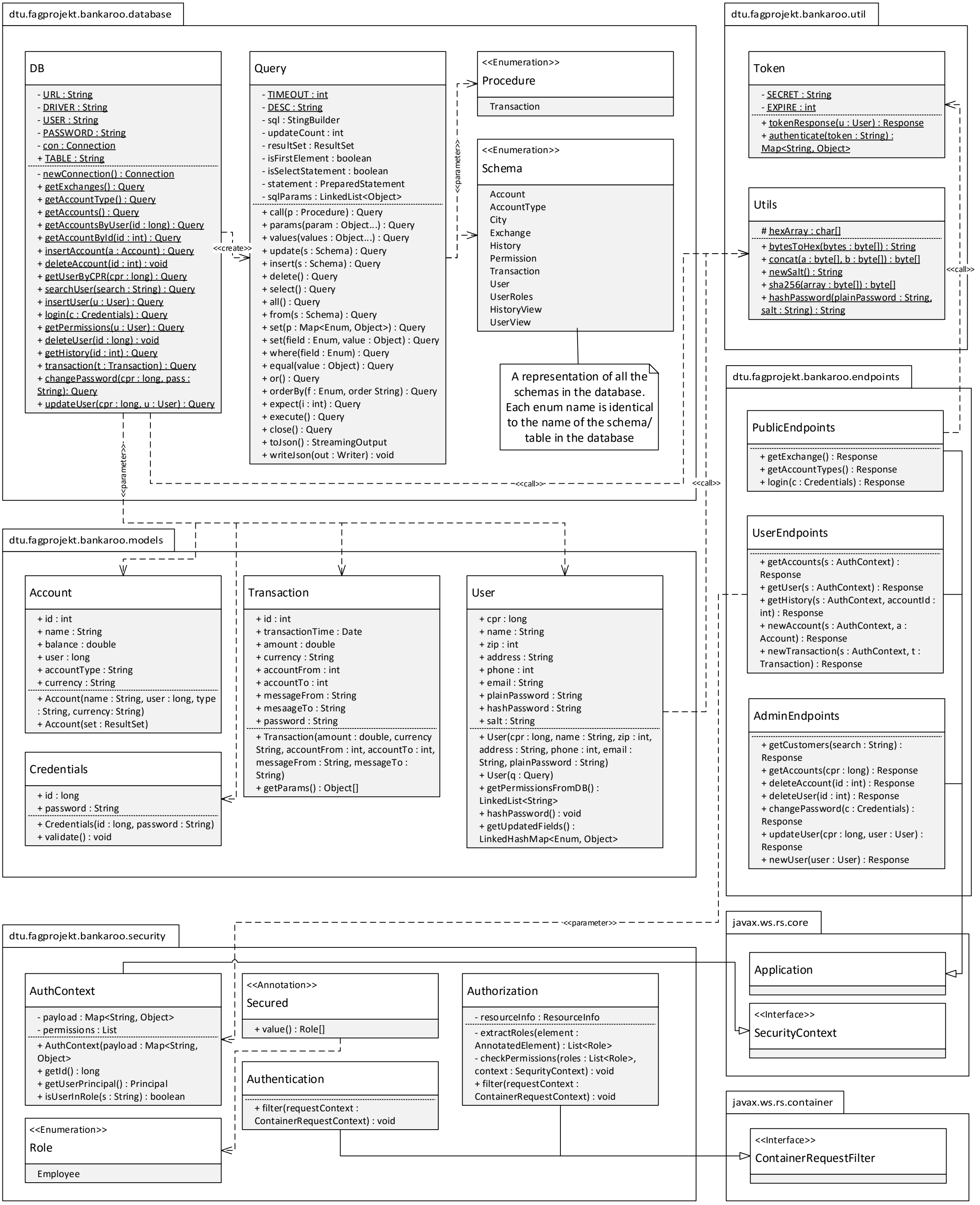
dtu.fagprojekt.bankaroo.database
DB
- URL : String
- DRIVER : String
- USER : String
- PASSWORD : String
- con : Connection
+ TABLE : String
- newConnection() : Connection
+ getExchanges() : Query
+ getAccountType() : Query
+ getAccounts() : Query
+ getAccountsByUser(id : long) : Query
+ getAccountById(id : int) : Query
+ insertAccount(a : Account) : Query
+ deleteAccount(id : int) : void
+ getUserByCPR(cpr : long) : Query
+ searchUser(search : String) : Query
+ insertUser(u : User) : Query
+ login(c : Credentials) : Query
+ getPermissions(u : User) : Query
+ deleteUser(id : long) : void
+ getHistory(id : int) : Query
+ transaction(t : Transaction) : Query
+ changePassword(cpr : long, pass : String): Query
+ updateUser(cpr : long, u : User) : Query
<<create>>
Query
- TIMEOUT : int
- DESC : String
- sql : StingBuilder
- updateCount : int
- resultSet : ResultSet
- isFirstElement : boolean
- isSelectStatement : boolean
- statement : PreparedStatement
- sqlParams : LinkedList<Object>
+ call(p : Procedure) : Query
+ params(param : Object...) : Query
+ values(values : Object...) : Query
+ update(s : Schema) : Query
+ insert(s : Schema) : Query
+ delete() : Query
+ select() : Query
+ all() : Query
+ from(s : Schema) : Query
+ set(p : Map<Enum, Object>) : Query
+ set(field : Enum, value : Object) : Query
+ where(field : Enum) : Query
+ equal(value : Object) : Query
+ or() : Query
+ orderBy(f : Enum, order String) : Query
+ expect(i : int) : Query
+ execute() : Query
+ close() : Query
+ toJson() : StreamingOutput
+ writeJson(out : Writer) : void
<<parameter>>
<<Enumeration>>
Procedure
Transaction
<<Enumeration>>
Schema
Account
AccountType
City
Exchange
History
Permission
Transaction
User
UserRoles
HistoryView
UserView
A representation of all the schemas in the database. Each enum name is identical to the name of the schema/ table in the database
<<call>>
<<call>>
<<parameter>>
dtu.fagprojekt.bankaroo.util
Token
- SECRET : String
- EXPIRE : int
+ tokenResponse(u : User) : Response
+ authenticate(token : String) : Map<String, Object>
Utils
hexArray : char[]
+ bytesToHex(bytes : byte[]) : String
+ concat(a : byte[], b : byte[]) : byte[]
+ newSalt() : String
+ sha256(array : byte[]) : byte[]
+ hashPassword(plainPassword : String, salt : String) : String
<<call>>
dtu.fagprojekt.bankaroo.endpoints
PublicEndpoints
+ getExchange() : Response
+ getAccountTypes() : Response
