Criterion B: Solution Overview

System diagram

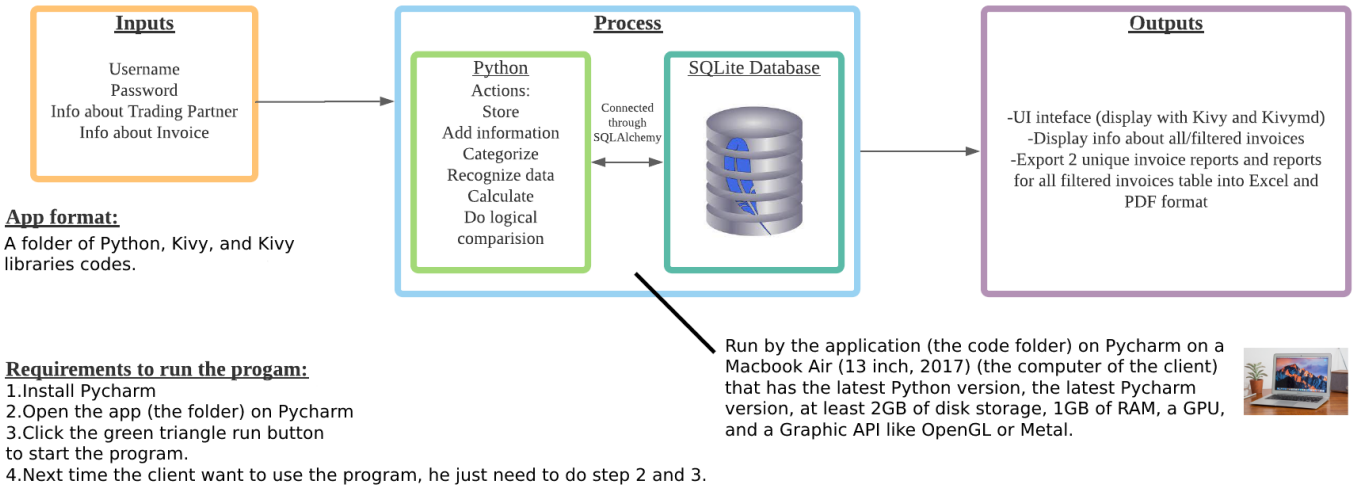


Image 1: System diagram of the program

Explanation: The program requires Mr Lam's Macbook Air to have the latest Python version, the latest Pycharm version, at least 2GB of disk storage, 1GB of RAM, a GPU, and a Graphic API like OpenGL or Metal. The program (which runs on Pycharm) lets Mr Lam input username, password, information about various trading partners and invoices. The program will then process them through actions such as storing those inputs, categorize those data and display them on filtered/searched invoices table, calculate multiple data (such as payment_unpaid_amount = invoice_amount - paid_amount) and do multiple logical comparison (such as is report_generate_day - actuatpayment_date > 0 or <0). A lot of these actions will need to do with a SQLite Database. The program will connect with this database through Python and SQLAlchemy. The program will then output those processed data through an UI interface, displaying all or filtered invoices. The program will let the client export 2 unique reports as requested or any other filtered invoices table into Excel and PDF format.

The diagram above is crucial because it allows the client to understand how to install the program, how to run it, how it runs, what it does, and what requirements his

computer needs in order to run the program. Without this diagram, it will be much harder for clients to be able to use the program.

Design of the program's front end screens

(All green colored squares in all screen designs are buttons. Anything other than them are texts or text fields. The client Mr Lam and his workers will be referred to through the word "user" throughout the explanations below for easier understanding.)

All the designs below are important because they ensures that I will be able to create easy to use GUI with Kivy for my client, improving the user experience for him with my program:

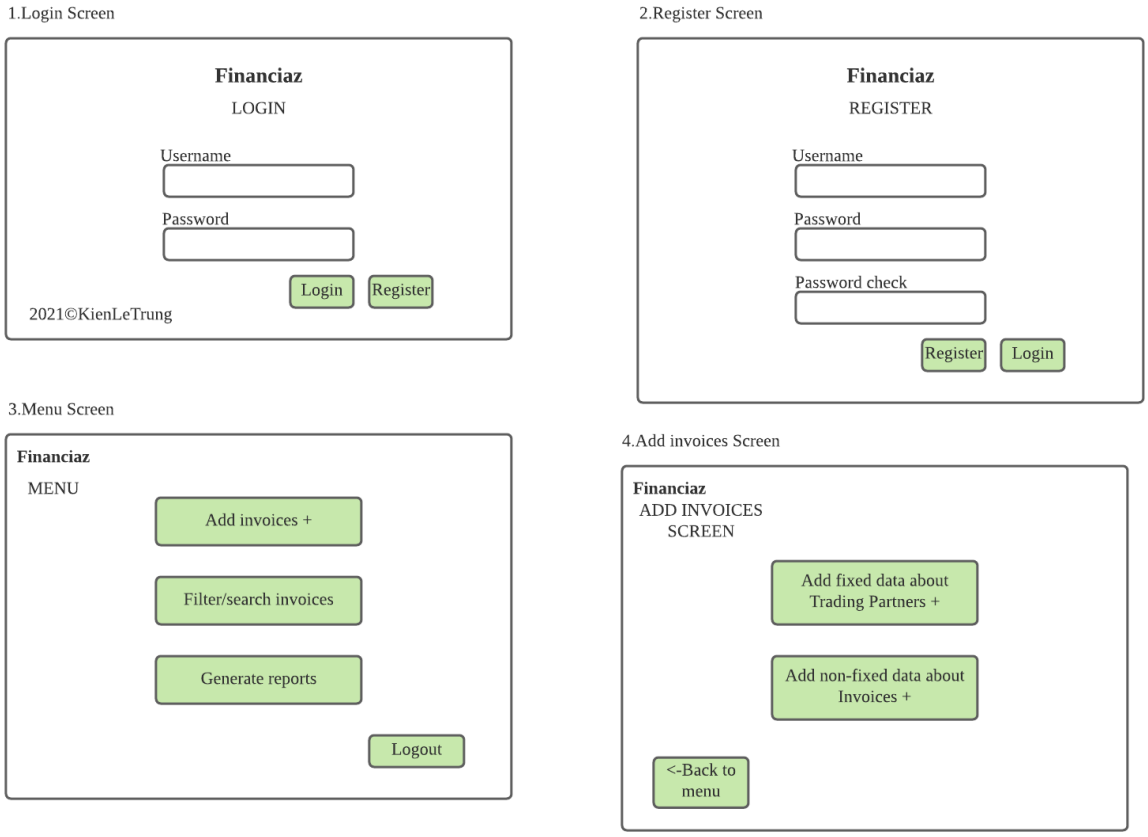


Image 2: Design of Login Screen, Register Screen, Menu Screen, Add invoices Screen

Explanation: The Login Screen (Screen no.1) lets the user login and use the program through inputting an username and a password. The user could then click the "Login" button for the authentication process. If the user hasn't created an account yet, they can click the "Register" button to go through the Register process at the Register Screen (Screen no.2). The user can create an account with an username that hasn't been registered before, and input the password twice before clicking the "Register button".

