GitTor

Senior Design Team 15

Cameron, Isaac, Jayson, Phu, Seth, Tyler

IOWA STATE UNIVERSITY

What Is GitTor?

Problem

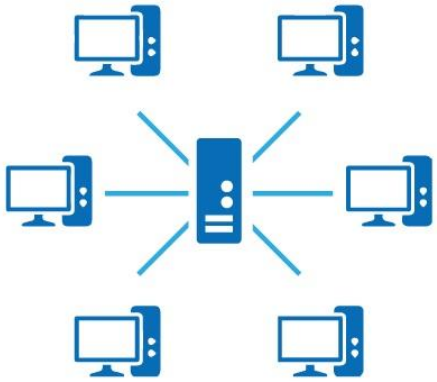
Git hosting services have issues:

- Single point of failure
- No fallback collaboration system
- Judge / Jury / Executioner
- Commit addition/removal
- Commit injection

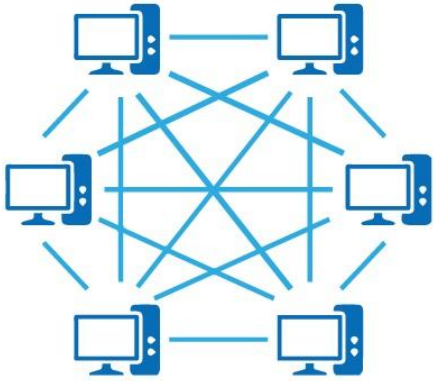
Solution

Decentralized Git hosting system:

- Every contributor shares the repository
- Layers of redundancy
- Power is in the people
- No third-party access



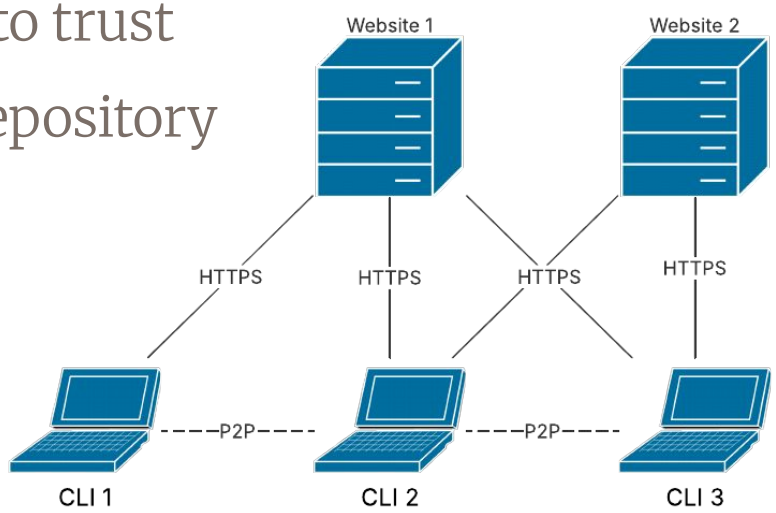
A **Server** based Network



A **Peer-to-Peer** based Network

Design

- Upload the “link” to the GitTor website
- Others get the “link” from the website
- Share repository via P2P
- Choose websites instances to trust
- The “link” changes when repository is updated



Project Setup

Repositories

- CLI Tool
  - Written in C
- Web Application
  - Frontend
    - TypeScript w/ Angular
  - Backend
    - Java w/ Spring Boot

Styling

- All codebases have designated linter and formatter
  - Uniform
  - Increase readability
  - Reduce code smells

Testing

- CLI Testing
  - Unity Framework
  - Integrated into our automated build pipeline
- API Testing
  - JUnit Framework
  - 94% code coverage, 100% of methods
- E2E Testing
  - Cypress

Figure 1: *Unity Coverage Reports*

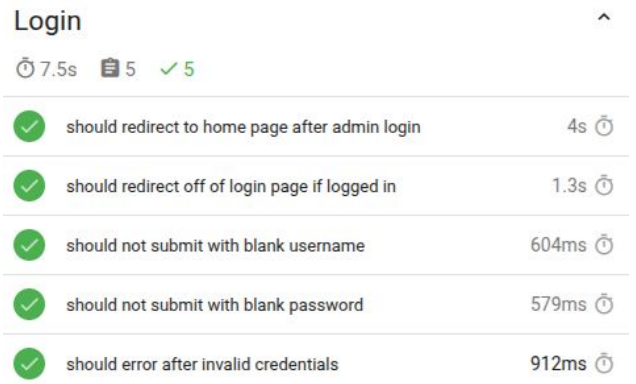


Figure 2: *Cypress Test Reports*

CI/CD

- Automated CI/CD Pipeline in GitHub
  - Tests ran for every push to main and PR
  - All tests must pass before PR approval
- Coverage Reports
  - Unity Framework

