U.P.B. Sci. Bull., Series C, Vol. 84, Iss. 4, 2022 ISSN 2286-3540

PADDY - A DIGITAL ASSISTANT FOR MEDICAL PRESCRIPTIONS

Flavia Oprea[1], Teodora Costinescu[1], Teodor Mutu[2], Daniel Rosner[3]

In the previous two decades, the pharmaceutical industry has experienced remarkable growth, which has lead to increased drug consumption. Therefore, it has gotten significantly more critical to understand how they work. Sometimes, taking many medications together can cause fatal side effects. This paper aims to develop a platform that alerts users about potentially harmful drug interactions. The purpose of this application is to provide the patient with an easily accessible drug storage and a better understanding of the interactions between the medicine supplied. On the other hand, the doctors can access a patient's drug history with ease, update the current medical prescription and check interactions between various drugs. Users of the application can write and distribute prescriptions in addition to discovering drug interactions. Both the author and the medical staff have the authority to change these prescriptions.

Keywords: pharmacovigilance, drug-to-drug interaction, medical platform

1. Introduction

The pharmaceutical industry has experienced tremendous growth during the past ten years. For instance, developing a COVID-19 vaccine in less than a year, making significant strides toward curing HIV and AIDS, and having the second HIV patient make a full recovery [1]. Immunotherapy [2], a cancer treatment that strengthens the immune system, was another achievement in the last decade. Understanding and reading the leaflet are more crucial than ever given the increase in drug consumption. Despite this, very few people actually read the medical prospect [4]. For leaflets provided with new medications, 49.2%, 21.2%, 16.0%, and 13.7% reported reading the leaflets always, often, seldom, or never, respectively [3]. Additionally, 63.8% and 35%, respectively, of the patients said the material was extremely beneficial or moderately

[1]Teaching Assistant, University POLITEHNICA of Bucharest, Romania, e-mail `flavia.oprea@upb.ro`

[2]Student, University POLITEHNICA of Bucharest, Romania, e-mail `teodora.costinescu@stud.acs.upb.ro`

[3]Student, University POLITEHNICA of Bucharest, Romania, e-mail `teodor.mutu@stud.acs.upb.ro`

[4]Lecturer, University POLITEHNICA of Bucharest, Romania, e-mail `daniel.rosner@upb.ro`

useful [3]. In other words, the leaflet provides pertinent information regarding drug use, information that could help avoid the estimated 1.5 million medical errors that occur each year in the United States alone [9]. Approximately 30% of the unpredicted adverse events are those caused by drug-drug interaction.

The leaflets contain essential information about how the drug is going to affect someone's body, yet those pamphlets are filled with complex and specific medical terms which are not suited for everyone [7]-[8]. For example, advil (ibuprofen) is commonly used and does not require any prescription, but it has a lot of unwanted interaction with different drugs, one of them being spironolactone, which is a potassium-saving diuretic used mostly by the elders. This leads to adverse reactions that can cause pain and in some unfortunate cases, even the death of the consumer [5].

The objective of this research is to create a platform that can help prevent medical errors from happening, especially those regarding the interaction between different drugs, by simplifying the content of leaflets. Instead of showing the full leaflets of each drug individually, the platform will display short messages if the provided pharmaceutical products may have adverse reactions. Furthermore, the users will be allowed to store their prescription, check the interactions between drugs in it, allow an authorized party to modify it and share it without the need of an email client.

2. **State of the Art**

The National Agency for Medicines and Medical Devices of Romania provides an online platform [10] where all the drugs that are available to be commercialized in Romania can be found in the section "Medicamente de uz uman" and "Nomenclatorul medicamentelor pentru uz uman". The approach is not simplifying the content of leaflets for the common user (non-professionals) to have a better understanding in regard with the drugs that are prescribed. This solution is more suited as an alternative to the printed leaflets that can be lost or deteriorated.

Mediately is a company operating in the Central and Eastern Europe that helps doctors with drug information. The application "Lista Medicamantelor" [11] is a web platform that allows users to search information about a drug by the brand name, active ingredient, or the Anatomical Therapeutic Chemical code, search information about a disease using ICD-10 (International Classification of Diseases, Tenth Revision, Clinical Modification), list all the available instruments and use them. The application displays a multitude of interactions in the form of tables with two columns, one for the drug category and the other for the negative effects. They are difficult to follow, especially when they split without translating the text from the preceding row to the new row. The amount of information is dense with medical terms, making it difficult for non-professionals to understand.

Drugs.com [13] is a web platform that provides users various tools such as pill identifier, side effects, drug-to-drug interactions checker, treatment guides, comparison of drugs and many more. The interaction checker information is split into two categories: consumer and professional. The material for consumers is easy to understand and gives indications on how the drugs interact with one another, a list of actions that you can follow to be sure that everything is under control. Apart from the drug interactions the application provides interaction between drugs and food, informing the users what aliments to avoid or consume. However, the report is filled with complex information that is suited to trained personnel, alongside with studies that are relevant to the found interaction. Another important part of the application is the ability to share the found interactions. The downside to this solution is the need of an email client installed on the user machines.