+ login(c : Credentials) : Response
UserEndpoints
+ getAccounts(s : AuthContext) : Response
+ getUser(s : AuthContext) : Response
+ getHistory(s : AuthContext, accountId : int) : Response
+ newAccount(s : AuthContext, a : Account) : Response
+ newTransaction(s : AuthContext, t : Transaction) : Response
AdminEndpoints
+ getCustomers(search : String) : Response
+ getAccounts(cpr : long) : Response
+ deleteAccount(id : int) : Response
+ deleteUser(id : int) : Response
+ changePassword(c : Credentials) : Response
+ updateUser(cpr : long, user : User) : Response
+ newUser(user : User) : Response
dtu.fagprojekt.bankaroo.models
Account
+ id : int
+ name : String
+ balance : double
+ user : long
+ accountType : String
+ currency : String
+ Account(name : String, user : long, type : String, currency: String)
+ Account(set : ResultSet)
Transaction
+ id : int
+ transactionTime : Date
+ amount : double
+ currency : String
+ accountFrom : int
+ accountTo : int
+ messageFrom : String
+ mesaageTo : String
+ password : String
+ Transaction(amount : double, currency String, accountFrom : int, accountTo : int, messageFrom : String, messageTo : String)
+ getParams() : Object[]
User
+ cpr : long
+ name : String
+ zip : int
+ address : String
+ phone : int
+ email : String
+ plainPassword : String
+ hashPassword : String
+ salt : String
+ User(cpr : long, name : String, zip : int, address : String, phone : int, email : String, plainPassword : String)
+ User(q : Query)
+ getPermissionsFromDB() : LinkedList<String>
+ hashPassword() : void
+ getUpdatedFields() : LinkedHashMap<Enum, Object>
Credentials
+ id : long
+ password : String
+ Credentials(id : long, password : String)
+ validate() : void
<<parameter>>
dtu.fagprojekt.bankaroo.security
AuthContext
- payload : Map<String, Object>
- permissions : List
+ AuthContext(payload : Map<String, Object>
+ getId() : long
+ getUserPrincipal() : Principal
+ isUserInRole(s : String) : boolean
<<Annotation>>
Secured
+ value() : Role[]
Authentication
+ filter(requestContext : ContainerRequestContext) : void
Authorization
- resourceInfo : ResourceInfo
- extractRoles(element : AnnotatedElement) : List<Role>
- checkPermissions(roles : List<Role>, context : SequrityContext) : void
+ filter(requestContext : ContainerRequestContext) : void
<<Enumeration>>
Role
Employee
javax.ws.rs.core
Application
<<Interface>>
SecurityContext
javax.ws.rs.container
<<Interface>>
ContainerRequestFilter