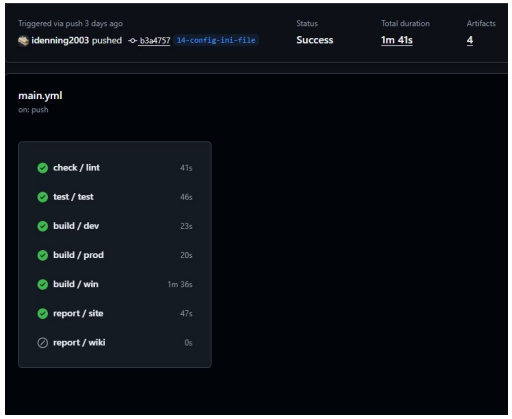


Figure 1: *Passing CI/CD Test Results*

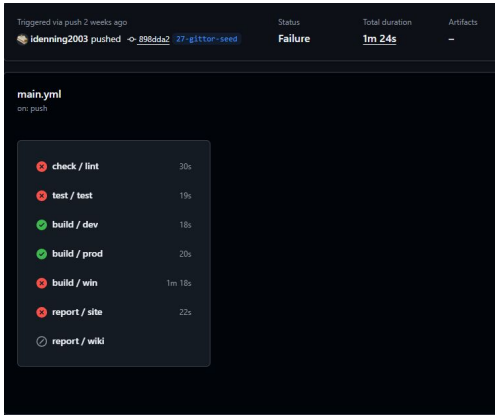


Figure 2: *Failing CI/CD Test Results*

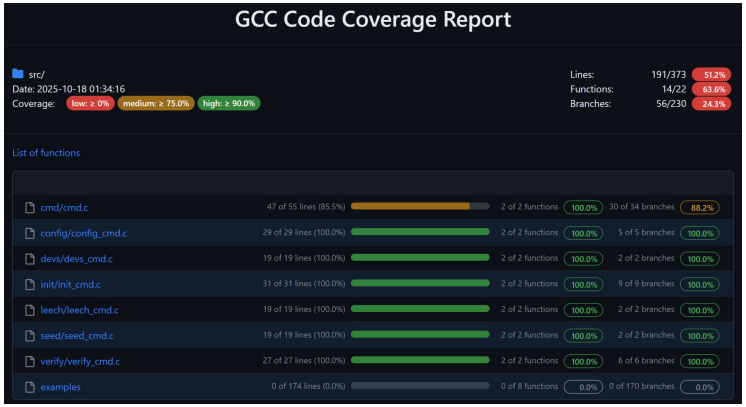


Figure 3: *Unity Coverage Reports*

Command Line Interface

Prototyping

- Proof of Concept
  - Torrenting a repository manually
- Seeding
  - Sharing the repository
  - qBittorrent
- Leeching
  - Downloading the repository
  - libTorrent

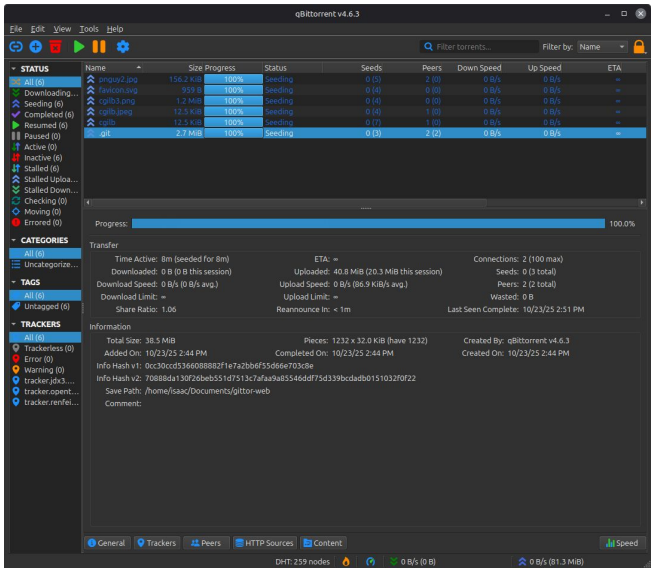


Figure 1: *qBitTorrent Seeding*

```
C:\Users\cgilb\Downloads\gittor-exe (2)>gittor tor
seeding 885 kB/s 2915 kB (100%) downloaded (0 peers)
saving session state

done, shutting down
```

Figure 2: *GitTor application leeching using libTorrent*

Build

Makefile defines build targets

- ➔ Development / Production
- ➔ Test / Reports
  - ◆ Automated with memory leak detection
  - ◆ Code coverage report
- ➔ Install / Uninstall

Windows build validated with GitHub Actions

```
·$ make
make: ./obj/prod/
make: ./obj/prod/gittor.o
make: ./obj/prod/login/
make: ./obj/prod/login/login_cmd.o
make: ./obj/prod/login/login.o
make: ./obj/prod/cmd/
make: ./obj/prod/cmd/cmd.o
make: ./obj/prod/init/
make: ./obj/prod/init/init_cmd.o
make: ./obj/prod/init/init.o
make: ./obj/prod/utils/
make: ./obj/prod/utils/utils_remote.o
make: ./obj/prod/utils/utils_git.o
make: ./obj/prod/verify/
make: ./obj/prod/verify/verify_cmd.o
make: ./obj/prod/devs/
make: ./obj/prod/devs/devs_cmd.o
make: ./obj/prod/config/
make: ./obj/prod/config/config_cmd.o
make: ./obj/prod/config/config.o
make: ./obj/prod/service/
make: ./obj/prod/service/service_com.o
make: ./obj/prod/service/service_seed.o
make: ./obj/prod/service/service_cmd.o
make: ./obj/prod/service/service_utils.o
make: ./obj/prod/service/service.o
make: ./obj/prod/seed/
make: ./obj/prod/seed/seed.o
make: ./obj/prod/seed/seed_cmd.o
make: ./obj/prod/api/
make: ./obj/prod/api/api.o
make: ./obj/prod/api/torrents.o
make: ./obj/prod/api/heartbeat.o
make: ./obj/prod/api/internal.o
make: ./obj/prod/leech/
make: ./obj/prod/leech/leech_cmd.o
make: ./obj/prod/leech/leech.o
make: ./bin/prod/
make: ./bin/prod/gittor
isaac@isaac-asus:~/Documents/gittor-cli·main
·$ make install
make: /home/isaac/.local/bin/gittor
```

