ANEXO IV

Diseño del sistema software.

Diseño y desarrollo de una plataforma robótica para la agilización de procesos en la hostelería

Trabajo de Fin de Grado

Grado en Ingeniería Informática

VNiVERSiDAD D SALAMANCA

CAMPUS DE EXCELENCIA INTERNACIONAL

Julio 2022

Autor

David Cruz García

Tutores

André Filipe Sales Mendes

Gabriel Villarrubia González

Juan Francisco de Paz Santana

Tabla de contenido

1 INTRODUCCIÓN 5
2 MODELO DE DISEÑO 6
2.1 PATRONES DE ARQUITECTURA Y DE DISEÑO 6
2.1.1 PATRÓN BACKEND FOR FRONTEND (BFF) 6
2.1.2 PATRÓN ACCESS TOKEN 7
2.1.3 PATRÓN FACADE 8
2.2 SUBSISTEMAS DE DISEÑO 9
2.3 CLASES DE DISEÑO 11
2.3.1 SERVIDOR WEB O FRONTEND 11
2.3.2 SERVIDOR API O BACKEND 12
2.3.3 ROBOT 13
2.4 VISTA ARQUITECTÓNICA 15
2.5 REALIZACIÓN DE LOS CASOS DE USO 16
2.5.1. PAQUETE GESTIÓN DE USUARIOS 16
2.5.1. PAQUETE GESTIÓN DE BEBIDAS 25
2.5.1. PAQUETE GESTIÓN DE RECETAS 33
2.5.2 PAQUETE GESTIÓN DE ROBOT 45
3 DISEÑO DE LA BASE DE DATOS 50
4 MODELO DE DESPLIEGUE 51
5 REFERENCIAS 53

Índice de ilustraciones

Ilustración 1. Patrón BFF. 6
Ilustración 2. Patrón API Gateway. 7
Ilustración 3. Patrón Access Token con JWT 7
Ilustración 4. Patrón Facade. 8
Ilustración 5. Subsistema de diseño. 10
Ilustración 6. Vista web. 11
Ilustración 7. Controlador de peticiones. 12
Ilustración 8. Controlador API. 12
Ilustración 9. Datos barbot. 13
Ilustración 10. Controlador API Robot. 13
Ilustración 11. Vista web robot. 14
Ilustración 12. Vista arquitectónica. 15
Ilustración 13. Diagrama de secuencia UC-001. 16
Ilustración 14. Diagrama de secuencia UC-002. 17
Ilustración 15. Diagrama de secuencia UC-003. 17
Ilustración 16. Diagrama de secuencia UC-004. 18
Ilustración 17. Diagrama de secuencia UC-005. 18
Ilustración 18. Diagrama de secuencia UC-006. 19
Ilustración 19. Diagrama de secuencia UC-007. 20
Ilustración 20. Diagrama de secuencia UC-008. 20
Ilustración 21. Diagrama de secuencia UC-009. 21
Ilustración 22. Diagrama de secuencia UC-010. 21
Ilustración 23. Diagrama de secuencia UC-011. 22
Ilustración 24. Diagrama de secuencia UC-012. 23
Ilustración 25. Diagrama de secuencia UC-013. 23
Ilustración 26. Diagrama de secuencia UC-014. 24
Ilustración 27. Diagrama de secuencia UC-015. 25
Ilustración 28. Diagrama de secuencia UC-016. 26
Ilustración 29. Diagrama de secuencia UC-017. 27
Ilustración 30. Diagrama de secuencia UC-018. 28
Ilustración 31. Diagrama de secuencia UC-019. 29
Ilustración 32. Diagrama de secuencia UC-020. 30
Ilustración 33. Diagrama de secuencia UC-021. 31
Ilustración 34. Diagrama de secuencia UC-022. 32
Ilustración 35. Diagrama de secuencia UC-023. 33
Ilustración 36. Diagrama de secuencia UC-024. 34
Ilustración 37. Diagrama de secuencia UC-025. 35
Ilustración 38. Diagrama de secuencia UC-026. 36
Ilustración 39. Diagrama de secuencia UC-027. 37
Ilustración 40. Diagrama de secuencia UC-028. 38
Ilustración 41. Diagrama de secuencia UC-029. 39
Ilustración 42. Diagrama de secuencia UC-030. 40
Ilustración 43. Diagrama de secuencia UC-031. 41
Ilustración 44. Diagrama de secuencia UC-032. 42
Ilustración 45. Diagrama de secuencia UC-033. 43
Ilustración 46. Diagrama de secuencia UC-034. 44
Ilustración 47. Diagrama de secuencia UC-035. 45
Ilustración 48. Diagrama de secuencia UC-036. 45

Ilustración 49. Diagrama de secuencia UC-037.....46
Ilustración 50. Diagrama de secuencia UC-038.....47
Ilustración 51. Diagrama de secuencia UC-039.....48
Ilustración 52. Diagrama de secuencia UC-040.....48
Ilustración 53. Diagrama de secuencia UC-041.....49
Ilustración 54. Diseño de la base de datos. 50
Ilustración 55. Diagrama de despliegue.....51

1 INTRODUCCIÓN

Una vez definidos todos los casos de uso del sistema y analizados presentando el modelo de dominio del sistema, hay que comenzar a diseñar el sistema, tanto su estructura mediante el uso de patrones como las diferentes clases de diseño, preparando el proyecto para su implementación.

Por tanto, en este anexo se recogerá toda la documentación referente al diseño del sistema. Para ello, se centrará el documento en el dominio de la solución del problema, dividiéndolo para ello en los siguientes apartados:

- ***Modelo de diseño:*** este modelo se basará en el modelo de análisis explicado en el documento "*Anexo III: Análisis de requisitos"* describiendo con mayor detalle la estructura del sistema y como será implementada dicha estructura. Podemos dividir el modelo de diseño en diferentes apartados:
    - ***Patrones de arquitectura:*** se explicará el uso de patrones usado para el diseño del sistema.
    - ***Subsistemas de diseño:*** se organizará el sistema en diferentes paquetes y como se relacionan dichos paquetes
    - ***Clases de diseño:*** se mostrará el conjunto de clases de diseño explicando sus operaciones, atributos y relaciones en base a los apartados y anexos anteriores.
    - ***Vista de arquitectura:*** vista por capas según el patrón utilizado.
    - ***Realización de casos de uso:*** al igual que en el análisis del sistema, se explicarán los diagramas de secuencia según las clases de diseño del sistema.
- ***Diseño de la base de datos:*** se mostrará un diagrama entidad-relación mostrando la información que debe almacenar y manejar el sistema.
- ***Modelo de despliegue:*** se muestra la forma de despliegue de todo el sistema, mostrando para ellos los nodos y artefactos participantes.

2 MODELO DE DISEÑO

Basándonos en el modelo del dominio del problema y el análisis del sistema junto a los distintos casos de uso explicados en anexos anteriores, se explicará el modelo de diseño del sistema.

Durante esta fase de desarrollo del modelo de diseño del sistema se acercará el desarrollo del sistema a la implementación mediante clases de diseño que disponen de atributos y funciones que posteriormente serán implementadas.

Para el desarrollo del modelo de diseño se hará uso de:

- ***Node.js***: entorno de desarrollo para los servidores (frontend y backend).
- ***Express***: framework web de código abierto para el desarrollo de APIs en Node.js. Se utilizará para el desarrollo del servidor backend (API REST)
- ***React***: biblioteca javascript de código abierto para el desarrollo de aplicaciones web. Se utilizará para el desarrollo del servidor frontend (la web). Trabajará junto a ***HTML, CSS y JS.***
- ***Octoprint API:*** API proporcionada por el SSOO OctoPi para el control del robot.

2.1 PATRONES DE ARQUITECTURA Y DE DISEÑO

En este apartado se plantearán los distintos patrones arquitectónicos que se utilizarán para el correcto diseño e implementación del sistema.

2.1.1 PATRÓN BACKEND FOR FRONTEND (BFF)

El patrón ***BFF o Backend-For-Frontend*** es un patrón de diseño software enfocado a servicios REST.

Con este patrón se enfoca el desarrollo del backend según las necesidades del frontend, con lo que priorizamos la interfaz de usuario y las opciones que esa UI necesita. Así, el desarrollo se centrará principalmente en el usuario y en la interfaz con la que trabajará, desarrollando una API REST totalmente personalizada para dicho frontend (aunque dicha API podría usarse para otro tipo de aplicaciones).

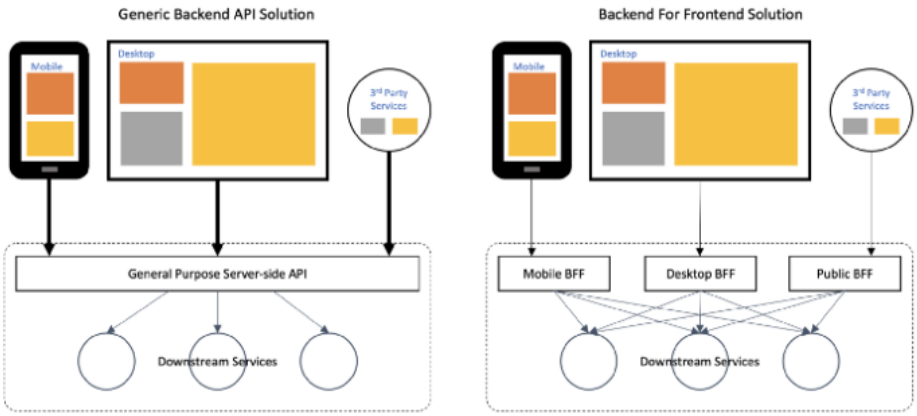


Ilustración 1. Patrón BFF.

Este patrón es una variante del patrón ***API Gateway***. Este otro patrón permite centralizar y enrutar todas las peticiones hacia un servicio único y personalizado.

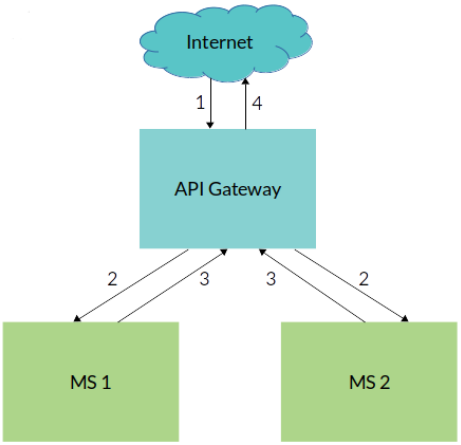


Ilustración 2. Patrón API Gateway.

Con el uso del patrón BFF como variante del patrón API Gateway se podrá enfocar el sistema directamente a ofrecer los servicios únicamente necesarios para el correcto funcionamiento de la web y para que los usuarios trabajen de forma cómoda con el robot y la aplicación.

2.1.2 PATRÓN ACCESS TOKEN

El patrón ***Access Token*** es un patrón arquitectónico que sirve para autenticar, utilizando para ello un token de acceso, las peticiones enviadas. Este patrón trabaja junto al patrón *API Gateway* explicado en el apartado anterior para un correcto desarrollo de la API REST.

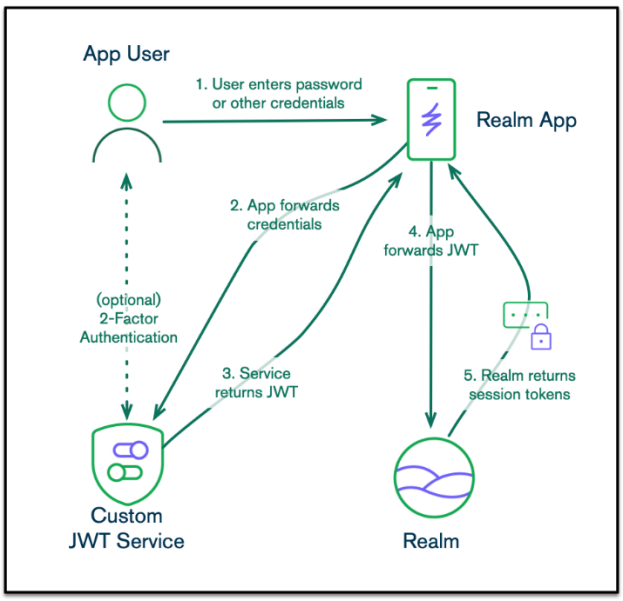


Ilustración 3. Patrón Access Token con JWT

Gracias al uso de este patrón, podemos ofrecer distintos servicios a través de la API y protegerlos (o no) ante usuarios no deseados. Esto se hará utilizando un token ***JWT*** para la autenticación de usuarios que se generará de forma única o se eliminará cuando un usuario haga login o logout en el sistema respectivamente (o en caso de que expire el tiempo de uso del token).

2.1.3 PATRÓN FACADE

El patrón ***Facade o Fachada*** es un patrón de diseño estructural que actúa de intermediario ocultando la complejidad para acceder a un determinado objeto o servicio, proporcionando para ello una interfaz de alto nivel con la que acceder de una manera mucho más sencilla y legible para el programador.

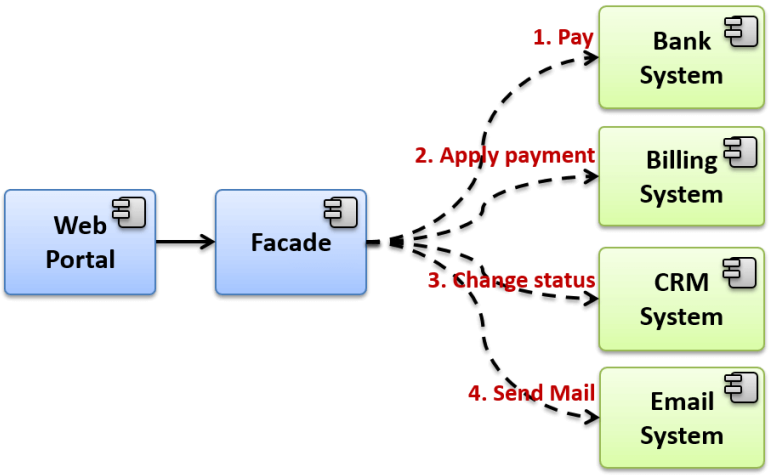


Ilustración 4. Patrón Facade.

Este patrón se utilizará para enmascarar en un objeto en el servidor Frontend (web) toda la lógica necesaria para hacer las peticiones al servicio correspondiente del servidor Backend (API REST).

2.2 SUBSISTEMAS DE DISEÑO

Una vez superada la etapa de análisis del proyecto y establecidos los patrones arquitectónicos que se utilizarán durante el diseño, se realiza una división del sistema en subpaquetes más específicos dentro de la estructura general de la aplicación:

- ***Servidores:*** este subpaquete del sistema hace referencia a los dos servidores desarrollados siguiendo el patrón BFF (*Backend For Frontend)* explicado anteriormente: el servidor web o frontend y el servidor api o backend. En cada servidor a su vez nos encontramos con varios paquetes:
    - ***Servidor web o frontend:*** servidor que proporciona la web a los distintos usuarios para soliticar la preparación de bebidas al robot, gestionar el robot, gestionar los datos de la aplicación, etc. Incluye 2 subpaquetes:
        - ***Vista web:*** hace referencia a las distintas páginas web que verán los usuarios y donde se les presentará la información.
        - ***ControladorPeticiones:*** hace referencia a un objeto que será el encargado de hacer las peticiones a la API del servidor backend para trabajar con los distintos datos y con el robot. Este controlador sirve como fachada (patrón *Facade*) para las vistas que necesitan
    - ***Servidor API o backend:*** servidor encargado de proporcionar una API a medida para el backend (patrón BFF) para poder trabajar con los distintos datos almacenados en el sistema y con el propio robot. Este servidor se encargará de procesar las peticiones recibidas por el servidor frontend y gestionar los datos, además de realizar las peticiones necesarias hacia la API del robot. Contiene 2 subpaquetes:
        - ***ControladorAPI***: recoge toda la lógica necesaria para ofrecer los distintos servicios a través de rutas (API) y se encarga además de comunicarse con el robot según esas peticiones recibidas.
        - ***DatosBarbot:*** incluye todos los datos almacenados en el sistema, tanto los datos guardados en la base de datos como las imágenes de las recetas (se guardarán en el backend mientras que se guardará la ruta a dichas imágenes en la base de datos para no sobrecargarla).
- ***Robot:*** este subpaquete representa todo el robot en conjunto, incluyendo tanto el firmware a instalar para el movimiento del robot como la API expuesta para poder comunicarse con él y la web para poder gestionarlo. Este paquete incluye:
    - ***APIRobot:*** subpaquete que incluye toda la funcionalidad referente a la API con la que el servidor backend se comunicará para gestionar el robot y la web a la que podrá acceder el administrador para poder ver el estado del robot. Se divide a su vez en 2 subpaquetes:
        - ***ControladorAPIRobot:*** incluye toda la funcionalidad para la comunicación con el robot.
        - ***Vista web robot:*** incluye la web para la gestión del robot.
    - ***FirmwareRobot:*** incluye toda la funcionalidad dentro del controlador de los motores necesaria para convertir las órdenes enviadas por el *ControladorAPIRobot* para el movimiento del robot.

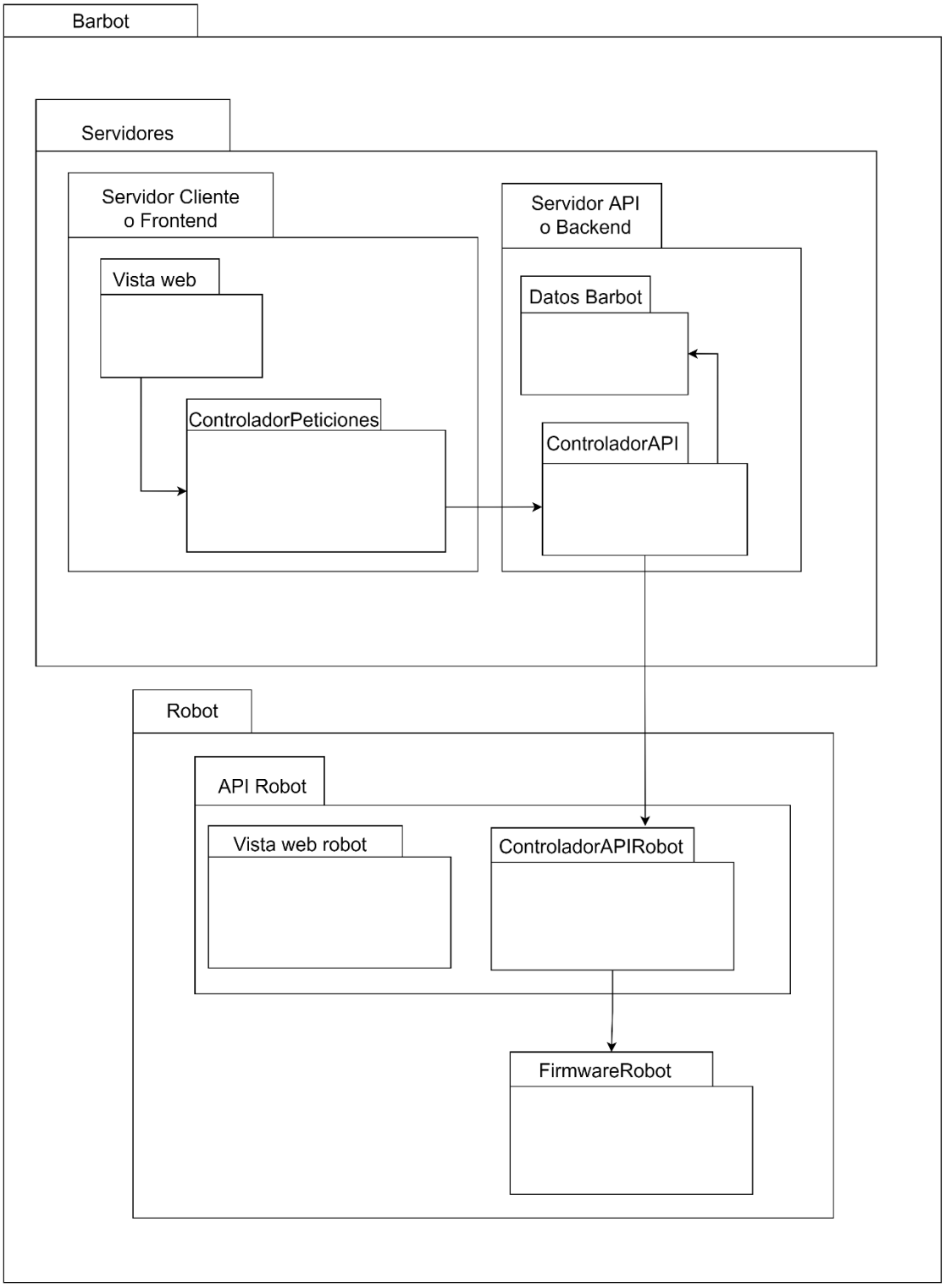


Ilustración 5. Subsistema de diseño.

2.3 CLASES DE DISEÑO

Este apartado recopilará los contenidos del sistema en clases con sus respectivos métodos (clases de diseño).

2.3.1 SERVIDOR WEB O FRONTEND

- ***Vista web:*** es la encargada de proporcionar las interfaces necesarias mediante clases o componentes para los usuarios.

