

Vinnitsia National Technical University
Faculty of Intelligent Information Technology and Automation
Department of Automation and Intelligent Information Technologies

MASTER'S THESIS

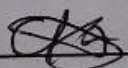
on the subject:

**Mobile application development for filtering and searching of relevant
information**

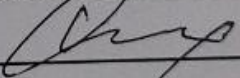
Performed by: 2nd year student of group 3AKIT-22m
specialty 151 - Automation and computer-integrated
technologies

Oleksandr DRANCHUK  08.12.2023

Supervisor: Candidate of Technical Sciences,
Associate Professor of the Department of AIT

Yaroslav KULYK  13.12.2023

Opponent: Candidate of Technical Sciences,
Associate Professor of the Department of CS

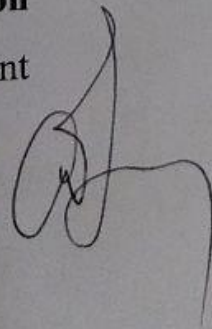
Oleksiy SILAGIN  14.12.2023

Admitted to representation

Head of the AIT Department

D., prof. Oleh BISIKALO

"18" 12 2023



Vinnitsia VNTU - 2023

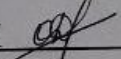
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА


на тему:

**Розробка мобільного додатку фільтрації та пошуку релевантної
інформації**


Виконав: студент 2-го курсу групи ЗАКІТ-22м
спеціальності 151 - Автоматизація та
комп'ютерно інтегровані технології

Олександр ДРАНЧУК  08.12.2023

Керівник: к.т.н., доцент кафедри АІТ

Ярослав КУЛИК  13.12.2023

Опонент: к.т.н., доцент кафедри КН

Олексій СЛАГІН  14.12.2023

Допущено до захисту

Зав. кафедри АІТ

д.т.н., проф. Олег БІСІКАЛО

"18" 12 2023 р.

Вінниця ВНТУ - 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій
Рівень вищої освіти другий (магістерський)
Галузь знань 15 – Автоматизація та приладобудування
(шифр і назва)
Спеціальність 151 – Автоматизація та комп'ютерно-інтегровані технології
(шифр і назва спеціальності)
Освітньо-професійна програма Інформаційні системи і Інтернет речей
(назва освітньо-професійної програми)

ЗАТВЕРДЖУЮ
Завідувач кафедри АІТ
д.т.н., проф. Олег БІСІКАЛО
18. 12 2023 року

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Дранчуку Олександрю Ігоровичу

(прізвище, ім'я, по батькові)

1. Тема роботи: «Розробка мобільного додатку фільтрації та пошуку релевантної інформації»

Керівник роботи Ярослав КУЛИК, доцент, доцент кафедри АІТ
(Ім'я ПРИЗВИЩЕ, науковий ступінь, вчене звання)

Підставою для виконання роботи є наказ № 247 по ВНТУ від 18.09.2023 року.
Затверджене протоколом №1 засідання кафедри АІТ від 30.08.2023 року.

2. Термін подання студентом роботи 15.12.2023

3. Вихідні дані до роботи:



вид програмного забезпечення – мобільний додаток; підтримка ОС – iOS; підтримка версії ОС – 9.0+; розмір додатку – 25 МВ; мови графічного інтерфейсу – українська, англійська.

4. Зміст текстової частини

Вступ. Аналіз об'єкта дослідження. Формування мети дослідження. Постановка задач дослідження. Вибір інструментарію для досягнення поставленої мети дослідження. Проектування програмного забезпечення. Реалізація програмного забезпечення. Тестування програмного забезпечення. Аналіз результатів тестування програмного забезпечення. Висновки. Список використаних джерел. Додатки.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)
UML-діаграми алгоритму додатку. Модель Lean Canvas роботи. Структура бази даних. Результати тестування додатку.

6. Консультанти розділів роботи

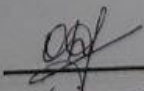
Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Виконання прийняв
Спеціальна частина	к.т.н., доцент, доцент кафедри АІТ Ярослав КУЛИК	25.09.2023 	05.12.2023 

7. Дата видачі завдання 25.09.2023

КАЛЕНДАРНИЙ ПЛАН


№	Назва та зміст етапу	Термін виконання		Прим.
		початок	закінчення	
1.	Вибір, узгодження та затвердження МКР	18.09.2023	25.09.2023	Вик.
2.	Аналіз літературних джерел. Попередня розробка основних розділів	26.09.2023	17.10.2023	Вик.
3.	Розробка технічного завдання (ТЗ)	18.10.2023	24.10.2023	Вик.
4.	Вибір технології для реалізації системи	25.10.2023	30.10.2023	Вик.
5.	Розробка програмного забезпечення	01.11.2023	20.11.2023	Вик.
6.	Тестування системи	21.11.2023	24.11.2023	Вик.
7.	Нормоконтроль	27.11.2023	01.12.2023	Вик.
8.	Попередній захист МКР, допрацював, рецензування МКР	05.12.2023	14.12.2023	Вик.
9.	Захист МКР	18.12.2023	18.12.2023	Вик.

Студент


(підпис)

Олександр ДРАНЧУК
(Ім'я ПРІЗВИЩЕ)

Керівник роботи


(підпис)

Ярослав КУЛИК
(Ім'я ПРІЗВИЩЕ)

Анотація

УДК 004.622

Магістерська робота складається з 70 сторінок формату А4, яка містить 26 рисунків, список використаних джерел з 52 найменувань.

Основною метою роботи є створення програмного забезпечення для пошуку та перегляду новин з фільтрацією за ключовими словами, яке є більш ефективним у відборі необхідної інформації порівняно з аналогами.

У загальній частині роботи представлено особливості розробки сучасних програмних продуктів, проведено аналіз існуючих методів фільтрації інформації. У розрахунково-проектній частині виконано проектування архітектури додатку та бази даних. У технологічній частині розроблено програмне забезпечення для фільтрації та пошуку релевантної інформації в новинах.

Ключові слова: інформація, метод найближчого сусіда, новини, мобільний додаток.

ANNOTATION

The masters thesis consists of 70 pages of A4 format, which includes 26 figures, a list of references contains 52 names.

The main goal of the work is to create a software for searching and browsing news with keyword filtering, which is more efficient in selecting the necessary information compared to analogues.

In the general part of the work presented features of the development of modern software products, an analysis of current methods of filtration of information. In the calculation and design part, the design of the application architecture and data base was performed. In the technological part, the software for filtration and search of relevant information in the news was developed.

Keywords: information, closest neighbor method, news, mobile application.

CONTENTS

INTRODUCTION.....	4
1 GENERAL THEORETICAL INFORMATION.....	7
1.1 General Information about Information Presentation Services.....	7
1.2 Key Concepts of the Theory of Software Development for Mobile Devices....	9
1.3 Target Audience Selection.....	11
1.4 Typology of Mobile Applications.....	13
1.5 Mobile Application Platforms.....	15
1.6 Mobile Application Development Environments.....	18
1.7 Architecture of Mobile Applications.....	20
1.8 Methodologies and Development Methods for Software Products.....	22
1.9 Overview of Existing Filtering and Information Retrieval Algorithms.....	26
1.10 Review of the Selected Filtering Algorithm.....	27
1.11 Relevance of Developing the Current Software Product and Its Purpose.....	29
1.12 Conclusion.....	30
2 CHOICE OF TOOLS AND ENVIRONMENT FOR DEVELOPING THE SOFTWARE PRODUCT.....	32
2.1 Programming Language Selection.....	32
2.2 Database Management System Selection.....	34
2.3 Selection of a Database Management System Client.....	37
2.4 Selection of Development Environment.....	39
2.5 Selection of Development Methodology.....	42
2.5 Evaluation of Results.....	43
2.5.1 Accuracy Analysis:.....	43
2.5.2 Precision and Recall Assessment:.....	45

2.5.3 F1 Score Integration:.....	46
2.6 Conclusion.....	47
3 IMPLEMENTATION OF THE MOBILE SEARCH AND FILTER	
APPLICATION.....	49
3.1 Design and Architecture.....	49
3.2 Form Development.....	52
3.3 Database Design and Development.....	58
3.4 Implementation of the Connection between Forms and the Database.....	62
3.5 Conclusion.....	66
4 TESTING OF DEVELOPED SOFTWARE.....	68
4.1 Choice of Testing Approach.....	68
4.2 Application Testing.....	70
4.3 Conclusion.....	73
CONCLUSIONS.....	74
REFERENCES.....	75
ATTACHMENTS.....	82
Attachment A (required) Technical Assignment.....	83
Attachment B (informational) Illustration Part.....	86
Attachment C (optional) Program listing.....	91
Attachment D (required) Protocol of verification.....	92

INTRODUCTION

Relevance of the Work. In modern society, news plays a crucial role in human activities. To stay informed about the latest developments in various fields, individuals routinely turn to information search services. The primary functions of information services include presenting photos, videos, and textual information to users in a format that is convenient for perception.

Modern information presentation services allow users to choose a news delivery system based on their interests and preferences, making them interesting and useful for consumers [1-2]. However, not all services currently enable the qualitative and rapid analysis and selection of information, especially on devices with limited RAM.

The relevance of this work lies in analyzing modern technologies for information retrieval and filtering, and developing software to implement algorithms for selecting information by keywords in a mobile application for news playback and storage. The software includes content selection based on user preferences, with an independent server component facilitating interaction with a database [3].

Objective of the Work. The goal of this work is to enhance the algorithm for searching and filtering information by keywords and to develop software for playing and storing news with content customization based on reader preferences. Unlike other services, this software allows for the qualitative and rapid analysis and selection of information, particularly on devices with limited RAM. The information retrieval

algorithms based on keywords will more effectively filter information and larger proportion of the algorithm's predictions align with the actual relevance of the articles in the dataset [6].

Research Object. The object of the study is the process of filtering incoming information and the development of algorithms for processing incoming information.

Research Subject. The subject of the study is the algorithm for selecting information by keywords, which will filter data according to the user's query, returning the most relevant information.

Tasks to be Accomplished. To develop this software product, the following tasks need to be accomplished:

- Conduct a review of existing services for presenting new information to users and analyze existing methods of selecting contextual information for users.
- Design a service for processing incoming information that allows for the qualitative and rapid filtering of information based on user preferences.
Develop a service for processing incoming information. Design a graphical interface for presenting processed information to users.
- Implement a content selection system based on user preferences.
- Test the software with the simulation of simultaneous access by a large number of users.

Novelty. The scientific novelty of the research results lies in the use of a unique algorithm for selecting and filtering information that works equally well on devices with limited RAM. Data processing will be based on the selected method of searching

for relevant information, after which the data will be presented to the user in a convenient form [3-8].

Practical Value. The practical value of the research results is to allow users to view news based on their own interests, either by selecting one or several standard categories or by finding relevant information by entering keywords for search in the appropriate field. The user is provided a mobile application with a fast, convenient, simple, and intuitively understandable interface and an improved algorithm for finding relevant information, which will improve the efficiency of information search.

Approval of Research Results. The main goals of the master's qualification work were published in the materials of the annual regional scientific and practical Internet conference for students, postgraduates, and young researchers "Youth in Science: Research, Issues, Prospects" (Vinnytsia, VNTU, 2021), and in the materials of the annual scientific and technical conference of teachers and students of VNTU (Vinnytsia, VNTU, 2021) [1].

Research on the topic of the master's qualification work will be conducted based on the individual assignment developed and approved by the Department of Automation and Intelligent Information Technologies of VNTU, as well as the technical task for scientific research.

1 GENERAL THEORETICAL INFORMATION

1.1 General Information about Information Presentation Services

The evolution of mass media, spanning "traditional" language-based channels like television, radio, cinema, compact discs, or DVD discs, and print media, to the vast expanse of the Internet and virtual services like the World Wide Web, has redefined the fabric of societal connectivity [3]. Mass media has transcended its conventional role, now an indispensable force that shapes our perceptions, interactions, and daily lives. In the contemporary landscape, envisioning a world without the influence of television, email, video-sharing websites, internet news portals, or blogs is a challenge, underscoring the extraordinary significance of mass media. Its traditional role as a window to the world continuously expands, reflecting the dynamic nature of the digital age.

Moreover, the metamorphosis of mass media goes beyond dissemination—it has acquired multifaceted functions, evolving into a dynamic forum that not only conveys information but also catalyzes social interaction and communication. The boundaries between content consumers and producers blur in the digital realm, with individuals empowered not only to consume but also to contribute to the ever-expanding pool of media content. This democratization of information has given rise to a virtual agora, a space where diverse voices converge in a global exchange of ideas, opinions, and experiences.

In parallel, information, an enduring value in societal development, has historically found its sanctuary in libraries, serving as the bastions of documented knowledge [1-4]. However, societal dynamics have undergone a seismic shift, transitioning from an era grappling with an information deficit to one inundated with an information surplus. The contemporary information flow is a complex mosaic generated by a myriad of internet resources—from news and socio-publicistic portals to feeds from government institutions, non-governmental organizations, personal websites, blogs, and the social media pages of public figures.

The exponential growth of information resources has given rise to an imperative for specialized programs and professionals equipped to navigate and process this digital deluge [2-7]. Information-analytical centers, spanning business, economics, politics, law, social sciences, and beyond, confront the formidable task of processing and analyzing vast volumes of both structured and unstructured data. As the digital age unfolds, the challenge of optimizing methods for working with this burgeoning information becomes increasingly apparent. A growing global network user base demands up-to-date information, emphasizing the need to optimize the time spent searching for relevant content [4]. This challenge has spurred the development of new tools, including specialized services—aggregators designed to accumulate information from diverse sources.

In the context of this rapidly evolving landscape, the exploration of news aggregators in the creation of information-analytical products emerges as a relevant and vital area of research. This work is dedicated to unraveling the intricate interplay

between news aggregators and the practice of synthesizing information for analytical purposes, shedding light on the evolving tools that shape our engagement with the vast sea of information in the digital era.

1.2 Key Concepts of the Theory of Software Development for Mobile Devices

The expansive concept of "Software Development" encompasses a diverse array of software engineering principles and knowledge areas, with mobile application development emerging as one of the most dynamic and influential directions in contemporary times. This facet of software development is a multifaceted process dedicated to creating and maintaining the functionality, quality, and reliability of specific software products. It draws on modern technologies and methodologies from various disciplines, including intelligent information sciences, project management, and engineering.

In the realm of software development, challenges abound, ranging from ensuring the quality and reliability of the end product to addressing issues of affordability and compliance with local laws or the internal standards set by developers themselves. The intricacies of crafting software products often involve the creation of thousands of lines of code, intricate connections, and dependencies. This

complexity is on par with the intricate design of today's automated machines, highlighting the sophistication inherent in modern software systems.

Mobile Application Development, as a specialized subset of software development, is the intricate process of crafting software tailored to the unique architectural features of mobile devices such as PDAs, smartphones, and phones. These applications can either be installed during the device's development or downloaded through specialized platforms, reflecting the versatility and accessibility that defines the mobile application landscape. The realm of mobile development is witnessing an exponential increase in practitioners, a growth that mirrors the rapid evolution of mobile devices themselves.

