могат да се направят корекции, за да отговарят на специфични изисквания или да се настанят допълнителни модули.

Flow Chart of Model



3. Заключение

Един ефективен модел за изчисления в облак за организационни административни бази данни, държи ключа към рационализираните операции и информираното вземане на решения. Гъвкавостта и мащабируемостта на този модел са от съществено значение, тъй като организациите се развиват, поддържайки оперативно съвършенство. Въпреки това, за да се постигне, разбирането на целите на организацията е от първостепенно значение, което гарантира, че моделът на облачните изчисления е в съответствие със стратегическата насока и ценности на организацията, служейки като основа за дългосрочен успех.

Процесът на вземане на решения по отношение на облачни среди и решения за бази данни е от решаващо значение, тъй като пряко влияе върху оперативната ефективност, намаляването на разходите и сигурността на данните. Докато традиционните релационни бази данни може да са подходящи за много предприятия, скалируемостта и гъвкавостта на NoSQL базите данни предлагат предимства, особено за организации с разнообразни данни или очакващи бърз растеж. Дизайнът на базата данни, мащабируемостта, мерките за сигурност, интеграцията, архивирането, потребителският интерфейс, поддръжката и обучението на потребителите играят основна роля за гарантиране, че облачната база данни остава надеждна, ефективна и адаптивна. Холистичният подход към управлението на бази данни, обхващащ както технологичните аспекти, така и човешкия елемент, е от съществено значение за успешната цифрова трансформация в административните настройки, насърчавайки доверието, съответствието и оптималната ангажираност на потребителите.

Трета глава: Внедряване - Експериментални данни и резултати от приложението на предложената система

1.Въведение

Третата глава обсъжда разработването на платформа за създаване на облачна конфедерация в отговор на нарастващото значение на облачните изчисления. Тази платформа има за цел да преодолее пропастта между местната инфраструктура и глобалния пазар, позволявайки на доставчиците на облачни услуги да се свързват и на потребителите да намират оптимални доставчици въз основа на техните нужди. Изследва нерелационни бази данни, стандартизация на ресурсите, широкомащабна производителност и прилагане на справедлива система за предотвратяване на злоупотреби. Концепцията за облачни федерации се подчертава като взаимосвързан „облак от облаци“, преодолявайки ограниченията на отделните доставчици като ограничени ресурси и географски ограничения. Този подход позволява на доставчиците да споделят инфраструктура и предлага на потребителите по-широка гама от опции за услуги, насърчавайки демократизацията на пазара и позволявайки на по-малките доставчици да се конкурират с по-големите. Целта на платформата включва управление на облачни изчислителни инфраструктури, обработка на потребителски заявки за виртуални машини, оптимизиране на разпределението на сървърите, гарантиране на справедливост на потребителите и основна сигурност. Внедряването включва уеб рамката Flask, REST протокол, MongoDB за управление на данни и вариант на алгоритъма glicko за справедливост на потребителите. Владеенето на Python е необходимо за разработването на платформата, като целта е да поеме голяма потребителска база.

2. Анализ

2.1 Облачни федерации: Основни изисквания за платформата

Има изисквания, които поставят основата за разработването на платформа за облачни конфедерации, целящи да осигурят безпроблемно и честно изживяване за потребителите и доставчиците на облачни услуги и те могат да бъдат обобщени, както следва:

а)Модел на облачни изчисления: Облачните изчисления представляват революционен оперативен модел за разпределени центрове за данни, подобен на моделите като при електроснабдяването и мобилните телекомуникации.

б) **Ограничения на ресурсите:** Дори големите облачни доставчици имат ограничени физически ресурси и ограничен географски обхват, което налага формирането на облачни федерации за обединяване на ресурси.
в) **Междинна платформа:** За да се даде възможност на доставчиците да си сътрудничат и да споделят ресурси в рамките на федерацията, е необходима междинна платформа.
За да бъде полезна и функционална, такава платформа трябва да има някои основни характеристики и да отговаря на някои основни спецификации. Някои от тях може да са:
а) **Регистрация на потребител:** Платформата трябва да позволява на клиентите-доставчици да се регистрират доброволно в системата и да записват своите данни в база данни.
б) Регистрация на ресурси: Участващите членове трябва да могат да регистрират ресурсите, които искат да предоставят на федерацията, като съхраняват тази информация в база данни.
в) **Търсене на ресурси:** Регистрираните членове могат да поискат конкретни ресурси, налични във федерацията, и платформата трябва да избере най-подходящия член, който да изпълни заявката.
г) **Честно отношение към потребителите:** Като доброволна система, платформата трябва да използва алгоритъм, за да гарантира справедливо отношение към потребителите, особено в неконкурентна среда.
д) **Основна сигурност**: Удостоверяването на потребителя е основно изискване за предотвратяване на имитиране и подобряване на сигурността в рамките на платформата.

