제20차 회의록

회의날짜	@11/22/2024
Tags	전체
작성자	박지현

참석

채현, 지현, 소희, 준하

회의 안건

BE, FE 연동

회의 내용

board.js 마지막 부분

```
// * 상점별 게시물 조회
export const getBoardsByShop = (searchTerm) => {
  return instance
    .get(`/api/board/shop?searchTerm=${encodeURIComponent(sea
    .then((response) => {
      return response.data.data; // 데이터 구조에 따라 수정 필요
    })
    .catch((error) => {
      console.error('상점별 게시글 조회 에러:', error.response.dat
      throw error;
    });
};
```

BoardListShop.jsx 스타일 전까지

```
function BoardListShop() {
  const { searchTerm } = useParams(); // URL 파라미터로 상점명 가
  const navigate = useNavigate();

  // 상점별 게시글 리스트 조회
  const { data: boardData, isLoading, isError } = useQuery(
    ['getBoardsByShop', searchTerm],
    () => getBoardsByShop(searchTerm), // `searchTerm`을 바로 
    {
      enabled: !!searchTerm, // searchTerm이 있을 때만 쿼리 실행
    }
  );

  if (isLoading) {
    return <Loading />;
  }

  if (isError) {
    return <Error />;
  }

  // 데이터가 없을 경우
  if (!boardData || boardData.length === 0) {
    return (
      <Layout>
        <NullAlert alertMessage={`${searchTerm} 상점에는 판매 게
      </Layout>
    );
  }

  // 상세 페이지로 이동
  const setPageChange = (boardId) => {
    navigate(`/BoardDetail/${boardId}`);
  };

  return (
    <Layout>
      <Title>{searchTerm}의 상품 목록</Title>
```

```
      <ListSection>
        {boardData.map((board) => (
          <ListOneDiv onClick={() => setPageChange(board.id)}
            <Image
              width="130px"
              height="130px"
              borderradius="10px"
              src={`http://localhost:8001${board.image}`}
              alt="상품 이미지"
            />
            <ListInfoDiv>
              <ListTitleH1>{board.title}</ListTitleH1>
              <ListDetailH3>
                <span>{board.nickname}</span>
              </ListDetailH3>
              <ListPriceH2>
                {board.status && <StatusButton color="black">
                {Number(board.price).toLocaleString()}원
              </ListPriceH2>
            </ListInfoDiv>
          </ListOneDiv>
        ))}
      </ListSection>
    </Layout>
  );
}
```

BoardList.jsx 스타일 전까지

```
function BoardList() {
  const [boardData, setBoardData] = useState([]);
  const [selectedCategory, setSelectedCategory] = useState(nu
  const [searchTerm, setSearchTerm] = useState('');
  const navigate = useNavigate();

  const getBoardList = (categoryId = null) => {
    const setPage = { page: 0, size: 100, sort: ["createdAt,D
```

```
    getBoardListMutation.mutate({ setPage, categoryId });
  };

  const getBoardListMutation = useMutation(
    ({ setPage, categoryId }) => getBoards(setPage, categoryI
    {
      onSuccess: (response) => {
        setBoardData(response);
      },
      onError: (error) => {
        console.error('Error fetching boards:', error);
      },
    }
  );

  useEffect(() => {
    getBoardList(); // 모든 게시글 조회
  }, []);

  const handleCategorySelect = (categoryId) => {
    setSelectedCategory((prevCategory) =>
      prevCategory === categoryId ? null : categoryId
    );
    getBoardList(categoryId);
  };

  const filteredBoardData = selectedCategory
    ? boardData.filter((board) => board.categoryId === select
    : boardData;

  const handleSearch = (event) => {
    if (event.key === "Enter") {
      navigate(`/BoardListShop/${searchTerm}`);
    }
  };

  return (
    <Layout>
```

```
        <SearchBar>
          <input
            type="text"
            placeholder="상점명 입력"
            value={searchTerm}
            onChange={(e) => setSearchTerm(e.target.value)}
            onKeyDown={handleSearch}
          />
        </SearchBar>

        <CategorySection>
          {categories.map((category) => (
            <CategoryItem
              key={category.id}
              onClick={() => handleCategorySelect(category.id)}
              isSelected={selectedCategory === category.id}
            >
              <CategoryImage>
                <img src={category.image} alt={category.name} /
              </CategoryImage>
              <span>{category.name}</span>
            </CategoryItem>
          ))}
        </CategorySection>

        <ListSection>
          {filteredBoardData.map((board) => (
            <ListOneDiv key={board.id} onClick={() => navigate(
              <Image
                width="130px"
                height="130px"
                borderradius="10px"
                src={`http://localhost:8001${board.image}`}
                alt="상품 이미지"
              />
              <ListInfoDiv>
                <ListTitleH1>{board.title}</ListTitleH1>
                <ListDetailH3>{board.nickname}</ListDetailH3>
```

```
                <ListPriceH2>{Number(board.price).toLocaleStrin
              </ListInfoDiv>
            </ListOneDiv>
          ))}
        </ListSection>
      </Layout>
    );
}
```

