

1. 템플릿 만들기

GET /admin/shops/detail/:id

아래와 같은 화면을 만들기

주소 : 서울 동작구 흑석로 13 / 121동
전화 : 02-543-222
운영시간 : 11시~20시
핸드폰 번호 : 010-3332-2222

메뉴 작성하기

메뉴명 :

가격 :

메뉴작성

목록으로

수정

templete/admin/detail.html

```
<hr />

    메뉴 작성하기

    <form action="" method="post">
      <div class="col-xs-3">
        메뉴명 : <input type="text" class="form-control" name="name" required />
      </div>

      <div class="col-xs-3">
        가격 : <input type="text" class="form-control" name="price" required />
      </div>
      <div class="col-xs-3">
        <button class="btn btn-primary" style="margin-top: 19px">메뉴작성 </button>
      </div>
    </form>
```

```

20      운영시간 : {{ shop.open_time }}
21      <br />
22      핸드폰 번호 : {{ shop.cell_phone }}
23
24      <hr />
25
26      ..... 메뉴 작성하기
27
28      ..... <form action="" method="post">
29      ..... <div class="col-xs-3">
30      ..... 메뉴명 : <input type="text" class="form-control"
31      ..... </div>
32
33      ..... <div class="col-xs-3">
34      ..... 가격 : <input type="text" class="form-control"
35      ..... </div>
36      ..... <div class="col-xs-3">
37      ..... <button class="btn btn-primary" style="margin-t
38      ..... </div>
39      ..... </form>
40
41      </div>
42    </div>
43
44    <a href="/admin/shops" class="btn btn-default">목록으로</a>
45    <a href="/admin/shops/edit/{{ shop.id }}" class="btn btn-
46    </div>
47

```

2. 모델작성

models/Shops.js

```

// 메뉴 모델 관계도
Shops.associate = (models) => {

  // 메뉴 모델에 외부키를 건다
  // onDelete 옵션의 경우 제품 하나가 삭제되면 외부키가 걸린 메모들도 싹다
  삭제해준다
  Shops.hasMany( models.ShopsMenu ,
    { as: 'Menu' , foreignkey: 'shop_id', sourceKey: 'id' , onDelete: 'CASCADE' }
  );

}

```

```

1  const moment = require('moment');
2
3  module.exports = (sequelize, DataTypes) => {
4    const Shops = sequelize.define('Shops',
5      {
6        id: { type: DataTypes.INTEGER, primary
7        name : { type: DataTypes.STRING , com
8        address : { type: DataTypes.STRING ,
9        location : { type: DataTypes.STRING ,
10       phone : { type: DataTypes.STRING , co
11       open_time : { type: DataTypes.STRING
12       cell_phone : { type: DataTypes.STRING
13     }
14   });
15
16   // 메뉴 모델 관계도
17   Shops.associate = (models) => {
18
19     // 메뉴 모델에 외부키를 건다
20     // onDelete 옵션의 경우 제품 하나가 삭제되면 외부키
21     Shops.hasMany( models.ShopsMenu ,
22       { as: 'Menu' , foreignkey: 'shop_id',
23     });
24
25   }
26
27   Shops.prototype.dateFormat = (date) => (
28     moment(date).format('YYYY-MM-DD')

```

models/ShopsMenu.js

```

module.exports = (sequelize, DataTypes) => {
  const ShopsMenu = sequelize.define('ShopsMenu',
    {
      id: { type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true },
      name : { type: DataTypes.STRING , comment: '메뉴명' },
      price : { type: DataTypes.INTEGER , comment: '가격' }
    }, {
      tableName: 'ShopsMenu'
    }
  );

  return ShopsMenu;
}

```

app.js DB sync -> 주식해제 -> 저장 -> 다시 주식

```
38
39
40 }
41
42 dbConnection(){
43     // DB authentication
44     db.sequelize.authenticate()
45     .then(() => {
46         console.log('Connection has been established');
47         return db.sequelize.sync();
48     })
49     .then(() => {
50         console.log('DB Sync complete.');
```

