

# iHealth Birth Certificate App User Manual

---

**Author:** Team iHealth

## Introduction

The collection and submission of birth certificate information to state vital statistics agencies is currently the responsibility of birth certificate clerks, typically staff in medical records departments of hospitals. For each birth occurring in a hospital, birth clerks are required to abstract clinical information from the records of both the mother and the newborn. This information is hand-written onto a facility worksheet. A second worksheet, the mother's worksheet, is completed by the mother and father, if applicable, and contains demographic information about the parents and the desired name of the newborn child. Both worksheets are entered into a web-based form called an electronic birth registration system (EBRS) and submitted to the state health department birth certificate registry.

## System Requirements

- **Operating Environment:** IE, Safari, Chrome or Firefox.
- **CPU:** 1.2 GHz and higher
- **RAM:** 1.0 GB and larger
- **Internet access** through cellular or WLAN is available when necessary.

## Set up:

### 1. Connect with the FHIR server

Run Docker

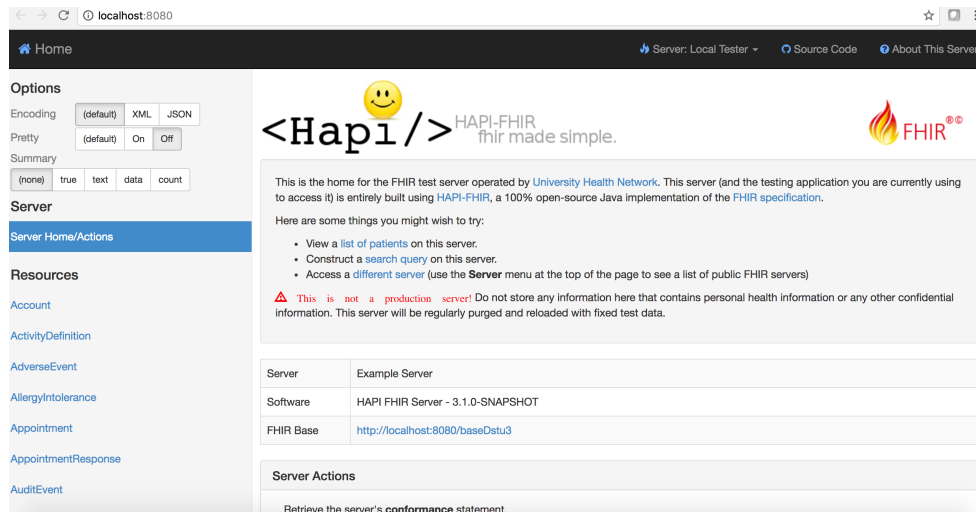
From command line, type:

```
docker run -d -p 8080:8080 --name=my-fhir-server djohnson325/hapi-fhir-jpaserver:stu3
```

Then point your web browser to:

<http://localhost:8080/>

You should see the following screen



## 2 Up load newborn's information to the server

We use tag-uploader to upload newborn's data to the FHIR server  
(The details can be seen at <https://github.com/smart-on-fhir/tag-uploader>)

```
In command line/terminal type: git clone https://github.com/smart-on-fhir/tag-uploader.git  
cd tag-uploader  
npm i
```

Create a new folder called STU3, put BabyBoy iHealth.json to the STU3 folder  
From command line/terminal, type

```
node . -d ../path/to/BabyBoy iHealth.json -S http://localhost:8080/baseDstu3
```

You should be able to see the file get processed as shown in the following image

```
0% ██████████ Processing file
100% ██████████
Done 1 files processed, 4 resources tagged, 1 resources uploaded, 0 resources
failed to upload
```

In the server, click patient, you should see the patient (newborn) data there. The id is cf-1509653935164

Home > Patient > Search for Resources

Server: Local Tester Source Code About This Server

**Options**

Encoding: (default) XML JSON

Pretty: (default) On Off

Summary: (none) true text data count

**Server**

Server Home/Actions

**Resources**

Immunization 1

Observation 1

**Patient 1**

Procedure 1

Account

ActivityDefinition

AdverseEvent

AllergyIntolerance

Appointment

AppointmentResponse

**Request**

GET http://localhost:8080/baseDstu3/Patient?\_pretty=true

**Request Headers**

Accept-Charset: utf-8  
Accept: application/fhir+xml;q=1.0, application/fhir+json;q=1.0, application/xml+fhir;q=0.9, application/json+fhir;q=0.9  
User-Agent: HAPI-FHIR/3.1.0-SNAPSHOT (FHIR Client; FHIR 3.0.1/DSTU3; apache)  
Accept-Encoding: gzip

**Response** ✓ HTTP 200 OK

**Response Headers**

last-modified: Thu, 16 Nov 2017 22:55:11 GMT  
server: Jetty(9.4.7.v20170914)  
transfer-encoding: chunked  
x-powered-by: HAPI FHIR 3.1.0-SNAPSHOT REST Server (FHIR Server; FHIR 3.0.1/DSTU3)  
content-type: application/fhir+json;charset=utf-8

