Mobile App Researcher Toolkit

User Guide And Developer Document

Yanbo Li

JACOBS
TECHNION-CORNELL
INSTITUTE
AT CORNELL TECH

JACOBS TECHNION-CORNELL INSTITUTE, CORNELL TECH, CORNELL UNIVERSITY

GITHUB.COM/SUNNYLIYANBO1357/MART

This was done under the supervision of Professor Deborah Estrin, technical adviser Michael Carroll, and HCI adviser Nicola Dell within a specialization project, from September 4th to December 8th of 2016.

First release, December 2016

Contents

1. Introduction

1.1 Abstract

Mobile App Researcher Toolkit (MART) is a webapp that provides an intuitive way for researchers to modify their medical research app. In MART, the app's configuration JSON files would be structured in graphic design, so that researchers can easily edit the survey questions, activities, and consent file as the medical research requirement changes. By building a bridge between researchers and software developers, MART can significantly reduce the development time and communication costs in medical research.

ResearchSuite, which includes sample research apps in both ResearchKit-iOS and ResearchStack-Andriod, is also developed for researchers and developers to start with. Both can use MART to change the content.

1.2 Background

1.2.1 Motivation

Last semester, I developed MyMigraine, an iOS app for migraine research. By building a patient-oriented data collection app and working with researchers and physicians, I understood the process of a medical research, the requirement for research apps, and the basic ResearchKit development rules, which enable me to get the background knowledge of building an interface for researchers to create their own apps.

In early July 2017, I was reached by Michael Carroll, who introduced me to the idea of building a JSON editor to help with research mobile app development. As he said

> The rundown is that the surveys in RK/RS apps are powered by JSON that defines all the elements of the survey. In "the bad old days" (aka right now) researchers communicated how to change these files to the app devs, app devs made the changes, researchers QA, researchers requested more adjustments, [repeat that process as many times as necessary]. With the new tool, researchers can drop the JSON for the surveys into their

browser, see how it looks automatically, edit as they see fit, and they just give the new JSON back to the devs.

The idea of generating an App totally made me excited. Since many research apps are in the similar format, it makes perfect sense to have a standard for research app generation, and all it leaves is for researchers to change the content. I think it is an interesting project that could really make a difference in the way that research apps are developed, so I started with research on current research apps and interview with researchers and developers.

1.2.2 ResearchKit/ResearchStack

ResearchKit and ResearchStack (RK/RS) are medical research app frameworks for iOS and Android platforms respectively. Since their appearance, it has never been easier for medical research to reach out patients. However, the software development cost (money, time, resource) for RS/RK app is still beyond the capability of many researchers. And even during the development, most of the costs has spent on communication and the duplicated works for small changes. By building a translator between medical requirement and computer language, the costs for developing and modifying apps can be significantly reduced, which enable more researches to use mobile apps.

In order to understand the JSON file structure of ResearchKit/ResearchStack Apps, I checked the JSONs configuration files every open source ResearchKit apps that are available on Github, including mPower, Asthma, GlucoSuccess, Mole Mapper, and MyHeart Counts.

App Name	Files	Surveys	Structure	Schedule file location
mPower	9	3	JSONS/(files) | JSONS/SurveyJSONs	SurveyJSONs
Asthma	10	7	JSONS/(files) | JSONS/SurveyJSONs	SurveyJSONs
GlucoSuccess	7	6	JSONS/Surveys	JSONs
Mole Mapper	0	0	Sage Bio API	-
MyHeart Counts	9	6	JSONS/(files) | JSONS/Cardiosurveys	JSONs

Table 1.1: ResearchKit App JSON Files Summery

Most of the ResearchKit apps follow the same structure, besides the Sage Bionetworks app Mole Mapper is using a totally different API. Therefore, it is possible that MART can modify most of the ResearchKit apps, or in minimum would generate a functional app using a sample app. There isn't much ResearchStack app available for checking, but basically it follows the structure of ResearchKit according to the Sample App. So it is possible that MART can edit for both RK/RS apps.

