Documento de Descripción de la Arquitectura
Equipo D

Yessica Katherine Chautá Insuasty
ykchautai@unal.edu.co

Brayan David Cajicá Murcia
bdcajicam@unal.edu.co

Jesús Alejandro Noguera Ballén
janoguerab@unal.edu.co

Anderson Kavir Osorio Delgado
akosoriod@unal.edu.co

David Santiago Barrera Gonzalez
dsbarrerag@unal.edu.co

Jerson Steven Bustos Beltran
jsbustosb@unal.edu.co

17 de octubre de 2017

1. Nombre del sistema

SAM (Software Architecture Mail)

2. Alcance

El Sistema SAM puede ser usado por clientes a través del servicio web y móvil (android); Permite el registro y autenticación de usuarios dentro del sistema, haciendo uso de un usuario y una contraseña, tiene como funcionalidades el envío y almacenamiento de correos electrónicos, únicamente entre usuarios registrados en el sistema.
El envío de correos también puede ser programado para que sea realizado a una hora y fecha determinadas por el usuario que envía el correo. Los correos pueden contener un archivo adjunto, si el usuario así lo desea; éste puede ser en formato pdf, zip, y otros, estos formatos están sujetos a los tipos de archivos soportados por Rails, los cuales están especificados en la documentación de mime_types. El archivo tiene un tamaño límite máximo de $20MB$.
Los usuarios del sistema reciben notificaciones en sus clientes (web y/o móvil) al momento de recibir un correo. En cuanto al manejo de correos electrónicos, los usuarios pueden almacenarlos como borradores antes de enviarlos, así como también pueden diferenciarlos, dependiendo el proceso que realice, por enviados, recibidos (leídos y no leídos), borradores, y correos urgentes. Los usuarios pueden hacer uso del CC (Copia de Carbón) para enviar una copia del correo a otro usuario.

3. Interesados (Stakeholders) e Intereses (Concerns)

	Stakeholders	Concerns
1.	Equipo desarrollador (Equipo D)	Desarrollo, despliegue y ejecución completa del sistema según especificaciones dadas por el product owner a través de las historias de usuario y los requisitos técnicos establecidos.
2.	Usuarios del sistema	Análisis y evaluación del buen funcionamiento del sistema según sus especificaciones.
3.	Profesor (Product Owner)	Registro, autenticación, envío, programación, y recepción de correos dentro del sistema. Acceso al sistema por medio de un cliente web y/o móvil (android).

Tabla 1: Tabla de interesados e intereses.

4. Vistas arquitectónicas.

4.1. Vista de modelo de datos.

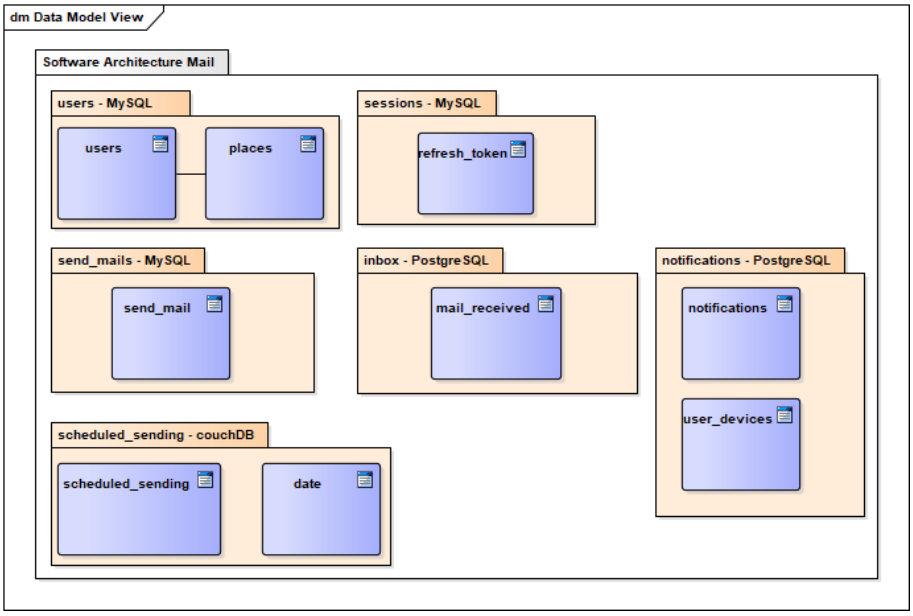

Figura 1: Vista - Modelo de base de datos.

