SDI E-Transport Marketplace

Group_B3_SDI_Report

PROJECT MANAGER: MAYA RHOADES – (N0901105)

SOFTWARE ARCHITECT: SAM ALLSOP - (N0894808)

SOFTWARE DEVELOPER: THOMAS MONKHOUSE – (N0913804)

SOFTWARE DEVELOPER: SHIVANGI PRAJAPATI– (N0909947)

SOFTWARE TESTER: TUGRA KARAKUS - (N0912548)

Lab Tutor: Amin Safaei

SOFT20091: Software Design and Implementation

NOTTINGHAM TRENT UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

UNITED KINGDOM

APRIL 2021

Abstract

The use of e-commerce is on the rise with the European market of e-commerce expect to grow by £31 billion from 2020 to 2024 ((Mitchell 2021). This number then increase due to the COVID-19 pandemic with online sales growing by over 50 percent for some companies (Popov 2021). While there are existing solutions such as *clickTrans* this report documents a different e-marketplace. This report features a working application that is database driven that allows drivers, cargo owners, and transport companies to all come together to get goods shipped fast and cheaply. The users can submit orders, accept, or deny orders, update order status, and more. The application provides an efficient, reliable solution to an e-transport marketplace.

Plagiarism

This report and the software it documents are the result of my own work. Any contributions to the work by third parties, other than tutors, are stated clearly below this declaration. Should this statement prove to be untrue I recognise the right and duty of the Board of Examiners to take appropriate action in line with the university's regulations on assessment.

Name: Maya Rhoades – N0901105

Name: Sam Allsop - N0894808

Name: Thomas Monkhouse – N0913804

Name: Shivangi Prajapati – N0909947

Name: Tugra Karakus - N0912548

Revision History

Version	Issue Date	Stage/Changes	Author
0.1	30/01/2022	Database Creation	Thomas Monkhouse
0.2	03/02/2022	GUI Creation	Maya Rhoades
0.3	25/02/2022	Calculations	Shivangi Prajapati
0.4	30/03/2022	Improved UI with some functions	Thomas Monkhouse Maya Rhoades
0.5	15/04/2022	Notifications Added	Thomas Monkhouse
0.6	16/04/2022	Order History added	Shivangi Prajapati
1.0	24/04/2022	Final submission validation fixes	Thomas Monkhouse Shivangi Prajapati

Table of Contents

Contents

List of Tables

List of Images

Introduction

E-Marketplaces are more popular than ever in a post pandemic world. Companies during the pandemic were left to sink or swim when it came to adapting to e-commerce ((Guajardo 2021). This report seeks to demonstrate an e-transport marketplace that could be used to ensure the survival of road transportation parties in the current commerce climate. The project begins with listing all the necessary items an e-transport marketplace must have and growing from there as shown in the rest of the report.

Background

UML Creation

Eclipse Papyrus was used to create all UML diagrams for this project. Papyrus is an open-source tool in Eclipse that can be used to quickly make UML diagrams. It provided a range of tools and suggestions to make clear and precise diagrams.

IDE

QT Creator was the integrated development environment for the project. QT makes graphical user interfaces efficiently and easily with its used of multiple languages such as C++, JavaScript, and QML. The use of QT brought the application to life and increased user experience.

Database

A MYSQL database held on a Google Cloud Platform was used for the database for the application. It is a simplified way to create, control, and connect your database to your application. This database was chosen as it was the most reliable for our database driven application, having this on the cloud means that no SQL file is required to use this application and any connection apart from NTU-Wifi can connect to this database, as cloud databases are easy to access all information will be encrypted to combat this.

Libraries

Table 1 - Libraries used

Library/Framework	Description
Vector	1D Array used for pushing and popping elements.
QStackedLayout, QListWidget,QListWidgetItem, QtWidgets.	UI elements within QT used for displaying information from calculations or SQL queries

QDateTime	Used for retrieving date and time from users' machine
QRegularExpression, regex	Used for formatting data input such as driving license and CPC number
Math.h	Header defines various mathematical functions and one macro
QSqlDatabase, QtSql,QSqlQuery,QSqlError,	Used for database connection and executing queries to our database for data pull and push
QMessageBox	Used for asking the user for order acceptance and declining them too.
QDebug	QDebug is used whenever the developer needs to write out debugging or tracing information to a device, file, string, or console
List, QString	Used for storing variables from both SQL queries and variables on the UI

Design

Planned Architecture

The planned architecture is centred around object-oriented programming and data entry into databases. This will help to coordinate and designate the functions and data, make code easier to both read and write for the application and be able to hold a large amount of data sets. This is chosen as the approach due to the application being centred around different users accessing different information. Each user will have different functions to manipulate the data as well as their own attributes so separating them into different objects would be the best approach. A database is used to store all related information being easily contained and maintained.

Requirements and Restrictions

Table 2 - Requirements and Restrictions

#	Functional Requirements	Description	Implication	Task	Actor
	Must Requirements				
1	User can create account	Allow all users to sign up to the system. Depending on type of user will ask for certain information. E.g., Driver must enter lorry details.	This will streamline the features based upon user type and allow use of the software.	T1 – create user class for each type of user with access to specific features	Cargo Owner, Driver, Transport Company
2	User can login/out	Allows only valid users to enter the system and to have stored information.	Decreases amount of time spent entering information by	T2 – Validation check of credentials	Cargo Owner, Driver, Transport Company

			signing in to saved account.		
3	Cargo Owner can place an Order	System allows Cargo Owner to place order containing source, destination, cargo dimensions, weight, and condition.	The variety of order types allows for a variety of orders and will affect further calculations and delivery.	T3 – Create order class with input from cargo owner for order specifics	Cargo Owner
4	Users can change and view order status	System must allow user and company to track the status of the order as indicated by the driver. Status such as loading, enroute, or delivered.	This will give peace of mind to all users and allow for a record if something goes wrong with delivery.	T4 – Use message passing to notify of order status with the message being sent from driver to company.	Cargo Owner, Transport Company
5	Cargo Owner can calculate shipping price	System will calculate shipping price based on the source, destination, and needed lorry for each order.	Allows flexible price of orders and negates the need for cargo owner to contact company for prices therefore improving customer service.	T5 -Create function to calculate prices based upon variables given by user class	Cargo Owner
6	Transport company can calculate commission	System must calculate the company's commission of each order based upon shipping rate.	Results in the company receiving their cut of each order.	T6 – Create function to calculate commission based upon shipping rate function.	Transport Company
7	Drivers can accept or reject orders until a Driver is found	System will notify driver of available order and driver can accept or reject the cargo. If rejected will send to another driver.	Allows for only available drivers to accept cargo and ensures order will be passed until a driver is found decreasing driver loads and delivery times.	T7 – Create a function that allows order to be re sent to another driver if rejected	Driver
8	Drivers can verify their information	System should verify CPC/lorry	Improves communication	T8 – Implement API verification	Driver

		registration number via API	of software and ensures only validated drivers can transport cargo.		
9	System must encrypt all information	Systems should encrypt user details when stored.	Improves security and user protection.	T9 – Implement encryption software	Cargo Owner, Driver, Transport Company
10	System must transfer files	System must have a file transfer system to send/ receive orders, notifications, etc.	Creates a data exchange so the software can function.	T10 – Create file sharing function	Cargo Owner, Driver, Transport Company
11	System must use message passing	System must user message passing to select and run code.	Allows for software to run functions in specific orders and after certain triggers, improving software process.	T11 – Create message passing system to be used in other requirements.	Cargo Owner, Driver, Transport Company
	Should Requirements				
12	Users can send and receive notifications	System must send notifications to company when they receive an order, driver when new order is available, and cargo owner when order is accepted.	Keeps all parties apprised of success of order improving user mood and starts several other processes of software such as invoicing.	T12 – Message passing implemented to send notifications.	Cargo Owner, Driver, Transport Company
13	Display delivery success notification and evidence of delivery	System must display delivery notification along with proof of deliver.	Decreased liabilities for company as proof is available upon delivery.	T13 – Include an image file of proof of delivery in notifications	Cargo Owner, Driver, Transport Company
14	Users can leave and view feedback	Allows Cargo owners to leave feedback and comments about orders and delivery.	Increased customer satisfaction as they can have an impact on the company.	T14 – Create user accessible comment file and display in interface.	Cargo Owner
15	Transport company can send invoice to cargo owner	Systems must send invoice (based upon shipping rate) to	This ensure that when the company accepts the delivery they	T15 – create invoice function that calculates the rate and	Cargo Owner, Transport Company

		cargo owner upon acceptance of driver.	are being paid for their time.	sends it to cargo owner.	
	Could Requirements				
16	Allow creation of tenders to assist users in choosing carriers etc.	Allow shipper to choose one of several prices and accept the tender. Allow each tender to be created by a proposed price from Carrier etc.	Allows for customer satisfaction through best prices	T16 – Create a tender class that can be create and accepted.	Cargo Owner
17	Users can view order or shipping history	Allows cargo owner/company to view order history and drivers to view shipping history.	Increased customer services as customers will know what they have ordered before. Drivers also have proof as what orders they have handled before showing experience in transport.	T17 – Create secure files for each user to access that are updated as more orders are placed.	Cargo Owner, Driver, Transport Company
18	System could store information to allow contact of users	System must store and track information of users provided in sign up function and order history.	Increases efficiency of process as information is store and easily accessible.	T18 – File management of user information	Cargo Owner, Driver, Transport Company
19	Users can edit their details	Allow user to update information.	This will allow all information to stay up to date and increase efficiency of orders/delivery.	T19 – Allow user class to access certain files	Cargo Owner, Driver, Transport Company
20	Users can track lorry and driver	Allow real time gps tracking of lorry and driver.	Allows users to efficiently use their time as they will know when orders are leaving/arriving.	T20 – Create a simulated tracking area	Cargo Owner, Transport Company
21	Allows offline message storing	System must store messages and notifications until user is active.	Improves user experience as they will not miss any notifications.	T21 -Create a system to store messages until active status is engaged.	Cargo Owner, Driver, Transport Company

22	Verify phone number	System should verify phone number before adding to profile.	Increases security and safety of users.	T22 – Verify the legitimacy of phone number	Cargo Owner, Driver, Transport Company

Non-Functional Requirements	Description	Implications	Tasks
User Friendly			
Response Time of < 4 seconds	Application respond time must be below 4 seconds for any action	Improves user experience	NF01 – Carefully choose options to reduce response time
System must have GUI	Users must interact with program through a graphical user interface	Improves user experience and understanding	NF02 – Create a GUI
Allow a minimum of 3 clicks to get to any given feature	Users must be able to access a feature in a minimum of 3 clicks.	User Satisfaction and decreased user time.	NF03 – Simplify GUI layout
System must have a clear file management system	System must have file management system to store user details and order information clearly and repeatable.	Improves access to information needed.	NF04 – Create file management system
Legal			
Follow GDPR regulation	System needs to adhere to the Data Protection Principles by following data protection and privacy law.	Creates security of user information and user peace of mind.	NF05 – Allow users access to secure information and ability to delete information
Security			
Ensure complexity of passwords	Password should be 8 characters long, contain a capital letter and number. The more complex the number the more secure it can be.	Strong password ensures secure account protection for user.	NF06 – Create loop to check for valid password
Ensure each user type only has access to their features	A Driver cannot access the Cargo Owners abilities vice versa, etc. Limiting capabilities to only relevant user roles.	Provides users with streamlined service and secures information of other users.	NF07 – Deny access to other user information
Reliability			
System must Handle errors	Software should handle improper input or any errors.	Makes software more useable and reliable.	NF08 – Implement error handling code such as try/catch
System must handle at least 10 users at any time	System must handle at least 10 users as it is small scale but still needs to handle stress.	Software will be more reliable and open to mire users.	NF09 – System handles multiple users.

Risk Management

Table 3 - Risk Management

N0.	Risk	Probability	Impact	Description	Action Plan
			Time Risks		
1	Underestimated Task duration	5	3	Task takes longer than the planned timeline allowed.	Team will work together to finish Task well and get back on track.
2	Change in Time Frame	2	5	Deadlines are moved either forward or backwards.	Project schedule will shift to accommodate but stay the same if deadline is moved back.
			Cost Risk		
3	Lack of Budget	3	2	There is no monetary budget as all aspects of project should be free.	If money is needed to fulfil software needs, then a monetary source will be located.
			Technical Risk		
4	Lack of Software Knowledge	4	4	Members have not done this type of work and design before.	Further research shall be conducted to gain the lacked information.
5	Change in requirements	4	3	Requirements are removed or add based upon project scale or assessment.	Project will be implementing the must have requirements first so any further changes will be minimal.
6	Overcomplication of Tasks	2	1	Too many tasks were taken onboard.	Project will be reduced to the must have requirements as to complete a final product.
7	Technology Failure	3	1	A technological failure such as a hard drive or computer.	Members will save all work to multiple locations and online in GitHub to reduce damage and will

					share technology is possible.
Human Resource Risk					
8	Member Illness	5	3	A member becomes ill and cannot participate in the project for a given time.	Team will work to help take up the slack while the member recovers and give the member plenty of ways to make up the work.
9	Lack of Contribution	1	3	A member fails to contribute an equal amount or at all.	Project Manager will report said person to the module leader.
11	Communication Failure	2	1	Communication failure such as Microsoft teams is down.	Communication will be restored through another means such as email and skype.
12	Overwhelming Members	3	3	A team member becomes overwhelmed with the amount of work or receives more of the work than others	All team members will help to work each other through the project. If one becomes overwhelmed team will shift balance accordingly.

Monitoring Tools

GitHub

GitHub will be used to monitor and change our code. It allows multiple people to work on the project and combine their work. It is also useful for storing any previous versions of the code to go back to if serious problems arise.

Link to GitHub -> B3/SDI_Project (ntu.ac.uk)

Microsoft Teams

Microsoft Teams allows the team to effectively communicate. It provides a method to notify teammates of changes, holds files, conduct biweekly meetings, and work on code together.

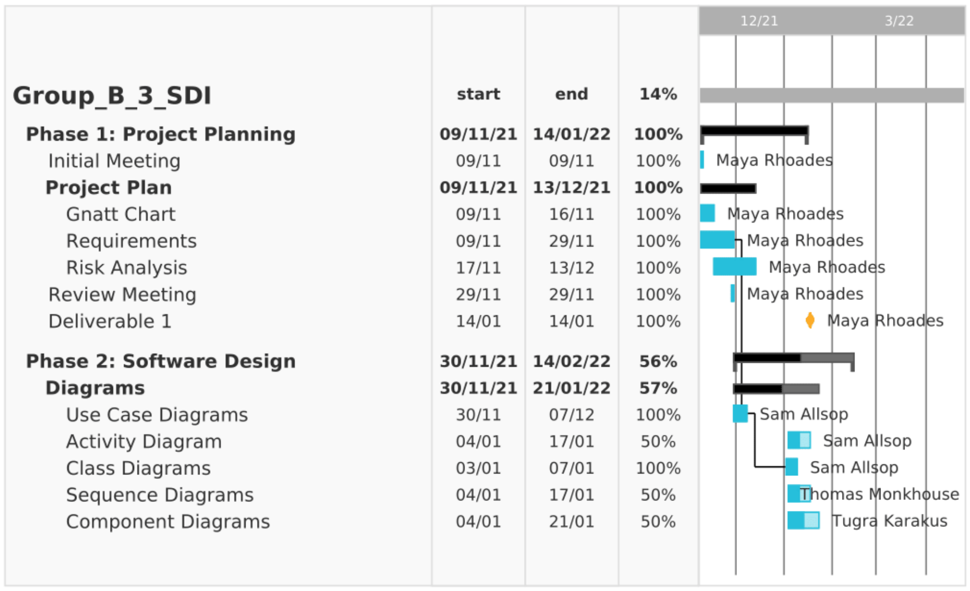
Trello

Trello's usage in the project is for keeping track of weekly tasks. Each week new task is posted so team members are kept up to date and tracked on tasks.

Time Plans

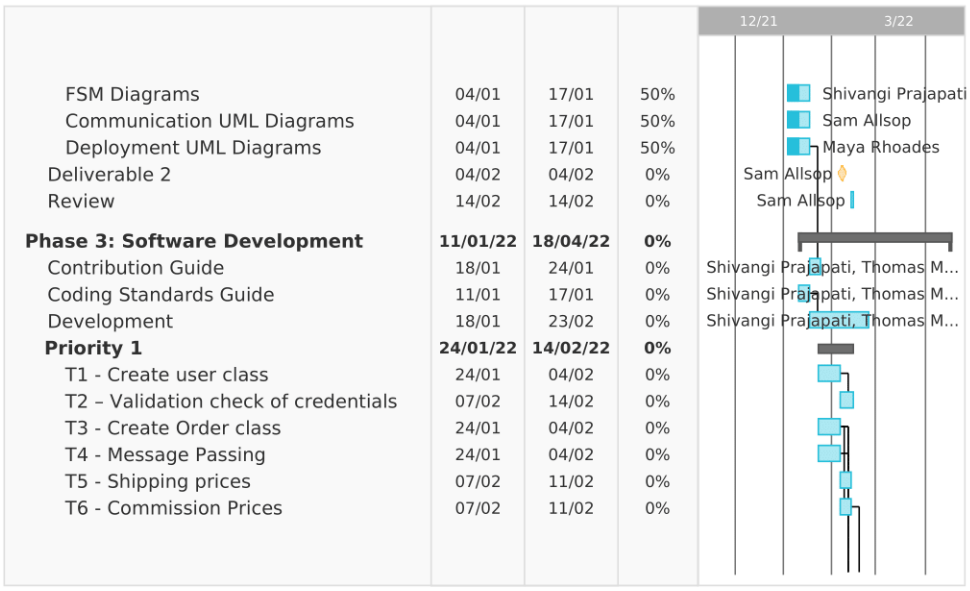
Depicted below are two version of the Gantt chart for the project. The First shows all tasks, who they belong to, and when they should be completed. The second shows an enlarged view of the scheduling of the project to better demonstrate the time plan.

Group_B3_SDI_Gnatt_Chart_List



Group_B_3_SDI	**start**	**end**	**14%**
Phase 1: Project Planning	**09/11/21**	**14/01/22**	**100%**
Initial Meeting	09/11	09/11	100%
Project Plan	**09/11/21**	**13/12/21**	**100%**
Gnatt Chart	09/11	16/11	100%
Requirements	09/11	29/11	100%
Risk Analysis	17/11	13/12	100%
Review Meeting	29/11	29/11	100%
Deliverable 1	14/01	14/01	100%
Phase 2: Software Design	**30/11/21**	**14/02/22**	**56%**
Diagrams	**30/11/21**	**21/01/22**	**57%**
Use Case Diagrams	30/11	07/12	100%
Activity Diagram	04/01	17/01	50%
Class Diagrams	03/01	07/01	100%
Sequence Diagrams	04/01	17/01	50%
Component Diagrams	04/01	21/01	50%

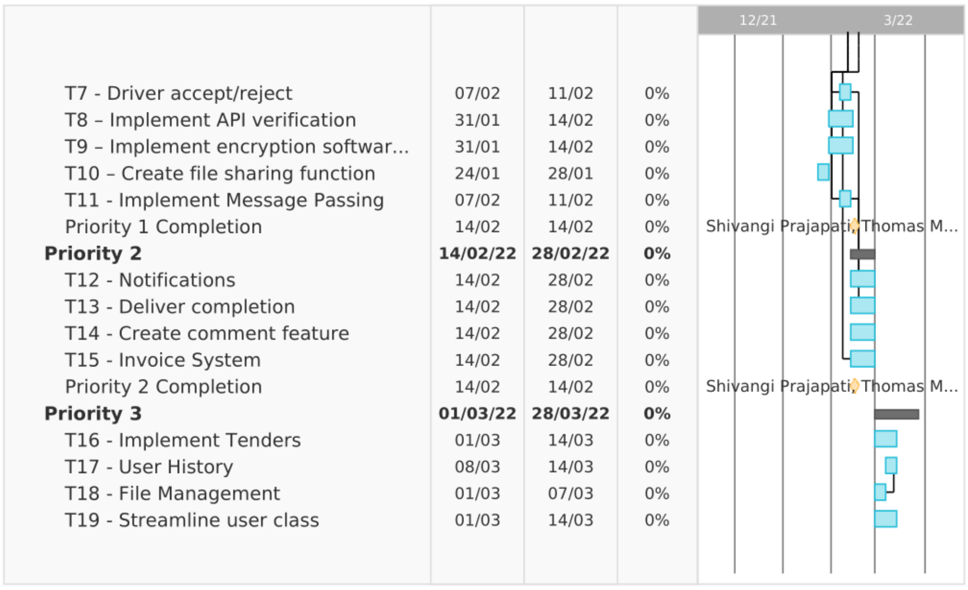
Figure 1: Gantt Chart List 1



FSM Diagrams	04/01	17/01	50%
Communication UML Diagrams	04/01	17/01	50%
Deployment UML Diagrams	04/01	17/01	50%
Deliverable 2	04/02	04/02	0%
Review	14/02	14/02	0%
Phase 3: Software Development	**11/01/22**	**18/04/22**	**0%**
Contribution Guide	18/01	24/01	0%
Coding Standards Guide	11/01	17/01	0%
Development	18/01	23/02	0%
Priority 1	**24/01/22**	**14/02/22**	**0%**
T1 - Create user class	24/01	04/02	0%
T2 – Validation check of credentials	07/02	14/02	0%
T3 - Create Order class	24/01	04/02	0%
T4 - Message Passing	24/01	04/02	0%
T5 - Shipping prices	07/02	11/02	0%
T6 - Commission Prices	07/02	11/02	0%

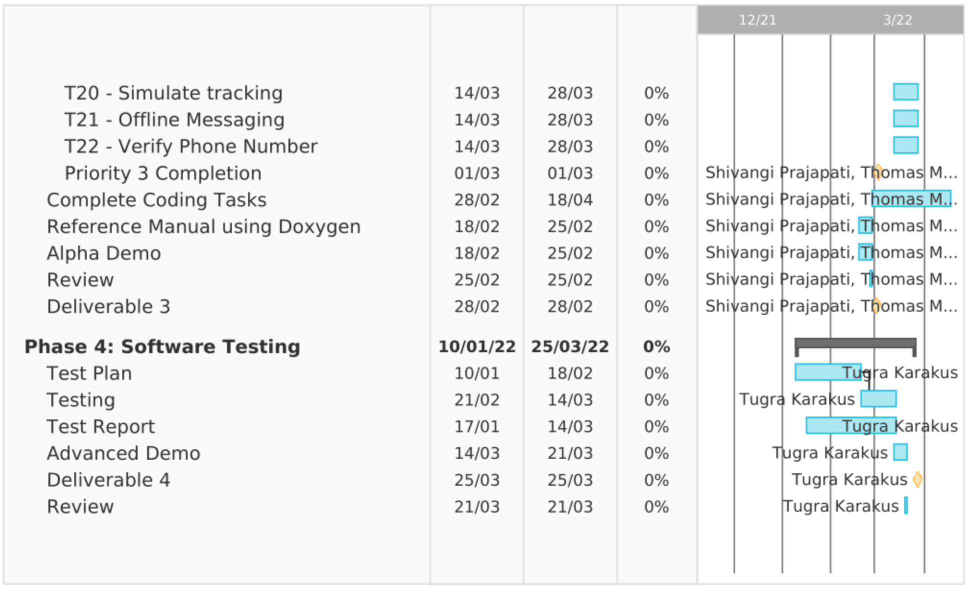
12/21
3/22
Shivangi Prajapati
Sam Allsop
Maya Rhoades
Sam Allsop
Sam Allsop
Shivangi Prajapati, Thomas M...
Shivangi Prajapati, Thomas M...
Shivangi Prajapati, Thomas M...

