

ASDs

API Design

Product description

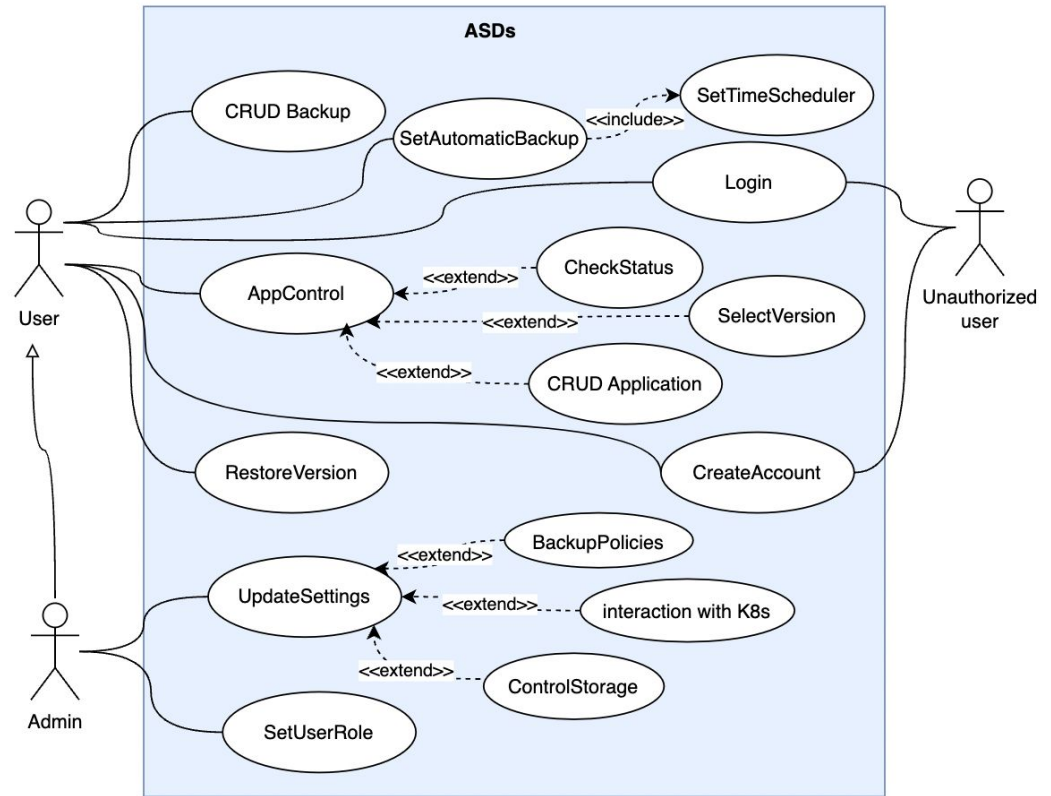
An application continuity service dynamically backs up and restores the state of applications running within the framework on K8s. The service uses the DSL of the framework to determine the relevant application state and its backup policies. Developers of an application are able to backup and restore 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, Derezhovskiy Ilya, Sigal Lev