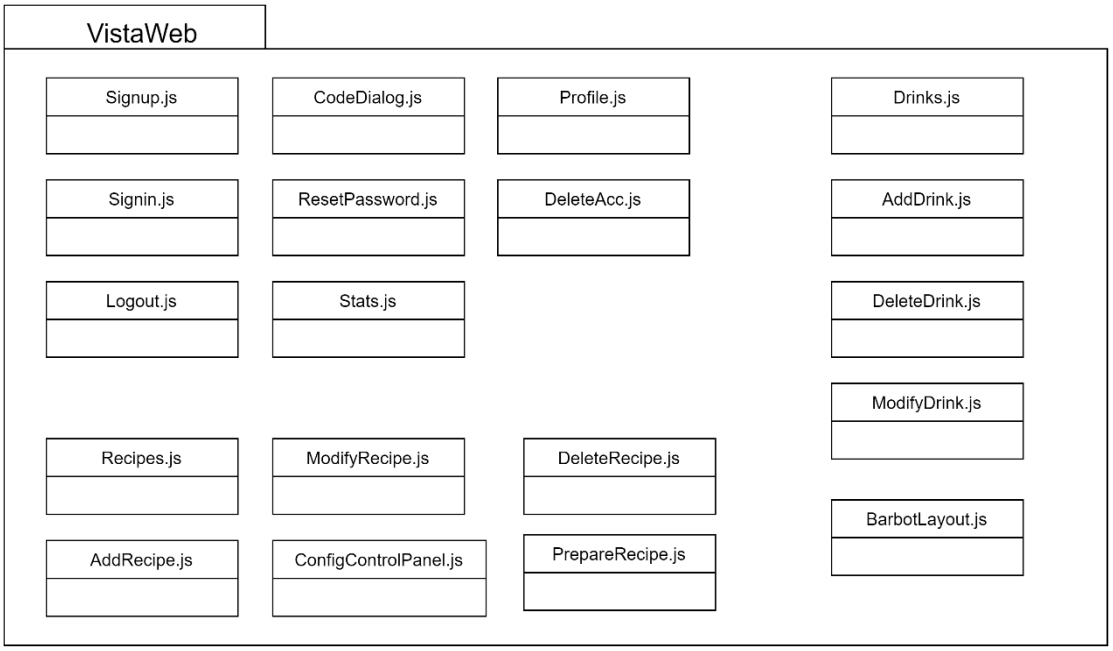


Ilustración 6. Vista web.

- ***ControladorPeticiones:*** Este controlador sirve de fachada (patrón *Facade)* para los elementos de la vista enmascarando las peticiones a la API del servidor backend con tan solo llamar a los elementos de estas clases.

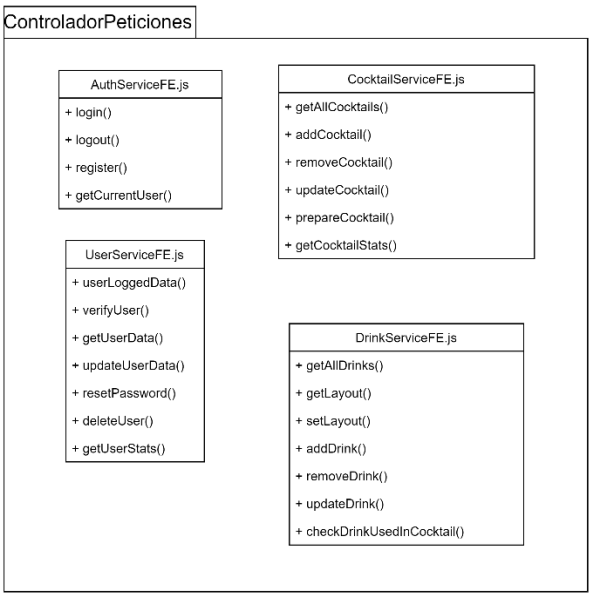


Ilustración 7. Controlador de peticiones.

2.3.2 SERVIDOR API O BACKEND

- ***ControladorAPI:*** controlador encargado de la API que recibe peticiones y realizar las acciones necesarias sobre los datos y sobre el robot.

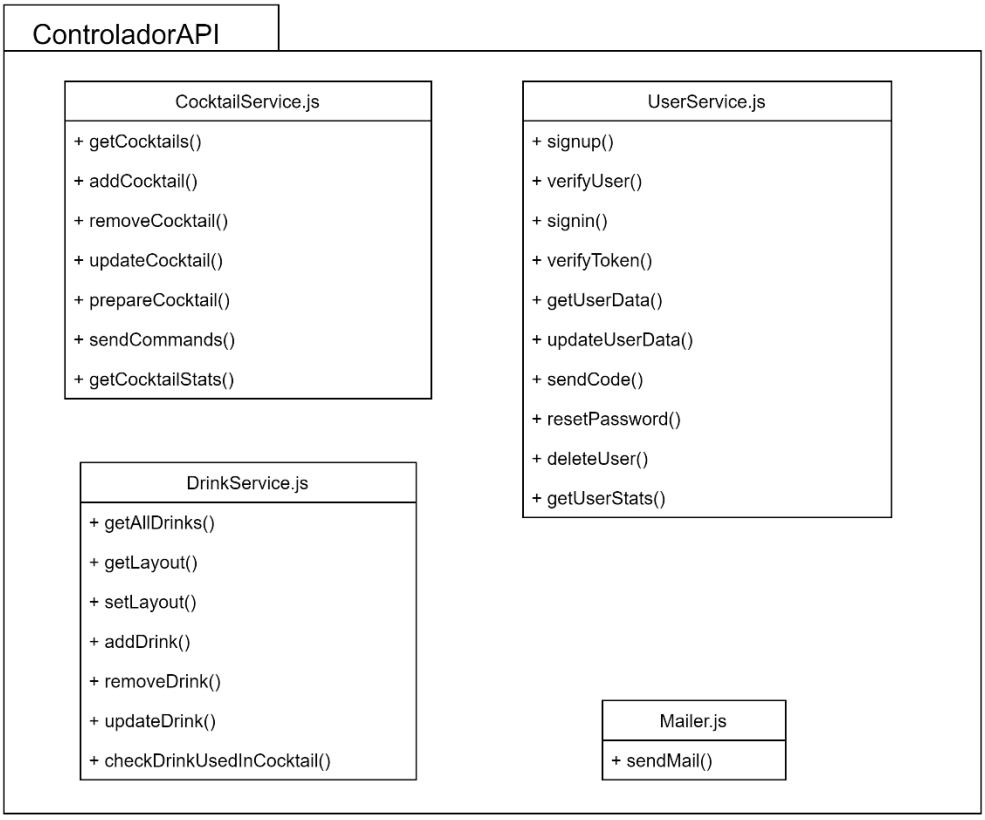


Ilustración 8. Controlador API.

- ***DatosBarbot:*** datos que gestionará el sistema para las bebidas, recetas, usuarios, etc.

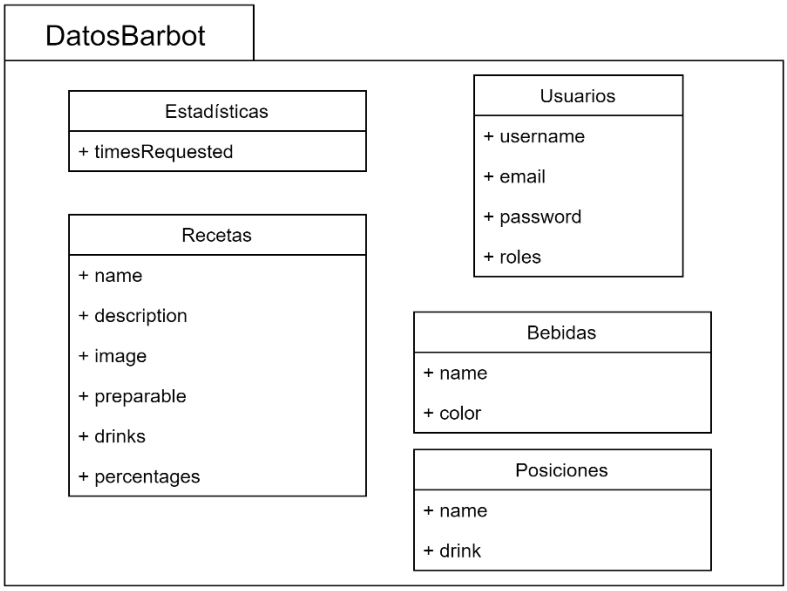


Ilustración 9. Datos barbot.

2.3.3 ROBOT

- ***ControladorAPIRobot:*** encargado de gestionar las peticiones recibidas del robot. Será gestionado por la API ofrecida por el SSOO OctoPi, la cual gestionará las peticiones enviadas al robot, el servidor web para su gestión y los datos del robot.

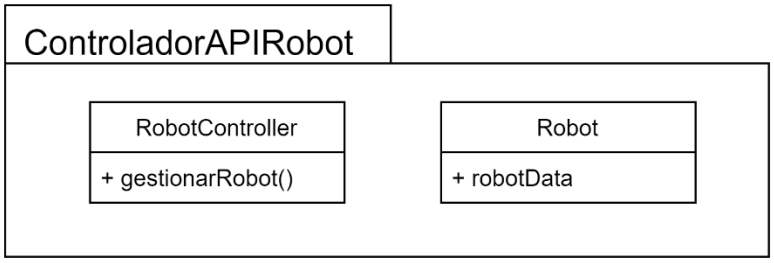


Ilustración 10. Controlador API Robot.g

- ***VistaWebRobot:*** Web de gestión del robot para consultar su estado, el estado de la conexión con el firmware, …

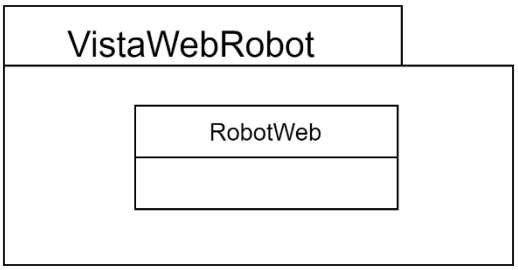
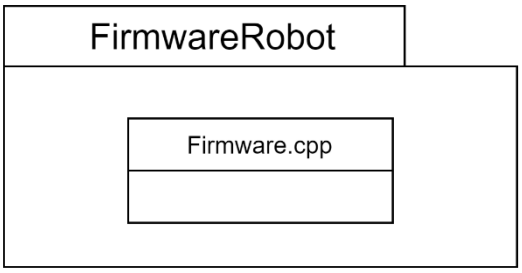


Ilustración 11. Vista web robot.

- ***FirmwareRobot:*** firmware que sirve para mover el robot y gestionar otros componentes electrónicos como el LED.



2.4 VISTA ARQUITECTÓNICA

Gracias a la vista arquitectónica, se da una imagen de la implementación del patrón arquitectónico BFF (*Backend For Frontend)* en la estructura del sistema diseñado. Además, se pueden dividir, al igual que en la etapa de análisis, las clases de diseño según el patrón MVC (Modelo-Vista-Controlador), dando una visión general del sistema separada por las vistas, los controladores y los datos almacenados (modelo).

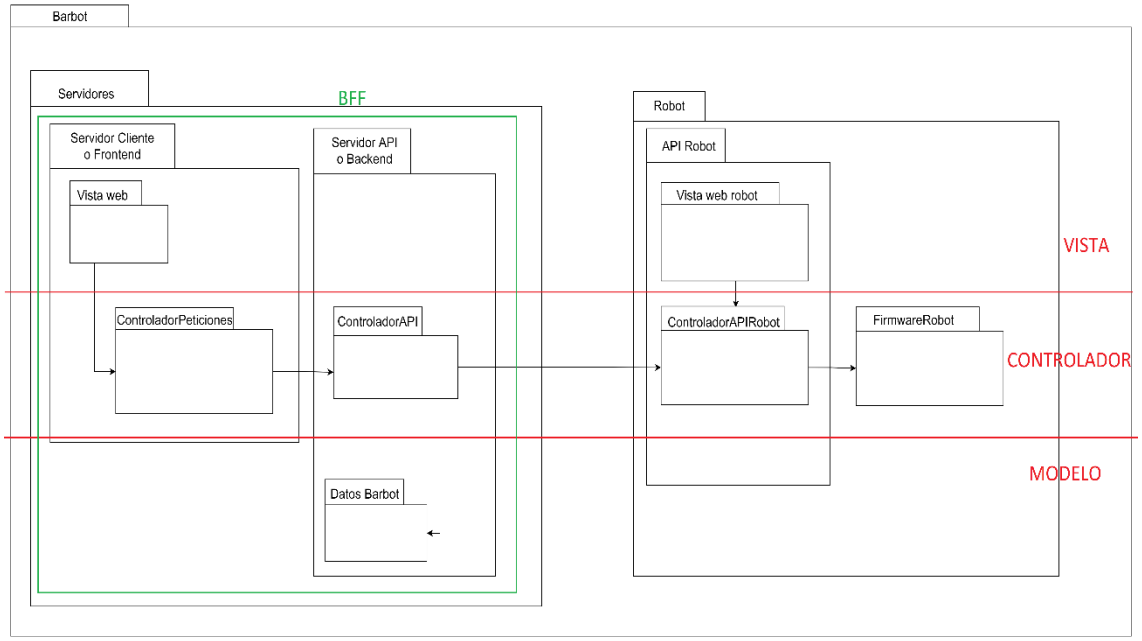


Ilustración 12. Vista arquitectónica.

2.5 REALIZACIÓN DE LOS CASOS DE USO

2.5.1. PAQUETE GESTIÓN DE USUARIOS

UC-001	**Registro**

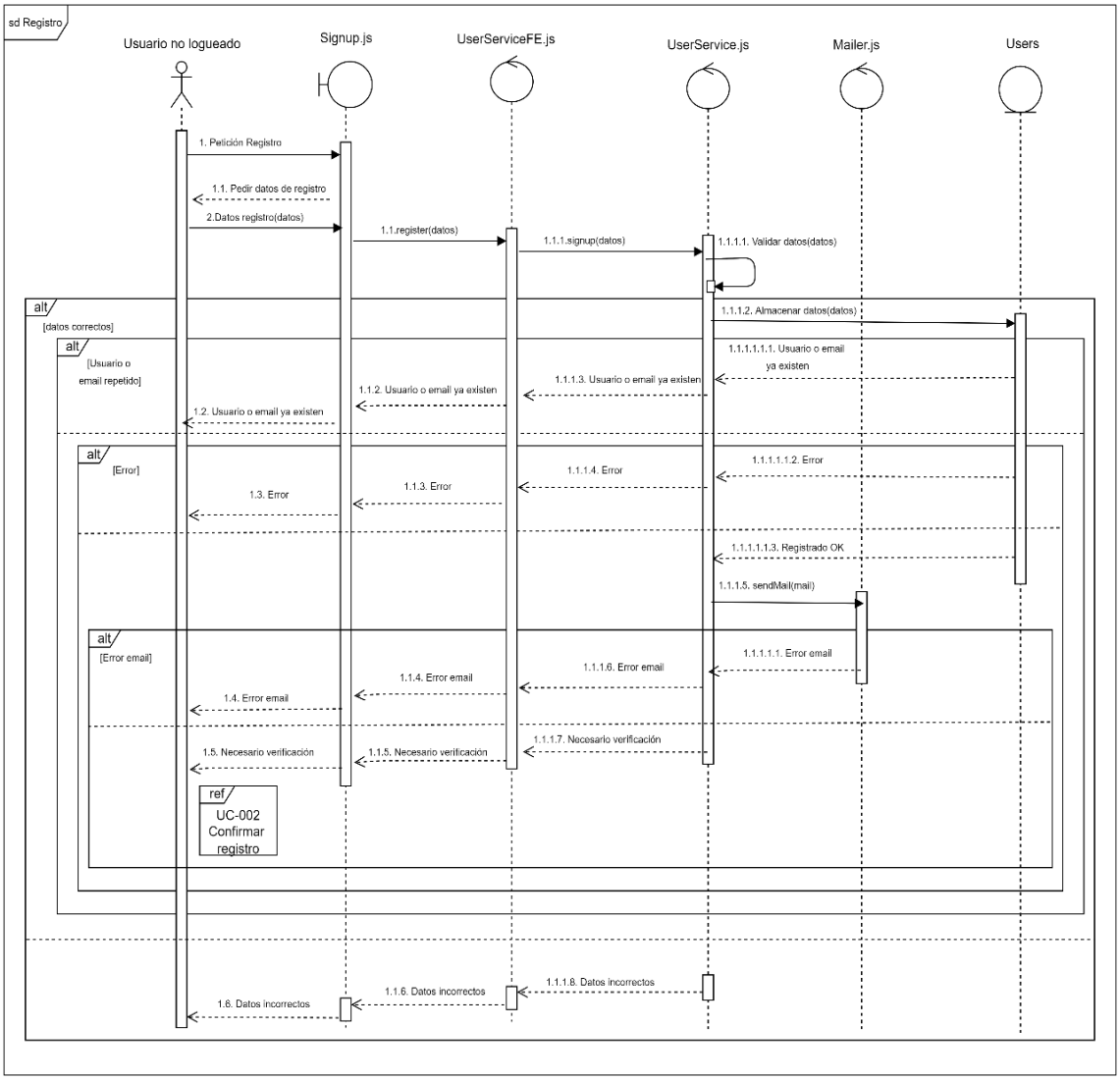


Ilustración 13. Diagrama de secuencia UC-001.

UC-002	**Confirmar registro**

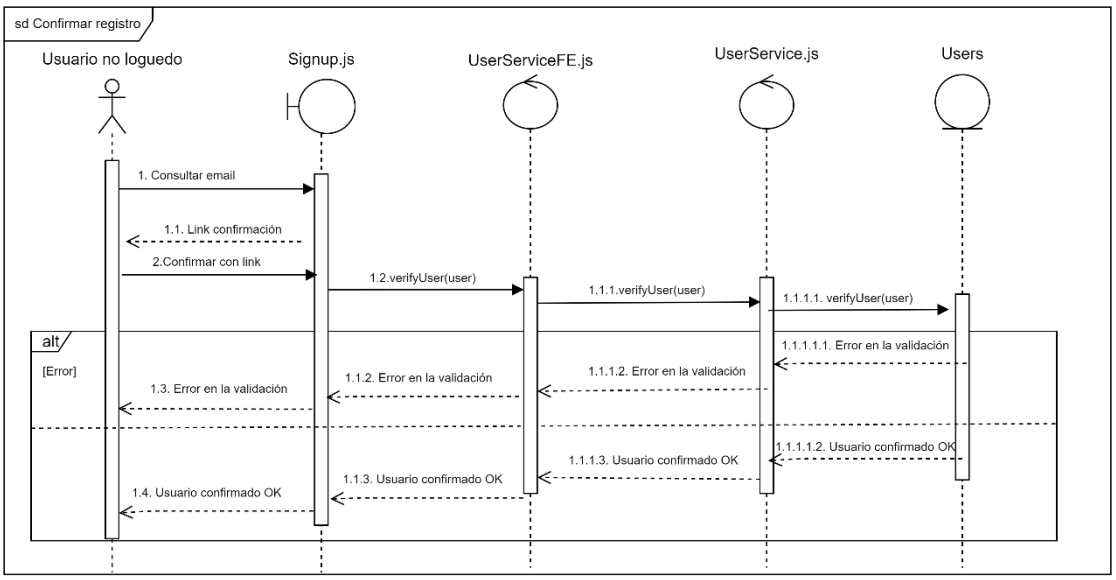


Ilustración 14. Diagrama de secuencia UC-002

UC-003	**Login**

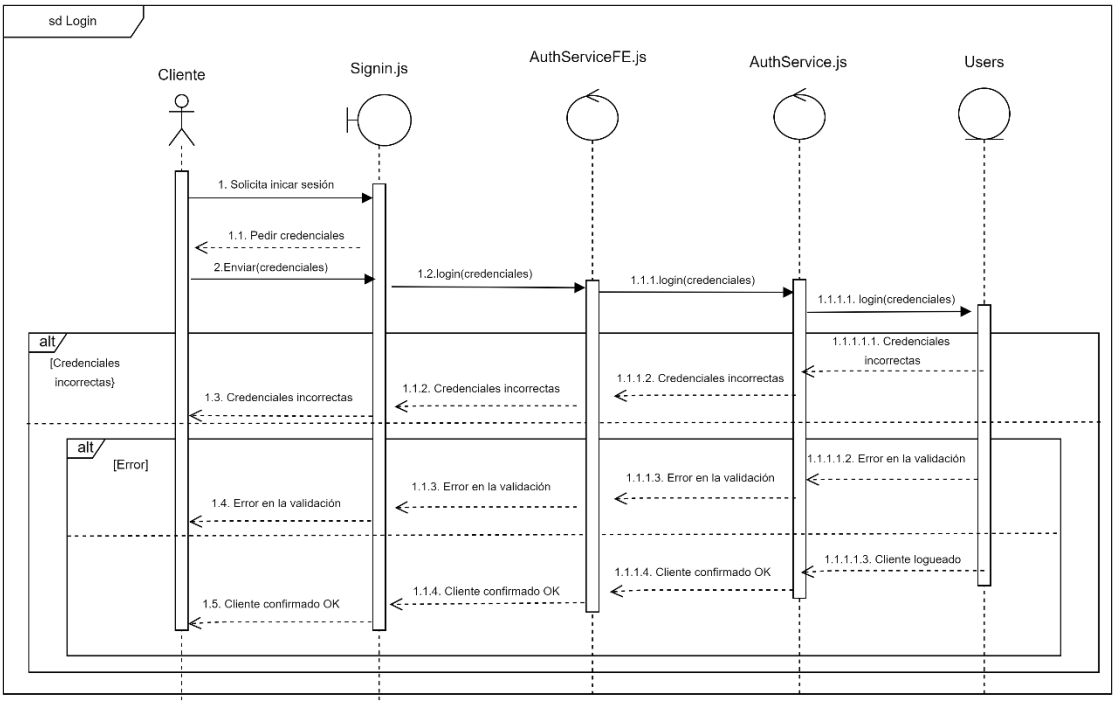


Ilustración 15. Diagrama de secuencia UC-003.