After successfully registering, the user will be redirected back to the Login Screen. After successfully logged in, the user will be redirected to the Menu Screen, where user can click on the "Add invoices +" button and go to the Add invoices Screen (Screen no.4) or click on the "Filter/search invoices" button and go to the Filter/search invoices Screen (Screen no.7) or click on the "Generate reports" button and go to the Generate reports Screen (Screen no.9) or click the "Logout" button, logging out of their account and get redirected to the Login Screen (Screen no.1). Inside the Add invoices Screen, there are 2 buttons for the user to click. The "Add fixed data about Trading Partners +" button will lead the user to the Add fixed data about trading partner Screen (Screen no.5) when clicked whereas the "Add non-fixed data about Invoices" button will lead the user to the Add non-fixed data about invoices Screen (Screen no.6) when clicked. Excluding the Login Screen (Screen no.1), the Register Screen (Screen no.2), and the Menu Screen (Screen no.3), all other screen has a “Back to menu” button that when clicked, will redirect the user back to the Menu Screen (Screen no.3).

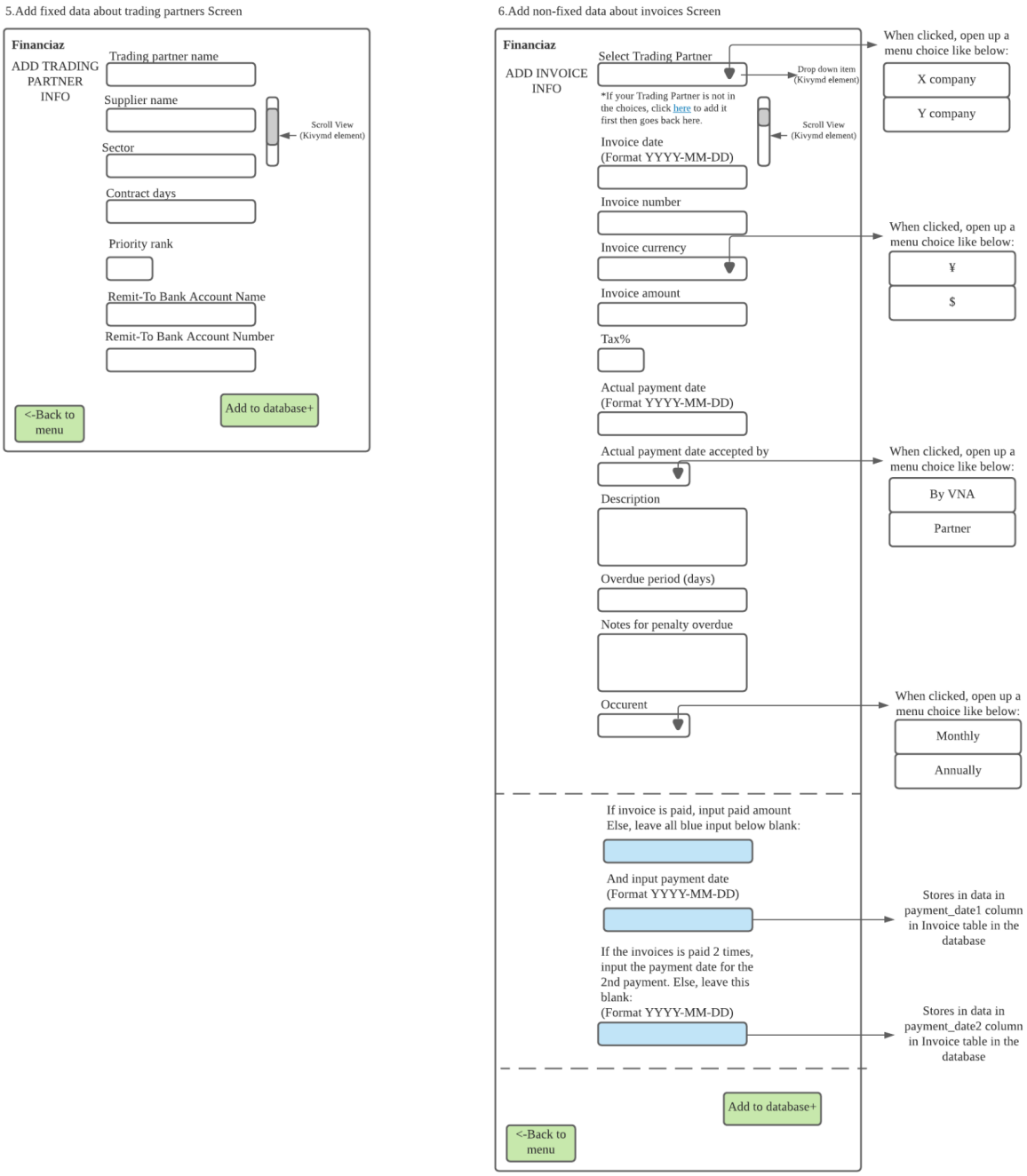


Image 3: Design of Add fixed data about trading partner Screen, Add non-fixed data about Invoice screen

Explanation: Once in the Add fixed data about trading partners Screen (Screen no.5), the user can add information about a Trading Partner by filling out all the text fields of “Trading partner name”, “Supplier name”, “Sector”, “Contract days”, “Priority rank”, “Remit-To Bank Account Name”, “Remit-To Bank Account Number”. The user can activate this adding trading partner information action by clicking the “Add to database

+” button. Once in the Add non-fixed data about invoices Screen (Screen no.6), the user can add information about an Invoice by filling out all the text fields, choosing from choice fields such as “Trading Partner”, “Invoice date”, “Invoice number”, “Invoice currency”, “Invoice amount”, “Tax%”, “Actual payment date accepted by”, “Description”, “Overdue period (days)”, “Notes for penalty overdue”, “Occurrent”. If the invoice is paid, or paid twice, the user can input optional text fields such as the paid amount, payment date, and 2nd payment date. The user can activate this adding invoice information action by cycling the “Add to database +” button.

