{github기반 포트폴리오 제작 서비스}

GitFolio

산학협력캡스톤설계1
gitfolio조
김우진 김해찬 송현섭 심재성

뽑히는 지원서

지원자 #1

- 프론트엔드와 백엔드 개발을 함께 진행하며 데이터 시각화 기능을 구현했습니다.
- 이 프로젝트를 통해 웹 개발 경험을 쌓았습니다.

지원자 #2

- 금융 데이터 처리 속도를 40% 개선하기 위해 비동기 데이터 처리 방식 적용 (Celery + Redis 활용).
- 초기 버전에서는 데이터 시각화가 비효율적이었으나, Chart.js를 활용하여 대시보드 UI를 최적화.
- AI 모델의 예측 정확도가 낮았던 문제를 해결하기 위해 데이터 전처리 방식 개선, 이상치 제거 알고리즘 추가하여 정확도 15% 향상.

뽑히는 지원서

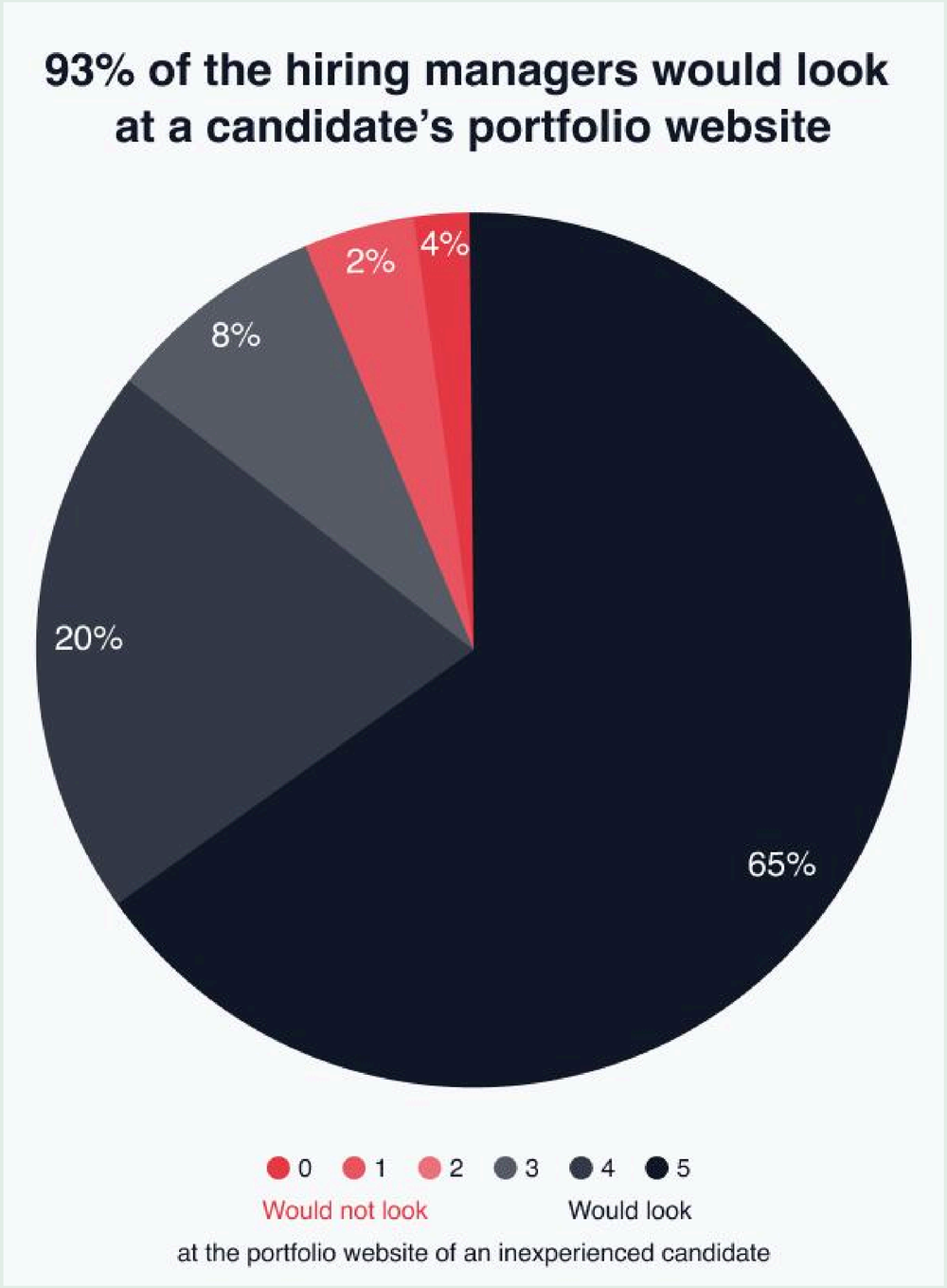
지원자 #1

- 프론트엔드와 백엔드 개발을 함께 진행하며 데이터 시각화 기능을 구현했습니다.
- 이 프로젝트를 통해 웹 개발 경험을 쌓았습니다.

지원자 #2

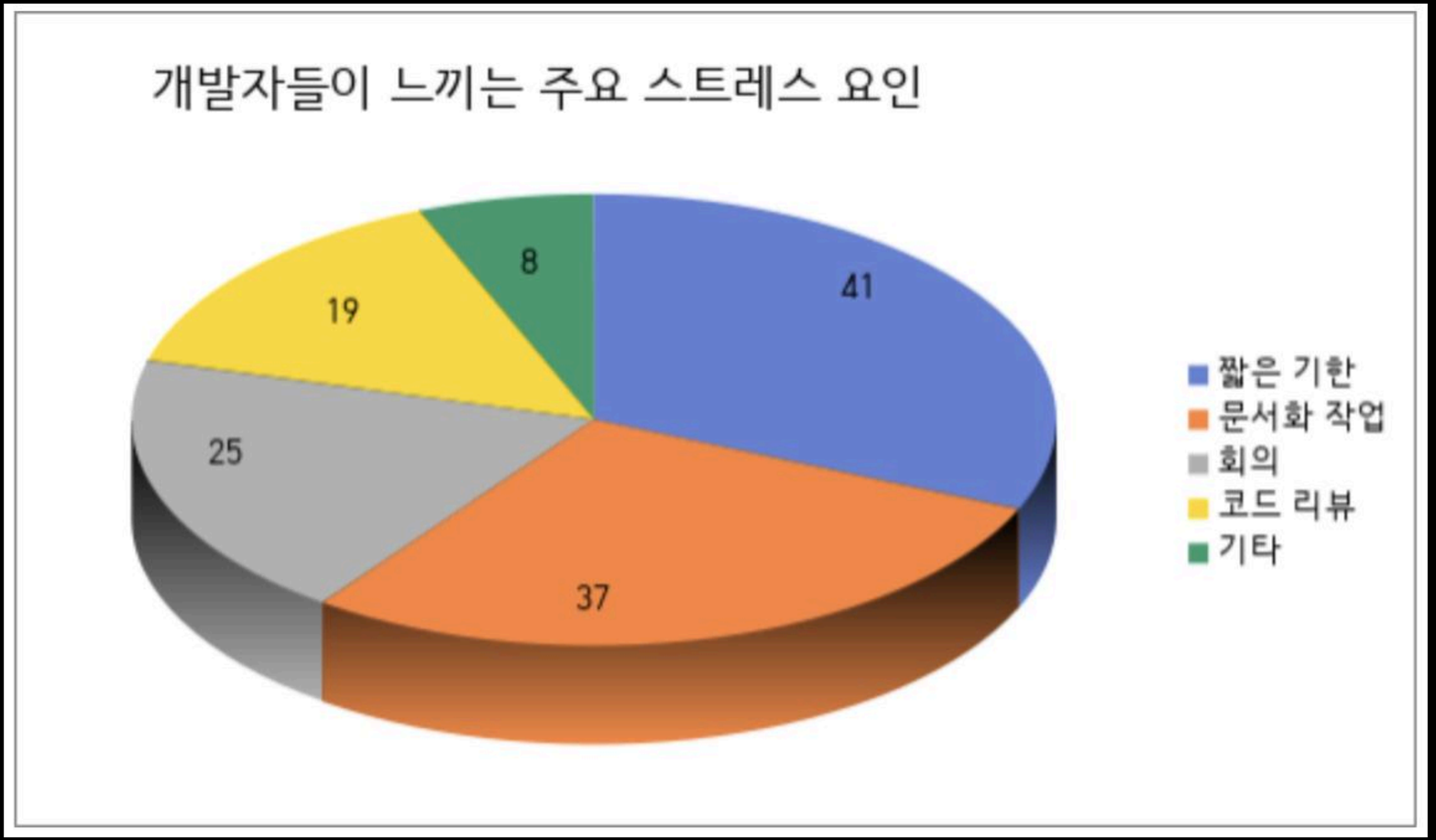
- 금융 데이터 처리 속도를 **40% 개선**하기 위해 비동기 데이터 처리 방식 적용 (Celery + Redis 활용).
- 초기 버전에서는 데이터 시각화가 비효율적이었으나, **Chart.js를 활용**하여 대시보드 UI를 최적화.
- AI 모델의 예측 정확도가 낮았던 **문제를 해결**하기 위해 데이터 전처리 방식 개선, 이상치 제거 알고리즘 추가하여 **정확도 15% 향상.**