UC-004	**Restablecer contraseña**

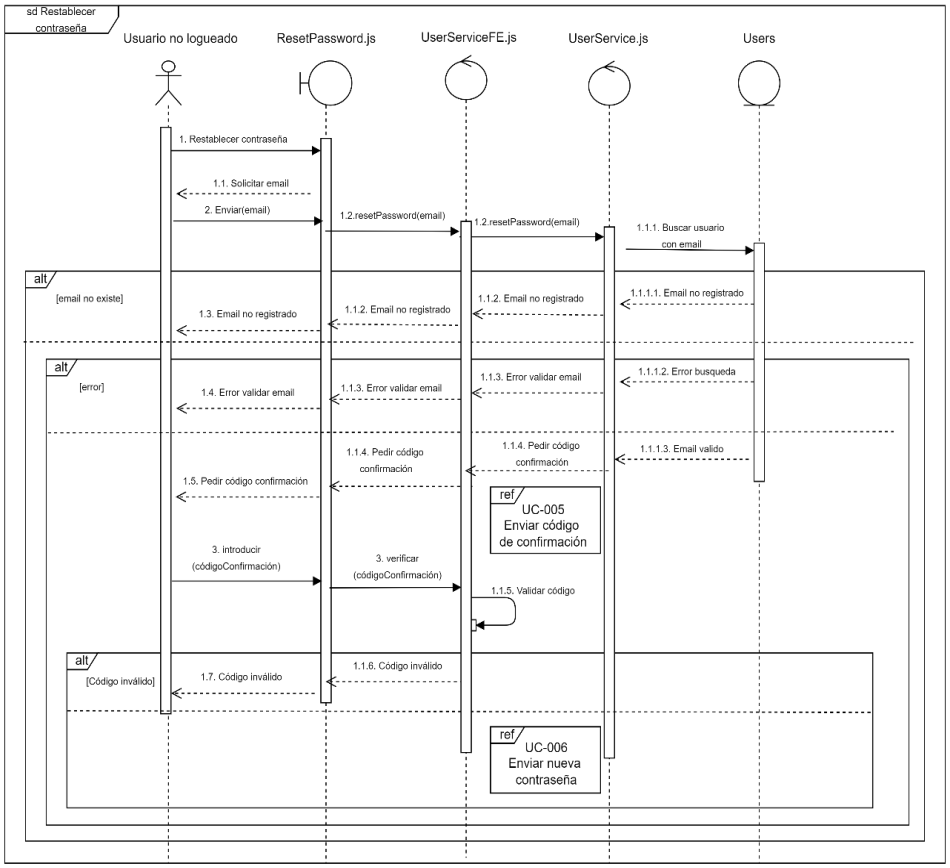


Ilustración 16. Diagrama de secuencia UC-004.

UC-005	**Enviar código de confirmación**

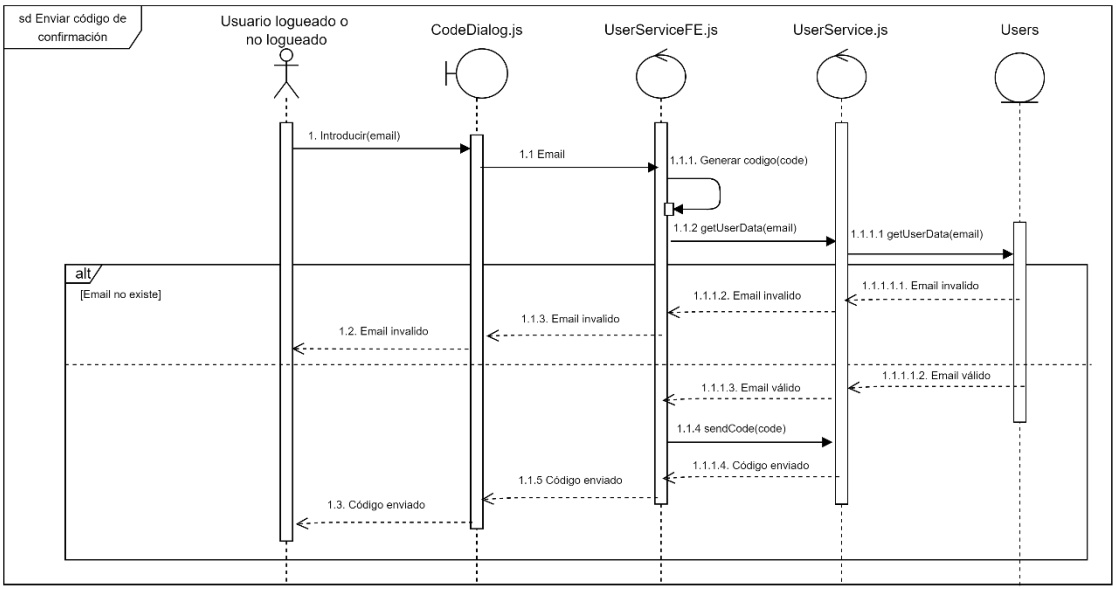


Ilustración 17. Diagrama de secuencia UC-005.

UC-006	**Enviar nueva contraseña**

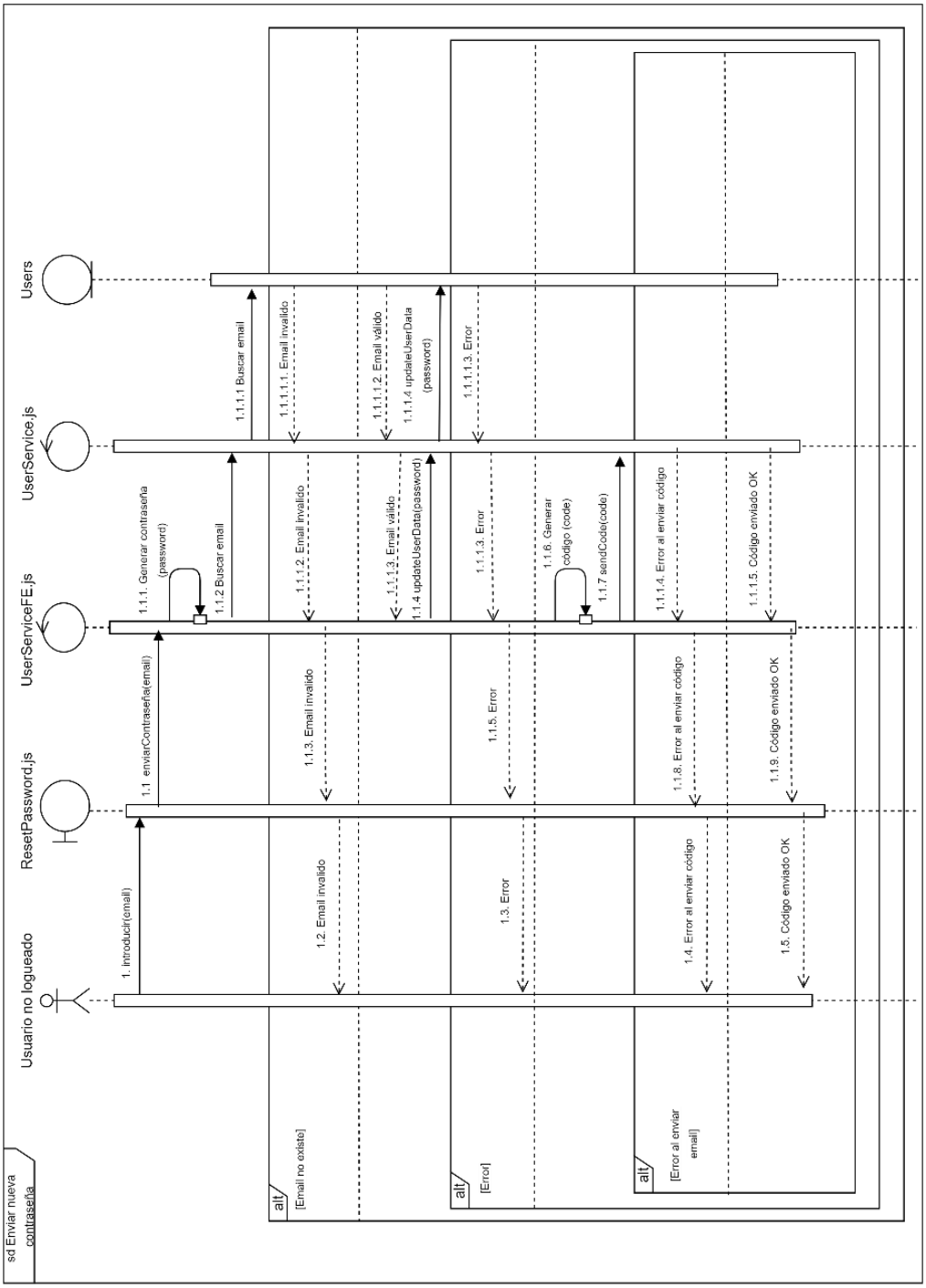


Ilustración 18. Diagrama de secuencia UC-006.

UC-007	**Modificar información perfil**

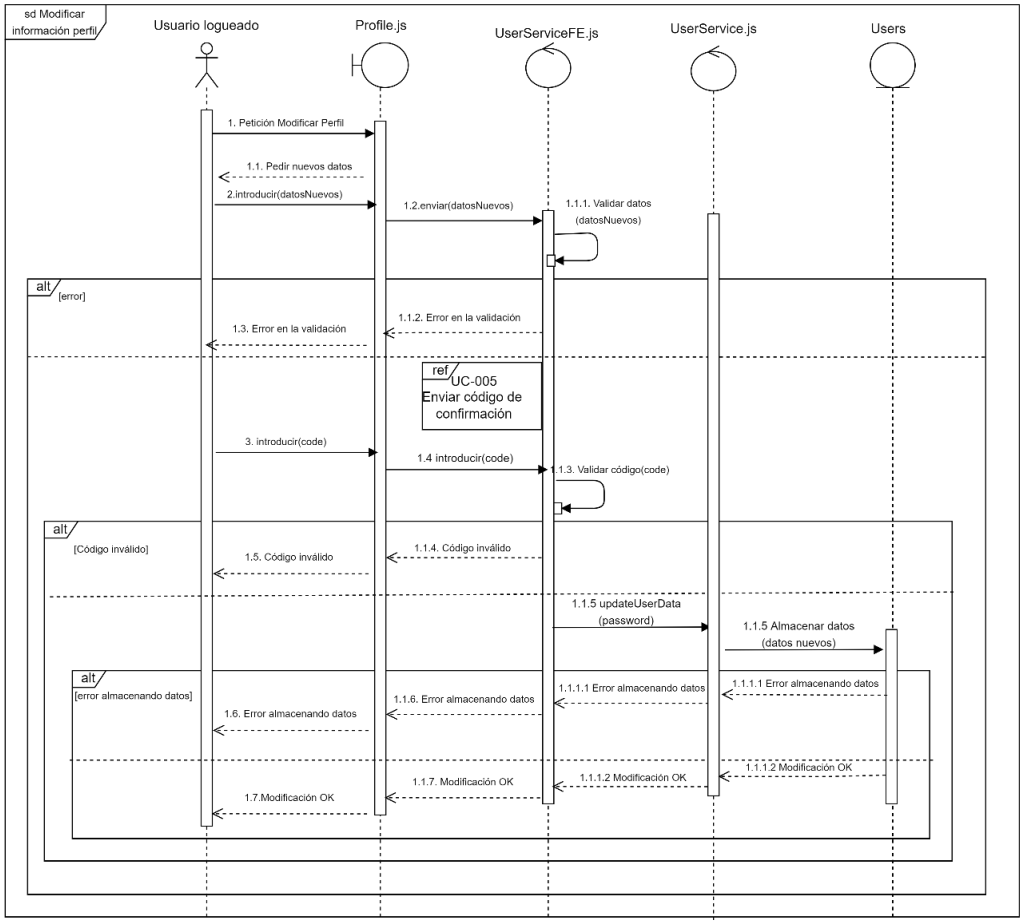


Ilustración 19. Diagrama de secuencia UC-007.

UC-008	**Logout**

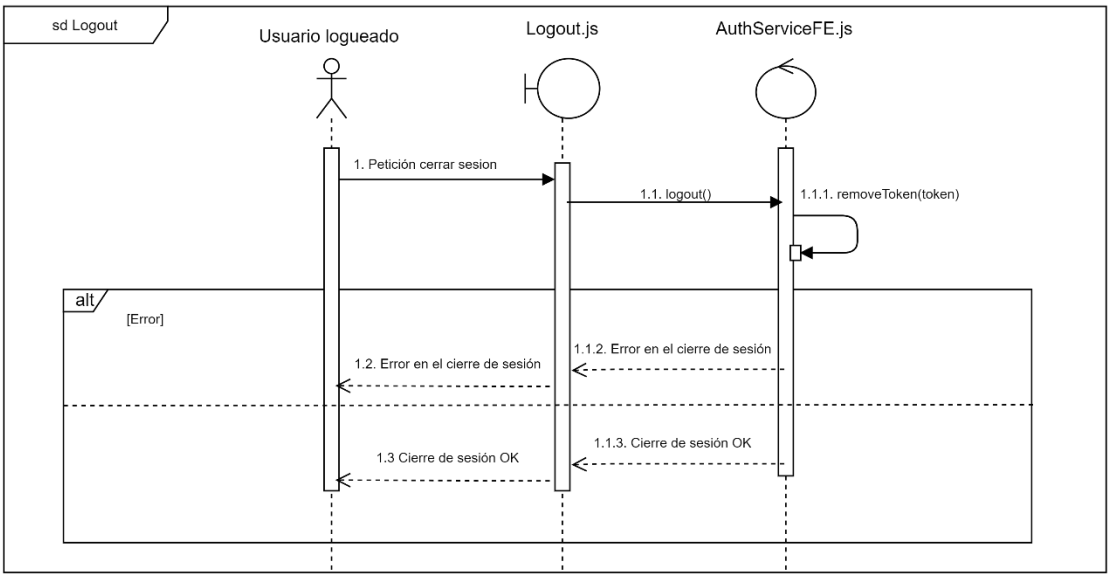


Ilustración 20. Diagrama de secuencia UC-008.

UC-009	**Consultar información perfil**

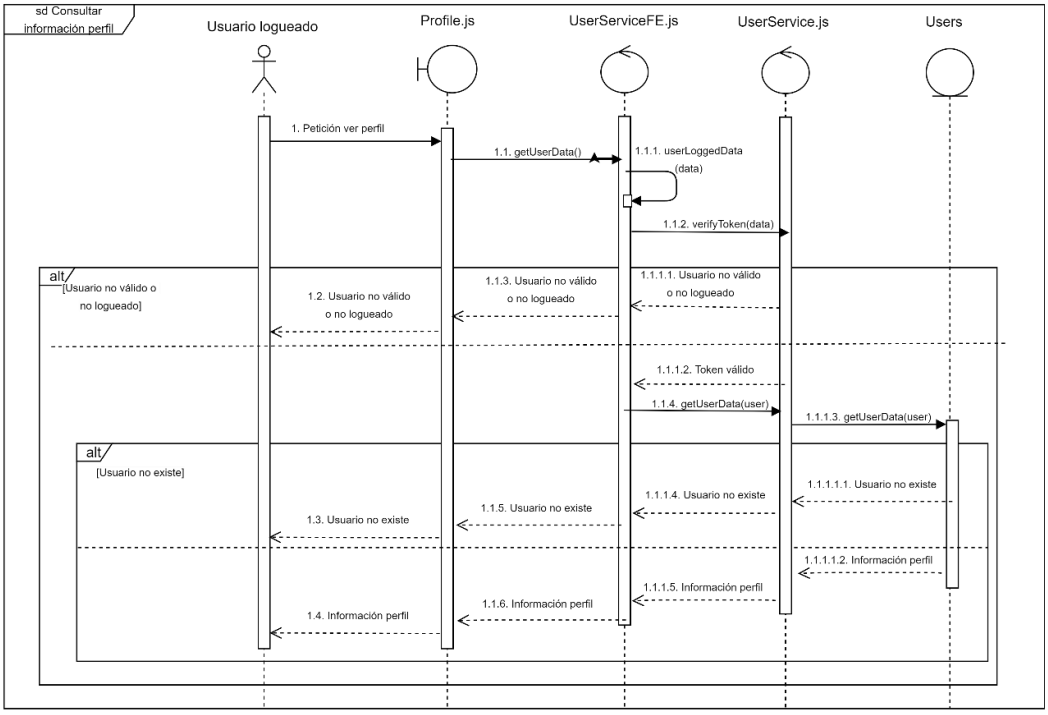


Ilustración 21. Diagrama de secuencia UC-009.

UC-010	**Consultar estadísticas personales**

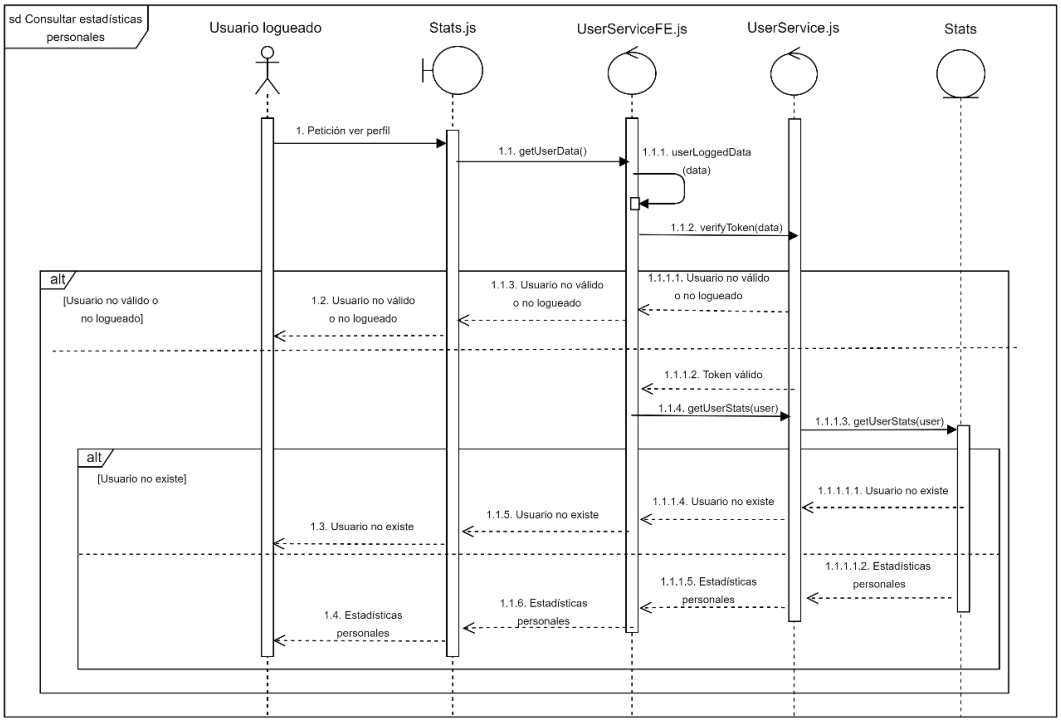


Ilustración 22. Diagrama de secuencia UC-010.

UC-011	**Eliminar cuenta**

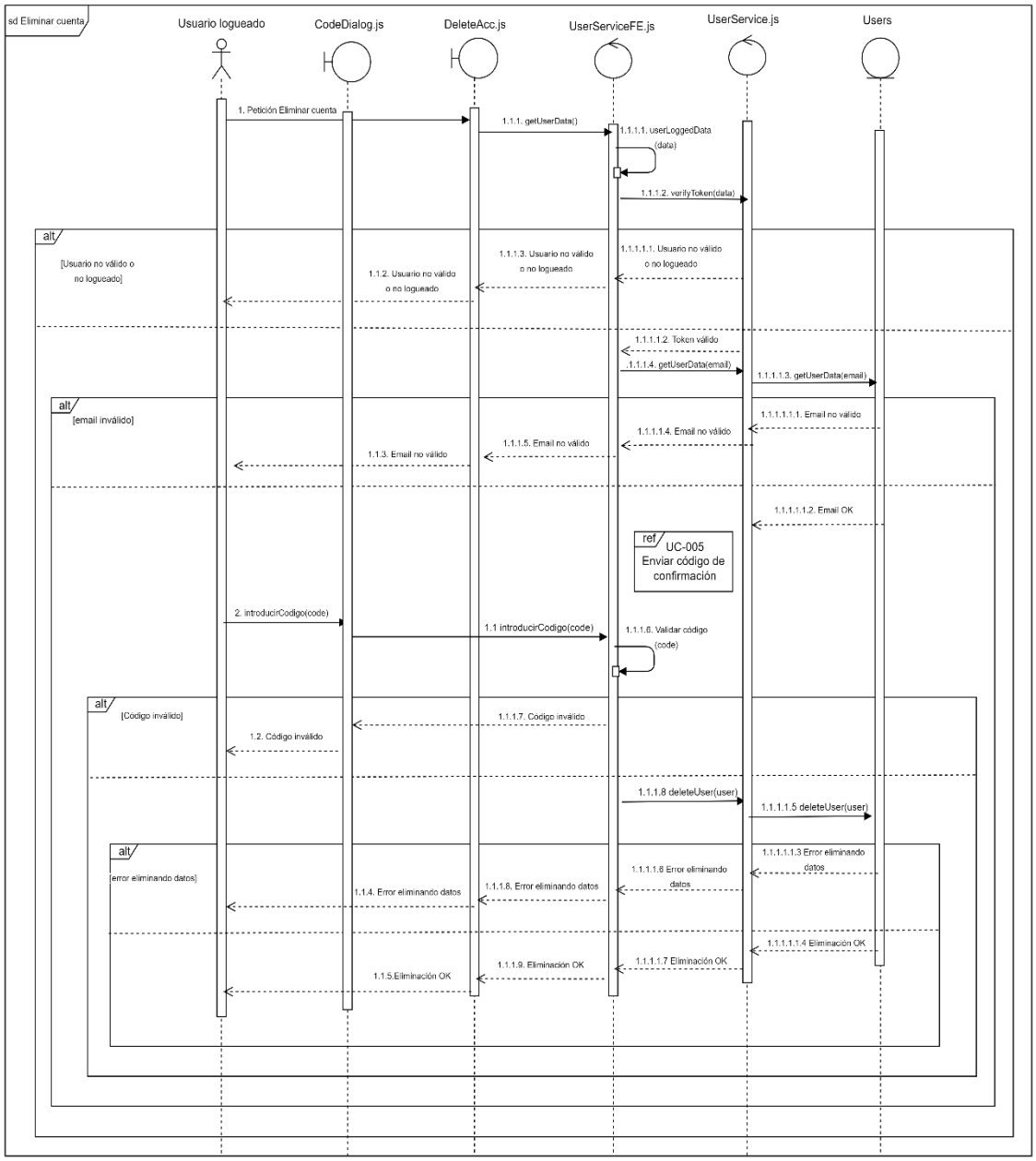


Ilustración 23. Diagrama de secuencia UC-011.

