Capstone Project Proposal: Customer Feedback Application

Daniel Kushner

CST 499 Directed Group Capstone

California State University, Monterey Bay

August 25, 2019

Table of Contents:

Introduction

The following sections of this paper will cover a proposal for gathering customer feedback. Below, the project definition, goals and objectives will be defined. This will also include evidence the project is needed and the approach that will be taken to gather customer feedback. It will also include an environmental scan, who the stakeholders are as well as ethical implications. And last, it will address project scope and final deliverables.

Project Description

From nonprofits and small businesses to large enterprises, everyone needs a pulse on their target audiences. A user may not be able to find how to donate, volunteer, or maybe the user couldn't find the right results from their search, or the product they were looking for. Many times this is handled by advertisers claiming they can target the business respect audiences via various proprietary tools or embedded analytics platforms such as google analytics. No matter which way this problem is approached, solving this problem inevitably requires customer interaction, and sometimes no amount of code or fancy analytics can beat direct customer feedback.

Businesses could add in forums and allow product reviews, but how does a business get honest feedback, and what if they want the feedback to be private or to get feedback without requiring the user to identify themselves? And what of businesses with limited resources? How do they gather this feedback without having to hire engineers to build it or pay for expensive software or subscriptions? A fantastic solution is the customer feedback widget! With one simple widget, a business could privately, anonymously gather direct customer feedback with the ability to add customized analytics later on. And this customer feedback widget can be built

using a pay for what you use model, making it affordable to businesses of any size. A simple example of a feedback widget would be as follows: The user sees a pop up widget on the page. The widget displays a simple question such as “Did you find what you were looking for?” If they select yes, it submits a yes response and says thank you for you feedback, if they select no, it then displays a new textbox asking for feedback. In the case of this project, it will specifically appear after a search to help determine if the customer found the results they seek.

Project Goals

- Allow business owners to easily deploy a feedback widget to a webpage for gathering direct feedback
- Make the widget an affordable option compared to competitors by following a pay only for what you use model
- Create the widget to provide a smooth and easy user experience for the customer
- Ensure the feedback is stored in a secure location
- Prevent basic spamming and abuse, so customer data isn’t compromised (i.e. you can’t submit the widget over and over)
- Store the data in a way that is easy for non-technical personnel to easily export into common tools such as Excel.

Project Objectives

- Develop an initial generic widget for gathering feedback
- Create easy deployments using infrastructure as code
- Write clear documentation showing the application architecture and how to continue custom development of the application

- Create a tutorial for the non-technical user showing how to deploy the application, attach it to a web page and retrieve customer feedback data.
- Test the application with at least 3 individuals or individual groups

Environmental Scan

There are numerous competitors in this market that provide various customer feedback tools. Among the largest is Zendesk, who also happens to directly provide a customer feedback widget. Each of the major competitors such as Zendesk, Freshdesk, Liveagent and others, all use a subscription based model, that are based on number of users or number of agents. This means regardless of usage you must pay a flat fee and must pay more to move up support tiers for access to additional features. This can be cost prohibitive to smaller businesses and nonprofits. The proposed project addresses this by using a pay for what you use model, to the extent that most small business will have no cost due to the generous allowances under Amazon Web Services (AWS) free tier.

These same competitors not only limit customers by price tiers but also in terms of functionality and user experience. There are limitations on the styling, formatting and integration of their products. If you want that feedback widget to send data to google analytics or integrate it with a custom chatbot, it still requires a custom solution that may not be possible, depending on what api's those feedback widget expose. In the case of google analytics, I have to either separately add it to my webpage on a per case basis, or hope that Zendesk or a competitor has an option on their ui for when I configure the widget. But why would they allow you to send data to a competitor for free? This project is intended to provide

a fully customizable widget that can allow businesses to either develop addition themselves or outsource a simple contract add those additions, keeping their costs limited to the way for what you use approach.