3. **Proposed Solution**

The N-tier architecture or the multilayer architecture is a client-server architecture in which the presentation, the application processing and the data management functions are physically and logically separated. Another important aspect of this architectural pattern is that the layer above can access resources from the layer below, but the other way around is impossible. Considering that, the tiers can easily be hosted on several machines or clusters ensuring that services are provided without resources being shared. The widespread use of multilayer architecture is represented by the three-tier architecture [6].

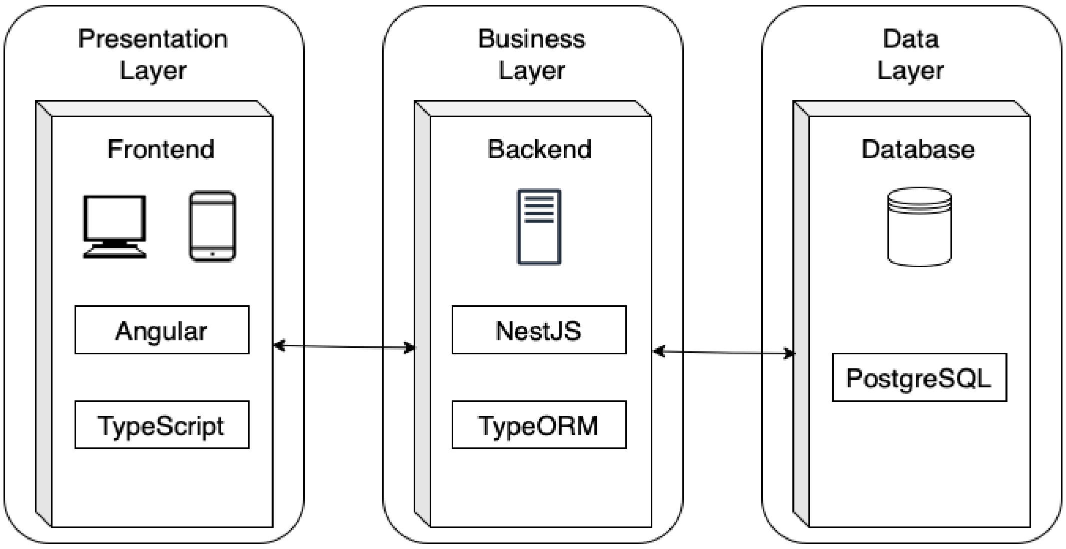


FIGURE 1. Architecture of Proposed Solution

The Presentation tier is the first layer present, and it has the responsibility to display information related to the business logic. This tier can communicate with the business layer directly and displays the result to the user. The Business tier is the second layer, and it contains the application logic. The Data tier is the last layer, and it includes the data persistence mechanism such as database server as well as the data access layer that should allow the retrieving of data.

The three layers of the architecture will be split into three servers each handling its own function. The first server will oversee the presentation layer, the second server will contain the business logic while the third will be responsible for data storage, as shown in Figure 1.

4. **Frontend Implementation**

The design for the main screen of the PADDY platform can be visualized in Figure 2. In this chapter we will briefly describe the technologies that were used in order to achieve our platform's design and frontend development.

To implement the presentation layer, the technology used is Angular 2+. Angular is a Typescript open-source application framework developed by the Angular team from Google and individuals for single page applications. The framework architecture relies on eight building blocks, which are: modules, components, templates, metadata, data binding, directives, services, and dependency injection.

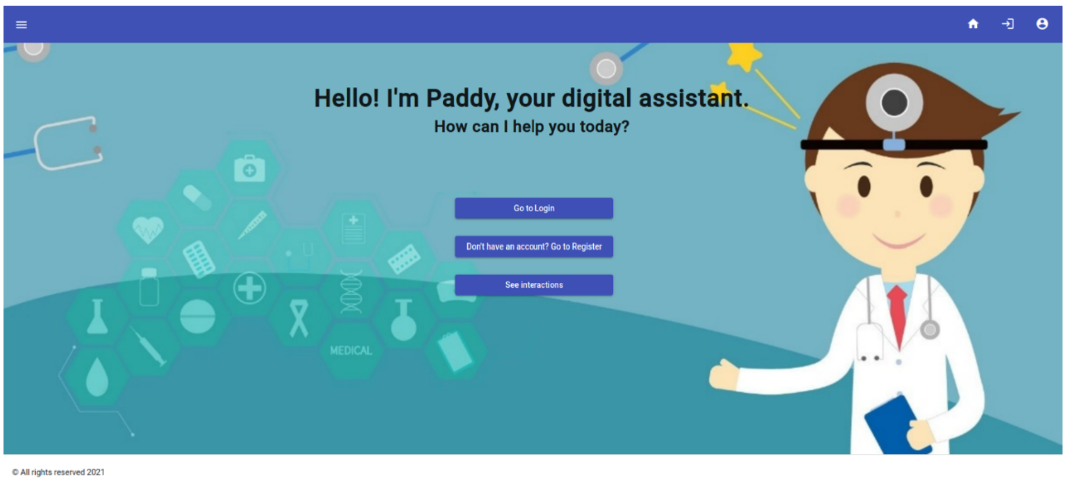

Hello! I'm Paddy, your digital assistant.
How can I help you today?
Go to Login
Don't have an account? Go to Register
See interactions
MEDICAL
© All rights reserved 2021

Figure 2. Interface of the Home Screen

Modules are a cohesive set of functionalities that focus on a specific application need, such as user workflow, routing, or forms. It contains information about components that are specific to it inside the declaration array and imports a list of modules that are needed inside the modules. Additionally, it provides an array of services that are to be injected into the components of the modules and exports an array of elements that will be usable by other modules when importing the source module.

