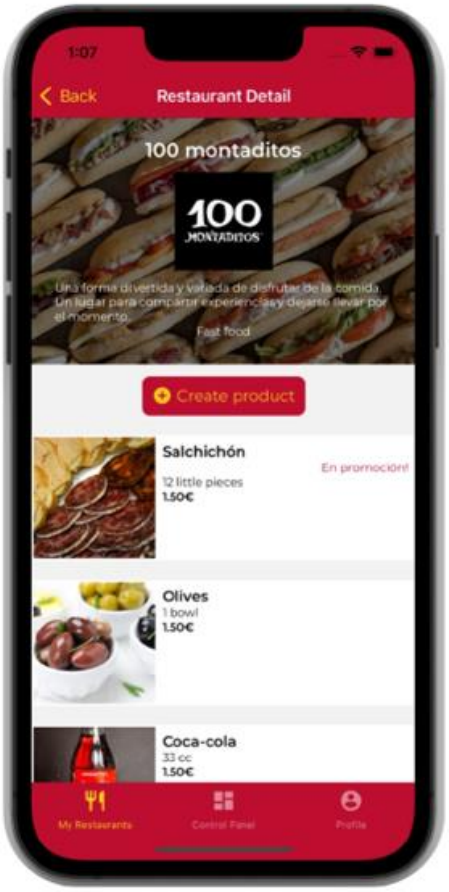
EXAMEN PRODUCT PROMOTED

Individual laboratory evaluation. June 2022.

Featured products

Una vez que se haya instalado la primera versión de DeliverUS lanzado, nuestros inversores han solicitado la inclusión de una nueva funcionalidad que consiste en ofrecer a los propietarios la posibilidad de promocionar un producto para cada uno de sus restaurantes.
Si el producto promocionado de un restaurante existe, será mostrarse en la parte superior de los listados de productos que son presentado tanto a los propietarios como a los clientes. Además a ser presentado en la parte superior de la lista, el
El producto promocionado debe resaltarse visualmente, por lo que un se mostrará la etiqueta de texto "Destacado".
Regla comercial: los propietarios solo pueden promocionar un producto
para cada uno de sus restaurantes, por ejemplo un propietario puede promocionar el producto "Filete" del Restaurante1, y "Bocadillo de queso y tomate" de Restaurant2, pero nunca dos productos de Restaurant1.
Ejercicio 1: Implementar todos los cambios necesarios en el proyecto back-end para cumplir con el nuevo requisito.
Ejercicio 2: Implementar todos los cambios necesarios en el proyecto de front-end para cumplir con el nuevo requisito.



El planteamiento de este problema , creamos una propiedad nueva , en productos llamada 'promoted' , y añadir un boton switch en el create product.

Añadir una restriccion para que solo haya 1 producto promocionado por restaurante

BACKEND

Models PRODUCT.JS

```
Product.init({
  name: DataTypes.STRING,
  description: DataTypes.STRING,
  promoted: {
    type: DataTypes.BOOLEAN,
    defaultValue: false
  },

  price: DataTypes.DOUBLE,
  image: DataTypes.STRING,
  order: DataTypes.INTEGER,
  availability: DataTypes.BOOLEAN,
  restaurantId: DataTypes.INTEGER,
  productCategoryId: DataTypes.INTEGER
}, {
```

Añadimos la propiedad 'promoted'

Migration CREATE-PRODUCT

```
    },
    description: {
      type: Sequelize.TEXT
    },
    promoted: {
      type: Sequelize.BOOLEAN,
      defaultValue: false

    },
    price: {
      allowNull: false,
      type: Sequelize.DOUBLE
    },
    image: {
      type: Sequelize.STRING
    },
```