There are several core parts of how this application differs from others, aside from the pay for what you use model. First, the infinite customization that's available. If we look at Zendesk's support widget and it's documentation (Zendesk.com) a user has to build a form from their zendesk account and then follow one of three options: hyperlink, popup hyperlink and embed code. All three of which require the user to add html elements directly to their website. This project will seek to avoid this by allowing a simple html link element to a javascript file containing the customer feedback widget, that will then automatically place the widget on the specified pages. Second, Zendesks approach provides no additional option for gathering meta-data, especially on authenticated users. The proposed customer feedback application will include a data object where the user can choose to bundle any additional meta-data they choose vis javascript and time permitting, or as a future enhancement via the same dynamic JSON formatting that will be described next. Third, while Zendesk, Freshdesk and other competitors state "For more advanced customizations, you can enter in your own custom CSS" (Knowledgebase.proprofs.com), it requires the user to write all their own css. A core differentiator here is the user will be able to create dynamic widgets by allowing the user to define the form elements, form data, and theming using simple json based on Mozilla's react-jsonschema library (/rjsf-team.github.io/react-jsonschema-form). Using this library has the added bonus of not just making the theming dynamic, but the forms themselves. Lastly, a core differentiator is over the ownership of the collected data. In the case of Zendesk

Stakeholders and Community

Potential client/end users have been referenced several times already, but the general focus being smaller businesses and nonprofits due to those entities typically having less financial resources than larger enterprises. However clients/end users are obviously not limited to those smaller institutions. Large enterprises with a do it in house mentality, business recreating their product to use microservices, those simply fed up with trying to integrate various services or wanting to add it to an existing one may find a feedback widget of value. It might even suit a future student learning new technologies or integrating it into a future capstone for a business or non-profit they want to help out. The client/end users really have an open and unlimited potential, simply due to the use of open source technology and a pay for what you use model. To reiterate the point that truly sets this apart and where it can truly impact stakeholders, is the pay for what you use model. Starting and growing businesses, charities or nonprofits is hard work and this provides one more cost effective tool to help make good decisions and drive growth.

The primary stakeholder, whom does not wish to be identified here, frequently gets complaints that the search results of it's return nothing relevant. In this case, they have a single business analyst who is supposed to figure out what's relevant for thousands of customers. This business analyst is also responsible for and attempting to tweak search parameter results, despite lacking any technical knowledge of the underlying search engine. Taking the manual labor of gathering customer data will allow the individual to focus solely on tweaking necessary search relevance.

The impact it has for this stakeholder is high. Per Petra Martíšková and Roman Švec's work on E-shops customer feedback, " Customer feedback can have a form of suggestions for product/service improvements, but it can be also complaints (Celuch, Robinson and Walsh 2015) which indicate a certain level of customer dissatisfaction"(Littera Scripta, 2017). This is near identical to what little customer feedback is currently available and a core goal of the stakeholder here is to address this as simply as possible, hopefully with just a single widget. Petra Martíšková and Roman Švec's research indicated that "About 60% of customers give their feedback to an e-shop where these customers have made a purchase" (Littera Scripta, 2017), showing a high willingness of customer to help improve the experience of the websites they use. That willingness to provide feedback can impact the stakeholder by providing better information on what's wrong, allowing for more accurate improvements, better customer satisfaction and improve customer retention and sales.

Methodology

A basic description of how the feedback application works is as follows: An AWS Cloudformation template is created to allow the repeated provisioning of resources, so each customer can deploy it themselves. An api for sending the customer feedback widget and receiving the feedback data is handled using AWS ApiGateway. AWS ApiGateway calls AWS Lambda functions to execute the code that servers up the feedback widget and stores the data to AWS DynamboDB. The widget that's displayed to the user will be built in react, since it's popular, robust and has strong community support. To simplify the widgets addition to a webpage, webpack will be used to encode the html, images etc into the end javascript that react generates, so only a url is required. AWS DynamboDB is where the feedback data will be

stored and was chosen for its simplicity. Including the ability to directly export of the data to a csv, that a customer with no coding skills can easily import into excel. The cost to business are minimized because react is open source and all of the AWS services are a pay for what you use model. For example, with AWS Lambda and ApiGateway, the first million requests per month are free. The entire project will be placed in a public AWS s3 bucket, so any user can directly add the cloudformation template and deploy the entire project directly from an aws account with a few clicks of a mouse.

