TigerAware Dashboard: An Improved Survey Generation and Response Visualization Dashboard

A Project

Presented to

The Faculty of the Graduate School

At the University of Missouri

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

Implemented and Defended by:

Jayanth Kanugo

Prof. Yi Shang, Advisor

May 2018

TABLE OF CONTENTS

[List of Figures iii](#_Toc510712120)

[List of Tables v](#_Toc510712121)

[Acknowledgements vi](#_Toc510712122)

[Abstract vii](#_Toc510712123)

[1. Introduction 1](#_Toc510712124)

[1.1. Problem Description 2](#_Toc510712125)

[1.2. Proposed Solution 4](#_Toc510712126)

[2. Background and Related Works 8](#_Toc510712127)

[3. TigerAware Design 10](#_Toc510712128)

[3.1. Survey Grammar 10](#_Toc510712129)

[3.2. TigerAware Architecture 13](#_Toc510712130)

[3.2.1. Central Database 14](#_Toc510712131)

[3.2.2. Interpreters 15](#_Toc510712132)

[3.3. New TigerAware Dashboard Design 16](#_Toc510712133)

[3.3.1. Dashboard Design 16](#_Toc510712134)

[3.3.2. Dashboard Architecture 18](#_Toc510712135)

[3.3.3. Technology Stack 19](#_Toc510712136)

[3.3.4. Front-End and Client Side 19](#_Toc510712137)

[3.3.5. Back-End and Server Side 19](#_Toc510712138)

[3.3.6. Other utilities 19](#_Toc510712139)

[4. System Implementation 21](#_Toc510712140)

[4.1. Server side Implementation 22](#_Toc510712141)

[4.1.1. Notification Scheduler 22](#_Toc510712142)

[4.1.2. Authentication Module 24](#_Toc510712143)

[4.1.3. Survey Module 26](#_Toc510712144)

[4.1.4. Users Module 27](#_Toc510712145)

[4.1.5. Utility Scripts 29](#_Toc510712146)

[4.2. TigerAware Web Dashboard Modules 35](#_Toc510712147)

[4.2.1. Account Management Module 36](#_Toc510712148)

[4.2.2. Survey Builder Module 37](#_Toc510712149)

[4.2.3. Survey Administration Module 42](#_Toc510712150)

[4.2.4. Survey Visualization Module 43](#_Toc510712151)

[4.2.5. Template Module 45](#_Toc510712152)

[5. TigerAware Applications 48](#_Toc510712153)

[5.1. Privacy and Anonymity 48](#_Toc510712154)

[5.2. Focus Group 48](#_Toc510712155)

[5.3. Google Assistant Based Diabetes Self-Management Study 50](#_Toc510712156)

[5.4. Bilingualism and Aphasia Study 50](#_Toc510712157)

[5.5. Driving After Drinking Alcohol Study 51](#_Toc510712158)

[5.6. Implicit Racial Bias Study 52](#_Toc510712159)

[6. Conclusion and Future Work 54](#_Toc510712160)

[6.1. Future Work 54](#_Toc510712161)

[7. References 56](#_Toc510712162)

# List of Figures

Figure 1 TigerAware System Architecture and Components 5

Figure 2 An Example of a Step in TigerAware 11

Figure 3 An Example of a Conditional Step in TigerAware 11

Figure 4 Survey Blueprint Structure in TigerAware 11

Figure 5 TigerAware Dashboard Architecture Diagram 18

Figure 6 TigerAware Dashboard Implementation 21

Figure 7 Illustration of Notification Scheduler 22

Figure 8 Background Listener for Change in Notifications 23

Figure 9 Illustration of OAuth 2.0 Flow 25

Figure 10 XML Response from PROMIS Measures API with List of Forms 30

Figure 11 XML Response with List of Questions in a Form 30

Figure 12 Pseudocode to Convert XML file to TigerAware Survey Grammar 31

Figure 13 Pseudocode to Extract Step Type 33

Figure 14 Pseudocode to Extract Step Constraints 34

Figure 15 Pseudocode to Extract Condition on a Step 35

Figure 16 Login and Registration Interface 36

Figure 17 Code for Registering New User 37

Figure 18 Survey Builder Interface 38

Figure 19 Create Question Modal 40

Figure 20 Notification Builder Interface 41

Figure 21 Add New Notification Modals 42

Figure 22 Survey Administration Interface 43

Figure 23 Survey Overview Interface with Ability to Activate or Deactivate a Survey 43

Figure 24 Different Visualization Available in Survey Visualization Module rendered based on Step Type 44

Figure 25 Visualization for Location Metadata representing Where Survey was Taken 45

Figure 26 Template Dashboard Interface 46

Figure 27 Autofill Enabled Modal to Search and Add Templates to a Survey 47

# List of Tables

Table 1 Comparing Standalone Systems with Exsisting TigerAware Dashboard 9

Table 2 Survey Grammar that Defines Survey Syntax 13

Table 3 List of Survey Module API End Points and Their Reponses 28

Table 4 List of User Module End Points and Their Responses 29

Table 5 Mapping of Headers in PhenX CSV file to TigerAware step-attributes 33

Table 6 New TigerAware Dashboard Features Compared to Exisiting Systems 49

# Acknowledgements

Firstly, I would like to thank Prof. Yi Shang for his mentorship, guidance and support throughout my graduate school and through this research project. I would also like to thank Tim and Denis for the opportunity to collaborate with the Department of Psychological Sciences at University of Missouri and their valuable feedback in development of various features in this project.

I would also like to thank everyone from the Computer Science lab, especially Nickolas Wergeles, Luke Guerden, William Morrison, Connor Rowland and Siyang Li for sharing their experiences and being an integral part of this project. I would also like to show appreciation to Jodie Lenser and Shirley Holdmeier for guiding me through my application process to Mizzou as well as advising me on my courses.

I would also like to thank Yeshwanthi Pachalla, who inspired my application to Mizzou. Without her helpful advice, guidance and support throughout my application process, I would not have been a part of Mizzou.

Finally, I would like to thank my family and friends. Without their support, trust and belief in me, I would not be where I am today. I hope I have done you proud.

# Abstract

From health to community assessment, mobile phones have become a cornerstone in many research areas for data collection. However, smartphone-based studies are difficult to develop and deploy, as they are usually standalone applications with dedicated development team that consume a major chunk of research budgets. Current platforms for data collection, are limiting researchers by just providing a standalone platform for data collection without the flexibility to modify, update and integrate new sources of information collection. To reduce the burden of costs associated with these research studies and provide flexibility, a new platform called TigerAware is developed. TigerAware is designed to be a generic and customizable platform which allows researchers to create surveys which can collect a wide range of data including but not limited to question responses, external sensor data and other metadata which can help researchers to get better insight for future data analytics.

In this Masters project a real-time web application was created for TigerAware to help researchers build surveys and control who can participate in a survey or deploy to a public audience, with additional ability to set scheduled and random notifications. The dashboard also provides researcher capabilities to structure, organize, analyze and visualize their results. TigerAware Dashboard is a full stack application consisting of a combination of RESTful Web services, an OAuth 2.0 endpoint and other background services hosted on Amazon EC-2 and consumed by an AngularJS front end framework, which uses modular design methodology along with various open source libraries like Highcharts.js to display rich and interactive graphs.

The architecture of the platform as a whole is highly modular and generic in design allowing TigerAware to be used in studies like Driving After Drinking Alcohol Study and plans for several other pilot studies in pipeline.

# Introduction

Researchers in different disciplines are always looking for new ways to gather novel, insightful data to draw more robust conclusions. Over the past decade, this seek has led researchers to rely heavily on the most ubiquitous device available--- smartphones. Smart phones have quickly become predominant in the United States, with 77% of the population using it, up from just 35% in 2011 [1]. These smart devices are even more predominant in younger populations, with 92% of Americans aged 18 to 29 years owning one [1]. Smartphones are cheap and are a reliable source to collect information without a major learning curve. These modern devices also come equipped with a wide array of specialized sensors along with ambient light sensor, GPS, proximity sensor, accelerometer, digital compass, microphone, and camera. Researchers can use each of these data sources to explore new scientific questions.

Yet using these devices in a research setting pose as a significant challenge because of unique needs of each study which requires custom software and a committed development team. For example, a medical researcher may need to collect answers to one set of questions at certain time intervals and be notified when a patient enters an actionable survey response. In a different scenario, a psychologist may need to collect answers to another set of questions and also the ambient light data. Upon data arrival, researchers may also need to visualize and interact with data in different ways. The software development to accommodate these needs is time consuming and expensive, diverting resources from the research study. To greatly reduce the software development and budgetary restrictions TigerAware is created. It is flexible and is capable in conducting smart device-based research. As the application is on a multi-tenant platform, the researchers can easily author, deploy, and manage survey and sensor-data based studies.

TigerAware can be easily incorporated into a clinical study with little or no modifications to the existing workflow. It also reduces the necessity of allocating significant funds for development of IT systems. Furthermore, the platform is highly configurable, allowing researchers to integrate novel data sources and interfaces. For example, the integration of blood alcohol level data type and voice based assistant interface were implemented with minimal effort.

TigerAware has significant advantages over previous commercial and academic research data collection platforms. By including existing state-of-the-art features and considering various sensors used in existing research studies, we were able to greatly improve upon existing platforms. In addition, data collected from the sensors and survey responses can be visualized on TigerAware's dashboard in real time. As a result, TigerAware is developed to offer a generic, customizable survey creation and deployment, data collection from survey and sensors for researchers who want to use smartphone-based surveys in conjunction with various internal or external data sources including wireless-wearable sensors and Internet of Things (IoT) devices. TigerAware is highly modular and uses advanced Web and Mobile technologies to incorporate diverse data sources with a rich set of survey question types, requiring little development work by researchers for their studies. TigerAware has been applied to a focus group and several pilot studies and shown excellent capabilities to be easily adaptable and deployable for new types of data collection tasks and a wide range of studies.

## Problem Description

Researchers usually tackle the need to collect empirical data by either incorporating an off-the-shelf solution which are commercially available or creating a custom standalone solution each with their own set of drawbacks.

Researchers prefer creating a standalone version of their application and dashboard to deliver a survey and integrate other external sensors necessary for a study, it could take anywhere between 6 months to develop a capable smartphone application and similar amount of time to develop a real time dashboard based on the number of dedicated developers working closely with the research team which consumes a significant portion of research budget. Any major change in the application would need a redeployment of application leading to significant delay in collection of information.

Commercially available systems on the other hand come with their own set of drawbacks. For example, REDCap which is widely used for generating surveys which help in conducting clinical studies is highly capable platform but is limited to qualitative data collection with question for external data to have of information that can be collected by on the platform. The limitation can be attributed to the lack of a capable mobile application which collect such information. Other commercially available survey development applications like SurveyMonkey, SurveyNuts, etc. [2] are readily available in the market for creating and deploying surveys for data collection. However, they come at a premium. These off-the-shelf products provide numerous features, which may be limited when creating a custom application. Firstly, many of these applications could cost from $10-$70 per month depending on certain user requirements, deployment platforms, and the application itself [2]. And the ones that come free of charge only provide limited features and only for a certain trial period. They also come with a restriction on customizability [3]. Similarly, applications such as SurveyMonkey provide data analysis options with graphs and charts but do not readily provide a method of verifying the data as per our requirements. The lowest price plan in SurveyMonkey that generates graphs, charts and downloading responses and reports in Comma-Separated Value (CSV) in called the Standard Plan, which costs $35 per month [4]. Of the two-aforementioned commercial platform only SurveyMonkey has the ability to integrate and deploy surveys to a mobile client, yet there is no way for users to be notified about the time to take a survey.

## Proposed Solution

As smartphones have become ubiquitous, they have allowed for novel methods of investigating a broad array of fields ranging from social science, to psychological science, to clinical research. Examples include ecological momentary assessment (EMA) for psychological studies [5, 6, 7], clinical assessment and self-management [8], and remote monitoring [9]. Though each of these domains focuses on a different research question, they each have common technical components. These often include:

* Recurring data input by a user on a mobile device, such as providing answers to survey questions.
* Collecting data from a mobile device's internal sensors or external wirelessly connected sensors.
* Managing multiple participants and their data across time.
* Visualizing and analyzing collected data on a Web dashboard.

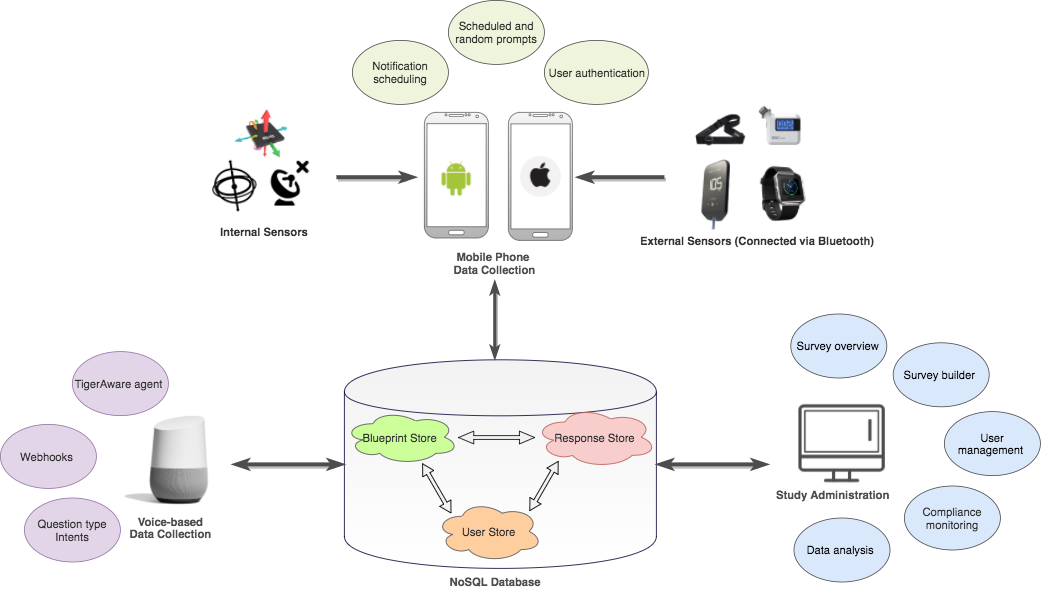


Figure 1 TigerAware System Architecture and Components

Based on the understanding of other systems and their technical components a combination of data collection systems and a central web dashboard were proposed as a solution. The mobile phone or voice-controlled data collection side of the platform is used in field to collect all responses and upload them directly to our NoSQL database, while the study administration, creation and management is handled on the web dashboard.

During the development of the dashboard, constant meetings were held with the client to discuss and understand the complexity of their surveys. Applying an incremental development strategy, the dashboard was developed by creating incremental versions where features of all the Study Administration part of the platform were upgraded without affecting any previous deployed versions of the platform to accommodate a wide variety of surveys structures with multiple branches and complex question flow. Various suggestions were made by focus groups, of which the ability incorporate pre-existing surveys from standard health measures constantly used in field of study would greatly improve the deployment of surveys at a faster rate. To accommodate this: PROMIS® (Patient-Reported Outcomes Measurement Information System) a set of person-centered measures helped in evaluating and monitoring physical, mental, and social health in adults and children; PhenX Toolkit a catalog of recommended and standard measures of phenotypes and environmental exposures for use in biomedical research were added to database to improve the time taken to create a survey.