🥹🥹🥹🥹🥹🥹🥹🥹

문제의 원인 분석

- **URL Routing Issue:** 개발자 도구에서 "No routes matched location"이라는 메시지가 나타나는 것은 보통 프론트엔드에서 해당 URL에 대한 라우터 경로가 설정되지 않았다는 것을 의미해요.
- **네트워크 요청 로그 미출력:** 개발자 도구 Network 탭에 아무런 요청이 표시되지 않는 것은 아예 프론트에서 백엔드로 요청이 전송되지 않았다는 뜻이에요. 따라서 프론트엔드 라우터 설정 또는 프론트에서 요청하는 코드에 문제가 있을 수 있어요.
- **상점별 데이터가 전달되지 않음:** URL에 포함된 한글 데이터 (`%EB%8F%99%EA%B5%AD%EB%B9%B5%EC%A7%91`)가 백엔드에서 제대로 처리되지 못하는 인코딩 문제도 있을 수 있습니다.

```
import React from 'react';
import { styled } from 'styled-components';
import { Layout, Image, StatusButton } from '../components/el
import { useNavigate, useParams } from 'react-router-dom';
import { useQuery } from 'react-query';
import { getBoardsByShop } from '../api/boards';
import Loading from './statusPage/Loading';
import Error from './statusPage/Error';
import NullAlert from './statusPage/NullAlert';
```

```
function BoardListShop() {
  const { searchTerm } = useParams(); // URL 파라미터에서 상점명
  const navigate = useNavigate();

  // 상점별 게시글 리스트 조회
  const { data, isLoading, isError } = useQuery(
    ['getBoardsByShop', searchTerm],
    () => getBoardsByShop(searchTerm),
    {
      onError: (error) => {
        console.error('Error fetching shop boards:', error);
      },
    }
  );

  if (isLoading) {
    return <Loading />;
  }

  if (isError) {
    return <Error />;
  }

  // 데이터가 없을 경우
  if (!data || data.length === 0) {
    return (
      <Layout>
        <NullAlert alertMessage={`${searchTerm} 상점에는 판매 게
      </Layout>
    );
  }

  // 상세 페이지 이동 함수
  const setPageChange = (boardId) => {
    navigate(`/BoardDetail/${boardId}`);
  };
```

```
  return (
    <Layout>
      <Title>{searchTerm}의 상품 목록</Title>
      <ListSection>
        {data.map((board) => (
          <ListOneDiv key={board.id} onClick={() => setPageCh
            <Image
              width="130px"
              height="130px"
              borderradius="10px"
              src={`http://localhost:8001${board.image}`}
              alt="상품 이미지"
            />
            <ListInfoDiv>
              <ListTitleH1>{board.title}</ListTitleH1>
              <ListDetailH3>
                <span>{board.nickname}</span>
              </ListDetailH3>
              <ListPriceH2>
                {board.status && <StatusButton color="black">
                {Number(board.price).toLocaleString()}원
              </ListPriceH2>
            </ListInfoDiv>
          </ListOneDiv>
        ))}
      </ListSection>
    </Layout>
  );
}

export default BoardListShop;
```

App.js (프론트)

```
<Route path="/BoardListShop/:searchTerm" element={<BoardListS
```

백엔드쪽....