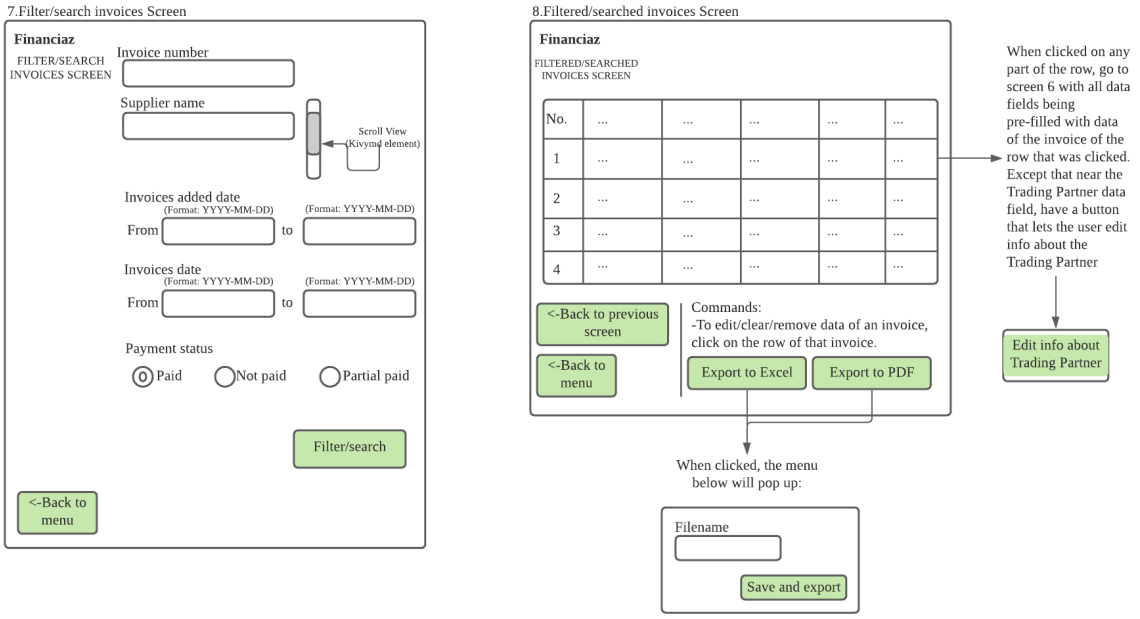


Image 4: Design of Filter/search invoices Screen, Filtered/searched invoices Screen

Explanation: Once in the Filter/search invoices Screen (Screen no.7), the user can search multiple invoices or one invoice by filling out some or all text fields such as “Invoice number”, “Supplier name”, “Invoices added date”, “Invoices date” and choosing or not choosing an option from the choice field “Payment status”. The user can activate this searching action by clicking on the “Filter/search” button. Once in the Filtered/searched invoices Screen (Screen no.8), the user can see the filtered/searched invoices table and export that table to Excel by clicking on the “Export to Excel” button or to PDF by clicking on the “Export to PDF” button. The user can click on the “Back to previous screen” button to go back to the Filter/search invoices Screen (Screen no.7) with all input fields before filled out.

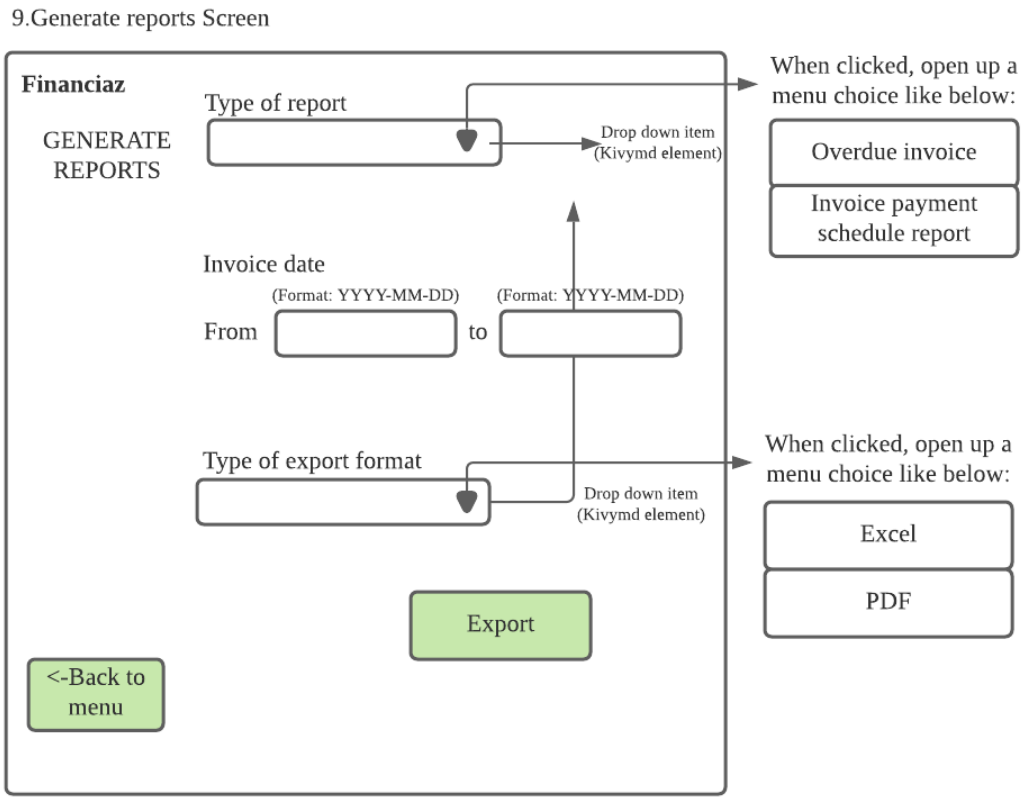


Image 5: Design of Generate reports Screen

Explanation: Once in the Generate reports Screen (Screen no.9), the user can choose the type of report they want to export through the choice field “Type of report”, then filling out the “Invoice date” text field and choosing the export format through the choice field “Type of export format”. The user can activate this export process by clicking on the “Export” button.

