GitTor

Senior Design Team 15

Cameron, Isaac, Jayson, Phu, Seth, Tyler

IOWA STATE UNIVERSITY

What Is GitTor?

Problem

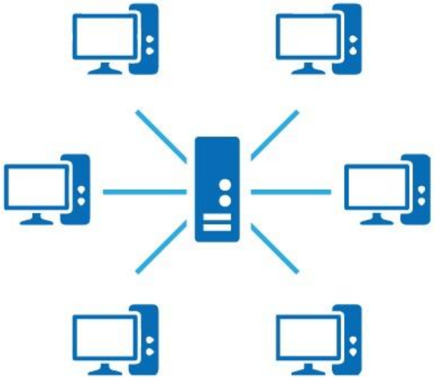
Git hosting services have issues:

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

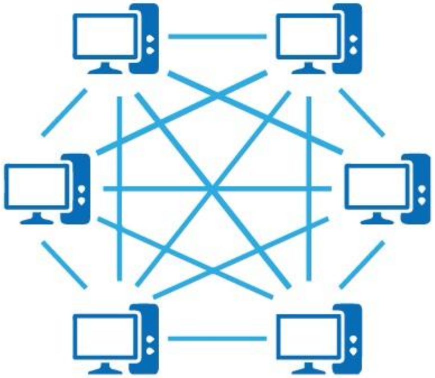
Solution

Decentralized Git hosting system:

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



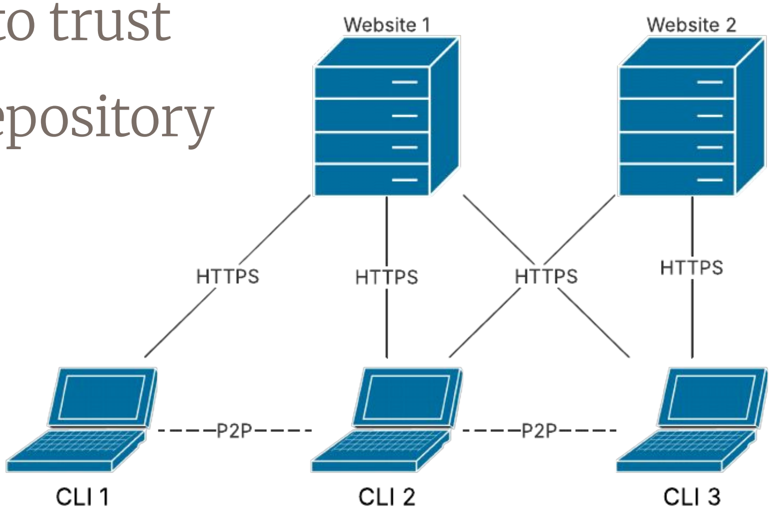
A **Server** based Network



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



What We’ve Done

Setup

Repositories

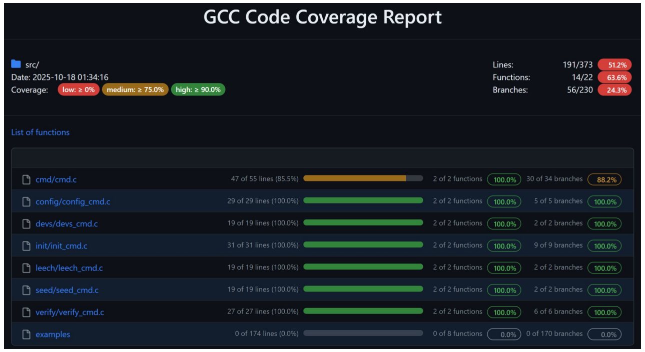
- CLI Tool
  - Written in C
- Web Application
  - Frontend
    - Written in TypeScript w/ Angular
  - Backend
    - Written in Java w/ Spring Boot

Styling

- All codebases have designated linter and formatter
  - Uniformity/Appearance
  - Reduce code smells
  - Reduce errors
- CLI - CppLint, Clang-Tidy
  - Google C Style
- API - Maven Checkstyle Plugin, Eclipse formatter
  - Google Java Styles
- UI - Eslint, Prettier
  - Along with many eslint plugins

Testing

- CLI Testing
  - Unity
  - Integrated into our automated build pipeline
- API Testing
  - JUnit Framework
  - 94% code coverage, 100% of methods
- Code Reviews
  - Kanban Board
  - Discord



GCC Code Coverage Report

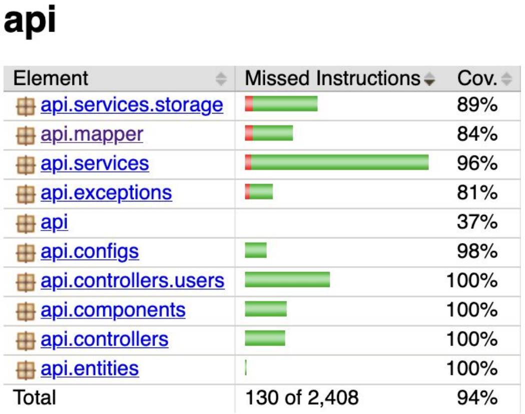
src/
Date: 2025-10-18 01:34:16
Coverage: low: ≥ 0% medium: ≥ 75.0% high: ≥ 90.0%

