Demo Abstract: SmartSAT - A Customizable Secure App for San Antonio Transit Pilot Project

Jeong Yang
Dept. of Computing & Cyber Security
Texas A&M University-San Antonio
San Antonio, TX, USA
jyang@tamusa.edu

Young Lee
Dept. of Computing & Cyber Security
Texas A&M University-San Antonio
San Antonio, TX, USA
ylee@tamusa.edu

William Noonan
Dept. of Computing & Cyber Security
Texas A&M University-San Antonio
San Antonio, TX, USA
wnoonan@tamusa.edu

Anoop Abraham
Dept. of Computing & Cyber Security
Texas A&M University-San Antonio
San Antonio, TX, USA
aabraham@tamusa.edu

ABSTRACT

The purpose of this demo abstract is to present a pilot SmartSAT project, a customizable mobile web app for San Antonio Transit to make critical services available for transit users. SmartSAT app provides real-time bus arrivals, seat capacity information, and instant alert messages on schedule changes. The app also securely collects data and feedback from riders on their commute experience. Researchers and students from diverse disciplines (sociology, computer science, cyber security, & information science) work collaboratively for developing this tool, collect commuters' feedback, and deploy data to improve the quality and inclusivity of transit service. The development efforts demonstrate that SmartSAT is a customizable tool with not only provides transportation information to the riders but also collects their commute behaviors and experience feedback toward improving the public transportation system in San Antonio, TX.

CCS CONCEPTS

• Software and its engineering • Software and application security • Information systems applications

KEYWORDS

Smart city, Smart mobility, Transit, Google map, Google Maps API, Django, Cloud SQL, Real-time arrivals, GPS, Google Cloud

ACM Reference format:

Jeong Yang, Young Lee, William Noonan, & Anoop Abraham . 2022. Demo Abstract: SmartSAT - A Customizable Secure App for San Antonio Transit Pilot Project. *In Proceedings of the 20th ACM International Symposium on Mobility Management and Wireless Access (MobiWac '22), October 24-28, 2022, Montreal, QC, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3551660.3560910

MobiWac '22, October 24–28, 2022, Montreal, QC, Canada

ACM ISBN 978-1-4503-9480-2/22/10. https://doi.org/10.1145/3551660.3560910

1. BACKGROUND

Public transportation connects people to their jobs, schools, medical and recreational facilities, friends, and family. Public transportation users in most American cities have disproportionately lower incomes than commuters who use automobiles [1]. San Antonio (SA) follows this trend, with 67% of SA's VIA Transit users falling below the poverty line. Providing reliable and accessible public transit will benefit all transit users and, therefore, promote the upward mobility, health, and wellness of all San Antonians.

To improve the reliability of transit information and the quality of the rider experience, many transit authorities have developed smartphone apps [2, 3]. Indeed, the public demand for public transportation smartphone apps is increasing in many urban areas [4, 5, 6], particularly among people in lower income brackets who primarily access the internet through their smartphones [7]. However, maintaining sustainable, high-quality service and information on these apps is challenging [4, 5]. Some smartphone transit apps can collect users' GPS-tagged and Quality of Service (QoS) data to evaluate on-time arrival and rider experience, but these apps often fail to characterize trips, neglect to consider the impact of social factors like race/ethnicity and socioeconomic status and lack privacy protection and decision support systems [1].

To serve the needs of VIA Transit riders, especially those who are lower income, a smartphone transit app must provide reliable information about the arrival and seating capacity of buses, which can reduce wait time, reduce travel time, and ensure accessibility [8, 2, 9]. Furthermore, an app must provide a secure platform to collect rider information and feedback so that this information can be used by city planners to make data-driven decisions about transit allocation and service improvements that will benefit all San Antonians.

Google Maps provides public transit information, including route planning and bus arrival and departure times for transit agencies who partner with Google Transit [10]. However, its utility falls short in many other respects that are important for riders (Table 1). Using Google Maps API, other transit authorities have developed smartphone apps that offer enhanced services.

