✦ Member-only story

Elastic Search — Day 9: Backup and Restore!!

Navya Cloudops · Following

9 min read · 4 days ago

Welcome back to Day 9 of our 10-day DevOps Elasticsearch course! Today, we delve into one of the most critical aspects of Elasticsearch management: Backup and Restore. As data integrity and availability are paramount in any production environment, understanding how to backup and restore your Elasticsearch data is indispensable.

Importance of backups in Elasticsearch:

The importance of backups cannot be overstated, especially in a production environment where data integrity and availability are critical. Here’s a more detailed explanation:

1. Data Loss Prevention:

- Elasticsearch is often used to store vast amounts of valuable data, including logs, metrics, and application-generated data.
- Any unexpected data loss due to hardware failures, software bugs, or human errors can have severe consequences, leading to business disruption and financial losses.
- Backups serve as a safety net, enabling you to restore lost or corrupted data and minimize downtime.

2. Business Continuity:

- In the event of a disaster or system failure, the ability to quickly restore data is essential for maintaining business continuity.
- Backups provide a means to recover data and resume operations with minimal disruption, ensuring that critical services remain available to users and customers.

3. Regulatory Compliance:

- Many industries and jurisdictions have stringent data retention and compliance requirements.
- Backups help organizations meet regulatory standards by providing a historical record of data that can be audited and verified as needed.

4. Disaster Recovery:

- Natural disasters, cyber attacks, and other unforeseen events can result in data loss or corruption.
- Backups form an integral part of disaster recovery plans, allowing organizations to recover from catastrophic events and restore data to a known good state.

5. Version Upgrades and Migrations:

- Upgrading Elasticsearch to a new version or migrating to a different infrastructure can pose risks to data integrity.

- Backups serve as a safety mechanism during version upgrades and migrations, enabling you to roll back changes if compatibility issues or data inconsistencies arise.

6. Peace of Mind:

- Knowing that your data is backed up and protected provides peace of mind to stakeholders, administrators, and end users.
- It instills confidence in the reliability and resilience of the Elasticsearch infrastructure, fostering trust and credibility within the organization.

Snapshot and restore strategies:

These are essential components of data management in Elasticsearch. They involve creating point-in-time backups of your indices and metadata, which can be restored in case of data loss or corruption. Here's a detailed explanation of snapshot and restore strategies:

Snapshotting:

Incremental Backups:

- Elasticsearch supports incremental backups, allowing you to capture changes since the last snapshot. This minimizes the amount of data transferred and reduces backup times.
- Incremental backups are particularly useful for large datasets where full backups are impractical or resource-intensive.

Scheduled Snapshots:

- Schedule snapshots at regular intervals to ensure that your backup data is up to date.
- Determine an appropriate snapshot frequency based on your data retention policies, recovery objectives, and resource constraints.

Offsite Storage:

- Store snapshots in offsite locations or cloud storage to protect against localized failures or disasters.

- Offsite storage provides an extra layer of redundancy and ensures that backup data remains accessible even if the primary Elasticsearch cluster becomes unavailable.

Snapshot Lifecycle Management:

- Implement policies to manage snapshot retention and lifecycle.
- Define retention periods for snapshots based on business requirements, compliance regulations, and available storage capacity.
- Periodically review and prune old snapshots to free up storage space and maintain efficient backup operations.

Restore Strategies:

Point-in-Time Recovery:

- Elasticsearch snapshots capture the state of your indices at a specific point in time.
- Use point-in-time recovery to restore indices to a known good state, allowing you to recover from data loss, corruption, or accidental deletions.

Partial Restores:

- Elasticsearch allows you to restore individual indices or specific data subsets from snapshots.
- Perform partial restores to recover specific datasets or address localized issues without affecting the entire cluster.

Parallel Restores:

- Elasticsearch supports parallel restore operations, enabling you to expedite data recovery and minimize downtime.
- Distribute restore tasks across multiple nodes or instances to leverage parallel processing capabilities and improve restore performance.

