Table of Contents

Background of H-E-B

Established in 1905, H-E-B Grocery Company, LP, started as a local family business and is now an American privately held supermarket chain. Headquartered in San Antonio, Texas, H-E-B operates more than 300 stores with over 10000 employees in around 150 communities across Texas. H-E-B Digital is a section of H-E-B that concentrates on providing software solutions to various business problems within the company. H-E-B digital has teams of engineers, analysts, designers, researchers, eCommerce experts and managers. As part of our capstone project, we will work with the Enterprise Engineering team and Healthcare & wellness team, subparts of H-E-B digital. We would also be expected to constantly interact with the Cloud infrastructure and Software security teams.
Per Forbes, cybercrimes will amount to $1-2 trillion annually. Cybercrimes can cause significant loss in operations, customer relations and brand image to an organization. Mistakes can happen during design and architecture as well as during implementation and process. Flaws can also exist across diverse programming languages and computer system components, leading to various vulnerabilities. Unknown software defects are known as zero-day vulnerabilities. These are particularly dangerous because until they're identified and fixed, they can be exploited by attackers. Code vulnerability can create a potential risk of compromising security. Enforcing security standards and practicing them during development can prevent software security vulnerabilities.

Business Problem

H-E-B faces challenges keeping the teams informed regarding approaching deadlines for addressing various issues associated with projects in a scheduled manner.

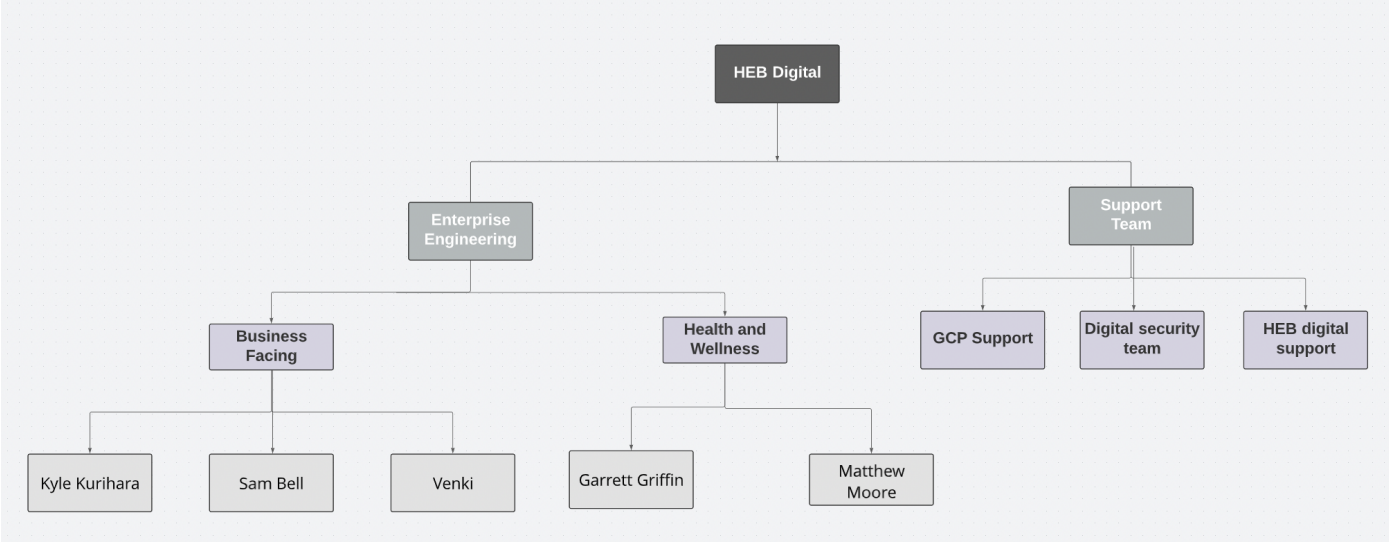
Business Problem Overview

H-E-B faced a security breach on or around January 11, 2021, impacting H-E-B Center IT systems. A third-party service provider that performs IT services for H-E-B Center discovered that its systems had been attacked by sophisticated malware. The attacker had gained access to various H-E-B Center IT systems, encrypted their contents, and exfiltrated this data to the attacker's own systems. These systems contained confidential and personal

information of current and former H-E-B Center employees, such as their name, address, contact information, date of birth, and in some cases, social security numbers. Given recent security breach incidents, H-E-B would like to proactively reduce the attack vector for potential security threats. Post the attack, they limited external access to all IT systems and engaged outside experts to investigate the incident and attempt remediation efforts. To combat the issues related to susceptibility in code, H-E-B's software security team has implemented a system that performs daily scans.