ER (Entity-relationship) diagram

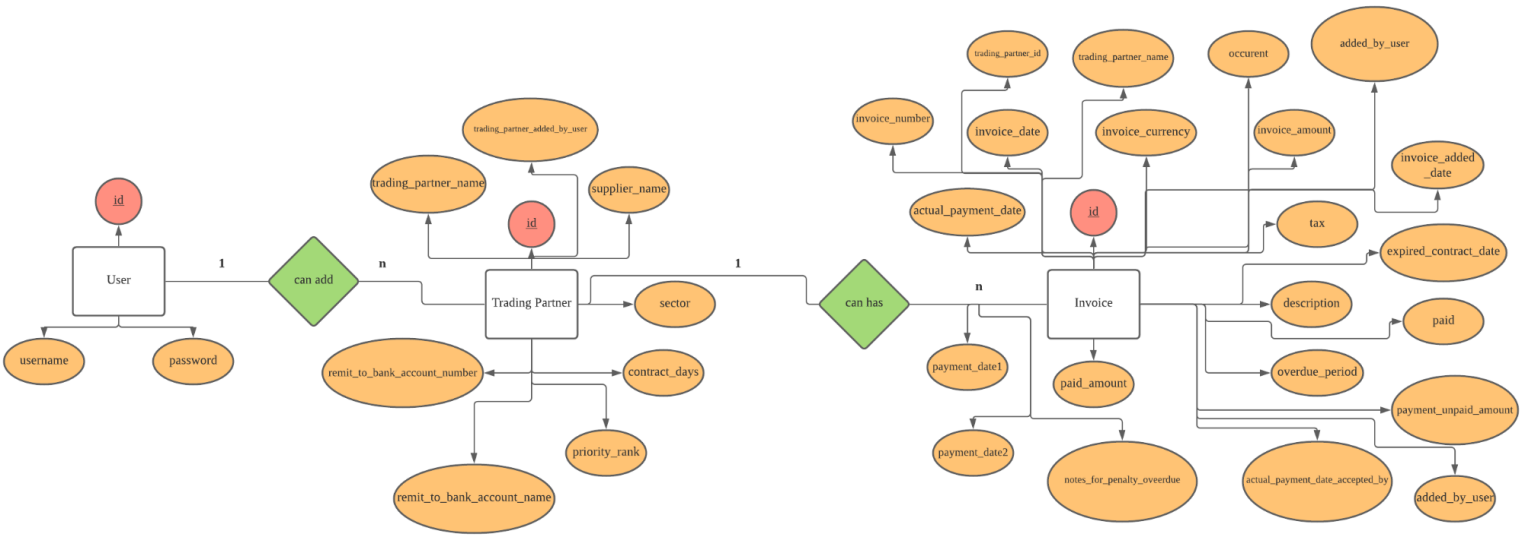


Image 6: ER (Entity-relationship) diagram of the program

Explanation: This diagram shows the relationships between the 3 classes of the 3 data tables in the database of the program. It shows that an user can add multiple trading partners, and a trading partner can be used for multiple invoices. This image is important because it allows me to design the database for my program in a way that fit my client's requirements (of allowing him to add in multiple trading partners but also pre-storing the information about an input trading partner so that all invoices added later don't require him to input the information about the trading partner again).

UML diagram/Sequence diagram

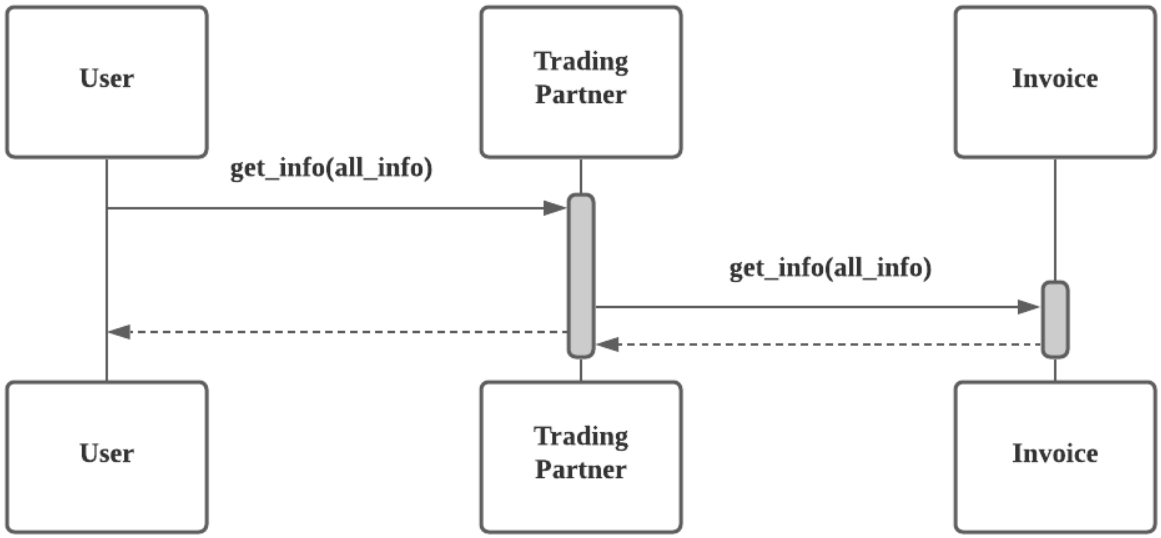


Image 7: UML diagram/Sequence diagram of a method in the program

Explanation: This diagram shows how a method can be carried out through the 3 data tables “User”, “Trading Partner”, “Invoice”. In this diagram, the get_info() method in the User table is being carried out. First, it needs to request wanted information (all_info) from the Trading Partner data table; the Trading Partner data table then needs to request wanted information (all_info) from the Invoice data table. The sequence diagram above is important because it demonstrates the function that fulfills the client’s requirement of being able to filter/search invoices on the program.