포트폴리오 = 자기경험


93% of the hiring managers would look at a candidate's portfolio website
4%
2%
8%
20%
65%
0
1
2
3
4
5
Would not look
Would look
at the portfolio website of an inexperienced candidate

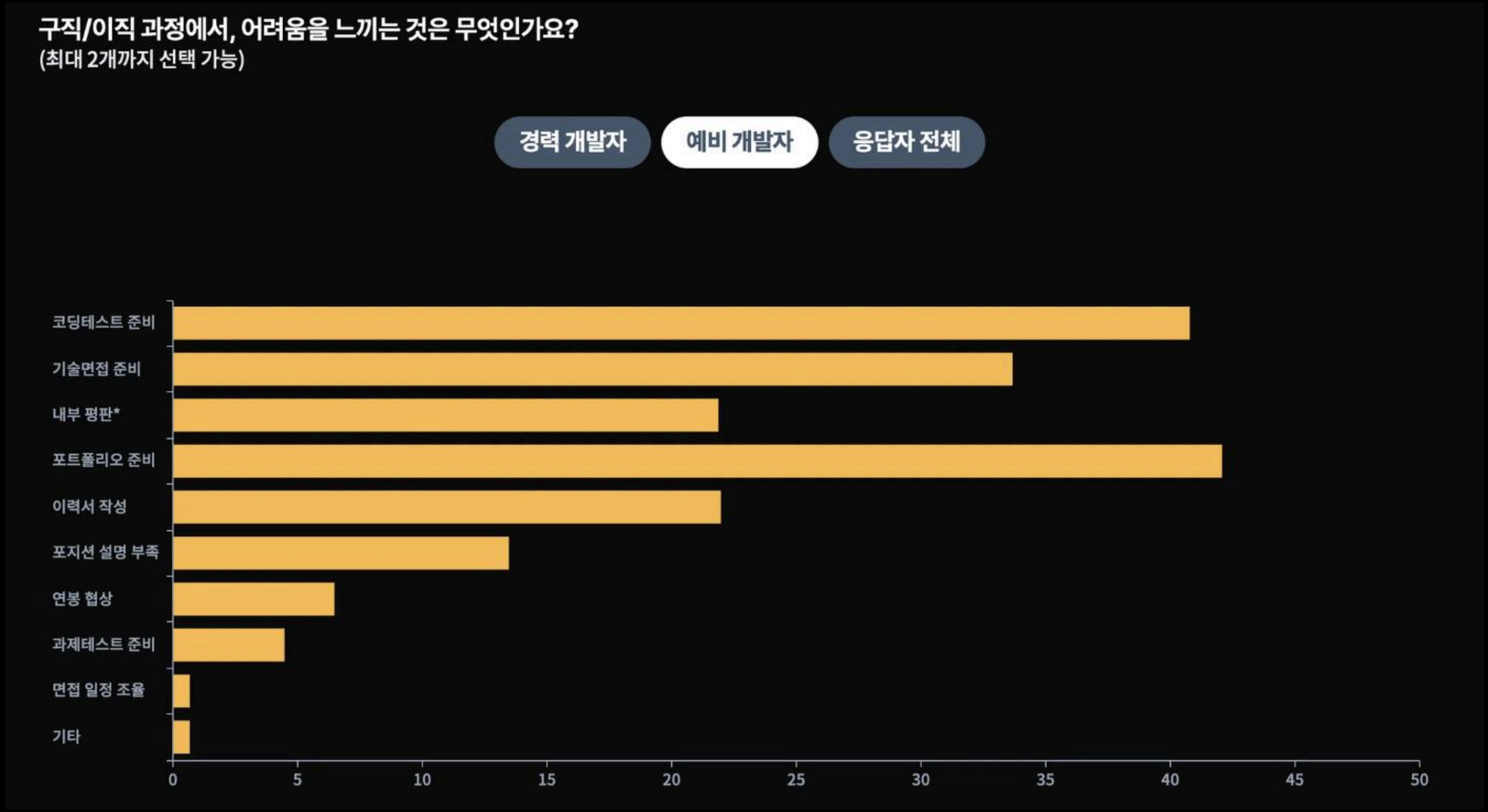
문제점

자기경험 = 어렵다



문제점

자기경험 = 어렵다


구직/이직 과정에서, 어려움을 느끼는 것은 무엇인가요?
(최대 2개까지 선택 가능)
경력 개발자
예비 개발자
응답자 전체
코딩테스트 준비
기술면접 준비
내부 평판*
포트폴리오 준비
이력서 작성
포지션 설명 부족
연봉 협상
과제테스트 준비
면접 일정 조율
기타
0
5
10
15
20
25
30
35
40
45
50

문제점

1

개발 후 시간이 지나면
본인의 **코드 의도를 파악**하기 힘듦

2

코드 문서화가 충분히 진행되지 않음

아이디어

깃허브
커밋 데이터

내가 작성한 **코드** + **고민**한 흔적

내가 기여한
내용을 분석

정량적인 **데이터가 포함된 텍스트**로 제공

“내 코드를 분석해주고 정리해주는 구글 확장프로그램”

