Mädchenflohmarkt Android Remote Case.

Impress us with product viewer app!

We'd like to ask you to implement a little Android app that allows showing information about some items from our service. The app should support devices with API 19 or newer and supports landscape and portrait orientations. The app should have 2 screens:

- Product list. This screen displays list with products with short product' overview like name, price and concierge/not concierge item. For concierge items use red text and black for regular items. On user' tap should be opened next screen with more detailed information about the selected product.
- Product page. This screen should display at least one product image, name, description, brand, category, price, shipping cost and "open in browser" button. Also, this screen should display username and rounded user photo.

Please, use Gitlab, Bitbucket or Github service for storing your source. We would like to see git project history.

The app algorithm.

In each app start it should load root model. By path `links.products` you could find array with links to products feeds. Please, use the first link in array if it available. By this link you could find a dictionary with many properties, where:
`links.next` *– link to next product page. Please, implement infinitive loading. Could be nil (it means that loaded all elements).*
`products` – array with products. Could be empty.

Product model has many values, you could use any values but some links could return 404 error. The most interesting fields for this project:
`links.base` *– string. link to full product information. A model by this link has user data.*

`links.websiteUrl` – string. link for "open in browser". Should open the website in embedded browser or in external.
`concierge` – bool value. Required use alternative text color for product description.
`name` – string. Product name.
`description` – string. Product description.
`brand.name` – string. Brand name.
`category.name` – string. Category name.
`price` – number. Product price in EURO.
`priceShipping` – number. Product shipping cost in EURO.
`images` – an array of image object. Each image object has 3 important fields:
`width` and `height` – number. original image size. Could be used for calculation image' aspect ratio.
`url` – string. Universal image link. **Important!** You should replace strings `_WIDTH_` and `_HEIGHT_` with real required image size. E.g. for link `https://mfcdn.de/product/_WIDTH_x_HEIGHT_/tunika-bluse-h-m-cf96a9.jpeg` should be used `https://mfcdn.de/product/260x195/tunika-bluse-h-m-cf96a9.jpeg`.

Also full product model has `user` model. This model has:
`username` – string. Visible username
`photo` – string. Link to profile image. Always square. Link could be to an internal resource with `_WIDTH_x_HEIGHT_` or to external resource, e.g. facebook. This value could be nil.

Api

- root model.
  The root model with all information required for the app work. It has an array of possible feeds.
  `https://prelved.app/v1/root.json` In test project always use this url.

- products feed.
  Feed with products and other information, like pagination and other.
  `https://prelved.app/v1/:feedId.json`
  Example data https://prelved.app/v1/products.json

- Full product model.
  Product model with user information.

`https://prelved.app/v1/products/:productId.json`

Example data https://prelved.app/v1/products/3806090.json

Some option tasks.

They are not required, but it would be nice if you could find some time to do it:

- Add product image to the product list.
- Use grid layout in product's list.
- Show all available images in product detail page.
- Display any extra useful information product details.