Abstract—The need of processing medical data services is rapidly growing. However, the current local data health centers and systems have a shortage to manage this data proliferation and users’ needs. Mobile applications can work in dynamic wireless networks that cover wide geographical areas, thus it can enhance the medical data services. In this paper, we design a mobile application method that supports e-health data services, performs remote data transactions and determines patient’s location and the nearby health centers. This approach is built on interface components, cloud virtual machines, and e-health databases. The main contribution of this paper is building an integrated e-health data services from independent mobile and wireless devices. This will facilitate the development of large number of different mobile e-health applications and facilitate the application’s maintaining processes.

Keywords—E-health, Android, RESTful Web service API, JSON, HTTP standard, Database Handler, Back-End Communication, SQL.

I. INTRODUCTION

Health Centers are one of the important real life premises. With no regard to the actual reasons for visiting clinic centers, most of the patients want to make a treatment or to make an appointment with a specialist, they either need to go to that clinic or by calling the healthcare center to submit their appointment. Patient will make reservation and wait a long time to get treatment or even an appointment. However, it is common if patients complain for not being satisfied about provided service. There are many reasons leading to the dissatisfied feeling including being served late in terms of medication by the specialist. On the other hand, people sometimes visit the city for the first time where they do not know the health centers’ locations around them, what kind of clinic centers, what kind of medications they support and how to reach them.

In this context, smartphones play an important role which are able of showing all patients wherever they are; what types of health clinics are nearest to them with providing a route and ability of making appointments as well. The new technology of Android operating system [1] enables developing promised applications for smartphones that help the patients to find the preferred hospitals or health centers in a specific area and according to the specified treatment. Patients’ location determined by Google Maps [2] helps to show the nearby health centers based on the user’s current location and a route to a particular one.

Moreover, providing patients with a list of physicians and specialties offered by the health center can help much in saving the patients’ expenses, time and effort. In addition, this technology supports patients with a possibility to make their appointment reservations in advance and remotely in their favorite time to their favorite clinic.

This paper is organized as follows: Section II describes the related work and Section III illustrates the method requirements and architecture. Creating mobile data services and the implementation is discussed in section IV. Finally, the conclusion and future work are drawn in section V.

II. RELATED WORK

Intensive works are done on developing mobile applications. Some are focused on the interaction between patients and doctors, others are handle sending and receiving text and images, but few are consider e-health services based on patients’ locations. Rao et al. [3] presented a mobile application for smartphone on Android platform to facilitate interaction between the patient and doctor. However, this application doesn’t represent an actual patient data nor connected to actual hospital database. Chun et al. [4] proposed a reusable data management and messaging platform that provides a simple and unified abstraction for mobile applications. However, this platform doesn’t support patient’s location services. Gheorghe [5] identified and analyzed the importance of using mobile technologies for developing healthcare solutions and described a medical application as an e-health solution, which can be used to improve care services. However, he does not describe any health care service technically nor discuss the challenges in this field.

Authors in [10] presented a web based mobile system that has three main components: the electronic medical application, which is comprised of the patient record and diagnosis module, the requests/approval application, and the dashboard application for data visualization. This mobile system provides the web auxiliary entry, aggregated disease report application and usage monitoring. However, the monitoring time complexity is high due to iterative involvement of multi-stakeholders, which is necessary to ensure higher acceptance and adoption. To improve the Quality of Service (QoS) of remote monitoring of patients and elderly, authors in [11] proposed a design of a new context-aware mobility management system that is able to
take benefit from multiple available access wireless networks (3G and WiFi) to offload traffic and maintain the QoS of critical services such as e-health services. However, the large amount of 3G data traffic transmitting through this network is expected to grow rapidly with the popularity of mobile applications and therefore degrade the overall service.

