**우리 “시각화” 팀은!?**

어떤 데이터들을 보기 좋게 표현하여 사람에게 정보를 쉽고, 정확하게 전달하는 것이 목표예요.

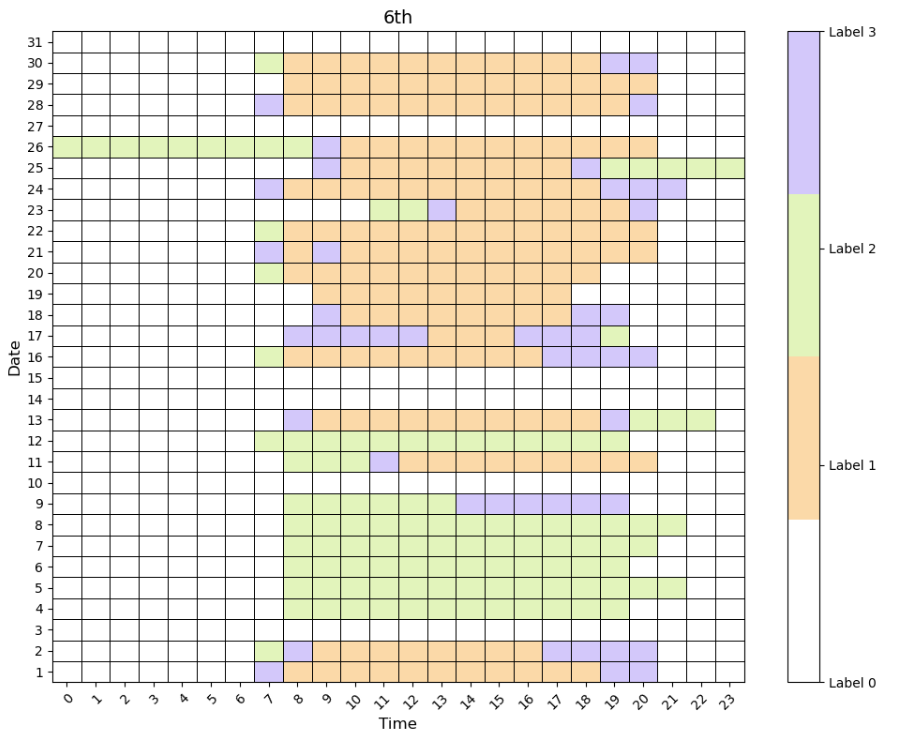
우리의 데이터 표현 방법, 협업 방식에 대해 알아보아요.

우리 팀은 “파스텔 색”을 사용하는 것이 원칙이에요.

“파스텔 색”은 톤이 연하고, 채도가 낮기 때문에 밝고, 부드러운 효과를 낼 수 있어요.

“원색”은 톤이 진하고, 채도가 높기 때문에 뚜렷하고, 강렬한 효과를 낼 수 있어서 정보 전달을 할 때 “원색”을 이용하는 것이 뚜렷하다고 생각할 수 있어요. 하지만 아래 그림을 보면 알 수 있는데, 뚜렷하고 강렬하다 해서 정보 전달이 잘 되는 것은 절대 아닙니다.

오히려 데이터를 여러 가지로 라벨링해야 한다면 부드럽게 이어지는 “파스텔 색”을 이용하는 것이 시각적으로 더 좋은 효과를 가져올 수 있습니다.

도표, 스크린샷, 라인, 다채로움이(가) 표시된 사진

자동 생성된 설명다만 “파스텔 색”은 비슷한 색을 이용할 경우 분류의 효과가 너무 떨어질 수도 있으니 주의하세요.

**코드 스타일 가이드**

1. 들여쓰기와 공백:

- 들여쓰기: 4개의 스페이스 사용

- 연산자 주변에 공백 추가: x = y + z

2. 변수와 함수명:

- 의미 전달: user\_name, calculate\_total\_amount()

- 카멜 케이스 또는 밑줄 사용

3. 주석:

- 코드 의도와 기능 설명

- 복잡한 부분이나 중요한 정보 주석으로 기록

4. 함수와 클래스:

- 한 가지 기능만 수행하는 함수

- 클래스는 한 가지 책임 가지기

5. 문자열 리터럴:

- 작은 따옴표(') 또는 큰 따옴표(") 사용

- 같은 프로젝트 내에서 동일한 스타일 유지

6. 예외 처리:

- 각 예외 상황에 맞는 에러 메시지 제공

- 필요한 곳에서 예외 처리 사용

팀 내에서 이러한 가이드라인을 공유하고 따르면서 코드 품질을 유지하고 협업을 원활하게 진행하세요. 주기적으로 가이드라인을 검토하며 프로젝트의 요구사항과 팀의 선호도에 맞게 조정할 수 있습니다.

**GITHUB(협업) 가이드**

1. 레포지토리 생성:

- 새로운 레포지토리 생성 시, 명확하고 간결한 이름 선택

- README 파일에 프로젝트 개요와 사용 방법 작성

2. 브랜치 관리:

- 기본 브랜치는 'main' 로 설정

- 기능별 브랜치 생성: feat/기능명, bugfix/버그명 등

- Pull Request를 통해 기능별 브랜치를 메인 브랜치로 병합

3. 커밋과 푸시:

- 커밋 메시지는 명확하게 작성 (예: "기능 추가: 사용자 등록 기능 구현")

- 커밋 시 자동 생성되는 코드 포맷팅 등 훅을 활용하여 일관성 유지

- 작업 중간 커밋을 자주 하여 작업 이력 관리

4. Pull Request (PR):

- PR을 통해 기능 또는 수정 사항을 메인 브랜치로 통합

- PR 제목과 설명을 상세히 작성하여 리뷰어에게 명확한 정보 제공

5. 코드 리뷰:

- PR에 리뷰어를 지정하고, 코드 변경 사항을 검토하도록 요청

- 리뷰어가 주석을 달고 피드백을 주면, 개선사항 반영 후 다시 커밋하여 반영

6. 머지와 충돌 해결:

- PR에 충돌이 발생할 경우, 충돌 지점을 해결하고 다시 커밋하여 PR에 반영

- 리뷰어와 함께 충돌을 해결하고, 최종적으로 머지

7. 프로젝트 문서화:

- README 파일을 업데이트하여 프로젝트 개요, 설치 및 사용 방법 제공

- 필요한 경우 Wiki를 활용하여 추가 문서 작성

8. 자동화:

- CI/CD 파이프라인을 설정하여 자동 빌드, 테스트, 배포 자동화

GitHub을 효과적으로 활용하여 협업을 관리하면서 프로젝트를 개발하고 관리하세요. 팀원들과의 소통과 효율적인 협업을 위해 GitHub 기능을 최대한 활용하세요.