GitFoliio

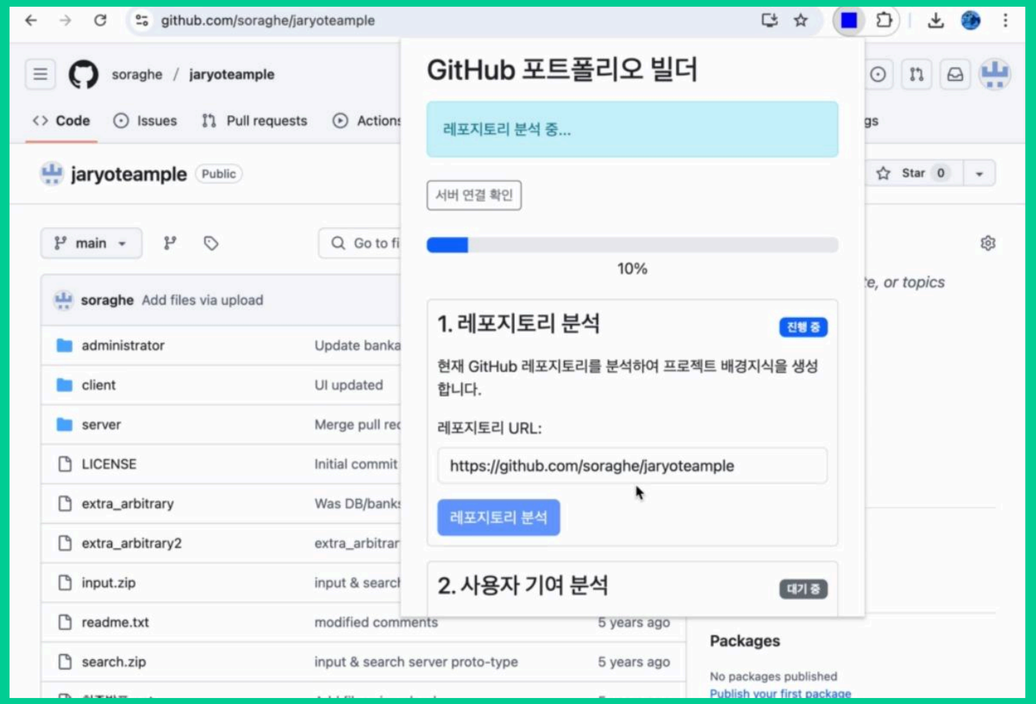
“내 경험을 분석해주고 정리 해주는 구글 확장프로그램”