Lines:	191/373	51.2%
Functions:	14/22	63.6%
Branches:	56/230	24.3%

List of functions

cmd/cmd.c	47 of 55 lines (85.5%)	2 of 2 functions	100.0%	30 of 34 branches	88.2%
config/config_cmd.c	29 of 29 lines (100.0%)	2 of 2 functions	100.0%	5 of 5 branches	100.0%
devs/devs_cmd.c	19 of 19 lines (100.0%)	2 of 2 functions	100.0%	2 of 2 branches	100.0%
init/init_cmd.c	31 of 31 lines (100.0%)	2 of 2 functions	100.0%	9 of 9 branches	100.0%
leech/leech_cmd.c	19 of 19 lines (100.0%)	2 of 2 functions	100.0%	2 of 2 branches	100.0%
seed/seed_cmd.c	19 of 19 lines (100.0%)	2 of 2 functions	100.0%	2 of 2 branches	100.0%
verify/verify_cmd.c	27 of 27 lines (100.0%)	2 of 2 functions	100.0%	6 of 6 branches	100.0%
examples	0 of 174 lines (0.0%)	0 of 8 functions	0.0%	0 of 170 branches	0.0%

Figure 1: *Unity Coverage Reports*



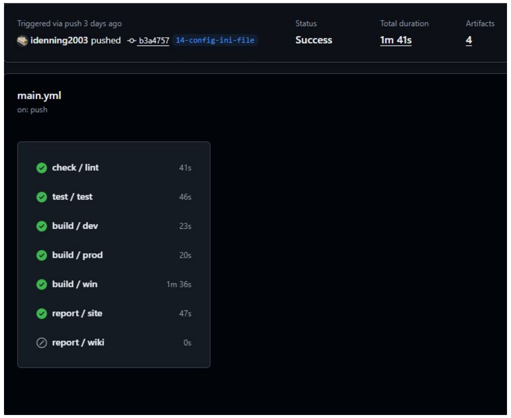
api

Element	Missed Instructions	Cov.
api.services.storage		89%
api.mapper		84%
api.services		96%
api.exceptions		81%
api		37%
api.configs		98%
api.controllers.users		100%
api.components		100%
api.controllers		100%
api.entities		100%
Total	130 of 2,408	94%

Figure 2: *JUnit Coverage 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



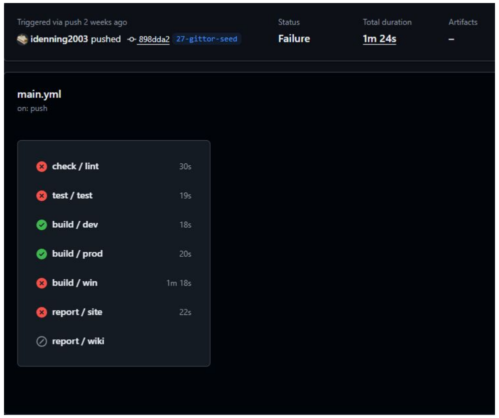
Triggered via push 3 days ago
idenning2003 pushed b3a4757 14-config-ini-file

Status: Success
Total duration: 1m 41s
Artifacts: 4

main.yml
on: push

✓	check / lint	41s
✓	test / test	46s
✓	build / dev	23s
✓	build / prod	20s
✓	build / win	1m 36s
✓	report / site	47s
⊘	report / wiki	0s

Figure 1: *Passing CI/CD Test Results*



Triggered via push 2 weeks ago
idenning2003 pushed 898dda2 27-gittor-seed

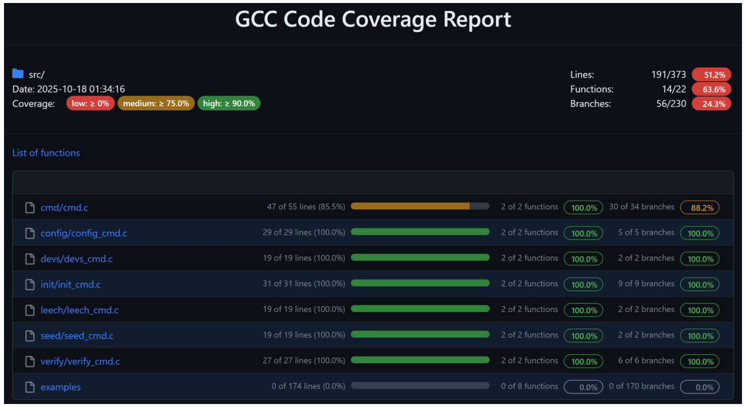
Status: Failure
Total duration: 1m 24s
Artifacts: –

main.yml
on: push

✗	check / lint	30s
✗	test / test	19s
✓	build / dev	18s
✓	build / prod	20s
✗	build / win	1m 18s
✗	report / site	22s
⊘	report / wiki	