2.2 Платформата Intercloud

Платформата Intercloud има централен фокус в текста, предназначен да реализира концепцията за облачни конфедерации и да даде възможност на потребителите-доставчици да споделят ресурси. Платформата включва няколко ключови функции, включително регистрация на потребители, удостоверяване и възможност за ефективно управление на различни потребителски дейности.
Ключовите точки за платформата Intercloud включват:
а) **Име на платформата:** Платформата, която се разработва, се нарича Intercloud Exchange Platform, подчертавайки нейната роля в създаването на облачни конфедерации.
б) **Класификация на потребителите:** Потребителите се категоризират като клиенти и доставчици и потребителят може да има и двата статуса. Регистрацията е задължително условие за всички потребители, което гарантира сигурността и целостта на системата.
в) **Поверителност, цялост и наличност (Confidentiality, Integrity, and Availability - CIA):** Регистрацията на потребителя е от решаващо значение за поддържане на поверителността, целостта на данните и наличността на системата. Платформата прилага мерки за сигурност, като комбинации парола-потребителски идентификатор, за да поддържа тези принципи.
г) **Удостоверяване и база данни:** Внедрена е система за удостоверяване за проверка на самоличността на потребителя. Платформата е проектирана да обработва многобройни заявки ефективно, дори в случай на системни повреди, с подходяща база данни за сигурност и задържане на данните.
д) **Функции на доставчика:** Доставчиците могат да предоставят изчислителни ресурси на конфедерацията, да променят съществуващи ресурси и да премахват ресурси от федерацията. Съобщенията във формат JSON обикновено се използват за комуникация.
е) **Клиентски функции**: Клиентите могат да поискат конкретни ресурси, които отговарят на техните нужди. Платформата обработва тези заявки, търси в базата данни за подходящи доставчици и връща информация на клиентите за по-нататъшни взаимодействия.
ж) **Справедливо отношение към потребителите:** Като се има предвид доброволният характер на облачната конфедерация, се въвежда система за оценка на потребителите, за да се гарантира справедливо отношение. Тази система възнаграждава участващите потребители и ги насърчава да поддържат активно участие, като същевременно обезсърчава експлоатацията.

2.3 Рационализиране на облачните конфедерации: стандартизация, потребителски оценки и интеграция на MongoDB

1. Необходимост от стандартизиране на заявките: Растежът на доставчиците на облачни услуги доведе до вариации в терминологията и описанията на ресурсите. За да се създаде

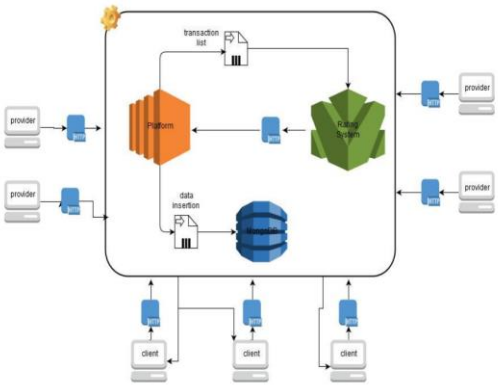
функционална платформа за облачни конфедерации, стандартизацията на комуникацията е от решаващо значение. Това включва използването на HTTP протокола със специфични заглавки за идентифициране на формата на данните. Показването и съхранението на ресурсите са стандартизирани въз основа на модели, открити във водещи облачни доставчици. Стандартизираните съобщения се използват за търсене на ресурси.

2.**Система за оценка на потребителите на платформата:** Членството в облачната конфедерация е доброволно, така че е разработена система за оценка на потребителите, за да възнагради активните участници, особено доставчиците. Потребителите се оценяват въз основа на стандартизирана оценка и отклонение в оценката. Платформата избира подходящи доставчици въз основа на потребителски оценки. Алгоритъмът Glicko, адаптиран за неконкурентна среда, актуализира оценките на потребителите независимо.

3.Избор на MongoDB за база данни: MongoDB, широко използвана NoSQL система за управление на данни и документи, е избрана заради нейните характеристики. Използва BSON за съхранение на данни, поддържа рамка за агрегиране и предлага индексиране за подобрена производителност при търсене. MongoDB е подходящ за приложения с неструктурирани и полиморфни данни, което го прави стабилен избор за платформата за облачни конфедерации.

3. Разработка на платформата

Този скрипт указва процедурата, следвана за регистриране на ново лице - потребител в платформата. По време на процеса на разработване на платформата са разработени следните компоненти: процедури за регистрация на потребител, функция за импортиране на ресурс на доставчик, функция за коригиране на елемент от ресурс на доставчик, функция за изтриване на ресурс на доставчик, функция за обслужване на заявки от клиенти за търсене на ресурси и накрая, оценка на комуникационната функция на платформата и системата. Следващата фигура показва основната архитектура на облачната конфедерация, която ще бъде разработена за нуждите на този дисертационен труд.



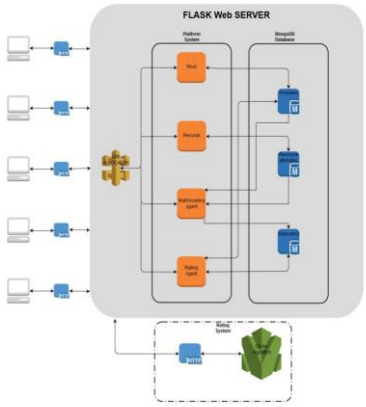
Фигура 3: основната архитектура на облачната конфедерация, която ще бъде разработена

4. Системна архитектура и дизайн

4.1. Описание на цялостната работа на системата и схематична структура.

Философията за проектиране на системата, която е приета, се основава на принципите на архитектурата на софтуерната система и е извършена с оглед на нуждите на потребителя, както и с ефективността, гъвкавостта и сигурността на системата. Този подход на проектиране дава приоритет на нуждите на потребителите, ефективността на системата, гъвкавостта и сигурността. Главата очертава процеса на дефиниране на системната архитектура, подсистеми, модели, интерфейси и данни, за да отговори на системните изисквания. Също така подчертава, че системата не съществува изолирана, а се влияе от околната среда, което изисква отчитане както на дизайнерските, така и на потребителските нужди, както и разпоредби за решаване на настоящи и бъдещи проблеми. Цялостният процес, от първоначалния дизайн до окончателното изпълнение, включва цялостно изследване на

съответните фактори и изчисляване на необходимите спецификации въз основа на технически и теоретичен анализ. Фазата на проектиране е критичен компонент от разработката на системата, започвайки анализ, след завършването на фазата. Важно е, че резултатите от фазата на анализ, заедно със спецификациите, извлечени от нея, служат като входни данни за етапа на проектиране, което в крайна сметка води до ефективното внедряване на системата.