Validation and Testing:

- Validate the integrity of snapshot data before initiating restores in a production environment.
- Conduct periodic restore tests in a controlled environment to ensure that backup and restore processes function as expected and meet recovery objectives.

Snapshot and restore strategies are essential components of Elasticsearch backup and disaster recovery plans. By implementing robust snapshotting mechanisms and restore procedures, you can safeguard your data against unexpected failures, ensure business continuity, and minimize the impact of downtime on critical operations.

Using built-in snapshot features:

This provides a convenient and reliable way to create backups of your data. Elasticsearch offers robust snapshot and restore capabilities out of the box, allowing you to capture the current state of your indices and metadata and store them in a repository for safekeeping. Here’s a detailed explanation of how to use built-in snapshot features in Elasticsearch:

1. Setting Up a Repository:

Before you can create snapshots, you need to set up a repository where Elasticsearch will store the backup data. Elasticsearch supports various repository types, including:

- **File System Repository:** Stores snapshots on a shared file system accessible to Elasticsearch nodes.
- **Amazon S3 Repository:** Stores snapshots in Amazon S3 buckets.
- **Azure Repository:** Stores snapshots in Azure Storage containers.
- **Google Cloud Storage Repository:** Stores snapshots in Google Cloud Storage buckets.

To set up a repository, use the **‘_snapshot’** API with the PUT method:

```
PUT /_snapshot/my_backup
{
  "type": "fs",
  "settings": {
    "location": "/mnt/backups/my_backup"
  }
}
```

In this example, we create a file system repository named **‘my_backup’** located at **‘/mnt/backups/my_backup’**.

2. Creating Snapshots:
Once you have set up a repository, you can create snapshots using the **‘_snapshot’** API. Specify the indices you want to include in the snapshot and any additional settings:

```
PUT /_snapshot/my_backup/snapshot_1
{
  "indices": "index1,index2",
  "ignore_unavailable": true,
  "include_global_state": false
}
```

- **snapshot_1**: Name of the snapshot.
- **indices:** Comma-separated list of indices to include in the snapshot.
- **ignore_unavailable:** (Optional) Ignore unavailable indices during snapshot creation.
- **include_global_state:** (Optional) Include the global cluster state in the snapshot.

3. Restoring Snapshots:
To restore a snapshot, use the **‘_snapshot’** API with the POST method:

```
POST /_snapshot/my_backup/snapshot_1/_restore
```

Elasticsearch will restore the indices included in the specified snapshot to their previous state.

4. Snapshot Lifecycle Management:
To manage snapshot lifecycle, you can define policies to automate snapshot creation and retention. Elasticsearch Curator is a popular tool for managing snapshot lifecycle. You can define Curator actions to schedule snapshot creation and deletion based on predefined criteria such as age or size.

Using built-in snapshot features in Elasticsearch provides a reliable mechanism for creating backups of your data.

Implementing automated backup solutions:

It is crucial for ensuring data integrity, minimizing the risk of data loss, and streamlining operational processes. Automation reduces the likelihood of human error and ensures that backups are performed consistently and according to predefined schedules. Here’s how you can implement automated backup solutions for Elasticsearch:

1. Use Snapshot Lifecycle Management:
Elasticsearch allows you to define policies for managing snapshot lifecycles. You can specify rules for automatically creating, retaining, and deleting snapshots based on criteria such as age, size, or the number of snapshots. Snapshot lifecycle management helps optimize storage usage and ensures that you have an appropriate number of snapshots available for recovery.

2. Leverage Elasticsearch Curator:
Elasticsearch Curator is a powerful tool for managing Elasticsearch indices and snapshots. You can use Curator to automate snapshot creation, deletion, and other maintenance tasks. Curator provides a flexible and expressive configuration language for defining actions and filters based on various criteria. For example, you can schedule snapshots to run daily or weekly and specify retention policies to keep a certain number of snapshots based on your backup requirements.

3. Integrate with Task Schedulers:
You can integrate Elasticsearch snapshot commands with task schedulers like cron jobs (Unix/Linux) or Task Scheduler (Windows) to automate backup tasks at specified intervals. Schedule snapshot creation and cleanup tasks using cron expressions or task scheduler configurations to ensure that backups are performed regularly and according to your organization’s backup policy.

