

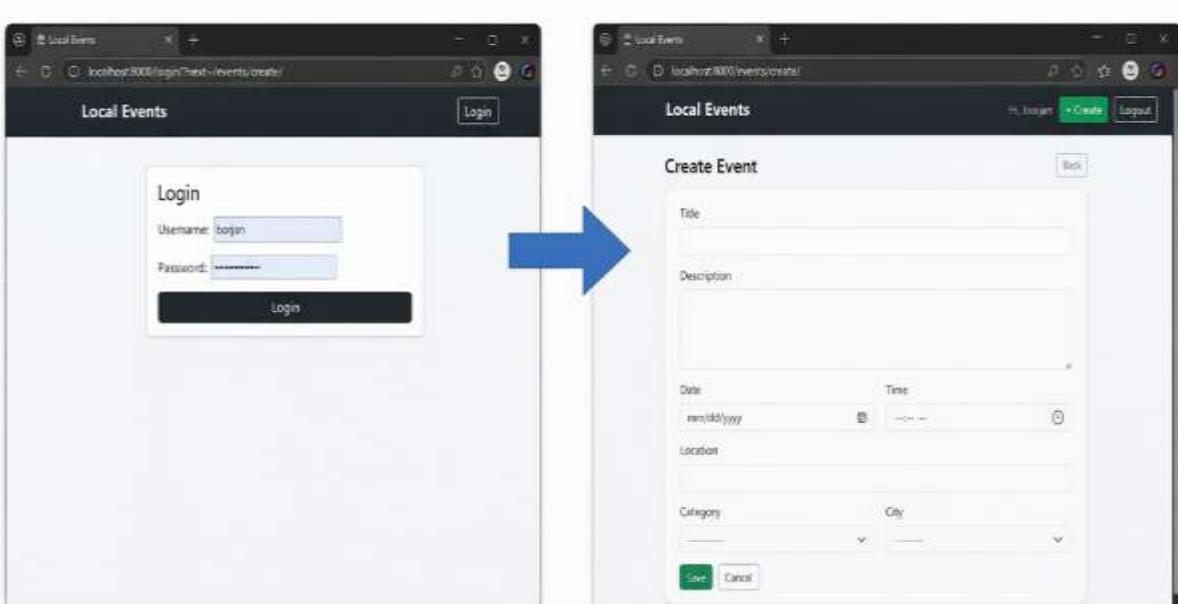
Web апликација за управување со локални настани

Local Event

■ ЗАДАЧА 1

Автоматско пренасочување кон login при неовластен пристап

- Ако корисникот отвори /events/create/ без да е најавен
- автоматски се префрла на /login/?next=/events/create/

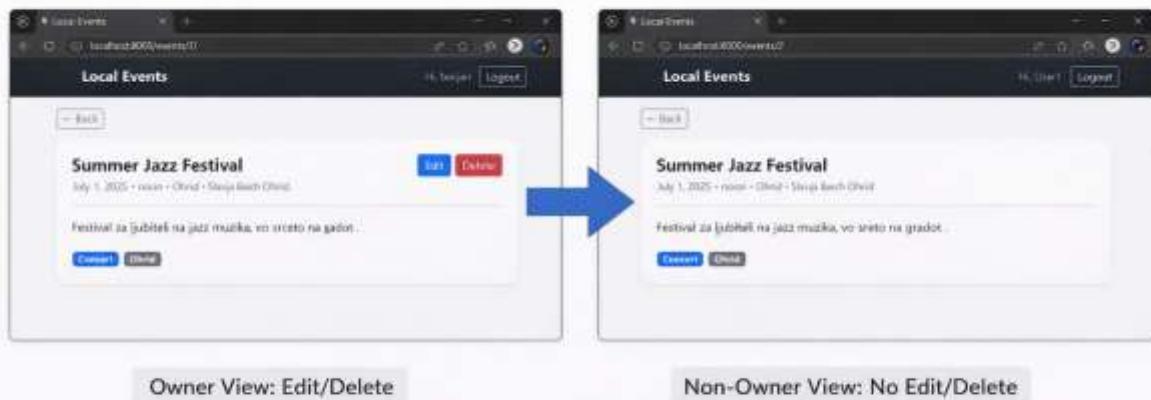


При обид за пристап до заштитена функционалност без најава, системот автоматски го пренасочува корисникот кон login страницата со next параметар. По успешна најава, корисникот се врaka на претходно бараната Create Event страна.