Фигура 4: Системна архитектура

4.1.1. Структурата на колекцията от доставчици

Очертана е структурата на колекцията от доставчици.

- Системата се състои от три независими колекции от обекти: Колекция от доставчици, Колекция от атрибути на ресурси, Колекция от списък с транзакции.
- Колекцията на доставчиците дефинира данни, свързани с всеки доставчик в рамките на конфедерацията.
- Документите в тази колекция включват полета като _id (уникално потребителско име), Active (указващо активно участие), рейтинг (изразяващ ефективността на участието на доставчика), RD (отклонение на рейтинга въз основа на системата glicko), url (URL за комуникация на доставчика) и pwd (хеширана парола).
- Тази колекция се използва от подкласове Root, Matchmaking-agent и Rating-agent.
- Предоставен е примерен документ от колекцията на доставчиците, представящ активен доставчик с име „user1“ със стандартизирани стойности на рейтинг и отклонение на рейтинга.

```
/* 1 */
{
    "_id" : "user1",
    "active" : true,
    "rating" : 1500,
    "RD" : 350,
    "pwd" : "$6$rounds=656000$MKevrlUEYv5PjjM9$dph6XL3Z4DlKYth5tbXie0/VyzBXjrKd3e5KDvu3ciy8cQYt.K.Fy1
}
```

Фигура 5: код за user1, който е активен и току-що регистриран в системата

4.1.2. Структурата на колекцията атрибути на ресурси

- Структурата на колекцията Resource_attributes се върти около категоризирането на инфраструктурите като услуги, комбинирайки ги с машинни изображения в един обект, известен като виртуална машина (VM).
- Характеристиките, определящи уникалността на VM, търсенето и предлагането на пазара, се записват в тази колекция.
- Категоризацията на пазара на инфраструктурата като услуга не е чисто техническа; услугите се категоризират въз основа на тяхното предназначение и адаптивност към нуждите на потребителите.
- Полетата на базата данни са приведени в съответствие с характеристиките, представляващи интерес.

- Връзките между характеристиките са описани, като множество местоположения за една инфраструктура, различни услуги за изображение на машина, множество услуги за типове оптимизация, количествени характеристики за всеки вид и клас виртуална машина и уникалното определяне на цената и наличността на услугата.
- Структурата на колекцията от атрибути на ресурси е показана както следв като пример.

Фигура 6: структура на колекцията атрибути на ресурси

- Документите за колекцията са уникално индексирани от полето за потребителско име.
- Колекцията съдържа таблица с местоположения с полета за регион и инстанции.
- Полето за инстанции включва OS и Тип на инстанцията, което допълнително съдържа полета за оптимизация, продукти, Размер на инстанцията и други подробности.
- Полето с подробности включва вградени атрибути като vCPU, памет, инстанция за съхранение, цена и наличност.
- "Вградените" документи се използват за организиране на данни, всеки с локален уникален индекс към друго поле.
- Тази структура е избрана поради стабилните връзки на характеристиките, избягване на прекомерна документация и предимствата на търсачката.
- Размерът на документа намалява при повече услуги поради "вградени" таблици.
- Ограничението от 16 MB на документ не се очаква да бъде надвишено, като се запазва производителността на базата данни.
- Колекцията се използва от подкласовете Records и Matchmaking-agent.

4.1.3. Колекцията списък с транзакции

- Документите за събиране на списъка с транзакции, описват елементите на транзакцията.
- Всеки документ има уникален идентификатор на транзакция.
- Полетата на документа включват _id, име на доставчик, клиент и резултат.
- Името на доставчика е променливо, заменено от потребителското име на доставчика, обслужващо заявка.
- Клиентът е потребителското име на доставчика, искащ услуги.
- Резултатът приема стойности 0 или 1 при успех или неуспех на транзакцията.
- Долупосоченият примерен документ показва успешна транзакция, при която user5 предоставя ресурси на user1.

Фигура 7: примерен документ на колекцията, в която променливата е зададена на името на доставчика user5, който е удовлетворил искането на user1 за ресурси

4.1.4. Използване на API на Python

За комуникация с MongoDB, както от страна на платформата, така и от страната на flask_web_server, е използван официалният API на Python (pymongo и flask_pymongo), предоставен от MongoDB, и кодът е написан на езика за програмиране Python.

Процесът използва тръбопроводен механизъм, при който всеки етап на обработка захранва следващия.

4.2. Системата Platform

- Intercloud Exchange Platform, разработена да обслужва нуждите на потребителите.
- Flask_restful разширение на flask_web_server, използвано за разработка.
- Позволява създаване на API като обект на Python, следващ REST архитектура.
- API картографира HTTP методите към подсистемите на класовете на платформата.
- Всяка подсистема е подклас на класа Resource.
- Четири независими подсистеми: Root, Records, Matchmaking Agent и Rating Agent.

4.2.1. Подсистемата Root

Използването на подсистемата Root се активира, когато Flask_Web_Server получи HTTP заявка с целеви URL адрес /root/.

4.2.2. Подсистемата Records

Използването на подсистемата Records се активира, когато Flask_Web_Server получи HTTP заявка с целеви URL адрес /update/. Подсистемата комуникира само с колекцията Resource_attributes на базата данни.

4.2.3. Подсистемата Matchmaking_agent

Използването на подсистемата Matchmaking_agent се активира, когато Flask_Web_Server получи HTTP заявка с целеви URL адрес /search/.

4.2.4. Подсистемата Rating_agent.

Използването на подсистемата Rating_agent се активира, когато Flask_Web_Server получи HTTP заявка с целеви URL адрес /rating_agent/.

4.3. Проектиране на база данни за администрацията

За да се създаде схема на база данни за Intercloud Exchange Platform, ще трябва да се дефинират таблици и връзки между тях, като се вземе предвид различните потребителски роли и функционалности, описани по-горе.

