DESOFS 2024 M1A Group 2 - Phase 1

Members

- 1190326 - Afonso Machado
- 1190535 - Domingos Machado
- 1230201 - Nuno Ribeiro
- 1230211 - Rui Neto
- 1230212 - Simão Santos

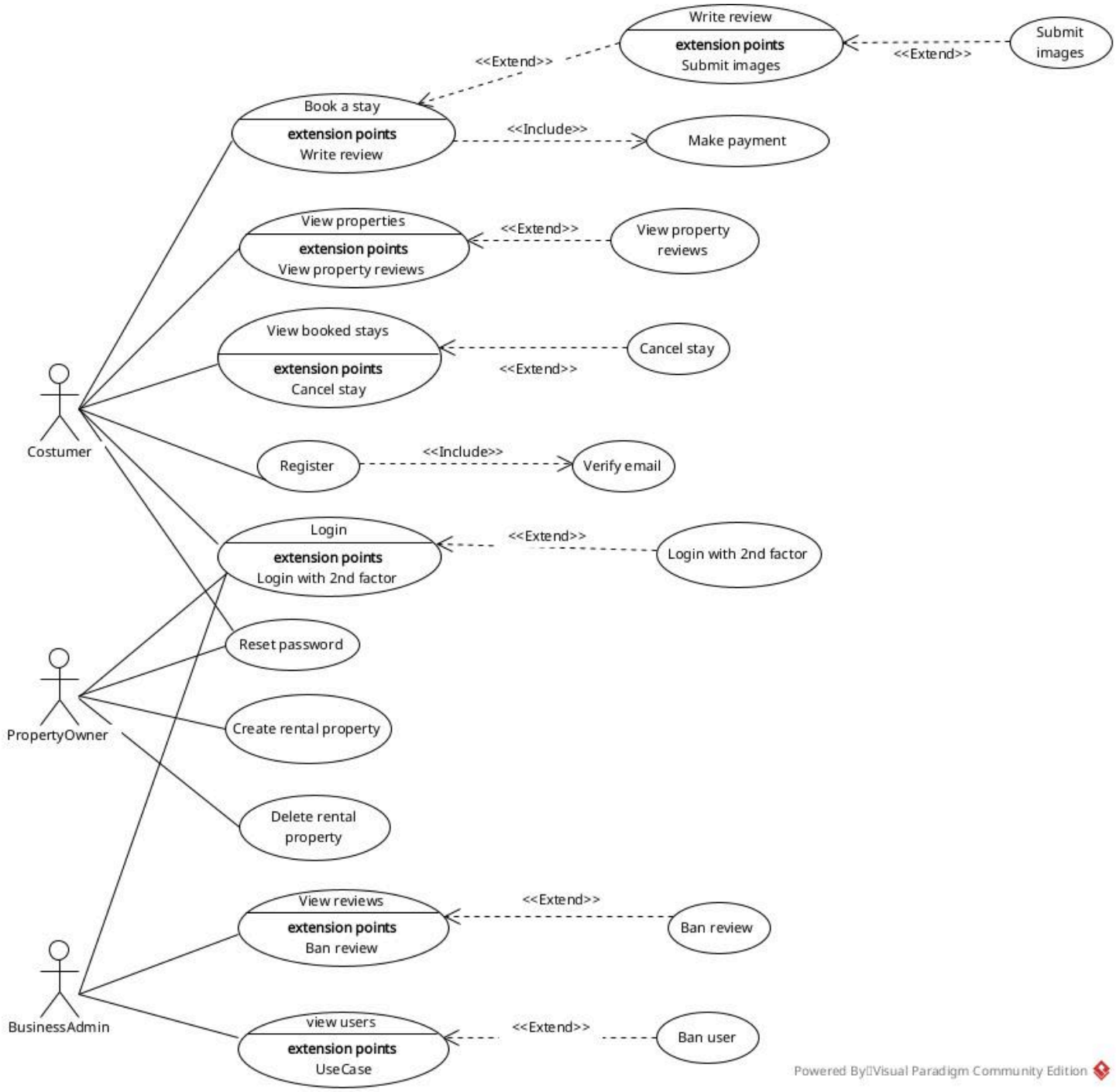
1. SSDLC

1.1 Analysis (Requirements gathering)

1.1.1 Use cases

TripNau is a web application similar to airbnb and booking that aims to ease the process of booking properties for
whatever purpose the customer wants, such as vacations, work, etc. Basically, property owners can easily list their
properties on the platform, setting nightly rates for specific time frames and providing detailed information such as
location, number of bedrooms, bathrooms, and amenities. Then, costumers will be able to scroll through a list of properties and will have the option to perform bookings for specific time intervals whenever these are
available. Moreover, the platform facilitates post-stay reviews, allowing users to provide feedback to property owners and
offering insights to potential guests before making a booking.

The following diagram showcases the use cases of the TripNau system.