Within the sphere of mobile application development, a comprehensive understanding is crucial. It involves a set of processes and methodologies that encompass essential planning and development steps, all directed towards the creation of a final software product tailored for the target audience. Therefore, the task of developing mobile software necessitates a methodical approach. It begins with the fundamental step of determining the target audience, followed by defining the type of application and specifying the devices it is intended for. Choices must be made regarding the platform on which it will operate, considering factors such as whether a native or cross-platform implementation is more suitable. This decision, in turn, influences the selection of the programming language, development environment, architecture, and methodology that will be employed in the creation of the software program.

The dynamic landscape of mobile application development demands not only technical expertise but also strategic foresight. The choices made at each stage of the development process have far-reaching implications, impacting user experience, accessibility, and the overall success of the software in the ever-evolving mobile ecosystem. As the digital era progresses, the theory of software development for mobile devices continues to evolve, shaping the way we engage with technology on a daily basis.

1.3 Target Audience Selection

The target audience is a group of people for whom a specific product or software product is intended. Forming a target audience is based on the purpose and goals of the implemented product.

To determine the target audience, the following questions should be addressed:

- What is the purpose of your product?
- Who will buy your product, and for whom is its use beneficial?

Forming a target audience is a fundamental task for creating a successful product. The target audience can be considered a specific group of people who share certain demographic data and interests.

To determine demographic correspondences, the following factors can be considered:

- Age: The age range should be reasonably broad but not excessively so. For example, a range of 15-70 may not be effective, as the interests of a 15-year-old and a 50-year-old are likely to differ. There are also differences in perception and usability preferences; older individuals may find a simpler, less information-overloaded interface more convenient. Ideally, the age should remain within one generation, such as 13-18 or 20-30.
- Gender: Men and women are equal but not identical in their tastes and desires. The stylistic design and emphasis of the application can vary significantly depending on the user. Research shows that men make purchases because they need to, marking it off their to-do list. Women make purchases because they enjoy it; they want to browse and learn more about the product. Considering this aspect is important for the success of the application when forming the target audience.
- Education: Education can include several levels, such as elementary school, middle and high school, university, or continuous education. The level of education directly influences the perception and feelings of the audience towards the application, requiring a different approach in conveying ideas and forming the marketing campaign of the application.
- Location: The world is vast, so if the goal is to attract the maximum number of users, it is necessary to encompass diverse nations, cultures, and values. While addressing more localized needs, it is crucial to understand the specific characteristics of the target audience in that region.

Issues at the regional, state, or even national level may have a smaller scope but a more qualitative and interested audience. The popular aspects influencing the choice of the project's target audience have been listed above. Additionally, numerous other indicators, such as occupation, family status, interests, and more, play a crucial role in defining the specific characteristics of the target audience for a project.

1.4 Typology of Mobile Applications

Mobile applications can be categorized into:

- Native Applications
- Web Applications
- Hybrid Applications

Native applications are programs developed using a programming language specialized for the chosen operating system. For example, Objective-C and Swift for iOS applications, and Java and Kotlin for Android. Advantages of developing native applications include:

- Speed and stability.
- Absence of functional constraints.
- Platform tools that allow developers to flexibly configure all aspects of the application for optimization and maximum performance.

Web applications are programs developed using technologies such as HTML, JavaScript, and CSS. They require a configured browser on the device with active internet access for execution. HTML is used for graphical interface markup, CSS assists in describing the visual part and mutual positioning of control elements, while JavaScript is used to implement the program's logic. Advantages of using web applications include:

- Ability to use (or a larger part of) functionality on devices with different operating systems.
- Do not require significant use of device graphic resources.
- Availability of a wide range of development tools that facilitate the development process.

Hybrid applications are applications that partially utilize the capabilities of both native and web applications. Advantages of using hybrid applications include:

- Simultaneous distribution to multiple operating systems.
- Sufficient speed due to the native component of the application.
- Possibility of initial publication as a finished product or temporary release until the full version of the application is launched in the future.

A comparative analysis of application types is provided in Figure 1.1.

	Native App	Web Based APP	Hybrid App
Native UI	√		
Accessing device features	√		√
Performance	√		
Multiplatform support		√	
Distribution	Application Portal	Web access	Application Portal

Figure 1.1 - Comparison of application types.

1.5 Mobile Application Platforms

Mobile devices have evolved into universal and portable computers, leading to the development of various platforms and operating systems for implementing mobile products. Today, the mobile device's operating system (OS) is as crucial as the device manufacturer. The OS directly influences the program's speed, user interface, app availability, and their functionalities.

Currently, the most widespread mobile platforms are iOS by Apple and Android by Google.

iOS is a mobile operating system developed by Apple for its devices. Initially designed for the iPhone, it later extended to iPod Touch and iPad. iOS is exclusive to

Apple's products, prohibiting its use on devices from other manufacturers. The OS interface is derived from the Multi-Touch concept, featuring control elements like fields, switches, and buttons to provide users with an interface for interacting with the application's main logic. The built-in accelerometer allows automatic device orientation switching and can be used for specific purposes, such as surface tilt measurement.

iOS stands out among other operating systems for its superior speed and smooth operation of programs and the user interface. Since the release of the first iPhone, the design of the OS and applications has become a benchmark for other mobile devices.

iOS is also one of the most secure operating systems overall. Each application requests the user's permission to use services like geolocation, gallery access, phone book, and more. A detailed list of all application settings is available in the device's main settings menu, allowing users to make changes at any time.

App Store is the application store for Apple devices, boasting over 2 million apps for various purposes. Through the App Store, users can browse and download the programs they need, while developers have the opportunity to publish apps for download.

Android is an operating system developed by Google for mobile devices, tablets, e-books, media players, smartwatches, gaming consoles, multimedia systems, televisions, and even Google Glass. The OS kernel is Linux, and it features Google's implementation of the Java virtual machine. Initially developed by Android Inc.,

Google acquired it in 2005. Subsequently, Google initiated the creation of the Open Handset Alliance (OHA), currently responsible for supporting and further developing the platform. Android allows the creation of native applications using Java and Kotlin, managing the device using Google-developed libraries. Android is an open system, enabling manufacturers of various devices to use it for their own needs.

Google Play Market is the store for apps, games, books, and music on Android, serving as an equivalent to the App Store on iOS. It allows developers to upload their own apps for users to purchase. However, there are restrictions on this platform; for example, developers from certain countries may not publish paid apps. Google Play Market emerged from the rebranding of the Android Market on March 6, 2012. In May 2014, the option to pay for apps using PayPal was added, and at the end of July 2017, an embedded antivirus—Google Play Protect—was introduced.

In Android version 1.6, developers were provided with the Native Development Kit, allowing the creation of low-level modules for the system in C/C++, relying on standard Linux libraries.

Specialized applications developed by Google are required for the device to access Google services, and the right to install them is granted to manufacturers of "smart" technology only after entering into a contract with Google.

1.6 Mobile Application Development Environments

The choice of development tools depends directly on the operating system, the purpose of the application, its complexity, and specific features. The development environment provides access to tools for managing the operating system and various phone components. High-performance programming languages are used to write software, enabling the achievement of maximum application speed.

Different programming languages are used for various platforms:

- For devices running the Android operating system, programming languages such as Java and Kotlin are predominantly used. However, the use of languages like C/C++ is also possible, allowing for increased productivity in specific code sections.
- For devices operating on the iOS system, programming languages like Objective-C and Swift are employed. Swift, introduced by Apple in 2014, is a relatively new programming language.

The most common Android application development environments are shown in Figure 1.2.

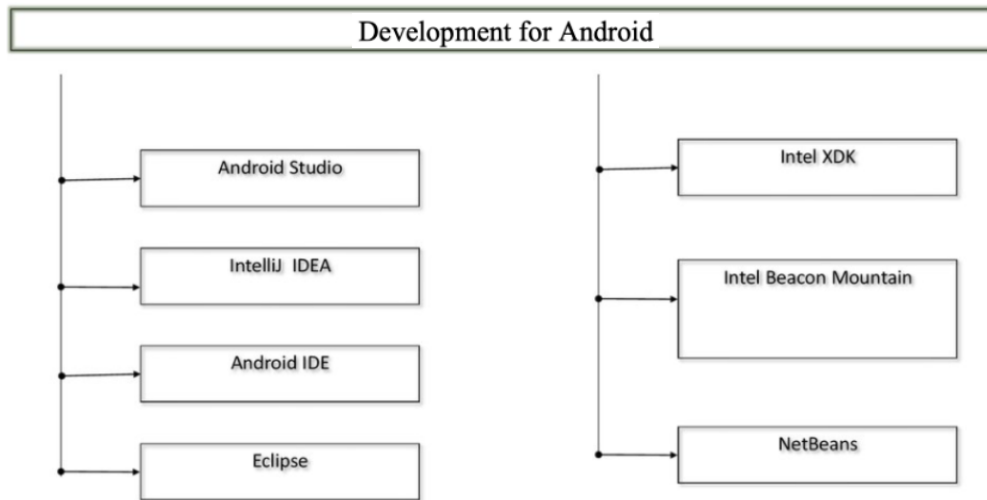


Figure 1.2 - Android application development environments.

The most popular iOS application development environments are shown in Figure 1.3.

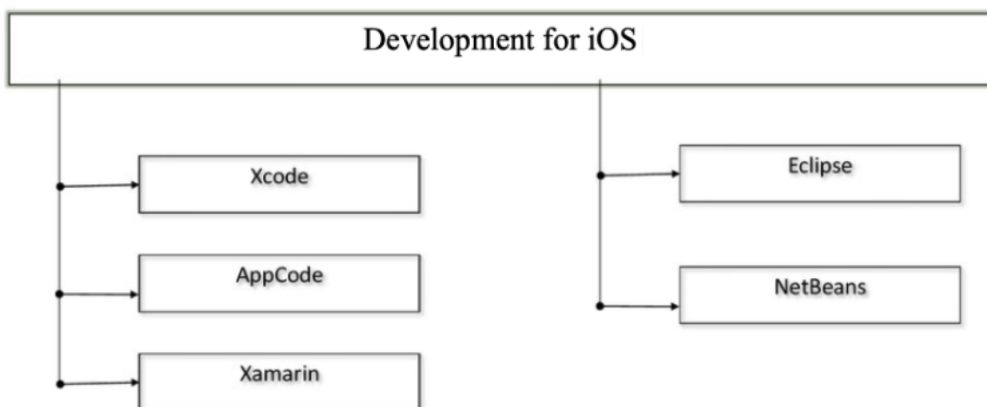


Figure 1.3 - iOS application development environments.

1.7 Architecture of Mobile Applications

The architecture of any application is an integral part of its composition. A good architecture is considered one that combines the following characteristics:

- Independence from External Modules and Frameworks: A well-developed architecture should not rely on the existence of any external modules or frameworks. This allows developers to use frameworks as tools without being restricted by their capabilities.
- Capability for Automated Testing: The application's business logic should be adaptable for automated testing, enabling the improvement of the quality and efficiency of the application.
- Independence from the User Interface: It should be easy to modify and expand the user interface without significant changes to the application's business logic.
- Independence from External Agents: The business logic should not have a rigid connection to any external resource.
- Openness and Extensibility of Software Components: The architecture should be open, allowing for the extension of software components without modifying them.

The reflection of these principles of interaction in the application architecture is illustrated in Figure 1.4.

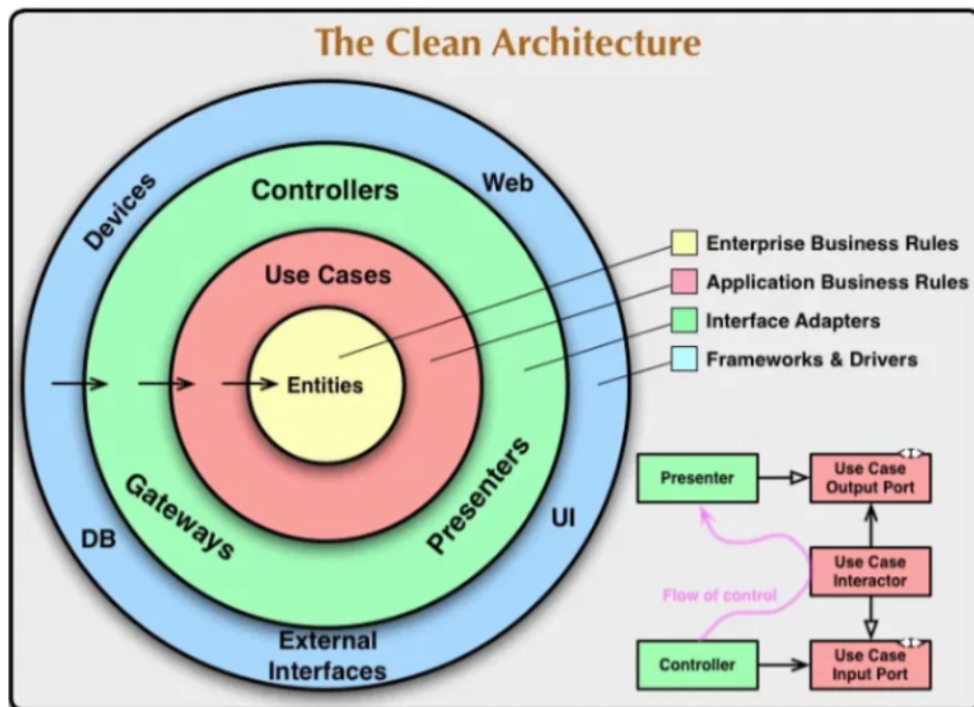


Figure 1.4 - Main application architecture patterns.

In this scheme, it is worth noting:

- Entities: The business logic of the application.
- Use Cases: Scenarios of use and logic of the program that manages the data flow from the previous layer.
- Interface Adapters: Adapters that provide data conversion and communication between Use Cases and the external layer, including the UI elements.

- Frameworks & Drivers: The external layer containing frameworks and external modules of the application. It also includes the visual component of the application.

1.8 Methodologies and Development Methods for Software Products

Throughout many development cycles of applications, development management methodologies have changed repeatedly, taking various forms, often following strict laws of planning, documentation, and reporting. However, due to the rapid and extensive development of the IT industry, project management has transformed from a set of rules into a complex area with a large number of methodologies, each built on its own connections and principles.

Among the most widespread development methodologies currently, the following can be noted:

- Agile;
- Scrum;
- Kanban;
- Waterfall.

Let's consider each of them in more detail.

Agile Methodology consists of several repetitive, short cycles (about 2-3 weeks each) of teamwork. At the end of each cycle, the development team informs the

customer about the results and makes adjustments to the application if necessary. While more than one cycle is needed to obtain the final product, each cycle, upon completion, presents a ready part of the application's functionality that can be tested or replaced. At the end of the cycle, the team summarizes the results, analyzes customer feedback, and plans for the next cycle.

Advantages of this methodology include:

- Thorough planning due to short cycle durations;
- Quick problem detection and resolution. Agile minimizes the possibility of serious errors in the application's operation;
- Constant communication with the customer. If changes to the application's operation are necessary, this can be easily done without significantly disrupting the development process.

Scrum is part of the Agile methodology and represents a more structured approach that requires the presence of certain mandatory components. The main principles of the Scrum methodology are division and optimization. All large tasks should be divided into smaller parts, distributed among team members, and worked on sequentially. Additionally, Scrum involves holding regular meetings during the cycle, allowing for constant awareness of the status of the development of various modules and promptly identifying and resolving issues.