Parser

- Argp interface from GNU C Library
  - We define the options and their behavior
  - It handles the parsing and validation
  - Provides error handling and help text generation

Config

- Manages gittor settings
  - Follows Git config pattern for familiarity
  - Allows global and local configurations
  - Example:

```
[network]
port=12345
api_url=https://gittor.rent/api
tracker1=https://tracker.moeblog.cn:443/announce
tracker2=https://tr.nyacat.pw:443/announce
tracker3=https://tr.highstar.shop:443/announce
tracker4=https://tracker.gcrenwp.top:443/announce
```

Init

- Initializes new Git repository
- Creates an initial commit for an identifier
- Makes the repository in desired location

```
·$ gittor init --help
Usage: gittor init [OPTION...] [DIRECTORY]
Initializes a new GitTor repository in the current directory.

  -?, --help                  Give this help list
      --usage                 Give a short usage message
```

```
isaac@isaac-asus:/tmp/project$ gittor init
isaac@isaac-asus:/tmp/project·main
·$ git lg
6683180 N - <Isaac Denning> (4 seconds ago): init  (HEAD -> main, origin/main, origin/HEAD)
isaac@isaac-asus:/tmp/project·main
·$ ls ~/.config/gittor/repos/
6683180baab273c7680a3f930b996dd6c10284bb
```

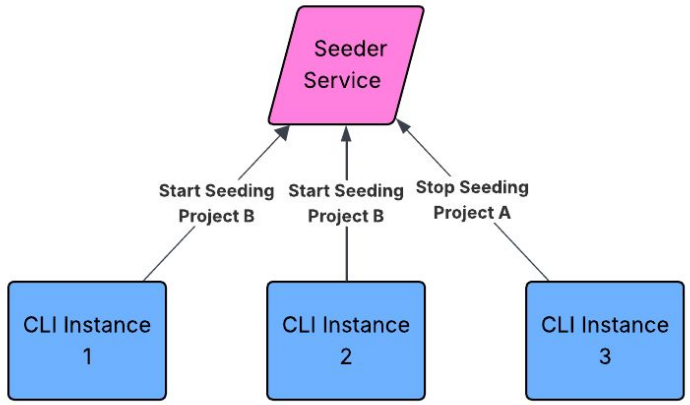
Login

- Commands that interacts with the web need authorization
- Prompts the user for email / username and password
- Temporarily and securely stores info

```
~/GitTor-Cli> gittor login
Email or username: sethc
Password:
Login successful.
~/GitTor-Cli> cat ~/.gittorconfig
[auth]
access_token=eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJzZXR
CT4skDW3RUsUOEbfwbascQ9hF-u62OH5HOLTV_8cP003A
token_type=Bearer
expires=2026-04-27T20:22:47.152+00:00
```

Seeder Service

- CLI starts and stops
- Seeding runs forever
- Seperate process for seeding
- Each CLI makes requests to the service
- Manage the service via CLI




```
·$ gittor service --help
Usage: gittor service [OPTION...]
COMMANDS:

  start    Ensures the GitTor service is running
  stop     Ensures the GitTor service is not running
  restart  Stops and starts the GitTor service
  status   Prints the GitTor service status (up, down)

OPTIONS:

  -?, --help                 Give this help list
      --usage                Give a short usage message
```

Seed

- Equivalent to `git push`
- Sends the repository to the seeder service
- Uploads the torrent file to the Web application
- Outputs repository information

```
isaac@isaac-asus:/tmp/repo·main
·$ gittor seed
GitTor service started.
Seeding Repository!
Repository ID: 375b00d5e5e99fcdf3bc4ec0403fde0e84a35223
Torrent File: /home/isaac/.config/gittor/repos/375b00d5e5e99fcdf3bc4ec0403fde0e
84a35223.torrent
Magnet Link: 'magnet:?xt=urn:btih:3774f1aed8329042fcb118fbf2c60aa1667080b3&xt=u
rn:btmh:12207cd200db7f583d03f3bde3ac520145c89e160d5fe9a24c469319d8efd6ae2186&tr
=https%3a%2f%2ftracker.gcrenwp.top%3a443%2fannounce&tr=https%3a%2f%2ftr.highsta
r.shop%3a443%2fannounce&tr=https%3a%2f%2ftr.nyacat.pw%3a443%2fannounce&tr=https
%3a%2f%2ftracker.moeblog.cn%3a443%2fannounce'
```

Figure 1: *Seeding basic repository*

Leech

- Leech with different Identifiers
  - Torrent File
  - Magnet Link
  - Repository ID
- Begins seeding after leeching

```
isaac@isaac-asus:/tmp$ gittor leech 'magnet:?xt=urn:btih:4da8597b113028b9d4312e
fad87252ca4245214f&xt=urn:btmh:1220129c976b21fc870527ce117e16e841f6f6d6cb53ef60
51e9e85897a6f3f9b2bc&dn=0d686ff9761d4b86be801bbc2cd35f15df8561fb&xl=163840&tr=h
ttps%3A%2F%2Ftr.highstar.shop%3A443%2Fannounce&tr=https%3A%2F%2Ftracker.gcrenwp
.top%3A443%2Fannounce&tr=https%3A%2F%2Ftr.nyacat.pw%3A443%2Fannounce&tr=https%3
A%2F%2Ftracker.moeblog.cn%3A443%2Fannounce' repo
Leech complete. Saving session state...
Closing leeching client...
```