1.1.2 Security related requirements

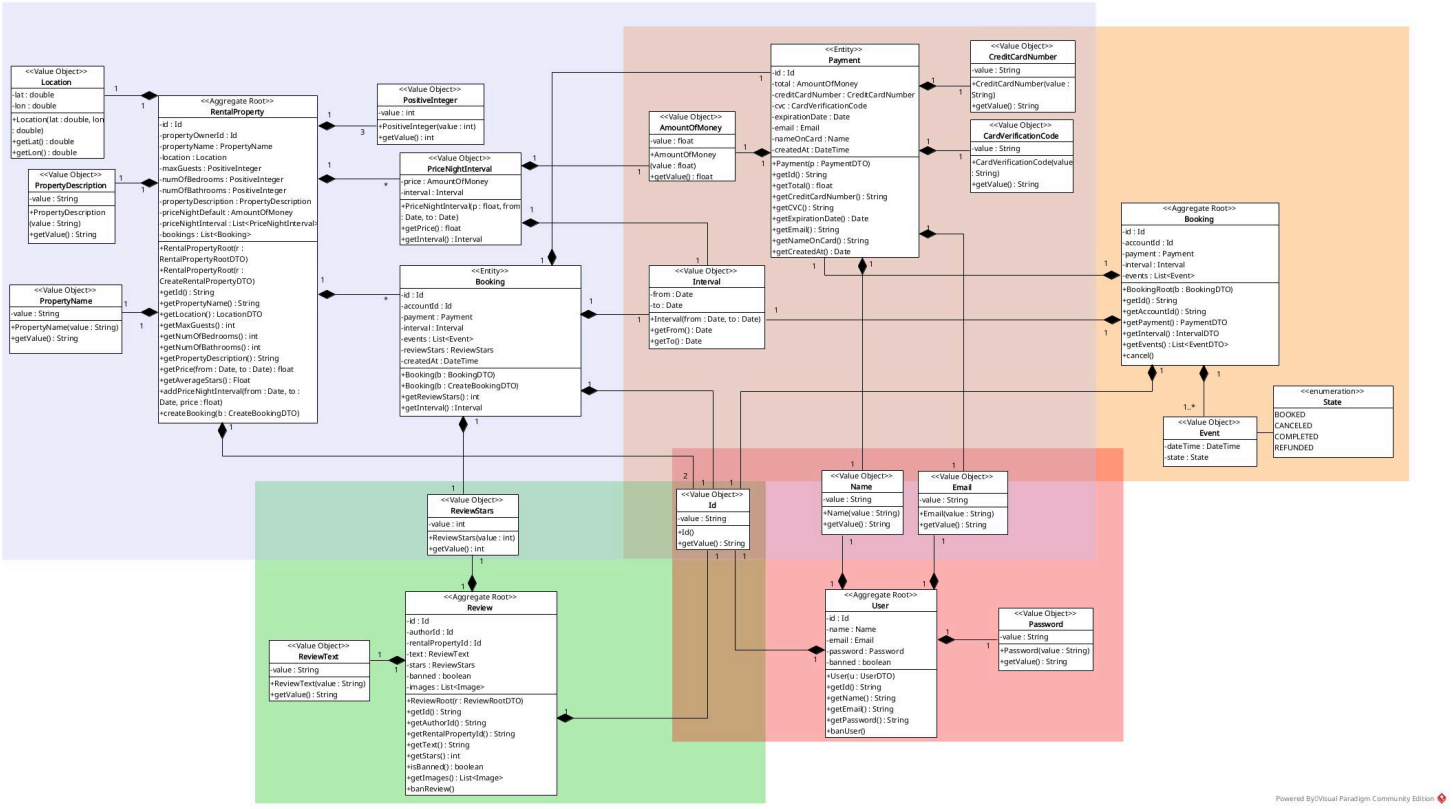
Go to Security related requirements Excel

