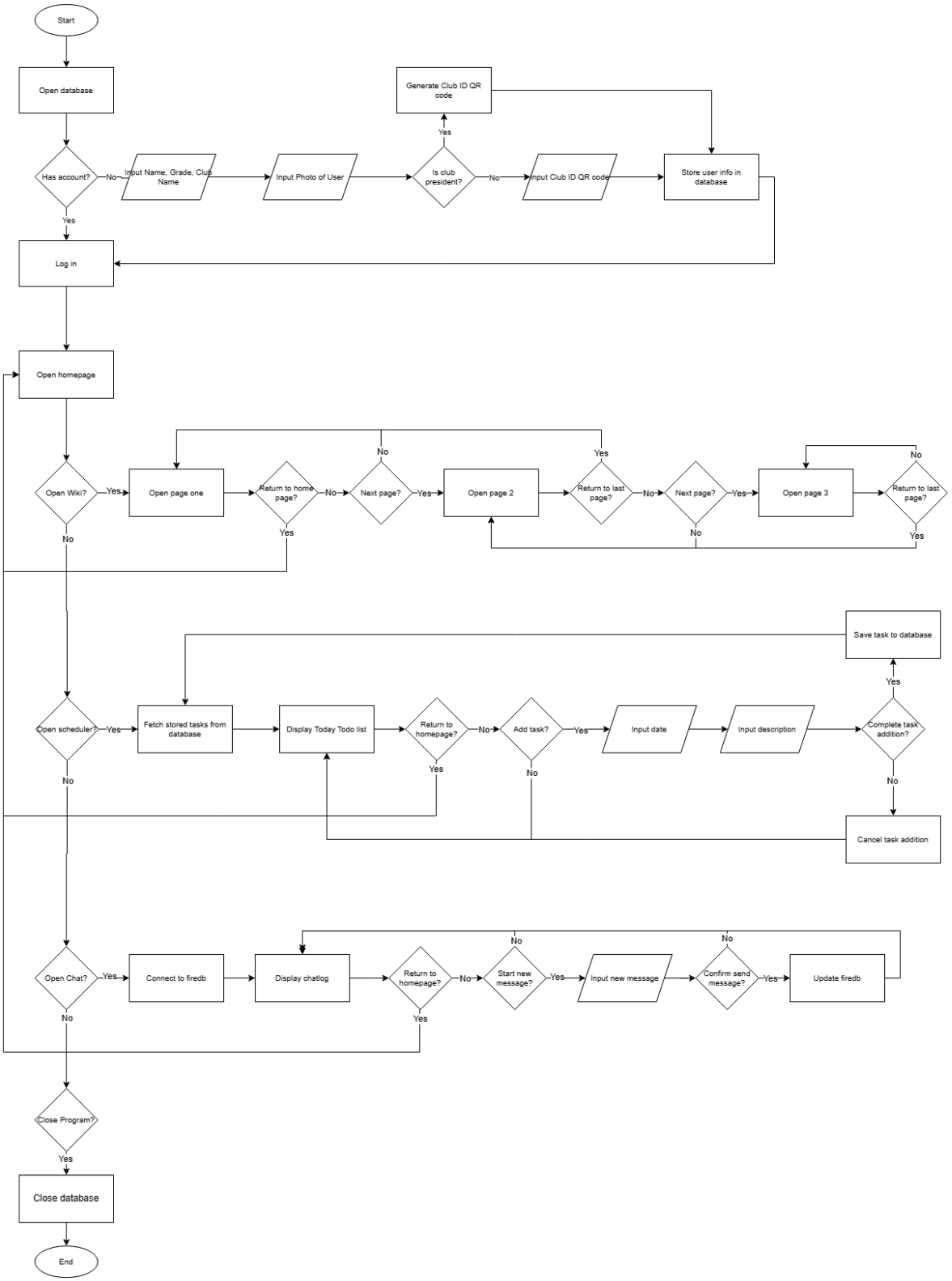
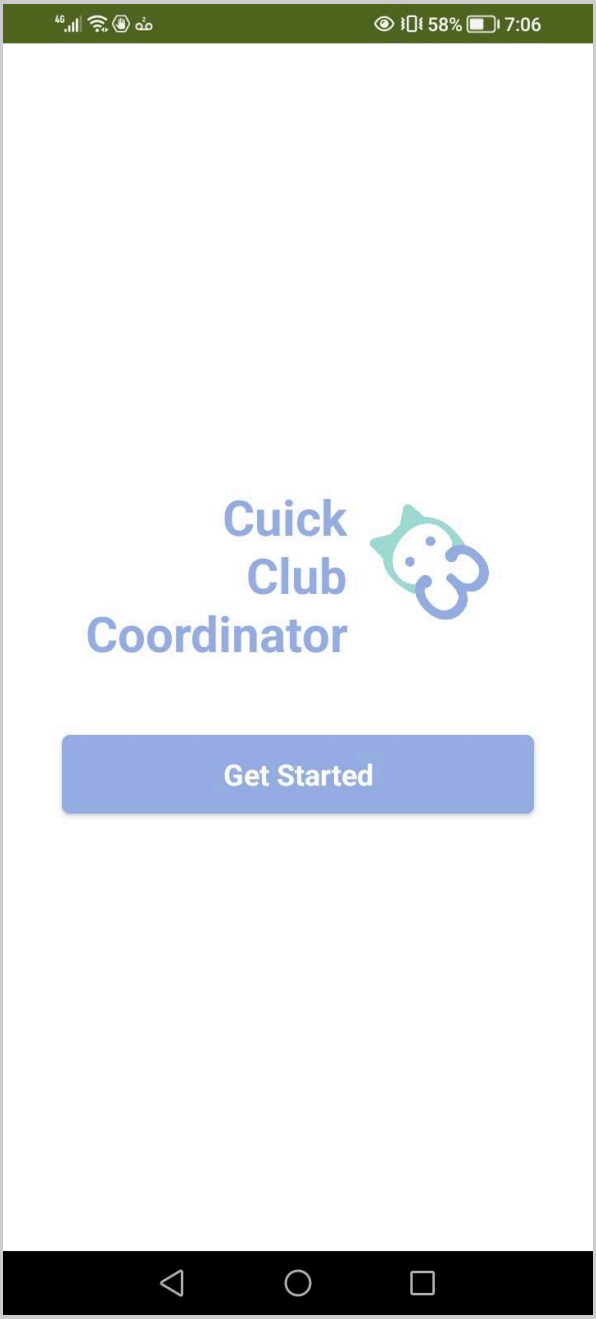
1. Flowchart


Start
Open database
Has account?
No
Input Name, Grade, Club Name
Input Photo of User
Is club president?
Yes
Generate Club ID QR code
No
Input Club ID QR code
Store user info in database
Yes
Log in
Open homepage
Open Wiki?
Yes
Open page one
Return to home page?
No
Next page?
Yes
Open page 2
Return to last page?
No
Next page?
Yes
Open page 3
Return to last page?
No
Yes
No
Yes
No
Yes
No
Yes
No
Yes
Save task to database
Open scheduler?
Yes
Fetch stored tasks from database
Display Today Todo list
Return to homepage?
No
Add task?
Yes
Input date
Input description
Complete task addition?
Yes
No
Yes
No
Cancel task addition
Open Chat?
Yes
Connect to firedb
Display chatlog
Return to homepage?
No
Start new message?
Yes
Input new message
Confirm send message?
Yes
Update firedb
No
No
Yes
No
Close Program?
Yes
Close database
End

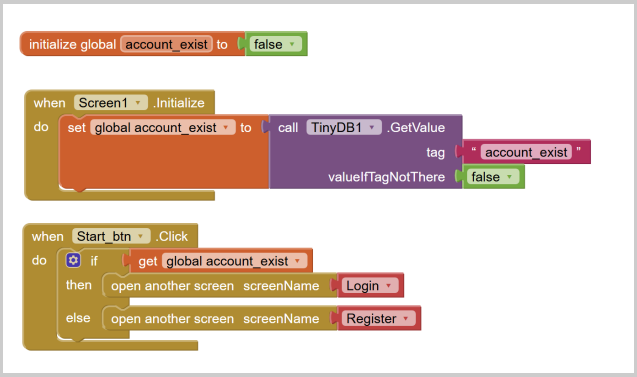
2. Screen UI and background elements + Code + Screenshot

Screen1		
User Interface	**Name/Purpose**	
Buttons	Start If pressed, sends the user to registration or login depending on if the user has an account.	
Images	Logo Displays an image of the app logo.	
Labels	Name Displays the name of the app.	Label2 Provides spacing.
TinyDB	TinyDB1 The main database of the app. Stores the user account information used.	

Screen1



Screen1



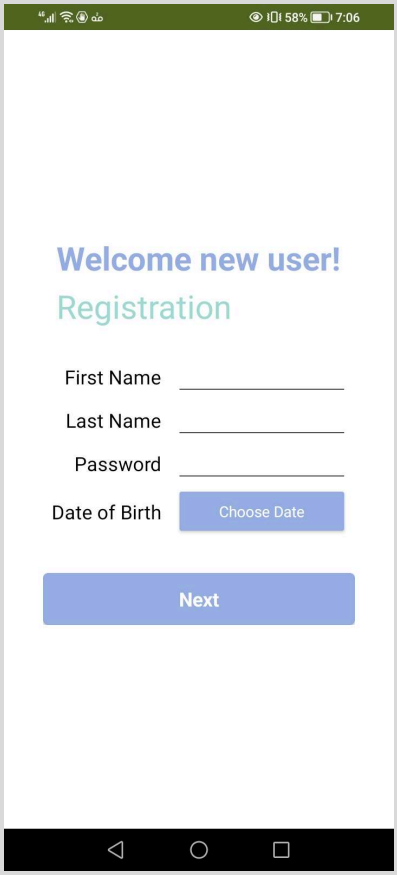
Checks if the user has an account. If not, then redirects the user to the register screen when the start button is pressed. Otherwise directs the user to the login screen.

