**Car Rental Software Requirements Specification Document**

Author: Alexander Zucker

Delivered September 27, 2024

**Section 1: Document introduction and system overview**

**1.1: Document contents**

This document will enumerate all requirement specifications pertaining to the software system ordered by Hertz Global Holdings, Inc (hereafter referred to as “client”). This encompasses the following categories of information:

* Section 1 – Introduction ……………………………………………………………………………………… 1
  + 1.1 – Document contents ………………………………………………………………………… 1
  + 1.2 – System overview ……...…………………………………………………………………….. 1
* Section 2 – User requirements ……………………………………………………………………………. 2
  + 2.1 – General user requirements …………………………………………………………….. 2
  + 2.2 – Customer user requirements ………………………………………………………….. 3
  + 2.3 – Employee user requirements ………………………………………………………….. 3
* Section 3 – System requirements ……………………………………………………………………….. 3
  + 3.1 – Functional requirements ………………………………………………………………… 3
  + 3.2 – Nonfunctional requirements …………………………………………………………… 12
* Section 4 – Miscellaneous information ……………………………………………………………..... 14
  + 4.1 – Data security risk assessment …………………………………………………………. 14

**1.2: System overview**

The client has ordered the creation of a software system that will be used to manage their rental car inventory. Due to the client’s status as a major international car rental corporation, the scale of their operations has rendered the use of physical documents and records impractical. Therefore, the aim of this software system is to effectively digitize all the client’s day-to-day operations.

As this software will serve as a digital representation of the company’s business, both the client’s customers and employees will be interacting with it extensively. All actions required for a customer to rent a vehicle will be a functional component of this software system. Similarly, the workload of the client’s non-executive employees will now almost exclusively be performed by interfacing with this software.

Successful design and implementation of this software system will both dramatically speed up and improve the ease of the customer experience. Likewise, deployment of this software can be expected to cause a notable increase in the efficiency of the client’s employees. The resulting reduction in time taken to perform internal logistical and managerial tasks is projected to reduce employee workload by several hours per week.

**Section 2: User requirements**

This section will describe the individual functions of the software system at a high level. Functions that are specific to either customers or employees are separated into their respective subsections.

**2.1: General user requirements**

* The software system should be accessible by users via iOS and Android mobile applications, as well as web browsers.
  + The system must only ensure support for the latest versions of iOS and Android at the time of development.
  + The system must also ensure support for the latest versions of all major public web browsers.
* User interface details were not specified by the client and are therefore left to the discretion of the UI/UX implementation team.
  + No additional localization is required by the client; both mobile application and web browser versions of the software system need only support the English language.
* The use of a specific programming language was not mandated by the client. This decision is therefore left to the discretion of project management, provided the software system can run on the hardware previously specified by the client.
* The software system must provide all users, both customers and employees, with a method of registering a user account.
  + The account registration system must support opt-in two-factor authentication methods.
* The software system must be capable of interfacing with major electronic payment processors to handle customer billing.

**2.2: Customer user requirements**

Customers utilizing the software system are expected to be able to perform the following general operations:

* View available rental vehicles.
* Select and rent an available vehicle.
* View a history of previously rented vehicles.
* Configure GPS services for a currently rented vehicle.
* Add, update, and remove payment methods.
* Make payments for any balance assessed to their user account.
* View a history of previous payments.
* View and sign contracts relevant to the vehicle renting process.
* Upload proof of age verification in the form of an image of a government issued id or other legal identification.

**2.3: Employee user requirements**

Employees utilizing the software system are expected to be able to perform the following general operations:

* View the status of customers’ current vehicle rentals.
* View customers’ rental history.
* View rental contracts associated with customers’ current or previous vehicle rentals.
* Manage employee-only notes attached to a customer’s account.
* View the current rental vehicle inventory.
* Modify the rental cost of vehicles in the inventory.
* Add or remove available vehicles from the inventory.
* Submit maintenance orders for vehicles in the inventory.