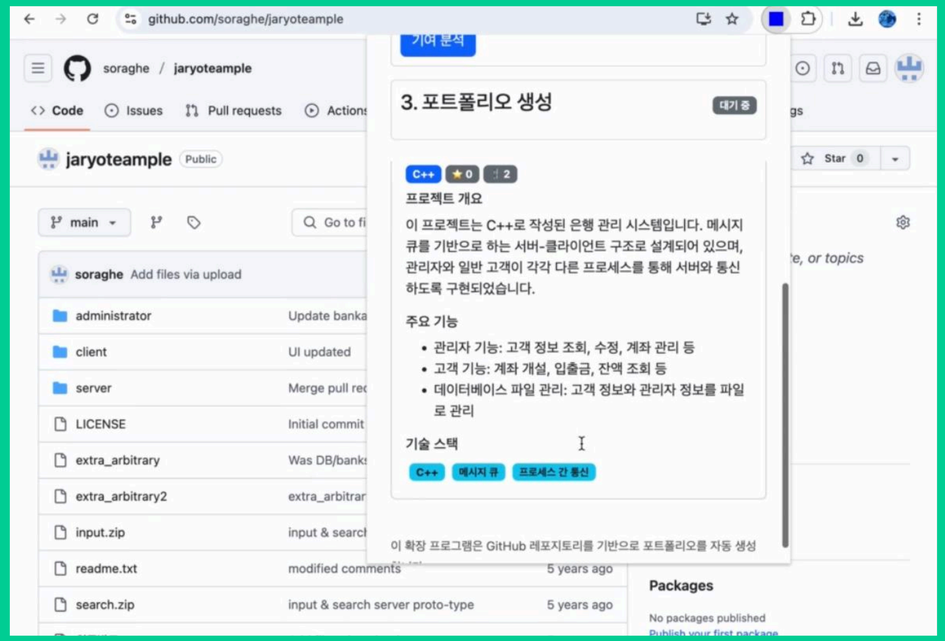
GitFoliio

서비스 소개

1. 레포지토리 보고서

사용자 선택한 레포지토리를 분석하여
프로젝트 개요에 대한 보고서를 작성

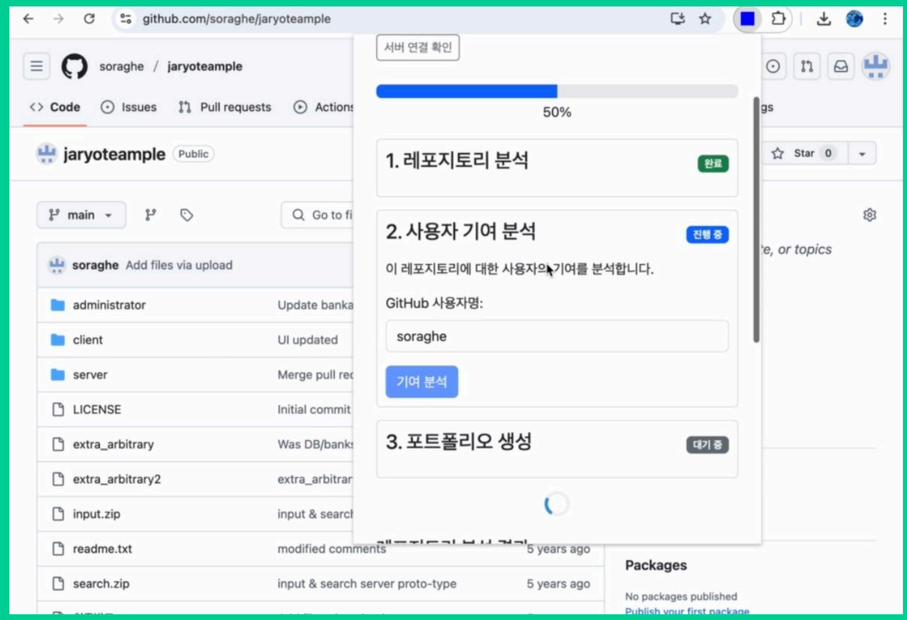


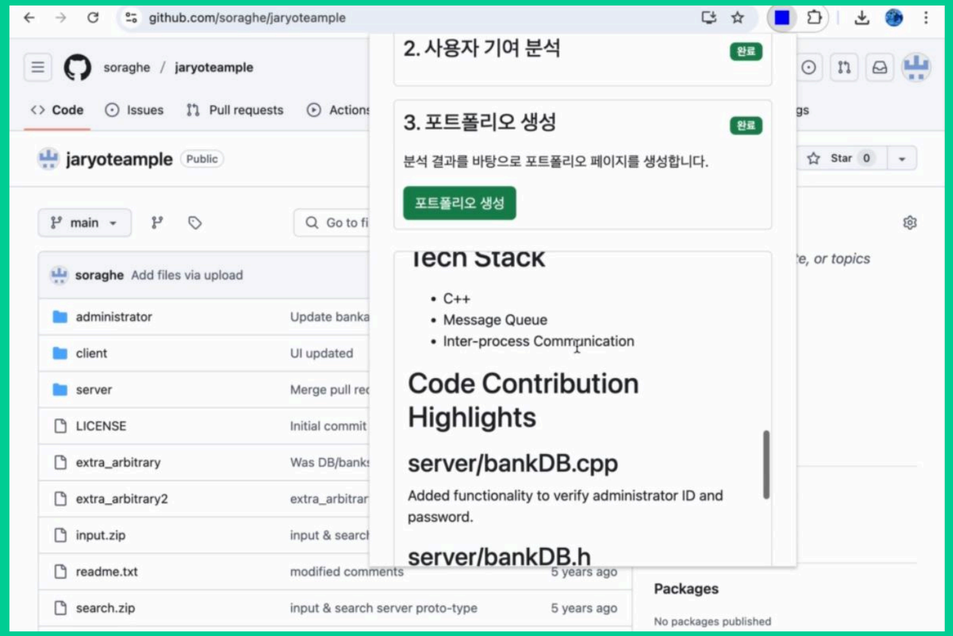


서비스 소개

2. 포트폴리오 작성

사용자의 커밋 데이터를 분석하여
기여 내용을 바탕으로 포트폴리오 작성

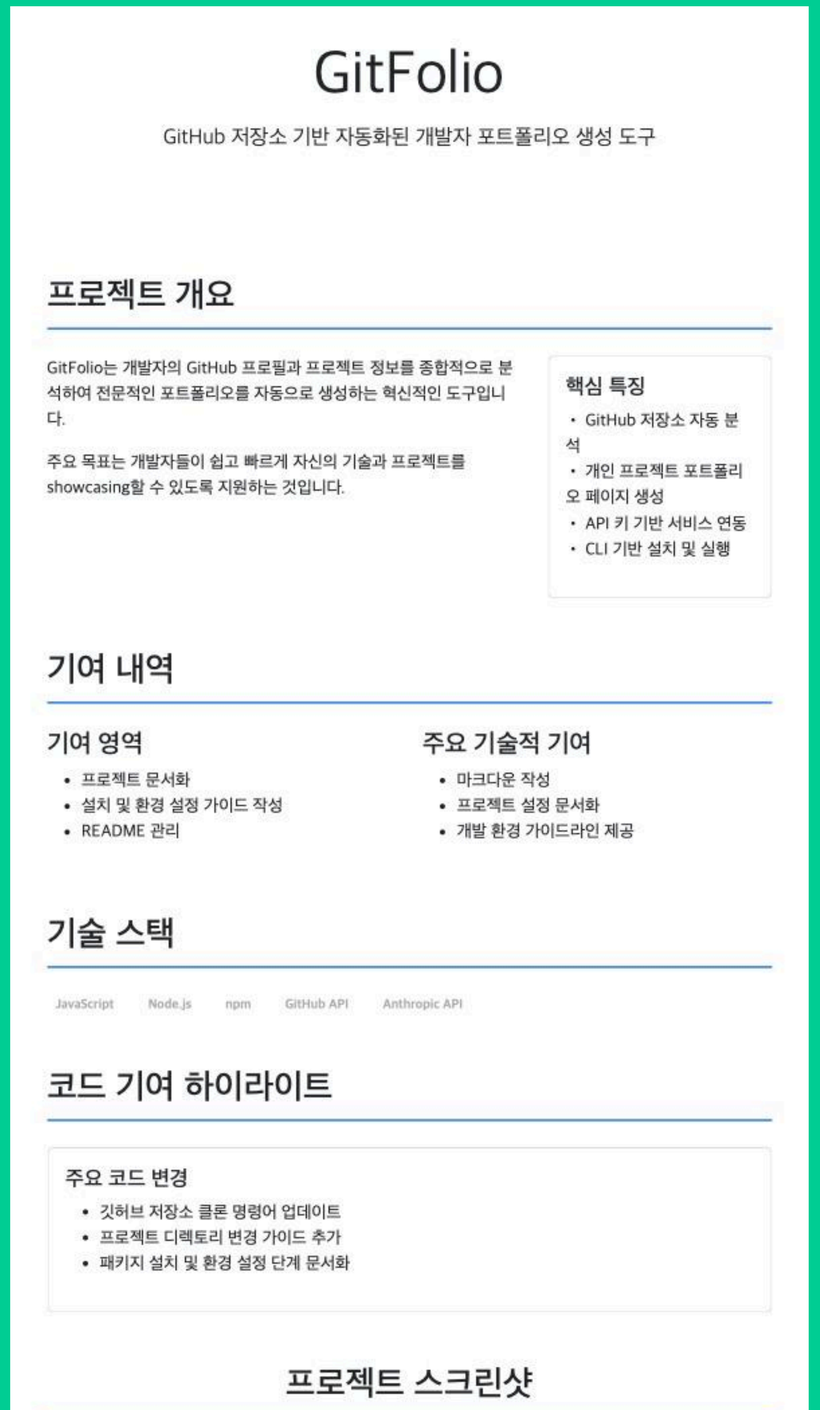




서비스 소개

2. 포트폴리오 작성

사용자의 커밋 데이터를 분석하여
기여 내용을 바탕으로 포트폴리오 작성



관련 연구

1. GitCommitsAnalysis

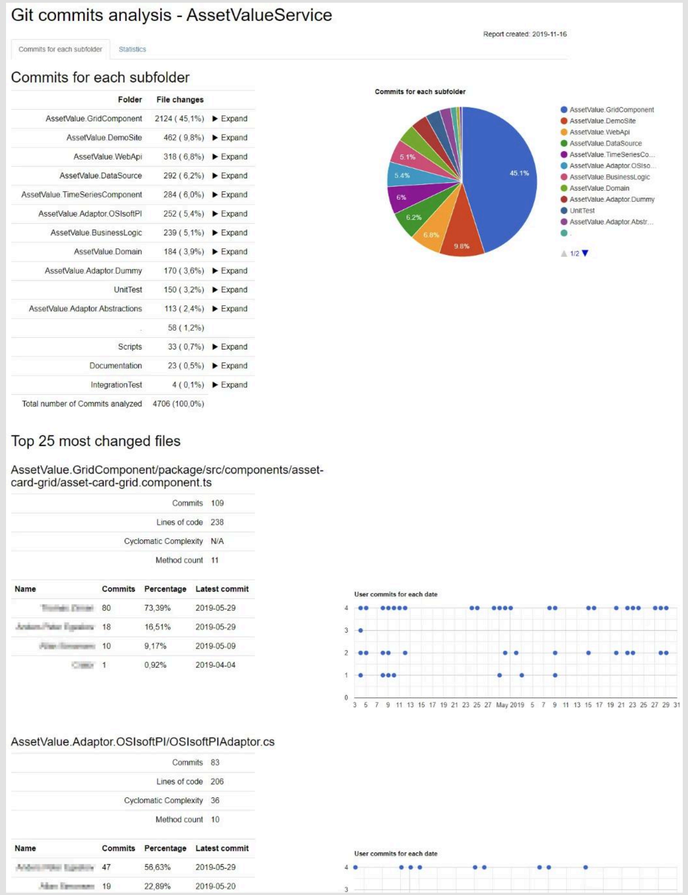
커밋 기록을 분석하여 코드 파일의 길이, 복잡도, 변경 빈도를 기반으로 리팩토링이 필요한 파일을 식별

