ВИКОРИСТАННЯ МІРКОСЕРВІСНОЇ АРХІТЕКТУРИ ДЛЯ РЕКОМЕНДАЦІЙНИХ СИСТЕМ

ПІДГОТУВАВ СТУДЕНТ 3 КУРСУ КН
КУЦЕНКО АНДРІЙ

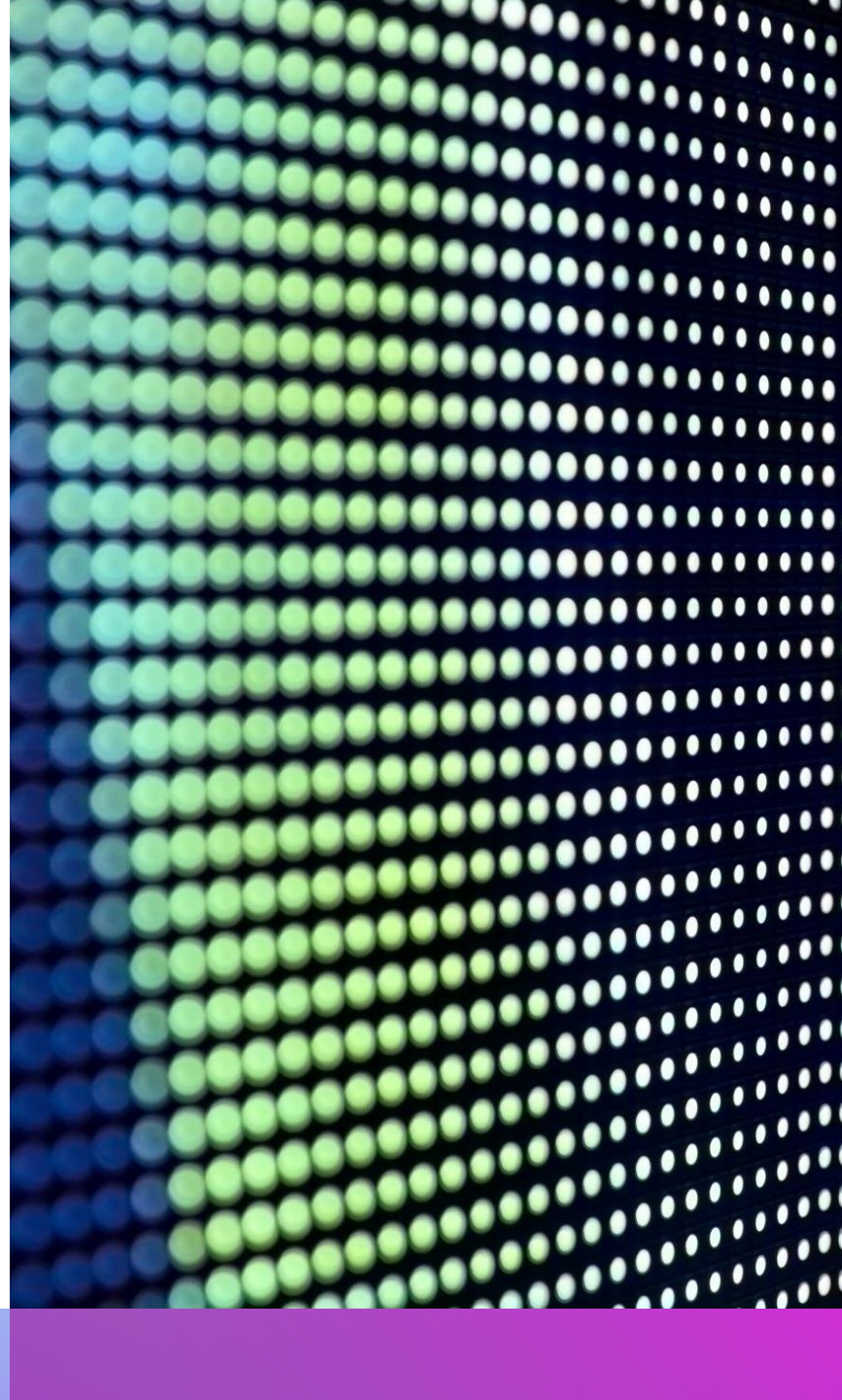
НАУКОВИЙ КЕРІВНИК БАБИЧ ТРОХИМ

В С Т У П

- Рекомендаційні системи: актуально, є попит, розвиваються
- Мікросервісна архітектура: невід’ємна частина сучасних веб-додатків
- Мета: створення системи рекомендацій соцмережі для пошуку стартапів