UC-012	**Cambiar configuración panel de control**

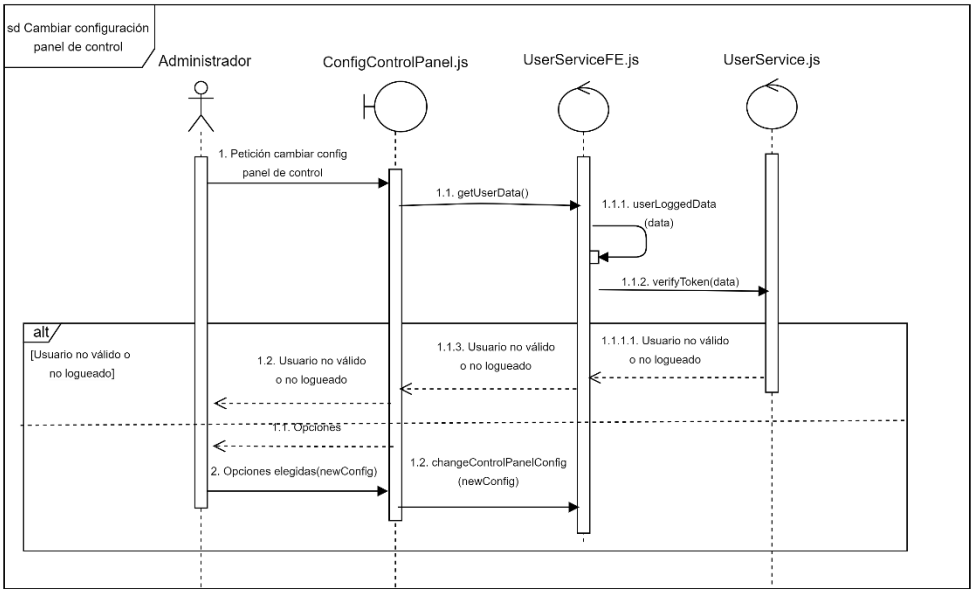


Ilustración 24. Diagrama de secuencia UC-012.

UC-013	**Consultar estadísticas generales**

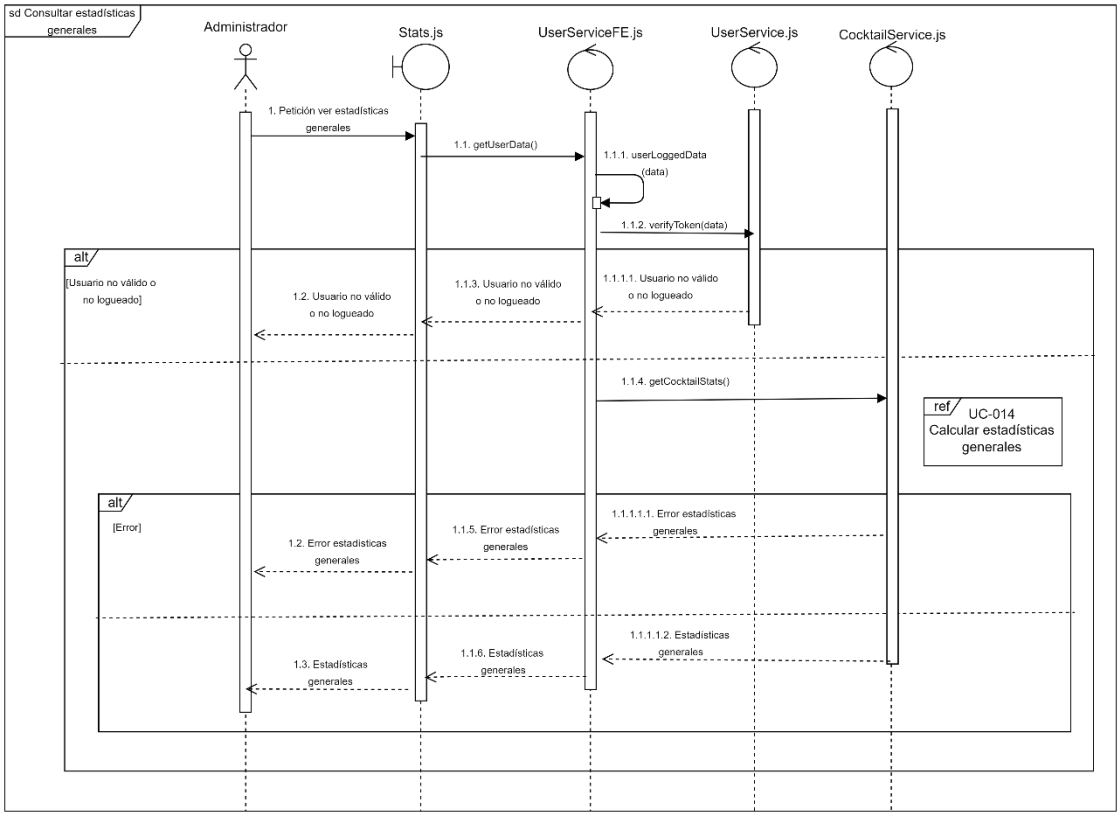


Ilustración 25. Diagrama de secuencia UC-013.

UC-014	**Calcular estadísticas generales**

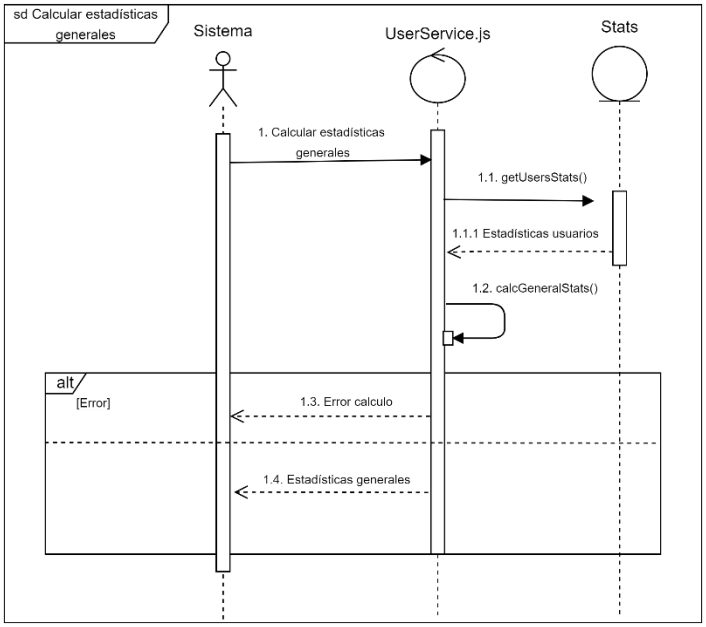


Ilustración 26. Diagrama de secuencia UC-014.

2.5.1. PAQUETE GESTIÓN DE BEBIDAS

UC-015	**Añadir bebida**

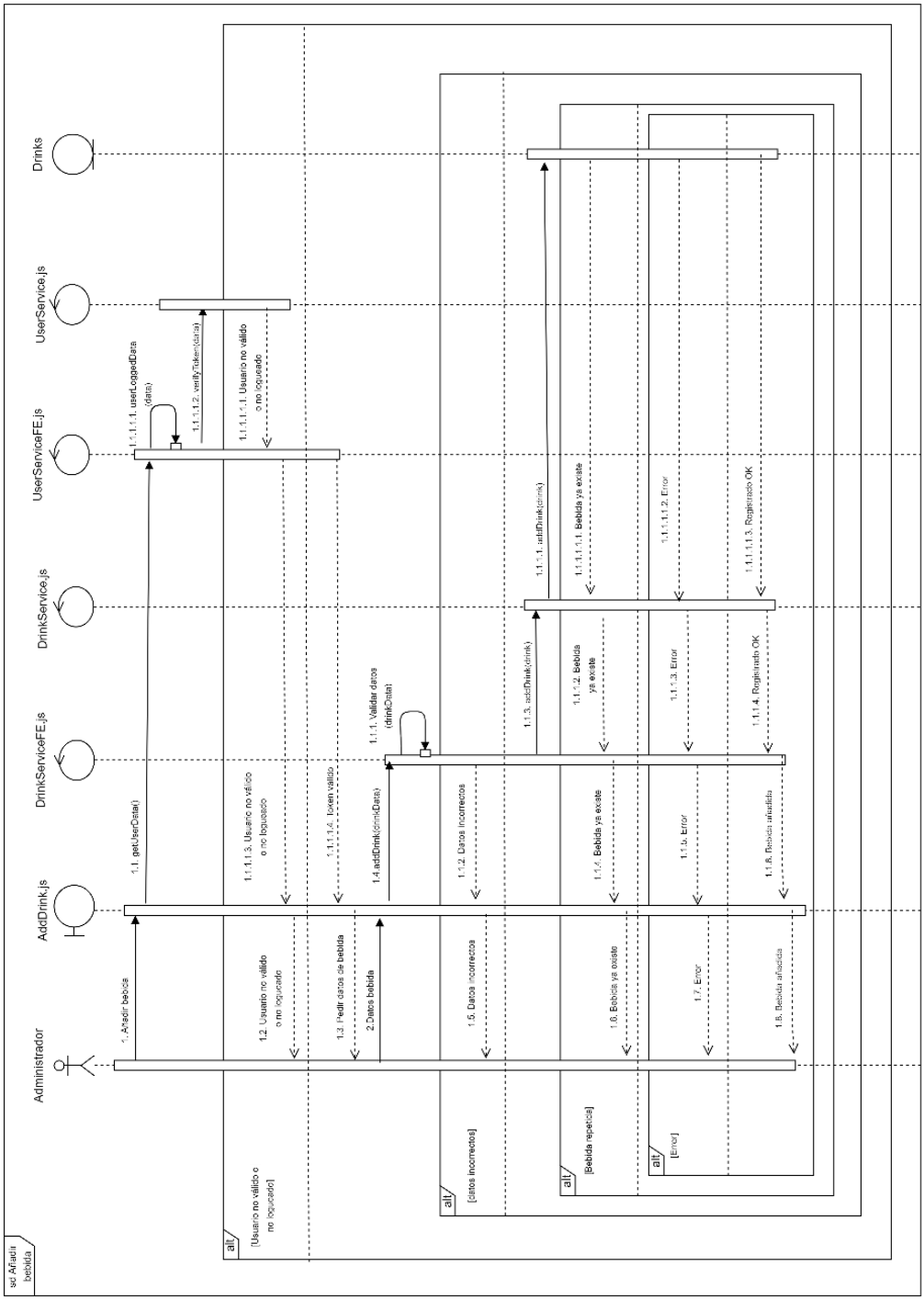


Ilustración 27. Diagrama de secuencia UC-015.

UC-016	**Modificar bebida**

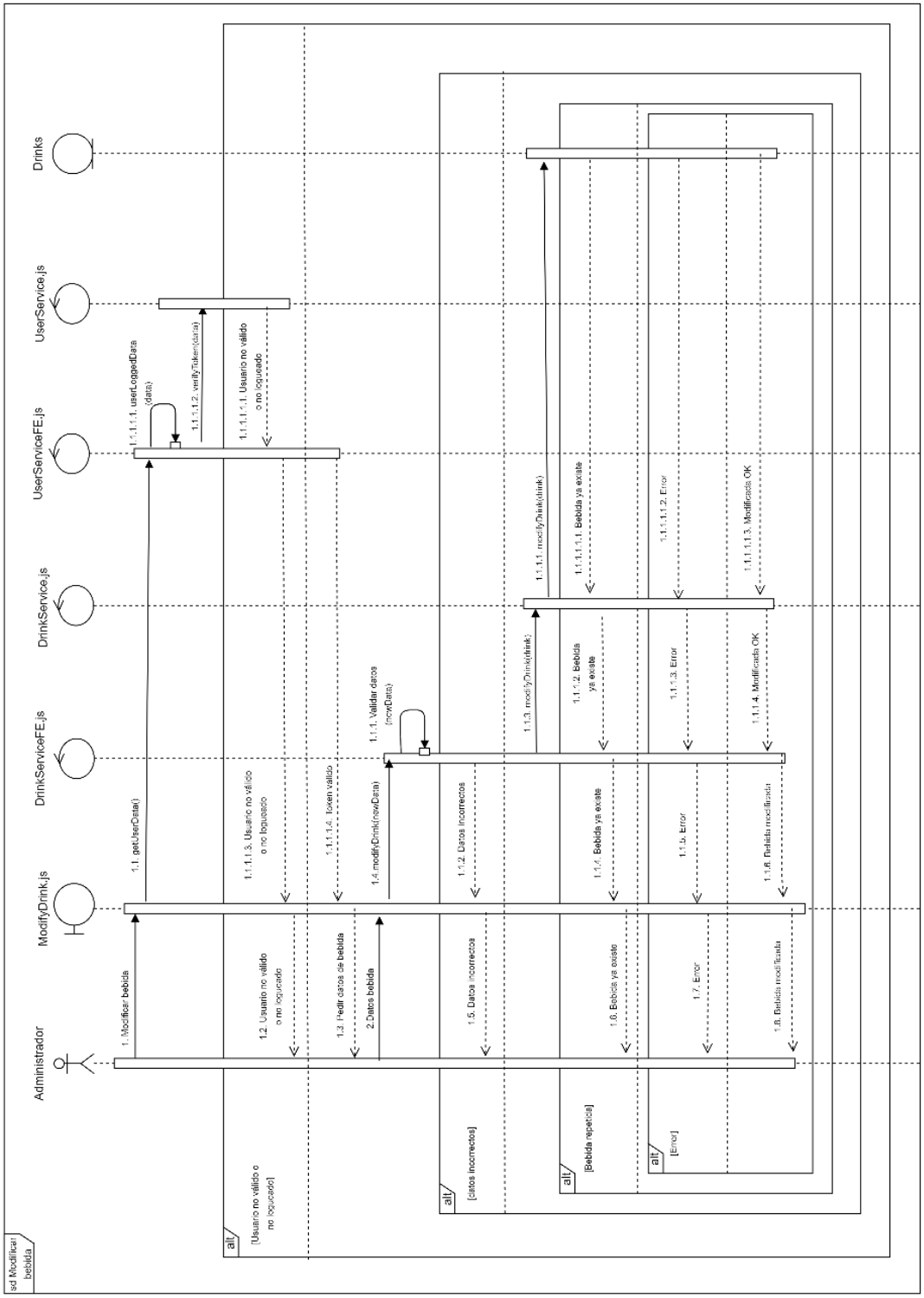


Ilustración 28. Diagrama de secuencia UC-016.

UC-017	**Eliminar bebida**

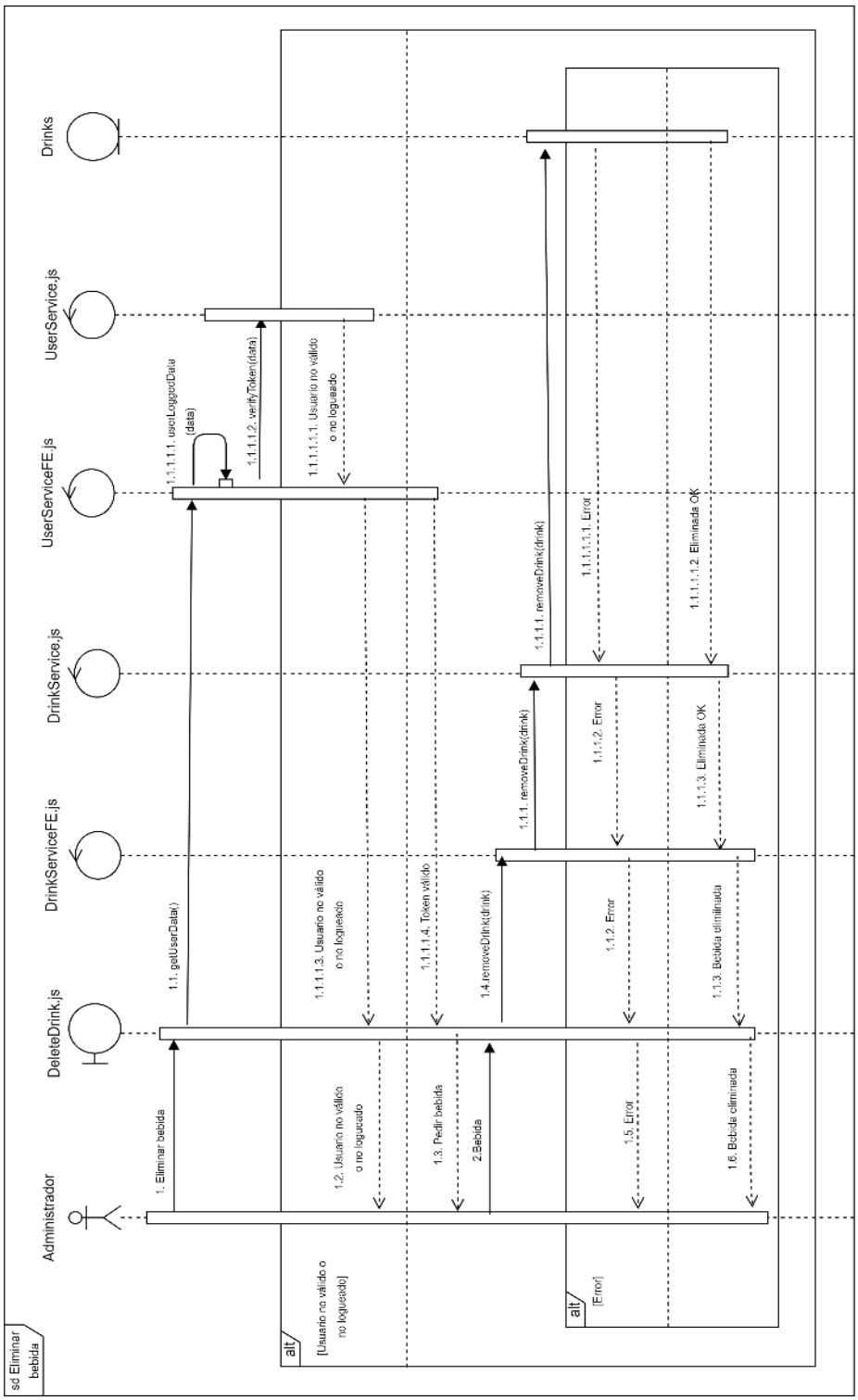


Ilustración 29. Diagrama de secuencia UC-017.

UC-018	**Activar bebida**

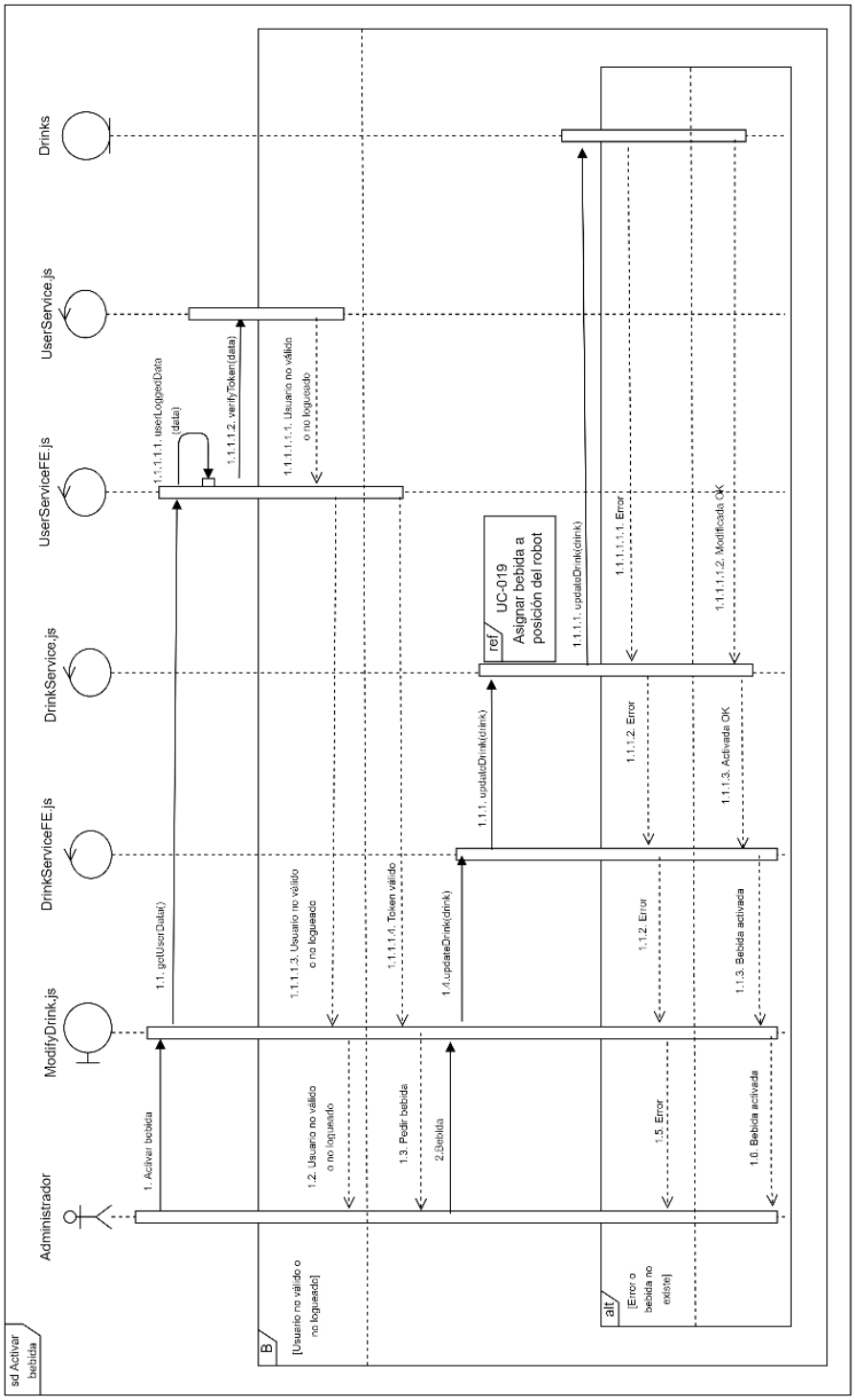
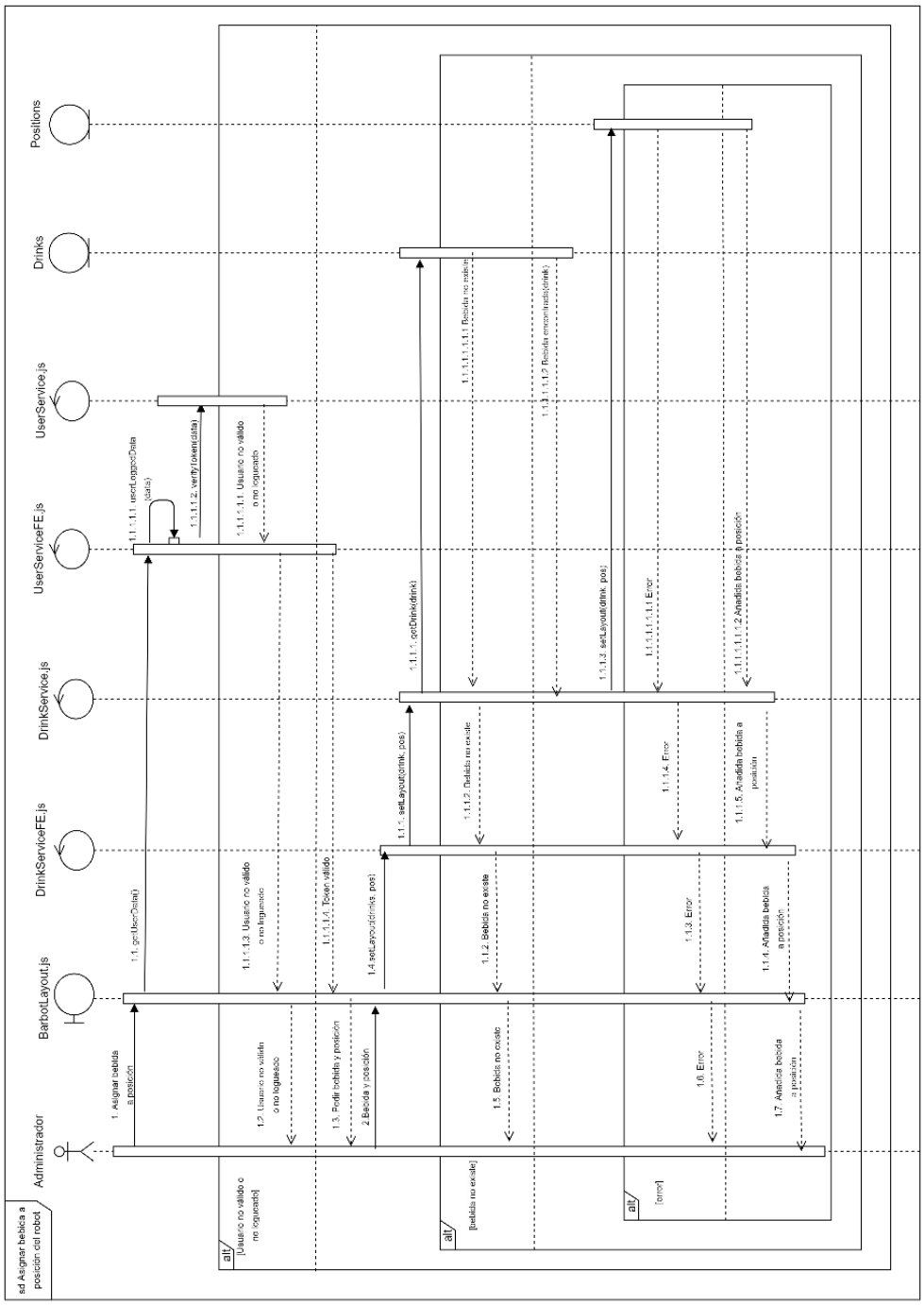


Ilustración 30. Diagrama de secuencia UC-018.

