Design and Testing Document

Project 1: API Development and Testing
University of British Columbia Okanagan
COSC499 - Summer 2022

Date: Jun 9, 2022

Contacts:

Wasi-ul-Hassan Raza(wasiulhassanraza@gmail.com)

Calvin Qu(Calvin000128@gmail.com)

Po-Chia Wei(lunch33665566@gmail.com)

Austin Wong(austinwong1@yahoo.ca)

1. Software Description

HelpYa is a trade service procurement app that connects clients and businesses. Using the platform, customers will be able to purchase, search, and negotiate services with the business of their choosing. The client would like to design and develop a backend for an ios app for now with plans for cross-platform for android and the web in the future. During our time in capstone, our scope is to create an API that can handle login authentication, create delete updates and report user information and is able handle monetary transactions.

2. User Groups

- Consumer account (users who are looking for services)
- Bussiness account (companies or personal industries that are struggling to get more views and link popularity)
- Admin account (staff members who can access the backend and frontend of the software and edit the data)
- Helpya account (used by the owners of the company to create subscriptions for businesses)

3. User Interface

3.1 List of features

1. Consumers can search businesses by their name and/or a keyword

2. Ability to message a business with any inquiries
3. Purchase a service from the consumer's desired choice
4. Favourite certain business of the consumers choosing
5. Ability to create an account, either a consumer or business
6. Consumers can upgrade to a business account later on

3.2 USER INTERFACE DESCRIPTION

Brief description of the different interfaces the users will be able to see, however, this is already going to be done and we are only responsible for the back-end.

3.2.1 CHATTING AMONGST THE USERS

The chatting system will not be done in real-time but will be done through posting and receiving, like an email of sorts. The users will post a message, and when the receiving user logins and enters that chat they will be able to send, receive and see the message sent.

3.2.2 PAYMENT HISTORY

3.2.2.1 USER ACCOUNTS

Both the business and consumer user will have a history of the payments. Consumers will be able to see the previous purchases they've made, this will have things like, how much the purchase was, what purchase was made, what business it was purchased from, along with the date. These are the receipts for payments, whether you received or sent a payment.

3.2.2.2 ADMIN ACCOUNTS

For the purpose of statistics the admins will be able to see how many purchases are made from a business. This will have the service purchased, the business it was purchased from along with the date.

3.2.2.3 HELPYA ACCOUNTS

Helpya account will have no access to payment history. Its only purpose is to distribute subscriptions.

3.2.3 PROMOTING THE BUSINESS

Business users will be create ads to promote their business, this will be done on a subscription based setup. Also after an approval process, business are able to create and draft stories, that are temporary to help promote their business as well. [WILL ELABORATE MORE]

4. NON FUNCTIONAL REQUIREMENTS

4.1 USER PLATFORM

The platform for both consumers and business users will be the same upon login and account creation, however more information will be needed if the user is a business user.

4.2 ACCOUNT FUNCTIONALITIES

All users which inclues, consumers, business and admins can manipulate their account by creating, updating, viewing, and deleting information on their profile.

5. FUNCTIONAL REQUIREMENTS

5.1 PROCEEDING WITH A SERVICE AND PAYMENT

After the consumer and business message each other and agree on a purchase they can proceed with payment. [UPDATE MORE AFTER CLIENT MEETING]

5.1.1 BANK INFORMATION & CREDIT CARD INFORMATION

Consumers will be able to add credit card information to be able to make payments to business'. Business' will be able to add their bank information to receive payments they will also be able to add credit card information in the case they want to purchase a service from another business. All payment information will only be accessible to the user account it belongs to, admins will not be able to see this information.

5.2 SEARCHING AND FINDING A BUSINESS SERVICE

When searching for a business, a consumer needs to simply enter what service they want along with a keyword. They will then be shown multiple businesses that match the criteria they entered in. They will then have the option to proceed with payment right away or send a message to the business

5.3 ACCOUNT DELETION

All users will be able to delete their accounts. However, the account won't be permanently deleted. The account will be tombstoned to prevent further accounts being created using the same username or email.