La vista muestra las entidades que conforman el modelo de base de datos del proyecto, el cual esta separado en 6 bases de datos, una por micro-servicio.

- Elementos:
  - MS de registro: Almacena los datos de registro de los nuevos usuarios del sistema, las tablas tienen una relación 1 a 1.
    - Tabla users:
      - id (int)
      - first_name (string)
      - last_name (string)
      - username (string)
      - password (string)
      - birth_date (date)
      - gender (string)
      - mobile_phone (int)
      - current_email (string)
      - location (string)
    - Tabla places:
      - id (int)
      - name (string)
  - MS de autenticación:
    - Tabla refresh_token: Guarda la información de los refresh tokens activos.
      - id (int)
      - id_token (int)
      - username (string)
      - created_date_time (date)
  - MS de envió de correos: Almacena la información de los correos enviados a un usuario
    - Tabla sent_mail:

        - id (int)
        - sender (string)
        - recipient (string)
        - cc (string)
        - distribution_list (string)
        - subject (string)
        - message_body (text)
        - attachments (string)
        - send_date_time (date)
        - created_date_time (date)
        - draft (bool)
        - urgent (bool)
        - confirmation (bool)
- MS de recepción de correos: Almacena la información de los correos recibidos por un usuario.
    - Tabla mail_received:
        - id (int)
        - sender (string)
        - recipient (string)
        - cc (string)
        - distribution_list (string)
        - subject (string)
        - message_body (text)
        - attachments (string)
        - send_date_time (date)
        - created_date_time (date)
        - read (bool)
        - urgent (bool)
- MS de notificaciones: Se encarga de guardar la información necesaria de los correos recibidos y los dispositivos del usuario al cual se le va a hacer la notificación push.
    - Tabla notificacion:
        - id (int)
        - username (string)
        - sender (string)
    - user_devices:
        - id (int)
        - username (string)
        - device_id (string)
- MS de envió de correos programados: No relacional, almacena los datos de la fecha en la cual se quiere enviar cierto correo.
    - Colección schenduled_sending:
        - ids
        - date
    - Colección ids
        - user_id
        - mail_id
    - Colección date
        - year
        - month
        - day
        - hour
        - minutes

4.2. Vista de descomposición.

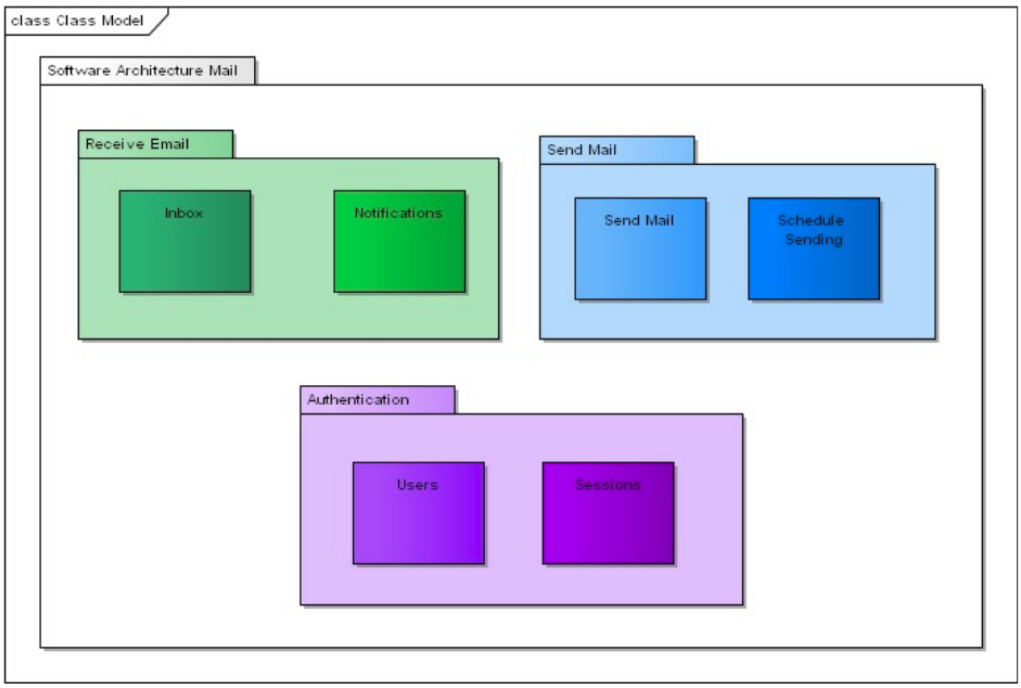

Figura 2: Vista - Descomposición.

