Building enterprise grade technology applications is hard. Before the first line of code is written, there are a myriad of decisions that need to be made concerning the infrastructure of the application. What operating systems do we use? What application web server do we use? What languages to we use? How do we connect to our existing databases? **Where do we even start?**

KRM'S LEAF

LEAF is derived from a VA-funded project, the Electronic Health Management Platform (eHMP) meant to replace CPRS with a modern, web-based interface that would integrate the patient record across VistA sites to allow clinicians to have a full, seamless view of patient care across the enterprise. With the advent of the Cerner transition, this project was shelved by the VA. KRM was integral in the development of eHMP, and acquired the source code under open source, and has continued the development of the framework under KRM's Light Enterprise Application Framework (LEAF) branding.

KRM started the LEAF migration more than 7 years ago. KRM's goal with LEAF was to build upon VA's investment and further develop it as a robust application platform which is not confined to healthcare applications. KRM has invested significant resources and time to develop LEAF as a flexible, lightweight platform for application modernization projects, including healthcare EHRs as well as other applications. Specific activities KRM conducted to develop LEAF include:

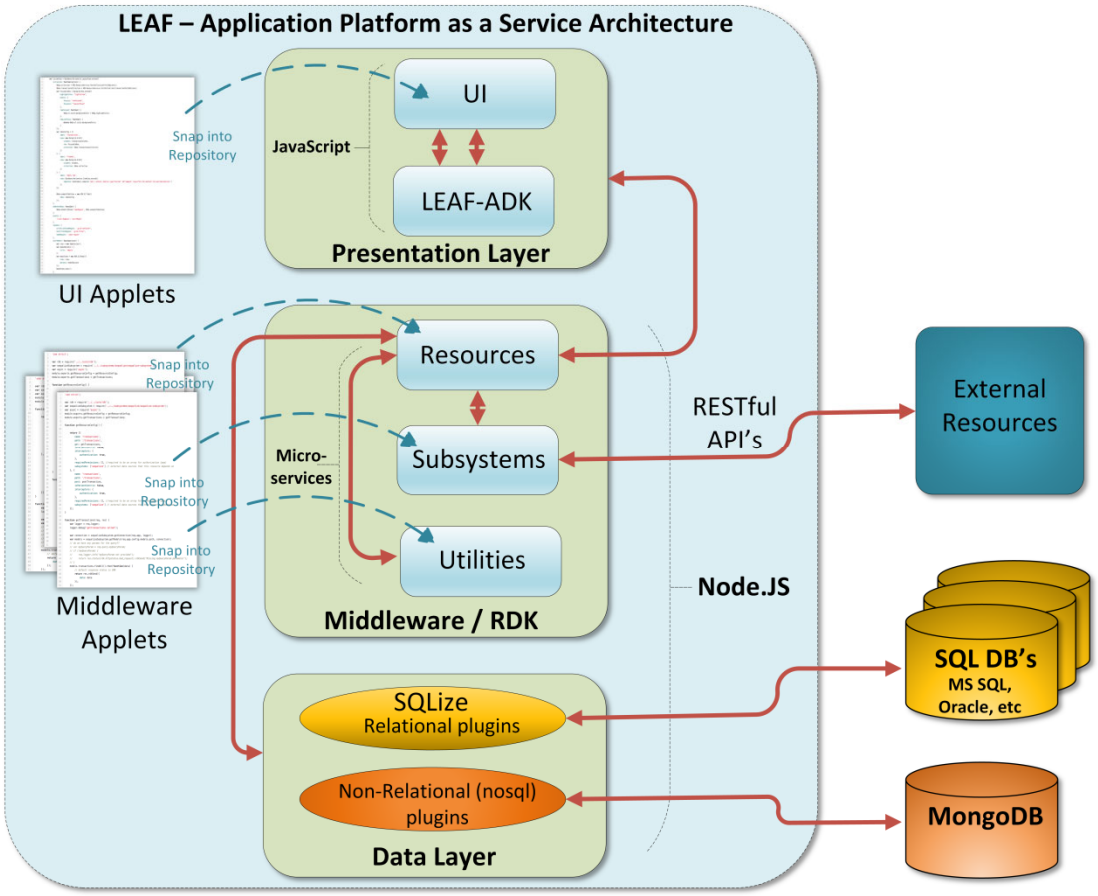
- Forking the original code to LEAF and resolving issues caused by VA redaction
- Rebuilding the UI in a modern framework (React)
- Rebuilding the supporting infrastructure to use containers (Docker) and targeting deployment for the cloud (e.g., AWS/VA EC)
- Rebuilding healthcare-specific applets and functions
- Removing VA specific code, replacing VistA specific Data Access Layer with a generic database layer
- Reorganizing the code, enforcing the SDK paradigm, creating starter projects to allow for quick project initialization and efficient 3rd party development