**Result Body**

JSON bundle  
(1421 bytes)

Bundle contains 1 / 1 entries

ID	Updated
<a href="#">Read</a> <a href="#">Update</a> Patient/cf-1509653935164/_history/1	22:47:18

**Raw Message**

```
{
  "resourceType": "Bundle",
  "id": "bc34e6e-8615-4f11-9a79-3daee6a38823",
  "meta": {
    "lastUpdated": "2017-11-16T22:55:11.189+00:00"
  },
  "type": "searchset",
  "total": 1,
  "link": {
    "relation": "self",
    "url": "http://localhost:8080/baseDstu3/Patient?_pretty=true"
  },
  "entry": [
    {
      "fullUrl": "http://localhost:8080/baseDstu3/Patient/cf-1509653935164",
      "resource": {
        "resourceType": "Patient",
        "id": "cf-1509653935164",
        "meta": {
          "versionId": "1",
          "lastUpdated": "2017-11-16T22:47:18.269+00:00"
        },
        "text": {
          ...
        }
      }
    }
  ]
}
```

### 3 Run the app.

Local server for the web application can be set up using Xampp Apache (port need to configured at 8080).

Database can also be setup using Xampp Mysql, the name of the database need to be "registration", the table can be set up based on the account.sql. As shown in the image:

localhost:8080/phpmyadmin/db\_structure.php?server=1&db=registration

Server: localhost Database: registration

Structure SQL Search Query Export Import Operations Privileges Routines Events More

Filters

Containing the word:

Table	Action	Rows	Type	Collation	Size	Overhead
account	<a href="#">Browse</a> <a href="#">Structure</a> <a href="#">Search</a> <a href="#">Insert</a> <a href="#">Empty</a> <a href="#">Drop</a>	7	InnoDB	latin1_swedish_ci	16 KiB	-
<b>1 table</b>	<b>Sum</b>	7	InnoDB	latin1_swedish_ci	16 KiB	0 B

Check all With selected:

Print Data dictionary

Create table

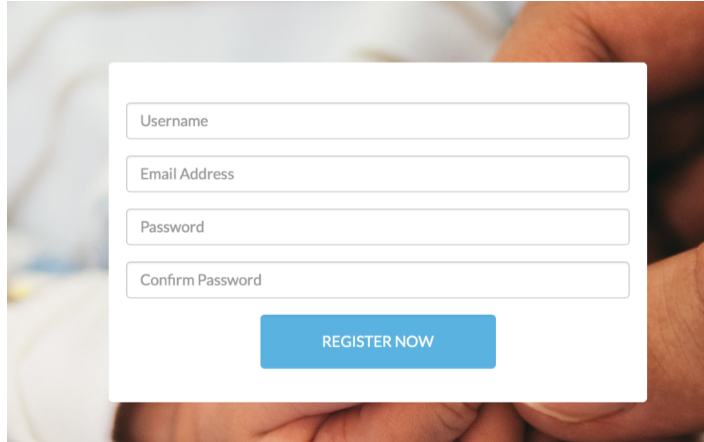
Name: Number of columns: 4

Download the project folder and put it at xampp/hotdocs/

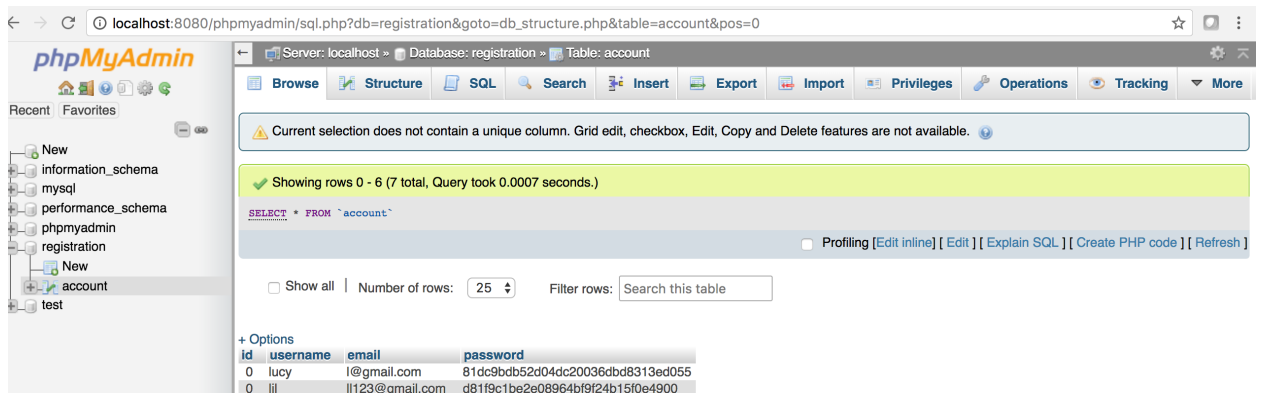
Access the web application at <http://localhost:8080/.../index.php> (point to the location of index.php file.)