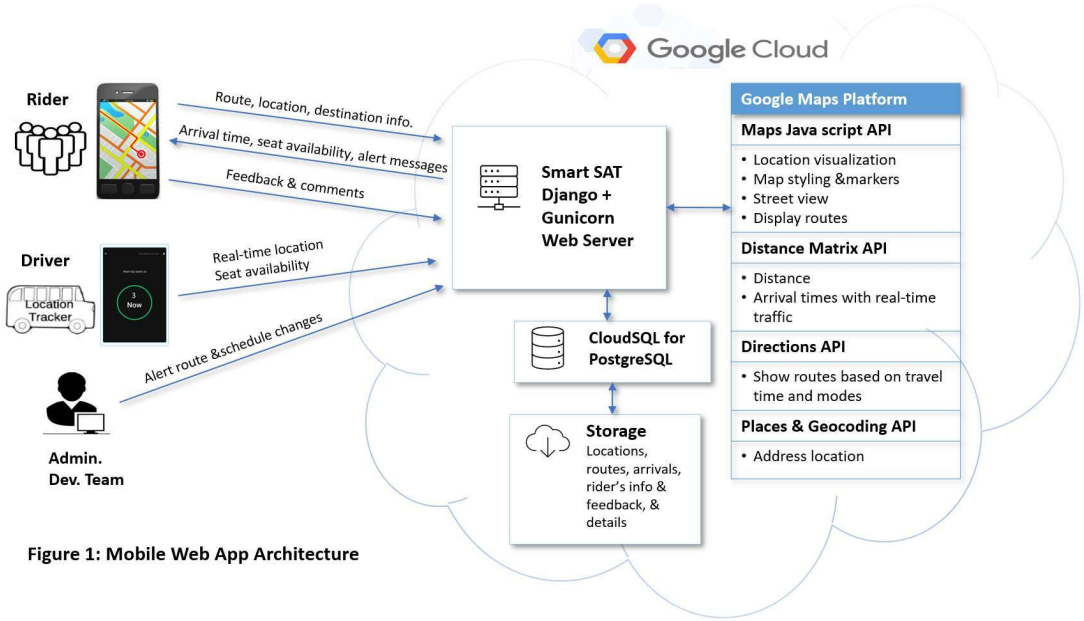

Figure 1: Mobile Web App Architecture

For example, California's Orange County Transportation Authority (OCTA) launched a Transit app in 2020 to provide real-time bus capacity so riders can check enough available seats [2, 3]. The Los Angeles County Metropolitan Transportation Authority partnered with the Transit to improve customer experience by providing accurate real-time bus arrival information [3, 15].

The NSF-sponsored Pitt Smart Living project developed a platform-integrated information system to improve public transportation service with real-time information on arrival and utilization of relevant options of public transit [16, 18]. AC Transit for the Bay Area in California provides real-time bus arrival and bus capacity information and has a platform for accepting rider's feedback [17]. Apps developed by these other transit authorities indicate a demand for enhanced transit information beyond what Google Maps provides.

This demo presents a customizable secure SmartSAT app for San Antonio's Transit pilot system. The app has been designed and developed with the goal of providing real-time bus arrivals and seat capacity information, sending instant alert messages on schedule changes and other important messages to riders, and securely collecting data and feedback from the riders on their commute experience.

2. PROPOSED WORK

SmartSAT is a customizable mobile app for the San Antonio (SA) Transit to make critical services available for San Antonian transit users. While Google Maps and others such as Moovit and Transit provide similar services [10], SmartSAT is designed to build a mobility network with a focus on improving the transit experiences of lower-income people who depend on the SA' VIA Transit because

Table 1. Transit app service comparison between SmartSAT and Google Maps

	SmartSAT	Google Maps
Purpose	To improve the service and rider experience for all San Antonians who depend on transit for work, education, leisure, and other purposes.	To provide the best route options for driving, taking transit, walking, etc.
Target Users	SA VIA riders, specifically those in poverty	Everyone including travelers & visitors
Real-Time Information	Bus arrival times on all stops for rider's route	Bus arrival times for major stops only
	Bus capacity with limited seats available	No service
Feedback	Rider specified comments & survey on trip experience for social impact research	General options to provide feedback about crowdedness, temperature, accessibility
Alert Service	Instant text messages to riders & alerts on maps	Alerts on maps