Figure 1: *From magnet link*

```
isaac@isaac-asus:/tmp$ gittor leech 0d686ff9761d4b86be801bbc2cd35f15df8561fb repo
Leech complete. Saving session state...
Closing leeching client...
isaac@isaac-asus:/tmp$ cd repo/
isaac@isaac-asus:/tmp/repo·main
·$
```

Figure 2: *From repository ID*

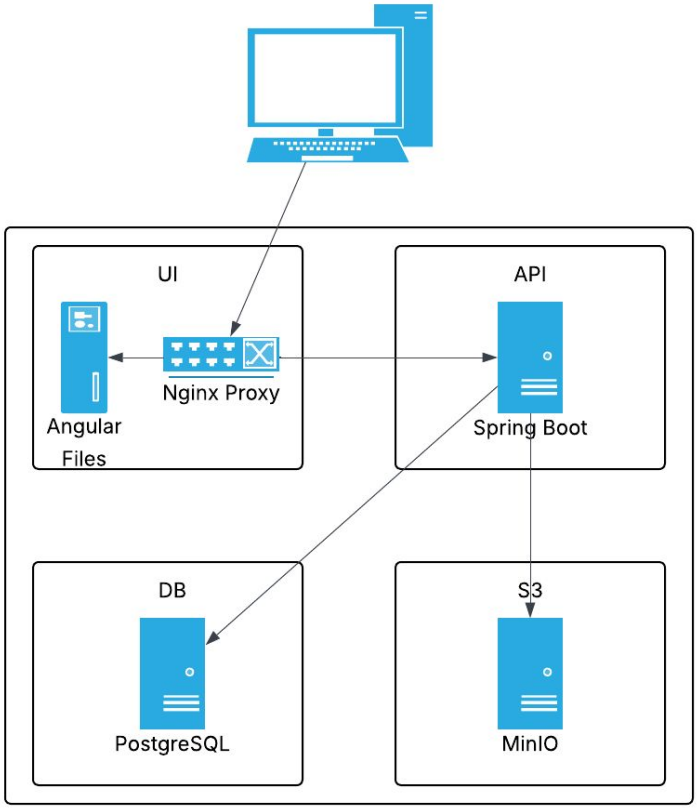
```
isaac@isaac-asus:/tmp/repo·main
·$ gittor leech
Leech complete. Saving session state...
Closing leeching client...
Fetched latest changes. Run `git pull` in '/tmp/repo' to apply the latest changes
```

Figure 3: *Leeching newest state*


Web Application

Docker

- Four containers (UI, API, DB, S3)
- Managed via Docker Compose
- Internal Network
- Only Nginx proxy is exposed (for security)

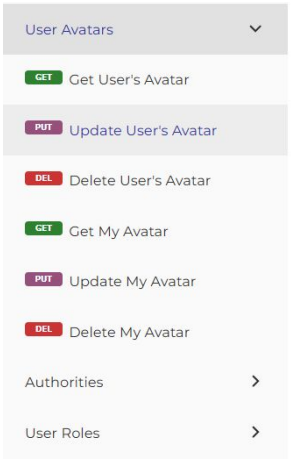


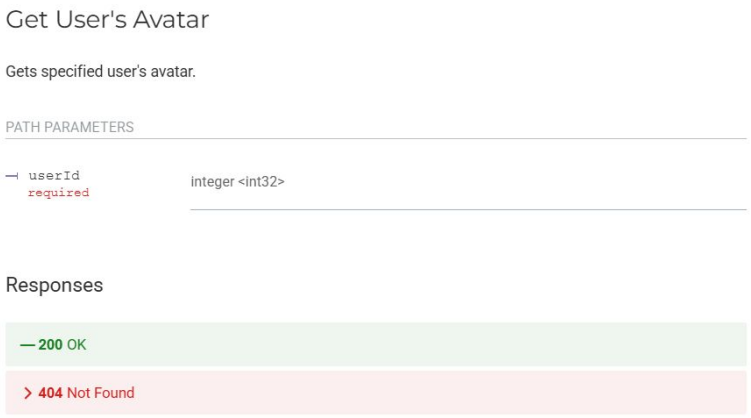
Endpoints

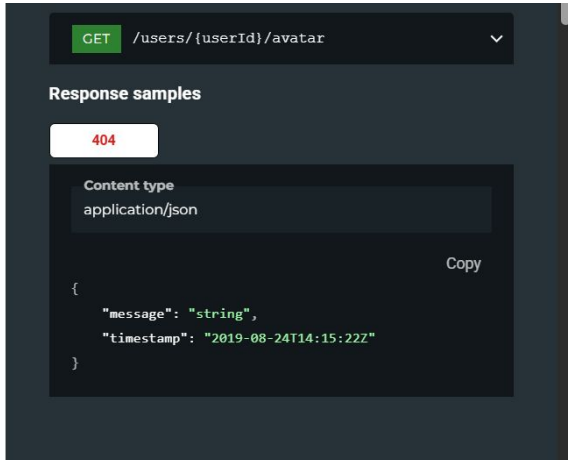
- Auth Endpoints
  - /authenticate — handles user login and registration
    - Login, register, refresh, log out
  - /roles and /authorities — handle user permissions
- User Endpoints
  - Get, edit, and delete users
- Torrent Endpoints
  - Upload torrent files w/ metadata
  - Get / update metadata or torrent file