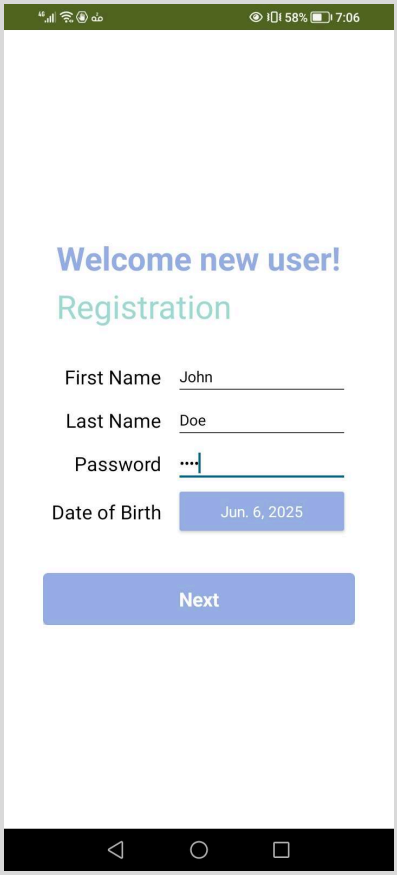
Register

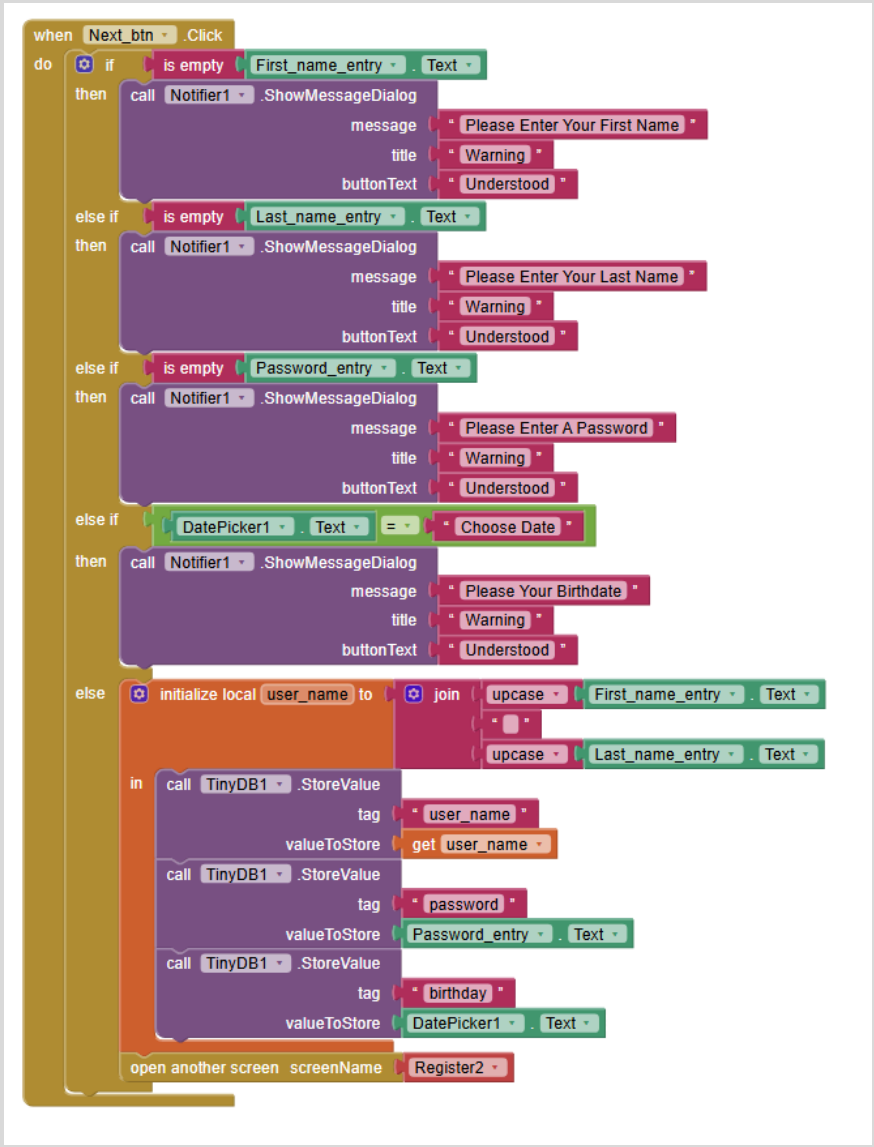
User Interface	**Name/Purpose**								
Buttons	Next_btn If pressed, check that all input fields have been completed. If completed, send the user to registration2. Otherwise, call the notifier.								
Labels	Title Displays the text “Welcome new user”.	Subtitle Displays the text “registration”.	First_name_label Displays the text “First Name”.	Last_name_label Displays the text “Last Name”.	Password_label Displays the text “Password”.	Dob_Label Displays the text “Date of Birth”.	Label7 Provides spacing for the entry fields	Label9 Provides spacing for the next button and entry fields.	Label10 Provides spacing for the header.
TinyDB	TinyDB1								

Register									
	The main database of the app. Stores the users new account information.								
Textbox	First_name_entry Provides a spot for the user to input their first name.	Last_name_entry Provides a spot for the user to input their last name.							
Datepicker	Datepicker1 Allows the user to select the day of their birth.								
Password Textbox	Password_entry Allows the user to enter their password.								
Notifier	Notifier1 Display warning messages if registration is incomplete.								
Sensors	Clock1 Formats datepicker instants for storage and display.								

Register


58% 7:06
Welcome new user!
Registration
First Name
Last Name
Password
Date of Birth
Choose Date
Next

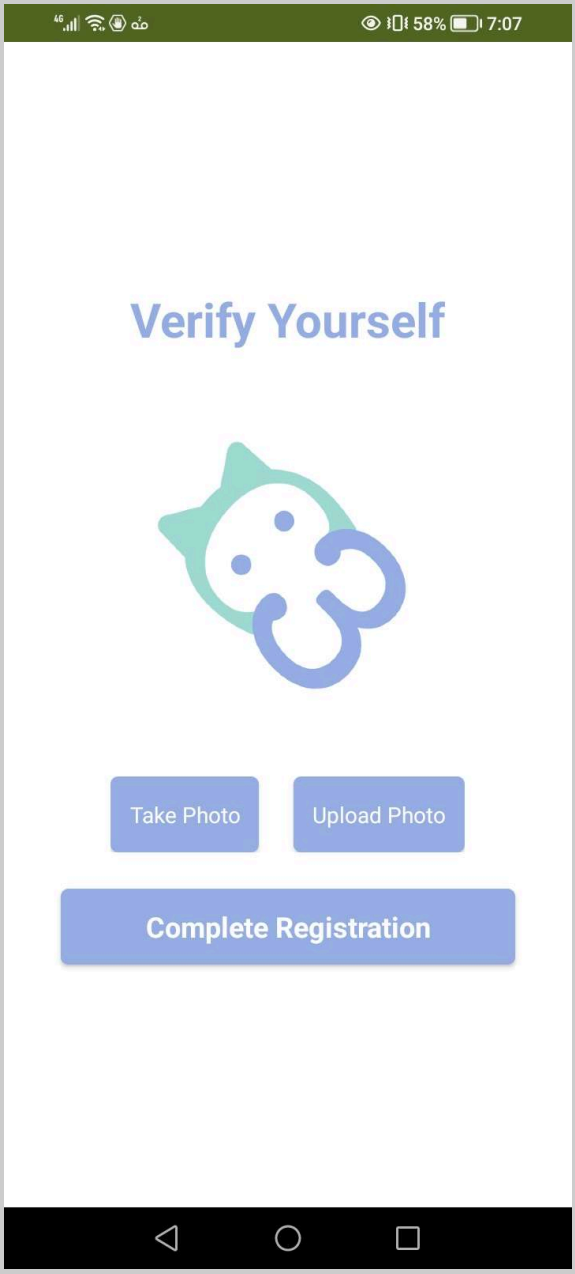

58% 7:06
Welcome new user!
Registration
First Name
John
Last Name
Doe
Password
Date of Birth
Jun. 6, 2025
Next

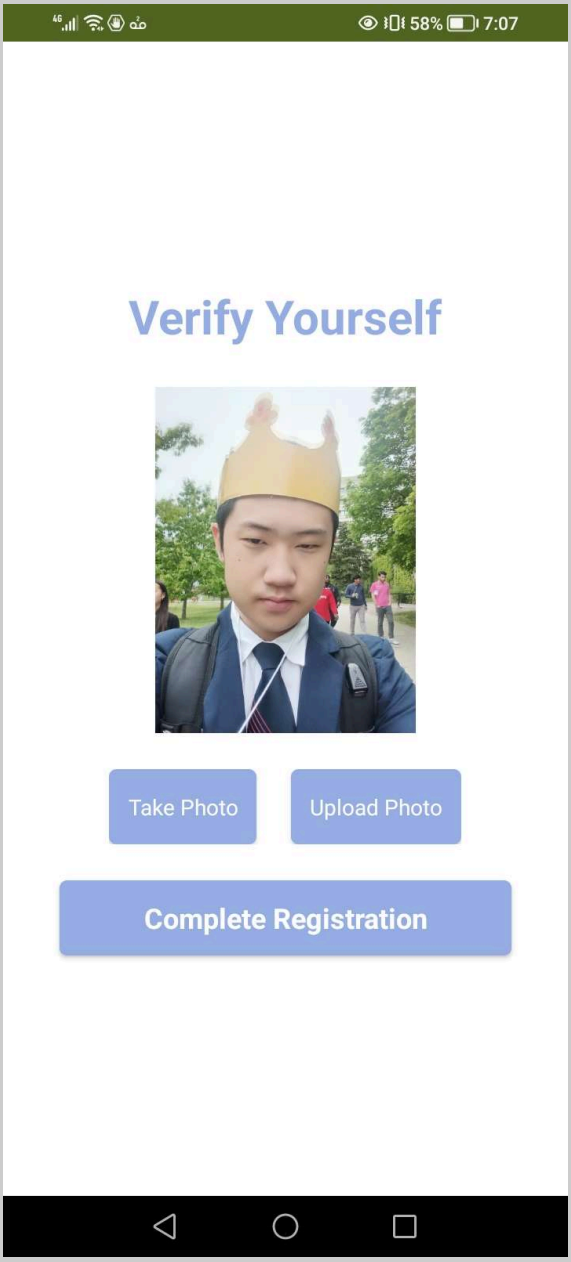

when Next_btn .Click
do if is empty First_name_entry . Text
then call Notifier1 .ShowMessageDialog
message " Please Enter Your First Name "
title " Warning "
buttonText " Understood "
else if is empty Last_name_entry . Text
then call Notifier1 .ShowMessageDialog
message " Please Enter Your Last Name "
title " Warning "
buttonText " Understood "
else if is empty Password_entry . Text
then call Notifier1 .ShowMessageDialog
message " Please Enter A Password "
title " Warning "
buttonText " Understood "
else if DatePicker1 . Text = " Choose Date "
then call Notifier1 .ShowMessageDialog
message " Please Your Birthdate "
title " Warning "
buttonText " Understood "
else initialize local user_name to join upcase First_name_entry . Text
" "
upcase Last_name_entry . Text
in call TinyDB1 .StoreValue
tag " user_name "
valueToStore get user_name
call TinyDB1 .StoreValue
tag " password "
valueToStore Password_entry . Text
call TinyDB1 .StoreValue
tag " birthday "
valueToStore DatePicker1 . Text
open another screen screenName Register2

Register
When the user submits their registration, the program will validate the user's input. If all input is valid, the program stores it into the account database in the correct format. After, the program sends the user to registration2. If the input is invalid, the program will call the notifier to give a warning..