app.js

```
const express = require('express');
const cookieParser = require('cookie-parser');
const morgan = require('morgan');
const path = require('path');
const session = require('express-session');
const nunjucks = require('nunjucks');
const dotenv = require('dotenv');
const passport = require('passport');
const cors = require('cors'); // CORS 모듈 추가

dotenv.config();
const pageRouter = require('./routes/page');
const authRouter = require('./routes/auth');  // authRouter 가
const postRouter = require('./routes/post');
const userRouter = require('./routes/user');
const mypageRouter = require('./routes/mypage'); // 마이페이지에
const { sequelize } = require('./models');
const passportConfig = require('./passport');

const app = express();
passportConfig(); // 패스포트 설정
app.set('port', process.env.PORT || 8001);
app.set('view engine', 'html');
nunjucks.configure('views', {
  express: app,
  watch: true,
});
sequelize.sync({ force: false })
  .then(() => {
    console.log('데이터베이스 연결 성공');
  })
  .catch((err) => {
    console.error(err);
  });
```

```
app.use(morgan('dev'));
app.use(express.static(path.join(__dirname, 'public')));
app.use('/img', express.static(path.join(__dirname, 'uploads'
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser(process.env.COOKIE_SECRET));
app.use(session({
  resave: false,
  saveUninitialized: false,
  secret: process.env.COOKIE_SECRET,
  cookie: {
    httpOnly: true,
    secure: false,
  },
}));
app.use(passport.initialize());
app.use(passport.session());

app.use(cors({
  origin: 'http://localhost:3000', // 프론트엔드 도메인 허용
  credentials: true, // 쿠키 공유를 위해 true 설정
}));


app.use('/', pageRouter);
app.use('/auth', authRouter);
app.use('/post', postRouter);
app.use('/user', userRouter);
app.use('/board', mypageRouter);

// *** 프론트와 api 연결!!! ***
app.use('/api/member',authRouter); // api/member경로로 들어오는
app.use('/api/board',postRouter)  // board.js요청. 게시물 작성,
app.use('/uploads', express.static('uploads'));
app.use('/api/mypage', mypageRouter)  // 사장님유저 마이페이지
```

```
app.use((req, res, next) => {
  const error =  new Error(`${req.method} ${req.url} 라우터가 없
  error.status = 404;
  next(error);
});


app.use((err, req, res, next) => {
  res.locals.message = err.message;
  res.locals.error = process.env.NODE_ENV !== 'production' ?
  res.status(err.status || 500);
  res.render('error');
});

app.listen(app.get('port'), () => {
  console.log(app.get('port'), '번 포트에서 대기중');
});
```

routes/post.js

```
const express = require('express');
const multer = require('multer');
const path = require('path');
const fs = require('fs');

const { afterUploadImage, uploadPost } = require('../controll
const { isLoggedIn, isOwner } = require('../middlewares');

const router = express.Router();

const { getPosts } = require('../controllers/post');
const { getPostDetail , editPost, deletePost } = require('../
const { getBoardsByShop } = require('../controllers/post');

try {
  fs.readdirSync('uploads');
```

```
} catch (error) {
  console.error('uploads 폴더가 없어 uploads 폴더를 생성합니다.');
  fs.mkdirSync('uploads');
}

const upload = multer({
  storage: multer.diskStorage({
    destination(req, file, cb) {
      cb(null, 'uploads/');
    },
    filename(req, file, cb) {
      const ext = path.extname(file.originalname);
      cb(null, path.basename(file.originalname, ext) + Date.n
    },    // 이미지.png -> 이미지12312315.png
  }),
 // limits: { fileSize: 5 * 1024 * 1024 },
});


// *** 프론트요청 ***
router.post('/write',isLoggedIn, isOwner, upload.single('imag
router.get('/',getPosts); // /api/board?page=${setPage.page}
router.get('/detail/:boardId', getPostDetail); // /api/board/
router.put('/modify/:boardId', isLoggedIn, isOwner, editPost)
router.delete('/delete/:boardId',isLoggedIn, isOwner, deleteP
router.get('/shop', getBoardsByShop); // 상점별 게시글 리스트 조회

// POST /post/img
router.post('/img', isLoggedIn, upload.single('img'), afterUp

// POST /post
const upload2 = multer();
router.post('/', isLoggedIn, upload2.none(), uploadPost);

module.exports = router;
```

controllers/post.js

```
const { Post, Hashtag } = require('../models');
const { User } = require('../models');


// 상점별 게시물 조회
exports.getBoardsByShop = async (req, res, next) => {
  try {

    let { searchTerm } = req.query;
    searchTerm = decodeURIComponent(searchTerm); // 인코딩된 문

    console.log(`상점명으로 검색: ${searchTerm}`);


    // 검색어와 일치하는 상점명(nick)을 가진 유저 조회
    const users = await User.findAll({
      where: {
        nick: {
          [Op.like]: `%${searchTerm}%` // 부분 일치하는 상점명도 검
        },
        userType: 'owner', // owner 타입의 유저만 조회
      },
      include: [
        {
          model: Post,
          required: false, // LEFT JOIN: 게시물이 없는 경우도 포함
        }
      ],
      attributes: ['id', 'nick'] // 필요에 따라 반환할 속성을 조정
    });

    console.log('조회된 유저: ',users)

    // 응답 데이터 생성
    const responseData = users.map(user => ({
      id: user.id,
```

```
      nick: user.nick,
      posts: user.Posts // 게시물이 있는 경우 배열로 반환, 없으면 빈
    }));

    console.log("응답데이터: ",responseData)

    res.status(200).json({ data: responseData });
  } catch (error) {
    console.error('Error fetching boards by shop:', error);
    res.status(500).json({ error: '상점별 게시물 조회 중 오류가 발생
  }
};
```

boards.js

```
export const getBoardsByShop = async ({ searchTerm }) => {
  return instance.get(`/api/boards/shop?searchTerm=${encodeURI
    .then(response => response.data)
    .catch(error => {
      console.error('상점별 게시글 조회 실패:', error.response?.da
      throw error;
    });
};
```