The web dashboard, was developed using latest technologies based of JavaScript and Firebase stack (AngularJS, ExpressJS, Firebase and node.js). The dashboard provides researchers a stage to create, deploy, modify any survey at any point in time and also have the ability to verify data in real-time with the help of visualization, as well as functionalities such as downloading data as CSV files, setting up random notifications across time zones. To protect the sensitive data, a way to assign user roles and access restrictions based should be available on the web dashboard.

The backend of the platform relies heavily on node.js as the runtime environment which made it easier to setup Application Programming Interfaces or API’s which improves the ability of the platform to perform various other tasks. It also made it easier to integrate Google Home, a popular home device which uses Google Assistants voice-based feature to act as a virtual assistant and ask questions and gather responses by deploying the surveys created using the survey engine. An end point was implemented which authenticates the users using custom authentication mechanism instead of relying on Google based login credentials which doesn’t restrict the users to specifically use Google Accounts to use the service. The authentication was based on an open standard protocol OAuth 2.0 where the server handles the necessary authorization workflows and creates a unique access token for every user on database when necessary.

The new and improved TigerAware dashboard improves the research cycle by providing a one stop shop for researchers with the following contributions.

* Create surveys and quizzes.
* Perform user management and survey administration.
* A general and easy to use set of applications for iOS, Android, and web development
* Ability to view, verify and analyze real-time data

The remainder of the report is organized as follows: In Chapter 2, related works in data collection and their strengths and limitations are discussed. The design consideration for the development cross platform survey mechanism are discussed in Chapter 3. Chapter 4 presents the implementation of the TigerAware Dashboard. The results of implementing these applications are discussed along with application and feedback for the platform in Chapter 5. The report is concluded in Chapter 6 and discusses potential future works

# Background and Related Works

Data Several cross-platform systems are currently available to suit the needs all of researchers providing components in context of a specific research question. For instance, **Psychlog** is a mobile platform designed to monitor participants' mental states, and collects users' psychological, physiological, and activity information [10]. Psychlog also allows users to enter self-report data so changes in mental state can be tracked over time. Psychlog offers advanced physiological data collection capabilities, including a real-time readout of ECG and accelerometer data on a mobile phones screen.

**REDCap** is another such clinical research tool developed by Vanderbilt University that can be used to author, deploy, and analyze a wide array of clinical studies [11]. It also uses a metadata-driven method of survey creation and deployment but is limited primarily to a medical context. REDCap allows for creation of projects backed by a SQL data store including five predefined tables such as metadata, logging, and regular data.

**AndWellness** is another well-being centered platform focusing on data collection using mobile phone for the purpose of self-monitoring [12]. AndWellness offers an advanced suite of study management tools, including mobile phone and dashboard components. AndWellness also supports research across different fields, but does not come with a builder module; thus it requires programmer’s assistance for each new study. Several community monitoring and science platforms have also been developed which allow users to author new studies dynamically. From systems enabling epidemiologists to view changing conditions throughout a community [13] to platforms monitoring community atmosphere and noise levels [14], generic surveys have also become pivotal to community monitoring efforts.

Some recently developed systems aim to synthesize device data, survey responses, and physiological data obtained from third party devices. DEMONS is an integrated framework for synthesizing active and passive data sources using smartphone data and a wearable sensor [15]. DEMONS offer a generic authoring framework for new studies. DEMONS allow researchers to author new surveys including physiological data without any programming necessary; however, DEMONS restrict researchers to using only one external device (Basis Peak fitness tracking watch). Wearable chest sensors have also been used in conjunction with a survey application to better understand mood, dysregulation [16] and alcohol consumption [17] Other work has incorporated device data and sensor data to critically evaluate how each can contribute to mobile health (mHealth) applications [18]. In each of these scenarios, survey data is combined with independent data sources collected from a mobile phone sensor, or an externally connected device (i.e. Bluetooth wearable).

Table 1 Comparing Standalone Systems with Exsisting TigerAware Dashboard

| Survey Platfroms / Functionalities | REDCap | DEMONS | AndWellness | Survey Monkey | Existing TigerAware Dashboard |
| --- | --- | --- | --- | --- | --- |
| Quiz | Yes | Yes | No | Yes | No |
| Survey | Yes | Yes | Yes | Yes | Yes |
| Response based Branching | Yes | No | Yes | Yes | No |
| Notification Scheduler | No | No | No | Email Based Notifications | No |
| Survey Visualization | No | Yes | Yes | Yes | Limited |
| External Data Sources | No | Mobile | Mobile | Mobile | Mobile |
| Survey Templates | Yes | No | No | No | No |

Each of these previous efforts proposes a system for generating custom surveys to address a specific research question, at times allowing for integrating outside data sources. These works, were essential in identifying the existing approaches, and how they can be utilized for developing the TigerAware platform. They were useful in designing an overarching tool for authoring surveys across various disciplines and yet be flexible to integrate new data sources, wireless sensors, and IoT devices build a well-rounded cross-platform survey collection system, which have been further explored in the following chapters.

# TigerAware Design

This chapter discusses the design of the web application for TigerAware as well as the factors considered in its design. The chapter also covers the benefits and drawbacks of the development technologies used for the web application.

## Survey Grammar

|  |  |
| --- | --- |
| Figure 2 An Example of a Step in TigerAware | Figure 3 An Example of a Conditional Step in TigerAware |
| Figure 4 Survey Blueprint Structure in TigerAware | |

Research study is defined as a collection of survey responses gathered over time. Each survey is composed of a series of questions which are formally defined by extending the JSON format to create a survey grammar. Table 1 shows the rules for constructing a survey including two step types: a conditional-step shown in Figure 3 and a survey step shown in Figure 2 Steps are combined to form a survey blueprint as in Figure 4 and a combination of these steps can create any type of survey.

Table 2 Survey Grammar that Defines Survey Syntax

|  |  |
| --- | --- |
| ***survey*** | *[steps]* |
| ***steps*** | *conditional-step*  *step*  *step, steps*  *conditional-step, steps* |
| ***conditional-step*** | *{ conditional-step-attributes }* |
| ***conditional-step-attributes*** | *conditional-pairs, conditional-step-attributes* |
| ***conditional-pairs*** | *‘title’: string*  *‘id’: string*  *‘subtitle’: string*  *‘type’: conditional-types*  *‘choices’: choices-value*  *‘conditions’: conditions-value*  *‘conditionalDefault’: string* |
| ***choices-value*** | *array*  *null* |
| ***conditions-value*** | *[ condition ]*  *null* |
| ***conditional-types*** | *“BAC”*  *“ContinuousScale”*  *“MulitpleChoice”*  *“Scale”*  *“YesNo”* |
| ***condition*** | *{ “trigger”: value, “toID” : string }*  *{ “trigger”: value, “toID” : string }, condition* |
| ***step*** | *{ step-attributes }* |
| ***step-attributes*** | *step-pairs*  *step-pairs, step-attributes* |
| ***step-pairs*** | *‘title’: string*  *‘id’: string*  *‘subtitle’: string*  *‘type’: type-values*  *‘conditionalDefault’: string* |
| ***type-values*** | *“TimeInt”*  *“TextFeild”*  *“TextSlide”* |

## TigerAware Architecture

TigerAware embraces a modular and generic design in its entire system to enable broad cross platform capabilities as well as rapid development of new functionalities.

TigerAware has two major components: a central database and interpreters.

### Central Database

TigerAware's central database is a NoSQL database consisting of three key parts: a blueprint store, a response data store, and a user store. The blueprint store holds instructions to define TigerAware modules, as illustrated in Figure 4.In the current version, most modules are surveys or data collection modules. Survey instructions are comprised of information such as the survey name and instructions to make each step in the survey (survey blueprint). Each step in a survey could be a text slide, which delineates a section of a survey, or a point of input like a yes/no question. Each step is kept as its own unit to allow the interpreter to decide whether it wants to combine the steps into a single interface or separate interfaces. Currently supported step types, and their configuration, are shown in Table 1. Each step may include configuration metadata, and all step types include step information such as the title, description, and question label which are unique within the survey. All parts of survey instructions, including the survey steps, are left generic to enable different interpretations of questions. Despite the genericness of the survey, all the information necessary to define the steps is conveyed in the instruction. Researchers can add new step types by conforming to the step type interface. These step types could be traditional survey types, such as multiple choice or Likert scale, or more advanced types, such as physiological or device data (e.g., that from a wireless breathalyzer device).

The second important component of the database is the data-store which keeps track of survey response from the mobile device. The storage of response data alongside the corresponding blueprint data is important because the blueprint data enables interpretation of the survey data.

The last element is the user-store used for storing user information. This is stored separately from the blueprint and the responses. If desired, these database parts can be stored in three distinct databases to ensure privacy of users, improve scalability, or isolate data pertinent to different use cases. In this way, the three stores function independently of one another to enable more modularity and functionality.

A NoSQL database was used because of its high levels of modularity and genericness. One impediment that was encountered when trying to develop TigerAware using a SQL database was that each survey required a different schema, since each step corresponds to a different column in a SQL style store. Moreover, we use Firebase as our NoSQL data store because it decreased development and deployment time. The process of creating a new Firebase database and connecting it to a new platform takes just a few minutes. Because the database is structured from the frontend interpretive nodes, after a Firebase database has been created and connected to the nodes, backend development is essentially complete. An added benefit is that Firebase can also be used on a large number of diverse platforms, including mobile and the web.

### Interpreters

TigerAware's interpreters are comprised of three key parts: database interface, controller, and interpretation.

The database interface reads or writes parts of the Firebase. An interpreter can read from or write to any of the three segments of the database. Because of the high level of modularity in the Firebase, the database interface segments of the interpreters are consequently highly modular and generic as well.

The second part of interpreters is the controller, which takes what has been read and builds TigerAware surveys or survey related entities. The controller is modular and comprehensive because the higher levels area based on the same characteristics. These features are also essential in the controller to maximize reusability. For example, although TigerAware's iOS app launches a survey when a user taps a survey's table view cell, a survey could be launched based on any action like a button press or a rotation of phone.

The final part of interpreters is interpretation. In iOS, ResearchKit is primarily used as the interpretation part. ResearchKit is an SDK for rapid modular development of surveys. If the controller calls for a binary answer, the interpretation part converts its call to ResearchKit instructions to produce a yes or no question and adds it to a survey. Once a survey is produced, the controller enables a user to take it at any time. When user takes the survey, ResearchKit produces a result. The controller receives the result and passes it to the database interface, which stores it in the Firebase. ResearchKit thus provides a modular protocol for interpreting survey instructions and producing survey data from those instructions. ResearchStack was used in place of ResearchKit on Android, and the TigerAware team has developed replacements for ResearchKit on platforms that have no substitutes, such as the Google Home smart speaker. Figure 1 shows an overview of the system architecture, components and how they interact with the database.

## New TigerAware Dashboard Design

### New Dashboard Design

The web dashboard will help researchers view and modify their surveys, control access to each survey, assign notifications and post data collection they will be able to gather insight into data collected through visualization module of the dashboard.

The major design considerations [19] that were considered right at the beginning of the design of the dashboard were:

1. Modularity: as TigerAware platform is designed to cater a wide range of research studies, it is highly likely that new features will be requested based on the research requirement. By following a modular design philosophy, each new component can be developed and tested in isolation before being integrated to create a desired software system.
2. Extensibility: was vital component design choices made, as the current implementation, takes into consideration future growth. The use of MVC design pattern provides a configurable model and plug-in architecture for any features or components that are being added into the system which led to the selection of AngularJS as our front-end framework.
3. Usability: Users are highly habituated to using software and there is nothing more off putting than using unintuitive software and one’s especially with a learning curve, hence making applications that use intuitive design principles is another challenge for developing web applications. Choosing the right design tools so that it allows minimum development effort and provides optimum UI design features played a great part in choosing the tech stack for the application.

### New Dashboard Architecture

The architecture of the TigerAware Dashboard comprises of a web application hosted on EC2 Instance of Amazon Web Service (AWS) EC2. The dashboard, which was developed using the a modified version of the MEAN stack, which is a combination of ExpressJS, AngularJS and node.js integrating with Firebase as the backend for the platform.

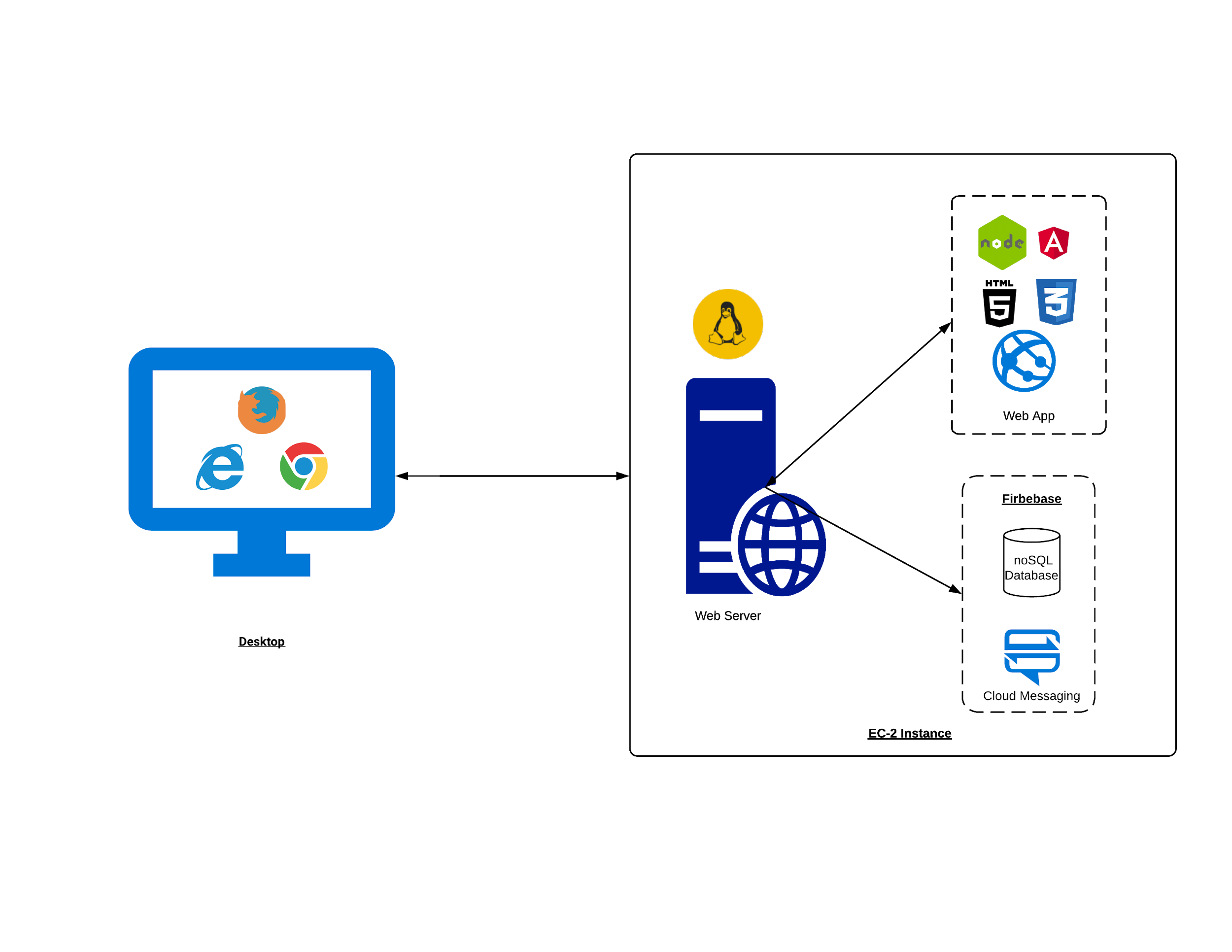


Figure 5 TigerAware Dashboard Architecture Diagram

The Figure 2 shows the overall dashboard architecture, the ExpressJS and node.js based backend provides all the necessary REST APIs which allow for information to be retrieved, saved, updated and transformed on the front end. The frontend views which can be accessed by any web browser were created following a MVC design pattern using AngularJS provides controllers and services to consume the RESTful Web Service and manipulate the client facing UI using HTML5 directives and Materialize CSS design. Highcharts.js forms a crucial building block of the graphing mechanism in for response visualizations.