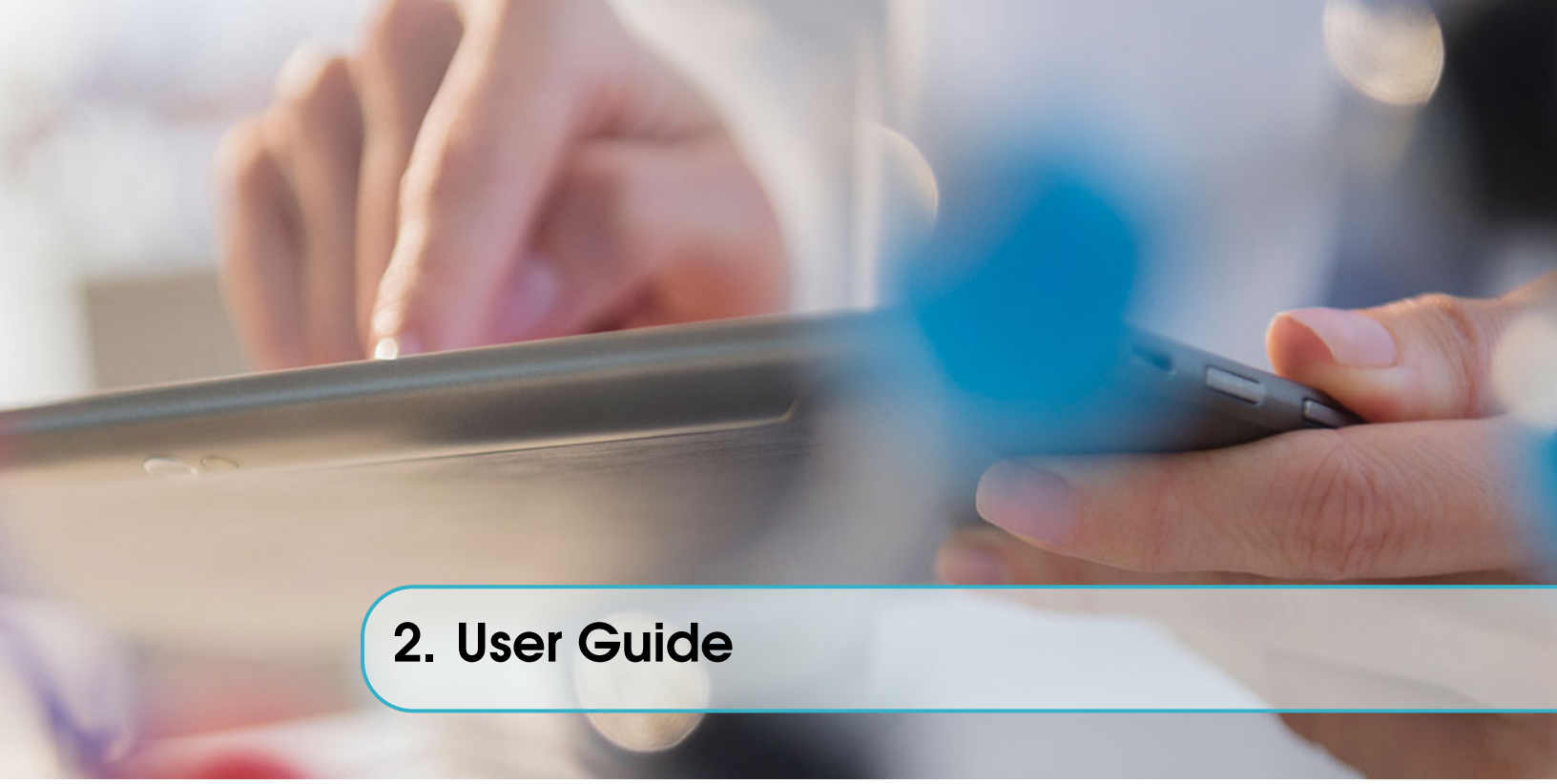
1.2.3 Researcher/Developer Communication

The communication between researchers and developers has never been easy. Email or work files doesn't always deliver the research content without much communication effort, and thus the app development process could be inefficient and inaccurate.

I interviewed with two interns at Cornell Tech, who are developing a ResearchKit app for Columbia University researchers. They use emails and spreadsheet to communicate. The spreadsheet is basically a third language that translating between medical research language and programming language, which requires twice communication effort from both researchers and developers. And whenever there is a change, they need to communicate with email first, and then check this spreadsheet and translate as many times as needed. By using MART, the communication effort should be

significantly reduced to one step – researchers make the change on MART, and then get the new app with the updated content directly.

I also interviewed the researchers and developers at Northwell. In order to provide precise information to developers, researchers would use RedCap, which is a web-based survey system that Northwell researchers has been used for years, to write the surveys. Then developers would develop the app accordingly. Some researchers tried to use online JSON file editor, but since these editors are not exactly for a mobile app or research, the generated JSON files still cannot be used directly.



2. User Guide

This is a guide for any user who is using MART to develop an iOS research app based on the iOS Sample app in ResearchSuite. The user doesn't need any coding experience to follow the instruction. But the user do need to have Xcode(Version 7.3 or above) installed to generate an iOS app. Download Xcode from Mac App Store.

For MART Demo Video, please go to: `https://youtu.be/41xe61KN5gY`.

