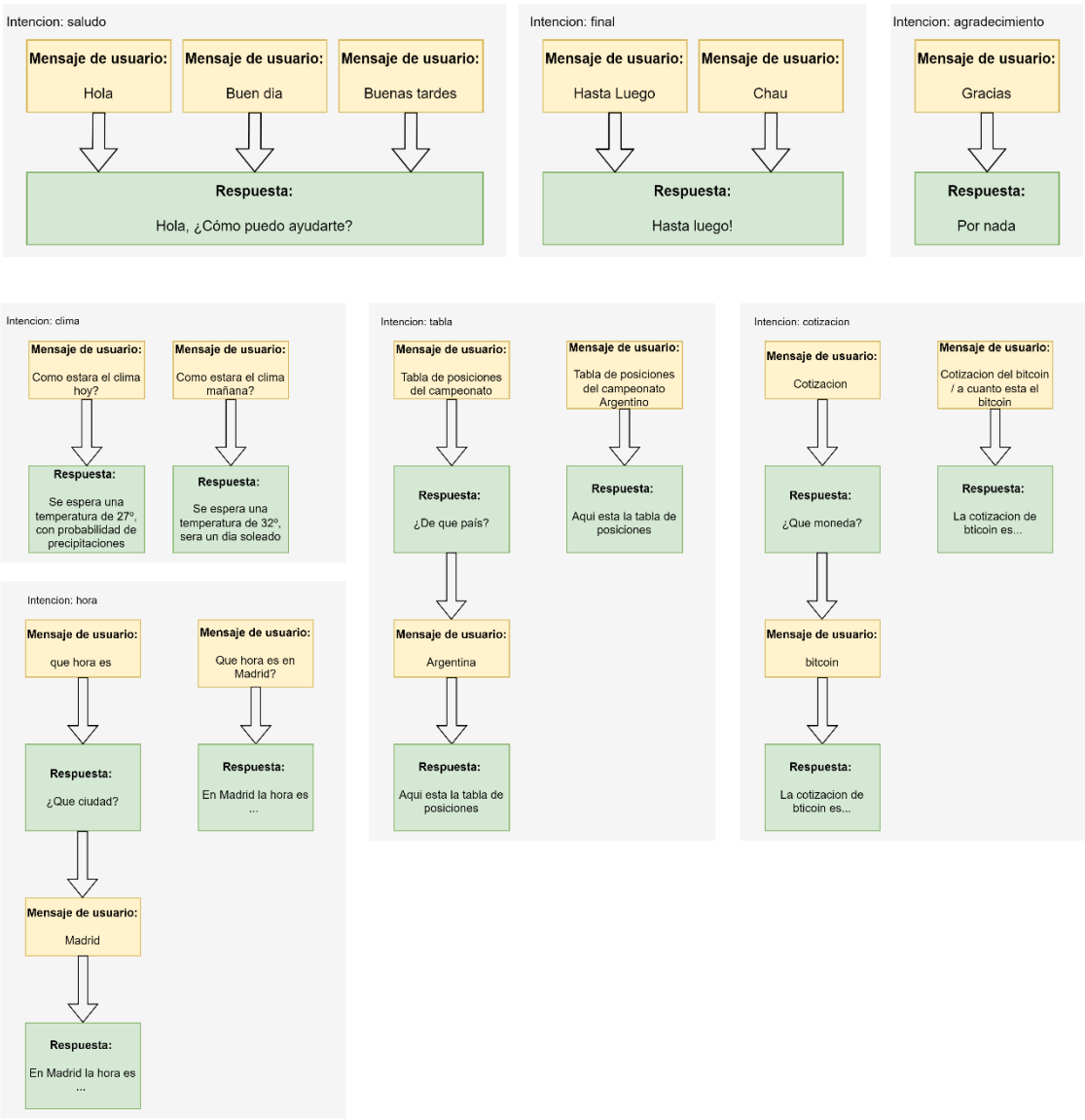
Propuesta de solución prueba técnica

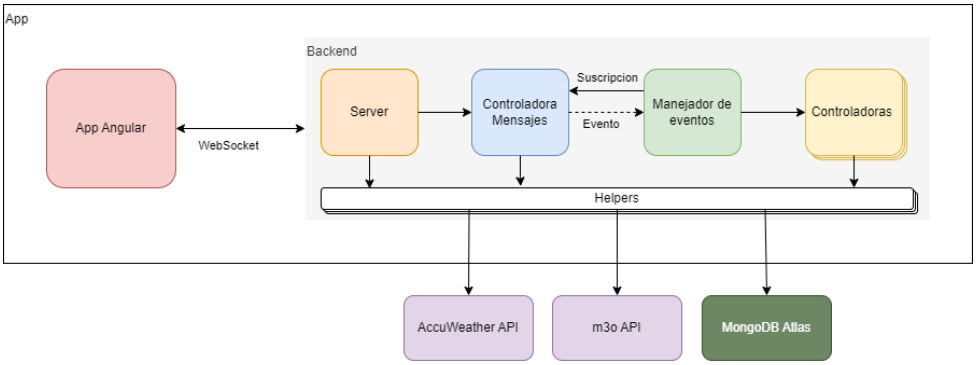
La solución propuesta se compone de dos elementos, por un lado el front end desarrollado con Angular y por otro lado el backend desarrollado con NodeJS y Express.

El primer paso fue ver el funcionamiento de un bot en general y pensar en los flujos de cara al usuario, ver las intenciones de este y diferenciar las entidades.

Al pensar en los flujos vemos los básicos para comenzar con los saludos, y luego las intenciones que van a la lógica: clima, tabla, cotización y hora.



Una vez hecho esto, pensé en la arquitectura, y se diseño la siguiente arquitectura en capas



El backend se compone de distintas capas:

- **Server:** Es la capa donde se aloja el servidor web, recibiendo las requests y encargado del websocket.
- **Controladora Mensajes:** Recibe los mensajes que el server recibe por el websocket, llama al helper de intenciones para encontrar la intención del mensaje y luego emite el evento correspondiente a la intención del mensaje recibido.
- **Manejador de eventos:** Se suscribe a los eventos de intenciones de los mensajes recibidos, y continua el flujo por cada evento, pasando el control a la controladora correspondiente.
- **Controladoras:** Reciben el mensaje, llaman al helper de entidades para obtener la entidad presente en el mensaje y ejecutar la acción correspondiente a la intención y entidad. También utilizan otros helpers de a cuerdo con la acciona realizar.
    - **Clima:** Obtiene la entidad “localidad” del mensaje y llama al helper de Accuweather para obtener los datos del clima y reponder al cliente.
    - **Cotizacion:** Obtiene la entidad “moneda” del mensaje y llama al helper de m3o para obtener la cotización de la cripto solicitada (bitcoin o cardano).
    - **Horarios:** Obtiene la entidad “localidad” del mensaje y llama al helper de m3o para obtener la fecha, hora y huso horario de la localidad encontrada.
    - **Tabla:** Obtiene la entidad “país” del mensaje y realiza una consulta a una mongoDB para obtener la tabla de posiciones del país encontrado.