### Technology Stack

There were multiple tools used for designing, developing and testing the application. They can be categorized into the front-end and client side, which were used for displaying the interface on the web dashboard and the back-end and server side, which are the storage, and languages used to interact with the storage; and other utilities, like the software programs and cloud services.

### Front-End and Client Side

1. HTML5: Markup Language on which the front-end interface is based for both the web dashboard.
2. CSS3: Materialize CSS implements the responsive design on the web dashboard
3. Javascript: AngularJS framework by Google, provides increased interactivity and background processing, with fewer page reloads along with modular features.

### Back-End and Server Side

1. node.js: A server side scripting language capable of multi-processor deployment.
2. Firebase: Scripting language used on the web dashboard and interaction between the mobile application and the web server
3. Amazon Web Services: This is the cloud service provided by Amazon, where the data for the web dashboard was hosted on an EC2 instance running a Linux OS

### Other utilities

1. Visual Studio Code: A highly capable and feature-driven text editor
2. Git and GitHub: A file-sharing, collaboration and version control service widely used among the open source community to maintain the code base and back the code up regularly and securely
3. FileZilla: A secure and reliable FTP software to transfer files between the local system and the remote server and vice versa
4. PM2: A node.js library which acts an advanced node.js process manager, it's a powerful tool which helps with monitoring managing application processes, logging, and more

# New TIGERAWARE SYSTEM Implementation

This chapter focuses on the implementation of the entire TigerAware dashboard and detailed explanation of each and every module implemented in the application. A module level design of the TigerAware platform is as shown in Figure 3 . The following topics will discuss special features in the applications, key architectural considerations and its implementation and the overall flow of data from Firebase to the backend API and to the front-end dashboard view. The work done to improve the platform can be divided into improvement made on the server side of the application and the client side of the application.

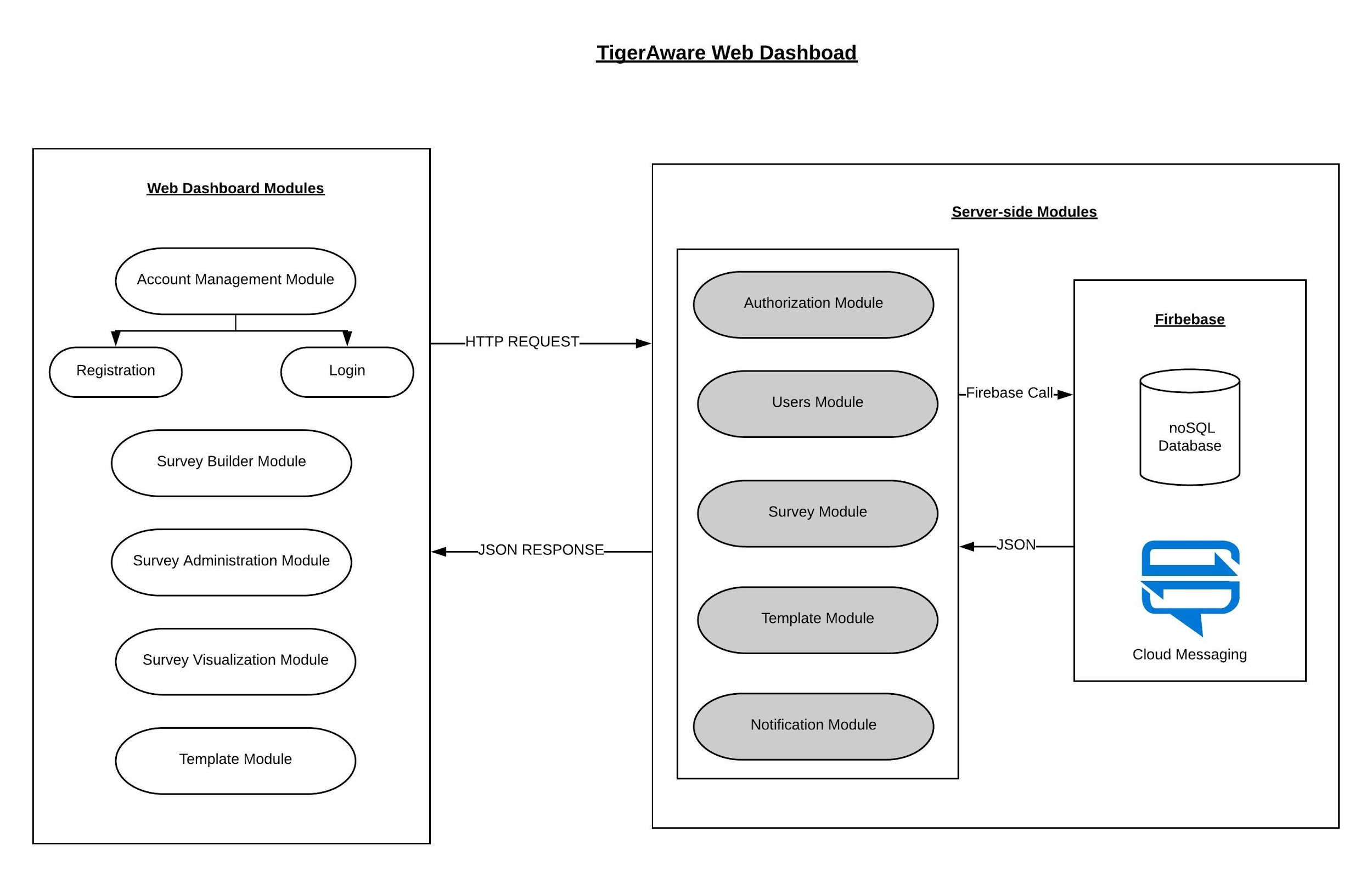


Figure 6 TigerAware Dashboard Implementation

## Server side Implementation

The server side of the application concentrates on the business logic involved in the application. It holds modules that provide various API’s necessary for the dashboard to deliver different functionalities. The server relies on the express.js library, a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. It also integrates with Firebase Admin SDK on the server side as it allows us to interact with Firebase from our server environments to provides admin privileges, allowing notification delivery, authorization token creation and various other content management privileges.

### Notification Scheduler

Research studies usually want to collect information at random from its participants in order to a get a general understanding of the participants daily life. Which would make it vital for the platform to notify users at random times to take a survey. This is achieved by the notification scheduler which is a background process running on the server constantly listening for changes in the database.

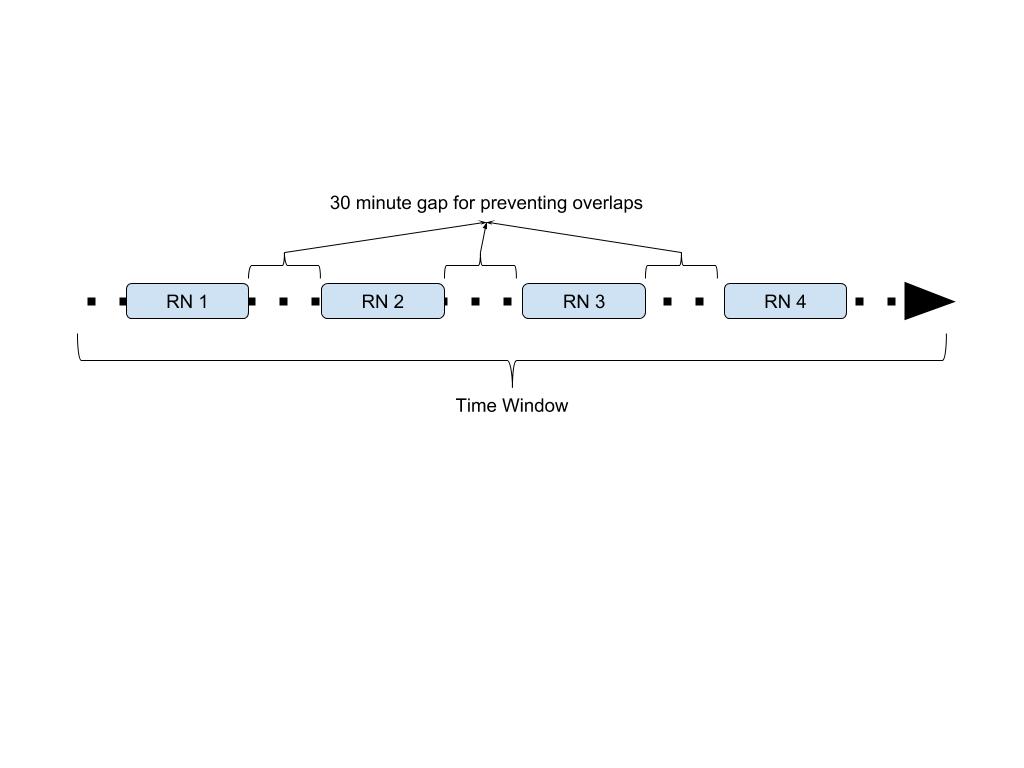


Figure 7 Illustration of Notification Scheduler

When the administrator adds a random notification to a survey, the notifications node is updated with information regarding to notification, every random notification creates has five important attributes. Every notification to be randomized provides a window in each the notifications need to delivered along with the number of notifications that need to be scheduled and expected end for the random notifications.

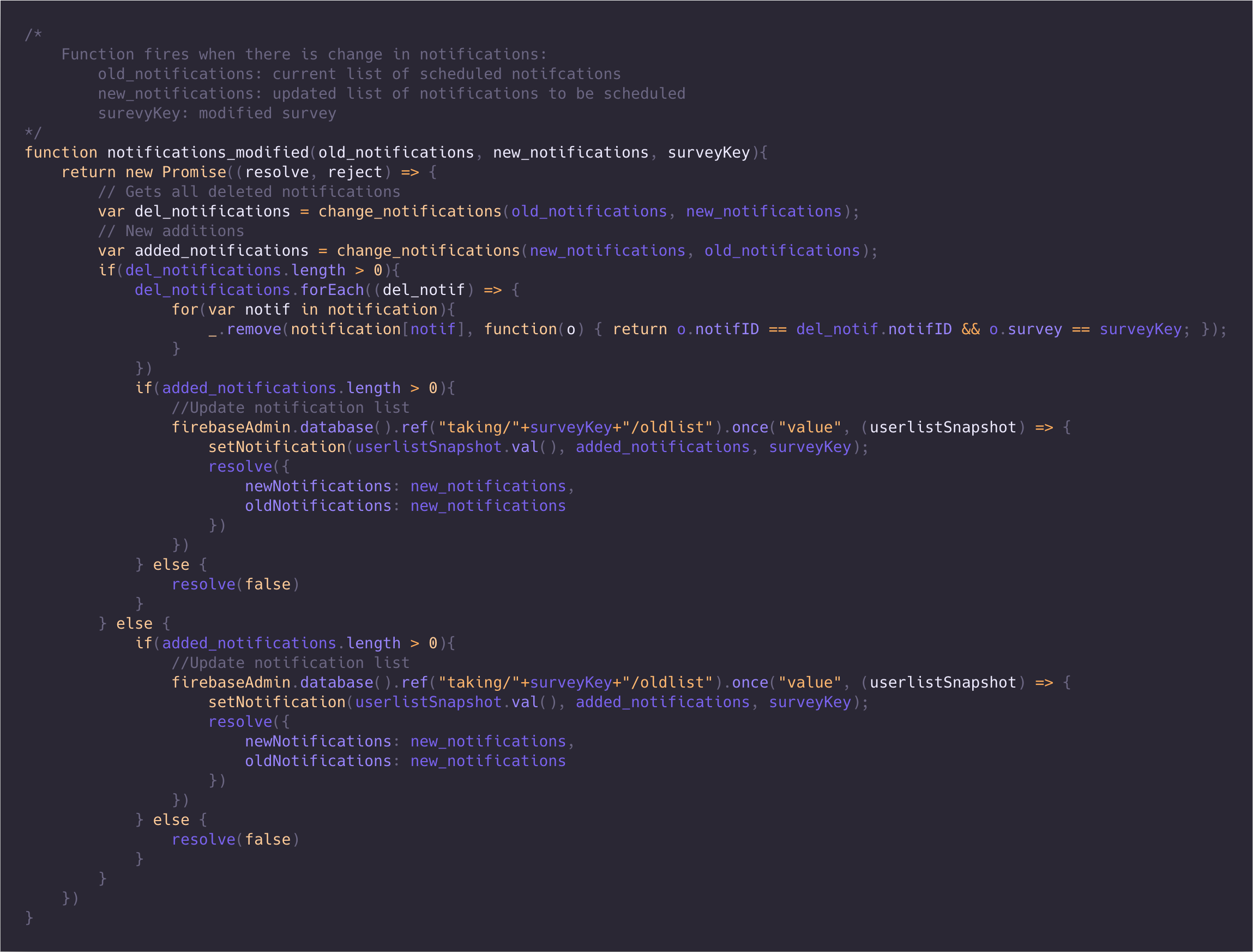


Figure 8 Background Listener for Change in Notifications

Based on constraints provided by the administrator, the notification scheduler creates new notifications every night until the last day of the random notification. Figure 4 best illustrates what notification scheduler is trying to achieve, for every participant in the survey a background process tries to generate N random times in each window. Also, when the server clock matches “00:01” in a time zone a rescheduling process is initiated to update all notifications for surveys in that time zone. The scheduler also is waiting on changes in notifications and taking nodes of the database to modify or create necessary notifications. Figure 5 shows once such background listener which reschedules notifications if any new notifications where added or modified.

