An Android Application of a Canteen Ordering System

**Interim Report**

Despina Dorotheou (dd186)

Table of Contents

[Aims: 2](#_Toc531789677)

[Objectives: 2](#_Toc531789678)

[Survey of Literature/Information Sources: 3](#_Toc531789679)

[Description of Prototype: 4](#_Toc531789680)

[Software Architecture: 7](#_Toc531789681)

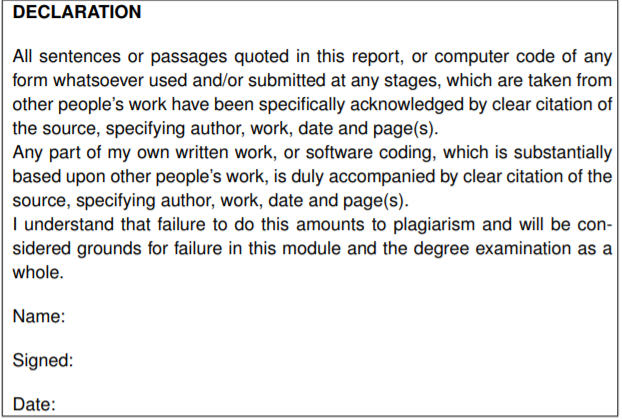
[Planning and Timescales: 8](#_Toc531789682)

[Completed Requirements: 9](#_Toc531789683)

[Remaining Requirements: 9](#_Toc531789684)

[Bibliography: 11](#_Toc531789685)

The first thing that I did was, a background research on the Android applications. Since I have never been involved with Android applications before, I wanted to get more familiar with the Android SDK. In order to do that, I followed some tutorials and I created a simple, hello-world style Android application.[[1]](#footnote-1) I also researched similar applications to the one I am developing currently, to get an idea about their functionalities and their designs. The research also helped me to figure out things that I would like to avoid or do better in my application



05/12/2018

Despina Dorotheou

Despina Dorotheou

# Aims:

My main goal for this project is to make a user-friendly and helpful Android application of a canteen ordering system.

I aim to learn as many new things as possible about designing, implementing and testing an Android application, as well as making sure there is a secure way of payment through it.

I intend to make an easy to use web application where administrators without any MySQL or Computer Science knowledge can add, delete and change information in the database, which will be connected to the Android application. Also, I intend to deploy both applications to the cloud for scalability purposes.

Moreover, I intend to improve my problem-solving skills by tackling any new possible challenges that might come up during the project. I also would like to improve my time management skills since it is a project that is going to take limited time, and there will be many challenges that will need to be accomplished during that period.

# Objectives:

To achieve my goals, firstly, on the one hand, I needed to acquire familiarity with Android SDK which entailed understanding concepts of Android and using said concepts programmatically with Java. Also, I needed to improve my knowledge regarding XML for the app design. On the other hand, I needed to do deeper research on Spring boot Web applications as well as on RESTful API which is based on representational state transfer technology that helped me to force these two applications (Android and Web) to communicate with each other and share information, more specifically, database information, through [HTTP](https://searchwindevelopment.techtarget.com/definition/HTTP) requests.

Secondly, I will need to do a market research, collect valuable information that will help me find out whether there is a market for the product/service (application) that I intend to develop during this project and will help me make some the correct business decisions. In addition, I will need to do a research on similar applications, find some new ideas, figure out how they work and what things I should do or avoid.

Thirdly, I will need to be able to make the best decisions possible regarding any new challenges that might come up while working on the project.

Finally, I want to make my application easy to use with many helpful and functional features, so every student and staff of the university would need to have it on his / her phone.

# Survey of Literature/Information Sources:

The first thing that I did was, a background research on the Android applications. Due to the fact that I have never been involved with Android applications before, I wanted to get more familiar with the Android SDK. In order to do that, I followed some tutorials and I created a simple, hello-world style Android application.[[2]](#footnote-2) I also researched similar applications to the one I am developing currently, to get an idea about their functionalities and their designs. The research also helped me to figure out things that I would like to avoid or do better in my application. Some of these applications were the Upay[[3]](#footnote-3) and the Uber Eats[[4]](#footnote-4). I researched the Upay application because the university is currently using it to share their canteen’s promotions and because it helps the users to pay for their meals faster at the counter. The reason I researched Uber Eats application is because of its design, I believe is very simple and user-friendly, so it inspired me to do something similar.

