UNIVERSITY OF WOLLONGONG AUSTRALIA

CSIT321 Final Year Project
Project Progress Report
27 Mar 2021

Project Group No: FYP-21-S1-02

Project Topic: Typing Habit Gesture Authentication System

Project Supervisor: Mr Sionggo Japit

Project Members:

Name	UOW ID	Email
Alicia Tan Chun Ying	6457794	cyatan005@mymail.sim.edu.sg
Terrence Yap Yong Ting	6573307	yttyap002@mymail.sim.edu.sg
Tng Min Li	6649737	mltng001@mymail.sim.edu.sg
Chea Cui Hui	6647546	chchea001@mymail.sim.edu.sg
Joel Teo Han Wen	6456467	jhwteo002@mymail.sim.edu.sg

Project Website: https://fyp21s102.wixsite.com/fyp21s102

Table of Contents

1 Introduction

Overview

In the digital age, security is of great importance. Authentication has become common activity in daily life. As passwords pose a risk of being compromised due to a variety of reasons such as reusing passwords or using simple passwords, exploring a different approach to authentication could be useful.
The purpose of this project is to develop a typing gesture habit authentication system that can authenticate a person based on his typing habits.

Project Scope

Project Objective

Complete a working typing gesture habit authentication system that can recognize user based on typing habit.

1.2.2 Goals
The website must be able to recognize the user based on their typing habits.

1.2.3 Milestones

- Literature review
- Project specification report
- Progress report
- Coding of system

Project Description

This project is to authenticate users based on their typing habits instead of using passwords. Through analysis of the typing patterns using attributes that are recorded from the user, analysis would be performed on them and then determined if they match the typing habits of the user when he initially registered.

1 Feasibility Studies

Competitors Analysis

There are many analyses done about typing gesture habit authenticated system and one of our competitors is TypingDNA.

TypingDNA is an existing system that is available in the market and they introduce themselves as “A smarter, user-friendly authentication that replaces SMS 2FA codes and reduces costs by an order of magnitude.”.

It recognizes users by the way they type on their keyboards as their typing biometrics engine, which is exposed by a RESTful API, is able to analyze typing patterns and accurately determines if they are a match with a known, enrolled user. This allows TypingDNA to protect trusted user accounts with powerful typing biometrics analysis, accurately and passively. (Authentication API Documentation, 2020)

It can be deployed in scenarios in which, the identity of a user needs confirmation -- such as enforcing password resets, complimenting primary authentication means, with an additional layer of security, or in the place of OTP flows. The API is not constrained to specific use-cases or authentication stacks, and therefore can be incorporated anywhere within the architecture that end-users are typing.

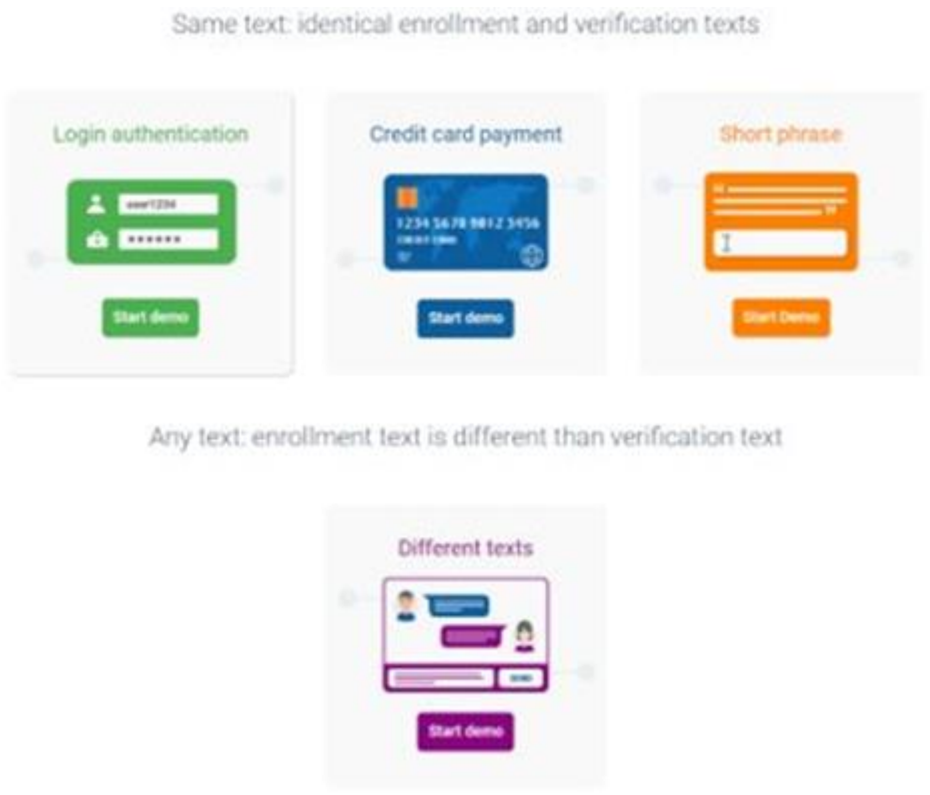
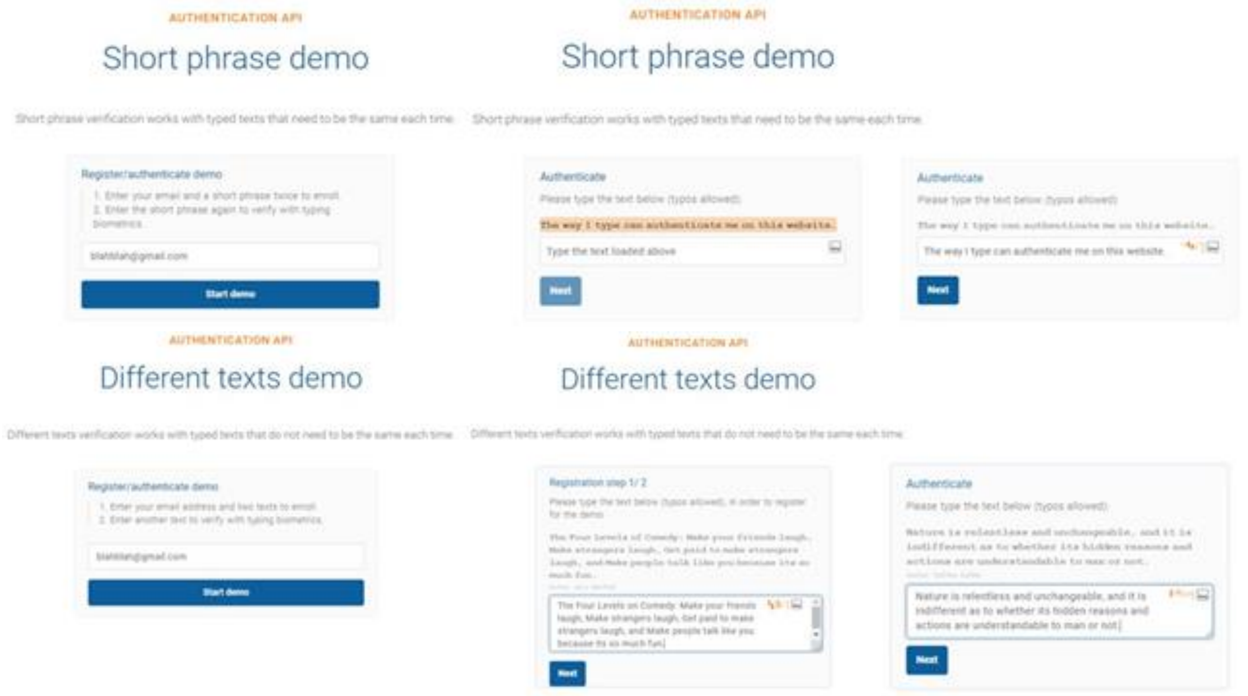

Same text: identical enrollment and verification texts
Login authentication
Start demo
Credit card payment
Start demo
Short phrase
Start Demo
Any text: enrollment text is different than verification text
Different texts
Start demo

Figure 1: Demos available for TypingDNA

From the diagram above (refer to Figure 1), we see that TypingDNA has 4 demos where it is categorized into 2 groups. The first group are the ones where it uses the same text, for example identical enrollment and verification text, while the other group are the ones where it uses any text, for example the enrollment text is different than the verification text.

We will be analyzing more about Short Phase and Different Text as it is more in line with our project specifications.


AUTHENTICATION API

Short phrase demo

Short phrase verification works with typed texts that need to be the same each time.

Register/authenticate demo

1. Enter your email and a short phrase twice to enroll.
2. Enter the short phrase again to verify with typing biometrics.

blahblah@gmail.com

Start demo

AUTHENTICATION API

Short phrase demo

Short phrase verification works with typed texts that need to be the same each time.

Authenticate

Please type the text below (typos allowed):

The way I type can authenticate me on this website.

Type the text loaded above

Next

Authenticate

Please type the text below (typos allowed):

The way I type can authenticate me on this website.

The way I type can authenticate me on this website

Next

AUTHENTICATION API

Different texts demo

Different texts verification works with typed texts that do not need to be the same each time.

Register/authenticate demo

1. Enter your email address and two texts to enroll.
2. Enter another text to verify with typing biometrics.

blahblah@gmail.com

Start demo

AUTHENTICATION API

Different texts demo

Different texts verification works with typed texts that do not need to be the same each time.

Registration step 1/2

Please type the text below (typos allowed), in order to register for the demo.

The Four Levels of Comedy: Make your friends laugh, Make strangers laugh, Get paid to make strangers laugh, and Make people talk like you because its so much fun.

The Four Levels on Comedy: Make your friends laugh, Make strangers laugh, Get paid to make strangers laugh, and Make people talk like you because its so much fun|

Next

Authenticate

Please type the text below (typos allowed):

Nature is relentless and unchangeable, and it is indifferent as to whether its hidden reasons and actions are understandable to man or not.

Nature is relentless and unchangeable, and it is indifferent as to whether its hidden reasons and actions are understandable to man or not|

Next

Figure 2: Short phrase demo vs Different texts demo from TypingDNA

For both Short Phase and Different Text, we can see that they have similar flow where we are required to register with our emails. But after that, the Short Phase demo show that the input statement is the same for both segments, whereas Different Text demo shows 2 different paragraphs.

Figure 3: Typing pattern visualization from TypingDNA

This small icon (Refer to figure 3) that you see within the right side of the input text box shows our typing pattern which consists of data on the timing and durations of various key press events. This data is then used for the verification process.

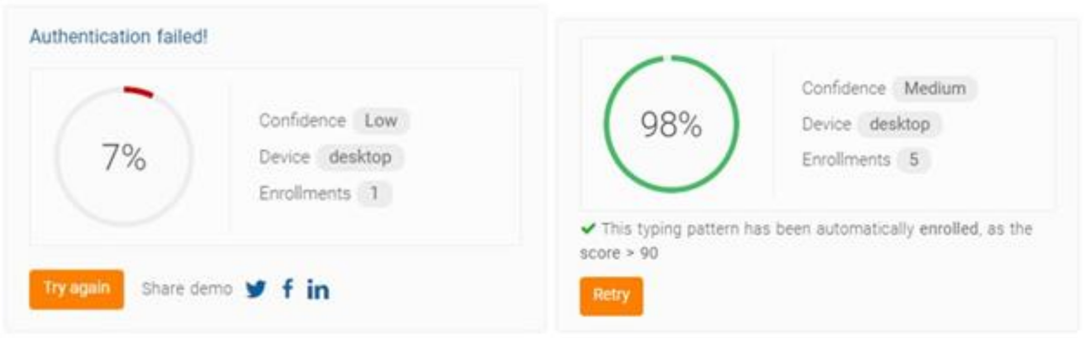
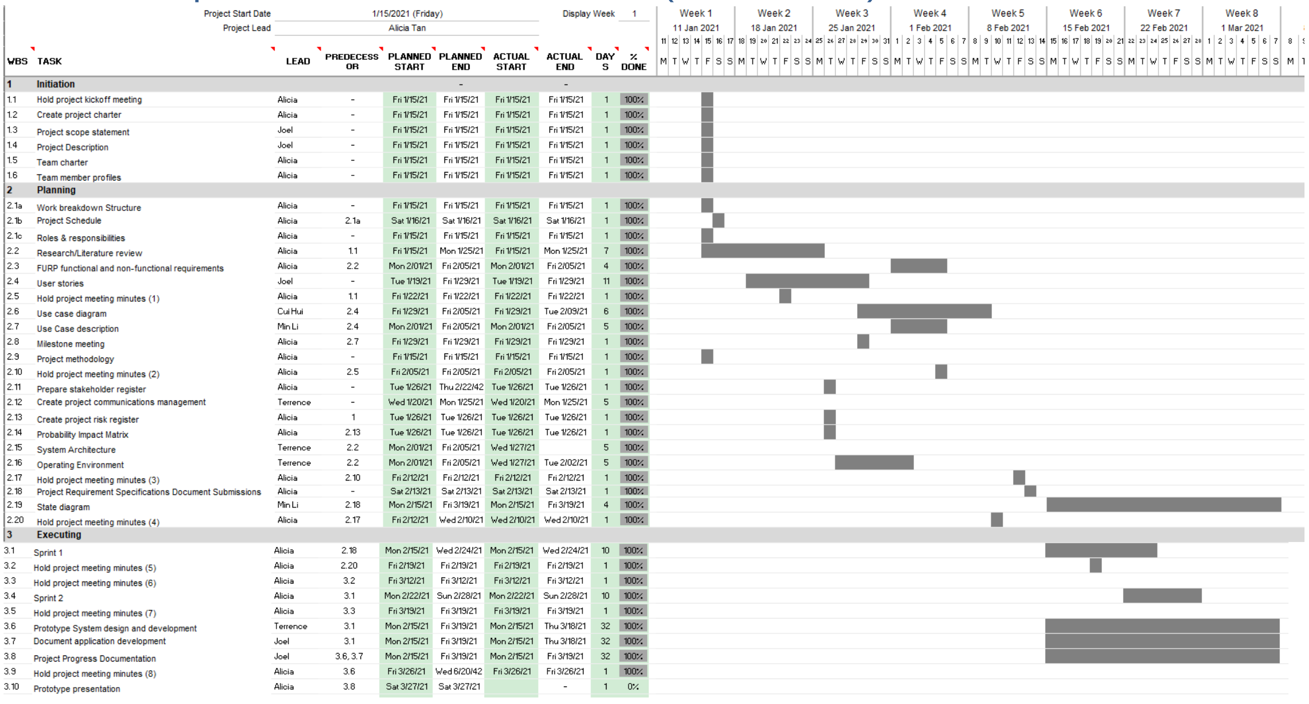


Figure 4: Authentication Pass vs Fail from TypingDNA

After Verification process, TypingDNA is then able to analyze and show the outputs as shown above. These 2 different outputs show the differences between Pass and Failed. As seen, the ones that have passed would have the typing pattern automatically enrolled into the system. (Try Our Demo, n.d.)

2 Work completed

List of Completed Work Items/Activities (Gantt Chart)



	Project Start Date	1/15/2021 (Friday)					Display Week	1	
	Project Lead	Alicia Tan							

WBS	TASK	LEAD	PREDECESSOR	PLANNED START	PLANNED END	ACTUAL START	ACTUAL END	DAYS	% DONE
1	**Initiation**				-		-		
1.1	Hold project kickoff meeting	Alicia	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
1.2	Create project charter	Alicia	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
1.3	Project scope statement	Joel	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
1.4	Project Description	Joel	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
1.5	Team charter	Alicia	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
1.6	Team member profiles	Alicia	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
2	**Planning**								
2.1a	Work breakdown Structure	Alicia	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
2.1b	Project Schedule	Alicia	2.1a	Sat 1/16/21	Sat 1/16/21	Sat 1/16/21	Sat 1/16/21	1	100%
2.1c	Roles & responsibilities	Alicia	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
2.2	Research/Literature review	Alicia	1.1	Fri 1/15/21	Mon 1/25/21	Fri 1/15/21	Mon 1/25/21	7	100%
2.3	FURP functional and non-functional requirements	Alicia	2.2	Mon 2/01/21	Fri 2/05/21	Mon 2/01/21	Fri 2/05/21	4	100%
2.4	User stories	Joel	-	Tue 1/19/21	Fri 1/29/21	Tue 1/19/21	Fri 1/29/21	11	100%
2.5	Hold project meeting minutes (1)	Alicia	1.1	Fri 1/22/21	Fri 1/22/21	Fri 1/22/21	Fri 1/22/21	1	100%
2.6	Use case diagram	Cui Hui	2.4	Fri 1/29/21	Fri 2/05/21	Fri 1/29/21	Tue 2/09/21	6	100%
2.7	Use Case description	Min Li	2.4	Mon 2/01/21	Fri 2/05/21	Mon 2/01/21	Fri 2/05/21	5	100%
2.8	Milestone meeting	Alicia	2.7	Fri 1/29/21	Fri 1/29/21	Fri 1/29/21	Fri 1/29/21	1	100%
2.9	Project methodology	Alicia	-	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	Fri 1/15/21	1	100%
2.10	Hold project meeting minutes (2)	Alicia	2.5	Fri 2/05/21	Fri 2/05/21	Fri 2/05/21	Fri 2/05/21	1	100%
2.11	Prepare stakeholder register	Alicia	-	Tue 1/26/21	Thu 2/22/42	Tue 1/26/21	Tue 1/26/21	1	100%
2.12	Create project communications management	Terrence	-	Wed 1/20/21	Mon 1/25/21	Wed 1/20/21	Mon 1/25/21	5	100%
2.13	Create project risk register	Alicia	1	Tue 1/26/21	Tue 1/26/21	Tue 1/26/21	Tue 1/26/21	1	100%
2.14	Probability Impact Matrix	Alicia	2.13	Tue 1/26/21	Tue 1/26/21	Tue 1/26/21	Tue 1/26/21	1	100%
2.15	System Architecture	Terrence	2.2	Mon 2/01/21	Fri 2/05/21	Wed 1/27/21		5	100%
2.16	Operating Environment	Terrence	2.2	Mon 2/01/21	Fri 2/05/21	Wed 1/27/21	Tue 2/02/21	5	100%
2.17	Hold project meeting minutes (3)	Alicia	2.10	Fri 2/12/21	Fri 2/12/21	Fri 2/12/21	Fri 2/12/21	1	100%
2.18	Project Requirement Specifications Document Submissions	Alicia	-	Sat 2/13/21	Sat 2/13/21	Sat 2/13/21	Sat 2/13/21	1	100%
2.19	State diagram	Min Li	2.18	Mon 2/15/21	Fri 3/19/21	Mon 2/15/21	Fri 3/19/21	4	100%
2.20	Hold project meeting minutes (4)	Alicia	2.17	Fri 2/12/21	Wed 2/10/21	Wed 2/10/21	Wed 2/10/21	1	100%
3	**Executing**								
3.1	Sprint 1	Alicia	2.18	Mon 2/15/21	Wed 2/24/21	Mon 2/15/21	Wed 2/24/21	10	100%
3.2	Hold project meeting minutes (5)	Alicia	2.20	Fri 2/19/21	Fri 2/19/21	Fri 2/19/21	Fri 2/19/21	1	100%
3.3	Hold project meeting minutes (6)	Alicia	3.2	Fri 3/12/21	Fri 3/12/21	Fri 3/12/21	Fri 3/12/21	1	100%
3.4	Sprint 2	Alicia	3.1	Mon 2/22/21	Sun 2/28/21	Mon 2/22/21	Sun 2/28/21	10	100%
3.5	Hold project meeting minutes (7)	Alicia	3.3	Fri 3/19/21	Fri 3/19/21	Fri 3/19/21	Fri 3/19/21	1	100%
3.6	Prototype System design and development	Terrence	3.1	Mon 2/15/21	Fri 3/19/21	Mon 2/15/21	Thu 3/18/21	32	100%
3.7	Document application development	Joel	3.1	Mon 2/15/21	Fri 3/19/21	Mon 2/15/21	Thu 3/18/21	32	100%
3.8	Project Progress Documentation	Joel	3.6, 3.7	Mon 2/15/21	Fri 3/19/21	Mon 2/15/21	Fri 3/19/21	32	100%
3.9	Hold project meeting minutes (8)	Alicia	3.6	Fri 3/26/21	Wed 6/20/42	Fri 3/26/21	Fri 3/26/21	1	100%
3.10	Prototype presentation	Alicia	3.8	Sat 3/27/21	Sat 3/27/21		-	1	0%

