Utopia Architecture Notes (2020) Has Changed

utopia_diagram.png
Image not found or type unknown

Download link: Utopia Architecture Notes.docx

DataAccess Notes

- Contains DB entities
- Method Classes
  - Used for uniform retrieval of data from the database that doesn't require knowledge of business rules
  - Used by BusinessLogic to compous various, reusable IQueryables to get informationit needs for a given operation
  - Used for updating in case of additional database specific work. i.e. When adding a DbNode, it sets the CreatedOn and ModifiedOn values prior to adding them to the DbContext
  - The primary goals for the Method classes in DataAccess are to make Get, Add, Update, and Delete reusable and abstract so any change to the Database layer (i.e. going to NoSQL) will require far less work
- Contains common extension methods for any of the DB entities
- Contains any classes that are required to do optimized batch queries (i.e. Retrieving permissions by NodeID, although this could probably be converted to some form of IGrouping in the future)

BusinessLogic Notes

- Contains all Business Logic
- Split into three layers: **BaseLogic**, **Logic**, **FacadeLogic**

BaseLogic Layer

  - Used for sharing reusable logic
    - Getting permissions for nodes for NodeLogic operations
  - Used for intercommunication between logics

        - Think Validation operations
    - These logics act as parent objects for their corresponding Logic layer counterparts. This allows logic that needs to be reused in both the Logic and Base Logic layers to be a part of that logic only.
        - For example: A method in NodeBaseLogic is marked as private, enabling it's use by NodeLogic due to NodeLogic inheriting from NodeBaseLogic. This ensures that the method can only be used from these two classes.
    - This layer is up for debate.
        - We have also considered a *Logic Components* section that contains the reusable logic, as well as a *Validation* section that contains all the logic required for validation in a formal format (most cross-logic communication occurs due to validation requirements).

Logic Layer

- Primary layer containing most logic
- There is no cross-communication between the Logics
- Utilizes Base Logics for reusable logic
- Utilizes DataAccess Methods for DB interactions as much as possible
    - Should avoid accessing the DbContext directly, where possible

FacadeLogic Layer

- Used primarily by External Node Providers
- A transparent layer to whatever wishes to support external Node Providers (Google Drive, OneDrive, etc.)
- Does no other logic than getting the correct provider for the given ID/Operation
- Bypassed by any operation that is not supported for external providers
    - Currently everything except Node Operations, File Download Operations, and File Upload Operations bypasses this layer and goes directly to the Logic layer

EventSystem Notes

- Lives in BusinessLogic
- Consists of 3 parts: **Events**, **Listeners**, **EventManager**
- Any call to the EventManager, regardless of location, will initialize the Manager and invoke all BusinessLogic level listeners. This allows for Audit Logging, EventTrigger Servicing, Notification Creation, and other "reactions" to system events to occur transparently
- Events are handled Asynchronously to allow original requests to process as quickly as possible while another thread handles all "reactions"
- Listeners can be hooked up from upper layers
    - i.e. SignalR on the App Server

UtopiaSharedClass Notes

- Intended to be shared with customers and thus should contain no sensitive information
- DataAccess utilizes UtopiaSharedClass mostly to reuse it's Enums

Security

- https://cheatsheetseries.owasp.org/cheatsheets/DotNet_Security_Cheat_Sheet.html

---

Revision #3
Created 18 July 2022 23:37:41 by Bryce Holloway
Updated 9 February 2023 20:20:49 by Bryce Holloway