Furthermore, I researched Spring boot Web applications despite developing a simple Spring boot Web application before. I watched a course-type tutorial in order to get some help regarding which frameworks to choose, i.e. code libraries, tool sets, application programming interfaces etc, and dependencies I was going to use.[[5]](#footnote-5)

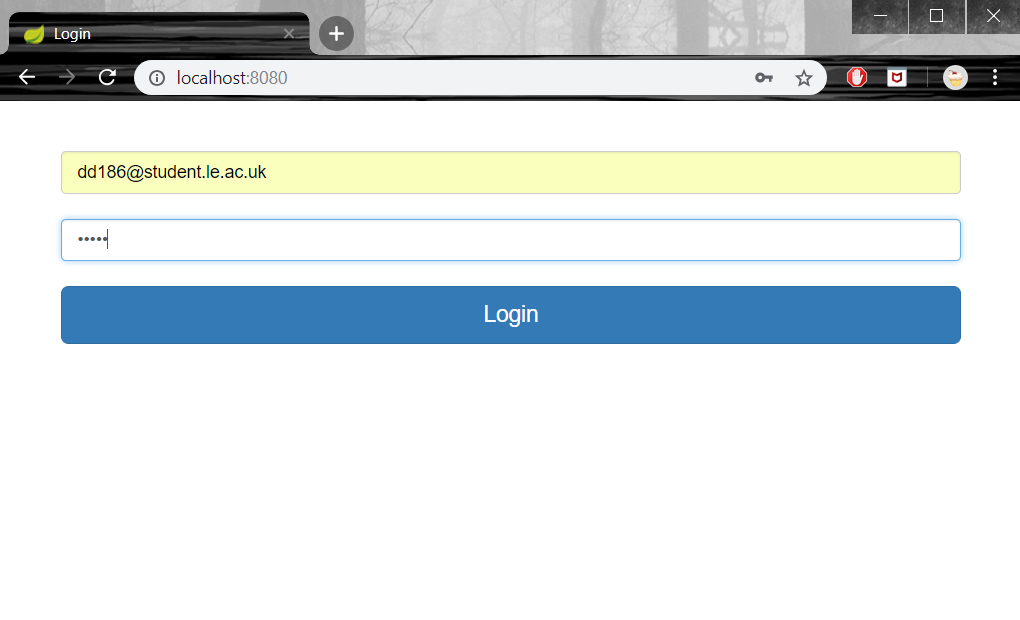
In addition, I did a background research on how to connect my database to the system. At the beginning, I was researching on how to connect the database directly to the Android application but then I decided to connect the database to the web application and the web application to the Android one, using the RESTful API. I chose the RESTful API mainly because of its client-server principle – “By separating the user interface concerns from the data storage concerns, we improve the portability of the user interface across multiple platforms and improve scalability by simplifying the server components.”[[6]](#footnote-6).

Finally, the last research I did was on RESTful API concerning how to use it in order to send HTTP requests from an Android app to a web application but also how to handle the responses. I watched a course-type tutorial explaining how the HTTP requests and responses work and how to connect the two applications.[[7]](#footnote-7) Later, I did a research on how to serialize and deserialize objects since I wanted to send the list of products (objects) from the web application to the Android application.[[8]](#footnote-8)

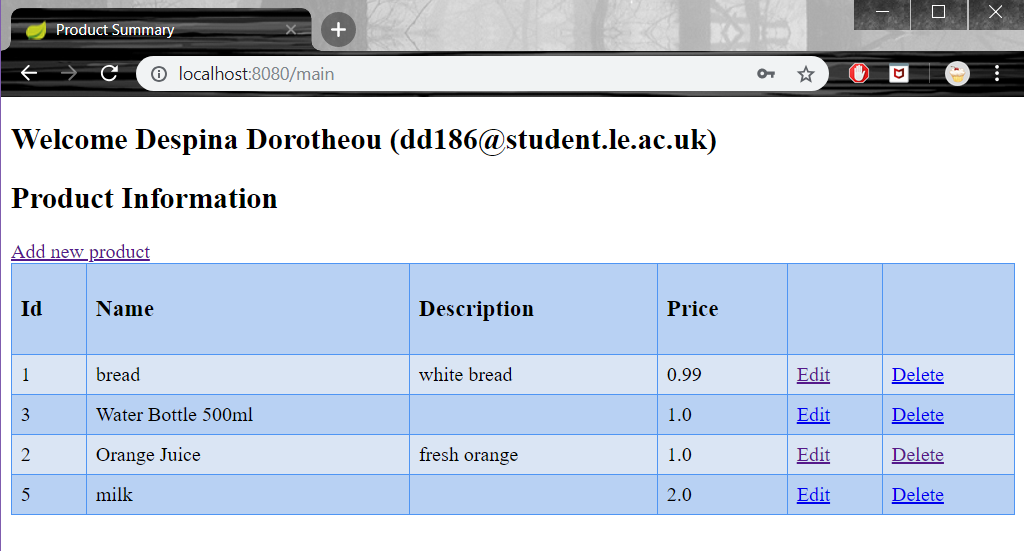
# Description of Prototype:

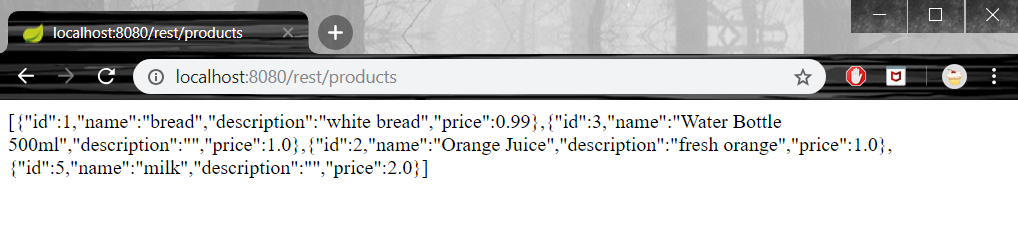
My project’s prototype consists of an Android application and a Spring boot Web application.

The web application has the login page where only the administrators can login in to, using their already provided email and password (*R: The admin won’t have to sign up, his /her account will be given & R: The admin will be able to login into a web application*).

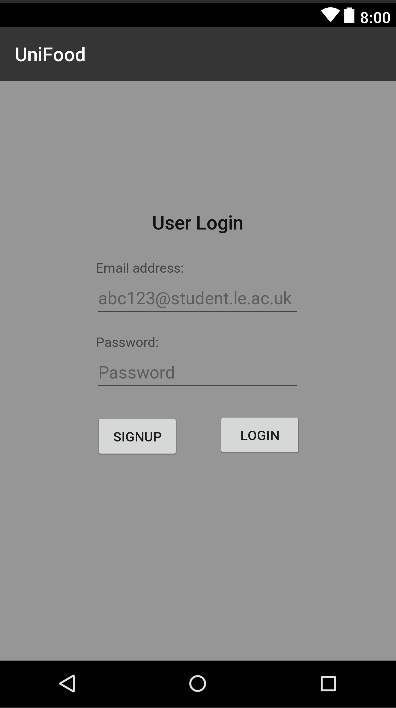
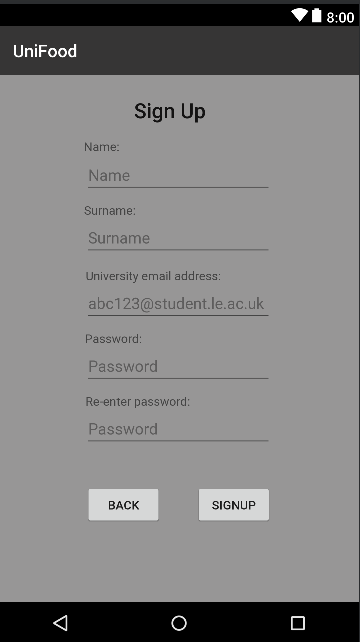


There is also the main page where the administrators can see (*R: The admin will be able to see a table of the products*) and modify the products’ table, for instance, they can add, delete or edit a specific product in the table and the database is automatically updated (*R: The admin will be able to add new products & R: The admin will be able to delete products & R: The admin will be able to update the description of the product & R: The admin will be able to update the price of the product*).



The web application is also used to respond to the Android application’s HTTP requests. In other words, the Android application will send requests for functionalities, for example, login and sign up with the user’s data. Moreover, the web application will check if the user exists in the database in case the user wants to login. In another case, where the user wants to sign up, the web application will create a new user account (*R: The app will send HTTP request to check the user’s data for the login & R: The app will send HTTP request with the new user’s data to create a new account for the sign up*). I intend to use HTTP requests in order to take more data from the database like products, deals and categories.