Figure 5: Top of the gantt chart showing the completed progress

Sprint 1

User Stories

#2 As a user, they should be able to log out so that they can exit the application safely.
#3 As a user, they should be able to view a sentence upon login so that they can be able to type and authenticate themselves
#4 As a user, they should be able to see the generated sentence so that they can type it for authentication
#11 As an admin, they should be able to log out so that they can exit the system

Use Case Diagrams

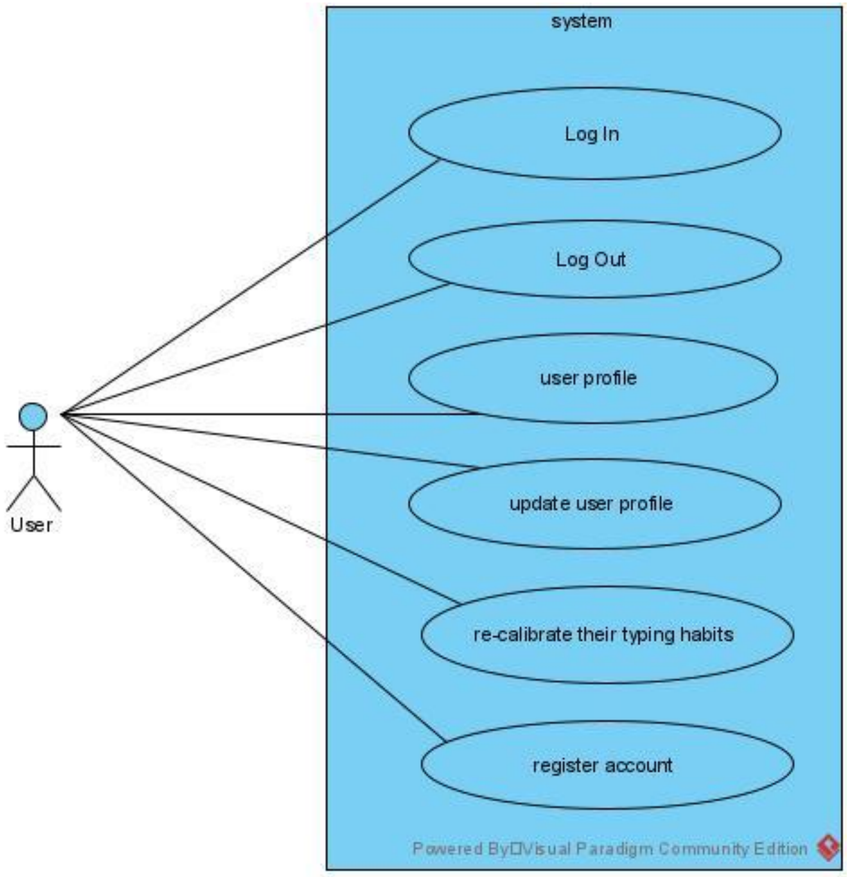

system
Log In
Log Out
user profile
update user profile
re-calibrate their typing habits
register account
User
Powered By Visual Paradigm Community Edition

Figure 6: Use case diagram for User role

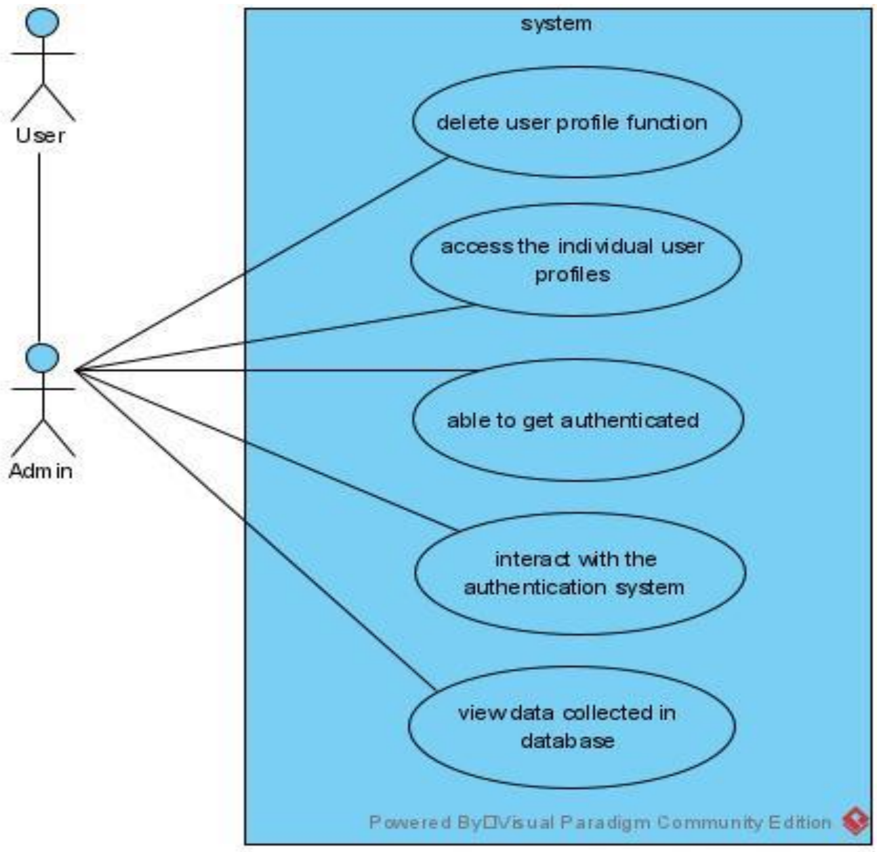

system
User
Admin
delete user profile function
access the individual user profiles
able to get authenticated
interact with the authentication system
view data collected in database
Powered By Visual Paradigm Community Edition

Figure 7: Use case diagram for Admin role

Use Case Description

User: User Story #2

Table 1: User Story #2

User Story #2: As a user, they should be able to log out so that they can exit the application safely.	
Name: Logout(user.py)	**Taiga ID:** 2
Stakeholders and Goals: User wants to logout from the Typing Habit Gesture Authentication System.	
Descriptions: The user is able to log out from the system.	
Actors: User	
Trigger: 1. User clicks on the Logout button on their user page.	
Normal Flow: 1. User clicks on the Logout button on their user page. 2. The system retrieves and displays the Welcome page. 3. End.	
Sub flows: None	
Alternate/Exception flows: 2a The system is unable to retrieve the welcome page. The system will display an error message. The use case terminates.	

User: User Story #3

Table 2: User Story #3

User Story #3: As a user, they should be able to see the generated sentence so that they can type it for authentication.	
Name: generateSentence(TypingHabitUser.py)	**Taiga ID:** 3
Stakeholders and Goals: User needs to be able to type the generated sentence.	
Descriptions: The user is able to type the generated sentence in the blank space provided below the shown sentence.	
Actors: User	
Trigger: 1. User clicks on the submit button on their Login page after entering username.	
Normal Flow: 1. User clicks on the submit button on their login page after entering username. 2. The system retrieves sentence and display on the TypingHabitUser page. 3. End.	
Sub flows: 2a User selects the Back option The system shows the login page where user can enter his username. 2b User selects the Refresh option The system will show a different sentence for authentication.	
Alternate/Exception flows: 2a The system is unable to retrieve the Login page. The system will display an error message. The use case terminates. 2b The system is unable to display the sentence. The system will display an error message. The use case terminates.	

User: User Story #4

Table 3: User Story #4

User Story #4: As a user, they should be able to view a sentence upon login so that they can be able to type and authenticate themselves.	
Name: generateSentence(TypingHabitUser.py)	**Taiga ID:** 4
Stakeholders and Goals: User needs to be able to view the generated sentence.	
Descriptions: The user is able to view the generated sentence.	
Actors: User	
Trigger: 1. User clicks on the submit button on their login page after entering username.	
Normal Flow: 1. User clicks on the submit button on their login page after entering username. 2. The system retrieves and displays the TypingHabitUser page. 3. User types the sentence in the blank space provided. 4. End.	
Sub flows: 2a User selects the Back option The system shows the login page where user can enter his username. 2b User selects the Refresh option The system will show a different sentence for authentication.	
Alternate/Exception flows: 2a The system is unable to retrieve the Login page. The system will display an error message. The use case terminates. 2b The system is unable to display the sentence. The system will display an error message. The use case terminates.	

Admin: User Story #11

Table 4: User Story #11

User Story #11: As an admin, they should be able to log in as admin so that they can be distinguished as a different role.	
Name: checkUser_Name (Login.py)	**Taiga ID:** 10
Stakeholders and Goals: Admin needs to be able to log in and see the admin page.	
Descriptions: Admin needs to be able to log in and see the admin page, which has admin functions that normal users do not have.	
Actors: Admin	
Trigger: 1. Admin clicks on the Submit button on the Login page (Login.py).	
Normal Flow: 1. Admin enters username and clicks the submit button. 2. System verifies username, retrieves and display the TypingHabitAdmin page to verify their typing habit (TypingHabitAdmin.py). 3. Admin types the generated sentence, clicks on the Log In button on their TypingHabitAdmin page 4. The system retrieves and displays admin page (admin.py). 5. End.	
Sub flows: 2a User selects the Back option The system shows the login page where user can enter his username. 2b User selects the Refresh option The system will show a different sentence for authentication.	
Alternate/Exception flows: 2a The system is unable to retrieve the login page. The system will display an error message. The use case terminates. 2b The system is unable to display the sentence. The system will display an error message. The use case terminates.	

Sequence Diagram

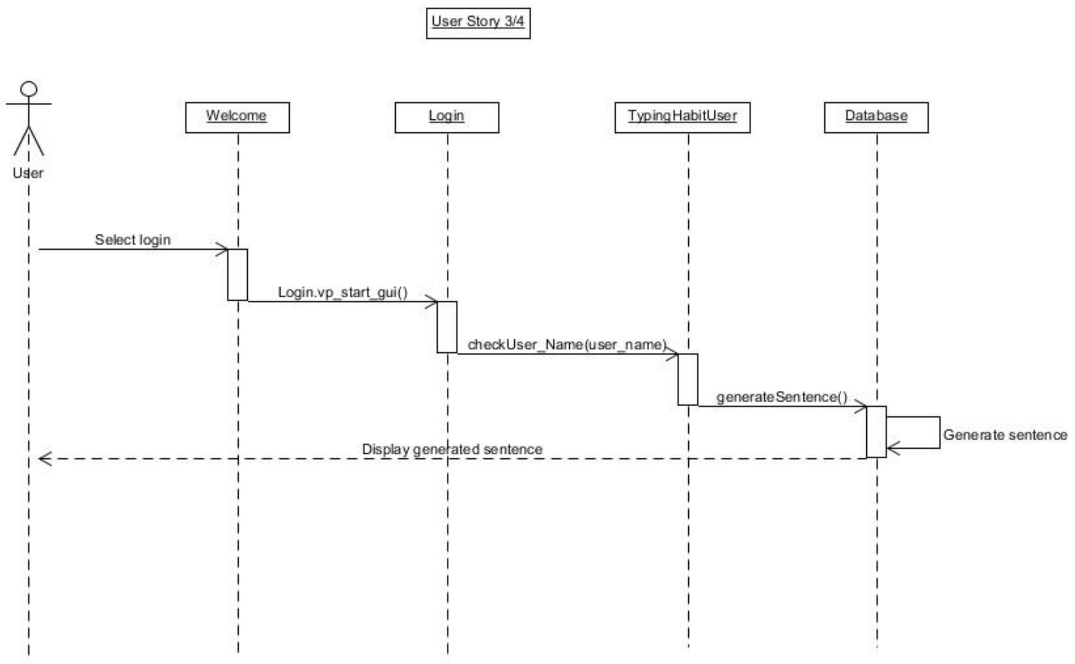


Figure 8: Sequence diagram for user story 3 and 4 (Able to see generated sentence)

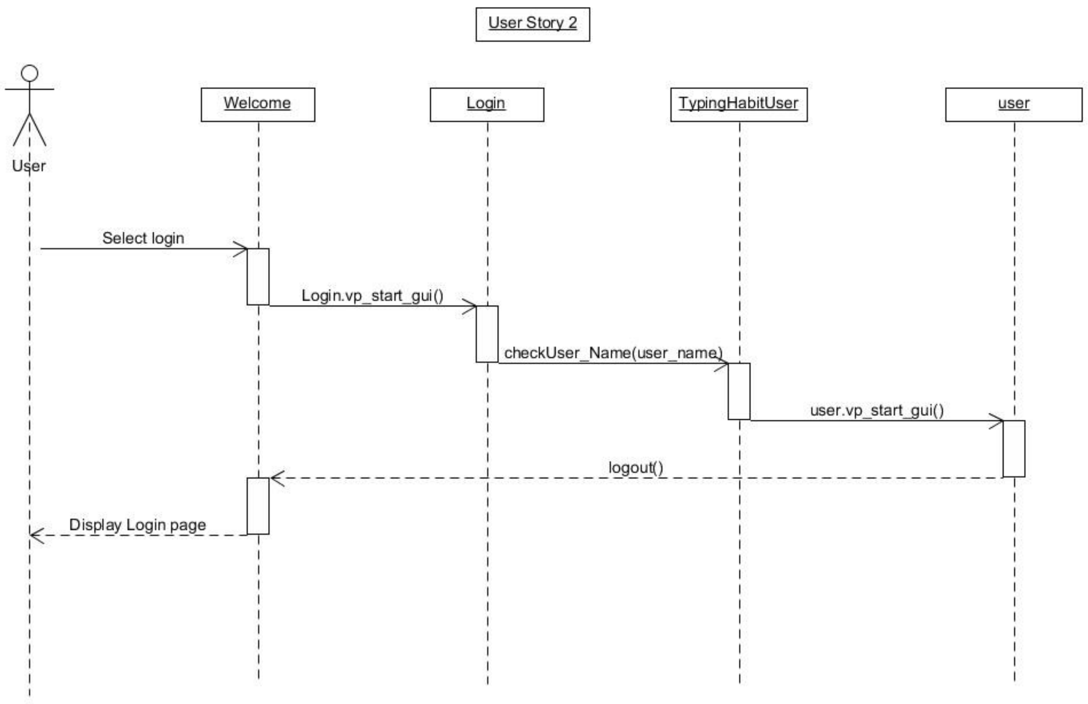


Figure 8: Sequence diagram for user story 2 (Able to log out safely)

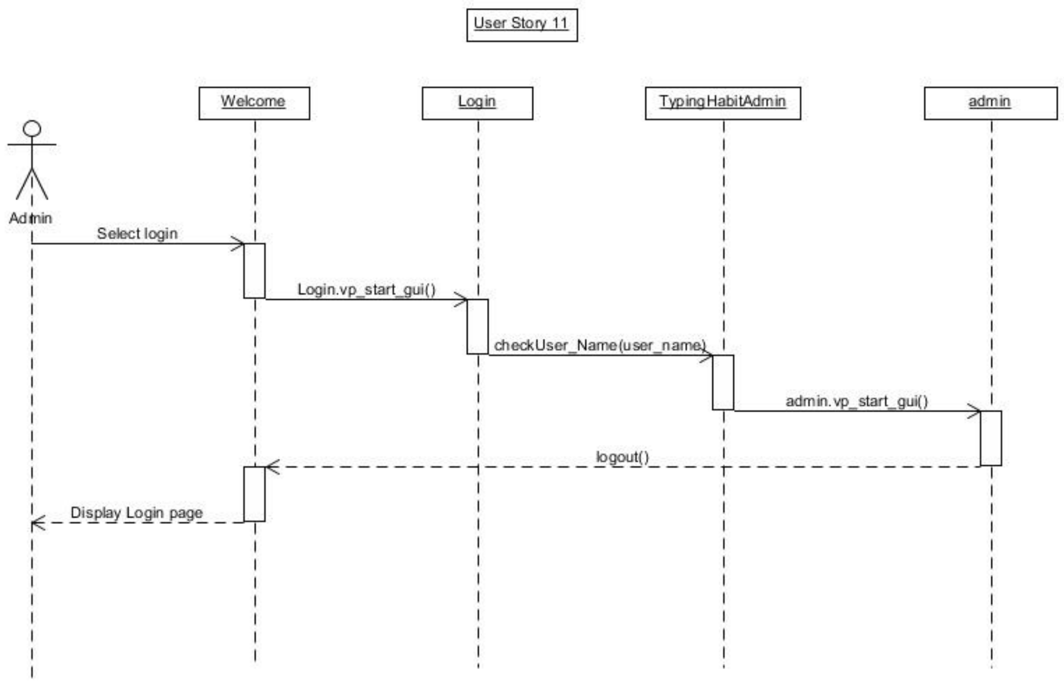


Figure 9: Sequence diagram for user story 11 (Log in as admin)