5.4 SIGNING UP FOR AN ACCOUNT AND ACCOUNT LOGIN

When the user first enters the app they will be prompted to either log in or create an account. If the user already has an account they will be prompted to log in. Otherwise, they will be prompted to choose from either a consumer account or business account and then follow the instructions accordingly.

5.5 ACCOUNT RECOVERY AND FUNCTIONALITY

5.5.1 CONSUMER ACCOUNT

When a consumer is logged in they will have the capability to change their password, change their profile as well as delete or pause their account. During login they will have the ability to recover their password as well if it is forgotten.

5.5.2 BUSINESS ACCOUNT

When a business is logged in they will be able to do the same functions with their account as the consumers can but they will also have access to their ads. They will be able to create, see, delete and edit their ads. Business' will also be able to enter into a time-based subscription to increase ads.

5.5.3 ADMIN ACCOUNT

Admin accounts will allow access to user account management. The admin account will allow changes to the account itself. Admins will have the basic functions of an account such as changing their password, viewing their account, deleting information as well as updating. They will also have the capability to ban users, unban users, delete users and in some cases edit user accounts. Regular users will not be able to see an account that is an admin. Admins will also have the ability to refund subscription fees.

5.5.4 HELPYA ACCOUNT

There will also be an account specifically dedicated to subscriptions as well. The helpya account will control and create subscriptions and discounts business's can use on the app.

6. TECHNICAL REQUIREMENT

6.1 STORING BANKING INFORMATION

The banking information for both users will be stored in the database. The database is used to update and delete any banking information upon request.

6.2 COMMUNICATION WITH BACKEND

The API uses Node .js to handle requests from the front end database and the login authentication. The API will output a JSON which will be queried by the backend.

