DOI: 10.32347/2412-9933.2024.58.139-145
UDC 004.27

Poliakov Mykyta
Postgraduate student Department of Information Technology Design and Applied Mathematics,
https://orcid.org/0000-0002-5061-4866
Kyiv National University of Construction and Architecture
Yeremenko Bohdan
PhD, Associate Professor Department of Management Technologies,
https://orcid.org/0000-0002-3734-0813
Taras Shevchenko National University of Kyiv

INFO-COMMUNICATION SYSTEM OF THE ELECTRONIC ESTIMATION OF ADOLESCENTS' SPECIAL ABILITIES

Abstract. *The focus of this study is aimed at the career guidance of adolescents. It is shown that currently there are various methods and tools for the decision support of the youth regarding the future profession choice, but the vast majority of them are either long-term or accompanied by the risks of a negligent attitude to the process of career guidance on the adolescents' side due to misunderstanding or loss of interest in the process itself. That is why this work is a continuation of research aimed at the development of info-communication systems that use game technologies in order to attract the attention of an adolescent when performing professionally oriented tasks for self-assessment of their own special abilities. In this work, the architecture of the "Electronic Estimation of Adolescents' Special Abilities" system is built based on the available hardware and software, and the technologies that can be used in the construction of an info-communication system are considered. The main idea is to build a micro service architecture that has the potential to expand and be able to store copies of data in cloud storage. It is suggested to use such technologies as: Kafka, Celery, Redis, PostgreSQL, AWS S3 Storage, FastAPI. Examples of determining the final conclusion of the system with a recommendation on the profession are described. An example of system operation from the user's side using a game is shown.*

Keywords: background tasks; data backups; framework; message queue system; microservices; system architecture

Introduction

Career guidance is an essential component of school and university-level education when trying to ensure that adolescents effectively use the skills they gained throughout their studying in the real jobs. With the rapidly evolving and increasingly globalised job market, demands are also being placed on the nature of this guidance and its need to support the development of a wide range of career skills and capabilities [1].

There are many approaches on how to help young people with the decision of their future profession across different continents and countries. This article is also dedicated to solving the current problem of professional orientation of teenagers.

One of the most used tools to give career guidance support is the test.

In Europe one of the most popular tools for career guidance is the "Magellano" complex. It does not require software installation and covers a wide range of specialties but which is adapted to European universities only [2; 3].

There's a career guidance method in Japan that helps adolescents to consciously choose their professional path. This technique is based on the Fukuyama test [2]. The test also reflects the situation on the labour market.

The SAT Reasoning Test is a standardised test in the United States that helps universities evaluate applicants' academic knowledge and analytical capabilities. The test has two major sections: the evidence-based reading and writing section and Math [4].

In Ukraine the popular service called Diia created a career-guidance test. The test suggests how to make decisions and perceive the world around the person [5].

The test includes 94 questions and requires to be completed in 30 minutes. The test suggests the person to select an answer to a question without hesitation for too long, that is, to act on the first instinct and to better understand which type of activity brings the most pleasure. The big advantage of the test is that the adolescent can complete it at home [5].

The test can be performed rather quickly, but the conclusion precision can be very low. Sometimes the test can be taken at home but the adolescent can be not fully

involved and easily distracted. In addition, not understanding the question may lead to frustration.

The test can take a couple of days but can lead to a higher stress connected with the stakes of not entering college. It can take years, like in the case of the F-test but requires involvement of psychologists and teachers that help with the career guidance. However, there are various game applications that provide the opportunity to teach adolescents the basics of the profession in a game form. These games might sustain a higher level of interest of a young individual as the person does activities and achieves visible progress in the game.

Thus, the development of intelligent info-communication systems, which are able to provide the adolescent with automatic career guidance support, that can be taken in a comfortable environment, with a small involvement of people resources and be performed quickly with the high interest of the adolescent becomes very important.

Goal and Tasks of the Publication

The goal of this study is to analyse existing tools to develop the Electronic Estimation of Adolescents' Special Abilities info-communication system and define its architecture.