El sistema se descompone en 3 sub módulos, estos son:

- Recibir Correos: Se encarga de almacenar los correos de la bandeja de entrada del usuario, de recibir los correos entrantes y de notificar al usuario cuando un correo ha llegado a su bandeja. Estas responsabilidades se separan en 2 microservicios:
  - Inbox o bandeja de entrada: recibe y almacena los correos.
  - Notificaciones: Se encarga de enviar notificaciones push a los dispositivos en los que el usuario tenga activa una sesión.
- Enviar Correos: Se encarga de crear y enviar correos, también de almacenar los borradores y hacer envíos programados. Se divide en 2 micro servicios:
  - Envío de correos: Se encarga de almacenar borradores y enviar correos.
  - Programación de envíos: Se encarga de almacenar la información acerca de cuándo se debe enviar un correo, y accionar el envío del mismo

  Autenticación: Se encarga de almacenar información sobre el usuario, autenticarlo y generar sesiones. Se divide en 2 micro servicios:
  - Usuarios: Se encarga de almacenar la información básica del usuario y autenticar al mismo con usuario y contraseña.
  - Sesiones: Se encarga de firmar los jwt, validarlos y refrescarlos.

4.3. Vista de capas.

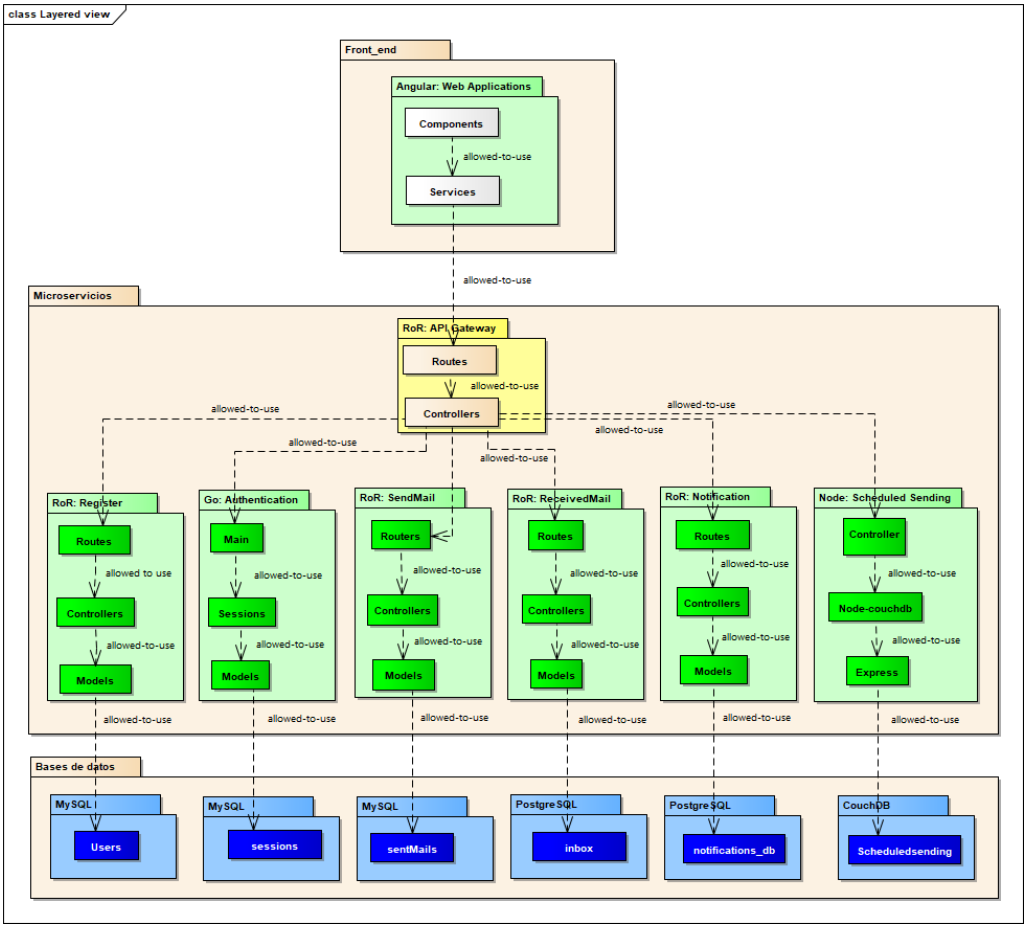

Figura 3: Vista - Capas.