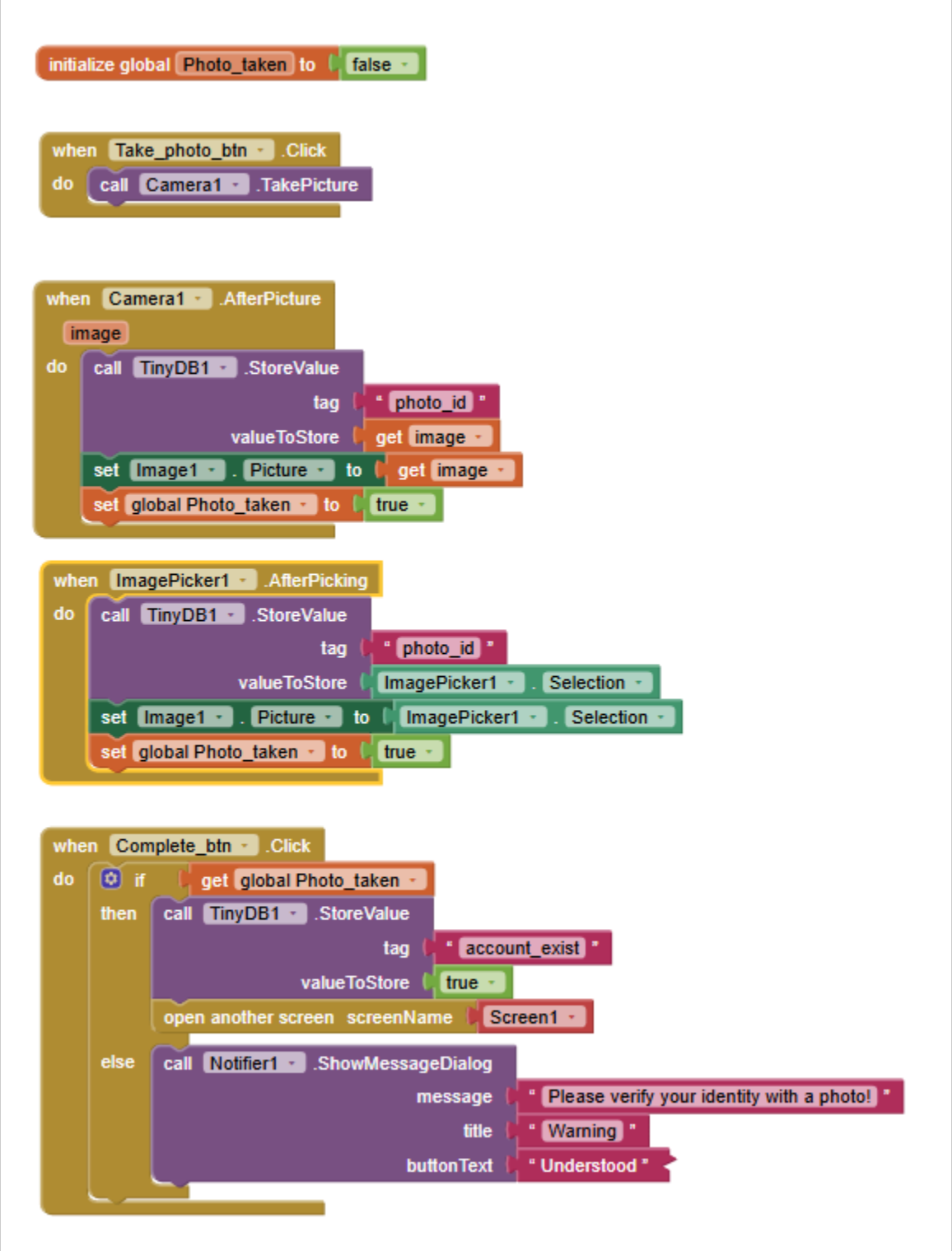
Register2					
User Interface	**Name/Purpose**				
Buttons	Take_photo_btn Activates the camera for the user to take a photo.	Complete_btn If pressed, check that all fields have been completed. If complete, send the user back to screen1. Otherwise, call the notifier.			
Image	Image1 Empty image box used to display the photo the user takes/uploads. Default set to the app logo.				
Labels	Title Displays the text "Verify yourself".	Label5 Provides spacing for the title.	Label2 Provides spacing for the image.	Label3 Provides spacing between camera button and image picker.	Label4 Provides spacing between complete button and image buttons.
Media	Imagepicker1 Allows the user to select an image stored on the device.	Camera1 Allows the app to use the user device's camera.			
TinyDB	TinyDB1 The main database of the app. Stores the user's photo file path.				

Register2					
Notifier	Notifier1 Display warning messages if registration is incomplete.				





Register2

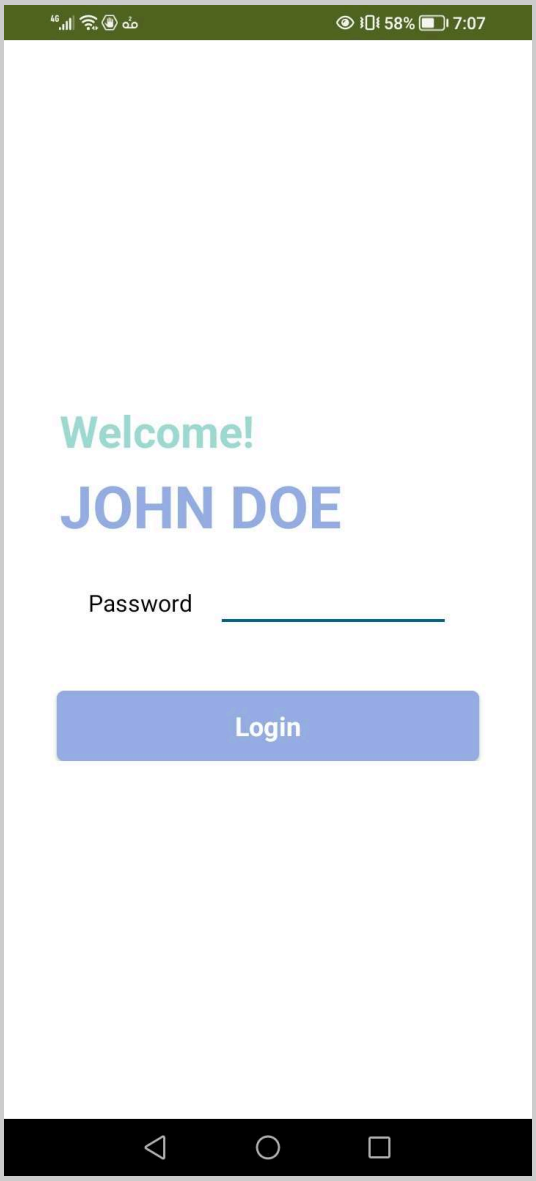


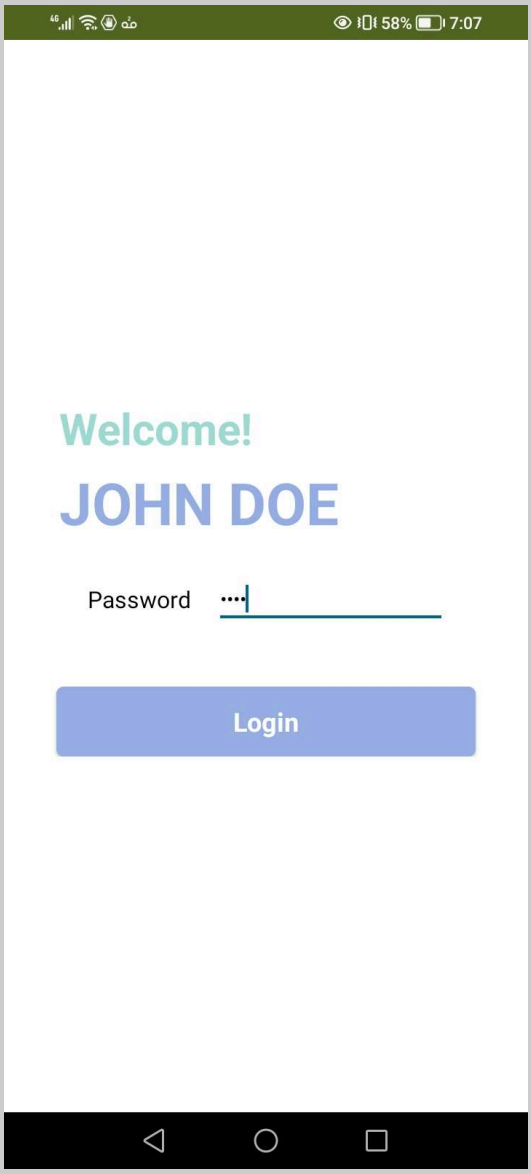
This code updates the image to display whatever photo the user selects or takes. When the user completes this

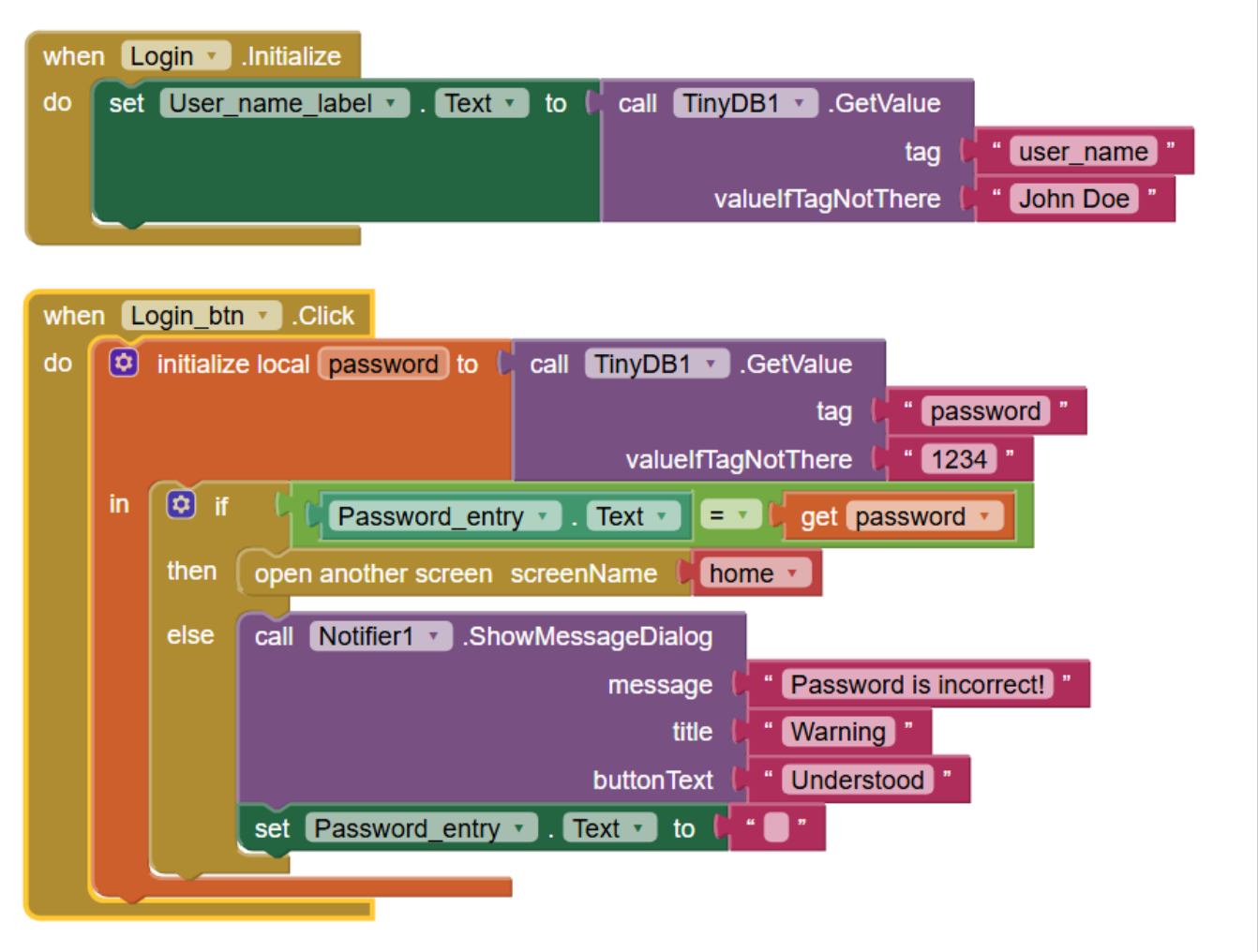
Register2
registration, the program will validate the input. If the input is valid, the photo file path is stored in the database and the user is redirected to screen1 of the program. If the input is invalid, the program calls the notifier to give a warning.