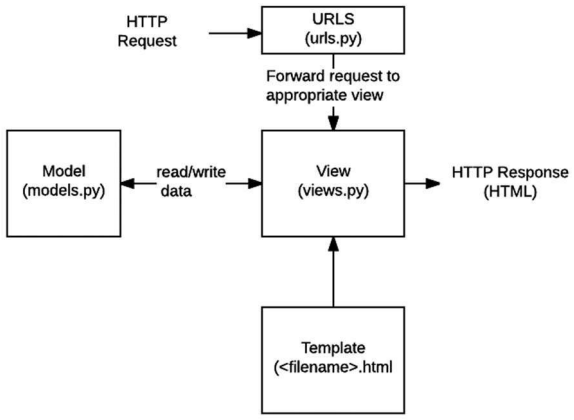

Figure 2: Django Framework

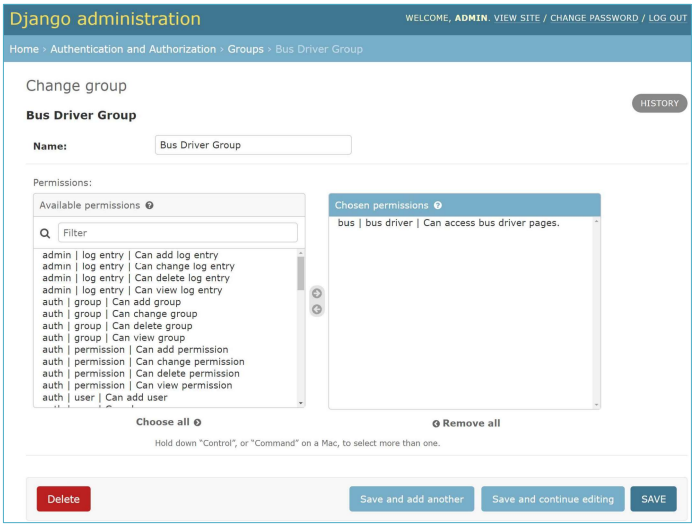

Figure 3: Creating Groups and User Permissions

they cannot afford to drive or lack a driver's license. Through the analysis of actual bus arrival times, the research team will determine the accuracy of current scheduled times and if they need to be changed. We will also assess rider satisfaction and the quality of transit experience, with a focus on those who live at the poverty level. The SmartSAT app is being systematically tested for all aspects of security attacks and to protect rider data privacy and authentication.

The developed app has the following four major functionalities:

1) Providing real-time bus tracking and arrival times at each bus stop for six piloted VIA routes
2) Providing seating availability alerts for buses
3) Sending instant alerts to registered riders about important updates, schedule and route changes, and ridership reminders.
4) Collecting direct information on riders' wait time and feedback about their transit experience to better understand the quality of transit service.

3. DEVELOPMENT

The backbone of the SmartSAT app is Web Architecture. All designed interfaces respond to different screen sizes of devices and different browsers and run well on any mobile phone, tablet, laptop, and desktop.

3.1 Architecture and Framework

Figure 1 presents a system architecture. The app is being developed on a Django + Gunicorn web server that is deployed on Google Cloud using a Docker container. Django is a Python web framework that follows the model-template-views architectural pattern as shown in Figure 2. Django was designed to work with relational databases. The default database is SQLite. When we create a model in Django, which is a Python class that inherits from a built-in Django class, Django automatically sets up an SQLite database table based on the model. "Template" in "model-template-views" refers to HTML templates. We can make Django substitute the values of Python variables into an HTML template before it is sent to a browser to be rendered by using Django's built-in templates engine syntax. "Views" refers to the functions that render the template.

If there is an HTML template called "template.html", then you could create a view function to render this template. When we request the associated URL, Django executes this function and substitutes the values of the keys in the dictionary called "context" for its keys where they exist in the template inside double curly braces. Before running the server, you must associate a URL with this view function. When the server is running and somebody visits a valid URL, the server sends the data from the browser in an object called "request" to the associated view function as an input argument.

Django is hosted on a Gunicorn server being run by Google Cloud Platform's Cloud Run service. It is connected to a PostgreSQL instance created in the Google Cloud Platform. User data, bus data, and route data such as associated bus stop locations and scheduled times are stored in our PostgreSQL instance. When a user interacts with our map, data is communicated with our server via AJAX executions. AJAX is an asynchronous tool that requests a URL in the background allowing the user to interact with the map without having to reload the page. We created special URLs where our server handles requests from AJAX executions on the front end. Google Maps API is used to calculate various things like the transit distance and duration between the coordinates of a bus and the coordinates of a given bus stop.