Components are the fundamental units of each Angular application. Each component defines the class that contains application data and logic, and it is associated with the HTML template. A template is a combination of HTML and Angular markup that can modify HTML elements before they are displayed. Metadata is another important feature that tells Angular how to process a class and it can be attached by using a decorator.

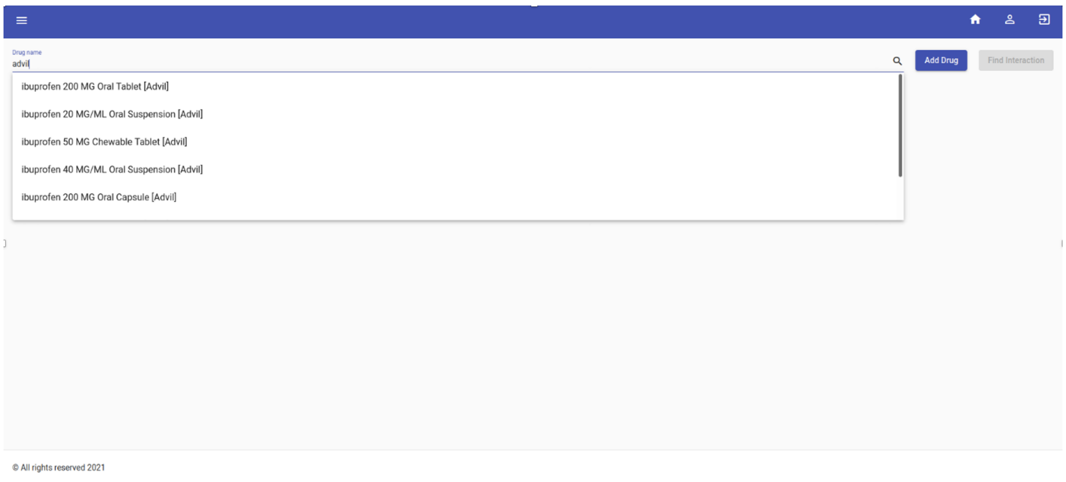
Data binding is an Angular communication mechanism between a template and its component. There are 2 types of bindings: one-way bindings, which go from either the component to the template (property binding), or vice versa (event binding), and two-way bindings, which connect the component to the template in both ways.

Directives are classes that inject additional behavior into elements inside the application. Services are classes that do not hold logic for a particular view and that need to be shared among different components. Dependency injection is a design pattern in which a class requests dependencies from external sources rather than creating them. The reasons for choosing Angular were the modularization of the application and its easy-to-understand structure.

The three important elements that make up the NestJS architecture are controllers, providers, and modules. A controller handles receiving requests and sending responses back to the client. The server accepts HTTP requests, and the configured routing mechanism distributes the requests to the appropriate controller. The role of modules is to group components in order to divide the entire application into independent elements that can communicate with each other. Each program has a hierarchy of modules; the hierarchy is composed of at least one module. The core module is the one that deals with drawing a graph to link and resolve dependencies and relationships between providers and modules.

To abstract the relationship with the PostgreSQL database, the concept of Object Relational Mapping (ORM) was used. It helps to manipulate and query data through an object-oriented paradigm. Through this mechanism, you no longer have to write SQL requests directly in the code. Each table can be represented by a class in the programming language of your choice. The advantages of ORM come from reusing code and data and simplifying create, insert, update, and delete (CRUD) operations. TypeORM is the concept introduced by the NodeJS community regarding the connection between objects and entities in the database.

TypeScript can be used for both front-end and back-end development. It contains patterns, and problems in the code are easier to solve because the types of the variables are known. This language also intervenes in error handling by announcing type mismatches through error messages. Being an open-source and popular programming language, development environments (IDEs)



ibuprofen 200 MG Oral Tablet [Advil]
ibuprofen 20 MG/ML Oral Suspension [Advil]
ibuprofen 50 MG Chewable Tablet [Advil]
ibuprofen 40 MG/ML Oral Suspension [Advil]
ibuprofen 200 MG Oral Capsule [Advil]

© All rights reserved 2021

Figure 3. Appearing Screen and Results Returned when Searching for the Medication "advil"

offer developers many features in order to facilitate people's work: displaying information about an object when dragging the mouse over it, autocompletion of some code segments, type checking. Another advantage of using this language is the good integration with the back-end for this project. Both parts of the program are based on TypeScript, which becomes the common component between them.