El sistema se compone de tres capas, estas son:

- Capa de front end.
  - Angular web application.
- Capa de microservicios.
  - API Gateway
  - MS Register
  - MS Authentication
  - MS SendMail
  - MS RecivedMail
  - MS Notification
  - MS Scheduled Sending
- Capa de datos.
  - BD users
  - BD sessions
  - BD sent mails
  - BD inbox
  - BD notifications
  - BD scheduled sending

4.4. Vista de componentes y conectores.

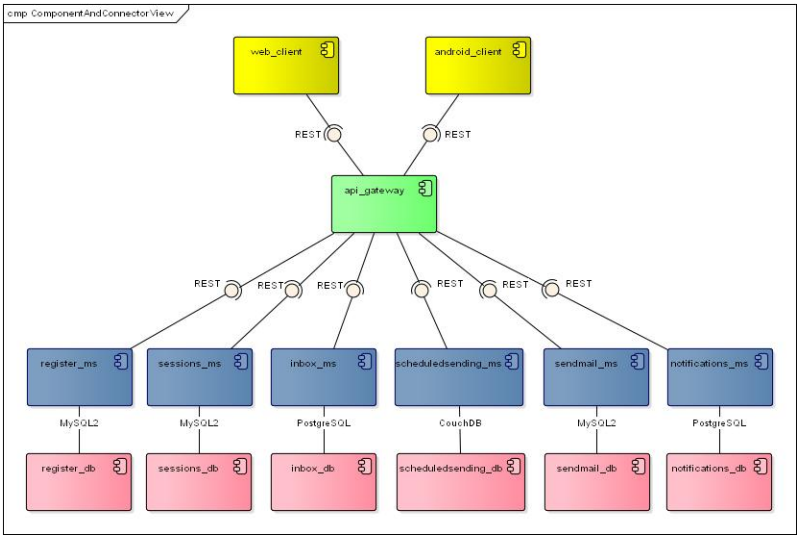

Figura 4: Vista - Componentes y conectores.

- **Componentes**
  - Api Gateway: el api gateway se encarga de recibir los datos desde el usuario y procesarlos a través de cada uno de los microservicios existentes. Dentro de las funciones que realiza, se tienen:
    - Registro y autenticación del usuario: recibe los datos básicos del registro, y los almacena en la base de datos de Usuario, y para la autenticación valida el usuario y la contraseña que se alojen en la base de datos de usuario y una vez válido, genera el token desde el microservicio de sesiones para poder seguir con el proceso.
    - Creación y envío de correos (Borradores y enviados): realiza el registro de los correos electrónicos en la base de datos de correos enviados o por enviar.
    - Recepción de correos y clasificación en etiquetas(Urgente, recibido): una vez enviado un correo, se debe clasificar de acuerdo a los parámetros que se reciban y almacenarse en la base de datos del microservicio de recepción de correos, de tal forma que el usuario que los reciba pueda verlos desde el api gateway como consulta.
    - Programación del correo borrador que se desea enviar: haciendo uso del microservicio de programación y de creación de borradores, se realiza el envío del correo en la hora programada y se registra en correos enviados por el usuario. Una vez realizado el proceso, el usuario que recibe también puede consultar su correo desde el api.
    - Notificación hacia los dispositivos registrados: cuando un correo es recibido se envía la notificación al dispositivo registrado en la base de datos de las notificaciones y se revisan las bandejas de entrada de correo una vez sea enviada la notificación.
    - Cliente Web: Se encarga de dar la vista desde navegadores web hacia el usuario final. Cuenta con las vistas de todas las funcionalidades del sistema y se conecta con el api gateway a través de peticiones REST para mostrar al usuario los datos almacenados y permitir crear nuevos datos dentro de los microservicios. Es el componente de frontend web de la aplicación.
    - Cliente Android: Se encarga de dar la vista desde dispositivos con sistema operativo Android hacia el usuario final. Al igual que el componente web, cuenta con las vistas de todas las funcionalidades del sistema y se conecta con el api gateway a través de peticiones REST para mostrar al usuario los datos almacenados y permitir crear nuevos datos dentro de los microservicios. Es el componente de frontend móvil de la aplicación.
  - Microservicios