4. Monitor Backup Status and Health:
Implement monitoring and alerting mechanisms to track the status and health of automated backup processes. Monitor snapshot creation, completion, and errors using Elasticsearch monitoring features or third-party monitoring solutions. Set up alerts to notify administrators or operations teams in case of backup failures or issues requiring attention.

5. Test Backup and Restore Procedures:
Regularly test your automated backup and restore procedures to ensure their reliability and effectiveness. Conduct backup and restore drills in a controlled environment to validate that backups are created successfully and can be restored in case of data loss or system failure. Testing helps identify and address potential issues before they impact production environments.

6. Implement Redundancy and Disaster Recovery Plans:
Consider implementing redundancy and disaster recovery plans to safeguard against data loss and minimize downtime. Store backups in multiple locations or repositories to protect against localized failures or disasters. Establish procedures for restoring backups in different scenarios, such as hardware failures, data corruption, or accidental deletions.

Automating backup solutions for Elasticsearch is essential for maintaining data availability, integrity, and compliance with regulatory requirements.

Below is an example project demonstrating how to set up Elasticsearch snapshots using Amazon S3 as the repository for backups. This project assumes you have an AWS account and Elasticsearch cluster already set up.

Prerequisites:

- An AWS account with access to S3.
- An Elasticsearch cluster running on AWS or accessible from your local environment.

Step 1: Set Up Elasticsearch Repository on Amazon S3

First, you need to create a repository in Amazon S3 where Elasticsearch will store its snapshots.

- Log in to the AWS Management Console.

- Navigate to the S3 service.
- Create a new S3 bucket to store Elasticsearch snapshots. Note down the bucket name.

Step 2: Configure Elasticsearch Repository

Once the S3 bucket is created, you need to configure Elasticsearch to use it as a repository.

- Open your Elasticsearch configuration file (elasticsearch.yml).
- Add the following configuration to specify the S3 repository:

```
repositories:
  s3:
    bucket: "your-s3-bucket-name"
    region: "your-s3-bucket-region"
```

Replace “**your-s3-bucket-name**” and “**your-s3-bucket-region**” with the appropriate values for your S3 bucket.

Step 3: Take a Snapshot

Now, you can take a snapshot of your Elasticsearch indices and store it in the S3 repository.

```
PUT /_snapshot/s3_repository_name/snapshot_name
{
  "indices": "index1,index2",
  "ignore_unavailable": true,
  "include_global_state": false
}
```

Replace “**s3_repository_name**” with the name of the repository you configured in Elasticsearch and “**snapshot_name**” with the desired name for your snapshot.

Step 4: Restore from Snapshot

To restore indices from a snapshot stored in the S3 repository, use the ‘**_restore**’ API:

```
POST /_snapshot/s3_repository_name/snapshot_name/_restore
```

Step 5: Automate Backups with Curator and AWS Lambda (Optional)

For automated backups, you can use Elasticsearch Curator and AWS Lambda:

- Set up Curator to create snapshots on a schedule.
- Configure an AWS Lambda function triggered by a CloudWatch Events rule to execute Curator commands at specified intervals.

This example project demonstrates how to set up Elasticsearch backups and restores using Amazon S3 as the repository. By following these steps, you can ensure that your Elasticsearch data is securely backed up and easily recoverable in case of data loss or system failure.

Here are **scenario-based interview questions** covering Elasticsearch backup and restore.

1. Can you outline a backup strategy for an Elasticsearch cluster handling sensitive financial data? What factors would you consider?
2. Suppose a critical Elasticsearch index containing customer orders becomes corrupted. Walk me through the steps you would take to restore the index from a snapshot.
3. How would you implement automated backups for an Elasticsearch cluster using Amazon S3 as the repository?
4. What steps would you take to ensure effective disaster recovery for an Elasticsearch cluster?
5. Your organization has implemented a data retention policy mandating that Elasticsearch indices older than six months be archived. How would you ensure compliance with this policy while managing backups?
6. During a routine backup operation, the Elasticsearch cluster experiences a failure while creating a snapshot. How would you troubleshoot and address this issue?