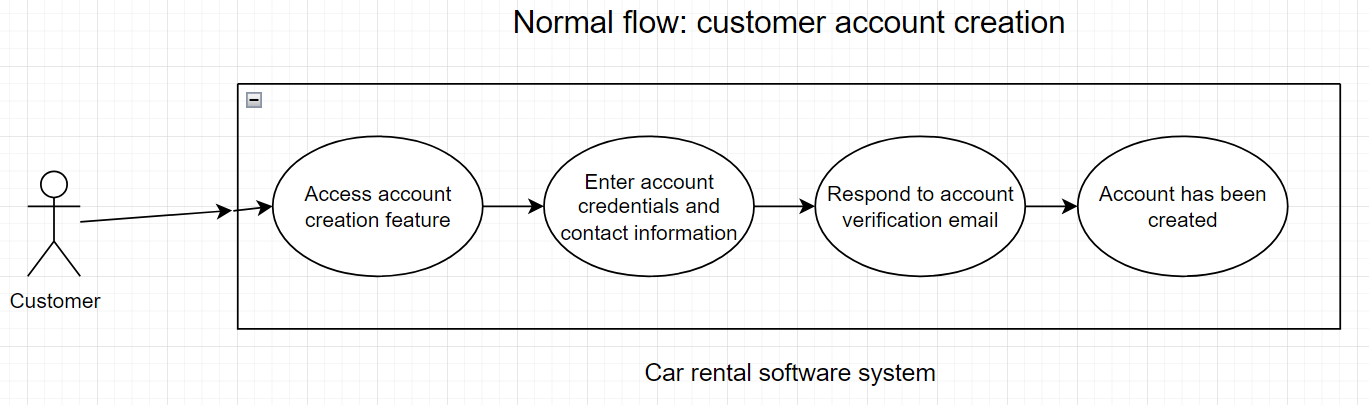
**Section 3: System requirements**

**3.1 Functional requirements**

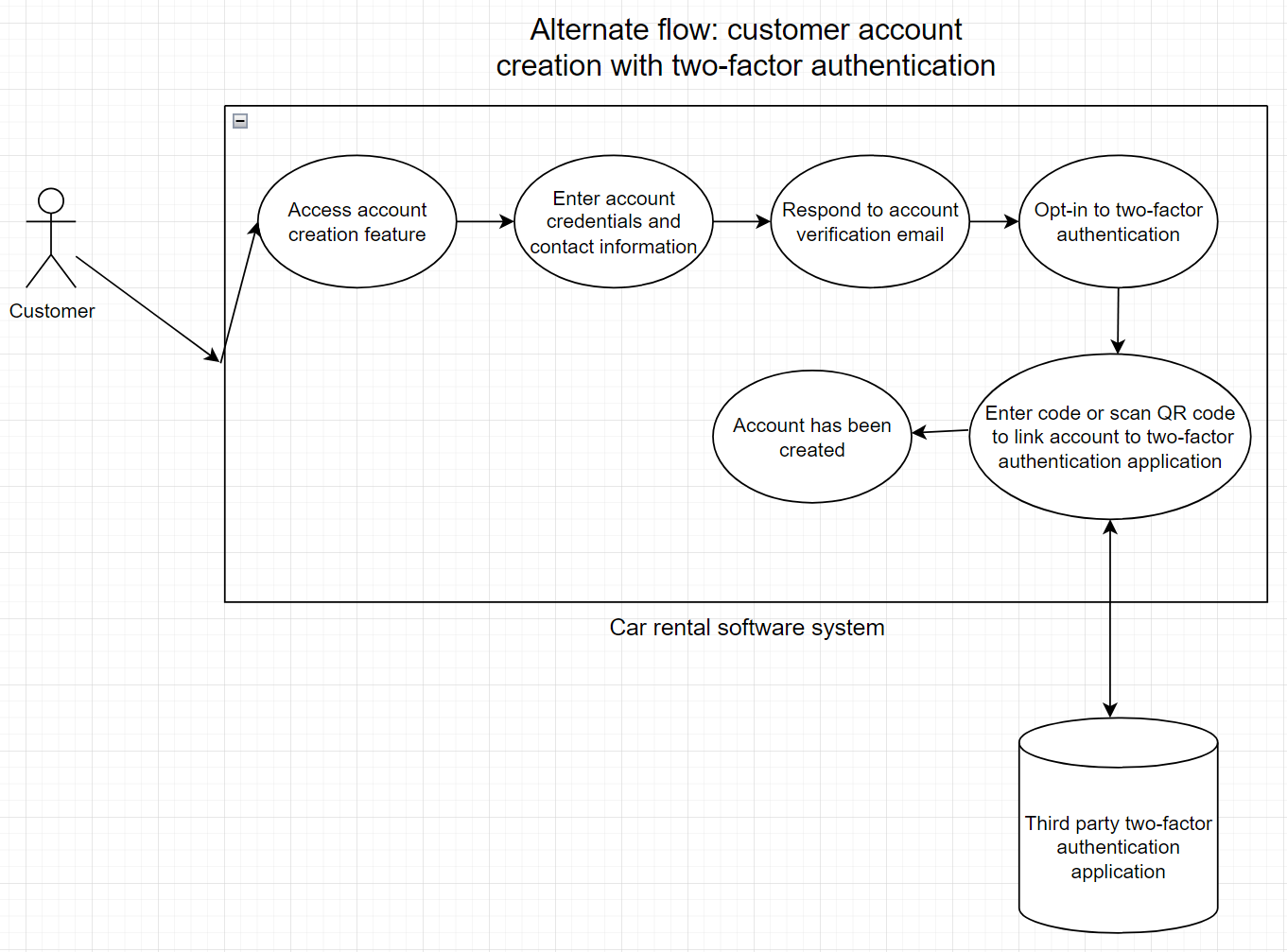
This section contains specific descriptions of all system functions. Each function will also be accompanied by at least one use case diagram which illustrates how the function interacts with the software system, as well as any other users or external systems. Some functions will be accompanied by multiple use case diagrams which depict alternate or exceptional flows.