The android application has the login page somewhere that both users and administrators can login into and the sign-up page where non-registered users can create an account (*R: The user must login into his /her account & R: The user will be able to create an account through the sign up page*). When the user is trying to login or sign up, the application sends a HTTP request and waits for a response. When this situation occurs, the application waits for an “ok” response and as soon as it gets it, the main page is displayed for login or the login page is displayed for sign up (*R: Respond to HTTP request for login with “ok” if the user exists and respond with “Invalid” if the user does not exist in the database & R: Respond to HTTP request for sign up with “ok” if the user account was created successfully and respond with “Already existing user” if the user exists in the database*). Non-registered users cannot login until they register. After the user is logged in, she / he can see the main page which consists of a bottom-navigating menu for home, search / categories and account. In the home fragment there will be displayed all the products and deals which the user can easily see by scrolling down. Also, in the search fragment there will be a search bar and certain categories where the user will be able to find a specific product under a certain category or by typing keywords in the search bar. In the account fragment, the user has the choice to see her / his order history, his / her favourite products, his / her current order status and log out. For every choice in the home fragment there is a corresponding, empty, at the moment, fragment.

# Software Architecture:

Android Application

RESTful

API

MySQL

Spring Boot Web Application

HTTP request

HTTP response

My software system consists of three components, MySQL, the Spring Boot Web application and the Android application. The web application is directly connected with the MySQL database, which will be used by the whole software system. At the prototype state, there are four tables in the database, the “user” table, the “role” table, a “user\_role” table that stores the user id and the corresponding role id, and finally the last table is for the products.

The web application has two purposes, the first is to serve the administrators so they can modify the data and the tables in the database automatically. The second is to receive HTTP requests from the Android application and respond back with the information that is been requested, to do so, the web application must serialize[[9]](#footnote-9) the information. This is achieved using the RESTful API whose Uniform interface principle benefits the whole system architecture – “By applying the software engineering principle of generality to the component interface, the overall system architecture is simplified, and the visibility of interactions is improved. In order to obtain a uniform interface, multiple architectural constraints are needed to guide the behaviour of components. “[[10]](#footnote-10).

The Android application is the main component of the software system and is the only part that interacts directly with the user. It takes its data from the web application by sending the appropriate HTTP requests. When the web application sends back the response to the request, the Android application takes the response, deserializes[[11]](#footnote-11) and stores the information that is needed.

# Planning and Timescales:

While I was working on the prototype, I came across with some unexpected challenges that caused certain changes in the initial plan.

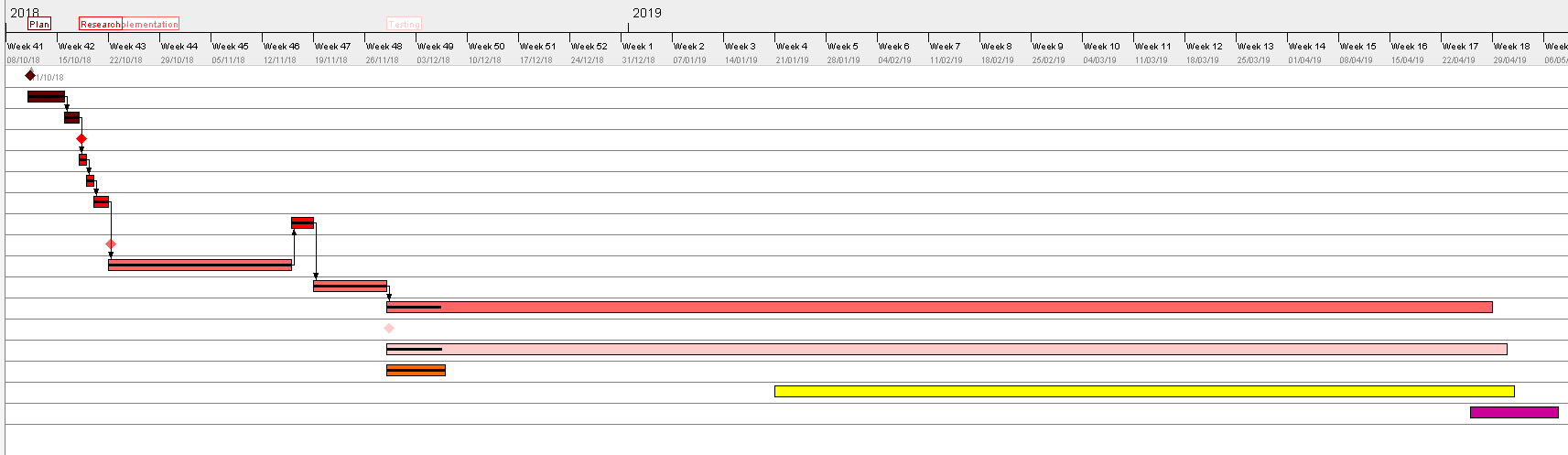
At the beginning, I did not plan to develop a Spring boot Web application for the administrators, which is the main reason I did not mention anything about the research in the initial plan. After I developed the web application, I had to connect the two together, something that I also did not plan initially. Therefore, I had to conduct a research on the RESTful API. Moreover, these plan changes had an immediate effect on the whole prototype since I completed less requirements for the Android application than what I initially expected.