7. How would you ensure data consistency between the primary Elasticsearch cluster and its backup copies stored in Amazon S3?

Conclusion:

Backup and restore strategies are fundamental pillars of Elasticsearch management. By implementing robust snapshotting mechanisms and automated backup solutions, you ensure the safety and integrity of your data, facilitating seamless operations and disaster recovery.

In our final day, we'll explore about Deployment Strategies and Continuous Integration/Continuous Deployment (CI/CD). Stay tuned for more insights!

Elasticsearch DevOps Interview Learning Course

N

Following

Written by Navya Cloudops

514 Followers

More from Navya Cloudops

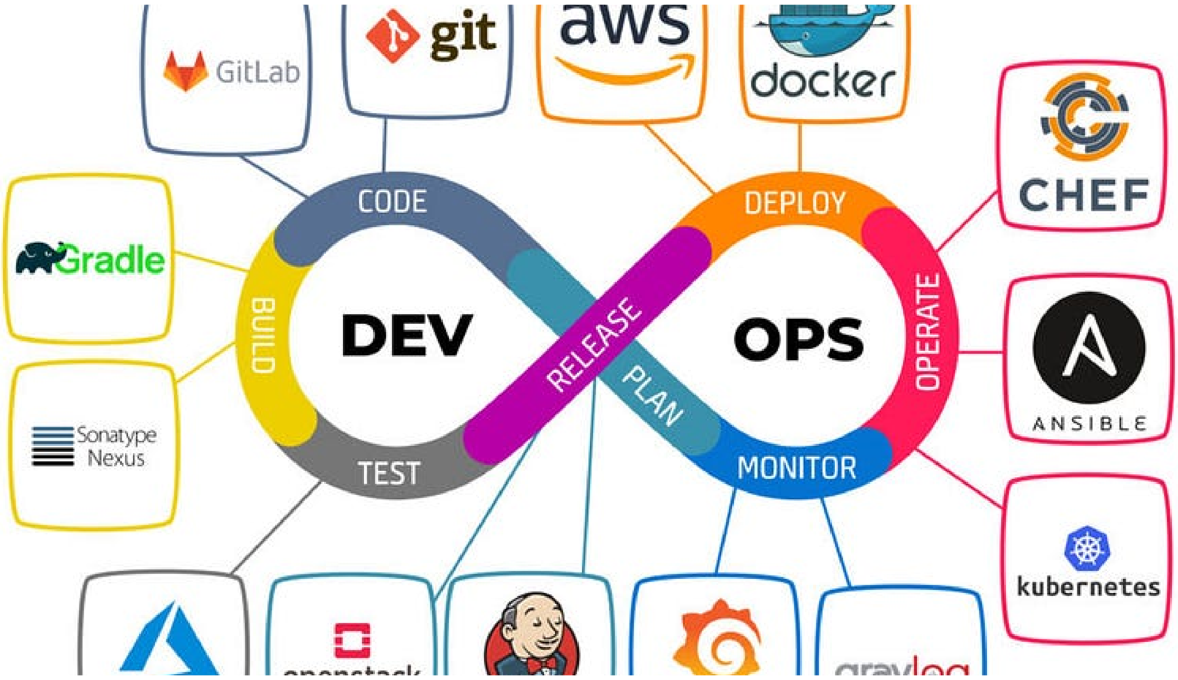
Navya Cloudops

Real-time Interview Questions for 4 years Experience!!

Here are some scenario based interview questions with respect to 4 years experience for a position in CITI Bank.

✦ · 3 min read · Jan 24, 2024

88



Navya Cloudops

DevOps Zero to Hero in 30 days!!

Here's a 30-day DevOps course outline with a detailed topic for each day!

4 min read · Jul 12, 2023

26 1

Navya Cloudops

DevOps Zero to Hero — Day 14: Release Management!!

Welcome to Day 14 of our 30-day course on Software Development Best Practices! In today's session, we'll delve into the critical aspect of...