Figure 2: Gantt Chart List 2



T7 - Driver accept/reject	07/02	11/02	0%
T8 – Implement API verification	31/01	14/02	0%
T9 – Implement encryption softwar…	31/01	14/02	0%
T10 – Create file sharing function	24/01	28/01	0%
T11 - Implement Message Passing	07/02	11/02	0%
Priority 1 Completion	14/02	14/02	0%
Priority 2	**14/02/22**	**28/02/22**	**0%**
T12 - Notifications	14/02	28/02	0%
T13 - Deliver completion	14/02	28/02	0%
T14 - Create comment feature	14/02	28/02	0%
T15 - Invoice System	14/02	28/02	0%
Priority 2 Completion	14/02	14/02	0%
Priority 3	**01/03/22**	**28/03/22**	**0%**
T16 - Implement Tenders	01/03	14/03	0%
T17 - User History	08/03	14/03	0%
T18 - File Management	01/03	07/03	0%
T19 - Streamline user class	01/03	14/03	0%

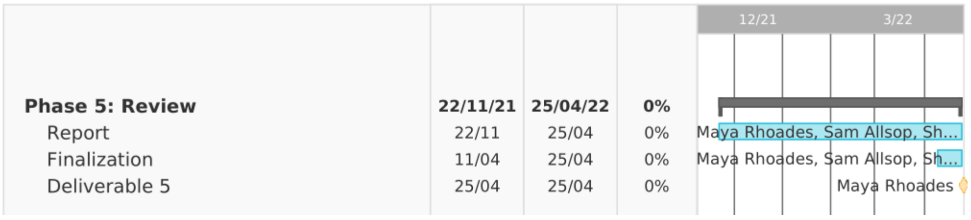
Figure 3: Gantt Chart List 3



T20 - Simulate tracking	14/03	28/03	0%
T21 - Offline Messaging	14/03	28/03	0%
T22 - Verify Phone Number	14/03	28/03	0%
Priority 3 Completion	01/03	01/03	0%
Complete Coding Tasks	28/02	18/04	0%
Reference Manual using Doxygen	18/02	25/02	0%
Alpha Demo	18/02	25/02	0%
Review	25/02	25/02	0%
Deliverable 3	28/02	28/02	0%
Phase 4: Software Testing	**10/01/22**	**25/03/22**	**0%**
Test Plan	10/01	18/02	0%
Testing	21/02	14/03	0%
Test Report	17/01	14/03	0%
Advanced Demo	14/03	21/03	0%
Deliverable 4	25/03	25/03	0%
Review	21/03	21/03	0%

12/21
3/22
Shivangi Prajapati, Thomas M…
Shivangi Prajapati, Thomas M…
Shivangi Prajapati, Thomas M…
Shivangi Prajapati, Thomas M…
Shivangi Prajapati, Thomas M…
Shivangi Prajapati, Thomas M…
Tugra Karakus
Tugra Karakus
Tugra Karakus
Tugra Karakus
Tugra Karakus
Tugra Karakus

Figure 4: Gantt Chart List 4



Phase 5: Review	**22/11/21**	**25/04/22**	**0%**
Report	22/11	25/04	0%
Finalization	11/04	25/04	0%
Deliverable 5	25/04	25/04	0%

Figure 5: Gantt Chart List 5

Group_B3_SDI_Gnatt_Chart_Enlarged

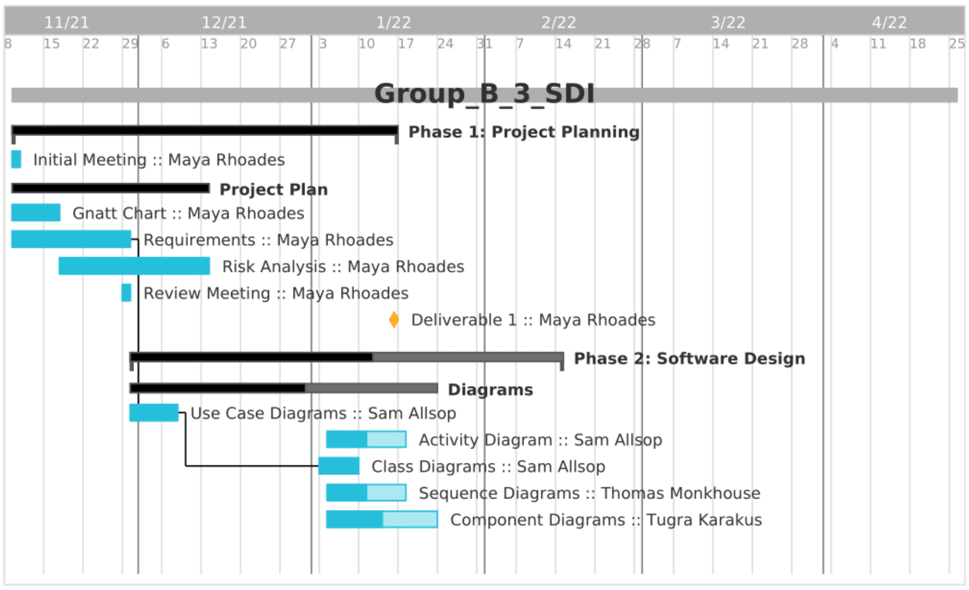


Figure 6: Gantt Chart Enlarged 1

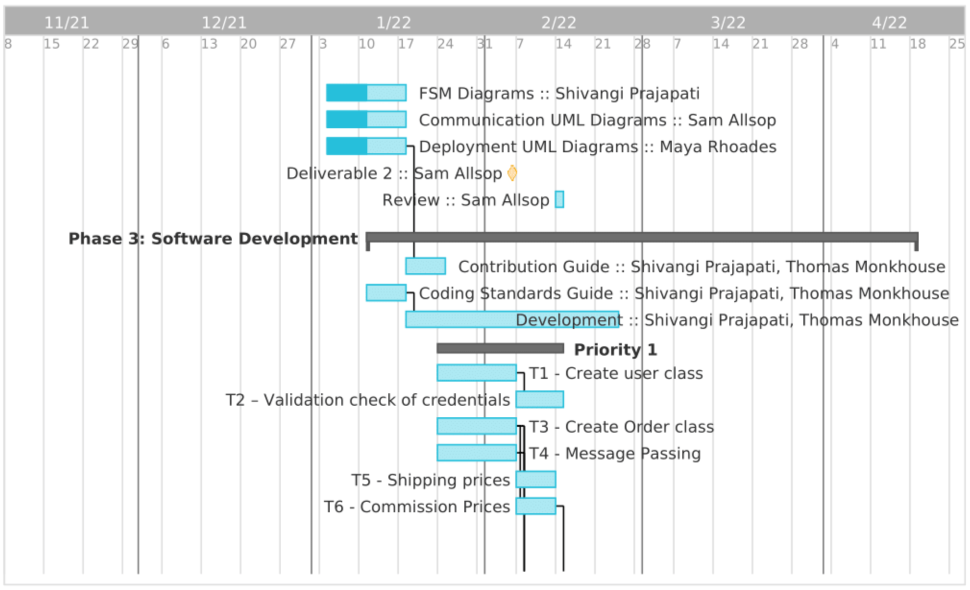


Figure 7: Gantt Chart Enlarged 2

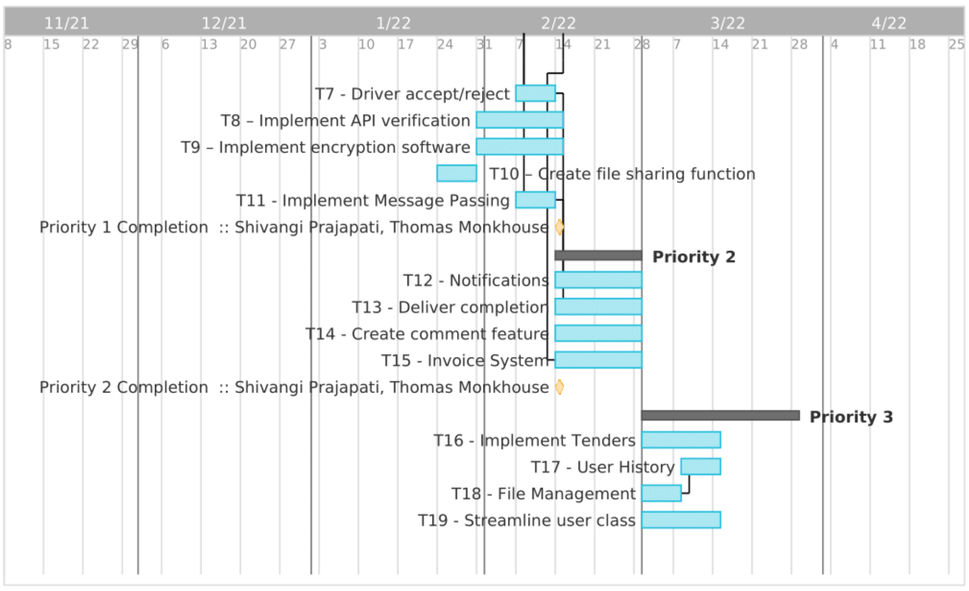


Figure 8: Gantt Chart Enlarged 3

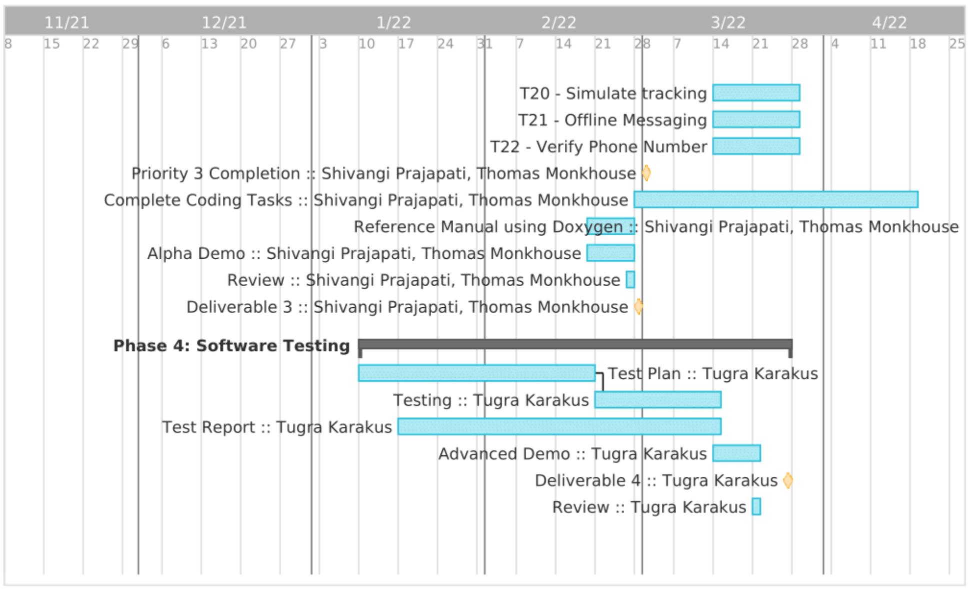


Figure 9: Gantt Chart Enlarged 4

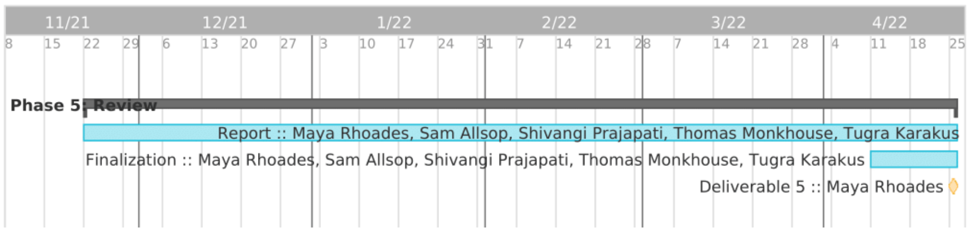


Figure 10: Gantt Chart Enlarged 5

Adopted Coding Standards

1. Naming conventions for local variables, global variables, constants, and functions:
   a. Use of Camel Case and Snake Case for naming based upon which section is being named.

2. Indentation:
    a. Each block of code should be properly indented
    b. All equals, commas, and other symbols should be followed by a space.
    c. Opening brackets should stay on the same line as their statement with no white space between the bracket and the statement.
    d. Two tabs worth of space between any code and a comment that are on the same line.
3. Functions
    a. Length of Functions should not exceed 100 lines to make functions concise. If they exceed, they should be simplified with smaller functions or the use of inheritance.
    b. Any repeated sections of code that for multiple lines shall be turned into their own function.
4. Commenting
    a. Each block of code shall have a comment describing its use above it on its own line.
5. Scope Declaration
    a. Do not declare scope such as namespace std.
6. Naming Files
    a. .h and .cpp should be named similarly with no unnecessary repetition.
    b. .h and .cpp names shall not include digits, spaces, or symbols.
    c. Avoid the use of sources file names
7. Git Commits
    a. Commits should be on a branch first to be reviewed before committing on main branch.
    b. Commit message shall be a straightforward list just listing any additions or changes.

Implementation

The requirements are for designing and implementing an e-transport marketplace application. This report will detail through many diagrams the makeup of how this system will work and its implementation.

The project is to create a virtual platform for road transport that connects cargo owners, drivers, and shipping companies from all over the country. It aims to have a direct impact in reducing costs thus increasing revenue while also having the ability to track the cargo online.

Diagrams

Use Case Diagram

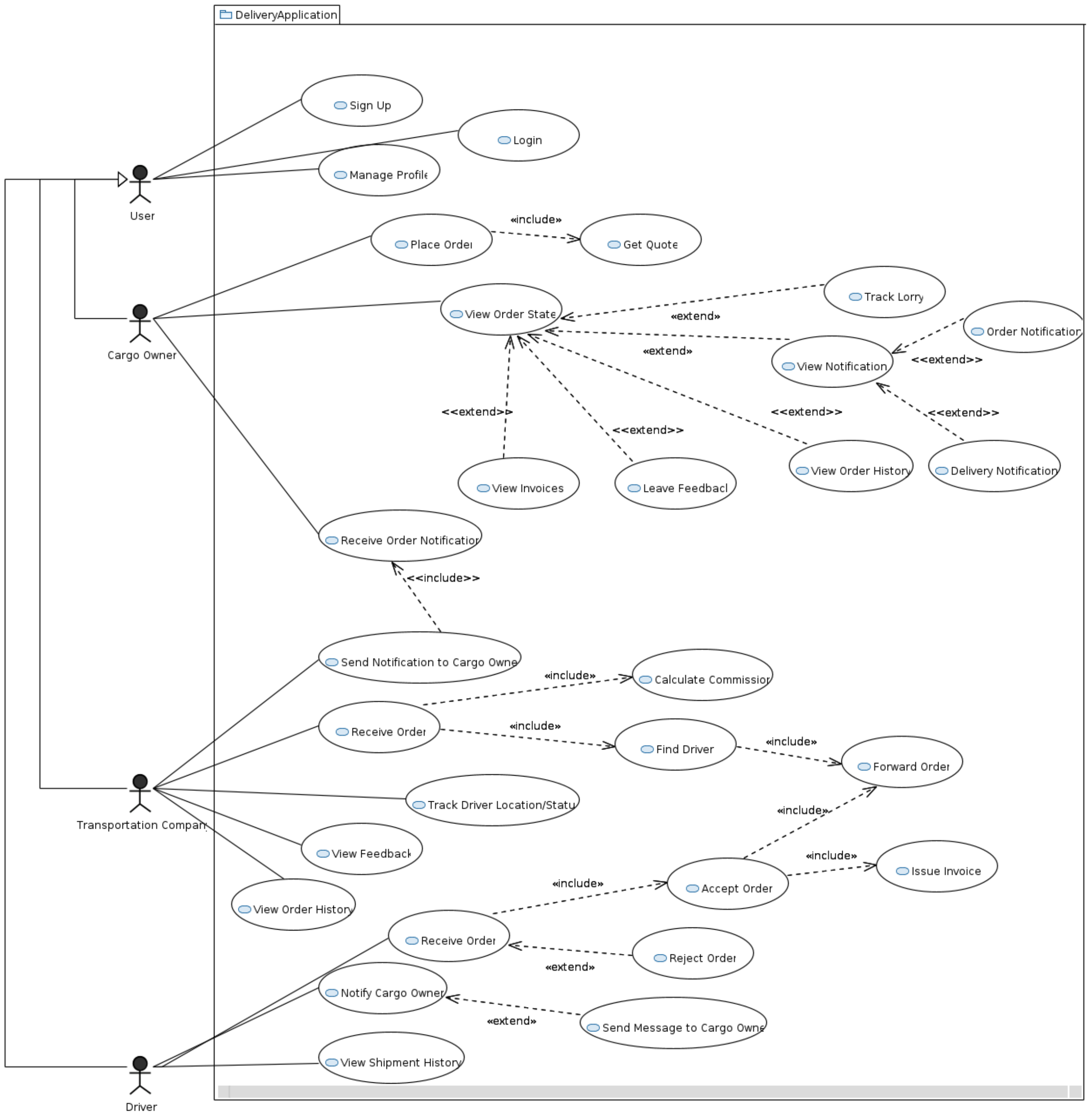


Figure 11: E-Transport Marketplace Application Use Case Diagram

The use case diagram displays the many different users involved as well as their specific functions within the system. The system would involve three types of users, the Cargo Owner, Transportation Company, and the Driver. All users have access to basic universal cases of ‘Sign Up’, ‘Login’, and ‘Manage Profile’. Other use cases are specific to their subclass such as the Cargo Owner being able to make an order, the Transportation Company receiving said order and forwarding to a Driver. A lot of use cases that get shared concerning the order are to be managed within an order class. This will make it easier to organise order data, update notifications and state, as well as sectioning off data so only the relevant information is shown.