**Function 1: User account creation**

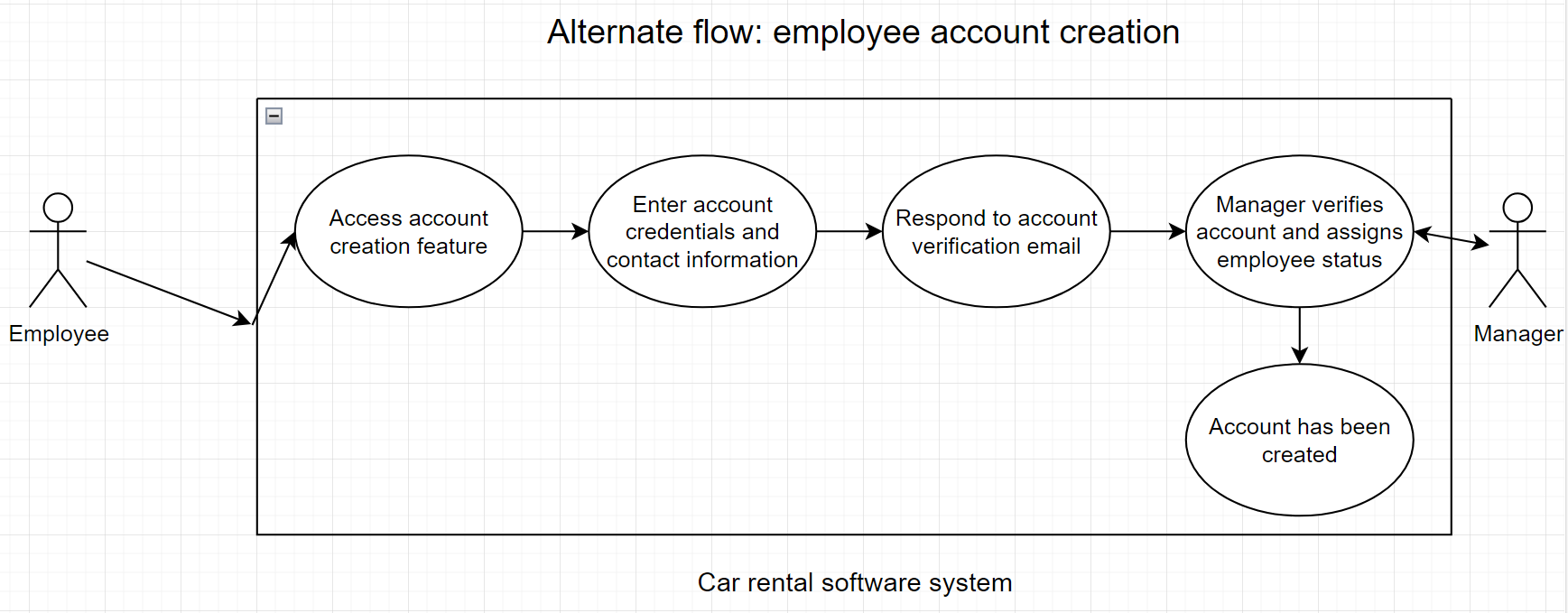
Both customers and employees of the client will have the ability to register a user account with the software system. The signup process will differ depending on whether the user is a customer or employee, as well as whether two-factor authentication is being utilized. These divergences will be illustrated via alternate flow use case diagrams.



This use case depicts the most simplistic version of the account creation process, in which the user is a customer and does not opt-in to two-factor authentication.



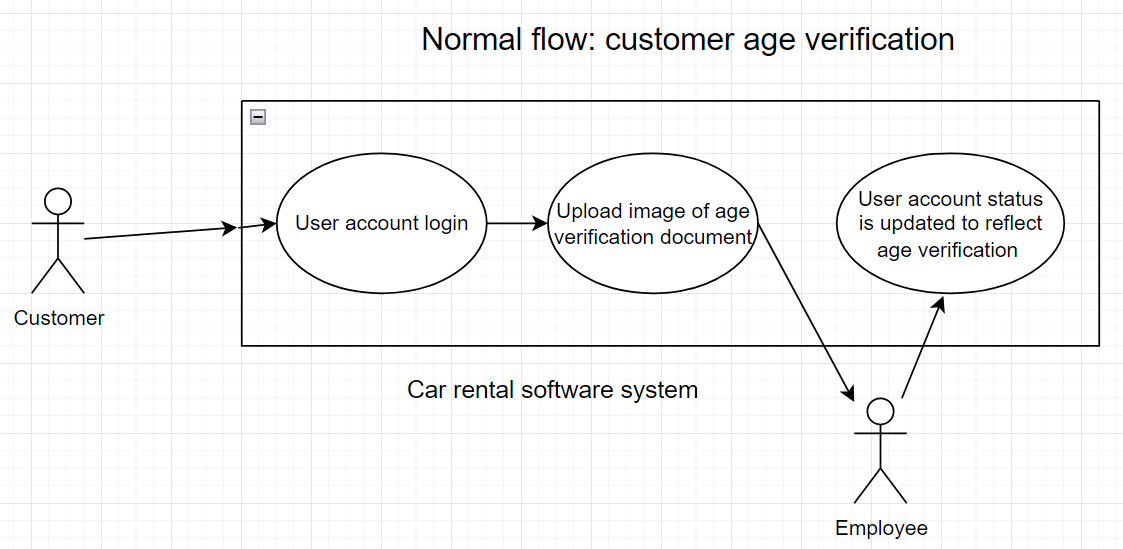
This use case depicts an alternate sequence of the customer account creation process, in which the customer decides to utilize two-factor authentication. Numerous two-factor authentication methods are supported such as Google Authenticator, Authy, or SMS messaging, but the diagram simply abstracts the application and does not distinguish between them.



This use case depicts the process of an employee’s account creation. As employee system accounts are granted special user permissions, a manager is required to validate any employee account creation requests. Upon successful validation, the newly created account will be assigned the appropriate permissions required by the employee.