1.2 Design

1.2.1 System architecture

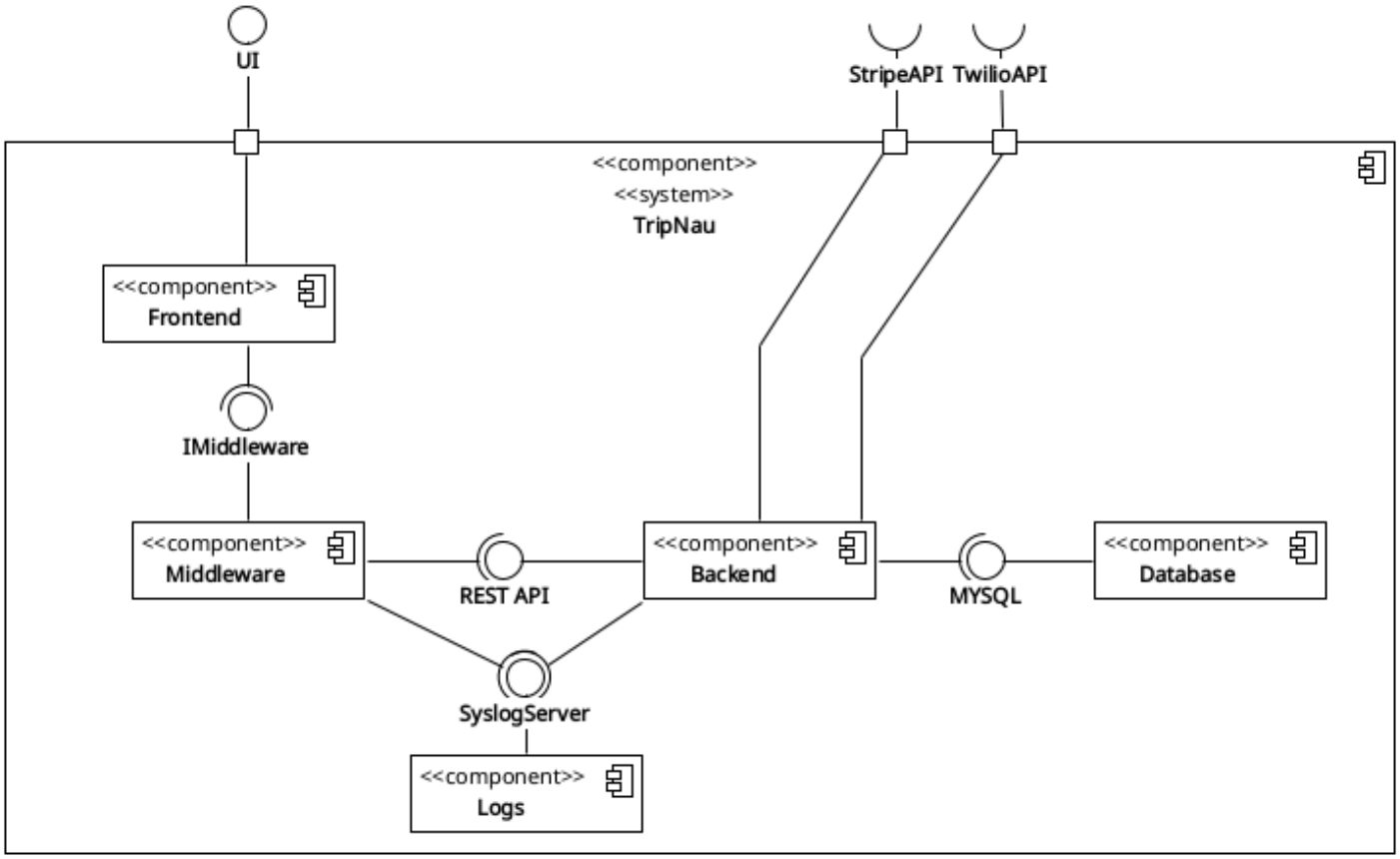
1.2.1.1 Class diagram

The TripNau system employs the Domain-Driven Design (DDD) pattern, which emphasizes the encapsulation of domain logic
within distinct aggregates. These aggregates include 'RentalProperty,' 'Review,' 'Booking,' and 'User.' Each aggregate is
represented by a specific color in the following diagram. This architectural approach enhances clarity, organization,
and maintainability within the system by grouping related components together.

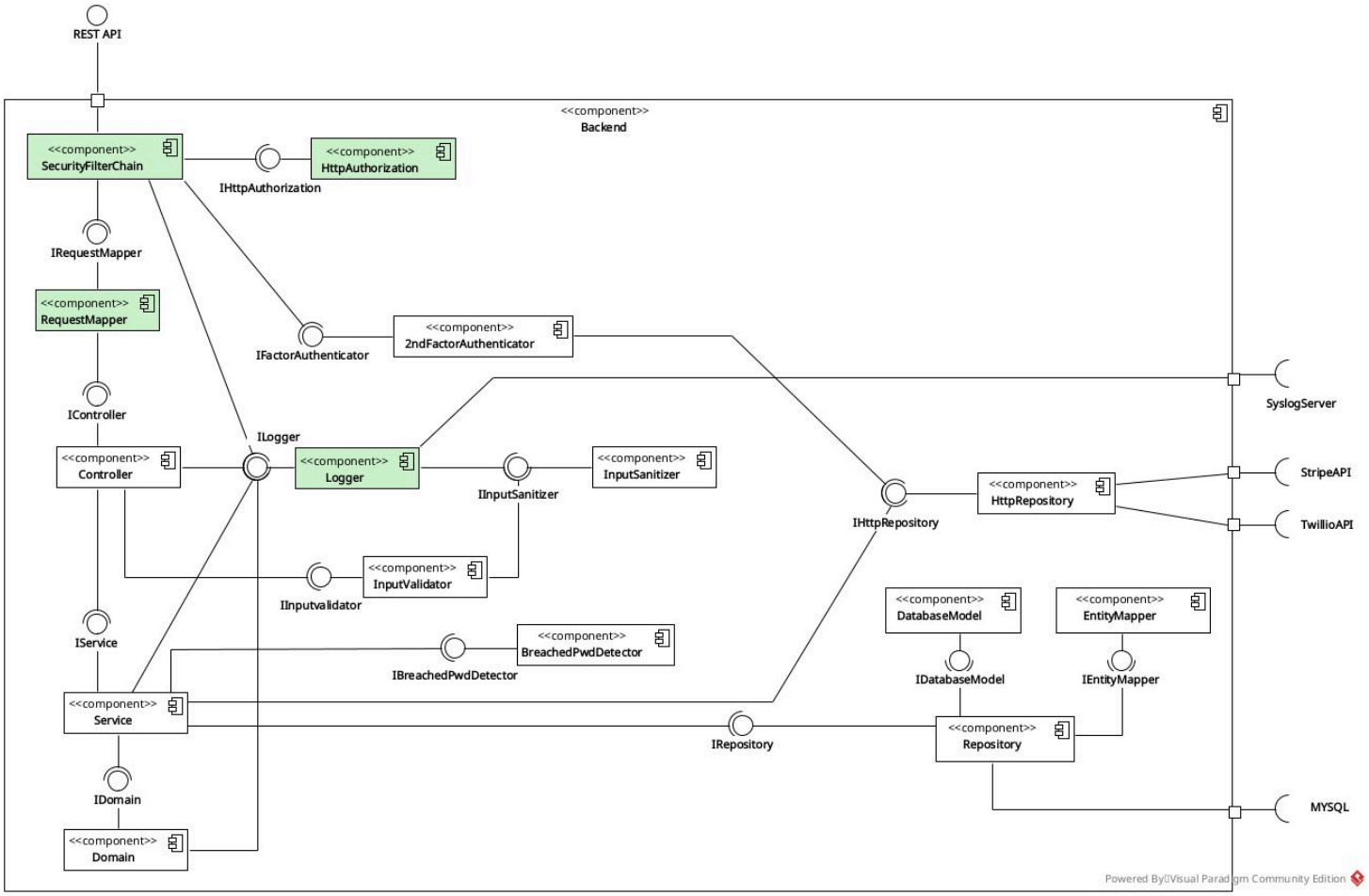


1.2.1.2 Components diagram

This section provides a logical representation of the system architecture for TripNau. The following diagram illustrates
how the TripNau system can be divided into various components with a low level of granularity, as well as the interfaces
provided and required by each component.



The diagram below presents an in-depth view of the backend component of the TripNau system, focusing on its various elements
with a higher level of granularity.



SecurityFilterChain: Spring Security maintains a filter chain internally where each of the filters has a particular responsibility and filters
are added or removed from the configuration depending on which services are required (**Chain of responsibility pattern**).
Filters can be used for a number of different purposes, like authentication, authorization, exploit protection, and more
(ref).