Activity Diagram

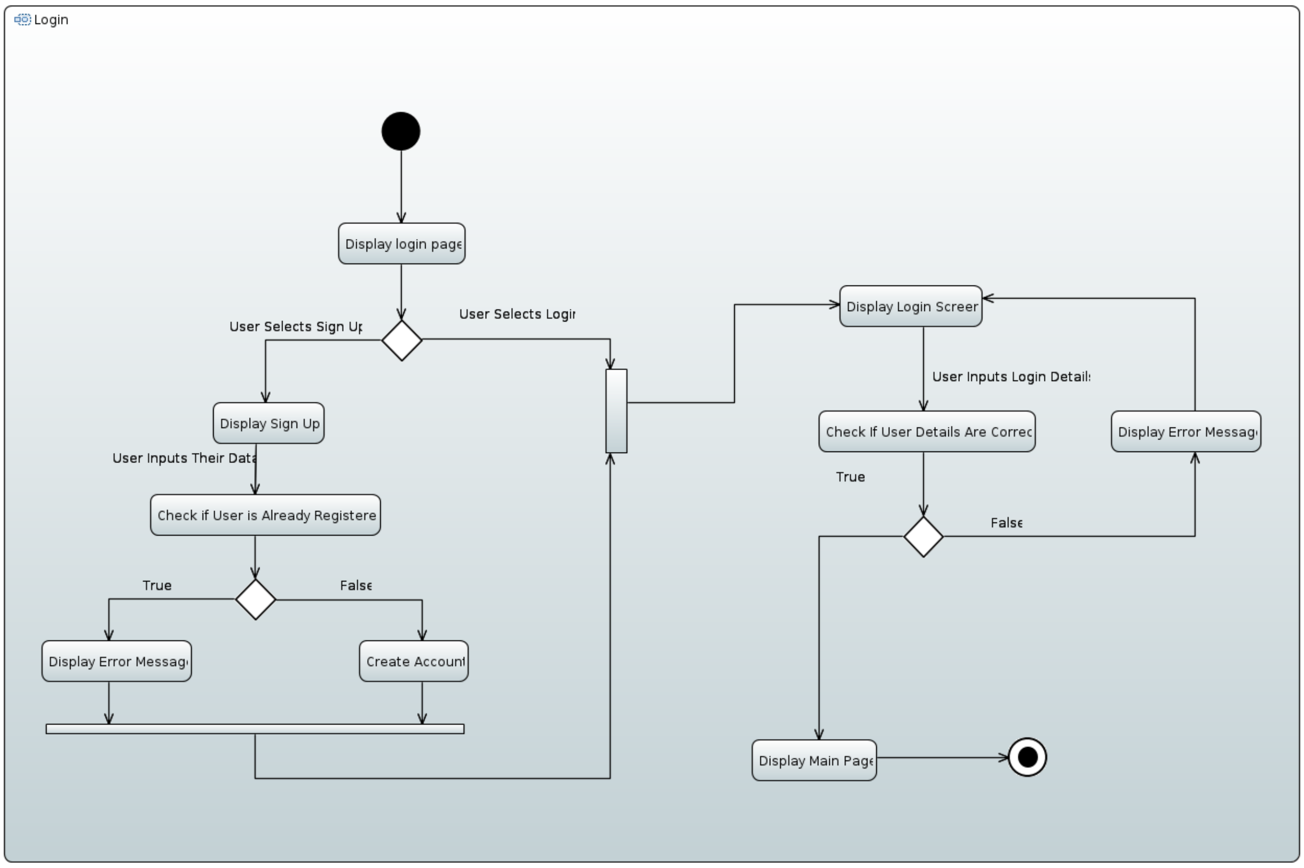


Figure 12: Login Activity Diagram

The first activity diagram showcases the login system. Initially, it will follow through with the sign-up/ login processes. During sign-up, users would input their data which then gets compared in the database for an existing/duplicate user to create their account. After, it returns to the main login screen before being able to access the main pages. The main page is dependent on the user type; a cargo owner will have a different main page than a transport company for example.

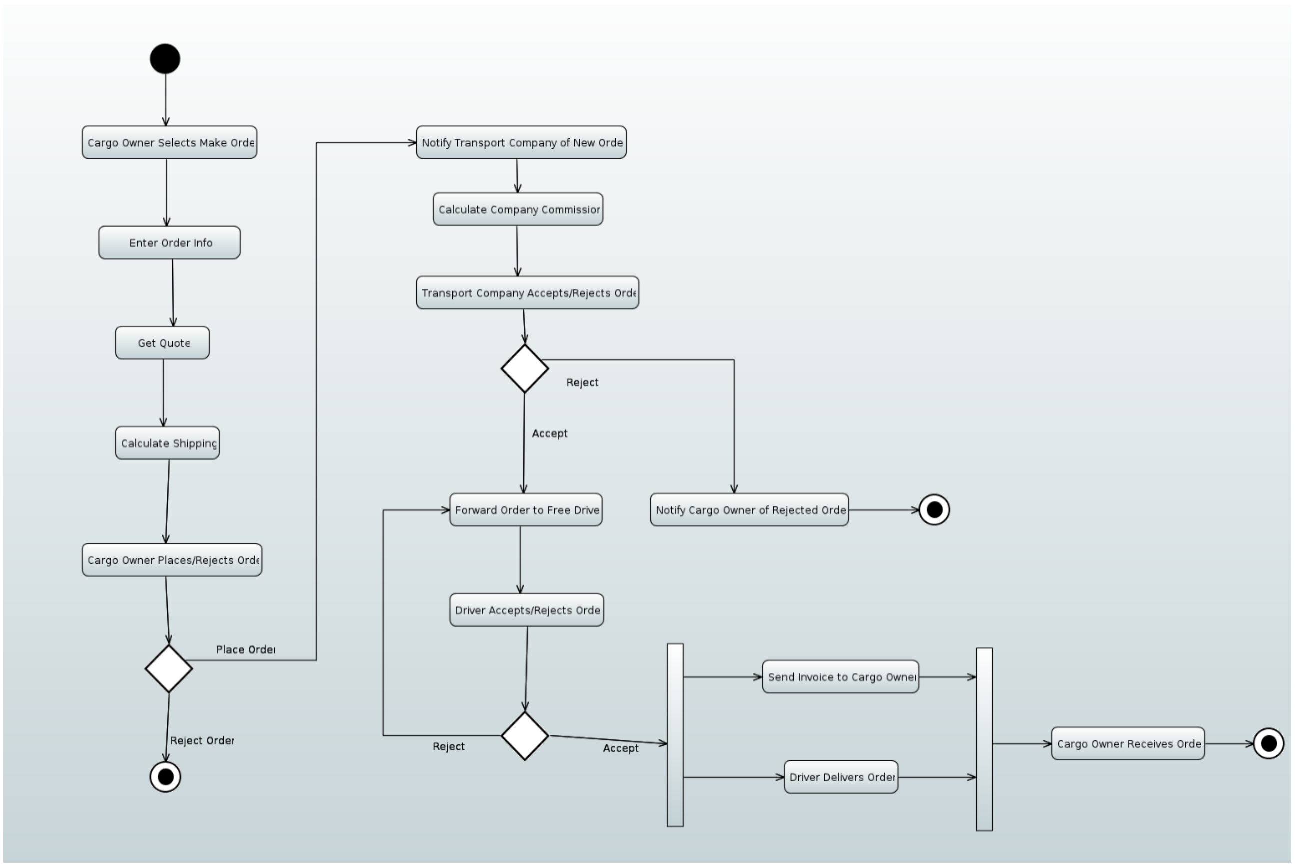

Cargo Owner Selects Make Orde
Enter Order Info
Get Quote
Calculate Shipping
Cargo Owner Places/Rejects Orde
Place Order
Reject Order
Notify Transport Company of New Orde
Calculate Company Commission
Transport Company Accepts/Rejects Orde
Reject
Accept
Forward Order to Free Drive
Notify Cargo Owner of Rejected Orde
Driver Accepts/Rejects Orde
Reject
Accept
Send Invoice to Cargo Owner
Driver Delivers Order
Cargo Owner Receives Orde

Figure 13: Order Activity Diagram

This activity diagram shows the place order process. When an order is to be placed, the cargo owner enters the details for the order. After, the cargo owner receives a quote. Shipping rates are then calculated in which the Cargo Owner can either cancel their order or follow through and notify the transport company. The transport company will receive their calculated commission and can then choose to either accept/reject the order. If accepted, a driver would need to be found. The driver themselves can also reject an order therefore it will loop so if a driver rejects, a new driver can be found. Once accepted, the cargo owner is notified of the accepted order.

Class Diagram

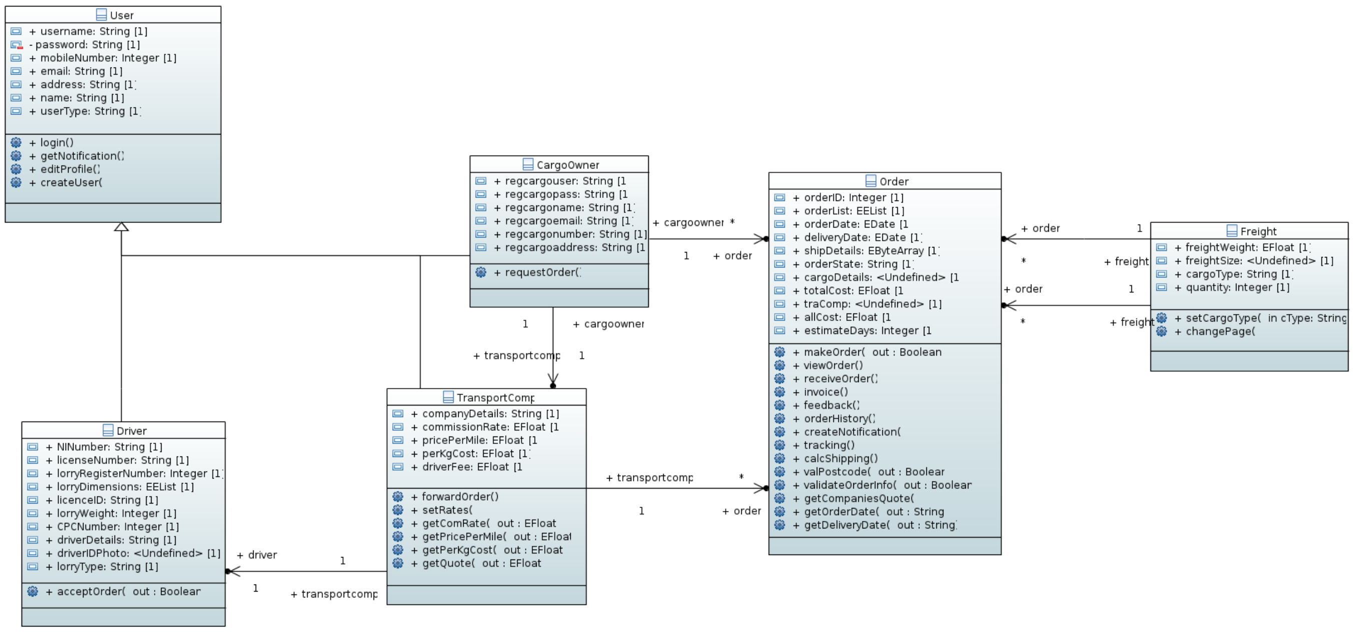

User
+ username: String [1]
- password: String [1]
+ mobileNumber: Integer [1]
+ email: String [1]
+ address: String [1]
+ name: String [1]
+ userType: String [1]
+ login()
+ getNotification()
+ editProfile()
+ createUser()
CargoOwner
+ regcargouser: String [1]
+ regcargopass: String [1]
+ regcargoname: String [1]
+ regcargoemail: String [1]
+ regcargonumber: String [1]
+ regcargoaddress: String [1]
+ requestOrder()
Order
+ orderID: Integer [1]
+ orderList: EEList [1]
+ orderDate: EDate [1]
+ deliveryDate: EDate [1]
+ shipDetails: EByteArray [1]
+ orderState: String [1]
+ cargoDetails: <Undefined> [1]
+ totalCost: EFloat [1]
+ traComp: <Undefined> [1]
+ allCost: EFloat [1]
+ estimateDays: Integer [1]
+ makeOrder(out : Boolean
+ viewOrder()
+ receiveOrder()
+ invoice()
+ feedback()
+ orderHistory()
+ createNotification(
+ tracking()
+ calcShipping()
+ valPostcode(out : Boolean
+ validateOrderInfo(out : Boolean
+ getCompaniesQuote(
+ getOrderDate(out : String
+ getDeliveryDate(out : String
Freight
+ freightWeight: EFloat [1]
+ freightSize: <Undefined> [1]
+ cargoType: String [1]
+ quantity: Integer [1]
+ setCargoType(in cType: String
+ changePage(
TransportComp
+ companyDetails: String [1]
+ commissionRate: EFloat [1]
+ pricePerMile: EFloat [1]
+ perKgCost: EFloat [1]
+ driverFee: EFloat [1]
+ forwardOrder()
+ setRates(
+ getComRate(out : EFloat
+ getPricePerMile(out : EFloat
+ getPerKgCost(out : EFloat
+ getQuote(out : EFloat
Driver
+ NINumber: String [1]
+ licenseNumber: String [1]
+ lorryRegisterNumber: Integer [1]
+ lorryDimensions: EEList [1]
+ licenceID: String [1]
+ lorryWeight: Integer [1]
+ CPCNumber: Integer [1]
+ driverDetails: String [1]
+ driverIDPhoto: <Undefined> [1]
+ lorryType: String [1]
+ acceptOrder(out : Boolean
+ cargoowner *
1 + order
+ order 1
* + freight
+ order 1
* + freight
1 + cargoowner
+ transportcomp 1
+ transportcomp *
1 + order
+ driver 1
1 + transportcomp

Figure 14: Class Diagram

The class diagram is to showcase the different objects that make up the system, their properties and functions, and base communications between them. This class diagram demonstrates the attributes of all users, both shared and independent. This with attributes and functions like ‘username’, ‘password’, login (), and editProfile() being shared across all users. Meanwhile more specific attributes/functions such as companyDetails or calculateCommission () are specified to that user. The class ‘Order’ contains a lot of the functionality of the system and so most other classes are related to this class. Users will need to access the order for different functions and to see the relevant data. The freight class exists as the original order placement details centring around size, quantity, and type.

Sequence Diagram

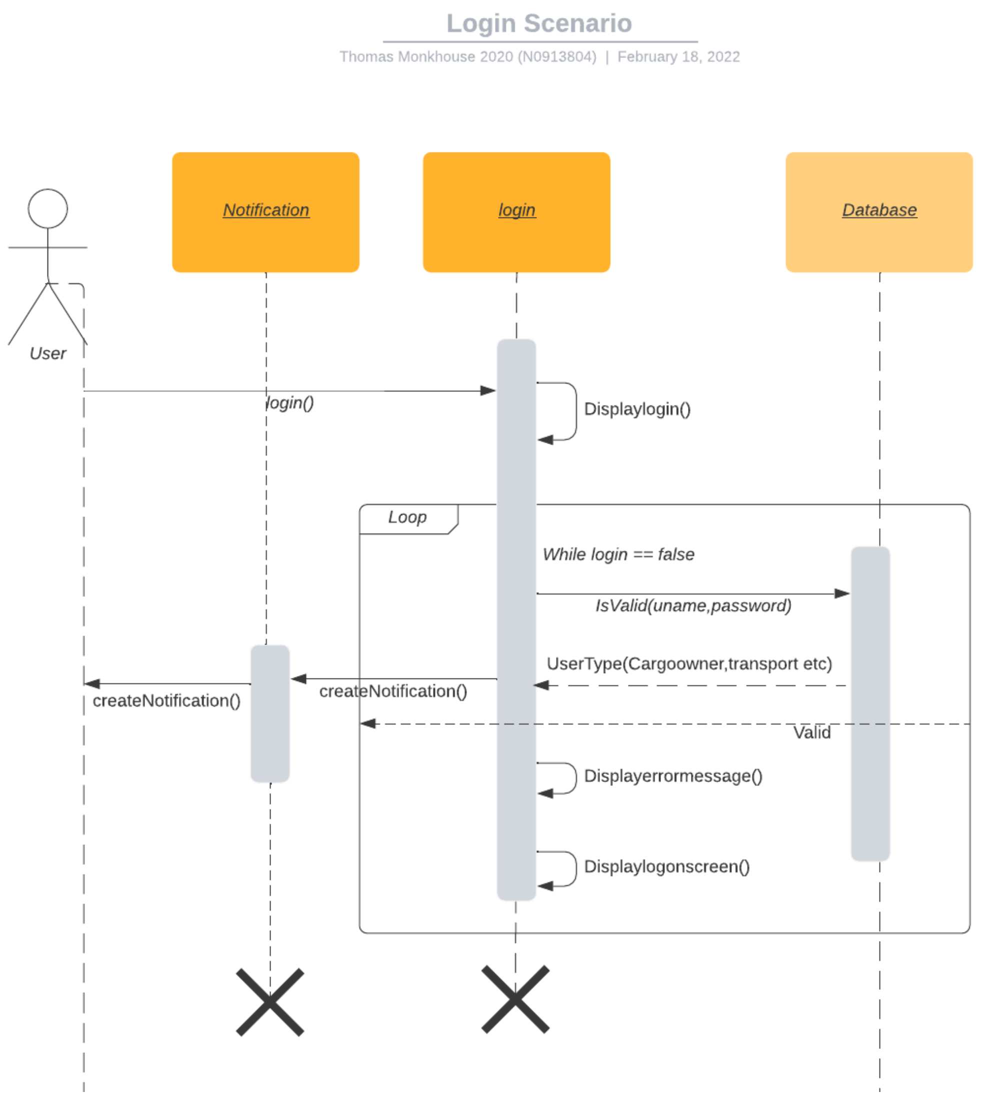


Figure 15: Logon Sequence Diagram

This sequence diagram showcases the general flow of the login system. Upon login request from the user, the UI updates ready for valid a valid username and password which is then found in the database and finally retrieves the userType. For an invalid entry, an error message is displayed instead before returning to the logon screen.

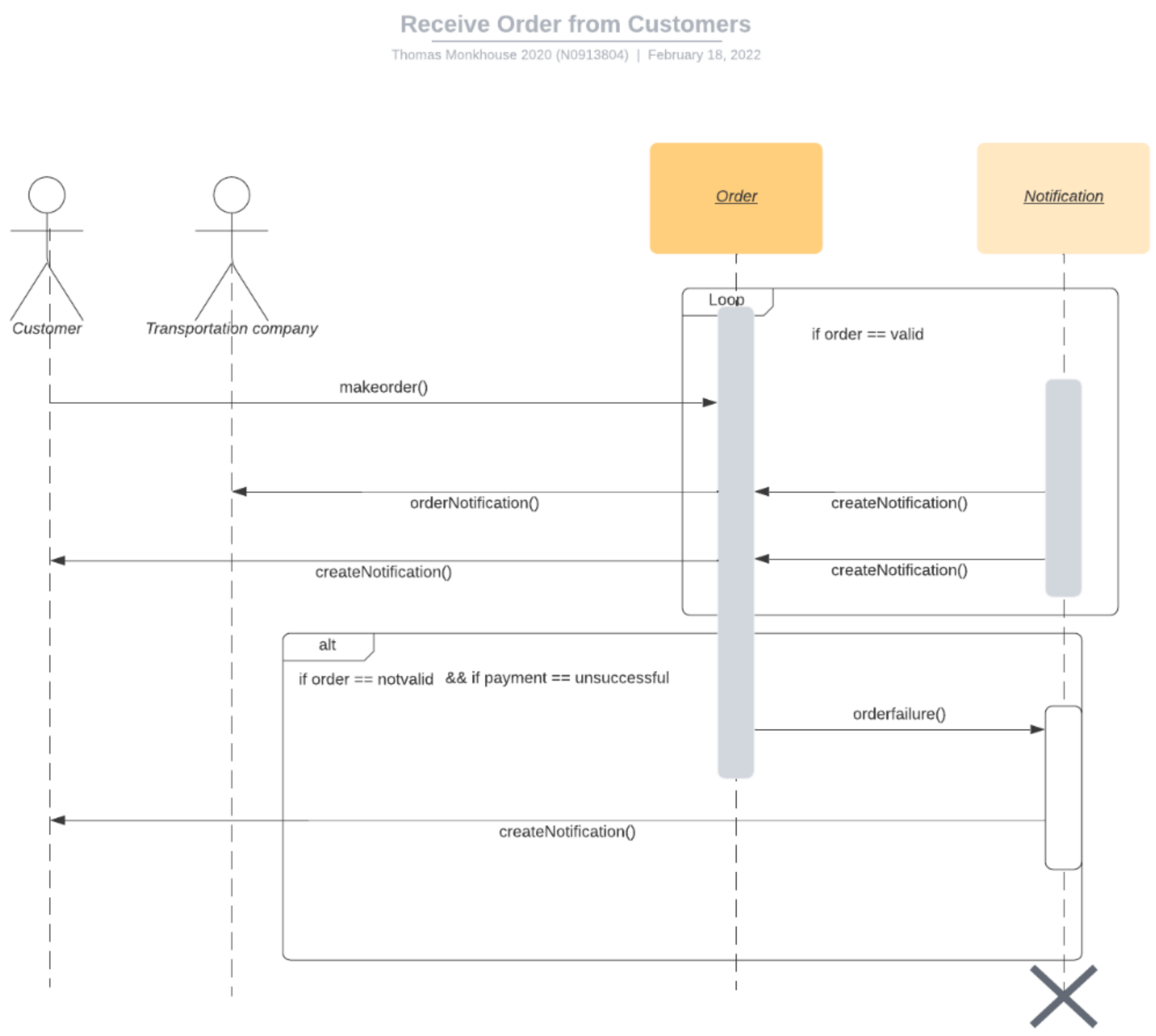


Figure 16: Order Placed Sequence Diagram

The above diagram describes the order placement. When a cargo owner places the order, they receive a notification confirming the order. Also, another notification is sent to the transport company signalling a new order has been made. Upon an invalid order, the cargo owner will instead receive a failed order notification.