A general breakdown/approach of how the application will be developed:

- Create a generic customer feedback widget with react
- Configure webpack to package all non-js files such as images as part of the final compiled react
- Create a generic template for deploying the infrastructure using infrastructure as code. This will provision at least an AWS Apigateway, AWS DynamoDB table(s), several AWS Lambda functions, and store the compiled react widget in an AWS S3 bucket
- Build out the two AWS Lambda functions generated above, 1 for serving up the feedback widget, 1 for processing the returned data and putting it into a database
- Develop documentation and tutorials for the technical and non-technical users

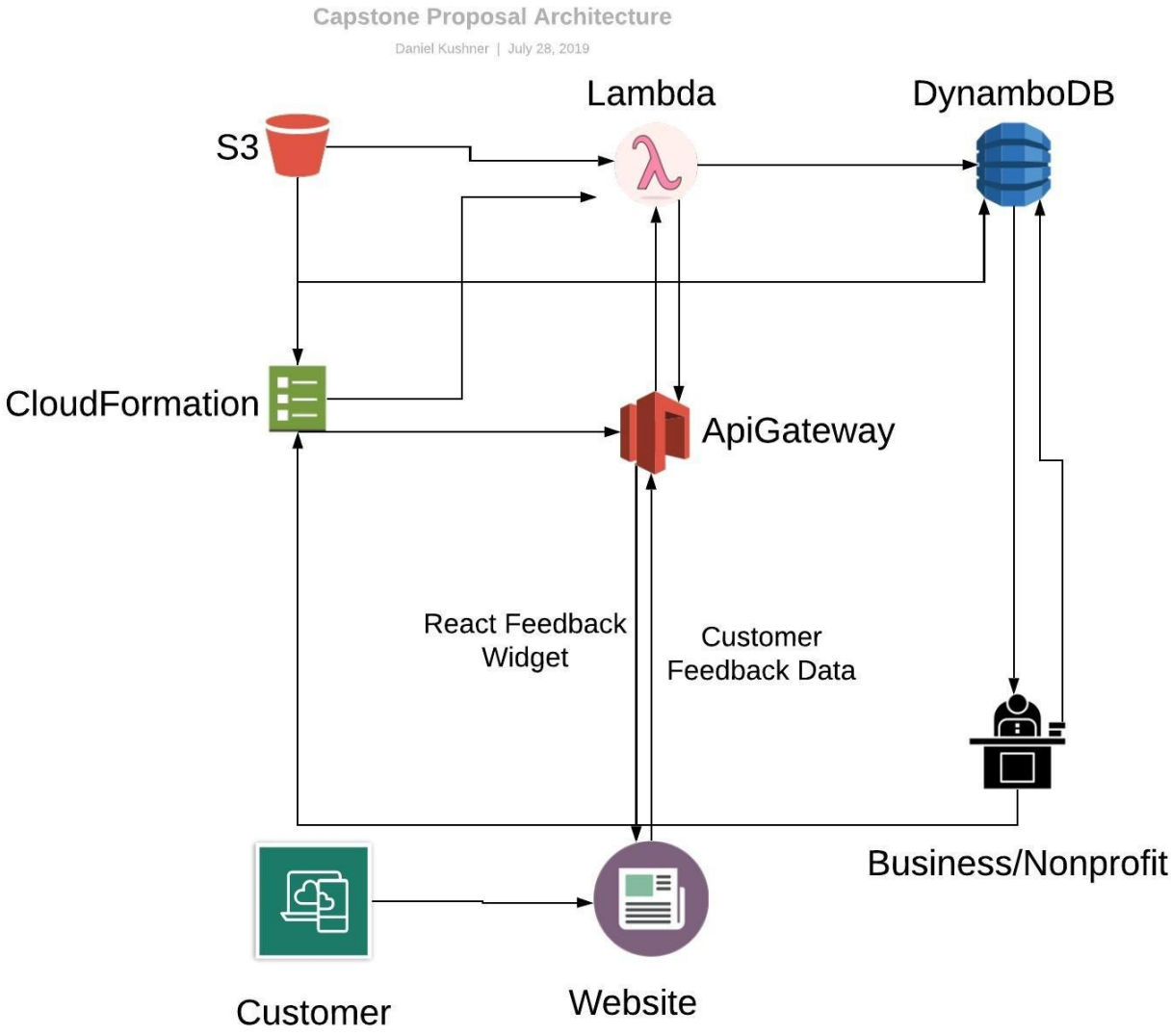
Figure 1: Architeture Diagram of Customer Feedback Application