2.1 Sample App

First, make sure you have Xcode installed. Then go to the Mart repo on Github: `https://github.com/sunnyliyanbo1357/MART` , clone or download the repo.

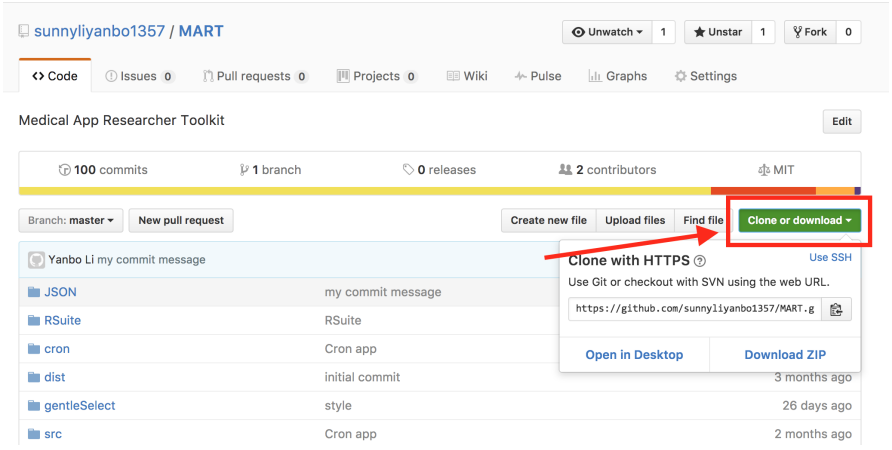


Figure 2.1: Github MART Repo

2.1.1 Build iOS App

Open RSuite folder, double clicked **RSuite.xcodeproj**. In the opened Xcode window, click the "play" button on the upper left corner to build the app.

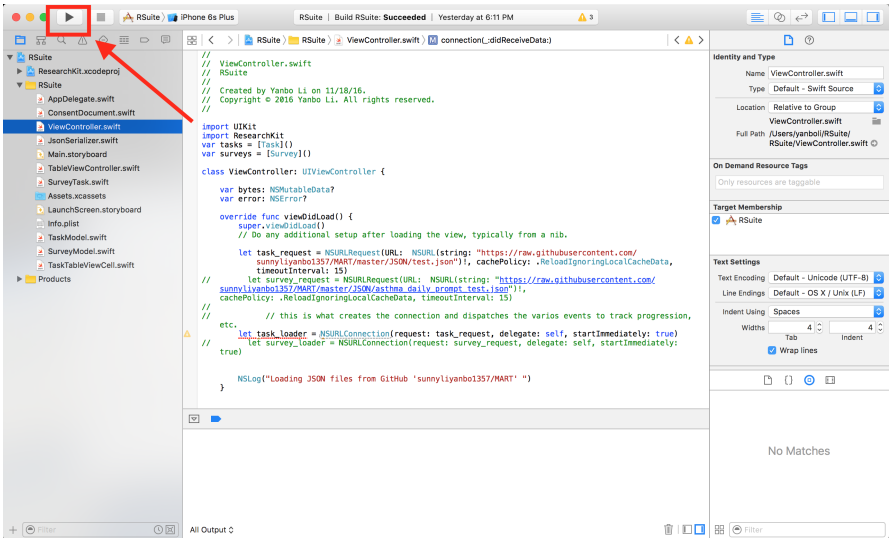
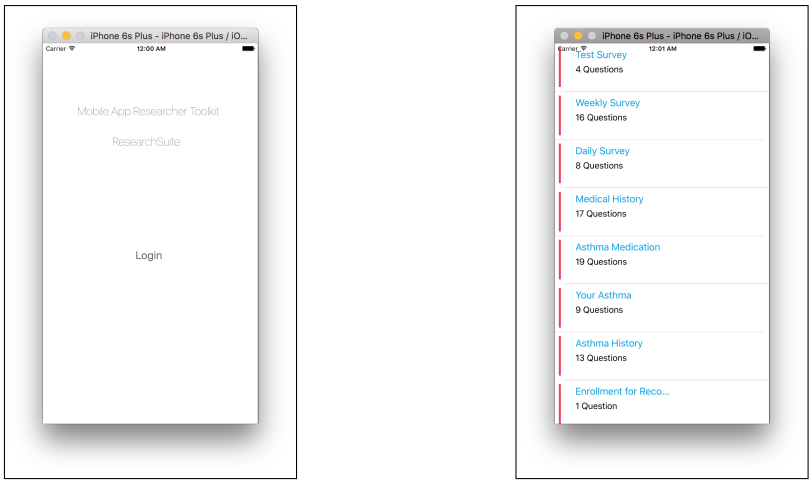


Figure 2.2: Run the project in Xcode

Then you will see an iPhone simulator pops up, and then it will opens the ResearchSuite iOS app automatically. Click **Login** to preview the ResearchSuite App.