Login						
User Interface	**Name/Purpose**					
Buttons	Login_btn If pressed, check if the user password is correct. If correct, send the user to home. If incorrect, call the notifier.					
Password Textbox	Password_entry Allows the user to enter their password.					
Labels	Welcome_label Displays the text “Welcome!”	User_name_label Displays the user’s name	Label9 Provides spacing for the login entry and header	Password_label Displays the text “Password”.	Label6 Provides spacing between the Password label and entry	Label5 Provides spacing between the login button and the password entry field
TinyDB	TinyDB1 The main database of the app. Stores the user's account information used in verifying the login.					
Notifier	Notifier1 Display warning messages if login is incorrect.					

Login


58% 7:07
Welcome!
JOHN DOE
Password
Login

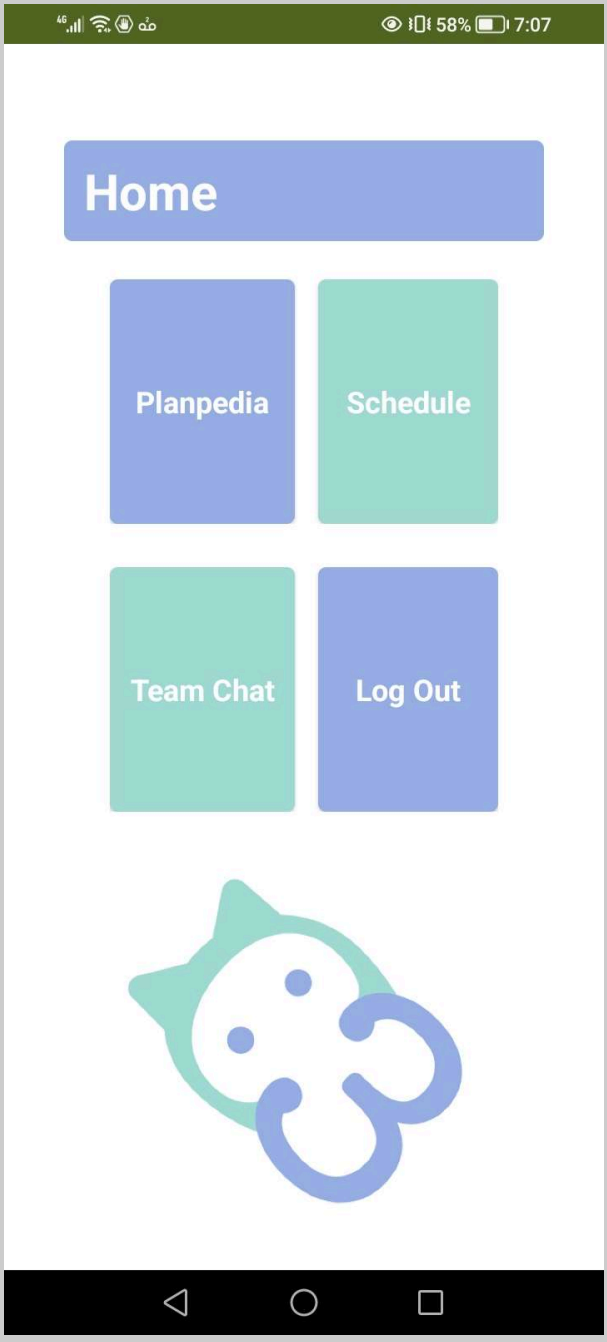

58% 7:07
Welcome!
JOHN DOE
Password
Login

Login
 When the user presses the login button, the program gets the inputted password and checks it against the account password stored in the database. If the password is correct, the program sends the user to the home screen. If not correct, the program calls the notifier to send a warning.

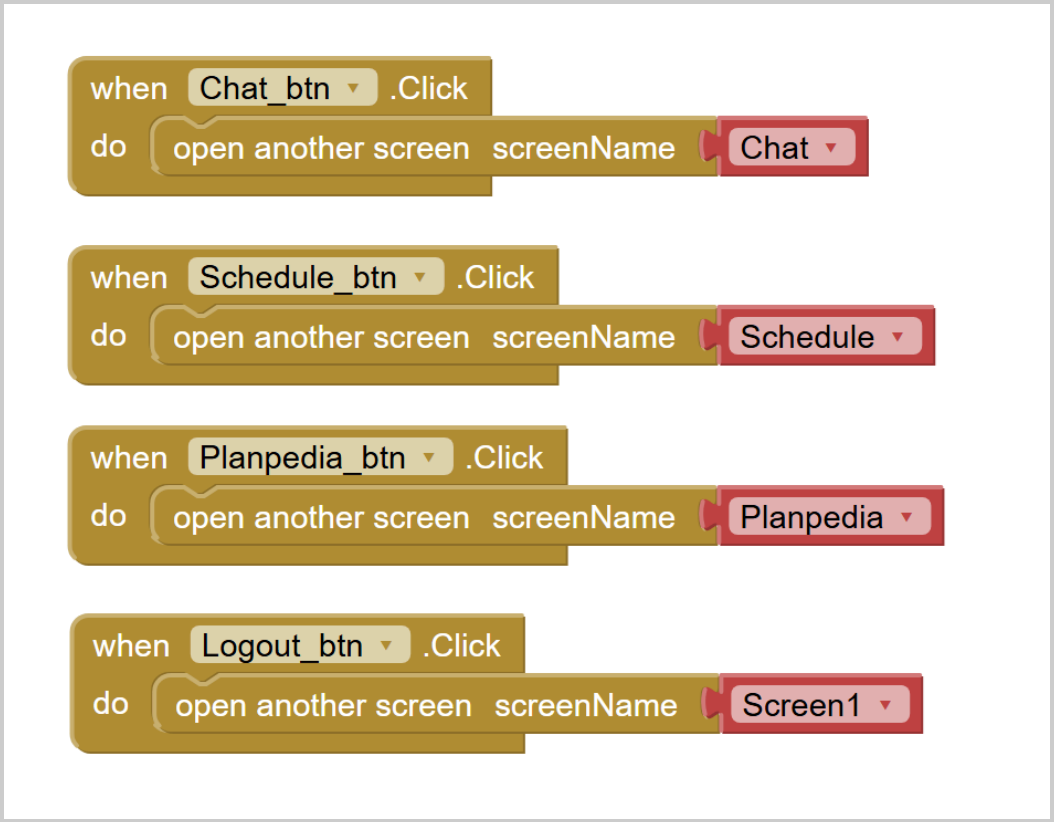
Home				
User Interface	**Name/Purpose**			
Buttons	Title	Chat_btn	Planpedia_btn	Schedule_btn

Home				
	Non functional button used for the rounded border aesthetics like a label. Displays the text “Home”.	Send the user to the chat screen.	Send the user to the plantpedia screen.	Sends the user to the schedule screen.
Image	Logo Displays the app logo.			
Labels	Label4 Provides spacing for the header.	Label1 Provides horizontal spacing between navigation buttons	Label2 Provides vertical spacing between navigation buttons	

Home



Home

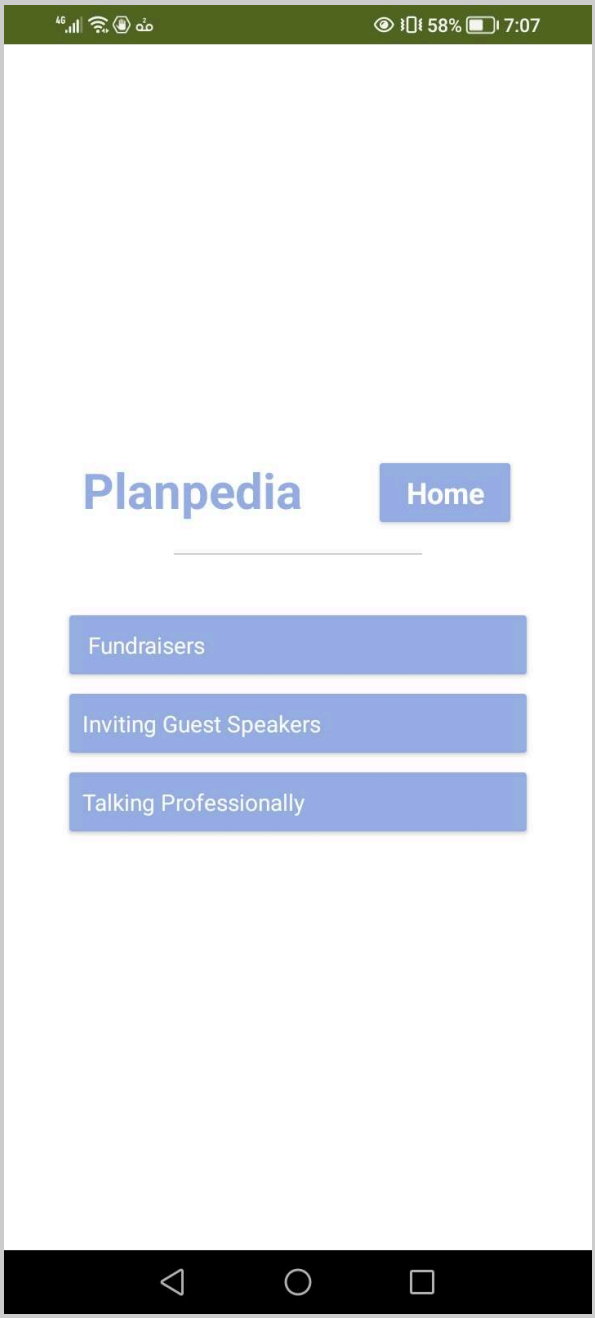


Depending on the button pressed, the program sends the user to the appropriate screen.