UC-019	**Activar bebida a posición del robot**


sd Asignar bebida a posición del robot
Administrador
BarbotLayout.js
DrinkServiceFE.js
DrinkService.js
UserServiceFE.js
UserService.js
Drinks
Positions
1. Asignar bebida a posición
1.1. getUserData()
1.1.1.1.1. userLoggedData (data)
1.1.1.1.2. verifyToken(data)
1.1.1.1.3. Usuario no válido o no logueado
1.1.1.1.4. Token válido
1.2. Usuario no válido o no logueado
1.3. Pedir bebida y posición
2.Bebida y posición
1.4.setLayoutDrinks(pos)
1.1.1.1.5. getDrink(drink)
1.1.1.1.6. setLayout(drink, pos)
1.1.1.1.

UC-020	**Consultar bebidas**

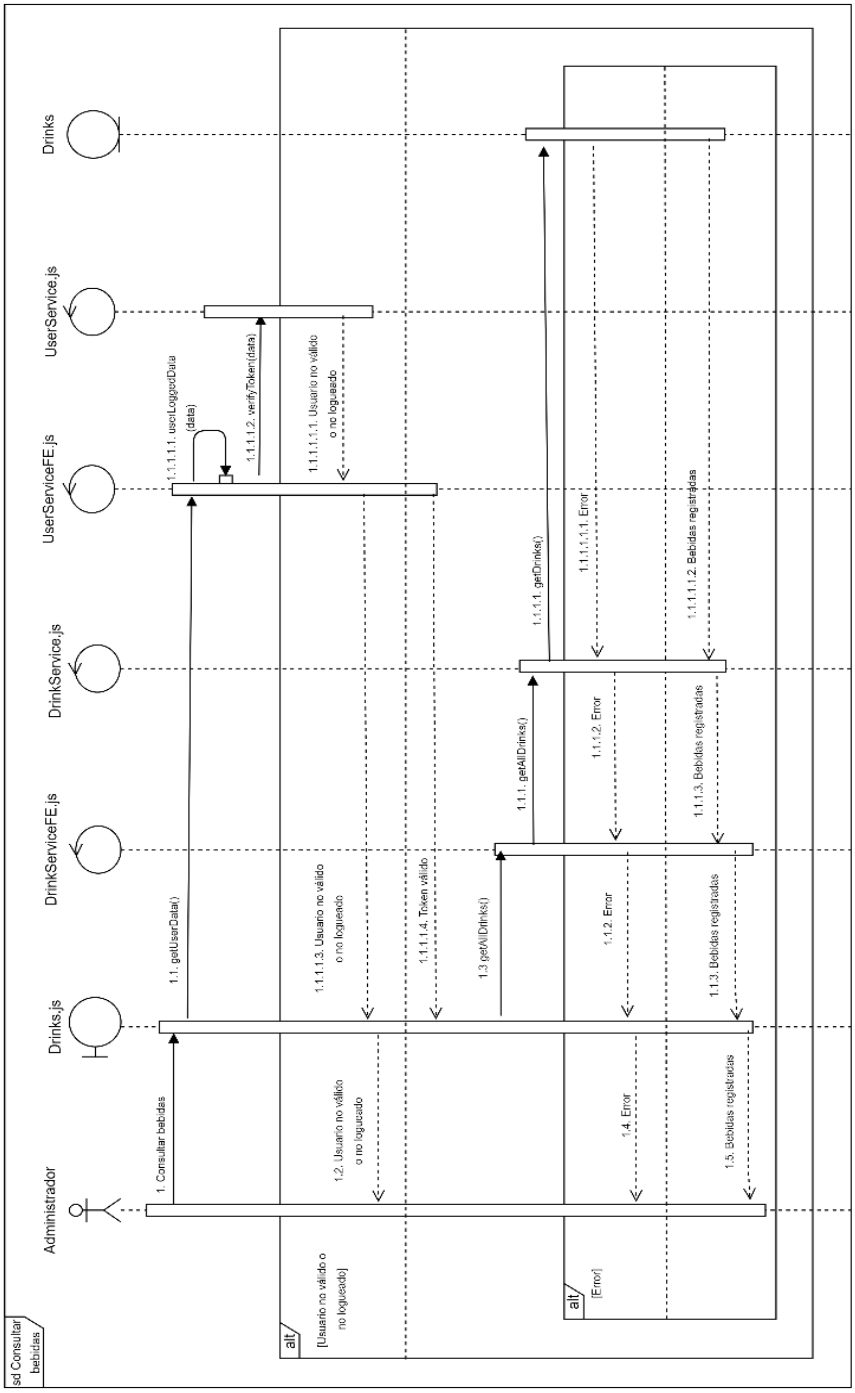


Ilustración 32. Diagrama de secuencia UC-020.

UC-021	**Buscar bebidas**

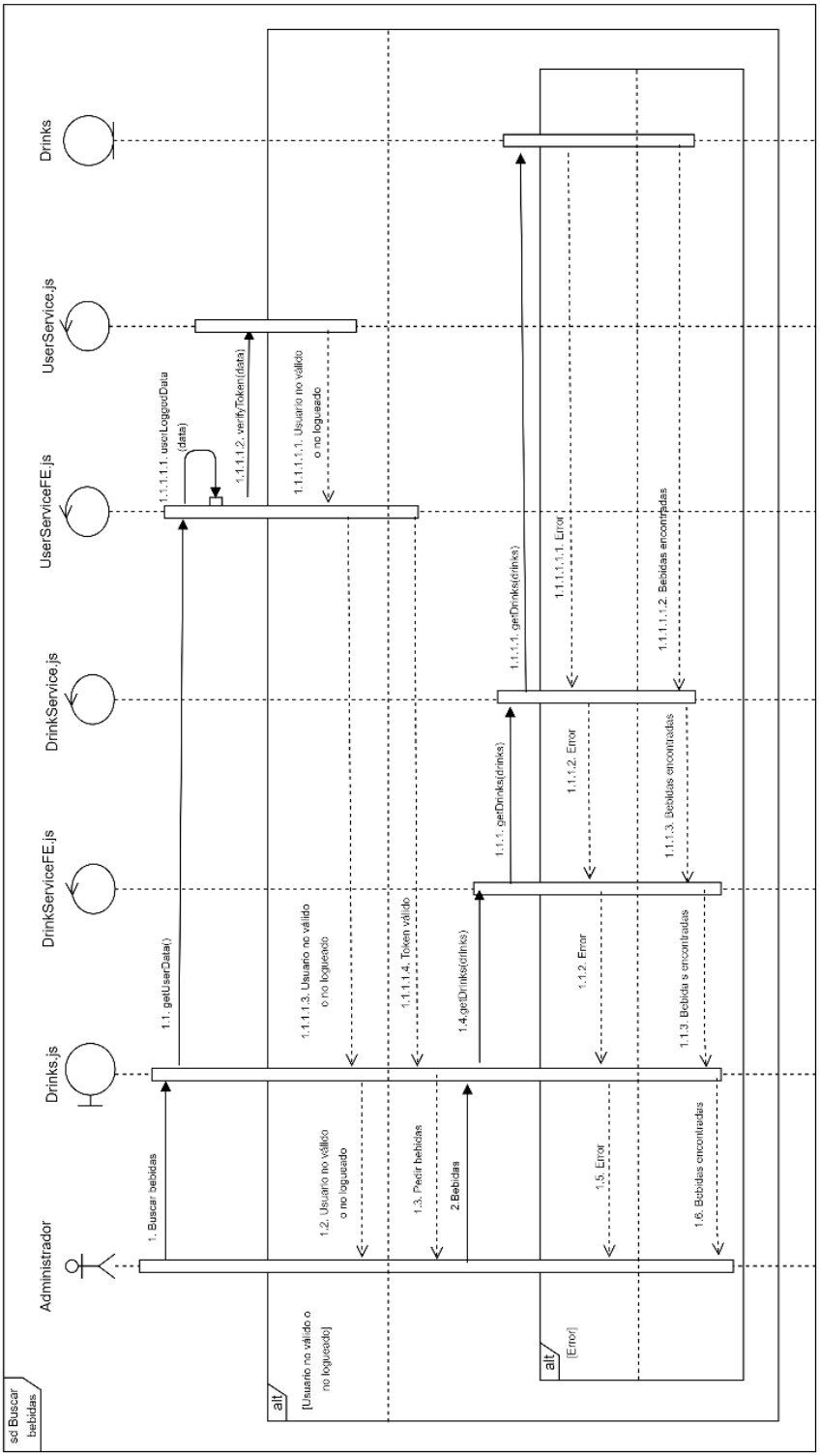


Ilustración 33. Diagrama de secuencia UC-021.

UC-022	**Consultar bebidas posiciones robot**

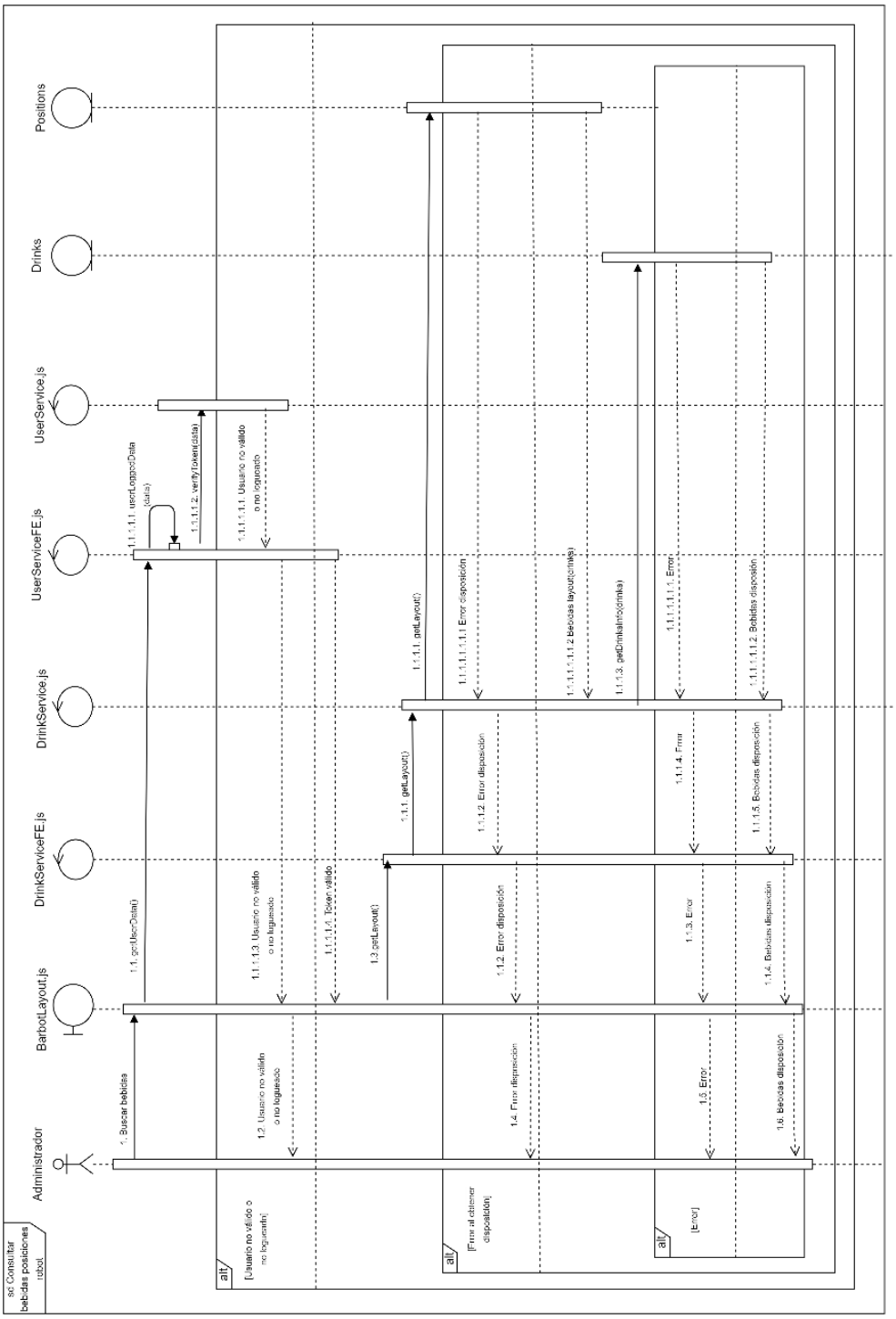


Ilustración 34. Diagrama de secuencia UC-022.

2.5.1. PAQUETE GESTIÓN DE RECETAS

UC-023	**Añadir receta**

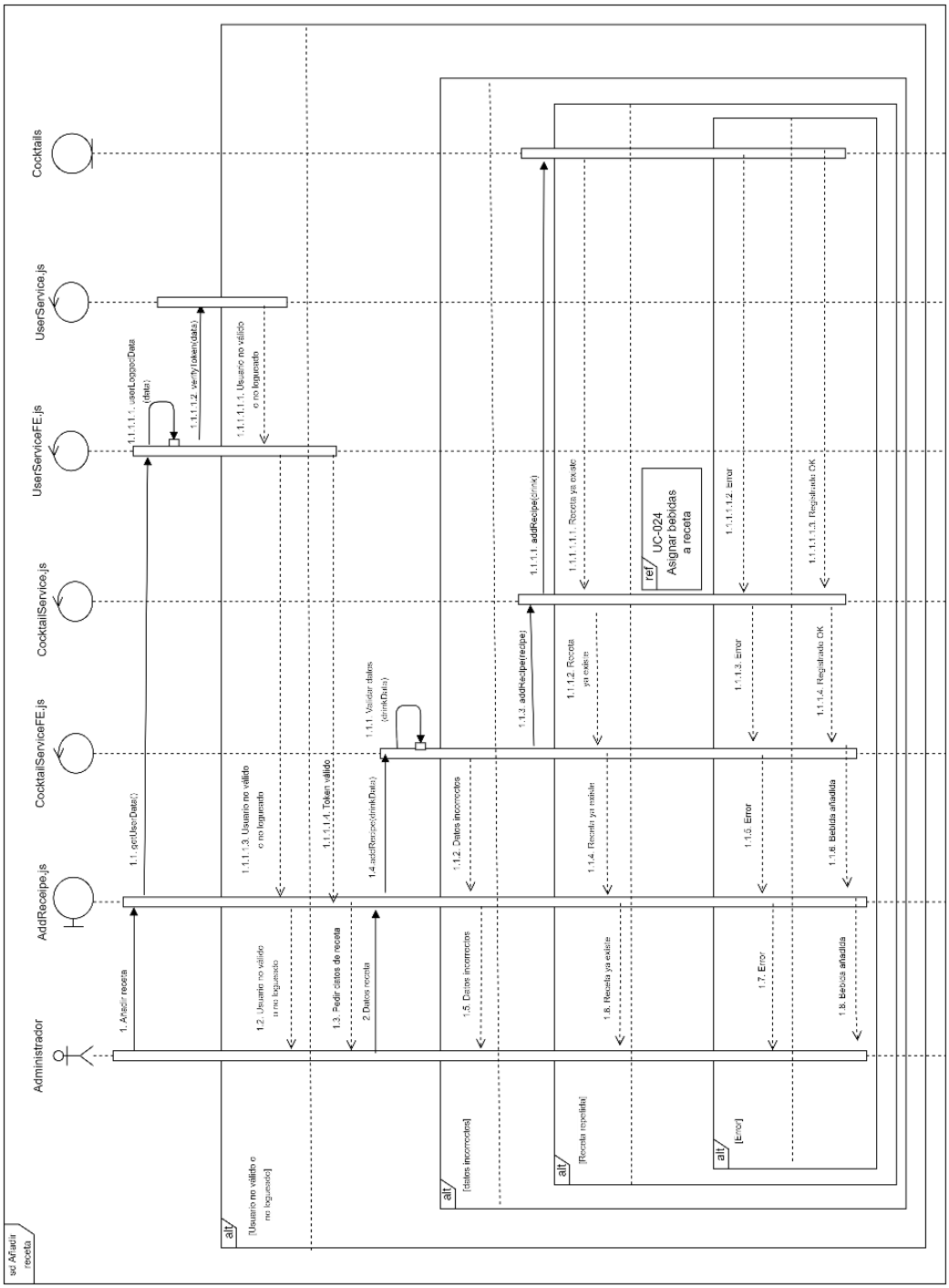


Ilustración 35. Diagrama de secuencia UC-023.

UC-024	**Asignar bebidas a receta**

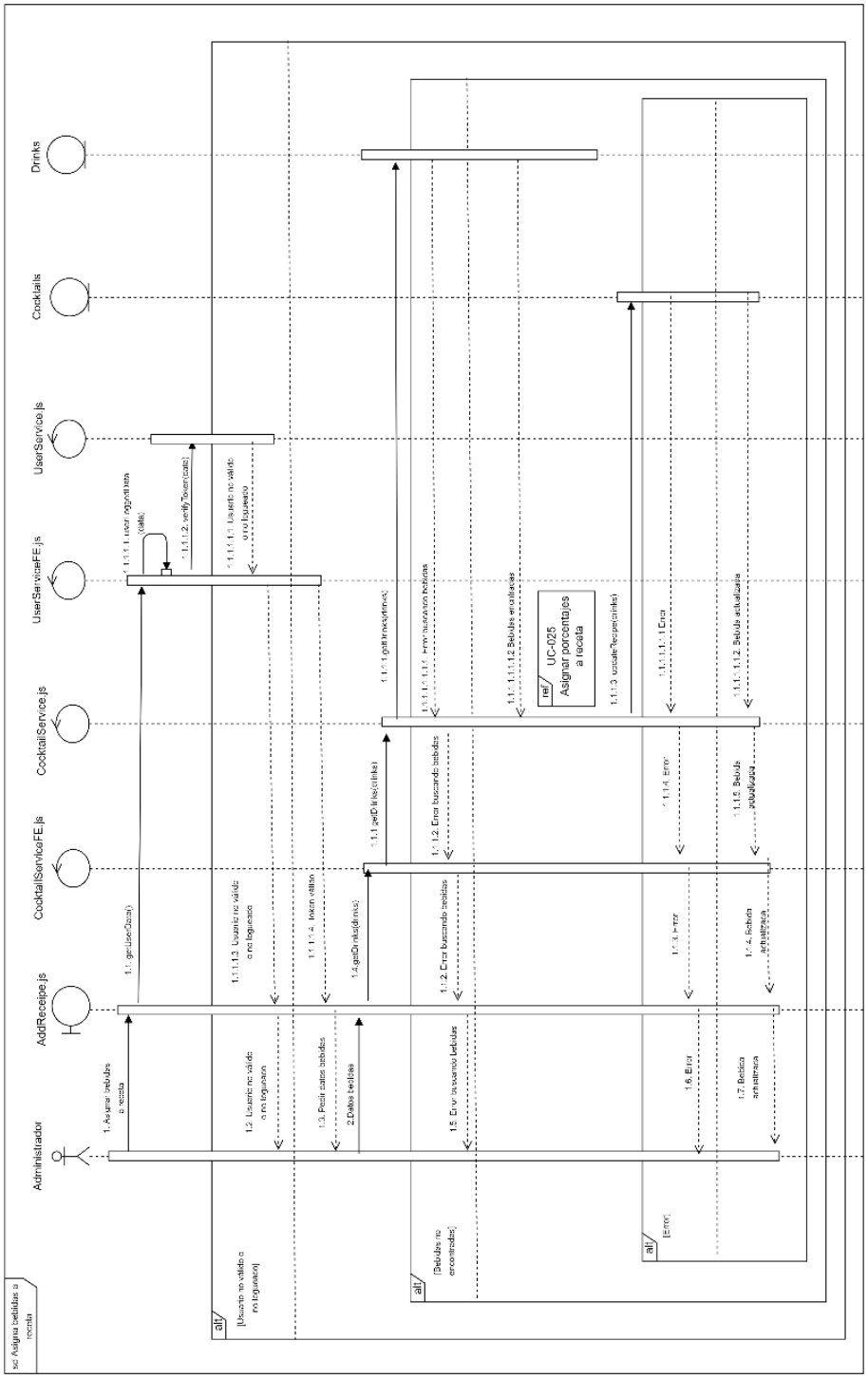


Ilustración 36. Diagrama de secuencia UC-024.