    - Register_ms: se encarga del registro de los usuarios de la plataforma con los datos básicos, valida el usuario en la contraseña a través del login.
    - Sessions_ms: se encarga de la autenticación del usuario y la generación de un token de validación del mismo para acceder a los demás microservicios cuando lo requiera.
    - Inbox_ms: se encarga de mostrar las bandejas de entrada de cada uno de los correos electrónicos que tenga el usuario en base de datos y clasificarlos por categorías.
    - Scheduled_sending_ms: tiene la función de programar correos y enviarlos a una hora determinada, la cual es definida por el usuario.
    - Sendmail_ms: se encarga del envío de correos electrónicos entre los usuarios de la plataforma.
    - Notifications_ms: envía las notificaciones de nuevos mensajes cuando se tenga un nuevo registro.
  - Bases de datos correspondientes
  - Register_db: almacena los datos básicos del usuario, su nombre de usuario y contraseña.
  - Sessions_db: almacena los token asignados para las sesiones de cada usuario.
  - Inbox_db: almacena los correos recibidos por el usuario y su contenido.
  - Scheduling_sending_db: almacena los registros de programación de correos del usuario.
  - Sendmail_db: almacena los correos que son enviados por el usuario y su contenido y las fechas de envío.
  - Notificatios_db: Contiene la lista de dispositivos del usuario y el registro de las notificaciones.

- **Conectores**

- Desde el api gateway hacia los microservicios: se mantiene un funcionamiento a través del protocolo REST entre el api y los microservicios creados.

- Desde los microservicios a las bases de datos: para las conexiones entre los microservicios y las bases de datos se tienen determinados los siguientes servicios de bases de datos:

  - Entre los ms de registro, sesiones y envío de correos se usan bases de datos relacionales en MySQL.
  - Para los ms de notificaciones y bandeja de correos se tienen bases de datos relacionales en posgreSQL.
  - Para el ms de programación de correos se tiene una base no relacional en CouchDB.

4.5. Vista de despliegue.

- Maquina rancher_server con IP 192.168.99.100.
  - Contenedor #1.
    - Entorno de ejecución en rancher puerto 8080.
      - El cual despliega rancher server.
- Maquina rancher_node2 con IP 192.168.99.101.
  - Contenedor #1.
    - Entorno de ejecución en Ruby on Rails puerto 4000.
      - El cual despliega sam_api_gateway.
  - Contenedor #2.
    - Entorno de ejecución en Ruby on Rails puerto 3004.
      - El cual despliega sam_register_ms.
  - Contenedor #3.
    - Entorno de ejecución en MySQL puerto 8812.
      - El cual despliega sam_register_db.
  - Contenedor #4.
    - Entorno de ejecución en GO puerto 3005.
      - El cual despliega sam_sessions_ms.
  - Contenedor #5.
    - Entorno de ejecución en MySQL puerto 8814.
      - El cual despliega sam_sessions_db.
  - Contenedor #6.
    - Entorno de ejecución en Ruby on Rails puerto 3002.
      - El cual despliega sam_inbox_ms.
  - Contenedor #7.
    - Entorno de ejecución en PostgreSQL puerto 8808.
      - El cual despliega sam_inbox_db.
  - Contenedor #8.
    - Entorno de ejecución en Ruby on Rails puerto 3001.
      - El cual despliega sam_sendmail_ms.
  - Contenedor #9.
    - Entorno de ejecución en MySQL puerto 8806.
      - El cual despliega sam_rsendmail_db.
  - Contenedor #10.
    - Entorno de ejecución en NodeJS puerto 3006.
      - El cual despliega sam_scheduledsending_ms.
  - Contenedor #11.
    - Entorno de ejecución en CouchDB puerto 5984.
      - El cual despliega sam_scheduledsending_db.
  - Contenedor #12.
    - Entorno de ejecución en Ruby on Rails puerto 3003.
      - El cual despliega sam_notification_ms.
  - Contenedor #13.
    - Entorno de ejecución en PostgreSQL puerto 8810.
      - El cual despliega sam_notifications_db.