OpenAPI

- Document our API endpoints
  - Details about response codes, inputs, mapping, etc.
- Auto-generated services and models in UI
  - Keeps consistency without any upkeep

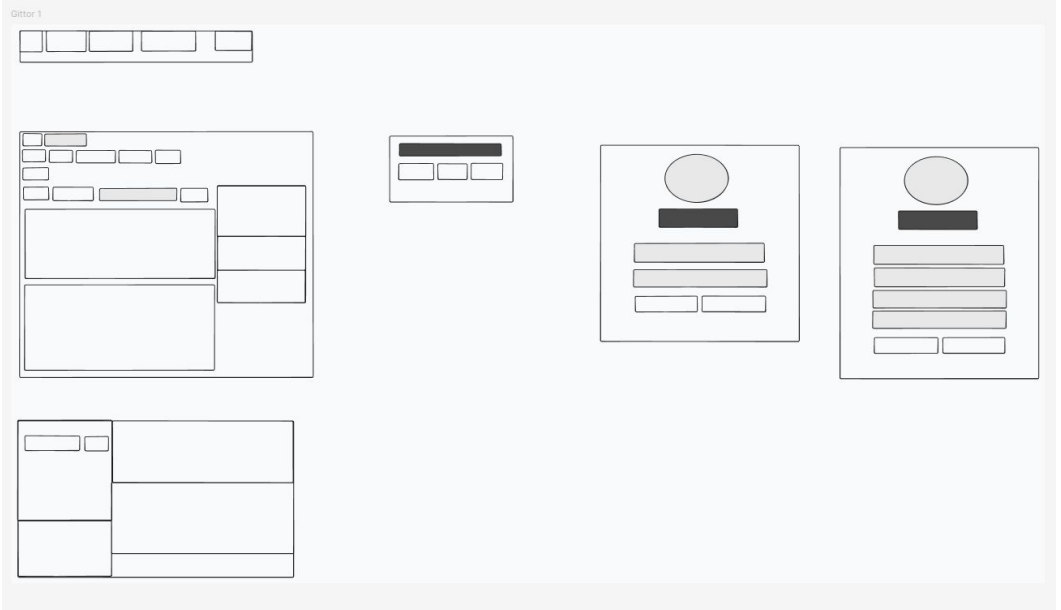






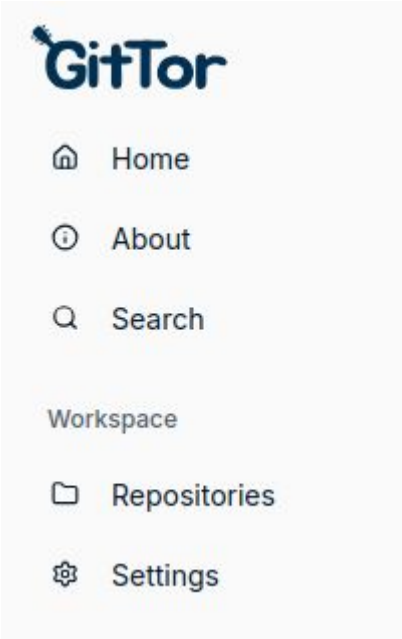
Frontend Design

- Used Figma and TLDraw to create a rough design
- Similar to Github's design yet still distinct
- Theme switching



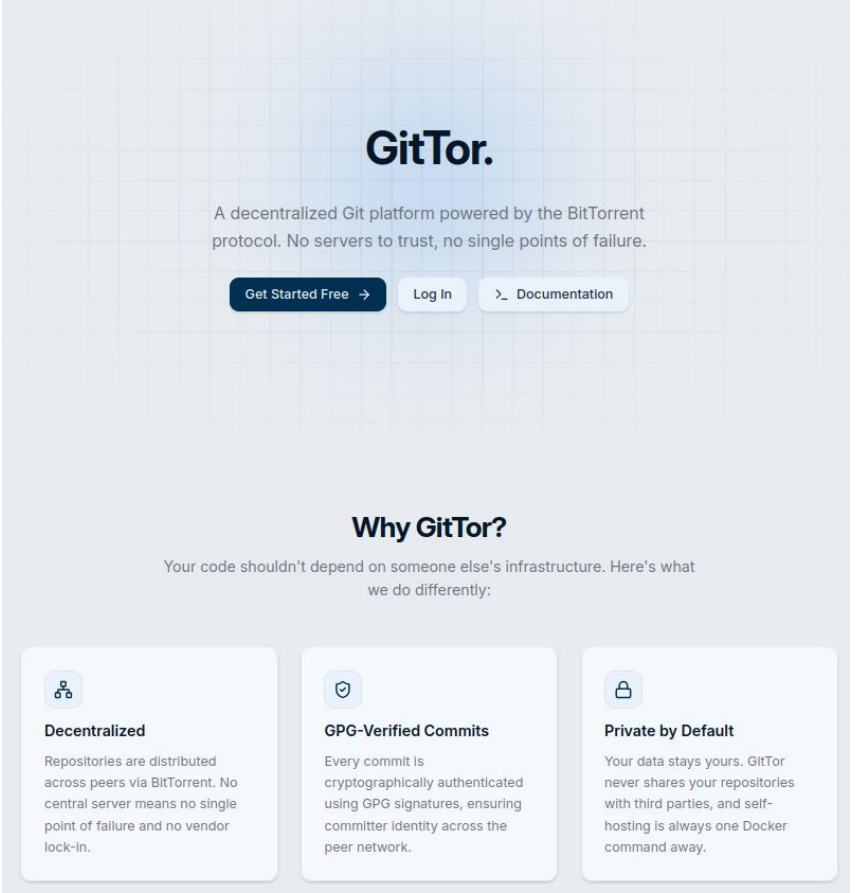
Navigation (Landing Page/Sidebar)

