# 이미지 필터 조정 및 특징을 나타내는 웹 사이트 개발

조규철*, 한용준O
*인하공업전문대학 컴퓨터정보과,
O인하공업전문대학 컴퓨터정보과
e-mail: kccho@inhatc.ac.kr*, gksdydwns34@naver.comO

# Development Web Application for Image Filters and Features Information

Cho Kyu Cheol*, Han Yong JunO
*Dept. of Computer Science, Inha Technical College,
ODept. of Computer Science, Inha Technical College

● 요 약 ●

본 논문에서는 '그림'이라는 주제를 통해 자기 작품을 그림에 관심이 많은 사람과 자유롭게 소통하고 작품성을 높이기 위해 이미지 필터 조정 및 특징을 표현하는 웹사이트를 제작하였다. '그림'을 통해 대중들에게 자기 작품을 보여 줄 수 있는 활동은 '전시회' 이외에는 없기 때문에 이 웹사이트에 게시판 기능을 통해 자기 작품을 더 쉽게 공개함과 동시에 google cloud vision api를 활용해 자기 작품에 특징을 그래프로 시각적인 효과를 이용해 추출하고 필터를 변경하여 좀 더 작품에 대한 완성도를 더해주며, 본 웹사이트를 통해 그림에 대한 문화 활동이 다른 문화 활동에 비해 좀 더 앞서갈 것이라고 기대한다.

**키워드:** 그림(art), Google Cloud Vision API, Filter

## I. 서론

최근 인터넷과 디지털 기술의 발전으로 인해 사람들은 더 많은 시각적 콘텐츠에 노출되고 있다. 그림 작품에 대한 시각적 요소들은 사람들의 감정을 자극하고 정보를 전달하는 중요한 매체로 사용되고 있다. 그래서 그림을 주제로 만든 웹사이트를 활용하여 자신이 직접 만든 작품을 공유할 수 있는 시스템이 필요하다.

개인과 커뮤니티 간의 연결성이 더욱 중요하다고 생각하고 있으므로 그림 공유와 소통을 위한 웹사이트는 사회적 필요성에 부합하는 해결책으로 주목받고 있다. 자기 창작물을 공유하고, 다른 이들과 의견을 나누면서 창작에 대한 영감과 피드백을 얻고 사진 특징 추출과 같은 기능을 통해 사용자들이 그림에 담긴 의미와 특징을 더욱 깊이 있게 탐구할 수 있도록 도와주는 것이 연구의 목적이다.

지난해 우리나라 미술시장이 코로나19와 세계 경제 위축에도 불구하고 역대 최초로 미술품 유통액 1조 377억 원을 달성했다.


Fig. 1. 2022년 국내 미술시장 규모 추이

문화체육관광부는 그림1과 같이 '2022년 미술시장 규모 추산 결과'를 발표하면서 2021년 7563억 원 대비 37.2% 성장했다고 밝혔다[1]. 그림 1은 경매와 아트 페어의 유통액과 화랑 매출액의 예측치를 합산한 것이며, 분야별로 아트 페어 매출액이 59.8% 성장했고 화랑의 판매액도 59.8%를 증가하지만, 경매를 통한 판매액은 30.9% 감소했다는 걸 볼 수 있다.

## II. 개발환경 및 구조

프로젝트 개발환경은 다음과 같다.

Tomcat 서버를 기반으로 HTML, CSS, JAVA SCRIPT, JSP와 JAVA Spring Framework를 이용한 Web Back-End를 개발하였으며, Google Cloud Vision API[2]를 사용하였다. 그리고 MySQL을 이용해서 게시글 작성 및 해당 게시글의 댓글을 남기는 유저정보 등을 관리하고 있다.

본 프로젝트는 STS(Spring Tool Suite 3)를 사용하여 개발하였고 Database로는 Mysql 8.0을 이용하였다. 그림들의 보기 좋은 배치와 깔끔한 디자인을 위해 BootStrap 5.3을 사용하였다.

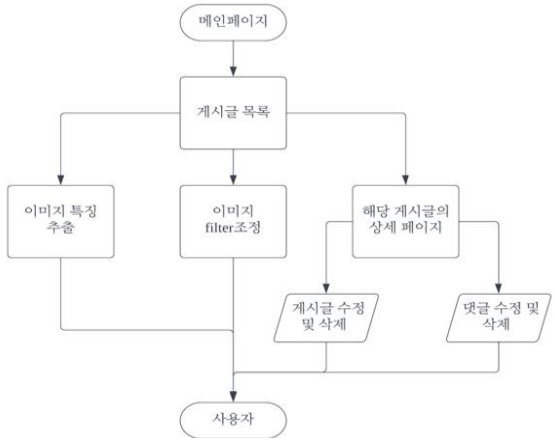


Fig. 2. 기능 흐름도

그림 2는 기능 흐름도로 시스템의 전체적인 흐름을 Flow Chart를 통해 표현했으며, 시스템 처리를 거쳐 사용자에게 도달하는 주요 기능의 흐름을 확인할 수 있다.