**Function 2: Age verification process**

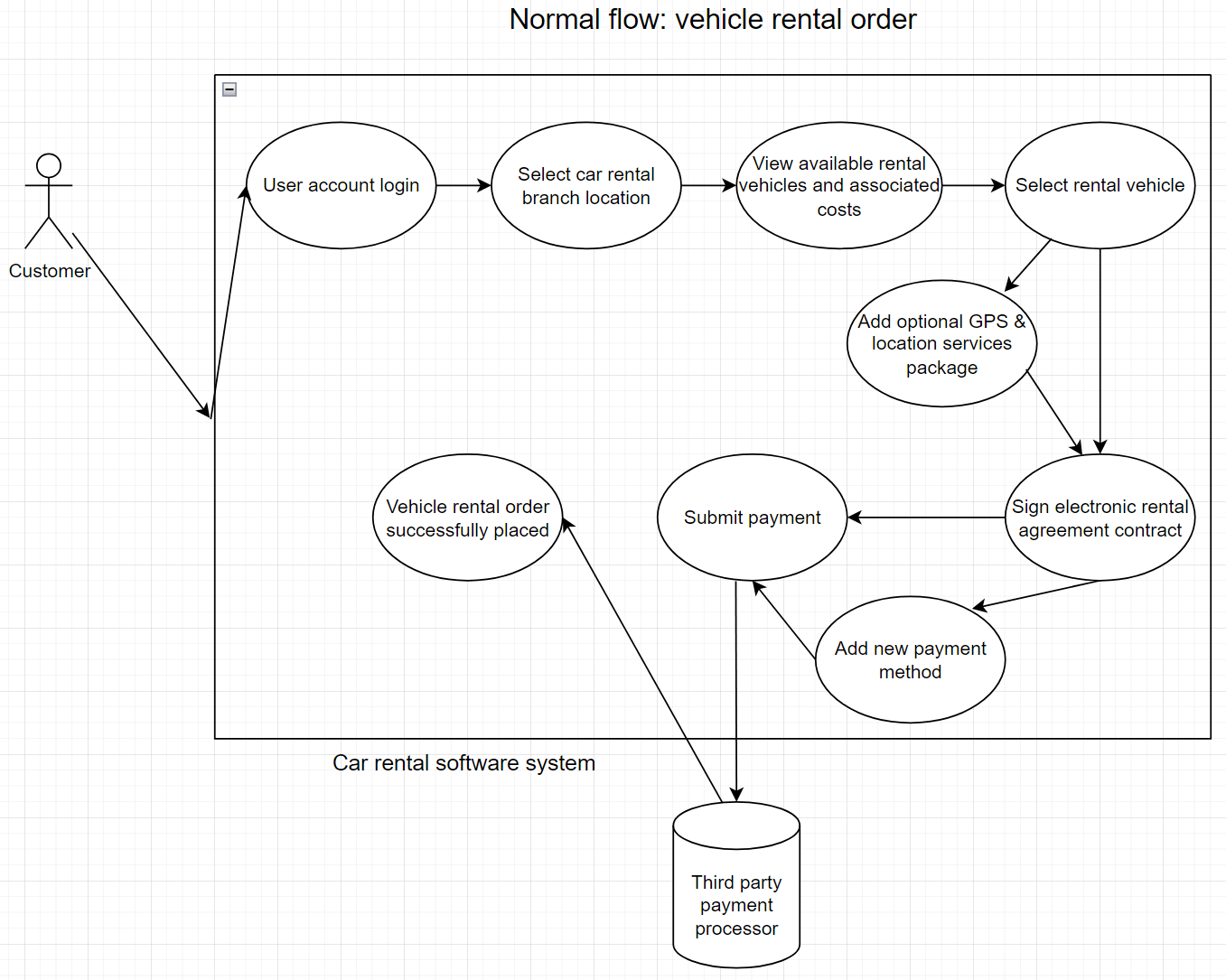
As the client will only rent vehicles to customers age 25 or older, the software system must contain a function to verify customers’ age. This will be performed by having the customer upload an image of a government issued id or other legal documentation, which will then be verified by an employee.



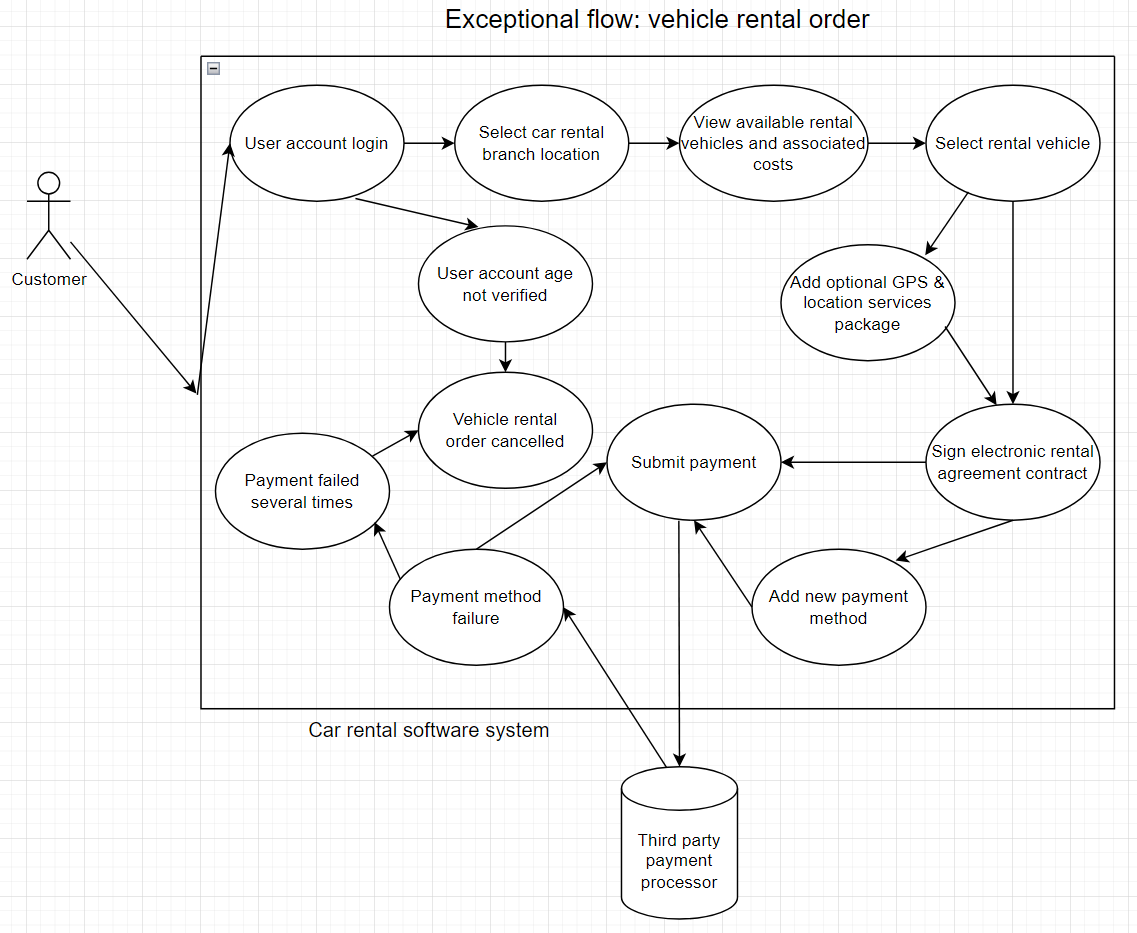
This use case depicts the successful verification of a customer’s age. As the verification process is manual and requires an employee to interact with the system, the time it takes to complete the verification request will vary depending on employee availability. Once verification occurs, a corresponding status will be applied to the customer’s account, allowing them to now begin the vehicle rental process.