7. System Architecture

7.1 Database ER DIAGRAM

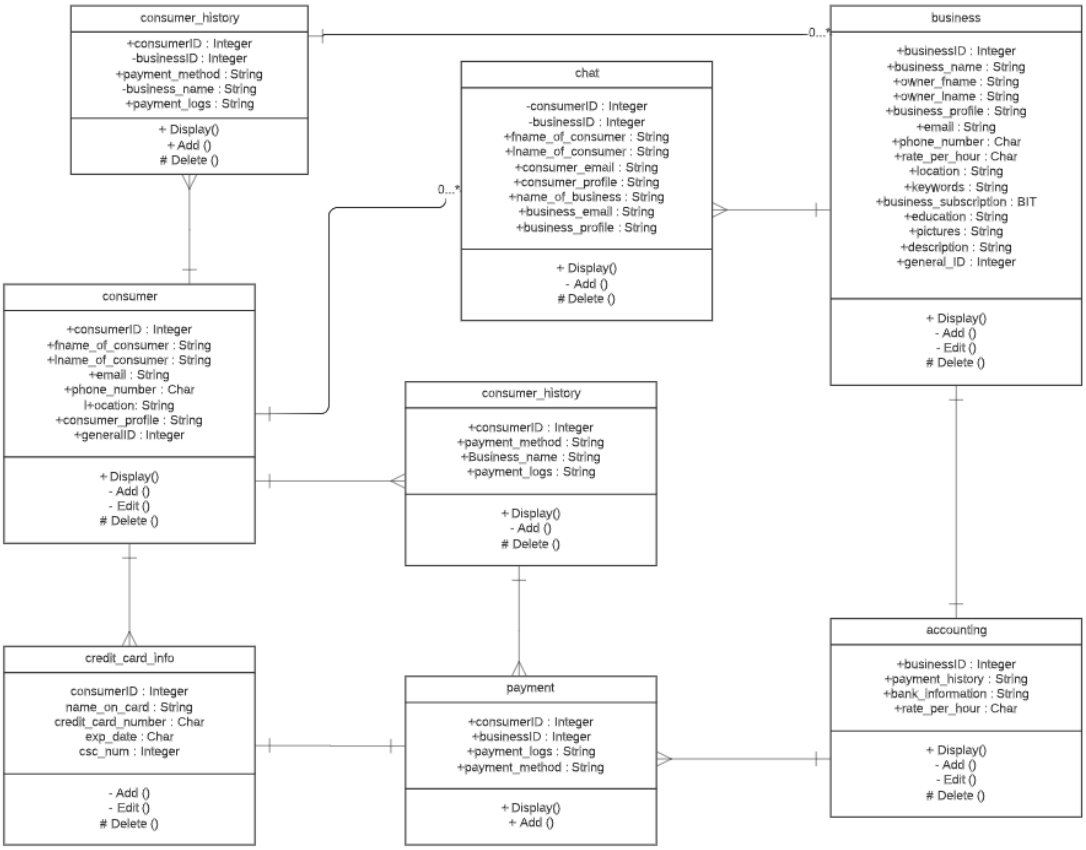
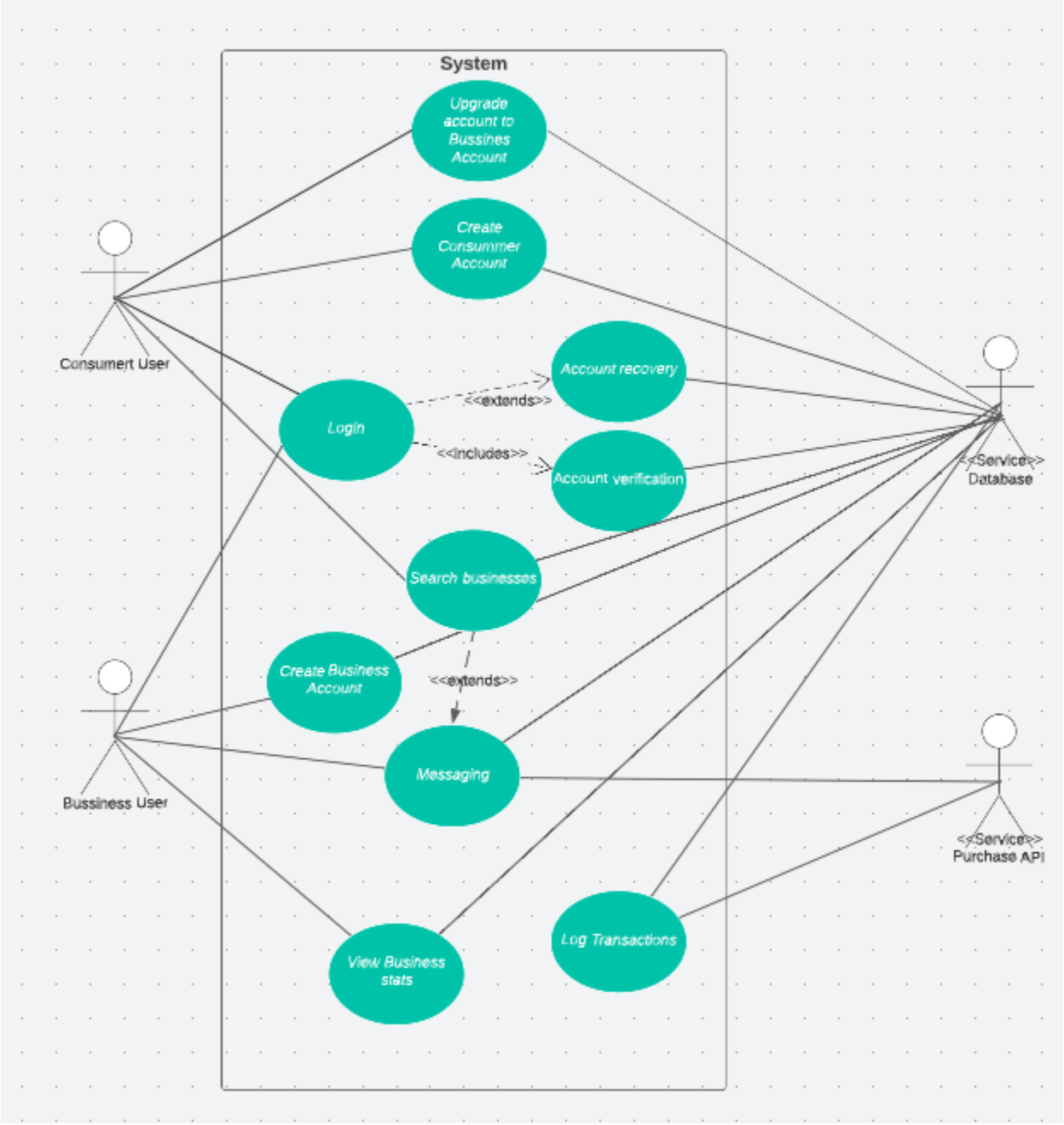