## III. 주요 기능

첫 번째 기능으로는 게시판에 게시글을 보여주는 페이지다. 그림 3과 같이 게시글을 작성하게 되면, grid 배치를 이용한 이미지(게시글)가 화면에 표출되게 된다.

Fig. 3. 그리드 배치를 통한 이미지 출력

두 번째 기능으로는 게시글 목록에서 해당 게시글의 제목을 누르면 해당 게시글의 조회수 및 게시글 수정, 삭제 그리고 댓글을 작성할 수 있는 페이지로 이동한다.

```
//게시물 삭제
@RequestMapping(value="/postdelete",method = {RequestMethod.POST, RequestMethod.GET})
public String postdelete(@RequestParam(value="userpassword",required = false) String password,
        @RequestParam(value="bno")int bno ,Model model, RedirectAttributes ra) throws Exception {

    BoardVO board = service.detail(bno);
    BCryptPasswordEncoder encoder = new BCryptPasswordEncoder();

    if (encoder.matches(password, board.getUserPassword())) {
        service.delete(bno);
        ra.addAttribute("deresult", "deleteOK"); // 삭제 성공 여부를 RedirectAttributes에 추가
        return "redirect:list";
    } else {
        ra.addAttribute("deresult", "deleteFail"); // 삭제 실패 여부를 RedirectAttributes에 추가
        return "redirect:detail?bno=" + bno;
    }
}
```

Fig. 4. Password check

그림 4는 controller에서 삭제와 수정 이벤트가 발생하면 어떻게 동작하고 있는지를 구현하였다. 해당 게시글을 지우기 위해선 그 게시글이 가지고 있는 고유 번호를 보내기 위해 GET을 사용하고 작성한 게시글의 비밀번호를 controller에 보내기 위해 POST를 동시에 사용하고 있다. view에서 전송하는 게시글의 비밀번호는 spring security를 이용해서 게시글의 작성 시, 비밀번호를 암호화시켰다. 그리고 controller에서 BCryptPasswordEncoder()를 이용해 view에서 보내는 비밀번호를 복호화 시켜 matches를 이용해 전송한 비밀번호와 데이터베이스에 있는 해당 게시글의 비밀번호가 일치하는지 비교하고 일치할 경우와 일치하지 않을 경우를 구분해서 기능을 수행하도록 구현하였다. 그리고 댓글 수정, 삭제 기능도 동일한 모듈을 활용하여 구현하였다.

## IV. API 활용

업로드된 사진의 label과 객체 값을 추출하기 위해 google cloud vision api 사용하였다. api는 label, object, text, properties, safe search 등의 주요 기능들이 있다. 본 웹사이트에서 그림에 대한 정보를 추출하는 것은 설계단계에서 객체와 label 등으로 충분하다고 판단했기 때문에 이 두 결과를 선택해서 활용하였다.

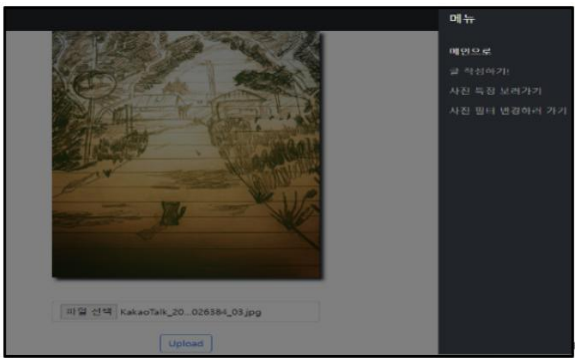


Fig. 5. 사진의 label과 객체 값을 추출해주기 위해 사진을 upload할 수 있는 페이지

그림 5와 같이 그림 파일을 upload 시키고 화면상의 upload 버튼을 클릭하게 되면 해당 이미지에 label 값과 객체 값이 추출된다.