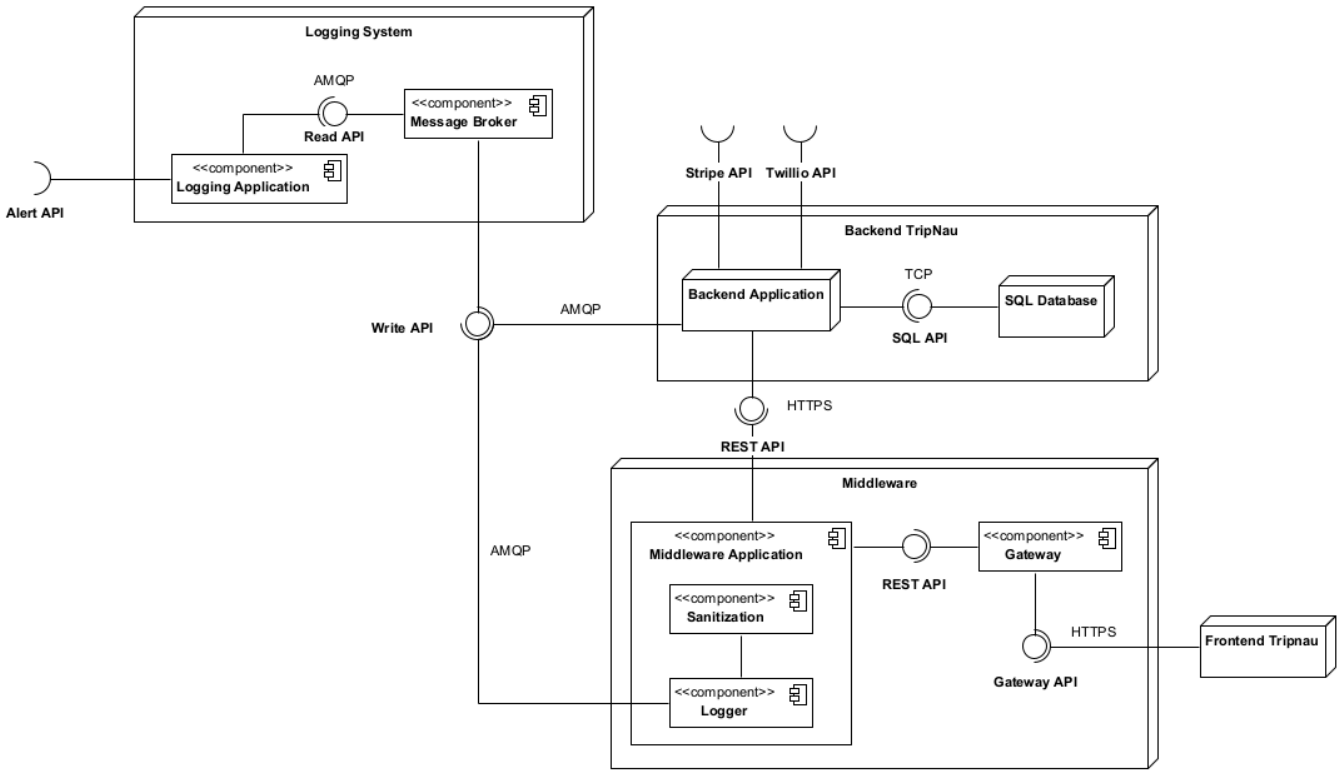
Security mechanisms must be designed so that a failure will follow the same execution path as disallowing the
operation (ref). A security filter chain is designed in such a way that
downstream filter instances aren't invoked when there is a failure (ref),
adhering to the **Fail Securely Security Principle**.

HttpAuthorization: Spring request authorization components permit the definition of match rules by HTTP method
(ref).
This adheres to the **Least Privilege Security Principle**.

InputSanitizer: Ensures that an input to a system function does not trigger an unexpected and unauthorized behavior.

InputValidator: Performs abstract level validations with no domain considerations.

1.2.1.3 Deployment diagram



Since this is a web application, the client-server architecture was adopted, and in order to relieve the processing load on the
Backend, acting as a gateway, a Middleware component was created that is responsible for filtering out requests that should
not reach the Backend, e.g. requests for authenticate routes that carry invalid tokens. In addition, the Middleware component
is also responsible for sanitizing and validating the payloads of the requests on a technical level, e.g. ensuring that
strings meet the minimum size or that requests with excessively large strings do not reach the Backend.

The Backend, in turn, will be responsible for carrying out all the validation at business level. This component will only
accept requests from Middleware and will therefore ignore all requests from other sources. The

database that the Backend
communicates with is hidden from the outside world, being on a private network. This component is also responsible for
communicating with two external APIs in order to fulfill the necessary business flows, in this case the Stripe API and the
Twilio API.

Finally, there is the Logs component, which is included in the entire system, as it is responsible for receiving all the
actions carried out in the system and thus recording everything according to a standard. In order not to block any part of
the system, a message broker was added to the communication, so that actions can be written and read asynchronously.

1.2.2 Threat model (Software-Centric approach)

1.2.2.1 Application decomposition

1.2.2.1.1 Threat Model Information

Application Version: 1.0

This website is designed for the travel sector, focusing on facilitating the purchase of stays from customer to customer,
emphasizing a peer-to-peer C2C model while excluding commercial entities in this first implementation. The platform supports
three distinct user roles:

- Customer
- Property Owners
- System Admins

Property owners will be able to login and have access to features that allows them to register a new property and view
the booking history for their listings. Users, on the other hand, can browse through a list of available properties without
login, but will be able to login and initiate booking processes, and provide feedback through reviews. System admins will
be able to login and play a role in approving reviews and managing the properties submitted to the platform.