I believe that despite having trouble with specific challenges that occurred, I managed to develop the core of the project for the prototype and I am confident about any further steps.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Begin date | End date | Status |
| Plan | 11 / 10 / 18 | 18 / 10 / 18 | Completed |
| * Purpose | 11 / 10 / 18 | 15 / 10 / 18 | Completed |
| * Project Planning | 16 / 10 / 18 | 17 / 10 / 18 | Completed |
| Research | 18 / 10 / 18 | 19 / 11 / 18 | Completed |
| * Market Research | 18 / 10 / 18 | 18 / 10 / 18 | Completed |
| * Research on similar apps | 19 / 10 / 18 | 19 / 10 / 18 | Completed |
| * Research on android apps | 20 / 10 / 18 | 21 / 10 / 18 | Completed |
| * Research on spring boot apps | 22 / 10 / 18 | 23 / 10 / 18 | Completed |
| * Research on Restful API | 16 / 11 / 18 | 18 / 11 / 18 | Completed |
| Implementation | 22 / 10 / 18 | 29 / 04 / 19 | Completed |
| * Prototype | 22 / 10 / 18 | 15 / 11 / 18 | Completed |
| * Final Prototype Changes | 19 / 11 / 18 | 28 / 11 / 18 | Completed |
| * Finalise SW system | 29 / 11 / 18 | 28 / 04 / 19 | Started |
| Testing | 30 / 11 / 18 | 01 / 05 / 19 | Started |
| * SW testing | 30 / 11 / 18 | 30 / 04 / 19 | Started |
| Interim Report | 29 / 11 / 18 | 06 / 12 / 18 | Completed |
| Dissertation | 21 / 01 / 19 | 01 / 05 / 19 | Unstarted |
| Presentation | 26 / 04 / 19 | 07 / 05 / 19 | Unstarted |

Completed Tasks

Remaining Tasks



Due to the background research I did on the Android and the Spring Boot Web applications, I had to change some of my project’s requirements and add more in order to have a better idea than I did before about the whole software system that I am about to develop.

## Completed Tasks /Requirements:

User/Android app

1. The user must login into his /her account.
2. The user will be able to create an account through the sign up page.
3. The app will send HTTP request to check the user’s data for the login.
4. The app will send HTTP request with the new user’s data to create a new account for the sign up.

Admin/Web app

1. The admin won’t have to sign up, his /her account will be given.
2. The admin will be able to login into a web application.
3. The admin will be able to see a table of the products.
4. The admin will be able to add new products.
5. The admin will be able to delete products.
6. The admin will be able to update the description of the product.
7. The admin will be able to update the price of the product.
8. Respond to HTTP request for login with “ok” if the user exists and respond with “Invalid” if the user does not exist in the database.
9. Respond to HTTP request for sign up with “ok” if the user account was created successfully and respond with “Already existing user” if the user exists in the database.

## Remaining Tasks /Requirements:

User/Android app

1. The user will be able to create an account if he /she is student or staff of the university. (Can sign up only with the university address)
2. The app will send HTTP request to take all the products from the database.
3. The app will send HTTP request to take all the deals from the database
4. The user will be able to see a list of the products.
5. The user will be able to see daily /weekly offers.
6. The user will be able to see recommendations by the application based on his/her order history.
7. The user will be able to see what products are trending at the time.
8. The user will be able to see the availability of the items.
9. The user will be able to see a description of the items including their ingredients.
10. The user will be able to see his /her order history.
11. The user will be able to search through a list of options based on the ingredients.
12. The user will be able to search for meals under different food categories.
13. The user will be able to set dietary constraints. (vegan, vegetarian etc.)
14. The user will be able to set cost constraints.
15. The user will be able to add items in the basket.
16. The user will be able to remove items from the basket.
17. The user will be able to change the quantity of an item in the basket.
18. The user will be able to pay for their order with credit /debit card through the app.
19. The user will be able to select the cash option at the checkout.
20. The user will be able to add items in his /her order, after the checkout, for a limited time.
21. The user will be able to remove items from his /her order, after the checkout, for a limited time.
22. The user will be able to delete his /her order, after the checkout, for a limited time.
23. The user will be able to receive a confirmation of his /her order.
24. The user will be able to receive a notification saying that his /her order is been processing and he /she cannot make any changes.
25. The user will be able to receive a notification, if he /she updated his /her order after checkout, saying that the changes he /she made are approved.
26. The user will be able to receive a notification saying that his /her order is finished and a unique order number.
27. The user will be able to add ranking on the items.
28. The user will be able to add a review with his /her ranking on the items.