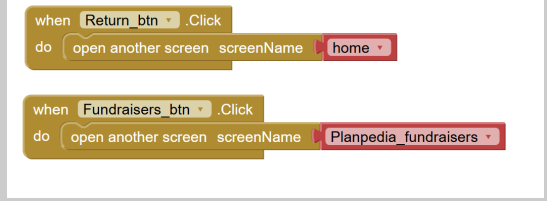
Planpedia

User Interface	**Name/Purpose**			
Buttons	Return_btn Sends user to the home screen	Fundraisers_btn Sends users to the Planpedia fundraiser article screen.	Guest_speak_btn Sends users to the planpedia guest speaker article. (Does not exist yet)	Talking_btn Send users to the plantpedia public speaking article. (Does not exist yet)
Password Textbox	PasswordTextbox 1 Non Functional text box used to place an aesthetic			

Planpedia				
	line.			
Labels	Title Displays the text “Plantpedia”.	Label1 Provides spacing for header		



Planpedia

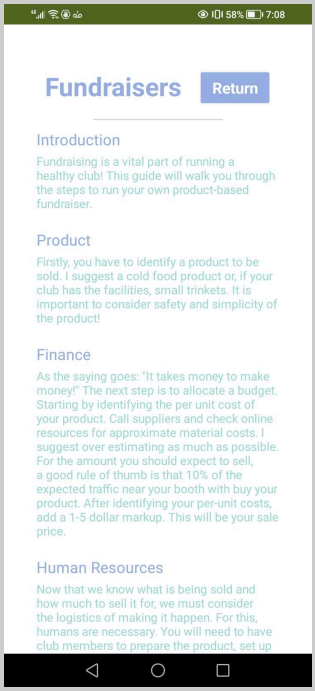


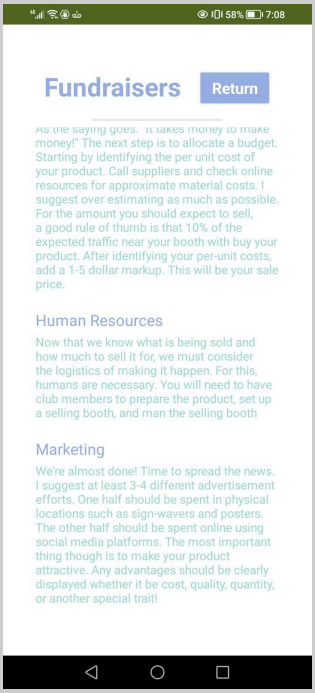
Sends the user to the screen of the selected option when that option button is pressed.

Planpedia_fundraiser

User Interface	**Name/Purpose**											
Buttons	Return_btn Sends user to the plantpedia hub screen											
Password Textbox	PasswordTextbox 1 Non Functional text box used to place an aesthetic line.											
Labels	Title Displays the text “Fundraiser”.	Subheading1 Subheader for introduction.	Paragraph1 Introduction article.	Subheading2 Subheader for product.	Paragraph2 Product article.	Subheading3 Subheader for finance.	Paragraph3 Finance article.	Subheading4 Subheader for human resources.	Paragraph4 Human resources article	Subheading5 Subheader for publicity.	Paragraph5 Publicity articles.	Label2 Bottom spacing for the text articles.

Planpedia_fundraiser



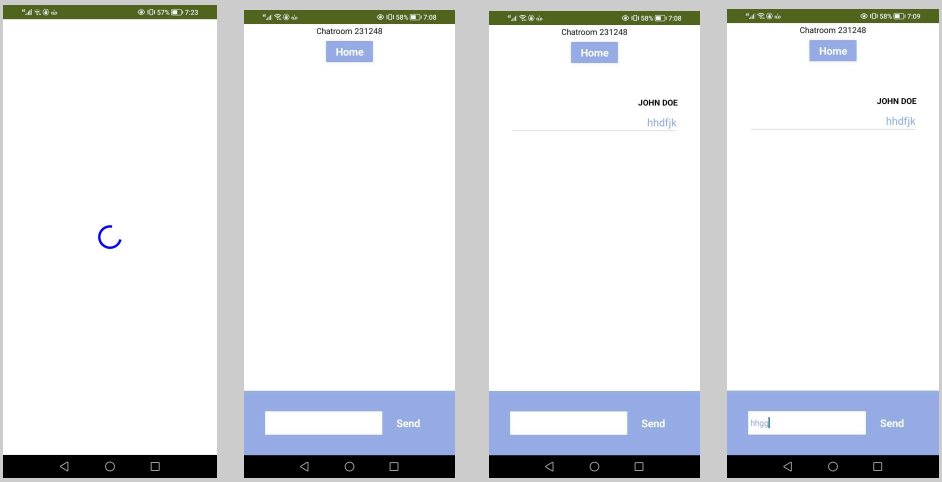


Planpedia_fundraiser

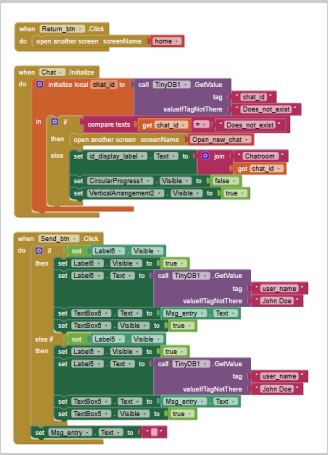
Sends the user to the plantpedia screen when the return button is pressed.

Chat			
User Interface	**Name/Purpose**		
Buttons	Return_btn Sends user to the home screen	Send_btn Displays the message that a user types.	
TinyDB	TinyDB1 Fetches the user's chat ID data.		
Circular progress	Circularprogress1 A spinning wheel displayed while loading the chat system.		
Textbox	Textbox6 Displays the message a user sends.	Textbox5 Displays the message a user sends.	Msg_entry Allows the user to type in a message.
Labels	Id_display_label Displays the ID of the current chat.	Label6 Empty text box used to display the user's name when they send a message.	Label5 Empty text box used to display the user's name when they send a message.

Chat


Chatroom 231248
Home
JOHN DOE
hhdfjk
Send
Chatroom 231248
Home
JOHN DOE
hhdfjk
Send
Chatroom 231248
Home
JOHN DOE
hhdfjk
Send

Chat

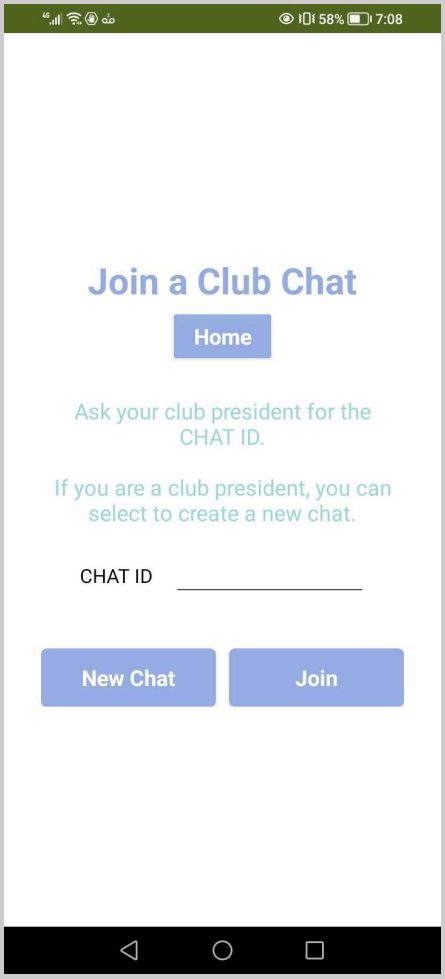


When the screen initializes, the program checks if the user is in a chat room via the chat_id tag in the database. If the user is not in a chatroom, the user is redirected to the open_new_chat screen. If the user is already in a chat, the program hides the spinning loading wheel and reveals the chatroom UI elements. The user can now interact with the program via two buttons. The return button will send the user back to the home screen. The send button will get the text stored in the msg_entry written by the user and paste it into the chatroom along with the user's name stored in the database.

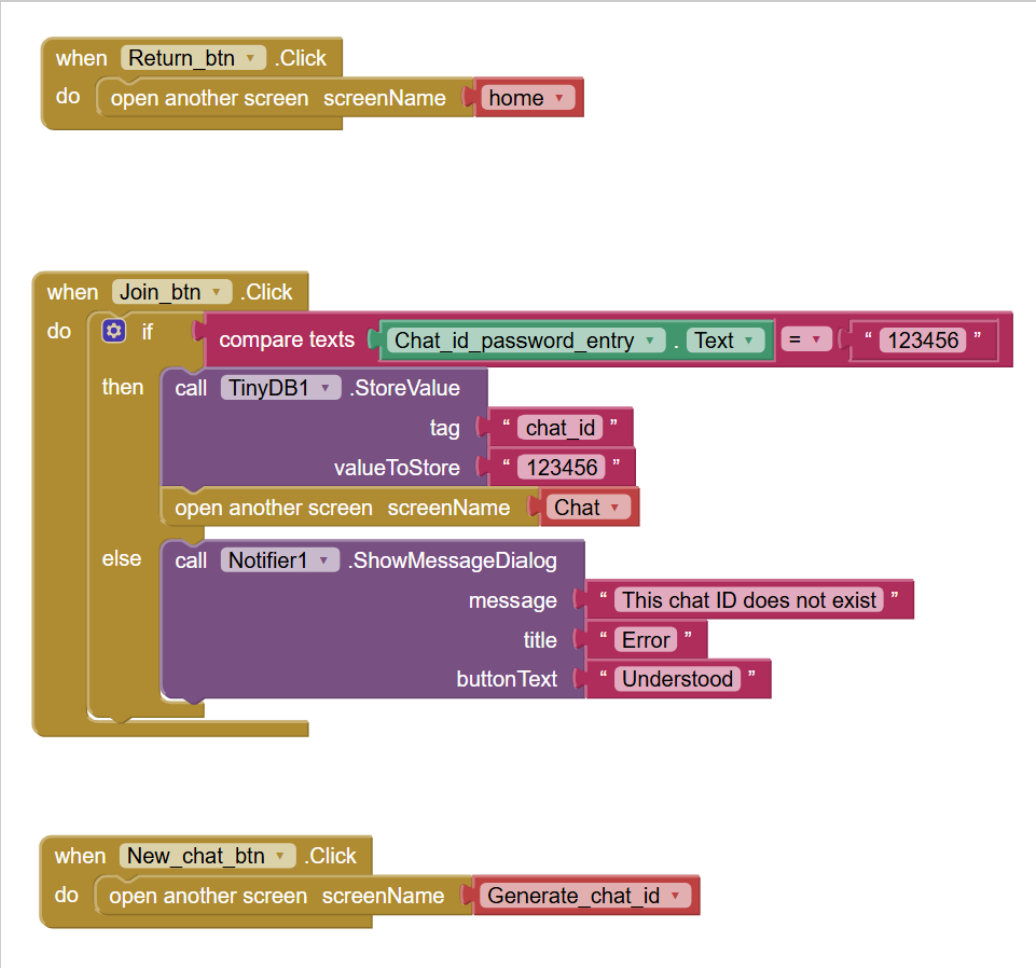
Open_new_chat								
User	**Name/Purpose**							

Open_new_chat								
Interface								
Buttons	Return_btn Send the user to the home screen.	New_chat_btn Sends the user to the Generate_chat_id screen.	Join_btn Stores the given chat ID and redirects the user to the chat screen.					
TinyDB	TinyDB1 Stores the entered chat ID into the user's profile.							
Notifier	Notifier1 Warns the user if the chat ID is invalid.							
Password Textbox	Chat_id_password_entry Provides a textbox for the user to enter their Chat ID.							
Labels	Title Displays the text "Join a Club Chat".	Label11 Provides title spacing.	Info_label Displays instructions on how to use the screen.	Label9 Provides spacing for the entry field.	Chat_id_label Displays the text "CHAT ID".	Label6 Provides spacing.	L:abel5 Provides spacing.	Label10 Provides spacing.