1.2.2.1.2 External dependencies

ID	Description
1	The TripNau system will run on docker containers. All the images should be verified.
2	The system will utilize a MySQL database for data storage. Communication between the backend application and the MySQL database will be secured over TLS.
3	The system will integrate with the Stripe API for managing booking payments securely.
4	Message notifications will be managed using the Twilio API to ensure reliable communication with users.
5	All system logs will be saved to a Syslog server for auditing and monitoring purposes.
6	The system backend and middleware will be developed using Spring Boot, utilizing its libraries extensively.
7	The system frontend will be built using Angular.js, leveraging its libraries for dynamic user interfaces.
8	Communications between backend and middleware, and between middleware and frontend will be secured using HTTPS to ensure data confidentiality and integrity.
9	Integration with RabbitMQ as a message broker using the AMQP protocol will be used for handling asynchronous communication between the backend application and message broker, between the logger from frontend and the message broker, and between message broker and the logging application.
10	The system deployment will heavily rely on Github actions.
11	The source code will be managed on the Github version control system.

1.2.2.1.3 Entry points

ID	Name	Description	Trust Levels
1	HTTPS Port	The booking website will only be accessible via TLS. All pages within the website are layered on this entry point.	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business admin
1.1	Website home page	The splash page for the booking website is the entry point for all	(1) Anonymous Web User, (2) Customer, (3)

		users.	Property Owner, (4) Business admin
1.2	Login Page	Customers, property owners and business admins must log in to the booking website before they can carry out any of the use cases.	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business admin
1.2.1	Login Function	The login function accepts user supplied credentials and compares them with those in the database.	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business admin
1.2.1.1	Two factor authentication Function	The two factor authentication function accepts user information and guarantee that is correct.	(2) Customer, (3) Property Owner, (4) Business admin
1.2.2	Reset password function	The reset function accepts all users, since they provide a valid email.	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business admin
1.3	Signup Page	Customers and properties owners that don't have credentials must register in order to realize the login.	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business admin
1.3.1	Signup function	The Signup function accepts any registration data and guarantee that is a valid account before creating a new user.	(1) Anonymous Web User
1.4	Properties entry page	The page used to visualize properties.	(2) Customer, (3) Property Owner, (4) Business admin
1.4.1	Property search function	The page used to search for properties.	(2) Customer, (3) Property Owner, (4) Business admin
1.5	Property detail page	The page used to specify detail information about the property.	(2) Customer, (3) Property Owner, (4)

			Business admin
1.5.1	Book property function	The book function accepts the user information and guarantees that all the information is valid and correct.	(2) Customer
1.6	Own bookings page	The page used by customers to see their bookings.	(2) Customer
1.6.1	Write property's review function	The write property's review function accepts the review information and guarantees that all the information is valid and correct.	(2) Customer
1.7	Own property list page	The page used to list the properties owned by a property owner.	(3) Property Owner
1.7.1	Create new property function	The create new property function accepts the property input and validates in order to guarantee that is valid.	(3) Property Owner
1.7.2	Delete a property function	The delete a property function a property as input and verify if exists in the database.	(3) Property Owner
1.8	Pending review list page	The page where is possible to see the pending reviews.	(4) Business admin
1.8.1	Approve/Reject review function	The approve/reject review function takes information about one review and verify if exists in the database.	(4) Business admin

1.2.2.1.4 Exit points

ID	Name	Description	Trust Levels
1	HTTPS Port (Response)	The response from the booking backend, served via LTS.	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business administrator

2	Login Status	Error messages returned to the user via the the log in page might allow for entry point attacks, such as account harvesting (username not found), and SQL injection (SQL exception errors).	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business administrator
3	Register Status	Error messages returned to the user via the the register page might allow for entry point attacks, such as account harvesting (username not found), and SQL injection (SQL exception errors).	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business administrator
4	Reset Password Status	Error messages returned to the user via the the reset password returned status page might allow for entry point attacks.	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business administrator
5	Create Rental Property returned page	The rental property page returned to the user after creation might allow for entry point attacks.	(3) Property Owner
6	Owned Bookings List returned page	The list of updated owned bookings returned to the user after the creation might allow for entry point attacks.	(2) Customer, (3) Property Owner
7	Cancel a Stay Status	When a user cancels a stay, a page with the confirmation is returned and it might allow for entry point attacks.	(2) Customer
8	Create Review returned page including image submission	The updated reviews page returned to the user after the review with image creation might allow for entry point attacks.	(2) Customer
9	Error Messages	Error messages displayed all over the pages in response to failed actions attempts. These messages might inadvertently disclose information about the existence of important information.	(1) Anonymous Web User, (2) Customer, (3) Property Owner, (4) Business administrator

1.2.2.1.5 Assets

ID	Name	Description	Trust Levels
1	User related data	Assets related to user information.	
1.1	Auth details for TripNau users	The login credentials that a costumer, property owner or a business admin will use to log into the TripNau.	(2) Customer (3) Property owner (4) Business admin (5) Backend server user process (6) Middleware server user process (10) DB read/write user (11) DB admin
1.2	User payment data	Credit card related data such as credit card number and card verification code.	(2) Customer (3) Property owner (5) Backend server user process (6) Middleware server user process (10) DB read/write user (11) DB admin
1.3	User booking related data	Booking information such as dates, status, etc.	(2) Customer (3) Property owner (5) Backend server user process

ID	Name	Description	Trust Levels
			(6) Middleware server user process (10) DB read/write user (11) DB admin
1.4	Read access to property related data	Property to book information such as available dates, price per night, num of bedrooms, etc.	(1) Anonymous user (3) Property owner (5) Backend server user process (6) Middleware server user process (10) DB read/write user (11) DB admin
1.5	Read/Write access to property related data	Full access to property related data.	(3) Property owner (5) Backend server user process (6) Middleware server user process (10) DB read/write user (11) DB admin
2	System	Assets related to the underlying system.	
2.1	Availability of the TripNau	TripNau should be available 24 hours a day and be accessible by all it's users.	(5) Backend server user process