Неовластен пристап → автоматски redirect → успешен login → пристап дозволен

■ ЗАДАЧА 2 Owner authorization

Како се имплементира owner-based authorization за Edit/Delete



Кога корисникот е сопственик на настанот, опции за уредување и бришење се достапни.

Кога корисникот не е сопственик, опциите стануваат невидливи.

```
<div class="card shadow-sm border-0 rounded-4">
    <div class="card-body p-4">

        <div class="d-flex justify-content-between align-items-start gap-3">
            <div>
                <h3>{{ event.title }}</h3>
                <div class="text-muted small">
                    {{ event.date }} - {{ event.time }} - {{ event.city.name }} - {{ event.location }}
                </div>
            </div>
        </div>

        <div class="mt-3" style="font-size: 0.8em; font-weight: bold; color: #555; margin-bottom: 10px;">
            &#9702; user.is_authenticated and user.id == event.owner_id
        </div>

        <div class="d-flex gap-2">
            <a href="{% url 'event_edit' event.id %}" class="btn btn-primary btn-sm">Edit</a>
            <a href="{% url 'event_delete' event.id %}" class="btn btn-danger btn-sm">Delete</a>
        </div>
        <br>
        <p>{{ event.description }}</p>
        <div class="d-flex flex-wrap gap-2">
            <span class="badge text-light-primary">{{ event.category.name }}</span>
            <span class="badge text-light-success">{{ event.city.name }}</span>
        </div>
    </div>
</div>
```

■ ЗАДАЧА 3

Како корисникот може да пребарува настани преку полето за пребарување?

The screenshot shows a web application interface titled "Local Events". At the top, there is a search bar with the placeholder "Search" containing the text "nova". Below it are two dropdown menus: "Category" set to "Concert" and "City" set to "Ohrid". There are "Apply" and "Reset" buttons. The main content area is titled "Discover local events" with the subtitle "Concerts, festivals, meetups and more. Find what's happening in your city.". It displays a single event card for "Nova godina" on Dec. 31, 2023, in Ohrid, categorized as a Concert. The event card includes a "Details" button.

Системот користи параметри од URL адресата за да ги филтрира настаните.
Корисникот може да пребарува по текст и истовремено да избере категорија и град.

The screenshot shows a code editor with a Python file named "views.py" open. The code defines a function-based view "event_list" that filters events based on category and city. It uses Django's QuerySet filtering methods like ".filter(category_id=category_id)" and ".filter(city_id=city_id)". It also handles dropdown lists for categories and cities by filtering the queryset if the dropdown values are not empty. The code includes imports for "Category" and "City" models, and a paginator for displaying the filtered events.

```
def event_list(request):
    if request.GET.get("category") or "":
        category_id = request.GET.get("category") or ""
    if request.GET.get("city") or "":
        city_id = request.GET.get("city") or ""

    events = Event.objects.all().order_by("-id")

    # Search
    if "q" in request.GET:
        events = events.filter(
            Q(title__icontains=q) |
            Q(description__icontains=q) |
            Q(location__icontains=q) |
            Q(categories__name__icontains=q) |
            Q(city__name__icontains=q)
        )

    # Filters
    if category_id:
        events = events.filter(category_id=category_id)

    if city_id:
        events = events.filter(city_id=city_id)

    # Dropdowns
    categories = Category.objects.all().order_by("name")
    cities = City.objects.all().order_by("name")

    # Pagination
    paginator = Paginator(events, per_page=8) # 8 events per page
    page = request.GET.get('page')
    events = paginator.get_page(page)

    context = {
        "events": events,
        "categories": categories,
        "cities": cities
    }

    return render(request, 'events/list.html', context)
```

■ ЗАДАЧА 4

Како системот овозможува прикажување на голем број настани на повеќе страници?

The screenshot shows a search interface titled "Discover local events" with the subtitle "Concerts, festivals, meetups and more. Find what's happening in your city." It displays a list of events under the heading "Events" with "7 total". Two events are visible: "Modern Art" and "Wizz maraton". Each event card includes a date, location, category, and a "View" button. Below the cards is a navigation bar with "Previous", "Page 1 of 2", and "Next" buttons, with a red arrow pointing to the "Next" button.

Настаните се поделени на страници со ограничен број резултати по страница.

This screenshot shows the same search interface but with three events per page. The events listed are "Balkan Business", "Nova godina", and "Biznis Nastan". Each event has a similar structure with a date, location, category, and a "View" button. The navigation bar at the bottom shows "Previous", "Page 2 of 2", and "Next", with a red arrow pointing to the "Next" button.

