러닝 페어 중간 보고서 (5점)

- 10페이지 이내 개발 계획서 작성 후 pdf 파일로 제출

- 분량, 형식: 10 페이지 이내에서 자유 형식

- 참고 사항

1. 프로젝트 진행 현황:

- 목표한 프로그램이 얼마나 진행되었는가? **(적당히 살 좀 붙여주십쇼)**

계획대로 원활히 진행되었으며, 현재 예상한 모든 기능은 프로토타입으로 구현하였음. 개발된 프로토타입을 중심으로 팀원들 간 회의를 통해 중간까지의 수정사항과, 최종까지의 예정사항을 선정하여 반영하였음.

수정사항:

- 본인의 글이면 완료 버튼이, 타인의 글이면 쪽지 버튼이 출력되도록 UI 간소화

- [찾습니다] 선택 시 사례 입력란이, [주웠어요] 선택 시 위치 태그가 출력되도록 함

- ‘이차원 보관소’의 이름에 맞는 배경화면을 추가함

- 여러 줄의 쪽지를 보낼 수 있도록 Entry에서 Text로 변경함

- 로그아웃, 종료 버튼 추가

- 글 번호와 글쓴이가 병행 표기되도록 변경함

**- 팀원들이 각자 맡은 역할을 충실히 수행했는가? (그럴듯하게 잘 부탁드립니다)**

2. 설계 내용을 충실하게 이행했는가?

- 프로그램의 전체적인 설계와 구조 (이미지 포함 가능)

**- 이차원 보관소의 순서도 (필수 내용)**

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

**- GUI 및 사용자 경험**

Tkinter 모듈을 활용한 GUI를 구현하여 CUI 방식보다 효과적으로 사용자에게 정보를 전달하고, 입력값을 받을 수 있다. ‘버튼을 눌러서 원하는 기능을 실행한다’라는 직관적인 사용자 경험이 사용자로 하여금 프로그램에 더욱 몰입할 수 있게끔 하였다. 가독성이 좋은 폰트인 ‘나눔글꼴’을 사용하였으며, 배경이 지나치게 화려하여 화면의 가시성을 저하하지 않도록 배경에 검은색 레이어를 덧씌워 어둡게 하였다. 로그인과 회원가입 시 사용자가 이해할 수 있도록 안내 문구를 표기하였고, 값을 잘못 입력하였을 시 잘못된 값에 맞게 안내 문구가 실시간으로 변경된다. 로그인 후 모든 화면의 제목과, main 화면의 상단에 사용자의 이름을 표기하여 사용자에게 친숙하게 다가갈 수 있게끔 하였다. 글을 쓸 때 [찾습니다]와 [주웠어요] 중 한 개를 선택하고, 선택에 필요한 입력란만 활성화시켜 불필요한 입력란으로 인해 사용자에게 혼동을 주지 않도록 하였다. 사례를 입력한 글의 색상을 붉은색으로 변경하여 다른 글 대비 강조하였다.

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

\* 배경화면 출처: <https://wallhere.com/ko/wallpaper/1057962>

-**자료 구조 (필수 내용)**

저장하는 정보는 크게 3가지로 나뉘는데, 사용자의 정보, 게시글, 그리고 각 게시글의 구성요소에 해당하는 Tkinter 모듈이다. 각 정보는 여러가지의 자료형을 포함하므로 모든 자료형을 수용 가능한 List 자료형에 저장된다. 각 List에 저장되는 정보는 다음과 같다.

텍스트, 스크린샷, 도표, 라인이(가) 표시된 사진

자동 생성된 설명

먼저 userlist는 사용자가 학번(id)과 비밀번호(pw), 이름(name)을 입력한 후 회원가입(register())하였을 때의 정보를 저장한다. mailbox는 사용자의 쪽지함이며, 다른 사용자가 보낸 쪽지가 저장된다. 최대 10명이 사용 가능한 크기(len(mailbox) == 10)로 선언되어 있고, 회원가입 시 사용자에게 자동으로 할당된다. 회원가입 시 각 정보를 list로 묶어 append하는데, 이 때의 인덱스는 len(userlist)-1과 같다. 따라서 사용자에게 할당된 mailbox의 인덱스도 len(userlist)-1로 주어진다(userlist의 0번 인덱스인 사용자는 mailbox의 0번 인덱스를 사용). id는 이후 각 함수나 Tk 인스턴스에서 값을 처리하는데 사용된다.