### Authentication Module

Primarily to increase the number of clients through which the participants can respond to a survey. The platform should be able to provide a way to authenticate with third party devices without providing the login credentials. This can be achieved by using OAuth 2.0, an authorization framework which enables a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service, or by allowing the third-party application to obtain access on its own behalf.

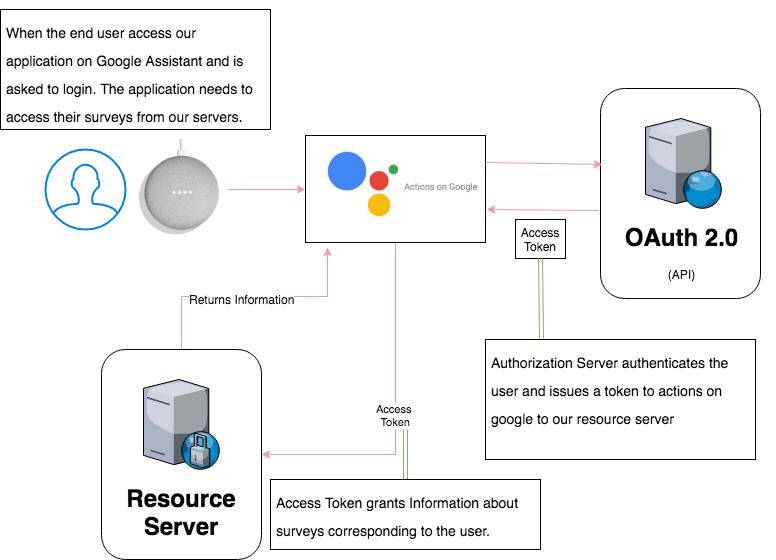


Figure 9 Illustration of OAuth 2.0 Flow

The Figure 6 illustrates the OAuth 2.0 flow for linking a Google Account with the users TigerAware account to collect responses for surveys and store them. When the user launches TigerAware Chabot for the first time by saying “OK Google! Talk to Tiger Aware” actions on Google provides a button which redirects the user to [*https://tigeraware.com/dialog/oauth?response\_type=code&client\_id=*](https://tigeraware.com/dialog/oauth?response_type=code&client_id=)*<CLIENT\_ID>&redirect\_uri=<REDIRECT\_URI> ,* after the server endpoint validates the *REDIRECT\_URI* and requesters CLIENT\_ID. The user is redirected to the login page. On successfully logging in the user is redirected to a dialog box which asks the user to explicitly allow a third party service to access their information on the TigerAware server. After clicking "Allow", the user will be redirected back to the application with a URL like *https://tigeraware.com/authorize?code=AUTHORIZATION\_CODE*. Which is followed by the token exchange flow, the client-server exchanges the authorization code with the POST *https://tigeraware.com/token?grant\_type=authorization\_code&code=AUTH\_CODE\_HERE&redirect\_uri=REDIRECT\_URI&client\_id=CLIENT\_ID&client\_secret=CLIENT\_SECRET*.

### Survey Module

The Survey module plays an important role in the entire platform. It provides endpoints to perform some of the core business logic necessary for the platform. It works in tandem with the front end of the application to provide vital functions like addition, deletion and modification of the survey blueprint. The survey blueprint holds all the information necessary for the client side applications to render a survey to the participant. The figure below shows the structure of a survey blueprint. It’s a combination of survey metadata and survey information to handle different use cases corresponding to the survey.

When the administrators uses the survey builder to add questions to their survey and clicks on the Save & Deploy/Save & Continue button on the survey builder a HTTP POST request is initiated at the */surveys/saveNewSurvey* API endpoint of the server. The request body consists of id used to authenticate the request and newSurvey which contains the JSON object which represents the blueprint. The server then calls the firebase.database() to create a random id called survey\_key to associate with the survey and store the information at */blueprints/<survey\_key>/* after successfully storing this information the */user/id/surveys* is also updated with the generated key as the person creating the survey is also assigned the administrator role. The table list all the endpoints currently available in the survey module.

Table 3 List of Survey Module API End Points and Their Reponses

|  |  |  |  |
| --- | --- | --- | --- |
| END POINT | HTTP METHOD | REQUEST PARAMETERS | RESPONSE |
| /users/saveNewSurvey | POST | *id*: “Authenticaiton token”  *newSurvey*: “JSON Object of blueprint” | Status Code: 200, TRUE: on successful addition  Status Code: 500, REQPARAMMISSING |
| /users/updateSurvey | POST | *id:* “Authentication Token”  *survey\_key:* “Survey Key”  *blueprint:* “JSON object of updated blueprint” | Status Code:200, true|false  Status Code: 500, REQPARAMMISSING |
| /users/deleteSurvey | POST | *id:* “Authentication Token”  *survey\_key:* “Survey Key” | Status Code:200, true|false  Status Code: 500, REQPARAMMISSING |

Table 2 lists out all the API’s supported in the survey module. It was created keeping in mind that the TigerAware end goal is acting as a robust platform as a service. As every call rests on JSON object to create/update/delete survey. Newer clients can be added to the platform to act as survey builders.

### Users Module

The User Module integrates with the User Management Module on client side to assign and delegate access to various users registered to use TigerAware. There are two major roles that are available survey administrator and survey participant. Survey administrators are able to view data, add participants, assign other administrators to their deployed survey. Survey Participant as the name suggest this role is specific to users participating in survey. The user can only take the survey on the mobile application and the access to survey is controlled by the survey administrator. The application is modular to handle any new roles that are added in the future. Each role has its own purpose, with some having limitations as well.

Table 4 List of User Module End Points and Their Responses

|  |  |  |  |
| --- | --- | --- | --- |
| END POINT | HTTP METHOD | REQUEST PARAMETERS | RESPONSE |
| /users/addUserOrAdmin | POST | *id*: “Authenticaiton token”  *survey\_key*: “Survey to assign the role for”  *user\_email*: “user email”  *isAdmin*: “TRUE: if admin/FALSE if participant” | Status Code: 200, TRUE: on successful addition  Status Code: 500, REQPARAMMISSING |
| /users/getAdminInStudy | POST | *id:* “Authentication Token”  *surveyKey:* “Survey Key” | Status Code:200, [user\_ids]  Status Code: 500, REQPARAMMISSING |
| /users/getUserInStudy | POST | *id:* “Authentication Token”  *surveyKey:* “Survey Key” | Status Code:200, [user\_ids]  Status Code: 500, REQPARAMMISSING |
| /users/getUserSurveys | POST | *id:* “Authentication Token” | Status Code:200,{surveys}  Status Code: 500, REQPARAMMISSING |
| /users/removeUserAndData | POST | *id:* “Authentication Token”  *survey\_key:* “Survey Key”  *user\_key:* “User key | Status Code:200,TRUE  Status Code: 500, REQPARAMMISSING |

When the administrator add users through the web dashboard, it creates a HTTP POST request to the */users/addUserOrAdmin* API endpoint of the server with an request body containing authorization token as ‘id’, ‘user\_email’, ‘survey\_key’, ‘isAdmin’ parameters. The survey\_key is added to either to */users/<id>/taking* node if the user is not an admin or */users/<id>/surveys* if the user is an admin. The API’s call responds with true/false if the survey\_key was successfully added to the user specific information. The Table 3 lists out all the API endpoints provided by the user module.

### Utility Scripts

To reduce the amount of time taken to create surveys, the platform needed to provide the ability to import preexisting surveys easily into a survey. PROMIS Measure and Phenx ToolKit are two widely used collection of surveys in health management and research studies to collect standardized information from a participants.

#### PROMIS Measures

PROMIS Measure are free to use but the calculation of outcome measures through their API has an yearly license fee associated with it. So, a utility script was created to use the API to collect all surveys. The API provides two endpoints, FORMS.xml which gives a list of all measures available in PROMIS API but they provide surveys in both Spanish and English languages as shown in Figure 7, the FORMS.xml was preprocessed to remove all forms which had a name attribute in Spanish language and another endpoint to access each of the forms based on the *OID* attribute for each *form* tag from the preprocessed FORMS.xml.

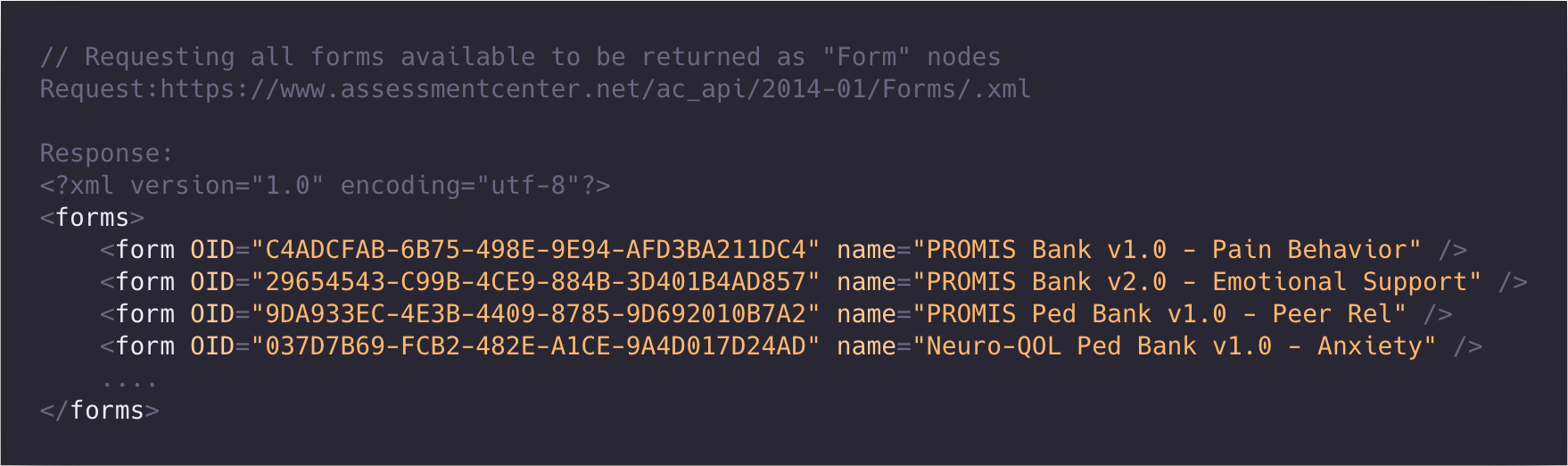


Figure 10 XML Response from PROMIS Measures API with List of Forms



Figure 11 XML Response with List of Questions in a Form

The Figure 8 shows how data is received from an API call, each of the XML forms are passed through a converter script which generates a new JSON object compatible with TigerAware survey structure.

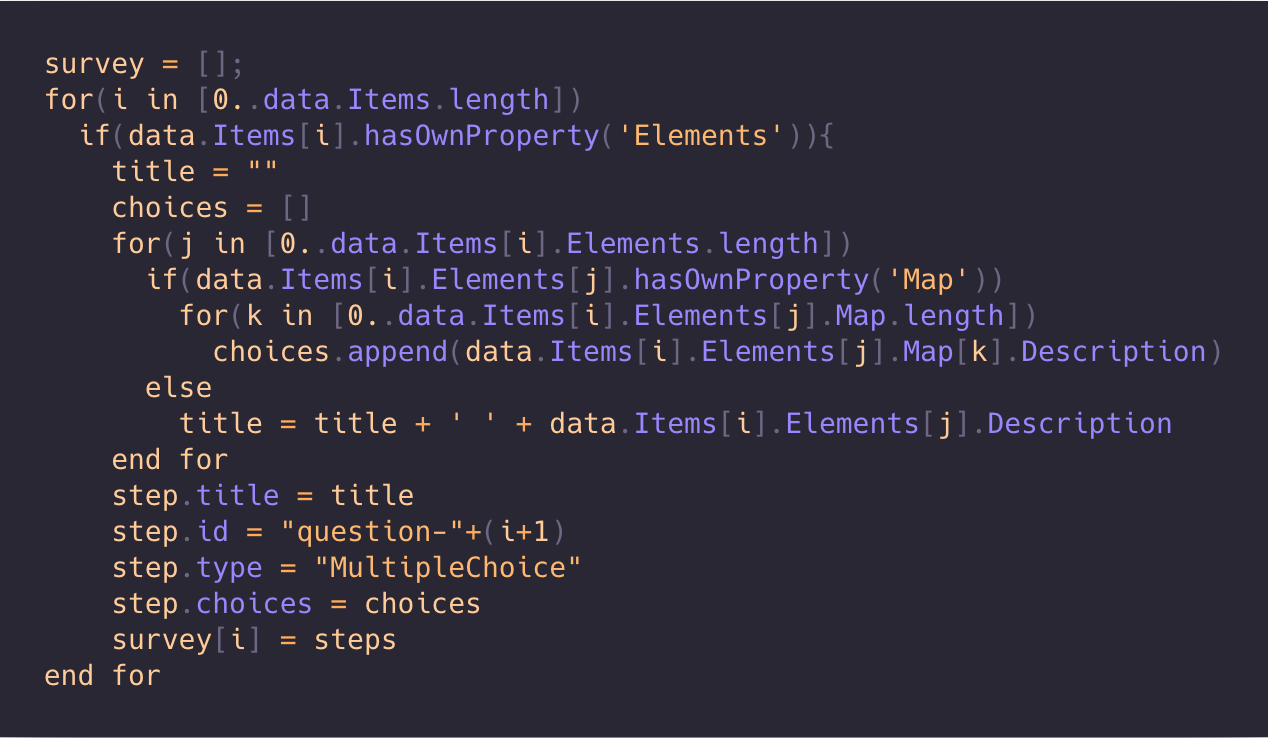


Figure 12 Pseudocode to Convert XML file to TigerAware Survey Grammar

The Figure 9 shows pseudocode for the converter, the XML file for each form has multiple *Item* which represents a question. Nested within each item are *Element* tags which can either be part of question title or response options for a question based on the association *Map* tag with *Element* tag. So the converter loops through each the *Item* tags to generate each step of survey which are added to our survey array. This array is later upload to our Firebase database under template node with a simple function call. Currently, a total of 300+ PROMIS Measure are available on platform accessible through *template dashboard* in TigerAware web dashboard

#### PhenX Toolkit

PhenX (consensus measures for Phenotypes and eXposures) is a community-driven effort to provide standard measures to help investigators identify opportunities for collaborative biomedical research which facilitates combining data from a variety of studies, and makes it easy for investigators to expand a study design beyond the primary research focus.

Table 5 Mapping of Headers in PhenX CSV file to TigerAware step-attributes

|  |  |
| --- | --- |
| **CSV HEADERS** | **TIGERAWARE STEP ATTRIBUTES** |
| Variable / Field Name | id |
| Form Name | Survey Name |
| Section Header | - |
| Field Type | type |
| Field Label | title |
| Choices, Calculations, OR Slider Labels | choices |
| Field Note | subtitle |
| Text Validation Type OR Show Slider Number | type |
| Text Validation Min | constraints |
| Text Validation Max | constraints |
| Identifier? | - |
| Branching Logic (Show field only if...) | conditions |
| Required Field? | - |
| Custom Alignment | - |
| Question Number (surveys only) | - |
| Matrix Group Name | - |
| Matrix Ranking? | - |

There are a total of 600+ PhenX measures available on the web portal in a CSV format with headers as show in Table 4, it describes which of the headers are useful in data parsing.