The scans are also performed when the user triggers a merge request. These scans are run on the entire code base, check for flaws, and return a report. The report contains detailed information regarding the potential issues with the code. Thereby providing an opportunity to remediate the problem before releasing it into production. These reports are highly beneficial for the company and the engineers or developers involved in active development. While these scans act as a proactive measure towards code awareness, they have limitations. For example, the teammates must manually look up the report to get information regarding the raised bugs. This costs the employees valuable resources and time. Furthermore, employees may not perform these report checks frequently for various reasons. Teams can grow complacent over a while. Since automated JIRA bug creation is not incorporated with the security scan feature, employees need a way of being notified. Finding and assessing the bugs at the earliest stage is crucial to every product cycle. Streamlining the process of identifying the bugs and notifying the responsible team and the employees involved is of the utmost importance to H-E-B.

The Who's-Who Of Our Project



HEB Point-of-Contact chart

S.No.	Name	Role	Responsibilities
1	Garrett Griffin	Software Engineering manager (Pharmacy)	Plan strategies for the development, design and products on schedule. To assist in budget management.
2	Kyle Kurihara	Software Engineering manager (BF)	Analyze, propose and manage tasks, technologies and resources for projects.
3	Matthew Moore	Software Engineer (Pharmacy)	Assist in developing software design, architecture and document development phases.
4	Sam Bell	Systems Engineer (BF)	Point of contact for issues, act as a collaborator and help communicate with internal teams and vendors to fix problems.
5	Venki	Systems Engineer (BF)	Provide input in strategic technical decisions and solutions as and when needed.

(BF) - Business Facing

Client Project Goals

Currently, H-E-B has excellent vulnerability scanning for applications, but using this information effectively is where the hardship is. To help H-E-B act on their security findings, a tool is needed to take in the Common Vulnerabilities and Exposures (CVEs) and notify the teams of the deadlines around remediation so that no vulnerability will go unfixed by the teams before the deadline. By doing this, H-E-B can create more secure software that meets their criteria. With these reminders in place, there is a lesser chance for the teams to forget that they must work through these various issues.

So, the idea is to create a Slackbot that will integrate with H-E-B's Gitlab repositories and send the vulnerability scan information to the respective team's slack channel. Having reminders sent to various slack channels that have issues to remediate will be leveraged by the teams across the engineering space. Setup of the completed Slackbot should allow the user to specify the channel to report to. That is, which repositories to report the vulnerabilities for, and when to report the vulnerabilities. Reported CVEs should be in the order of longest-lived vulnerabilities. Allowing the users to create JIRA tickets based on the vulnerabilities presented.

This table below maps business goals with technical goals:-

ID	Business Goals	Technical Goals ID
BG1	Read the Common Vulnerabilities and Exposures (CVEs) scans of each repository	TG1
BG2	Notify respective teams of the deadlines for fixing vulneribilities	TG1, TG2, TG3
BG3	Report the vulnerabilities to multiple channels responsible for fixing it	TG2, TG3
BG4	Teams should prioritize ones with closest dealines and with high level of vulnerability over others	TG5, TG6, TG4
BG5	No need to notify the teams during holidays	TG7
BG6	Frequency of notifications of the vulnerability to increase with moving closer to deadline	TG4

ID	Technical Goals	Business Goals ID
TG1	Create a Slackbot to integrate with Gitlab repositories and send the vulnerability scan information	BG1, BG2
TG2	Send the deadline reminders to various slack channels	BG2, BG3
TG3	Slackbot should allow the user to specify the channel to report to	BG2, BG3
TG4	When and how frequently to report these vulnerabilities	BG6, BG4
TG5	Reported CVEs should be in the order of longest-lived vulnerabilities	BG4, BG2
TG6	Mute specific levels of vulnerability	BG4
TG7	Automated Holiday mute	BG5