**Function 3: Vehicle rental process**

Given the client’s status as a vehicle rental company, the primary function of this software system is to enable customers to electronically submit vehicle rental orders. As age verification and valid payment submission are requirements to successfully complete an order, events leading to an order cancellation will be illustrated via an additional exceptional flow diagram.



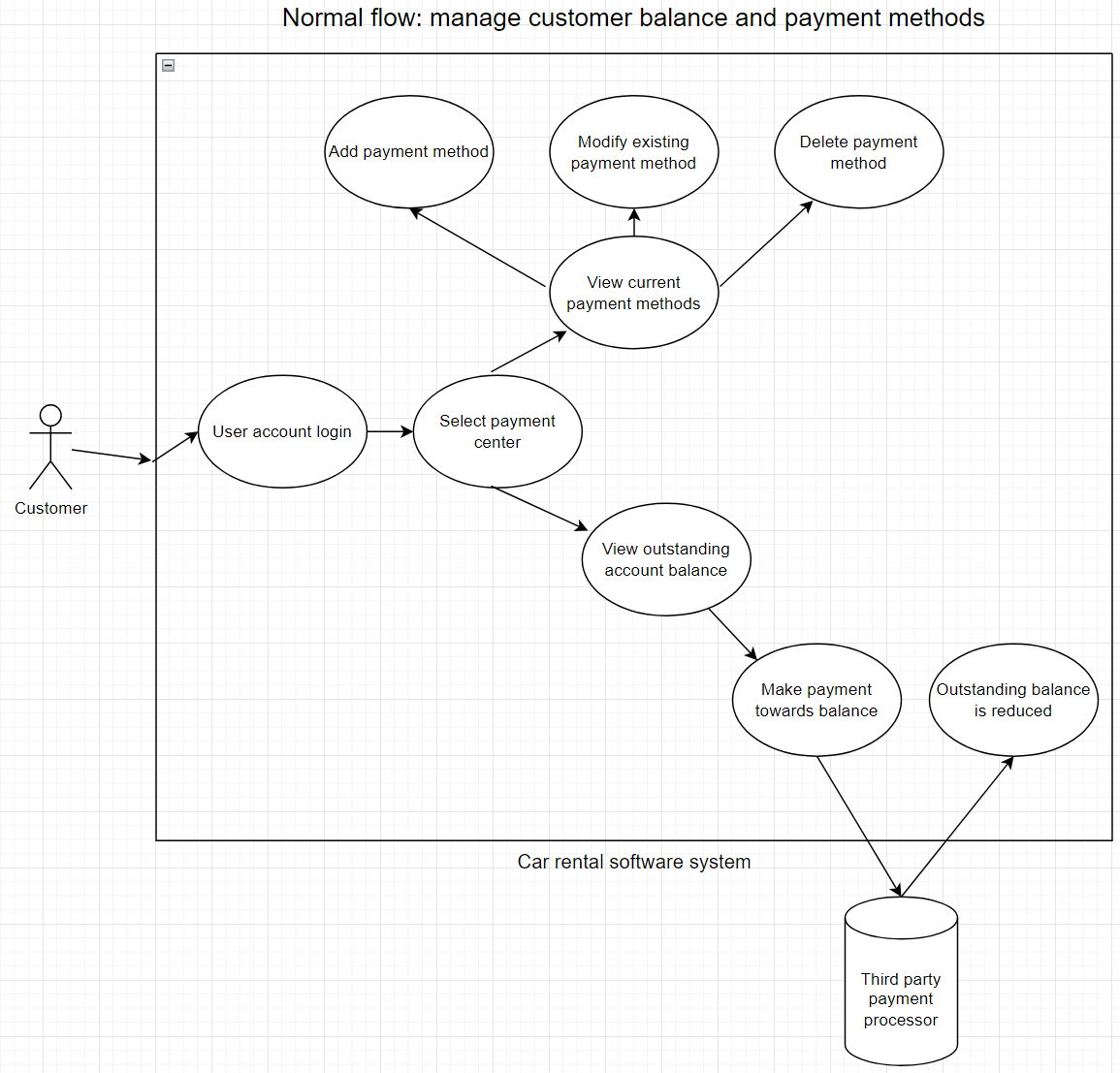
This use case diagram depicts the vehicle rental order process from start to finish. As the customer can opt to purchase additional GPS services to be included with their rental vehicle, this is represented by a branching flow in the diagram. Similarly, if the customer needs to add a new payment method in order to complete their purchase, they will be prompted to do so during this process. As is the case with nearly all online purchases, payments will be handled through a third party payment processor, and the software system will simply interface with their API.



This use case diagram depicts two of the conditions that may lead to the cancellation of a customer’s rental vehicle order. Should a customer attempt to initiate the vehicle rental process before having their age validated, the majority of the function’s logic is skipped and the customer is informed that they must upload age verification documents before proceeding further. In the event of a payment failure, the customer is given the opportunity to try again using a different method of payment. Should payment continue to fail, the customer’s order will be cancelled.