Flow diagram of general processes

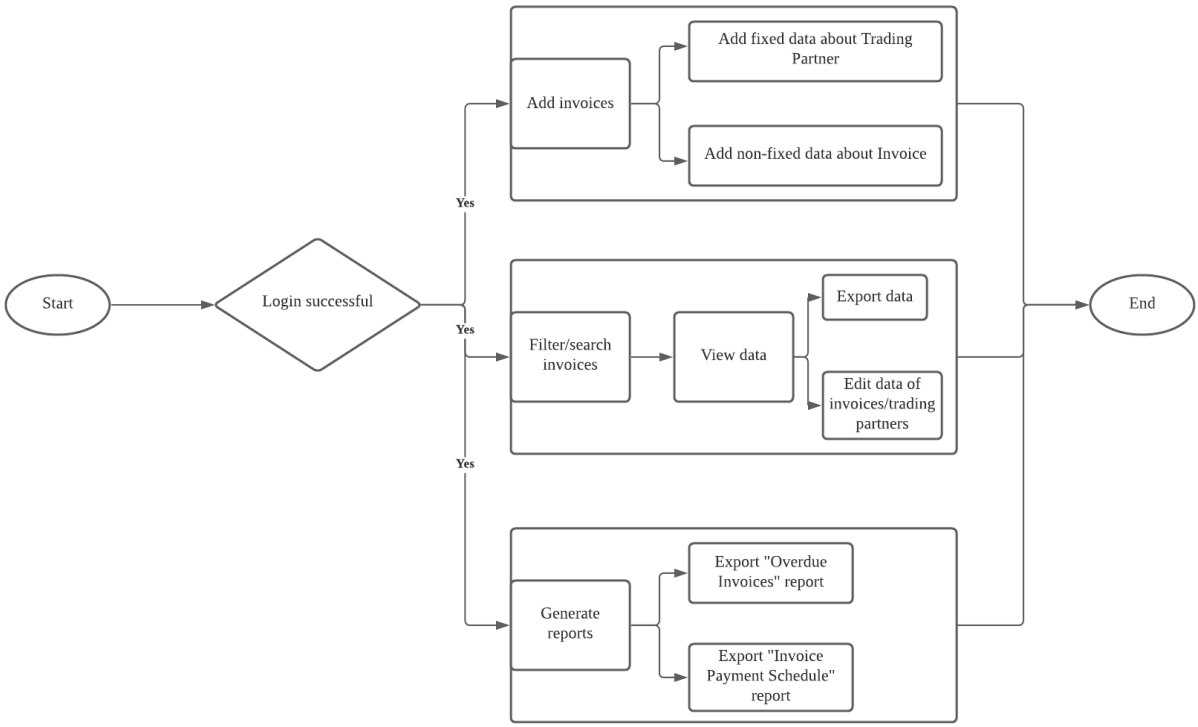


Image 8: Flow diagram of general processes of the program

Explanation: This diagram shows the general processes an user could carry out with the program.

Flow diagram of the 1st significant algorithm

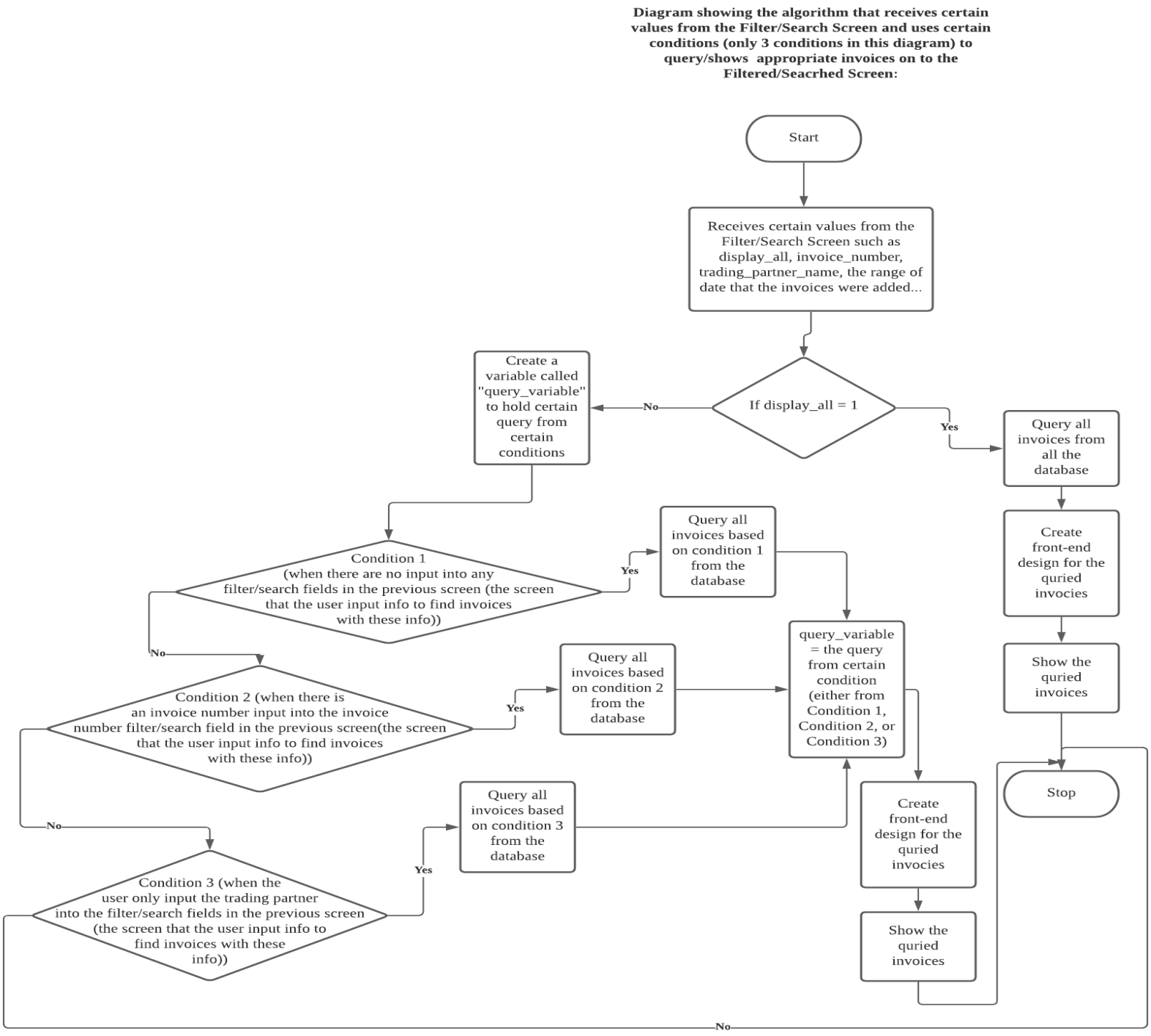


Image 9: A diagram showing the algorithm that receives certain values from the Filter/Search Screen and uses certain conditions (only 3 conditions in this diagram) to query/shows appropriate invoices on to the Filtered/Searched Screen

Explanation: this diagram shows an algorithm that receives certain values from the Filter/Search Screen and uses certain conditions (only 3 conditions (...,... , and ...) in this diagram) to query/show appropriate invoices on to the Filtered/Search Screen. The nice thing about this algorithm lies in the codes of the 3 conditions. By creating one query variable in the beginning of the 3 conditions, it removes the need for me (the developer) to code multiple same lengthy front-end codes. Using the algorithm for all invoices search conditions, I'm able to use only one long front-end code for different back-end search results.

