deti universidade de aveiro
departamento de eletrónica, telecomunicações e informática

TQS: Product specification report

Project: Techflux

Group:

- Aneta Pawelec
- Gonçalo Almeida
- Pedro Candoso
- Ricardo Antão

1 Introduction

1.1 Overview of the project

This project was introduced in the course of Test and Quality of Software (TQS). The objectives were to work in a team in order to create a multi-layered, enterprise web application and apply a Software Quality Assurance strategy during the development process.

The developed application is an online marketplace named “TechFlux”, where users are able to look for electronic devices or components, buy products from other users or publish a product they offer to sell. In order to make purchases or sell products, users must first sign up and sign into the website using an email and password.

On our platform users can find multiple types of electronic devices and also single components for devices. Users are able to search the product by name or by one of the categories, in which products are divided. Once the user finds their desired product, they can add it to the shopping cart or immediately buy it.

Users who want to sell their products can fill out a form with its information and include photos, the products are automatically listed and associated with their contact information.

2 Project Concept

2.1 Vision

This product aims to provide a fair marketplace for both buyers and sellers. This means a seller will be able to sell his products at a demandable price (negotiable) anda buyer will be able to look for the offer that best suits himt, maximizing the possible profit on both sides.

It is expected to result in happy sellers who could drain their products and happy buyers who got the best bargain.

2.2 Personas

Buyer: Vicente is a 40 year old electronics repairer. He needs to find electronic components at the best price, in order to profit the most he can from each repair, keeping competitive prices against his market opponents who, like him, own small businesses.

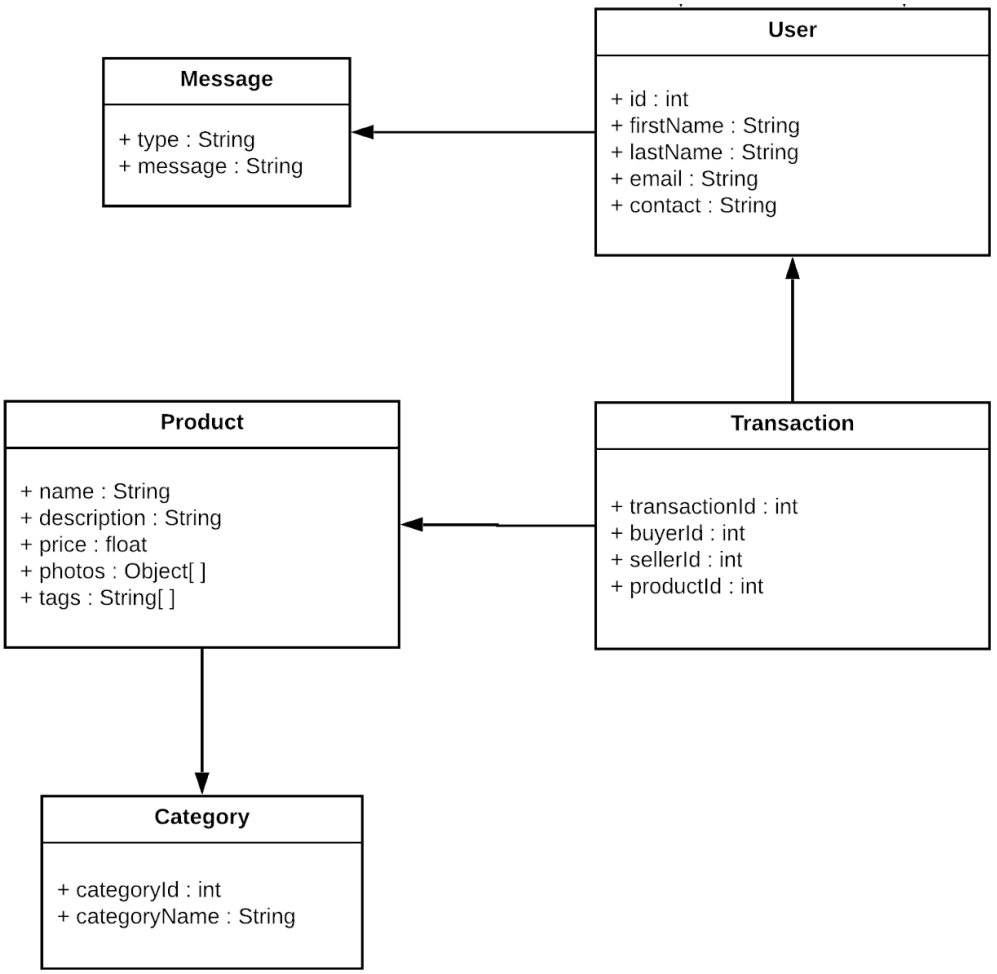
Seller: Tomás is a 31 year old businessman who owns a company that produces electronic components. He is looking for consumers for his products and he is in absence of a large deal with technology companies, therefore he is looking for a retailer where it becomes possible to unload his goods and make some profit.

2.3 Main Scenarios

The following scenarios illustrate the main ways in which our users will want to interact with the marketplace.

- Vincente searches for a product he wants to buy;
- Tomás publishes a product for sale;
- Vincente purchases a product;
- Vincente checks his transaction history;
- Tomás checks his account details and updates it (email, contact, password, etc).