7 min read · Jul 26, 2023

1

N Navya Cloudops

DevOps Zero to Hero — Day 20: Deployment Strategies

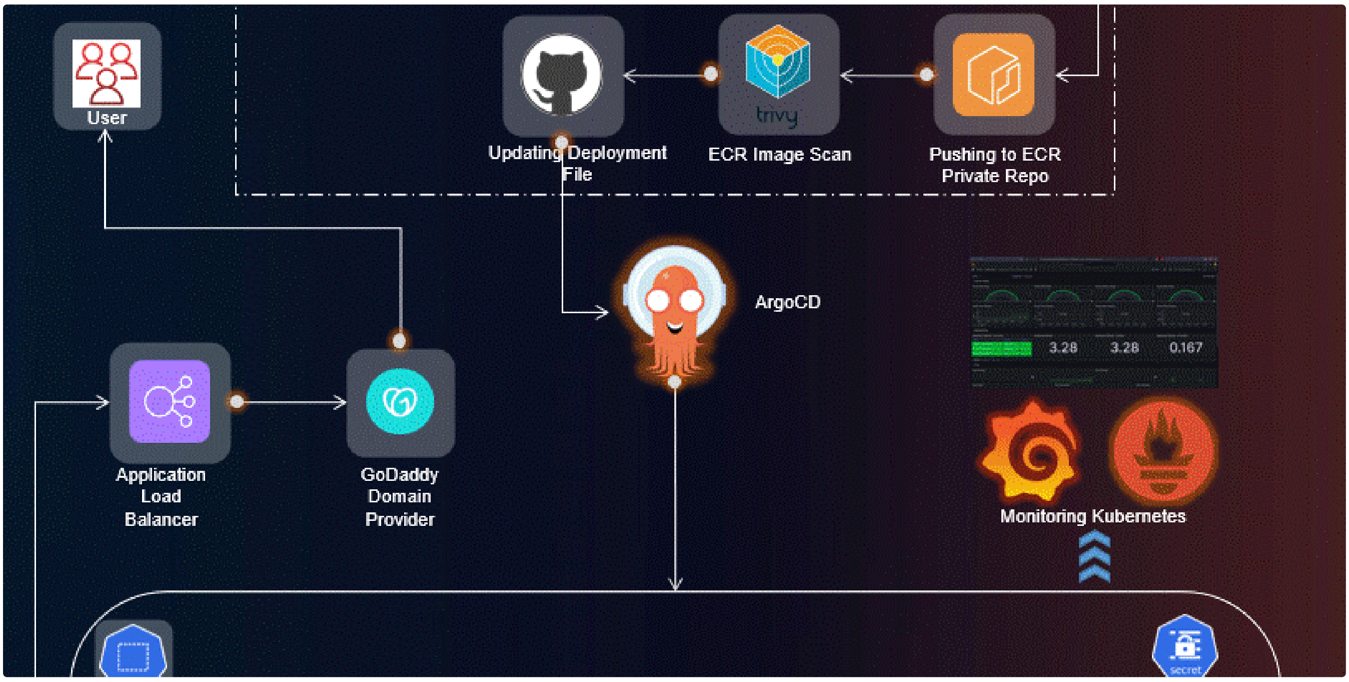
Welcome to Day 20 of our comprehensive 30-day course on Application Deployment! In this segment, we will delve into various deployment…