board의 경우 사용자가 선택한 tag와 입력한 글제목, 글내용, 연락처를 저장한다. 마지막 인덱스에 tag가 [찾습니다](tag[0])일 경우 사례를 저장하고, [주웠어요](tag[1])일 경우 place 내의 ‘원위치’, ‘위탁’, ‘취득’ 셋 중 하나의 값을 저장한다(place[0],[1],[2]). 사용자가 선택한 tag에 따라 입력할 수 있는 값이 변경되므로 값이 잘못 입력되는 경우를 방지하였다. 사례와 연락처는 ‘선택 사항’이므로, 값을 입력하지 않았을 때 ‘N/A’를 전달받는다.

마지막으로 boardnum은 main에 존재하는 Tk 인스턴스들을 행으로 묶어서 저장한다. boardnum은 완료 버튼(comp\_btn)을 눌렀을 때 해당하는 인덱스의 행을 비활성화시키거나, 쪽지 버튼(mail\_btn)을 눌렀을 때 해당하는 글쓴이(id)의 쪽지함에 쪽지를 보내는 데 사용된다. 현재 최대 10개의 글을 작성할 수 있도록 설정되었다.

**주요 기능: Initial() 알고리즘 (필수 내용)**

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

함수 밖까지 값을 유지해야 하므로 Tk 인스턴스들을 보관할 변수들을 global로 선언하고, 초기 y값을 설정했다. Initial() 함수는 글을 작성할 때마다 실행되는데, 두 번 이상 실행 시 기존의 값이 배경에 그대로 남아있어 화면이 지저분해진다. 따라서 매개변수로 init 변수를 받아 처음 실행과 그 이후 실행의 경우를 구분하였다. main 화면 진입, 즉 첫 번째 실행(login()) 시 매개변수로 True를 전달하여 if문이 실행되지 않게 하기 위해 main 화면에 init 변수를 True로 선언하였다. 두 번째 실행(getdata())부터는 False를 직접 인자로 설정하여서, if문이 실행된다. if문은 boardnum의 길이만큼(글의 개수) 반복하여 각 Tk 인스턴스들을 화면에서 지우는 역할을 한다. Tk 인스턴스를 받는 변수의 빈 공간에는 ‘N/A’가 저장되어 있으므로, ‘N/A’가 아닐 때까지만 for문을 반복한다.

for문은 in enumerate(board)를 활용하여 현재 저장된 글의 개수만큼 반복하고, 반복횟수(인덱스)를 index로, board의 각 값을 i로 받는다. 처음에 접속한 사용자의 id와 글쓴이의 id를 비교하여 만약 같으면 is\_user에 1을, 다르면 0을 그대로 사용한다. is\_user가 1이면 본인의 글이므로 쪽지 버튼 대신 완료 버튼을 생성하고, 0이면 타인의 글이므로 쪽지 버튼을 생성한다. 두 경우 모두 생성하지 않은 버튼에 ‘N/A’를 할당한다(1). i[1], 즉 tag가 [찾습니다]이고, i[-1], 사례가 ‘N/A’가 아닌 경우에 붉은색 글씨를, 나머지는 검은색 글씨로 글을 작성한다. 글에는 글 번호(index+1), 가운데를 ‘\*’로 필터링한 사용자 이름, tag(i[1]), 제목(i[2])가 포함된다. 이후 x,y 좌표를 옮겨가며 글, 내용 버튼, 완료/쪽지 버튼을 생성한다. 이 때 각 버튼의 command에 lambda x = index:함수(x)의 형태, 즉 for문 실행 당시의 index값을 인자로 하는 함수를 전달한다(2). 한 번의 for문이 끝나면 생성된 Tk 인스턴스들을 List로 묶어 boardnum의 [index]에 대입한다. comp\_chkr는 해당하는 인덱스의 boardnum에서 완료 버튼이 눌렸는지를 확인하는 List로, 완료 버튼이 눌리면 해당 인덱스를 comp\_chkr의 인덱스로 받아 True의 값을 저장한다. 따라서 이후로 Initial() 함수가 실행되어도, 완료 버튼이 눌린 인덱스의 comp\_chkr의 값이 ‘N/A’가 아니므로 comp() 함수가 실행되어 완료처리를 유지할 수 있다.