Figure 2: *Failing CI/CD Test Results*

- → Coverage Reports
  - ◆ Unity



GCC Code Coverage Report

src/
Date: 2025-10-18 01:34:16
Coverage: low: ≥ 0% medium: ≥ 75.0% high: ≥ 90.0%

Lines: 191/373 51.2%
Functions: 14/22 63.6%
Branches: 56/230 24.3%

List of functions

cmd/cmd.c	47 of 55 lines (85.5%)	2 of 2 functions	100.0%	30 of 34 branches	88.2%
config/config_cmd.c	29 of 29 lines (100.0%)	2 of 2 functions	100.0%	5 of 5 branches	100.0%
devs/devs_cmd.c	19 of 19 lines (100.0%)	2 of 2 functions	100.0%	2 of 2 branches	100.0%
init/init_cmd.c	31 of 31 lines (100.0%)	2 of 2 functions	100.0%	9 of 9 branches	100.0%
leech/leech_cmd.c	19 of 19 lines (100.0%)	2 of 2 functions	100.0%	2 of 2 branches	100.0%
seed/seed_cmd.c	19 of 19 lines (100.0%)	2 of 2 functions	100.0%	2 of 2 branches	100.0%
verify/verify_cmd.c	27 of 27 lines (100.0%)	2 of 2 functions	100.0%	6 of 6 branches	100.0%
examples	0 of 174 lines (0.0%)	0 of 8 functions	0.0%	0 of 170 branches	0.0%

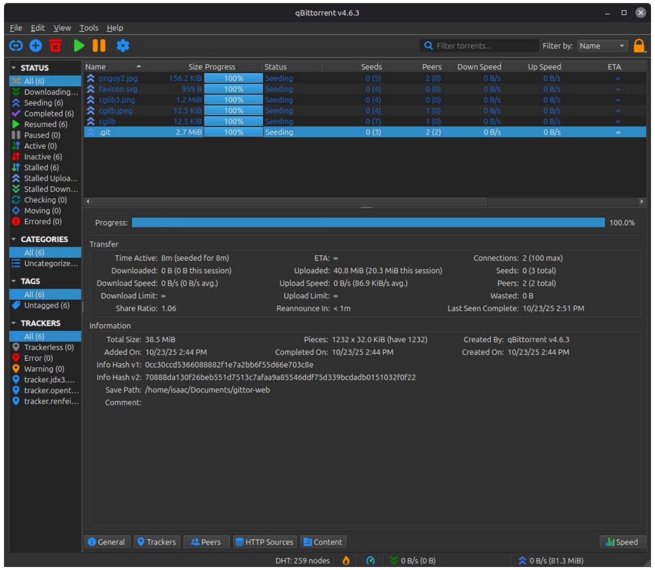
Figure 3: *Unity Coverage Reports*

What We’ve Done

Command Line Interface

Prototyping

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



Filter by: Name

Name	Size	Progress	Status	Seeds	Peers	Down Speed	Up Speed	ETA
pnguy2.jpg	156.2 KiB	100%	Seeding	0 (5)	2 (0)	0 B/s	0 B/s	∞
favicon.svg	959 B	100%	Seeding	0 (4)	0 (0)	0 B/s	0 B/s	∞
cgilb3.png	1.2 MiB	100%	Seeding	0 (4)	0 (0)	0 B/s	0 B/s	∞
cgilb.jpeg	12.5 KiB	100%	Seeding	0 (4)	1 (0)	0 B/s	0 B/s	∞
cgilb	12.5 KiB	100%	Seeding	0 (7)	1 (0)	0 B/s	0 B/s	∞
.git	2.7 MiB	100%	Seeding	0 (3)	2 (2)	0 B/s	0 B/s	∞

Progress: 100.0%

Transfer

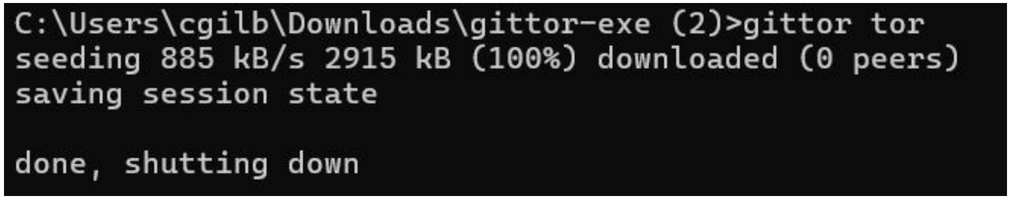
Time Active: 8m (seeded for 8m)
ETA: ∞
Connections: 2 (100 max)
Downloaded: 0 B (0 B this session)
Uploaded: 40.8 MiB (20.3 MiB this session)
Seeds: 0 (3 total)
Download Speed: 0 B/s (0 B/s avg.)
Upload Speed: 0 B/s (86.9 KiB/s avg.)
Peers: 2 (2 total)
Download Limit: ∞
Upload Limit: ∞
Wasted: 0 B
Share Ratio: 1.06
Reannounce In: < 1m
Last Seen Complete: 10/23/25 2:51 PM