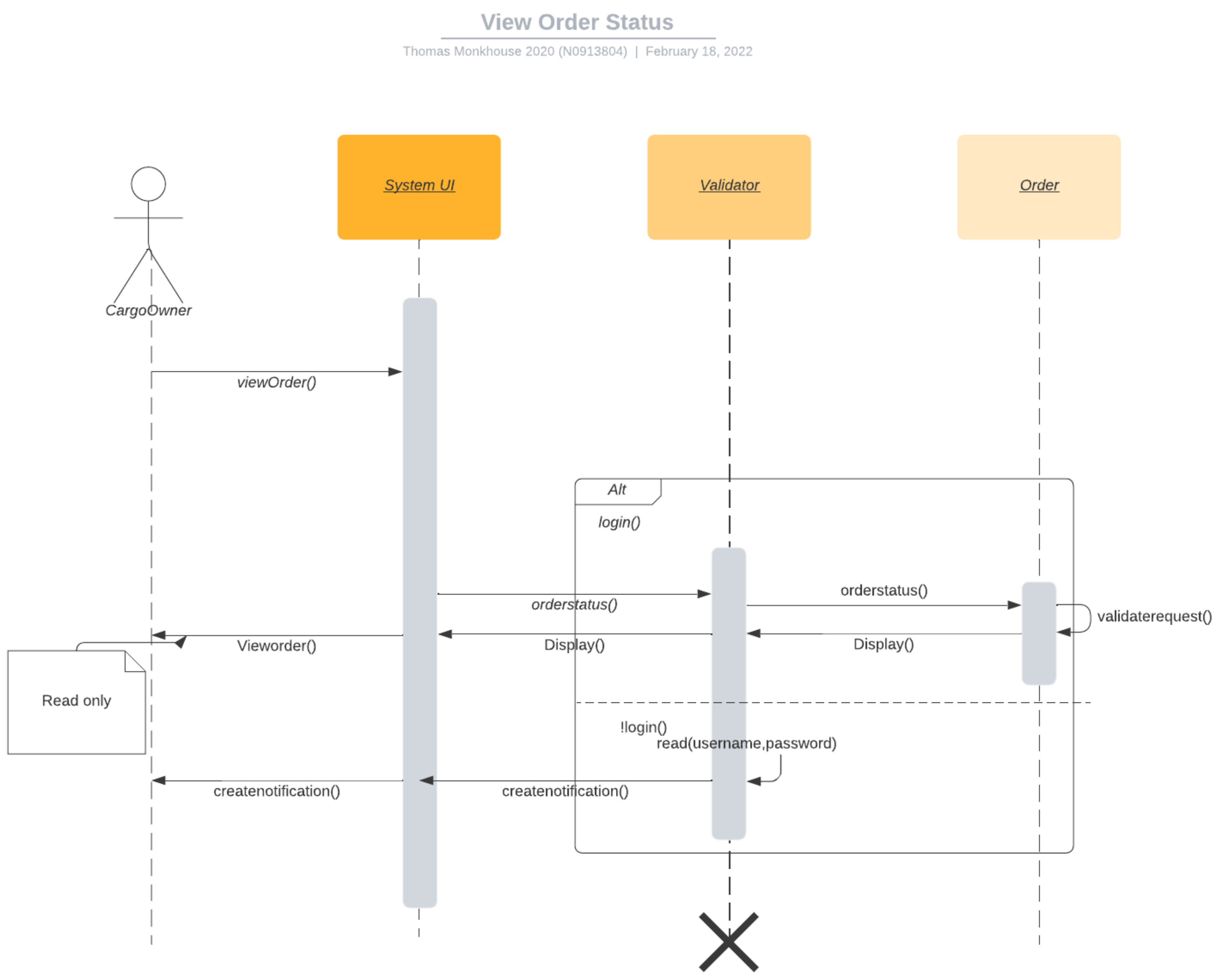


Figure 17: Order Status Sequence Diagram

The above figure describes the view order status for the cargo owner. After the order status has been called, the account is verified before validating the request which is then returned and displayed for the user as read only. Else, an error is instead displayed.

Component Diagram

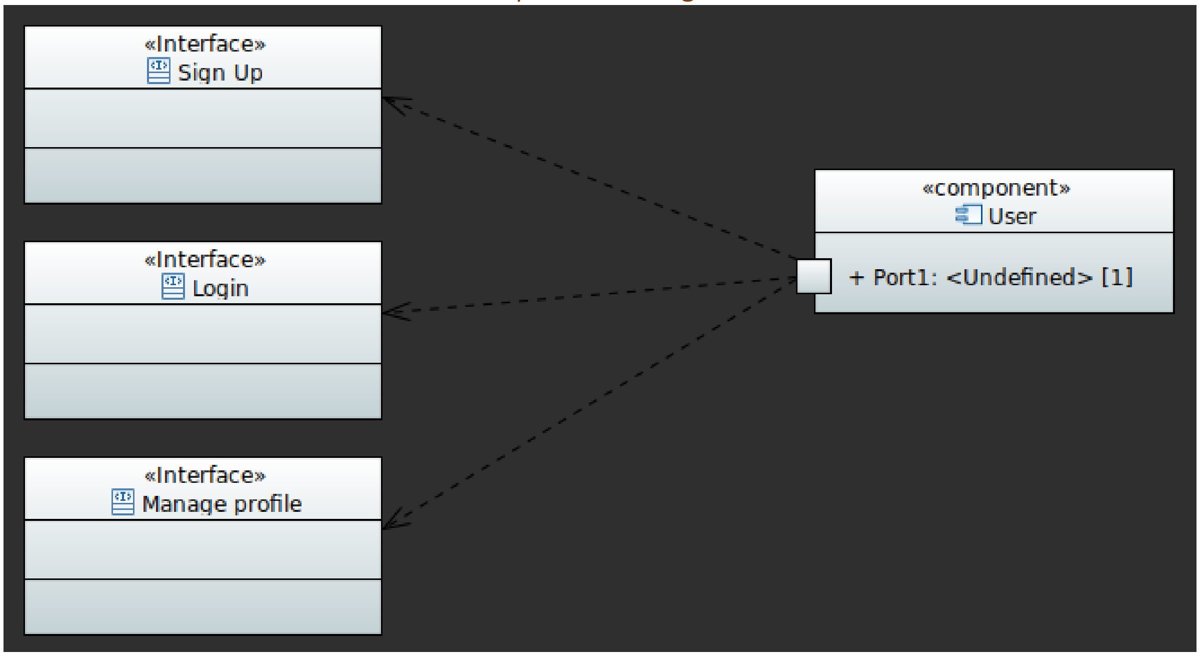


Figure 18: User Component Diagram

This diagram displays the functionality options of all Users allowing base methods of login, sign up, and profile management.

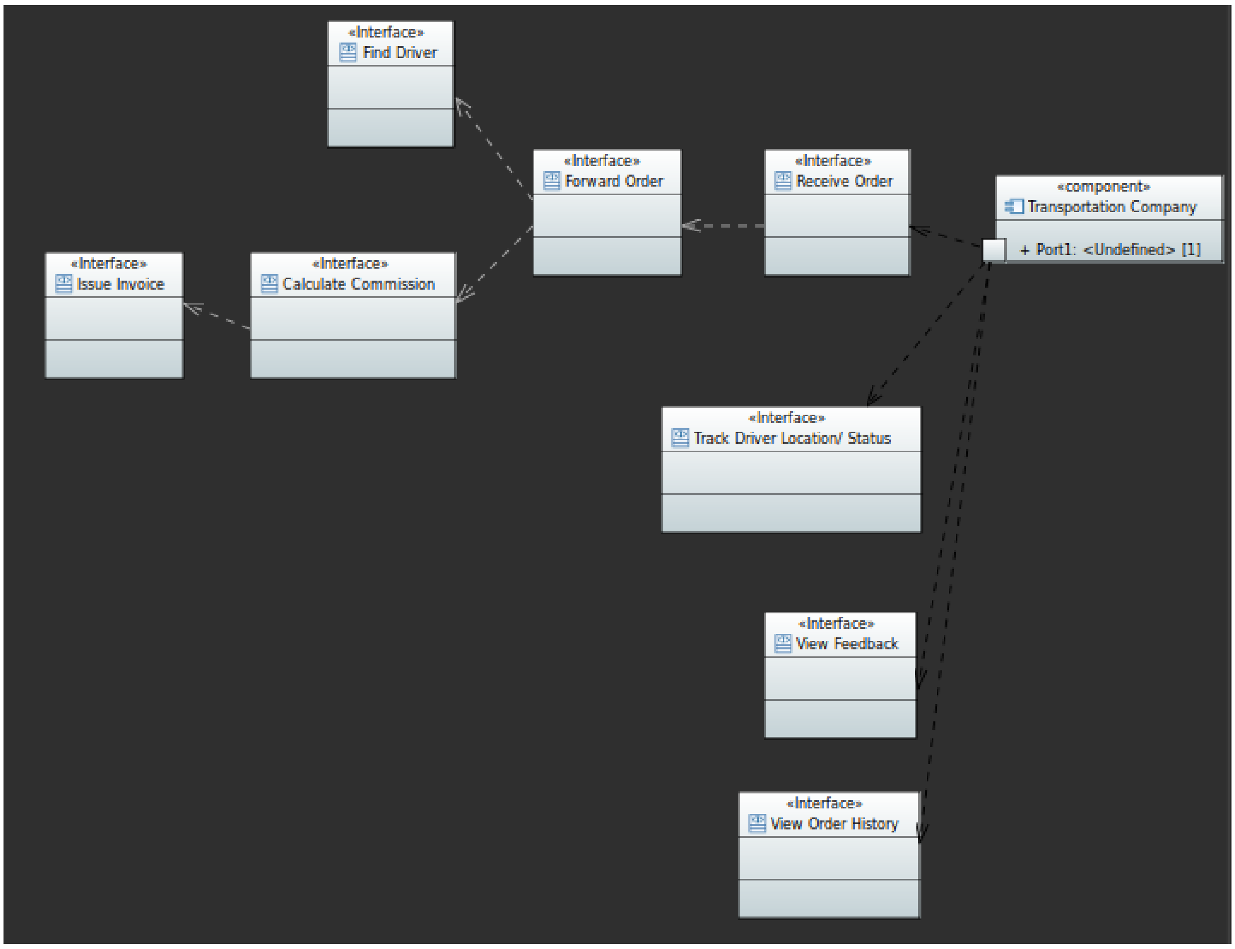


Figure 19: Transportation Company Component diagram

The diagram displays the functionality and order of the transportation company user. Some baseline functions such as view order history and feedback are recall methods. The order methods are related and proceeded one after another while also splitting to find driver.

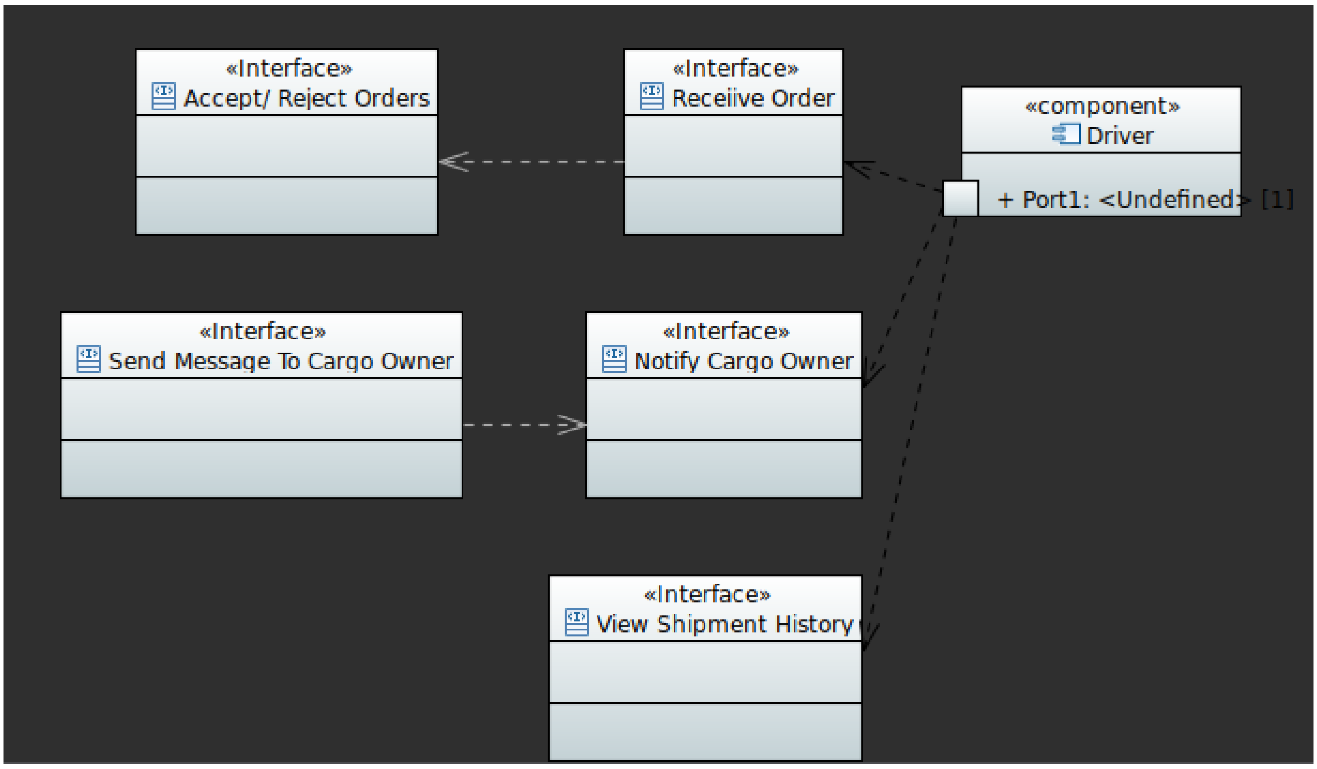


Figure 20: Driver Component Diagram

This diagram showcases the driver user. The driver can view shipment history, receive order to then accept/reject it, and can notify the cargo owner with an optional message.

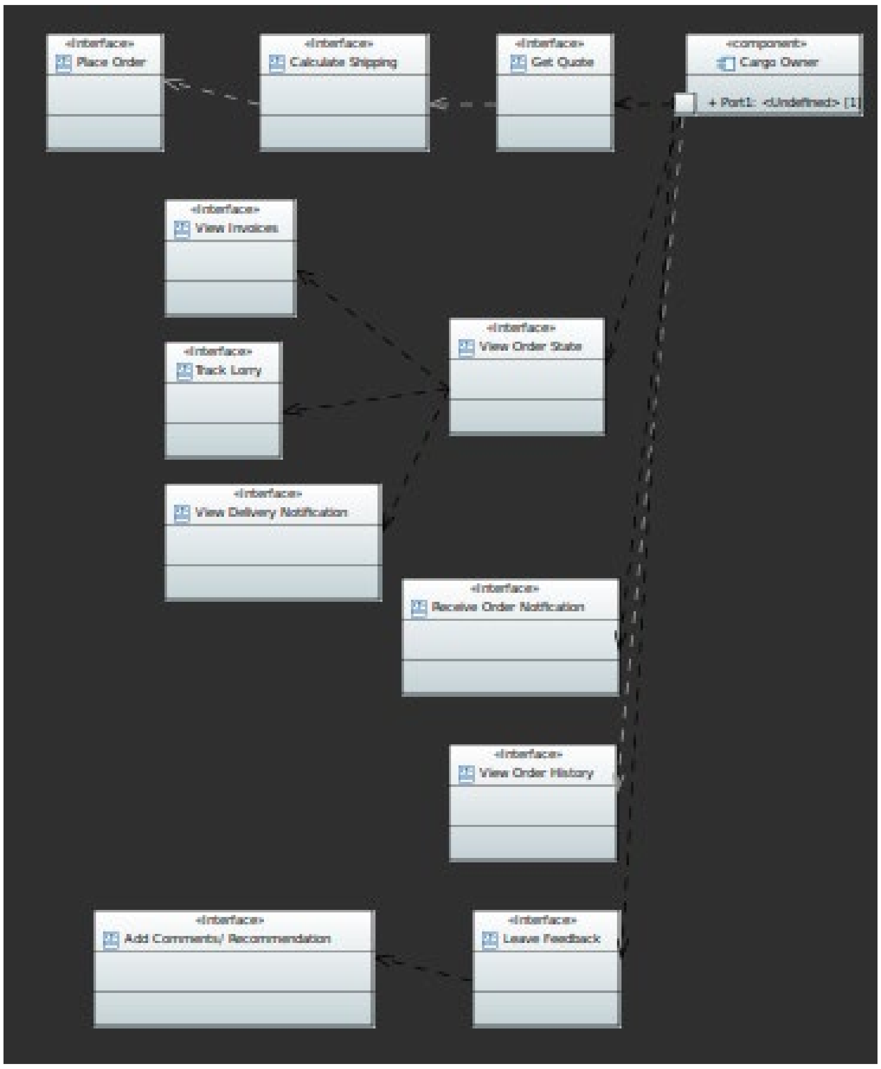

«interface» Place Order
«interface» Calculate Shipping
«interface» Get Quote
«component» Cargo Owner
+ Port1: <Undefined> [1]
«interface» View Invoices
«interface» View Order State
«interface» Track Lorry
«interface» View Delivery Notification
«interface» Receive Order Notification
«interface» View Order History
«interface» Add Comments/ Recommendation
«interface» Leave Feedback

Figure 21: Cargo Owner Component Diagram

The diagram shows the cargo owner user. The cargo owner can place an order via getting quote and calculating shipping. Within the view order state, the user can access a few options such as viewing invoices and lorry tracking. The cargo owner can also receive notifications and view history as well as leave feedback with additions to add comments/recommendation.

FSM Diagram

The Finite State Machine or FSM diagram aims to illustrate the states of a system. Moreover, it also displays how the system responds to events by passing from one state to another. The diagram below represents the states of the e-transport system and transitions causing the change of these.

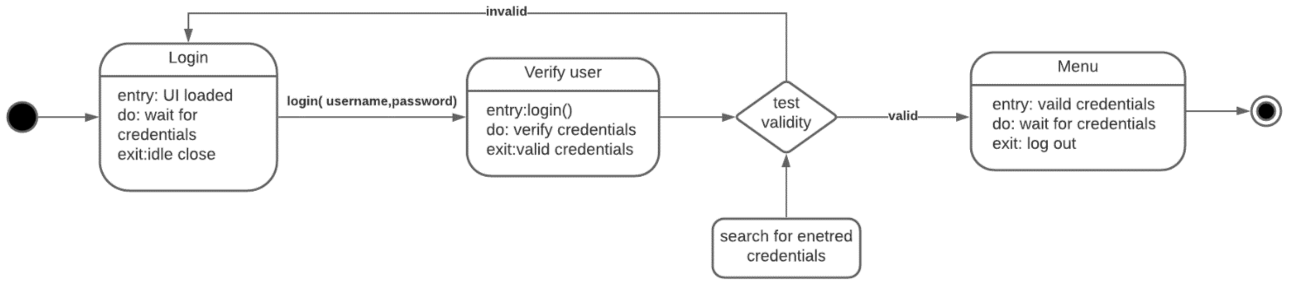


Figure 22: Login FSM Diagram

The diagram above describes the change in states in the login process. The first state is "Login" where the user's credentials are requested. The transaction to next state takes place through the login() method. The login credentials can be either valid or invalid and based on the verification the system can either display the main menu or take back the user to login page.

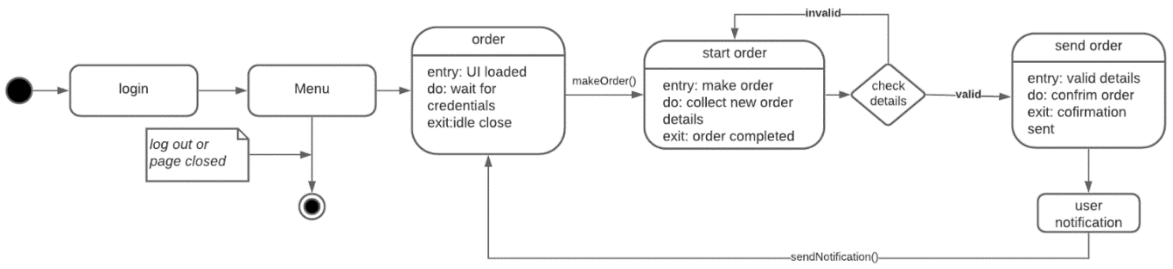


Figure 23: Order FSM Diagram

The image above is the representation of the states of the system in the process of placing an order. These states are most likely to happen when the user is the cargo owner. Assuming the user has provided valid credentials and wants to place a new order. The state "order" is changed to "start order" by the makeOrder() method. If the provided details are valid, the order is placed and notification is sent, if not, the user is taken back to the previous page.

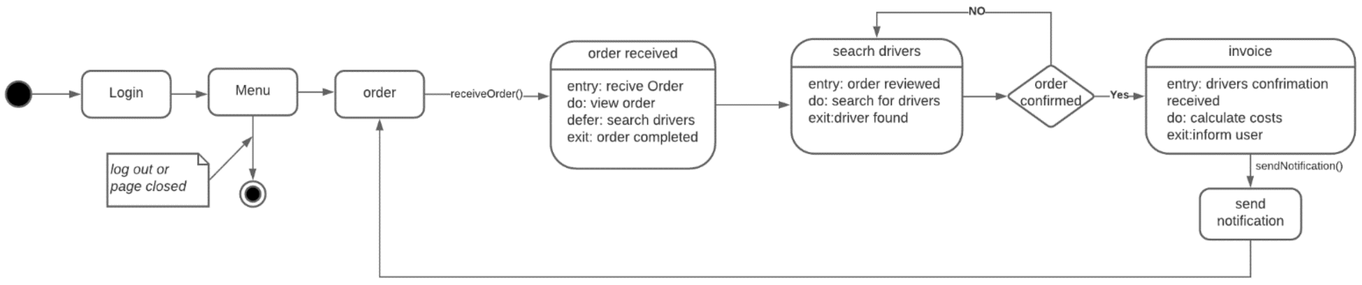


Figure 24: Transport Company FSM Diagram

The transport company is one the users of the system and it mainly receives orders from cargo owners. Assuming, the company has been successful in logging in to the system, the first state will be order and the transition will take place when a new order is received. This will start the process of searching for available drivers. Once this is confirmed, the system will produce an invoice with all the costs and notify the customer.