(a) iOS App Welcome View (b) iOS App Task List View

Figure 2.3: App Preview

2.2 MART

Go to `https://sunnyliyanbo1357.github.io/MART/`. Now you can view and edit the root JSON file *tasks_and_schedules.json*. The task list shown in the iOS App is defined in *tasks_and_schedules.json* file, which is the root JSON file. Full JSON file structure can be illustrated as below.

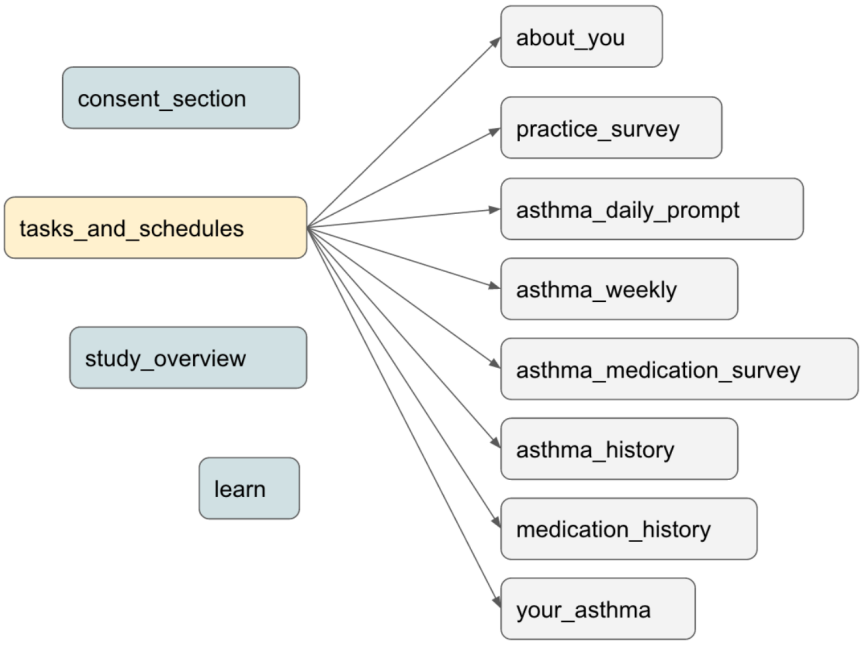


Figure 2.4: JSON Files Structure

As shown in the figure 2.4, *tasks_and_schedules.json* is the entry point in MART. From here, you can go to each survey respectively by clicking Edit Survey. Now since only the Asthma Daily Survey is included, all links is redirect to the Asthma Daily Prompt.

2.2.1 Edit Task List

The JSON file is structured like a tree. There are 9 *Tasks* in the list. You can fold the Task information by clicking the square triangle button on the right *Task1*, or delete this block by clicking 'x', or move this block by one up or down using the up and down arrow. By clicking 'JSON', you will see the JSON file of this block. 'Properties' shows all JSON schema in this block.

Figure 2.5: Block manipulating button

For each Task block, it has scheduleType, delay, scheduleString, and task detail. Each of these is a JSON schema that represents one vital of the task. You can find the detailed description of each JSON schema at the end of this chapter.