To achieve the goal, it was necessary to solve the following issues:

1. Design an architectural schema of the Electronic Estimation of Adolescents' Special Abilities info-communication system;
2. Choose development tools;
3. Define the user story of the application.

Architectural Design

Fig. 1 is proposed the generalised architecture of the info-communication system of the Electronic Estimation of Adolescents' Special Abilities.

User's interaction with the system is happening with the help of the Backend service that contains endpoints for frontend needs.

Database (DB) Application Programming Interface (API) is the service that maintains the connection to the database. It is important to have only one data source point. It is possible to have connections to DB from different services but the main priority is the safety of data. If one of the services stops working in the middle of interaction with data and at the same time the other service will rely on the same data, unexpected behaviour can occur.

The other possible issue is unstructured data operations. This problem is also solved by the selected approach because inside the DB API service it can be strictly defined what operations are desirable and which are not. Queries to the database can be regulated by the logic written inside the service.

The users should be notified if there are ready results, generated in the Neuro-Fuzzy Inference System. To achieve that the system requires a Cache Storage, where data can be saved before returning to the Frontend (FE), and can be retrieved in case of system failure.

The system has a mechanism to save backups of the data to the cloud storage. DB API is able to create background tasks to save database dumps, so later the data can be restored or needs to be transferred to another place.

Message queue system ensures that the information between the services is shared and won't be lost if any service stops its work.

The other way of communication between the services is the direct API calls. Having this approach the service that makes a call will receive an exception or any other message can be shown to the user saying that something is wrong.

Analysis of existing development tools

Programming languages

Starting with the programming language it was decided to use Python because it is the interpreted programming language that has a large number of libraries that simplify the development, training, and usage of machine learning models.

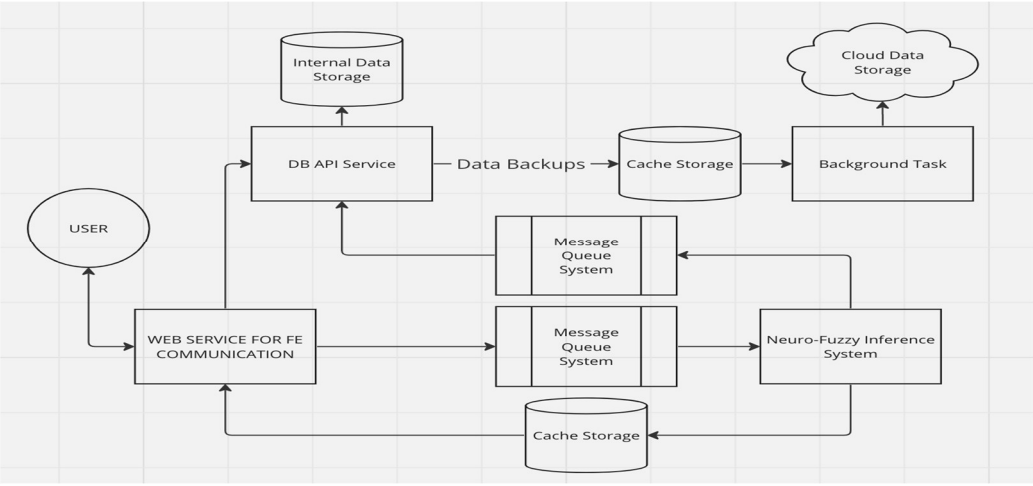


Figure 1 – The generalised architecture of the Electronic Estimation of Adolescents' Special Abilities info-communication system

Architecture model

A monolithic architecture is a traditional approach to designing software where an entire application is built as a single, indivisible unit [6].

On one hand it gives us the opportunity to develop faster and simplifies the implementation because the codebase is stored in one place. But on the other hand it makes the application hardly scalable and won't give us the opportunity to handle a large volume of traffic.

In a microservices architecture, an application is built as a collection of small, independent services, each representing a specific business capability. These services are loosely coupled and communicate with each other over a network, often using lightweight protocols like Hypertext Transfer Protocol (HTTP) or messaging queues [6].

