ASDs

Detailed design

Product description

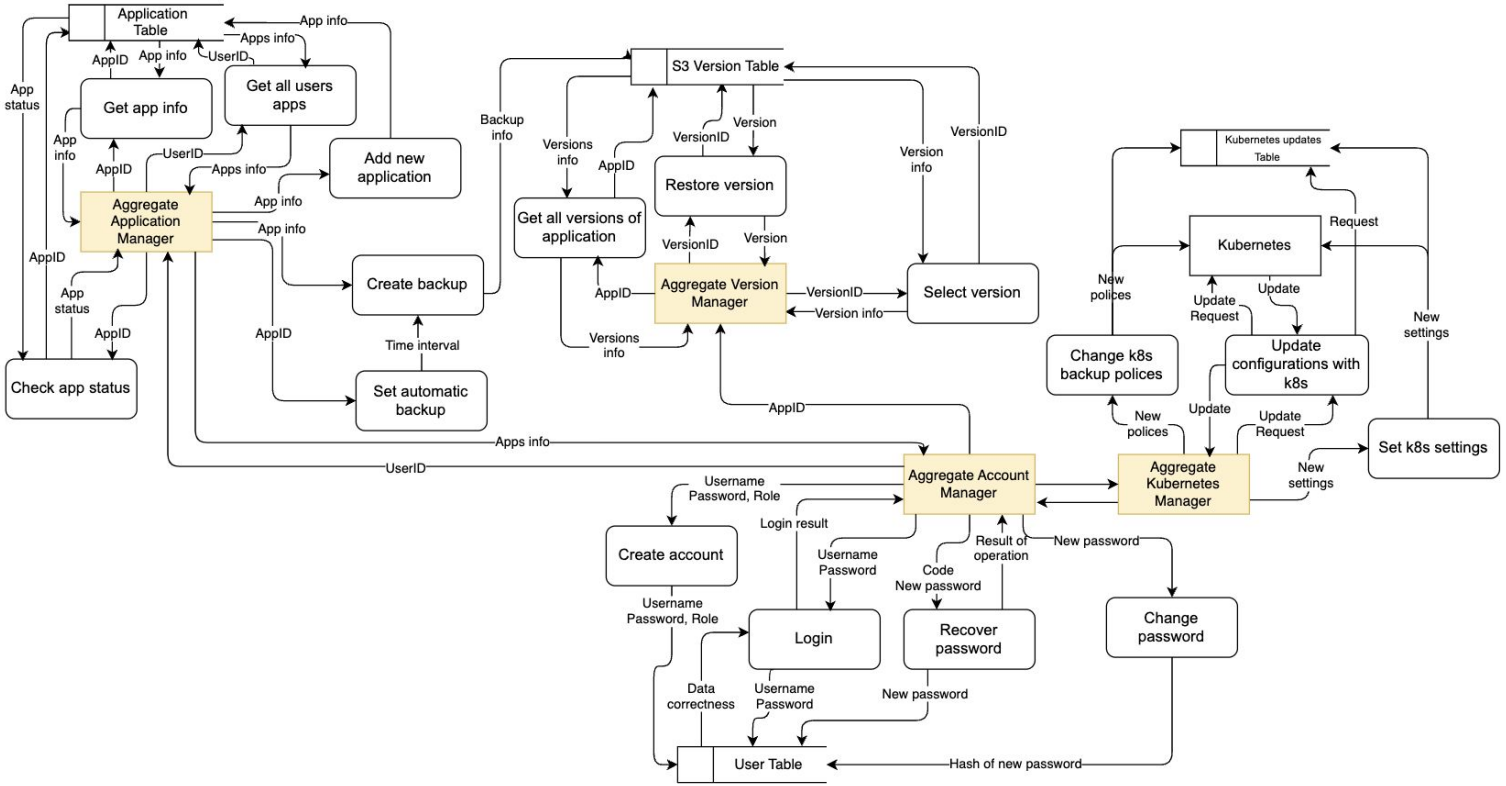
An application continuity service dynamically backs up and restores the state of applications running within the framework on K8s. The service uses the DSL of the framework to determine the relevant application state and its backup policies. Developers of an application are able to backup and restore a specific version from images and relevant state from the service. The application state is stored externally on the given S3-compatible object storage.

Team: Gorelyi Mikhail, Lukashin Daniil, Derezovskiy Ilya, Sigal Lev

Repo: https://github.com/gorelyi-code/advanced_software_designers

Report: a link to this slides within project repo/doc storage

System architecture



System architecture

Micro service architecture

Principle: The system should be designed as a set of independent microservices that interact via an API.