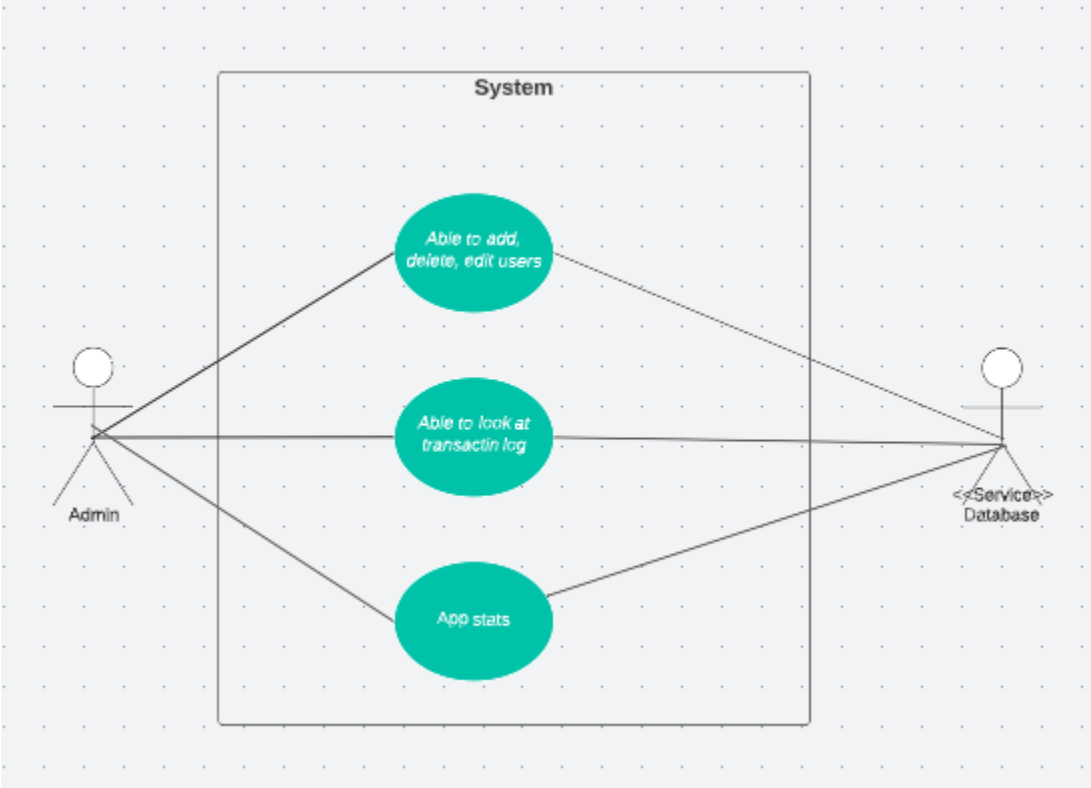
Figure 3.1.1 The following database ER diagram shows how data will be represented as in our relational schema.