The diagram above is important because it allows me to add in more invoice search conditions based on further requests by the client based on the way he wants to search the invoices.

Flow diagram of the 2nd significant algorithm

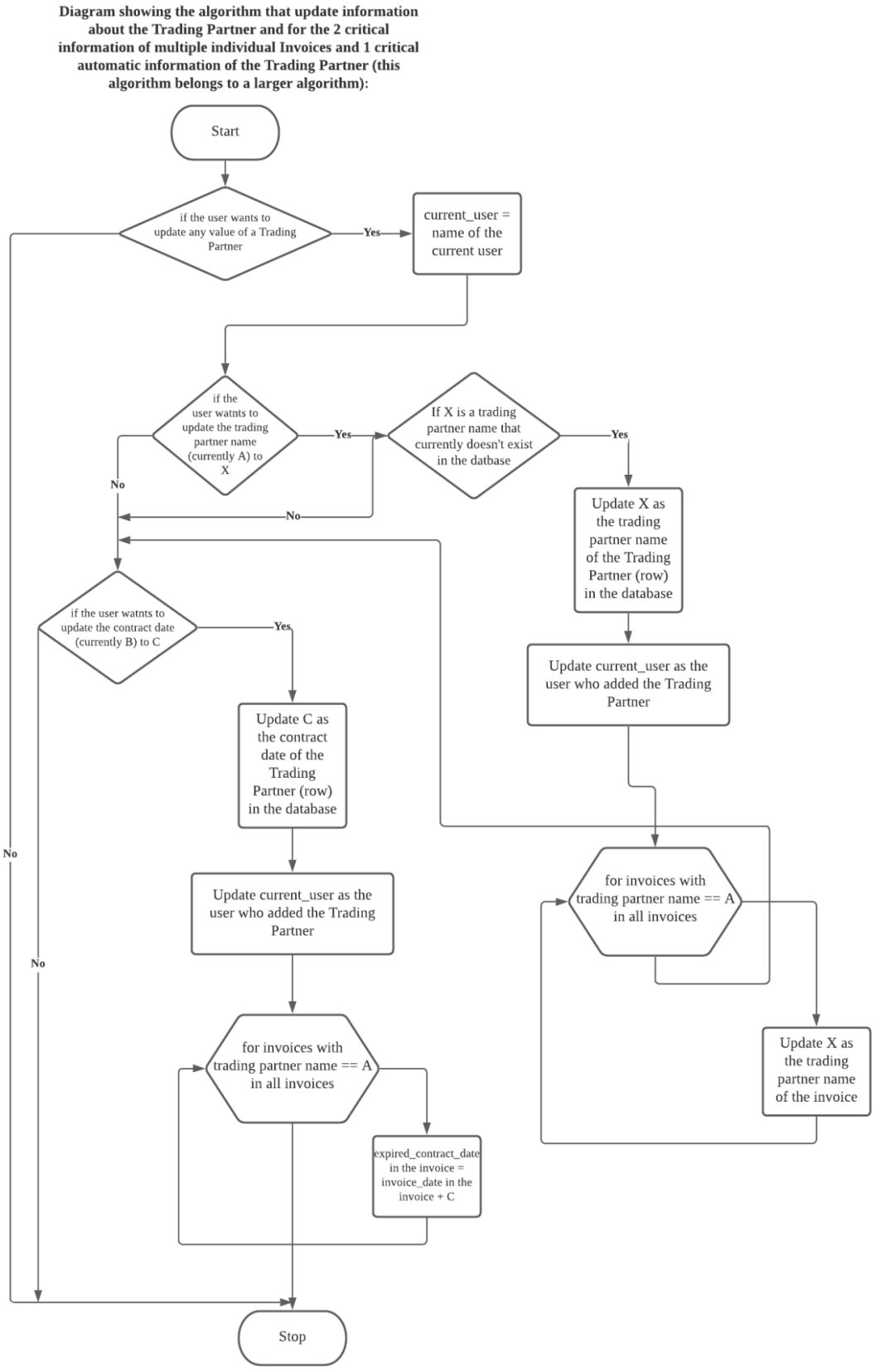


Diagram 2: a diagram showing the algorithm that update information about the Trading Partner and for the 2 critical information of multiple individual Invoices and 1 critical automatic information of the Trading Partner (this algorithm belongs to a larger algorithm)

Explanation: this diagram shows an algorithm that updates information about the Trading Partner and for the 2 critical information of multiple individual Invoices and 1 critical automatic information of the Trading Partner. As you can see from the diagram, not only does it update information of the Trading Partner, but it can also update all trading partner name, contract day, expired contract date of all Invoices with the Trading Partner that is updated, and automatically update the user who added the Trading Partner. This algorithm is critical to maintain data integrity and connection between Invoices and their Trading Partners, ensuring that the information of every invoice is properly changed and saved for the client. Without the algorithm in the diagram above, information of the invoices will be wrong after the client changed some information in the trading partner, negatively affecting the client' goal of correctly keeping track of invoices with the program.

Database normalization:

1. **For the User class:**

id	username	password
1	Lam	hash_password
2	X	hash_password
3	Y	hash_password

Table 1: A data table of the User class in the database in the program with example data Explanation: This table shows the User data table/class in the database. The table will contain the primary key attribute "id", the unique attribute "username", and a normal attribute called "password".

This table is important because it shows me how I should design the data table to store various accounts' information for the program, which will help me to fulfill many requirements given out by the client. An example of a requirement that can be fulfilled is for the client to be able to allow multiple users to use the program.

2. **For the Trading partner class:**

id	trading_partner_name	supplier_name	sector	contract_days	remit_to_bank_account_name	remit_to_bank_account_number	priority_rank	trading_partner_added_by_user
1	X partner	X	Entertainment	14	Mistsubishi	00012	3	Kien
2	YKJ	YKJ	Financial	7	Yuchou	00001	1	Kien
3	ooo.com	ooo	Security	14	Miyazaki	02200	1	Son

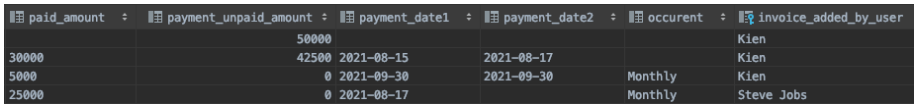
Table 2: A data table of the Trading partner class in the database in the program with example data Explanation: This table shows the Trading partner class in the database. The table will contain the primary key attribute "id", the foreign key attribute "trading_partner_added_by_user", the unique attribute "trading_partner_name", and various other normal attributes such as "supplier_name", "sector".

