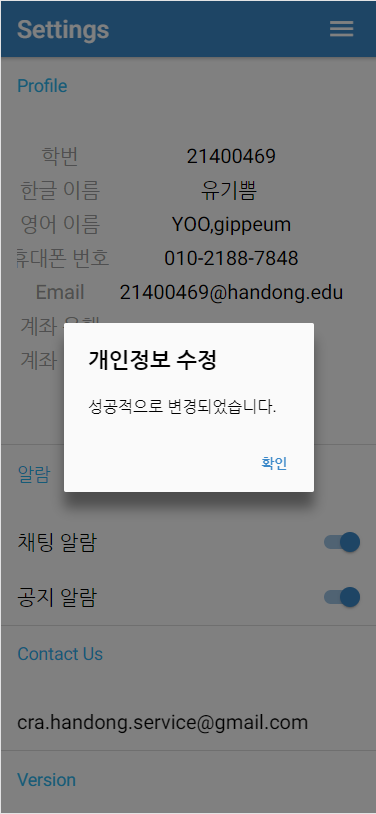
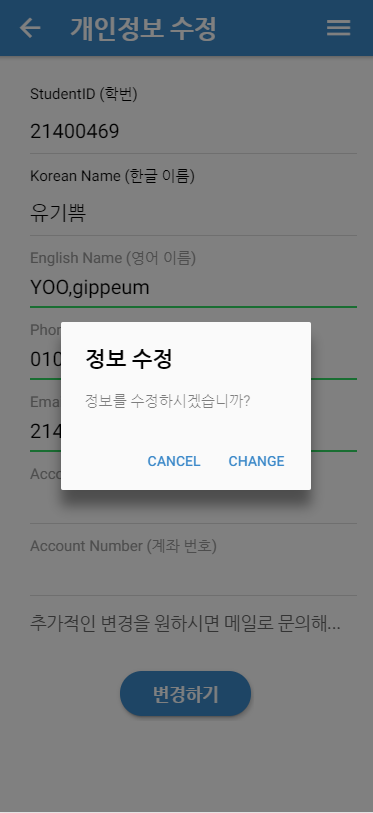
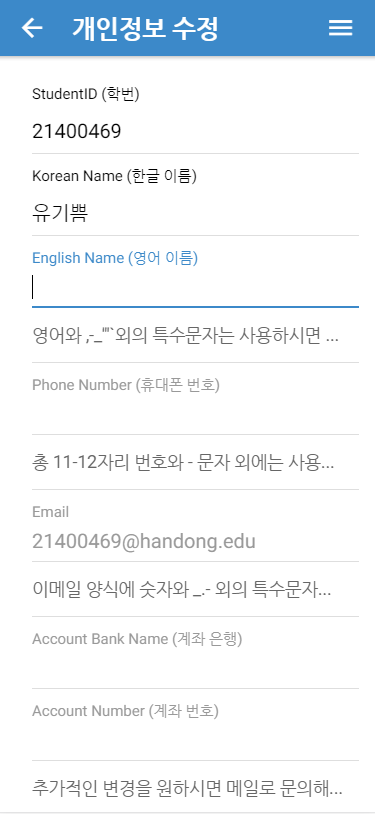
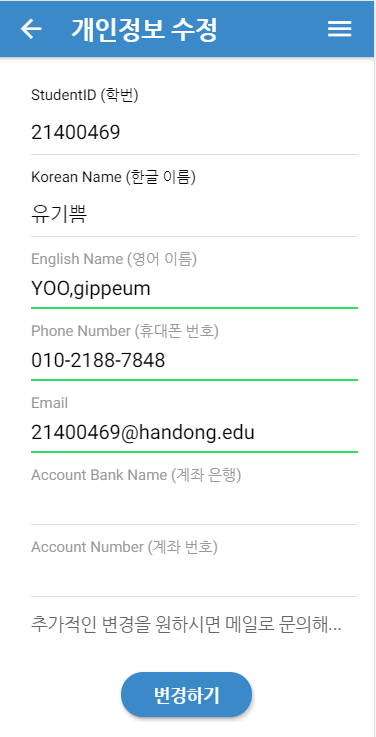
**Personal-Info Page**

3

2

1



* + - 1. Form

userProvider를 이용해 firebase의 /userProfile/studentID 디렉토리에 해당하는 정보를 가져와 userInfo 변수에 넣는다. Form을 만들어 userInfo의 값들을 표시하고 유저가 원하는 값으로 변경할 수 있도록 한다. 이때 학번과 한글이름은 변경할 수 html의 label을 이용해 직접 표시하고 나머지 정보에 대해서만 Form으로 만든다. Form을 만들때에는 유저의 입력값이 유효한지 판단하는 Validator도 함께 지정해주어야 하며, control마다 별도의 Validator가 쓰이는데 이는 src/validator 폴더에 있다. Validator는 정규식을 이용해 입력값이 유효한지 검사하며, 유효한 값이 입력된 경우에는 null 값을, 유효하지 않은 경우 validation errors 값을 반환한다. 계좌은행과 계좌번호는 필수 입력값이 아니기 때문에 입력값이 없는 경우에도 유효하도록 Validator를 설정해주어야 한다.

* + - 1. 정보 수정

Form의 각 영역은 ngIf를 이용해 입력값이 유효하지 않거나 변경된 경우 아래 쪽에 입력가이드가 표시되도록 한다. 이메일 영역은 placeholder를 이용해 유저의 학생계정 메일을 표시해준다. ‘변경하기’ 버튼은 form의 모든 입력값이 유효한 경우에만 활성화되도록 한다.

* + - 1. Alert 창

‘변경하기’ 버튼이 활성화 된 경우 유저가 클릭하면 정보수정 여부를 확인하는 alert 창이 생성된다. 또한 ‘변경하기’ 버튼은 form의 submit 기능을 담당하며, 클릭시 updateUserInfo() 함수가 실행되어 form에 적힌 내용으로 userInfo의 값이 update된다. ‘CHANGE’ 버튼을 누르면 수정이 되었음을 확인시켜주는 새로운 alert 창을 생성하고 ‘확인’을 누르면 settingPage로 돌아가기 위해 pop()을 한다.

* + - 1. Back Button 동작

registerBackButtonAction() 함수를 이용해 유저가 하드웨어의 back button을 눌렀을 때 alert 창이 있으면 해당 창을 닫게하고, 그렇지 않은 경우 이전페이지로 돌아가도록 pop()시켜준다.

사용된 변수

userInfo - firebase로부터 받아온 유저의 프로필 정보를 담기위해 사용한다.

updateForm – userInfo를 이용해 form을 만들기 위해 사용한다.

수정이 필요한 부분

* 글자 잘리는 거
* 11-12자리 -> 10-11자리
* [class.invalid] 왜 필요한지?
* updateForm.controls.accountNumber.dirty 없어도 잘 작동함 왜 필요한지