ID	Name	Description	Trust Levels
			(6) Middleware server user process (11) DB admin
2.2	Ability to execute code as a backend server user	This is the ability to execute source code on the backend server as a server user.	(5) Backend server user process
2.3	Ability to execute SQL as a DB Read/Write user	This is the ability to execute SQL. Select, insert, and update queries on the database and thus have read and write access to any information stored within the Database.	(10) DB read/write user
2.4	Ability to request Stripe API	This is the ability to perform requests to the Stripe API with the purpose of creating check out sessions.	(5) Backend server user process
2.5	Ability to request Twilio API	This is the ability to perform requests to the Twilio API with the purpose of sending e-mails and messages.	(5) Backend server user process
2.6	Ability to execute code as a middleware server user	This is the ability to execute source code on the middleware server as a server user.	(6) Middleware server user process
2.7	Ability to request backend server	This is the ability to request the backend provided endpoints	(6) Middleware server user process
2.8	Ability to put messages into the message broker	This is the ability to enqueue log messages into the message broker	(9) Message queue write user
2.9	Ability to read messages from the message broker	This is the ability to dequeue log messages from the message broker	(8) Message queue read user
3	Infrastructure	Assets related to infrastructure access	

ID	Name	Description	Trust Levels
3.1	Access to the Database server	Gives full access to the data contained within the database	(11) DB admin
3.2	Access to Audit data	Audit data shows all the events that occurred in the TripNau system	(7) Logs admin
3.3	Access to the backend server	Full access to the machine where the backend server is hosted	(5) Backend server user process
3.3.1	Access to API keys and secrets	Access to sensitive data that serves as authentication to third party's among other secrets	(5) Backend server user process
3.4	Access to the middleware server	Full access to the machine where the middleware server is hosted	(6) Middleware server user process

1.2.2.1.6 Trust Levels

ID	Name	Description
1	Anonymous web user	A user that has connected to the website but hasn't provided credentials.
2	Customer	The customer can see the booking list, see their personal information, and book a stay. Also, they can submit reviews of bookings that they stayed.
3	Property owner	A property owner can register a new property to be booked, see their personal information and see the bookings that are associated with their properties.
4	Business administrator	The business admin can approve or reject reviews sent by users, as the properties sent by the properties owners.
5	Backend server user process	This is the user responsible for the backend process execution.
6	Middleware server user process	This is the user responsible for the middleware process execution.

7	Logs administrator	The administrator can analyze the logs and be notified of alerts originated by the log system.
8	Message queue read user	The user only has access to dequeue messages.
9	Message queue write user	The user only has access to enqueue messages.
10	DB read/write user	The user that only has access to read/write queries.
11	DB user admin	The user that has permissions to administrate the database.

1.2.2.1.7 Data Flow Diagrams

Go to threat model report

1.2.2.2 Rank and determination of threats

1.2.2.2.1 Threat Categorization (STRIDE)

Go to threat model report

1.2.2.2.2 Threat Analysis

1.2.2.2.2.1 Abuse Cases

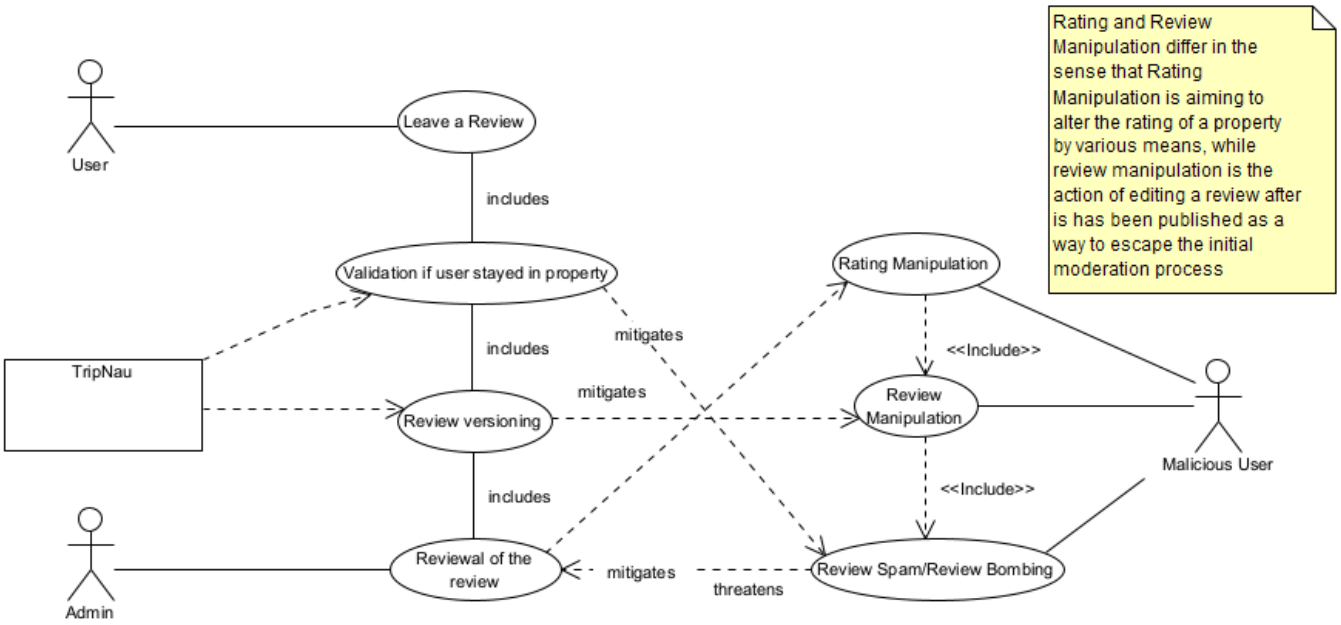
ID	Abuse Case	Description
1	As a malicious user, I will perform an injection attack on the application	This includes injection attacks like XSS, SQL Injection, OS commands, among others.
2	As a malicious user, I will try to access other's accounts.	This can include brute force attacks, dictionary tool attacks
3	As a malicious user, I will try to manipulate property ratings	This can include removal of negative reviews or usage of bots to add positive/negative reviews on a property
4	As a malicious user, I will try to evade review monitoring	This can be done by trying to manipulate reviews after they have been monitored. Another way would be review bombing and flooding the moderators queue.
5	As a malicious user, I will try to access restricted areas without	This might be done by trying to log into an admin account with default credentials, or simple trying to