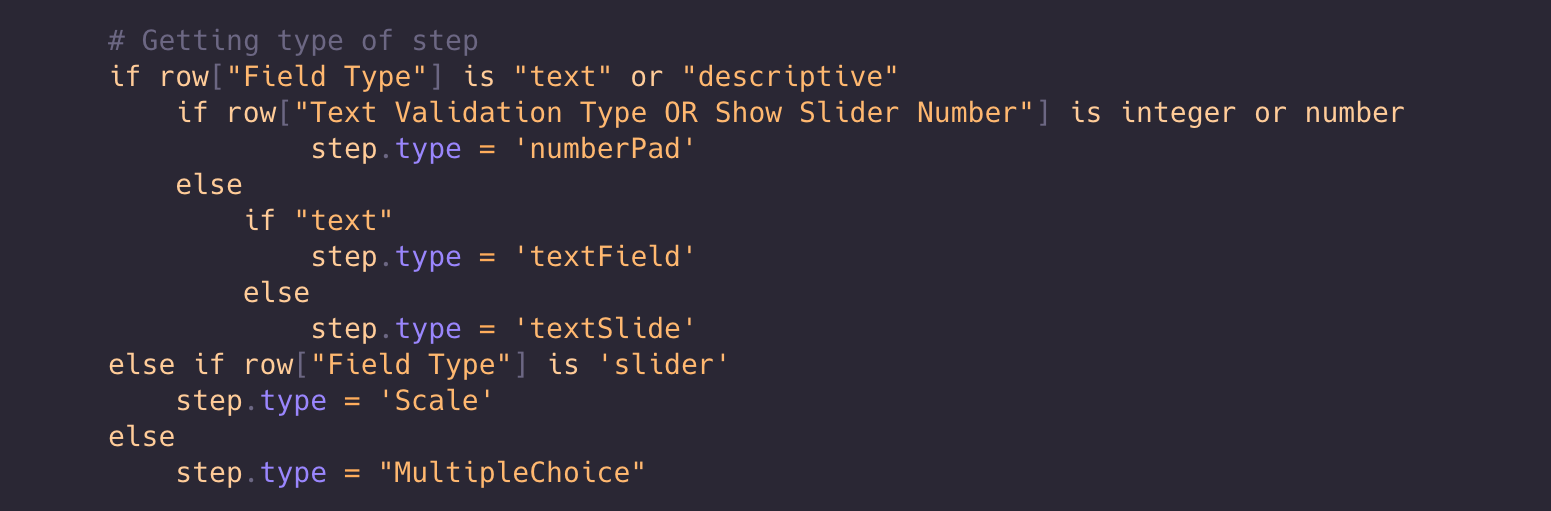


Figure 13 Pseudocode to Extract Step Type

There are three important steps in converting the CSV files, first being able to determine the type of question, this was pretty straight forward, there were two different fields that needed to be checked *“Field Type”* and “*Text Validation Type OR Show Slider Number*” based on the text in these field one of the four possible types where determined as show in Figure 10.

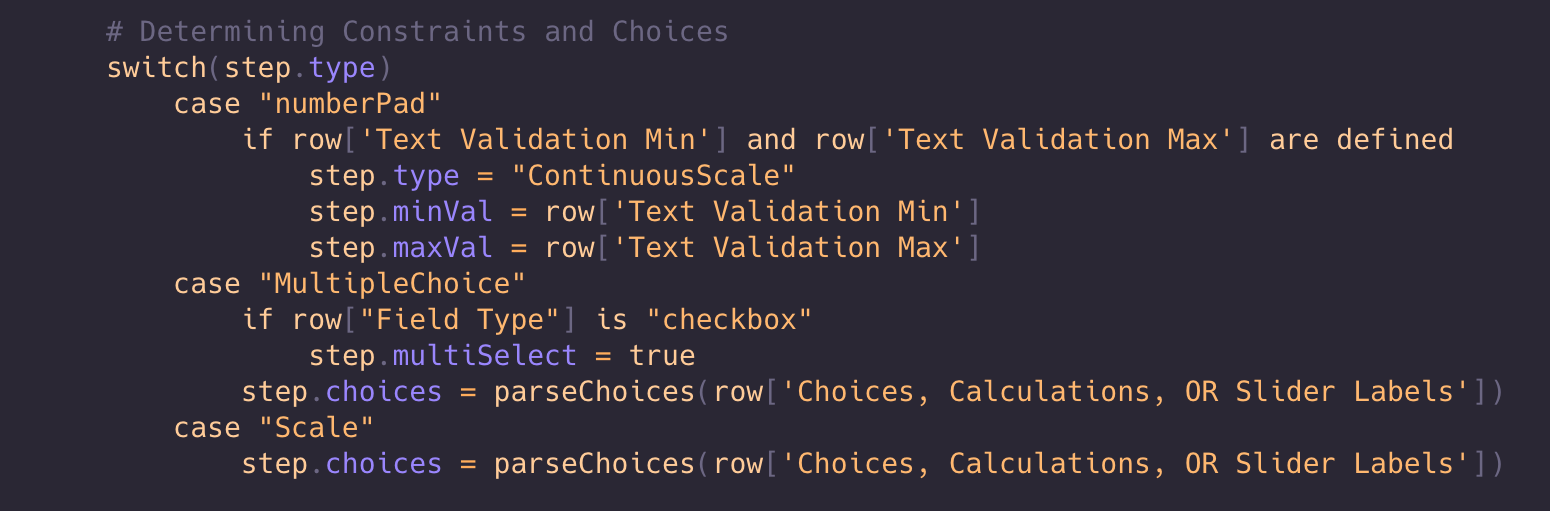


Figure 14 Pseudocode to Extract Step Constraints

Second is determining all the choices in question and constraints on the response of a question. Based on the type of question determined earlier, it was easier to parse necessary constraints involved in the question. “*Choices, Calculations, OR Slider Labels*” field is pipe delimited string of choices which was parsed to store all response choices in a question. “*Text Validation Min*” and “*Text Validation Max*” field was useful in selecting a better question type to gather information as show by pseudocode in Figure 11.

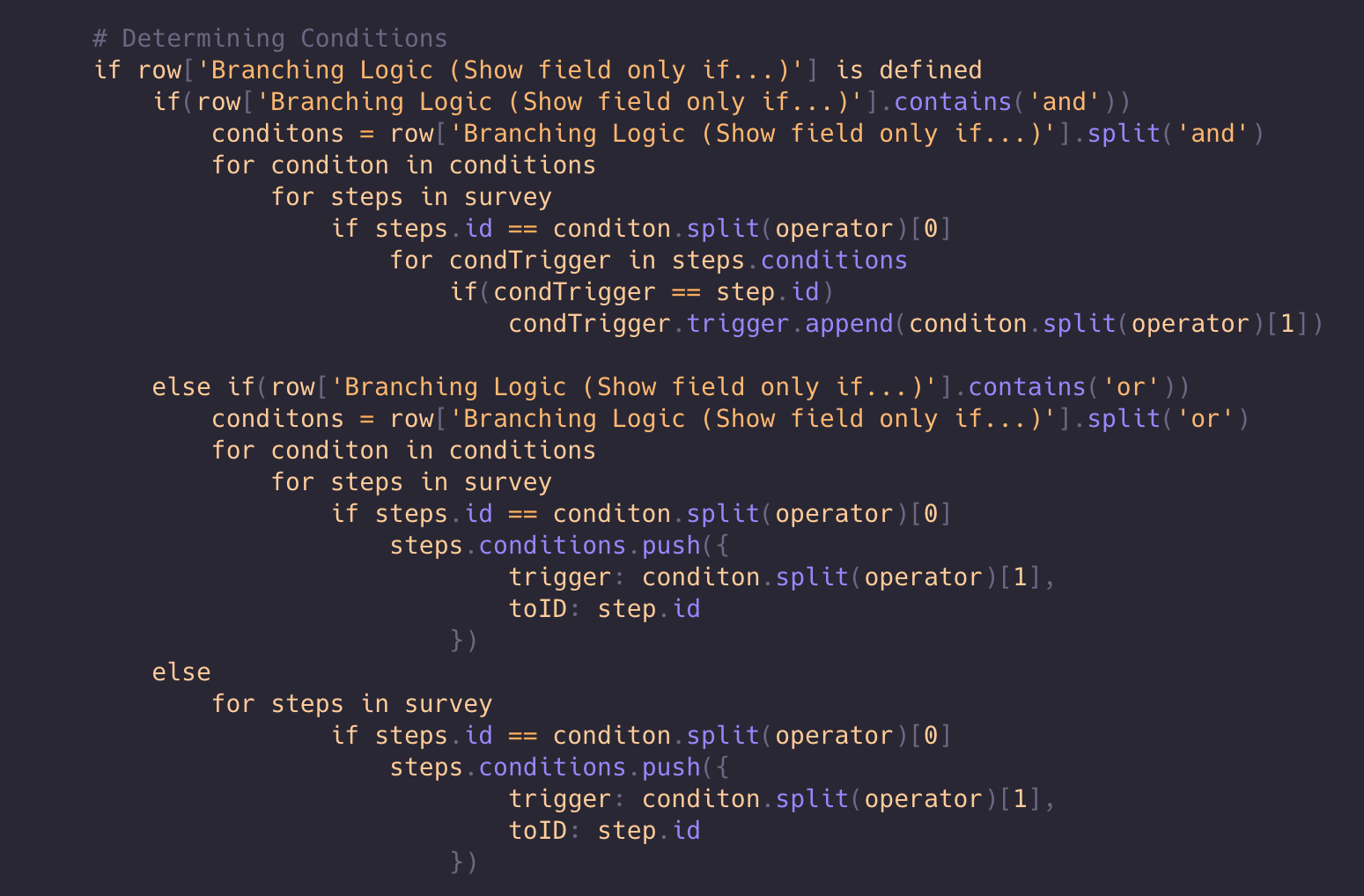


Figure 15 Pseudocode to Extract Condition on a Step

Lastly, it was vital to parse any information necessary for branching to and from a question in the survey. “*Branching Logic (Show field only if...)*” field in the CSV file was parsed for branching logic which is determined using the pseudocode in Figure 12.

## TigerAware Web Dashboard Modules

The TigerAware dashboard plays an important role in the entire platform as it allows the administrators to view, edit and create surveys. It also allows survey administrators to view responses gathered for a survey from all the clients in platform. The dashboard is developed using AngualrJS to follow MVC architecture in the front end of application development. The interface was styled using Materialize CSS3 library to improve usability and he innate responsiveness feature is used for providing multi-platform access to the dashboard like, desktop, mobile phones and tablets.. The application relies on APIs that are created using ExpressJS framework to establish a communication channel between the front-end and back-end of the platform.

### Account Management Module

Before being able to use the dashboard, users will need to register and login into the application. The first page that any user will come across in the website is the login page. For a first-time user, there is an option to sign up, where they are required to fill out some basic information (name, email address, username and password).

|  |  |
| --- | --- |
|  |  |

Figure 16 Login and Registration Interface

Security is paramount when creating accounts and creating a secure password prevents any unauthorized access to the application. Firebase as a platform made it easier to create an authentication mechanism for the application and also perform user management using it's console. The entire authentication process for registering new users, session management and password management usually implemented on other systems can be reduced few simple function calls.



Figure 17 Code for Registering New User

### Survey Builder Module

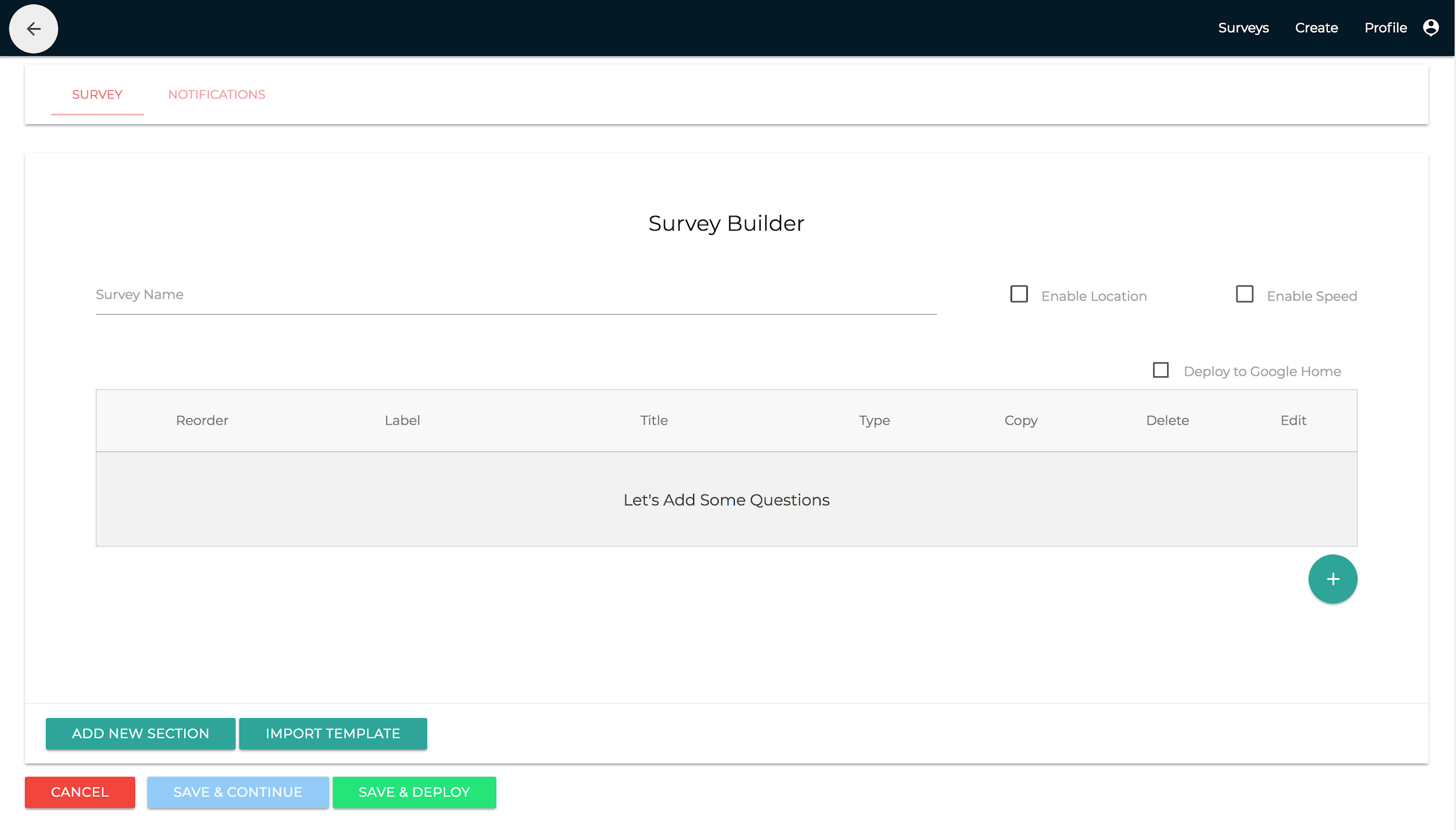


Figure 18 Survey Builder Interface

The Survey Builder is the most important part of the dashboard as it allows users to create complex surveys for the participants to undertake. The user can navigate to the Survey Builder page either by clicking on the Create Survey button on the overview page or by clicking the edit button of a pre-existing survey on the overview page. The user interface show in Figure 15 is broken down into two sections based on the functionality they offer.