A network infrastructure is proposed by authors in [12] which is based on the Software Defined Networking (SDN) paradigm to enable quality-oriented mobility control capabilities in the Context-Aware Mobile Approach (CAMA) to deploy mobility prediction, Point of Attachment (PoA) decision and handover setup meeting both application quality requirements and current wireless quality conditions, then improving the dependability and reliability of the e-Health biofeedback systems. However, the proposed study can’t offer an efficient remote data centers e-health services which are needed for controlling patients in position monitoring. Authors in [13] presented PISCES, a responsive health monitoring mobile information system, which enables remote monitoring of the patients’ health-status and physical performances, react in any case of deficiency, as well as giving the opportunity to increase the level of the physical activities. However, this study focuses on controlling patients’ e-health services from one e-health center, but the patient may need care of multi-health centers simultaneously where coordination of multi-health services is required.

An anonymous on-the-fly secure data exchange protocol [14] proposed for preventing health data from attacks based on pairing-based cryptography. The proposed solution allows cloud peers to dynamically generate temporary identities that are used to produce a session key for each session of data exchange. The proposed protocol is designed against different attacks, such as target-oriented, man-in-the-middle, masquerade, and message manipulation attacks. However, multi-data exchange between different e-health centers enforce this protocol to reschedule the e-health service process, which lead to communication delay, and incomplete processes.

III. E-HEALTH SERVICES REQUIREMENTS AND ARCHITECTURE

The main contribution of this paper is building an integrated e-health data services from independent mobile and wireless components. If any changes are needed to be done on a certain e-health data service component, it does not require modifying on any other components. The e-health data system is built on RESTful Web service API [6], which acts as a middle layer to communicate between client/cloud server sides. RESTful APIs are defined with these aspects whenever an HTTP [7] request is made:

- Base URL, such as http://localhost/resources/
- An Internet media type for the data. (e.g., JSON [8]).
- Standard HTTP methods (e.g., GET, POST, PUT, or DELETE).
- Hypertext links to reference state and related resources.

The proposed e-health data services architecture supports dependable software components, reliable real time system applications and maintains data integrity. The proposed e-health system consists of the following components:

- User Interface
- Database Handler Interface
- Back-End Communication Interface
- SQL Database
- Location API
- Server (Database and API)

Figure 1 describes how the process goes through between the e-health system components, relations, and its properties. In what follows a brief description of different functions and components is provided needed to provide the e-health services is provided.

a. Patient Interface

The user interface component describes the activities of the application which represents the view and the controller of e-health model. For instance, the activity Google maps inflates it's view which represents a list view to all nearby health centers from the patient’s location and a button in order to display it on the map with markers. The activity is also responsible to perform its task of searching for the nearby health centers and set those in the adapter in order to be shown in the list view for the user.

![Fig.1. E-Health Data Services System Architecture](image)

b. The Back-End Communication Interface:

The main functionalities of this interface are:

- Receives the user interface request, parses its parameters and communicates with the server over an HTTP request.
- The server then sends back an HTTP response, which indicates whether the service has been performed successfully to the Back-End Communication Interface.
Back-End Communication Interface in turn sends this server response (Model Container) back to the user interface.

The model container represents the response coming back from the Back-End Communication Interface as complete object. For example, when a certain activity communicates with Back-end interface to perform such a service of fetching items from a specific health center, the response which is coming back to that activity is in JSON format where it is used to get the properties of the item like ID, name, description, and create an instance with these properties from the health item model. So, the health item class represents as model container.

c. Database Handler Interface:
The main functionalities of this interface are:

- The user interface component communicates with the Database Handler Interface to perform the required service.
- Database Handler Interface communicates with a SQLite Database, which is a local Database to perform user’s requests.

d. Location API
The Google Location Services API, part of Google Play services, provides a more powerful, high level framework location services that provide new features, such as activity detection, that are not available in the framework API. The Location API is used to determine the current geographic location of the patient. This can be done via GPS module or via cell tower triangulation or via WIFI networks.