Information

Total Size: 38.5 MiB
Pieces: 1232 x 32.0 KiB (have 1232)
Created By: qBittorrent v4.6.3
Added On: 10/23/25 2:44 PM
Completed On: 10/23/25 2:44 PM
Created On: 10/23/25 2:44 PM
Info Hash v1: 0cc30ccd5366088882f1e7a2bb6f55d66e703c8e
Info Hash v2: 70888da130f26beb551d7513c7afaa9a85546ddf75d339bcdadb0151032f0f22
Save Path: /home/isaac/Documents/gittor-web
Comment:

DHT: 259 nodes   0 B/s (0 B)   0 B/s (81.3 MiB)

Figure 1: *qBitTorrent Seeding*

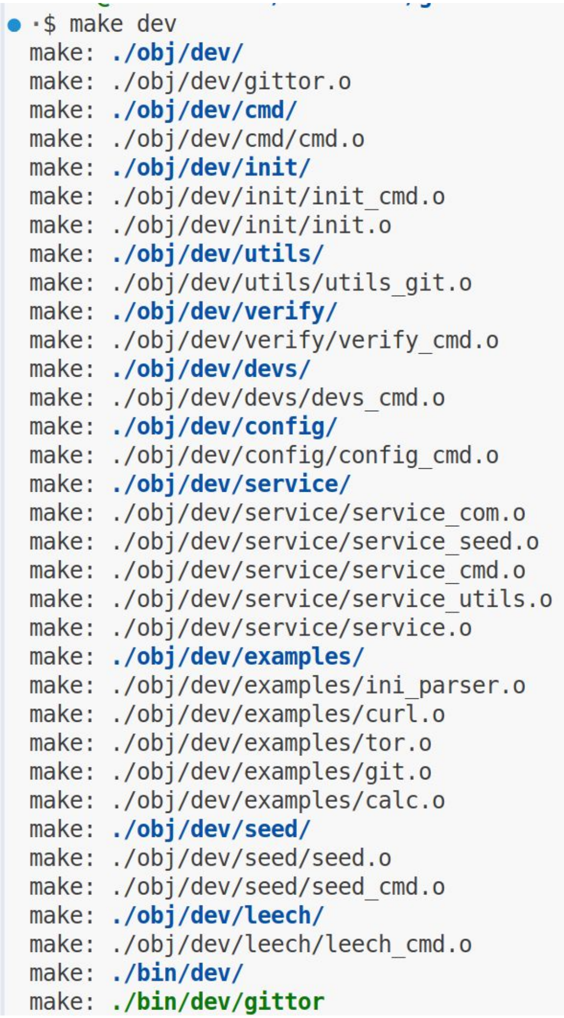


```
C:\Users\cgilb\Downloads\gittor-exe (2)>gittor tor
seeding 885 kB/s 2915 kB (100%) downloaded (0 peers)
saving session state

done, shutting down
```

Figure 2: *GitTor application leeching using libTorrent*

Build

- Makefile to define build targets
  - Development
  - Production
  - Test / Reports
- Added libraries
- C and C++ linking
- Windows build with Github Actions