**Function 4: Manage account balance and payment methods**

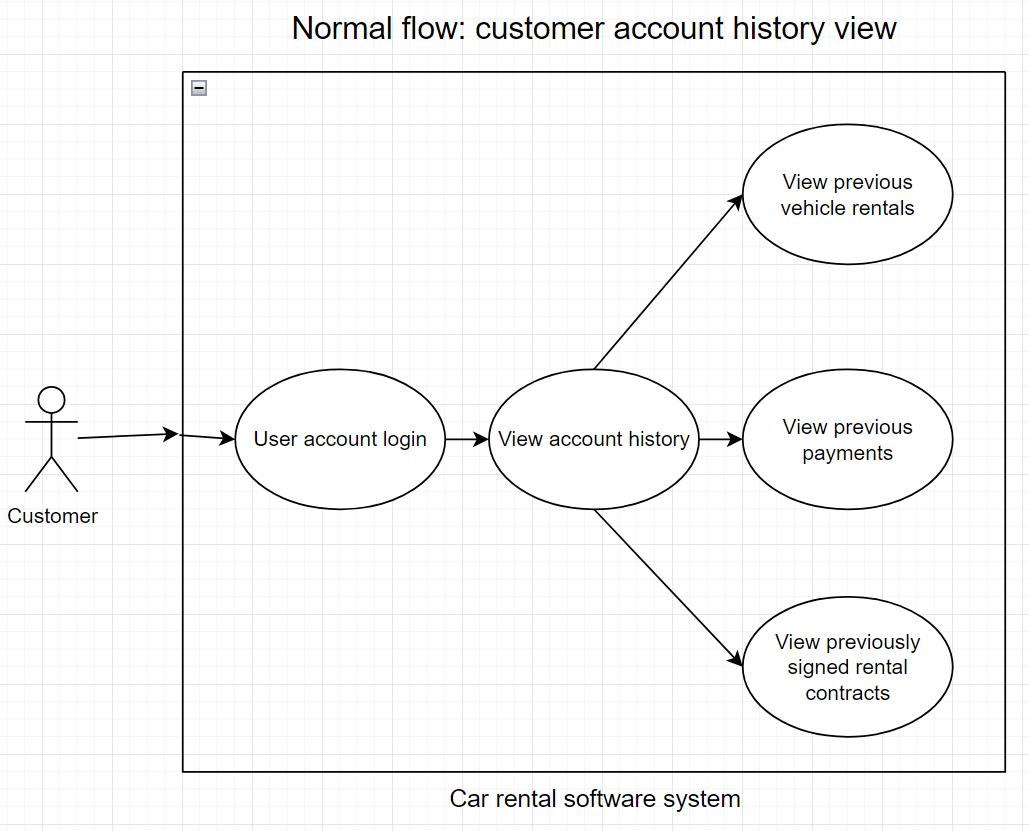
In addition to making a payment during the vehicle rental process, customers will have the ability to view and pay any outstanding fees that have been assessed to their account. This function will also provide customers with the ability to add, update, or remove payment methods from their user account.



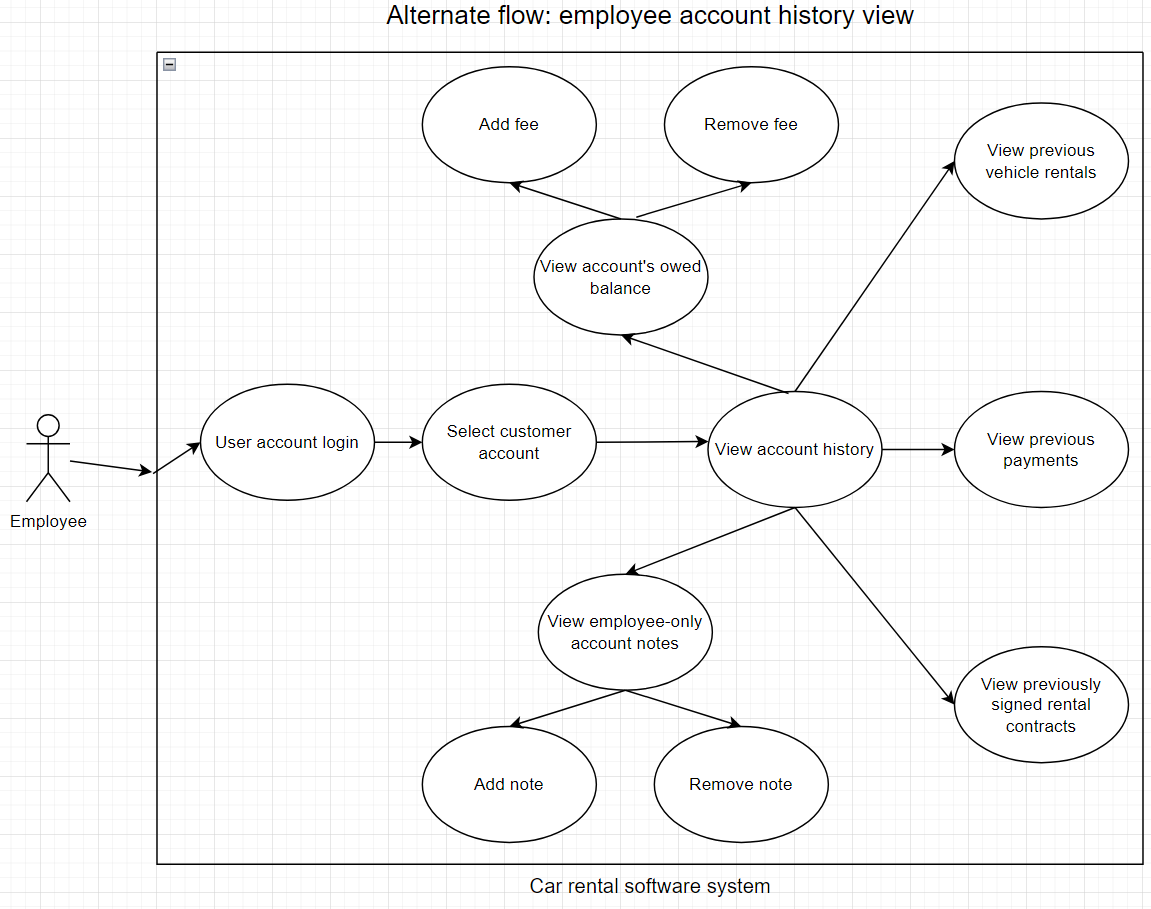
This use case diagram depicts the two major functions available in the customer’s payment center. As with payments made during the vehicle rental process, any payments submitted towards and account’s balance will be handled through a third party payment processor. Valid payment methods that can be linked to a customer’s account include debit cards, credit cards, and bank accounts.