Validation PRODUCT-VALIDATION

```
const checkOnly1ProductPromoted = async (value, { req }) => {
  if (value) {
    try {
      const product = await Product.findOne({ where: { restaurantId: req.body.restaurantId, promoted: true } })
      if (product !== null) {
        return Promise.reject(new Error('Ya hay un producto promocionado en este restaurante'))
      } else { return Promise.resolve() }
    } catch (err) {
      return Promise.reject(new Error(err))
    }
  }
}
module.exports = {
  create: [
    check('name').exists().isString().isLength({ min: 1, max: 255 }).trim(),
    check('description').optional({ checkNull: true, checkFalsy: true }).isString().isLength({ min: 1 }).trim(),
    check('promoted').custom(checkOnly1ProductPromoted),
    check('price').exists().isFloat({ min: 0 }).toFloat(),
    check('order').default(null).optional({ nullable: true }).isInt().toInt(),
    check('availability').optional().isBoolean().toBoolean(),
    check('productCategoryId').exists().isInt({ min: 1 }).toInt(),
    check('restaurantId').exists().isInt({ min: 1 }).toInt(),
    check('restaurantId').custom(checkRestaurantExists),
    check('image').custom((value, { req }) => {
      return checkFileIsImage(req, 'image')
    }).withMessage('Please upload an image with format (jpeg, png).'),
```

CONTROLLER

Product-Controller

```
exports.popular = async function (req, res) {
  try {
    const topProducts = await Product.findAll(
      {
        include: [{
          model: Order,
          as: 'orders',
          attributes: []
        },
        {
          model: Restaurant,
          as: 'restaurant',
          attributes: ['id', 'name', 'description', 'promoted', 'address', 'postalCode', 'url', 'shippingCosts', 'averag
          include:
        {
          model: RestaurantCategory,
          as: 'restaurantCategory'
        }
        }
        ],
```

Añadimos la propiedad para poder llamarla en el fronted

Restaurant-CONTROLLER

```
exports.show = async function (req, res) {
  // Only returns PUBLIC information of restaurants
  try {
    const restaurant = await Restaurant.findByPk(req.params.restaurantId,
      attributes: { exclude: ['userId'] },
      include: [{
        model: Product,
        as: 'products',
        include: { model: ProductCategory, as: 'productCategory' }
      },
      {
        model: RestaurantCategory,
        as: 'restaurantCategory'
      }],
      order: [[{ model: Product, as: 'products' }, 'promoted', 'DESC']]
    }
    )
    res.json(restaurant)
  } catch (err) {
    res.status(500).send(err)
  }
}
```

Modificamos el orden en el que aparecen los productos , para que aparezcan por orden de 'promoted'

FRONTED

RESTAURANT-DETAIL-SCREEN

```
const renderProduct = ({ item }) => {
  return (
    <ImageCard
      imageUri={item.image ? { uri: process.env.API_BASE_URL + '/' + item.image } : defaultProductImage}
      title={item.name}
    >
      <TextRegular numberOfLines={2}>{item.description}</TextRegular>
      <TextSemiBold textStyle={styles.price}>{item.price.toFixed(2)}€</TextSemiBold>
      {!item.availability &&
        <TextRegular textStyle={styles.availability }>Not available</TextRegular>
      }
      <View style={{ position: 'absoluted', top: -20, right: -1050 }}>
      {item.promoted
        ? <TextRegular textStyle={{ color: 'red' }}>Promoted!!</TextRegular>
        : <TextRegular textStyle={{ color: 'red' }}>Not Promoted</TextRegular>}
      </View>
      <View style={styles.actionButtonsContainer}>
       <Pressable
        onPress={() => navigation.navigate('EditProductScreen', { id: item.id })
        }
        style={({ pressed }) => [
          {
            backgroundColor: pressed
              ? GlobalStyles.brandBlueTap
              : GlobalStyles.brandBlue
          },
```

Añadimos eso para poder ver a la derecha , en los recuadros que ponen los detalles de los productos

CREATE-PRODUCT-SCREEN

```
/>
<ErrorMessage name={'availability'} render={msg => <TextError>{msg}</TextError> }/>

<TextRegular style={styles.switch}>Promoted?</TextRegular>
<Switch trackColor={{
    false: GlobalStyles.brandSecondary,
    true: GlobalStyles.brandPrimary
}} thumbColor={values.promoted
    ? GlobalStyles.brandSecondary
    : '#f4f3f4'} value={values.promoted}
    style={styles.switch}
    onValueChange={value => setFieldValue('promoted', value)} />

<Pressable onPress={() =>
  pickImage(
    async result => {
```

Añadimos un boton switch , que indique si el promoted , es true or false

SOLUCION