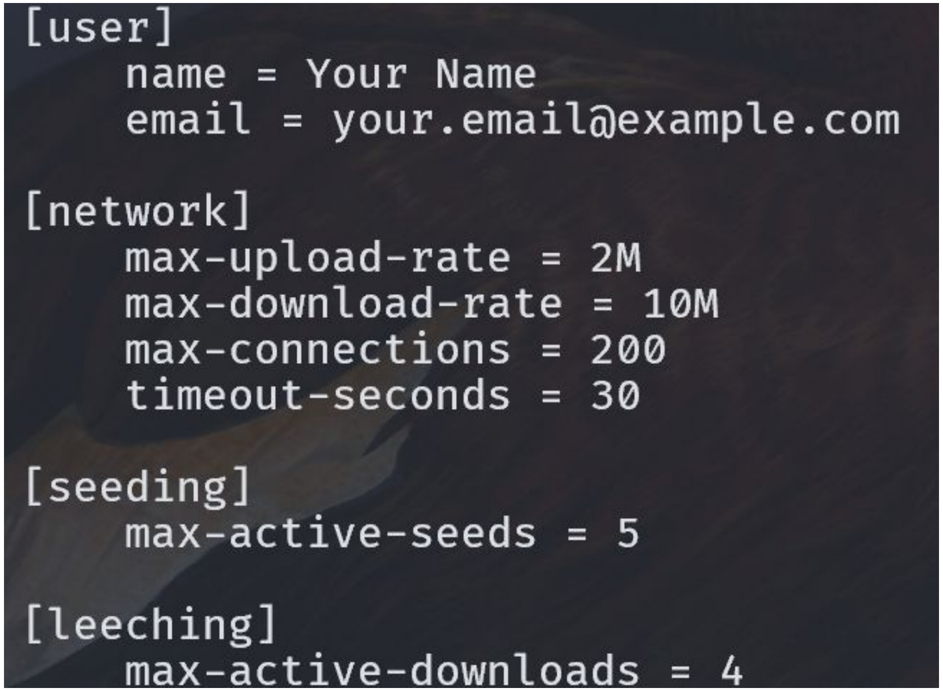
```
● ·$ make dev
  make: ./obj/dev/
  make: ./obj/dev/gittor.o
  make: ./obj/dev/cmd/
  make: ./obj/dev/cmd/cmd.o
  make: ./obj/dev/init/
  make: ./obj/dev/init/init_cmd.o
  make: ./obj/dev/init/init.o
  make: ./obj/dev/utils/
  make: ./obj/dev/utils/utils_git.o
  make: ./obj/dev/verify/
  make: ./obj/dev/verify/verify_cmd.o
  make: ./obj/dev/devs/
  make: ./obj/dev/devs/devs_cmd.o
  make: ./obj/dev/config/
  make: ./obj/dev/config/config_cmd.o
  make: ./obj/dev/service/
  make: ./obj/dev/service/service_com.o
  make: ./obj/dev/service/service_seed.o
  make: ./obj/dev/service/service_cmd.o
  make: ./obj/dev/service/service_utils.o
  make: ./obj/dev/service/service.o
  make: ./obj/dev/examples/
  make: ./obj/dev/examples/ini_parser.o
  make: ./obj/dev/examples/curl.o
  make: ./obj/dev/examples/tor.o
  make: ./obj/dev/examples/git.o
  make: ./obj/dev/examples/calc.o
  make: ./obj/dev/seed/
  make: ./obj/dev/seed/seed.o
  make: ./obj/dev/seed/seed_cmd.o
  make: ./obj/dev/leech/
  make: ./obj/dev/leech/leech_cmd.o
  make: ./bin/dev/
  make: ./bin/dev/gittor
```

Parser

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

Config

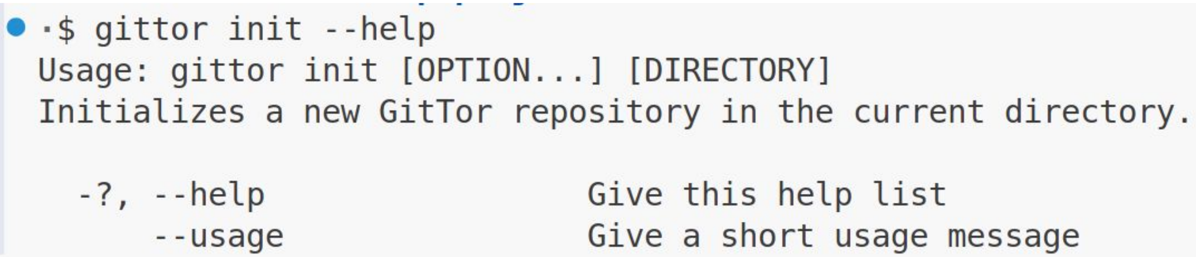
- Manages gittor settings
  - Follows git config pattern for familiarity
  - Allows global and local configurations
  - Example:



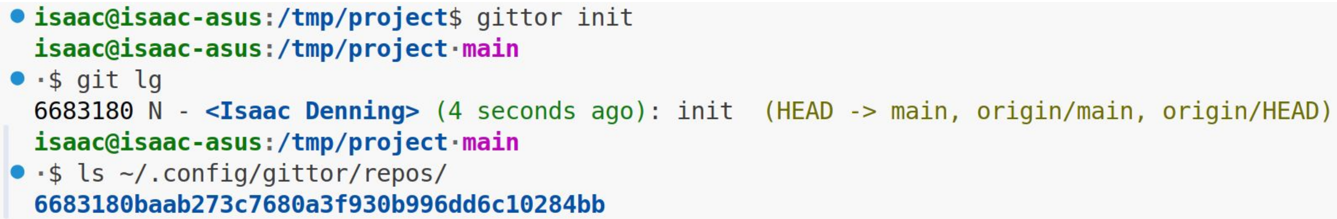
```
[user]
    name = Your Name
    email = your.email@example.com

[network]
    max-upload-rate = 2M
    max-download-rate = 10M
    max-connections = 200
    timeout-seconds = 30

[seeding]
    max-active-seeds = 5

[leeching]
    max-active-downloads = 4
```

Init

- Initializes new Git repository
- Creates an initial commit
- Clones the repository in desired location