- Signed in users get the Sidebar
  - Home Repositories
  - Documentation
  - Search
  - Settings
- Logged out users get Landing Page



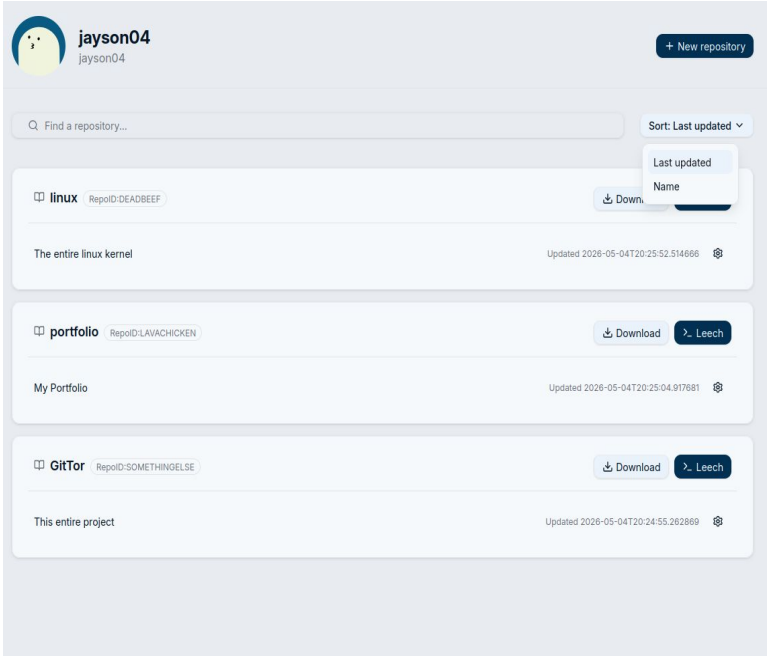
Landing Page

- Default page for new Users
  - Register
  - Log in
  - Documentation



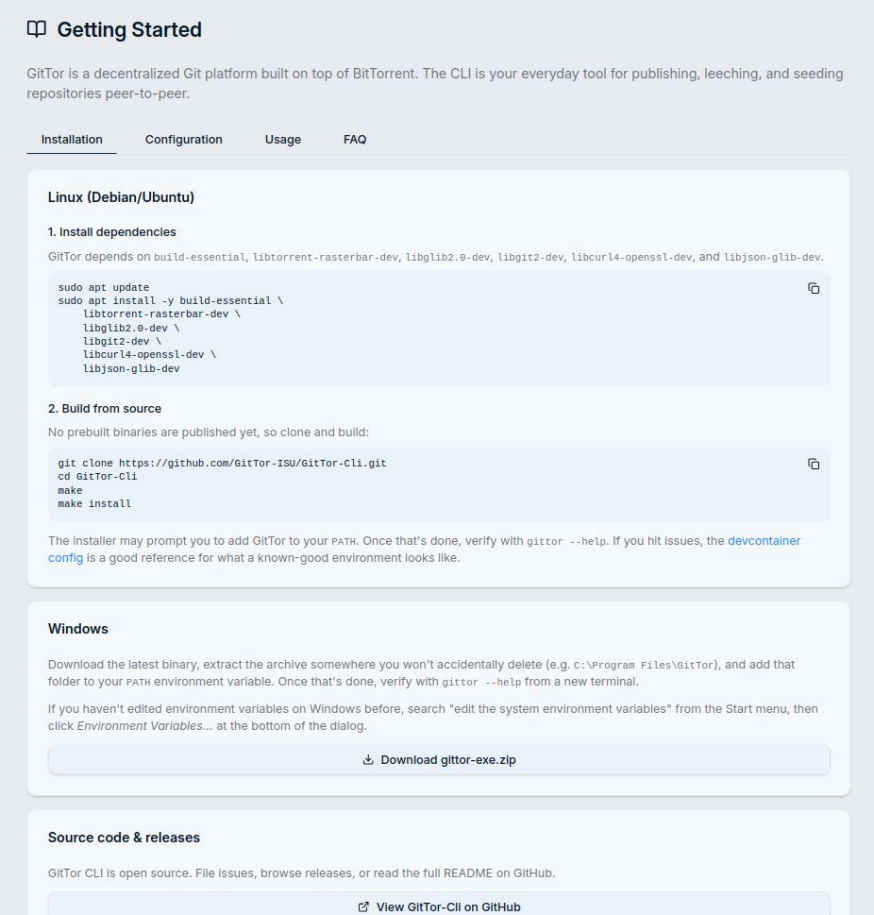
Repository List

- Default page for logged in Users
  - Search for your own Repos
    - Sorting
      - Name
      - Last Updated
  - Torrent Files
    - Leech command
    - Download .torrent file