1. Таблица на потребителите

user_id (Primary Key, Auto Increment)
username
password (hashed)
company_name
role (ENUM: 'client', 'provider', 'both')
rating (FLOAT)

2. Таблица с ресурси

resource_id (Primary Key, Auto Increment)
user_id (Foreign Key, references Users.user_id)
resource_type (ENUM: 'compute', 'storage', 'network', etc.)
available_capacity
total_capacity
status (ENUM: 'available', 'in-use', 'deleted')

3. Таблица за модификации на ресурси

modification_id (Primary Key, Auto Increment)
resource_id (Foreign Key, references Resources.resource_id)
user_id (Foreign Key, references Users.user_id)

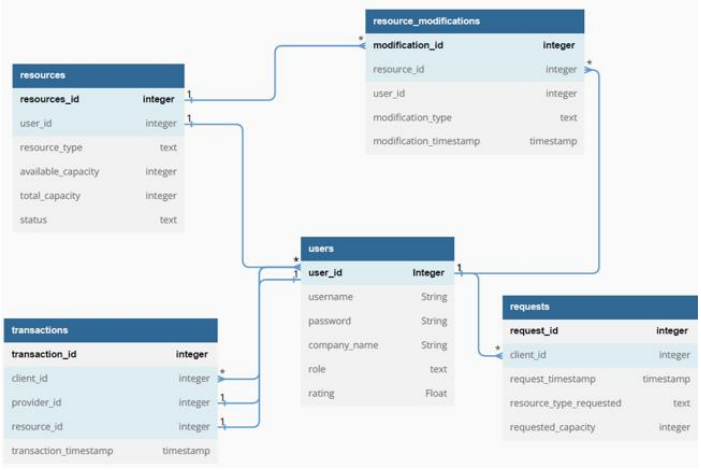
modification_type (ENUM: 'update', 'delete')
modification_timestamp (DATETIME)

4. Таблица на заявките

request_id (Primary Key, Auto Increment)
client_id (Foreign Key, references Users.user_id)
request_timestamp (DATETIME)
resource_type_requested
requested_capacity

5. Таблица на транзакциите

transaction_id (Primary Key, Auto Increment)
client_id (Foreign Key, references Users.user_id)
provider_id (Foreign Key, references Users.user_id)
resource_id (Foreign Key, references Resources.resource_id)
transaction_timestamp (DATETIME)



Фигура 8: Схема на база данни

Схемата на базата данни и кодът на Python, предоставени по-горе, са предназначени за администриране на модел за облачни изчисления, по-специално Intercloud Exchange Platform. Тази платформа свързва различни облачни доставчици и клиенти в облачна конфедерация, позволявайки на потребителите да споделят ресурси и улеснява работата на облачната екосистема. Схемата на базата данни включва различните роли на потребителите (клиент, доставчик или и двете) и проследява ресурсите, модификациите на ресурсите, клиентските заявки и транзакциите между клиенти и доставчици. Също така прилага система за рейтинг, за да възнагради потребителите, които допринасят за конфедерацията, и да накаже тези, които експлоатират системата.

Използвайки SQLAlchemy, кодът на Python дефинира моделите на базата данни и предоставя примери за това как да се взаимодейства с базата данни, за да се извършват различни операции като добавяне на потребители, ресурси и запитване за информация. Тази настройка на база данни може да се използва за управление и администриране на Intercloud Exchange Platform и да служи като основа за изграждане на по-сложни и богати на функции системи за управление на облак.

4.4. Системата за оценяване

Системата за оценяване, използваща алгоритъма Glicko, оценява избора на потребителите в рамките на Intercloud Exchange Platform. Ключовите компоненти включват използване на алгоритъма Glicko, анализиране на потребители въз основа на проценти и отклонения в процентите, дефиниране на периоди на рейтинг и провеждане на две основни стъпки за актуализации на рейтинга. Отличава се от Glicko, като включва два възможни изхода (победа и загуба), специално кодирани като 0 и 1, като резултатите се прилагат само за доставчици. Системата възнаграждава участващите доставчици, предлагащи качествена услуга, обезсърчава злоупотребата с платформата, като приема поражението като единствен резултат

за клиентите и се активира периодично в рамките на определени периоди на оценка, като гарантира честни и ефективни актуализации.

4.5. Flask Web Server

Уеб сървърът на Flask действа като централен център за управление на всички заявки към платформата, независимо дали идват от потребители или системата за оценка. По-специално, RESTful интерфейс се използва за методи, отговарящи на тези заявки, позволявайки безпроблемна комуникация с всяка друга платформа чрез прости HTTP заявки. Внедрени като RESTful API с помощта на разширението Flask RESTful, основните подсистеми и механизми на платформата за HTTP комуникация са структурирани, като обхващат аспекти като удостоверяване на потребителя, записи на ресурси, съвпадение, рейтинг, извличане на данни, сигурност, обработка на данни и формат на отговора. Това внедряване, изградено върху Flask RESTful, служи като основа за Intercloud Exchange Platform, улеснявайки различни функционалности като потребителски взаимодействия, управление на ресурсите, намиране на съвпадения и комуникация със системата за рейтинг.