The LEAF platform is written using modern software tools and methodologies:

- Offers a full-stack software development kit – front-end (applet development kit/ADK) and back-end (resource development kit/RDK)
- Includes extensible interactive API documentation built on OpenAPI standards and example interactive UI component examples
- Modern graphing libraries for beautiful and interactive data display and several graph types
- User- and pre-defined composable workspaces combined with fine-grained administrative privileges allow for complex customizability of different data displays to accommodate any user role
- Data filtering including date-range to focus all displayed data on a specific user-configured time period
- Written in HTML5/CSS3, Node.js, JavaScript and modern user interface frameworks like React and Redux
- Backend is built in Node.JS - which is an open-source, cross-platform JavaScript runtime environment used in such web sites as PayPal, Netflix, and Uber
- Containerized with Docker to run anywhere
- Uses modern testing tools and boasts high unit test code coverage
- Uses RESTful web services and APIs for data interchange – even inside the SDK
- Built for the cloud -- take advantage of scaling, cloud-native databases, automated deployments, and more
- Extensible database layer allows connections to virtually any relational DB or NoSQL DB (includes connectors to PostgresDB, MongoDB, and Google GCP Firestore out of the box)

The LEAF Platform can be logically divided into three parts: the presentation layer, business layer and data access layer. A simplified diagram representing the current architecture is shown on the next page.

To learn more, contact us today:
304-876-6600 or info@krminc.com



Applets are developed in JavaScript and, once registered with the platform, can be customized and arranged by users to their liking.

The business layer or Resource Development Kit (RDK) is responsible for moving data and information to and from the UI to the data layer or other externally connected system. The RDK is comprised of server side "resources", which are reusable components that perform data reading and writing functions as well as communications to external systems. These resources allow LEAF to deliver a "service-oriented architecture" (SOA) and provide standardized access methods to common databases or external systems. The RDK is also responsible for providing central logging, metrics, and error handling.

Together, the RDK and UI make up the LEAF SDK allowing the development of custom applets and the associated data-reading and writing resources to provide information to the applet. All development work or enhancements to LEAF are performed in accordance with the standards established in the SDK. Having a robust and flexible SDK allows for the addition of new functionality into the platform in a consistent and structured methodology, and the fact that LEAF uses a consistent language such as JavaScript, enables the easy extension of functionality by third party developers.

BUILT TO SCALE

The entire platform is designed for flexibility in both the application development and deployment – whether it is on-premise servers or cloud based systems.

Built around the concept of DevOps and Continuous Integration, LEAF is ideal for cloud-based environments like Amazon Web Services as it can be designed to horizontally scale to meet cyclical or peak performance demands. As micro-services are utilized, additional temporary containers can be automatically spawned to meet the increased demand on those services. When the demand decreases the system will scale back and dispose of the temporary containers. This functionality allows LEAF to scale to meet the most demanding enterprise functionalities.

By using LEAF, enterprises can enable rapid application development, enforce application consistency and delivery, as well as the streamlining of application provisioning and deployment.

To learn more, contact us today:
304-876-6600 or info@krminc.com