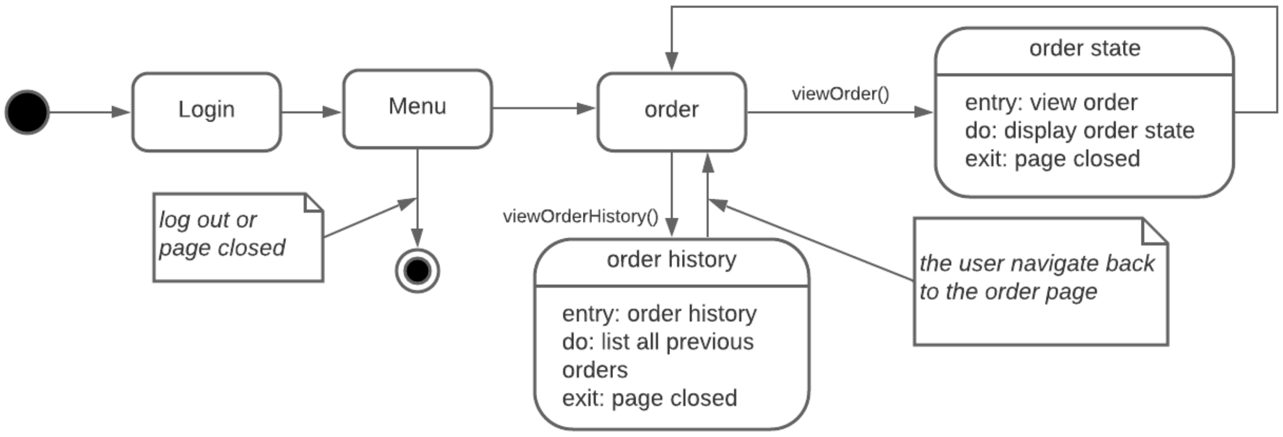


Figure 25: Order State/History FSM Diagram

The diagram represents the other possible states of the system. The order can be paced as well as tracked and this is one of the states of the system. Similarly, the user can view previously place orders.

Communication Diagram

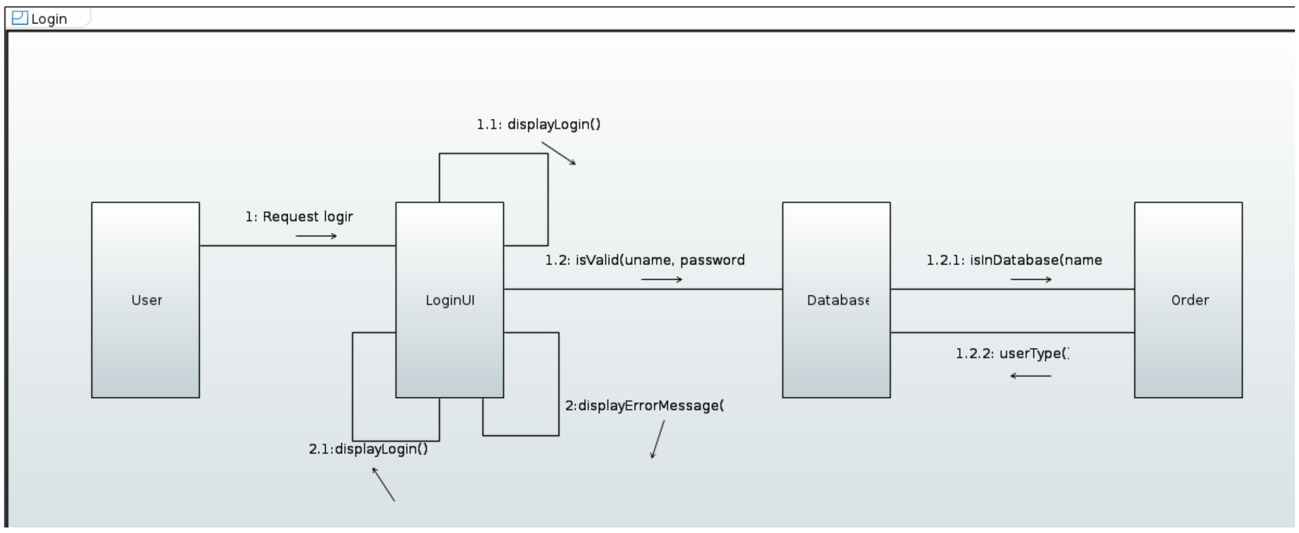


Figure 26: Login Communication Diagram

The above communication diagram describes the verification of the login system. Upon request of login, the UI will display ready for input. Upon incorrect input, an error message will be displayed before returning to the login page. Otherwise, it's checked through the database for an existing user, then the userType is fetched.

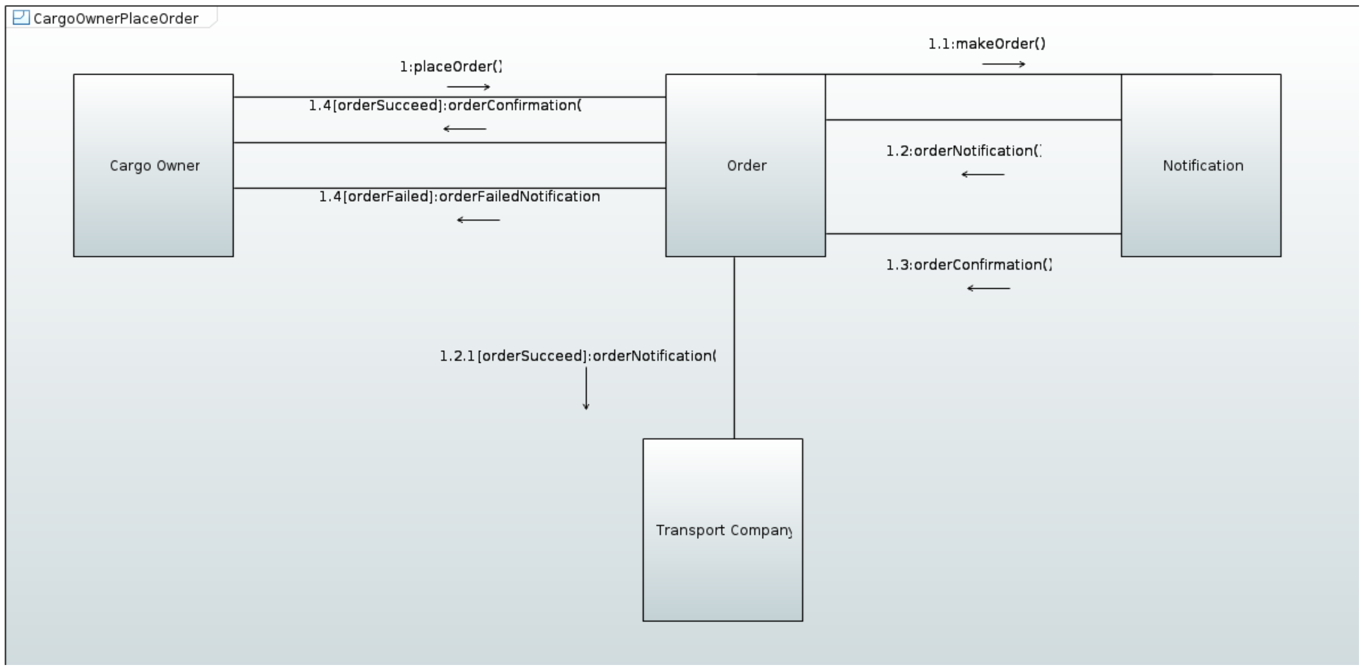


Figure 27: Place Order Communication Diagram

The place order communication diagram shows the relations between Cargo Owner and the Transport Company upon making an order. After placing an order, a notification is created for both the Transport Company and the Cargo Owner confirming a successful order, otherwise, a notification is sent to the Cargo Owner signifying that the order has failed.

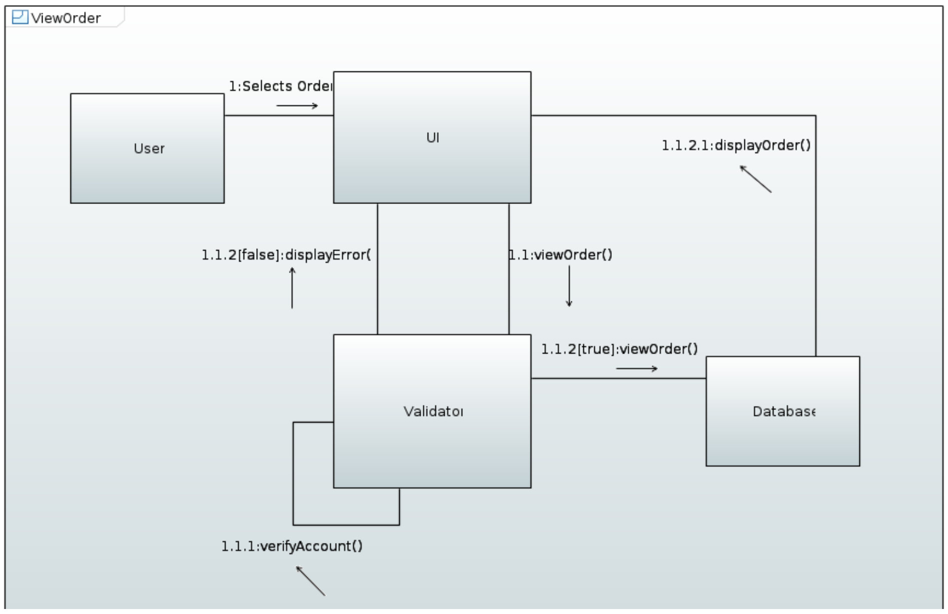


Figure 28: View Order Communication Diagram

The view order communication diagram describes said process of the user selecting their order. Initially, it gets verified and would display an error if false, else, it fetches the order data from the database and is then displayed via the UI.

Deployment Diagram

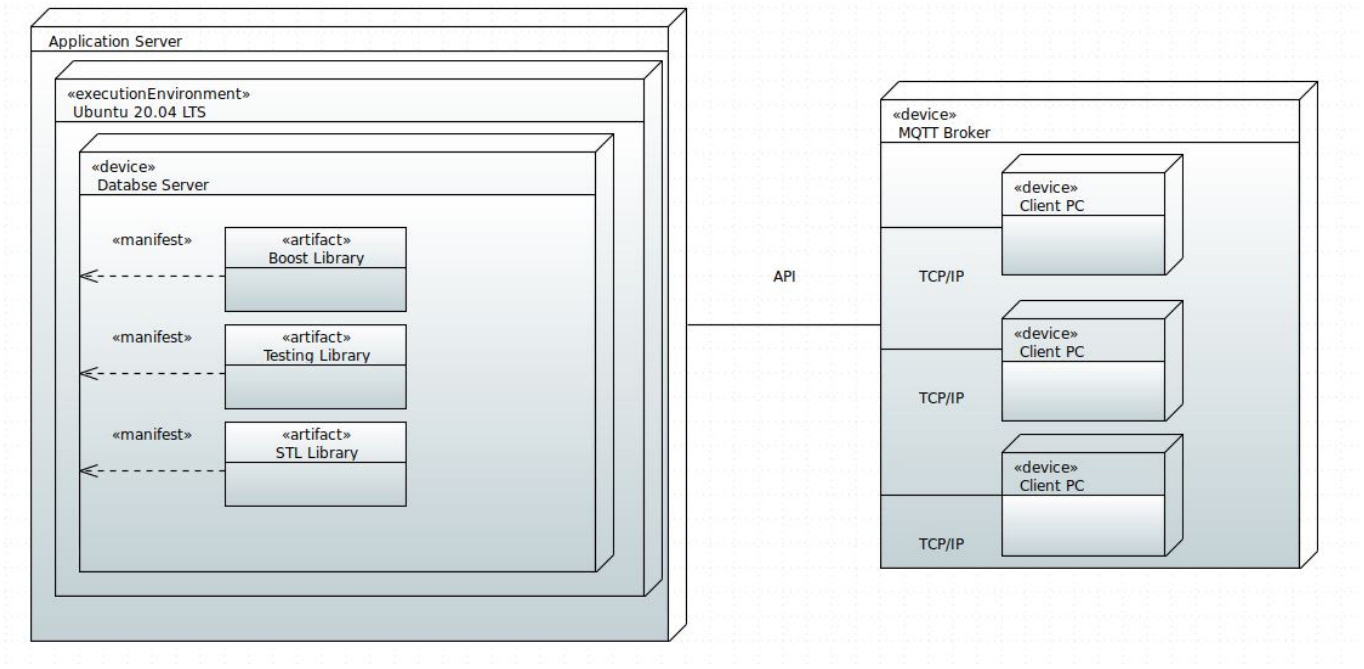
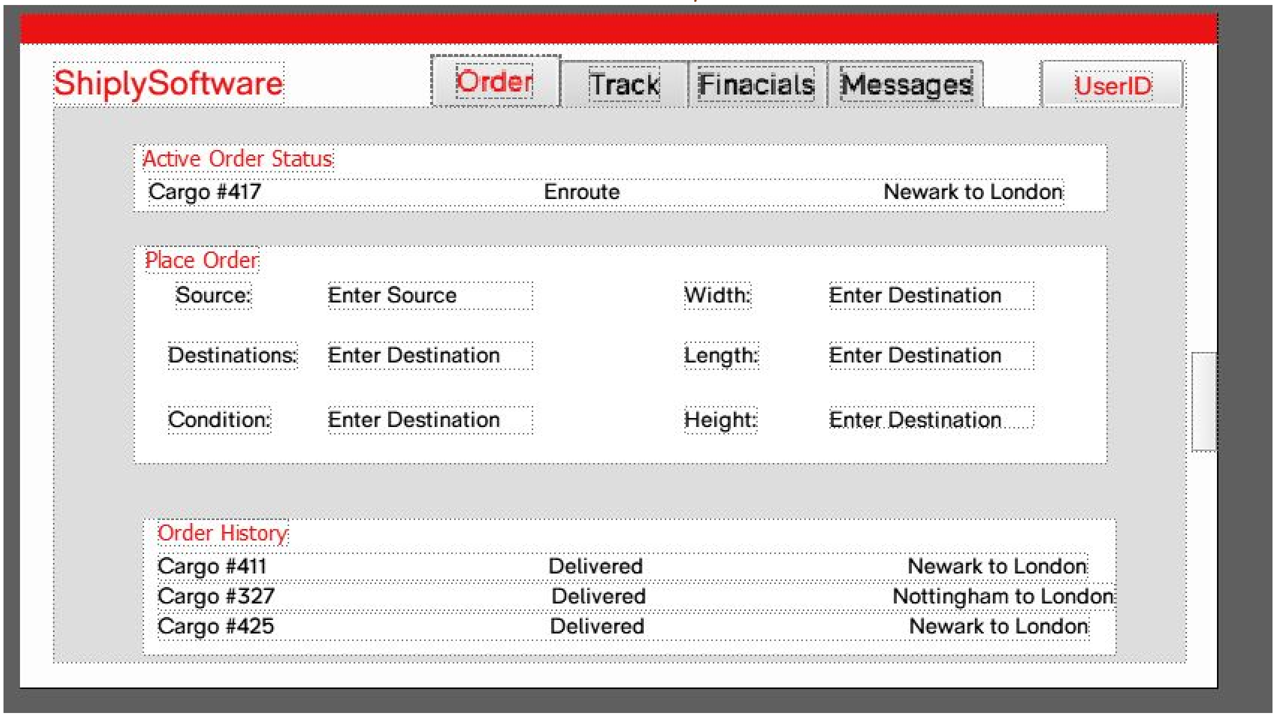


Figure 29: Deployment Diagram

The Deployment Diagram shows the architecture of a system, its hardware and software. The application is stored on a server. Within the server is the executable environment Ubuntu. The Database server stores all the libraries and source code for the application. When the client wants to access the application, they do so through the mqtt broker. The mqtt broker serves as encryption for the user. The communication from the client broker to the application is done via API. Data is therefore sent from the client through the mqtt broker, hits the server through the executable ubuntu, checks the libraries and executes code, then sends data back to the client.

GUI

Mock Up



Active Order Status

Cargo #417	Enroute	Newark to London

Place Order

Source:	Enter Source	Width:	Enter Destination
Destinations:	Enter Destination	Length:	Enter Destination
Condition:	Enter Destination	Height:	Enter Destination

Order History

Cargo #411	Delivered	Newark to London
Cargo #327	Delivered	Nottingham to London
Cargo #425	Delivered	Newark to London

Figure 30: GUI Original Mock-up

This is a Mock-up interface for the application. It was created in QT Creator. The interface features several tabs all with their own information. The order tab that is shown in the figure above allows the cargo owner to see active orders, place an order, and view order history. This page will change according to what type of user is viewing it. For example, the driver will not see a place order section but instead an accept order section. The track tab will allow users to track orders by a live map and with an indicator of where the cargo is in the delivery process (packing, enroute, delivered). The financials page will contain invoices and the ability for certain users to calculate shipping or commission rates. The message tab is for any notifications a user may receive from delivery notification to available orders for the driver. Finally, the UserID page will allow the user to enter all their information (username, password, phone number, driverID, etc.). Each page will be set up similarly to this one in design and simplicity.

Actual GUI

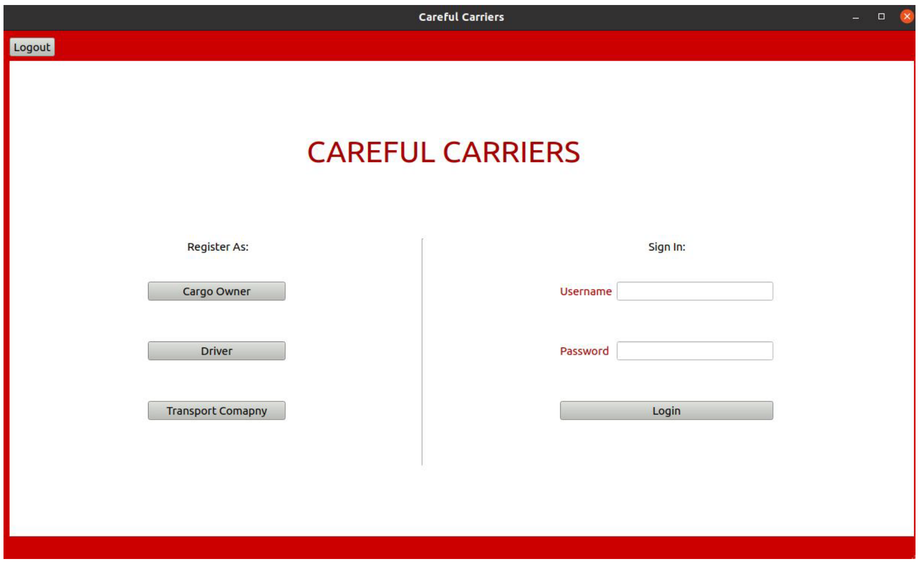
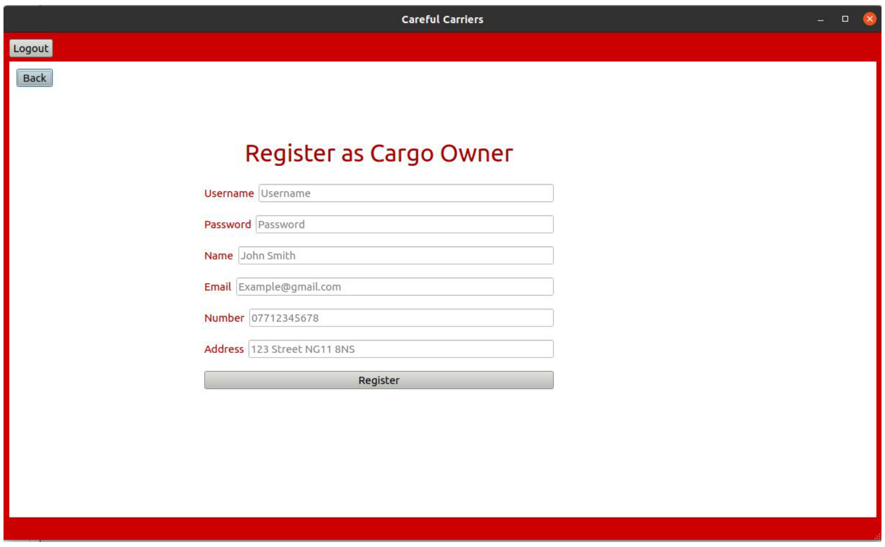


Figure 31: Main Menu

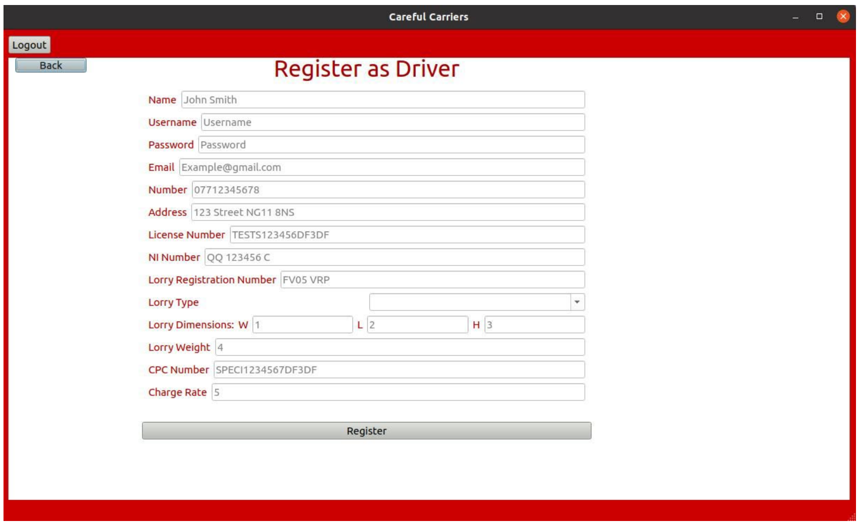
This figure shows the main menu allowing user to either go to the relevant registry page or to login.