### Register

1. Go to the website, click the "Get Started!" button
2. Click "Register" button on the upper right.
3. Input "Username", "Email Address", "Password" and "Confirm Password" in the blanks for the new register. The new register cannot be duplicated with old ones. The encoded phrases should be encoded with a simple substitute cipher. The phrases are limited to 512 characters.
4. Click "REGISTER NOW" button to create the new register.
5. You should able to see database get updated



A registration form with four input fields: Username, Email Address, Password, and Confirm Password. Below the fields is a blue button labeled "REGISTER NOW".



The screenshot shows the phpMyAdmin interface. The left sidebar shows the database structure with 'registration' selected. The main area shows the 'account' table with 7 rows. A message states: "Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available." Below this, a green bar indicates "Showing rows 0 - 6 (7 total, Query took 0.0007 seconds.)". The SQL query shown is "SELECT \* FROM `account`". Below the query, there are options for "Show all", "Number of rows: 25", and "Filter rows: Search this table". At the bottom, a table lists the data for the 'account' table.

id	username	email	password
0	lucy	l@gmail.com	81dc9bdb52d04dc20036dbd8313ed055
0	ljl	ll123@gmail.com	d81f9c1be2e08964bf9f24b15f0e4900

## Login

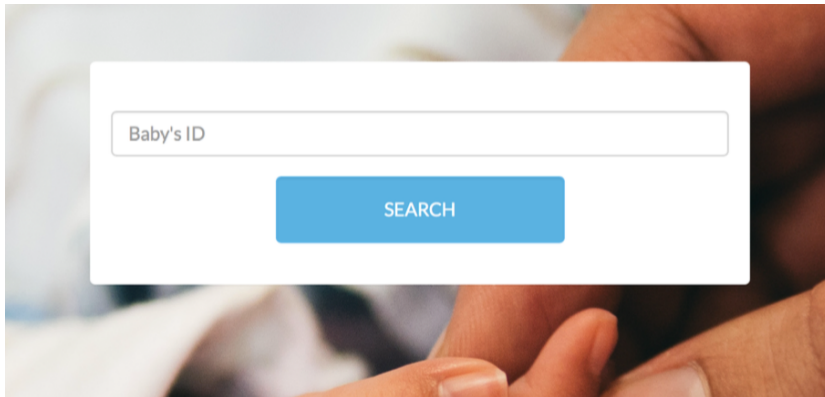
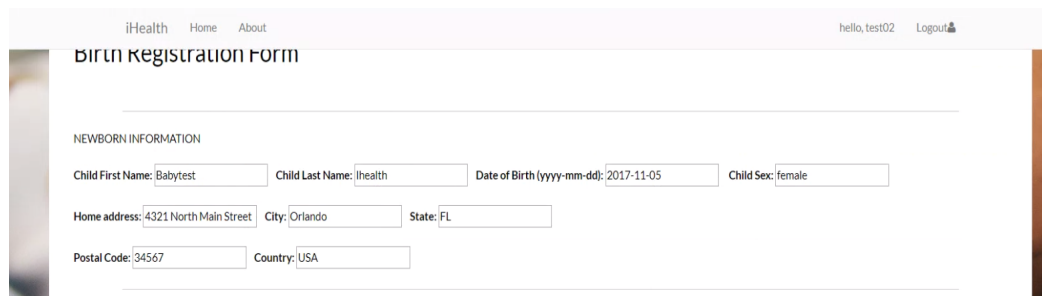
1. Go to the website, click the "Get Started!" button
2. Select the "Login" button on the upper right.
3. Input "Username", "Password".
4. Click "LOGIN IN"

## Logout

1. Click the "Logout" button in the menu.

## Search

1. Login to the website.
2. Input a valid patient information ( cf-1509653935164 ) .
3. Click the "Search" button.
4. You would see the details of patient information.

A white rectangular search form overlay is centered on a blurred background of a person's hands. The form contains a text input field with the placeholder text "Baby's ID" and a blue rectangular button with the text "SEARCH" in white capital letters.A screenshot of the "iHealth Birth Registration Form" web page. The page has a light purple header with "iHealth Home About" on the left and "hello, test02 Logout" on the right. The main content area is white and titled "BIRTH REGISTRATION FORM". Below the title is a section labeled "NEWBORN INFORMATION" containing several input fields: "Child First Name" (filled with "Babytest"), "Child Last Name" (filled with "Ihealth"), "Date of Birth (yyyy-mm-dd)" (filled with "2017-11-05"), "Child Sex" (filled with "female"), "Home address" (filled with "4321 North Main Street"), "City" (filled with "Orlando"), "State" (filled with "FL"), "Postal Code" (filled with "34567"), and "Country" (filled with "USA").

## TECHNICAL SUPPORT

Technical support: iHealth Team