**GitLet Architecture**

**Functionalities**

* *Commit*, saving the content of entire directories of files.
* *Checking out*, restoring the version of a specific commit.
* *Log*, viewing the history of your backups.
* *Branches,* maintaining parallel sequences of commits.
* *Merge,* combining branches together.

A picture containing drawing

Description automatically generated

* We are able to change where the head is pointed at.
* At the head GitLet is also able to branch out into different paths.
* **Commit trees are immutable.**

**Structures**

* *Blobs:* the content of files. Basically, just a file object.
* *Trees:* Directory mapping, a tree representing the structure of a git repository.
* *Commits:* combinations of log messages, other metadata (time stamp, commit message, map of filenames to blob objects, primary and secondary commit reference for normal commits and merges respectively), a reference to a tree, and references to parent commits. (kind of like a repository?)
  + Commits and blobs have distinct SHA-1 hash values.
  + Blobs with same content are considered to have the same hash. Commits with same value have the same hash.

**Specs**

* Everything needs to be stored in .gitlet (old copies of files and metadata.)
* If a user doesn't input any arguments, print the message Please enter a command. and exit.
* If a user inputs a command that doesn't exist, print the message No command with that name exists. and exit.
* If a user inputs a command with the wrong number or format of operands, print the message Incorrect operands. and exit.
* If a user inputs a command that requires being in an initialized Gitlet working directory (i.e., one containing a .gitlet subdirectory), but is not in such a directory, print the message Not in an initialized Gitlet directory.