#### Survey Builder

##### Google Assistant Surveys

The administrator at any point when creating the survey can request the application to deliver just to Google Assistant. This can be done with the help of a simple checkbox on the survey builder page. When the administrator toggles the deploy to google home checkbox, they’re notified that the survey builder functionality is limited, viz., they cannot create section or use any sort response based branching. Currently, the dashboard is limited to creating open ended questions for the Google Assistant without an option of response based branching, as the dashboard is restricted and dependent based on the client frameworks ability to deploy surveys. Once TigerAware will be able to completely leverage features available on Google Assistant, all the features can be enabled on the dashboard by simply removing the *ng-if=”vm.isAssistantSurvey”* attribute in the builder.html page.

##### Smartphone Surveys

The survey builder is full featured to support development of surveys for smartphone based surveys. The builder page is broken down into two different parts: Survey Builder and the Notification builder by using a tabbed layout. In the Survey Builder tab, the administrator can add a question by clicking the Green ‘+’ Floating Action Button(FAB), the reason for using FAB for adding a question instead of going with generic button was to help administrators identity and provide a way to easily perform the most repetitive task the administrator would perform on the survey builder.

The ‘+’ FAB would launch question-modal show in Figure 16 which would act as the central place for all the information pertaining to a question in the survey. When the user clicks the save button, a helper function stored provided by SchemaHelperService service is called which checks to see if the question has a unique id, if the SchemaHelperService returns true the question is updated or saved and the corresponding vm.steps variable in the builder.controller.js file is updated. The smartphone based surveys have access to a full array of question types, currently the dashboard allows for 12 different questions types, out which currently 1 is a sensor based integrations and other background metadata that can be added to survey collection.

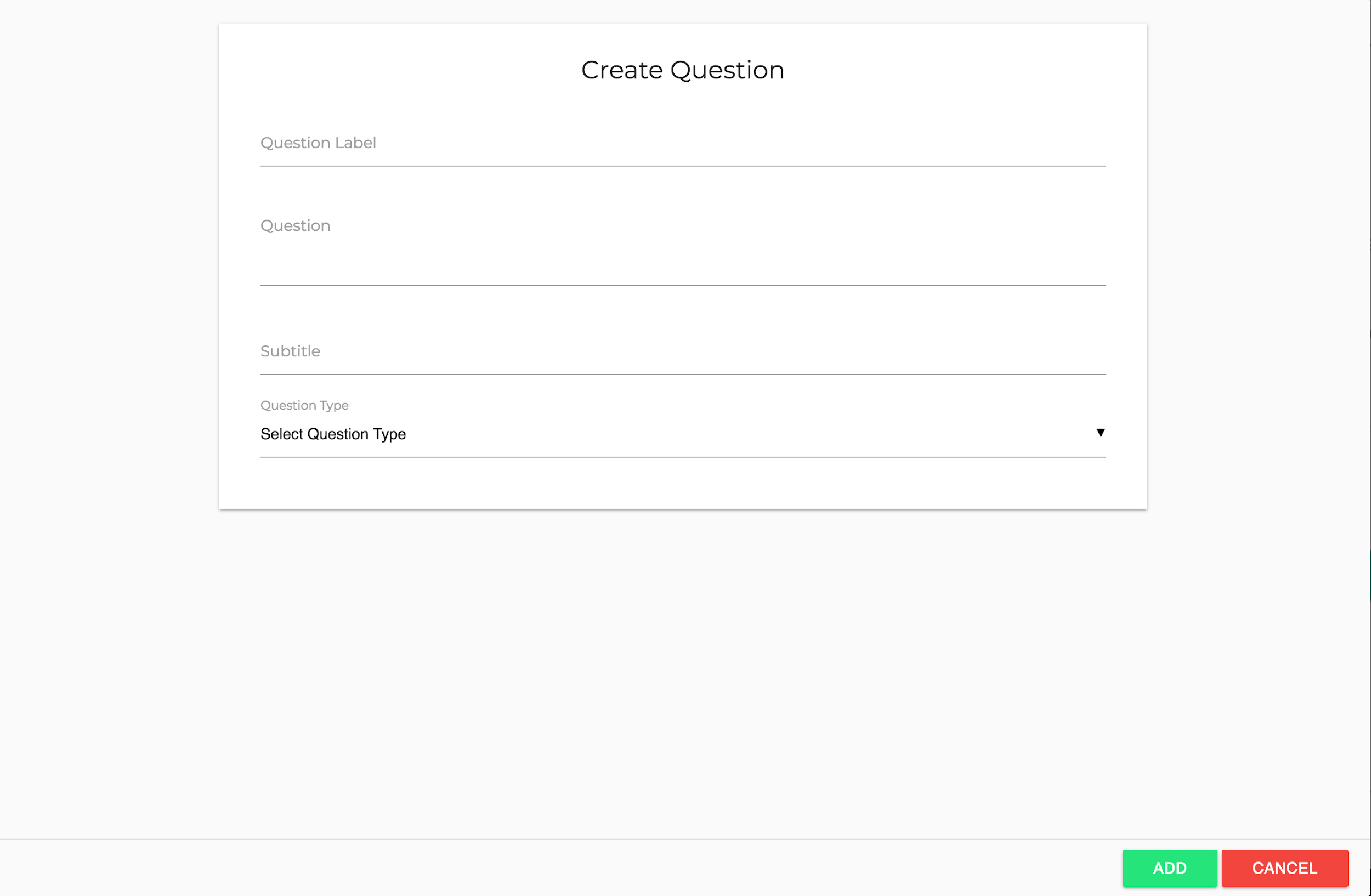


Figure 19 Create Question Modal

Currently the dashboards supports branching on four different question types. The administrator can create a section by clicking the “Add Section” button at the bottom of the main survey UI. This creates a section which holds the follow up questions, the *vm.sections* is an object with a new section id added to it every time ‘add new section’ button is clicked. Similar to the main survey section any new questions added to the survey is checked to maintain unique question id.

*SurveySerializationHelper* Angular Service, the *serialize* function available in the service converts all section objects created in survey builder page to a directed graph where each of node represents either a question or a section of questions. Based on survey grammar mentioned in *3.1*, the function uses the *conditionDefault* *step-attribute* is updated either with question id of the next following question or *^eos* a special character used to indicate the end of survey. The convention helps TigerAware smartphone based clients to render the necessary question in the required order. Similarly, when the administrator wishes to edit a created survey *deserialize* function flattens graph to recreate all the objects necessary to render the survey builder page.

#### Notification Builder

When conducting a survey based research the administrator would require all their participants to maintain compliance necessary to successfully collect information to gain insight. Notification builder allows administrators to set up survey level notifications to remind the participants of a pending survey.

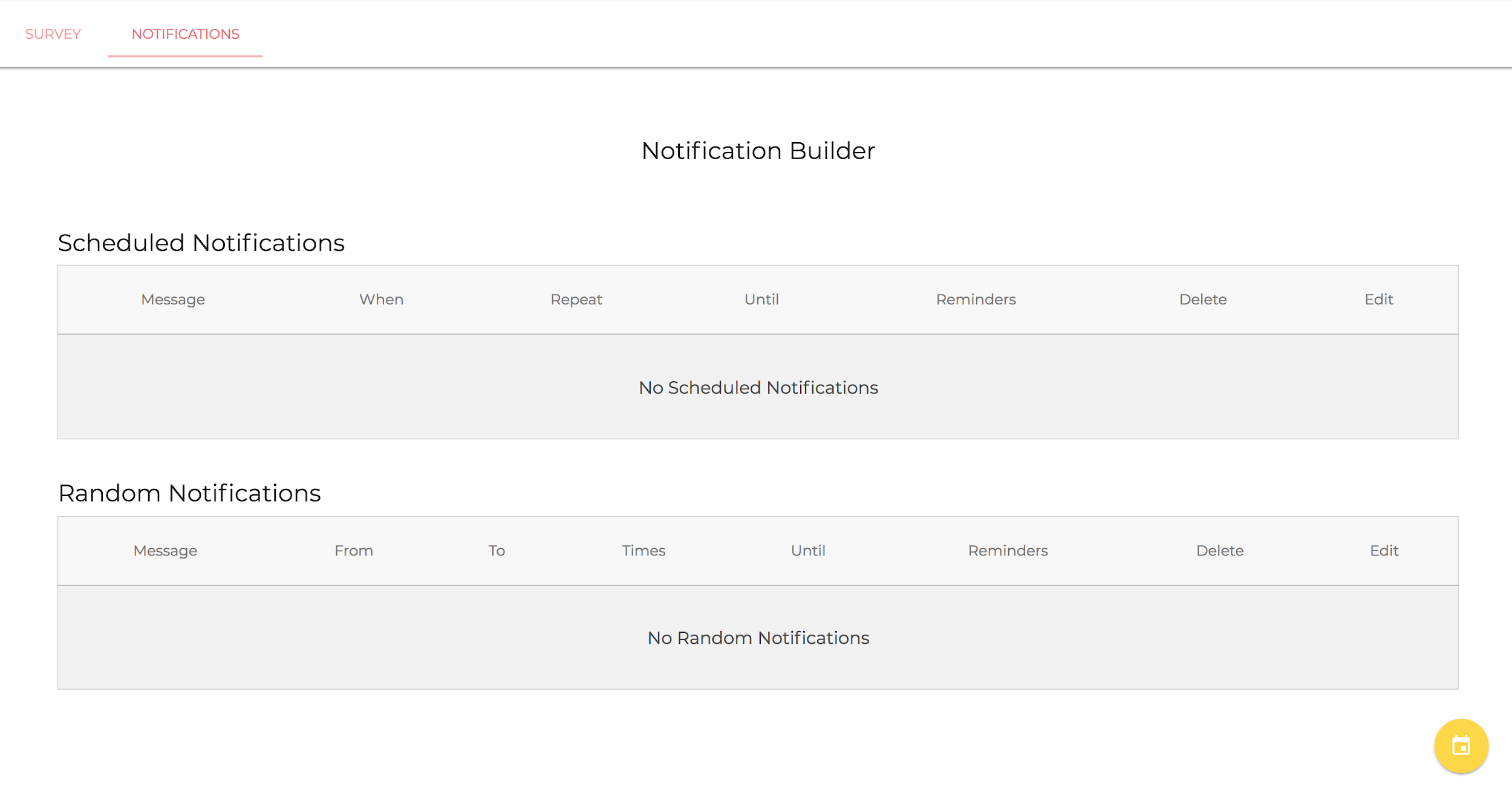


Figure 20 Notification Builder Interface

The FAB button in the notification tab show in Figure 17 launches the notification modal Figure 18 which provides a wide range of constraints administrator can set on the notification. There a functionally two different ways an administrator can notify all the participants: administrators can setup scheduled notifications, where all the users are notified at a specific time each day to collect information; they can also set up random notifications where each user taking the survey is randomly notified each day. The administrators can also select a time window and number of random notifications in that time window and the last day to notify the users randomly.

|  |  |
| --- | --- |
|  |  |

Figure 21 Add New Notification Modals

### Survey Administration Module

The survey administration module allows administrators to perform administrative operations corresponding to each of their surveys. An administrator can add users with one of the predefined roles viz., administrator or participant of the survey. This module also has an interface to show all the users currently in the database. Administrators are the only users who can view, add and edit a participants to their survey.

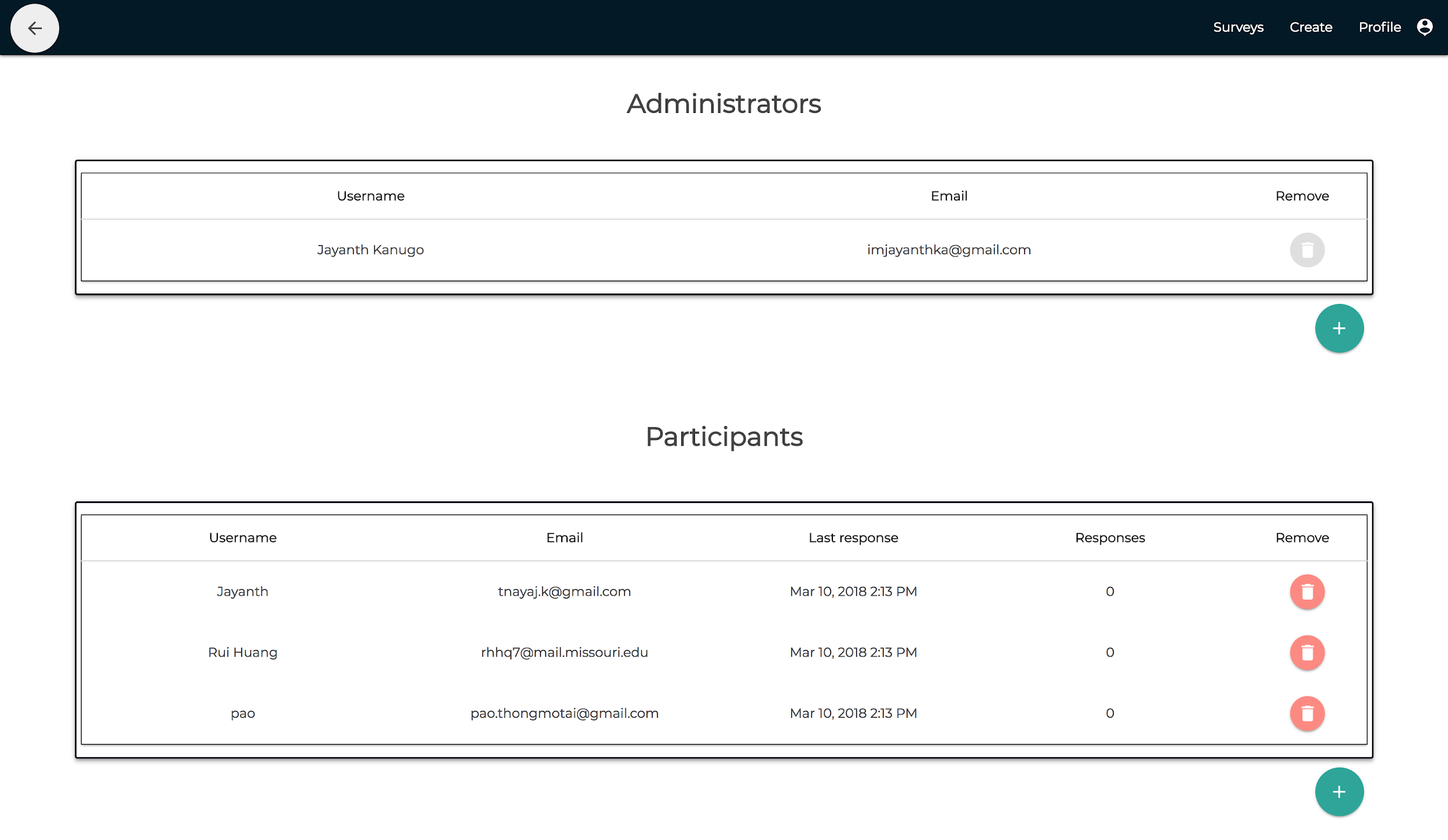


Figure 22 Survey Administration Interface

The module also provides the ability to the administrator to control the access associated to each of their surveys. They can disable any of their currently deployed surveys with the flip of a switch shown in Figure 20.