This approach allows us to be very scalable as each service can be scaled on demand and doesn't affect other services. But a noticeable disadvantage is the complexity of the solution because communication between microservices must be robust and this approach forces us to monitor many components in the system, which might be a challenging task.

Considering two options it was decided to proceed with microservice architecture. It can be very hard to maintain but it gives us a very big plus in scalability.

Web frameworks

Python web frameworks support the Representational State Transfer (REST) API, which is an API that conforms to the constraints of REST architectural style.

REST is a set of architectural constraints, not a protocol or a standard. [7].

This means that the communication in the microservice architecture via HTTP, the common protocol used for communication between web servers and clients, is handled by selecting the web framework.

Three Python web frameworks were considered as a possible choice: Django [8], Flask [9] or FastApi [9].

Django's main advantages are speed, scalability, documentation and security. Cons of Django are: not good for simpler projects and lower performance [8].

Advantages of Flask are lightness and flexibility. Disadvantages of Flask are lack of built-in features and lack of standardisation.

FastAPI is a modern framework based on asynchronous programming, known for its excellent performance and low latency. It supports the use of type annotations to enhance code readability and maintainability. Also FastAPI excels in rapid development, making it suitable for building prototypes, Proof of Concept, and applications with quick iterations.

On the other hand, FastAPI is relatively new and may lack mature solutions and community support in certain areas. Also because of the asynchronous nature it may be harder to learn how to use this framework.

So considering 3 options it's decided to use FastAPI. Django was dropped because it's good for building large and complex web applications rather than small microservices. And the advantage is given to the FastAPI over the Flask because of the asynchronous nature which will give us a better performance.

Message queue system

The other communication method in the microservice architecture is messages. The candidates for the message queue system are: Kafka and RabbitMQ.

RabbitMQ has minimal guarantees regarding ordering messages within a stream which may be a crucial disadvantage for the system. On the other hand, Kafka provides message ordering thanks to its partitioning.

The other feature that is present in Kafka but not in RabbitMQ is delivery guarantees. In a partition, Kafka guarantees that the whole batch of messages either fails or passes, when RabbitMQ doesn't guarantee atomicity, even in relation to transactions involving a single queue [10]. Moreover, such characteristics of Kafka like High-throughput, Fault-Tolerance, Durability and Scalability [11] made the choice of Kafka pretty clear for the system.

Communication with client (Frontend)

The communication with the Frontend side of the application could be done using HTTP connection. But this approach won't allow us to build the communication between Frontend and Backend in both ways. With HTTP Frontend sends requests to Backend and the latter responds to them but Frontend will never know if some work is completed on the Backend side.

So it's decided to additionally use websocket connection - technology that makes it possible to open a two-way interactive communication session between the user's browser and a server [12]. However, this approach is harder to maintain because it requires the services to be tolerant to the connection loss.

To solve this problem, Redis topics are used.

Redis is a key-value database which is suitable when the application requires handling a large volume of small and continuous reads and writes and also supports Publish/Subscribe messaging paradigm [13, 14]. Backend pushes friendly messages to the Redis topics and if the websocket connection is closed, messages could still be retrieved from the database and be sent to the Frontend when the connection is open again.

Background tasks

The data backup process can be solved by using asynchronous tasks.

FastAPI provides built-in background task support [15].

The other considered option is Celery.

Celery is an open-source distributed task queue framework written in Python. It's designed to manage and distribute tasks (often time-consuming or resource-intensive) across multiple worker processes or machines.

This is particularly useful for handling asynchronous tasks in web applications, batch processing, and other scenarios where there's a need to offload tasks from the main application. This tool's attributes ideally fit our application. Moreover it has next advantages: Scalability, Task Priority and Scheduling, Retry Mechanism, Result Tracking.

As a disadvantage Celery relies on a message broker for communication, which introduces an additional dependency to a component that needs to be set up and maintained. Moreover if the message broker fails, the processing of tasks can be affected which leads to a decrease of the application reliability.

Following Celery documentation, it supports Redis, which is already used in the application so we decided to reuse the key-value database here as well, as a message broker that sends tasks to the worker (backend). In addition Redis is supported by Celery as a backend – results storage of the completed task [16].