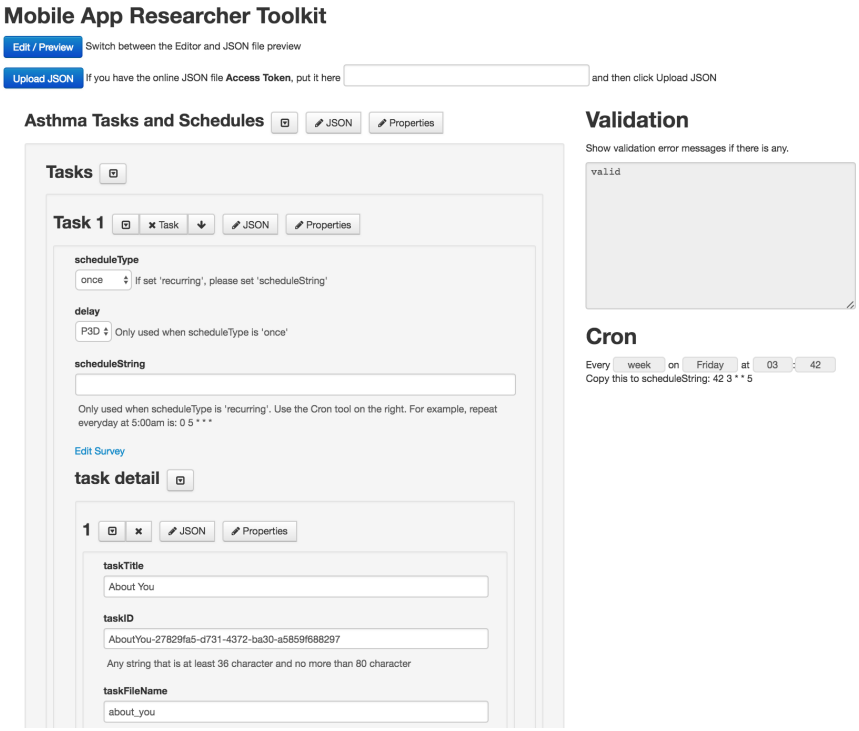


Figure 2.6: Editor

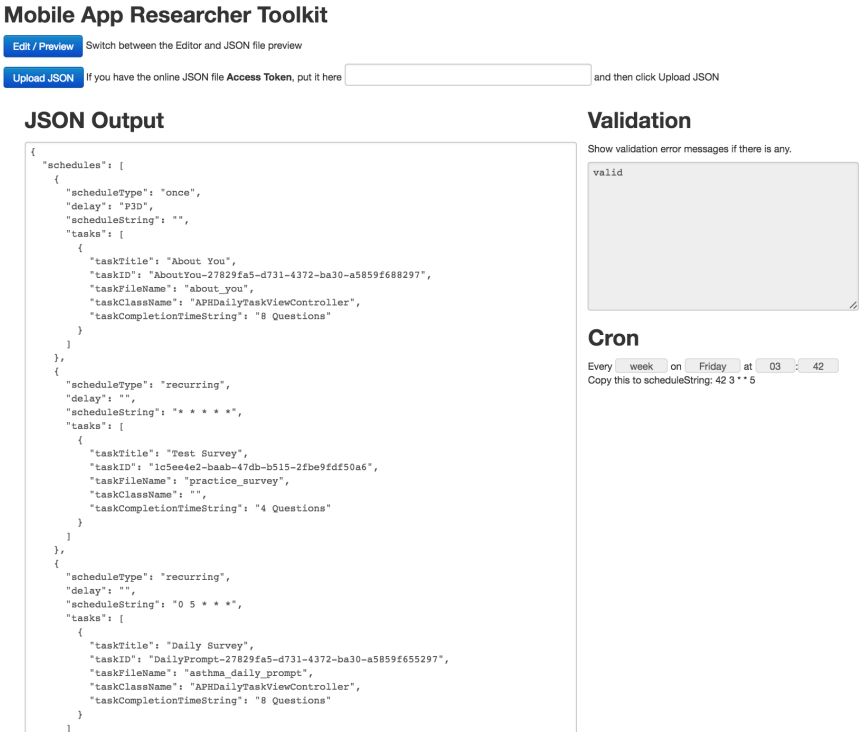


Figure 2.7: JSON Preview

2.2.2 Preview JSON

After you finish editing, you can click on the *Edit / Preview* to preview JSON file as shown in figure 2.7, or change back to the editor.

2.2.3 Upload JSON File

After finish editing, you can either copy the JSON to your local JSON file in the project folder, or type in the Access Token to upload the JSON file to the online repo. If you have successfully uploaded the JSON file, rebuild your sample app in Xcode, then you will have the app with updated content, since it reads directly from the new JSON file in the online repo.

Please email *yl2556@cornell.edu* and ask for Access Token.

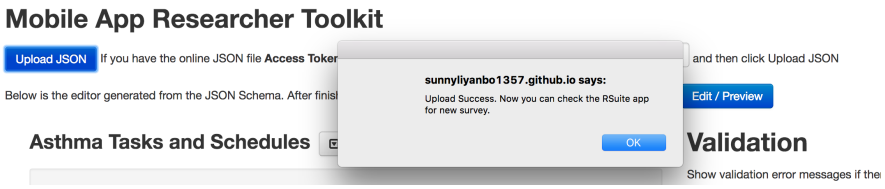


Figure 2.8: Upload Success

2.2.4 Rebuild The App

Now, in Xcode, rebuild your app by clicking the triangle button showed at the beginning of this chapter. You should be able to see the new content.

If the content doesn't change, this could result from that the system stored last JSON file in local cache instead of the new JSON files from the repo. Wait for up to 5mins for the cache to expire and try again.