In Figure 3 we can observe how the results of a drug term are displayed when being typed or searched. In the example depicted in Figure 3 the user searches for the term "advil". Details on how the results are found, retrieved, and how the drug-to-drug interaction is identified will be explained in the section Backend Implementation.

To manage the styling of user interfaces in an easier way, Angular proposes a library called Angular Material. It helps to quickly create visually pleasing, intuitive to use web pages or applications that can be ported and used on any device. The following components can be created through this library: buttons, dialog boxes, autocomplete, forms, icons, menus, tables and many other elements.

5. Backend Implementation

5.1. Technologies Used

The business layer is a NestJs-based REST API (representational state transfer). Nest is a Node.js application framework that has complete Typescript support while also allowing developers to code in plain JavaScript. Nest is built on top of two dependable HTTP Server frameworks: ExpressJs and Fastify. The resemblance between Nest and Angular, with various architectural components acting in the same way, was the basis for picking Nest to develop the application. Instead of relying on components, Nest includes controllers that perform comparable functions. The modules, services, and dependency injection work precisely like their Angular counterparts.

TypeORM, an ORM (Object Relational Mapping) framework, is another technology employed by the business layer. An ORM is a tool that implements a strategy that allows developers to use an object-oriented paradigm to query and manipulate data from a database. A migration is a file that contains information on a minor incremental, reversible change to the schema of a relational database. A schema migration is performed on a database anytime an upgrade or reversion to a previous or newer version is required. TypeORM's rich command line simplifies the migration process, making development in the process straightforward.

The relations between tables in the database architecture are:

- User to Role relationship - is needed in order to associate a user with a single role hence the relation is many-to-one
- Role to Role-permissions - this relation associates a role with multiple entries in the role_permissions table, this relation is one-to-many. However, the relation could be one-to-many directly with Permission table, in this case there would be multiple duplicate entries
- Role-permissions to Role - the sole purpose of this relation is to avoid duplicate entries as explained in the Role – Role-permissions relation.
- User to Medication - this relation associates a user with multiple entries from the medication table, the nature of this relation is one-to-many
- Medication to Drug - This relation associates a medication with multiple drugs, it is a one-to-many relation

The application's features are divided into three controllers: "UserController," "RoleController," and "MedicationController," each with its own set of capabilities. The logic for the following functionalities is contained in the "UserController": authentication, authorization, user creation, user deletion, and user modification. "RoleController" ensures that role-based access control is maintained by allowing users to create, update, and delete roles. Finally, the "MedicationController" logic is responsible for creating, modifying, and sharing prescriptions.

5.2. Role-Based Access Control

RBAC is a method of restricting system access to authorized users that is based on roles and privileges. This mechanism is used by the backend server to restrict authorized users' access to endpoints based on their role. A role is a bundle of permissions that specify a user's access to a database table. There are four actions that can be performed on a table: insert, read, update, and delete, which translate into four permissions:

- Create needed to be able to create entries into the database. E.g., "create-user"

• List needed to get rows from the database. E.g., “list-user”
• Modify needed to change rows in the database. E.g., “modify-user”
• Delete needed to remove rows from the database. E.g., “delete-user”

To implement this feature, NestJs has a mechanism of guards. A guard is a function that will be executed before the logic of an endpoint. The server has two guards: authentication and permission.

The authentication guard is using the PassportJs strategy to verify the JWT. When a request comes to an endpoint that is protected by this guard, the request must contain the Authorization header. If the header is not present or the token has expired, the guard will send a response with the status HTTP 401 Unauthorized and an error message. If there is no error, then the guard will inject the user into the request object.

The permission guard checks if the user role has the required privilege to access the endpoint. The implementation relies on the authentication guard which inserts the user into the request. After that the permission guard checks if the role of the user has the required permissions to perform the action. If the role does not have the required permissions, then a response with the status HTTP 401 Unauthorized and an error message will be sent.