A general breakdown of how the will application function is described below.

- Create a generic customer feedback widget with react

- Configure webpack to package all non-js files such as images as part of the final compiled react
- Create a generic template for deploying the infrastructure using infrastructure as code. This will provision at least an AWS Apigateway, AWS DynamoDB table(s), several AWS Lambda functions, and store the compiled react widget in an AWS S3 bucket
- Build out the two AWS Lambda functions generated above, 1 for serving up the feedback widget, 1 for processing the returned data and putting it into a database
- Develop documentation and tutorials for the technical and non-technical users

Figure 1, below, is a visual representation of how the application works. The arrows indicate the path data travels through from the customer all the way to the stakeholder.


Capstone Proposal Architecture
Daniel Kushner | July 28, 2019
Lambda
DynamboDB
S3
CloudFormation
ApiGateway
React Feedback Widget
Customer Feedback Data
Business/Nonprofit
Customer
Website

Ethical And Legal Considerations

Ethical concerns for this application are largely centered on how customer data is gathered and handled. This includes what data should be gathered, how is it protected and how is it used. Unfortunately, since this will be self hosted by each business there is a limitation on how some of this is handled. The application itself will use secure libraries, comply with aws privacy and security standards, store data in a secure location and apply best coding practices to ensure common methods of hacking such as SQL injection and cross-site scripting are prevented. Just as importantly, since the application is hosted by the stakeholder, any concerns involving the parties that provide customer feedback tools are mitigated.

Work done by some students at MIssouri University of Science and Technology break down ethical concerns of customer data into several components, social graph, ownership of data, data memory (how long data is stored), data collection, and privacy (slideshare.net). Each presents an ethical challenge that stakeholders themselves will have to grapple with. The best approach to mitigate problems is to provide documentation addressing each of these. The documentation should include a legal disclaimer, that the stakeholder should ensure they are complying with the law and company policy as well as encouraging maximum transparency and clear communication and to customers. Per Meridith Powell, an expert on business growth, “Transparency is key” and “Trust is everything” (meridithelliotpowell.com). This means making this clear and visible to the stakeholders that will use the application and their end users. An additional possible enhancement to promote this idea is to bake a link into the customer

feedback application that directs the customer to a page containing policies derived from the suggestions described above.

Project Scope

Week 1	Design Core Infrastructure	Build a base widget
Week 2	Add form to widget	Define the data object that will be sent for storage
Week 3	Create configurations to encode objects such as images and html into a single compiled javascript file.	Add general javascript functionality for submitting the form to the widget
Week 4	Create an api gateway for the data to be sent to and from	Define the lambda functions that will send the application and receive the data
Week 5	Provide capability for the widget to send the data and regression testing on api gateway and data storage	First stakeholder testing
Week 6	Add theming to the widget	Develop unit tests to prevent regressions
Week 7	Full stakeholder participation and regression testing	Any bells and whistles there might be time for
Week 8	Final Application Submission	

Budget and Resources Required

The budget in this case is minimal, as the aws free tier will cover all the requirements. AWS apigateway and lambda allow the first million executions per month for free and dynamoDB allows 25GB storage for free. Additionally, resources required are minimal as well, since it's a cloud native application, all that is required is an aws account and a computer for development.

