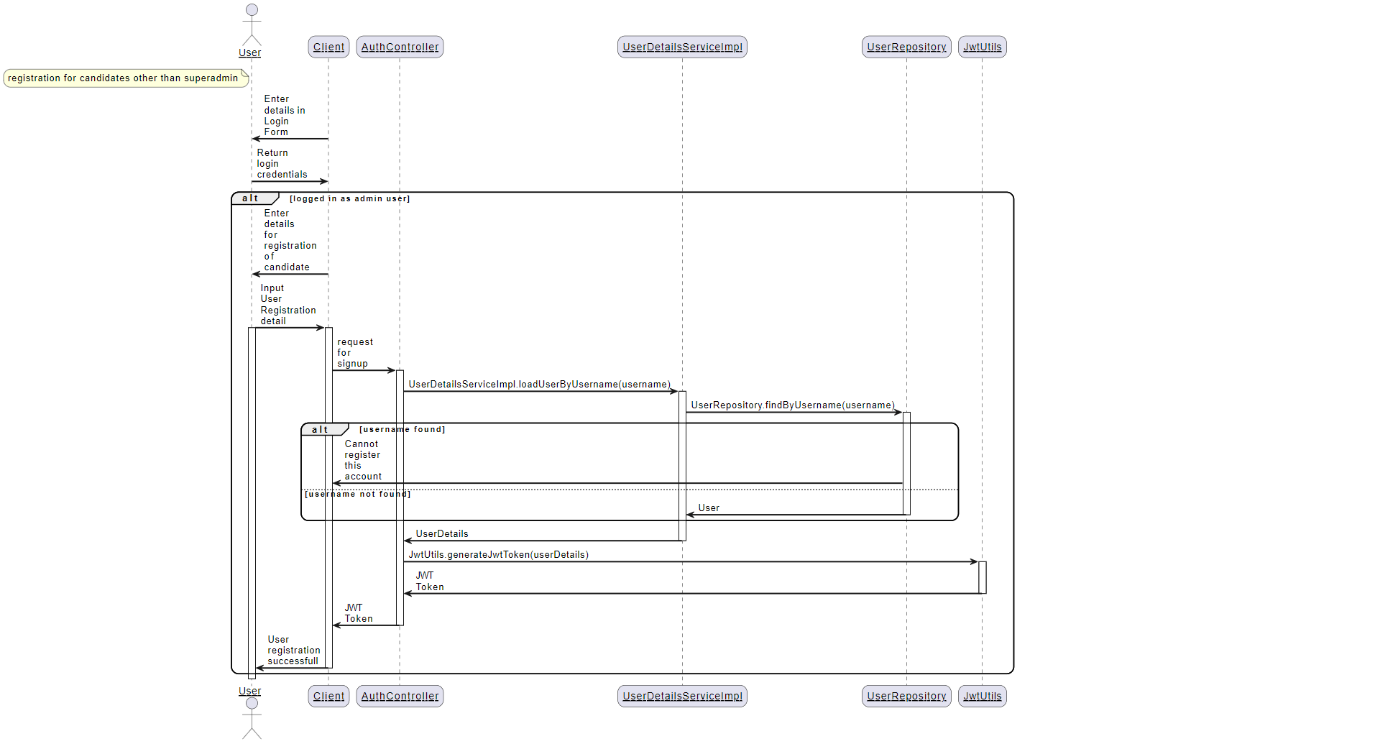
Diagrams For JWT’s

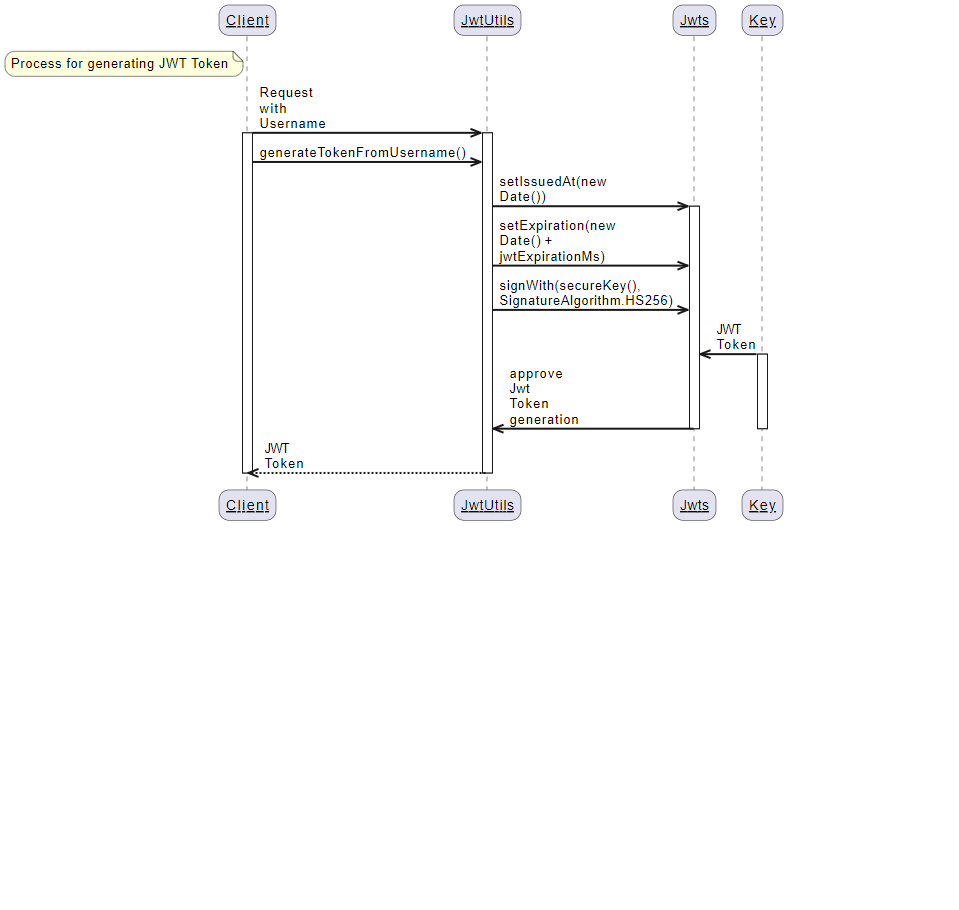
Sign Up (Normal User)



SignIn

A screenshot of a computer

Description automatically generated



* The **signWith(secureKey(), SignatureAlgorithm.HS256)** method from the **Jwts** library is responsible for signing the JWT token using the HMAC algorithm (with SHA-256 hashing) and the secure key generated by **secureKey()**. This ensures the token's authenticity and integrity.
* The generated JWT token is returned to the **JwtUtils**, ready to be sent back to the client.
* These diagrams illustrate how the JWT generation and validation processes leverage the HMAC algorithm for securing and verifying tokens within the **JwtUtils** class.

A screenshot of a computer

Description automatically generated

* The **parse(authToken)** method from the **Jwts** library is responsible for decoding and validating the JWT token. It internally utilizes the HMAC algorithm with the secret key provided by **setSigningKey()** to verify the token's authenticity and integrity.
* If the token parsing is successful, the result indicates that the token is valid. Otherwise, it signifies that the token is invalid.