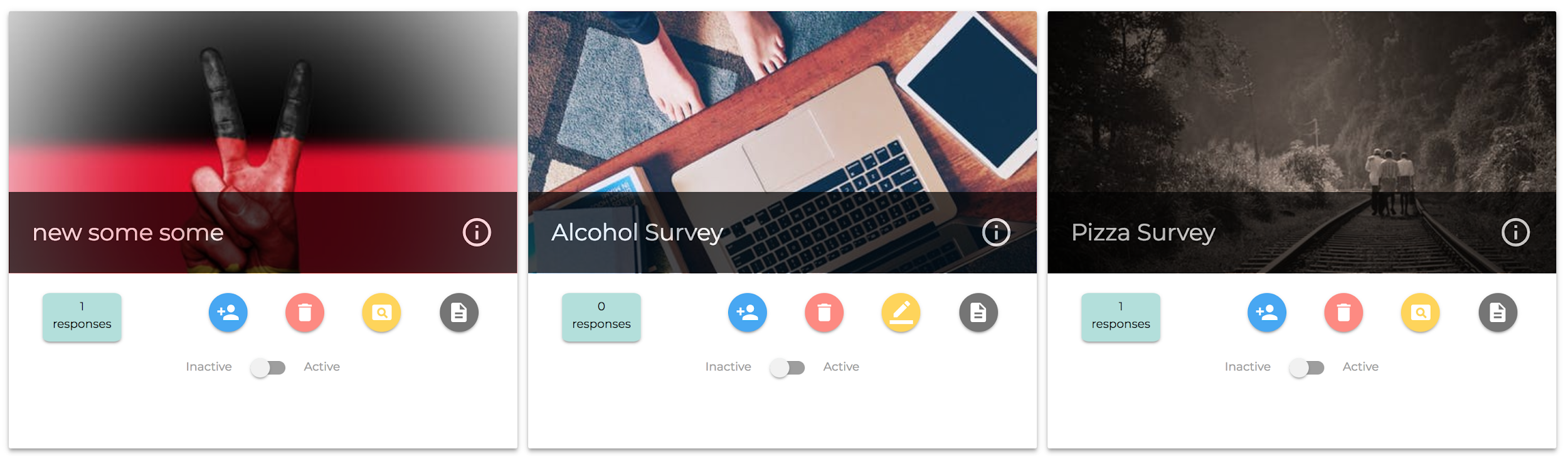


Figure 23 Survey Overview Interface with Ability to Activate or Deactivate a Survey

### Survey Visualization Module

The dashboard also provides a way to visualize all responses collected from the study group the survey is assigned to. The survey visualization module works in real time to display information being collected, it improves the workflow of each research study as it eliminates the necessity for administrators to export the data collected and use a third party software to visualize and gather insights from data.

|  |  |
| --- | --- |
|  |  |
|  | |

Figure 24 Different Visualization Available in Survey Visualization Module rendered based on Step Type

The *surveyDisplayController* is the controller for Survey Display page and it works in tandem with *dynamicGraphService:* an Angular Service created to generate all graphs necessary to visualize the gathered information based on type of each question in survey as show in Figure 21. *dynamicGraphService* service also provides information of compliance by visualizing compliance information which helps researcher to track participant’s compliance to a study.

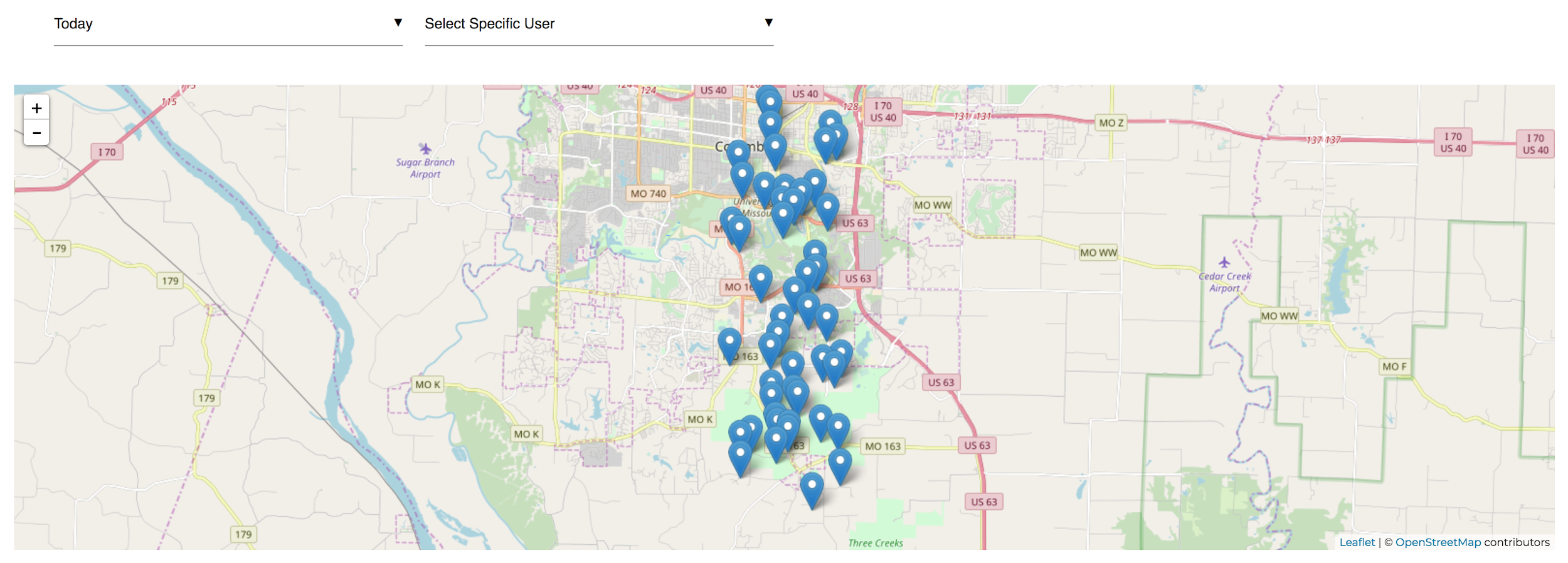


Figure 25 Visualization for Location Metadata representing Where Survey was Taken

It also provides methods to visualize metadata collected for a survey, the location of each user-response is mapped on a graph to represent all positions and speed associated to the time of taking the survey as shown in Figure 22.

Another nifty utility that is provided by the survey display page is to allow administrators to download all the responses to their local system from TigerAware platform into a CSV file. This, offers administrators flexibility to use third party data analytics tools on the collected data.

### Template Module

The TigerAware dashboard allows survey administrators to import from a wide range of pre-existing surveys to reduce the amount of time taken to create a survey from scratch. When on the overview page, the administrator can click on import template button. The user is then navigated to Template Dashboard page rendered as show in Figure 23. This page is populated based on a service call to getTemplateList() service function which obtains all templates stored in the database and displays them as a paginated list of surveys names.

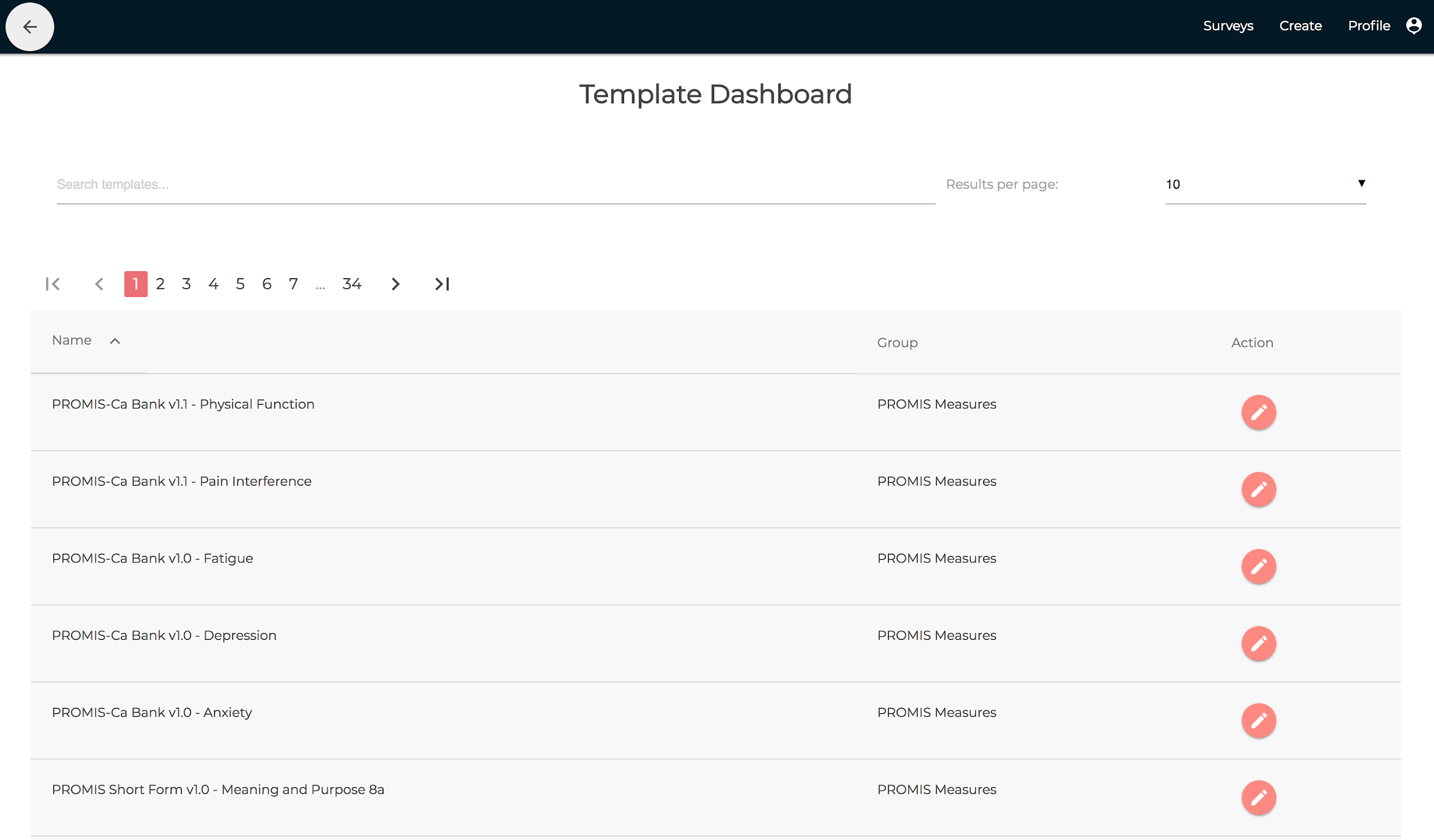


Figure 26 Template Dashboard Interface

The administrator can select one of the many available surveys, on clicking Action button it retrieves all the steps by making a HTTP post request to the backend and the administrator is redirected to the survey builder page with steps pre-populated according to the template selected.

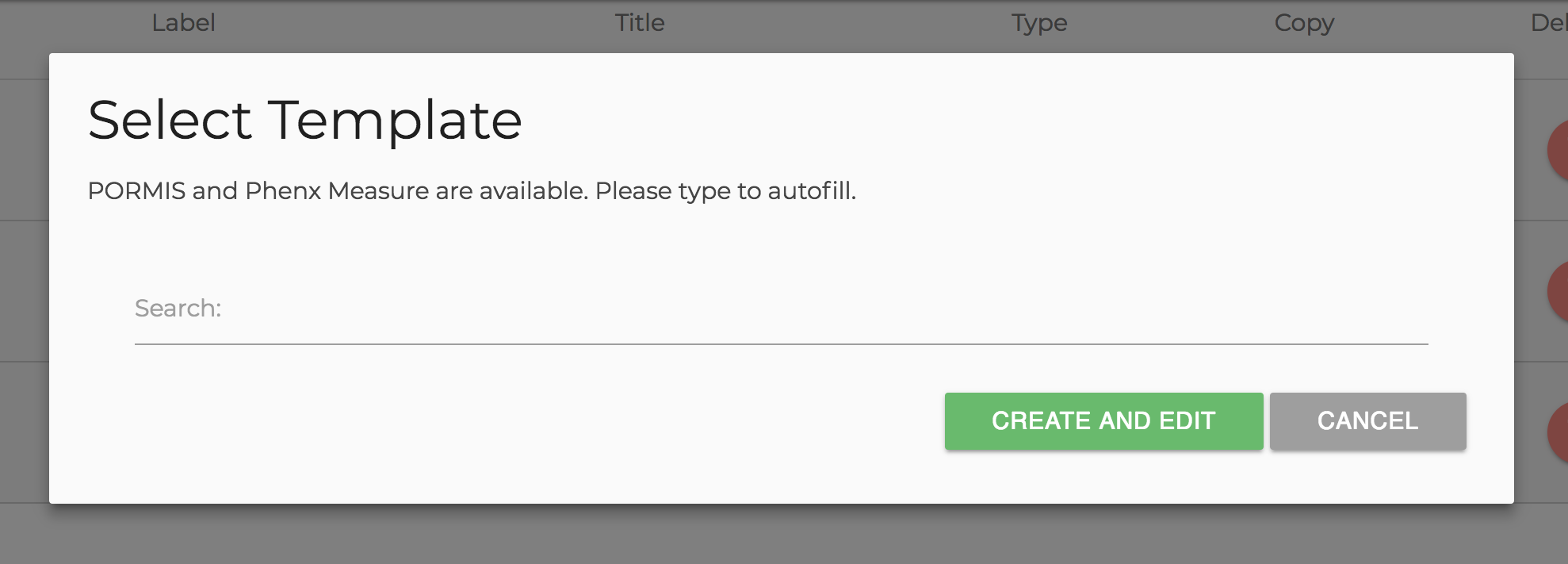


Figure 27 Autofill Enabled Modal to Search and Add Templates to a Survey

Once in survey builder, an administrator can import any number of templates as part of a section by clicking on ‘import template’ at the bottom of each section as show in Figure 15. This launches a template search model as show in Figure 24 that would allow the administrators to name search from various templates on the platform. Once imported all the question can be rearranged between the main and various other sections allowing user to create a complex set of question to form a survey.