Rationale: Allows for scalability, flexibility, simplified deployment and fault isolation.

Division of Responsibility (SRP)

Principle: Each component or microservice should perform only one responsibility (Single Responsibility Principle).

Rationale: Simplifies testing, support, and modification of components.

Scalability and fault tolerance

Principle: The system should be designed to support horizontal scaling and minimize the impact of failures of one component on the entire system.

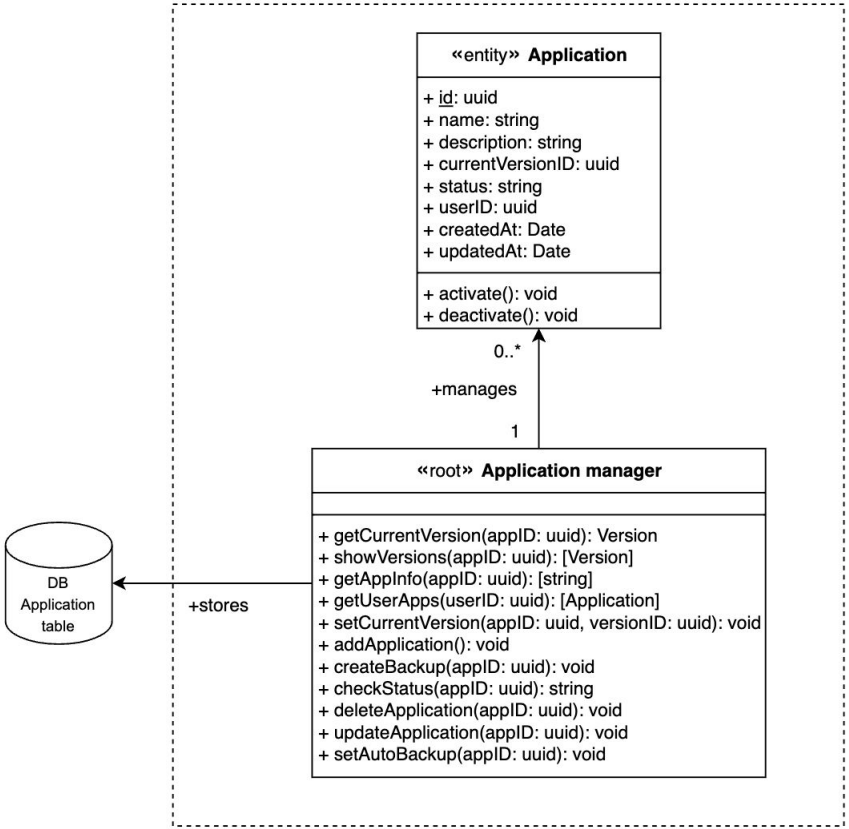
Justification: Takes into account the requirements for system performance and availability.

Design case Application Manager

Problem: Application Versioning: It is necessary to ensure that multiple versions of the same application are stored and managed so that users can choose whether to update or roll back to previous versions.

Solution: Facade

Application Manager provides a single access point for all operations on applications. This makes it easier to add new features, such as monitoring application activity.