2. 시각화를 통한 프로젝트 통찰 연구

GitHub 이슈와 커밋 데이터를 시각화하여 문제 해결 시간, 버그 발생 빈도가 높은 컴포넌트, 개발자 기여 패턴 등을 분석하는 도구 개발

3. Git History Analyzer and Code Quality Predictor

머신러닝을 이용해 커밋 패턴과 코드 변경 이력을 분석하고, 잠재적인 버그 핫스팟이나 리팩토링 필요성을 예측



Git commits analysis - AssetValueService

Report created: 2019-11-16

Commits for each subfolder | Statistics

Commits for each subfolder

Folder	File changes	
AssetValue.GridComponent	2124 (45,1%)	▶ Expand
AssetValue.DemoSite	462 (9,8%)	▶ Expand
AssetValue.WebApi	318 (6,8%)	▶ Expand
AssetValue.DataSource	292 (6,2%)	▶ Expand
AssetValue.TimeSeriesComponent	284 (6,0%)	▶ Expand
AssetValue.Adaptor.OSIsoftPI	252 (5,4%)	▶ Expand
AssetValue.BusinessLogic	239 (5,1%)	▶ Expand
AssetValue.Domain	184 (3,9%)	▶ Expand
AssetValue.Adaptor.Dummy	170 (3,6%)	▶ Expand
UnitTest	150 (3,2%)	▶ Expand
AssetValue.Adaptor.Abstractions	113 (2,4%)	▶ Expand
.	58 (1,2%)	
Scripts	33 (0,7%)	▶ Expand
Documentation	23 (0,5%)	▶ Expand
IntegrationTest	4 (0,1%)	▶ Expand
Total number of Commits analyzed	4706 (100,0%)	

Commits for each subfolder

Top 25 most changed files

AssetValue.GridComponent/package/src/components/asset-card-grid/asset-card-grid.component.ts

Commits	109
Lines of code	238
Cyclomatic Complexity	N/A
Method count	11

Name	Commits	Percentage	Latest commit
[redacted]	80	73,39%	2019-05-29
[redacted]	18	16,51%	2019-05-29
[redacted]	10	9,17%	2019-05-09
[redacted]	1	0,92%	2019-04-04

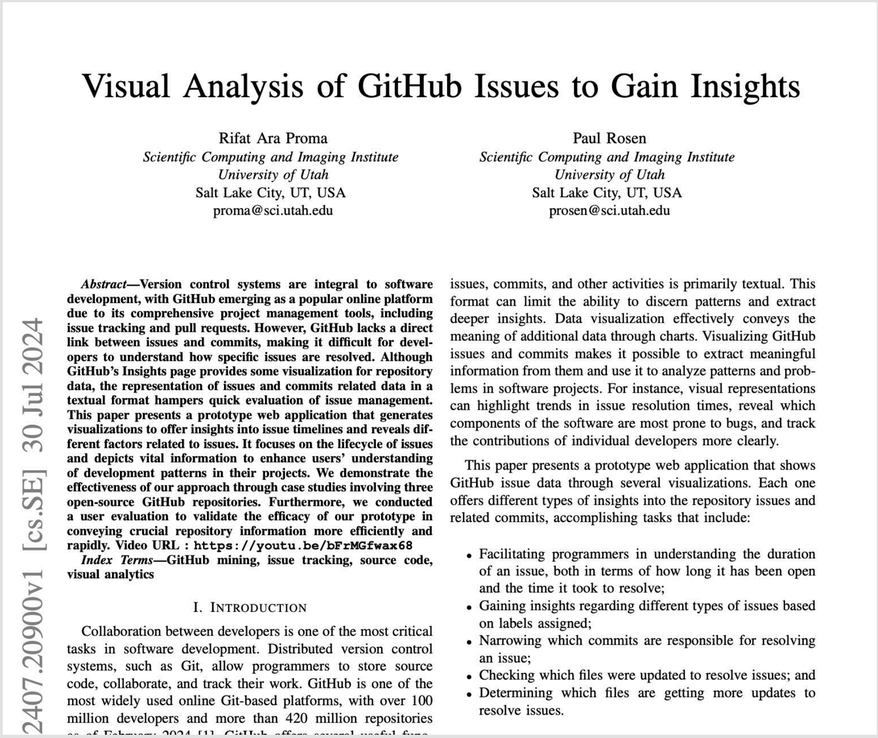
User commits for each date

AssetValue.Adaptor.OSIsoftPI/OSIsoftPIAdaptor.cs

Commits	83
Lines of code	206
Cyclomatic Complexity	36
Method count	10

Name	Commits	Percentage	Latest commit
[redacted]	47	56,63%	2019-05-29
[redacted]	19	22,89%	2019-05-20

User commits for each date



Visual Analysis of GitHub Issues to Gain Insights

Rifat Ara Proma
Scientific Computing and Imaging Institute
University of Utah
Salt Lake City, UT, USA
proma@sci.utah.edu

Paul Rosen
Scientific Computing and Imaging Institute
University of Utah
Salt Lake City, UT, USA
prosen@sci.utah.edu

***Abstract*—Version control systems are integral to software development, with GitHub emerging as a popular online platform due to its comprehensive project management tools, including issue tracking and pull requests. However, GitHub lacks a direct link between issues and commits, making it difficult for developers to understand how specific issues are resolved. Although GitHub's Insights page provides some visualization for repository data, the representation of issues and commits related data in a textual format hampers quick evaluation of issue management. This paper presents a prototype web application that generates visualizations to offer insights into issue timelines and reveals different factors related to issues. It focuses on the lifecycle of issues and depicts vital information to enhance users' understanding of development patterns in their projects. We demonstrate the effectiveness of our approach through case studies involving three open-source GitHub repositories. Furthermore, we conducted a user evaluation to validate the efficacy of our prototype in conveying crucial repository information more efficiently and rapidly. Video URL : `https://youtu.be/bFrMGfwax68`**

***Index Terms*—GitHub mining, issue tracking, source code, visual analytics**

I. Introduction

Collaboration between developers is one of the most critical tasks in software development. Distributed version control systems, such as Git, allow programmers to store source code, collaborate, and track their work. GitHub is one of the most widely used online Git-based platforms, with over 100 million developers and more than 420 million repositories as of February 2024 [1]. GitHub offers several useful func-

issues, commits, and other activities is primarily textual. This format can limit the ability to discern patterns and extract deeper insights. Data visualization effectively conveys the meaning of additional data through charts. Visualizing GitHub issues and commits makes it possible to extract meaningful information from them and use it to analyze patterns and problems in software projects. For instance, visual representations can highlight trends in issue resolution times, reveal which components of the software are most prone to bugs, and track the contributions of individual developers more clearly.

This paper presents a prototype web application that shows GitHub issue data through several visualizations. Each one offers different types of insights into the repository issues and related commits, accomplishing tasks that include:

- Facilitating programmers in understanding the duration of an issue, both in terms of how long it has been open and the time it took to resolve;
- Gaining insights regarding different types of issues based on labels assigned;
- Narrowing which commits are responsible for resolving an issue;
- Checking which files were updated to resolve issues; and
- Determining which files are getting more updates to resolve issues.