Wireframes

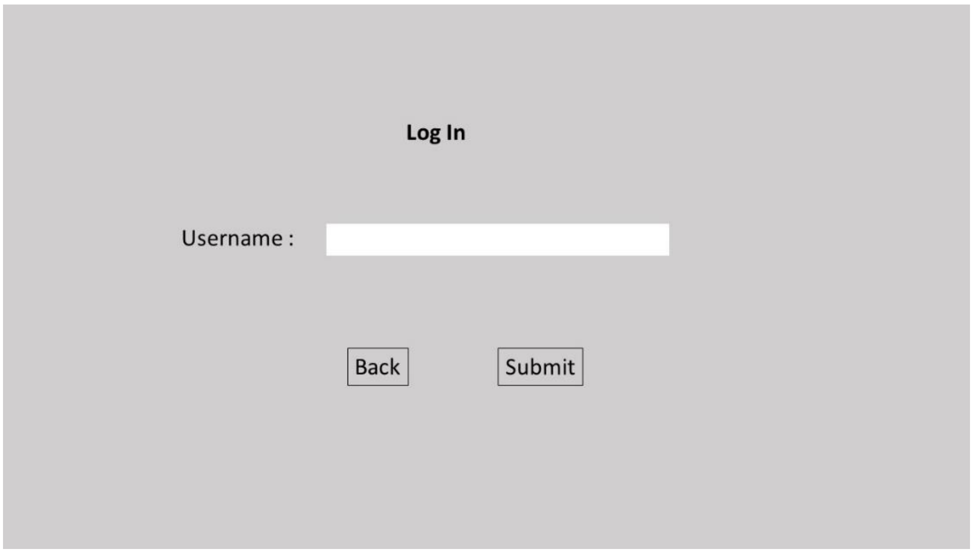


Figure 9: Wireframe for login.py (Login page)

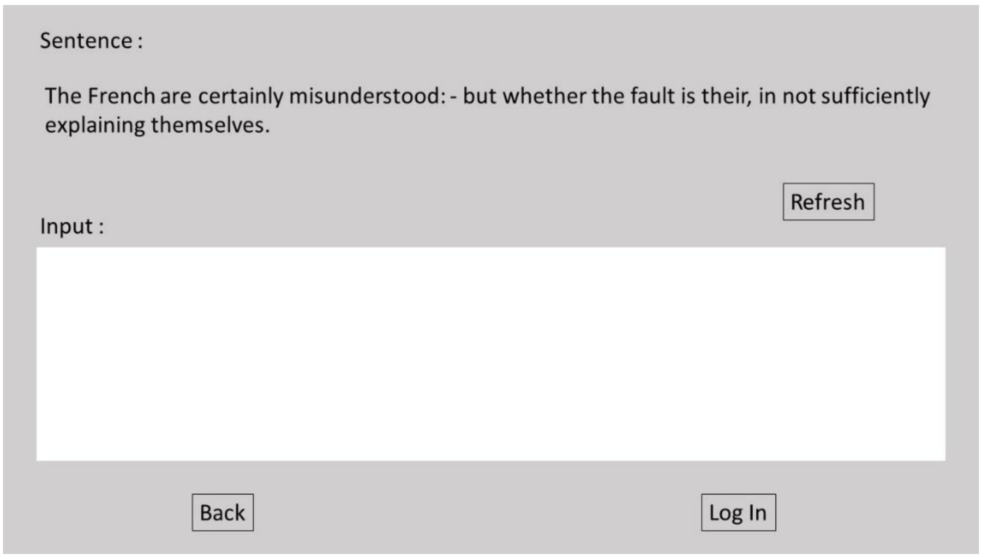

Sentence :
The French are certainly misunderstood: - but whether the fault is their, in not sufficiently explaining themselves.
Refresh
Input :
Back
Log In

Figure 10: Wireframe for TypingHabitUser.py and TypingHabitAdmin.py

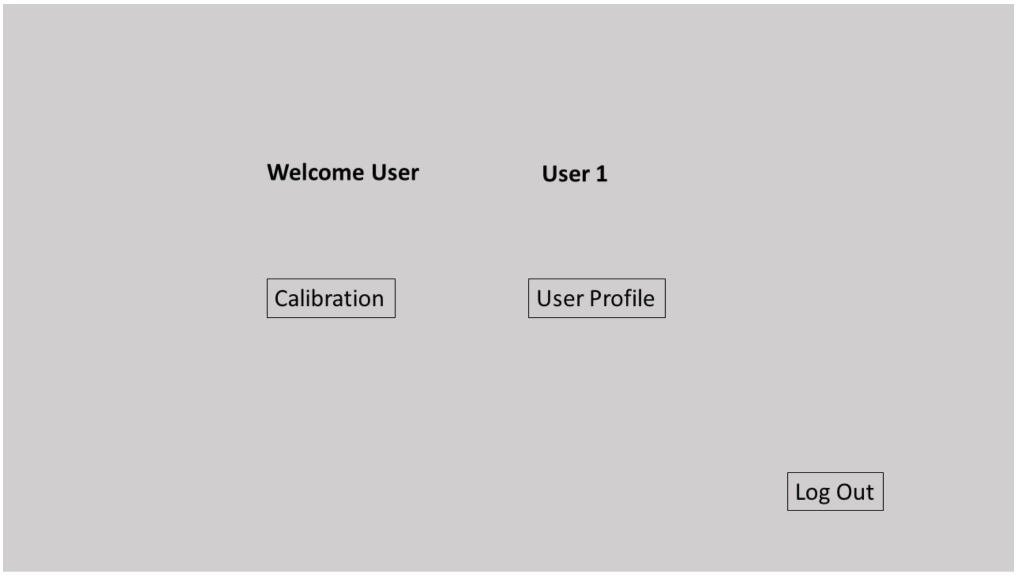


Figure 11: Wireframe for logging in as user (user.py)

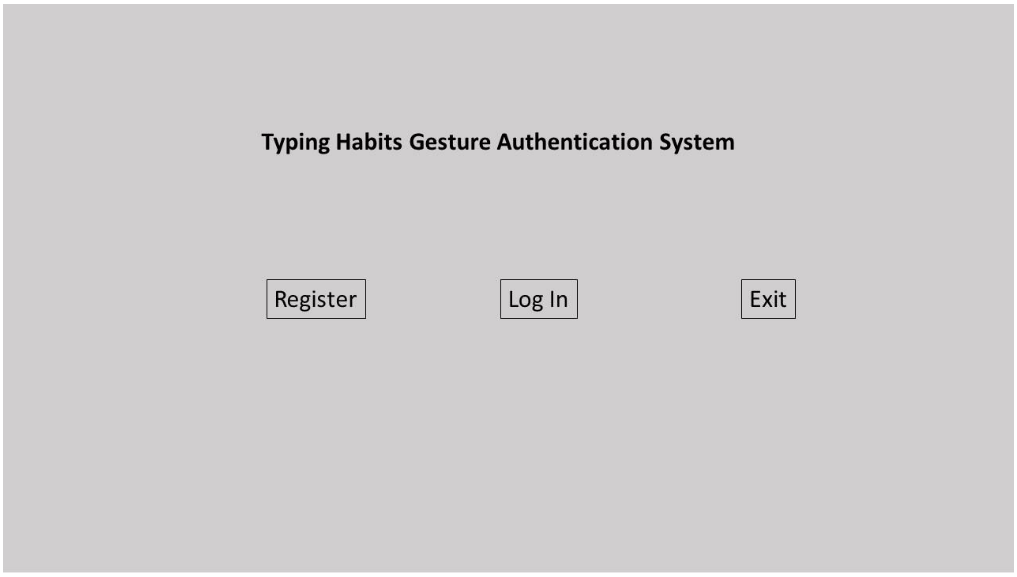


Figure 12: Wireframe for welcome.py (Welcome page)

Test Case

Table 5: Test case for user story #2

User Story :	2			
User Story Description :	As a user, they should be able to log out so that they can exit the application safely.			
Actor :	User			
Trigger :	User clicks on the Logout button on their user page.			
Flow	**Step**	**Description**	**Pass/Fail**	**Remarks**
A	1	User clicks on the Logout button on their user page	Pass	
	2	The system is unable to retrieve the welcome page. The system will display an error message.	Pass	"The system unable to retrieve the welcome page" message box will pop out

Table 6: Test case for user story #3 and 4

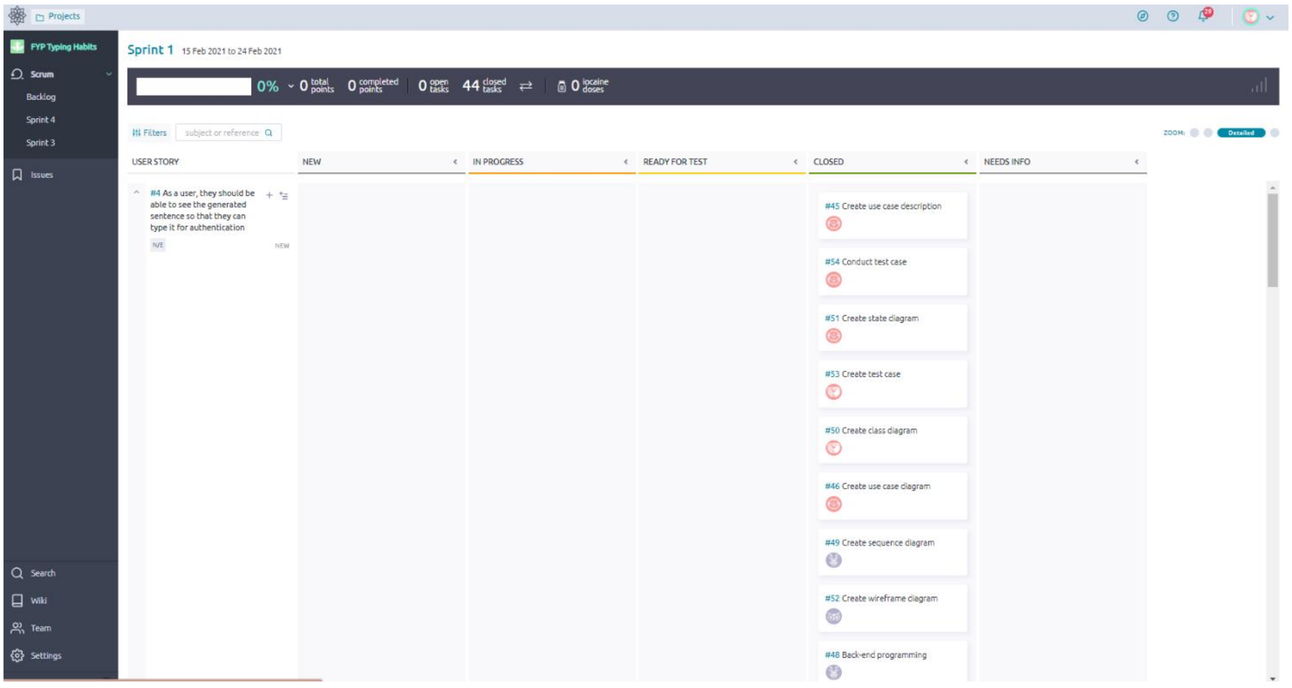
User Story :	3, 4			
User Story Description :	As a user, they should be able to see the generated sentence so that they can type it for authentication.			
Actor :	User			
Trigger :	User clicks on the submit button on their Login page after entering username			
Flow	**Step**	**Description**	**Pass/Fail**	**Remarks**
A	1	User clicks on the Back button on the TypingHabitUser page.	Pass	
	2	The system is unable to retrieve the Login page. The system will display an error message.	Pass	"The system unable to retrieve the Login page" message box will pop out
B	1	User clicks on the Refresh button on the TypingHabitUser page.	Pass	
	2	The system is unable to display the sentence. The system will	Pass	"The system unable to retrieve sentence" message box will pop out

		display an error message.		

Table 7: Test case for user story #11

User Story :	11			
User Story Description :	As an admin, they should be able to log in as admin so that they can be distinguished as a different role.			
Actor :	Admin			
Trigger :	Admin clicks on the submit button on the login page.			
Flow	**Step**	**Description**	**Pass/Fail**	**Remarks**
A	1	Admin clicks on the Back button on the TypingHabitAdmin page.	Pass	
	2	The system is unable to retrieve the Login page. The system will display an error message.	Pass	"The system unable to retrieve the Login page" message box will pop out
B	1	Admin clicks on the Refresh button on the TypingHabitAdmin page.	Pass	
	2	The system is unable to display the sentence. The system will display an error message.	Pass	"The system unable to retrieve sentence" message box will pop out

Taiga Screenshots

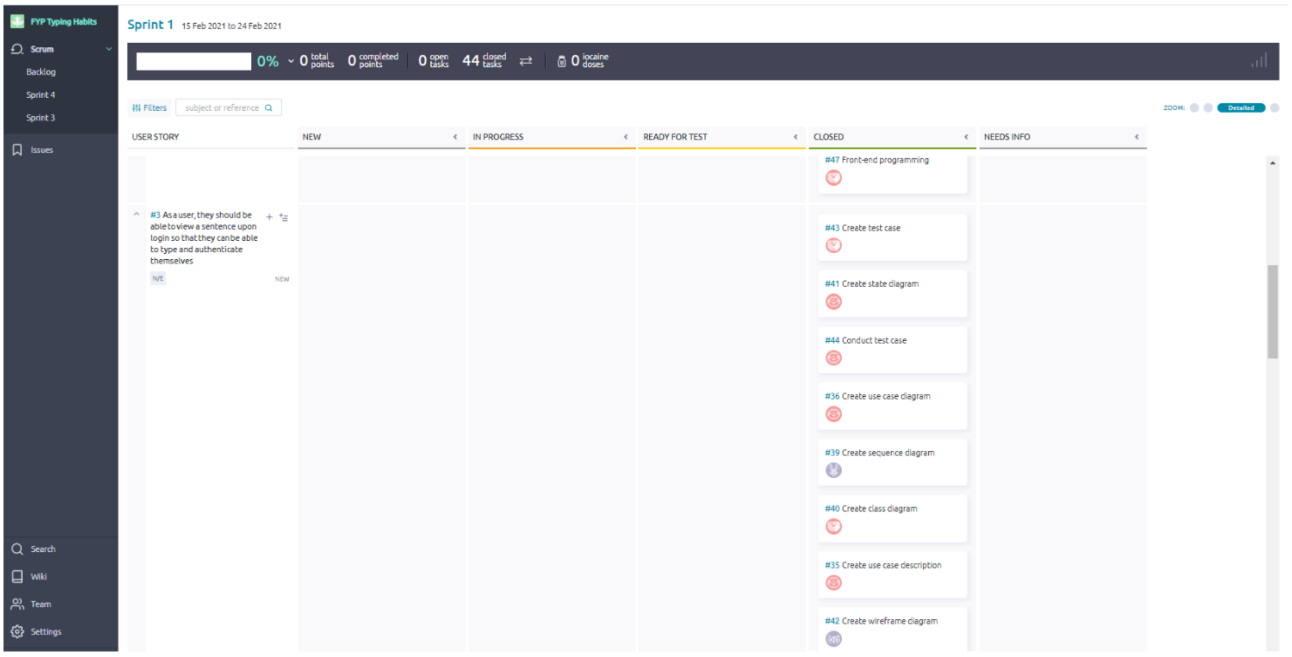


Sprint 1 15 Feb 2021 to 24 Feb 2021

0% 0 total points 0 completed points 0 open tasks 44 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
#4 As a user, they should be able to see the generated sentence so that they can type it for authentication N/E NEW				#45 Create use case description #54 Conduct test case #51 Create state diagram #53 Create test case #50 Create class diagram #46 Create use case diagram #49 Create sequence diagram #52 Create wireframe diagram #48 Back-end programming	

Figure 13a: Taiga screenshot for sprint 1

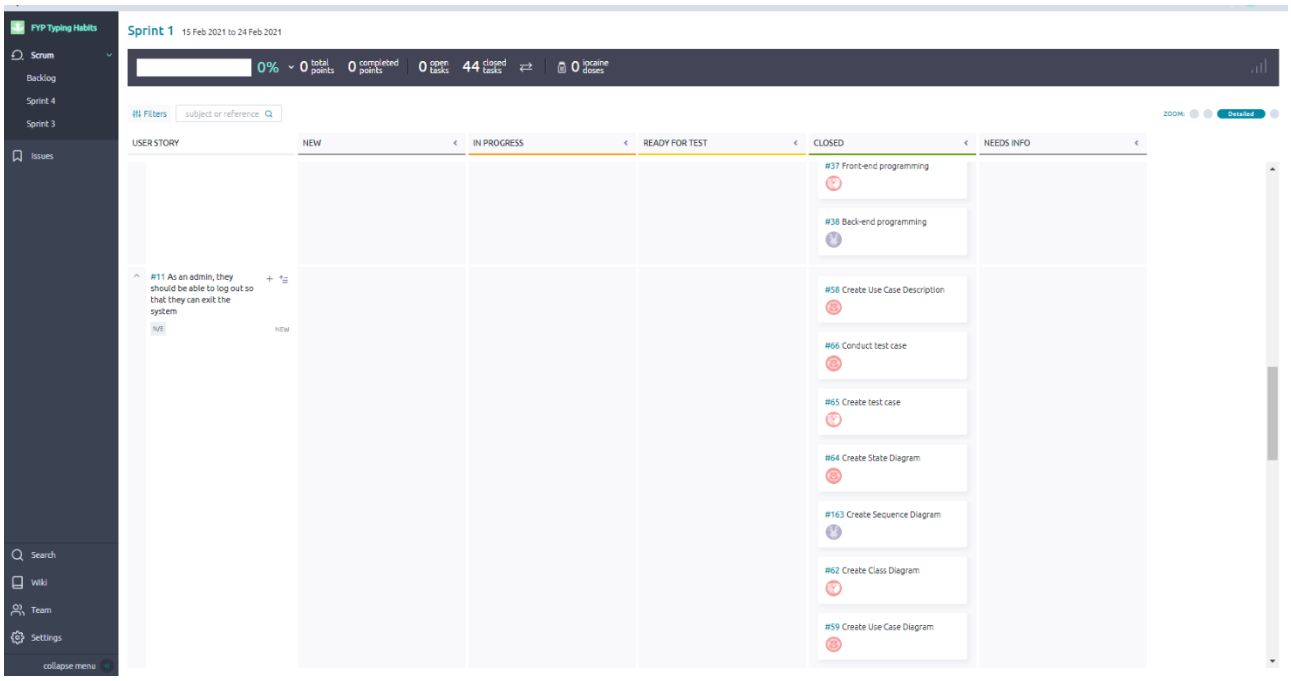


Sprint 1 15 Feb 2021 to 24 Feb 2021

0% 0 total points 0 completed points 0 open tasks 44 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
				#47 Front-end programming	
#3 As a user, they should be able to view a sentence upon login so that they can be able to type and authenticate themselves N/E NEW				#43 Create test case #41 Create state diagram #44 Conduct test case #36 Create use case diagram #39 Create sequence diagram #40 Create class diagram #35 Create use case description #42 Create wireframe diagram	