7.2 USE Case Diagrams

7.2.1 User side USE Case Diagram

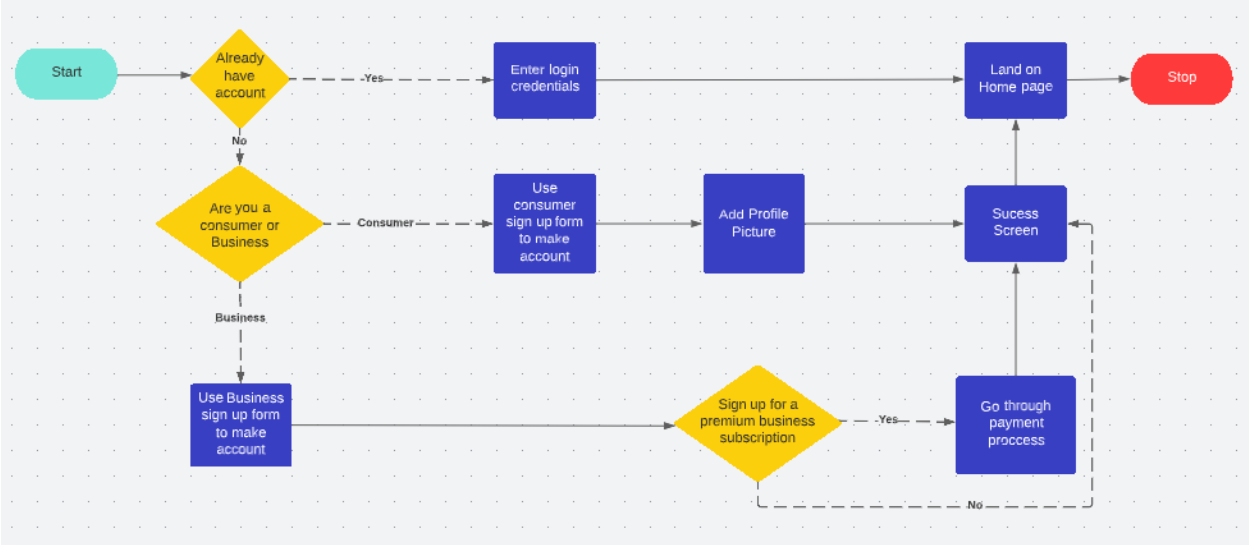


7.2.2 Admin side USE Case Diagram

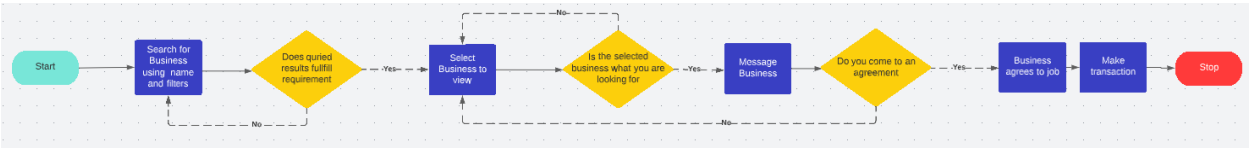


7.3 Data Flow Diagrams

7.3.1Creating an account



7.3.2 Purchasing a service



8. ENTITY ATTRIBUTE DESCRIPTION

8.1 Consumer

A consumer is one of the clients that can access the app. The consumer will need to create an account and provide the specified information, which includes their name, email, phone number, location, and password. Attributes like a profile picture can also be added later.

Attribute	Description
consumerID: INT **[PK]**	Unique identifier given to every consumer. Admins will be able to use this ID to find consumers.
fname_of_consumer: varchar(150)	Attribute holds first name of the consumer. Has a NOT NULL condition because businesses need a way of identifying the consumer.
lname_of_consumer: varchar(150)	Attribute holds the last name of the consumer. Has a NOT NULL condition because businesses need a way of identifying the consumer.
email: VARCHAR(150)	Consumer's email is used as a way of sending any notifications. Because the consumer needs a method of receiving these notifications the value can not be NULL. The attribute also needs to be UNIQUE so multiple accounts can not be created under the same email.
phone_number: CHAR(15)	Attribute will hold the phone numbers of the consumers. A Char value of 15 is given to include international numbers and area codes as well. The phone number will be accessible to the business only after an appointment has been made.
location: VARCHAR(50)	Attribute is used to filter the business within the consumer's general location. This value can not be NULL because businesses with the same location need to be shown to the consumer.
consumer_password: VARCHAR(50)	Attribute will hold the password the consumer sets for their account.

consumer_profile: varchar(250)	Attribute will hold an image link that will have a photo that can be used as a profile picture.
generalID: INT	Given to all users, this includes both consumers and businesses. Used for admins to identify general users of the app.

