

```

1 {-# LANGUAGE OverloadedStrings #-}
2 {-|
3   This is the Haskell implementation of the Datatypes
4   described in the UML-class-diagram from storedData.pdf.
5   It is also converted into a .pdf and included into storedData.pdf.
6 -|}
7 module OpenBrain.Data where
8
9 import Data.Aeson ((.=), ToJSON(..), object)
10 import Data.Function (on)
11 import Happstack.Server (FromReqURI(..))
12 import qualified Data.Aeson as Aeson
13
14 import OpenBrain.Data.Id
15 import OpenBrain.Data.Hash
16 import OpenBrain.Data.Json
17 import OpenBrain.Data.Salt
18
19 data Description = Description {
20     descriptionId :: DescriptionId
21 , author        :: Author
22 , headline      :: Headline
23 , description   :: String
24 , creationTime  :: Timestamp
25 , deletionTime  :: Timestamp
26 } deriving (Show)
27
28 data Article = Article {
29     articleId    :: ArticleId
30 , content       :: String
31 , children      :: [ArticleId]
32 , aDescription  :: Description
33 } deriving (Show)
34
35 data Relation = Relation {
36     relationId   :: RelationId
37 , source        :: ArticleId
38 , target        :: ArticleId
39 , rType         :: RelationType
40 , rDescription  :: Description
41 } deriving (Show)
42
43 data RelationType = RelationAttack | RelationDefense
44     deriving (Show, Read, Eq, Ord, Enum)
45
46 data Collection = Collection {
47     collectionId :: CollectionId
48 , articles      :: [ArticleId]
49 , cDescription  :: Description
50 } deriving (Show)
51
52 data Discussion = Discussion {
53     discussionId :: DiscussionId
54 , participants  :: [UserId]
55 , deadline      :: Timestamp
56 , weights       :: [(UserId, Weight, RelationId)]
57 , result        :: Maybe Result
58 , dCollection   :: Collection
59 } deriving (Show)
60
61 data Result = Result {
62     resultId     :: ResultId
63 , choices       :: [(CollectionId, Votes)]
64 , voters        :: [(UserId, Voted)]
65 } deriving (Show)
66
67 data User = User {
68     userId       :: UserId
69 , username      :: String
70 , userhash      :: Hash

```

```

71 , userSalt      :: Salt
72 , userCreation :: Timestamp
73 , lastLogin    :: Timestamp
74 , isAdmin      :: Bool
75 , profile      :: Maybe ArticleId
76 , session      :: Maybe SessionKey
77 } deriving (Show)
78
79 {-| Type aliases: |-}
80 type Author      = UserId
81 type Count       = Int
82 type Headline    = String
83 type Heir        = UserId
84 type IsAdmin     = Bool
85 type Limit       = Int
86 type Offset      = Int
87 type SessionKey  = String
88 type Timestamp   = String
89 type Username    = String
90 type Voted       = Bool
91 type Votes       = Int
92 type Weight      = Int
93
94 {-| Instances of Eq: |-}
95 instance Eq Description where
96     (==) = (==) `on` descriptionId
97 instance Eq Article where
98     (==) = (==) `on` articleId
99 instance Eq Relation where
100     (==) = (==) `on` relationId
101 instance Eq Collection where
102     (==) = (==) `on` collectionId
103 instance Eq Discussion where
104     (==) = (==) `on` discussionId
105 instance Eq Result where
106     (==) = (==) `on` resultId
107 instance Eq User where
108     (==) = (==) `on` userId
109
110 {-| Instances of Ord: |-}
111 instance Ord Description where
112     compare = compare `on` descriptionId
113 instance Ord Article where
114     compare = compare `on` articleId
115 instance Ord Relation where
116     compare = compare `on` relationId
117 instance Ord Collection where
118     compare = compare `on` collectionId
119 instance Ord Discussion where
120     compare = compare `on` discussionId
121 instance Ord Result where
122     compare = compare `on` resultId
123 instance Ord User where
124     compare = compare `on` userId
125
126 {-| Instances of ToJSON: |-}
127 instance ToJSON Description where
128     toJSON d = object [
129         "descriptionId" .= descriptionId d
130         , "author"       .= author       d
131         , "headline"    .= headline     d
132         , "description"  .= description  d
133         , "creationTime" .= creationTime d
134         , "deletionTime" .= deletionTime d
135     ]
136 instance ToJSON Article where
137     toJSON a = merge (toJSON $ aDescription a) o
138     where
139         o = object [
140             "articleId" .= articleId a

```

```

141         , "content"      .= content    a
142         , "children"     .= children   a
143     ]
144 instance ToJSON Relation where
145     toJSON r = merge (toJSON $ rDescription r) o
146     where
147         o = object [
148             "relationId" .= relationId r
149             , "source"    .= source     r
150             , "target"    .= target     r
151             , "rType"     .= rType      r
152         ]
153 instance ToJSON RelationType where
154     toJSON = toJSON . show
155 instance ToJSON Collection where
156     toJSON c = merge (toJSON $ cDescription c) o
157     where
158         o = object ["collectionId" .= collectionId c, "articles" .= articles c]
159 instance ToJSON Discussion where
160     toJSON d = merge (toJSON $ dCollection d) o
161     where
162         o = object [
163             "discussionId" .= discussionId d
164             , "participants" .= participants d
165             , "deadline"    .= deadline    d
166             , "weights"     .= weights     d
167             , "result"      .= result      d
168         ]
169 instance ToJSON Result where
170     toJSON r = object [
171         "resultId" .= resultId r
172         , "choices" .= choices  r
173         , "voters"  .= voters   r
174     ]
175 instance ToJSON User where
176     toJSON u = object [
177         "userId"      .= userId      u
178         , "username"  .= username    u
179         , "userCreation" .= userCreation u
180         , "lastLogin" .= lastLogin   u
181         , "isAdmin"   .= isAdmin     u
182         , "profile"   .= profile     u
183     ]
184 {-| Instances of FromReqURI |-}
185 instance FromReqURI RelationType where
186     fromReqURI "RelationAttack" = Just RelationAttack
187     fromReqURI "RelationDefense" = Just RelationDefense
188     fromReqURI _ = Nothing

```