BoardListShop.jsx

```
const { data, isLoading, isError } = useQuery(
['getBoardsByShop', searchTerm],
() => getBoardsByShop(searchTerm),
{
onSuccess: (response) => {
console.log("API 호출 성공:", response);
},
onError: (error) => {
```

```
console.error("API 호출 에러:", error);
},
}
);

useEffect(() => {
console.log("데이터 확인:", data);
if (data) setBoardData(data); // 데이터 상태 업데이트
}, [data]);
```

boards.js

```
export const getBoardsByShop = (searchTerm) => {
  const encodedSearchTerm = encodeURIComponent(searchTerm); /
  return instance
    .get(`/api/board/shop?searchTerm=${encodedSearchTerm}`) /
    .then((response) => {
      return response.data.data;
    })
    .catch((error) => {
      console.error('상점별 게시글 조회 에러:', error.response.dat
      throw error;
    });
};
```

boards.js (new!) → 이거랑 맞는 BoardListShop.jsx 필요해요

```
export const getBoardsByShop = ({ searchTerm }) => {
  return instance
    .get(`/api/board/shop`, { params: { searchTerm } }) // se
    .then((response) => {
      if (!response.data || !response.data.data) {
        throw new Error('상점별 게시글 데이터가 비어 있습니다.');
      }
      return response.data.data;
    })
```

```
    .catch((error) => {
      console.error('상점별 게시글 조회 에러:', error);
      throw error;
    });
};
```

BoardListShop. 최종!!!

```
import React, { useState, useEffect } from 'react';
import styled from 'styled-components';
import { Layout, Image, StatusButton } from '../components/el
import { useNavigate, useParams } from 'react-router-dom';
import { useMutation, useQuery } from 'react-query';
import { getBoards } from '../api/boards';
import { getBoardsByShop } from '../api/boards';
import Loading from './statusPage/Loading';
import Error from './statusPage/Error';
import NullAlert from './statusPage/NullAlert';

function BoardListShop() {
  const [boardData, setBoardData] = useState([]);
  const { searchTerm } = useParams(); // URL 파라미터로 상점명 가
  const navigate = useNavigate();

  // 상점별 게시글 리스트 조회
  const { data, isLoading, isError } = useQuery(
    ['getBoardsByShop', searchTerm],
    () => getBoardsByShop({ page: 0, size: 100, sort: ["creat
    {
      onSuccess: (response) => {
        setBoardData(response); // 응답 데이터 설정
      },
    }
  );

  if (isLoading) {
    return <Loading />
```

```
  }

  if (isError) {
      return <Error />
  }

  // 데이터가 존재하지 않을 경우
  if (!data || data.length === 0) {
    return (
      <Layout>
        <NullAlert alertMessage={`${searchTerm} 상점에는 판매 게
      </Layout>
    );
  }

  // 상세 페이지로 이동
  const setPageChange = (boardId) => {
    navigate(`/BoardDetail/${boardId}`);
  };

  return (
    <Layout>
      <Title>{searchTerm}의 상품 목록</Title>

      {/* 게시물 리스트 섹션 */}
      <ListSection>
        {boardData.map((board) => (
          <ListOneDiv onClick={() => setPageChange(board.id)}
            <Image
              width="130px"
              height="130px"
              borderradius="10px"
              src={`http://localhost:8001${board.image}`}
              alt="상품 이미지"
            />
            <ListInfoDiv>
              <ListTitleH1>{board.title}</ListTitleH1>
              <ListDetailH3>
```

```
                <span>{board.nickname}</span> {/* 상점명(사장님의
              </ListDetailH3>
              <ListPriceH2>
                {board.status && <StatusButton color="black">
                {Number(board.price).toLocaleString()}원
              </ListPriceH2>
            </ListInfoDiv>
          </ListOneDiv>
        ))}
      </ListSection>
    </Layout>
  );
}

export default BoardListShop;

// 스타일 정의
const Title = styled.h1`
  text-align: center;
  margin: 20px 0;
  font-size: 24px;
  font-weight: bold;
  color: #333;
`;

const ListSection = styled.section`
  display: flex;
  flex-direction: column;
  margin-top: 10px;
`;

const ListOneDiv = styled.div`
  padding: 15px 5px;
  display: flex;
  border-bottom: 1px solid lightgrey;
  cursor: pointer;
`;
```

```
const ListInfoDiv = styled.div`
  margin-left: 20px;
  display: flex;
  flex-direction: column;
  justify-content: center;
`;

const ListTitleH1 = styled.h1`
  font-size: 18px;
  font-weight: 500;
  margin: 0;
`;

const ListPriceH2 = styled.h2`
  margin: 0;
  & span {
    font-size: 20px;
    font-weight: 600;
    margin: 0;
  }
`;

const ListDetailH3 = styled.h3`
  margin: 10px 0 7px 0;
  font-size: 15px;
  font-weight: 300;
  color: grey;
`;
```