**mailscreen() 함수와 mailupdate() 알고리즘**

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

mainscreen() 함수는 쪽지함을 출력하고 실시간으로 정보를 업데이트하는 mailupdate() 함수를 내부에서 실행하는 기능을 한다. namechkr() 함수로 id에 해당하는 name을 title에 전달하기 위해 id를 매개변수로 받았다. login 화면을 부모로 하는 Toplevel mbox를 선언하여, 화면 구성에 필요한 요소들을 배치하였다. 이 때 여러 개의 쪽지를 한 번에 볼 수 있도록 Text 및 Scrollbar Tk 인스턴스를 Frame에 같이 배치하였다.

쪽지를 실시간으로 갱신하기 위해 mailupdate() 함수를 따로 구성하여, 실질적인 쪽지함 역할을 하는 Text Tk 인스턴스에 after 메서드를 사용하여 0.2초마다 mailupdate() 함수를 실행하게 하였다. 빈 list mails를 선언하고, 현재 Text Tk 인스턴스의 y position 값을 pos에 저장한다. .yview() 메서드는 y position의 값을 (분자, 분모)의 형태로 표시하는데, 최상단의 값은 (0, n)이고, 최하단의 값은 (n, 1)이다. 최상단일 때의 n의 값은 0보다 크며, 최하단일 때의 n의 값은 1보다 작다. 쪽지함을 갱신하기 앞서 먼저 쪽지함을 초기화해야 한다. Text Tk 인스턴스의 처음 값은 ‘1.0’부터 시작하므로, .delete(‘1.0’, END) 메서드로 처음부터 끝까지 값을 제거하였다. for문을 사용하여 userlist에서 사용자의 id와 같은 list를 찾아 3번 인덱스에 저장되어 있는 mailbox의 값을 추출하여 mails에 저장하고, 값을 찾았으므로 break로 for문을 종료한다. mails의 길이가 0보다 크면, 즉 쪽지가 한 개라도 있으면 for문을 통해 한 줄씩 Text Tk 인스턴스에 insert한다. 가장 최근의 값은 가장 마지막 인덱스에 저장되어 있으므로, 첫 번째 인덱스부터 차례대로 END 위치에 이름과 내용을 작성한다. mailupdate() 함수는 0.2초마다 실행되므로, 매번 스크롤바의 위치가 최상단으로 고정된다. 따라서, 스크롤바의 위치를 현재 위치로 고정하기 위해, mailupdate를 실행했을 때 저장해둔 y position의 분자값(인덱스 0번)으로 MOVETO하여 스크롤바의 이동을 막는다.

**getdata() 함수와 sending() 함수**

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

getdata() 함수는 post 화면에 내장된 함수로, 글을 작성하는 기능을 한다. 연락처가 입력되지 않았으면 ‘N/A’를, 입력되면 그 값을 tel에 저장한다. tagvar는 [찾습니다]와 [주웠어요]를 결정하는 Radiobutton Tk 인스턴스로 [찾습니다]는 0, [주웠어요]는 1에 해당한다. 0인 경우 사례를(없으면 ‘N/A’) last에 저장하고, 1인 경우 [원위치, 위탁, 취득] 중 해당하는 값을 last에 저장한다. 제목과 내용은 필수 입력사항이므로, 제목과 내용이 모두 공백이 아닌 경우에 board의 0번 인덱스에 정보가 insert된다. 새롭게 작성된 글이므로, comp\_chkr의 0번 인덱스에도 ‘N/A’를 insert하여 에러를 방지한다. 글을 갱신하기 위해 initial() 함수를 실행하고, 이 경우 두 번째 실행이므로 init 인자에 False를 입력하여 initial() 함수의 글 초기화 기능을 활성화한다. 글을 작성했으므로 post 화면을 destroy하여 종료한다. 만약 제목과 내용 둘 중 하나라도 입력되지 않았다면 if문이 아닌 else문으로 빠져 재입력 안내문구인 void\_lab을 place한다.

sending() 함수는 send 화면에 내장된 함수로, 쪽지를 작성하는 기능을 한다. user\_id는 글 작성자의 id로, 글 작성자의 id에 일치하는 userlist의 mailbox의 0번 인덱스에 값을 insert하고, 쪽지를 전송하였으므로 입력창(send\_ent)의 값을 delete한다.