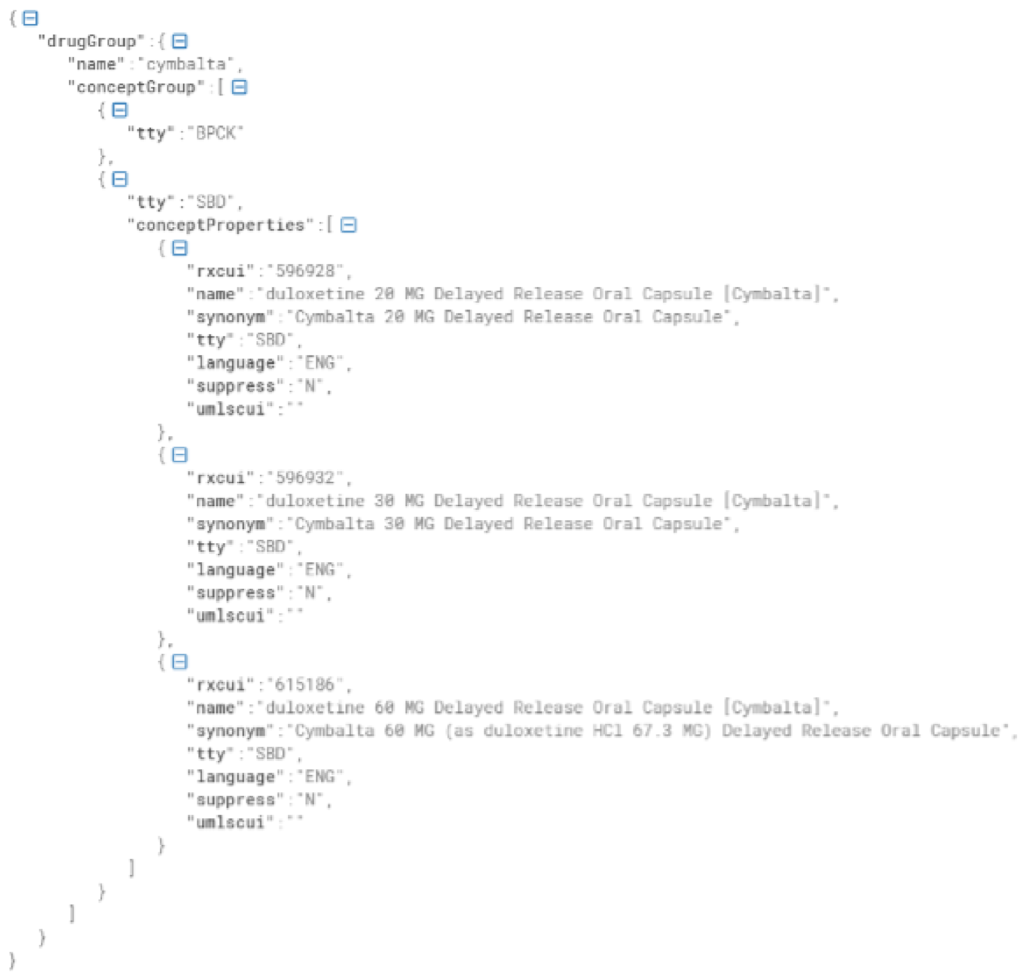
5.3. Medication Controller

The medication controller oversees the following operations:

• Create prescription – gives the possibility to a user to create a prescription
• List prescriptions – returns a list will all the prescription a user has
• Modify prescription – gives the ability to a user to modify an existing prescription
• Spell suggestion – helps a user to find a drug with a certain name
• Share prescription – allows a user to share its prescription
• Find interactions – allows the user to check the interaction between various drugs

Spelling suggestion is a feature is intended to assist users in searching for medications by name. To use the resource, an HTTP GET request is sent to the endpoint ”/medication/suggestions” with the drug name as a query argument. Only the authentication guard protects this endpoint. If the user is not authenticated, an HTTP 401 Unauthorized response will be delivered. After passing the guard, the drug’s name will be entered into the ”MedicationService,” which will begin the search process. RxNav will use a third-party API to obtain appropriate information [14].

There are two APIs that will be used in this case. Spelling Suggestions API returns strings similar to the specified string [14]. For example, if you search for ”adilv,” one of the suggestions will be ”advil.” Get Drug API returns the drug products connected with a given string [14]. For example, if the search phrase is ”cymbalta,” the result is shown in Figure 4.



```
{
   "drugGroup":{
      "name":"cymbalta",
      "conceptGroup":[
         {
            "tty":"BPCK"
         },
         {
            "tty":"SBD",
            "conceptProperties":[
               {
                  "rxcui":"596928",
                  "name":"duloxetine 20 MG Delayed Release Oral Capsule [Cymbalta]",
                  "synonym":"Cymbalta 20 MG Delayed Release Oral Capsule",
                  "tty":"SBD",
                  "language":"ENG",
                  "suppress":"N",
                  "umlscui":""
               },
               {
                  "rxcui":"596932",
                  "name":"duloxetine 30 MG Delayed Release Oral Capsule [Cymbalta]",
                  "synonym":"Cymbalta 30 MG Delayed Release Oral Capsule",
                  "tty":"SBD",
                  "language":"ENG",
                  "suppress":"N",
                  "umlscui":""
               },
               {
                  "rxcui":"615186",
                  "name":"duloxetine 60 MG Delayed Release Oral Capsule [Cymbalta]",
                  "synonym":"Cymbalta 60 MG (as duloxetine HCl 67.3 MG) Delayed Release Oral Capsule",
                  "tty":"SBD",
                  "language":"ENG",
                  "suppress":"N",
                  "umlscui":""
               }
            ]
         }
      ]
   }
}
```