Advantages of the Scrum methodology include:

- Elimination of ambiguity and misunderstandings. Thanks to detailed planning, breaking tasks into smaller and regular checks of the completion status, situations of ambiguity in the development process are minimized;
- Good organization. Breaking down complex tasks into manageable parts makes Scrum suitable for executing large projects. Clear delineation of responsibilities and planning ensures transparency and quality control over the development process.

Kanban emphasizes continuous delivery of updates to the customer without overloading the development team. The main principles of Kanban are illustrated in Figure 1.5.

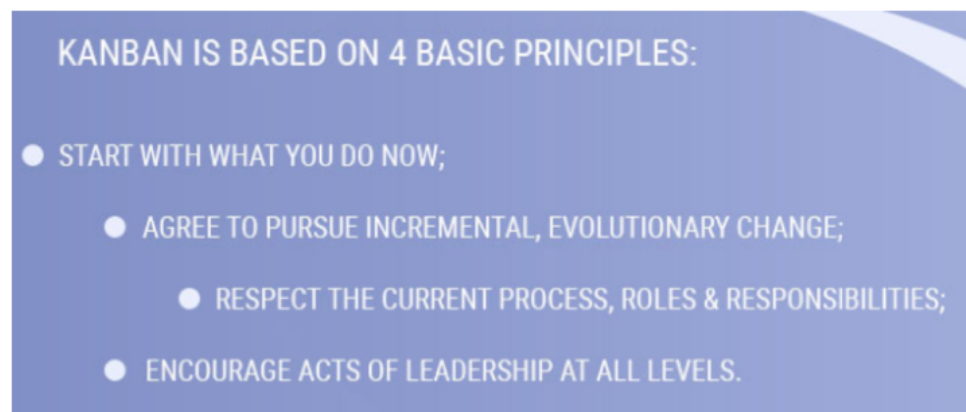


Figure 1.5 - Basic principles of the Kanban methodology.

Visualization of work elements typically occurs through a Kanban board, providing the team with the ability to visualize progress, aiding in a clear definition of the workflow.

Among the advantages of the Kanban methodology are:

- Flexibility in planning: The development team focuses on one task, but the client can change task execution priorities during development. This allows for quickly addressing the most prioritized tasks. Upon completing the current task, the development team moves on to the next, concluding any unfinished work.
- Increased efficiency: Visualization of the development process helps quickly identify areas of inefficiency. Once the team identifies a problem, they can immediately address it.
- Reduced workload: Traditional management methodologies rely on pre-planning the work of developers, leading to rushed task completion to meet cycle deadlines, resulting in developer fatigue and, in some cases, decreased task execution quality.

Waterfall is a linear and sequential software development methodology, where each task follows the completion of the previous one.

Projects using this methodology are implemented in a single long-term cycle. All requirements are defined at the beginning of the project, and in case of failure to fulfill some of them, the team needs to go through all stages from design to

implementation or bug fixing from the very beginning. Each stage concludes with a transition to the next.

Advantages of the Waterfall methodology include:

- Clear structure and organization: Waterfall focuses on the clear and sequential execution of a defined set of steps.
- Ease of control and management: Each stage has a set of necessary procedures and defined checking processes. This work can be easily organized by following hierarchies.
- Testing: Testing scenarios are defined in the functional specification of the project, making the testing process simpler and more straightforward.

Jira service, based on the Kanban methodology, was selected as development methodology for the product, involving task planning and assignment for the development team.

1.9 Overview of Existing Filtering and Information Retrieval Algorithms

Automatic information retrieval systems are utilized to reduce "information overload." A search system examines all available information units (documents) in the collection and selects documents according to the information query [3, 4]. Since real search systems do not find all relevant documents, one can discuss the precision of search systems. The result of a search system is a list of selected documents,

among which there are those relevant to the query. For an ideal search system, the lists of selected documents and those relevant to the query should match. In real search systems, the lists of selected documents also contain irrelevant ones [5]. Filtering information messages is carried out not only to determine relevance but also to address issues related to the ambiguity of language—where the same term can denote various concepts, and the same concept can be denoted by different terms.

In general, the essence of the filtering procedure is an algorithm that, by reviewing a set of documents (D_1, D_2, \dots, D_n), establishes their relevance to a search query [3].

The mathematical foundation of information processing algorithms includes methods such as:

- Support Vector Machine (SVM) method;
- k-Nearest Neighbors (k-NN) method;
- Rocchio method.

In this work, information retrieval and filtering will be performed using the k-Nearest Neighbors (k-NN) method.

1.10 Review of the Selected Filtering Algorithm

The "nearest neighbor" algorithm starts at an arbitrary point and gradually "visits" each nearest point that has not been "visited" yet. The traversal points are

sequentially included in the route. Each subsequent point added to the route must be the nearest to the last selected point among all other points not yet included in the route. The algorithm terminates when all points have been "visited" [5]. The route's points are the content components of the document.

The input data for the set of points V is of dimension N . The output data is a route T consisting of a sequence of visited points from set V . The algorithm's operation sequence includes:

1. Select an arbitrary point V_1 ;
2. $T_1 = V_1$;
3. for $i = 2$ to $i = N$, do;
4. Select point V_i closest to point T_{i-1} ;
5. $T_i = V_i$;
6. $T_{(N+1)} = V_1$;

End of the algorithm and decision-making on the relevant information source.

The relevant source is selected based on the minimum distance $V_i, T_{(i-1)}$.

The algorithm is simple to implement, executes quickly, but, like other "greedy" algorithms, may yield suboptimal solutions. The computational complexity of the algorithm is $O(n^2)$. The result of the nearest neighbor algorithm is a route approximately 25% longer than the optimal one. One heuristic criterion for evaluating the solution is the rule: if the path taken in the last steps of the algorithm matches the path taken in the initial steps, the found route can be conditionally considered

acceptable; otherwise, there may be better solutions. Another criterion involves using the lower-bound estimation algorithm [5].

Advantages of this method compared to analogs include:

- Faster solution finding speed;
- The goal of the search is not a guaranteed correct solution but rather the best among the possible ones.

Disadvantages of this method include:

- Difficulty in choosing the proximity measure (metric), resulting in a large volume of records to be stored in memory to achieve the necessary classification or prediction;
- Typical tasks for this method involve small dimensionality in terms of the number of classes and variables.

1.11 Relevance of Developing the Current Software Product and Its Purpose

The need for developing this software product lies in enabling users to view news based on their interests. Users can achieve this by selecting one or several standard proposed categories or by finding relevant information by entering keywords for search into the corresponding field. This includes models of phones with limited RAM.

Data processing will be carried out using one of the presented methods for searching relevant information, specifically the Nearest Neighbor method. Subsequently, the data will be presented to the user in a convenient and perceptible form [4, 5].

The user is provided with a fast, convenient, simple, and intuitively understandable interface that does not require special skills for using the application. Support is implemented for all mobile devices running on the iOS operating system version 9.0 and above.

1.12 Conclusion

This section provides general theoretical information about information presentation services, key concepts of software development for mobile devices, target audience selection, typology of mobile applications, mobile application platforms, mobile application development environments, architecture of mobile applications, methodologies and development methods for software products, overview of existing filtering and information retrieval algorithms, review of the selected filtering algorithm, and the relevance and purpose of developing the current software product. The text covers various aspects related to mass media evolution in the digital age as well as the intricacies involved in developing software for mobile

devices. It also discusses different methodologies used in software development and provides an overview of filtering algorithms used in information retrieval systems.

An analysis of the general concepts related to the research object was conducted, and the main methods of information filtration necessary for creating a high-quality software product were discussed. These methods include:

- Support Vector Machine (SVM) method
- Rocchio method
- Nearest Neighbor method

The chosen data filtration algorithm, specifically the Nearest Neighbor method, was examined in detail. The most popular platforms for product implementation and current development environments for implementing the required functionality were reviewed. An analysis of architecture and development methodologies was also conducted.

The Jira service based on the Kanban methodology was selected for task planning and assignment for the development.

2. CHOICE OF TOOLS AND ENVIRONMENT FOR DEVELOPING THE SOFTWARE PRODUCT

2.1 Programming Language Selection

In the intricate realm of software development for mobile devices, the selection of a programming language plays a pivotal role in determining the success and efficiency of the final product. In the specific context of developing a software application for users with iOS devices or tablet computers with internet access, the choice of programming language becomes even more critical. For this task, Swift, Apple's high-level programming language, was deemed the most fitting.

Swift, while belonging to the category of high-level languages, defies the typical trade-off of being less performant than low-level languages. Apple has invested significant efforts in optimizing Swift for speed, making it a standout choice for mobile application development. According to developers, Swift's search algorithm runs up to 2.6 times faster than its predecessor, Objective-C, and an impressive 8.4 times faster than Python 2.7 [8]. The language's efficiency is underscored by its ability to handle memory management independently through Automatic Reference Counting (ARC). This feature ensures that memory leaks, a common concern in languages like Objective-C, are virtually eliminated, allowing programmers to focus on the core logic of their applications and innovative capabilities [9].

Swift's prominence is further supported by survey data from Stack Overflow, where nearly 80% of professionals express satisfaction with or plan to work with Apple's development tool. In a survey of over 26,000 professionals from more than 150 countries, about a third of respondents were primarily engaged in mobile software development on the iOS platform. Interestingly, less than half were involved in creating applications for the competing Android OS, and approximately 20% of respondents remained undecided, likely including experts who regularly develop software for various platforms [8-10]. This data highlights the robust adoption and satisfaction levels among professionals working with Swift.

In the broader landscape of software development, the selection of a programming language is only one facet of the overall architecture. For interacting with the database in our case, the SQL (Structured Query Language) programming language was chosen. SQL, a declarative programming language, is renowned for its use in interacting with databases. It facilitates the formulation of queries, updates, and management of relational databases, as well as the creation and modification of database schemas and access control.

Unlike conventional programming languages like C or Pascal, SQL operates as a declarative language for forming interactive queries and acts as a set of data management instructions when embedded in applications [13, 14]. Its core functionality revolves around forming commands for searching, inserting, updating, and deleting data, complemented by robust management and administrative functions

[15]. This makes SQL an ideal choice for interacting with databases and performing the essential tasks required for the seamless functioning of the software application.

In conclusion, the strategic choice of Swift for the application's core logic and SQL for database interaction aligns with the intricacies of the task at hand, ensuring a robust and efficient development process. The symbiotic use of these languages reflects a thoughtful approach to technology selection, underpinning the success of the software application designed for iOS devices.

2.2 Database Management System Selection

In a dynamic mobile application development environment, careful selection of a suitable database management system (DBMS) is of paramount importance as it has a significant impact on the efficiency, security, and overall performance of the application. A DBMS serves as the foundation for organizing, searching, and storing data in an application's database in a seamless manner. Among the many relational DBMS options available, including MySQL Server, MySQL, PostgreSQL, and Oracle, a wise choice depends on a fine-grained evaluation of specific features aligned with diverse application requirements.

The following popular relational DBMS are distinguished:

- MySQL Server;
- MySQL;

- PostgreSQL;
- Oracle.

The needs of modern mobile applications require a balanced approach to choosing a DBMS, taking into account numerous requirements:

- simplicity and flexibility in creating applications;
- multi-user and multi-aspect use of data;
- ease of use, simplicity, and flexibility;
- ease and flexibility in changing, expanding, and configuring the database;
- efficiency and flexibility in storing and retrieving data;
- low cost of storing and using data;
- protection against unauthorized access, distortion, and destruction;
- maintenance of the necessary level of data independence;
- maintenance of the required level of data integrity;
- advanced administrative tools.

The harmonious integration of these facets is instrumental in realizing a resilient foundation for the database, essential for the success of the mobile application.

For the purposes of this master's work, MySQL is the best choice as it fully meets the key requirements outlined in the document. Known for its open nature, MySQL is recognized for its exceptional support for a wide range of programming languages. Its compact yet powerful architecture with multithreading capabilities

provides high performance, stability, and ease of use, making it the optimal solution for small and medium-sized applications.

The choice of MySQL is supported by its ability to seamlessly serve an unlimited number of users interacting with the database simultaneously. Convenient installation and use procedures, scalability with tables supporting up to 50 million rows, fast command execution, and a robust yet simple security system all combine to position MySQL as a versatile and reliable choice.

MySQL is a free relational database management system primarily used for creating various client-server applications due to its excellent support for various programming languages [12].

MySQL is a compact, multi-threaded database server known for its high speed, stability, and ease of use. It is considered a good solution for small and medium applications. The server source codes compile on various platforms, with the server's full capabilities revealed in UNIX systems, where multi-threading support enhances overall system performance [12-14].

Advantages of the MySQL server [13]:

- supports an unlimited number of users working with the database simultaneously;
- easy to install and use;
- tables can have up to 50 million rows;
- high command execution speed;
- features a simple and effective security system.

In conclusion, a comprehensive evaluation of the available DBMS options emphasizes that MySQL is the best choice for the mobile application under consideration. Its full compliance with the defined requirements combined with high-quality features positions it as a reliable choice for effective database implementation. The choice of MySQL ensures not only optimal performance but also data integrity throughout the entire application life cycle, thereby confirming its key role in the success of mobile application development.

2.3 Selection of a Database Management System Client

The following popular clients for managing MySQL databases are distinguished:

- MySQL Workbench;
- Navicat;
- PHPMyAdmin;
- HeidiSQL;
- SQLyog;
- MyDB Studio.

For the implementation of database management, the MySQL Workbench client was chosen.

MySQL Workbench is a tool that provides developers with a complete set of visual tools for creating, editing, and managing SQL queries, connections to databases, and objects.

One of the key features of MySQL Workbench is visual composition. After creating the necessary tables, you can generate ER diagrams to easily establish connections. Other undeniable advantages include schema synchronization and verification, reverse/forward engineering, syntax highlighting, input text error analysis, and code folding [13].

Key features of the MySQL Workbench client include:

- connecting to multiple servers in one window;
- connecting to servers using the command line;
- creating and editing tables, views, stored procedures, triggers, and scheduled events;
- exporting from one server or database directly to another server or database;
- convenient user management;
- importing text files;
- exporting to tabular formats: CSV, HTML, XML, SQL, LaTeX, wiki markup, and PHP arrays;
- viewing and editing tables using a convenient grid;
- mass table changes (type, sorting, etc.);
- bulk insertion of ASCII or binary files into tables;
- online query editor with syntax highlighting and code autocompletion;

- monitoring and closing client processes.

In conclusion, it can be stated that the MySQL Workbench client is suitable for solving the tasks at hand.

2.4 Selection of Development Environment

Currently, there are only a few IDEs for working with the Swift programming language, namely:

- Xcode;
- AppCode;
- Clion;
- CodeRunner.

For the implementation of this software product, the Xcode development environment was chosen because it has the following advantages over competitors:

- developed by Apple, just like the Swift programming language;
- has official support from Apple;
- has the ability to install applications on iOS.

Xcode is an integrated development environment (IDE) produced by Apple. The Xcode IDE provides everything needed for development, from professional code editors with code autocompletion and Cocoa refactoring to configuring open-source compilers [16].