Figure 13b: Taiga screenshot for sprint 1

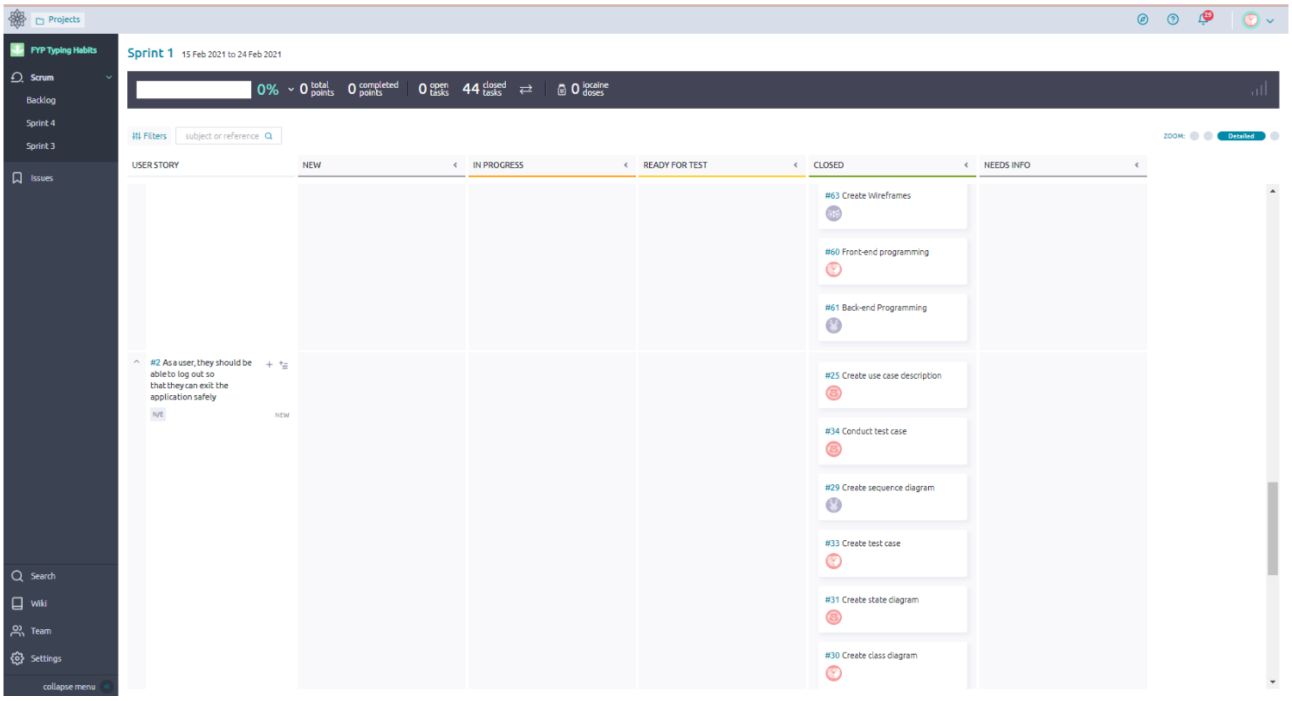


Sprint 1 15 Feb 2021 to 24 Feb 2021

0% 0 total points 0 completed points 0 open tasks 44 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
				#37 Front-end programming	
				#38 Back-end programming	
#11 As an admin, they should be able to log out so that they can exit the system N/E NEW				#58 Create Use Case Description	
				#66 Conduct test case	
				#65 Create test case	
				#64 Create State Diagram	
				#163 Create Sequence Diagram	
				#62 Create Class Diagram	
				#59 Create Use Case Diagram	

Figure 13c: Taiga screenshot for sprint 1

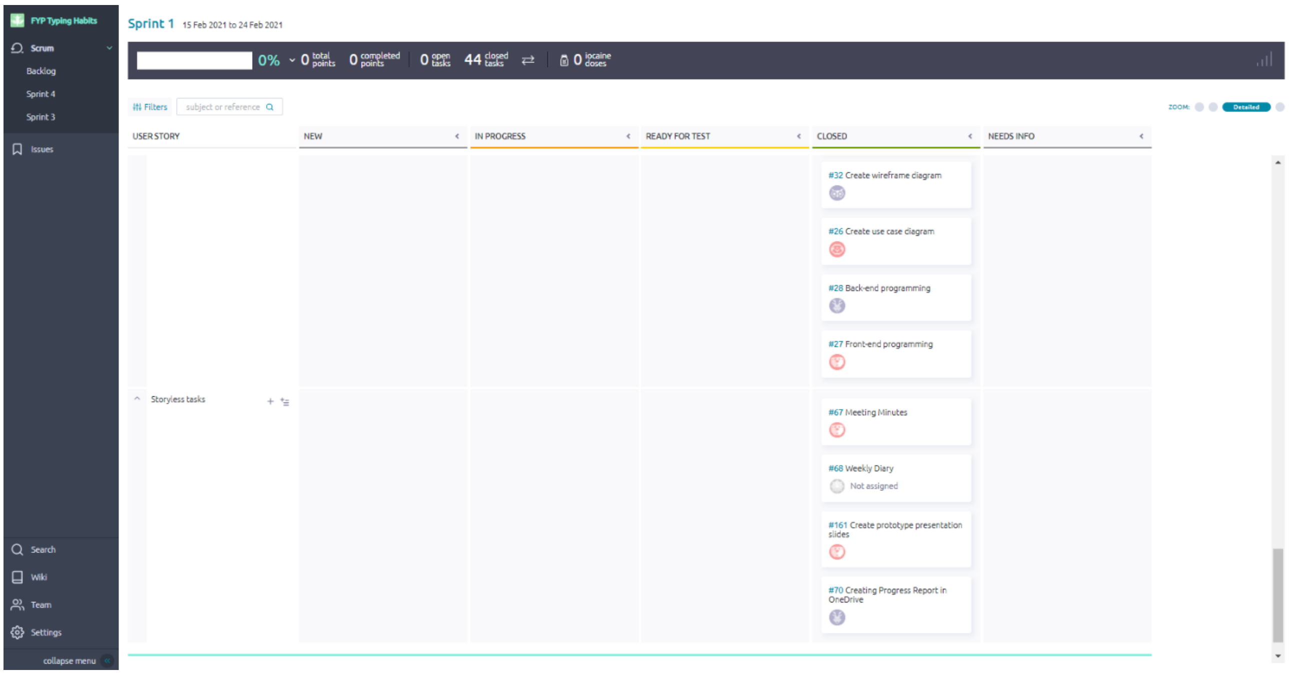


Sprint 1 15 Feb 2021 to 24 Feb 2021

0% 0 total points 0 completed points 0 open tasks 44 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
				#63 Create Wireframes	
				#60 Front-end programming	
				#61 Back-end Programming	
#2 As a user, they should be able to log out so that they can exit the application safely N/E NEW				#25 Create use case description	
				#34 Conduct test case	
				#29 Create sequence diagram	
				#33 Create test case	
				#31 Create state diagram	
				#30 Create class diagram	

Figure 13d: Taiga screenshot for sprint 1



Sprint 1 15 Feb 2021 to 24 Feb 2021

0% 0 total points 0 completed points 0 open tasks 44 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
				#32 Create wireframe diagram #26 Create use case diagram #28 Back-end programming #27 Front-end programming	
Storyless tasks				#67 Meeting Minutes #68 Weekly Diary Not assigned #161 Create prototype presentation slides #70 Creating Progress Report in OneDrive	

Figure 13e: Taiga screenshot for sprint 1

4 Current Work

List of Current Work Items/Activities

Table 8: Table for Current Work Items/Activities
Updated as of 24/3/2021:

S/No	Work Item	% completed	Expected Completion Date
1	Update of marketing website	100%	23/3/2021
2	Sprint 2	100%	24/3/2021
3	Prototype slides	100%	24/3/2021
4	Progress report	100%	24/3/2021

Sprint 2

User Stories

#6 As a user, they should be able to register account so that they can log in to the system
#7 As an admin, they should be able to see all the user profiles so that they can manage the user profiles
#8 As an admin, they should be able to delete user profiles so that they can remove profiles of inactive users
#9 As an admin, they should be able to access the individual user profiles so that they can see user profile details
#10 As an admin, they should be able to log in as admin so that they can be distinguished as a different role

Use Case Diagrams

The use case diagrams below are the same as seen in sprint 1 earlier (in Figures 6 and 7).

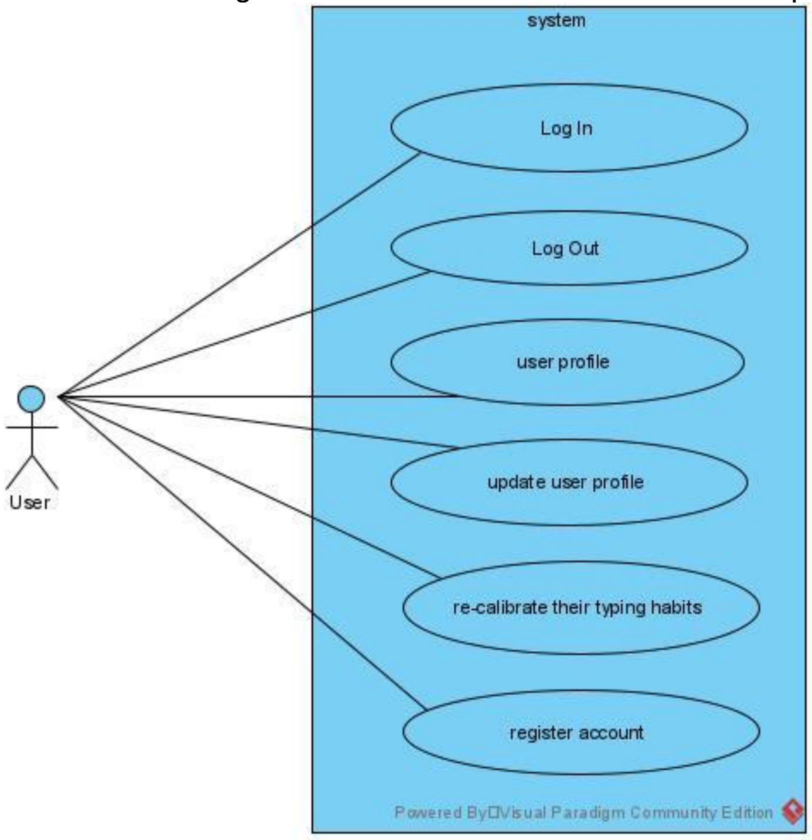

system
Log In
Log Out
user profile
update user profile
re-calibrate their typing habits
register account
User
Powered By Visual Paradigm Community Edition

Figure 14: Use case diagram for user role

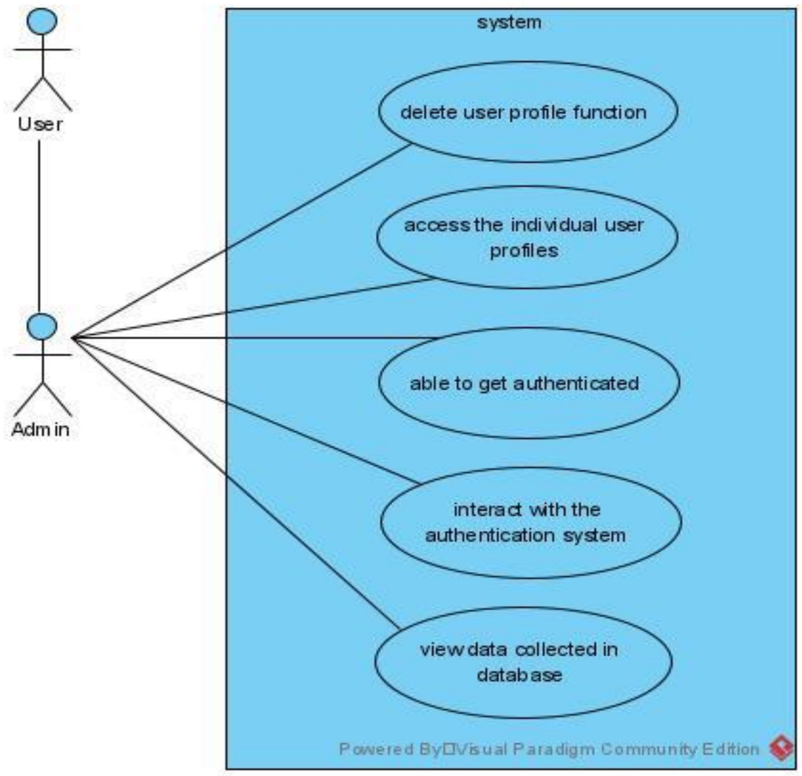

User
system
delete user profile function
access the individual user profiles
able to get authenticated
interact with the authentication system
view data collected in database
Admin
Powered By Visual Paradigm Community Edition

Figure 15: Use case diagram for admin role

Use Case Descriptions

User: User Story #6

Table 9: User Story #6

User Story #6: As a user, they should be able to register account so that they can log in to the system	
Name: Register (Register.py and Register_2.py)	**Taiga ID:** 6
Stakeholders and Goals: User wants to register an account.	
Descriptions: The user wants to create an account.	
Actors: User	
Trigger: 1. User clicks on the Register option on the Welcome page (Welcome.py).	
Normal Flow: 1. The system retrieves and displays the register page. 2. User enters first name, last name, username, email, security question, security question answer. 3. User clicks on the ‘Next button. 4. User registers typing habit 20 times 5. The system creates a new row in the database and displays a success message. 6. User clicks ‘Confirm’ button. 7. The system retrieves the user Welcome page (Welcome.py). 8. End.	
Sub flows: 2a User selects Back option on Registration page 1 (Registration.py) The system redirect the user back to the welcome page. 2b.1 User exits Registration on page 1 (Registration.py) The application ends. 2b.2 User exits Registration on page 2 (Registration_2py) The application ends.	
Alternate/Exception flows: 1a The system is unable to retrieve the Register page. The system will display an error message. The use case terminates. 2a No first name entered in the First Name text field. The system will prompt User to re-input first name at Step 2. 2b Invalid value in the First Name text field. The system will display an error message and prompt User to re-input first name at Step 2 2c No last name entered in the Last Name text field. The system will prompt User to re-input last name at Step 2. 2d Invalid value in the Last Name text field. The system will display an error message and prompt User to re-input last name at Step 2	

2e No username entered in the Username text field.
The system will prompt User to re-input username at Step 2.

2f Invalid value (username exists in database) in the Username text field.
The system will display an error message and prompt User to re-input username at Step 2

2g No email entered in the Email text field.
The system will prompt User to re-input email at Step 2.

2h Invalid value (email exist in database or value is not an email) in the Email text field.
The system will display an error message and prompt User to re-input email at Step 2.

5a The system is unable to retrieve the login page.
The system will display an error message. The use case terminates.

Admin: User Story #8

Table 10: User Story #8

User Story #8: As an admin, they should be able to see all the user profiles so that they can manage the user profiles.	
Name: ManageUser (adminPgManageUsers.py)	**Taiga ID:** 7
Stakeholders and Goals: Admin wants to view and manage all user profiles within the system.	
Descriptions: The user is able to access all user profiles within the system.	
Actors: Admin	
Trigger: 1. Admin clicks on the ManageUser button in admin page (admin.py).	
Normal Flow: 1. Admin clicks on the ManageUser button or icon in admin page(admin.py). 2. The system retrieves and displays the adminPgManageUsers page (adminPgManageUsers.py). 3. End.	
Sub flows: None	
Alternate/Exception flows: 2a The system is unable to retrieve the adminPgManageUsers page. The system will display an error message. The use case terminates.	

Admin: User Story #9

Table 11: User Story #9

User Story #9: As an admin, they should be able to delete user profiles so that they can remove profiles of inactive users.	
Name: DeleteUser (function is found in adminPageManageUsers.py)	**Taiga ID:** 8
Stakeholders and Goals: Admin wants to delete and remove user profiles of inactive users within the system.	
Descriptions: The admin is able to delete user profile within the system.	
Actors: Admin	
Trigger: 1. Admin selects one of the user profiles. 2. Admin clicks on the 'Delete' button on adminPgManageUsers page.	
Normal Flow: 1. Admin selects one of the user profiles. 2. Admin clicks on the 'Delete' button on adminPgManageUsers page. 3. The system retrieves and deletes the row containing the inactive user from the database and displays a success message. 4. The system displays a confirmation message. 5. Admin clicks 'Okay' button. 6. End	
Sub flows: None	
Alternate/Exception flows: 4a The system is unable to retrieve and delete the inactive user from the database. The system will display an error message. The use case terminates.	

Admin: User Story #10

Table 12: User Story #10

User Story #10: As an admin, they should be able to access the individual user profiles so that they can see user profile details.	
Name: getUser(function found in adminPageManageUsers.py)	**Taiga ID:** 9
Stakeholders and Goals: Admin wants to view individual user profile details within the system.	
Descriptions: The Admin is able to access and edit individual user profile details within the system.	
Actors: Admin	
Trigger: 1. Admin selects individual users on the adminPgManageUser page (adminPgManageUsers.py). 2. Admin selects Edit option.	
Normal Flow: 1. Admin selects individual users on the adminPgManageUser page (adminPgManageUsers.py). 2. Admin selects Edit option. 3. System retrieves user details and display it on edituser page. 4. End.	
Sub flows: None	
Alternate/Exception flows: 2a The system is unable to retrieve individual user details. The system will display an error message. The use case terminates. 3a The system is unable to display the edituser page. The system will display an error message. The use case terminates.	