Open_new_chat



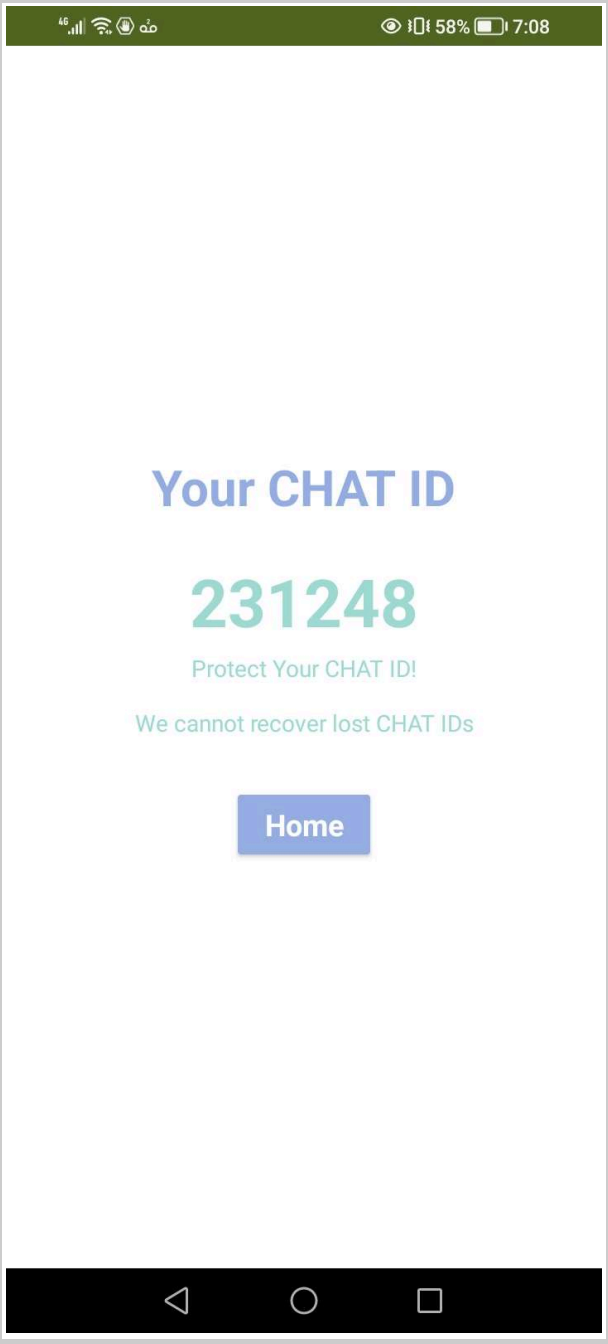
Open_new_chat



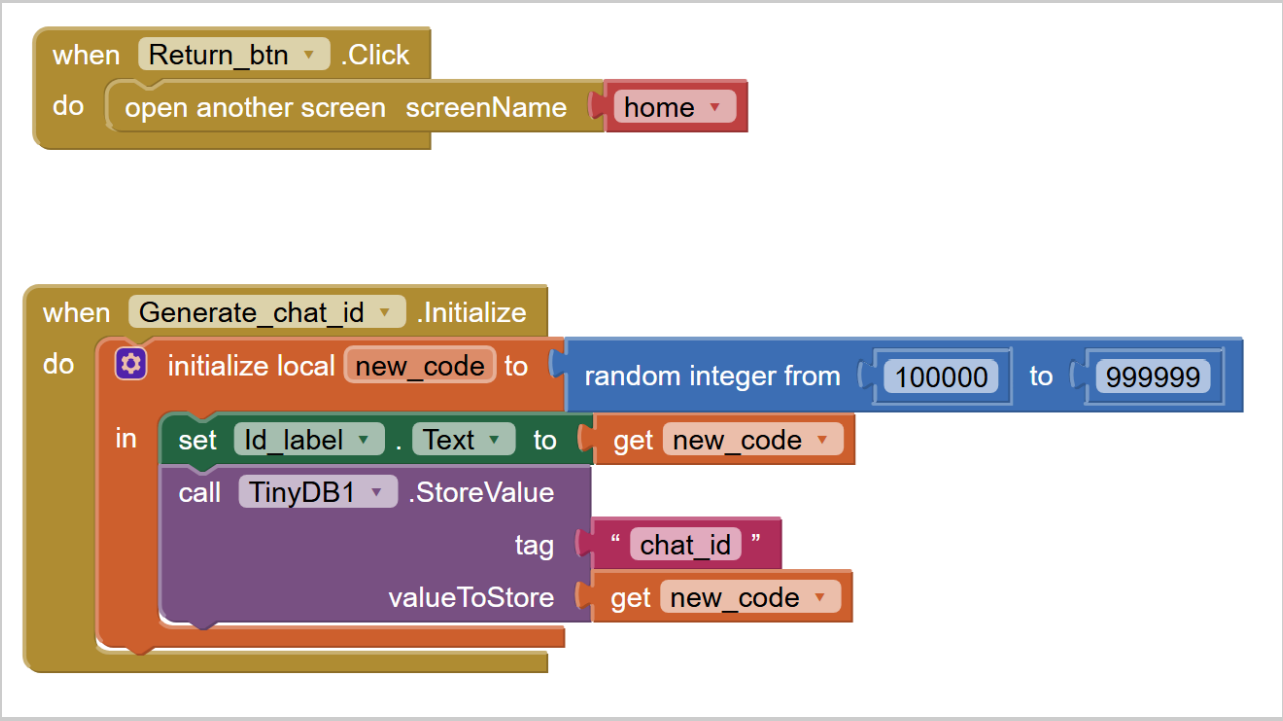
The user can press the return button and the program will send the user back to the home screen. The user can also press the new chat button and get redirected to the generate_chat_id screen. Otherwise, the user can input a chat ID into a text field. When the user presses the join button, the program will get the imputed 6-digit value and check if the ID is valid; the default chat ID I set was 123456. If the chat ID is valid, the program stores the ID into the database. If not, the notifier is called to give a warning.

Generate_chat_id					
User Interface	**Name/Purpose**				
Buttons	Return_btn Send the user to the home screen.				
TinyDB	TinyDB1 Stores the generated chat ID into the user's profile.				
Labels	Title Displays the text "Your CHAT ID".	Id_label Displays the generated chat ID.	Disclaimer_label Warns the user that the chat ID is not recoverable.	Label11 Provides spacing.	Label15 Provides spacing.

Generate_chat_id



Generate_chat_id



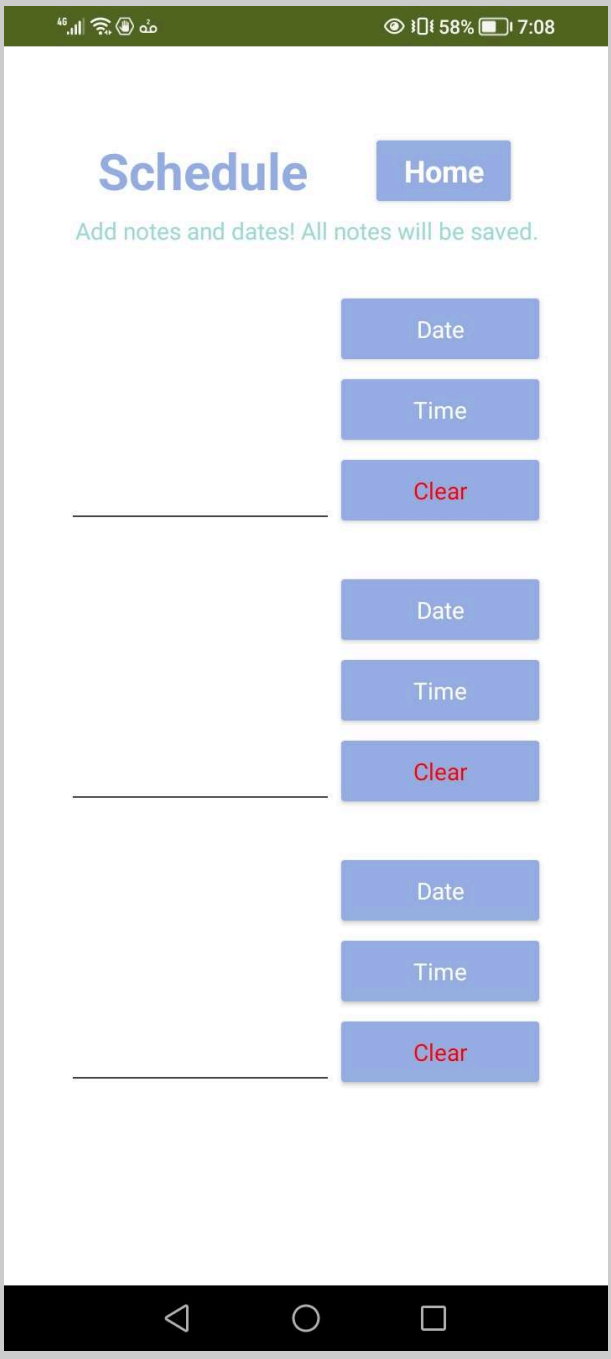
When the screen is initialized, the program pseudo randomly generates an integer from 100000 to 999999. The generated number is then displayed as the text of the ID label and stored to the database as the user's chatroom. The user can then press return to get sent to the home screen.