Technical goals

1. create a Slackbot that will integrate with Gitlab repositories and send the vulnerability scan information

2. Send the reminders to various slack channels

3. Slackbot should allow the user to specify the channel to report to

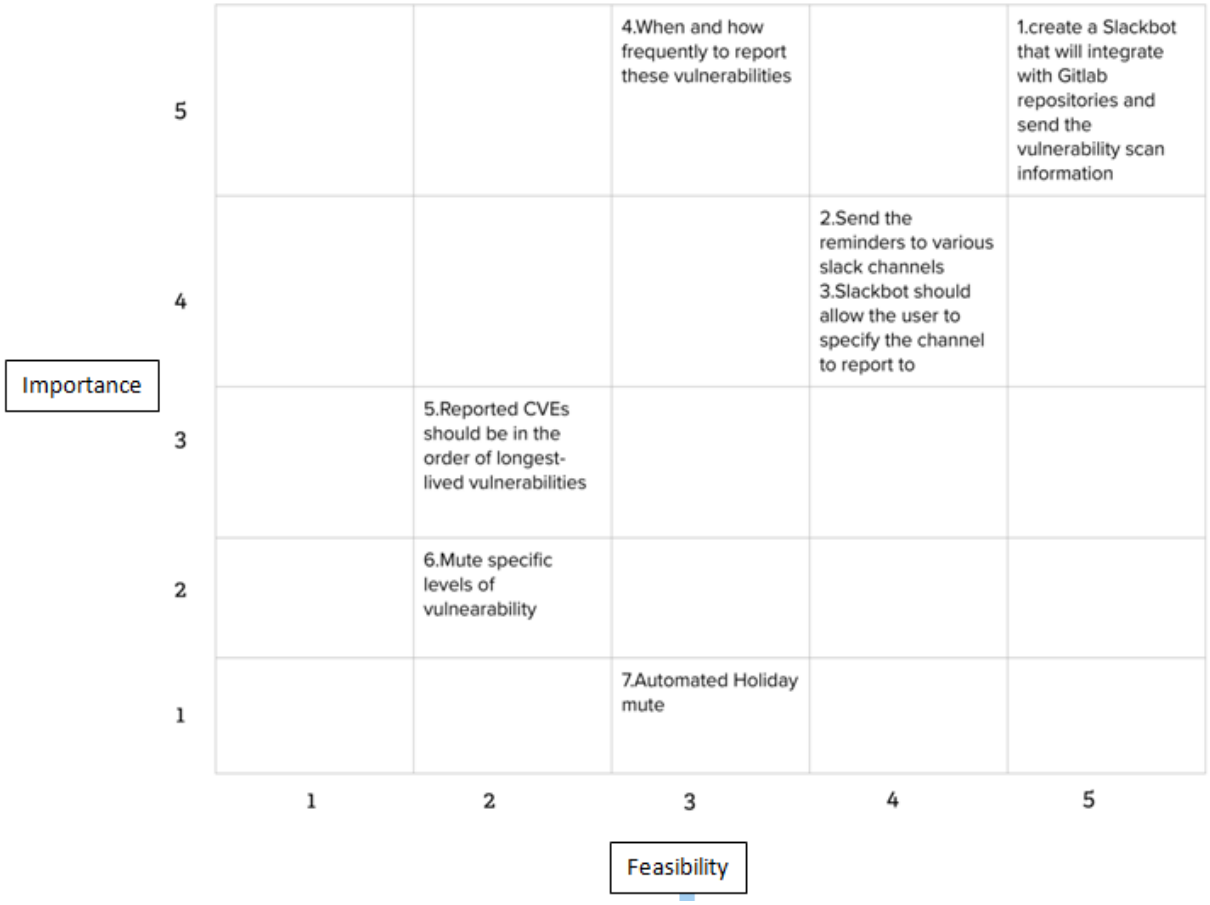
4. When and how frequently to report these vulnerabilities