Figure 4: Real Time Arrivals

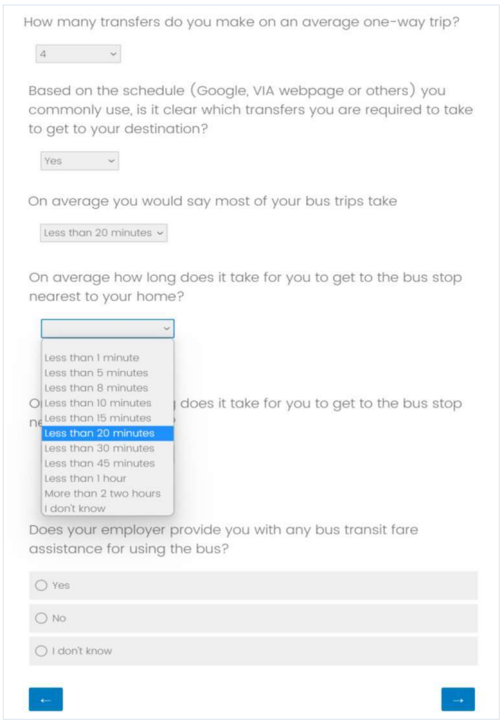

Figure 5: Rider's Feedback Survey

There are some key features of Django administration deployed for this application such as creating groups and user permissions and displaying and customizing navlinks based on user permission. For example, to create groups and user permission, we add custom permission to the Meta class of (any) Django model by setting the value of the Django keyword "permissions" (code below). This will make the permission available on the administration site.

```
class BusDriver(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE)

    class Meta:
        permissions = [
            ("access_busdriver_pages", "Can access bus driver pages.")
        ]

    def __str__(self):
        return self.user.username
```

In the administration site, we create a custom group with special permissions. As shown in Figure 3, a group is created called "Bus Driver Group" with the custom permission created. And then we add a user to a group to give them the permissions defined for that group and add Django's built-in login_required and permission_required decorators to a view function to restrict the access of a webpage to only those users who are logged in and hold the appropriate permissions (code below).

```
@login_required
@permission_required('bus.access_busdriver_pages', raise_exception=True)
def busdriver_view(request):
    context = {'allRoutes': commons.helper.getAllActiveRoutesDropDown()}
    return render(request, "bus/busdriver_2.html", context)
```

3.2 Functional Requirements

The app is designed to be the main connection point between riders, drivers, and administrators/development teams. There are four functional requirements for the app using the following implementation approach. As for the architectural part of the app, it was developed using the Django web framework, which is written in Python and uses the GUnicorn web server, which is a Python WSGI HTTP server.

3.2.1 Providing Real-Time Bus Arrivals

This is implemented with Google Maps Platform web services. Google Maps JavaScript API is the main APIs to retrieve the latest Maps and Routes in the development, where real-time traffic conditions are constantly updated with routes [11]. Directions for transit are obtained by Direction API, and Distance Matrix API is used to calculate travel times based on real-time traffic and distance for multiple stops on a route. Utilizing these APIs, we can conveniently build customized transit maps to fit our project needs. With some exceptions, the mobile web app routes are considered faster and cheaper than the native iOS or Android mobile app routes that use iOS or Android SDK [12, 13].

The Javascript API is also chosen for this part of the implementation because of its capability to enable map controls and gestures for any mobile devices running on web browsers, to provide directions services with real-time traffic information using a form of public transit, and their exclusive support of transit layer with public transport routes, and easy integration with other transit web sites [11, 12]. There would be no installation needed on the user(rider) side as the app is ready to be used through web browsers, but still, be a native app available on google play and apple stores. No updates and registrations are needed on Google Play Store and Apple Store on the developer side. Figure 4 displays a rider's view of a prototype with real-time arrival information for a selected stop on a rider's route on a mobile device.

3.2.2 Providing Bus Capacity with Limited Seats

A bus driver monitors seats and notifies the system when limited seats (≤3) are available. This is done with an easy-to-use interface on a touchscreen tablet attached to a dashboard area of the bus. The seat information is instantly provided in the rider's app as soon as the driver triggers the seat information.

As explained in the figure of Django's model-template-view architecture, communication between a bus and the cloud occurs by using HTTP requests sent to the server via jQuery-AJAX calls made from the client (front end) side. We set up unique URLs on the server where we send such HTTP requests, and these HTTP requests contain various data that we want to send to the server like GPS data and seat availability data, etc. A cellular network is used for the communication from the mobile user with the app's back-end hosted on google cloud as well as from the bus to the cloud implemented.

3.2.3 Alerting Important Announcement & Schedule Changes

Currently, when SA VIA services are detoured, changed, or canceled, alerts are posted on the VIA website, or riders can call customer service. In extraordinary circumstances, service may change without notice [14]. SmartSAT app sends instant alerts to registered riders about schedule changes, route changes, and other unexpected changes.