Backup data storage

Several options were considered as storage for the backup data. There are 2 storage types that were considered for storing the backup data: Physical storage (the data is saved on the hard drives) and Cloud storage (the data is saved on the web).

There can be issues with physical storage like where to keep the hard drives or how to transfer them, so it's decided to use cloud storage as a backup place.

Among many available options the decision was made between the next 3: Amazon Simple Storage Service (AWS S3), Azure Blob Storage and Google Cloud Storage [17].

AWS S3 is a highly scalable and durable cloud object storage. It offers secure storage for various data types, including images, videos, documents, backups, and application data.

Here are the key features and advantages that S3 provides: Performance, Flexible storage classes, Availability, Scalability.

Potential limitations of S3: Limited direct access, The complexity of Bucket Policies, Data transfer costs, Limited performance for high-frequency small object operations.

Google Cloud Storage (GCS) is a cloud object storage service by Google Cloud Platform. It was launched in 2010 and offers an affordable solution for storing and retrieving data in the cloud.

The platform also stores data as objects in buckets. These buckets can be assigned to four storage classes – Standard, Coldline, Nearline, and Archive.

GCS allows users to store and access unstructured data in a highly available and globally distributed manner. Key features and strengths of GCS: scalability, multi-regional and regional storage, data lifecycle management, Access control, Low latency.

Potential limitations of GCS are the following: No native indexing or search, Limited availability zones, Pricing complexity.

Azure Blob Storage is a cloud object storage service by Microsoft Azure. It was first launched in 2010 and has evolved to become a vital component of the Azure cloud ecosystem.

The fundamental data storage unit in Azure Blob Storage is a blob (binary large object). Blobs can store various types of unstructured data. Each blob is identified by a unique URL comprising the storage account name, container name, and blob name. Azure Blob Storage groups related blobs into logical units called Containers. Containers are analogous to folders in a file system.

Key features and strengths of Azure Blob Storage: multiple blob types, unlimited scalability.

Azure Blob Storage has certain disadvantages that users should consider.

Potential limitations of Azure Blob Storage: Data retrieval latency, Transaction and transfer costs.

It is decided to use AWS S3 Storage as a cloud storage for the backup data since it is highly available and scalable storage.

Application data storage

The next big step is to determine which type of the database to use: relational or non-relational [18].

Relational databases allow building relationships between data points or in other words tables. To do operations with data this type of databases use Structured Query Language (SQL), which makes it easy to query and update data. Relational databases allow to create indexes for the columns in tables to do the search quicker but as data grows they can take up a lot of space. Also the database structure should be defined very well from the very beginning to avoid making changes to the DB schema as the updates are time-consuming and complicated.

Non-relational databases do not require the data to be confined to a structured group. But this advantage produces a big disadvantage of querying the data. Data may not be consistent, meaning it could have different values for one key. Also data integrity can suffer if not managed correctly. Moreover there's no "universal" language to query the data as there's SQL for the relational databases.