UC-025	**Asignar porcentajes a receta**

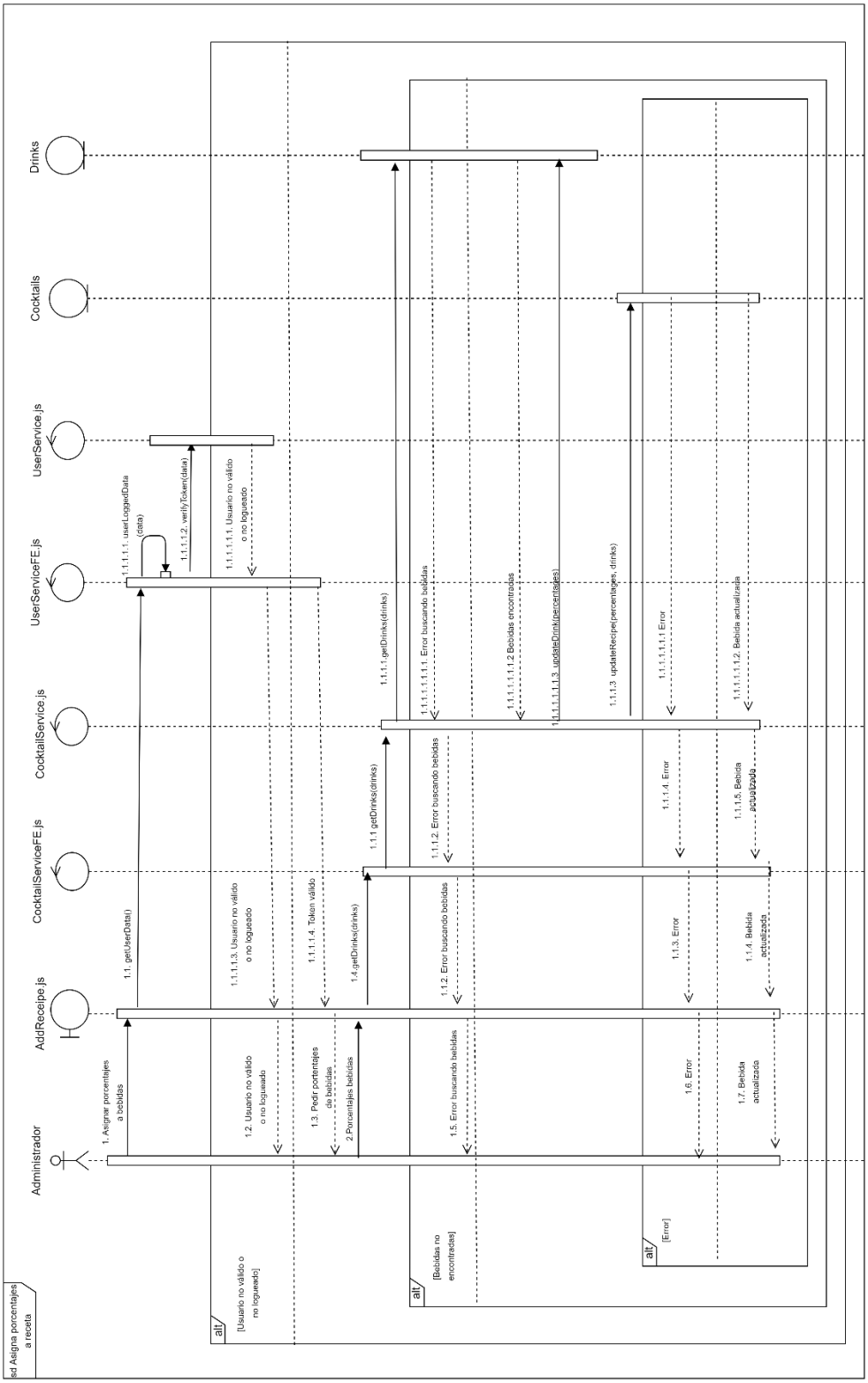


Ilustración 37. Diagrama de secuencia UC-025.

UC-026	**Modificar receta**

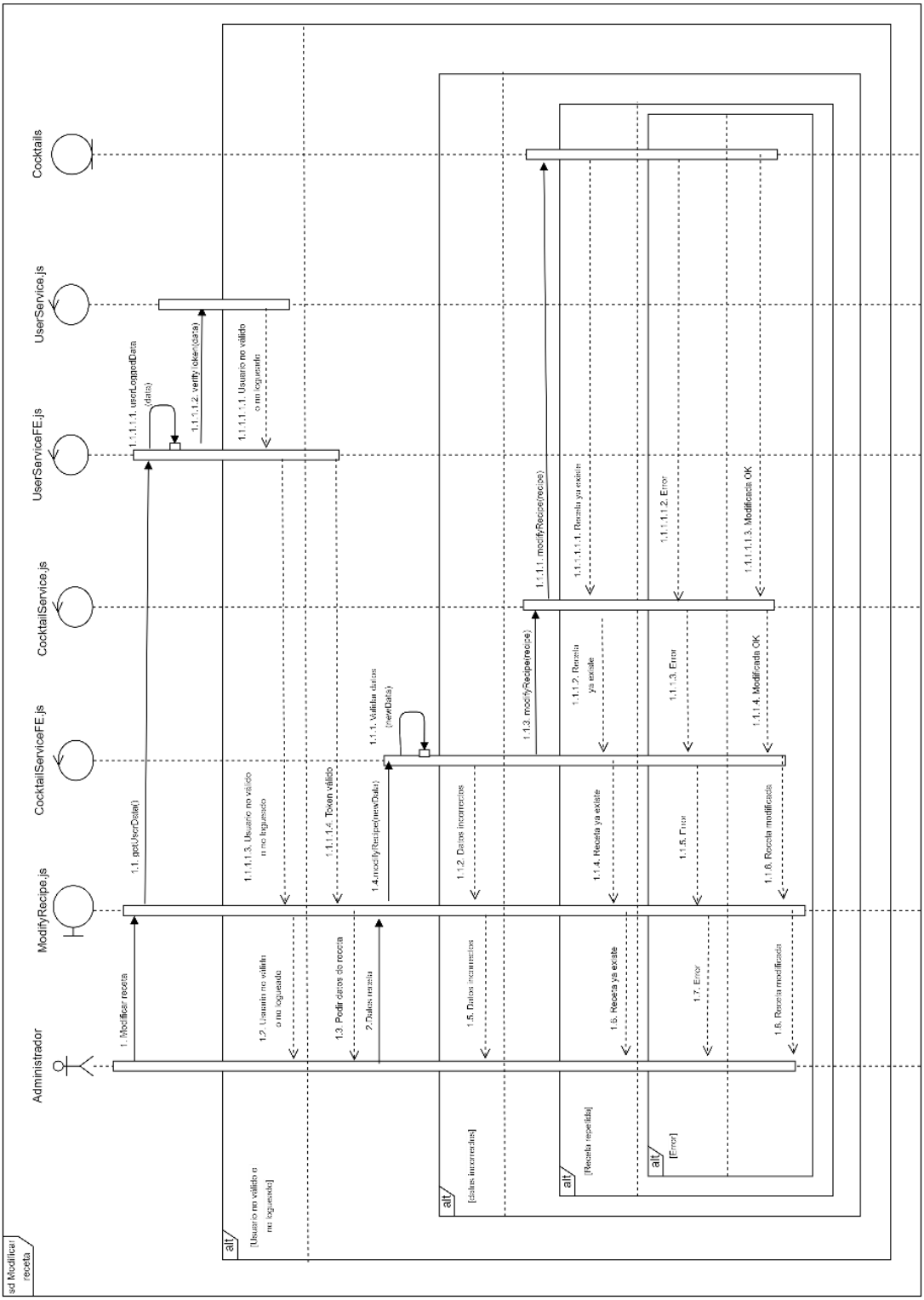


Ilustración 38. Diagrama de secuencia UC-026.

UC-027	**Eliminar receta**

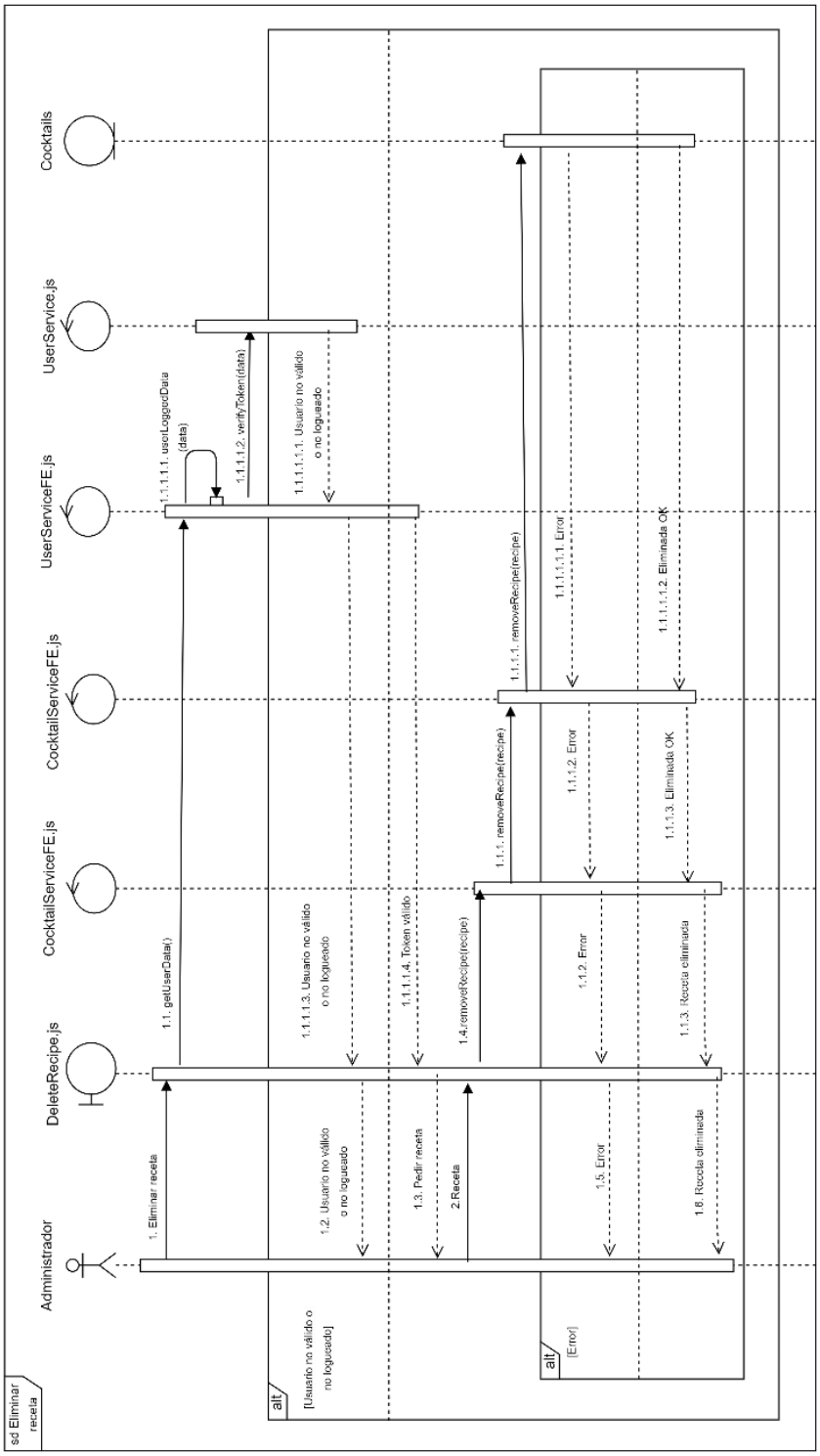


Ilustración 39. Diagrama de secuencia UC-027.

UC-028	**Eliminar receta**

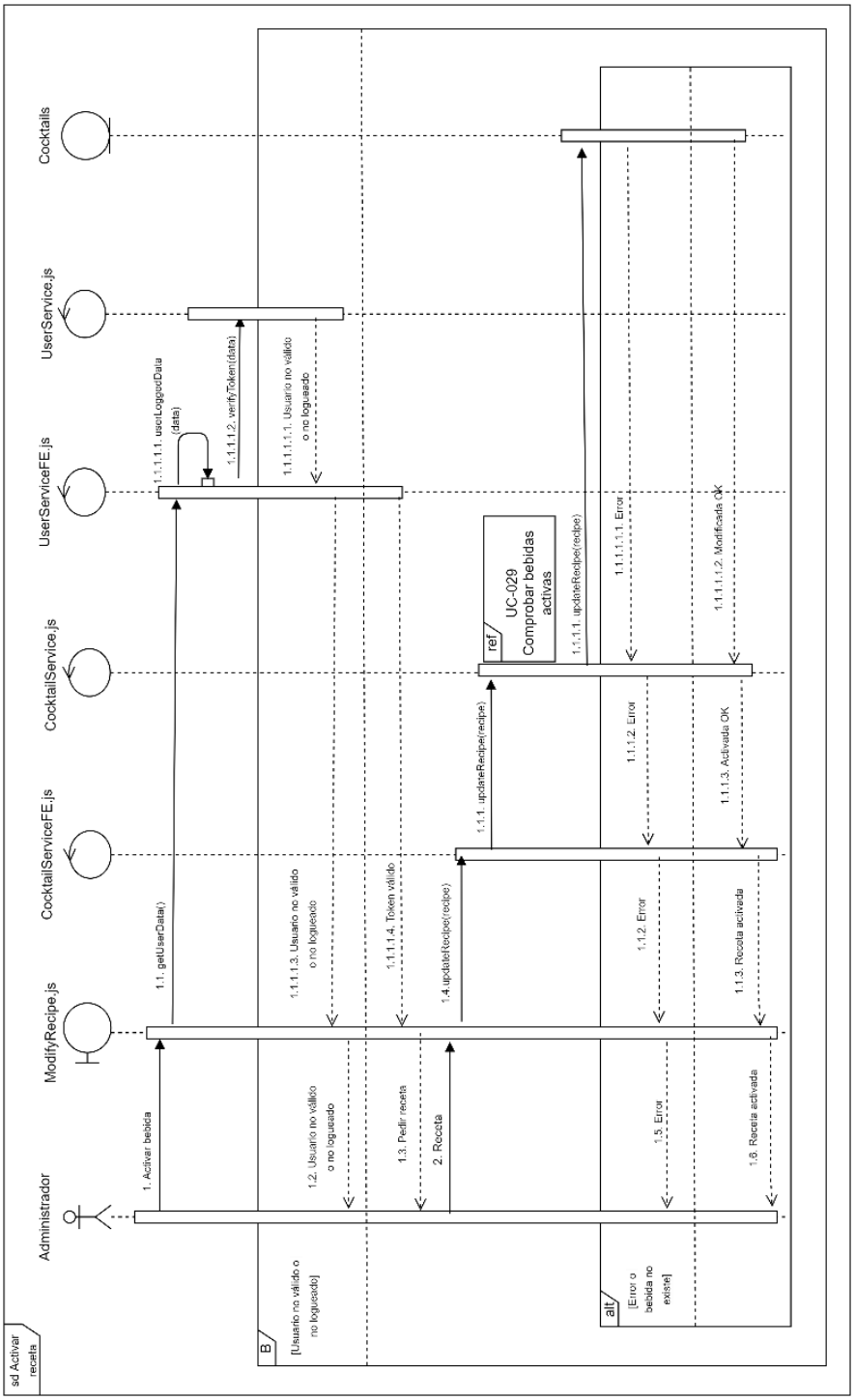


Ilustración 40. Diagrama de secuencia UC-028.

UC-029	**Comprobar bebidas activas**

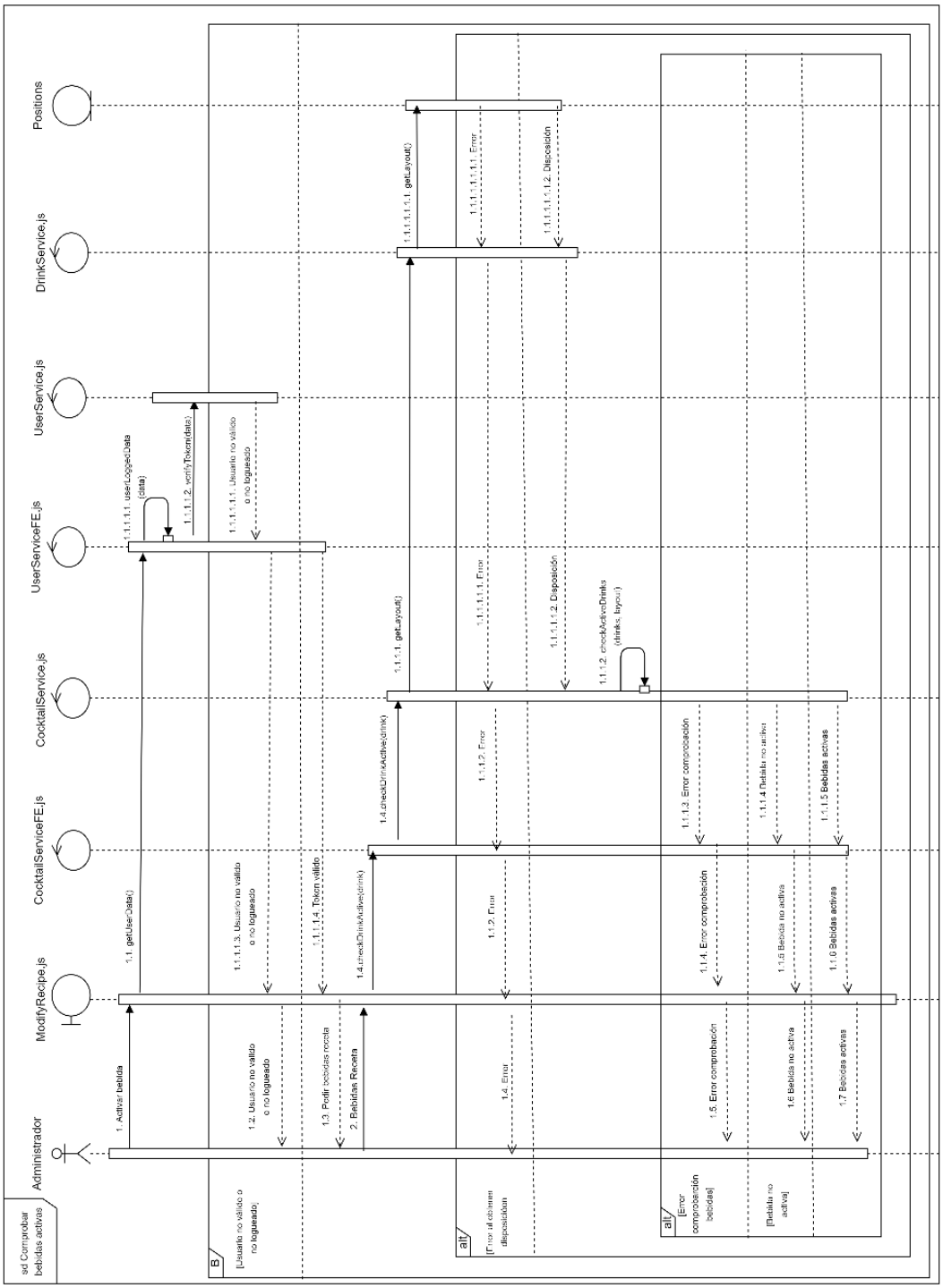


Ilustración 41. Diagrama de secuencia UC-029.

UC-030	**Consultar recetas**

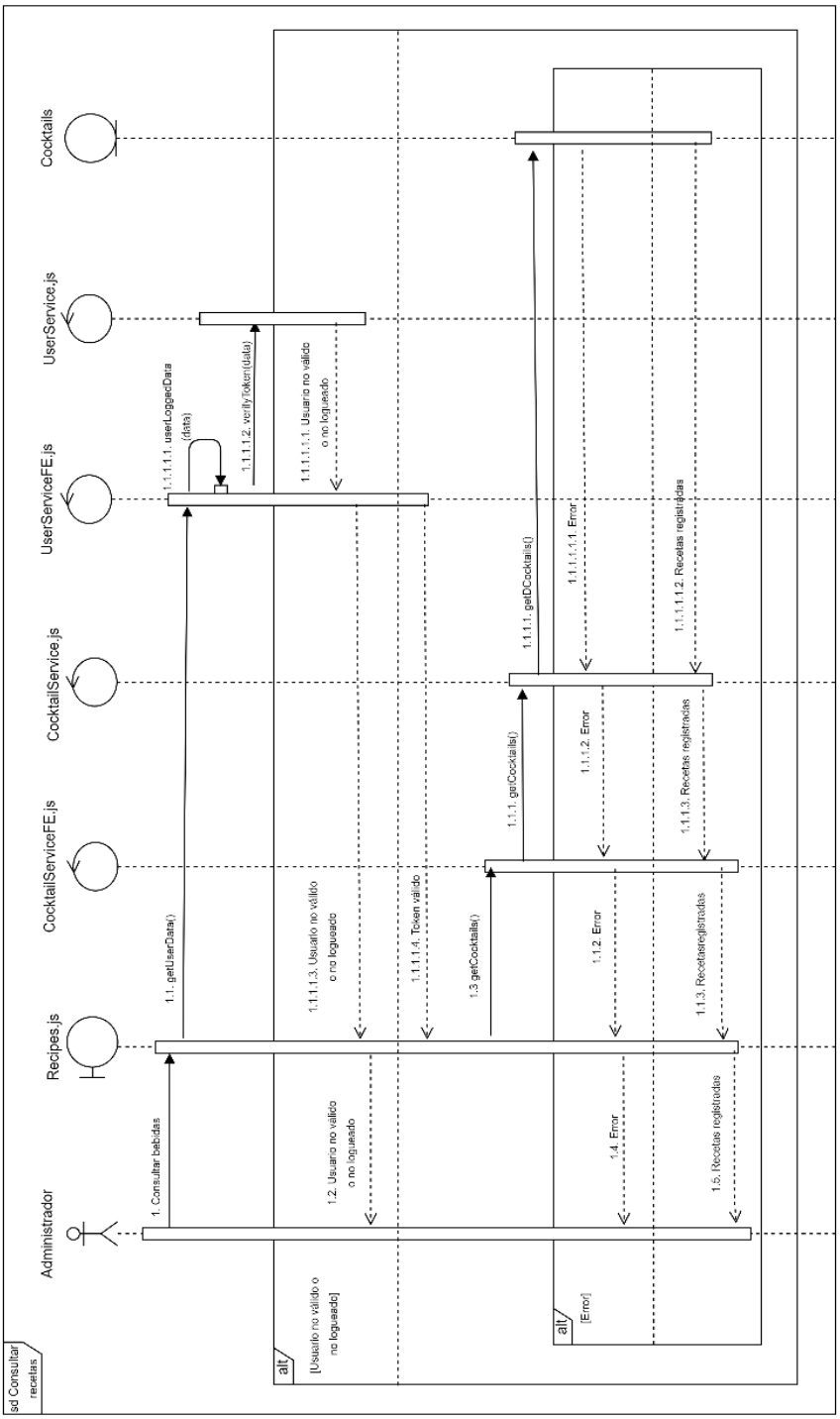


Ilustración 42. Diagrama de secuencia UC-030.

