Task 2 Test Log

Fr#	Task#	Testing type	Description of test	Expected outcome	Actual outcome	Notes
SPRINT 1						
1	1.1	Unit testing Functional test	Testing to see if flask import is working correctly	Server should be able to run on local host	Localhost:5000 is running	Localhost:5000 Is website
	1.2	Unit testing Functional test	Just to see if model runs	Model can run by itself once clicked run	Model runs with no errors	x
	1.3	Unit testing Functional test	Checking if TokaBase (database) is running	Should work as it suppose to	Not running due to not having a user created yet	Need to create a user for database
	1.4	Integration test Functional test	Checking to see if controller and model communicate with each other	Model should receive controllers' requests	Controller and model can communicate with each other by returning template	x
	1.5	unit testing integration testing	Webpage is made and displays Toka name Controller sends request to model to show database	On localhost user should be able to see Toka Fitness Title to be displayed	Toka Fitness Homepage is getting Displayed for the user once entered in search bar for localhost:5000/	
2	2.1	Unit testing Functional test	Creating a user in MySQL database and granting it access to everything	User should be created with all the permissions	User has been created and named as 'Remote'	User has now been created for database
	2.2	Unit testing Acceptance testing	Creating a form for user to be able to fill in with all the necessary details	All field names are in the form which is required and are drop downs work as well as text	Drop down works and text can be entered in the fields	x
	2.3	Unit testing	Create a form for Premium account	All field names are in the form which is required and are drop downs work as well as text	Drop down works and text can be entered in the fields	
	2.4	Unit testing	Add detail to be filled out in the forms	All information requested in the forms getting displayed	All requested information is displayed	x

	2.5	Unit testing	Making sure CSS file is linked to home page	If linked forms should be side by side instead of on top of each other	CSS has linked with the home page and now is side by side with each other	x
3	3.1	Unit testing Functional testing	Once can see and click on submit button	User can click on submit button but should do nothing	A user can now click on a button	x
	3.2	Unit testing Functional testing	Once user clicks on submit, they should be sent to a blank page (login page)	Once submit button is clicked, they get sent to another page	Once user clicks on login button it will lead them to page with login title	x
SPRINT 2						
1	1.1	Unit testing	When there is new data added to customer table it should increase by 1. This will be used to identify which customer is which as it's a unique one number	Under customeridfr there should be numbers appear starting from 1 once there is a record added	(to be tested)	This will need to be tested in the next task as right now we are not getting any new information from user
	1.2	System testing Integration testing	There should be communication from controller received user inputs from forms submit button pressed. Controller should get form information and store it in variables which then should be sent to model. Model will receive all the information sent from controller and send all data to the database file. The database should insert those variables which contain users' data into the table in SQL and return a page with a message depending on which form has been completed	Controller receive form details from user Controller sending details to model Model receiving details Model sending data to TokaBase TokaBase using the variables in inserting the data into the database	Users' data is now stored in the database. This means all information from forms is getting correctly received and sent across different modules (controller, model, tokabase). This also means that Tokabase is storing user data.	CustomerIDfr is now getting increased by 1 (1.1) Data can now be stored in the database

	1.3	Integration testing	When retrieving forms depending on the form submitted it should store the correct account type (FREE/PREMIUM) this will be used to distinguish if user is a premium user or free user	Controller should get AccountType variable with data inside. TokaBase should send data to SQL to store it	Controller now receives AccountType TokaBase saves it in the table	AccountType is hidden field user will not see it
2	2.1	Unit testing Functional testing	Checking to see once submit button is clicked it should move user to the login page	Once submit form button Is clicked user should be sent to a login page	User is now sent to login page once clicked on submit button	
	2.2	Unit testing	When user clicks on login button, they get sent to login page	User should be able to click on button and be sent across to different page	User is now able to click on login button and get sent to login page	
	2.3	Unit testing Functional testing	There should be a transition from click on login button to get to login page	Once user clicks on login button, they should be shown a form to fill in	User can click on login button and greeted with a form to fill in	
	2.4	Integrated testing	To test the login system, it will require me to get the information user has entered in login form and check it against existing users. As a result of this I need a cookies table where each user will be uniquely stored to retrieve their information and see if they can access database, they registered for	This cannot be tested or done till sprint 3	This cannot be tested till sprint 3 therefore needs to be done	SPRINT 3 to make this work
	2.5	Unit testing Accessibility testing	Just the making of the pages themselves	There should be a free dashboard page added to files	There is a Free Dashboard page	(will be tested in sprint 3)
	2.6	Unit testing Accessibility testing	Just the making of the pages themselves	There should be a premium dashboard page added to files	There is a Premium Dashboard page	(will be tested in sprint 3)
SPRINT 3						
2(BL)	2.4(bl)	Integrated testing	Testing to see if database can select the items from the database and check against user login information and if it	Users' credentials should get authenticated and	The users can now enter their details in the login form and once they click submit it authenticates	Variables were not lowercase instead of being