Username: Username
Password: Password
Name: John Smith
Email: Example@gmail.com
Number: 07712345678
Address: 123 Street NG11 8NS

Figure 32:Register Cargo Owner Page

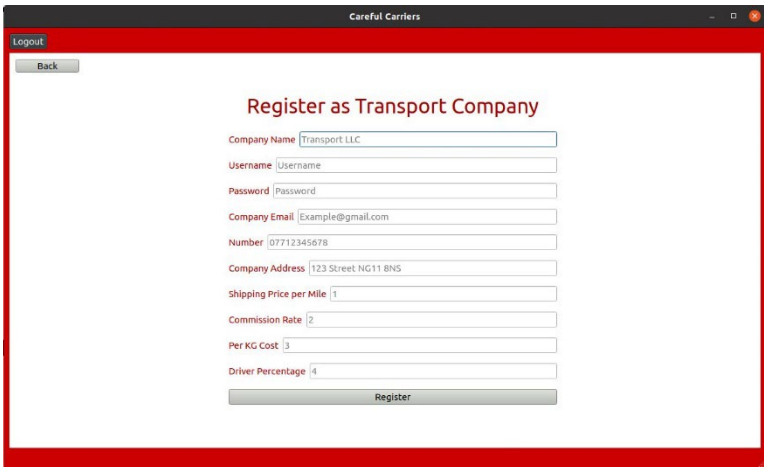
This figure shows the register cargo owner page where the cargo owner can input all their details and validate their number when clicking submit.



Name	John Smith
Username	Username
Password	Password
Email	Example@gmail.com
Number	07712345678
Address	123 Street NG11 8NS
License Number	TESTS123456DF3DF
NI Number	QQ 123456 C
Lorry Registration Number	FV05 VRP
Lorry Type	
Lorry Dimensions: W	1
L	2
H	3
Lorry Weight	4
CPC Number	SPECI1234567DF3DF
Charge Rate	5

Figure 33:Register Driver Page

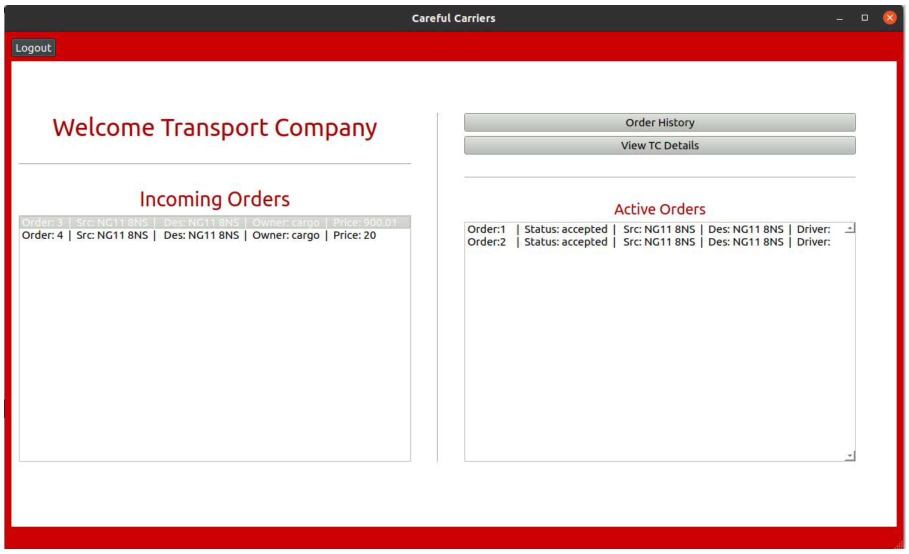
This figure shows the register driver page where the driver can input all their details validate their number, cpc, and license number when clicking submit.



Company Name	Transport LLC
Username	Username
Password	Password
Company Email	Example@gmail.com
Number	07712345678
Company Address	123 Street NG11 8NS
Shipping Price per Mile	1
Commission Rate	2
Per KG Cost	3
Driver Percentage	4

Figure 34: Register Transport Company Page

This figure shows the register transport company page where the transport company can input all their details and validate their number when clicking submit.



Welcome Transport Company

Incoming Orders

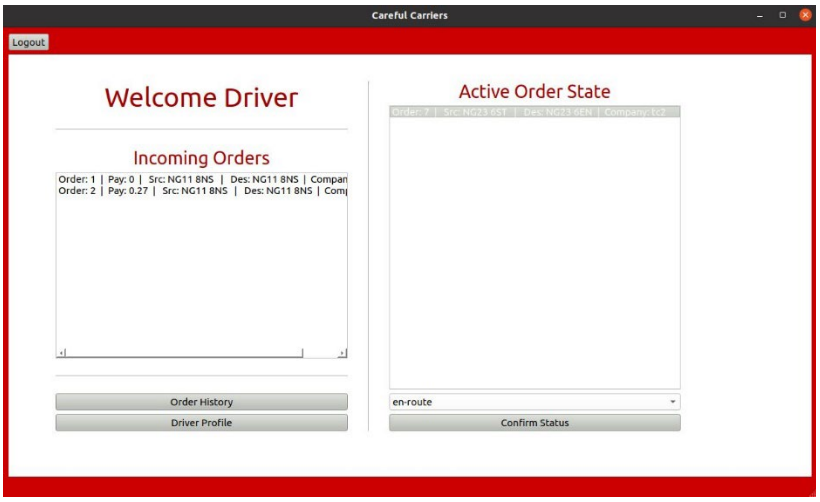
Order: 3 | Src: NG11 8NS | Des: NG11 8NS | Owner: cargo | Price: 900.01
Order: 4 | Src: NG11 8NS | Des: NG11 8NS | Owner: cargo | Price: 20

Active Orders

Order:1 | Status: accepted | Src: NG11 8NS | Des: NG11 8NS | Driver:
Order:2 | Status: accepted | Src: NG11 8NS | Des: NG11 8NS | Driver:

Figure 35: Transport Company Home Page

This figure shows the transport company home page where they can see incoming orders and active orders on a glance. When they click on incoming orders, they are shown all the information for the order and prompted to accept or reject the order. When clicking on an item in Active orders they will see additional information. They can also go to order history or the profile page



Welcome Driver

Incoming Orders

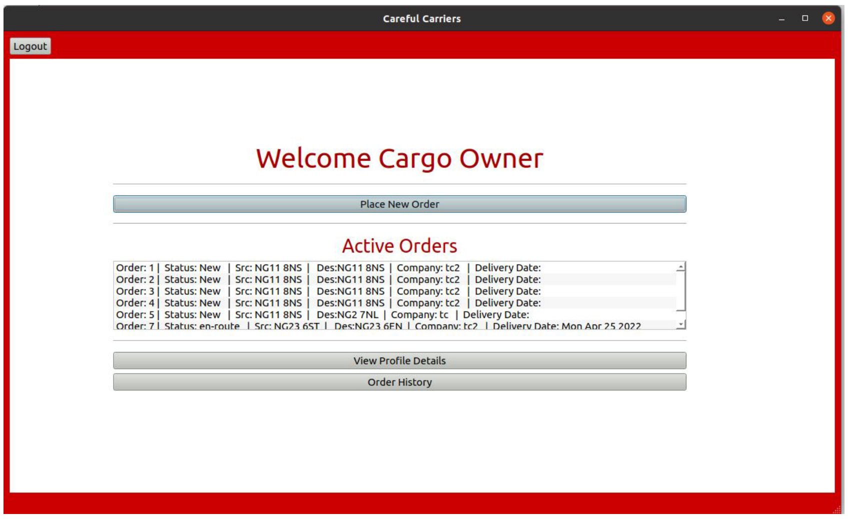
Order: 1 | Pay: 0 | Src: NG11 8NS | Des: NG11 8NS | Compan
Order: 2 | Pay: 0.27 | Src: NG11 8NS | Des: NG11 8NS | Com

Active Order State

Order: 7 | Src: NG23 6ST | Des: NG23 6EN | Company: tc2

Figure 36: Driver Home Page

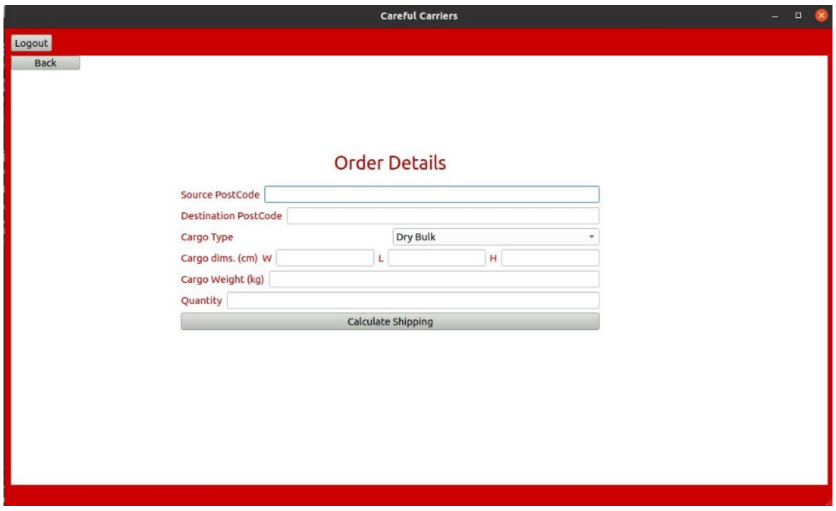
This figure shows the driver home page where they can see incoming orders and active orders on a glance. When they click on incoming orders, they are shown all the information for the order and prompted to accept the order. When clicking on an item in Active orders they will see additional information and can change the status in the combo box below with the addition of the confirm button. They can also go to order history or the profile page



```
Order: 1 | Status: New  | Src: NG11 8NS |  Des:NG11 8NS | Company: tc2  | Delivery Date:
Order: 2 | Status: New  | Src: NG11 8NS |  Des:NG11 8NS | Company: tc2  | Delivery Date:
Order: 3 | Status: New  | Src: NG11 8NS |  Des:NG11 8NS | Company: tc2  | Delivery Date:
Order: 4 | Status: New  | Src: NG11 8NS |  Des:NG11 8NS | Company: tc2  | Delivery Date:
Order: 5 | Status: New  | Src: NG11 8NS |  Des:NG2 7NL | Company: tc  | Delivery Date:
Order: 7 | Status: en-route  | Src: NG23 6ST |  Des:NG23 6EN | Company: tc2  | Delivery Date: Mon Apr 25 2022
```

Figure 37: Cargo Owner Home Page

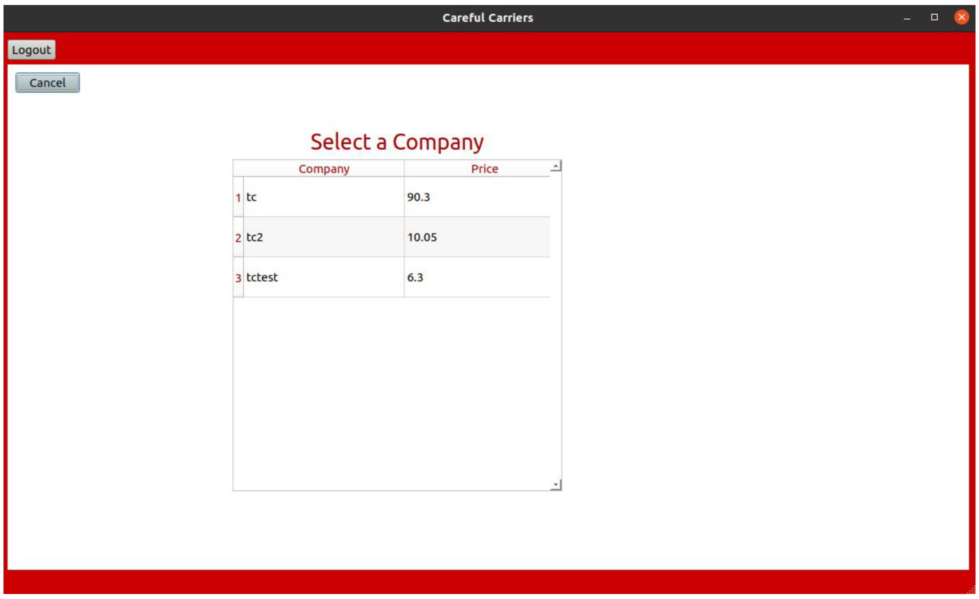
This figure shows cargo owner home page where they can see active orders on a glance. When clicking on an item in Active orders they will see additional information. They can also go to order history, profile, or new order screen.



Source PostCode
Destination PostCode
Cargo Type: Dry Bulk
Cargo dims. (cm) W L H
Cargo Weight (kg)
Quantity

Figure 38: Order Page

This figure shows the order page and all the information a cargo owner must enter to submit an order.

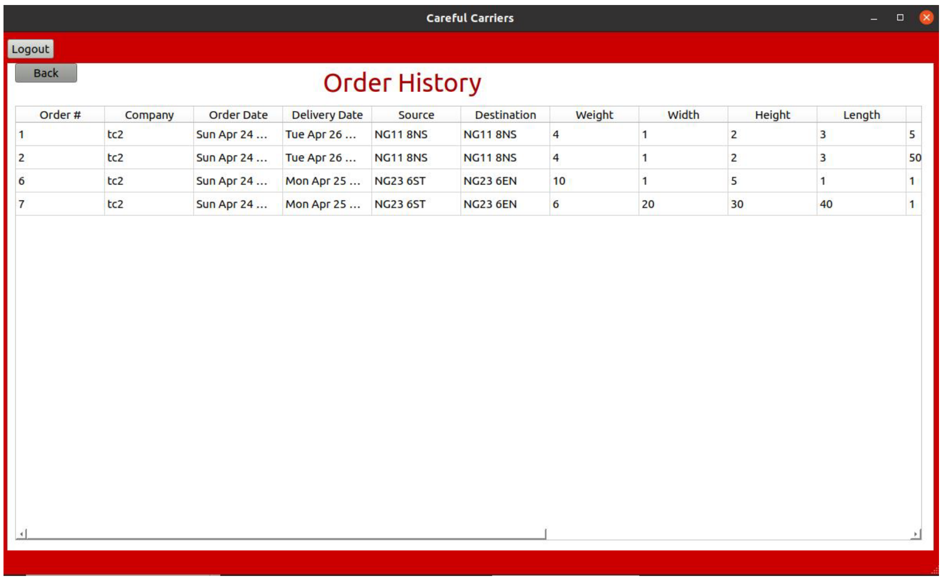


Select a Company

	Company	Price
1	tc	90.3
2	tc2	10.05
3	tctest	6.3

Figure 39: Order Selection Page

This figure shows the order selection page and allows the cargo owner to see every company that uses the application and their price for that order.



Order History

Order #	Company	Order Date	Delivery Date	Source	Destination	Weight	Width	Height	Length	
1	tc2	Sun Apr 24 ...	Tue Apr 26 ...	NG11 8NS	NG11 8NS	4	1	2	3	5
2	tc2	Sun Apr 24 ...	Tue Apr 26 ...	NG11 8NS	NG11 8NS	4	1	2	3	50
6	tc2	Sun Apr 24 ...	Mon Apr 25 ...	NG23 6ST	NG23 6EN	10	1	5	1	1
7	tc2	Sun Apr 24 ...	Mon Apr 25 ...	NG23 6ST	NG23 6EN	6	20	30	40	1

Figure 40: Order History Page

This figure shows the order history page. It is very similar for all users. It displays all delivered (and denied for the cargo owner) orders and their information. It also allows cargo owners to submit feedback that companies and drivers can view on their table.

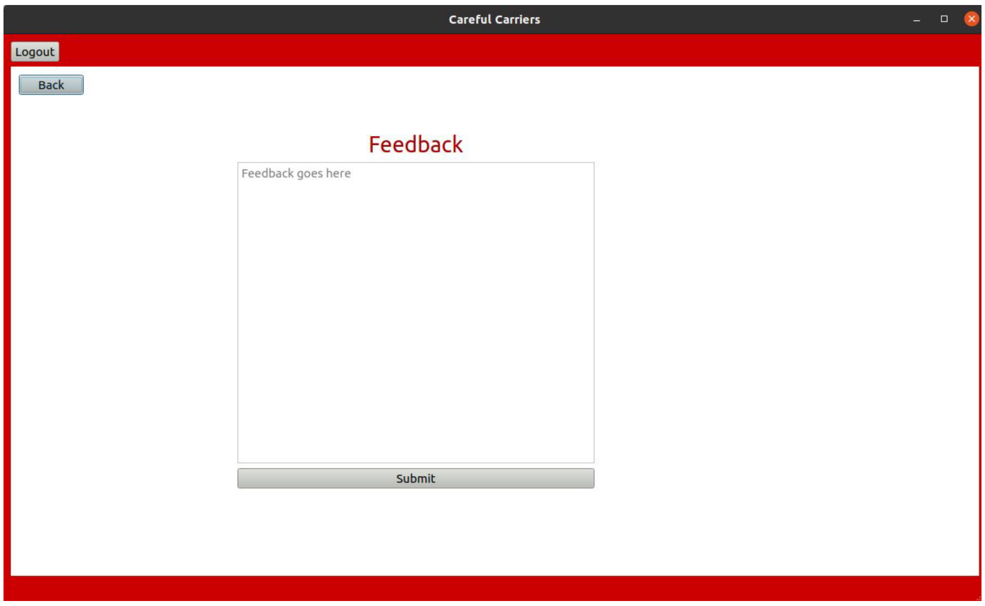


Figure 41: Feedback Page

This figure shows the feedback page where the cargo owner submits feedback for the driver and transport company.

System Analysis Conclusion

In summary, the system has been broken down into various cases and portrayals to better understand the project as well as improve the development process. The analysis lays out the major interactions between the users and the system as well as the functions of the system to perform the designated tasks.