In our method, we detect the patient's current location by the GPS Tracker class that is responsible for detecting the patient's position using either the GPS or the network provider. The class contains the (android. Location) package, which provides the API to determine the current Geo position. One of the API classes that the package provides is Location Manager Class, which provides access to the system location services. These services allow the applications to obtain periodic updates of the device's geographical location. This class was not instantiated directly; instead, it was retrieved through the Context.getSystemService(Context.LOCATION_SERVICE)

All location API methods require the following permissions in order to have an access to the GPS or passive location provides:

- ACCESS_COARSE_LOCATION
- ACCESS_FINE_LOCATION

The GPS location provider determines patient’s location using satellites. Depending on several conditions, this provider may take a while to return a location fix. It requires the permission AC-CESS_FINE_LOCATION, while the Network provider determines location based on availability of cell tower and WiFi access points, the results retrieved by means of a network lookup. In our method, we determine the patient’s location through the following steps:

- First we check which provider is enabled using the method isProviderEnabled(String Provider) which returns the current enabled/disabled status of the given provider.
- The location manager invokes the requestLocationUpdates() method for registering location updates using the named provider, minimum time interval between location updates (milliseconds), minimum distance between location updates (meters), and a location listener whose onLocationChanged(Location) method will be used for each location update.
- The location manager invokes the method getLastKnownLocation(String provider) which returns a location indicating the data from the last known location fix obtained from the given provider.
- The returned location is used to get the patient coordinates (latitude, longitude) and store them into the appropriate variables for later to be invoked from the maps Activity to show the patient's location on the map.

The main contribution of our approach is that, any modifications needed on the server, for instance, changing from SQLite database to any other database type, doesn't require changing the user interface component. The only part that needs to be modified is the communication with the database in order to be adapted for handling the database operations.

IV. CREATING MOBILE E-HEALTH DATA SERVICES

Creating e-health services will be done through HTTP requests to perform services that are typically used to implement the RESTful API. Different types of HTTP requests are created such as patient authentication, login, health center information and treatment services. Treatment and e-health services’ requests are described as follows:

a. Authentication Request
The following code run in the Android platform is made to authenticate the patient in the e-health system:

app->POST/register

- This request requires patient credentials like (email/name/id, and password) for the registration process.
- These parameters are sent via a request to the API, which calls the appropriate database function for storing the patient credentials.
- An access token is generated and sent back with the rest of patient object to the e-health system when the patient is registered in order to be used
with the requests that need to be authenticated.

- The response then is coming from the server after registration process is completed in JSON format.
- The response holds the patient’s information like the patient id, name, or email, and the API Key.

For example, the server response is shown below as JSON response:

```json
{
  "unique_id": "538c2779d71281.68838588",
  "user": {
    "name": "ABC",
    "email": "abc@health1.com"
  },
  "created_at": "2015-11-02 09:27:53",
  "updated_at": null,
  "api_key": "696cf635b5ea370245d42f9dbde181a",
  "success": 1,
  "message": "Registered successfully"
}
```

b. Login Request

To allow the patient login into the e-health system, we write the following request:

```php
$app->POST/login
```

- This request is sent whenever the patient loges into the e-health system.
- The required parameters are the email and the password.
- The API reads the POST parameters and checks against the database whether the patient’s credentials are correct or not.
- The API sends back the patient's information to the e-health system in case the patient enters the correct credentials.

c. Treatment Requests

Reserving an appointment for a medical treatment in a certain health center can be accomplished by the following request:

```php
$app->POST/reservation
```

This request needs to be authenticated before performing the reservation service, since we need to know who does the reservation and from which health center. This is done by authenticating the request which uses the access token that was already generated once the patient registered to get the patient identification number as a global parameters, which is used for making the reservation. In addition, the health center identification number and other required parameters to perform the reservation such as date, time, number of persons, name, email, and phone number are required to complete this request. The response of this request will return the reservation number for the patient. For example, the JSON response would be as follows:

```json
{
  "error": false,
  "message": "Reservation created successfully",
  "reservation number": "10"
}
```

Performing treatment service will be done through the following request:

```php
$app->POST/treatments
```

This request is considered to handle two actions as two requests are made one inside the other (internally):

- The first request stores the information about the health center that the patient wants to medicate at in case if it is not already stored in the database.
- The second request performs the treatment service itself.

d. Request Requirements:

- The parameters, which are needed for storing the health, center information such as reference, name, vicinity, address, and phone number.
- The list of treatments that patient has already request and the patient ID.