**3. 난관 및 문제점 (필수 내용)**

(1) 글 작성 후 main 화면 재실행 시의 에러

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

처음 Initial() 알고리즘 작성 당시 사용자의 글이면 완료, 아니면 쪽지 버튼을 생성하고 각 반대의 경우에 ‘N/A’를 할당하지 않았다. 이 경우 처음 실행 및 글 작성 시 정상적으로 동작했으나, main 화면을 닫고 login 화면에서 다시 main 화면을 실행 시 화면이 깨지고 에러가 발생하였다. 에러 메시지를 확인해보니 config할 대상이 없다고 출력되었다. 따라서 boardnum을 print해가며 에러를 추적하였다. 그 결과, main 화면을 닫고 다시 생성할 경우 기존 버튼인 .!toplevel.!button4가 destroy되고 새로운 버튼이 생성되는데, boardnum에 .!toplevel.!button4의 값이 한 인덱스 밀린 채로 남아있어서 comp() 함수에서의 config 명령이 정상적으로 작동하지 않음을 알게 되었다. main 화면을 여러 번 열어도 글이 정상적으로 생성될 수 있도록, boardnum에서 사용하지 않는 버튼의 값에 ‘N/A’를 입력하여 comp() 함수 시 불필요한 config 명령이 이루어지지 않게 하였다.

(2) Button Tk 인스턴스 command 미동작 에러

lambda 함수 없이 함수를 입력하고 글을 작성하면 버튼을 누르지 않았음에도 불구하고 버튼의 함수가 실행되어 비정상적인 동작을 하게 된다. 원인은 command에 할당된 함수가 인자를 필요로 하는 함수이기 때문이다. 인자가 없는 함수 a()를 사용할 때 command=a의 형태로 사용하는데, 이때 Button Tk 인스턴스는 command에 ‘a 함수를 실행하는 것’을 저장한다. 그런데 인자가 있는 함수 a(n)을 command=a(n)의 형태로 사용하게 되면, command에 ‘a(n)의 결과’가 저장된다. 따라서 ‘함수를 실행하는 것’을 저장해야 하는 command와 충돌하여 버튼이 눌리지 않았음에도 불구하고 a(n)이 실행되는 것이다. 따라서, ‘함수를 실행하는 것’을 전달할 수 있는 lambda 함수를 사용해야 한다.

lambda 함수를 사용했을 때 정상적으로 동작하는 것을 확인했으나, 최하단의 글에만 동작하는 문제가 발생하였다. 이는 lambda 함수에 index를 바로 인자로 사용했기 때문인데, 이 경우 lambda 함수는 ‘함수를 실행하는 것’을 전달하므로, lambda 함수의 인자에 값을 따로 할당하지 않기 때문에 for문을 모두 마친 뒤의 index가 lambda 함수에 일괄 적용된다. 따라서 모든 command가 마지막 index의 글에만 적용되는 것이다. 이를 막기 위해서 lambda x = index: 함수(x)의 형태로 ‘for문 실행 당시의 index값을 저장한 x를 인자로 하는 함수를 실행하는 것’의 기능을 하는 lambda 함수를 선언해야 한다.

4. 향후 계획 - 앞으로 완료해야 될 작업은 무엇이 남아 있는가?

**(적절히 살 좀 붙여주십쇼)**

- 찾습니다/주웠어요 게시글 별 시인성 개선

- 제목, 세부내용 글자수 제한 추가

- 마이페이지(본인이 쓴 글 확인) 추가

- 메인화면 ‘?’ 버튼 튜토리얼 구현

**(아래 두개는 못할 가능성이 크니 최대한 할듯 말듯하게 가거나 빼버립시다)**

- 쪽지함 편의성 개선

- 게시글 내용 항목 추가 및 개선