Fernando Gonzalez-Cruz fg32 120699549
Neyida Michel nm68 142843512
Roger Dai qd8 145816347

Design: The server package is responsible for processing all incoming http requests, and routing them to the appropriate handlers. We have four handler packages: resourceGetterService (for GET requests for resources), resourceDeleterService (for DELETE requests on resources), resourceCreatorService (for PUT and POST requests on resources), and resourcePatcherService (for PATCH requests). In addition, the auth package provides the server with an authentication mechanism via bearer tokens, ensuring only logged-in users can access the database. Once a request has been processed and forwarded to the correct handling package, it is dispatched to a Database struct (defined in package db), that then determines whether the request is to a top-level document, or to a child document/collection. If it is the latter, it finds the top-level document, implemented by the document package, and forwards the request. From there, the document is responsible for operations on collections of children. The validation package is used by both the resourceCreatorService and document to validate the creation and editing of documents. The subscriptionManager package contains functionality for both adding and notifying subscribers to documents and collections; it is injected into databases,documents, and collections, so that each can manage and process subscription requests. The patch package's sole responsibility is to perform patches on json objects; it is injected into documents into main so that they can patch their children. The concurrentSkiplist package implements a concurrent skiplist with lazy synchronization, and this is used throughout OwlDB as a performant way to store documents, collections, databases, authentication tokens, and active subscriptions. index_utils is solely to define the Pair[K,V] and updateCheck types, which are necessary for our skiplist implementation.

Concurrency: At the core of our concurrency handling is a concurrent skiplist that uses lazy synchronization with atomics to ensure safe creation,modification,retrieval,and deletion of databases, documents, and collections. Our databases hold documents within skiplists, and our documents contain skiplists that contain zero or more collections, which themselves contain skiplists of documents, ensuring concurrent synchronization. We also leveraged skiplists in our implementation of subscriptions and authentication, mapping unique IDs to subscribers in document and collection subscription managers, and mapping tokens to active users respectively.

In addition, we leveraged unbuffered channels to facilitate safe passing of SSEs from the server to the client, enabling our database to handle concurrent subscription requests.

Design: We leveraged the following design principles in our design:

1.Single Responsibility Principle: we leveraged the single responsibility principle by developing packages that had a single responsibility. To use an example, our auth and patch packages' sole functionality are to authenticate users and to perform patches on json objects respectively; they have no other utility and are exposed to other packages that need them via interface that is injected in main.

2. Interface segregation principle: we leveraged this mainly in our four packages for resource creation, getting, deleting, and patching. We originally had the server contain a giant interface with the methods of all four in one, but decided to split them up, as neither's functionality depended on each other, and doing so enabled easier testing.

3. Dependency inversion: Many of our packages rely on a concurrent associative array mapping keys to values for their operations (see the diagram). Thus, we decoupled our implementation from it's behavior within these packages via interfaces, which are instantiated and injected as needed in main, ensuring loose coupling between the concurrentSkiplist package and all other packages reliant on a concurrent skiplist.

(various): enables concurrent storage of key-value pairs

concurrentSkiplist

enables document to perform patches

Patcher

document*

enables document to validate patches

enables subscription management for documents and collections

validation

validates incoming docs

enables database to forward requests to create documents & collections

subscriptionManager*

enables database subscription management

Enables getting databases

db*

enables forwarding of patch requests to documents in a database

resourceGetterService

enables forwarding of deletion requests for documents/collections

resourceDeleterService

resourcePatcherService

enables getting creating databases & forwarding requests to make documents/collections

enables the server to forward get requests, and receive response bodies & status codes, also provides channels for subscriptions

resourceCreatorService

enables server to forward resource creation requests

enables deletion of databases & forwarding of deletion requests to documents/collections

enables server to forward patch requests to databases for a particular document

server

auth*

provides server with authentication via bearer tokens

All packages are imported in main, which injects dependencies within constructors and starts the server

main

Injects all dependencies within constructors; starts the server.

Legend:

- A black arrow from package A to B indicates that package A implements an interface defined in B
- A blue star indicates that the package has an interface implemented by skiplist
- An asterisk indicates that the package imports index_utils, which defines our Pair and updateCheck types for usage with skiplist