Xcode includes most of Apple's documentation and Interface Builder—an application used for creating graphical interfaces.

Key features of the Xcode development environment:

- support for CocoaPods;
- a convenient graphical editor for creating user interfaces;
- the presence of a built-in emulator;
- a large number of built-in frameworks;
- a built-in LLVM (Low Level Virtual Machine) compiler;
- the ability to create code comments;
- extensive and convenient functionality for version control of the application;
- uses a single workspace window, containing most of the necessary data for work;
- convenient project navigation;
- high-speed code compilation;
- built-in unit testing and test optimization;
- built-in Git version control system.

Using Interface Builder, you can create windows for the program by dragging pre-configured components into it. Components include standard system controls such as switches, text fields, buttons, etc. Components are placed in the window, after which they can be positioned by dragging throughout the window, adjusting attributes using the inspector, and establishing connections between these objects and the code. The content of the window is saved in a nib file, which is a resource file of a special

format. It contains all the information needed for the "UIKit" library to create the same objects in the application at runtime. Loading the nib file creates runtime versions of all objects stored in the file, configures them exactly as they were in Interface Builder [17-20]. Information about the relationship specified by the developer is also used to establish a connection between newly created objects and those already existing in the application. These connections provide the code with pointers to objects in the nib file and provide the objects themselves with the information needed to pass user actions to the source code [19].

In general, using Interface Builder saves a significant amount of time spent on creating the UI. Interface Builder eliminates the need to write code to create, configure, and position objects used for interface development. Therefore, you can see how the interface will look during runtime. The UI is essentially archives of Cocoa objects that do not require code generation [17-19]. Changes to the user interface (UI) do not require code recompilation, and changes in the code do not require UI recompilation.

Thus, it can be concluded that the Xcode development environment is well-suited for solving the tasks at hand.

2.5 Selection of Development Methodology

When creating a software product, a deliberate decision was made to use the Agile Scrum methodology, even if our team consists of one member. Agile Scrum's emphasis on iterative development fits well with the adaptive nature of the project, allowing for continuous improvement of features and quick response to changing requirements.

Despite the fact that Agile Scrum is an individual work, the methodology introduces a structured approach to development. Regular sprint meetings, although autonomous, provide a dedicated space for reflecting on progress, solving problems, and planning the next phase of development. This self-directed iteration allows for systematic exploration of tasks, ensuring that each sprint makes a significant contribution to the overall project goals.

The Trello platform is used to optimize project management within Agile Scrum. Trello's user-friendly interface makes it easy to visualize tasks, helping to manage them effectively and track progress. Since most of Trello's features are available for free, its adaptability to solo projects improves the organization and management of the development process.

The iterative nature of the Agile Scrum methodology is particularly beneficial for solo developers, facilitating adaptation to changing needs and creating a continuous feedback loop. Retrospective meetings, although self-reflective, provide

an opportunity to evaluate the development process and identify areas for improvement, contributing to a continuous improvement cycle.

The decision to implement Agile Scrum, even in a one-person team, stems from its proven ability to increase development agility. By focusing on delivering functional increments in short cycles, the methodology allows a single developer to focus on solving immediate problems and iteratively improving the product based on feedback. This iterative and self-reflective approach is expected to facilitate a fast and efficient development process, leading to the successful implementation of our software product.

2.6 Evaluation of Results

The effectiveness of the proposed algorithm in identifying relevant information within news articles is paramount to the success of this master's thesis. Evaluation metrics such as accuracy, precision, recall, and F1 score were employed to comprehensively assess the performance of the developed algorithm.

2.6.1 Accuracy Analysis:

The accuracy of the algorithm reflects its overall ability to correctly classify news articles as relevant or non-relevant. A higher accuracy indicates a more reliable

system in distinguishing between the two classes. The accuracy metric was computed using the formula (2.1):

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Predictions}} \quad (2.1)$$

Where:

- True Positives (TP): The algorithm correctly identifies and labels relevant articles.
- True Negatives (TN): The algorithm correctly identifies and labels non-relevant articles.
- False Positives (FP): The algorithm incorrectly identifies a non-relevant article as relevant.
- False Negatives (FN): The algorithm incorrectly identifies a relevant article as non-relevant.

This metric offers a global perspective on the algorithm's success, but it should be interpreted judiciously, especially in the context of imbalanced datasets.

A better accuracy implies fewer misclassifications (both false positives and false negatives), indicating that the algorithm is more reliable in distinguishing between relevant and non-relevant information. It is essential to consider the specific requirements of the task and the potential consequences of false positives and false negatives when interpreting the significance of accuracy improvements.

It's advisable to complement accuracy with other metrics, such as precision, recall, and F1 score, to gain a more comprehensive understanding of the algorithm's performance.

2.6.2 Precision and Recall Assessment:

Precision and recall were employed to delve deeper into the algorithm's performance by focusing on specific aspects of its predictions.

Precision emphasizes the accuracy of the algorithm when it predicts an article as relevant. A higher precision implies fewer instances where non-relevant articles are incorrectly classified as relevant.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (2.2)$$

Recall (Sensitivity): Recall gauges the algorithm's ability to capture and identify all relevant articles, minimizing instances where truly relevant articles are missed.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (2.3)$$

Balancing precision and recall is crucial, as they are often inversely related—improving one may come at the expense of the other. Achieving a good balance depends on the specific requirements of the task. For instance, in scenarios where false positives have high costs (e.g., misclassifying important news), higher precision may be prioritized. In situations where missing relevant articles is more critical, higher recall may be favored.

2.6.3 F1 Score Integration:

The F1 score, being the harmonic mean of precision and recall, served as a unified metric to balance the trade-off between minimizing false positives and false negatives. A higher F1 score indicates a more harmonious equilibrium between precision and recall:

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (2.4)$$

This comprehensive evaluation approach not only provides a global assessment of the algorithm's accuracy but also offers insights into its ability to make accurate positive predictions (precision) while capturing a high proportion of actual positives (recall).

As part of the analysis, special attention was given to scenarios where precision or recall needed to be emphasized based on the specific requirements of the task. The

chosen metrics were instrumental in fine-tuning the algorithm's parameters and achieving a balance that aligns with the objectives of this research.

2.7 Conclusion

In this section, an analysis of programming language selection for the development of the software product was conducted. Python was chosen for the development of the data filtering algorithm, while Swift and the iOS operating system were selected for displaying the obtained information and interacting with the user. Analogues were considered, and a well-founded choice of the database management system and client was made, namely MySQL and MySQL Workbench. Xcode was chosen as the development environment for the software product, alternative options were explored, and the advantages and disadvantages of its use were evaluated.

The evaluation metrics employed in this study offer a rigorous and nuanced assessment of the algorithm's performance in the context of identifying relevant information in news articles. The detailed analysis provides a foundation for understanding the algorithm's strengths and areas for improvement, contributing to the overall contribution of this master's thesis.

3 IMPLEMENTATION OF THE MOBILE SEARCH AND FILTER APPLICATION

3.1 Design and Architecture

The application follows an architecture based on the Model-View-Controller (MVC) design pattern. To implement this architectural pattern, all project files are grouped according to their purpose. This ensures convenient and fast navigation within the project, allowing the developer to easily locate the necessary modules of the program. The project structure is illustrated in Figure 3.1.

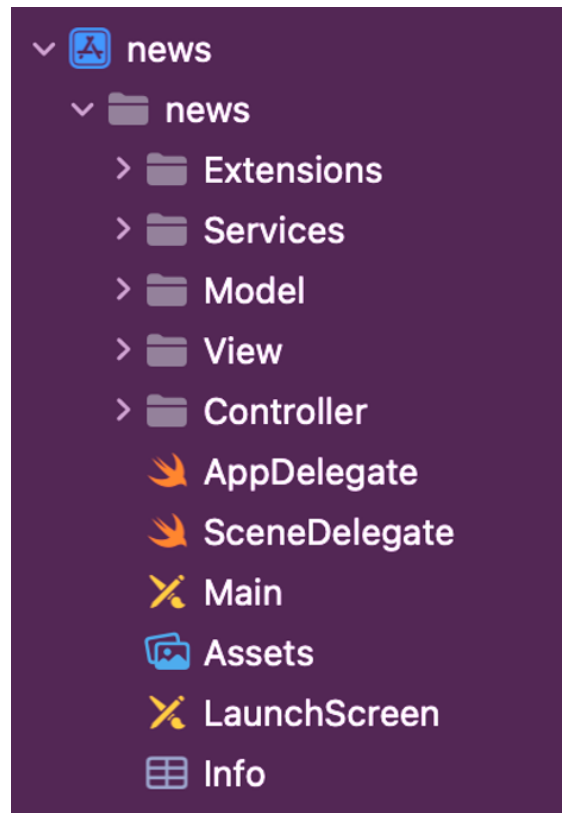


Figure 3.1 - Project structure.

The Extensions folder contains objects that extend program components, providing them with additional functionality needed to solve specific tasks.

The contents of the Extensions folder are shown in Figure 3.2.

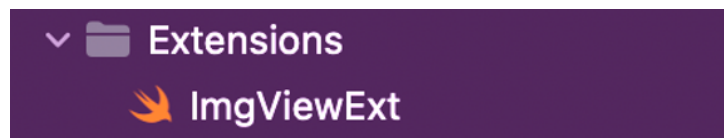


Figure 3.2 - Contents of the Extensions folder.

The Services folder contains the application modules that are responsible for the full implementation of the functionality assigned to them and return a ready-to-use value as a result.

For example, the NetworkService is responsible for the application's communication with the network, downloads and returns data models used for further processing and display to the user.

The contents of the Services folder are shown in Figure 3.3.



Figure 3.3 - Contents of the Services folder

The Model folder contains modules that correspond to the structure of objects and their properties that the view should reflect.

The contents of the Model folder are shown in Figure 3.4.



Figure 3.4 - Contents of the Model

The View folder contains classes of views that are reused in the application on different screens and do not contain the implementation of the application business logic.

The contents of the View folder are shown in Figure 3.5.

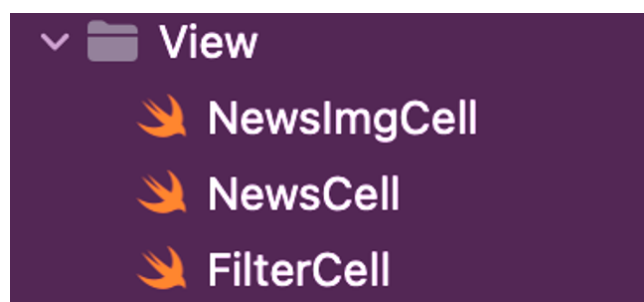


Figure 3.5 - Contents of the View

The Controller folder contains the main elements of all program screens. Controllers are elements that contain the main logic of the application and give instructions on what should be displayed on the screen using views.

The contents of the Controller folder are shown in Figure 3.6.

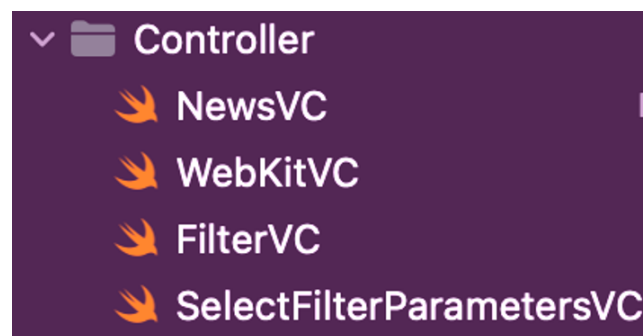


Figure 3.6 - Contents of the Controller folder

3.2 Form Development

Any application always includes a set of tools for interacting with the user – the user interface.

A form is a window that allows users to display and input information, sending it to the server for further processing [8]. The external appearance of the program is mainly presented through forms. Forms are the fundamental building blocks,

providing a container for various control elements. The event mechanism allows form elements to respond to user input, thus interacting with the user.

The form itself and its appearance can be created in the Xcode environment using individual control blocks, whose actions are described in the Swift programming language [8].

In the Swift programming language, a form is programmatically described as a class that inherits all the fields and methods of the View class [9]. The access modifier for form elements is private, meaning access to form elements can only be described within the inheriting class.

In the course project, the following forms were created:

- A form for viewing brief information about a news item.
- A form for selecting the displayed news category.
- A form for searching for news using keywords or news category.
- A form for viewing detailed information about a news item.

Figure 3.7 shows an example of a form for viewing brief information about a news item in the constructor mode in Xcode.

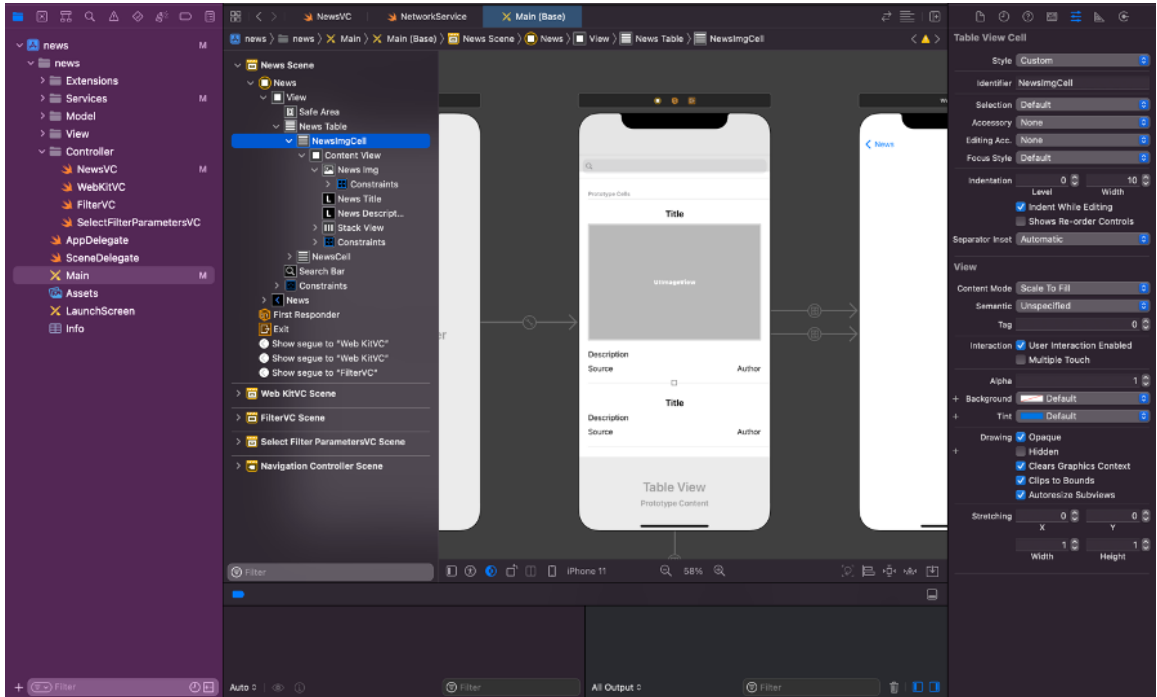


Figure 3.7 - View of the forms for viewing brief information about a news item, selecting a category, and searching for a news item in the designer mode in Xcode

For example, the action and connections with other elements when the left mouse button is pressed on a form can be described using an event handler (Figure 3.8).