4.6 Система за оценяване Glicko_algorithm

Системата за оценяване Glicko_algorithm инициира периода на оценка и изчислява нови оценки и отклонения в рейтинга за потребители, ангажирани в конфедерацията чрез транзакции. Взема предвид резултатите от транзакцията, вида на транзакцията и предишните потребителски оценки. Основната функционалност на системата е илюстрирана в Приложение 6. Методът "main_prog" комуникира със сървъра, като използва библиотеката "requests" за HTTP заявки, задава заглавки на заявки и полезно натоварване и инициализира сесии с определен параметър "c", използвайки "request. put()". Той извлича речник, съдържащ всички периодични транзакции чрез "request.get()", преобразува този речник във формат, който установява двойки клиент-доставчик, и впоследствие изчислява нови оценки както за доставчиците, така и за потребителите въз основа на техните заявки за ресурси. След това функцията "count_providers()" се извиква, за да изчисли нови ставки, придържайки се към дефинираните връзки в главата за проектиране. И накрая, поредица от заявки "request.patch()" се изпращат за актуализиране на тарифите и отклоненията в рейтинга на всички потребители. Системата Rating_agent работи периодично на интервали от един час, съответстващи на определена продължителност на рейтингов период.

4.7. Измервания

Оценката на ефективността на платформата се фокусира върху времето, необходимо за изпълнение на заявки за търсене, особено в сценарии с обширни данни и различни количества доставчици. Проучването разглежда заявките за услуги, като се приема най-лошият сценарий с доставчик, който разпределя целия си инвентар. Измерванията използват услугите, предлагани от Amazon Web Services (AWS) за физически лица, създавайки доставчик с 3318 виртуални машини. Оценката обхваща сценарии с 5, 10, 15 и 20 регистрирани доставчици, като симулира най-лошия сценарий за търсене и улеснява цялостна оценка на възможностите на платформата при различни условия.

5.Заключения

Писането на дисертационния труд и внедряването на платформата включват ангажиране с различни технологии, което довежда до полезни заключения. Разработването на платформата изисква задълбочен анализ, като се набляга на функционалността, ефективността и бързата обработка на различни потребителски заявки, като същевременно се гарантира сигурността на данните. Стандартизирането на визуализацията на ресурсите за универсална комуникация и осигуряването на съвместимост със стандартите на доставчиците са от решаващо значение за разработването на платформа за облачни услуги. Проучването на технологиите разкрива прозрения, с NoSQL бази данни, предпочитани за обработка на големи обеми данни, хоризонтално мащабиране и групова обработка. Уеб рамката на Flask, известна с адаптивността и удобната за потребителя документация, улеснява бързото разработване на приложения. Въпреки че е сравнително нов, Flask се препоръчва за ясно разбиране на системните компоненти и сведени до минимум неизползвани функции.

Четвърта глава: Първично проучване в гръцките компании

1. Основна методология на изследването

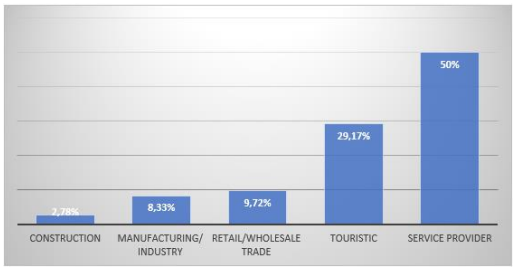
Тази глава очертава основната изследователска методология, използвана за проучването:

Дизайн на въпросника: Въпросникът е проектиран с акцент върху краткостта, като същевременно се гарантира, че покрива целите на изследването. Този подход има за цел да минимизира времето, необходимо на предприемачите за попълване на въпросника.

Период на изследването: Проучването е проведено между септември 2022 г. и януари 2023 г.

Извадка от проучването: Извадката на изследването се състои от фирми, опериращи в районите на Атина и Кардица. Този избор на райони е повлиян от мястото на постоянно пребиваване (Атина) и произхода на изследователя (Кардица). Извадката е взета както от Атина, където са включени приятели и познати, така и от Кардица, където е използвана случайна извадка.

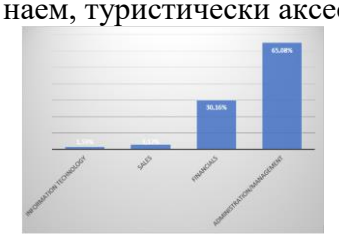
Фигура 9: Местоположение



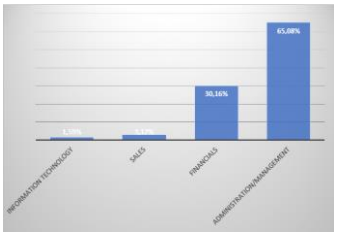
Фигура 1: Вашият бизнес принадлежи към индустрията

Характеристики на извадката от проучването: В проучването участват 144 компании. От анкетираните 65,97% се намират в Атина, докато 34,03% са в Кардица.

Фирмите са категоризирани, както следва:
Обслужване: 50%, туризъм: 29.17%, търговия на дребно/едро: 9,72%, производство: 8,33%, строителство: 2,78%.
Предоставена е допълнителна категоризация за доставчици на услуги (напр. здравеопазване, счетоводство, уроци, право) и туристически бизнес (напр. хотели, коли под наем, стаи под наем, туристически аксесоари).



Фигура 11: Години на трудов опи



Фигура 12: Брой служители, различни от собствениците

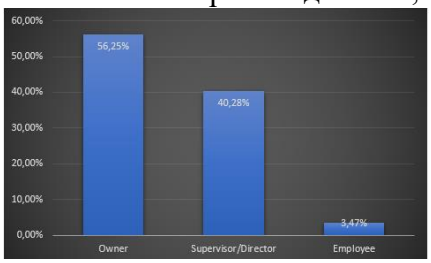
Години на опит: Анкетираните варират в годините на трудова дейност. Приблизително 36,11% са работили от 6-10 години, 34,72% от 11-20 години и 28,47% от повече от 20 години. Само 0,69% са работили по-малко от 5 години.

Брой служители: Що се отнася до броя на служителите (с изключение на собствениците), 47,22% от предприятията са имали по-малко от 10 служители, 38,89% са имали 11-50 служители, 13,19% са имали 51-250 служители и 0,69% са имали повече от 250 служители.