	proper permission.	reach restricted zones directly using a default account.
6	As a malicious user, I will try to register a property without owning it.	This can be either registering properties that aren't his or even that don't exist all.
7	As a malicious user, I will try to delete my property without refunding any bookings made to it.	This means that a malicious user will try to delete his property's listing without any intention to refund the bookings made to it.
8	As a malicious user, I will try to edit a property's information without owning it	This means that the malicious user might try to edit someone else's property from his account or even try to enter the owners account to edit the information.
9	As a malicious user, I will try to delete a property without actually owning it.	This means that the malicious user might try to delete someone else's property from his account or even try to enter the owners account to delete it.
10	As a malicious user, I will try to refund a booking after having already stayed there	This can mean he will try to refund the booking during or after the stay.
11	As a malicious user, I will try to view the bookings made by other people	This can be mean he will try to access other peoples details/bookings page or even try to forge requests to the API
12	As a malicious user, I will try to upload an image in a review with too big of a size	This can lead to a downtime on the server as it processes the image data as it is too big
13	As a malicious user, I will try to upload an image with faulty metadata	A malicious user might include malicious scripts in the image metadata that can be executed if it isn't properly treated
14	As a malicious user, I will try to simulate cancelling a booking to attempt to get a refund	This can mean the user will try to get a refund without cancelling the stay.
15	As a malicious user, I will try to book a stay on dates where the property is already booked	This can mean the user will try to override another user's stay.

1.2.2.2.2.2 Use and Abuse Case Graphs

The action of logging in as a user is prone to several types of attacks and there are many abuse cases associated with it. In order to mitigate these attacks the application should make sure not to show too much information on login failure as well as lock the account after 3 failure attemps. The amount of time locked out should be incremental as a way to mitigate dictionary attacks. As a way to prevent these attacks, 2FA is also in place.

The action of leaving a review can be used by both normal users as a way to show their liking on a booking, but it can also be used by malicious users as a way to influence the ratings on a property, giving it a better rating or worse according to their intents. As a counter-measure to this, the system should validate if users actually stayed in the property to be able to leave a review. Another thing users might do is alter reviews after the first moderation process done by a admin, as a way to leave review that don't follow TOS. For this a review versioning should be in place where whenever a review is altered it can go by an automatic reviewal process and eventually if necessary a manual one.

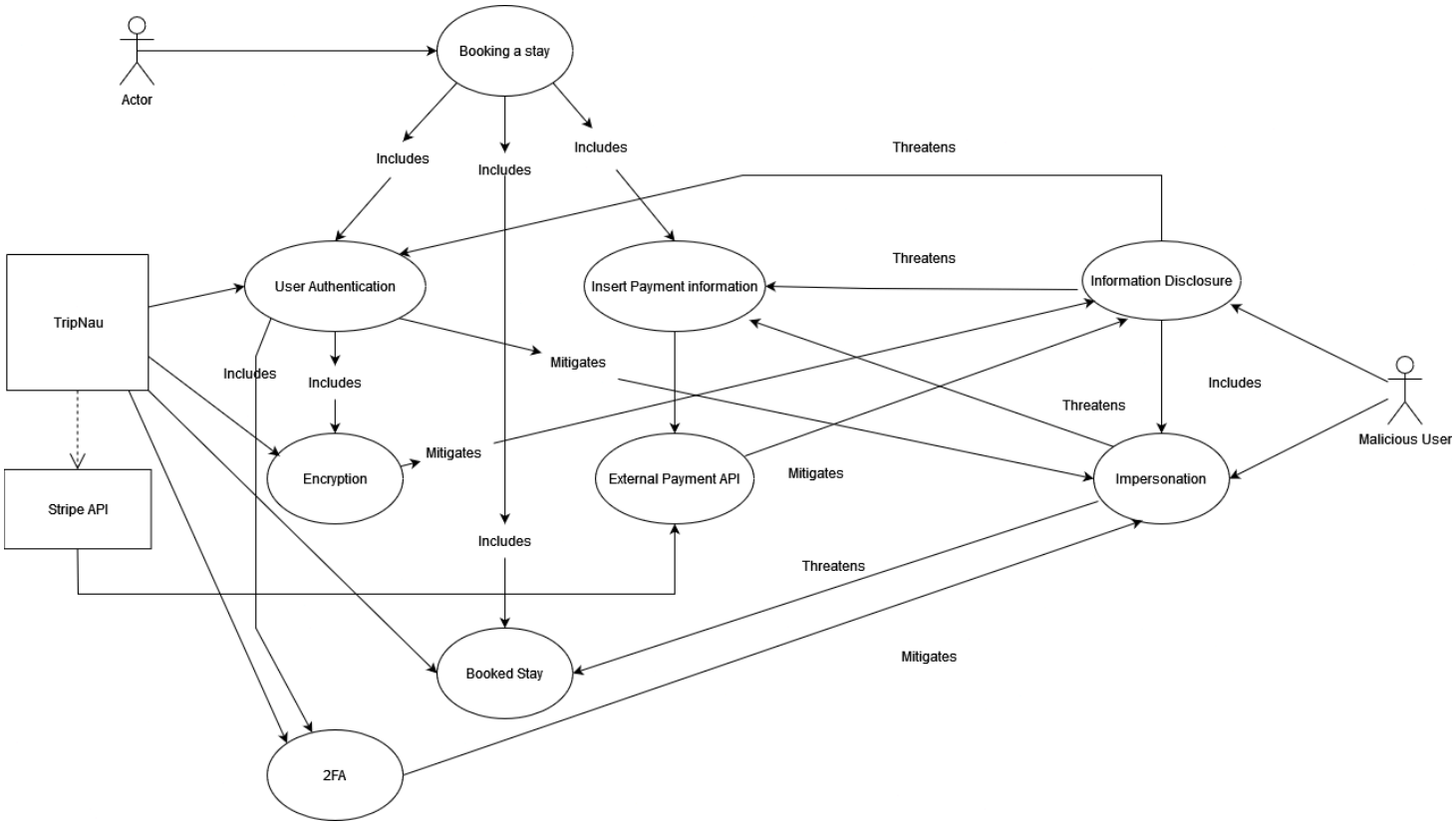


Booking a stay involves the input of a lot of sensitive data from the user including his personal information as well as payment details. This information is usually a target for attackers and needs to be properly secured.