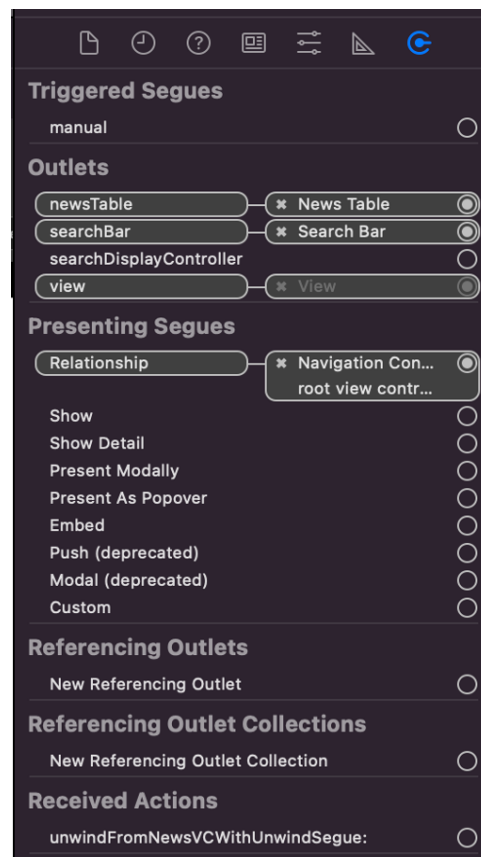


Figure 3.8 - An example of creating a form connection with an event handler

Also, the connection of the form with the event handler can be created by pressing the Ctrl + right mouse button in the constructor mode, and connect the form with the corresponding function - the handler in the code.

To demonstrate an example of a program description of events, we use the program description of the event of clicking on the filter button on the news list view screen, the function - the handler of this event is shown in Figure 3.9.

```
@IBAction func filterButtonTapped(_ sender: UIButton) {
    self.showFilters()
}
```

Figure 3.9 - Description of the ordered actions when you click the filter button

The use-case diagram is shown in Figure 3.10.

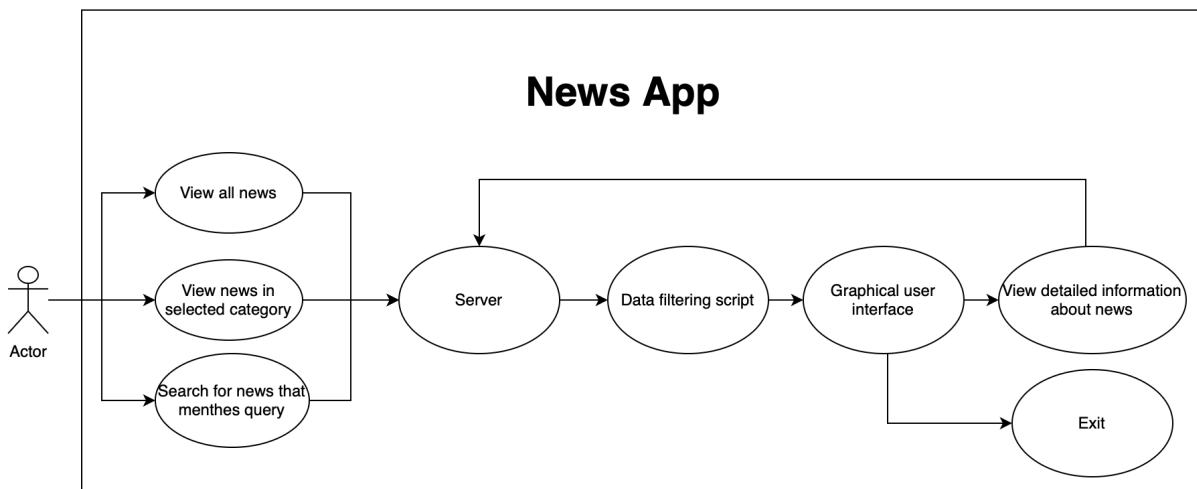


Figure 3.10 - Use-case diagram

The application class diagram is shown in Figure 3.11.

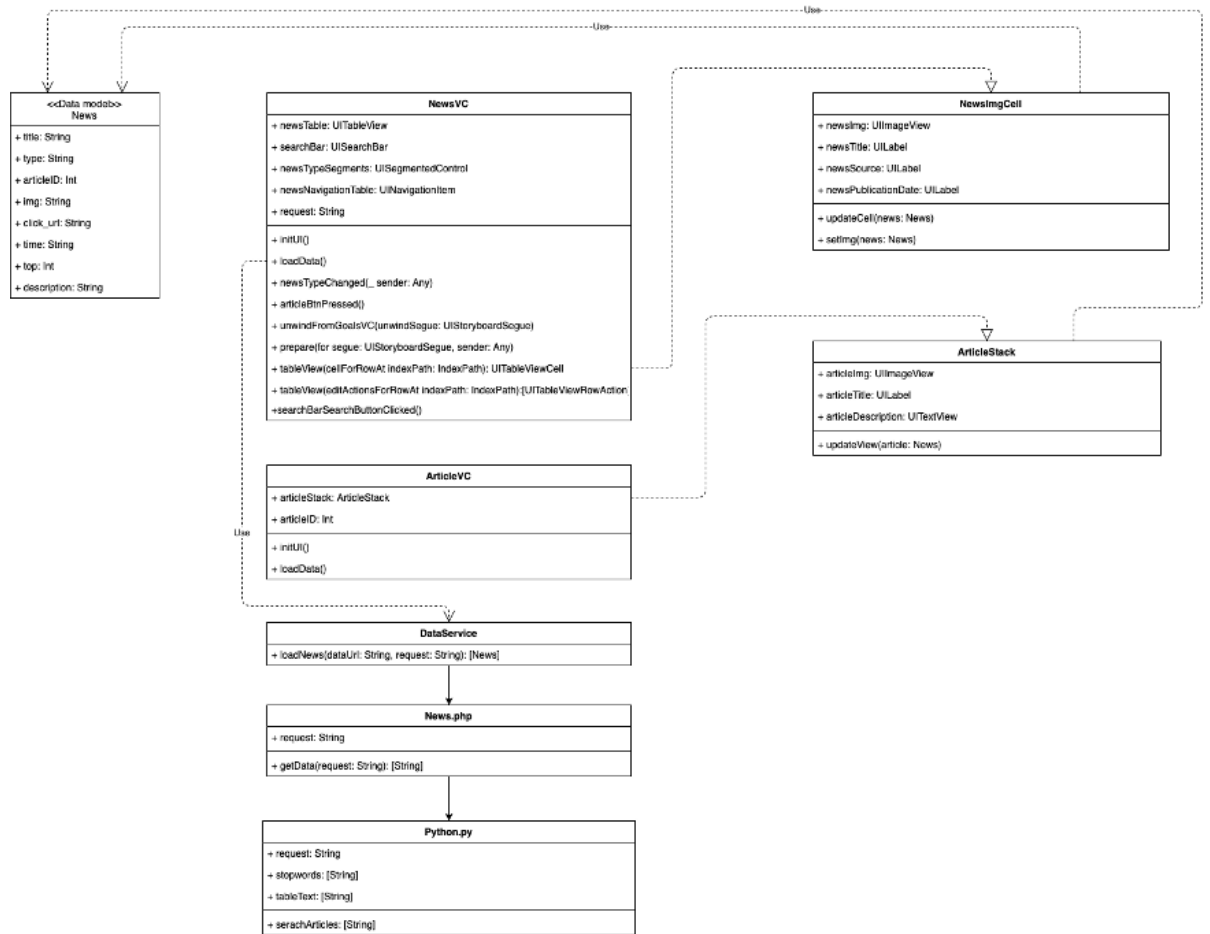


Figure 3.11 - Class diagram

As we can see from the class diagram, the following classes are created and used in this application: NewsVC, ArticleVC, DataService, NewsImgCell, ArticleStack, and the News data model.

3.3 Database Design and Development

An Entity-Relationship (ER) model is a representation of a database in the form of visual diagrams. The ER model visualizes the process that defines a specific subject area.

The ER model is merely a conceptual level of modeling; it does not contain implementation details. For the same ER model, the details of its implementation may vary.

An Entity-Relationship diagram is a graphical representation of entities, attributes, and relationships.

In a database, an entity is any object that can be identified based on the essence of the subject area for which this database is developed. The database developer must be able to correctly identify entities.

For this subject area in the ER model, the following entities and keys can be identified:

- News (articleID);
- Source (sourceID);
- News Level (levelID);
- News Rating (levelID);
- Author (authorID);
- News Type (typeID);
- Image (imgID);

- Like (likeID).

It is necessary to include attributes describing objects in the database:

- News (<articleID>, typeId, authorID, articleTitle, sourceID, subtitle, description, articleTop, publicationDate, levelID);
- Sources (<sourceID>, source);
- Article Level (<levelID>, levelName);
- Article Rating (<ratingID>, articleID, topDate, ratingNumber);
- Authors (<authorID>, authorName);
- News Type (<typeID>, typeName);
- Image (<imgID>, articleID, img);
- Likes (<likeID>, articleID, userName, isLike, likeDate).

Based on the necessary data, we will build the resulting ER diagram of the types of the subject area (Fig. 3.12). The data scheme is shown in Figure 3.13

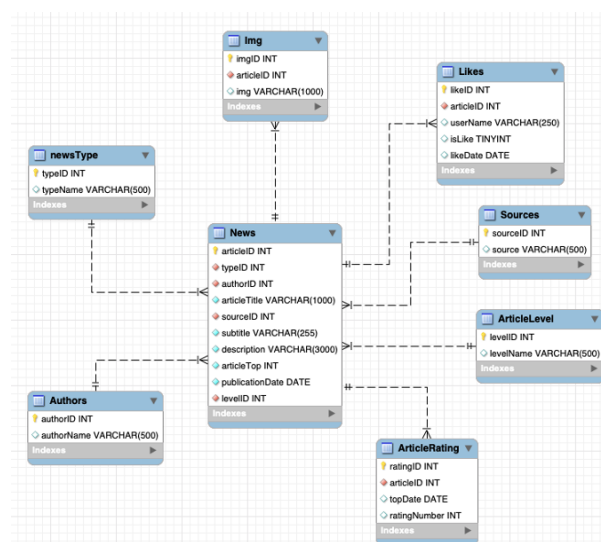


Figure 3.12 - The resulting ER diagram of the subject area types

The data scheme is shown in Figure 3.13.

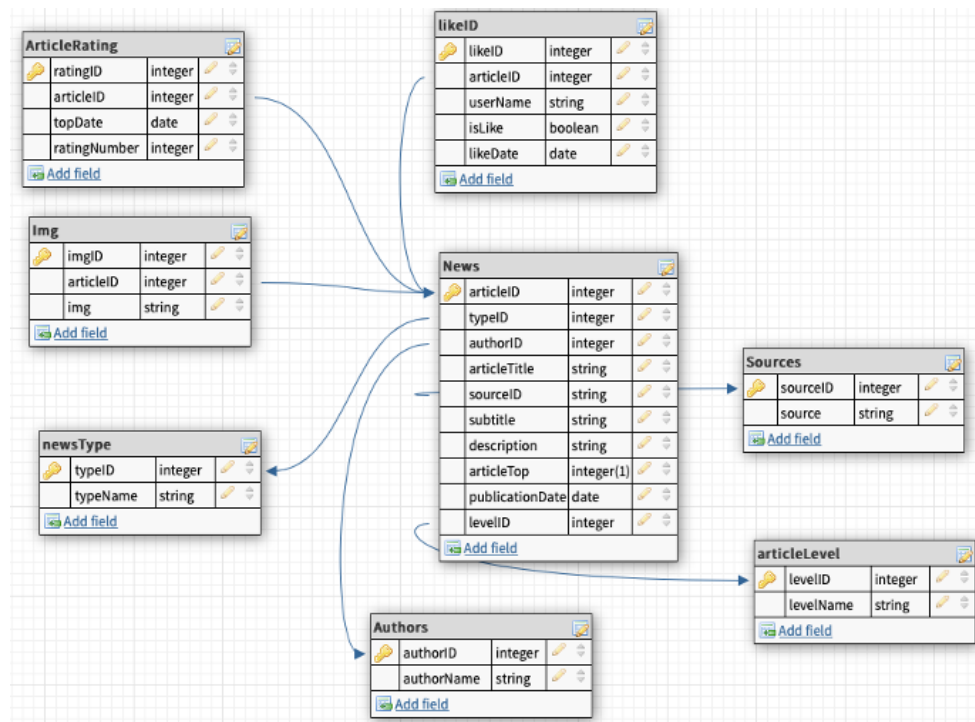


Figure 3.13 - Data schema

To enhance the functionality of the database, normalized relationships were designed within the database. Normalization is a process that eliminates design flaws in a database. The normalization process involves breaking down tables into smaller ones, resulting in a more efficient structure.

The goals of normalization are:

- Reducing data storage volume.
- Improving the efficiency of the database.

The normalization procedure is carried out step by step. Ideally, during normalization, each value should be stored in the database in a single instance, and

this value should not be derived through calculation from other data stored in the database.

In relational databases, the following sequence of normal forms is distinguished:

- First Normal Form (1NF).
- Second Normal Form (2NF).
- Third Normal Form (3NF).
- Boyce-Codd Normal Form (BCNF).

First Normal Form: A relation complies with 1NF when each intersection of every column and every row contains only elementary (indivisible) values of attributes and does not contain repeating groups.

Second Normal Form: A relation is in 2NF if it satisfies the constraints of 1NF, and each descriptive attribute is functionally fully dependent on the primary key (including composite keys).

Third Normal Form: A relation is in 3NF if it satisfies the constraints of 2NF, and all descriptive attributes of the relation are mutually independent and fully dependent on the primary key, meaning each descriptive attribute is not transitively dependent on the key.

The base relation is in Boyce-Codd Normal Form if and only if it is in the third normal form, and not only is any non-key attribute fully functionally dependent on any key attribute, but also any key attribute must be fully functionally dependent on any key attribute.

Normalization is performed following the algorithm and definitions of normal forms. The universal relation R is divided into the following relations:

- R1 <sourceID>, source;
- R2 <levelID>, levelName;
- R3 <ratingID>, articleID, topDate, ratingNumber;
- R4 <authorID>, authorName;
- R5 <typeID>, typeName;
- R6 <imgID>, articleID, img;
- R7 <likeID>, articleID, userName, isLike, likeDate.

Tuples in relations are not repeated, each attribute has only one value, all keys are simple, there are no transitive dependencies, and each relation has a single key. Therefore, it can be concluded that all relations are in Boyce-Codd Normal Form.

3.4 Implementation of the Connection between Forms and the Database

The connection between the interface and the database was realized using a PHP script and the standard MySQL Improved extension (mysqli). The script establishes a connection with the database and processes the received information, returning it to the form in JSON format.