8 min read · Aug 3, 2023

2

See all from Navya Cloudops

Recommended from Medium



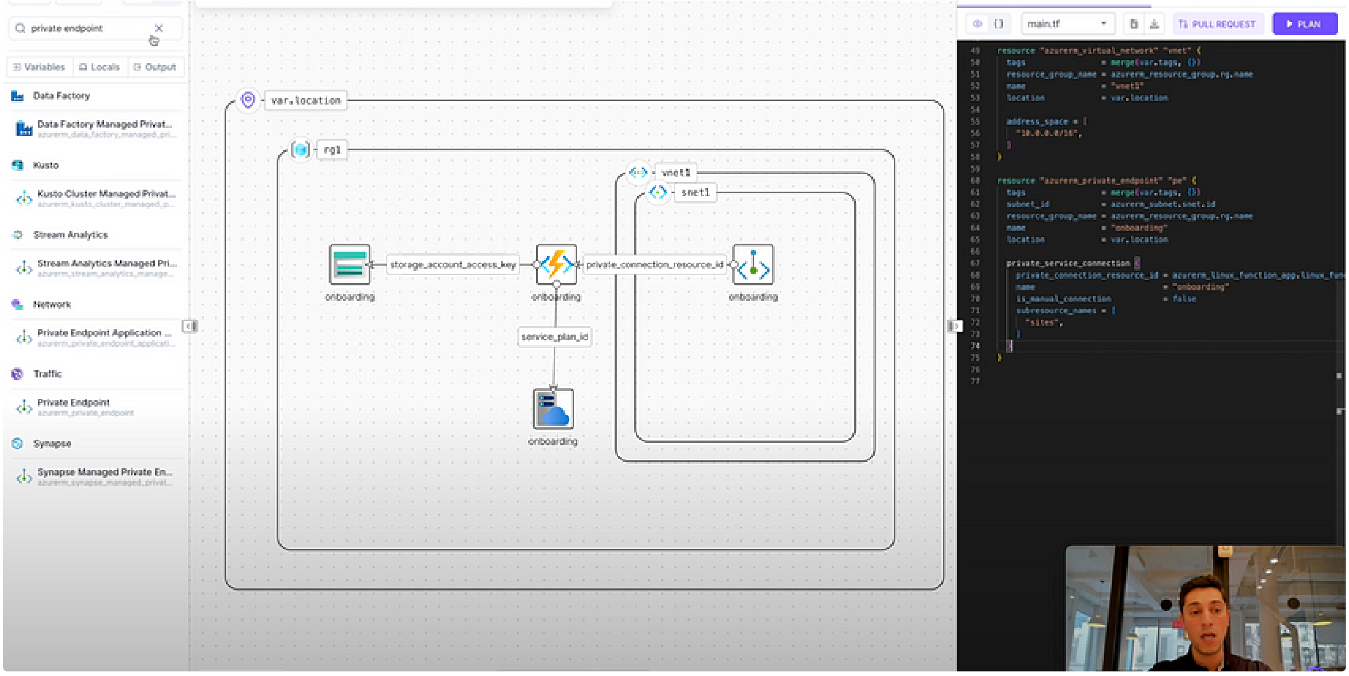
Aman Pathak in Stackademic

Advanced End-to-End DevSecOps Kubernetes Three-Tier Project using AWS EKS, ArgoCD, Prometheus...

Project Introduction:

23 min read · Jan 18, 2024

1.5K 15



```
resource "azurerm_virtual_network" "vnet" {
  tags                = merge(var.tags, {})
  resource_group_name = azurerm_resource_group.rg.name
  name                = "vnet1"
  location            = var.location

  address_space = [
    "10.0.0.0/16",
  ]
}

resource "azurerm_private_endpoint" "pe" {
  tags                = merge(var.tags, {})
  subnet_id           = azurerm_subnet.snet.id
  resource_group_name = azurerm_resource_group.rg.name
  name                = "onboarding"
  location            = var.location

  private_service_connection {
    private_connection_resource_id = azurerm_linux_function_app.linux_fun
    name                           = "onboarding"
    is_manual_connection           = false
    subresource_names = [
      "sites",
    ]
  }
}
```

Mike Tyson of the Cloud (MToC)

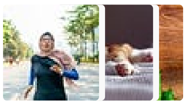
Complete Terraform Tutorial

Your journey in building a cloud infrastructure from scratch and learning Terraform with Brainboard starts here.