8.2 recent_searches

This table will be used so the consumer will be able to easily see the business they had recently viewed. They will be able to view things like the name of the business and the business's profile.

Attribute	Description
businessID: INT **[PK]**	Attribute is used to uniquely identify and show the business. In this instance, the attribute is used to save in the saved_searches table so clients can view the recently searched business.
consumerID: INT **[FK]**	Attribute holds the consumerID that the saved_searches is connected to.
store_name: VARCHAR(150)	Used to identify a business. Attribute won't be NULL because the value is needed to identify the business
store_profile: VARCHAR(250)	Profile picture attribute will hold the link to the image to be used to display the photo in the consumer's saved searches.

8.3 credit_card_info

Table will hold the credit card information of the consumer. The table will possess all the necessary information to process credit card payments.

Attribute	Description
consumerID: INT **[PK]**	Attribute is used here so the credit card can be associated with someone's account.
name_on_card: VARCHAR(150)	Attribute holds the name of the owner of the credit card. Needed to process the credit card payments so the value can not be NULL.
credit_card_number: CHAR(16)	Attribute holds the credit card number. Can not be NULL because the value is needed to

	process payments.
exp_date: CHAR(5)	Attribute holds the expiry date of the credit card. Needed to process payments - therefore values can't be NULL.
csc_num: INT	Attribute holds the CSC number, which is located at the back of the credit card. Value is needed to process payments, this value can not be NULL.

8.4 Payments

This table will be the connection between the consumer's credit card and the business's accounting table. This will possess the main information about the payment between the two clients.

Attribute	Description
consumerID: INT **[PK]**	consumerID in this table is used to track which consumer made a payment.
businessID: INT **[PK]**	Attribute in the table is used to identify which business is receiving payment.
Payment_logs: VARCHAR(250)	This attribute records all information about the payment, ie: time was made and amount of payment
payment_method: VARCHAR(250)	Value holds what method of payment the consumer used

8.5 Business

A business is one of the clients that can access the app. The consumer will need to create an account and provide the specified information, which includes the owner's name, business email, business phone number, location, password, business name, the rate they charge, keywords, education level, pictures and a general description. Attributes like a profile picture can also be added later.

Attribute	Description
businessID: INT **[PK]**	Unique identifier given to every consumer. Admins will be able to use this ID to find consumers.
business_name: VARCHAR(150)	Attribute holds the name of the business, which can be viewed by anyone on the app. Value can not be NULL because it is needed for users to identify.
owner_fname: VARCHAR(150)	Attribute holds the first name of the business owner(s). Provides information for other users to say who owns the business.
owner_lname: VARCHAR(150)	Attribute holds the last name of the business owner(s). Provides information for other users to say who owns the business.
business_password: VARCHAR(50)	Attribute will hold the password the business sets for their account.
business_subscription: BIT	This attribute is a value that returns true or false depending on if the business wants to purchase a premium account.
business_profile: VARCHAR(250)	Attribute will hold an image link that will have a photo that can be used as a profile picture.
email: VARCHAR(150)	business email is used as a way of sending any notifications. Because the business needs a method of receiving these notifications the value can not be NULL. The attribute also needs to be UNIQUE so multiple accounts can not be created under the same email.
phone_number: CHAR(11)	Attribute will hold the phone numbers of the business. A Char value of 15 is given to include international numbers and area codes as well. The phone number will be accessible to the business only after an appointment has been made.
rate_per_hour: CHAR(10)	Value holds the rate per hour, set by the business, that they are charging consumers for

	their services. Value can not be NULL because consumers need to know the view before proceeding.
location: VARCHAR(50)	Attribute is used to find consumers in the same areas as the business.
keywords: VARCHAR(50)	Keywords used to attach to the business. These could be subjects or areas the business relates to. Consumers can use certain keywords to find business concerning a specific term.
education: VARCHAR(150)	This attribute holds the education level of the business. This could be the certifications they have, qualifications they have or any professional documents they have received.
pictures: VARCHAR(500)	This attribute enables businesses to add photos to their account that are associated with their business. This can include examples of work done or any information where pictures are better than words.
general_ID: INT	Given to all users, this includes both consumers and businesses. Used for admins to identify general users of the app.
description VARCHAR(500):	This attribute holds the information of the business. Here, the business can enter anything about their business that consumers can see.