An example of establishing and verifying the connection to MySQL in the PHP script is illustrated in Figure 3.14.

```
// Create connection to database
$con=mysqli_connect("localhost","admin", "1234" ,"gameNews");

// Check connection
if (mysqli_connect_errno())
{
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}
```

Figure 3.14 Establishing a connection to MySQL

After the connection is established, you need to connect the script to the project. In order to connect the script to the project, the URLSession framework built into Swift is used, which allows you to establish a connection to external Internet resources (Figure 3.15).

```
URLSession.shared.dataTask(with: request)
```

Figure 3.15 - connecting a PHP script to a project

The next step is to define the body of the request that the PHP script will accept to execute the corresponding query to the database. For this purpose, the "loadNews" function is created (Fig. 3.16), responsible for the connection between the script and the project. It takes, as parameters, a reference to an external resource with which the connection needs to be established and an array of information to be passed to the script to execute the database query. In case of successful execution, this function

returns the data obtained from the database, structured into an object model according to the Decodable protocol.

If executed successfully, this function returns data received from the database, which is structured into an object model in accordance with the Decodable protocol.

```
func loadNews(dataUrl urlString: String, searchNews: [String : Any], completion: @escaping ([News]) -> Void) {
    guard let url = URL(string: urlString) else { return }

    var request = URLRequest(url: url)
    request.httpMethod = "POST"
    request.addValue("application/x-www-form-urlencoded", forHTTPHeaderField: "Content-Type")
    request.httpBody = searchNews.percentEscaped().data(using: .utf8)

    URLSession.shared.dataTask(with: request) { data, _, err in
        DispatchQueue.main.async {
            if let err = err {
                print("Failed to get data from url:", err)
                return
            }

            guard let data = data else { return }

            do{
                let news = try JSONDecoder().decode([News].self, from: data)
                completion(news)
            }catch let jsonErr{
                print("Failed to decode: ", jsonErr)
            }
        }
    }.resume()
}
```

Figure 3.16 - The "loadNews" function

After establishing communication with the form and the PHP script, we send from the form to the script an array of information about the news that we need to display. In the script, we receive the information sent (Figure 3.17) and, depending on the data received, execute the appropriate request.

```
$request = $con->real_escape_string($_POST['request']);
```

Figure 3.17 - getting data from a form in a PHP script

After receiving the data from the form, the PHP script communicates with the Python script to which it sends the received request and receives the filtered data that matches the request, then converts it to JSON format and sends it to the application.

```

$request = $con->real_escape_string($_POST['request']);
$result = exec("python python1.py ".$request);
$json_array = array();
while ($row = $result->fetch_assoc()) {
    $news_title = $row['title'];
    $news_type = $row['type'];
    $news_articleID = $row['articleID'];
    $news_img = $row['img'];
    $news_click_url = $row['click_url'];
    $news_time = $row['time'];
    $news_top = $row['top'];
    $news_description = $row['description'];

    $news = array('title' => $news_title, 'type' => $news_type, 'articleID' => $news_articleID, 'img' => $news_img,
        'click_url' => $news_click_url, 'time' => $news_time, 'top' => $news_top, 'description' => $news_description);
    array_push($json_array, $news);
}
echo json_encode($json_array);

```

Figure 3.18 - sending and receiving data between PHP and Python scripts

The Python script selects all news items containing the query, converts them into a list, and uses loops to search through all the data, evaluating the query's similarity index to the current data [13,14]. If the similarity index is greater than or equal to 0.1, we consider the data to be similar and continue the check. If the analyzed news contains 1 or more similarities to the query, this news is considered relevant and is converted to the form required for its further conversion to JSON format. To achieve greater filtering accuracy, we can increase the similarity index that will be checked and increase the minimum number of matches required for the news to be considered relevant.

The algorithm of the information search and filtering process is shown in Figure 3.19.

```

request = 'select articleTitle as title, newsType.typeName as type, News.articleID, img.img, Authors.authorName as click_url, publicationDate as time,
stopwords = ['.', ':', ';', '-', '?', '!', ',']
cursor = cnx.cursor()
searchWord = sys.argv[1]
cursor.execute(request)
tableText = cursor.fetchall()
simCount = 0
sim2count = 0
jsonData = []

i = 0
while i < len(tableText):
    j = 0
    mainResult = tableText[i]
    while j < len(tableText[i]):
        try:
            result = tableText[i][j].split()
        except AttributeError:
            sim2count += 1
        for n in result:
            if n not in stopwords:
                similarityIndex = 0
                try:
                    synset1 = wn.synsets(searchWord)[0]
                    synset2 = wn.synsets(n)[0]
                    similarityIndex = int(synset2.path_similarity(synset1))
                except IndexError:
                    sim2count += 1
                except TypeError:
                    sim2count += 1
                if similarityIndex >= 0.1:
                    simCount += 1
            j += 1
    if simCount >= 1:
        row_headers = [x[0] for x in cursor.description]
        jsonData.append(dict(zip(row_headers, mainResult)))
        print(jsonData)
    simCount = 0
    i += 1

```

Figure 3.19 - searching and filtering information that matches the query

A diagram of the information filtering algorithm is provided in Appendix A.

3.5 Conclusion

This section outlines the architecture of the application and discusses its main modules. Forms have been developed for displaying information and interacting with the user. The database has been designed and implemented to store an array of

necessary data. The connections between the forms and the database have been described and implemented. The application follows a Model-View-Controller (MVC) design pattern, with project files grouped according to their purpose. The Extensions folder contains objects that extend program components, while the Services folder contains modules responsible for implementing functionality. The Model folder holds modules reflecting object structure, and the View folder contains reusable view classes. The Controller folder houses main elements of program screens. Forms in the application are used for displaying and inputting information, and can be created in Xcode using control blocks described in Swift programming language. Database design involves an Entity-Relationship (ER) model represented by visual diagrams, where entities, attributes, and relationships are defined. Normalization is performed to improve data storage efficiency by breaking down tables into smaller ones based on normal forms such as 1NF, 2NF, 3NF, and BCNF. Connection between forms and database is established using a PHP script that communicates with MySQL through the MySQL extension.

4 TESTING OF DEVELOPED SOFTWARE

4.1 Choice of Testing Approach

Software testing is a crucial part of the application development cycle. By testing and evaluating the product, the development team can understand if the solution aligns with the specified requirements. Testing improves development quality and prevents errors that may lead to the deterioration or even failure of the application.

There are two main types of testing:

- Automated Testing
- Manual Testing

Automated testing involves using tools to detect errors during the execution of software modules and evaluate product performance. In automated testing, tests can be executed without human intervention, but the effectiveness of such tests is limited by the capabilities of the testing tool. This type of testing is suitable for functions that require simultaneous execution of a large number of conditions or processing a substantial amount of data.

Advantages of automated testing include:

- Early error detection: Issues, such as memory usage problems, can be identified faster through simultaneous execution of multiple tests.

- Reusability: Testers can reuse a defined set of conditions for testing once the code has been modified.
- Reliability: Human errors that often accompany manual testing are minimized with automated testing.
- On-demand test execution: This type of testing is useful for conducting time-consuming tests with a high volume of repetitive work.

Disadvantages of automated testing include:

- Limitations defined by the testing tool.
- Significant time investment in test creation and maintenance.
- Loss of efficiency when testing small functionalities.

Manual testing is a more conventional approach that relies solely on human intervention and is widely used in most companies. During manual testing, a specialist independently performs a test from start to finish without using automation tools.

Advantages of manual testing include:

- Accuracy: Testers can thoroughly test and better assess the final outcome of the program.
- Provides a better understanding of the problem and solution at a conceptual level.
- Greater variability in testing, unrestricted by testing tool capabilities.

Disadvantages of manual testing include:

- Potential for human error and higher testing inaccuracy compared to automated processes.
- Low productivity for large volumes of tests and extensive testing processes.

After evaluating the advantages and disadvantages of both approaches, manual testing was chosen for testing the application's functionalities.

4.2 Application Testing

To verify the functionality of the application, it is necessary to conduct testing of the logical operation of queries and the functioning of forms.

Forms and queries were tested during their development, and the analysis of their operation revealed no deviations. They work correctly, all fields and buttons perform their functions accurately, and data retrieval and display functions operate correctly.

As an example, let's test the process of searching for news based on a keyword and obtaining detailed information about it. It is necessary to establish a connection with the script responsible for data filtering, make a selection of news that match the query, and display the list of found news on the mobile device screen. Upon clicking on a news item, a screen with detailed information about the news should open.

Figure 4.1 shows an example of a completed news search field and the results of searching and filtering news.

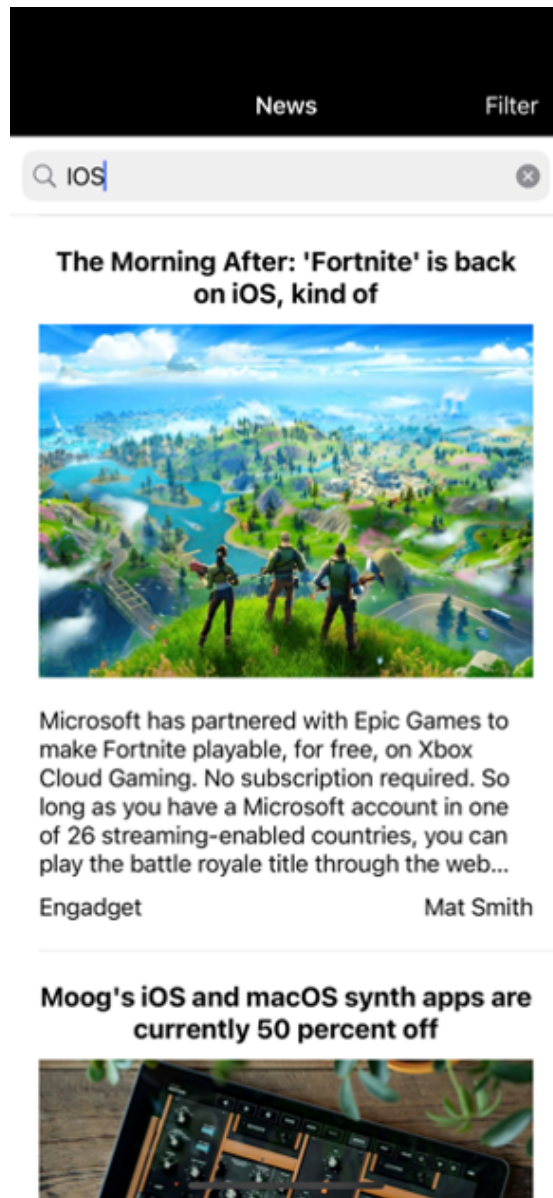


Figure 4.1 - results of filtering news containing the search query

The results of clicking on the found news are shown in Figure 4.2.

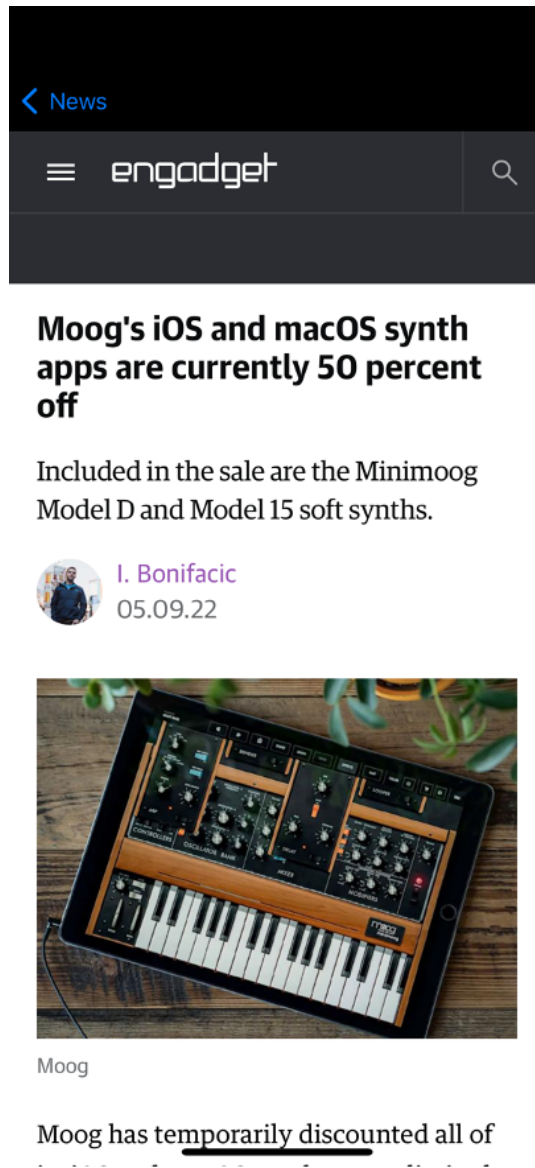


Figure 4.2 - results of clicking on a news item

So, we can conclude that the program works correctly and can be used to solve the tasks.

4.3 Conclusion

Software testing is an essential part of the application development cycle. There are two main types of testing: automated and manual. Automated testing allows for early error detection, reusability, reliability, and on-demand test execution. However, it has limitations defined by the testing tool and requires significant time investment in test creation and maintenance. On the other hand, manual testing offers accuracy, a better understanding of the problem and solution at a conceptual level, and greater variability in testing. It does have disadvantages such as potential for human error and low productivity for large volumes of tests. In this case, manual testing was chosen to evaluate the application's functionalities successfully.

This section has outlined the fundamental types of software testing, including automated and manual testing. After thorough investigation and evaluation of potential options, manual testing was selected, and the primary software modules were examined for proper functionality.

CONCLUSIONS

This master's thesis presents a comprehensive exploration of cutting-edge technologies for data selection and filtration, with a central focus on the Nearest Neighbor method's intricate sequential traversal approach. The study delves into the theoretical underpinnings of this algorithm, highlighting its application in efficiently "visiting" nearest unvisited points, sequentially incorporating them into a defined route.

In tandem with the algorithmic exploration, the research rigorously reviews existing services dedicated to presenting new information to users. An exhaustive analysis of general concepts related to the research object unfolds, accompanied by a detailed consideration of diverse methods for selecting contextual information. The primary objective of these services is to empower users to execute programmed requests to servers, facilitating the retrieval of essential information, which is subsequently presented through an interface designed for convenience and accessibility.

Swift, renowned for its speed, intuitive syntax, and optimization capabilities, is meticulously chosen as the programming language to implement the envisioned mobile application for filtering and searching relevant information. The thesis extends its focus to modern software development features, encompassing diverse methods of information filtration, and delves into the intricacies of software design and implementation. The Agile Scrum methodology is adopted, leveraging its

efficiency and adaptability to accommodate changes in requirements or user feedback.

The evaluation process incorporates a comprehensive set of metrics, including accuracy, precision, recall, and F1 score, to meticulously assess the algorithm's performance in accurately identifying relevant information within news articles. The implementation strategy adheres to a structured architecture based on the Model-View-Controller (MVC) design pattern, with forms seamlessly connected to a database through PHP scripting. Rigorous manual testing ensures the correct functionality of forms and query operations, further fortifying the robustness of the implemented solution. In essence, this master's thesis amalgamates theoretical exploration, technological analysis, and practical implementation, positioning it as a comprehensive contribution to the realm of data selection, filtration technologies, and mobile application development.