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

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

Use case diagram

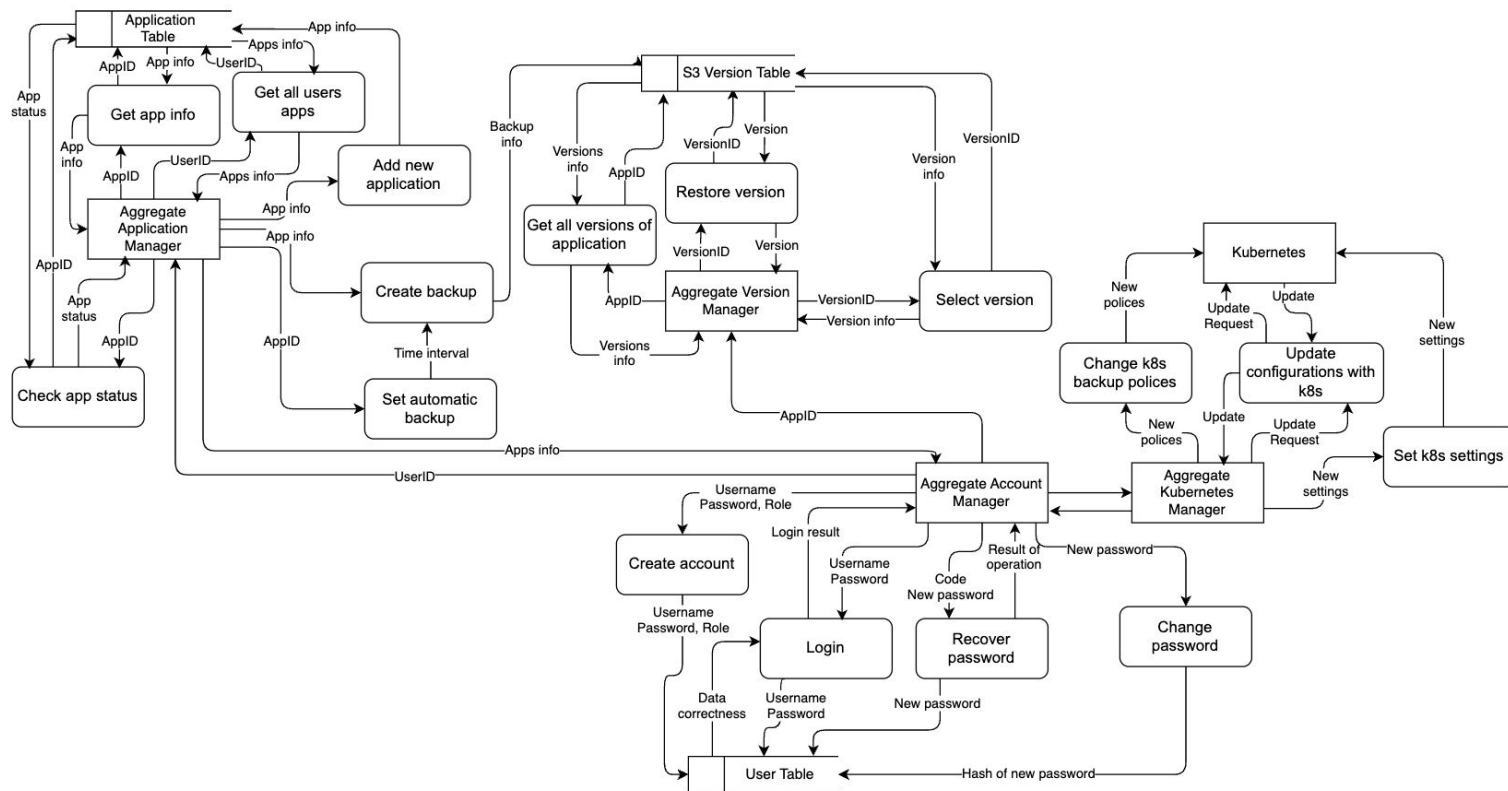


CRUD: create, read, update, and delete

Textual description in github:

https://github.com/gorelyi-code/advanced_software_designers/blob/main/Use%20case%20diagram/main_scenarios.md

Service diagram (DFD)



API usage Application Manager

Application Manager Operations about Application Manager



POST

/app Add a new application to the system



GET

/app/{appID} Get application info by appID



DELETE

/app/{appID} Deletes an application



GET

/app/{userID} Get application info by userID



GET

/app/status/{appID} Get application status by appID



POST

/app/backup Create a backup of application



PUT

/app/autobackup/{appID}/{timeInterval}
Update an autobackup timeinterval



API usage Version Manager

version_manager



PUT

/version/restoreVersion



GET

/version/selectVersion



DELETE

/version/deleteVersion



GET

/version/getAllVersionsOfApp



API usage Kubernetes Manager

Kubernetes Manager Managing kubernetes

**GET****/kubernetes/checkKubernetesStatus** Check state of kubernetes**PUT****/kubernetes/changeBackupPolicies** Update an existing backup policy**PUT****/kubernetes/setKubernetesSettings** Update kubernetes settings

https://github.com/gorelyi-code/advanced_software_designers/blob/main/hometasks/task_10/kubernetes_manager.yaml

API usage Account Manager

user_manager Operations about user



POST

/user/login Logs user into the system



GET

/user/logout Logs out current logged in user session



POST

/user/createNewAccount create new account



PUT

/user/changeUserRole Change user role



DELETE

/user/deleteAccount Delete user password



GET

/user/recoveryPassword Recovery user password



PUT

/user/changePassword Change user password



Solution stack (prepare)

Find an example implementation of a microservices application in the programming language chosen.
Specify one value for each option below

Implementation

- API definition OpenAPI
- Connection server for API python gunicorn
- App framework python FastAPI
- Serialization/state format json

Asynchronous interactions (optional)

- Message queue rabbitmq
- Messaging client library celery

Testing tools pytest

Operations

- App initializer systemd
- Code build makefile
- CI/CD pipeline gitlab
- Delivery method docker
- Logging & monitoring prometheus,
loki, grafana

Some references

<https://github.com/mfornos/awesome-microservices>

<https://awesomeopensource.com/projects/microservices-architecture>

<https://www.redhat.com/en/blog/comparing-openapi-grpc>

<https://cloud.google.com/apis/design/resources>