TEAM INSIGHTS ARCHITECTURE PRINCIPLES

Cloud Architecture

1. **AWS** - Team Insights will run in AWS.
2. **Managed Services** - We will leverage managed services and automate where possible to reduce operational costs (e.g. people). An engineer should be able to support the first 5 clients.
3. **Client Accounts** - Each customer will have their own dedicated AWS accounts (Prod and NonProd) that provide security, network and storage segregation at the client level.
4. **Automated Provisioning** - The setup of a new client will be automated from provisioning the account and resources to deploying the application. This should take less than 1 hour to complete.
5. **Multi-Region Active/Passive** - For increased resiliency and availability, the system will be capable of running in multiple U.S. geographic regions and multiple availability zones (e.g. data centers) in an Active/Passive scenario.

Application Architecture

1. **Domain Driven Design** - We will use a domain-driven design approach to modeling the team domain area. This design will be implemented using object-oriented principles
2. **API First** - All modules will expose API's for querying and modifying the system. These API's will be used internally and externally (by clients).
3. **Modular Services** - At runtime modular services will perform the application functions. These services will be developed in alignment with micro-service architecture principles and practices.
4. **Service Level Objectives (SLO's)** - Team Insights will be architected to meet the internal Service Level Objectives (SLO's) related to availability, response latency, throughput and error rate.
5. **Container First** - Each modular service will run in a Docker container that is deployed to a Kubernetes cluster at runtime.
6. **Mobile First** - clients interact with Team Insights with a user interface optimized for a mobile experience but that also looks good on web desktops and tablets.
7. **Deployed to App Store** - Team Insight's mobile application will be capable of being deployed to private app stores (both ours and our customer's) leveraging enterprise mobile device management capabilities.

Dev & Ops Processes

1. **Every Commit Triggers a Build** - Every commit to code repos will trigger a build job. Successful builds will result in release candidates stored in a binary repo.
2. **Automated Deploys** - Deploys can be triggered manually or via a successful build. Deploys may also be gated via automated checks and/or manual approvals.
3. **Infrastructure as Code** - All infrastructure will be defined in code and modified via pipelines. Once infrastructure is created it will not be modified, instead it will be rebuilt via automation. We will use CloudFormation for all infrastructure code.