Milestones

The following major milestones will mark that the project is proceeding as expected and is meeting the timeline to complete the project.

- Base infrastructure can be deployed via a cloudformation template
- The widget can load onto a webpage
- The widget can display a form with images etc, encoded into a single javascript file
- Users can submit the data and it's stored properly and securely
- The data can be retrieved and used by the stakeholder
- Stakeholders use the application and give feedback and approval

Risks and Dependencies

There are several risks and dependencies that could impact the completion of the project. The first is packing the application into an easy to deliver single file. It's unknown at this time how difficult encoding all html, images and other files will be. The second risk is regarding CORS or cross-origin resource sharing. It is possible that the application and the hosting website might prevent the widget from properly loading or sending data.

Aside from the risks, dependencies are minimal. All infrastructure and code libraries are open source or free. The largest dependency is working with stakeholders to ensure they're getting the application they need. Since this is a solo project, team related dependencies and barriers are nonexistent.

Final Deliverables

The customer feedback application will not only have stakeholder approval but be deployed live on their website and in their own aws accounts. This means the application is live, with customers sending it data and the stakeholder is retrieving the data for analysis. As the application will be in active use, it means good documentation and easy deployment is a core part of the final deliverable. To clarify, the final deliverable to the stakeholder, is the fully deployed live application.

Final deliverables for the course will have to vary slightly since it will be deployed in their own accounts and websites and contain private customer data. The final deliverable for academic purposes will contain a demo website with the widget and the infrastructure deployed in an independent aws account. The data contained in the independent aws account for demo purposes will be gathered during stakeholder testing and approvals. In short, the customer feedback application itself is what stakeholders will be asked to use for submitting their feedback and approvals.

Usability Testing and Evaluation

Phase 1 for testing will begin with component testing, this means individually testing out each part of the application individually and manually. These component tests will be the basic test

cases for the unit tests that will be written. Below is a list of each component that will be tested individually:

- The application can load into a webpage and the form displays correctly
- The infrastructure can be deployed via a cloud formation template
- Each Lambda function individually performs it's post and get http requests
- The api gateway endpoints are reachable via https requests
- The widget can successfully load the form and images
- The widget can successfully send data

Phase 2 will consist of integration testing as follows:

- The customer feedback widget can load from api gateway
- The widget can send data to the api gateway
- The api gateway successfully passes requests to the appropriate lambda functions
- The lambda functions correctly send/store data
- The data can be retrieved via csv from dynamoDB

Phase 3 for usability testing is live stakeholder testing. There will be two components to this. First a demo website, in which the stakeholder will submit their thoughts and feedback on the widget. Second will be a trial run of the widget on their own websites. Feedback for this component will be gathered via a simple google form survey .

References

Amazon web services free tier, Retrieved August 23, 2019 from https://aws.amazon.com/free/?all-free-tier.sort-by=item.additionalFields.SortRank&all-free-tier.sort-order=asc

Cardona, Mandal, Nadathur(Oct, 2016), Godse Ethical Issues with Customer Data Collection Retrieved August 23, 2019 from https://www.slideshare.net/PranavGodse/ethical-issues-with-customer-data-collection

Mozilla React JSON Schema, Retrieved August 23, 2019 from https://github.com/rjsf-team/react-jsonschema-form

Martíšková & Švec (2017), E-shops and customer feedback: experience by Czech B2C customers, Retrieved August 23, 2019 from https://www.academia.edu/34228901/E-shops_and_customer_feedback_experience_by_Czech_B2C_customers

Powell, Meridith (Jan 2015) What are the ethics of customer data mining?, Retrieved August 23, 2019 from https://www.meridithelliottpowell.com/ethics-customer-data-mining/

Zendesk Support Widget, Retrieved August 23, 2019from https://knowledgebase.proprofs.com/zendesk-support-widget