REFERENCES

1. Dranchuk O. I. Development of a Mobile Application for Viewing News with the Possibility of Selecting Relevant Information. Vinnytsia, 2023. URL: <https://conferences.vntu.edu.ua/index.php/mn/mn2021/paper/view/13240> (accessed on 14.02.2023).
2. The Role of Mass Media. URL: <https://www.coe.int/uk/web/compass/media> (accessed on 10.01.2023).
3. Mass Media (Media). URL: [https://vue.gov.ua/Засоби_масової_інформації_\(ЗМІ\)](https://vue.gov.ua/Засоби_масової_інформації_(ЗМІ)) (accessed on 10.01.2023).
4. Use of News Aggregators in the Practice of Library Information and Analytical Product Preparation. Kyiv: Sci. work of the Nat. Lib. of Ukraine named after V. I. Vernadsky, 2018. 278 p. URL: <http://nbuviap.gov.ua/images/naukprazi/49.pdf> (accessed on 10.01.2023).
5. Fundamentals of Information Systems: textbook / V. F. Sitnik et al. 2nd ed. Kyiv: KNEU, 2001. 420 p.
6. Intelligent Decision Support Systems: textbook / B. M. Herasymov et al. Kyiv, 2019. 400 p.
7. Gogertchak H. O. Information Systems and Databases: textbook. Kyiv, 2019. 400 p.

8. Permyakov O. Yu., Sbitnev A. I. Information Technologies and Modern Information Warfare. Luhansk: Znannia, 2008. 204 p.
9. Swift Programming Language. URL: <https://mobiz.com.ua/mova-prohramuvannia-swift.html> (accessed on 13.01.2023).
10. Swift Programming Language. URL: [https://ru.qaz.wiki/wiki/Swift_\(programming_language\)](https://ru.qaz.wiki/wiki/Swift_(programming_language)) (accessed on 15.11.2020).
11. Official Apple Swift Website. URL: <https://developer.apple.com/swift/> (accessed on 15.01.2023).
12. MySQL – Free Relational Database Management System. URL: http://baserpz.blogspot.com/2015/09/mysql_3.html (accessed on 16.01.2023).
13. MySQL Workbench – Visual Tool for Database Architects and Developers. URL: <https://www.make-info.com/what-is-mysql-workbench/> (accessed on 16.01.2023).
14. Official Apple Xcode Website. URL: <https://developer.apple.com/xcode/> (accessed on 17.01.2023).
15. Learn. URL: <https://learn.co/lessons/reading-ios-xcode-overview> (accessed on 01.02.2023).
16. Hacking With Swift. URL: <https://www.hackingwithswift.com/read/1/3/designing-our-interface> (accessed on 15.02.2023).

17. Official Apple Xcode-Interface Builder Website. URL:
<https://developer.apple.com/xcode/interface-builder/> (accessed on 23.02.2023).
18. Hacking With Swift. URL:
<https://www.hackingwithswift.com/quick-start/swiftui/swiftui-vs-interface-builder-and-storyboards> (accessed on 05.03.2023).
19. Wikipedia Swift programming. URL:
[https://en.wikipedia.org/wiki/Swift_\(programming_language\)](https://en.wikipedia.org/wiki/Swift_(programming_language)) (accessed on 19.04.2023).
20. SmartInnovations – URL:
<https://www.smartsight.in/technology/why-mobile-applications-are-important-for-businesses/> (accessed on 15.07.2023).
21. Techahead – URL:
<https://www.techaheadcorp.com/blog/importance-of-a-mobile-app-for-the-construction-business-and-its-productivity/> (accessed on 17.07.2023).
22. Mission20Zero – URL:
<https://m20zero.com/how-mobile-apps-can-help-optimize-your-business-prospects/> (accessed on 20.07.2023).
23. Postindustria – URL:
<https://postindustria.com/apple-core-ml-leveraging-the-power-of-machine-learning-for-mobile/> (accessed on 22.07.2023).

24. Mission20Zero – URL:
<https://m20zero.com/how-mobile-apps-can-help-optimize-your-business-prospects/> (accessed on 25.07.2023).
25. Infostretch – URL:
<https://www.infostretch.com/blog/a-primer-to-mobile-operating-systems/> (accessed on 28.07.2023).
26. Clarion Technologies – URL:
<https://www.clariontech.com/blog/top-mobile-app-development-frameworks-in-2019> (accessed on 01.08.2023).
27. Official Google Play Market Website – URL:
https://play.google.com/store/apps/details?id=com.fivesysdev.mathy&hl=en_US&gl=US (accessed on 03.08.2023).
28. Official Apple App Store Website – URL: <https://www.apple.com/ua/app-store/> (accessed on 06.08.2023).
29. McWherter, J., Gowell, S. Professional Mobile Application Development. 2012. URL: Not available (accessed on 08.08.2023).
30. Adamson, C., Clayton, J. iOS 10 SDK Development: Creating iPhone and iPad Apps with Swift. 2017. URL: Not available (accessed on 11.08.2023).
31. Jagannath S., Debasish D., Lov K., Aneesh K. Application Dev 18th ICDCIT 2022. URL: Not available (accessed on 13.08.2023).

32. Tamai S. Technological and business Fundamentals for mobile app dev. Springer Nature Switzerland AG 2022. URL: Not available (accessed on 16.08.2023).
33. InfoSystem – URL: <https://www.hyperlinkinfosystem.com/blog/the-relevance-of-mobile-app-development-to-the-modern-world> (accessed on 18.08.2023).
34. Armenski, G., Gusev, M. Architecture of Modern e-learning Systems. The 6th International Conference for Informatics and Information Technology, 2008. URL: <http://ciit.finki.ukim.mk/data/papers/6CiiT/6CiiT-09.pdf> (accessed on 21.08.2023).
35. Wikipedia. Swift programming. URL: [https://en.wikipedia.org/wiki/Swift_\(programming_language\)](https://en.wikipedia.org/wiki/Swift_(programming_language)) (accessed on 20.05.2021).
36. Blinco, K., Mason, J., McLean, N., Wilson, S. (2004). Trends and issues in e-learning infrastructure development. A White Paper for alt-i-lab 2004 Prepared on behalf of DEST (Australia) and JISC-CETIS (UK).
37. Yiyi, S. Practical Application Development. Apress, 2019. URL: Not available (accessed on 26.08.2023).
38. Martin, R. C. (2017). Clean Architecture. Prentice Hall. 432 p. URL: Not available (accessed on 28.08.2023).

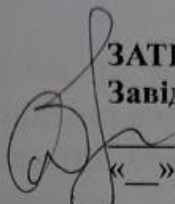
39. Teacher Training and Professional Development: Concepts, Methodologies, Tools and Application. Management Association, Information Resources. URL: Not available (accessed on 31.08.2023).
40. Griffiths, D. Head First Android Development: A Brain-Friendly Guide. 1st Edition. O'Reilly Media, 2015. 734 p. URL: Not available (accessed on 04.09.2023).
41. Murphy, M. The Busy Coder's Guide to Advanced Android Development. Murphy. CommonsWare, LLC, 2011. 630 p. URL: Not available (accessed on 07.09.2023).
42. Darwin, F. Android Cookbook: Problems and Solutions for Android Developers. 2nd Edition. O'Reilly Media, 2017. 772 p.
43. Schildt, H. Java: A Beginner's Guide, Eighth Edition. 8th Edition. Schildt. McGraw-Hill Education, 2018. 720 p. URL: Not available (accessed on 09.09.2023).
44. McCalister, J. Quickly Creating, Designing and Utilizing Mobile Apps for Your Business. 2nd Edition. CreateSpace Independent Publishing Platform, 2014. 170 p. URL: Not available (accessed on 12.09.2023).
45. Lee, V. Mobile Applications: Architecture, Design, and Development. Lee. Prentice Hall, 2004. 368 p. URL: Not available (accessed on 14.09.2023).
46. Iversen, J., Eierman, M. Learning Mobile App Development: A Hands-on Guide to Building Apps with iOS and Android. 1st Edition. Addison-Wesley Professional, 2013. 464 p. URL: Not available (accessed on 17.09.2023).

47. Vasic, M. Mastering Android Development with Kotlin: Deep dive into the world of Android to create robust applications with Kotlin. Packt Publishing, 2017. 378 p. URL: Not available (accessed on 20.09.2023).
48. Augmented Reality Applications in News Reporting. URL: https://www.researchgate.net/publication/340848497_Augmented_reality_as_news (accessed on 20.06.2023).
49. Digital Natives: Media Consumption Patterns of the New Generation. URL: <https://savanta.com/knowledge-centre/view/gen-z-unplugged-media-consumption-trends-of-the-digital-natives/> (accessed on 23.06.2023).
50. Hyperlocal News Websites: Reshaping Community Journalism. URL: <https://orca.cardiff.ac.uk/id/eprint/100797/1/2017harted.pdf> (accessed on 12.06.2023).
51. Impact of Artificial Intelligence on Content Curation. URL: <https://aicontentfy.com/en/blog/role-of-ai-in-content-curation-and-organization> (accessed on 28.06.2023).
52. Virtual Influencers: Redefining Celebrity in the Digital Age. URL: <https://www.linkedin.com/pulse/rise-virtual-influencers-redefining-social-media/> (accessed on 01.07.2023).

ATTACHMENTS

Додаток А
(обов'язковий)

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

 **ЗАТВЕРДЖУЮ**
Завідувач кафедри АІТ
д.т.н., проф. Олег БІСКАЛО
« » _____ 2023 р.

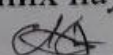
ТЕХНІЧНЕ ЗАВДАННЯ

на магістерську кваліфікаційну роботу

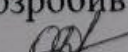
**РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ ДЛЯ ФІЛЬТРАЦІЇ ТА
ПОШУКУ РЕЛЕВАНТНОЇ ІНФОРМАЦІЇ**

08-31.МКР.002.00.000 ТЗ

Науковий керівник: кандидат
технічних наук, доцент АІТ

 Ярослав КУЛИК
"08" _____ 2023 р.

Розробив студент гр. ЗАКІТ-22м

 Олександр ДРАНЧУК
"13" _____ 2023 р.

1. Назва та галузь застосування.

Розробка мобільного додатку для фільтрації та пошуку релевантної інформації. Інформаційні системи та технології.

2. Підстава для проведення робіт.

Підставою для виконання роботи є наказ № 247 ВНТУ від 18.09.2023 р. та індивідуальне завдання на МКР, затверджене протоколом № 1 засідання кафедри АІТ від 30.08.2023 р.

3. Мета та призначення роботи.

Метою магістерської роботи є розробка та впровадження мобільного додатку для фільтрації та пошуку релевантної інформації в новинах для підвищення швидкості та якості знайденої інформації. Основним завданням є підвищення ефективності та точності знаходження інформації, оцінюючи точність, релевантність та ефективність алгоритму.

4. Джерела розробки:

1) Кветний, Р.М., Бевз, О.М., Бісікало, О.В. Методичні вказівки до виконання бакалаврських дипломних робіт (проектів) для студентів спеціальностей 126 - "Інформаційні системи та технології", 151 - "Автоматизація та комп'ютерно-інтегровані технології". Вінниця: ВНТУ, 2019. 26 с.

2) Лі, В. Мобільні додатки: Архітектура, дизайн та розробка. Prentice Hall, 2004. 368 с.

3) Іверсен, Й., Ейерман, М. Вчимося розробляти мобільні додатки: Практичний посібник зі створення додатків для iOS та Android, 1-е видання. Addison-Wesley Professional, 2013. 464 с.

5. Показники призначення

Основні технічні вимоги та мінімальні системні вимоги до програми:

Мобільні пристрої з операційною системою iOS версії 9.0 і вище.

Досягнуто показника релевантності в розробленому додатку на рівні 90%. Досягнуто показника точності, що перевищує 85%, що свідчить про точність розробленого алгоритму. Показник ефективності 92%, що підкреслює точність алгоритму у визначенні релевантної інформації. Коефіцієнт відповідності 88%, що відображає здатність алгоритму охоплювати значну частину актуальних статей.

Вихідні дані для проведення робіт:

Початкові параметри магістерської роботи передбачають розробку мобільного додатку для спрощеного пошуку інформації, включаючи функції, орієнтовані на користувача, такі як інтуїтивно зрозуміла фільтрація та персоналізовані алгоритми пошуку. Доопрацювання повинні включати комплексний план валідації, інтегруючи показники точності на основі відгуків користувачів та реальних сценаріїв. Етичні міркування щодо конфіденційності

та прозорості даних також повинні бути чітко враховані на початкових етапах розробки.

Методи дослідження:

Основну увагу приділено методу Найближчого Сусіда, що починається в довільній точці та поступово «відвідує» кожну найближчу точку, яка ще не була «відвідана». Пункти обходу плану послідовно включаються до маршруту.

Результати роботи програми

Результати програми включають повнофункціональний мобільний додаток з орієнтованими на користувача функціями для ефективного пошуку інформації, перевірених за допомогою таких показників, як точність, достовірність і пригадування. Крім того, результати включають технологічний аналіз, відгуки користувачів та документацію, що сприяють прогресу в розробці мобільних додатків для спрощеного доступу до інформації.

6. Стадії розробки:

а) Вибір, узгодження та затвердження теми МКР.	18.09.2023 - 25.09.2023
б) Аналіз літературних джерел.	
Попередня розробка основних розділів.	26.09.2023 - 17.10.2023
в) Розробка технічного завдання.	18.10.2023 - 24.10.2023
г) Вибір технології для реалізації системи.	25.10.2023 - 31.10.2023
д) Розробка програмного забезпечення.	01.11.2023 - 20.11.2023
е) Тестування системи.	21.11.2023 - 24.11.2023
ж) Нормоконтроль.	27.11.2023 - 01.12.2023
з) Захист МКР.	18.12.2023

7. Порядок контролю та приймання

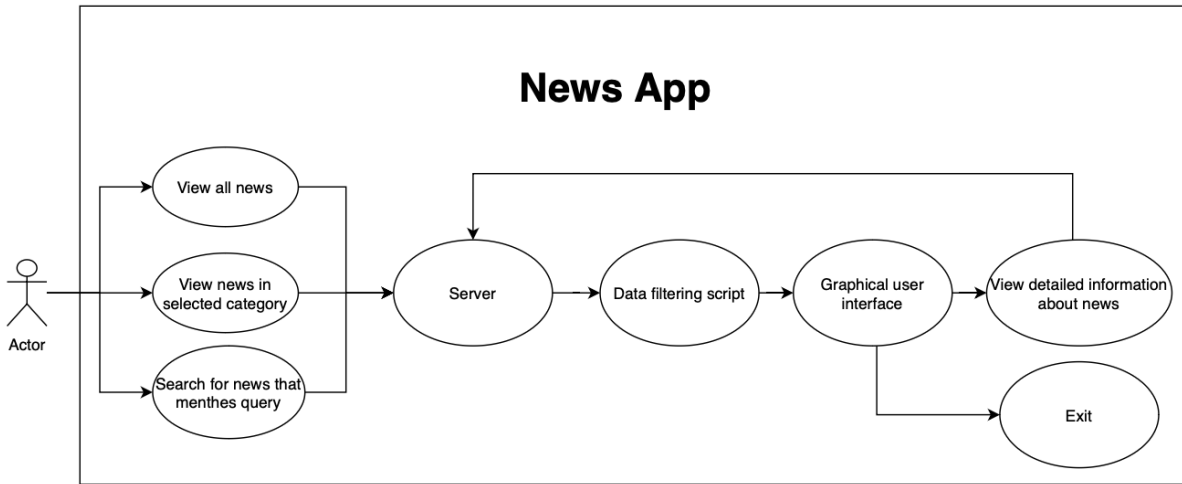
Рубіжний контроль провести до	01.12.2023
Попередній захист МКР провести	05.12.2023
Захист МКР провести до	18.12.2023