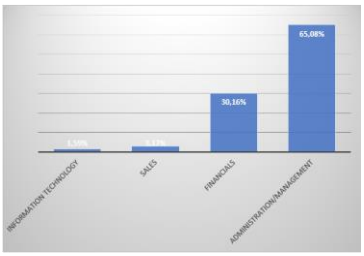
Обработка на данни: Данните, събрани от проучването, са обработени с помощта на SPSS версия 25, статистически софтуерен инструмент.

2. Резултати

Резултатите от проучването дават представа за демографските характеристики и свързаните с ИТ познания на респондентите, както следва:



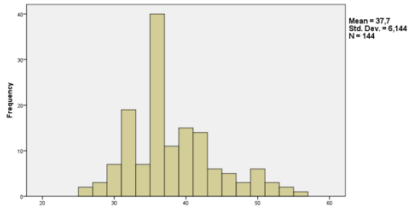
Фигура 13: Позиция във фирмата



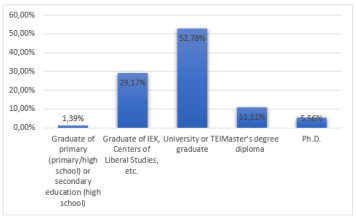
Фигура 14: Ако сте ръководител/директор или служител, моля, посочете в кой отдел работите

56,25% от анкетираните са собственици на фирми, 40,28% са супервайзъри/директори, а 3,47% са служители.

От тези респонденти, които са на ръководни длъжности 65,08% работят в администрация/управление, 30,16% във финанси, 3,17% в продажби и 1,59% в ИТ сферата.



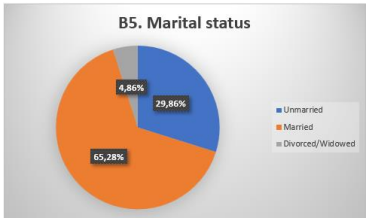
Фигура 15: Възраст



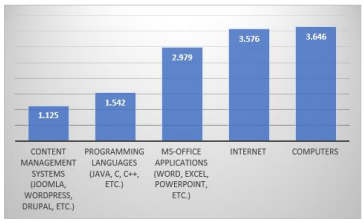
Фигура 16: Образователно ниво

Средната възраст на 144-те анкетирани е 37,7 години.

Образователни нива: 52,78% са завършили висши учебни заведения или технологични учебни заведения, 29,17% са завършили институти за професионално образование, либерални науки и др., 11,11% притежават магистърска степен, 5,56% имат докторска степен и 1,39% са завършили основно/средно образование.



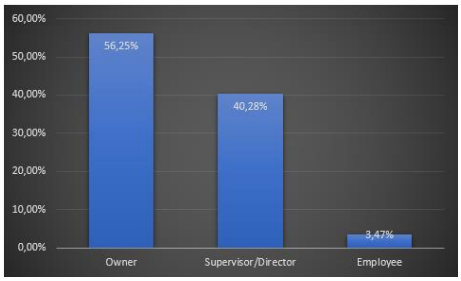
Фигура 17: Семейно положение



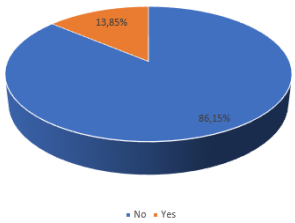
Фигура 18: Моля, запишете степента на познаване и боравене със следното: ***Променливи: 1 никаква, 2 малка, 3 умерена, 4 добра, 5 много добра***

Повечето анкетирани са омъжени/женени (65,28%), следвани от несемейни (29,96%), като 4,86% са разведени или овдовяли.

Анкетираните показват различна степен на умения в областта на компютъра и ИТ.



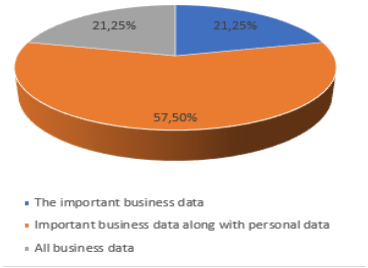
Фигура 19: Кой е отговорен във вашия бизнес за ИТ/компютъризация



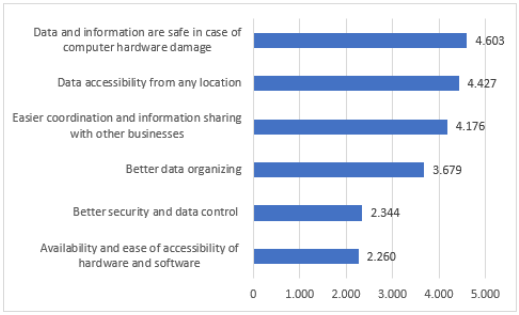
Фигура 20: Вашата компания има ли вътрешна мрежа (интранет) със сървър

Мнозинството (88,29%) разчитат на външни партньори за ИТ проблеми, докато 6,94% имат постоянна външна поддръжка, а 4,86% имат вътрешни ИТ специалисти.

86,11% съобщават, че тяхната компания няма вътрешна мрежа (интранет) със сървър, докато 13,89% имат такава.



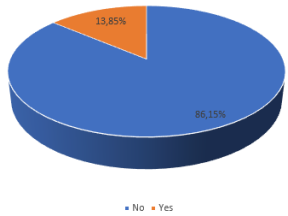
Фигура 21: Какво е вашето ниво на познания по отношение на облачните изчисления?



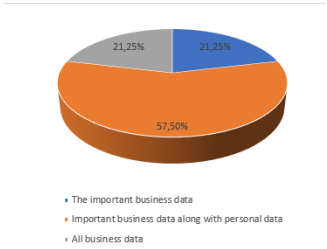
Фигура 22: Независимо дали облачните изчисления се използват или не във вашия бизнес, моля, оценете важността на следните предимства на облачните изчисления:

47,92% знаят за Cloud Computing, 43,06% твърдят, че имат достатъчно познания, а 9,03% нямат познания.