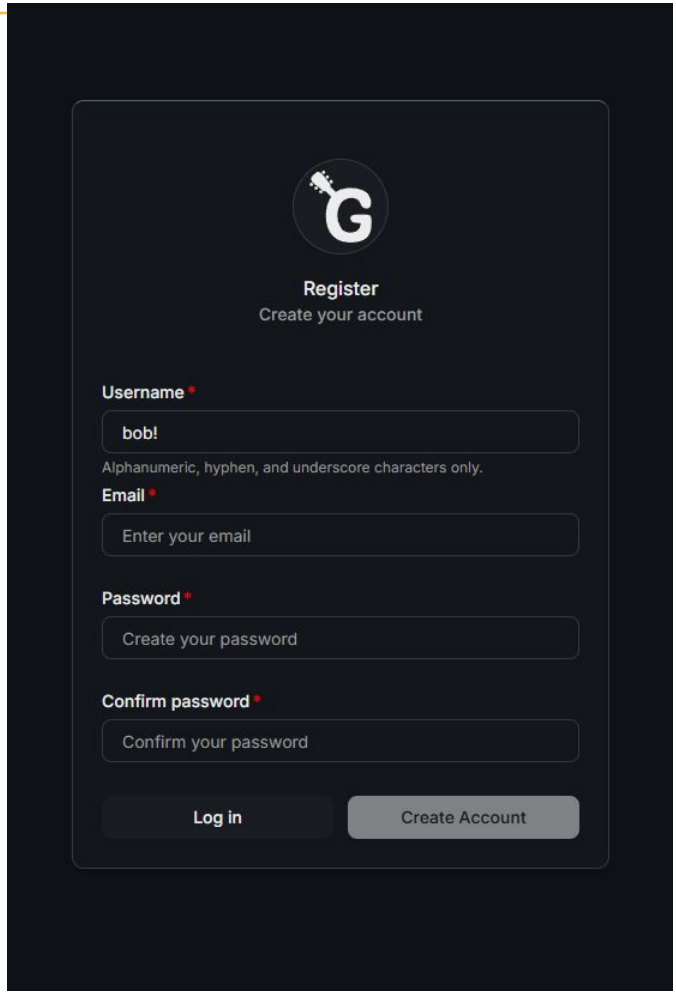
Documentation

- 4 Sections in a tab layout
  - Installation
    - Debian/Ubuntu
    - Windows
  - Configuration
  - Usage
  - FAQ



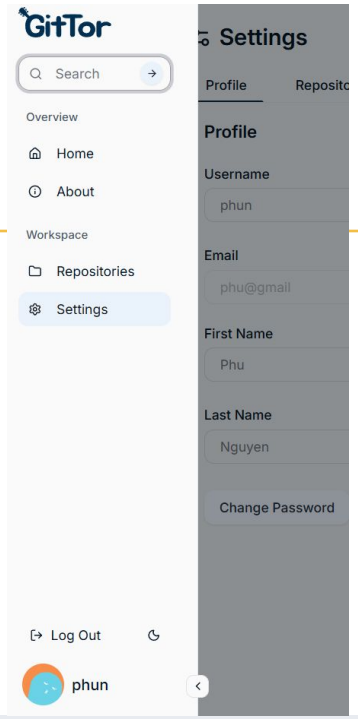
Login / Registration

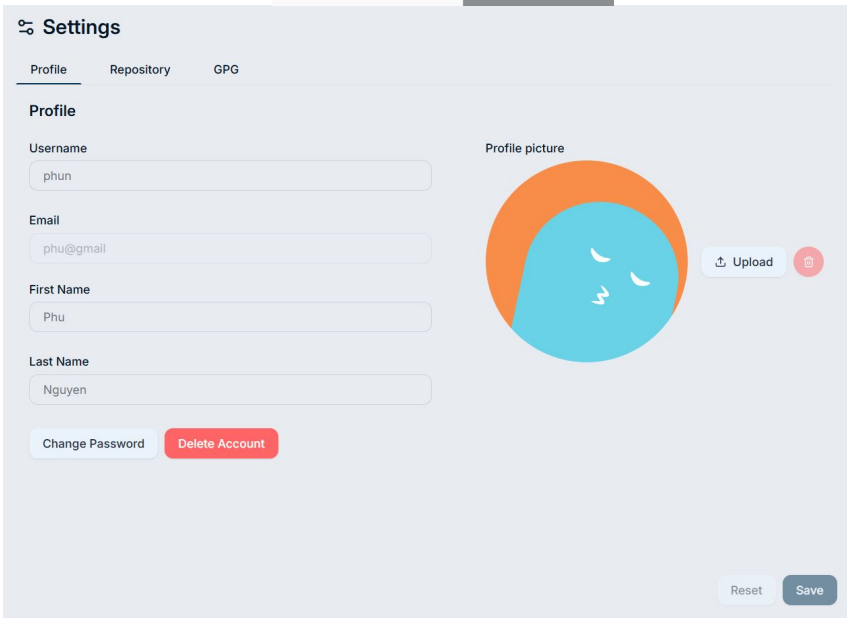
- Two token system
  - Access token - Access to API
  - Refresh token - Refresh access tokens
- Security
  - Access token always rotating
  - Refresh token stored in HTTP only cookie
- Form validation w/ clear directions for users



Settings

- Navigation
- Profile
  - Edit info
  - Generated default profile
- Repository
  - Edit repository
- GPG
  - WIP