In order to protect it Authentication is required to book stays which includes encryption of the data as well as 2FA, ensuring that, if a user has payment information associated to their account, even if the account login details are compromised, the payment data won't.

When booking a stay the user will perform the payment through the Stripe API, which in itself offers security measures, abstracting the requests with the data from the application, making it harder for bad actors to steal it.

2FA also helps in preventing unauthorized bookings in case a user, for example, leaves his account logged on in an unprotected machine.

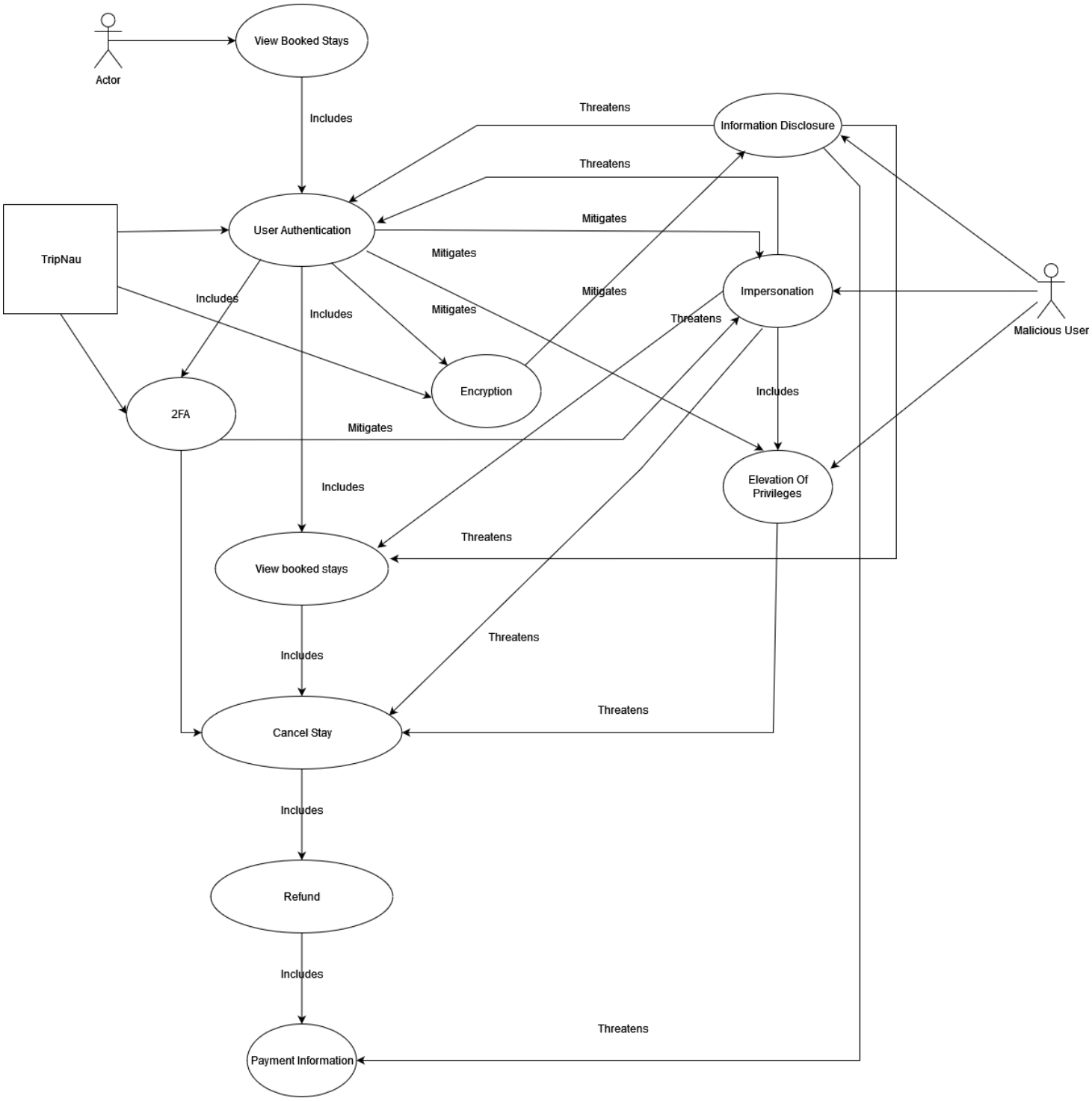


Similarly to booking stays, viewing booked stays can expose sensitive data. The same techniques are applied here.

However, viewing a user's booked stays opens up paths to exploits.

If a nefarious user has access to an account they might attempt to cancel a booked stay and divert the refunds to an account they control which is why 2FA is once again used as another layer of confirmation before a user is able to cancel their stay.

On the other hand, if a malicious actor manages to gain access to an account with special privileges they might try to cancel another user's stay to try and book their own. Therefore the authentication of moderator/admin accounts has layers of protection to try and ensure their use by authorized personell only.



1.2.2.2.3 Ranking of Threats

Go to threat model report

1.2.2.3 Countermeasures and Mitigation

Go to threat model report

1.3 Other artifacts

1.3.1 Tests Plan

Go to Tests Plan Excel

1.3.2 ASVS

Go to ASVS Excel