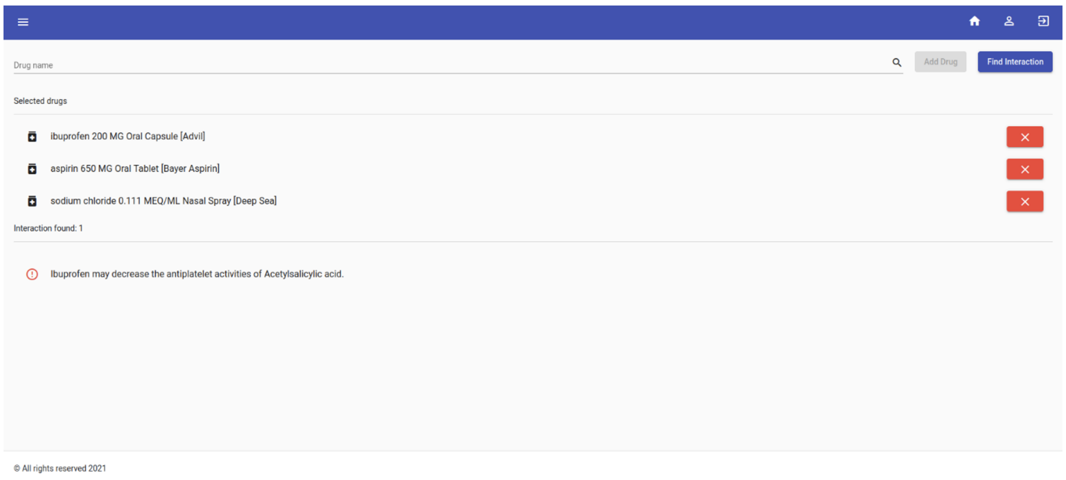
Figure 4. Response of Get Drug API for search term "cymbalta"

Both of these APIs are utilized together to increase search quality. However, "Get Drug API" takes precedence. If it produces a list of medications, that list will be processed and only the names returned; otherwise, the "Spelling Suggestions API" suggestions will be returned. This API will always deliver a response with the status HTTP 200 OK and a list after passing the authentication guard.

The endpoint of Interaction Checker, "/medication/ interactions," can be accessed by sending an HTTP GET request along with a query parameter containing a list of strings (drug names). To access the functionality, the user must be authenticated; otherwise, an HTTP 401 Unauthorized response and an error message will be delivered. Following successful authentication, the endpoint will always provide a response with the status HTTP 200 Ok and a list of interactions (the list can be empty). The query parameter is extracted and inserted into the "MedicationService."

After obtaining a list of drug names, the next step is to obtain an RXCUI (RxNorm concept unique identifier) for each drug [15]. A request will be sent to the "Get Drug API" for each drug name in the list to acquire all RXCUIs. The final step will be to make another request to the "findInteractionsFromList" API to retrieve the list of interactions. The user will receive a response with the status HTTP 200 OK and a list of interactions after processing the response to obtain only the interaction.

In Figure 5 can be observed how our solution displays the interaction between the three drugs that were selected: ibuprofen 200mg, aspirin 650mg and sodium chloride. Using a red exclamation point, the user is informed that Ibuprofen may decrease the antiplatelet activities of Acetylsalicic acid (aspirin).



Selected drugs

ibuprofen 200 MG Oral Capsule [Advil]

aspirin 650 MG Oral Tablet [Bayer Aspirin]

sodium chloride 0.111 MEQ/ML Nasal Spray [Deep Sea]

Interaction found: 1

Ibuprofen may decrease the antiplatelet activities of Acetylsalicylic acid.

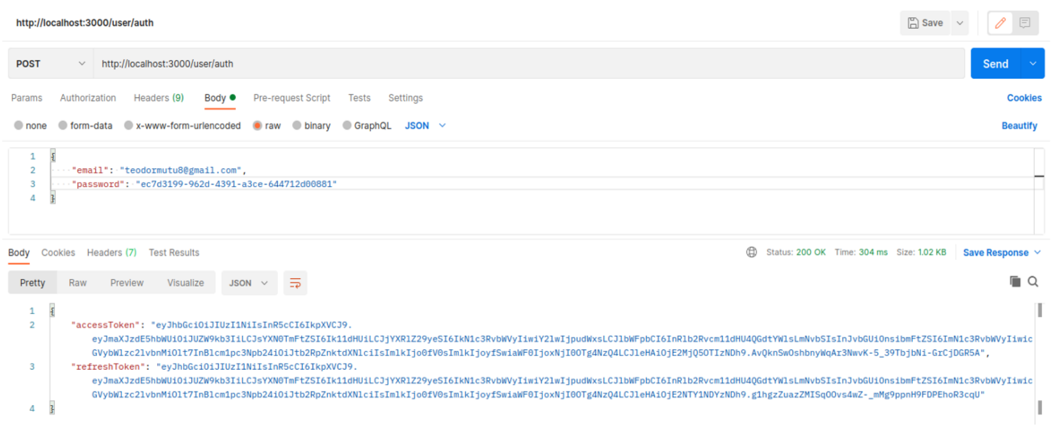
© All rights reserved 2021

FIGURE 5. Interactions found by PADDY between Ibuprofen, Aspirin and Sodium Chloride

6. Testing and Assessment

The behavior of the application's server or database is monitored via the backend testing process. Before the product is distributed to end users, this is done with the intention of discovering undefined and undesirable behaviors such as deadlock, data loss, and corruption. The validation method was divided into two parts so that a robust solution could be developed: manual testing and user testing.