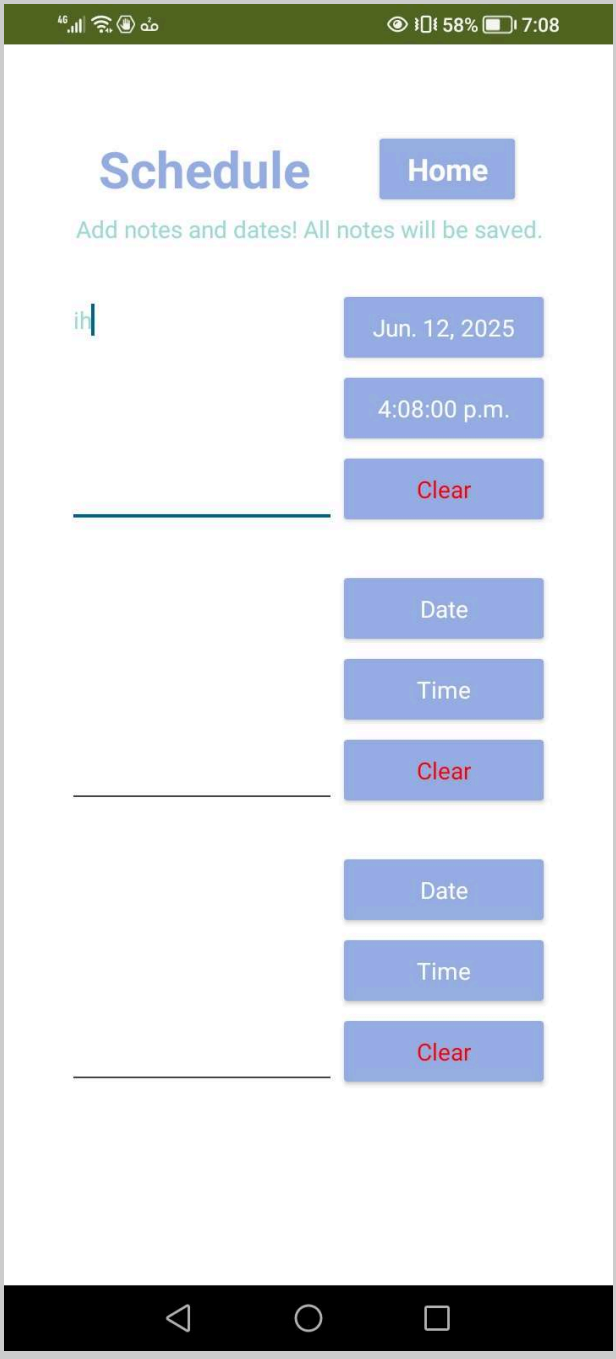
Schedule

User Interface	**Name/Purpose**				
Buttons	Return_btn Send the user to the home screen.	Clear_s1_btn Clears the entry fields and tinydb tags of all schedule1 data.			
TinyDB	TinyDB1 Fetches and stores the user's schedule data.				
Textbox	Schedule1	Schedule2	Schedule3		

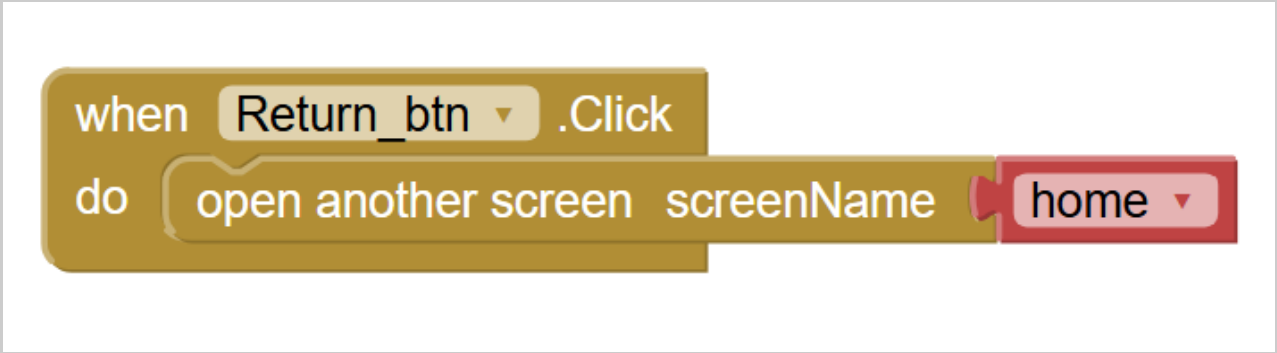
Schedule					
	Allows the user to write schedules for a day.	Allows the user to write schedules for a day.	Allows the user to write schedules for a day.		
Datepicker	Datepicker1 Allows the user to pick the date for schedule1.	Datepicker2 Allows the user to pick the date for schedule2.	Datepicker3 Allows the user to pick the date for schedule3.		
Timepicker	Timepicker1 Allows the user to pick the time for schedule1.	Timepicker2 Allows the user to pick the time for schedule2.	Timepicker3 Allows the user to pick the time for schedule3.		
Sensor	Clock1 Converts timepicker and datepicker instants into strings for storage and display.				
Labels	Title Displays the text “Schedule”.	Info_label Displays instructions for how to use the feature.	Label11 Provides spacing.	Label14 Provides spacing.	Label15 Provides spacing.

Schedule

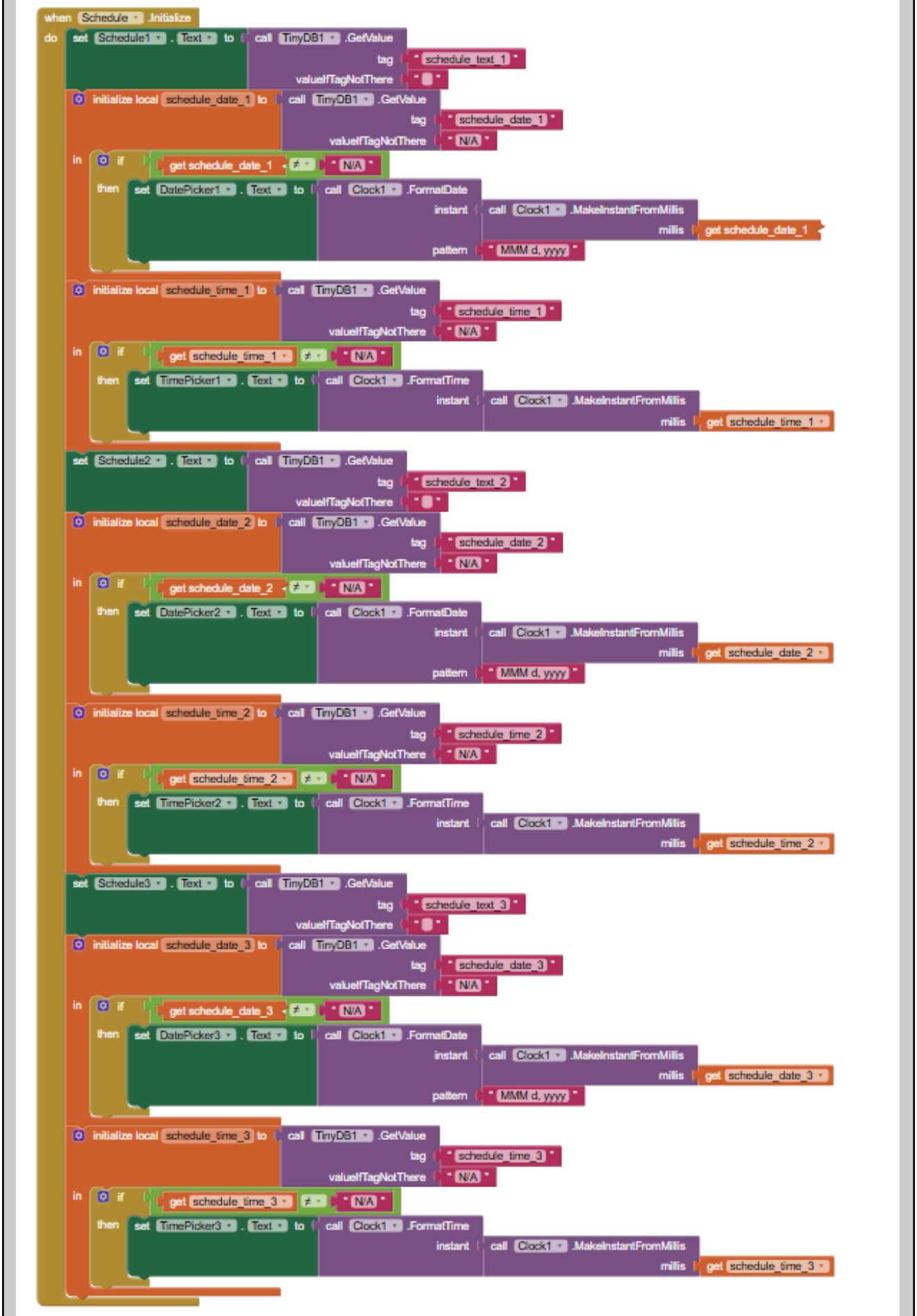

58% 7:08
Schedule
Home
Add notes and dates! All notes will be saved.
Date
Time
Clear
Date
Time
Clear
Date
Time
Clear


58% 7:08
Schedule
Home
Add notes and dates! All notes will be saved.
ih
Jun. 12, 2025
4:08:00 p.m.
Clear
Date
Time
Clear
Date
Time
Clear

Schedule



When the return button is pressed, the program sends the user to the home page.