8.6 consumer_history

This table contains information used and seen by only the consumer and admins. Has information on their previous purchases. Contains things like which business they bought from as well as the payment logs.

Attribute	Description
consumerID: INT	consumerID in this table is used to identify which consumer history is being viewed.
businessID: INT	Attribute in the table is used to identify which business is received payment.

payment_logs: VARCHAR(250)	This attribute records all information about the payment, ie: the time it was made and the amount of payment.
business_name: VARCHAR(150)	This holds the name of the business the consumer has purchased from.
payment_method: VARCHAR(50)	Value holds what method of payment the consumer used.

8.7 Chat

This table will contain information about the chatting system between the consumer and business. The messaging system will post a message and then receive a message. This is the method to use for communication between the two clients. The clients will be able to see very basic information about each other in the chat window.

Attribute	**Description**
consumerID: INT	Unique identifier to see which clientID is talking to the business.
businessID: INT	Unique identifier to see which clientID the business is talking to.
Fname_of_consumer: VARCHAR(150)	Attribute holds the first name of the consumer. Visible to the business when they are talking to the consumer.
Lname_of_consumer: VARCHAR(150)	Attribute holds the last name of the consumer. Visible to the business when they are talking to the consumer.
Consumer_email: VARCHAR(150)	Consumer's email is used as a way of sending any notifications, in this scenario, messages.
business_name: VARCHAR(150)	Attribute holds the name of the business, which can be viewed by the consumer whom they are talking with.
Business_email: VARCHAR(150)	business email is used as a way of sending any notifications, in this scenario, messages.
consumer_profile: VARCHAR(250)	Attribute will hold an image link that will have a photo that can be used as a profile

	picture. This will be viewable in the chat window for the business.
Business_profile: VARCHAR(250)	Attribute will hold an image link that will have a photo that can be used as a profile picture. This will be viewable in the chat window for the consumer.

8.8 Accounting

This contains information available to only the business and admins. Will contain the payment history the business receipts, account information for them to receive payment, as well the hourly rate they charge

Attribute	Description
businessID: INT **[PK]**	In this scenario, this attribute is used to identify which business accounting information you are looking at and which accounting information it is associated with.
payment_history: VARCHAR(150):	This attribute is available to the business to see the history of payments they have received and from who (ie a name).
bank_information: VARCHAR(150)	This attribute is used to receive payment on the business side. Businesses will need to enter the required details to be able to receive payment from consumers.
rate_per_hour: CHAR(10)	Value holds the rate per hour, set by the business, that they are charging consumers for their services.

9. Tech Stack

1. MySQL: Database Development
2. Node.js: API development
3. Express: Node.js library primarily used to create login authentication system
4. Stripe: Purchasing API
5. Mocha: Unit Tests

10. Test Plan

10.1 Scope

The purpose of this test plan is to ensure we meet all of our functional, non-functional, technical, and user requirements without the introduction of bugs or issues. Our overall test plan will consist of unit testing, code review, performance testing, and bug fixing. Testing is critical to the successful completion of this project and will be a continuous process throughout its duration.

10.2 Process

Testing will be integrated into the development process of the service. We will begin by developing the actual service. Once completed we will create a minimum of three unit tests to check and service behaviours

1. To check the expected output with expected input
2. To check the behaviour with unexpected input
3. To check the behaviour with nonexistent input

Other unit tests will be created on a case-by-case basis. Once the system passes all unit tests it will be handed over to the integration manager for code review. The integration manager will check the code to see if the code can be made more efficient is bug-free and would not break or collide with other systems. Once completed the system will be added to a developer build and its test cases added to the test cases for said build and ran. If all test cases pass then the system is deemed tested and operational.

10.3 Testing environment

The testing environment will be all Windows 10 machines running the latest version of VS Code. A local server will be used to test code on the machine it is being developed before it is handed to the integration manager for further testing. All unit tests will be written in mocha.