Admin/Web app

1. The admin will be able to see a table of the current offers.
2. The admin will be able to add offers for the day /week.
3. The admin will be able to remove offers for the day /week.

Bibliography:

*Create an Android project.*Available at: <https://developer.android.com/training/basics/firstapp/creating-project> (Accessed: Dec 5, 2018).

*Creating Your First Spring Boot Application.*Available at: <https://app.pluralsight.com/library/courses/spring-boot-first-application/table-of-contents>.

*Accessing data with MySQL.*Available at: <https://spring.io/guides/gs/accessing-data-mysql/> (Accessed: Dec 5, 2018).

*Bottom navigation.*Available at: <https://material.io/design/components/bottom-navigation.html#behavior> (Accessed: Dec 5, 2018).

*Building a RESTful Web Service with Spring Boot Actuator.*Available at: <https://spring.io/guides/gs/actuator-service/> (Accessed: Dec 5, 2018).

*Fragments.*Available at: <https://developer.android.com/guide/components/fragments> (Accessed: Dec 5, 2018).

*HTTP networking.*Available at: <https://classroom.udacity.com/courses/ud843/lessons/ca00b576-ec4d-4d0f-939e-6d28f55faf7b/concepts/2914a2ea-7f15-4915-8113-d0be6e023526>.

*Introducing JSON.*Available at: <https://www.json.org/>.

*Jackson 2 – Convert Java Object to / from JSON.*Available at: <https://www.mkyong.com/java/jackson-2-convert-java-object-to-from-json/>.

*Start Here.*(2018) Available at: <https://www.baeldung.com/start-here> (Accessed: Dec 5, 2018).

*What is REST – Learn to create timeless RESTful APIs.*Available at: <https://restfulapi.net/> (Accessed: Dec 5, 2018).

*What is RESTful API? - Definition from WhatIs.com.*Available at: <https://searchmicroservices.techtarget.com/definition/RESTful-API> (Accessed: Dec 5, 2018).

*Accessing Data with JPA.*Available at: <https://spring.io/guides/gs/accessing-data-jpa/> (Accessed: Dec 5, 2018).

*Developer Guides.*Available at: <https://developer.android.com/guide/>(Accessed: Dec 5, 2018).

*Spring 4 MVC REST Controller Example (JSON CRUD Tutorial).*Available at: <https://viralpatel.net/blogs/spring-4-mvc-rest-example-json/>.

1. https://developer.android.com/tra -project [↑](#footnote-ref-1)
2. First Android App tutorial. Available at: <https://developer.android.com/training/basics/firstapp/creating-project> [↑](#footnote-ref-2)
3. Upay. Available at: <https://www.upay.co.uk/> [↑](#footnote-ref-3)
4. Uber Eats. Available at: <https://www.ubereats.com/en-GB/> [↑](#footnote-ref-4)
5. First Spring Boot App tutorial. Available at: <https://app.pluralsight.com/player?course=spring-boot-first-application&author=dan-bunker&name=spring-boot-first-application-m1&clip=0&mode=live> [↑](#footnote-ref-5)
6. *WHAT IS REST API*. (n.d.). Retrieved from RESTful API Tutorial: <https://restfulapi.net/> [↑](#footnote-ref-6)
7. The course-type tutorial for HTTP requests. Available at: <https://classroom.udacity.com/courses/ud843/lessons/ca00b576-ec4d-4d0f-939e-6d28f55faf7b/concepts/2914a2ea-7f15-4915-8113-d0be6e023526> [↑](#footnote-ref-7)
8. <http://appsdeveloperblog.com/java-into-json-json-into-java-all-possible-examples/> [↑](#footnote-ref-8)
9. Serialization is a mechanism of converting the state of an object into a byte stream. [↑](#footnote-ref-9)
10. *WHAT IS REST API*. (n.d.). Retrieved from RESTful API Tutorial: <https://restfulapi.net/> [↑](#footnote-ref-10)
11. Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory. [↑](#footnote-ref-11)