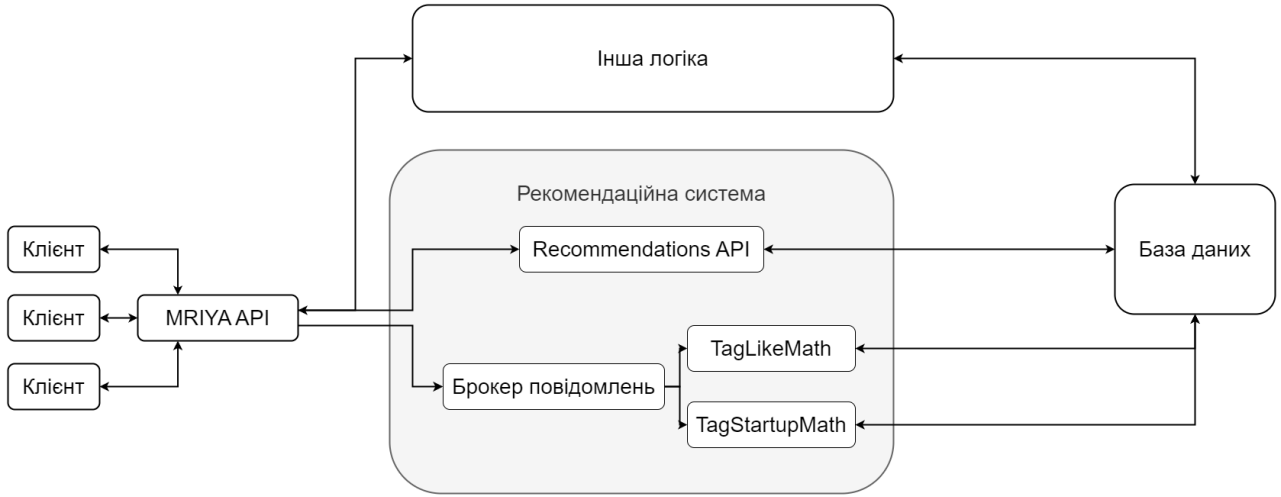
ТЕХНОЛОГІЇ

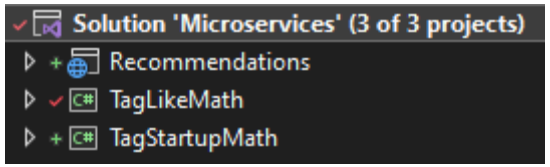
- ASP .Net Core 7.0
- PostgreSQL
- RabbitMQ




БАЗА ДАНИХ
Startup
login varchar(30)
owner integer
name varchar(50)
short_description varchar(400)
can_apply boolean
verify boolean
id integer
Startup_standard_Tag
startup_id integer
tag_id integer
Startup_stat_Tag
tag_count smallint
startup_id integer
tag_id integer
Post_owner_Startup
writer_id integer
show_writer boolean
startup_id integer
post_id integer
Startup_Languages
startup_id integer
language_id integer
Searching_Specialists
startup_id integer
specialist_id integer
Sex
sex varchar
id integer
Language
language varchar
id integer
Specialist
specialist varchar(50)
standard boolean
specialist_standard_ic integer
id integer
Tag
tag varchar
id integer
Post_stat_Tag
tag_count smallint
post_id integer
tag_id integer
Post_standard_Tag
post_id integer
tag_id integer
Post
title varchar(150)
text varchar(5000)
date_posting timestamp
id integer
Like
date timestamp
is_active boolean
user_id integer
post_id integer
User_history_Post
date timestamp
user_id integer
post_id integer
Subscription_Startup
date timestamp
user_id integer
startup_id integer
User_request_Startup
position varchar(50)
description varchar(300)
date timestamp
user_id integer
startup_id integer
User_Sex
user_id integer
sex_id integer
User_Languages
user_id integer
language_id integer
User_Specialists
user_id integer
specialist_id integer
Startup_employee_User
access_writer boolean
specialist_id integer
date_joining timestamp
user_id integer
startup_id integer
Post_owner_User
user_id integer
post_id integer
User_stat_Tag
tag_count smallint
user_id integer
tag_id integer
User_standard_Tag
user_id integer
tag_id integer
User
login varchar(30)
email varchar
name varchar(50)
description varchar(400)
verify boolean
date_joining timestamp
password varchar(256)
id integer
Subscription_user
date timestamp
user_id integer
user_subscribe_id integer
User_Chat
date_joining integer
chat_id integer
user_id integer
Chat
max_users integer
name varchar(50)
startup_id integer
date_of_creating timestamp
date_last_message timestamp
id integer
Message
user_id integer
chat_id integer
text varchar(500)
date timestamp
id bigint