Анкетираните оценяват достъпността и сигурността на облачните данни, докато организацията на данните, координацията и достъпът до хардуер/софтуер се считат за по-малко важни.



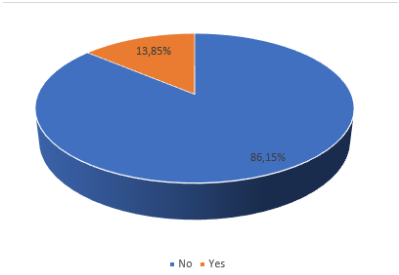
Фигура 23: Използвате ли облачни изчисления във вашия бизнес?



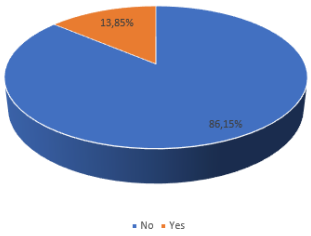
Фигура 24: Какъв вид информация съхранявате в облака?

Облачните изчисления се използват от 55,56% от компаниите на анкетираните.

57,50% съхраняват важни бизнес и лични данни в облака, 21,25% съхраняват само важни бизнес данни и 21,25% съхраняват всички бизнес данни.



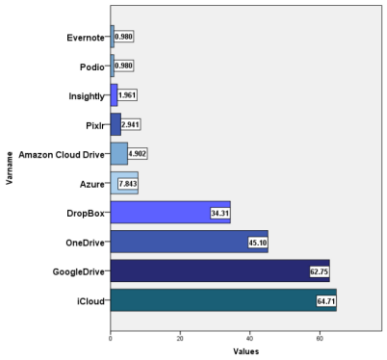
Фигура 25: Вашият бизнес има ли уебсайт?



Фигура 26: Използвате ли облачен хостинг за уебсайта си?

54,86% от фирмите имат уебсайт в интернет, докато 45,14% нямат.

Повечето респонденти (86,15%) не използват технология за облачни изчисления за своя уебсайт, като 13,85% я използват.



Фигура 27: Кои от следните услуги за облачни изчисления сте използвали лично или в бизнеса?

Популярните облачни услуги включват iCloud (64,71%), Google Drive (62,75%), OneDrive (45,10%) и Dropbox (34,31%). Други услуги имат по-ниски нива на използване.

3. Дискриминантен анализ на първични изследвания

3.1. Дискриминантен анализ

В тази точка дискриминантният анализ се прилага за създаване на функции за класиране за фирми, използващи облачни изчисления. Критериите за класиране се състоят от две категории: характеристики на гръцката компания (години на дейност и брой служители) и характеристики на бизнес мениджърите (възраст, образователно ниво и др.). Дискриминантният анализ има за цел да класифицира наблюденията в известни популации с известни разпределения, търсейки правила за точна класификация. Това е сложен, но ценен метод, широко използван в научната общност.

3.2. Дискриминантен анализ, базиран на характеристиките на бизнеса

Следващият раздел разглежда дискриминантния анализ, фокусиран върху бизнес характеристиките, по-специално годините на трудов опит и броя на служителите. Анализът разкрива значителни разлики в двете променливи между компаниите, използващи облачни изчисления, и тези, които не го правят. Въпреки липсата на значителни разлики в груповите средни стойности, се използват допълнителни тестове и функции за оценка, за да се създаде функция за класиране (Z), която класифицира компаниите въз основа на годините на трудов опит и броя на служителите. Анализът постига 61,1% правилен процент на дискриминация,

което предполага, че бизнес характеристиките влияят върху възприемането на облачните изчисления сред гръцките компании.

3.3Дискриминантен анализ, основан на характеристиките на предприемача/супервайзора.

Преминавайки към раздел 3.3, дискриминантният анализ се прилага въз основа на характеристиките на предприемача/супервайзора, като се изследват фактори като тяхната позиция в бизнеса, възраст и образователно ниво. Анализът разкрива значителни разлики в средната възраст и образователно ниво на предприемачите/ръководителите между групите. Въпреки това не се открива съществена разлика в позицията им в бизнеса. Използването на дисперсионни матрици, точкови функции и функция за класиране (Z) допринася за класифицирането на предприятията в групи въз основа на тези характеристики. Цялостният дискриминантен анализ постига правилен процент на разделяне от 63,2%, затвърждавайки идеята, че специфичните характеристики на предприемача/супервайзора играят роля при определянето на приемането на облачните изчисления в гръцките компании.

4. Заключение

Това проучване има за цел да оцени въздействието на навременното приемане на облачните изчисления върху конкурентоспособността на гръцката икономика и нейната интеграция от гръцкия бизнес. Предоставя общ преглед на облачните изчисления, подчертавайки основните характеристики и обясняйки услугите и моделите за внедряване. Изследването анализира въздействието му, наблягайки на потенциалното намаляване на разходите и подобрените услуги. Важен извод е, че облачните изчисления могат да спестят 4,8 милиарда евро през следващото десетилетие, предлагайки значителни ползи и генерирайки над 16 милиарда евро за гръцката икономика от 2010-2020 г. чрез повишена мащабируемост и навлизане на пазара. Проучването подчертава значението на добре обучен експертен екип за плавен преход.

5. ОБЩИ ЗАКЛЮЧЕНИЯ

В този дисертационен труд, основният фокус е върху оценката на значението на навременното приемане на облачните изчисления за конкурентоспособността на гръцката икономика и оценката на нивото на прилагането им сред гръцките предприятия. Проучването обхваща разработването на платформа, проучване на различни технологии и първични изследвания, което води до няколко ключови заключения. Разработването на успешна платформа включва внимателно разглеждане на функционалността, ефективността, сигурността на данните и потребителския опит, наблягайки на стандартизираната визуализация на ресурсите и справедливото отношение към потребителите. Тези фактори са ключови за постигане на целите на платформата.