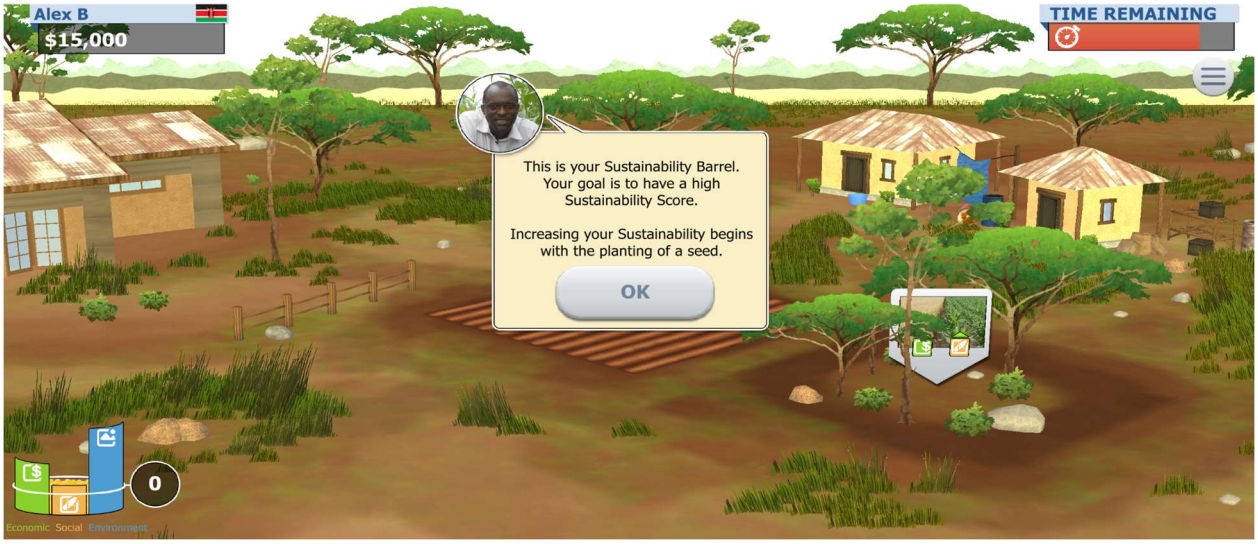
The preference is given to the relational database PostgreSQL as we need to support links between data. Also a big bonus is the existence of SQL.

Work Examples

The user story starts with a menu that allows adolescents to select a game. The game is picked and it automatically launches.

The example (Fig. 2) is shown based on the Journey 2050 game [19].

a



b

c

Figure 2 – The example of the Journey 2050 game: a – select level menu; b – level 1; c – level 4

The next step is to pick a level. The game consists of several game tasks of different complexity levels [20].

As the person completes levels, the results of the completion are passed to the Neuro-Fuzzy Inference System, where the final score is generated. Based on the previous works, to get a score the following formulas are used [20]:

$$v = \frac{t_{choice}}{N \cdot 60},$$

where t_{choice} – time in seconds that an adolescent spends choosing a computer game task; 60 – time in seconds, that is given to select a task; N is the ordinal number of the selected task ($N = 1,2,3$).

And [20]:

$$\theta = v + \zeta + \sum_{i=2}^{4} \frac{C_i}{\tau_i},$$

where ζ – estimation of the first level of the task; τ_i – the time of passing the *i*-th ($i = 2, 3, 4$) level of the computer game task; C_i – a constant determined by the time of execution of the i-th level of the task by a qualified specialist.

Based on the $(D = \theta - v)$ difference the recommendatory conclusion can be generated.

If $D > 4$, where 4 is the maximum achievable score that is defined by how the expert completed the game, then the profession can be recommended to the adolescent.

On the other hand $D < 1$ means that the task completely failed or the task wasn't finished, then the conclusion will be "try again" or "choose another game".

In the case of $1 < D < 4$ the results of the particular levels are considered. Let's assume the 1st and 2nd levels are passed with mistakes but 3rd and 4th, which are harder, are passed without mistakes. The conclusion can be "recommended" too, because the adolescent tried to improve results.

After results are calculated they are passed back to the user's profile, where the person can see all activities.

Conclusion

1. The generalised architecture of the info-communication system of the electronic estimation of adolescents' special abilities is provided.

2. Based on the analysis of the available development tools results, it was determined which of them should be used for the implementation of an info-communication system for the electronic estimation of adolescents' special abilities. The main criteria for the selected tools was to allow the info-communication system to be scalable in the future.

3. The example of the professional direction game shows the estimation of the adolescent's degree of interest in the future profession.

