**Documentatie Back End - Calendar**

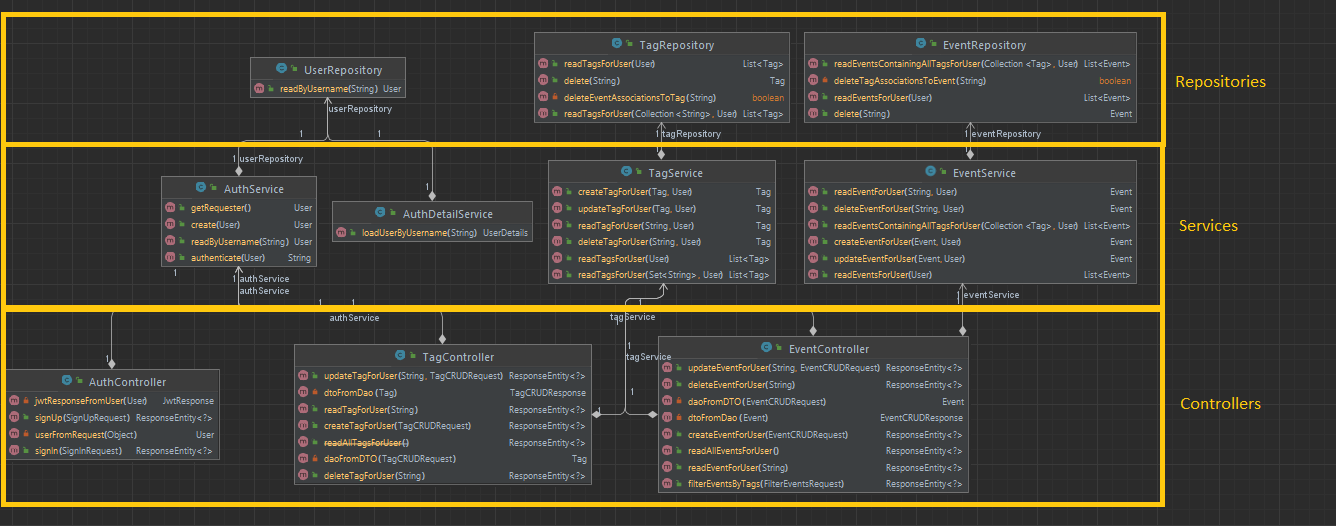
**Proiect colectiv**

**Tehnologii:**

* Back End-ul aplicatiei este un server web scris in Java, construit utilizand Spring Framework
* Serverul web serveste clientii prin intermediul REST controllers
* Autentificarea si autorizarea este realizata utilizand JWT si support-ul Spring pentru Security
* Persistenta datelor se realizeaza utilizand o baza de date PostgreSQL impreuna cu Hibernate framework

**Arhitectura:**

* Aplicatia noastra prezinta 3 layere ce comunica top-down:
  + Controllers – puntea de legatura ce utilizeaza REST endpoints, acestia sunt responsabili de conversiile dto-urilor in dao-uri si vice versa
  + Services – ce au ca scop rezolvarea unor probleme specifice domeniului propriu – astfel lucreaza cu dao-uri
  + Repositories – layer-ul de persistenta in care se realizeaza accesul la enitati, realizand inclusiv filtrarile
* Diagrama de clase:



* Pe langa arhitectura de prezentata mai sus, construim peste arhitectura existenta Spring, adauand propriul filtru pentru realizarea autorizarii userilor

**Configurare:**

* Confiugrarea se realizeaza utilizand clase speciale si mecanismul Spring pentru configuratii

**Controller Advice:**

* Pentru prinderea exceptiilor netratate am folosit un controller advice ce trateaza orice eroare nu este prinsa pe parcurs

**Model:**

* Modelul este format din obiecte de tip DTO, ce se folosesc pentru comunicarea cu exteriorul aplicatiei la nivelul REST Controllers, si obiecte de tip DAO, pentru comunicarea cu persistenta
* Mapparea dintre DTO si DAO se realizeaza utilizand
* DAO:
  + Toate entitatile prezinta un UUID generat de persistenta
  + DAO-urile sunt mapate la persistenta utilizand ORM-uri, mai exact Hibernate
  + Persistenta este reprezentata de o baza de date PostgreSQL
  + Entitatile de baza sunt User, Event si Tag
  + Relatii:
    - Un user detine mai multe tag-uri si event-uri
    - Un event e detinut de un user si este descris de mai multe tag-uri
    - Un tag e detinut de un user si descrie mai multe event-uri
* DTO:
  + Se impart in 2 categorii: request si response
  + S-au identiicat categoriile in care pica fiecare request/response si am incercat sa nu refolosim DTO-urile pentru a evita crearea de dependente intre request-uri care nu au legatura

**Exemple call-uri:**

* Toate request-urile sunt exemplificate intr-o colectie Postman atasata in proiect in directorul “documentation/api”