Architecture Overview

2.**Core RMM Server/Agent**

•A server and agents will be written in GO using the gorilla/mux framework.
•Agents will collect system information to register with the server as well as fulfill any needs not already provided by an integrated tool such as remote command and script execution, OS level data collection (version, specific hardware data, etc.), and more.
•Server will also house the web server for the Custom UI documented below which will serve as the users "entry point" to the platform.

3.**Microservices Architecture**

•Each tool (Netbox, UVDesk, CheckMK Raw, etc.) will be encapsulated into its own microservice.
•Each microservice will expose an API for communication with the server.

4.**API Gateway**

•Implement an API Gateway as a separate service.
•The API Gateway will route incoming requests to the appropriate microservice and aggregate responses.
•The API Gateway will also be written in Go using the gorilla/mux framework.

5.**Custom UI**

•Implement a Custom UI as a separate service.
•The UI will communicate with the backend services via the API Gateway.
•The UI can be built with a modern JavaScript framework like React, Vue, or Angular.

6.**User Management and Authentication Service**

•Implement a User Management and Authentication Service.
•This service will handle user registration, authentication, and role-based access control.
•It will integrate with all other services to ensure only authorized users can access certain functionalities.
•This service will also be written in Go using the gorilla/mux framework.

7.**Data Persistence Layer**

•Will use PostgreSQL database for storing agent registration data, user information, and additional "static" data.
•If dynamic data such as host metrics and other time-based data cannot be handled through microservice API's, will implement InfluxDB database to supplement where needed.

8.**Containerization and Orchestration**

•Dockerize each service for consistent deployment and scalability.
•Use Docker Compose for local development and testing.