# TigerAware Applications

Before, we discuss the real-world applications of TigerAware. Table 6 describes how the current platform can provide more robust features compared to the Existing Dashboard and other industry platforms.

Table 6 New TigerAware Dashboard Features Compared to Exisiting Systems

| Survey Platfroms / Functionalities | REDCap | DEMONS | AndWellness | Survey Monkey | Existing TigerAware Dashboard | New TigerAware Dashboard |
| --- | --- | --- | --- | --- | --- | --- |
| Quiz | Yes | Yes | No | Yes | No | Yes |
| Survey | Yes | Yes | Yes | Yes | Yes | Yes |
| Response based Branching | Yes | No | Yes | Yes | No | Yes |
| Notification Scheduler | No | No | No | Email Based Notifications | No | Yes |
| Survey Visualization | No | Yes | Yes | Yes | Limited | Yes |
| External Data Sources | No | Mobile | Mobile | Mobile | Mobile | Mobile and third party virtual assistants |
| Survey Templates | Yes | No | No | No | No | Yes |

The platform has been used in a number of real-world studies and the application has shown excellent capabilities in adaptability and deployment for new types of data collection tasks. The first part of the section discusses about the feedback from an initial focus group and a pilot study. followed by several ongoing use-cases of TigerAware.

## Privacy and Anonymity

One of the most common concerns for participants of mobile based studies(particularly clinical and behavioral health studies) is privacy and anonymity [20, 21]. Thus, steps were taken while developing TigerAware to preserve the anonymity and privacy of participants during and after the study. When participants create a profile to be invited to a study, they can enter a coded email given to them by their research group. This will conform to IRB policies by separating participant data and identity. We also restrict the availability of surveys to people who haven’t been assigned to them. Several rules are added to prevent researchers from knowing about the participants currently in the system, the most important one being that the survey created must know the participant’s complete email ID to add them(they can’t search via autocomplete, etc.) In an effort to prevent researchers from learning about participants enrolled in different studies, we added the restriction that researchers must also know participants' study email to add them (they can’t search via auto-complete, etc.). Apart from these measures, it is also ensured that information is encrypted before being transmitted and that participants and researchers are authenticated.

## Focus Group

During the development of TigerAware, we routinely consulted with psychological and social science professionals who intend to use the application to conduct research studies. This consultation helped us gain an understanding of which components of the platform work well, and which has a scope of improvement. After development of the beta version of the platform, the application was handed over to our focus group to use for two weeks. They subsequently gave feedback on five aspects of the application.

1. **Usability:** Many of the researchers appreciated that the platform is intuitive, simple, and easy to use. We noted the most problems during the survey creation process was due to the large number of options and branching creation process. To improve this, tooltips and icons were incorporated to supplement navigation. We also plan to develop instructional materials to help new researchers build and deploy new studies.
2. **Robustness:** One key concern was the robustness of the application. Given that recruiting participants and running a study is expensive, the researchers emphasized data persistence and backup at a mobile and database level. Though there was no data loss, systems will be capable to create local copies of the data in case some of the data is lost. This also applies to responses that were received when the user was offline
3. **Extensibility:** The focus group also observed that TigerAware easily extended to other study types. This is a key focus of the application, and effort will be put into making this process easier. Researchers also requested that previously build studies be copy able and shareable to save work when creating and sharing similar research. This feature was later added to the application
4. **Privacy and Anonymity:** As stated in section <4.1>, the personal anonymity and privacy of participants was a critical request
5. **Engagement:** One request of the group was to incorporate more ways of encouraging and tracking engagement of study participants. To encourage participants to use the platform, we developed a scheduled and random notification module that will allow researchers to request participants to complete studies. To track engagement, researchers also now have access to daily participation activity including missed, terminated, and completed surveys

## Google Assistant Based Diabetes Self-Management Study

In collaboration with the Department of Health Management and Informatics at the University of Missouri, the TigerAware system was adopted for data collection and diabetes self-management [22]. TigerAware is used to convert a set of questions into a Google Assistant based application which complements a preexisting mobile application. A chat function is developed to collect diabetes-related information, such as the users' blood glucose readings from their Blood Glucose Monitor, daily meals and activities, and overall well-being of the patient. The application advices the users based on their response for activity level and diet based on their responses. The mobile application and dashboard can be used by patients and physicians to monitor the information relating to patient’s wellbeing

## Bilingualism and Aphasia Study

Though many applications exist for language learning, few allow researchers to gain an accurate understanding of language progression over time [18]. In collaboration with the University of Missouri Bilingualism and Aphasia Lab, TigerAware will be used to monitor and improve the learning of new languages. The department plans to start the study using this application and test it for a span of few weeks. For this study, an image display step type and response time metadata field will be added to the framework in order to track the accuracy and speed of new word recall in Spanish and Mandarin Chinese. Plans to make changes in dashboard interpreters to enable detailed tracking of progress over time and feedback to users are being considered. This project will allow for further refinement of the platform and help streamline the process of adding new question types and response fields.

## Driving After Drinking Alcohol Study

A large body of research has examined which individuals are at risk for driving after drinking alcohol (DAD) for example, highly impulsive individuals or frequent risk takers in other domains are more likely to engage in DAD. However, almost no research has been conducted examining event-level predictors of DAD risk---what factors of the individual (e.g., current mood), drinking context (e.g., bar vs. restaurant, distance from home), or social context (drinking with one or two friends vs. party), make a DAD event more likely. Advances in mobile technology allow for the use of ambulatory assessment (AA) of such event-level characteristics to study DAD decisions in actual drinking contexts.

In collaboration with the University of Missouri Psychological Sciences Department and funded by an NIH grant, this project will provide some of the first data on how participants make decisions about DAD. Data will be collected from participant’s report (using TigerAware generated survey questions), mobile breathalyzer devices (recently integrated as a question type into TigerAware) and GPS/location information taken from a study provided smartphone. We will enroll 200 moderate-to-heavy drinking young adults and monitor them over a period of six weeks. This study will involve three separate surveys for data collection:

1. **Morning Report**: Completed by all subjects each day, 30 minutes after waking. This will consist of 12 questions consisting of yes/no, multiple choice, numeric and free response types.
2. **Drinking Report**: Completed by all subjects 4 times each day (at 3pm, 7pm, 10pm, and midnight; subjects may initiate report at other times if drinking alcohol). This survey will consist of three yes/no and multiple-choice questions, followed by a series of scale questions.
3. **Bedtime**: going to bed and setting wake-up reminder for next day.

## Implicit Racial Bias Study

This project intends to strengthen causal inferences that racial discrimination contributes to mental health outcomes among African-Americans by characterizing the temporal relationship between racial discrimination and psychological distress. An Ecological Momentary Assessment (EMA) approach will be used where African-American participants use TigerAware to report experiences of racial discrimination and momentary negative effects, anxiety, and depression symptoms (i.e.,psychological distress) over a four-week period. A targeted sample of 60 African American undergraduate and graduate students who are 18 or older will be recruited. During the EMA period, participants will complete two types of responses: Event-based reports and random prompts. Participants will initiate an event-based report when they experience a race-related incident, defined as determined from the previously conducted focus study. During each event-based report, participants will first report momentary psychological distress (affects and anxiety/depression symptoms), followed by questions specific to the race-related incident. Then, the participants will again be prompted via notification in the TigerAware app to report psychological distress thirty and sixty minutes following the race-related incident. Independently, participants will receive random prompts throughout the day between 9am and 10pm to assess fluctuations in affects and anxiety/depression using the same items as those in each of the event-based report.

# Conclusion and Future Work

Developing, TigerAware to act as a Platform as a Service(PaaS) has involved using some of the newest and popular web development technologies [23] [24], understanding how surveys are structured, and making design considerations on how survey data needs to be stored and preprocessed to reduce query complexities. All of this has been an enriching process in terms of knowledge gained about developing a modular system for survey collection and integrating various clients for survey deployment.

This application will greatly reduce the time taken for research studies to deploy their surveys and focus over gathering data and gain valuable insight through data analytics.

## Future Work

There are a number of important enhancements that the application can use and will be considered for implementation in near future. The platform can be made even more intends to develop a powerful analysis interpretive node that would be server based and maintain the interpretive node architecture. The server will use Firebase's API for its database interface. The flexible and high performance serving system called TensorFlow Serving will be examined for the new control part. Additionally, a series of deep learning servers will be developed separately using TensorFlow for the interpretation part. This architecture enables an easy integration of machine learning and deep learning methods for new studies. Similar to iOS platforms support for a range of specific question types, the analysis server will support a range of related data analysis types.

Furthermore, sobriety tests associated with the blood alcohol content question type will be developed. A new question type that administers an active sobriety test would be implemented. The sobriety test would return a value indicating a participant’s level of sobriety. The test value could be calculated by a pre-defined algorithm or using machine learning. If the values are calculated by machine learning, the blood alcohol content from the existing breathalyzer question type would be used to train the neural network. Essentially, TigerAware would collect its own training data for interesting machine learning tasks. Similarly, a sobriety test for cannabis usage will be created and deep learning will be used to recognize biological indicators of cannabis intoxication. Sobriety tests are important because if the tests become recognized by law, TigerAware will enable their instant deployment to all police officers' mobile devices. TigerAware allows for fascinating expansions of its core technology and enables new and exciting research opportunities in all fields.

# References

|  |  |
| --- | --- |
| [1] | Pew Research, "Mobile Fact Sheet," Jan 2018. [Online]. Available: http://www.pewinternet.org/fact-sheet/mobile/. [Accessed 2018 February 2018]. |
| [2] | M. Guay, "The 20 Best Online Survey Builder Tools," Zapier, 24 June 2016. [Online]. Available: https://zapier.com/learn/forms-surveys/best-survey-apps/. [Accessed 9 October 2017]. |
| [3] | C. Cohn, "Build vs. Buy: How to Know When You Should Build Custom Software Over Canned Solutions," Forbes, Inc., 15 September 2014. [Online]. Available: https://www.forbes.com/sites/chuckcohn/2014/09/15/build-vs-buy-how-to-know-when-you-should-build-custom-software-over-canned-solutions/#2416028cc371. [Accessed 9 October 2017]. |
| [4] | SurveyMonkey, "SurveyMonkey Plans and Pricing," SurveyMonkey, 2017. [Online]. Available: https://www.surveymonkey.com/pricing/details/. [Accessed 9 October 2017]. |
| [5] | K. E. Heron and J. M. Smyth, "Ecological momentary interventions: incorporating mobile technology into psychosocial and health behaviour treatments," *British journal of health psychology,* vol. 15, no. 1, pp. 1-39, 2010. |
| [6] | S. Shiffman, A. A. Stone and M. R. Hufford, "Ecological momentary assessment," *Annu. Rev. Clin. Psychol.,* vol. IV, pp. 1-32, 2008. |
| [7] | T. J. Trull and U. W. Ebner-Priemer, "Using experience sampling methods/ecological momentary assessment (ESM/EMA) in clinical assessment and clinical research: introduction to the special section.," American Psychological Association, 2009. |
| [8] | C. Free, G. Phillips, L. Watson, L. Galli, L. Felix, P. Edwards, V. Patel and A. Haines, "The effectiveness of mobile-health technologies to improve health care service delivery processes: a systematic review and meta-analysis," *PLoS medicine,* vol. X, no. 1, p. e1001363, 2013. |
| [9] | L. P. Malasinghe, N. Ramzan and K. Dahal, "Remote patient monitoring: a comprehensive study," *Journal of Ambient Intelligence and Humanized Computing,* pp. 1-20, 2017. |
| [10] | A. Gaggioli, G. Pioggia, G. Tartarisco, G. Baldus, D. Corda, P. Cipresso and G. Riva, "A mobile data collection platform for mental health research," *Personal and Ubiquitous Computing,* vol. XVII, no. 2, pp. 241-251, 2013. |
| [11] | P. A. Harris, R. Taylor, R. Thielke, J. Payne, N. Gonzalez and J. G. Conde, "Research electronic data capture (REDCap)—a metadata-driven methodology and workflow process for providing translational research informatics support," *Journal of biomedical informatics,* vol. XLII, no. 2, pp. 377-381, 2009. |
| [12] | J. Hicks, N. Ramanathan, D. Kim, M. Monibi, J. Selsky, M. Hansen and D. Estrin, "AndWellness: an open mobile system for activity and experience sampling," in *Wireless Health 2010*, ACM, 2010, pp. 34-43. |
| [13] | D. M. Aanensen, D. M. Huntley, E. J. Feil, B. G. Spratt and others, "EpiCollect: linking smartphones to web applications for epidemiology, ecology and community data collection," *PloS one,* vol. IV, no. 9, p. e6968, 2009. |
| [14] | E. Kanjo, "Noisespy: A real-time mobile phone platform for urban noise monitoring and mapping," *Mobile Networks and Applications,* vol. XV, no. 4, pp. 562-574, 2010. |
| [15] | P. Chow, W. Bonelli, Y. Huang, K. Fua, B. A. Teachman and L. E. Barnes, "DEMONS: an integrated framework for examining associations between physiology and self-reported affect tied to depressive symptoms," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, 2016. |
| [16] | P. Sun, N. M. Wergeles, C. Zhang, L. M. Guerdan, T. Trull and Y. Shang, "ADA-Automatic Detection of Alcohol Usage for Mobile Ambulatory Assessment," in *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2016. |
| [17] | N. M. Wergeles, "AMD, analysis of mood dysregulation: a machine learning approach," 2016. |
| [18] | M. Rabbi, S. Ali, T. Choudhury and E. Berke, "Passive and in-situ assessment of mental and physical well-being using mobile sensors," in *Proceedings of the 13th international conference on Ubiquitous computing*, 2011. |
| [19] | E. S. Raymond, "Software Design Principles," [Online]. |
| [20] | B. Mart'inez-P'erez, I. De La Torre-Di'ez and M. L'opez-Coronado, "Privacy and security in mobile health apps: a review and recommendations," *Journal of medical systems,* vol. XXXIX, no. 1, p. 181, 2015. |
| [21] | S. Arora, J. Yttri and W. Nilsen, "Privacy and security in mobile health (mHealth) research," *Alcohol research: current reviews,* vol. XXXVI, no. 1, p. 143, 2014. |
| [22] | A. Cheng, V. Raghavarajuy, J. Kanugo, Y. Handriantoz and Y. Shang, "Development and evaluation of a Healthy Copingvoice interface application using the Google Homefor elderly patients with Type 2 Diabetes," in *IEEE Consumer Communications and Networking Conference*, 2018. |
| [23] | Highcharts.js, "Highcharts.js – JS," [Online]. Available: http://www.highcharts.com/products/highcharts/#non-commercial. [Accessed 20 1 2018]. |
| [24] | AngularJS, "AngularJS," [Online]. Available: https://docs.angularjs.org/api. [Accessed 27 2 2018]. |
| [25] | A. Powell-Morse, "Iterative Model: What Is It And When Should You Use It?," Airbrake.io, 15 December 2016. [Online]. Available: https://airbrake.io/blog/sdlc/iterative-model. [Accessed 7 October 2017]. |