Розробив студент гр. ЗАІТ-22м _____ Олександр ДРАНЧУК

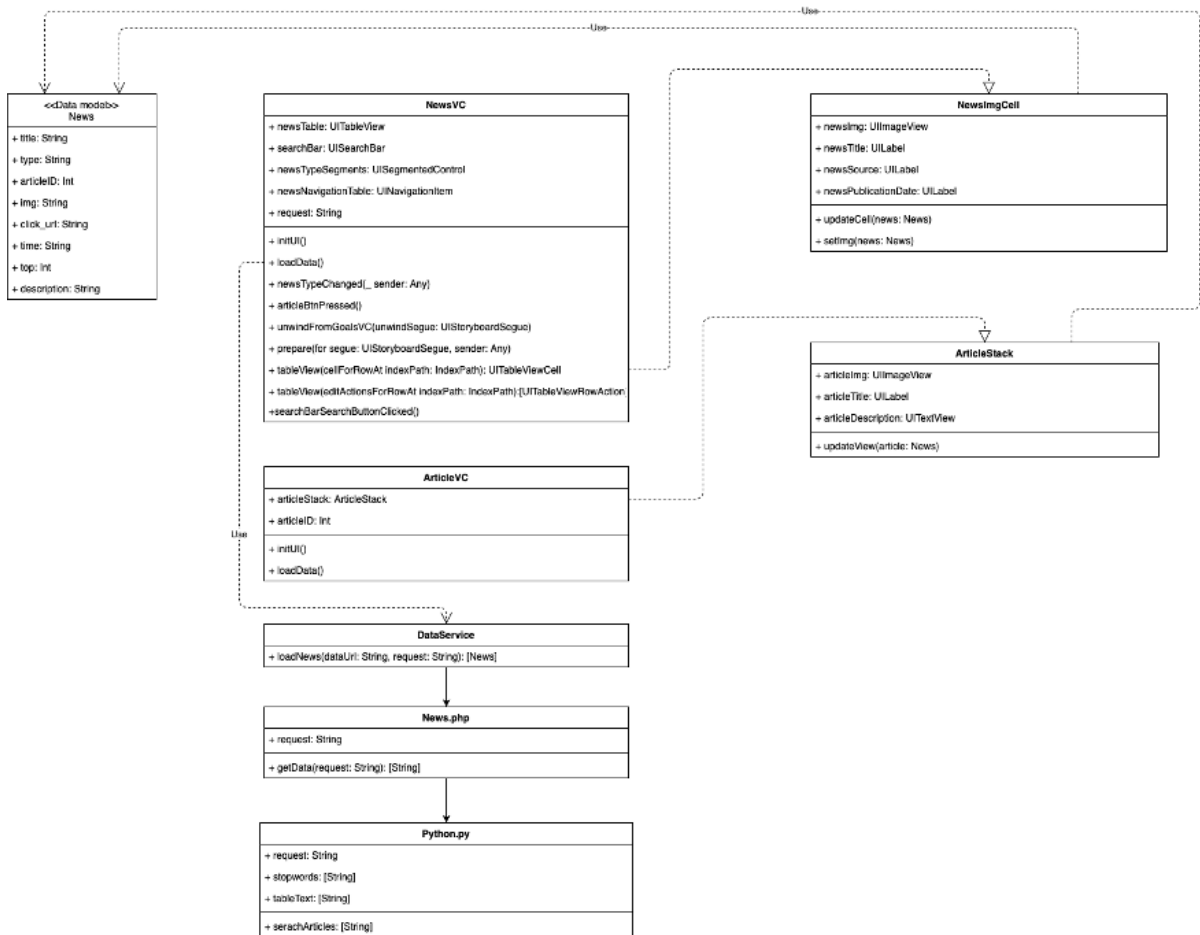
Attachment B
(required)

ILLUSTRATION PART

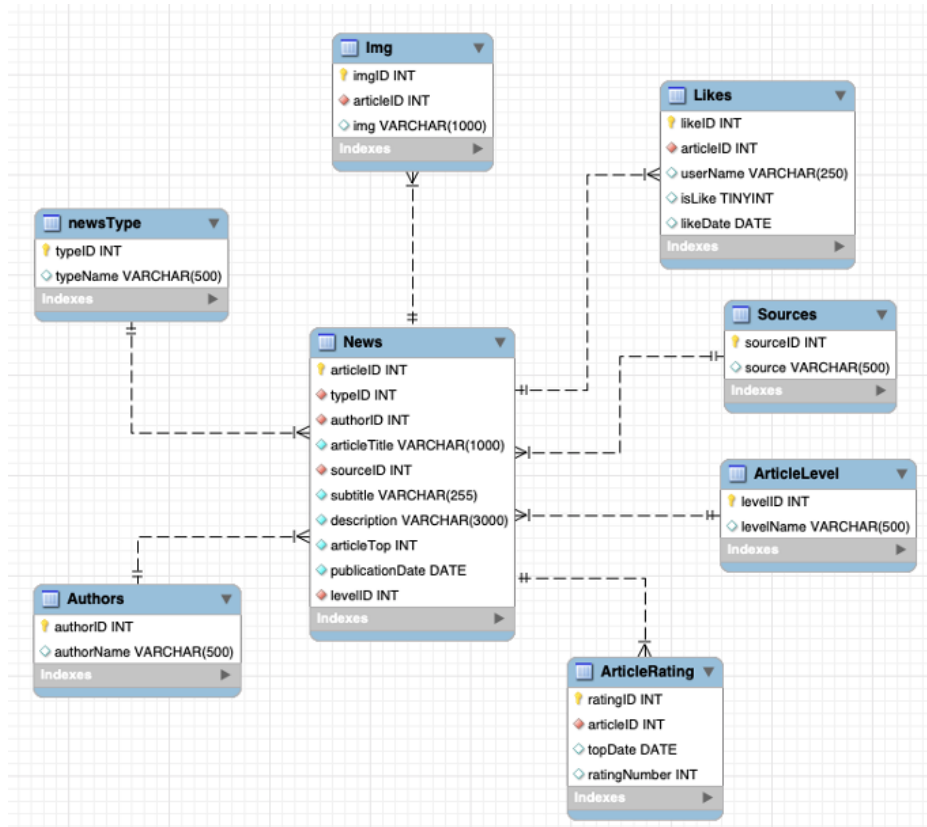
**Mobile application development for filtering and searching of relevant
information**



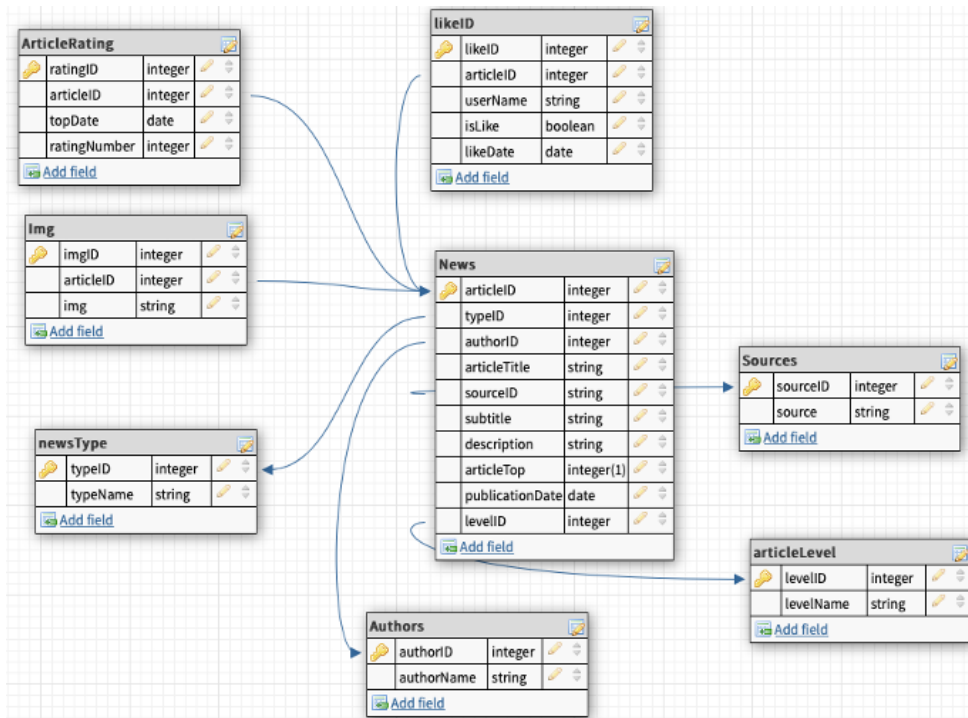
Attachment B.1 - Use-case diagram



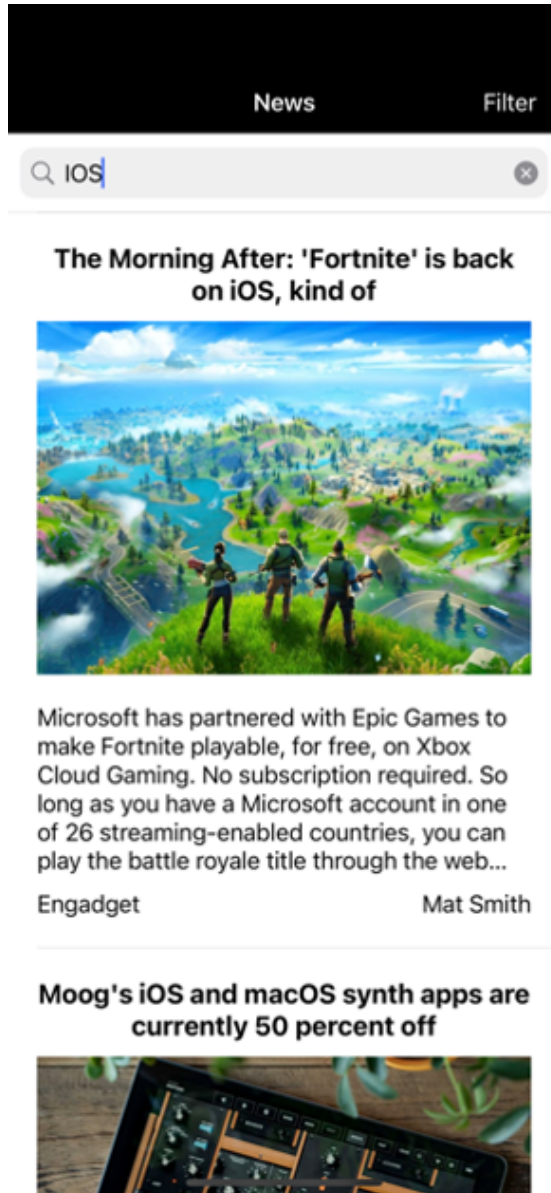
Attachment B.2 - Class diagram



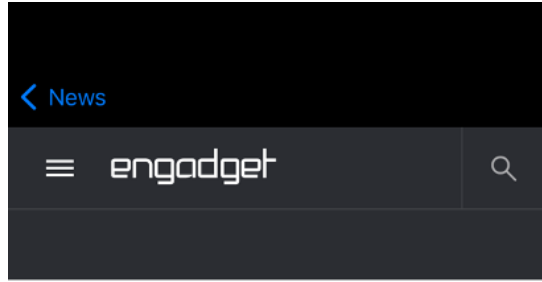
Attachment B.3 - The resulting ER diagram of the subject area types



Attachment B.4 - Data schema



Attachment B.5 - results of filtering news containing the search query



Moog's iOS and macOS synth apps are currently 50 percent off

Included in the sale are the Minimooog Model D and Model 15 soft synths.



I. Bonifacic

05.09.22



Moog

Moog has temporarily discounted all of

Attachment B.6 - results of clicking on a news item

Attachment C

Program listing

```

import json
import mysql.connector
import sys
from nltk.corpus import wordnet as wn
import nltk

cnx = mysql.connector.connect(user='root', password='root', host='127.0.0.1',
database='gameNews', auth_plugin='mysql_native_password')
request = 'select articleTitle as title, newsType.typeName as type, News.articleID,
Img.img, Authors.authorName as click_url, publicationDate as time, description,
articleTop as top from News inner join newsType, Img, Authors where News.articleID
= Img.articleID and News.authorID = Authors.authorID and News.typeID =
newsType.typeID;'
stopwords = ['.', ':', ';', '-', '?', '!', ',']
cursor = cnx.cursor()
searchWord = sys.argv[1]
cursor.execute(request)
tableText = cursor.fetchall()
simCount = 0
sim2count = 0
jsonData = []

i = 0
while i < len(tableText):
    j = 0
    mainResult = tableText[i]
    while j < len(tableText[i]):
        try:
            result = tableText[i][j].split()
        except AttributeError:
            sim2count += 1
        for n in result:
            if n not in stopwords:
                similarityIndex = 0
                try:
                    synset1 = wn.synsets(searchWord)[0]
                    synset2 = wn.synsets(n)[0]
                    similarityIndex = int(synset2.path_similarity(synset1))
                except IndexError:
                    sim2count += 1
                except TypeError:
                    sim2count += 1
                if similarityIndex >= 0.1:
                    simCount += 1
            j += 1
    if simCount >= 1:
        row_headers = [x[0] for x in cursor.description]
        jsonData.append(dict(zip(row_headers, mainResult)))
    simCount = 0
    i += 1
print(jsonData)

```

**Attachment D
(required)
Protocol of verification**

**ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ**

Назва роботи: Розробка мобільного додатку фільтрації та пошуку релевантної інформації.
 Тип роботи: магістерська кваліфікаційна робота
 Підрозділ: Кафедра автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації.

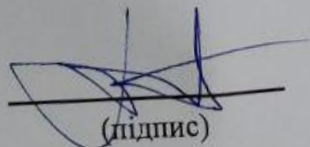
Показники звіту подібності Plagiat.pl (StrikePlagiarism)

Оригінальність 96.5% Схожість 3.5%

Аналіз звіту подібностей (відмітити потрібне):

- Запозичення, виявлені у роботі, оформленні коректно і не містять ознак плагіату.
- Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора.
Роботу направити на розгляд експертної комісії кафедри.
- Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

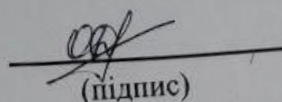
Особа, відповідальна за перевірку


(підпис)

Роман МАСЛІЙ
(Ім'я, ПРІЗВИЩЕ)

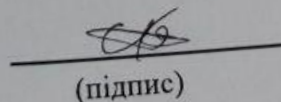
Ознайомлені з повним звітом подібності, який був згенерований системою Plagiat.pl (StrikePlagiarism) щодо роботи.

Автор роботи


(підпис)

Олександр ДРАНЧУК
(Ім'я, ПРІЗВИЩЕ)

Керівник роботи


(підпис)

Ярослав КУЛИК
(Ім'я, ПРІЗВИЩЕ)