The manual testing was done using Postman [12]. Each endpoint received multiple requests, both valid and invalid, in order to be certain that the application behavior is defined. A valid HTTP POST request with postman can be seen in Figure 6. Each endpoint was tested with multiple requests to cover all the possible responses.



```
{
    "email": "teodormutu8@gmail.com",
    "password": "ec7d3199-962d-4391-a3ce-644712d00881"
}
```

Status: 200 OK Time: 304 ms Size: 1.02 KB

```
{
    "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJmaXJzdE5hbWUiOiJUZW9kb3IiLCJsYXN0TmFtZSI6Ik11dHUiLCJjYXRlZ29yeSI6IkN1c3RvbWVyIiwiY2lwIjpudWxsLCJlbWFpbCI6InRlb2Rvcm11dHU4QGdtYWlsLmNvbSIsInJvbGUiOnsibmFtZSI6ImN1c3RvbWVyIiwic
        GVybWlzc2lvbnMiOlt7InBlcm1pc3Npb24iOiJtb2RpZnktdXNlciIsImlkIjo0fV0sImlkIjoyfSwiaWF0IjoxNjI0OTg4NzQ4LCJleHAiOjE2MjQ5OTIzNDh9.AvQknSwOshbnyWqAr3NwvK-5_39TbjbNi-GrCjDGR5A",
    "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJmaXJzdE5hbWUiOiJUZW9kb3IiLCJsYXN0TmFtZSI6Ik11dHUiLCJjYXRlZ29yeSI6IkN1c3RvbWVyIiwiY2lwIjpudWxsLCJlbWFpbCI6InRlb2Rvcm11dHU4QGdtYWlsLmNvbSIsInJvbGUiOnsibmFtZSI6ImN1c3RvbWVyIiwic
        GVybWlzc2lvbnMiOlt7InBlcm1pc3Npb24iOiJtb2RpZnktdXNlciIsImlkIjo0fV0sImlkIjoyfSwiaWF0IjoxNjI0OTg4NzQ4LCJleHAiOjE2NTY1NDYzNDh9.g1hgzZuazZMISqOOvs4wZ-_mMg9ppnH9FDPEhoR3cqU"
}
```

FIGURE 6. Postman Login Requests Testing Results

In order to test the application, two doctors and three non-professionals were asked to spend at least one week using the platform PADDY. In the first phase of the user testing, the users were asked to perform certain tasks for the first time and during their completion, we observed their action and where and if they hesitated or encountered difficulties. We also measured the amount of time they required for completing each task.

In Table 1 we can observe that difference between the reference time and average user time for each task is narrow, except for the complex longer tasks: creating an account, creating a prescription and deleting an account. Nonetheless, the average time a user takes to complete all the listed tasks is still ideal for a beginner.

Tasks	Time of Reference	Average User Time
Creating an account	26 seconds	45 seconds
Searching for a drug	10 seconds	12 seconds
Selecting and adding three drugs	30 seconds	47 seconds
Checking for interactions	6 seconds	13 seconds
Creating a prescription	92 seconds	110 seconds
Listing a prescription	10 seconds	14 seconds
Sharing a prescription	7 seconds	10 seconds
Modifying a prescription	12 seconds	13 seconds
Deleting an account	25 seconds	62 seconds

TABLE 1. User Testing Results - Task's Performance and Timing

After the first week of user testing, we asked our users to respond to nine questions about PADDY in order to figure out the advantages and the gaps that are missing in our solution. The same procedure was repeated for two other platforms: ANM [10] and Mediately [11].

- Q1: On a scale from one to ten, how easy was it to create a new account? (One means very hard, ten means very easy)
- Q2: On a scale from one to ten, how easy was it to search for drugs? (One means very hard, ten means very easy)
- Q3: On a scale from one to ten, how easy was it to find information about the interactions between drugs? (One means very hard, ten means very easy)
- Q4: On a scale from one to ten, how easy was it to understand the information regarding the interactions between drugs? (One means very hard, ten means very easy)
- Q5: Was the information provided sufficient? Rate on a scale from one to ten
- Q6: On a scale from one to ten, how easy was it to create a prescription? (One means very hard, ten means very easy)
- Q7: On a scale from one to ten, how easy was it to share a prescription? (One means very hard, ten means very easy)
- Q8: On a scale from one to ten, how safe did you feel sharing your prescriptions? (One means very unsafe, ten means very safe)
- Q9: On a scale from one to ten, how many of your mandatory features are met in this solution? (One means none, ten means all of the features)

The results were split into two tables: professional (Table 2) and professional (Table 3). The columns under an application represent the average score obtained at question.

Application	PADDY	Mediately	ANM
Question 1	10	7	3
Question 2	10	8	2
Question 3	10	9.5	7
Question 4	8	9	9
Question 5	5.5	9.5	8.5
Question 6	8	N/A	N/A
Question 7	10	N/A	N/A
Question 8	7.5	N/A	N/A
Question 9	7	8	7

Table 2. User Testing Results - Professionals

The consumers (non-professionals) considered the application extremely easy to use and quickly gained an idea on how the drugs interacted. The only aspect that our solution should be taking in consideration, according to the users feedback, is the amount of information displayed from an interaction. The data was sufficient to make the non-professionals seek a doctor advise.