5. Reported CVEs should be in the order of longest-lived vulnerabilities

6. Mute specific levels of vulnerability

7. Automated Holiday mute

Importance and Feasibility of Technical Goals



Updates in Goals/feasibility

For this report, we could phrase the goals better than the fall report as we better understand the business requirements more precisely. Apart from that, some of the updates included:

1. Change in the feasibility of Goal 6, "Mute specific levels of vulnerability": Initially, during the fall, we assumed the scan report would have some vulnerability score. But now we have information that the report will not consist of any score number. Implementing this feature might need extra work.
2. Change in the feasibility of Goal 7, "Automated Holiday mute": During the fall, we did not have an approach to address this goal, so we assigned lower feasibility. Now, we came up with a way of adding the feature by maintaining a table in the database with the holiday details and performing a lookup whenever necessary.
3. The goal in the fall report was "Setting a time frame for alerts based on priority and setting up customizable times for when Slackbot sends reminders". Since we better understood the term priority here, we updated the goal and added a new goal, " Reported CVEs should be in the order of longest-lived vulnerabilities". Also, we increased the importance of goal 4, based on the communication with the H-E-B team.

Scope and Requirements

In concrete terms, the project's primary goal is to link the GitLab security scan(s) to the Slack communication channel(s) at H-E-B. The project will include a Slack app which behaves similarly, in some ways, to a human Slack user and enables developers to perform remote procedure calls by mentioning the bot in their channel (e.g., "@securebot add gitlab.heb.com/example_repository") or using a slash-command (e.g., "/securebot-add-repository gitlab.heb.com/ex_repository"). The scale and extensibility of the project can vary greatly depending on how easily and quickly we can create this linkage. The most straightforward way to characterize the difference between the bare-minimum product and the top-of-the-line product is that the former need only communicates the security scan from GitLab to Slack. In contrast, the latter would include a wide variety of

customization opportunities for developers in various Slack channels, tracking different code repositories, with control over *what* content is shown in the channel (e.g., specific channels may only be concerned about high-priority vulnerabilities), *when* (e.g., some channels may prefer certain times of day/days of the week and exclude certain holidays), and *how often* (e.g., one repository may need to be scanned and reported on weekly, whereas another may be less important, needing only a monthly checkup).

Must include

The project must, at minimum, be capable of Slack-ing the results of GitLab security scan(s) from one hard-coded GitLab repository to one hard-coded Slack channel on demand (TG 1). With one call to an API / slash command / Slack event, the project must communicate the results of a scan from a specific repository/channel combination, pre-selected by the developers, via Slack.

This form of the deliverable is merely a proof-of-concept because the developers would be forced to manually add a repository by editing the code/environment variables of the project itself anytime they wanted to view the scans of an additional repository, remove a repository from scanning, etc. (TG 2, 3, and so on are missing). However, it would provide much of the boilerplate code necessary to create a more usable version of the project later.

breviter

- Tracking one repository's scans in one slack channel

Should include

The project should include customization options using a predetermined workflow that allow developers to add and remove specific repositories from Slack scan-forwarding without leaving the Slack workspace. In essence, a developer at H-E-B should be able to

follow a workflow which adds a GitLab repository to their channel (TG 2, 3). This workflow should include the option to specify the time and day of the week they will receive notifications about the repository (TG 4). The workflow should also have the opportunity to determine how frequently the channel will receive notifications regarding that particular repository (e.g., daily, weekly, monthly, etc.). The reverse operation must also be possible, allowing a developer to remove a hold from the list of repositories whose channel receives scans.

breviter

- Tracking multiple repositories in multiple slack channels
- Removal of specific repositories from tracking in channels as needed
- Nice UI to perform these two operations

Might include

Given a highly productive outcome, it may be possible to include quality-of-life features that don't strictly need to be in the final deliverables (TG 5). For instance, specific channels may wish to silence scanning notifications on certain days of the year (TG 7). A slash-command to do such a thing is possible to include and desirable, but not easy or essential enough to focus on initially. Additionally, certain teams may be concerned only with specific levels of security vulnerabilities in certain repositories (e.g., Team A only cares about critical vulnerabilities on repository B), so adding the ability to customize which types of scan results are sent to Slack is something we would like to be able to deliver (TG 6). Whether doing so is feasible or not depends mainly on the GitLab scan APIs in question.

