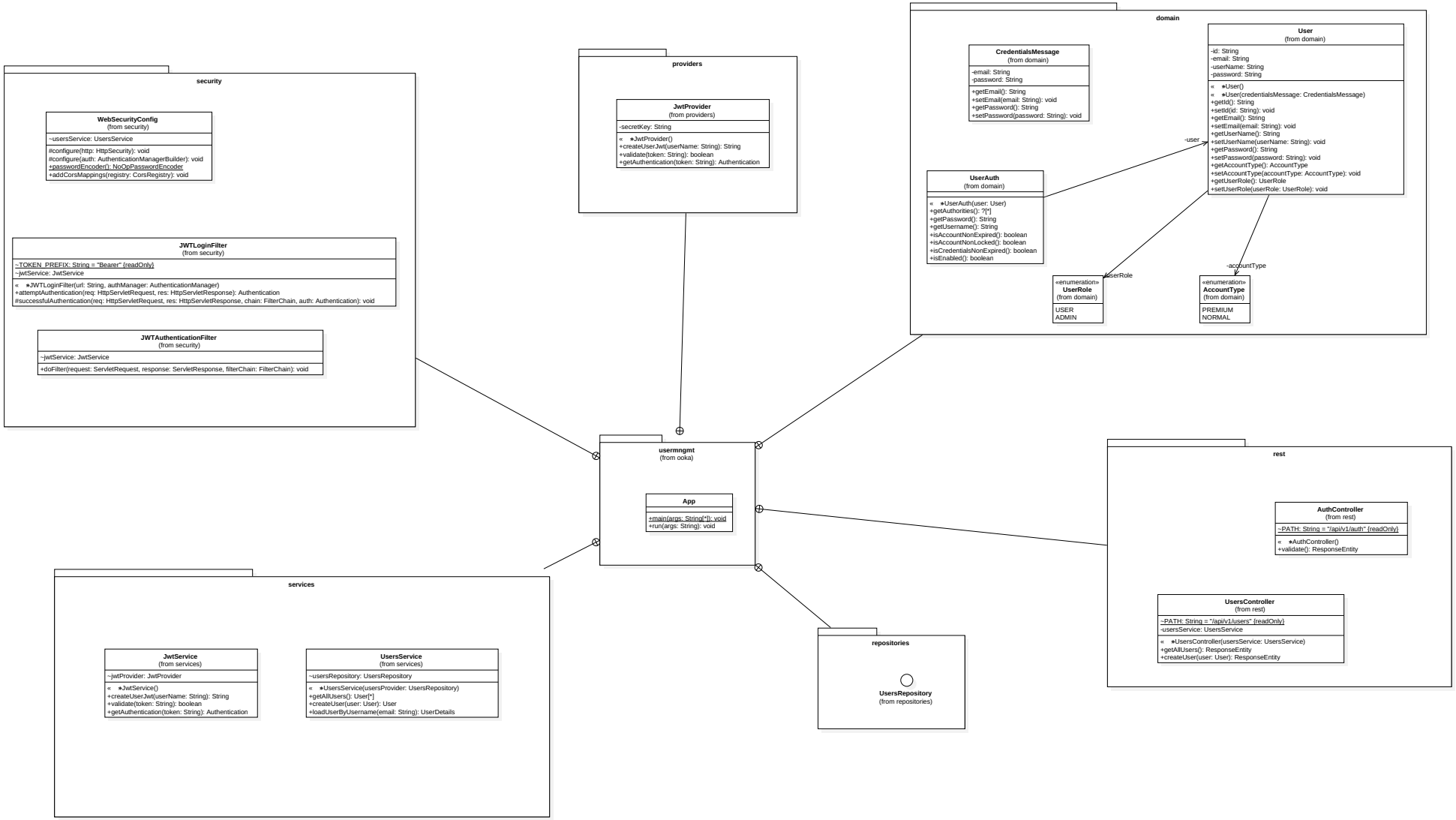
security
WebSecurityConfig
(from security)
~usersService: UsersService
#configure(http: HttpSecurity): void
#configure(auth: AuthenticationManagerBuilder): void
+passwordEncoder(): NoOpPasswordEncoder
+addCorsMappings(registry: CorsRegistry): void
JWTLoginFilter
(from security)
~TOKEN_PREFIX: String = "Bearer" {readOnly}
~jwtService: JwtService
«constructor»+JWTLoginFilter(url: String, authManager: AuthenticationManager)
+attemptAuthentication(req: HttpServletRequest, res: HttpServletResponse): Authentication
#successfulAuthentication(req: HttpServletRequest, res: HttpServletResponse, chain: FilterChain, auth: Authentication): void
JWTAuthenticationFilter
(from security)
~jwtService: JwtService
+doFilter(request: ServletRequest, response: ServletResponse, filterChain: FilterChain): void
providers
JwtProvider
(from providers)
-secretKey: String
«constructor»+JwtProvider()
+createUserJwt(userName: String): String
+validate(token: String): boolean
+getAuthentication(token: String): Authentication
domain
CredentialsMessage
(from domain)
-email: String
-password: String
+getEmail(): String
+setEmail(email: String): void
+getPassword(): String
+setPassword(password: String): void
User
(from domain)
-id: String
-email: String
-userName: String
-password: String
«constructor»+User()
«constructor»+User(credentialsMessage: CredentialsMessage)
+getId(): String
+setId(id: String): void
+getEmail(): String
+setEmail(email: String): void
+getUserName(): String
+setUserName(userName: String): void
+getPassword(): String
+setPassword(password: String): void
+getAccountType(): AccountType
+setAccountType(accountType: AccountType): void
+getUserRole(): UserRole
+setUserRole(userRole: UserRole): void
-user
UserAuth
(from domain)
«constructor»+UserAuth(user: User)
+getAuthorities(): ?[*]
+getPassword(): String
+getUsername(): String
+isAccountNonExpired(): boolean
+isAccountNonLocked(): boolean
+isCredentialsNonExpired(): boolean
+isEnabled(): boolean
-userRole
«enumeration»
UserRole
(from domain)
USER
ADMIN
-accountType
«enumeration»
AccountType
(from domain)
PREMIUM
NORMAL
usermngmt
(from ooka)
App
+main(args: String[*]): void
+run(args: String): void
rest
AuthController
(from rest)
~PATH: String = "/api/v1/auth" {readOnly}
«constructor»+AuthController()
+validate(): ResponseEntity
UsersController
(from rest)
~PATH: String = "/api/v1/users" {readOnly}
-usersService: UsersService
«constructor»+UsersController(usersService: UsersService)
+getAllUsers(): ResponseEntity
+createUser(user: User): ResponseEntity
services
JwtService
(from services)
~jwtProvider: JwtProvider
«constructor»+JwtService()
+createUserJwt(userName: String): String
+validate(token: String): boolean
+getAuthentication(token: String): Authentication
UsersService
(from services)
~usersRepository: UsersRepository
«constructor»+UsersService(usersProvider: UsersRepository)
+getAllUsers(): User[*]
+createUser(user: User): User
+loadUserByUsername(email: String): UserDetails
repositories
UsersRepository
(from repositories)