The health center information should be available before performing the treatment request to verify from which health center the patient has asked for treatment. After the treatment request is created successfully, the treatment ID is returned to the patient for acknowledgment purposes.

e. E-Health Services List Request

The list of the e-health services is simply created through the following request:

```php
$app->GET/services
```

This request is made in order to fetch the health center's services and display it for the patient. This provides the patient with rich and detailed treatment items, which enable him to navigate across and having a deep look that gives the patient a unique experience of which services should be selected. The required parameter for this request is the center ID in which is used to fetch the services that belong to that health center. The JSON response for this request is:

```json
{
  "error": false,
  "services": [
    { "Service_id":1, "title": "Emergency", "description": "An emergency cases that needs immediate medication and high care", "section": "1" },
    { "Service_id":2, "title": "Surgery", "description": "Include but are not limited to Bariatric, Breast, Colon and Rectal, Endocrine, Gynecological, Head and Neck, Neurosurgery, Orthopedic, Pediatric", "section": "2" }
  ]
}
```

In what follows, we describe some guidelines on how to implement the proposed system. Slim [9] is a PHP micro
framework that helps developers quickly to write simple yet powerful web applications and APIs. Slim is precursor to the popular micro framework movement which is a collection of bare necessities required to build a web application that facilitates receiving an HTTP request, routing the HTTP request to an appropriate controller, dispatching the controller and returning an HTTP response. It has a simple interface, intuitive, and extensively documented both online and in the code itself. Moreover, Slim supports all HTTP GET, POST, PUT and DELETE methods which are necessary for a REST API. In addition, Slim provides a middle layer architecture which will be useful to filter the requests. In our e-health system services design, Slim is used for verifying the API key. In the following Slim code, we declared $app as a variable instantiated from Slim library that is used to make an HTTP request for login using HTTP Post method.

```php
$app->post('/login', function() use ($app) {
    // check for required parameters
    verifyRequiredParams(array('email', 'password'));
    // read post parameters
    $email = $app->request()->post('email');
    $password = $app->request()->post('password');
    $response = array();
    // Some other requests need to be authenticated, so a second parameter is added when the request is made by using an instance from Slim. For instance, when the patient wants to make a treatment from a specific health center, it is required to have an authenticated request. The following Slim code illustrates how Slim manages to use authentication in the request using $app instance.
    $app->post('/health_centers', 'authenticate', function() use ($app) {
        // read post parameters
        $centerid = $app->request()->post('centerid');
        $name = $app->request()->post('name');
        $reference = array();
        // Some other requests need to be authenticated, so a second parameter is added when the request is made by using an instance from Slim. For instance, when the patient wants to make a treatment from a specific health center, it is required to have an authenticated request. The following Slim code illustrates how Slim manages to use authentication in the request using $app instance.
    });
```

V. CONCLUSION AND FUTURE WORK

In this paper, we described the major steps needed to design a mobile e-health system model that enables patients to find the health services based on their locations. This model provides a digital treatment’s menu, which gives the patient a unique visual experience of what health centers offer by showing several medication types and items provided with a rich description and images as well, where the patient can request for treatment and reserve an appointment from a particular health center. We also provided an insights on how an Android mobile e-health system (RESTful API) using Slim, PHP, and MySQL can be built. This API acts as middle layer, which communicates between the Application and the Database, handles all the requests coming from Android side and calls the database in order to perform the requested mobile services. Our experiments results show that our mobile e-health system is easy to use by patients via their smartphones where they manage their treatment appointments form a particular health center in a convenient date and time. As a future work, the paper can be improved to support a description on how to support more mobile devices platforms like tablets and iOS and extended to provide a website that enables the health center administrative to upload their medication services on the server. The mobile e-health system can suggest an appropriate health center for the patient who looks for some kind of treatment based on smart e-health data fragmentation, clustering and allocation techniques.

REFERENCES