АРХІТЕКТУРА





ФІЛЬТРАЦІЯ НА ОСНОВІ ВМІСТУ

S(score), де *n* - кількість рядків в матриці, *l* - кількість вподобайок постам, які користувач поставив постам користувача, або стартапу, кому належить цей пост, а *h* - кількість показів постів.

$$S = \left(1 - \frac{l}{h}\right) * \sum_{i=1}^{n} |Xi1 - Xi2|$$

ВЗАЄМОДІЯ З СКБД ЧЕРЕЗ ENTITY FRAMEWORK

```cs
internal class Post
{
    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]

    public int id { get; set; }

    [Required]
    [StringLength(150)]

    public string title { get; set; }

    [Required]
    [StringLength(5000)]
    0 references
    public string text { get; set; }

    [Required]
    0 references
    public DateTime date_posting { get; set; }
}
```

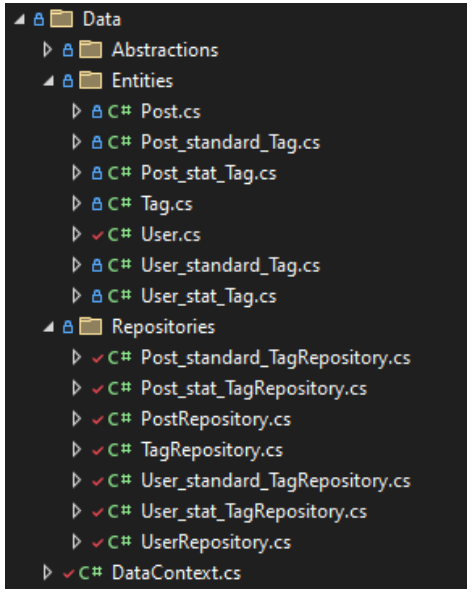
```cs
var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllers();

builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

builder.Services.AddScoped<IPostRepository, PostRepository>();
builder.Services.AddScoped<IUserRepository, UserRepository>();
builder.Services.AddScoped<IStartupRepository, StartupRepository>();
builder.Services.AddScoped<IPostRecomendationsService, PostRecomendationsService>();
```

```cs
public Post_standard_Tag? GetPost_standard_TagByIds(int post_id, int tag_id)
{
    return _context.Post_standard_Tag.Where(t => t.post_id == post_id && t.tag_id == tag_id).FirstOrDefault();
}
1 reference
public IQueryable<Post_standard_Tag> GetAll()
{
    return _context.Post_standard_Tag.AsQueryable();
}
2 references
public IQueryable<Post_standard_Tag >GetPost_standard_TagByPostId(int post_id)
{
    return _context.Post_standard_Tag.Where(t => t.post_id == post_id);
}
//----------------------------------------
1 reference
public bool Add(Post_standard_Tag newEntity)
{
    var entity = _context.Post_standard_Tag.Add(newEntity);
    return true;
}
```