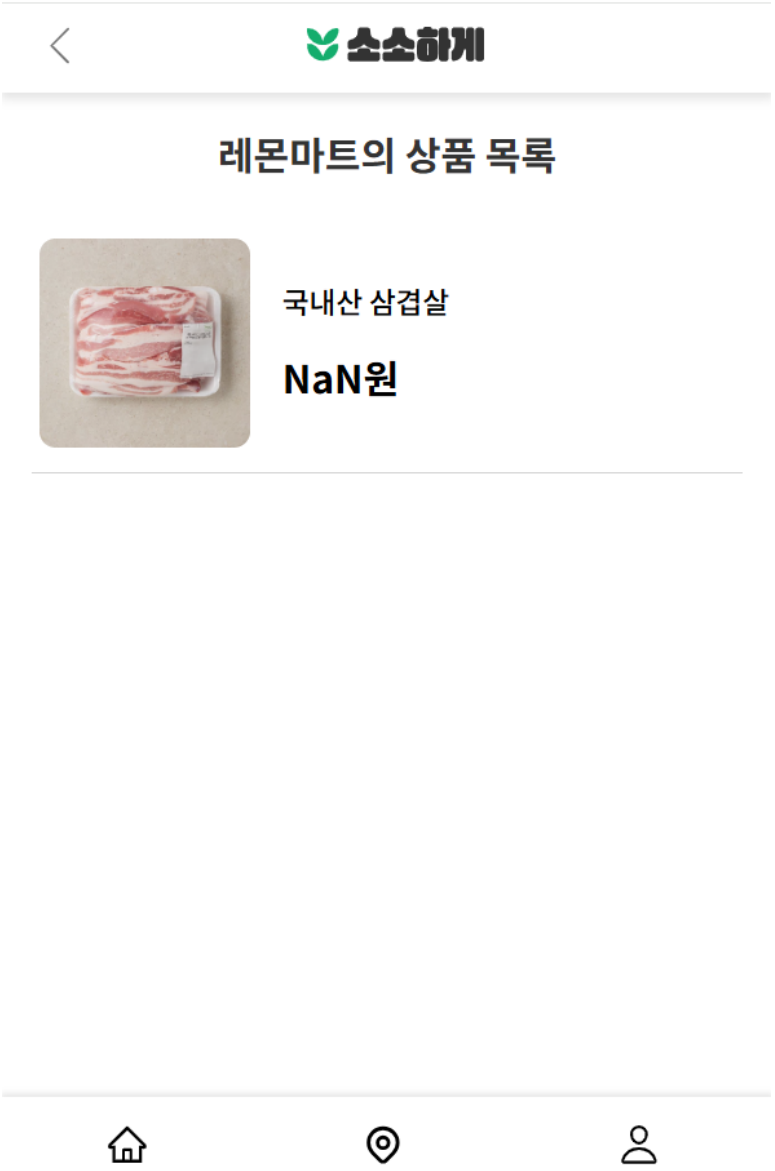
```
// * 상점별 게시물 조회
export const getBoardsByShop = (searchTerm) => {
  return instance
    .get(`/api/board/shop?searchTerm=${encodeURIComponent(sea
    .then((response) => {
```

```
        return response.data.data; // 데이터 구조에 따라 수정 필요
      })
      .catch((error) => {
        console.error('상점별 게시글 조회 에러:', error.response.dat
        throw error;
      });
};

<Route path="/BoardListShop" element={<BoardListShop />} />
```

백엔드 해야할것...

BoardListShop.jsx문제 해결!



- 가격 NaN으로 받아오는 문제 해결하기..
- 카테고리별 화면 뜨는거

공지사항

다음 회의

2024.11.25(월)