2.3 JSON Schema Description

File Name	Schema	Values(e.g)	Description
tasks_and_schedules	scheduleType	once/recurring	repeating task or only show once
tasks_and_schedules	delay	P2D/P3D/P4D	not used. See PFC 5545 Standard
tasks_and_schedules	scheduleString	e.g. 05***	Cron String. Use the Cron tool on the right. Must be set if *scheduleType* is set to *recurring*.
tasks_and_schedules	taskTitle	e.g About You	readable title shown on the list
tasks_and_schedules	taskID	e.g AboutYou-27829fa5-d731-4372-ba30-a5859f688297	Task unique identifier
tasks_and_schedules	taskFileName	e.g about_you	not used. Code file name.
tasks_and_schedules	taskClassName	e.g APHDailyTaskViewController	class name of the functional code for this task
asthma_daily_prompt	identifier	AsthmaDailyPrompt	unique identifier for this survey
asthma_daily_prompt	type	Survey	Task type. Mostly Survey
asthma_daily_prompt	name	AsthmaDailyPrompt	readable name of the task
asthma_daily_prompt	Question-identifier	day_symptoms	Unique identifier for this question in this survey
asthma_daily_prompt	Question-prompt	e.g In the last 24 hours, did you have any daytime asthma symptoms (cough, wheeze, shortness of breath or chest tightness)?	the question wording
asthma_daily_prompt	Question-uiHint	Checkbox/numberfiled/MultiValueConstrains	What answer UI is this question.
asthma_daily_prompt	Question-type	SurveyQuestion	Schema type.
asthma_daily_prompt	Question-guid	e.g e872f8fa-c157-457b-890f-9e28eeed6efa	Unique question global identifier.
asthma_daily_prompt	Constraints-dataType	boolean/integer	the data stored in database. boolean for Yes or No answers. integer of integer and multi-value answers
asthma_daily_prompt	Constraints-type	booleanConstraints / integerConstraints / MultiValueConstrains	answer data constrains according to data type
asthma_daily_prompt	Constraints-allowMultiple	true/false	allow multiple answer or not
asthma_daily_prompt	Skipping Rules-operator	eq	'eq' means equals to
asthma_daily_prompt	Skipping Rules-skipTo	e.g get_worse	skip to the question with this identifier
asthma_daily_prompt	Skipping Rules-type	SurveyRule	schema type
asthma_daily_prompt	Skipping Rules-value	e.g 0	skip when the answer data equals to this value
asthma_daily_prompt	Enumeration-type	SurveyQuestionOption	schema type
asthma_daily_prompt	Enumeration-value	e.g 1	the integer data representing this answer in database
asthma_daily_prompt	Enumeration-label	e.g Exercise	specific answer wording

Table 2.1: ResearchKit App JSON Files Summery

3. Developer Document

The system is really simple, MART is serials of single pages with no backend. ResearchSuite has one iOS sample app I developed for MART illustration, and one Android ResearchStack Sample App that Cornell Tech developed and tested on MART. The system involved researcher, developer, and end-users. System structure is shown in figure 3.1.

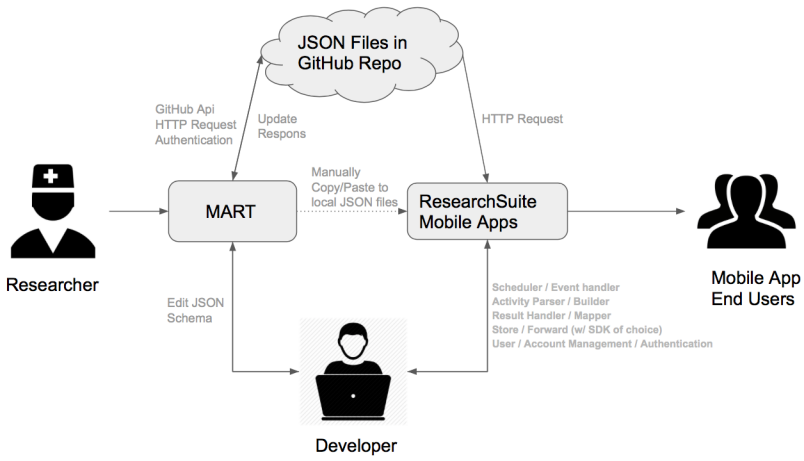


Figure 3.1: System Structure

3.1 MART

MART has three main features: JSON editor, tools, and upload. JSON editor was developed based on an open source library JSON Editor.