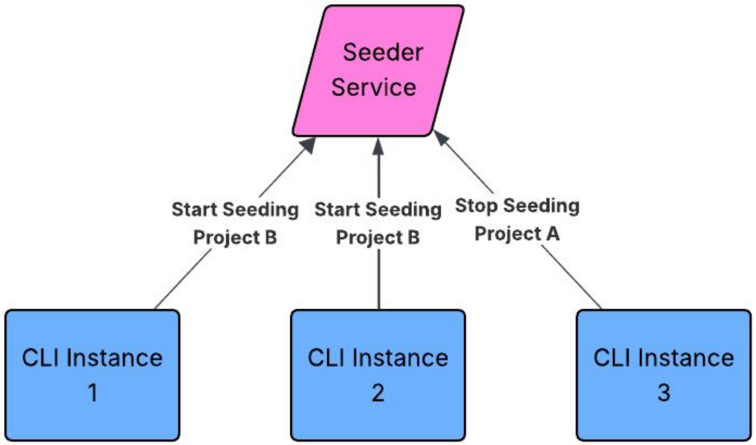
```
·$ gittor init --help
Usage: gittor init [OPTION...] [DIRECTORY]
Initializes a new GitTor repository in the current directory.

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

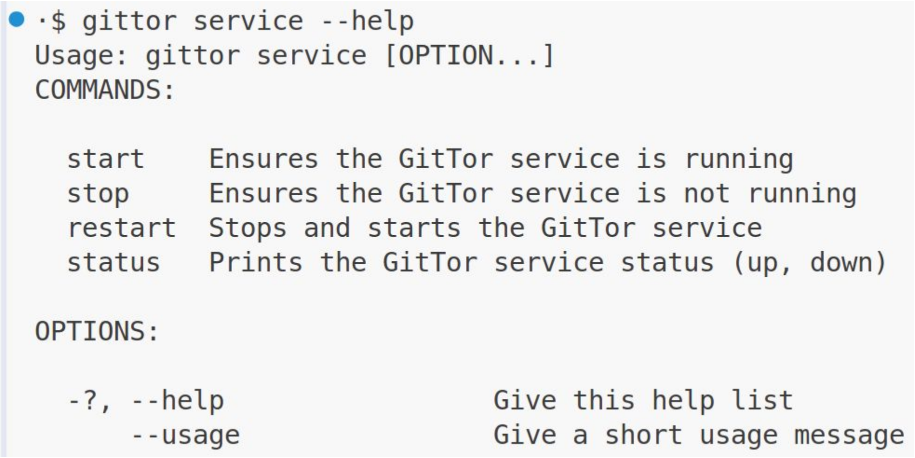


```
isaac@isaac-asus:/tmp/project$ gittor init
isaac@isaac-asus:/tmp/project·main
·$ git lg
6683180 N - <Isaac Denning> (4 seconds ago): init  (HEAD -> main, origin/main, origin/HEAD)
isaac@isaac-asus:/tmp/project·main
·$ ls ~/.config/gittor/repos/
6683180baab273c7680a3f930b996dd6c10284bb
```

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





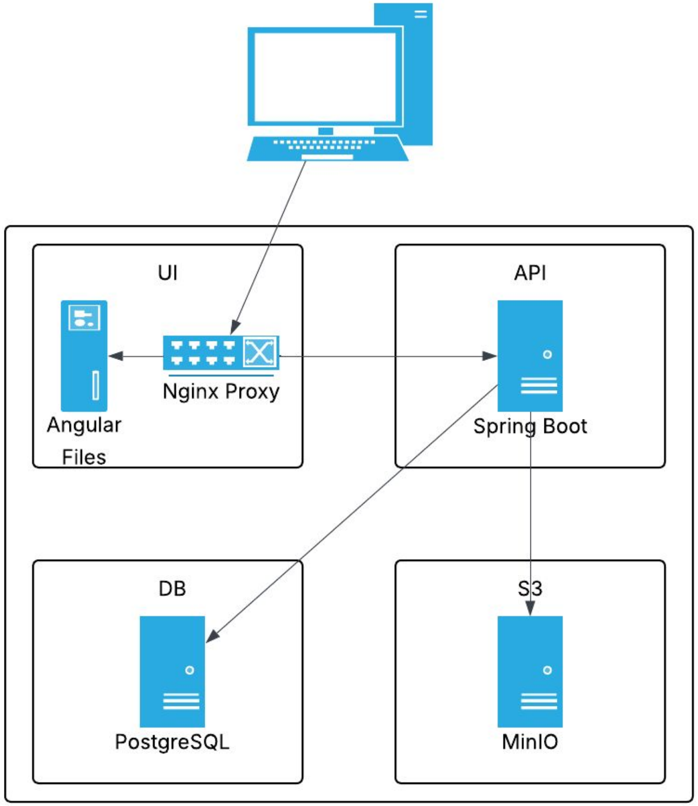
```
● ·$ gittor service --help
  Usage: gittor service [OPTION...]
  COMMANDS:

    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
```

What We’ve Done

Web Application

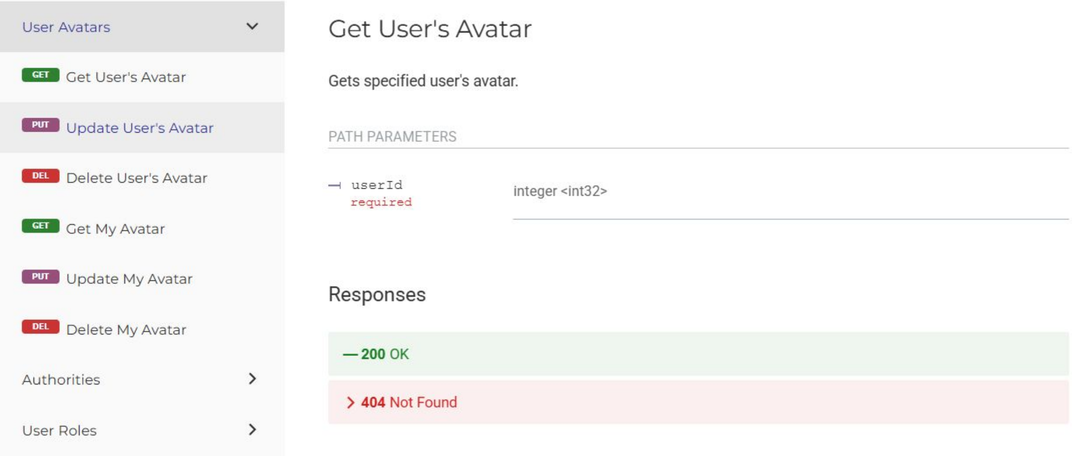
Docker

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



OpenAPI

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



User Avatars

GET Get User's Avatar

PUT Update User's Avatar

DEL Delete User's Avatar

GET Get My Avatar

PUT Update My Avatar

DEL Delete My Avatar

Authorities

User Roles

Get User's Avatar

Gets specified user's avatar.

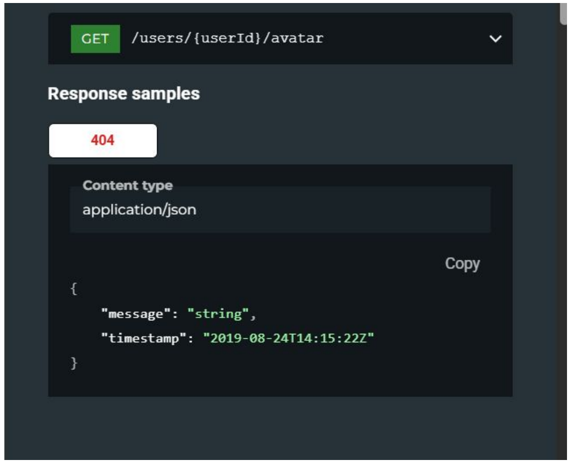
PATH PARAMETERS

userId required	integer <int32>

Responses

— 200 OK

> 404 Not Found



GET /users/{userId}/avatar

Response samples

404

Content type
application/json

Copy

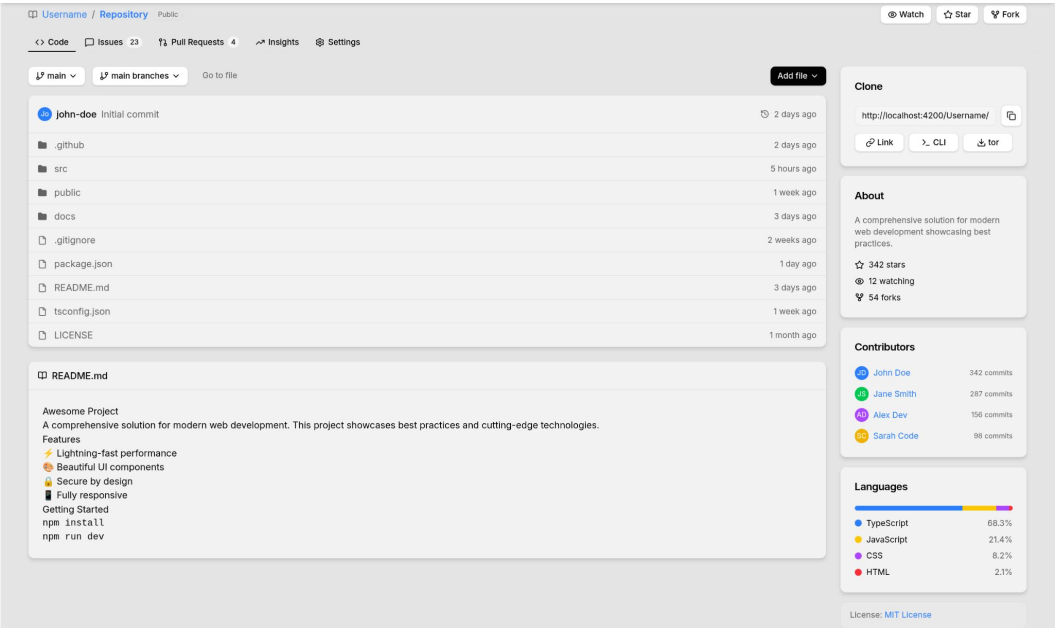
```
{
    "message": "string",
    "timestamp": "2019-08-24T14:15:22Z"
}
```

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

Frontend Design

- → Used Figma to create a rough design
- → Similar to Github’s design yet still distinct
- → Theme switching



john-doe Initial commit 2 days ago

.github	2 days ago
src	5 hours ago
public	1 week ago
docs	3 days ago
.gitignore	2 weeks ago
package.json	1 day ago
README.md	3 days ago
tsconfig.json	1 week ago
LICENSE	1 month ago

README.md

Awesome Project
A comprehensive solution for modern web development. This project showcases best practices and cutting-edge technologies.
Features
⚡ Lightning-fast performance
🎨 Beautiful UI components
🔒 Secure by design
📱 Fully responsive
Getting Started

