Report SUPD

Microservice 1

## Design decisions

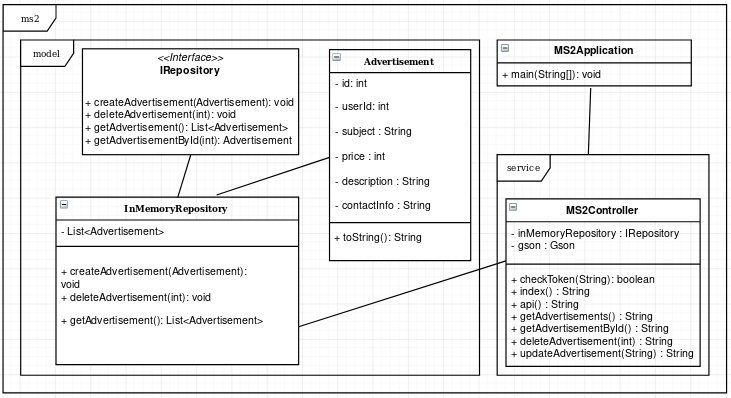
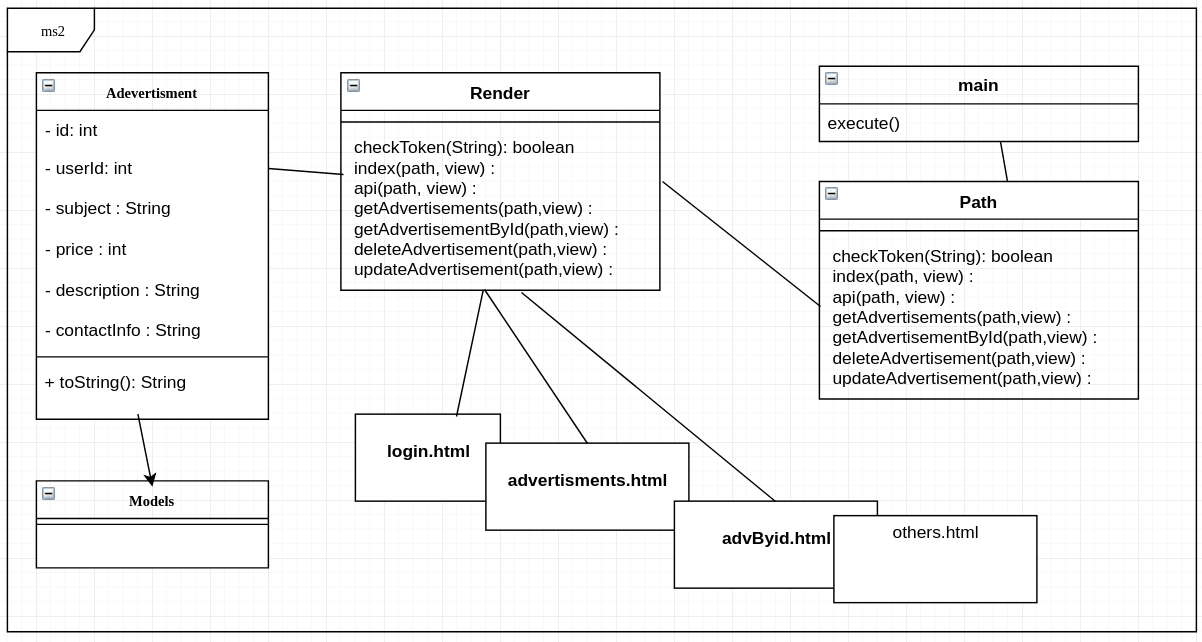
## Deployment

## Testing and presentation

## Status

Microservice 2

## Design decisions

* 1. 
  2. Picture 1 – UML\_1 (current implementation)
  3. 
  4. Picture 2 – UML\_2 (possible implementation)
  5. The second UML diagram shows another possible sollution using Django Framework and Python language with server side rendering of frontend interface. It will also use Sqlite dabase to save data of model definitions. Models are written with the help of built in objects-relational mapper, that will transform them in proper sql commands.
  6. As we learn mostly Java I decided to use java for the implementation of ms2. Spring Boot web project with maven is used. REST API architecture is used to communicate with other services. Information from ms2 will be send in JSON format. In case of object store I decided for in-memory- store. Each reboot of the server will delete all the data. Other possible solution were using DB integration or Java serialization.
  7. With request (create, update, delete, show all for current user) from the client ms2 will check the session token sent with the request, by sending request to authentication service. If everything is OK, the client will get the JSON data. Request of all available advertisements and one will not need any token.
  8. ms2 will be available on following paths:
  9. api/advertisements
  10. api/advertisements/{id}
  11. api/{user\_id}/advertisements
  12. api/{user\_id}/advertisements/{id}
  13. api/{user\_id}/create
  14. api/{user\_id}/update/{id}
  15. api/{user\_id}/delete/{id}

## Deployment

As we are using VPN all our services will be started on personal laptops with exposed ports to listen requests. In my ms2 it will be 8080 port. VPN connection will be needed every time to override possible firewall and proxy problems. Server is available on 10.101.104.13:8080.

## Testing and presentation

* 1. Current test were done manually without any constraints or JUnit tests. Coming soon… For test purposes Postman was used to send simple GET requests. On presentation the hole project will run in VPN network with working requests and responses between microservices.

## Status

* 1. Git workflow accepted and learned by all teammates.
  2. Data model is done without constraints.
  3. Creating objects is hardcoded for test purposes.
  4. Response is done in JSON format.
  5. Connection between services and simple requests are working already.
  6. Following paths are working, when service is started:
  7. 10.101.104.13:8080/api/advertisements
  8. 10.101.104.13:8080/api/advertisements/{id}
  9. TODO: make all paths working, work with token, request data validation, unit tests.

Microservice 4

## Design decisions

## Deployment

## Testing and presentation

## Status

API specification

### Show Advertisements

Returns json data about all available advertisements

* URL /api/advertisements
* Method: GET
* URL Params None
* Data Params None
* Success Response:
  + Code: 200   
    Content: [{"id":0,"userId":2,"subject":"first","price":50,"description":"computer nice","contactInfo":"tel. 473823748"},{"id":1,"userId":2,"subject":"second","price":50,"description":"computer nice","contactInfo":"tel. 473823748"}]
* Error Response:

Coming soon...

### Show Advertisement by ID

Returns json data about one specific advertisement

* URL /api/advertisements/{id}
* Method: GET
* URL Params

**Required**: id=[integer]

* Data Params None
* Success Response:
  + Code: 200   
    Content: {"id":1,"userId":2,"subject":"second","price":50,"description":"computer nice","contactInfo":"tel. 473823748"}
* Error Response:

Coming soon...

### Show user advertisements

Coming soon...

### Show user advertisement by ID

Coming soon...

### Create Advertisement

Coming soon...

### Update Advertisement by ID

Coming soon...

### Delete Advertisement by ID

Coming soon...