Проучването на различни технологии, като бази данни NoSQL и уеб рамката Flask, предостави ценна представа за техните силни и слаби страни, подпомагайки вземането на решения при избора на технологии. Първичните резултати от изследването показват потенциалните ползи от облачните изчисления за гръцката икономика, включително спестяване на разходи, увеличаване на доходите и създаване на работни места. Въпреки това, значителна част от гръцките фирми са в ранните етапи на интеграция на облачни изчисления, като само 55,56% в момента използват облачни услуги. Проучването подчертава липсата на знания за облачните изчисления сред предприемачи, мениджъри и служители. Препоръките за бъдещи изследвания, включват подобрения на платформата като разработване на графичен потребителски интерфейс, прецизна интеграция на географски данни, разглеждане на сложността на заявката, разширяване на услугите и внедряване на разпределени системи. Предлагат се по-нататъшни изследвания, включващи подобни проучвания за оценка на приемането на облачните изчисления в Гърция и репликация на проучването в различни региони за по-цялостно разбиране и усъвършенстване на дискриминантните функции.

АПРОБАЦИЯ

Моделът и получените резултати от изследването са използвани в администрацията на община Игуменица. Изглежда, че базата данни работи правилно и моделът е в състояние да поддържа потребители с много облаци. Въпреки това приложението е пуснато само пробно, тъй като за пълно инсталиране и контрол е необходим специален лиценз от компетентното министерство.

ОСНОВНИ ПРИНОСИ

Основните приноси на предложеното изследване могат да бъдат разделени на научни и с приложен характер.

Приноси с научен характер:

Предложена е стабилна рамка за оценка и рационализиране на прехода на традиционни релационни бази данни към облачни бази данни NoSQL, специално насочени към административните функции.

Създаден е цялостен модел за общинска администрация, използващ документно-ориентираната структура на базата данни на MongoDB. Този модел се задълбочава във функционалността, методологичния подход, технологичните императиви и основните математически конструкции.

Приноси с приложен характер:

Моделът, ориентиран към MongoDB, е проектиран с визията за подобряване на операциите на общинските служби, осигуряване на ефективно улавяне на данни, съхранение и извличане, особено за записи на жители, подробности за имоти и общински услуги.

Въведено е базирано на облак административно табло, което се комбинира с нашия модел, което позволява наблюдение в реално време, генериране на отчети и процеси на вземане на решения от общинските служители.

Разработена е, и въведена автоматизирана система за предупреждение и оповестяване, осигурява навременна обработка на заявки за услуги, жалби и одобрения, с което се преодолява дистанцията между жителите и общината.

Концептуализиран е обучителен модул, подкрепен от разработения модел, за да се улесни безпроблемното навлизане на общинския персонал, като гарантира, че те са оборудвани с ноу-хауто на новата система от бази данни.

Публикации, свързани с дисертационния труд

1. Briasouli, A. (2021). Research on advanced technologies–design and development of cloud computing model. Technology transfer: fundamental principles and innovative technical solutions, 7-9.
2. Briasouli, A., Minkovska, D., & Stoyanova, L. (2021). Development on advanced technologies–design and development of cloud computing model. Technology transfer: fundamental principles and innovative technical solutions, 13-16.
3. Briasouli, A., Minkovska, D., & Stoyanova, L. (2022). SECURITY OF CLOUD COMPUTING THROUGH BIOMETRIC FEATURES. International Journal of Advanced Research in Engineering and Technology (IJARET) V13 21-30.
4. Briasouli, A., (2022). DATA SECURITY AND PRIVACY IN CLOUD COMPUTING – AN EMPIRICAL STUDY. International Journal of Advanced Research in Engineering and Technology (IJARET) V13 22-35.

SUMMARY

In the thesis titled "Examination of advanced technologies - design and development of cloud computing model of databases for administration," authored by Briasouli Alexandra, the exploration begins with an overview of the evolution of data-driven decision-making and the crucial need for adaptable databases in administrative contexts. The study delves into the model's fundamental components, emphasizing scalability, integrity, security, automation, and efficiency. Moving forward, it explores various aspects of cloud computing architectures, service models, and their profound impact on modern organizations. The discussion extends to NoSQL cloud computing databases, database schema, Python code, Flask web server, and a specialized grading system tailored for the Intercloud Exchange Platform. The thesis concludes by thoroughly evaluating the platform's performance and providing valuable insights into the discriminant analysis of primary research, technological exploration, and forward-looking recommendations for future developments.

A significant aspect of the thesis involved discriminant analysis applied to primary research, focusing on business and entrepreneur/supervisor characteristics. Utilizing Discriminant Analysis, the research aimed to create ranking functions for businesses adopting cloud computing in Greece. It analyzed criteria such as years of operation, number of employees, entrepreneur/supervisor's age, position, and educational level.

The technological exploration undertaken in the study was crucial, providing in-depth insights into the strengths and weaknesses of various technologies. Noteworthy technologies explored included NoSQL databases and the Flask web framework. The knowledge gained from this exploration proved essential for informed decision-making in technology selection. Looking ahead, the thesis recommended key enhancements for the Intercloud Exchange Platform, encompassing the incorporation of a graphical user interface, integration of precise geographic data, consideration of request complexity, expansion of service offerings, and the implementation of distributed systems.

Additionally, future research endeavors were suggested, including conducting similar surveys in different regions of Greece to refine discriminant functions and further assess the extent of cloud computing adoption, contributing to the continuous improvement and evolution of cloud computing models for effective administration.