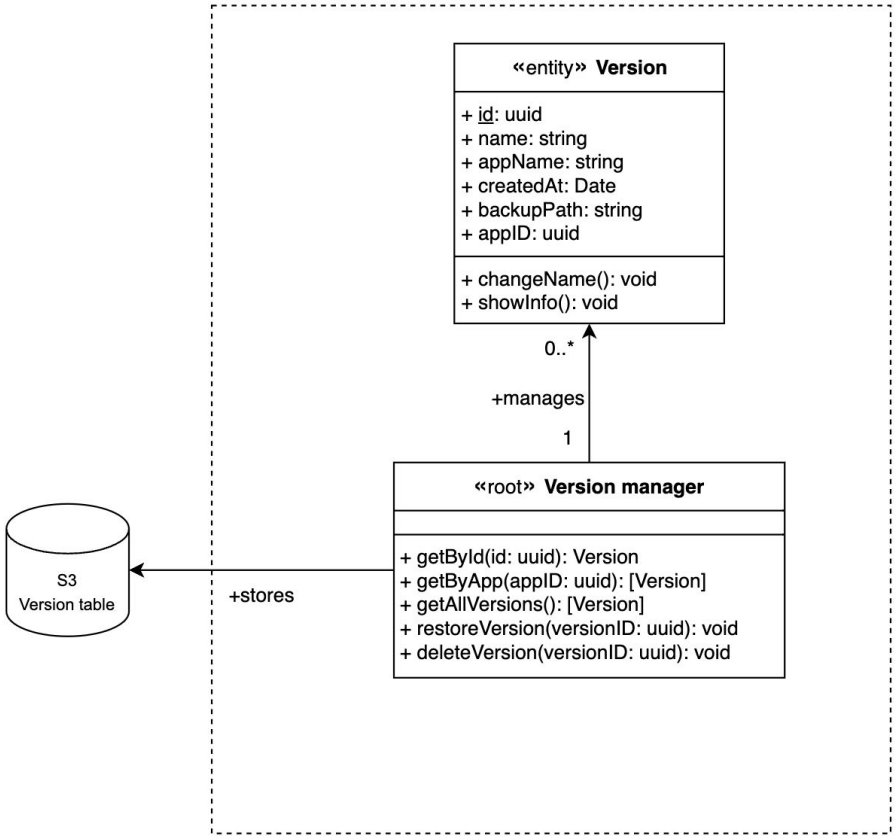


Design case Version Manager

Problem: Storage of data in distributed systems: Integration with cloud storage (for example, S3) is required for reliable storage of backup copies of versions.

Solution: Applied pattern: Adapter

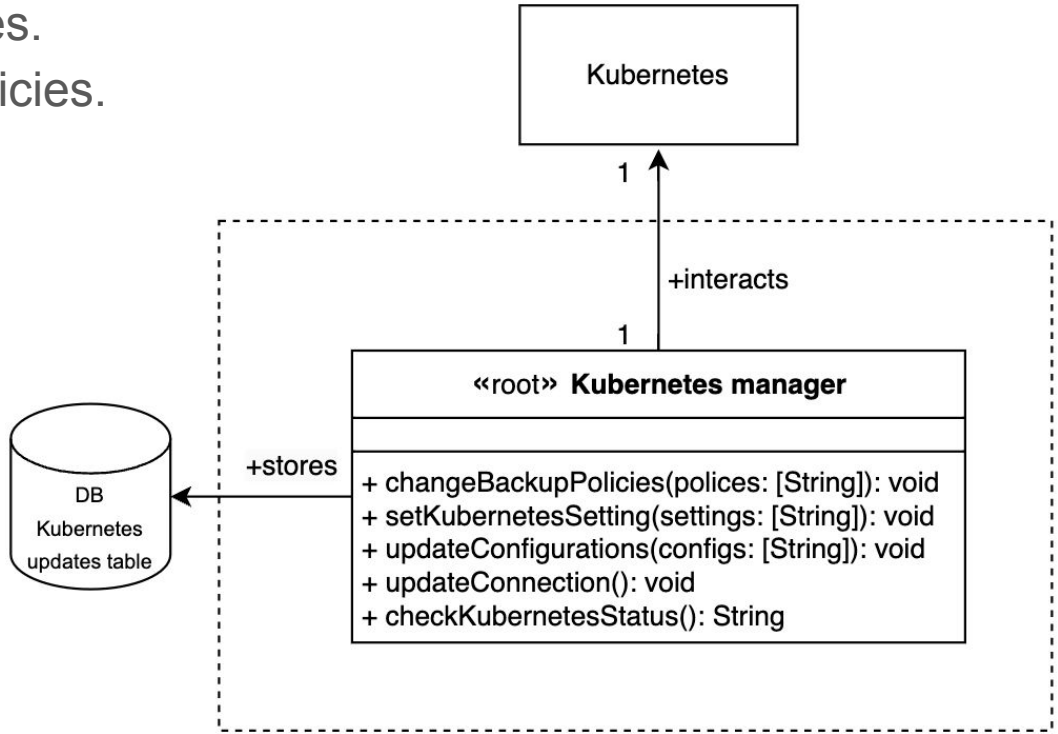
To interact with cloud storage, an adapter interface is used that abstracts the implementation details (for example, S3 API).



Design case Kubernetes Manager

Problem: Access to kubernetes.
Making changes. Updating policies.

Solution: API Gateway which abstracts the work with k8s



Design case Account Manager

Problem: Scaling difficulties: If a microservice is not able to scale effectively, this can lead to congestion, especially with a large number of requests.

Solution: CQRS (Command Query Responsibility Segregation) using the CQRS pattern will help to separate the logic of data recording