**Function 5: View customer account history**

Both customers and employees will be able to utilize the system to view a customer’s account history. This history will consist of previously rented vehicles, any previous payments, and any previously signed legal contracts. As employees will have access to additional components of an account’s history, the divergences between the customer and employee view will be illustrated via an alternate flow diagram.



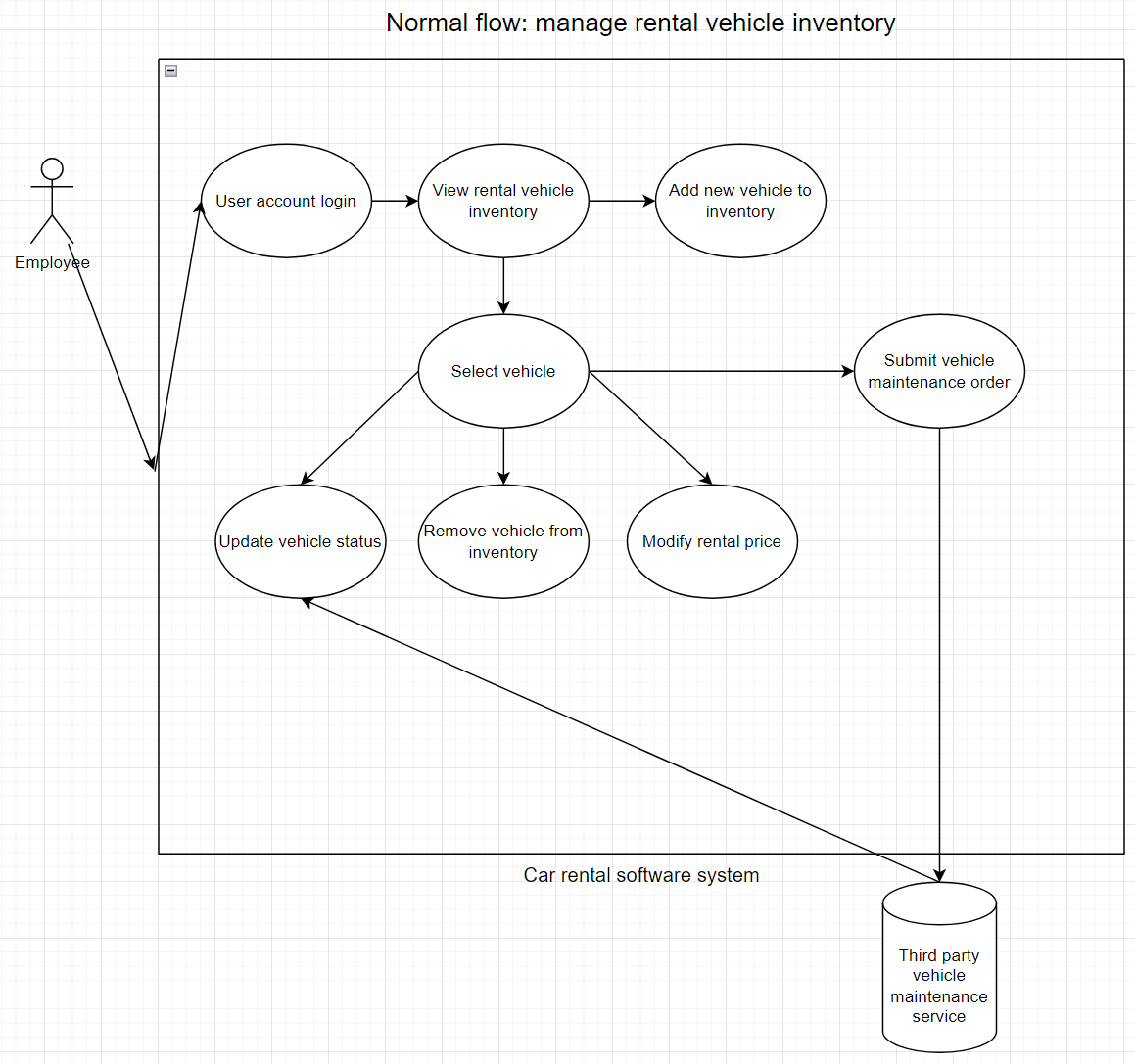
This use case diagram depicts the customer version of the view account history function. Given the user’s status as a customer, they are only capable of viewing the history for their individual account. After logging in and accessing the view history feature, they can select which of the three types of previous records to browse.



This use case diagram depicts the employee version of the view account history function. As the user is an employee, they can view the account history of any customer. In addition to the functionality included in the customer version of this function, employees have the ability to view and modify account notes and balances. Account notes are simple messages that can only be viewed by employees, and are used to document additional information not already covered by an account’s default data fields. Account balances represent any outstanding monetary amounts owed by a client. Employees have the ability to modify this amount by adding or removing fees, such as those that may be incurred when returning a rental vehicle in a damaged state.

