ENDPOINTS

1. register

2. login

3. google auth

4. user

add profile picture

update bio, social, ...

5. voyage

5.a. create voyage with profile image

5.b. upload voyage image

5.j. upload voyage profile image

5.c. delete voyage image

5.d. delete voyage

5.e. add waypoint for voyage

5.f. get voyages by 1st waypoint location

5.g. get voyages by date, vacancy, vehicle type

5.h. get user's voyages

5.i. get voyage's bids

6. vehicle

6.a. create vehicle with profile image

6.h. get vehicle by id

6.b. upload vehicle image

6.c. delete vehicle image

6.d. edit vehicle details

6.e. change vehicle profile image

6.f. delete vehicle

6.g. get voyages of vehicle

7. bid

7.a. create bid

7.b. change bid

7.c. delete bid

8. message

8.a. send message - signalR & db

8.b. get user's messages

**TOKEN STORAGE**

Flow:

1. On successful authentication, store the token in both Redux state and local storage.

2. Use the token from the Redux state for automatic updates in the UI.

3. On page reload or application restart, check local storage for an existing token. If found, fetch it and update the Redux state.

Considerations:

• Ensure that you handle token expiration and refresh appropriately.

• Implement security measures to protect against XSS attacks if storing tokens in local storage.

**SLICES**

1. **UserSlice**
   1. **State -> user**
      * **Id**
      * **Username (**AspNetUsers**)**
      * **Email**
      * **Title**
      * **Bio**
      * **Instagram**
      * **Facebook**
      * **PhoneNumber**
      * **ProfileImageUrl**
      * **UnseenMessages**
      * **ConnectionId**
      * **Vehicles**
      * **Voyages**
      * **Bids**
      * **SentMessages**
      * **ReceivedMessages**
   2. **Actions**
      * **Register**
        + **Id**
        + **Email**
        + **Username**

**+ Login**

* + - **add profile picture**
      * **ProfileImageUrl**
    - **update bio, social, ...**
      * **Title**
      * **Bio**
      * **Instagram**
      * **Facebook**
      * **PhoneNumber**
    - **Login**
      * **All**
    - **google authentication**
      * (AspNetUserTokens)
      * **All**

1. **VehicleSlice**
   1. **State -> array of vehicles**
      * **Id**
      * **Name**
      * **ProfileImageUrl**
      * **Type**
      * **Capacity**
      * **Description**
      * **UserId**
      * **User**
      * **VehicleImages**
      * **Voyages**
   2. **Actions**
      * **create vehicle with profile image**
      * **get vehicle by id**
      * **upload vehicle image**
      * **delete vehicle image**
      * **edit vehicle details**
      * **change vehicle profile image**
      * **delete vehicle**
      * **get voyages of vehicle**
2. **VoyageSlice**
   1. **State -> Array of Voyages**

* id
* name
* brief
* description
* vacancy
* startDate
* endDate
* lastBidDate
* minPrice
* maxPrice
* fixedPrice
* auction
* profileImage
* vehicleType
* waypoints
  + id
  + latitude
  + longitude
  + title
  + description
  + profileImage
  + order
  + voyageId
* voyageImages
  + id
  + voyageImagePath
  + voyageId
* userId
* user
  + id
  + userName
  + profileImageUrl
* vehicleId
* vehicle
  + id
  + name
  + profileImageUrl
  + type
  + capacity
  + description
  + userId
* bids
  + personCount
  + message
  + offerPrice
  + currency
  + dateTime
  + voyageId
  + userId
  1. **Actions** 
     + **create voyage with profile image**
     + **upload voyage image**
     + **upload voyage profile image**
     + **delete voyage image**
     + **delete voyage**
     + **add waypoint for voyage**
     + **~~get voyages by 1st waypoint location~~**
     + **get voyages by date, vacancy, vehicle type**
     + **get user's voyages**
     + **get voyage's bids**

1. **BidSlice**
   1. **State -> list of bids**
      * **Id**
      * **PersonCount**
      * **Message**
      * **OfferPrice**
      * **Currency**
      * **DateTime**
      * **VoyageId**
      * **Voyage**
      * **UserId**
      * **User**
   2. **Actions**
      * **create bid**
      * **change bid**
      * **delete bid**
2. **MessageSlice**
   1. **State -> array of messages**
      * **Id**
      * **Text**
      * **DateTime**
      * **Rendered**
      * **ReadByReceiver**
      * **SenderId**
      * **ReceiverId**
   2. **Actions**
      * **Send message**
      * **Get user’s messages**