```
npm install
npm run dev
```

Clone

http://localhost:4200/Username/

About

A comprehensive solution for modern web development showcasing best practices.

☆ 342 stars
12 watching
54 forks

Contributors

JD John Doe	342 commits
JS Jane Smith	287 commits
AD Alex Dev	156 commits
SC Sarah Code	98 commits

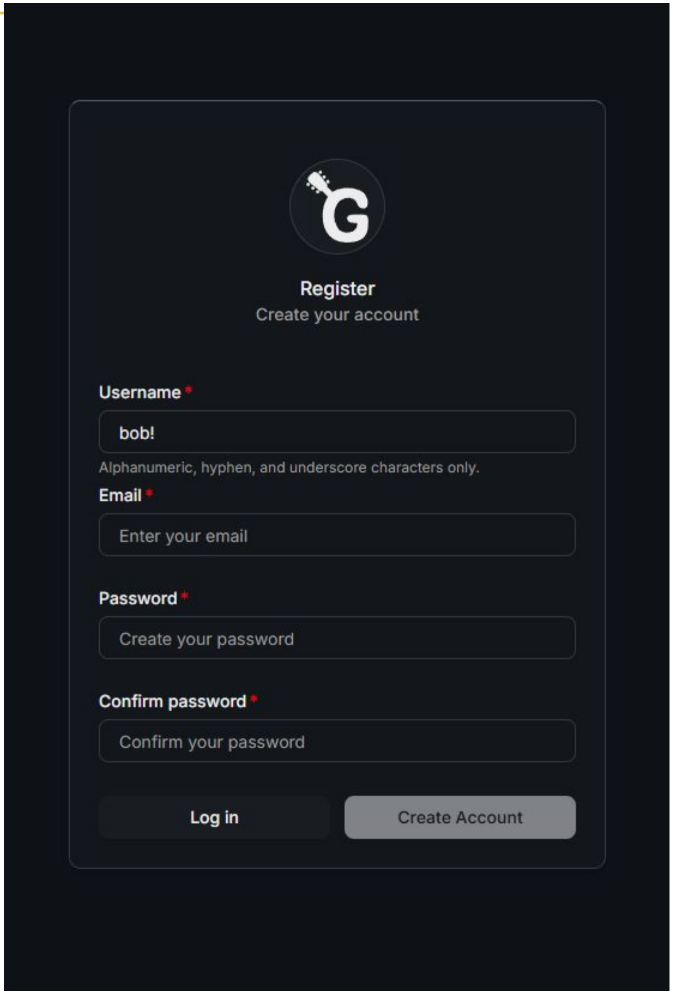
Languages

● TypeScript	68.3%
● JavaScript	21.4%
● CSS	8.2%
● HTML	2.1%

License: MIT License

Login / Registration

- Two token system
  - Access token - Short lived, JWT
  - Refresh token - Long lived, stored in DB
- Security
  - Refresh token stored in HTTP only cookie and revocable
  - Access token always rotating
- Secure approach while keeping UX smooth
- Form validation w/ clear directions for users



Username *

bob!

Alphanumeric, hyphen, and underscore characters only.

Email *

Enter your email

Password *

Create your password

Confirm password *

Confirm your password

What We Will Do

Command-Line Interface

- Seeding / Leeching
- API connection
- Authorization with GPG
  - Manage authorized developers
  - Verify entire repository validity

Web - Backend

Users need to be able to interact with web repositories.

Core endpoints nearly complete:

- ➔ Upload a torrent file & metadata
- ➔ Get metadata or download torrent file w/ id
- ➔ Update metadata or replace torrent file

More capabilities needed:

- ➔ List repositories for a user's account
- ➔ Control repository visibility
- ➔ View repository contents

Web - Frontend

- Repositories
  - Complete logic for repository pages
  - Displaying repositories:
    - Folder structure, text/code files, etc
  - Extra features associated with repos
- Homepage
  - For repository visibility
- Design
  - Complete design for other features
    - Planned: Homepage, Settings, PR, Sidebar

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