**Function 6: Manage rental vehicle inventory**

Employees will have the ability to view and manage the inventory of rental vehicles possessed by the client. Managing the inventory consists of the ability to add new vehicles, remove current vehicles, update vehicle status, modify rental prices, and submit vehicle maintenance orders.



This diagram depicts all features available within the inventory management function. In addition to manually updating a vehicle’s status to reflect damage or performance problems, employees can utilize the inventory management function to submit a maintenance request for a vehicle. As the client does not specialize in vehicular maintenance, these orders are contracted to a third party vehicle maintenance service. After information about the maintenance request is filled out by an employee, the software system will utilize the maintenance provider’s API to automatically file an order within their external system. Once this order is completed, the resulting repairs or modifications made to the vehicle will be reflected in the vehicle’s status.

**3.2: Nonfunctional requirements**

This section will expand on the more nebulous requirements specified by the client. While these requirements do not constitute a tangible function like those detailed in section 3.1, they are no less important to the successful implementation of the software system. Due to their intangible nature, these nonfunctional requirements will hereafter be referred to as “qualities” instead of “functions”.

**Quality 1: Data security**

As is the case with any software system that involves the creation of user accounts, the security of any and all sensitive data is of the upmost priority. For the purposes of this system, “sensitive” data refers to:

* User account credentials
  + Emails, passwords, and phone numbers
* Billing information
  + Credit and debit card information, bank account details
* Signed rental contracts

In order to protect user account credentials, passwords will only be stored in databases as salted one-way hashes using an industry standard hashing algorithm. As effective brute-force resistant hashing algorithms require a significant amount of computational power, it is inevitable that their implementation will lead to a slowdown in database read-write times. Within the confines of modern technology there is no realistic way to avoid this slowdown; therefore, a slightly slower read-write time is deemed an acceptable cost to ensure the security of user account credentials.

While billing information such as credit card, debit card, and bank account numbers will also be hashed prior to storage, the predictable numerical nature of these strings poses an inherent security risk. An attacker, for instance, knows that the typical Visa credit card number is 16 digits long. This dramatically increases the effectiveness of brute force attacks targeting payment credentials. To mitigate this risk, all payment information will be stored in a separate database which will only allow read-write requests from either the customer billing center, or a third party payment processor’s API. Implementation of this secured database should not adversely affect system performance speeds, but the requisite hardware will result in additional overhead costs.

Lastly, as rental contracts are sensitive, legally binding documents, they must also be stored in a secure manner. Due to their lengthy nature however, the same methods used to secure user account credentials and billing information cannot be effectively applied to rental contracts. As such, contracts will be encrypted prior to database storage via the joint use of symmetric and asymmetric encryption algorithms. Provided that these cryptography techniques are implemented properly (e.g. not solely relying on an asymmetrical algorithm due to the slow speed of encrypting large amounts of data), there should be no adverse effects on the software system at large.

**Quality 2: Customer privacy**

Whilst data security constitutes a large portion of what defines customer privacy, these two concepts do diverge within the software system. In order to protect the privacy of customers using the software system, it is not enough to simply hash or encrypt data; that data must also only be exposed to aspects of the software system that require access to it.

In order to observe this practice of data encapsulation, access to customer information must be restricted to employee user accounts of the relevant type. For instance, customer service representatives require the ability to view individual customer account histories, but have no reason to require database-level access to *all* customer account histories.

Additionally, logs should be kept of any employee access or modification of customer account information. This will result in a slight reduction of the software system’s performance speed, as well as a larger database storage space requirements.

**Quality 3: System reliability and redundancy**

Given the fact that the vast majority of the client’s customers are traveling on vacations or business trips, time-sensitive delivery of customers’ rental orders is critical. Should the software system experience an outage, customers may be left without their only means of securing transportation.

To prevent this, all data stored by the software system should be backed up at regular intervals. In order to ensure there is as little risk of data loss as possible, the 3-2-1 backup should be observed: data should be backed up by multiple services utilizing different storage methods at different locations, thereby all but eliminating the chance of all data storage systems becoming inaccessible simultaneously.

Performing regular backups on this scale will result in a large amount of database read requests. At this can result in a slowdown of the software system’s performance, backups should be scheduled to occur during times of minimal customer activity.

**Section 4: Miscellaneous information**

**4.1: Data security risk assessment**

Whilst section 3.2 outlines the security benefits of currently effective cryptography practices, it is imperative that any long-term software system considers the risk posed by technological advances that may be made in the future. One such potential risk stems from advances being made in the field of quantum computing.

Where classical computers use binary bits to represent information, quantum computers utilize quantum bits (qubits), which exist in multiple states at once via the principle of superposition. The nature of qubits allows quantum computers to execute specific types of algorithms with a time complexity that is exponentially less than any classical computer.

The potential security risk posed by quantum computers stems from a specific task that qubits excel in performing: the factorization of prime numbers. As modern encryption algorithms such as RSA revolve around the multiplication of massive prime numbers, it is theorized that sufficient advances in quantum computing could lead to efficient methods of defeating these cryptography techniques.

As experts in the fields of cryptography and quantum computing generally agree that modern encryption algorithms will not be rendered ineffective for numerous decades, this is not a critical imminent threat. However, cryptographers are already hard at work developing encryption algorithms that are resistant to the advantages of quantum computing. As it will likely be years before this endeavor produces high quality, reliable algorithms, this software system should still be designed using current industry standard encryption techniques. However, special care should be taken to implement this software system such a way that will allow pivoting to new quantum-resistant encryption algorithms in the future.