References

1. https://pure.coventry.ac.uk/ws/portalfiles/portal/11918037/chapter100comb.pdf
2. "Gamefication of Youth's Career Guidance Self-Identification" 2022 IEEE Smart Information Systems and Technologies (SIST), 28-30 April, 2022, Nur-Sultan, Kazakhstan.
3. Career guidance test battery "Magellano University". URL: https://magellano.com.ua/method/.
4. https://leapscholar.com/blog/sat-reasoning-test-everything-you-need-to-know/.
5. https://osvita.diia.gov.ua/en/prof-orientation-quiz.
6. https://www.geeksforgeeks.org/monolithic-vs-microservices-architecture/.
7. https://www.redhat.com/en/topics/api/what-is-a-rest-api.
8. https://careerfoundry.com/en/blog/web-development/django-framework-guide/.
9. https://medium.com/@tubelwj/comparison-of-flask-django-and-fastapi-advantages-disadvantages-and-use-cases-63e7c692382a.
10. https://www.upsolver.com/blog/kafka-versus-rabbitmq-architecture-performance-use-case
11. https://data-flair.training/blogs/advantages-and-disadvantages-of-kafka
12. https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API
13. https://redis.com/nosql/key-value-databases/
14. https://redis.io/docs/interact/pubsub/
15. https://fastapi.tiangolo.com/tutorial/background-tasks/
16. https://docs.celeryq.dev/en/stable/getting-started/backends-and-brokers/index.html
17. https://airbyte.com/data-engineering-resources/s3-gcs-and-azure-blob-storage-compared
18. https://aloa.co/blog/relational-vs-non-relational-database-pros-cons
19. https://www.journey2050.com/about-us/.
20. "Information technology of adolescents` professional self-identification" 3rd International Workshop on Intelligent Information Technologies & Systems of Information Security (IntelITSIS-2022) Khmelnytskyi, Ukraine, May 25 – 27, 2022.

Received 02.03.2024

Поляков Микита Олександрович
Аспірант кафедри інформаційних технологій проєктування та прикладної математики,
https://orcid.org/0000-0002-5061-4866
Київський національний університет будівництва і архітектури, Київ, Україна
Єременко Богдан Михайлович
Кандидат технічних наук, доцент кафедри технологій управління
https://orcid.org/0000-0002-3734-0813
Київський національний університет ім. Тараса Шевченка, Київ, Україна

ІНФО-КОМУНІКАЦІЙНА СИСТЕМА ЕЛЕКТРОННОГО ОЦІНЮВАННЯ СПЕЦІАЛЬНИХ ЗДІБНОСТЕЙ ПІДЛІТКІВ

Анотація. *Фокус цього дослідження спрямовано на профорієнтаційний супровід підлітків. Показано, що наразі існують різні методи і засоби підтримки рішення підлітка щодо вибору майбутньої професії, але переважна більшість з них або довготривалі, або супроводжуються ризиками халатного відношення до процесу проведення профорієнтації саме з боку підлітків через нерозуміння чи втрату інтересу до самого процесу. Саме тому ця робота є продовженням досліджень, спрямованих на розроблення інформаційно-комунікаційних систем, що використовують ігрові технології для того, щоб привертати увагу підлітка під час виконання професійно-орієнтованих завдань для самооцінки власних спеціальних здібностей. У пропонованій роботі на основі доступного апаратного і програмного забезпечення побудовано архітектуру системи електронного оцінювання спеціальних здібностей підлітків і розглянуто технології, які можуть бути використані при побудові інфо-комунікаційної системи. Головна ідея полягає в тому, щоб побудувати мікросервісну архітектуру, яка має потенціал до розширення і змогу зберігати копії даних у хмарному сховищі. Запропоновано використовувати такі технології, як: Kafka, Celery, Redis, PostgreSQL, AWS S3 Storage, FastAPI. Описано приклади визначення фінального висновку системи з рекомендацією щодо професії. Показано приклад роботи системи з боку користувача з використанням гри.*

Ключові слова: архітектура системи; мікросервіс; резервне копіювання даних; фонові завдання; фреймворк; система черги повідомлень

Link to publication

APA Poliakov, M., & Yeremenko, B., (2024). Info-Communication System of the Electronic Estimation of Adolescents' Special Abilities. *Management of Development of Complex Systems*, 58, 139–145, dx.doi.org\10.32347/2412-9933.2024.58.139-145.

ДСТУ Поляков, М. В., Єременко Б. М. Інфо-комунікаційна система електронного оцінювання спеціальних здібностей підлітків. *Управління розвитком складних систем.* Київ, 2024. № 58. С. 139 – 145, dx.doi.org\10.32347/2412-9933.2024.58.139-145.