			matches procced them to their dashboard	processed depending on if they match	their details by checking the database and if exists sends them to the dashboard, they registered for	uppercase, so form did not pick them up.
1	1.1	Unit testing Load testing (lots of details sent across and checked against)	Testing to see if user is getting sent to the correct dashboard	Users should be sent to the correct dashboard by their credentials	User is getting sent to either Free or Premium Dashboard depending on account they registered for	Only works while login in
	1.2	Unit testing User acceptance testing	Testing to see if the user doesn't exist are they still able to enter the dashboard through login	People who have not registered should not be able to log in	If a user tries to use email or password which doesn't exist in the database, they will be shown an error screen asking them to fill the details in again to confirm	If user did not enter correct details or entered blank, they will be greeted with fail page
2	2.1	Integrated testing	Testing to see if getting users name from database and trying to display it on dashboard to see if it works all well	User's name should be displayed on the dashboard title	Users FirstName is now getting displayed in their dashboard once they log in	Only works while login in
	2.2	Unit testing	Additional feature which allows users to click on show password checkbox to see their own password	Users should be able to click on a checkbox and see their password	Users can click on check box and see their password	Uses JavaScript For checkbox
	2.3	Unit testing Functional testing	Testing to see if buttons clicked go to their pages but still have no content on them/ testing to see if users can see buttons and are able to click on them	Once user logs in checking to see if they can click any of the dashboard buttons created and go to the pages	At the moment users can only see 3 buttons and once clicked on them they get internal server error (no pages made for them)	Pages will have to be made for each other content and functionality for sprint 4
SPRINT 4						
1	1.1	Unit testing	Checking to see if pages are successfully made	There should be pages made so next test can be done which user should be able to access those pages	Pages have now been built	Needs designing

	1.2	Unit testing Functional testing	Checking to see if user can click on a button and see the page	User should be able to click on button and view page	Pages can now be viewed by users once clicked on button	Pages have now been designed
2	2.1	Unit testing	Making sure buttons created are clickable	User should be able click on squats button	User can click on squats button	
	2.2	Integration testing	Making sure once button is clicked the counter would increase by 1 only and not anymore	Once user clicks on squats button the number 0 should increase by 1	When user clicks on squat button counter goes up by 1	
	2.3	Unit testing	Creating a button so making sure use can click and see the page	User should be able click on pushups button	User can click on push-up button	
	2.4	Integration testing	Making sure once button is clicked the counter would increase by 1 only and not anymore	Once user clicks on push ups button the number 0 should increase by 1	When user clicks on push-up button counter goes up by 1	
	2.5	Integration testing	Making sure tblcookie and tblworkout can communicate by getting customers ids and storing only their specific information	Making tblworkouts in MySQL with all fields necessary to store information in them	Creating tblworkouts with column such as squats and pushups	*tblworkout is created
3	3.1	Unit testing	Making sure save button can be pressed	User should be able to click on save	User can click on a save button	
	3.2	Integration testing	Once clicked on save button all the counter information gets saved	Once clicked on save button all the counter information should get saved	Once clicked on save button all the counter information gets saved	
	3.3	Integration testing	The tblworkout should store data such as push ups and squats once clicked on save	The tblworkout should store data such as push ups and squats once clicked on save	Tblworkout stores saved information from workouts	
4	4.1	Unit testing	Making sure that there is a table for information to be stored in	Records page contains a table showing pushups and squats	Records page now shows pushups and squats as well as results from user	
	4.2	Integration testing	Looking to see if the correct data is showing for the correct user such as is the correct number of pushups, and squats done by user correct.	Table should be storing data from tblworkout such as number of squats and pushups from other peoples too	The table now stores all users results and can be added to it if other users save their items.	

SPRINT 5						
1	1.1	Unit test	Creating blogs page	Blogs page should be created	Blogs page has now been made	Not yet displaying to user
	1.2	Unit test	Adding content to the blogs page	There should be some sentences or paragraphs of information	User can now see content on the blogs page (random content)	Blogs page is now displaying to the user
	1.3	Unit test	Testing read more button to see if extra text appears	When user clicks on read more button, they can see extra text on the screen.	Instead of button its now a box once clicked opens up sub box for more information	
	1.4	Unit test	Checking to see if images can be seen by the user and if they have good resolution	Users should be able to see the images in good quality	Images are from google with creative commons license meaning we are allowed to use them. The images are also high quality	
2	2.1	Unit test	Checking if the YouTube links selected are working	User should be able to see YouTube videos	YouTube link is just an example of how and where the video would be placed on the blogs page (but reference has been made once clicked it goes to creators channel)	
	2.2	Unit test	Testing to see if video's function when clicking play such as is it running and can you hear sound	User can watch the YouTube videos	User has all controls over the video including pause and full screen	
SPRINT 6						
1	1.1	Unit test	Checking to see if buttons are correctly places and can be seen by user and clicked	User should be able to see and click on back buttons on each dashboard page	Users can see and are able to click on back button but get frown internal server error as no page is yet getting returned	Throws internal error
	1.2	Integrated test	Checking to see if back buttons go back a page	Once user clicks on back button it goes to homepage	Users can now click on back button to go to homepage	

2	2.1	-	Just making sure all lines of code are commented to say what they do	-	Most of code is commented	