<<interface>>
@SIngleton PasswordKeeper

+ synchronized exists(String user, String password) : boolean
+ synchronized register(String user, String password) : boolean

---

AuthResource

- PasswordKeeper passwordKeeper
- HashManager hashManager

+ isLoggedIn(String username, byte[] ip, byte[] hash) : boolean
+ authorize(String username, String password) : byte[]
+ register(String username, String password) : boolean
+ active(String username, byte[] ip, byte[] hash) : bool

@Singleton AuthService

- @EJB AuthResource authResource

+ @POST authRequest(String json) : Response
+ @POST regRequest(String json) : Response

@Singleton UnencryptedPasswordKeeper

- credentialsFile : FileOutputStream
- userToPassword : Map<String, String>

+ constructor(String filename)
+ @Override synchronized exists(String user, String password) : boolean
+ @Override synchronized register(String user, String password) : boolean

---

Database

+ attribute1:type = defaultValue
+ attribute2:type
- attribute3:type

+ operation1(params):returnType
- operation2(params)
- operation3()

<<interface>>
@Singleton HashManager

+ generateHash(String username, byte[] ip) : byte[]
+ prolongHash(String username, byte[] ip, byte[] hash)
+ isLoggedIn(String username, byte[] ip, byte[] hash) : LoginState
+ expirationPeriodMinutes() : int
+ shutdown()

@Singleton OneToOneHashManager

- Map<LoginInstance,List<Byte>> userToHash;
- Map<List<Byte>,LoginInstance> hashToUser;
- Map<List<Byte>,ScheduledFuture<?>> hashToTask;
- Random random;
- int waitTimeMinutes;
- ScheduledExecutorServicetimerKeeper;

+ constructor(int waitTimeMinutes)
+ @Override generateHash(String username, byte[] ip) : byte[]
+ @Override prolongHash(String username, byte[] ip, byte[] hash)
+ @Override isLoggedIn(String username, byte[] ip, byte[] hash) : LoginState
+ @Override expirationPeriodMinutes() : int
+ @Override shutdown()