Sequence Diagram

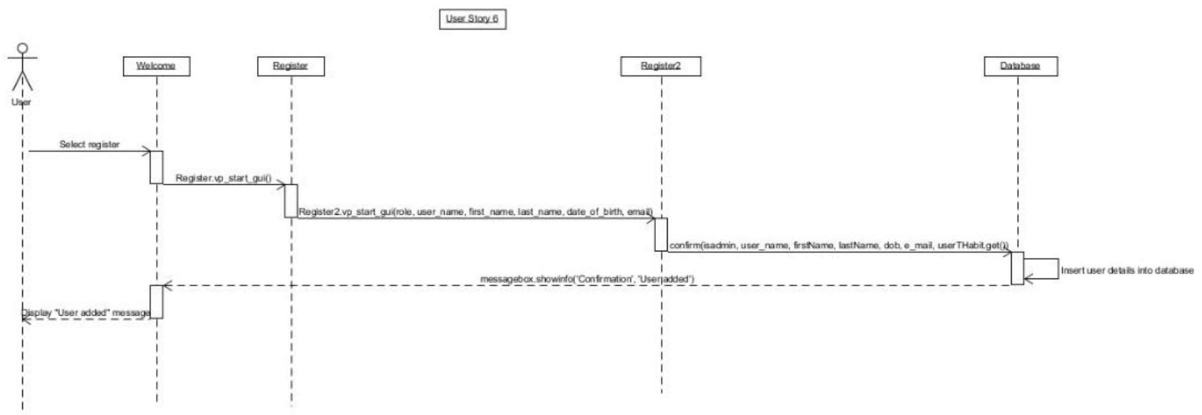

User Story 6
User
Welcome
Register
Register2
Database
Select register
Register.vp_start_gui()
Register2.vp_start_gui(role, user_name, first_name, last_name, date_of_birth, email)
confirm(isadmin, user_name, firstName, lastName, dob, e_mail, userTHabit.get())
Insert user details into database
messagebox.showinfo('Confirmation', 'User added')
Display "User added" message

Figure 16: Sequence diagram for user story 6 (Able to register as a user)

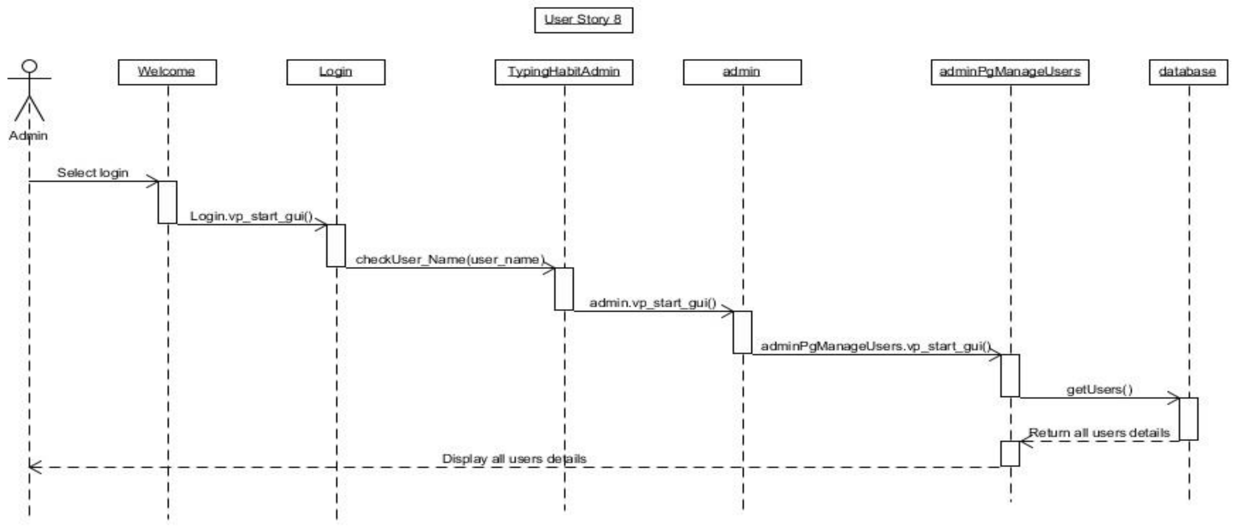


Figure 17: Sequence diagram for user story 8 (Taiga ID 7)

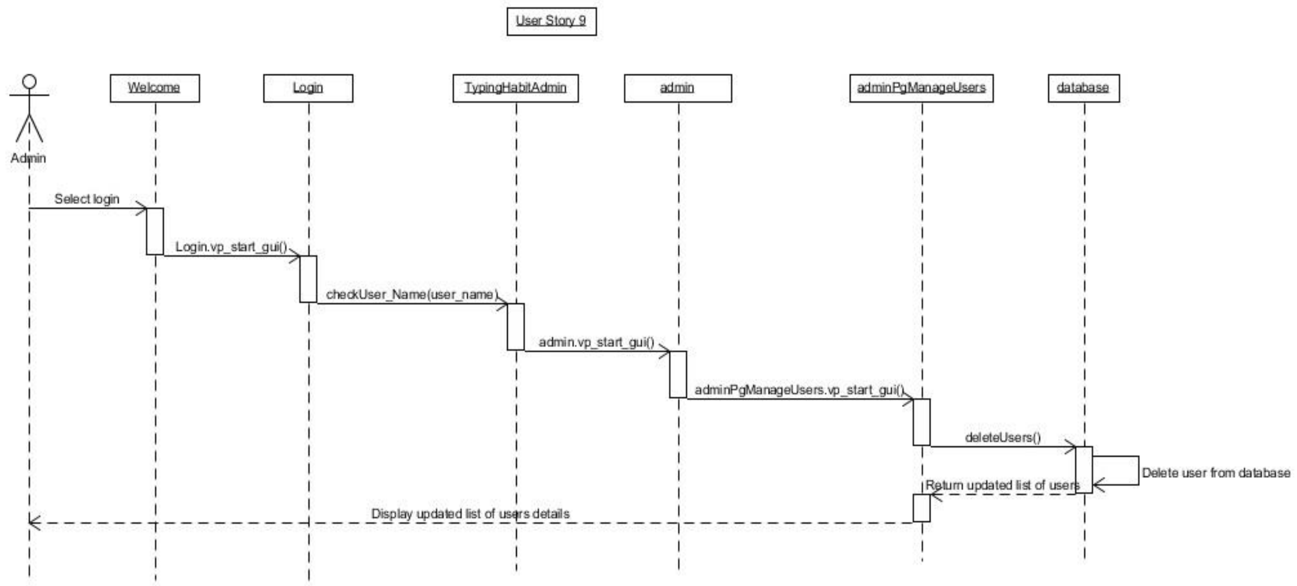

User Story 9
Admin
Welcome
Login
TypingHabitAdmin
admin
adminPgManageUsers
database
Select login
Login.vp_start_gui()
checkUser_Name(user_name)
admin.vp_start_gui()
adminPgManageUsers.vp_start_gui()
deleteUsers()
Delete user from database
Return updated list of users
Display updated list of users details

Figure 18: Sequence diagram for user story 9 (Admin can delete user profiles)

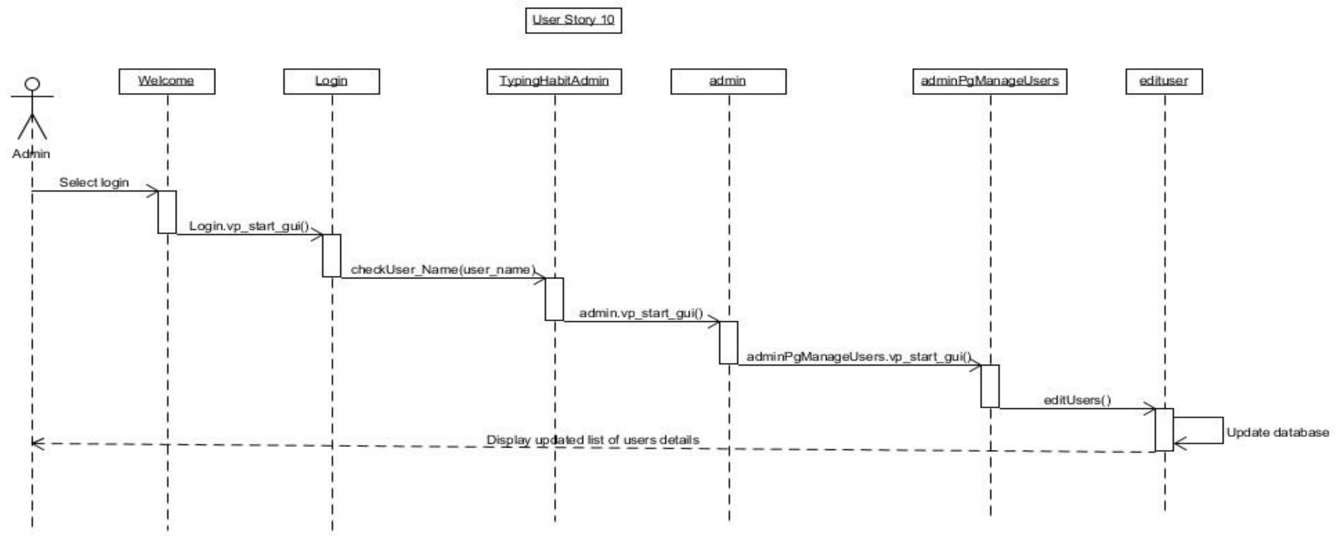


Figure 19: Sequence diagram for user story 10 (Admin can edit individual user profile)

Class Diagram (Overall)

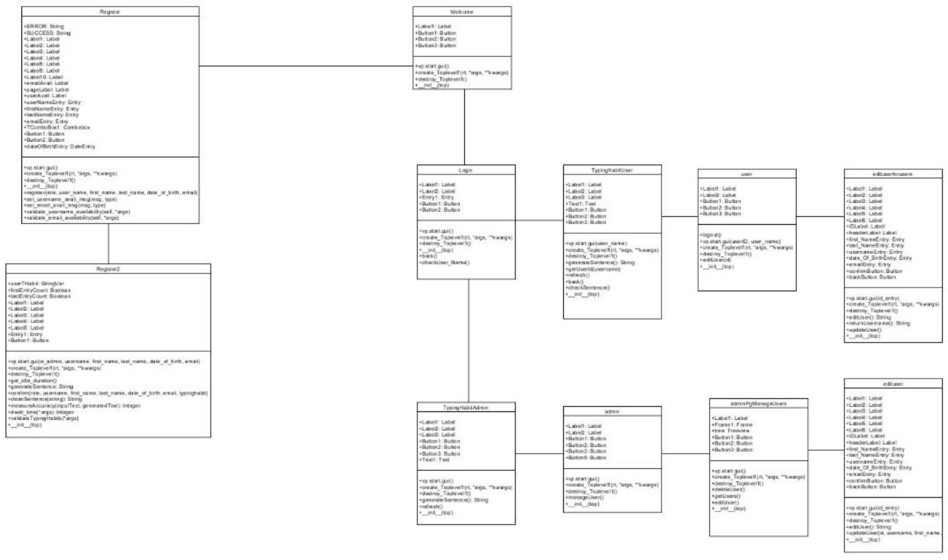


Figure 20: Overall class diagram

State Diagram (Overall)

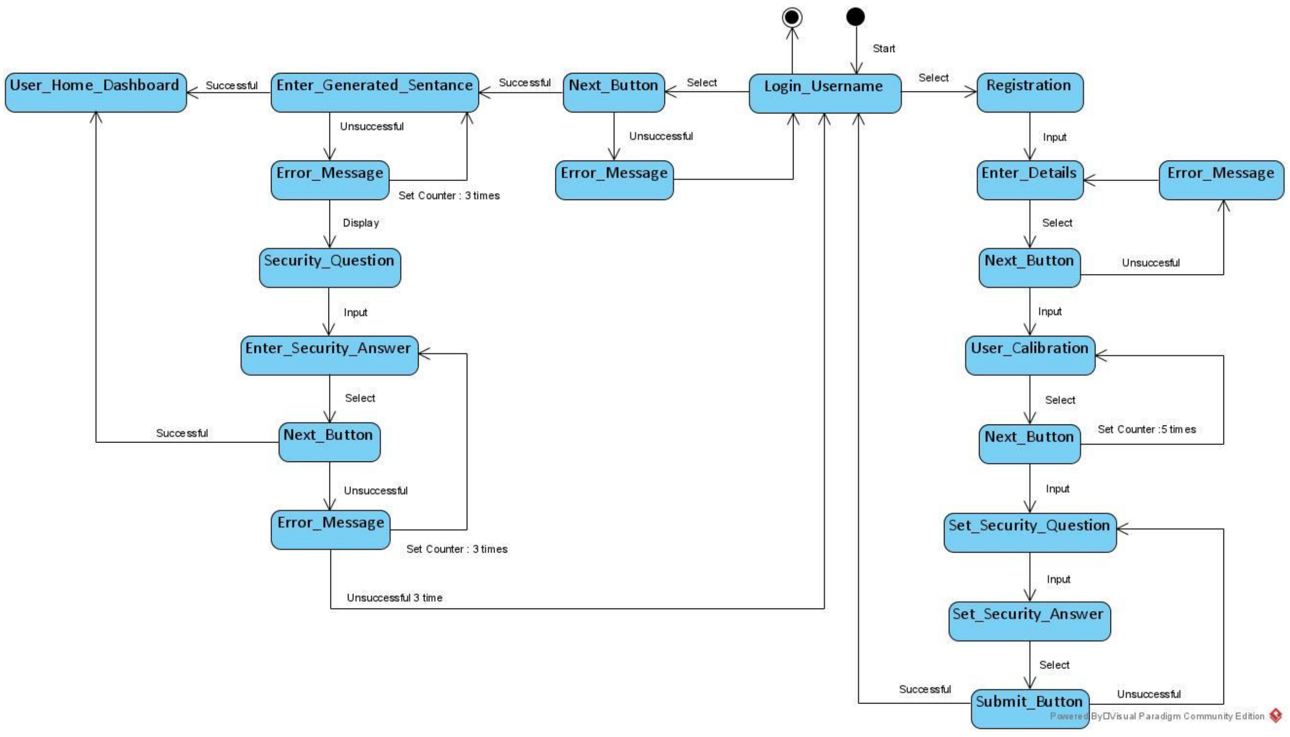
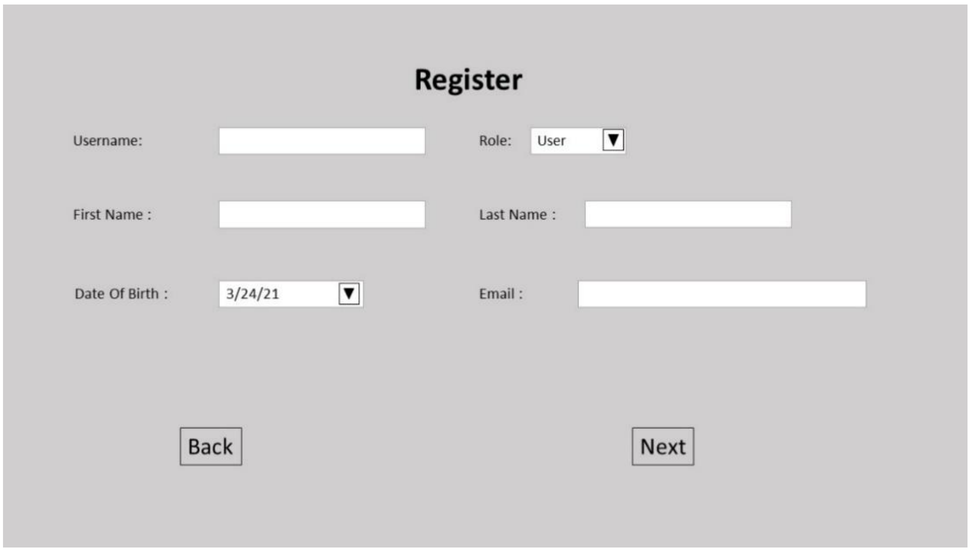


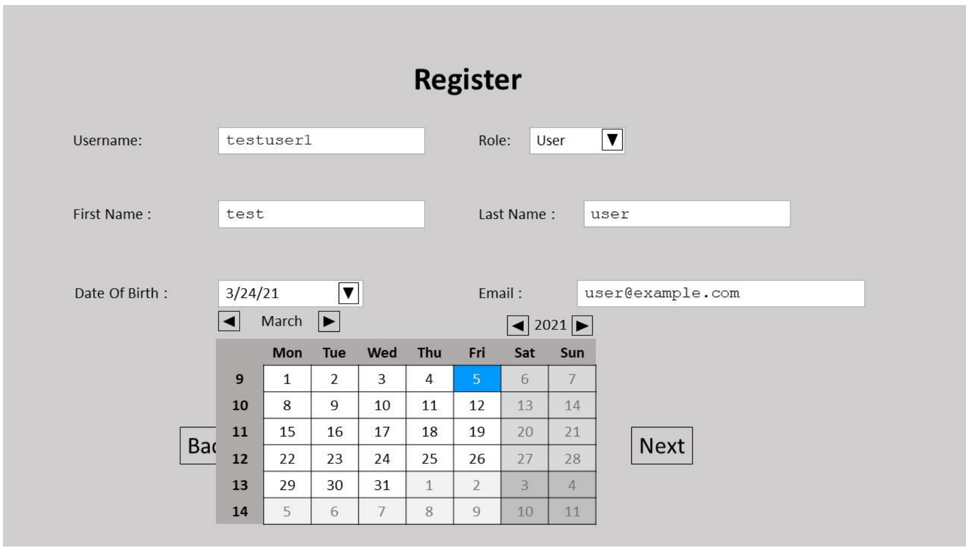
Figure 21: Overall state diagram

Wireframes



Username:		Role:	User
First Name :		Last Name :	
Date Of Birth :	3/24/21	Email :	