User Manual

Located in the documentation folder on the GitHub a direct link is:
[SDI_Project/GROUP_B3_SDI_User_Manual.pdf at master · B3/SDI_Project (ntu.ac.uk)]

Login Page

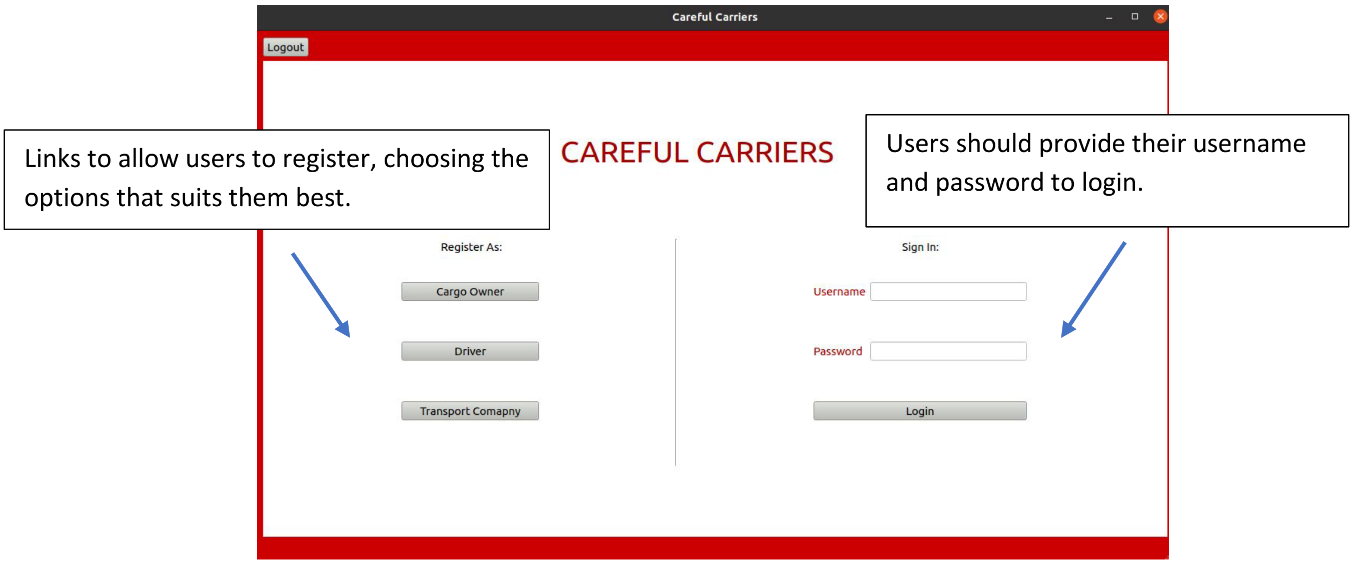
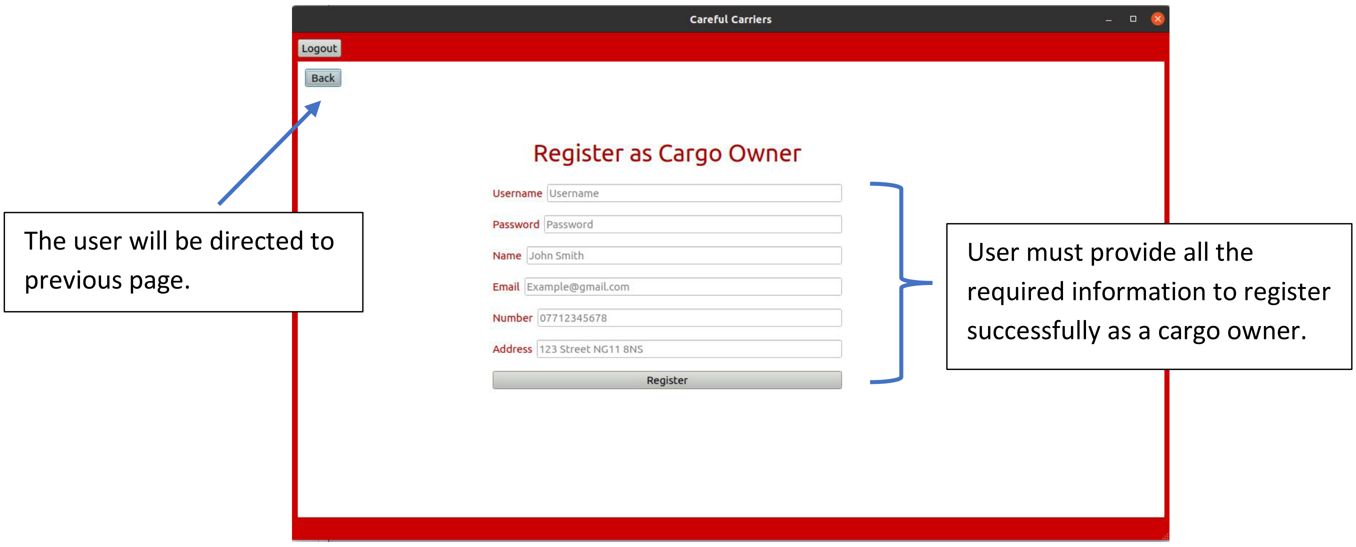

Careful Carriers
Logout
Links to allow users to register, choosing the options that suits them best.
CAREFUL CARRIERS
Users should provide their username and password to login.
Register As:
Cargo Owner
Driver
Transport Comapny
Sign In:
Username
Password
Login

Figure 42: User Manual Login

Cargo Owner Registration Page



Register as Cargo Owner

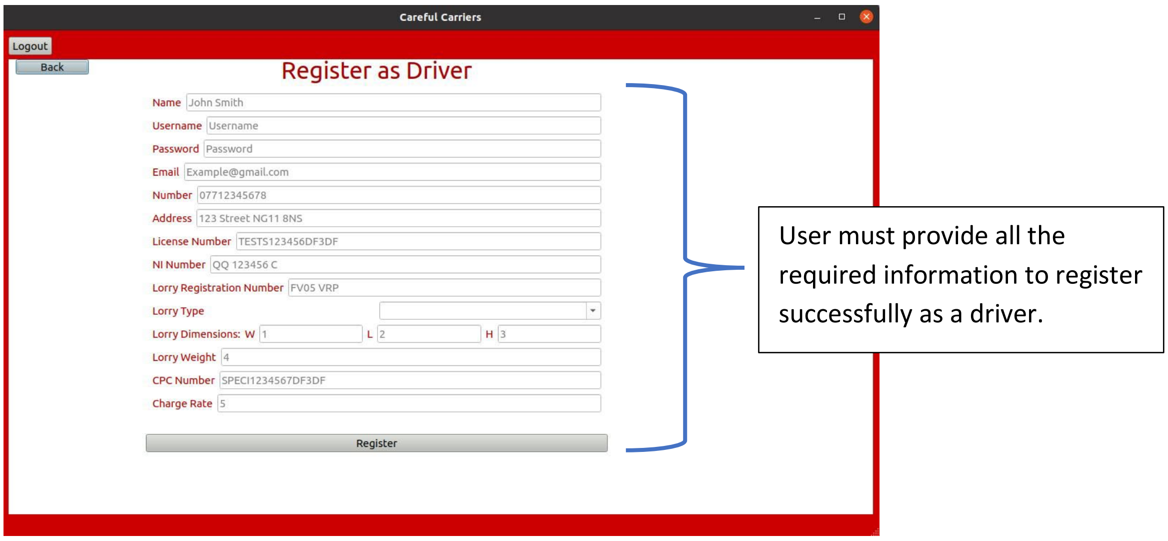
Username	Username
Password	Password
Name	John Smith
Email	Example@gmail.com
Number	07712345678
Address	123 Street NG11 8NS

The user will be directed to previous page.

User must provide all the required information to register successfully as a cargo owner.

Figure 43: User Manual Register Cargo Owner

Driver Registration Page



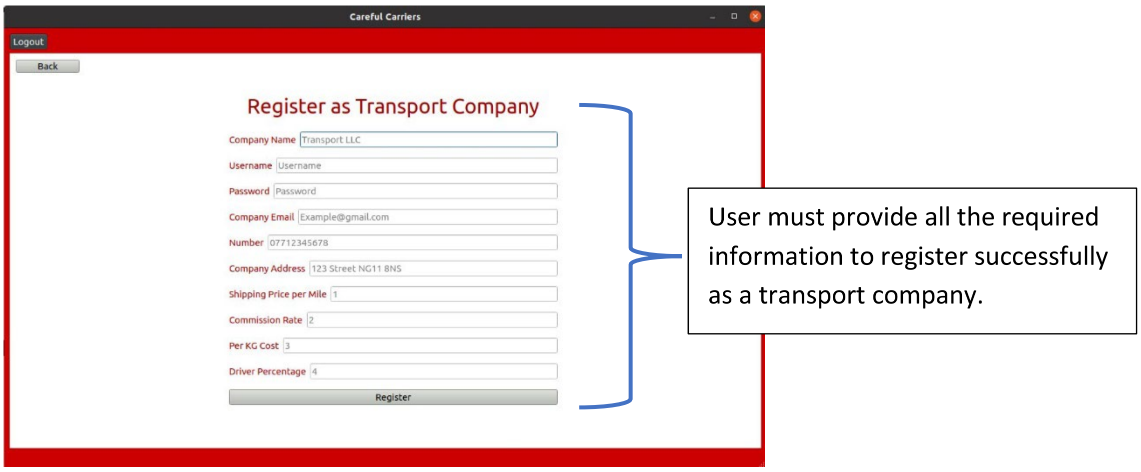
Register as Driver

Name	John Smith
Username	Username
Password	Password
Email	Example@gmail.com
Number	07712345678
Address	123 Street NG11 8NS
License Number	TESTS123456DF3DF
NI Number	QQ 123456 C
Lorry Registration Number	FV05 VRP
Lorry Type	
Lorry Dimensions: W	1 L 2 H 3
Lorry Weight	4
CPC Number	SPECI1234567DF3DF
Charge Rate	5

User must provide all the required information to register successfully as a driver.

Figure 44: User Manual Register Driver

Transport Company Registration Page



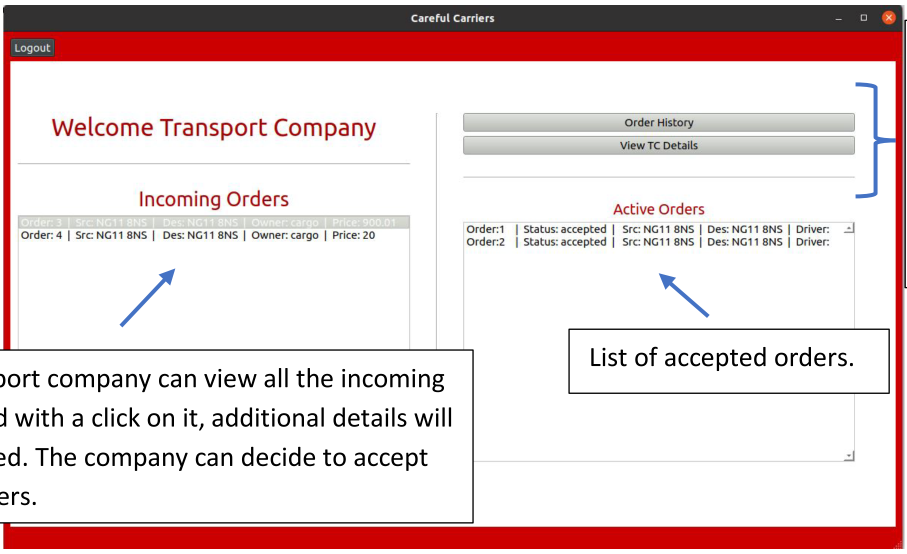
Register as Transport Company

Company Name	Transport LLC
Username	Username
Password	Password
Company Email	Example@gmail.com
Number	07712345678
Company Address	123 Street NG11 8NS
Shipping Price per Mile	1
Commission Rate	2
Per KG Cost	3
Driver Percentage	4

User must provide all the required information to register successfully as a transport company.

Figure 45: User Manual Register Transport Company

Transport Company home page



```
Order: 3 | Src: NG11 8NS | Des: NG11 8NS | Owner: cargo | Price: 900.01
Order: 4 | Src: NG11 8NS | Des: NG11 8NS | Owner: cargo | Price: 20
```

```
Order:1 | Status: accepted | Src: NG11 8NS | Des: NG11 8NS | Driver:
Order:2 | Status: accepted | Src: NG11 8NS | Des: NG11 8NS | Driver:
```

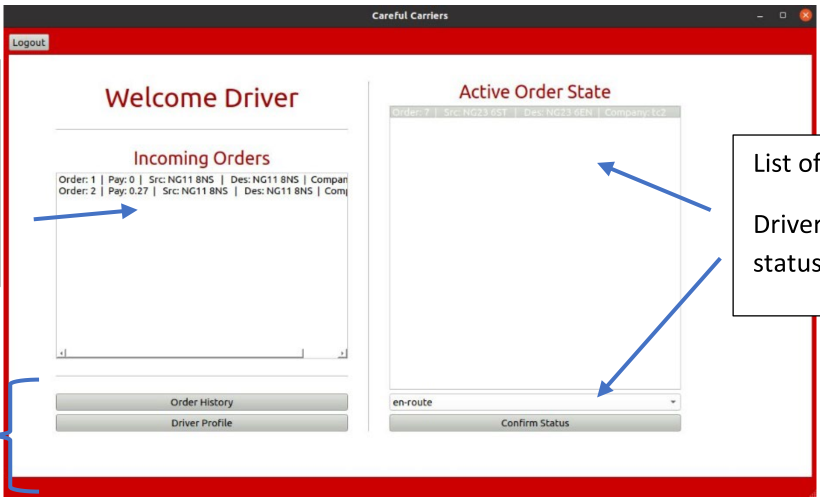
The user can view the order history by clicking on the first button and update its details using the second button.

List of accepted orders.

The transport company can view all the incoming orders and with a click on it, additional details will be provided. The company can decide to accept reject orders.

Figure 46: User Manual Transport Company Home

Driver Home Page



```
Order: 1 | Pay: 0 | Src: NG11 8NS | Des: NG11 8NS | Compan
Order: 2 | Pay: 0.27 | Src: NG11 8NS | Des: NG11 8NS | Comp
```

```
Order: 7 | Src: NG23 6ST | Des: NG23 6EN | Company: tc2
```

Drivers can view incoming orders and with a single click on it, additional details will be displayed. The driver can accept or

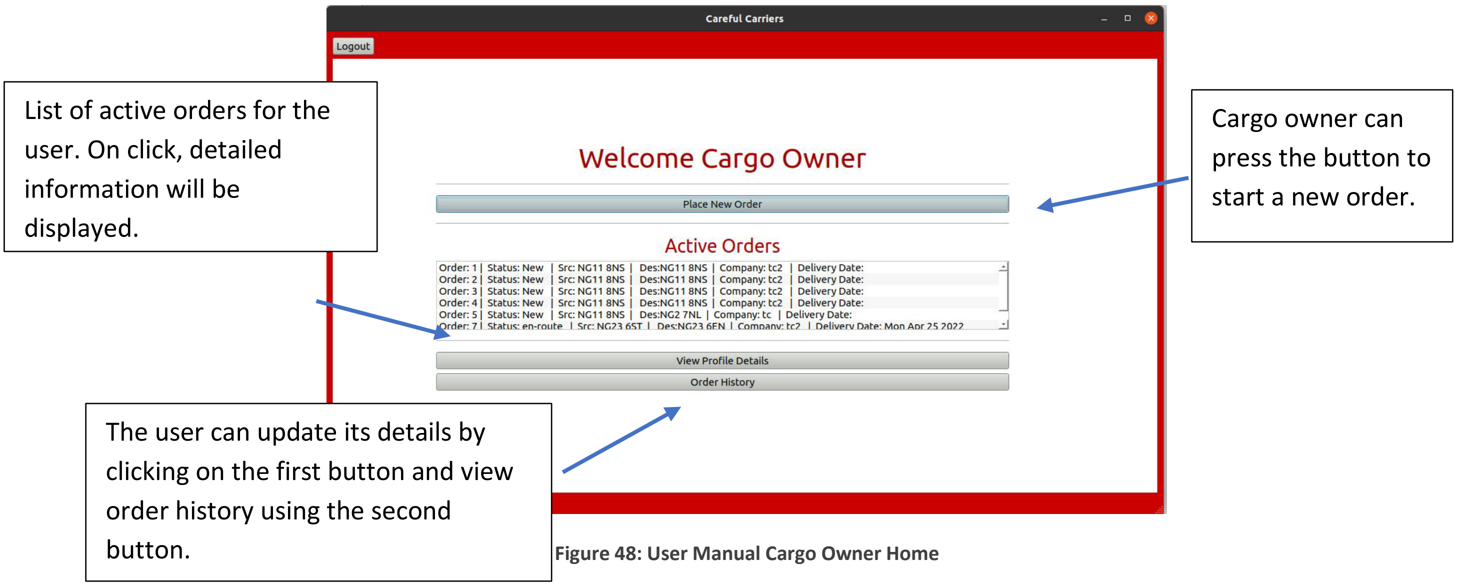
List of active orders.

Driver can change the order status using the combo box.

The user can view the order history by clicking on the first button and update its details using the second button.

Figure 47: User Manual Driver Home

Cargo Home Page



```
Order: 1 | Status: New  | Src: NG11 8NS |  Des:NG11 8NS | Company: tc2  | Delivery Date:
Order: 2 | Status: New  | Src: NG11 8NS |  Des:NG11 8NS | Company: tc2  | Delivery Date:
Order: 3 | Status: New  | Src: NG11 8NS |  Des:NG11 8NS | Company: tc2  | Delivery Date:
Order: 4 | Status: New  | Src: NG11 8NS |  Des:NG11 8NS | Company: tc2  | Delivery Date:
Order: 5 | Status: New  | Src: NG11 8NS |  Des:NG2 7NL | Company: tc  | Delivery Date:
Order: 7 | Status: en-route  | Src: NG23 6ST |  Des:NG23 6EN | Company: tc2  | Delivery Date: Mon Apr 25 2022
```

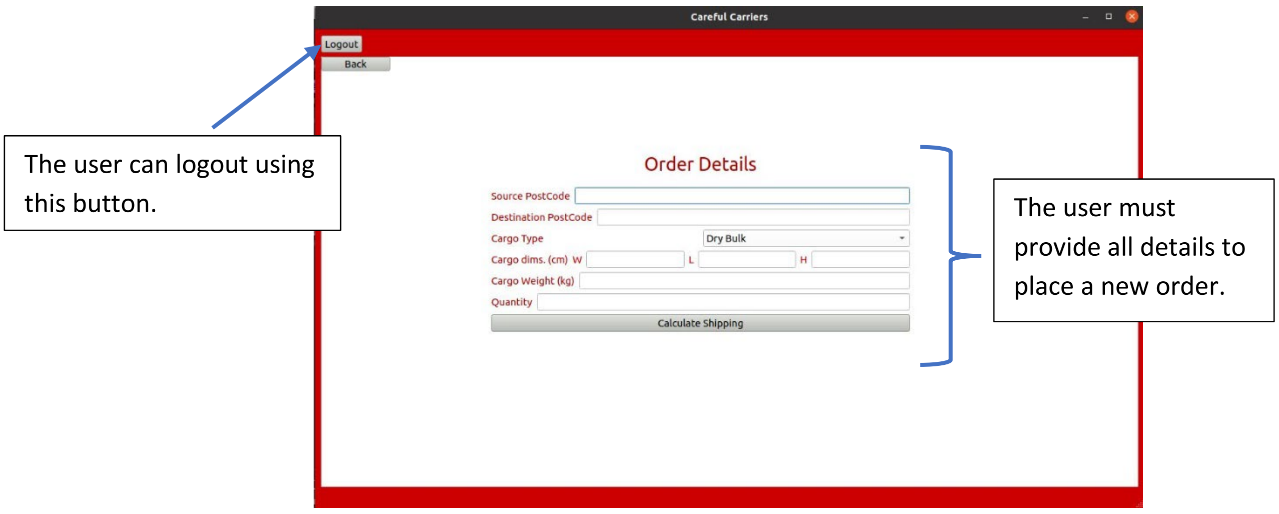
List of active orders for the user. On click, detailed information will be displayed.

Cargo owner can press the button to start a new order.

The user can update its details by clicking on the first button and view order history using the second button.

Figure 48: User Manual Cargo Owner Home

New order page

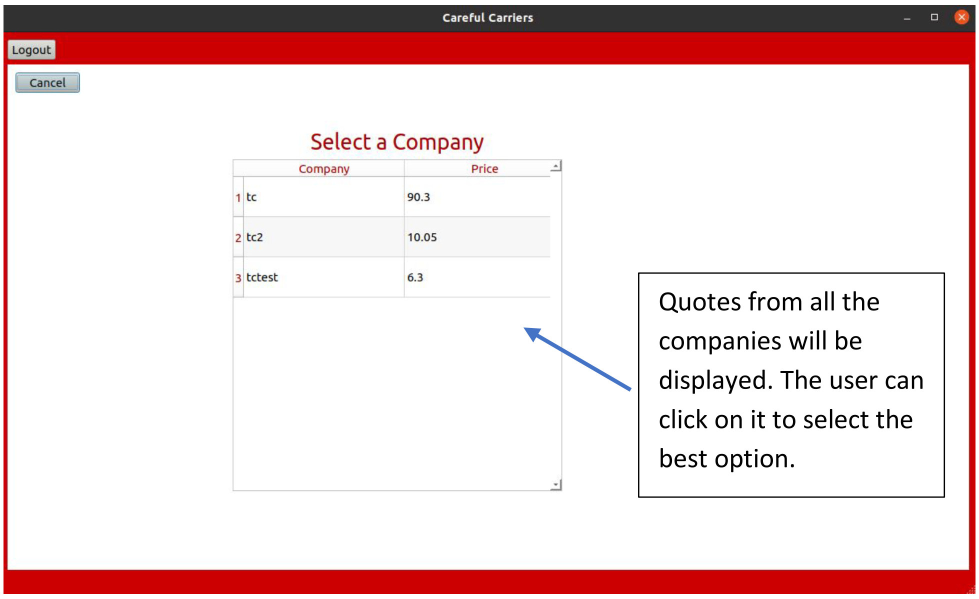


The user can logout using this button.

The user must provide all details to place a new order.

Figure 49: User Manual Order

Quotes Page

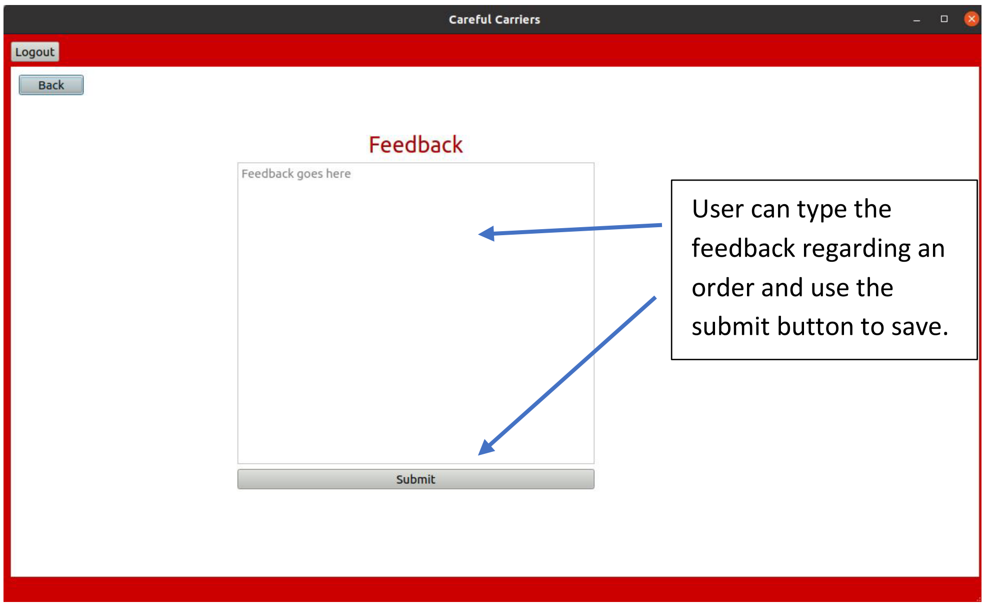


	Company	Price
1	tc	90.3
2	tc2	10.05
3	tctest	6.3

Quotes from all the companies will be displayed. The user can click on it to select the best option.