ShopsMenu 확인

```
[nodemon] restarting due to changes...
[nodemon] starting `node server.js`
Express listening on port 3000
Executing (default): SELECT 1+1 AS result
Connection has been established successfully.
Executing (default): CREATE TABLE IF NOT EXISTS `Shops` (`id` INTEGER auto
_increment, `name` VARCHAR(255), `address` VARCHAR(255), `location` VARCHAR
AR(255), `phone` VARCHAR(255), `open_time` VARCHAR(255), `cell_phone` VARCHAR
HAR(255), `createdAt` DATETIME NOT NULL, `updatedAt` DATETIME NOT NULL, PR
IMARY KEY (`id`)) ENGINE=InnoDB;
Executing (default): SHOW INDEX FROM `Shops`
Executing (default): CREATE TABLE IF NOT EXISTS `ShopsMenu` (`id` INTEGER
auto_increment, `name` VARCHAR(255), `price` INTEGER, `createdAt` DATETIME
E NOT NULL, `updatedAt` DATETIME NOT NULL, `shop_id` INTEGER, PRIMARY KEY
(`id`), FOREIGN KEY (`shop_id`) REFERENCES `Shops` (`id`) ON DELETE CASCADE
E ON UPDATE CASCADE) ENGINE=InnoDB;
```

POST /admin/shops/detail/:id 저장되는 부분 작성

controllers/admin/index.js

```
router.get('/shops/detail/:id', ctrl.add_menu );
```

controllers/admin/admin.ctrl.js

```
exports.add_menu = async(req, res) => {  
  
  try{  
  
    const shop = await models.Shops.findByPk(req.params.id);  
    // create + as에 적은 내용 ( shops.js association 에서 적은 내용 )  
    await shop.createMenu(req.body);  
    res.redirect('/admin/shops/detail/'+req.params.id);  
  
  }catch(e){  
    console.log(e)  
  }  
  
}
```

메뉴 저장해보고 workbench 에서 확인

[illegible]

아래와 같이 메뉴 명을 뿌려지게 작성

GET /admin/shops/detail/:id

localhost:3000/admin/shops/detail/2

ADMIN

22

작성일 : 2021-04-08

주소 : 서울 동작구 흑석로 13 / 121동

전화 : 02-543-222

운영시간 : 11시~20시

핸드폰 번호 : 010-3332-2222

메뉴 작성하기

- 닭발 / 1200 원 (삭제)
- 파스타 / 500 원 (삭제)

메뉴명 : 가격 :

template/admin/detail.html

메뉴 작성하기

```
<ul>
  {% for menu in shop.Menu %}
  <li>
    {{ menu.name }} / {{ menu.price }} 원
    (
      <a href="/admin/shops/delete/{{ shop.id }}/{{ menu.id }}"
        onclick="return confirm('삭제하시겠습니까?')">
        삭제
      </a>
    )
  </li>
  {% endfor %}
</ul>
```

```

20     운영시간 : {{ shop.open_time }}
21     <br />
22     핸드폰 번호 : {{ shop.cell_phone }}
23
24     <hr />
25
26     ..... 메뉴 작성하기
27
28     ..... <form action="" method="post">
29     ..... <div class="col-xs-3">
30     ..... 메뉴명 : <input type="text" class="form-control" n
31     ..... </div>
32
33     ..... <div class="col-xs-3">
34     ..... 가격 : <input type="text" class="form-control" n
35     ..... </div>
36     ..... <div class="col-xs-3">
37     ..... <button class="btn btn-primary" style="margin-t
38     ..... </div>
39     ..... </form>
40
41     </div>
42 </div>
43
44     <a href="/admin/shops" class="btn btn-default">목록으로</a>
45     <a href="/admin/shops/edit/{{ shop.id }}" class="btn btn-
46 </div>
47

```

controllers/admin/admin.ctrl.js

아래와 같이 get_shops_detail 에 shop 데이터 수정

```

exports.get_shops_detail = async(req, res) => {

  try{

    const shop = await models.Shops.findOne({
      where : {
        id : req.params.id
      },
      include :[
        'Menu'
      ]
    });

    res.render('admin/detail.html', { shop });

  }catch(e){
    console.log(e)
  }

}

```

```

exports.get_shops_detail = async(req, res) => {
  try{
    const shop = await models.Shops.findOne({
      where : {
        id : req.params.id
      },
      include : [
        'Menu'
      ]
    });

    res.render('admin/detail.html', { shop });

  }catch(e){
    console.log(e)
  }
}

```

삭제하기 작성

controllers/admin/index.js

```

// 메뉴 삭제
router.get('/shops/delete/:shop_id/:menu_id', ctrl.remove_menu );

```

controllers/admin/admin.ctrl.js

하단 추가

```

exports.remove_menu = async(req, res) => {
  try{
    await models.ShopsMenu.destroy({
      where: {
        id: req.params.menu_id
      }
    });

    res.redirect('/admin/shops/detail/' + req.params.shop_id );

  }catch(e){
  }
}

```


삭제 눌러보기

메뉴 작성하기

- 닭발 / 1200 원 ([삭제](#))
- 파스트 / 500 원 ([삭제](#))

메뉴명 .

가격 .