Fig. 6. 사진의 label과 그 신뢰도 수치를 그래프로 표현

그림 6은 업로드된 이미지와 그 이미지에 대한 특징(label)과 그 신뢰도 값이 결과로 추출된다.

```
// MultipartFile 객체를 바이트 배열로 변환
byte[] imageBytes = file.getBytes();

// 이미지를 Vision API로 전송하여 라벨을 추출
List<AnnotateImageRequest> requests = new ArrayList<>();
ByteString imgBytes = ByteString.copyFrom(imageBytes);
Image img = Image.newBuilder().setContent(imgBytes).build();

//이미지 label을 추출
Feature feat = Feature.newBuilder().setType(Feature.Type.LABEL_DETECTION).build();
// 객체 추출을 위한 Feature를 설정.
Feature objectLocalizationFeature = Feature.newBuilder().setType(Feature.Type.OBJECT_LOCALIZATION).build();
```

Fig 7. Upload한 이미지를 vision api로 호출해 label과 object로 추출해주는 function

view에서 전송된 파일을 byte 배열로 변환하여 vision api로 전송하고 LABEL_DETECTION 함수를 활용하여 이미지 lable을 추출시키고, OBJECT_LOCALIZATION 함수를 활용하여 그림 8의 결과처럼 객체와 신뢰도 값을 출력할 수 있다.


Fig. 8. 사진의 객체(Object)와 신뢰도 값을 추출

네 번째 기능은 업로드된 사진을 hexColor로 색을 조정해서 이미지 배경에 filter를 적용하는 기능이다.

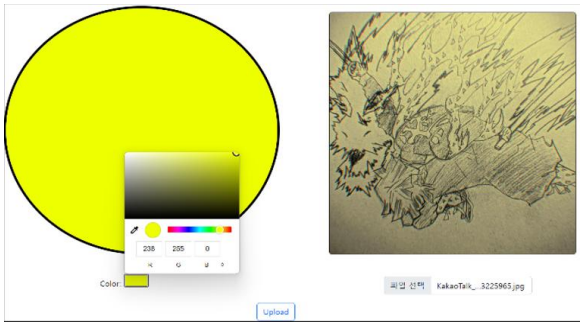


Fig. 9. 색을 조정해서 원본 이미지에 filter를 적용하는 화면

그림 9와 같이 filter를 적용하는 페이지로 이동하면, 왼쪽에 있는 color 버튼을 클릭하여 색을 조정할 수 있는 interface로 원하는 값을 입력함으로써 색을 찾거나 cursor로 움직여서 찾을 수도 있다. 오른쪽에는 자신이 색을 바꾸고 싶은 이미지를 업로드하면 바꾸고자 하는 원본 이미지를 미리보기 형식으로 출력되도록 구현했다.


Fig. 10. Upload한 원본 이미지와 filter가 적용된 이미지

그림 10은 업로드된 이미지의 원본 이미지와 색을 조정한 filter를 씌운 이미지의 값을 view에 출력하였다.

```
private BufferedImage applyFilter(BufferedImage image, String hexColor) {
    int rgb = Color.decode(hexColor).getRGB();
    int alpha = 128; // 투명도 (0-255 범위)

    BufferedImage filteredImage = new BufferedImage(image.getWidth(), image.getHeight(),
            BufferedImage.TYPE_INT_ARGB);
    Graphics2D graphics = filteredImage.createGraphics();
    graphics.drawImage(image, 0, 0, null);

    graphics.setColor(new Color(rgb, true));
    graphics.setComposite(AlphaComposite.getInstance(AlphaComposite.SRC_ATOP, alpha / 255f));
    graphics.fillRect(0, 0, image.getWidth(), image.getHeight());
    graphics.dispose();

    return filteredImage;
}
```

Fig. 11. hexColor의 초기값을 지정해 upload한 이미지와 조정한 hexColor를 통해 동작하는 controller

그림 11과 같이 ImageIO.read를 통해 업로드한 이미지를 읽고, 업로드 이미지와 조정한 hexColor에 대해서 applyFilter를 통해 원본 이미지에 조정한 hexColor를 적용하는 역할을 수행한다. 그리고 이미지 파일을 base64로 인코딩하여 문자열 형태로 변환한다. originalImage를 base64로 인코딩하여 originalEncodeImg에 저장하고 filteredImage를 png 형식으로 ByteArrayOutputStream에 기

록하고 해당 스트림을 byte 배열로 변환하여 base64로 인코딩한다.

filteredEncodeImage에 인코딩된 문자열을 저장한 후 이미지 파일을 base64로 인코딩하여 서버에서 클라이언트로 전달하면 filter가 적용된 이미지를 내려받을 수 있게 구현하였다.

## V. Conclusions

코로나19에도 불구하고 미술시장의 규모는 계속 성장하고 있어서 그림에 대한 열풍이 예전보다는 더 높아질 것으로 생각한다. 더불어 인터넷과 디지털 기술의 발전으로 미술에 대한 시각적 표현은 지금보다 더욱 발전할 것이며, 본 시스템의 서비스가 이미지를 공유하는 사용자를 만족할 뿐만 아니라 디지털 기술 발전에 조금이라도 도움이 될 것이라고 기대한다.

## REFERENCES

[1] https://www.korea.kr/news/policyNewsView.do?newsId=148910182

[2] https://cloud.google.com/visoin/docs