역할분담

김우진

프로젝트 매니저 & 백엔드 개발 담당

송현섭

코드 분석 및 정적 분석 담당

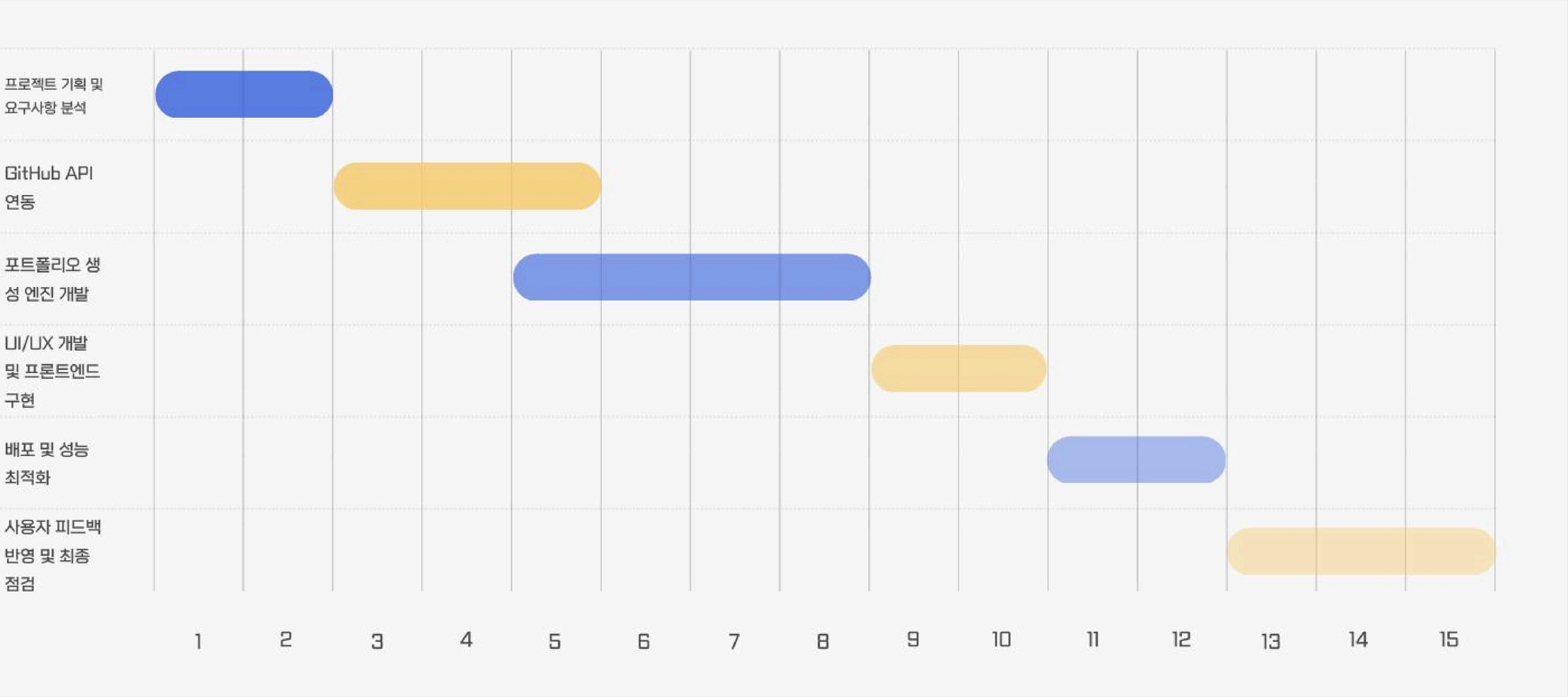
김해찬

AI 분석 및 포트폴리오 템플릿 담당

심재성

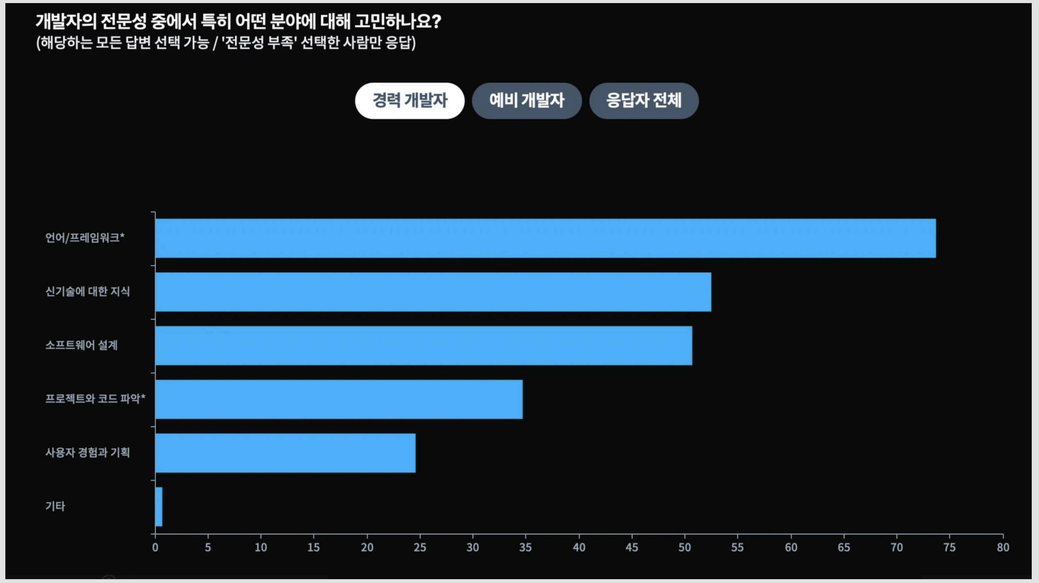
UI/UX, 통합 및 CI/CD 담당

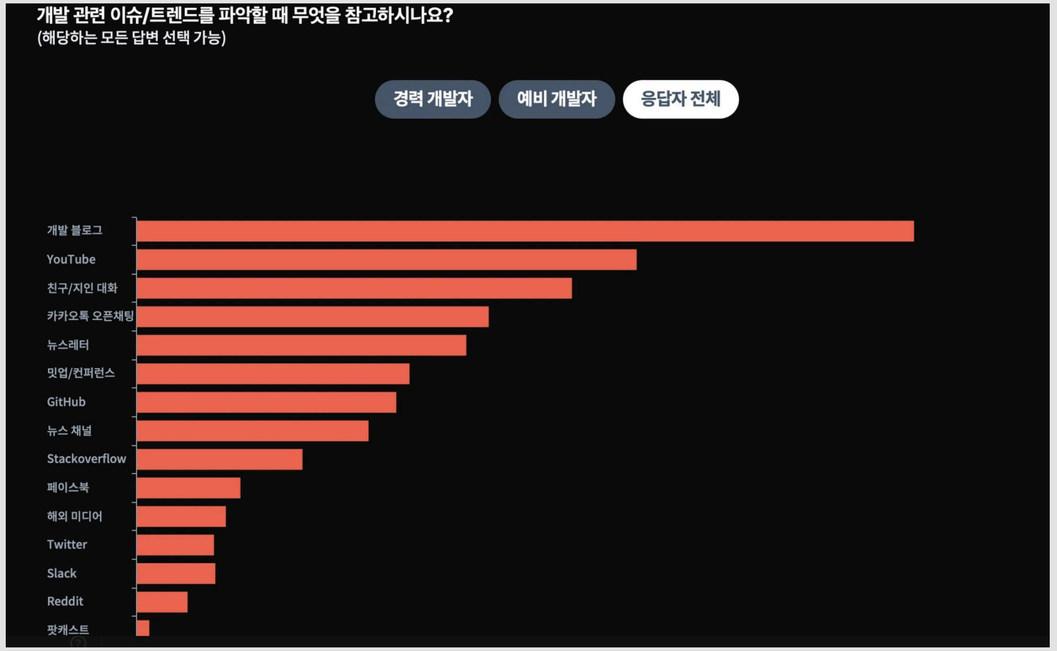
개발 일정


프로젝트 기획 및 요구사항 분석
GitHub API 연동
포트폴리오 생성 엔진 개발
UI/UX 개발 및 프론트엔드 구현
배포 및 성능 최적화
사용자 피드백 반영 및 최종 점검
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

