ATLAS Viewer User-Custom-Setting

💡 기존의 Context API로만 관리되던 상태에서 DB에 상태를 저장하여 변경된 상태를 유지할 수 있도록 개선

작업 내역

1. 서버 측 추가된 API (GET, PATCH) 들을 클라이언트 측에서 사용할 수 있도록 추가
    - GET - getUserCustomSetting
    - PATCH - patchUserCustomSetting
2. 전송데이터(Body) 동기화를 위해 UserCustomSetting type 정의

```
export interface UserCustomSetting {
   useScaleBar: boolean,
   useCaptureButton: boolean,
   useViewerRotationSlider: boolean,
   useGammaSlider: boolean,
   useViewerThumbnail: boolean,
   useAnnotation: boolean,
   useICCButton: boolean
 }
```

3. 정의된 타입에 따른 기존 변수 네이밍 수정 작업 및 API 사용 로직 추가

작업 이슈

1. 토글 값이 변경될 때마다 PATCH API를 계속해서 호출할 순 없다고 판단
    a. 최초 렌더링 시에는 getUserCustomSetting api를 호출하여 해당 데이터로 상태를 초기화 그 이후에는 토글 값을 로컬 변경 상태값으로 관리

```
const [userCustomSet, setUserCustomSet] = useState<UserCustomSetting>({
     useScaleBar: true,
     useCaptureButton: true,
     useViewerRotationSlider: true,
     useGammaSlider: true,
     useViewerThumbnail: true,
     useAnnotation: true,
     useICCButton: true,
 });

 useEffect(() => {
     const getUserSetting = async () => {
       const userSet = await getUserCustomSetting();
       return setUserCustomSet(userSet);
     }

     getUserSetting();
 }, [])
```

b. 방법 1) 새로고침과 페이지 이동 추적하기

```
const location = useLocation(); // react-router-dom 제공 hook

/* 새로고침 */
useEffect(() => {
  const updateUserSet = async () => {
    await patchUserCustomSetting(userCustomSet);
  }

  window.addEventListener('beforeunload', updateUserSet);

  return () => {
    window.removeEventListener('beforeunload', updateUserSet);
  }
}, [userCustomSet])

/* 페이지 변경 */
useEffect(() => {
  const handleOutPath = async () => {
    await patchUserCustomSetting(userCustomSet);
  }

  return () => {
    handleOutPath();
  }
}, [location, location.pathname, userCustomSet])
```

c. 방법 2) 저장 버튼 별도로 구성

```
const onSubmit = async () => {
    await patchUserCustomSetting(userCustomSet); // useCustomSet = context
}

...

<Button onClick={onSubmit}>설정 저장</Button>
```