3.1.1 JSON Editor

The main logic of JSON Editor is in *dist/jsoneditor.js*. The editor reads the schema rules from a schema JSON file wrote by developer (see *JSON/tasks_and_schedules_schema.json*). Then it translates the schema according to *type* and put the values in textboxes. Then it generates the JSON Output and updates it whenever it detects changes. For more detailed API explanation, see `https://github.com/jdorn/json-editor/`. For the use case in this project, see*index.html*

3.1.2 Tools

According to the RFC standard for scheduling in the mobile system, reminders are set as Cron string, which is a 5-digit string that each represents a time period, so together it shows cycle interval. Cron is really hard for researchers to read and write, so I include a cron tool on the right of the page. It basically calculates each digit from each selector values, and generate the cron string for researchers to copy and paste.

```
function getCurrentValue(c) {
        var b = c.data("block");
        var min = hour = day = month = dow = "*";
        var selectedPeriod = b["period"].find("select").val();
        switch (selectedPeriod) {
            case "minute":
                break;

            case "hour":
                min = b["mins"].find("select").val();
                break;

            case "day":
                min  = b["time"].find("select.cron-time-min").val();
                hour = b["time"].find("select.cron-time-hour").val();
                break;

            case "week":
                min  = b["time"].find("select.cron-time-min").val();
                hour = b["time"].find("select.cron-time-hour").val();
                dow  =  b["dow"].find("select").val();
                break;

            case "month":
                min  = b["time"].find("select.cron-time-min").val();
                hour = b["time"].find("select.cron-time-hour").val();
                day  = b["dom"].find("select").val();
                break;

            case "year":
                min  = b["time"].find("select.cron-time-min").val();
                hour = b["time"].find("select.cron-time-hour").val();
                day  = b["dom"].find("select").val();
                month = b["month"].find("select").val();
                break;

            default:
                // we assume this only happens when customValues is set
                return selectedPeriod;
```

```
        }
        return [min, hour, day, month, dow].join(" ");
    }
```

Listing 3.1: Cron Tool get value

3.1.3 Upload and Security

GitHub opened an API allowing repo content to be updated with HTTP Request. The user name and password information in authentication process can be replaced with an access token, which should not be shown in the repo. By giving this access token, the repo owner basically allows any change in public repos. Therefore the access token should not be given out randomly or shown in GitHub Repo.

In this specific HTTP Request, 'header' is the authentication information; 'content' is the 64Based encoded JSON output; 'sha' is a string representing target file content, which can be gotten from an HTTP request response.

```
$upload_JSON_button.addEventListener('click',function(){

  var uploadURL ="https://api.github.com/repos/sunnyliyanbo1357/MART/
      contents/JSON/test.json";
  // var content = $.base64.encode("this is a test");
  var content = btoa($output.value);
  console.log(atob(content));


  //Get original Sha
  $.ajax({
    url: uploadURL,
    beforeSend: function(xhr) {
      xhr.setRequestHeader('Authorization', "token " + $accessToken.
          value);
    },
    success: function(response) {
      console.log( response );
      //PUT new content
      $.ajax({
        type: "PUT",
        url: uploadURL,
        beforeSend: function(xhr) {
          xhr.setRequestHeader('Authorization', "token " + $accessToken.
              value);
        },
        contentType: "application/json",
        dataType: "json",
        data: JSON.stringify({
          "message": "my commit message",
          "committer": {"name": "Yanbo Li","email": "sunnyliyanbo@gmail.
              com"},
          "content" : content,
          "sha": response.sha}),

        success: function(response) {
          alert('Upload Success. Now you can check the RSuite app for
              new survey.');
```

```
34                console.log( response );
35              },
36              error: function(response) {
37                alert('Invalid Token');
38                console.log( response );
39              }
40            })
41          },
42          error: function(response) {
43            alert('Invalid Token');
44            console.log( response );
45          }
46        })
47      })
```

Listing 3.2: HTTP Request

3.2 ResearchSuite

ResearchSuite is a suite consist of sample research mobile apps for both iOS and Android platform.

3.2.1 ResearchKit Sample App

ResearchKit Sample App (RApp) use GitHub API to read raw JSON file content from the same location that MART uses for uploading. Then RApp parses the JSON response to different data models (TaskModel, SurveyModels) and display them within ResearchKit framework. Please see *RSuite* for the code.

3.2.2 ResaerchStack Sample App

Please refer to ResearchStack GitHub Repo for more information: `https://github.com/ResearchStack/ResearchStack`.

4. User Testing And Future Work

4.1 User Testing

4.1.1 OHSU

As soon as I have an MVP or MART, I started the interview and user testing with researchers, so that I can develop based on their feedback and finally develop something can actually being used. OHSU was one of the first organization I reached out for user testing.

It was a 45mins video interview with one researcher who mainly works with survey data in OHSU. I first asked her job responsibilities and any inconvenience in her daily work. Then I introduce her to MART (it was in very early stage, which can only edit one JSON file) and get some really useful feedback. For example, she got confused by some schema. Also, she couldn't find the place to change answers at the first glance. After my short explanation, she understood MART quickly and answered my question of how to skip questions correctly.