UC-031	**Buscar recetas**

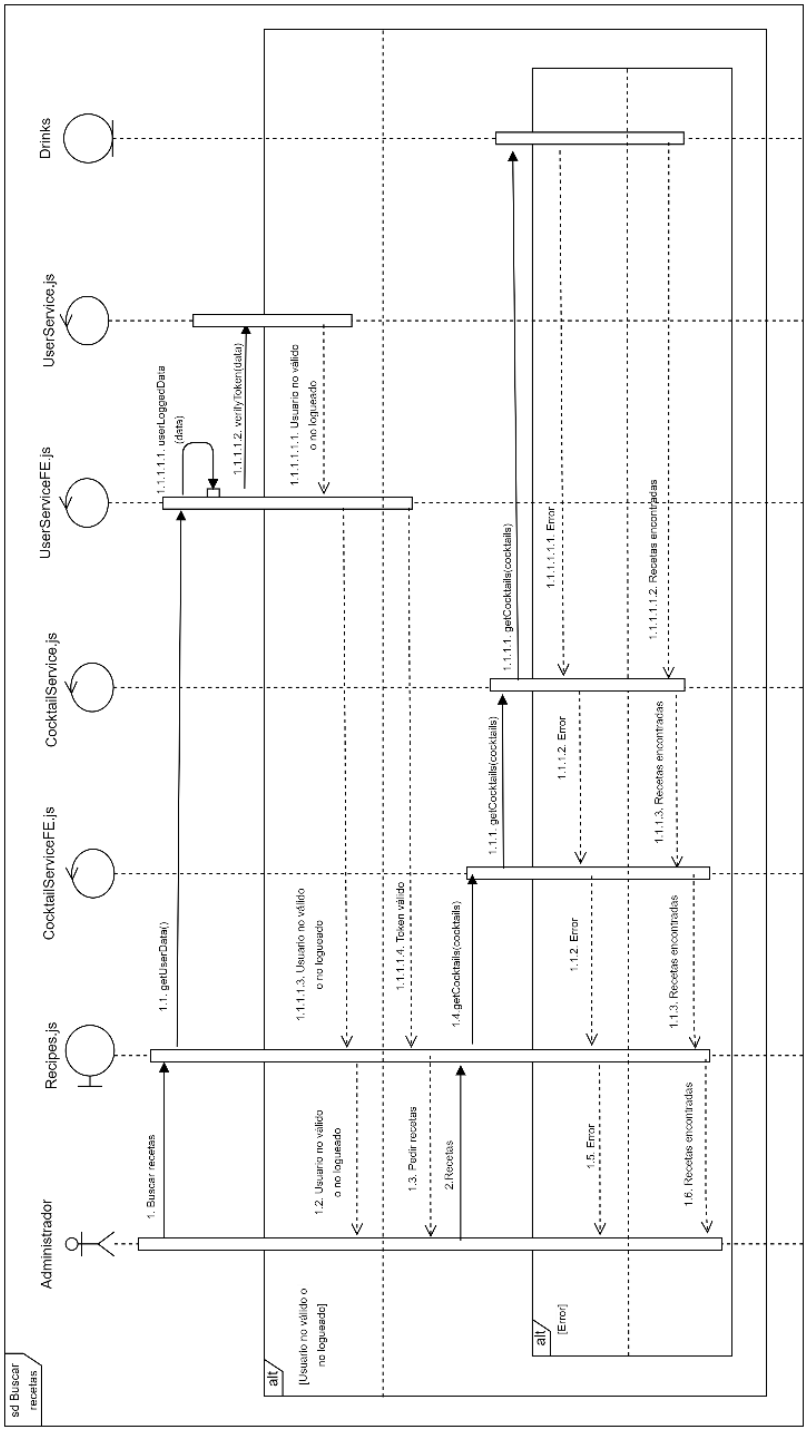
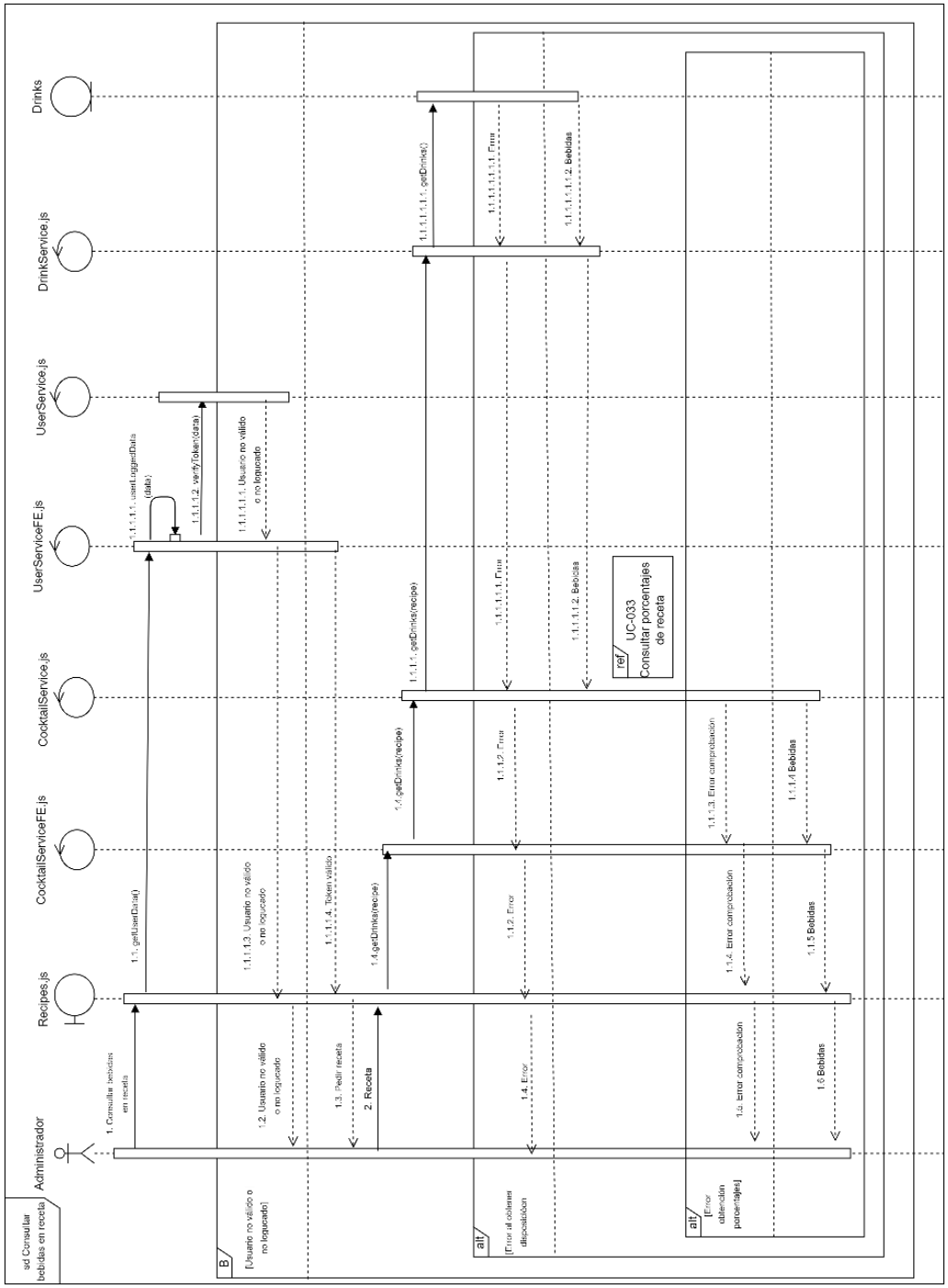


Ilustración 43. Diagrama de secuencia UC-031.

UC-032	**Consultar bebidas en receta**


sd Consultar bebidas en receta
Administrador
Recipes.js
CocktailServiceFE.js
CocktailService.js
UserServiceFE.js
UserService.js
DrinkService.js
Drinks
1. Consultar bebidas en receta
1.1. getUserData()
1.1.1.1.1. usertoggedData (data)
1.1.1.1.2. verify(token(data))
1.1.1.1.1. Usuario no válido o no logueado
1.1.1.1.3. Usuario no válido o no logueado
1.1.1.1.4. Token válido
1.1.1.1.1. getDrinks(recipe)
1.1.1.1.1.1. Error
1.1.1.1.1.1.1. Error
1.1.1.1.1.1.2. Bebidas
1.1.1.1.1.1.1.1. Error
1.1.1.1.1.1.1.2. bebidas
1.1.1.1.1.1.2. Error
1.1.1.1.1.1.2. Bebidas
1.1.1.1.1.1.3. Error comprobación
1.1.1.1.1.1.4. Error comprobación
1.1.1.1.1.1.5 Bebidas
1.1.1.1.1.1.6 Bebidas
1.1.1.1.1.1.6. Error comprobación
1.1.1.1.1.1.6 Bebidas
1.1.1.1.1.1.6 Bebidas
1.1.1.1.1.1.6 Bebidas

UC-033	**Consultar porcentajes en receta**

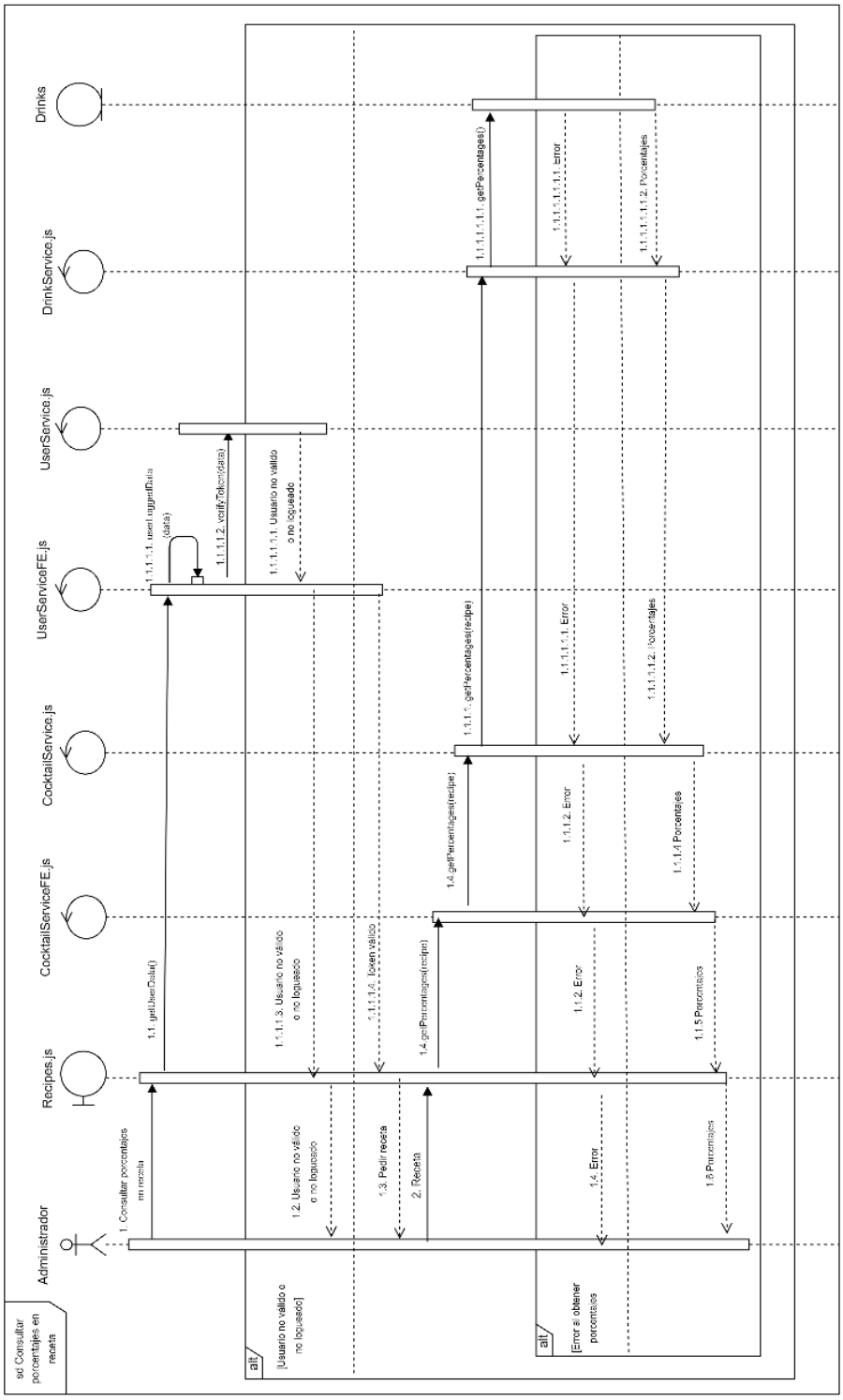


Ilustración 45. Diagrama de secuencia UC-033.

UC-034	**Preparar receta**

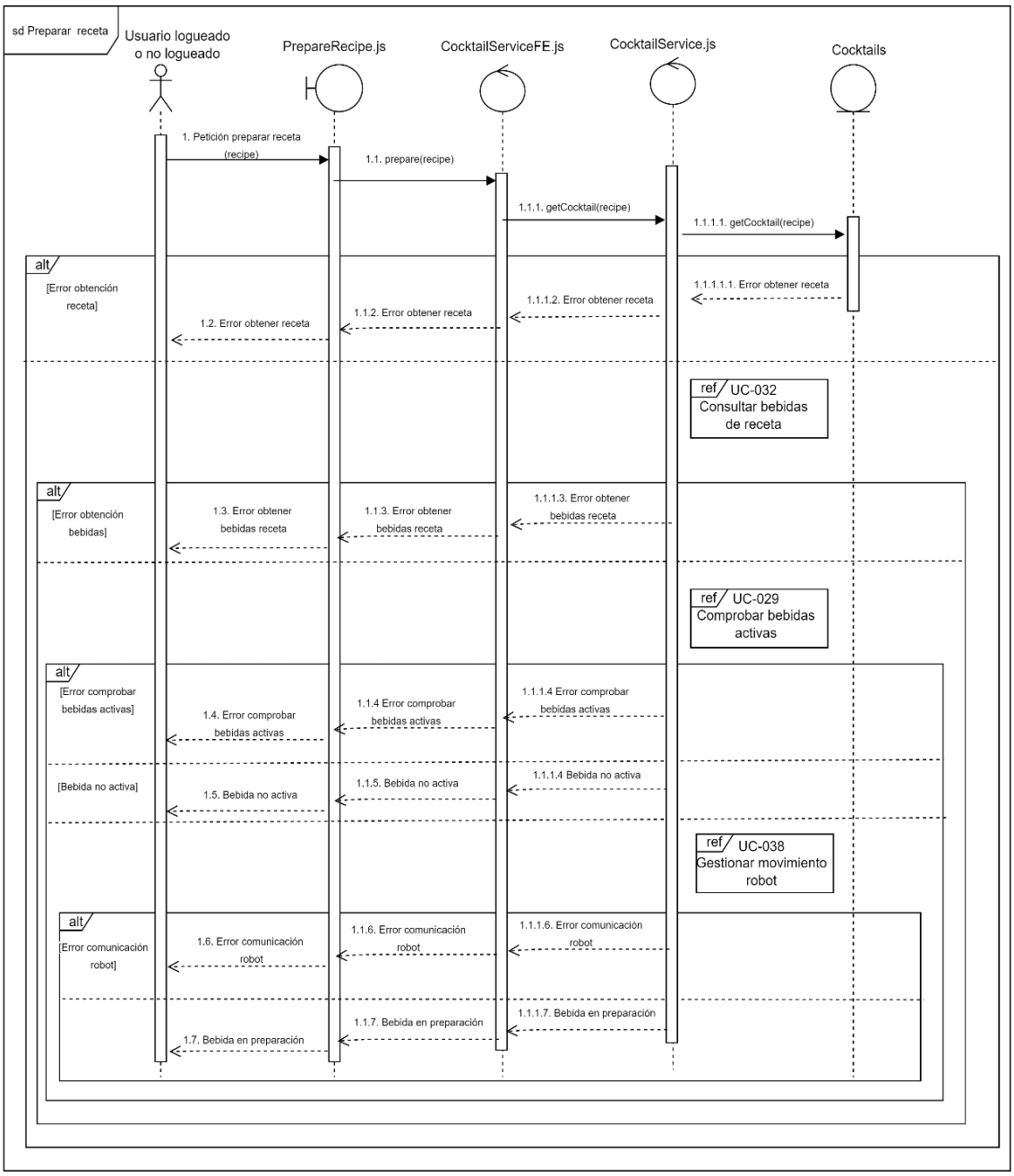


Ilustración 46. Diagrama de secuencia UC-034.

2.5.2 PAQUETE GESTIÓN DE ROBOT

UC-035	**Activar robot**

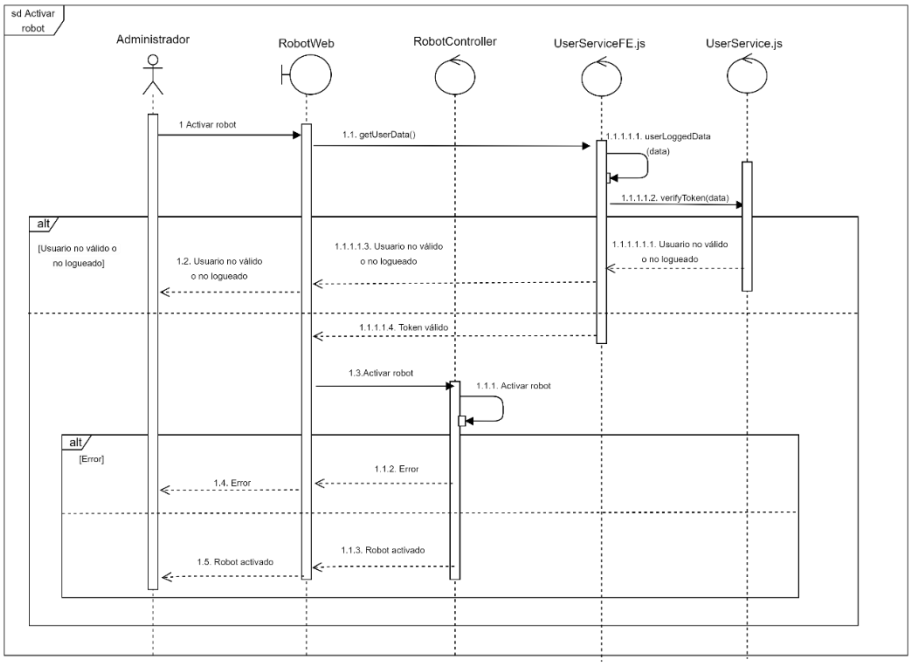


Ilustración 47. Diagrama de secuencia UC-035.

UC-036	**Parar robot**

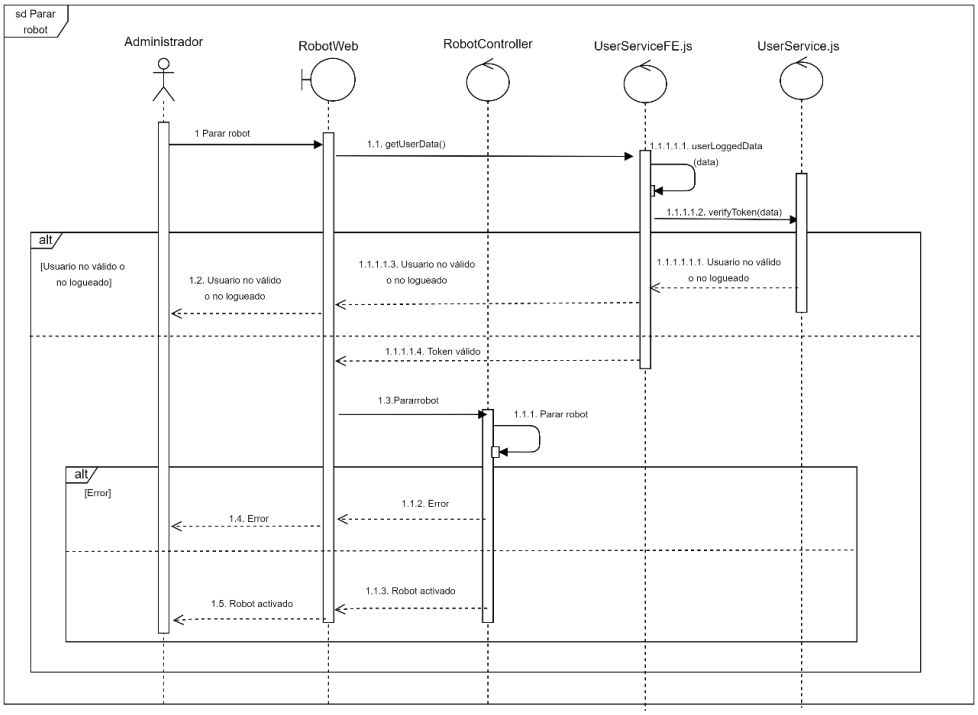


Ilustración 48. Diagrama de secuencia UC-036.