기대효과

포트폴리오
→ 문서화 분야로 확장

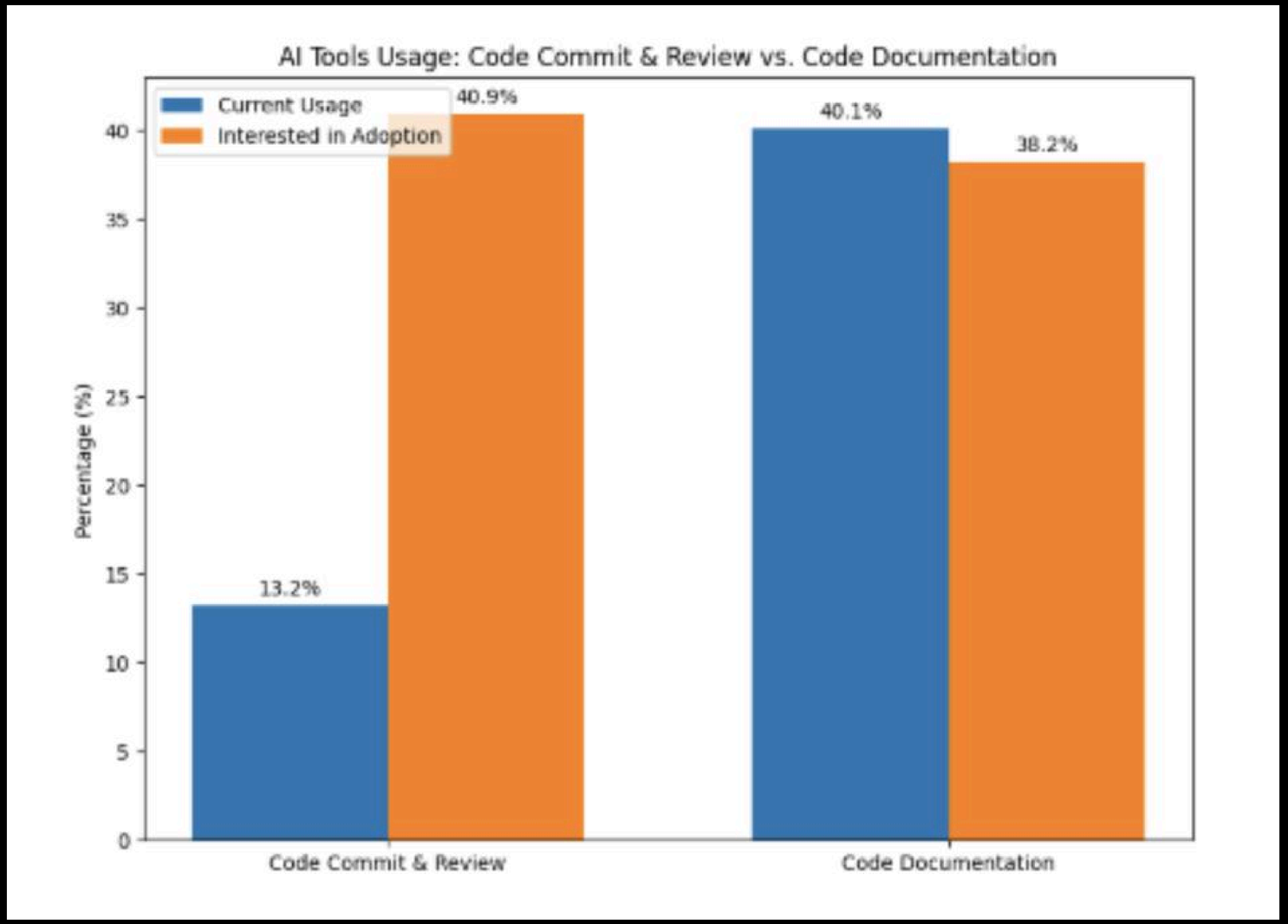




감사합니다

github기반 포트폴리오 제작 서비스
gitfolio조

appendix


AI Tools Usage: Code Commit & Review vs. Code Documentation
Current Usage
Interested in Adoption
Percentage (%)
0
5
10
15
20
25
30
35
40
13.2%
40.9%
40.1%
38.2%
Code Commit & Review
Code Documentation

appendix

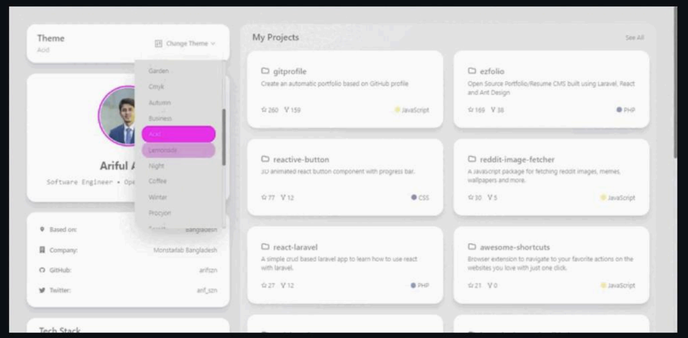
GitProfile

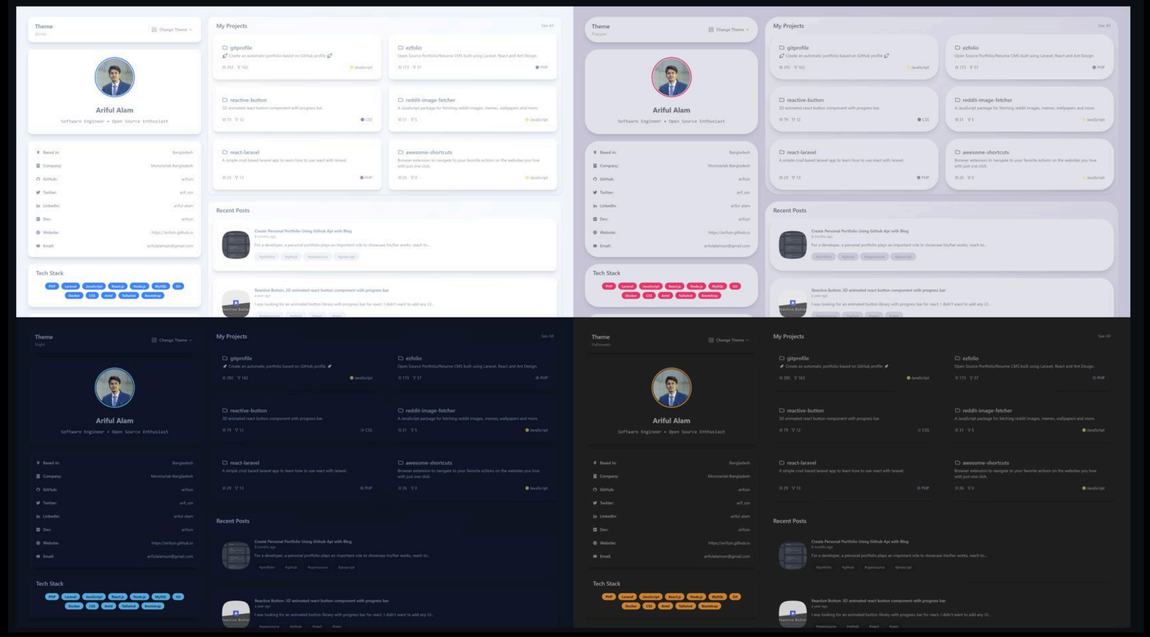
Easy to use automatic portfolio builder for every GitHub user!