Figure 22a: Wireframe for registration

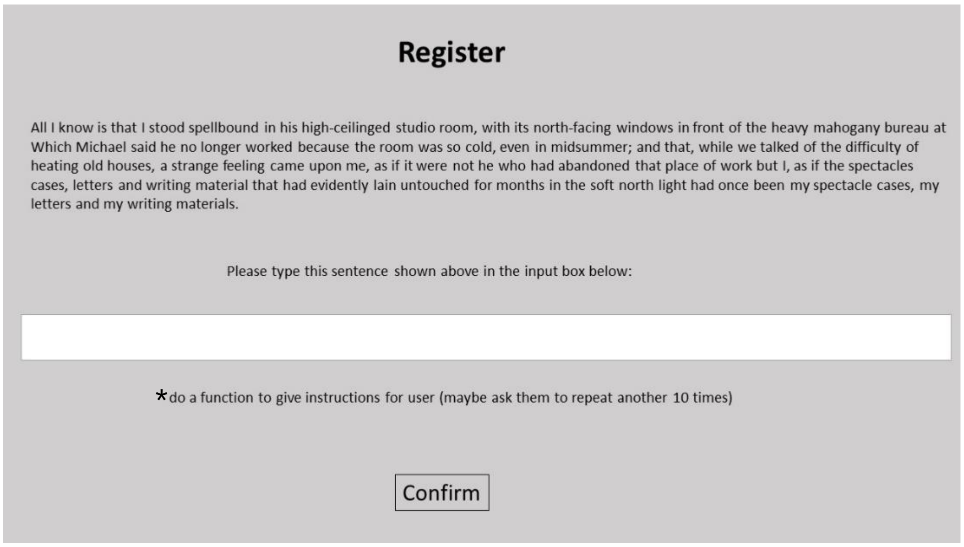


Username:	testuser1	Role:	User
First Name :	test	Last Name :	user
Date Of Birth :	3/24/21	Email :	user@example.com

March 2021

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
9	1	2	3	4	5	6	7
10	8	9	10	11	12	13	14
11	15	16	17	18	19	20	21
12	22	23	24	25	26	27	28
13	29	30	31	1	2	3	4
14	5	6	7	8	9	10	11

Figure 22b: Wireframe for registration (able to open calendar)



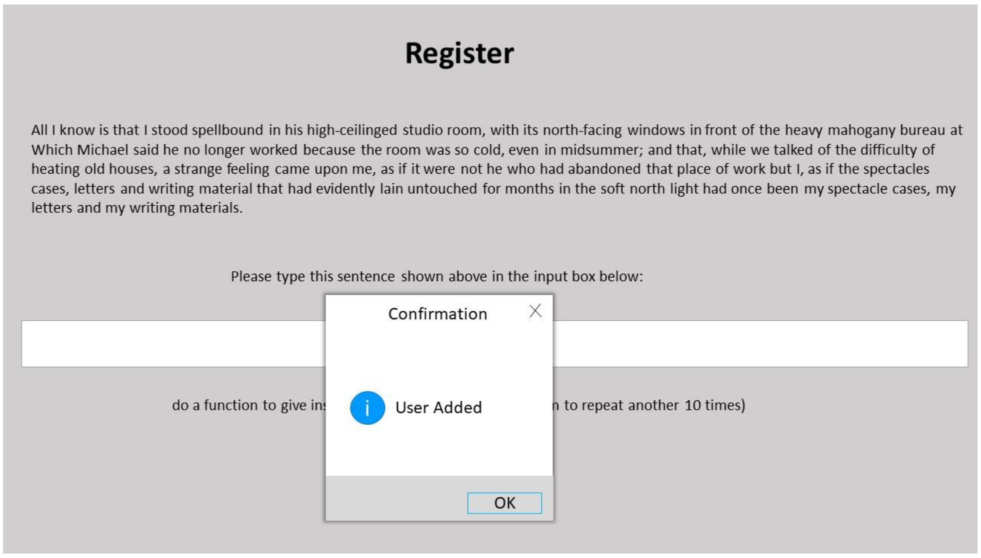
All I know is that I stood spellbound in his high-ceilinged studio room, with its north-facing windows in front of the heavy mahogany bureau at Which Michael said he no longer worked because the room was so cold, even in midsummer; and that, while we talked of the difficulty of heating old houses, a strange feeling came upon me, as if it were not he who had abandoned that place of work but I, as if the spectacles cases, letters and writing material that had evidently lain untouched for months in the soft north light had once been my spectacle cases, my letters and my writing materials.

Please type this sentence shown above in the input box below:

*do a function to give instructions for user (maybe ask them to repeat another 10 times)

**Sentence will change according to whether typing habit is successfully registered or not.*

Figure 22c: Wireframe for registration (register typing habit)


Register

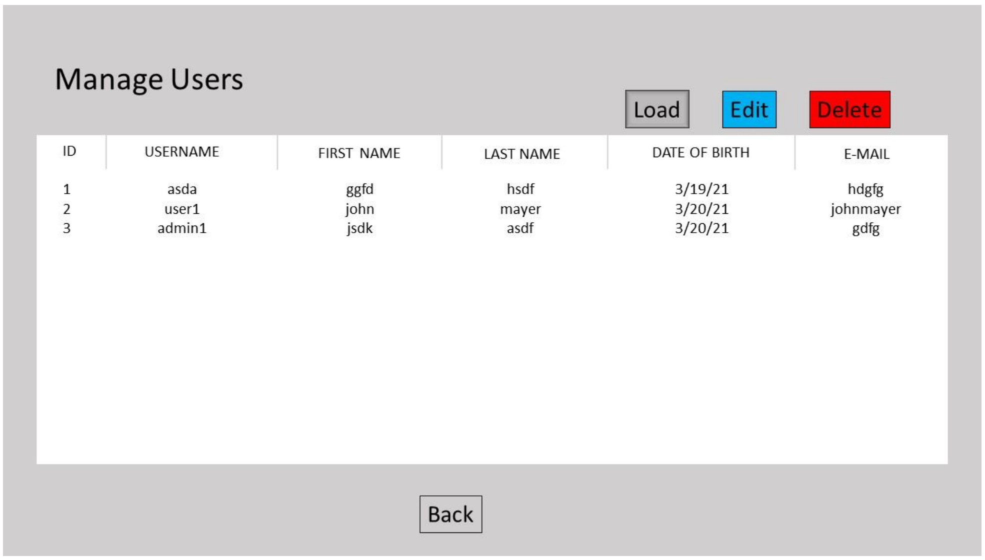
All I know is that I stood spellbound in his high-ceilinged studio room, with its north-facing windows in front of the heavy mahogany bureau at Which Michael said he no longer worked because the room was so cold, even in midsummer; and that, while we talked of the difficulty of heating old houses, a strange feeling came upon me, as if it were not he who had abandoned that place of work but I, as if the spectacles cases, letters and writing material that had evidently lain untouched for months in the soft north light had once been my spectacle cases, my letters and my writing materials.

Please type this sentence shown above in the input box below:

do a function to give in

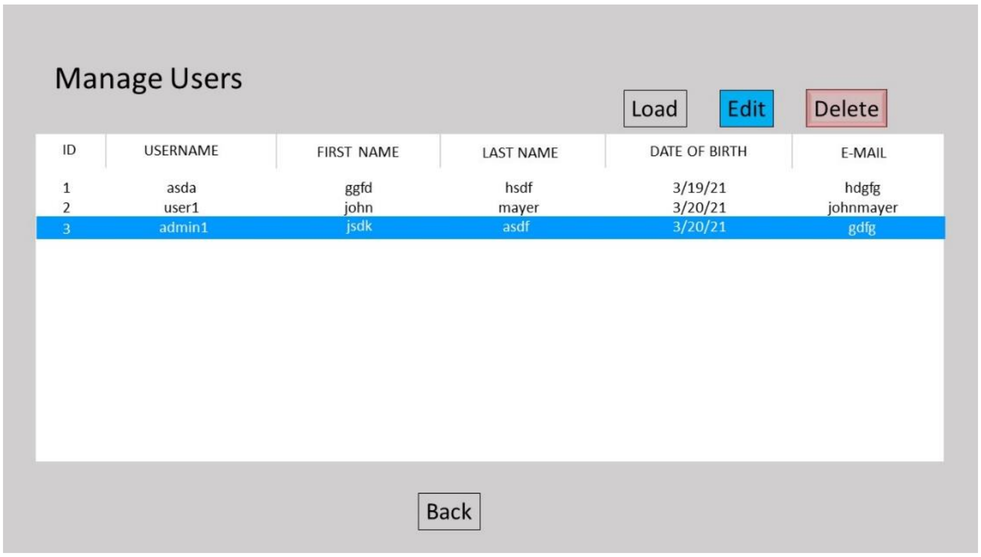
n to repeat another 10 times)

Figure 22d: Wireframe for successful registration


Manage Users

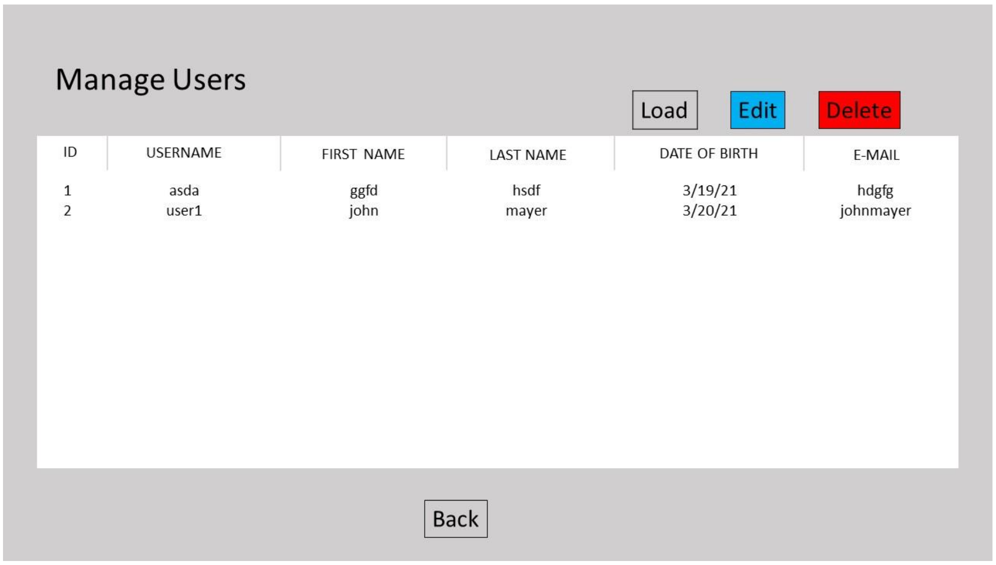
ID	USERNAME	FIRST NAME	LAST NAME	DATE OF BIRTH	E-MAIL
1	asda	ggfd	hsdf	3/19/21	hdgfg
2	user1	john	mayer	3/20/21	johnmayer
3	admin1	jsdk	asdf	3/20/21	gdfg

Figure 23a: Wireframe for managing user profiles


Manage Users

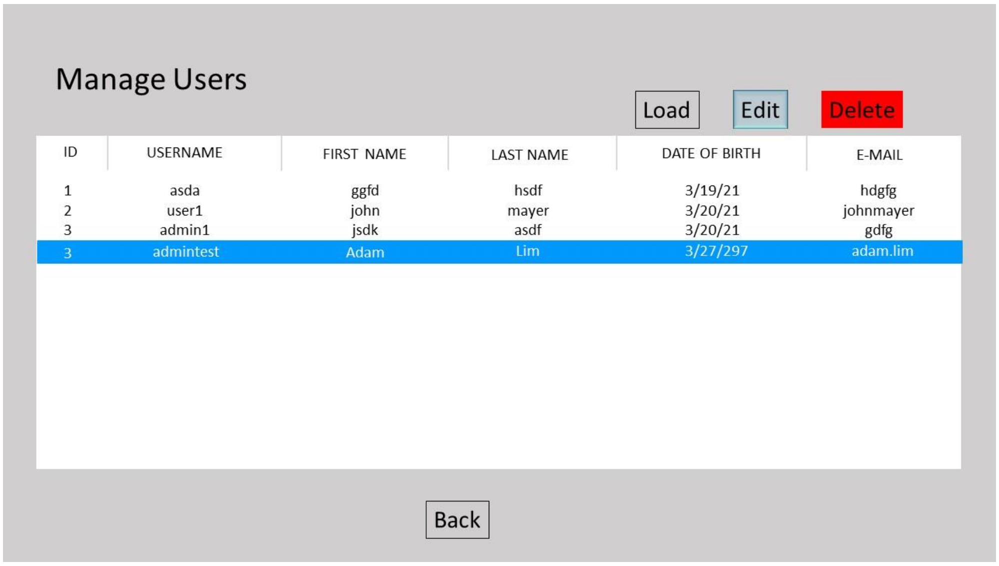
ID	USERNAME	FIRST NAME	LAST NAME	DATE OF BIRTH	E-MAIL
1	asda	ggfd	hsdf	3/19/21	hdgfg
2	user1	john	mayer	3/20/21	johnmayer
3	admin1	jsdk	asdf	3/20/21	gdfg

Figure 23b: Wireframe for managing user profiles (deleting a user)



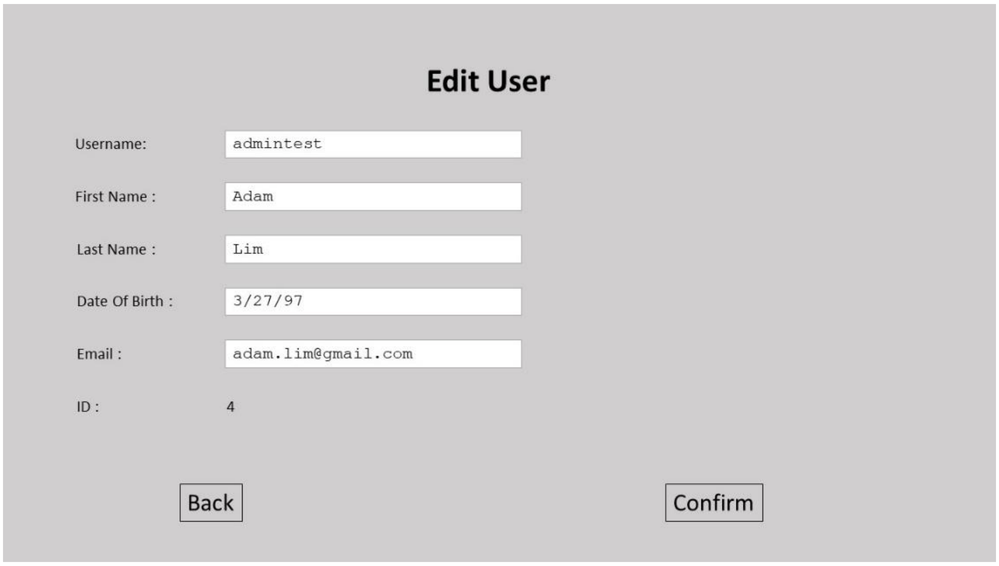
ID	USERNAME	FIRST NAME	LAST NAME	DATE OF BIRTH	E-MAIL
1	asda	ggfd	hsdf	3/19/21	hdgfg
2	user1	john	mayer	3/20/21	johnmayer

Figure 23c: Wireframe for managing user profiles (after delete)



ID	USERNAME	FIRST NAME	LAST NAME	DATE OF BIRTH	E-MAIL
1	asda	ggfd	hsdf	3/19/21	hdgfg
2	user1	john	mayer	3/20/21	johnmayer
3	admin1	jsdk	asdf	3/20/21	gdfg
3	admintest	Adam	Lim	3/27/297	adam.lim

Figure 23e: Wireframe for managing user profiles (editing a user)



Username:	admintest
First Name :	Adam
Last Name :	Lim
Date Of Birth :	3/27/97
Email :	adam.lim@gmail.com
ID :	4

Figure 23f: Wireframe for managing user profiles (editing as user - inside a user profile)

Test Case

Table 13: Test Case for user story #6

User Story :	6			
User Story Description :	As a user, they should be able to register account so that they can log in to the system.			
Actor :	User			
Trigger :	1. User clicks on the Register option on the Welcome page. 2. User has entered an invalid username on the login page and the system prompts user to register.			
Flow	**Step**	**Description**	**Pass/Fail**	**Remarks**
1a	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system is unable to retrieve the Register page. The system will display an error message. The use case terminates.	Pass	"Unable to display Register page" messagebox will pop up
2a	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system retrieves and displays the register page.	Pass	
	3	User does not input any first name. The system will display an error message. The use case terminates.	Pass	
2b	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system retrieves and displays the register page.	Pass	

	3	User input invalid values for first name.	Pass	
	4	User clicks on the ‘Next’ button.	Pass	
	5	The system will display an error message and prompt User to re-input the text-field at Step 2.	Pass	
2c	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system retrieves and displays the register page.	Pass	
	3	User does not input values for last name.	Pass	
	4	User clicks on the ‘Next’ button.	Pass	
	5	The system will display an error message and prompt User to re-input the text-field at Step 2.	Pass	
2d	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system retrieves and displays the register page.	Pass	
	3	User input invalid values for last name.	Pass	

	4	User clicks on the ‘Next’ button.	Pass	
	5	The system will display an error message and prompt User to re-input the text-field at Step 2.	Pass	
2e	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system retrieves and displays the register page.	Pass	
	3	User does not input values for username.	Pass	
	4	User clicks on the ‘Next’ button.	Pass	
	5	The system will display an error message and prompt User to re-input the text-field at Step 2.	Pass	
2f	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system retrieves and displays the register page.	Pass	
	3	User input invalid values (username already exists) for username.	Pass	
	4	User clicks on the ‘Next’ button.	Pass	

	5	The system will display an error message and prompt User to re-input the text-field at Step 2.	Pass	
2g	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system retrieves and displays the register page.	Pass	
	3	User does not input values for email.	Pass	
	4	User clicks on the ‘Next’ button.	Pass	
	5	The system will display an error message and prompt User to re-input the text-field at Step 2.	Pass	
2h	1	User clicks on the Register option on the Welcome page.	Pass	
	2	The system retrieves and displays the register page.	Pass	
	3	User input invalid values for email.	Pass	
	4	User clicks on the ‘Next’ button.	Pass	
	5	The system will display an error message and prompt User to re-input the	Pass	

		text-field at Step 2.		
5a	1	User clicks on the 'Confirm' button	Pass	
	2	The system is unable to retrieve the Welcome page. The system will display an error message. The use case terminates.	Pass	

Table 14: Test Case for user story #8

User Story :	8			
User Story Description :	As an admin, they should be able to see all the user profiles so that they can manage the user profiles.			
Actor :	Admin			
Trigger :	Admin clicks on the ManageUser button in admin page (admin.py).			
Flow	**Step**	**Description**	**Pass/Fail**	**Remarks**
2a	1	Admin clicks on the ManageUser button in admin page.	Pass	
	2	The system is unable to retrieve the adminPgManageUsers page. The system will display an error message. The use case terminates	Pass	"The system unable to retrieve the Manage User page" message box will pop out.