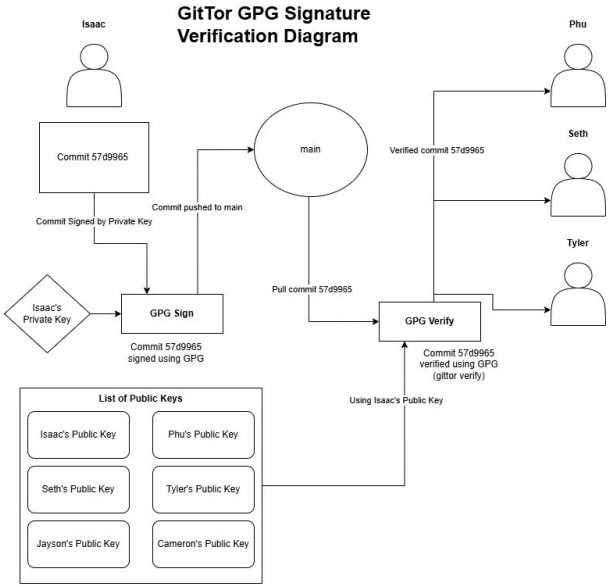


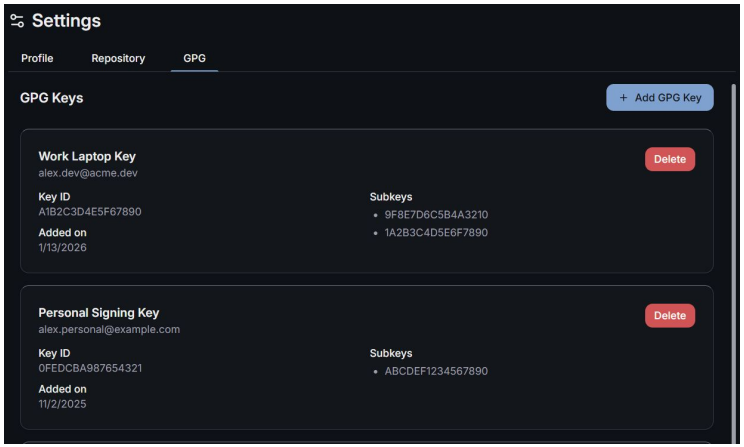


Future Work

GPG Authentication

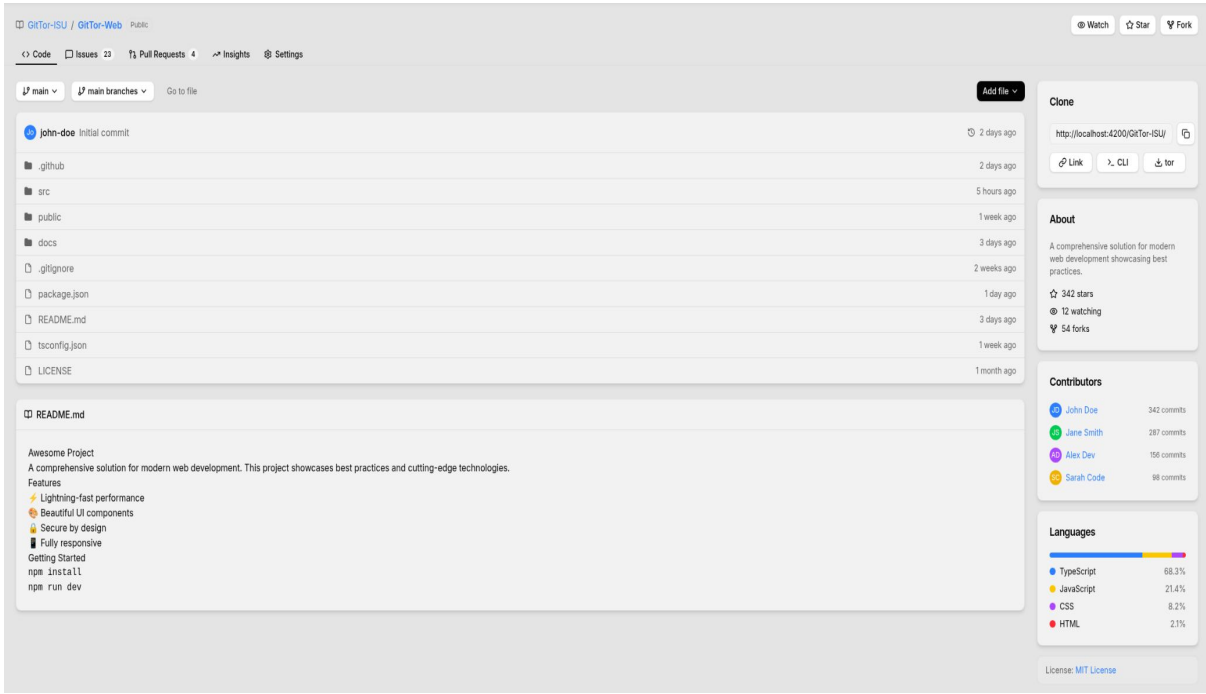
- Rollback repository to check each commit against authorized user file
- No Windows library support
- WebApp Integration





UI File View

- Viewing files from leeched torrent files
- Displaying file structure
- Displaying file contents



API Leeching and Seeding

- Backend only stores torrent files and metadata
  — can't see repository contents
- Need the API to leech the repository itself
- Two Methods:
  - Recreate torrent leeching & seeding functionality with Spring Boot
  - Embed CLI process in the API

Contact Us

sdmay26-15@iastate.edu

Cameron Gilbertson
Computer Engineering

cam2022@iastate.edu

github.com/cameron200316

Isaac Denning
Software Engineering

idenning@iastate.edu

github.com/idenning2003

Jayson Acosta
Computer Engineering

jayson04@iastate.edu

github.com/jacosta57

Phu Nguyen
Software Engineering

pnguyen2@iastate.edu

github.com/phu-n

Seth Clover
Software Engineering

sclover@iastate.edu

github.com/sethclover

Tyler Gorton
Software Engineering

tjgorton@iastate.edu

github.com/tjg23

Demo Video