|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Enabler Name\* | Federated Data Graph exploration using Apollo GraphQL Server and the Common Hotel Database(CHuD) | | | | | Infrastructure Enabler  Architecture Enabler  Epic Enabler | | | | Exploration Enabler  Compliance Enabler  Feature Enabler | |
| Summary\* | To consolidate the databases into an apollo GraphQL server and make it so there is one call that can go into the databases and microservices and pull out the specific data that the one call has made | | | | | | | | | | |
| Detailed Description\* | **Benefit hypothesis:**  *This section should provide the necessary information to convey the need and benefits of doing this enabler.*  By moving over to GraphQL ,we can mitigate a lot of performance issues by having to go into different databases and microservices and tailor the API call towards specific data that the user wants. It increases speed, efficiency, improve deployment, improve customer satisfaction and simplifies the current architecture  Need: HCCD is a developing platform to support multisource content. The quantity and quality of the data will be enhanced from added content from our multi source supplier (Gimmonix). A new NoSQL platform is being introduced (CosmosDB). Important to develop APIs that support delivery of that data in a concise and consistent way preventing over/under fetching currently found in existing model. Delivery of static content will require federation of data from different sources, including databases and REST services. Travelport+ next gen platform, corporation would like to perform simplification of architecture. HCP/Pre-CHuD previous platform, HCCD has experienced performance issues, resulting in the expansion of architecture complexity which causes challenges in maintenance and cost of HCCD platform.  Benefit: Determine if leveraging GraphQL Server can mitigate the previous performance and data access issues by creating a federated data graph of various data sources and the ability of GraphQL queries to allow user or service to consume only data that they need. Addition of technology should increase speed, efficiency, improve deployment of new features, improve customer satisfaction and improve architecture.  ***Background****:*  *The background section is optional, however, this could be a good area to cover any historical or architecturally relevant items as well as impacted features.*  *Lol basically:* [*https://docs.google.com/document/d/1krz08P8E-I0wUjs8AO2iCvqOBoR63f03VWFr\_0AzuZE/edit*](https://docs.google.com/document/d/1krz08P8E-I0wUjs8AO2iCvqOBoR63f03VWFr_0AzuZE/edit)  CHuD – HCCD Static Hotel Content Database built on CosmosDB using MongoDB interface  Apollo GraphQL server  Azure | | | | | | | | | | |
|  | | | | | | | | | | | |
| Technical Benefits\* | Scope\* | | | Architectural tiers/ layers | | | Technical Risks | | | | Assumptions |
| *Bullet out the benefits to the org (could be time, compliance, cost, support, HW/SW footprint)*   * Simplify architecture * Faster deployment * Allow customers to get data they want * Increase performance | *Can document what is in and out of scope – especially if this is an iterative enabler*  Bounded by working with CHuD and ancillary data on MS Azure platform | | | Presentation  Business rules  Channel  Domain  Atomic  Service  Data model  Database  Core  Support  Other: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | *Document risks for doing and not doing this enabler; can tie these to the business features as well as to customers* | | | | *Can be about the enabler or the dependent features*  Assume access to cloud platform and can stand up our own environment for the experiment and the subject matter experts will be available on a very limited basis to assist with any questions we may have about existing environment |
|  | | | | | | | | | | | |
| Acceptance Criteria\* | | Success Measures\* | | | Leading Indicators | | | | NFRs (following are of particular interest) | | |
| 1. *Detail out specifics that can be tested and approved*   *Pulling data from different microservices at a time and testing to make sure that it is the data that we want and was not under/over fetched. Then we move onto different services and multiple services and test for accuracy* | | *Be specific in how we determine if this enabler provided the right value – e.g. allowed NFRs to be met*  Build a new version using GraphQL of the CHuD query app and show off at BEER or system demo to HCCD and being able to prove that it is a viable alternative with just using REST  Leadership wanting more focus on R&D | | | *E.g.*  *Planned adoptions*  *Actual adoptions*  *NPS* | | | | Security/Privacy  Availability  Failover/Disaster recovery  Efficiency  Maintainability  Response time  Reliability  Scalability  Performance that can be compared against REST | | |
|  | | |
| Estimated Size | | Planned Start/End (month/qtr) |
|  | |  |
| Jira Epics, Stories, Aha! Features, and Any existing feature/story level documentation pertaining to PI1: | | | | | | | | | | | |
| Source (e.g. Jira, Aha, etc) | | | Link: | | | | | Friendly Description: | | | |
|  | | |  | | | | |  | | | |