Table 15: Test Case for user story #9

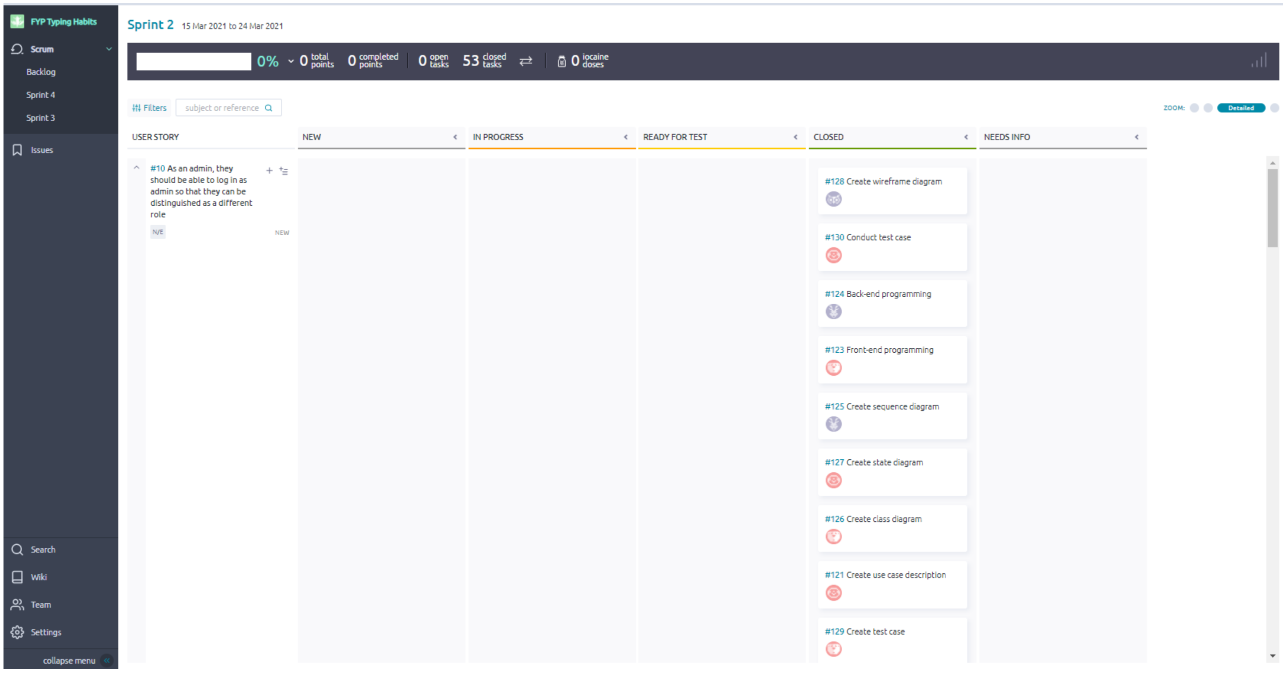
User Story :	9			
User Story Description :	As an admin, they should be able to delete user profiles so that they can remove profiles of inactive users.			
Actor :	Admin			
Trigger :	Admin clicks on the ‘Delete’ button on adminPgManageUsers page.			
Flow	**Step**	**Description**	**Pass/Fail**	**Remarks**
4a	1	Admin clicks on the ‘Delete’ button on adminPgManageUsers page.	Pass	

	2	The system is unable to retrieve and delete the inactive user from the database. The system will display an error message. The use case terminates.	Pass	

Table 16: Test Case for user story #10

User Story :	10			
User Story Description :	As an admin, they should be able to access the individual user profiles so that they can see user profile details.			
Actor :	Admin			
Trigger :	1. Admin selects individual users on the adminPgManageUser page (adminPgManageUsers.py). 2. Admin selects Edit option.			
Flow	**Step**	**Description**	**Pass/Fail**	**Remarks**
2a	1	Admin selects individual users on the adminPgManageUser page.	Pass	
	2	Admin selects Edit option.	Pass	
	3	The system is unable to retrieve the user details. The system will display an error message.	Pass	
3a	1	Admin selects Edit option.	Pass	
	2	System retrieves user details	Pass	
	3	The system is unable to display the edituser page. The system will display an error message.	Pass	

Taiga Screenshots

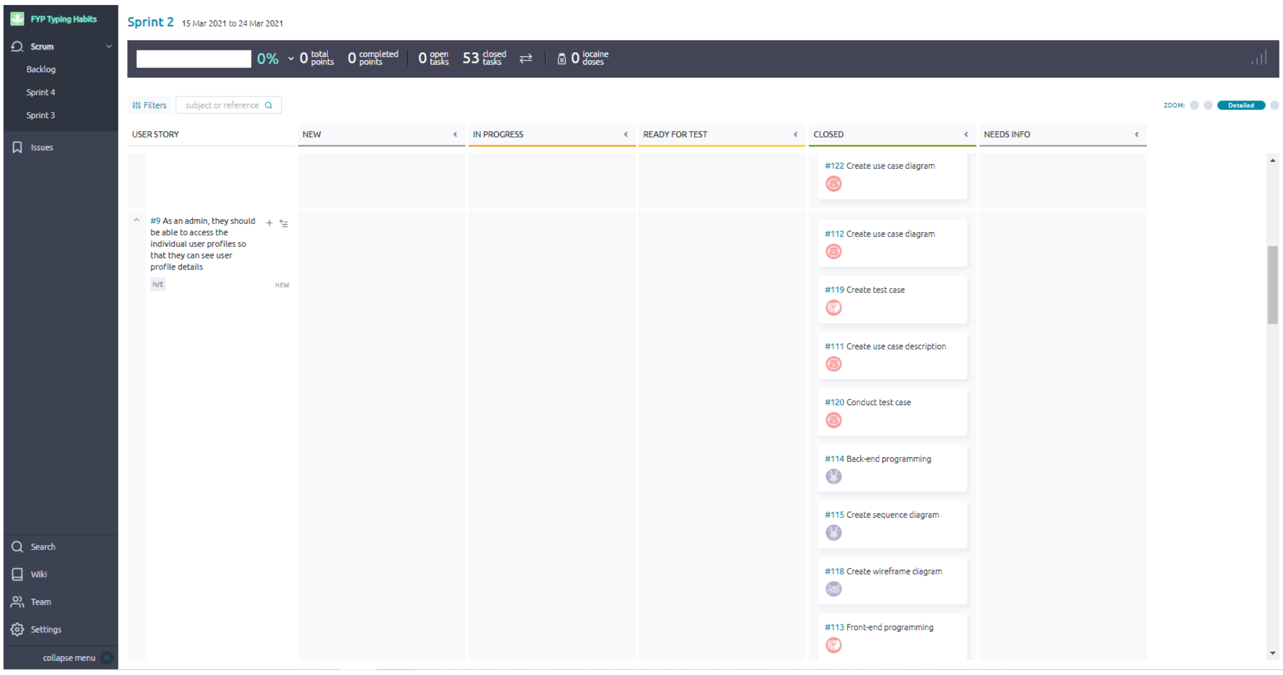


Sprint 2 15 Mar 2021 to 24 Mar 2021

0% 0 total points 0 completed points 0 open tasks 53 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
#10 As an admin, they should be able to log in as admin so that they can be distinguished as a different role N/E NEW				#128 Create wireframe diagram #130 Conduct test case #124 Back-end programming #123 Front-end programming #125 Create sequence diagram #127 Create state diagram #126 Create class diagram #121 Create use case description #129 Create test case	

Figure 24a: Taiga screenshots for sprint 2

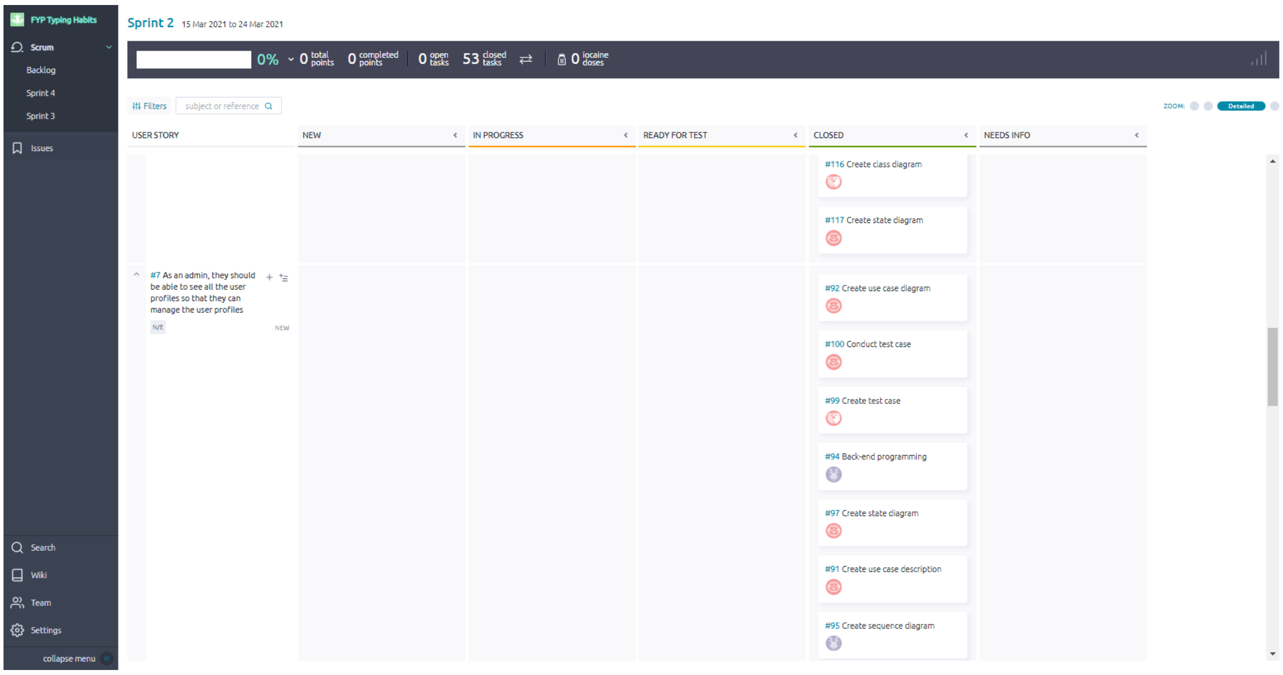


Sprint 2 15 Mar 2021 to 24 Mar 2021

0% 0 total points 0 completed points 0 open tasks 53 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
				#122 Create use case diagram	
#9 As an admin, they should be able to access the individual user profiles so that they can see user profile details N/E NEW				#112 Create use case diagram #119 Create test case #111 Create use case description #120 Conduct test case #114 Back-end programming #115 Create sequence diagram #118 Create wireframe diagram #113 Front-end programming	

Figure 24b: Taiga screenshots for sprint 2

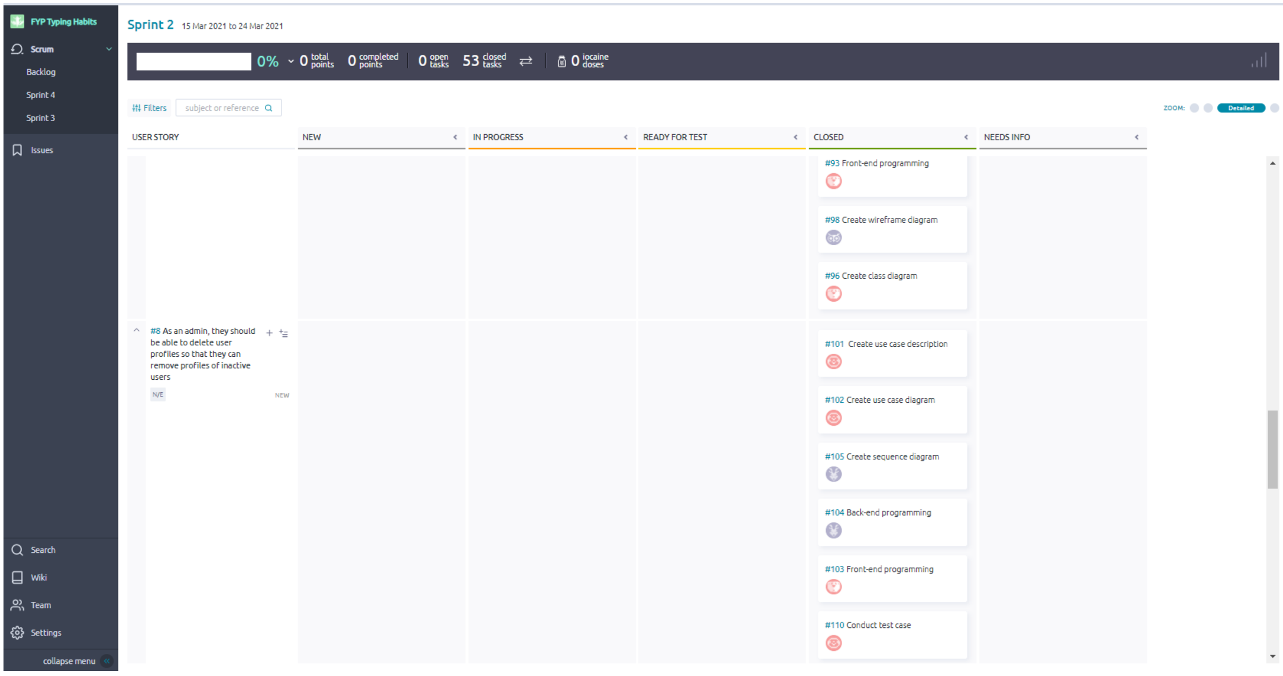


Sprint 2 15 Mar 2021 to 24 Mar 2021

0% 0 total points 0 completed points 0 open tasks 53 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
				#116 Create class diagram	
				#117 Create state diagram	
#7 As an admin, they should be able to see all the user profiles so that they can manage the user profiles N/E NEW				#92 Create use case diagram	
				#100 Conduct test case	
				#99 Create test case	
				#94 Back-end programming	
				#97 Create state diagram	
				#91 Create use case description	
				#95 Create sequence diagram	

Figure 24c: Taiga screenshots for sprint 2

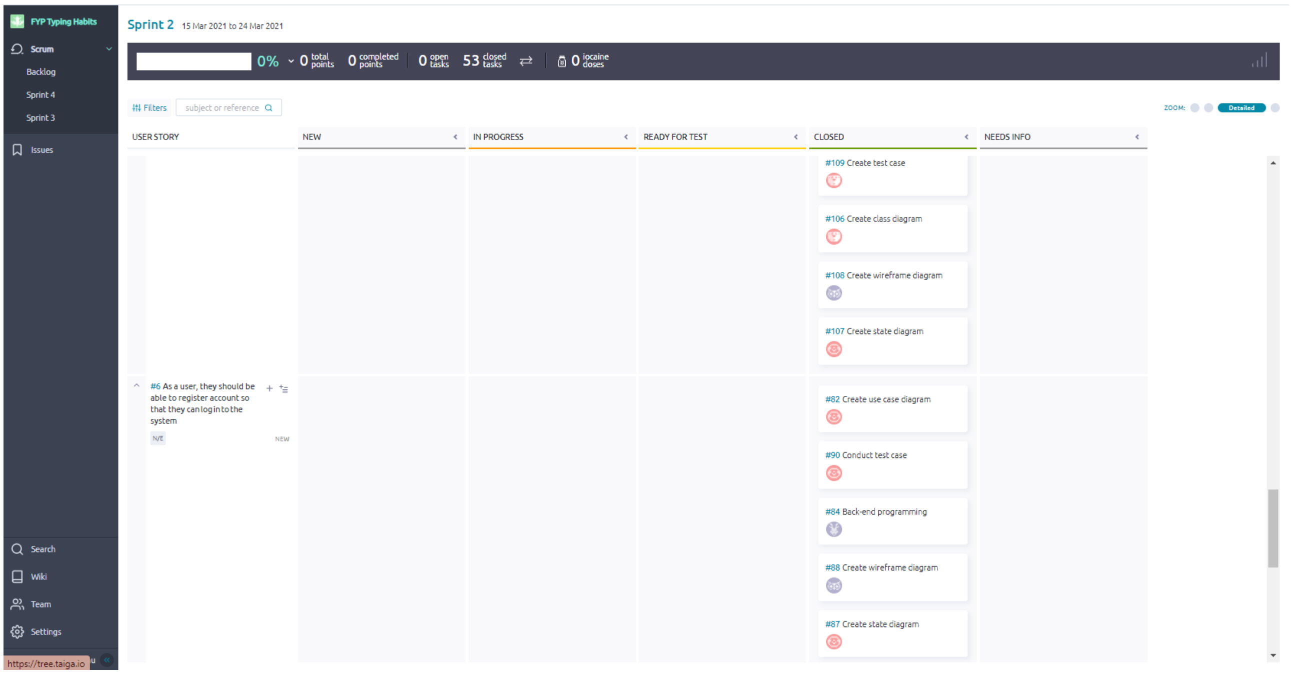


Sprint 2 15 Mar 2021 to 24 Mar 2021

0% 0 total points 0 completed points 0 open tasks 53 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
				#93 Front-end programming	
				#98 Create wireframe diagram	
				#96 Create class diagram	
#8 As an admin, they should be able to delete user profiles so that they can remove profiles of inactive users N/E NEW				#101 Create use case description	
				#102 Create use case diagram	
				#105 Create sequence diagram	
				#104 Back-end programming	
				#103 Front-end programming	
				#110 Conduct test case	

Figure 24d: Taiga screenshots for sprint 2


Sprint 2 15 Mar 2021 to 24 Mar 2021

0% 0 total points 0 completed points 0 open tasks 53 closed tasks 0 iocaine doses