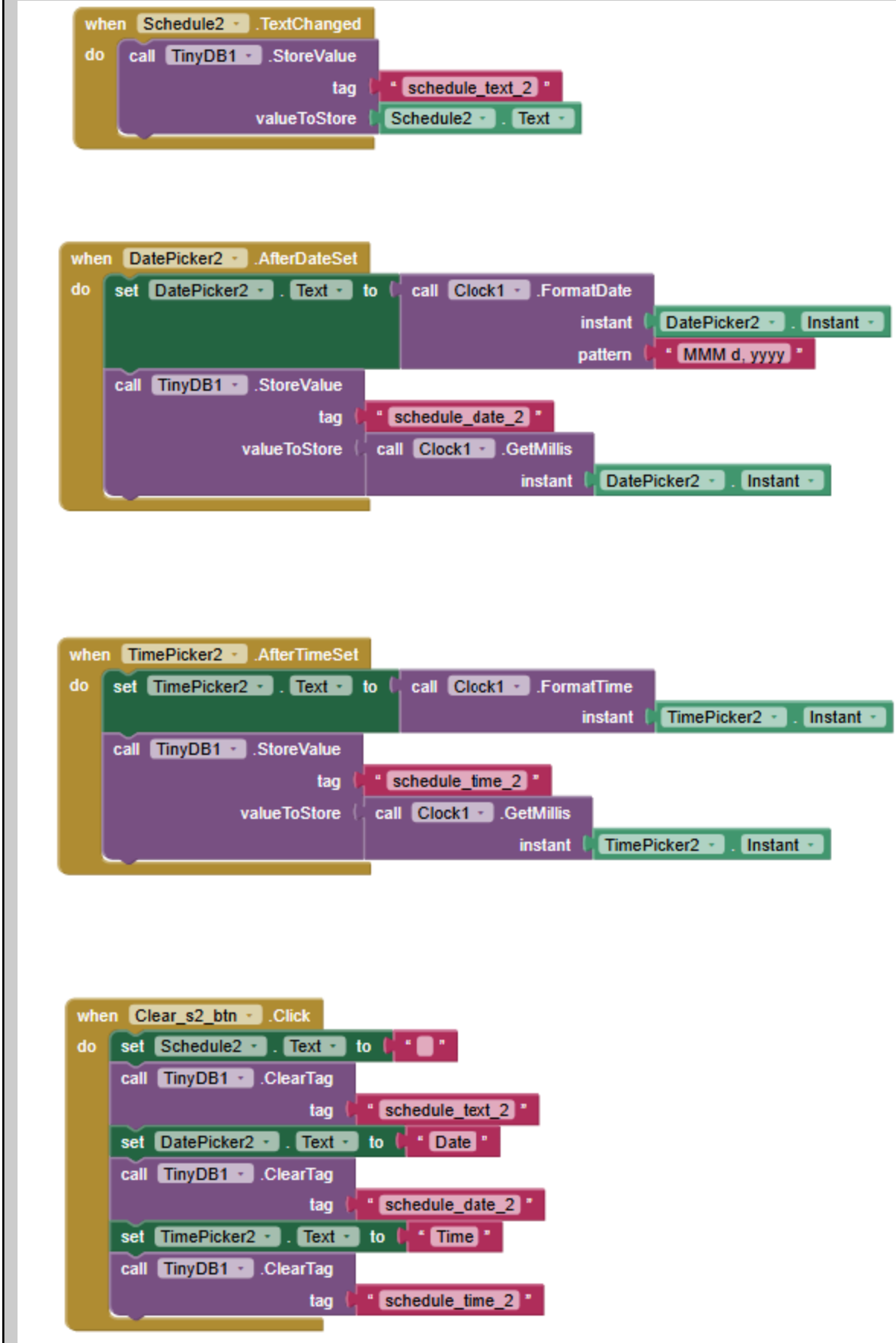

when Schedule .Initialize
do
set Schedule1 . Text to call TinyDB1 .GetValue
tag "schedule_text_1"
valueIfTagNotThere ""
initialize local schedule_date_1 to call TinyDB1 .GetValue
tag "schedule_date_1"
valueIfTagNotThere "N/A"
in if get schedule_date_1 ≠ "N/A"
then set DatePicker1 . Text to call Clock1 .FormatDate
instant call Clock1 .MakeInstantFromMillis
millis get schedule_date_1
pattern "MMM d, yyyy"
initialize local schedule_time_1 to call TinyDB1 .GetValue
tag "schedule_time_1"
valueIfTagNotThere "N/A"
in if get schedule_time_1 ≠ "N/A"
then set TimePicker1 . Text to call Clock1 .FormatTime
instant call Clock1 .MakeInstantFromMillis
millis get schedule_time_1
set Schedule2 . Text to call TinyDB1 .GetValue
tag "schedule_text_2"
valueIfTagNotThere ""
initialize local schedule_date_2 to call TinyDB1 .GetValue
tag "schedule_date_2"
valueIfTagNotThere "N/A"
in if get schedule_date_2 ≠ "N/A"
then set DatePicker2 . Text to call Clock1 .FormatDate
instant call Clock1 .MakeInstantFromMillis
millis get schedule_date_2
pattern "MMM d, yyyy"
initialize local schedule_time_2 to call TinyDB1 .GetValue
tag "schedule_time_2"
valueIfTagNotThere "N/A"
in if get schedule_time_2 ≠ "N/A"
then set TimePicker2 . Text to call Clock1 .FormatTime
instant call Clock1 .MakeInstantFromMillis
millis get schedule_time_2
set Schedule3 . Text to call TinyDB1 .GetValue
tag "schedule_text_3"
valueIfTagNotThere ""
initialize local schedule_date_3 to call TinyDB1 .GetValue
tag "schedule_date_3"
valueIfTagNotThere "N/A"
in if get schedule_date_3 ≠ "N/A"
then set DatePicker3 . Text to call Clock1 .FormatDate
instant call Clock1 .MakeInstantFromMillis
millis get schedule_date_3
pattern "MMM d, yyyy"
initialize local schedule_time_3 to call TinyDB1 .GetValue
tag "schedule_time_3"
valueIfTagNotThere "N/A"
in if get schedule_time_3 ≠ "N/A"
then set TimePicker3 . Text to call Clock1 .FormatTime
instant call Clock1 .MakeInstantFromMillis
millis get schedule_time_3

Schedule

When the screen is initialized, the program checks if any existing data for the text strings, date instants, or time instants for any of the three schedule slots is in the database. If there exists data, the data is displayed on the UI elements.

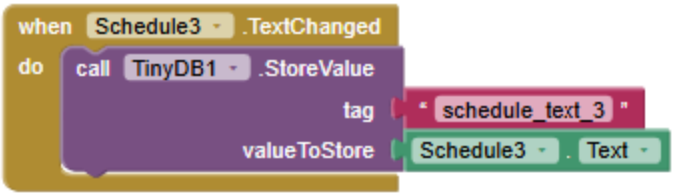
Detects changes to the schedule 1 text, timepicker, and datepicker to store into the database. When the clear schedule 1 button is pressed, it clears the text fields of the data and removes the data associated with schedule 1

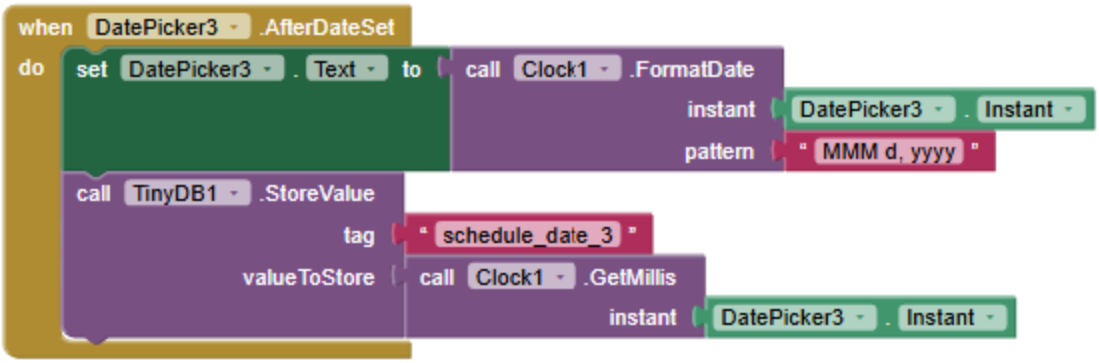
Schedule
from the database.

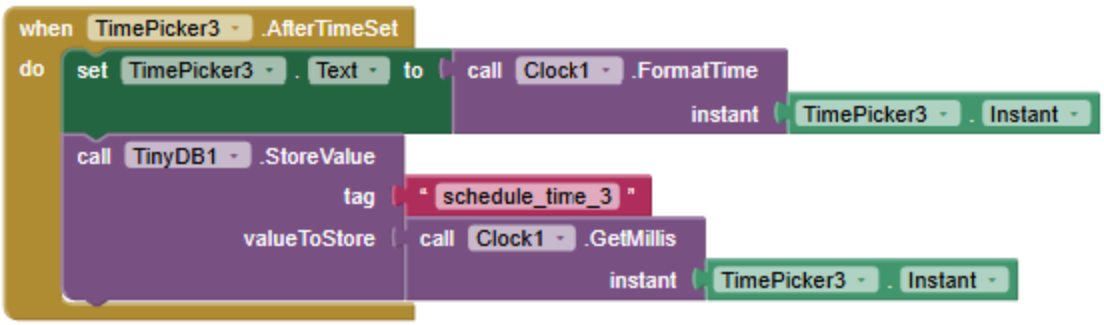

when Schedule2 .TextChanged
do call TinyDB1 .StoreValue
tag " schedule_text_2 "
valueToStore Schedule2 . Text
when DatePicker2 .AfterDateSet
do set DatePicker2 . Text to call Clock1 .FormatDate
instant DatePicker2 . Instant
pattern " MMM d, yyyy "
call TinyDB1 .StoreValue
tag " schedule_date_2 "
valueToStore call Clock1 .GetMillis
instant DatePicker2 . Instant
when TimePicker2 .AfterTimeSet
do set TimePicker2 . Text to call Clock1 .FormatTime
instant TimePicker2 . Instant
call TinyDB1 .StoreValue
tag " schedule_time_2 "
valueToStore call Clock1 .GetMillis
instant TimePicker2 . Instant
when Clear_s2_btn .Click
do set Schedule2 . Text to " "
call TinyDB1 .ClearTag
tag " schedule_text_2 "
set DatePicker2 . Text to " Date "
call TinyDB1 .ClearTag
tag " schedule_date_2 "
set TimePicker2 . Text to " Time "
call TinyDB1 .ClearTag
tag " schedule_time_2 "

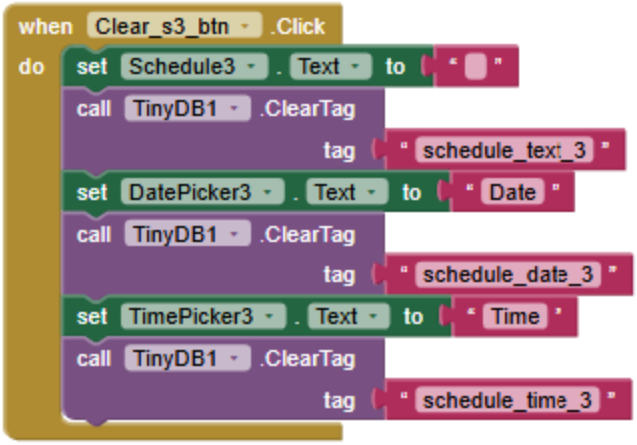
Schedule

Detects changes to the schedule2 text, timepicker, and datepicker to store into the database. When the clear schedule 2 button is pressed, it clears the text fields of the data and removes the data associated with schedule 2 from the database.


when Schedule3 .TextChanged
do call TinyDB1 .StoreValue
tag " schedule_text_3 "
valueToStore Schedule3 . Text


when DatePicker3 .AfterDateSet
do set DatePicker3 . Text to call Clock1 .FormatDate
instant DatePicker3 . Instant
pattern " MMM d, yyyy "
call TinyDB1 .StoreValue
tag " schedule_date_3 "
valueToStore call Clock1 .GetMillis
instant DatePicker3 . Instant


when TimePicker3 .AfterTimeSet
do set TimePicker3 . Text to call Clock1 .FormatTime
instant TimePicker3 . Instant
call TinyDB1 .StoreValue
tag " schedule_time_3 "
valueToStore call Clock1 .GetMillis
instant TimePicker3 . Instant


when Clear_s3_btn .Click
do set Schedule3 . Text to " "
call TinyDB1 .ClearTag
tag " schedule_text_3 "
set DatePicker3 . Text to " Date "
call TinyDB1 .ClearTag
tag " schedule_date_3 "
set TimePicker3 . Text to " Time "
call TinyDB1 .ClearTag
tag " schedule_time_3 "

Schedule

Detects changes to the schedule 3 text, timepicker, and datepicker to store into the database. When the clear schedule 3 button is pressed, it clears the text fields of the data and removes the data associated with schedule 3 from the database.