ВЗАЄМОДІЯ З СКБД ЧЕРЕЗ SQL ЗАПИТИ

```
public List<int> GeUserIDsRecomandationByStartupIDSpecialIDs(int startup_id, int specialistID, int limit)
{
    var sql = @"WITH user_stat_tags AS(
                    WITH user_tags AS (
                        SELECT user_id, tag_id, ROUND(tag_count * @notStandartValue) AS tag_count
                        FROM ""User_stat_Tag""
                    ),
                    user_standard_tags AS (
                        SELECT user_id, tag_id, @standartValue AS tag_count
                        FROM ""User_standard_Tag""
                    )
                    SELECT
                        COALESCE(user_tags.user_id, user_standard_tags.user_id) AS user_id,
                        COALESCE(user_tags.tag_id, user_standard_tags.tag_id) AS tag_id,
                        COALESCE(user_tags.tag_count, 0) + COALESCE(user_standard_tags.tag_count, 0) AS tag_count
                    FROM user_tags
                    FULL OUTER JOIN user_standard_tags ON user_tags.tag_id = user_standard_tags.tag_id AND user_tags.user_id = user_standard_tags.user_id
                ),
                startup_tags AS (
                    SELECT
                        s.id AS startup_id,
                        t.id AS tag_id,
                        COALESCE(st.tag_count, 0) AS tag_count
                    FROM ""Startup"" s
                    CROSS JOIN ""Tag"" t
                    LEFT JOIN ""Startup_stat_Tag"" st ON st.startup_id = s.id AND st.tag_id = t.id
                    WHERE startup_id = @startup_id
                ),
                user_standard_specialists AS (
                    SELECT user_id
                    FROM ""User_Specialists""
                    WHERE specialist_id IN (SELECT id
                                            FROM ""Specialist""
                                            WHERE id = @specialistID OR specialist_standard_id = @specialistID
                                            UNION
                                            SELECT id
                                            FROM ""Specialist""
                                            WHERE specialist_standard_id IN (SELECT id
                                                                             FROM ""Specialist""
                                                                             WHERE id = @specialistID OR specialist_standard_id = @specialistID
                                                                             ))
                ),
```

```
public List<int> GetUserPostsIDsSubscrription(int user_id)
{
    var sql = @"SELECT id
    FROM ""Post""
    WHERE (id IN (SELECT post_id
                  FROM ""Post_owner_User""
                  WHERE user_id IN (SELECT user_subscribe_id
                                    FROM ""Subscription_user""
                                    WHERE user_id = @user_id))
          OR id IN (SELECT post_id
                    FROM ""Post_owner_Startup""
                    WHERE startup_id IN (SELECT startup_id
                                         FROM ""Subscription_Startup""
                                         WHERE user_id = @user_id)))
    AND id NOT IN (SELECT post_id
                   FROM ""User_history_Post""
                   WHERE user_id = @user_id)
    AND date_posting >= (CURRENT_DATE - INTERVAL '1 month')
    ORDER BY date_posting DESC";

    return connection.Query<int>(sql, new { user_id = user_id }).ToList();
}
```

```
            user_language AS (
                SELECT user_id
                FROM ""User_Languages"" ul
                JOIN ""Startup_Languages"" sl ON sl.language_id = ul.language_id
                WHERE startup_id = 1
            )
            SELECT
                ut.user_id
            FROM user_stat_tags ut
            LEFT JOIN startup_tags st ON ut.tag_id = st.tag_id
            JOIN ""User"" sta ON ut.user_id = sta.id
            JOIN user_standard_specialists uss on uss.user_id = ut.user_id
            JOIN user_language ul on ul.user_id = ut.user_id
            GROUP BY ut.user_id
            ORDER BY SUM(ABS(ut.tag_count - st.tag_count))
            LIMIT @limit";

return connection.Query<int>(sql, new
{
    startup_id = startup_id,
    limit = limit,
    specialistID = specialistID,
    standartValue = TagStartupMathRules.PercentOfStandartStartup,
    notStandartValue = (float)TagStartupMathRules.PercentOfStandartStartup / 100
}).ToList();
```