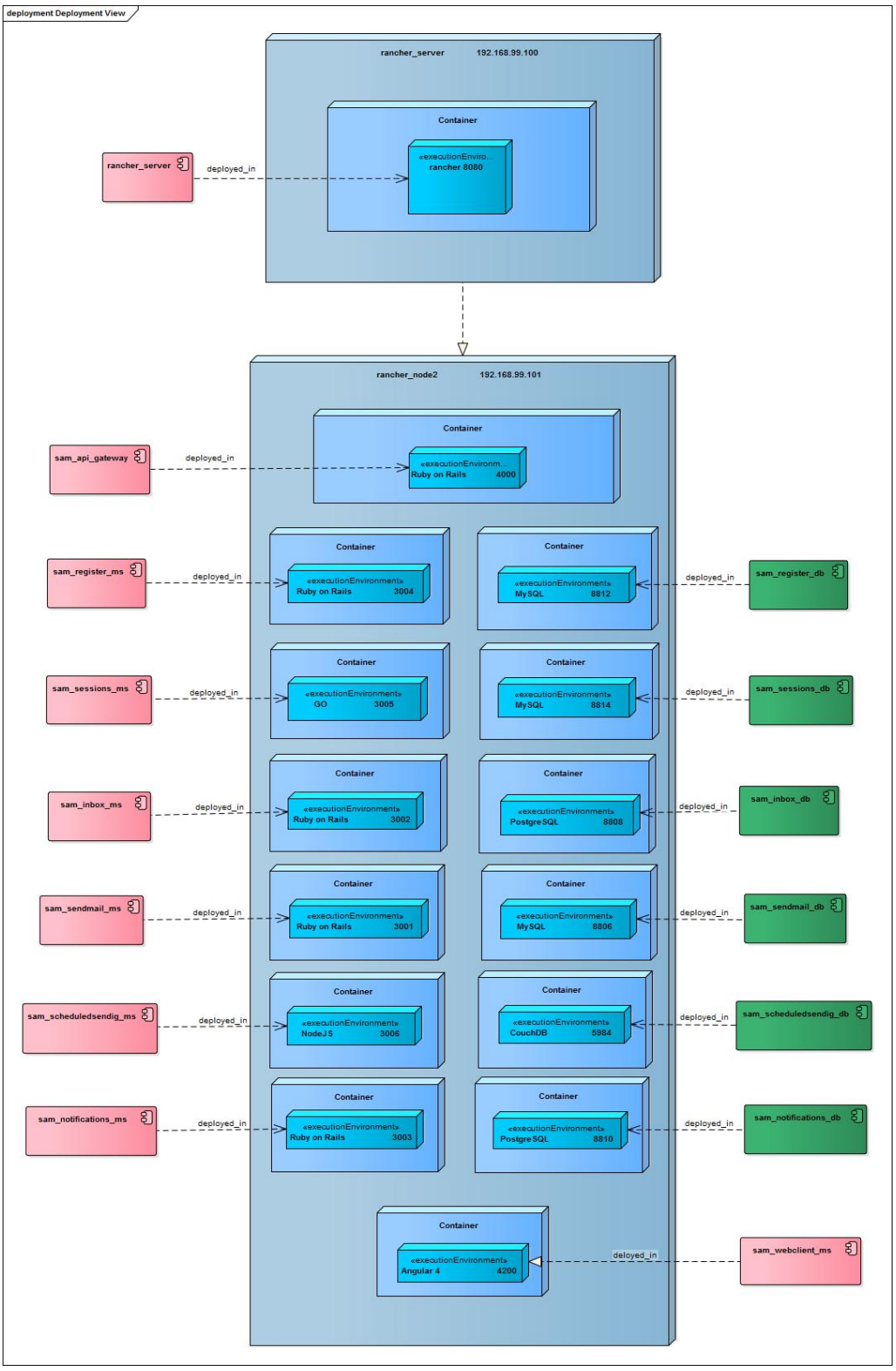

Figura 5: Vista - Despliegue