25 min read · Feb 8, 2024

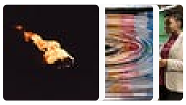
248

Lists



Self-Improvement 101

20 stories · 1364 saves

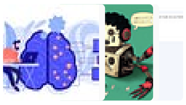


How to Find a Mentor

11 stories · 422 saves

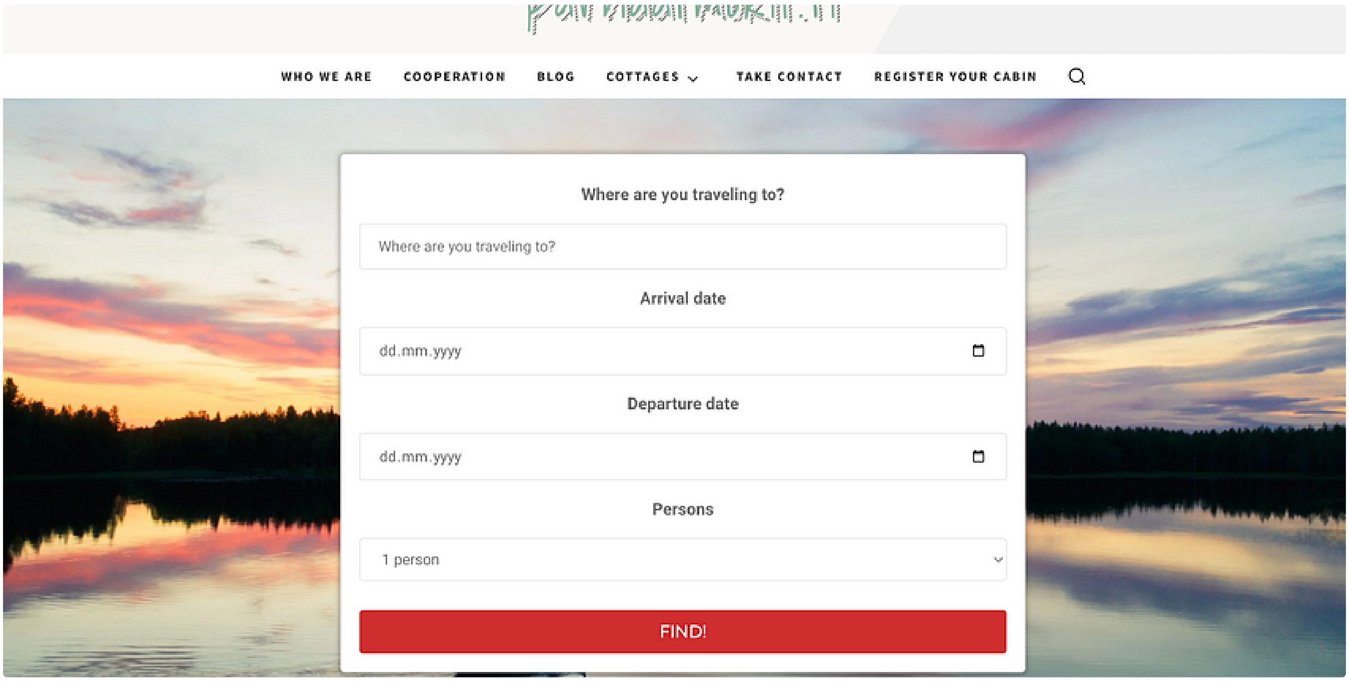
Good Product Thinking

11 stories · 470 saves



The New Chatbots: ChatGPT, Bard, and Beyond

12 stories · 307 saves



Where are you traveling to?

Where are you traveling to?

Arrival date

dd.mm.yyyy

Departure date

dd.mm.yyyy

Persons

1 person

FIND!

Artturi Jalli

I Built an App in 6 Hours that Makes $1,500/Mo

Copy my strategy!

✦ · 3 min read · Jan 23, 2024

9.2K 121

Chameera Dulanga in Bits and Pieces

Best-Practices for API Authorization

4 Best Practices for API Authorization

9 min read · Feb 6, 2024

1.1K 4

Akhilesh Mishra

Devops zero to hero #3—Everything you need to know about Dockers

A Complete Guide to start using Docker in your devops workflow

14 min read · Jan 23, 2024

308

Dolan Miu

Why have 100% Test Coverage

100% test coverage is somewhat of a taboo phrase in software. It’s “unachievable” with “diminishing returns” they would say. Non-developers...

4 min read · 3 days ago

53

See more recommendations