This table is important because it shows me how I should design the data table to store various trading partners' information for the program, which will help me to fulfill many requirements given out by the client. An example of a requirement that can be fulfilled is for the client to allow him and other accounts from other users to add information of multiple trading partners and connect them with invoices in the Invoice data table.

3. **For the Invoice class:**

id	trading_partner_id	trading_partner_name	invoice_number	invoice_date	invoice_amount	invoice_currency	invoice_added_date	tax
1	1	XX partner	237HUE28S3	2021-08-17	50000	yen	2021-08-17	10
2	1	XX partner	HJSE234HD1	2021-08-08	72500	yen	2021-08-17	10
3	1	XX partner	HD387SDJW	2021-06-04	5000	yen	2021-09-30	10
4	2	YKJ	KIENLETRUNG233	2021-07-01	25200	vietnam dong	2021-09-13	9

description	expired_contract_date	actual_payment_date	actual_payment_accepted_by	overdue_period	notes_for_penalty_overdue	paid
Invoice for toy X	2021-09-01 00:00:00	2021-09-01 00:00:00			+2.5% per year late	0
Invoice for X sandwiches	2021-08-23 00:00:00	2021-08-23 00:00:00	Partner	30	+1% every late month	0.5
Invoices for 3 pizzas	2021-06-19 00:00:00	2021-08-02 00:00:00	Partner	45		1
Invoices for advising in business 12	2021-07-31 00:00:00	2021-08-20	VNA	30	+5% per 4 months late	1



paid_amount	payment_unpaid_amount	payment_date1	payment_date2	occurent	invoice_added_by_user
	50000				Kien
30000	42500	2021-08-15	2021-08-17		Kien
5000	0	2021-09-30	2021-09-30	Monthly	Kien
25000	0	2021-08-17		Monthly	Steve Jobs

Table 3 (combined from the 3 table above, from top to bottom): A data table of the Invoice class in the database in the program with example data Explanation: This table shows the Invoice class in the database. The table will contain the primary key attribute "id", foreign key attribute "trading_partner_id", "trading_partner_name", "invoice_added_by_user" and various other normal attributes such as "invoice_date", "invoice_amount".

This table is important because it shows me how I should design the data table to store various invoices' information for the program, which will help me to fulfill many requirements given out by the client. An example of a requirement that can be fulfilled is for the client to allow him and other accounts from other users to add information of multiple invoices and connect them with their matching trading partners in the Trading partner data table.

Testing plan

Before presenting the program to Mr Lam, I plan to do various types of testings to make sure the program works as expected and fits for him. I've create a plan that I followed to test my program:

1. Alpha testing (I will check the program by myself while going through Alpha, Beta testing tables)
2. Beta testing with a computer science expert (I will ask my Computer Science classmate to check my program by using Alpha, Beta testing tables)
3. Beta testing with an external expert (I will ask an external expert, who is not taking Computer Science and not interested in coding at all, to check my program by using Alpha, Beta testing tables). This allowed me to make sure that my program could be used even by someone who is not familiar with Computer Science.
4. User acceptance testing (I will ask Mr Lam to use the program for about 1 week and then I will ask him to evaluate my program based on the User Acceptance testing table)

Testing plan for Alpha/Beta testing:

No.	Success Criteria (need to test all success criteria in Criterion A)	Type of testing	Procedures	Expected outcome
1	The program will have a login and logout screen	System Testing	1. To login when not registered an username yet: Run the program => redirected to the LOGIN screen=>click on "Register" button=>redirected to the REGISTER screen=> fill out a new username and set up a password for it=> click on "Regsiter" button again=> redirected back to the LOGIN screen=> input your just now registered username and password in the LOGIN screen=> click on "Log	1.The user will be able to login into the program when they didn't register an account for the program yet. 2.The user will be able to login into the program when they already registered an account for the program.

			in" button=> redirected to HOME screen 2. To login when already registered an username: Input your registered username and password in the LOGIN screen=> click on "Log in" button=> redirected to HOME screen 3. To logout, simply click on "Log out" button in the HOME screen=> redirected to the LOGIN screen.	3.The user will be able to log out of the program.
2	The program will let multiple user creates their account and passwords, storing those passwords in hashed	Unit Testing	Create an account at the “Register” screen of the program.	If the developers go into the database of the program to the place where it keeps user information, the developer will not be able to understand the real password of the users to log in with the user's account because all he or she can she is hashed password in the form of random combinations of letters and numbers such as "a38fjx3hsj3u".

3	Let the client add/edit/remove invoices.	Integration Testing	1.To edit an invoice, in the Filtered/Searched Invoices Screen, type in the invoice id and click the pencil icon. 2.To remove an invoice, in the Filtered/Searched Invoices Screen, type in the invoice id and click the trash icon. 3.To add an invoice, in the Filtered/Searched Invoices Screen, the user can go to the “Add Invoices/Trading partners Screen”=> go to the “Add Invoice Screen” and click the “Add to database+” button.	1.The user will then be redirected to a screen that will allow them to edit an invoice. 2.The invoice will be removed immediately from the program. 3.The invoice will be added to the database and can be checked if the user query all invoices in the “Filter/Search Invoices Screen”.
4	Program lets the client to input fixed data columns about the Trading Partner and pre-stored data in these fixed data columns for every subsequent	Integration Testing	The program should allow the user to input info about Trading partner only once and then can add multiple invoices with that Trading partner info without	Info about the invoices will be kept relatively with info of the trading partner in the database. When needed to show all information about an invoice, using the relatedness between the 2 database tables, the program could query up info from the Trading partner and show it alongside info of the invoice.

	invoices added to the program after the first time		having to input info about it again.	
5	Let the client filtered/search invoices based on required data columns such as Invoice Number, Trading Partner Name, Invoices added date, Invoices date, Payment information into tables that are viewable on the screen	Unit Testing	Go to “Filter/Search Invoices Screen”, input nothing, scroll down and click the “Filter/search” button.	In the “Filter/Search Invoices Screen”, the user can search/filter and then see invoices from the database over 16 different ways in the “Filtered/Searched Invoices Screen” (involving all required values like Invoice Number,Payment information, the Actual Payment time, Priority rank, Paid, Sector, Supplier Name)
6	Let the client export the filtered/search invoices tables (the invoices don't need to include trading partner data columns) to reports in Excel format.	Unit Testing	The user can export the filtered/search invoices table into an Excel file by clicking the “Export to Excel” button in the “Filtered/Searched Invoices Screen” => input the filename of the Excel report they want to export=>click the Export button.	The Excel report will appear inside the folder of their project/app.

