**Rollouts and Versioning in Deployment**

Whenever a new deployment is created or an existing deployment's container images are updated, a **rollout** is triggered. A rollout is the process of gradually deploying or upgrading application containers.

* When a deployment is first created, it initiates a rollout, generating **Deployment Revision 1**.
* If the application is later upgraded—such as updating the container version—a new rollout occurs, creating **Deployment Revision 2**.

This versioning system allows for **tracking changes** and provides the ability to **roll back** to a previous deployment if needed.

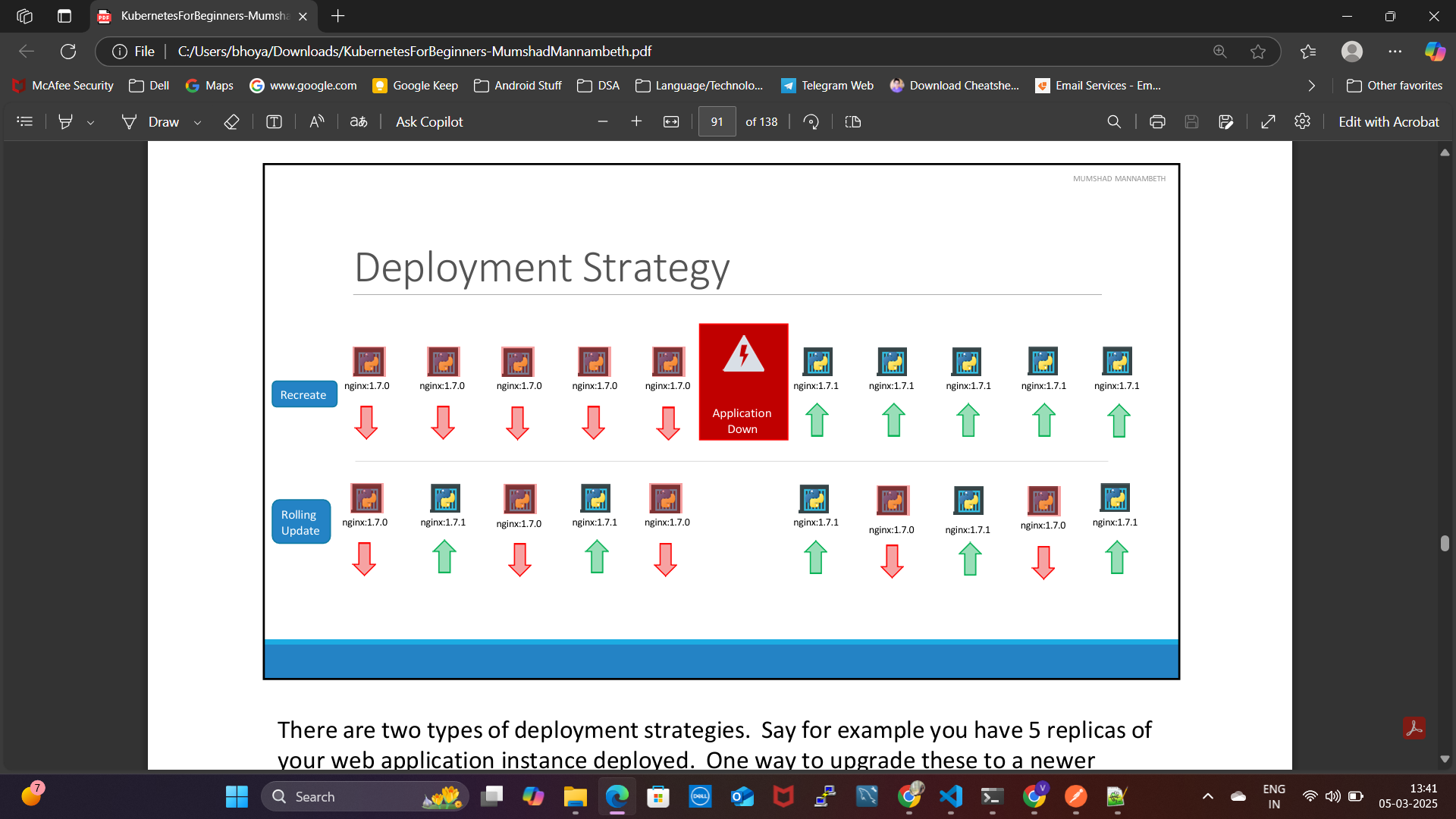
**Checking Rollout Status and History:**

* To check the current status of a rollout:

kubectl rollout status <deployment-name>

* To view the history of rollouts and their revisions:

kubectl rollout history <deployment-name>



**Deployment Strategies in Kubernetes**

Kubernetes supports two main deployment strategies for updating application instances.

**1. Recreate Strategy (Not the Default)**

Imagine you have **5 replicas** of your web application running. One way to upgrade them is to **delete all existing instances first** and then deploy new ones. This means:

1. Terminate all **5 running instances** of the current version.
2. Deploy **5 new instances** of the updated version.

The downside? During the transition, the application is completely **down** and inaccessible to users. This method is called the **Recreate strategy**, but **thankfully, it is NOT the default strategy** in Kubernetes.

**2. Rolling Update Strategy (Default)**

Instead of taking down all instances at once, this approach **gradually replaces old instances with new ones**, ensuring that the application remains available throughout the update.

* Older versions are **terminated one by one**, while new instances are brought up in parallel.
* This results in a **seamless upgrade** with zero downtime.

By default, if no strategy is specified, Kubernetes assumes **RollingUpdate** as the deployment strategy.

**Rolling Back to a Previous Version**

Kubernetes allows you to **rollback** to an earlier revision if needed. To undo the latest change, run:

kubectl rollout undo <deployment-name>

This command will **destroy the new pods** and bring back the older version from the previous ReplicaSet, restoring your application to its earlier state.