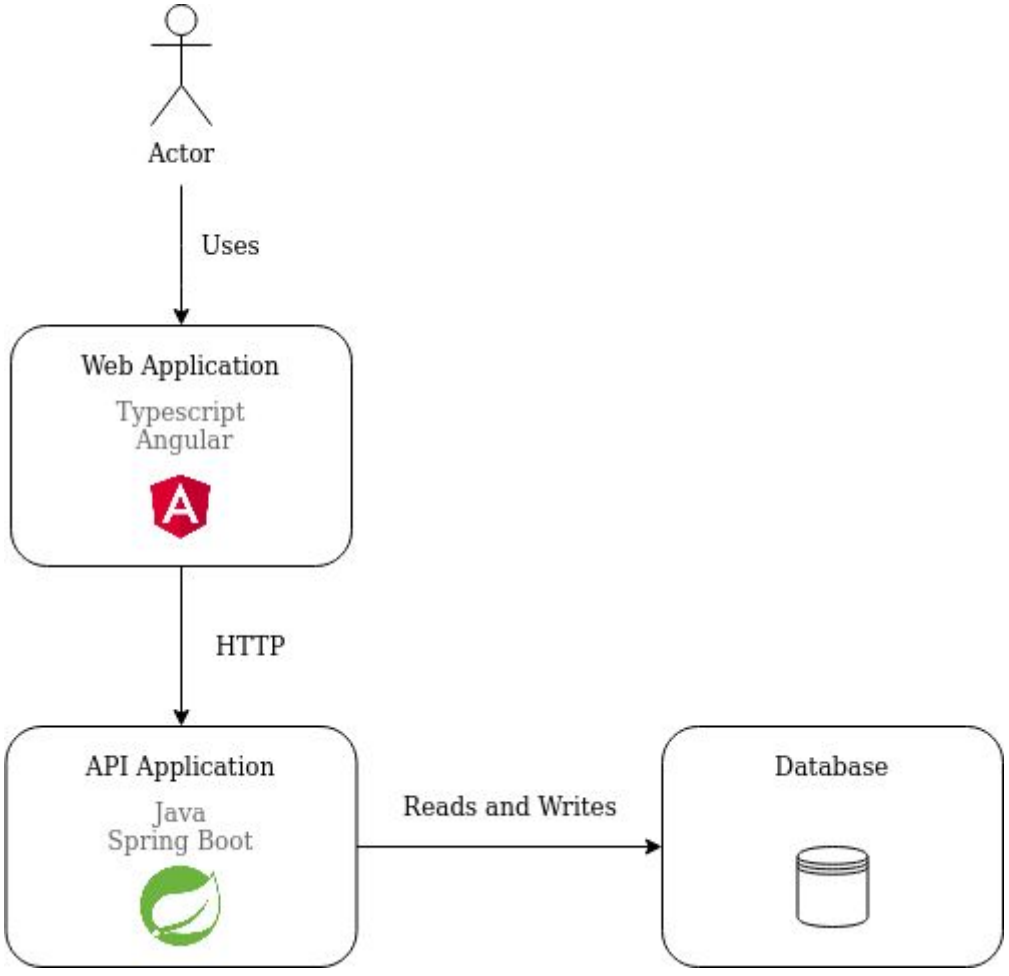
3 Domain Model



4 Architecture Notebook

4.1 Architectural view

The user is expected to interact with the system mainly through the web browser using a web page developed with the Angular framework. This application will use standard HTTP requests to access and manipulate the data on the database through the usage of a REST API developed with Spring Boot.



4.2 Deployment Architecture

The expected deployed solution requires two key components, the frontend client developed with the Angular framework, and the backend which comprises the Spring Boot API, business logic and database storage. These components will be hosted on the cloud platform Heroku separately and each will run on a single machine.

5 API for developers

AuthController:
/auth
POST: /auth/authenticate: logs in an user;

CategoryController:
/categories
GET: /categories/all: get a list of all categories;
GET: /categories/{categoryName}: get a category by its name;
GET: /categories/id/{categoryId}: get a category by its category ID;

ProductController:
/products
GET: /products/all: get a list of all products;
GET: /products/image/{productId}:
GET: /products/query/{productName}:
GET: /products/id/{productId}: get a product by its product ID;
POST: /products/add: used to add a new product;
PUT: /products/update: used to update a product;

TransactionController:
/transactions
GET: /transactions/{userId}/all: get a list of all transactions that involves a user by its user ID;
GET: /transactions/{userId}/id/{transactionId}: get a transaction that involves a user by its user ID and transaction ID;
POST: /transactions/add: used to add a new transaction;

UserController:
/users
GET: /users/{userPartialName}: get a user by their partial name;
GET: /users/id/{userId}: get a user by their user ID;
GET: /users/get: gets the user who owns a token;

UserinfoController:
GET: /me: current user details;

PUT: /update/details: update current user details;
PUT: /update/password: update current user password;
POST: /new: create a new user;

6 References and resources

Angular application on GitHub - The repository used for the frontend application
Spring application on GitHub - The repository used for the backend application
PivotalTracker - Used for project management and task assignments
Deployed product - the Angular application, deployed on Heroku
Deployed API - Spring Boot API, deployed on Heroku