Messenger

David Röbl, Felix Boxleitner,
Sara Dávila Méndez

System Architecture

- **Server:** Heroku
- **Webserver:** Spring
- **Persistence:** JPA (Hibernate)
- **DB:** PostgreSQL
- **Doc:** Swagger
- **Testbed:** Swagger

REST interface

Resource	POST	GET	DELETE
/users	creates a new user and returns the user object	returns a list containing all users	_
/messages	creates a new message object and sends it to the recipients	returns a list of all messages that have the current user as recipient	_
/messages/{id}	_	gets message with given id updates read_by!!	deletes message with given id

API URL: https://cc-ex2-messenger-105-7.herokuapp.com

Code Snippet I

```
@ManyToMany
@JoinTable(
    name="MESSAGE_RECIPIENT_USERS",
    joinColumns =
        @JoinColumn(name="msgid"),
    inverseJoinColumns =
        @JoinColumn(name="userid")
)
```

describes a the relation between message recipients and users (JPA annotations)

Code Snippet II - /messages call - GET

```
boolean unread; // url parameter unread set or not
String authToken; // authorization token passed in header
DatabaseConnection dbC = DatabaseConnection.getInstance();
List<Message> messages = null;
try {
    messages = dbC.getMessages(authToken, unread);
} catch (DatabaseConnection.UnauthorizedException e) {
    return new ResponseEntity<Messages>(HttpStatus.valueOf(400));
}
IdsWrapper idsWrapper = new IdsWrapper();
for(Message message : messages){
    IdWrapper id = new IdWrapper();
    id.setId(message.getId());
    idsWrapper.addIdsItem(id);
}
return new ResponseEntity<IdsWrapper>(idsWrapper, HttpStatus.OK);
```

Code Snippet III - retrieving messages from DB

```
boolean unread; //filter by unread messages
String authToken; //authorization token of user
Session session = sessionFactory.openSession();
DBUser user = getDBUser(session, authToken);
if(null == user){
   session.close();
   throw new UnauthorizedException();
}
List<Message> messageList = new ArrayList<>();
List<DBMessage> messages = getDBMessages(session);
for(DBMessage message : messages){
   if(message.getRecipients().contains(user)){
       if(!unread || !message.getReadBy().contains(user)){
           messageList.add(message.toMessage());
       }
   }
}
session.close();
return messageList;
```

Lessons learned

- API configuration with swagger takes long
- swagger can generate server code
- if you let swagger generate code that does not work with heroku, it won’t work
- JPA is only nice if you know what you’re doing
    - transactions!!!

Live Demo

With Swagger

https://cc-ex2-messenger-105-7.herokuapp.com

Q&A

Thanks for your attention!