Figure 50: User Manual Order Selection

Feedback page



User can type the feedback regarding an order and use the submit button to save.

Figure 51: User Manual Feedback

Reference Manual

Located in the documentation folder on the GitHub a direct link is: [SDI_Project/refman.pdf at master · B3/SDI_Project (ntu.ac.uk)]

Coding Contribution Guide

Contribution Guidelines

Team members will contribute to every aspect of the project. Tasks will be assigned on a weekly basis with short term goals and deadlines. Each task will be decided weekly and posted onto the Trello

Workspace so the task, assigned member, and due date can be noted. As tasks are completed or extended the Trello will be used to keep track of the amount of contribution of each member. Members will lead their relevant sections, but each member must still contribution equally. All contributions will review by the other members.

GitHub and Pull Requests

All files and data pushed to the GitHub will be first put on the pending branch. Any pull requests to the master branch shall be reviewed by every member of the team during the meetings. Any pushes shall have a detailed description of changes made. Any comments on the pull shall be clear and shall include what needs to be done to fix any issues.

Code of Conduct

Team Members are to be always professional and polite. Each member opinions will be taken into consideration for the project. Team Members shall be present in meetings and communicate any issues. Members will be on time for meetings and actively engage in the meetings. All contribution to meetings will be noted in the minutes. The Project Manager will record the meeting minutes and handle any issues from the members. Members will contribute to every part of the project and understand any parts they do not actively work on. All decisions shall be made by consensus of the team. The team will be professional, efficient, engaging, and organized.

Testing

Introduction

The approach to testing the program is to create tests for each requirement declared in the initial weeks of the project. The passing of these tests will determine the success of the project. Automated testing frameworks such as Boost were not used as a large majority of the requirements were functional requirements, opposed to expecting certain outputs from inputs

Test Tables

Table 4 - Sign up

Title: Sign up	
ID: 1	**Description:** Test to check account creation for user. Depending on the type of user the function will ask for certain information
Test type: Functional	**Success criteria:** User can create an account and is asked certain information based on type of user.
Number of attempts: 3	**Comments:** Total of 3 attempts one for each type of user
List of equipment/ requirements: User credentials	
Setup instructions: 1. Run the program and set up account	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	

Individual results: Attempt 1: Success Attempt 2: Success Attempt 3: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 5 - User Management

Title: User management	
ID: 2	**Description:** Test to check **s**ystem stores and tracks information of users provided in sign up function and order history.
Test type: Functional	**Success criteria:** System stores and tracks information of users provided.
Number of attempts: 3	**Comments:** Total of 3 attempts one for each user
List of equipment/ requirements: N/A	
Setup instructions: 1. Test 1 2. Check database is storing information	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success Attempt 2: Success Attempt 3: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 6 - Allow edit of user details

Title: Allow edit of user details	
ID: 3	**Description:** Test to check users can update their information.
Test type: Functional	**Success criteria:** Users can update their information
Number of attempts: 3	**Comments:** Total of 3 attempts one for each user
List of equipment/ requirements: N/A	
Setup instructions: 1. User logs in 2. User edits details	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success Attempt 2: Success Attempt 3: Success	**Video:**

Test date: 23/04/2022	**Result:** Success

Table 7 - Login/ Logout

Title: Login / Logout	
ID: 3	**Description:** Test to check system allows only valid users to enter the system.
Test type: Functional	**Success criteria:** Only users who have signed up can login/ logout
Number of attempts: 3	**Comments:** Total of 3 attempts one for each type of user
List of equipment/ requirements: N/A	
Setup instructions: 1. User logs in successfully with signed up details 2. User attempts login with non-signed up details Respective Credentials: User: cargo Password: cargo User: driver Password: driver User: tc Password: tc	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success Attempt 2: Success Attempt 3: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 8 - Calculate shipping rate

Title: Calculate shipping rate	
ID: 5	**Description:** Test to check system allows Cargo Owner to calculate shipping price based on the source, destination, and needed lorry for each order.
Test type: Functional	**Success criteria:** Cargo owner calculates shipping correctly
Number of attempts: 1	**Comments:**
List of equipment/ requirements: N/A	
Setup instructions: 1. Cargo owner logs in 2. Cargo owner retrieves calculated shipping price	

Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 9 - Place order

Title: Place order	
ID: 6	**Description:** Test to check system allows Cargo Owner to place order containing source, destination, cargo dimensions, weight, and condition.
Test type: Functional	**Success criteria:** Cargo owner can provide input for necessary fields
Number of attempts: 1	**Comments:**
List of equipment/ requirements: N/A	
Setup instructions: 1. Cargo owner logs in 2. Cargo owner can submit order details	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 10 - Accept/Reject Orders

Title: Accept/Reject Orders	
ID: 7	**Description:** Test to check system sends Driver notifications of available order and driver can accept or reject the cargo. If rejected will send to another driver.
Test type: Functional	**Success criteria:** Driver receives notification and can accept/ reject.
Number of attempts: 1	**Comments:**
List of equipment/ requirements: N/A	
Setup instructions: 1. Driver logs in 2. Driver views order notifications 3. Driver can accept/ reject order	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 11 - View/change order status

Title: View/change order status	
ID: 8	**Description:** Test to check system allows Cargo Owner and company to track the status of the order as indicated by the driver. Status such as loading, enroute, or delivered.
Test type: Functional	**Success criteria:** Cargo owner and company can track status of order
Number of attempts: 1	**Comments:**
List of equipment/ requirements: N/A	
Setup instructions: 1. Cargo owner/ Transport company logs in 2. Cargo owner/ Transport company can track status of order	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 12 - Calculate commission

Title: Calculate commission	
ID: 9	**Description:** Test to check system allows Transport company to calculate commission of each order based upon shipping rate.
Test type: Functional	**Success criteria:** Transport company calculates commission
Number of attempts: 1	**Comments:**
List of equipment/ requirements: N/A	
Setup instructions: 1. Transport company logs in 2. Transport company calculates commission	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 13 - Send/ receive notification

Title: Send/ receive notifications	
ID: 10	**Description:** Test to check Users receive notifications when transport company receive an order, driver when new order is available, and cargo owner when order is accepted.
Test type: Functional	**Success criteria:** User receives notification successfully
Number of attempts: 1	**Comments:**

List of equipment/ requirements: N/A	
Setup instructions: 1. Transport company logs in 2. Transport company receives order notification 3. Driver logs in 4. Driver receives order notification 5. Cargo owner logs in 6. Cargo owner receives notification	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 14 - Track Lorry/Driver Location

Title: Track Lorry/Driver Location	
ID: 11	**Description:** Test to check Cargo owner and transport company can get real time gps tracking of lorry and driver.
Test type: Functional	**Success criteria:** Cargo owner and transport company can track lorry and driver
Number of attempts: 1	**Comments:** Cargo owners and transport company can view status of Lorry
List of equipment/ requirements: N/A	
Setup instructions: 1. Transport company logs in 2. Transport company tracks lorry and driver 3. Cargo owner logs in 4. Cargo owner tracks lorry and driver	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 15 - Delivery Notification and report

Title: Delivery Notification and report	
ID: 12	**Description:** Test to check Driver provides delivery notification along with proof of deliver for cargo owner and transport company.
Test type: Functional	**Success criteria:** Driver provides delivery notification along with proof of deliver to cargo owner
Number of attempts: 1	**Comments:**
List of equipment/ requirements: N/A	
Setup instructions:	

1. Cargo owner logs in 2. Checks delivery notification from Driver 3. Transport company logs in 4. Checks delivery notification from Driver	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:** Success
Test date: 23/04/2022	**Result:** Success

Table 16 - Order/Shipping history

Title: Order/Shipping history	
ID: 13	**Description:** Test to check Cargo owner/ Transport company can view order history and drivers to view shipping history.
Test type: Functional	**Success criteria:** Cargo owner/ Transport company view order history. Drivers view shipping history
Number of attempts: 1	**Comments:** Every user type can login and view order history table
List of equipment/ requirements: N/A	
Setup instructions: 1. Cargo owner logs in 2. Views order history 3. Transport company logs in 4. Views order history 5. Driver logs in 6. Views shipping history	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 17 - Invoice system

Title: Invoice system	
ID: 14	**Description:** Test to check system sends invoice (based upon shipping rate) to cargo owner upon acceptance of driver.
Test type: Functional	**Success criteria:** Cargo owner receives invoice
Number of attempts: 1	**Comments:** Not so much an invoice but can view full order information
List of equipment/ requirements: N/A	
Setup instructions: 1. Cargo owner logs in 2. Views received invoice	

Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 18 - Allow Comments/ recommendation

Title: Allow Comments/ recommendation	
ID: 15	**Description:** Test to check Cargo owners can leave feedback and comments about orders and delivery.
Test type: Functional	**Success criteria:** Cargo owner can leave feedback and comments
Number of attempts: 1	**Comments:** Cargo owner unable to leave comments. Driver and Transport company can leave feedback
List of equipment/ requirements: N/A	
Setup instructions: 1. Cargo owner logs in 2. Leaves feedback and comments	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 19 - View feedback

Title: View feedback	
ID: 16	**Description:** Test to check Driver and Transport company can view feedback/comments from cargo owner.
Test type: Functional	**Success criteria:** Driver and Transport company view feedback/comments
Number of attempts: 1	**Comments:** Driver and Transport Company can view feedback/comments from one another. Not from cargo owner however- due to test 15 failing.
List of equipment/ requirements: N/A	
Setup instructions: 1. Driver owner logs in 2. Views feedback and comments 3. Transport company logs in 4. Views feedback and comments	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 20 - Database Management

Title: Database Management	
ID: 17	**Description:** Test to check System has file management system to store user details and order information.
Test type: Functional	**Success criteria:** File management system is present
Number of attempts: 1	**Comments:** Users can view profile details and order history.
List of equipment/ requirements: N/A	
Setup instructions: 1. View profile settings	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 21 - Offline Database System

Title: Offline Database System	
ID: 18	**Description:** Test to check System stores messages and notifications until user is active.
Test type: Functional	**Success criteria:** System stores messages and notifications
Number of attempts: 1	**Comments:** System stores incoming orders ready for relevant user to view and respond appropriately
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 22 - View/Accept Tender

Title: View/Accept Tender	
ID: 19	**Description:** Test to check System allows shipper to view and accept the tender. Allow each tender to be created by a proposed price from Carrier etc.
Test type: Functional	**Success criteria:** System allows shipper to choose prices and accept tender. Tender is created via proposed price- supplied by carrier
Number of attempts:	**Comments:**

List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 23 - User Interface

Title: User Interface	
ID: 20	**Description:** Test to check System has user interface
Test type: Interface	**Success criteria:** System has a user interface
Number of attempts:	**Comments:** User interface is present throughout the system
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 24 - Minimum Clicks

Title: Minimum Clicks	
ID: 21	**Description:** Test to check System allows Users to access any feature within 3 clicks
Test type: Interface	**Success criteria:** System has a cap of 3 clicks to access features
Number of attempts: 3	**Comments:**
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success Attempt 2: Success Attempt 3: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 25 - Response Time of < 4 seconds

Title: Response Time of < 4 seconds	
ID: 22	**Description:** Test to check System response time must is below 2 seconds
Test type: Non- Functional	**Success criteria:** System has a response time of 2 seconds
Number of attempts:	**Comments:** Used simple stopwatch to test response time of tasks such as: login; view order history; submit order etc.
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 26 - Follow GDPR regulation

Title: Follow GDPR regulation	
ID: 23	**Description:** Test to check System adheres to the Data Protection Principles
Test type: Non- Functional	**Success criteria:** System allows deletion of any of their information
Number of attempts: 3	**Comments:** Every user can alter details. Sensitive information such as passwords are hashed.
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success Attempt 2: Success Attempt 3: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 27 - **Phone number Verification**

Title: Phone number Verification	
ID: 24	**Description:** Test to check system verifies phone number
Test type: Non- Functional	**Success criteria:** System verifies phone number
Number of attempts:	**Comments:** System doesn't flag incorrect phone number. Validation was successfully implemented upon flagging this error to Software Developers.
List of equipment/ requirements: N/A	

Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Fail Attempt 2: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 28 - Verify CPC/Lorry Number

Title: Verify CPC/Lorry Number	
ID: 25	**Description:** Test to check system verifies CPC/Lorry number
Test type: Non- Functional	**Success criteria:** System verifies CPC/Lorry number
Number of attempts: 1	**Comments:** API validation is present for both CPC and Lorry number
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1:	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 29 -Password

Title: Password	
ID: 26	**Description:** Test to check system ensures password should be 8 character long, contain a capital letter and number.
Test type: Non- Functional	**Success criteria:** System ensures use of strong password
Number of attempts: 1	**Comments:** System allows any type of password- failing to enforce use of capitals; number; specific length. This was subsequently edited upon meeting with Software Developers
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Fail Attempt 2: Success	**Video:**

Test date: 23/04/2022	**Result:** Success

Table 30 - **Error Handling**

Title: Error Handling	
ID: 27	**Description:** Test to check system handles improper input.
Test type: Non- Functional	**Success criteria:** System has try/catch coding
Number of attempts:	**Comments:** Try catches are present throughout the source code
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

Table 31 - **Ensure each user type only has access to their features**

Title: Ensure each user type only has access to their features	
ID: 28	**Description:** Test to check system limits capabilities to only relevant user roles. A Driver cannot access the Cargo Owners abilities vice versa, etc.
Test type: Non- Functional	**Success criteria:** System secures information of other users
Number of attempts:	**Comments:**
List of equipment/ requirements: N/A	
Setup instructions: 1.	
Failure correction procedure: Debug the program and fix the bug	
Engineer/ Technician: Tugra Karakus	
Individual results: Attempt 1: Success	**Video:**
Test date: 23/04/2022	**Result:** Success

In conclusion the outcome of 28/28 tests passing indicates a successful program. Passing a test meant that the tests expectations were achieved successfully. Thorough testing is very important in ensuring the initial requirements are met. Some tests initially failed; however, this was dealt with appropriately by having meetings with Software Tester and Software Developers. The tester informed the developers of the failed tests, and these were fixed accordingly. Subsequently this resulted in a functional system.

Testing ensures not only that the code works but its implementation is working completely. Having the test cases based on the requirement list ensures the client is confident the system works as their requirements specified. Overall due to the number of tests that passed successfully we can say that the design reflects the implementation.

Demo

Link to demo on YouTube -> https://youtu.be/ebKizav_TeM

Conclusions and Future Work

Requirements Implemented

Table 32 - Requirements Implemented

Requirement	Completed
Cargo Owner	
MUST sign up and create new account (customer details, email, home/business address, username, password, ..., mobile number to receive verification message)	Done
COULD view profile, edit customer details.	Done
For security reasons customer details SHOULD be encrypted when stored into selected database (or plain files/json/csv,xml)	Done
MUST be able to sign in / logout	Done
MUST Calculate shipping rates (source / destination, lorry type)	Done
MUST place cargo order from a source to a destination (cargo details shall include cargo's dimensions, weight, conditions e.g. fragile or frozen goods)	Done
MUST View order status	Done
SHOULD receive notification when order was accepted by transportation company.	Done
COULD View invoices issued by transportation company.	Done
COULD Track lorry on the road	
SHOULD View delivery notification and report (including electronic proof of delivery	Done
COULD View order history/ feedbacks/ rate shipment	Done
SHOULD Add comments and recommendation	Done
Drivers	
MUST sign up and create new account (driver's details, NI number, Driving licence ID, photograph, email, home/business address, username, password, ..., mobile number to receive verification message, type of lorry	Done
SHOULD include following fields for the driver: lorry reg number, lorry dimension and weight, Driver Certificate of Professional Competence (CPC) number	Done
CPC and lorry reg number MUST be validated using an API.	Done
For security reasons driver details SHOULD be encrypted when stored into selected database	Done
SHOULD View/modify driver's details.	Done
MUST Sign in/ log out	Done

MUST Receive new orders notification and accept/reject cargos (order details including fee, source, destination, and cargo's details	Done
SHOULD Notify cargo owner (loading, on road, delivered)	Done
COULD View shipments history	Done
COULD Send message to cargo owner	
Transportation Company	
MUST Sign up and create account (company details)	Done
MUST Sign in/ log out	Done
MUST Receive order from customers (notifications)	Done
MUST Forward order to a free driver nearby the source (in case of rejecting order by one driver, the system try another driver until no free driver is found)	Done
SHOULD issuing invoice to the customer when the order was accepted by a driver	
MUST Calculate company's commission for each order.	Done
COULD Track driver location/status.	Done
SHOULD view feedbacks for each order (and driver)	Done
COULD View orders history	Done

Design Contributions to Development

The design was helpful to layout the provided classes and general flow of the application; it helped breakdown the application into the relevant components and use cases making the early stages of development smooth with classes, attributes, and function declarations. On implementation, the design helped to structure the flow of the application with the larger activities and components laid out making writing code for them more manageable.

Results

The implemented application is fully functioning and covers almost every aspect of the requirements. It is a reliable piece of software with many functions and makes use of several tool/libraries. The application varies somewhat from the design but only to make improvements upon the initial design. The application is well tested for errors and validation with any failed tests being fixed. Overall, it is an excellent demonstration of an e-transport marketplace that could be implemented in the future.

Future Work

The future work of the project could include:

-Invoice system

-Live Tracking of drivers

-A more complex driver order assignment

-a change from database driven design

An invoice system is an important factor if a transportation company wants a piece of software to do everything for them. The lack of one means it is less likely to be used if a similar application that has an invoice system is available for that company.

Live tracking improves the user experience so they can get reliable updates about their orders. It also makes the pickup and delivery of orders less stressful for the cargo owners.

Driver assignment for the most efficient systems is based off the closest available driver. Therefore, a system that would find the nearest driver and create efficient schedules for them would be a great improvement to the project.

Database driven designs rely on the speed in which you can connect to your database. If the application were to be move to a different approach, there could be a significant improvement in application speed.

Individual Contributions and Group Experience

Group Experience

The group has exceeded expectations in both attendance and contributions. Every member has fully participated in both their own roles and in helping others with their work. While group work is different to schedule and divide, the group had always made adequate alternatives for any of these issues. The group was a fully functioning amazing team that came together to make excellent work we are all proud of.

Contributions

Project Manager – Maya Rhoades

Group management and time management (Teams, Trello, Gantt Chart, Requirements)

Creation of report

Created basis of gui design

Software designer – Sam Allsop

Creation of UML diagrams with consistent edits to any design over the course of the project (UML Diagrams and explanations)

Documentation of design

Added to the report

Software Developer – Thomas Monkhouse and Shivangi Prajapati

Implementation of database and code (cloud MySQL and qt code)

Generated User manual and reference manual

Added to the report

Software Test – Tugra Karakus

Creation of tests and logging their results

Added to the report

Reflection

The client’s requirements are a good means of reflecting the success of the software. The passing of all requirements set out by the client is a necessity for well implemented and functioning product. Checking the product has met the must, should and requirements will be a suitable method for the client to verify the product has met their laid-out requirements. Knowing that our software passed all tests and met all requirements reflects that the product was a success.

Appendix

Coding Contribution Guide

In Documentation folder in GitHub link -> SDI_Project/Group_B3_SDI_Coding_Contribution_Guide.pdf at master · B3/SDI_Project (ntu.ac.uk)

Coding Standards Guide

In Documentation folder in GitHub link -> SDI_Project/Group_B3_SDSI_Coding_Standards.pdf at master · B3/SDI_Project (ntu.ac.uk)

GitHub

Link to GitHub -> B3/SDI_Project (ntu.ac.uk)

User Manual

In Documentation folder in GitHub link -> SDI_Project/GROUP_B3_SDI_User_Manual.pdf at master · B3/SDI_Project (ntu.ac.uk)

Reference Manual

In Documentation folder in GitHub link -> SDI_Project/refman.pdf at master · B3/SDI_Project (ntu.ac.uk)

References

AJOT, 2021. *How is technology changing the future of freight forwarders?* [online]. . Available at: https://logfret.com/how-is-technology-changing-the-future-of-freight-forwarders/ [Accessed 25/04/ 2022].

GUAJARDO, C., 2021. *Freight Technology: How Emerging Tech is Reshaping the Logistics Industry* [online]. . Available at: https://www.sdcexec.com/software-technology/article/21354913/forager-freight-technology-how-emerging-tech-is-reshaping-the-logistics-industry [Accessed 25/04/ 2022].

MITCHELL, S., 2021. *Growth of e-commerce driving the road freight transportation market in Europe* [online]. . Available at: https://www.reutersevents.com/supplychain/freight/growth-e-commerce-driving-road-freight-transportation-market-europe [Accessed 25/04/ 2022].

POPOC, A., 2021. *How COVID-19 triggered the digital and e-commerce turning point* [online]. . Available at: https://unctad.org/news/how-covid-19-triggered-digital-and-e-commerce-turning-point [Accessed 25/04/ 2022].