UC-037	**Gestionar estado robot**

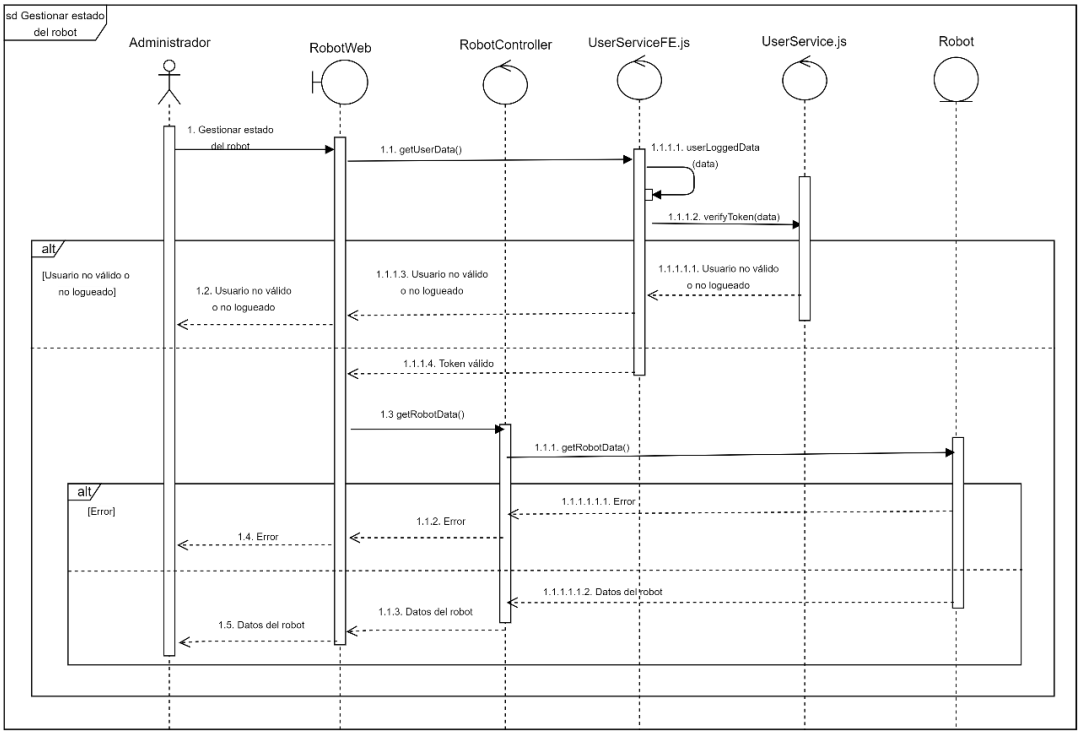


Ilustración 49. Diagrama de secuencia UC-037.

UC-038	**Gestionar movimiento robot**

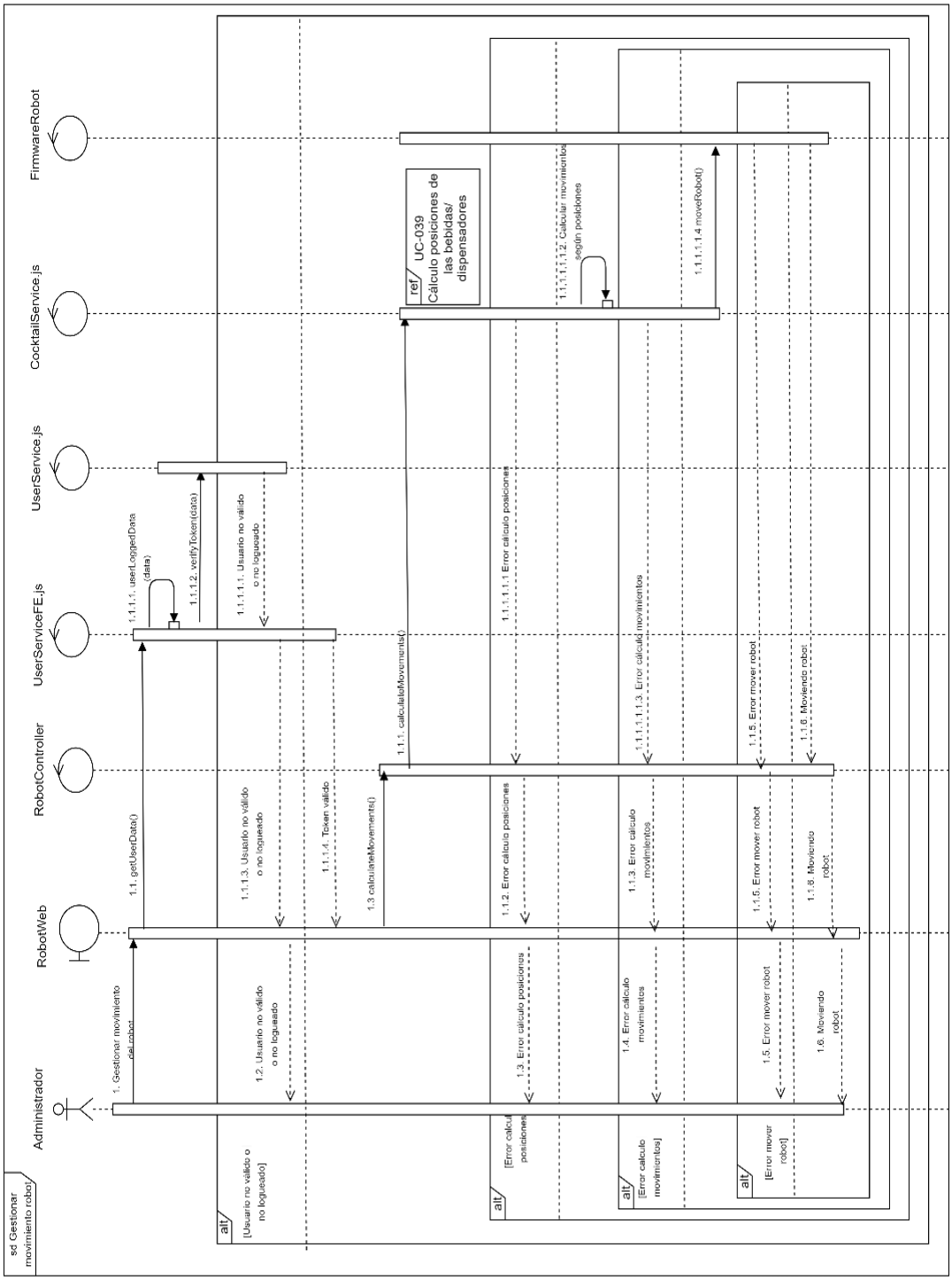


Ilustración 50. Diagrama de secuencia UC-038.

UC-039	**Cálculo posiciones de las bebidas/dispensadores**

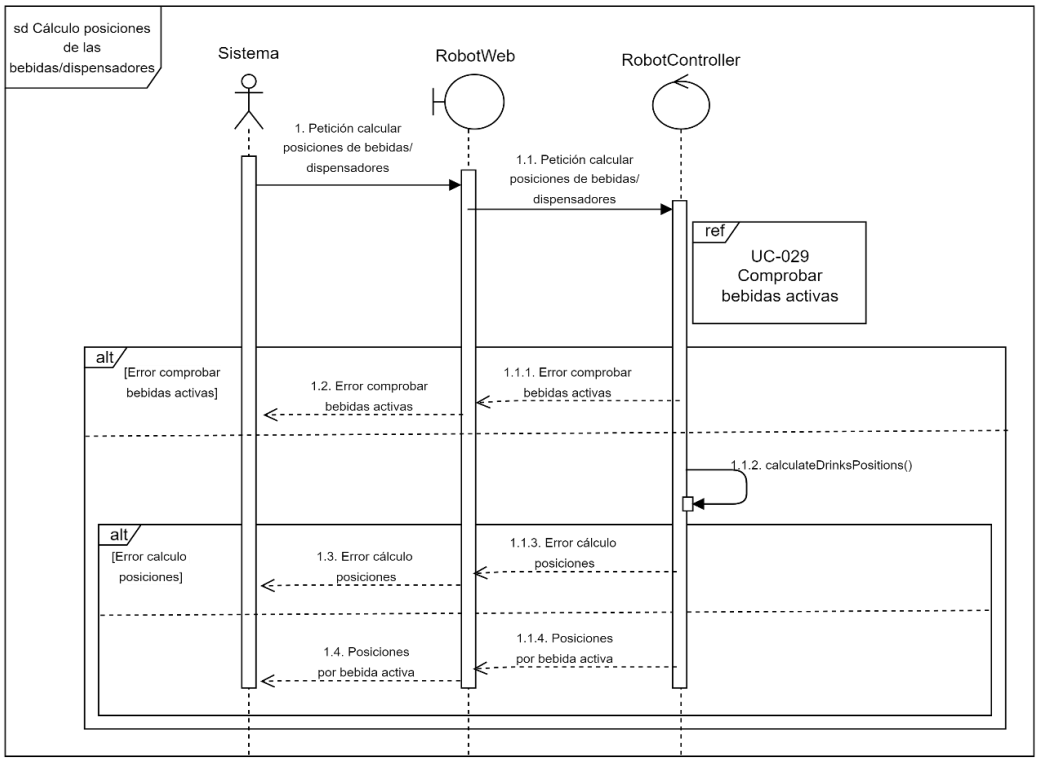


Ilustración 51. Diagrama de secuencia UC-039.

UC-040	**Gestión luces de aviso**

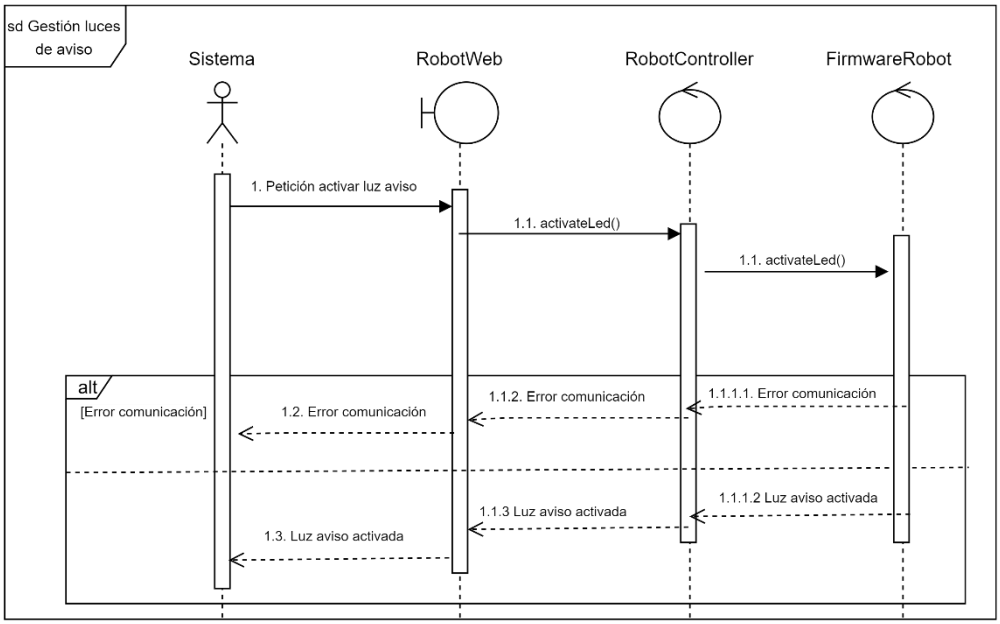


Ilustración 52. Diagrama de secuencia UC-040.

UC-041	**Control manual del robot**

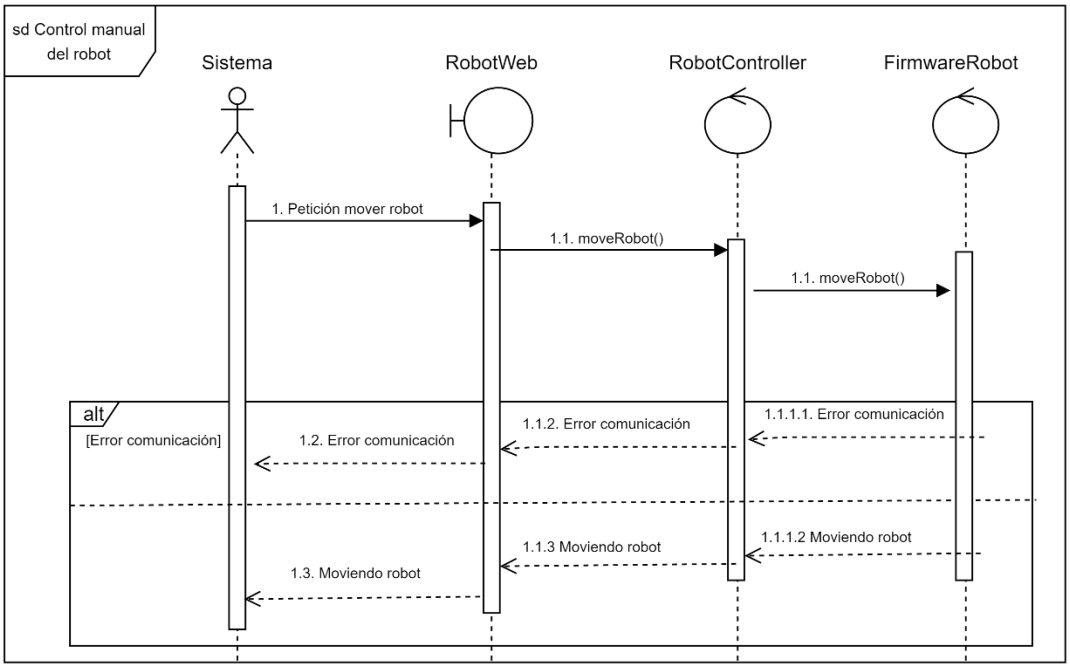


Ilustración 53. Diagrama de secuencia UC-041.

3 DISEÑO DE LA BASE DE DATOS

Para que el sistema pueda almacenar y gestionar correctamente la información, se utilizará una base de datos MySQL. En esta base de datos se guardarán diferentes tablas correspondientes a los datos de los usuarios, bebidas, recetas y otros de interés para el sistema.

Esta base de datos se describe según el siguiente modelo de entidad relación:

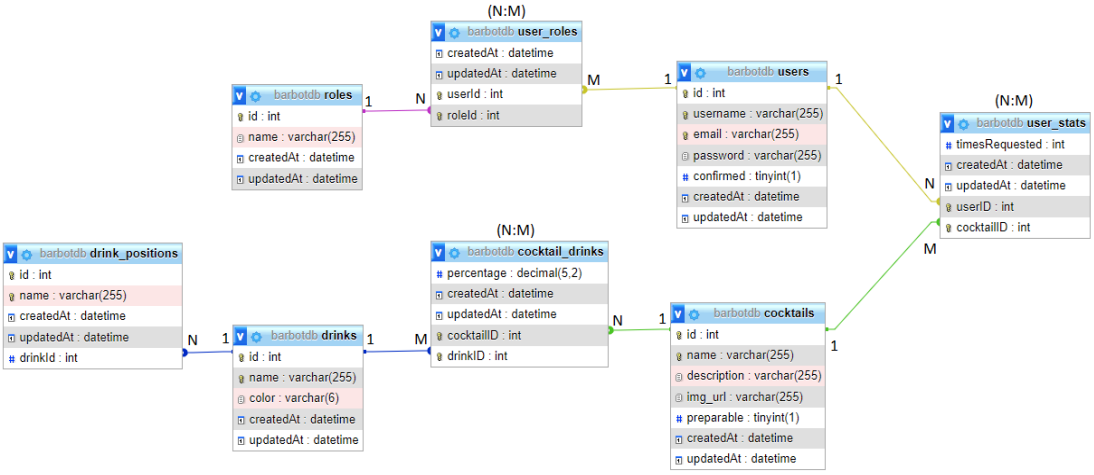


Ilustración 54. Diseño de la base de datos.

La base de datos dispondrá de las siguientes tablas principales:

- ***Roles:*** roles que puede tener un usuario.
- ***Users:*** información de los usuarios
- ***Cocktails:*** información sobre las recetas
- ***Drinks:*** información sobre las bebidas
- ***Drink_positions:*** información sobre las posiciones del robot y las bebidas registradas en ellas.

Estas tablas se relacionan a través de otras tablas (o directamente entre ellas) de la siguiente forma:

- ***User_roles:*** tabla intermedia que almacena los roles que tiene cada usuario. Sirve como unión para la relación N:M entre las tablas *Roles* y *Users.*
- ***User_stats:*** tabla intermedia que almacena las estadísticas que tiene cada usuario para cada receta registrada. Sirve como unión para la relación N:M entre las tablas *Cocktails* y *Users.*
- ***Cocktail_drinks:*** tabla intermedia que almacena las bebidas que componen cada receta registrada. Sirve como unión para la relación N:M entre las tablas *Cocktails* y *Drinks.*

La última relación es la relación N:1 entre las tablas *Drinks* y *Drink_positions.* Con esta relación se expresa el número de bebidas que puede haber en cada posición (que es 1) pero que esta bebida puede repetirse en varias posiciones.

4 MODELO DE DESPLIEGUE

El modelo de despliegue se representa utilizando para ello un diagrama de despliegue. Este diagrama muestra la arquitectura de ejecución de un sistema, incluyendo para ellos entornos de ejecución hardware, software y los middlewares que los conectan.

El diagrama de despliegue del sistema diseñado es el siguiente:

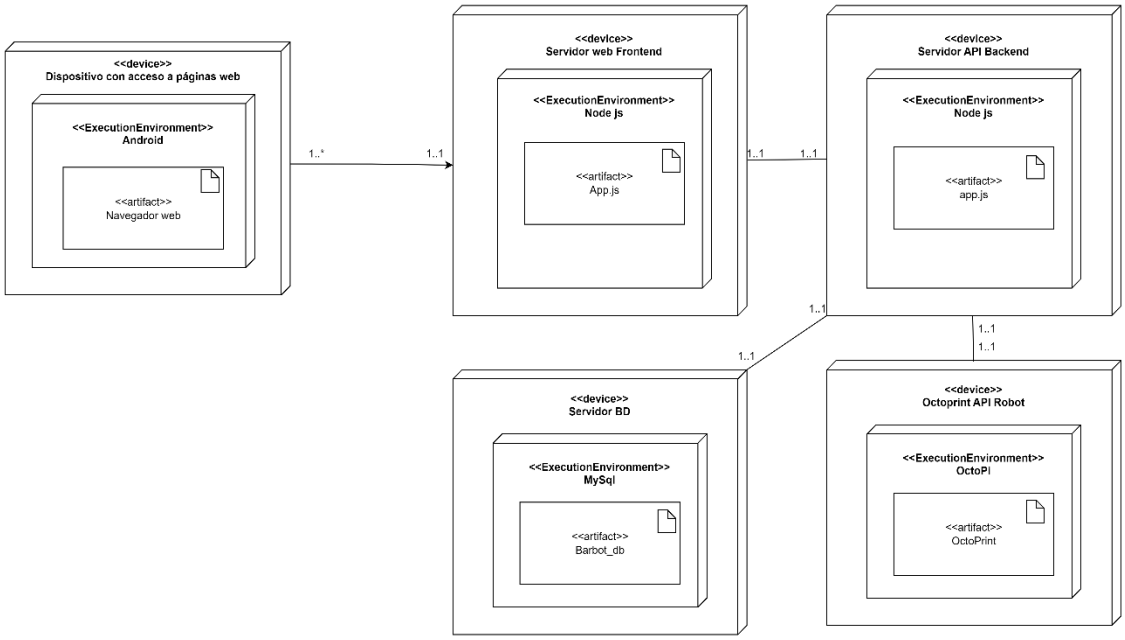


Ilustración 55. Diagrama de despliegue.

Como se puede observar en la figura, el sistema se compone de los siguientes nodos:

- ***Dispositivo con acceso a páginas web:*** dispositivo que dispone de un navegador web que permite acceder a páginas web. Con estos dispositivos los usuarios podrán acceder a la web del sistema y realizar distintas acciones.
- ***Servidor web Frontend:*** servidor que sirve de vista para el usuario. Proporciona la web a la que accederán los distintos usuarios. Se trata de un entorno Node.js.
- ***Servidor API Backend:*** servidor que proporciona los distintos servicios a los que accederán los clientes a través de la web. Se encargará de la parte logística del sistema proporcionando todos los servicios necesarios a través de una API, con lo que no solo se podrá trabajar a través del Frontend, sino que se pueden crear otras aplicaciones que trabajen junto con esta API. Se trata de un entorno Node.js.
- ***Servidor DB:*** servidor que se encargará de proporcionar acceso a la base de datos con la que trabajará el sistema y almacenando toda la información

necesaria para el correcto funcionamiento. Se trata de una base de datos MySQL.

- ***Octoprint API Robot:*** Este dispositivo llevará integrado el sistema operativo "*OctoPI*" desarrollado para el funcionamiento de impresoras 3D con motores paso a paso. Este sistema operativo proporciona una API con la que junto al firmware del robot se ejecutarán las órdenes de movimiento indicadas por el "*Servidor API Backend".*

5 REFERENCIAS

- Rumbaugh, J., Jacobson, I., Booch, G. “El Lenguaje Unificado de Modelado. Manual de Referencia”. 2ª ed., Addison-Wesley. 2007
- Pressman, R. S. “Ingeniería del Software: Un Enfoque Práctico”. 7ª Edición. McGraw-Hill. 2010.
- Documentación DrawIO (visitado el 30-05-2022)
    - https://drawio-app.com/tutorials/
- Documentación sobre patrones (visitados el 31-05-2022)
    - https://rlbisbe.net/2019/08/07/el-patron-bff-backend-for-frontend/
    - https://profile.es/blog/patrones-de-diseno-de-software/#Builder_Patterns
    - https://www.mongodb.com/docs/atlas/app-services/authentication/custom-jwt/