RABBITMQ

```
public Task Start()
{
    while (true)
    {
        if (activeProcessCount < processCount)
        {
            BasicGetResult result = channel.BasicGet(queuename, autoAck: false);
            if (result != null)
            {
                channel.BasicAck(result.DeliveryTag, multiple: false);
                byte[] body = result.Body.ToArray();
                string message = Encoding.UTF8.GetString(body);

                _ = Task.Factory.StartNew(rabbitMQReader, message);

            }
        }
    }
}
```

docker-compose.yml

```
version: "2.1"
services:
  rabbitmq:
    image: rabbitmq:3.11-management
    environment:
      - RABBITMQ_DEFAULT_USER=user
      - RABBITMQ_DEFAULT_PASS=password
      - RABBITMQ_SERVER_ADDITIONAL_ERL_ARGS=-rabbit log_levels [{connection,error},{default,error}] disk_free_limit 2147483648
    volumes:
      - ./rabbitmq:/var/lib/rabbitmq
    ports:
      - 15672:15672
      - 5672:5672
```

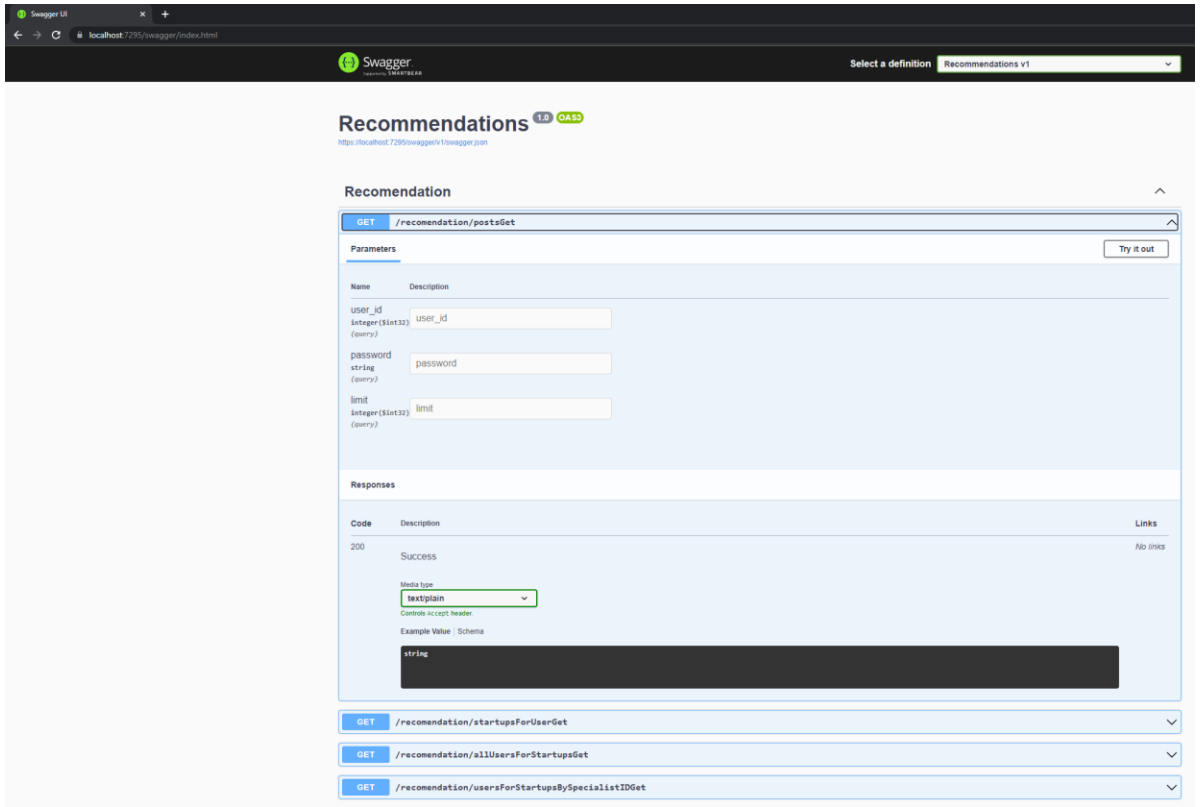
ТЕСТУВАННЯ

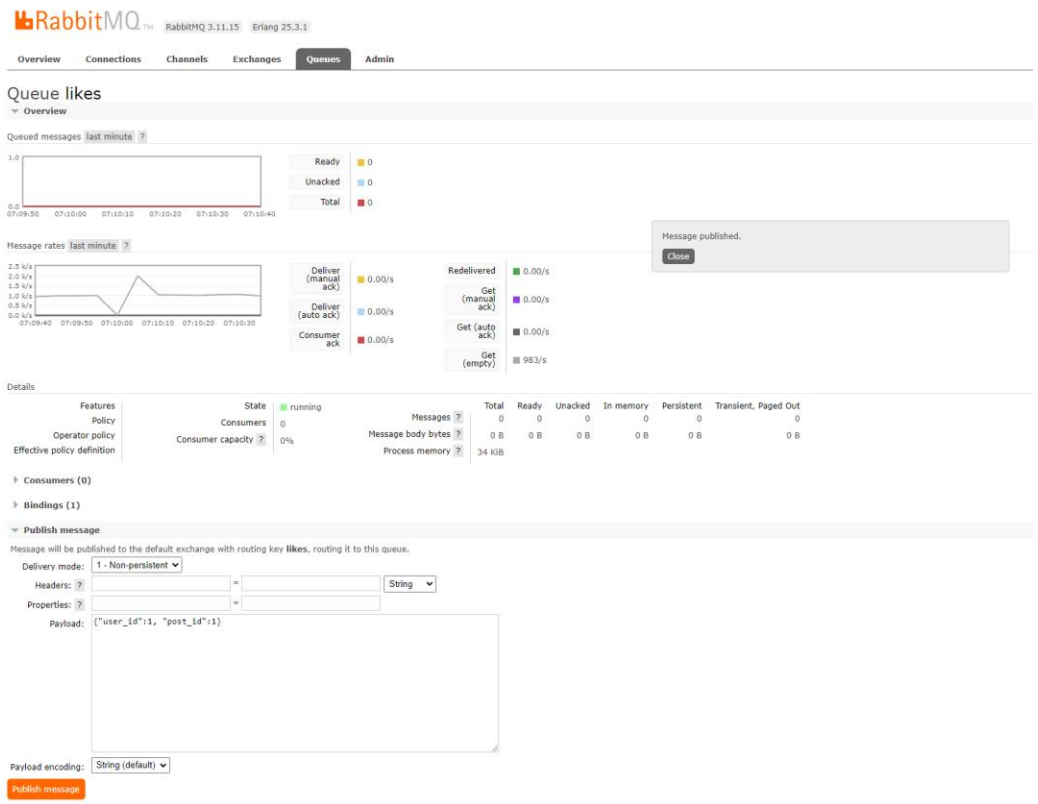