Based on her feedback, I added the schema descriptions and also separated the JSON output and editor view so that users don't get confused. I also added the validation and error alert so that user can get real-time feedback.

4.1.2 Northwell Interview

In this interview, I coordinated with five researchers from Northwell to learn their feedback on Mobile App Researcher Toolkit (MART), a web app that provides an easy way for researchers to modify their medical research app.

To start with, I shortly introduced myself and the purpose of this interview. Below is my script:

Thank you so much for your time. My name is Yanbo Li, a master student from Cornell Tech. I am helping Michael and James, developers of Northwell research mobile app, to test an additional tool called MART for researchers to better communicate with developers, and eventually, by using MART, a researcher can build an app without the need to know how to write code. This interview is only for educational use within Cornell Tech and Weill Cornell Medicine, and the recorded video/audio will only be used by myself to evaluate this MART system.

Then I asked their viewpoints on the technology they used for research in general. The answers

were surprising. They use mature online questionnaire service, such as RedCap, to build a survey and send out URL links in emails. These email addresses are from their patient database, so there is no need to go public or recruit participants. Frederick, the director of digital health interventions, said that he was aware of the existence of mobile apps that a lot of other hospitals are using, but chose not to use mobile app massively because, a) there is already well-developed technology (i.e. RedCap) that is fully functional; b) accessing mobile apps depend on largely on the smart devices with difference operation systems (iOS and Android), but there is no limitation on web questionnaire; c) it takes more effort for the user to conduct a survey on mobile. They need to go to app store, download the app (which is mission impossible for many small memory device users), figure out how to use this app, and then participate in the study; d) in general, researchers don't need users behavioral data such as location, heart rate, steps, sleep, from the mobile phone. If they really need these data, researchers would gather them from questionnaires.

However, they are developing one mobile app with Cornell Tech, but that is not for specific studies. They wanted the app to gather as much user data as possible so that the app can later be modified to fit different research use easily. This is exactly how MART can be useful.

I briefly introduced how MART can easily solve the problem, followed by a demo. Then I asked for feedback based on the SUS questionnaire. In short, they felt MART system is complicated and "defiantly need training", or at least an introduction video to start with. It doesn't necessarily mean that MART is awfully designed. According to a female researcher, RedCap has training sessions every 3 months. But MART should have the ease to use in order to obtain early adopters, i.e. those who chose MART over other web questionnaire systems.

In addition to MART itself, researchers expressed concerns with mobile research apps and gave insightful advice. First, some questionnaires are scheduled randomly within a time range. Based on that, apps can push task reminder only when the user is not in a call or driving (navigating with map app) within a time range. Second, servers should be able to send reminder according to time zone. For example, if a researcher wants the participants to take a survey at 9 am but they are in different time zone, the server should send reminder accordingly. Third, researchers want to have two-way communication with the participants within the app. In short, this interview was very insightful, gathering much information no only within MART, but also expectations and concerns within the mobile app for research purpose in general. These are valuable guidelines in research software development.

4.2 Future Work

This is a meaningful project that could significantly improve the research and development costs, which push the mobilization of research study forward. I was surprised that much research could have been much more efficient by using a mobile app or better technology, but didn't because the adopting cost was too high. Therefore, the gap between these research and technology could be minimized by building an easier to use and ready to use MART system. Both MART and ResearchSuite can have some improvement:

4.2.1 MART

- A mobile app preview as reference
- Finish the rest of survey editor
- Other editing UI/UX design such as Undo/Redo
- Better Error alert

4.2.2 ResearchSuite

- ResearchKit Tasks
- Concent Form
- Result mapping
- Design elements in JSON
- Easier and safer access to online JSON files

4.3 Useful Information

- MART: `https://sunnyliyanbo1357.github.io/MART/`
- MART Demo Video: `https://youtu.be/41xe61KN5gY`
- MART GitHub Repo: `https://github.com/sunnyliyanbo1357/MART`
- JSON Editor API: `https://github.com/jdorn/json-editor`
- ResearchStack Sample App GitHub: `https://github.com/ResearchStack/SampleApp`
- Developer contact information: yl2556@cornell.edu. Please feel free to email me with any question.

5. Acknowledgment

My sincere thanks to my mentor Deborah Estrin, who always gave me the most insightful opinion and correct direction. Especial thanks to Michael Carroll, who originally came up with the idea of JSON Editor for researchers, and constantly provided me with technical advice, researcher connections, and increasing interest in health tech. And real appreciation to Nicola Dell, my HCI class instructor and HCI adviser.