maintainability ? | Test Deployment passing | issues 0 open | Stars 1.8k | Forks 1.8k | last commit february

contributions welcome | license MIT | Open in IDX

View Demo · Report Bug · Request Feature





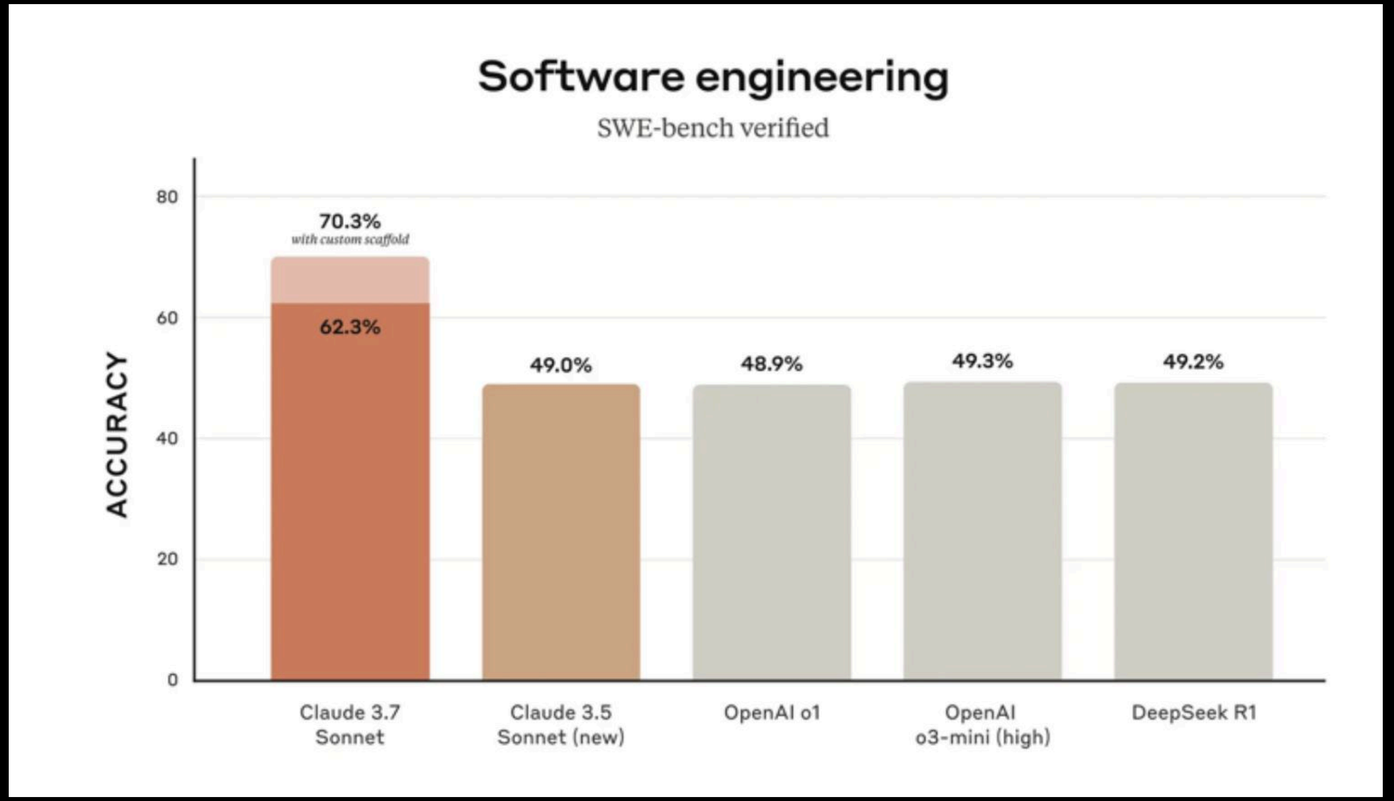
appendix

GPT-4 Technical Report

OpenAI*

Abstract

We report the development of GPT-4, a large-scale, multimodal model which can accept image and text inputs and produce text outputs. While less capable than humans in many real-world scenarios, GPT-4 exhibits human-level performance on various professional and academic benchmarks, including passing a simulated bar exam with a score around the top 10% of test takers. GPT-4 is a Transformer-based model pre-trained to predict the next token in a document. The post-training alignment process results in improved performance on measures of factuality and adherence to desired behavior. A core component of this project was developing infrastructure and optimization methods that behave predictably across a wide range of scales. This allowed us to accurately predict some aspects of GPT-4's performance based on models trained with no more than 1/1,000th the compute of GPT-4.



appendix

1. 리포지토리 관련 API

🔷 리포지토리 정보 조회
- GET /repos/{owner}/{repo} → 특정 리포지토리의 정보 조회
- GET /users/{username}/repos → 특정 사용자의 공개 리포지토리 목록 조회
- POST /user/repos → 새 리포지토리 생성
- PATCH /repos/{owner}/{repo} → 리포지토리 설정 변경
- DELETE /repos/{owner}/{repo} → 리포지토리 삭제

2. 커밋 및 코드 관련 API

🔷 커밋 데이터 가져오기
- GET /repos/{owner}/{repo}/commits → 특정 리포지토리의 커밋 목록 조회
- GET /repos/{owner}/{repo}/commits/{sha} → 특정 커밋의 상세 정보 조회

🔷 파일 및 브랜치 정보 조회
- GET /repos/{owner}/{repo}/branches → 브랜치 목록 조회
- GET /repos/{owner}/{repo}/contents/{path} → 특정 파일/디렉토리의 내용 가져오기
- PUT /repos/{owner}/{repo}/contents/{path} → 파일 생성 또는 업데이트

appendix

3. 이슈 및 PR 관련 API

🔹 이슈 조회 및 관리
- GET /repos/{owner}/{repo}/issues → 특정 리포지토리의 이슈 목록 조회
- POST /repos/{owner}/{repo}/issues → 새 이슈 생성
- PATCH /repos/{owner}/{repo}/issues/{issue_number} → 이슈 상태 업데이트
- GET /repos/{owner}/{repo}/issues/{issue_number}/comments → 이슈의 댓글 목록 조회

🔹 Pull Request(PR) 조회 및 관리
- GET /repos/{owner}/{repo}/pulls → PR 목록 조회
- GET /repos/{owner}/{repo}/pulls/{pull_number} → 특정 PR의 상세 정보 조회
- POST /repos/{owner}/{repo}/pulls → 새 PR 생성
- PUT /repos/{owner}/{repo}/pulls/{pull_number}/merge → PR 병합

4. 사용자 및 조직 관련 API

🔹 사용자 정보 조회
- GET /users/{username} → 특정 사용자 정보 조회
- GET /user → 인증된 사용자 정보 조회

🔹 조직 및 팀 관리
- GET /orgs/{org} → 특정 조직 정보 조회
- GET /orgs/{org}/teams → 조직 내 팀 목록 조회