    - **Agradecimiento:** Responde un mensaje predefinido al recibir una intención de agradecimiento.
    - **Saludo:** Realiza el saludo final al recibir una despedida en el mensaje y finaliza la conexión del websocket.
- **Helpers:** Son clases de soporte a la lógica principal:
    - **Sesión:** Es un manejador de la sesión, contiene un Map de las sesiones activas, con sus respectivos metadatos y se encarga de cualquier comunicación con cada cliente.
    - **Intenciones y Entidades:** Estos dos helpers simulan una capa de IA, obeniendo las intenciones y las entidades de los mensajes recibidos. Tienen dos arreglos de palabras que conforman el “modelo entrenado”. Se realiza de esta forma a modo de simulación para interiorizarme en el tema y ver la lógica detrás de esto.
    - **AccuWeather:** Se encarga de la comunicación con la API de AcuWeather.

- **M3O:** Se encarga de la comunicación con la API de m3o.
- **tabalPosiciones:** Se encarga de realizar la conexión a mongoDB y obtener la tabla de posiciones.

Respecto a la implementación, la realice en Azure, ambos componentes están dockerizados, se levantó una VM con Docker y ambos se alojan en ella.

La idea principal seria que el backend persista corriendo en AppService , ContainerIntances, o en Kubernetes y el front al ser una web estática en Static Web Apps, pero al alojarlo en estos, se vieron problemas al momento de establecer el websocket ya que al ser un proyecto de demostración, el backend no tiene certificados SSL y Static Web Apps en su tier gratuito no permite usar HTTP, fuerza el uso de HTTPS por lo que el navegador no se conecta a un websoket sin SSL, y por parte del backend de acuerdo a la documentación de Microsoft, en los tiers gratuitos o de máquinas de Linux para App Service y demás, tienen su websocket habilitado por defecto o no permiten desactivarlo, lo que no permite con el websocket del bakend en NodeJS.

Repo del backend: https://github.com/ignaciocolussi/bot

Repo del frontend: https://github.com/ignaciocolussi/bot_front_app