Корисникот може да се движи помеѓу страниците со помош на pagination.

```
news.py
if paginator = Paginator(events, 4) # events per page
    page_number = request.GET.get("page")
    page_obj = paginator.get_page(page_number)
```

A red arrow points to the line where the paginator object is assigned. Another red arrow points to the line where the page number is retrieved from the GET request.

Со Django Paginator, настани се делат на повеќе страници.

```
list.html
{
    ...
    <% if page_obj.paginator.num_pages > 1 %>
        <nav class="mt-4">
            <ul class="pagination justify-content-center">
                ...
                <% if page_obj.has_previous %>
                    <a href="?page={{ page_obj.prev_page_number }}>
                        page_number = request.GET.get("page")
                        page_obj = paginator.get_page(page_number)
                    </a>
                <% endif %>
            </ul>
        </nav>
    <% endif %>
}
```

A red arrow points to the line where the page number is assigned from the GET request. Another red arrow points to the line where the page object is assigned from the paginator.

```

<% if page_obj.paginator.num_pages > 1 %>
    <nav class="mt-4">
        <ul class="pagination justify-content-center">
            ...
            <% if page_obj.has_previous %>
                <a href="?page={{ page_obj.prev_page_number }}>
                    page_number = request.GET.get("page")
                    page_obj = paginator.get_page(page_number)
                </a>
            <% endif %>
        </ul>
    </nav>
<% endif %>
```

A red arrow points to the line where the previous page link is generated using the prev_page_number attribute of the paginator object.

Pagination во HTML template овозможува корисникот да се движи помеѓу страниците.

■ ЗАДАЧА 5

Како да се ограничи пристапот до одредени функционалности само за најавени корисници (Login Required)?

The screenshots illustrate the Local Events application's user interface and its underlying code structure.

Top Screenshot: Shows the 'Events' page with three event cards:

- Balkan Business:** Date: Jun 23, 2024 • 9:30 pm • Belgrade
Balkan meeting vo belgrad
- Nova godina:** Date: Dec 31, 2024 • noon • Ohrid
Koncert nova godina vo ohrid
- Bilans Nastan:** Date: Jan 18, 2025 • 8 am • Skopje
Bilans nastan konferencija tima za balkanske biznesi

Middle Screenshot: Shows the same 'Events' page with three different event cards:

- Modern Art:** Date: Jul 15, 2024 • 6 pm • Paris
Umetničko studio izlazak
- Wizz maraton:** Date: Jul 20, 2024 • 6 pm • Skopje
Maraton vo skopje
- Food Truck:** Date: Jul 19, 2024 • noon • Berlin
Food truck in the city

Bottom Screenshot: Shows the code editor in Visual Studio with a C# file named `Event.cs`. The code defines an `Event` class with properties like `Name`, `Date`, `Location`, and `Description`. It includes methods for `GetEventDetails`, `GetEventAttendees`, and `GetEventReviews`.

```
using System;
using System.Collections.Generic;
using System.Linq;

public class Event
{
    public string Name { get; set; }
    public DateTime Date { get; set; }
    public string Location { get; set; }
    public string Description { get; set; }

    public void GetEventDetails()
    {
        // Implementation
    }

    public void GetEventAttendees()
    {
        // Implementation
    }

    public void GetEventReviews()
    {
        // Implementation
    }
}
```

FIGMA

Local Events

Home + Create Event Login

Discover Local Events

Find the best concerts, sports, and other community gatherings happening in your area.

Search events by name, category... All Locations Search

Upcoming Events

All Categories This Week



Music

Aug 15, 2024 • 8:00 PM

Summer Jazz Festival 2024

New York, NY

Experience an evening of smooth jazz under the stars with top local artists.

[View Details →](#)



Tech

Aug 16, 2024 • 7:30 PM

Tech Startup Mixer

San Francisco, CA

Network with founders, investors, and developers in the bay area. Free drinks!

[View Details →](#)



Food

Aug 20, 2024 • 11:00 AM

Downtown Food Truck

Austin, TX

Taste the best local street food from over 30 vendors. Live music and family fun!

[View Details →](#)



Sports



Art

Aug 18, 2024 • 5:00 PM

Modern Art Exhibition

Chicago, IL

Be the first to see the new collection from emerging contemporary artists.

[View Details →](#)



Community

Aug 25, 2024 • 10:00 AM

Community Garden

Portland, OR

Learn sustainable gardening techniques and help plant the new fall harvest. Tools provided!

[View Details →](#)

[Load More Events](#)