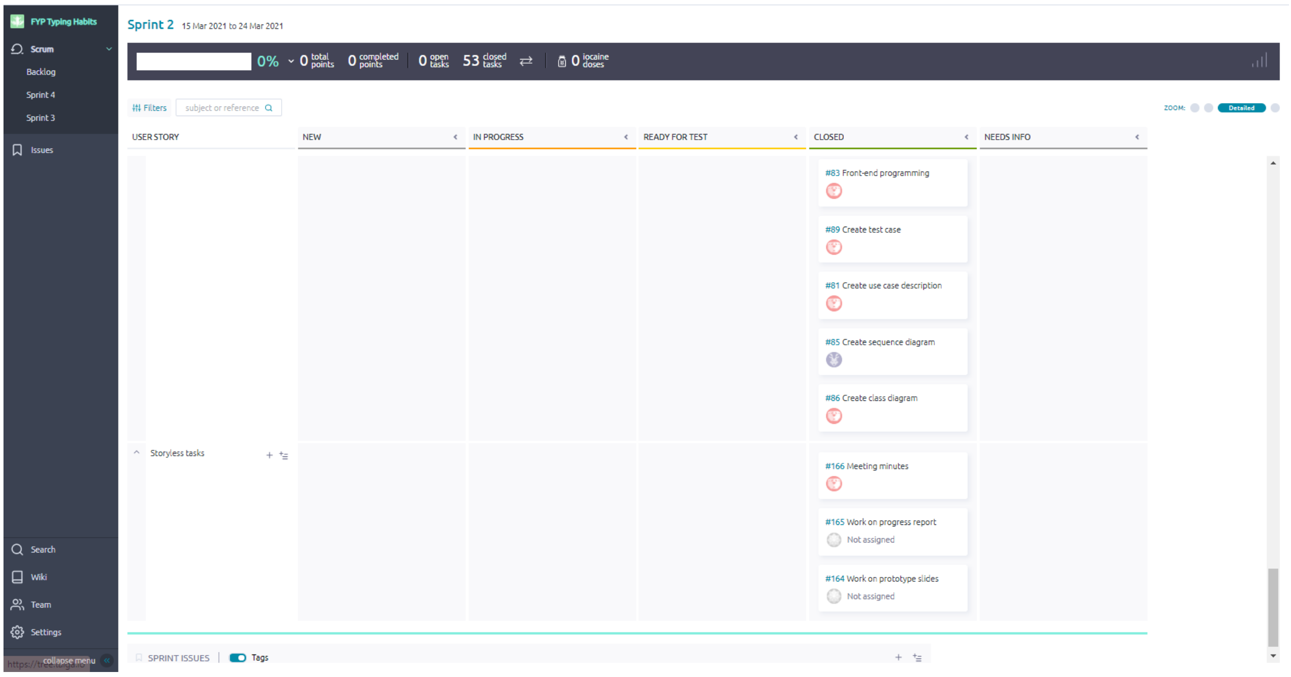
USER STORY | NEW | IN PROGRESS | READY FOR TEST | CLOSED | NEEDS INFO

CLOSED:
- #109 Create test case
- #106 Create class diagram
- #108 Create wireframe diagram
- #107 Create state diagram

#6 As a user, they should be able to register account so that they can login to the system — NEW

CLOSED:
- #82 Create use case diagram
- #90 Conduct test case
- #84 Back-end programming
- #88 Create wireframe diagram
- #87 Create state diagram

Figure 24e: Taiga screenshots for sprint 2


Sprint 2 15 Mar 2021 to 24 Mar 2021

0% 0 total points 0 completed points 0 open tasks 53 closed tasks 0 iocaine doses

USER STORY | NEW | IN PROGRESS | READY FOR TEST | CLOSED | NEEDS INFO

CLOSED:
- #83 Front-end programming
- #89 Create test case
- #81 Create use case description
- #85 Create sequence diagram
- #86 Create class diagram

Storyless tasks

CLOSED:
- #166 Meeting minutes
- #165 Work on progress report — Not assigned
- #164 Work on prototype slides — Not assigned

SPRINT ISSUES | Tags

Figure 24f: Taiga screenshots for sprint 2

5 Changes Implemented

Change of Programming language

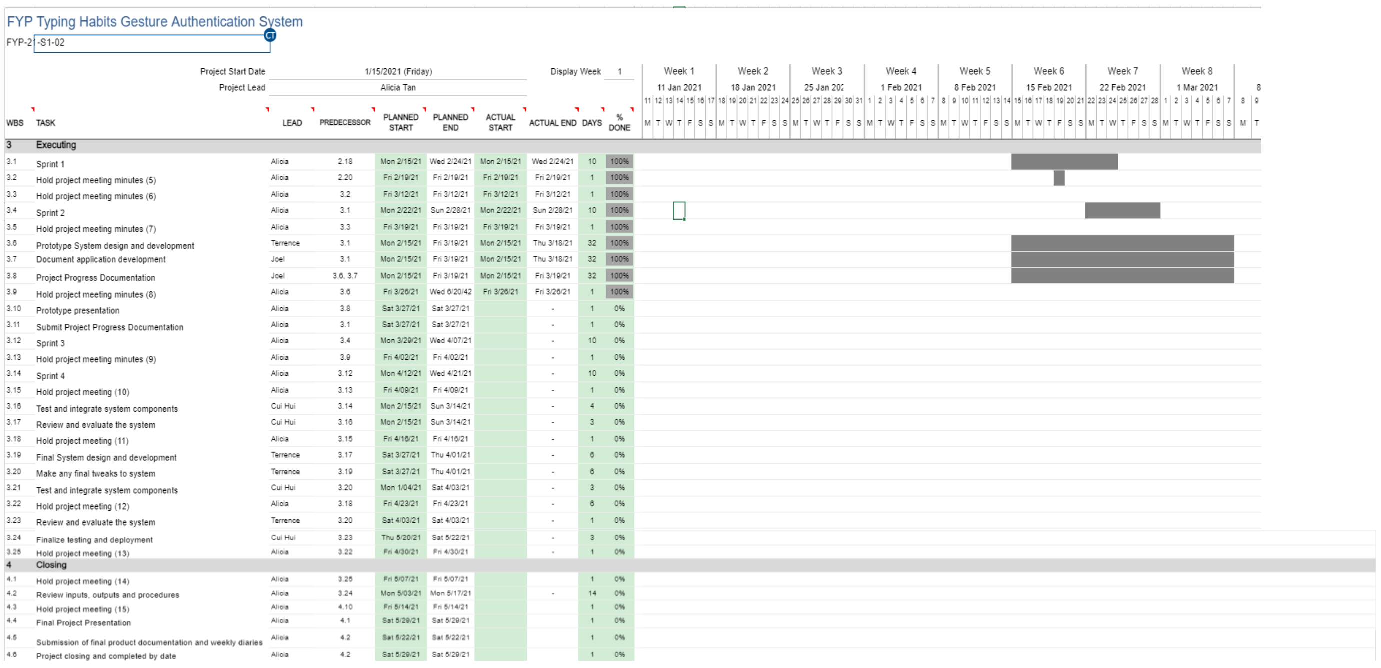
The team decided to switch from JAVA to Python programming language due to the extensive machine learning libraries found in python.

Impact

The cost of switching was minimal as it was implemented during the break in-between Sprint 1 and 2 (week 8 and 9). The buffer period allowed the programmers enough time to switch over as it was discussed early in week 8.

6 Future Work

Gantt Chart



FYP Typing Habits Gesture Authentication System

FYP-21-S1-02

Project Start Date: 1/15/2021 (Friday)

Project Lead: Alicia Tan

Display Week: 1

WBS	TASK	LEAD	PREDECESSOR	PLANNED START	PLANNED END	ACTUAL START	ACTUAL END	DAYS	% DONE
3	Executing								
3.1	Sprint 1	Alicia	2.18	Mon 2/15/21	Wed 2/24/21	Mon 2/15/21	Wed 2/24/21	10	100%
3.2	Hold project meeting minutes (5)	Alicia	2.20	Fri 2/19/21	Fri 2/19/21	Fri 2/19/21	Fri 2/19/21	1	100%
3.3	Hold project meeting minutes (6)	Alicia	3.2	Fri 3/12/21	Fri 3/12/21	Fri 3/12/21	Fri 3/12/21	1	100%
3.4	Sprint 2	Alicia	3.1	Mon 2/22/21	Sun 2/28/21	Mon 2/22/21	Sun 2/28/21	10	100%
3.5	Hold project meeting minutes (7)	Alicia	3.3	Fri 3/19/21	Fri 3/19/21	Fri 3/19/21	Fri 3/19/21	1	100%
3.6	Prototype System design and development	Terrence	3.1	Mon 2/15/21	Fri 3/19/21	Mon 2/15/21	Thu 3/18/21	32	100%
3.7	Document application development	Joel	3.1	Mon 2/15/21	Fri 3/19/21	Mon 2/15/21	Thu 3/18/21	32	100%
3.8	Project Progress Documentation	Joel	3.6, 3.7	Mon 2/15/21	Fri 3/19/21	Mon 2/15/21	Fri 3/19/21	32	100%
3.9	Hold project meeting minutes (8)	Alicia	3.6	Fri 3/26/21	Wed 6/20/42	Fri 3/26/21	Fri 3/26/21	1	100%
3.10	Prototype presentation	Alicia	3.8	Sat 3/27/21	Sat 3/27/21		-	1	0%
3.11	Submit Project Progress Documentation	Alicia	3.1	Sat 3/27/21	Sat 3/27/21		-	1	0%
3.12	Sprint 3	Alicia	3.4	Mon 3/29/21	Wed 4/07/21		-	10	0%
3.13	Hold project meeting minutes (9)	Alicia	3.9	Fri 4/02/21	Fri 4/02/21		-	1	0%
3.14	Sprint 4	Alicia	3.12	Mon 4/12/21	Wed 4/21/21		-	10	0%
3.15	Hold project meeting (10)	Alicia	3.13	Fri 4/09/21	Fri 4/09/21		-	1	0%
3.16	Test and integrate system components	Cui Hui	3.14	Mon 2/15/21	Sun 3/14/21		-	4	0%
3.17	Review and evaluate the system	Cui Hui	3.16	Mon 2/15/21	Sun 3/14/21		-	3	0%
3.18	Hold project meeting (11)	Alicia	3.15	Fri 4/16/21	Fri 4/16/21		-	1	0%
3.19	Final System design and development	Terrence	3.17	Sat 3/27/21	Thu 4/01/21		-	6	0%
3.20	Make any final tweaks to system	Terrence	3.19	Sat 3/27/21	Thu 4/01/21		-	6	0%
3.21	Test and integrate system components	Cui Hui	3.20	Mon 1/04/21	Sat 4/03/21		-	3	0%
3.22	Hold project meeting (12)	Alicia	3.18	Fri 4/23/21	Fri 4/23/21		-	6	0%
3.23	Review and evaluate the system	Terrence	3.20	Sat 4/03/21	Sat 4/03/21		-	1	0%
3.24	Finalize testing and deployment	Cui Hui	3.23	Thu 5/20/21	Sat 5/22/21		-	3	0%
3.25	Hold project meeting (13)	Alicia	3.22	Fri 4/30/21	Fri 4/30/21		-	1	0%
4	Closing								
4.1	Hold project meeting (14)	Alicia	3.25	Fri 5/07/21	Fri 5/07/21			1	0%
4.2	Review inputs, outputs and procedures	Alicia	3.24	Mon 5/03/21	Mon 5/17/21		-	14	0%
4.3	Hold project meeting (15)	Alicia	4.10	Fri 5/14/21	Fri 5/14/21			1	0%
4.4	Final Project Presentation	Alicia	4.1	Sat 5/29/21	Sat 5/29/21			1	0%
4.5	Submission of final product documentation and weekly diaries	Alicia	4.2	Sat 5/22/21	Sat 5/22/21			1	0%
4.6	Project closing and completed by date	Alicia	4.2	Sat 5/29/21	Sat 5/29/21			1	0%

Figure 25: Bottom half of the Gantt chart showing work to be completed

Implementation

Our current prototype is still in development. In the remaining 2 sprints, we plan to implement key features such as Machine Learning to authenticate users and admins, as well as additional security features should typing habit authentication fail due to various reasons.

Machine Learning

User stories to implement

1. User story #1

As a user, they should be able to get authenticated so that they can log into the application.

2. User story #5

As a user, they should be able to re-calibrate their typing habits so that they are able to pass the authentication system even if their typing habits change.

3. User story #12

As an admin, they should be able to get authenticated so that they can log into the application.

Approach

The machine learning algorithm will make use of the data collected (eg: accuracy, words per minute, time taken to time the sentence, dwell time, flight time) during registration.

The features will be compared using self-organizing map and suitable features will be used to train the model to identify and authentication users.

Two models will be adopted and compared: support vector machine (SVM) and logistic regression. The performance of the three models will be evaluated using a confusion matrix. The model with the highest accuracy and precision will be implemented in the project.

Data Visualization

Self-Organizing maps (SOM)

SOM is a neural network-based algorithm that aids in visualizing of data by reducing the dimensions. (Self-Organizing Map, 2014)

Visualizing data would be useful in showing patterns and relationships between the variables. Among the 5 variables that we will be recording, plotting SOM will show relationships between the variables and provide insight into which variables would be useful for training the machine learning algorithms. The identified variables would allow accuracy to be maximized.

Data Analytics (Training and Evaluation)

The data would be split 80:20 where 80% of the data would be assigned to train the model and 20% would be used to test the model.

Support Vector Machine (SVM)

One-class SVM would be adopted in this case as it is suitable for identifying outliers (One-class SVM with non-linear kernel (RBF), n.d.). This model is suitable for classification and regression tasks to identify outlier data.

One-class SVM model works by dividing the sample points in the training data into different types and increasing the gap between correct and incorrect data as large as possible. Incorrect training samples will be penalized and identified as incorrect. (One-Class Support Vector Machine, 2019)

The *OneClassSVM* module imported from *sklearn.SVM* library will be used to train and calculate the similarity scores for the test data set. (User Verification based on Keystroke Dynamics: Python code, 2017)

This model requires the parameters µ, σ, and kernel to be set. Then, the decision function from the model determines whether the data is an outlier. An anomaly is detected when the decision function returns false. (Zhang, Zhang, Muthuraman, & Jiang, 2007)

Logistic Regression
This model is used to calculate the probability and determine if the sample data point belongs to a particular class. For example, logistic regression could be used to estimate if the person logging in is indeed himself. The model returns a positive prediction if the estimated probability is higher than 50%. Otherwise, if predicts false instead.

Logistic regression relies on a bias term to compute a weighted sum of the features that are put into the model. However, unlike linear regression where the results are directly given, it outputs the logistic of the output result. (Géron, 2019)

Security feature

User stories to implement

1. Additional user story (Not uploaded in Taiga)

As a user, they should be able to authenticate themselves using a password in case their typing habits change.

Approach

Two user inputs will be implemented as required fields during registration (in page 1). User will have to set up their own security question and the answer to their question.

This implementation will be another way they could ensure they are able to verify themselves in the event that the authentication system is unable to identify them through their typing habit.

Admin features

User stories to implement

1. User story #13

As an admin, they should be able to interact with the authentication system so that they can perform maintenance on it

2. User story #14

As an admin, they should be able to be able to view data collected in database by authentication system so that they can analyze any anomalies

Approach

For user story #13, First, we plan to monitor the frequency of re-calibration needed for the existing users to determine the presence of odd behaviors from the user login. Then from there, the admin will be able to better manage user profile.

For user story #13, we want to implement a way using the statistics collected from users during the period of usage, like for an example, graph representation of the statistics, to better show the admin when they are carrying out any analysis.

7 Problems Encountered

The initial language chosen for programming was Java. However, after sprint 1, the team realized that implementing the machine learning algorithm in java would be challenging as the libraries available were limited (Brownlee, 2016). Thus, we decided to switch to Python as it is more popular for fraud detection and has a wider range of libraries available, which would be more applicable to this project. (Economics, 2017)

Python has a wide range of libraries for machine learning such as *scikit learn*. Scikit learn can be used for SVM and regression which are the machine learning algorithms used in the application.

The problem was quickly solved in week 9 after our exams because the idea of switching over was discussed early before Sprint 2 started. The programming team developed all the user stories in Sprint 1 and were ready to start Sprint 2 in week 10 according to the initial schedule.

8 Overall Assessment / Summary

The main objective of this project is to allow the typing habit gesture authentication system that can recognize the user based on typing habits. For the project, a Python GUI application is developed to recognize the user based on their typing habits. The project faced some difficulty along the way. However, it was quickly resolved without much impact to the timeline.

The application consists of 2 types of users – regular users and admins. Admins are able to manage users (edit and delete), while regular users are able to re-calibrate their typing habits.

To authenticate users, 2 key steps are needed: recording keystrokes and data analysis to identify users. Keystrokes are collected and 5 features are saved into individual (.txt) files accordingly. Then, the authentication system will make use of a machine learning model to identify users' typing habit and classify it is the user. To approach this, self-organizing maps are used to find the correlation between the 5 features. Then, one-class support vector machine and logistic regression are selected out of all the models to analyze the data and identify users correctly. Finally, these two models would then be compared using a confusion matrix and the model with higher accuracy will be picked.

9 References

Authentication API Documentation. (2020, November 6). Retrieved March 20, 2021, from TypingDNA: https://api.typingdna.com/index.html

Brownlee, J. (2016, July 15). *How to Use Machine Learning Algorithms in Weka*. Retrieved March 19, 2021, from Machine Learning Mastery: https://machinelearningmastery.com/use-machine-learning-algorithms-weka/

Economics, D. (2017, May 5). *What is the best programming language for Machine Learning?* Retrieved from Towards Data Science: https://towardsdatascience.com/what-is-the-best-programming-language-for-machine-learning-a745c156d6b7

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow* (2 ed.). (N. T. Rachel Roumeliotis, Ed.) O'REILLY. Retrieved March 19, 2021

One-Class Support Vector Machine. (2019, June 5). Retrieved from Microsoft Azure: https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/one-class-support-vector-machine#:~:text=More%20about%20one%2Dclass%20SVM,to%20one%20of%20two%20classes.

One-class SVM with non-linear kernel (RBF). (n.d.). Retrieved March 20, 2021, from Scikit Learn: https://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html

Self-Organizing Map. (2014). (Computer Aided Chemical Engineering) Retrieved March 2021, 20, from ScienceDirect: https://www.sciencedirect.com/topics/engineering/self-organizing-map

Try Our Demo. (n.d.). Retrieved from TypingDNA: https://www.typingdna.com/authentication-api.html#demo

User Verification based on Keystroke Dynamics: Python code. (2017, July 26). Retrieved from Machine Learning in Action: https://appliedmachinelearning.blog/2017/07/26/user-verification-based-on-keystroke-dynamics-python-code/

Zhang, R., Zhang, S., Muthuraman, S., & Jiang, J. (2007, December 14-16). One Class Support Vector Machine for Anomaly Detection in the Communication Network Performance Data. *5th WSEAS Int. Conference on Applied Electromagnetics, Wireless and Optical Communications*, 34. Retrieved from https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.332.4190&rep=rep1&type=pdf