However, the users wanted to have more insight on how the drugs would influence their body. Nonetheless, compared to other platforms the users were satisfied with the easy process of searching and understanding the interactions.

The professional users had a better understanding of the applications. For them it was easy to comprehend the information on all three platforms. However, the process of finding the interactions was easier on Paddy, unlike ANM or Mediately. They appreciated the idea of small and concise information, this way being less time consuming for them as proffesionals. The doctors found the saving, sharing, and modifying prescription features extremely useful. This aspect of the platform was initially overlooked by the non-professional users.

Application	PADDY	Mediately	ANM
Question 1	9.7	5.7	3
Question 2	10	9.7	3.7
Question 3	10	7.7	6.7
Question 4	8.7	6.7	5
Question 5	9.7	8.7	8
Question 6	8	N/A	N/A
Question 7	10	N/A	N/A
Question 8	6.7	N/A	N/A
Question 9	9.7	8	6.7

Table 3. User Testing Results - Non-Professionals

7. Conclusions

By providing the user with an easy way to verify drug interactions, our solution was able to reduce medical blunders. The information displayed is as clear and concise as feasible. However, the application's usefulness is not restricted to detecting drug interactions. The platform became a junction for doctors and patients, allowing them to write, share, and revise prescriptions. These qualities satisfied both non-professionals and doctors.

PADDY outperformed current alternatives in terms of consumer acceptance. Non-professional users had little trouble discovering various drug interactions and understanding the information given, as shown in Table 3. They had difficulty discovering interactions, particularly on ANM [10], and interpreting the information about the interactions while using the other platforms. Another feature that customers appreciated was the ability to share, write, and edit prescriptions. Even when polling expert users, PADDY's score in the first questions remained maintained as it can be seen in Table 2. The doctors appreciated how quick and simple the process was for providing them with information on drug interactions.

As can be seen, the project's solution addresses the two problems that deter non-professionals from reading flyers. PADDY is more than just an interaction monitor. It allows both experts and patients to generate, share, and edit medical prescriptions.

REFERENCES

[1] *M. Salzman*, Durable HIV Remission in London Patient, the Second Person Cured of HIV, TheBodyPro. **14**(2021) Available: https://www.thebodypro.com/article/durable-hiv-remission-london-patient-second-person-cured-hiv.

[2] Immunotherapy to Treat Cancer, cancer.gov, **2**(2019) Available: https://www.cancer.gov/about-cancer/treatment/types/immunotherapy

[3] *Nathan, Joseph Zerilli, Tina Cicero, Lorraine Rosenberg, Jack.*, Patients' Use and Perception of Medication Information Leaflets, The Annals of pharmacotherapy, **41**(2007), 777-82

[4] *B. J. Siriporn Burapadaja*, Determinants of Consumer's Drug Leaflet Reading, The CMU. Journal, **2**(2013), 777-82

[5] *Y. Noguchi, T. Tachi și H. Teramachi*, Review of Statistical Methodologies for Detecting Drug–Drug Interactions Using Spontaneous Reporting Systems, Frontiers in Pharmacology, **10**(2019)

[6] *L. Sun, X. Jiang, H. Ren and Y. Guo*, Edge-Cloud Computing and Artificial Intelligence in Internet of Medical Things: Architecture, Technology and Application, IEEE Access, **8**(2020), 101079-101092

[7] *L. Chen, C. Chu, Y.-H. Zhang, M. Zheng, L. Zhu, X. Kong și T. Huang*, Identification of Drug-Drug Interactions Using Chemical Interactions, Frontiers in Pharmacology, **12**(2017), 526-534

[8] *D. L. Schwappach, V. Mülders, D. Simic și P. A. Thürmann*, Is less more? Patients' preferences for drug information leaflets, Pharmacoepidemiology And Drug Safety, **20**(2011)

[9] *Institute of Medicine*, Preventing Medication Errors, The National Academies Press, Washington, 2007.

[10] *ANM*, Nomenclatorul medicamentelor de uz uman, Agenția Națională a Medicamentului și a Dispozitivelor Medicale din România, (2022) Available: https://www.anm.ro/medicamente-de-uz-uman/nomenclatorul-medicamentelor-de-uz-uman/

[11] *Mediately*, Lista Medicamentelor, Mediately (2022) Available: https://mediately.co/ro/drugs

[12] *Postman*, Postman Inc., Postman (2022) Available: https://www.postman.com/.

[13] *Drugs.com*, Drug Interactions Checker (2022) Available: https://www.drugs.com/drug_interactions.html.

[14] *USA GOV*, APIs Available for Drugs (2022) Available: https://rxnav.nlm.nih.gov/APIsOverview.html

[15] *USA GOV*, RxNorm (2022) Available: https://www.nlm.nih.gov/research/umls/rxnorm/overvGeiew.html.