7	Let the client create 2 main types of reports: Overdue Invoice Report and Invoice Payment Schedule Report and export them to reports in Excel format.	Unit Testing	1.The user click on the “Generate reports” button in the “Home” screen=> if the user wants to generate the Overdue Invoice Report, click the “Generate OVERDUE INVOICE REPORT” button. 2.If the user wants to generate the Invoice Report and Invoice Payment Schedule Report, click the “Generate INVOICE PAYMENT SCHEDULE REPORT” button.	1.The user will see an OVERDUE INVOICE REPORT that includes the filename the user input. 2.The user will see an INVOICE PAYMENT SCHEDULE REPORT that includes the filename the user input.

Table 4: Alpha, Beta testing table

Further Alpha, Beta testing table:

No.	**What I want to test**	**Type of testing**	**Procedures**	**Expected outcome**
1	Being able to view all	Unit Testing	In the Filtered/Search	The user will see very data

	data columns of each invoices save in the column		ed Invoices Screen, after clicking the button “Include Trading Partner Info” button.	columns of each filtered/search ed invoice
2	Each of data columns contains the required data format by the clients(so me are number, string, date)	Usability Testing	Query all invoices in the “Filter/Search Invoices Screen”.	When viewing the invoices, the user will see that all the data format of each invoice is appropriate to what he wants (some are number, string).
3	Let the client export the whole database of invoices to reports in Excel format.	Unit Testing	To do this, the user go to the “Filter/Search Invoices Screen”=> doesn’t set up any search conditions=>cli c k “Filter/search” button=>click “Export to Excel” button in the “Filtered/Searc h ed Invoices Screen” => input the	The Excel report of the whole invoices database will appear inside the folder of their project/app

			filename of the Excel report	
4	Each of the 2 main types of reports will need to be structured in the way that the client want (including headers, report generated time, font type)	Usability Testing	Click on the generated Overdue Invoice Report or Invoice Payment Schedule Report	In each of the Overdue Invoice Report and Invoice Payment Schedule Report, the user will see that there will be a header and a report generated date on top of each report. Also for the Invoice Payment Schedule Report, the user will see that overdue invoices will be on top of non-overdue but non-paid invoices.

“Code review”- only testing plan (additional testing plan for Alpha testing):

Criteria
Variables are named meaningfully and are easy to understand.
Functions are named meaningfully and are easy to understand.
There are explanation comments in the codes to help explain complex algorithms.

Table 5: Code review testing table

User Acceptance testing:

No.	Success Criteria	Procedures	Expected Outcome
1	The program will have a login and logout screen	1. To login when not registered an username yet: Run the program => redirected to the LOGIN screen=>click on "Register" button=>redirected to the REGISTER screen=> fill out a new username and set up a password for it=> click on "Regsiter" button again=> redirected back to the LOGIN screen=> input your just now registered username and password in the LOGIN screen=> click on "Log in" button=> redirected to HOME screen	1.The user will be able to login into the program when they didn't register an account for the program yet.
		2. To login when already registered an username: Input your registered username and password in the LOGIN screen=> click on "Log in" button=> redirected to HOME screen	2.The user will be able to login into the program when they already registered an account for the program.
		3. To logout, simply	3.The user will be

		click on "Log out" button in the HOME screen=> redirected to the LOGIN screen.	able to log out of the program.
2	The program will let multiple user creates their account and passwords, storing those passwords in hashed	Create an account at the “Register” screen of the program.	If the developers go into the database of the program to the place where it keeps user information, the developer will not be able to understand the real password of the users to log in with the user's account because all he or she can she is hashed password in the form of random combinations of letters and numbers such as "a38fjx3hsj3u".
3	Let the client add/edit/remove invoices.	1.To edit an invoice, in the Filtered/Searched Invoices Screen, type in the invoice id and click the pencil icon. 2.To remove an invoice, in the Filtered/Searched Invoices Screen, type in the invoice id and click the trash icon. 3.To add an invoice, in the Filtered/Searched Invoices Screen, the user can go to the	1.The user will then be redirected to a screen that will allow them to edit an invoice. 2.The invoice will be removed immediately from the program. 3.The invoice will be added to the database and can be checked if the user query all invoices in

		“Add Invoices/Trading partners Screen”=> go to the “Add Invoice Screen” and click the “Add to database+” button.	the “Filter/Search Invoices Screen”.
4	Program lets the client to input fixed data columns about the Trading Partner and pre-stored data in these fixed data columns for every subsequent invoices added to the program after the first time	The program should allow the user to input info about Trading partner only once and then can add multiple invoices with that Trading partner info without having to input info about it again.	Info about the invoices will be kept relatively with info of the trading partner in the database. When needed to show all information about an invoice, using the relatedness between the 2 database tables, the program could query up info from the Trading partner and show it alongside info of the invoice.
5	Let the client filtered/search invoices based on required data columns such as Invoice Number, Trading Partner Name, Invoices added date, Invoices date, Payment information into tables that are viewable on the screen	Go to “Filter/Search Invoices Screen”, input nothing, scroll down and click the “Filter/search” button.	In the “Filter/Search Invoices Screen”, the user can search/filter and then see invoices from the database over 16 different ways in the “Filtered/Searched Invoices Screen” (involving all required values like Invoice Number,Payment information, the Actual Payment time, Priority rank, Paid, Sector, Supplier Name)
6	Let the client	The user can export the filtered/search	The Excel report will appear inside the

	export the filtered/search invoices tables (the invoices don't need to include trading partner data columns) to reports in Excel format.	invoices table into an Excel file by clicking the “Export to Excel” button in the “Filtered/Searched Invoices Screen” => input the filename of the Excel report they want to export=>click the Export button.	folder of their project/app.
7	Let the client create 2 main types of reports: Overdue Invoice Report and Invoice Payment Schedule Report and export them to reports in Excel format.	1.The user click on the “Generate reports” button in the “Home” screen=> if the user wants to generate the Overdue Invoice Report, click the “Generate OVERDUE INVOICE REPORT” button. 2.If the user wants to generate the Invoice Report and Invoice Payment Schedule Report, click the “Generate INVOICE PAYMENT SCHEDULE REPORT” button.	1.The user will see an OVERDUE INVOICE REPORT that includes the filename the user input. 2.The user will see an INVOICE PAYMENT SCHEDULE REPORT that includes the filename the user input.

Table 6: User acceptance testing table