3.2.4 Collecting Rider's Feedback

SmartSAT app created a comprehensive survey system to collect direct information on riders' wait times and feedback about their transit experience (Figure 5). This is to better understand the quality of service provided by the current SA Transit service. To address the social impact of poverty and racial/ethnic status in the use of current VIA Transit systems, the project will assess riders' experiences and focus on how SmartSAT will make engagement with VIA's transit system more efficient for riders.

4. CONCLUSION

This demo paper presents a SmartSAT mobile web app that provides real-time bus location information, bus seating capacity, instant alert messages, and a secure platform for collecting rider feedback. The SmartSAT plays a connection point between riders, drivers, and administrators: 1) A rider uses the app to choose a route and ride from a bus stop and a destination stop and checks real-time bus arrivals, 2) A bus driver monitors seats and notifies the system when three or fewer seats are available, 3) The administrators deploy alerts about emergencies and schedule changes that are instantly pushed to riders' phones, and 4) Rider experience data is collected through a secure survey.

In the future, we hope that the SmartSAT project will provide more accurate bus arrival times to improve rider experience. From the collected feedback data from riders, we will determine how social class and race/ethnic backgrounds shape a rider's ability to engage the public transit system and whether this access can be improved through a more robust transit data delivery system. Collecting and analyzing the experiences of ridership provides a direct assessment of their interpretations of the usefulness of SmartSAT to improve the overall riding experience and improve the lives of riders through reliable transit data delivery.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation's Grant No. 2131193. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Fan Jiang, Bus Transit Time Prediction using GPS Data with Artificial Neural Networks, 2017, Machine Learning CMU report.
[2] Real Time Infor to Help Riders with OC Bus Capacity. https://www.metro-magazine.com/10124680/real-time-info-to-help-riders-with-oc-bus-capacity.
[3] Transit, Go Your Own Way, https://transitapp.com/.
[4] Smartphone Applications To Influence Travel Choices: Practices and Policies, US Department of Transportation, Publication #: FHWA-HOP-16-023, Last modified May, 2020.
[5] A. Arman, P. Bellini, P. Nesi, & M. Paolucci, Analyzing Public Transportation Offer wrt Mobility Demand, TESCA'19, 2019, New York, NY, USA, https://doi.org/10.1145/3364544.3364828.
[6] T. Jimbo & K. Fujinami, Detecting Mischoice of Public Transportation Route based on Smartphone and GIS, UbiComp/ISWC '15 Adjunct, September 7–11, 2015, Osaka, Japan, http://dx.doi.org/10.1145/2800835.280090.
[7] Marler, Will. "Accumulating Phones: Aid and Adaptation in Phone Access for the Urban Poor." Mobile Media & Communication 7, no. 2 (May 2019): 155–74. https://doi.org/10.1177/2050157918800350.
[8] Carlos Romero, Andrés Monzón, Andrea Alonso, Raky Julio, Added value of a customized transit app for metropolitan bus trips, Transportation Research Procedia, Volume 47,2020, Pages 513-520, ISSN 2352-1465, https://doi.org/10.1016/j.trpro.2020.03.126.
[9] Rajput, P., Chaturvedi, M., and Patel, P. 2019. "Advanced Urban Public Transportation System for Indian Scenarios," Proceedings of the 20th International Conference on Distributed Computing and Networking, pp. 327-336.
[10] Google Transit Basics, About Google Transit, https://support.google.com/transitpartners/answer/1111471?hl=en&ref_topic=3521043.
[11] Google Maps Javascript API, https://developers.google.com/maps/documentation/javascript/overview.
[12] Brainsmiths Blog, https://blogs.brainsmiths.com/post/2020/06/26/developing-mobile-web-applications-when-why-and-how.aspx.
[13] Mobile Website vs Native App vs. Mobile Web App, https://www.bluefountainmedia.com/blog/mobile-app.
[14] VIA Service Alerts, https://www.viainfo.net/service-alerts/.
[15] Metro Magazine, https://www.metro-magazine.com/.
[16] Pitt Smart Living Project. https://pittsmartliving.org/.
[17] AC Transit, https://www.actransit.org/.
[18] Bushman, Kristi and Labrinidis, Alexandros. "Utility-based scheduling for public displays with live content," Proceedings of the 8th ACM International Symposium on Pervasive Displays - PerDis'19, 2019.