Startup_Languages [Mriya] | User_Languages [Mriya] | User [Mriya] | Post [Mriya] | User_stat_Tag [Mriya] | Post_stat_Tag [Mriya] | Startup_stat_Tag [Mriya] | public | Searching...pecialists [Mriya]

5 rows | Tx: Auto | DDL

WHERE | ORDER BY

	id	login	email	name	description	verify	date_joining	password
1	5	andrij	andrij@gmail.com	Andrij	asdfg	true	2023-05-08 17:24:47.030335	123
2	3	riper	kutsenkoandrij2003@gmail.com	Kutsenko Andrij	asdfg	true	2023-05-06 00:55:53.000000	123
3	4	maria	maria@gmail.com	Maria	asdg	true	2023-05-08 17:24:47.030335	123
4	2	lansselot	somemail@gmail.com	Dima Pojarov	asdf	false	2023-05-07 02:20:16.094235	123
5	1	kirill	somemail@gmail.com	Kirill	asf	true	2023-05-08 17:23:46.586413	123





В И С Н О В О К

- Завдяки цій роботі вдалося дослідити, на зараз, важливу тему рекомендаційних систем, а також використання для їх побудови мікросервісної архітектури, та застосувати дані знання на проєкті, який в подальшому може бути застосований в реальному веб-додатку.

ДЯКУЮ ЗА УВАГУ!