breviter

- Silencing of scanning notifications on certain days of the year
- Silencing of certain vulnerability/exposure levels

Won't include

The project will not include a direct interface with other services which H-E-B currently uses, like Jira. Given that the security scans reflect issues in the code repositories which the teams must solve, and the teams track issues in the code which they must solve using Jira, it made sense to our team to pursue a direct creation of work tickets in Jira from the results of the scans. Although we still believe this idea is valuable, H-E-B's developers informed us that the Jira API is unwieldy and not worth pursuing this project.

breviter

- No Jira integration (perhaps desirable but unfeasible)

Updates

Due to the sensitivity of the information being communicated with the Slack bot, the connection to the Slack bot and communication with any other services must be secure. In pursuit of this security, H-E-B informed us that they want the bot to maintain a connection to Slack via WebSocket mode rather than an exposed HTTP endpoint. Additionally, H-E-B expressed that they would prefer the architecture be constructed in the Google Cloud Platform. No updates to the functionality/purpose of the project have been given at the time of writing.

breviter

- Secure connection *via* websocket
- Architecture built on Google Cloud Platform

Timeline and Milestones

Project Timeline Flow

Sprint Roadmap (Until February 24th)

PROJECT ROADMAP : SECURITY THIRD - HEB

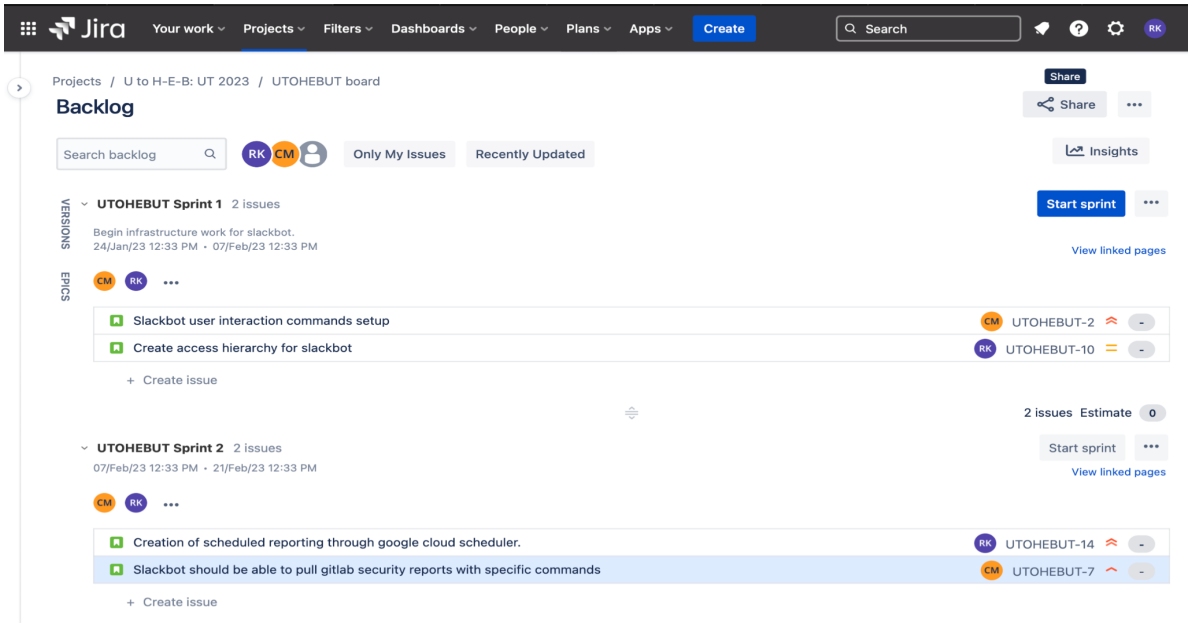
Project Setup Work

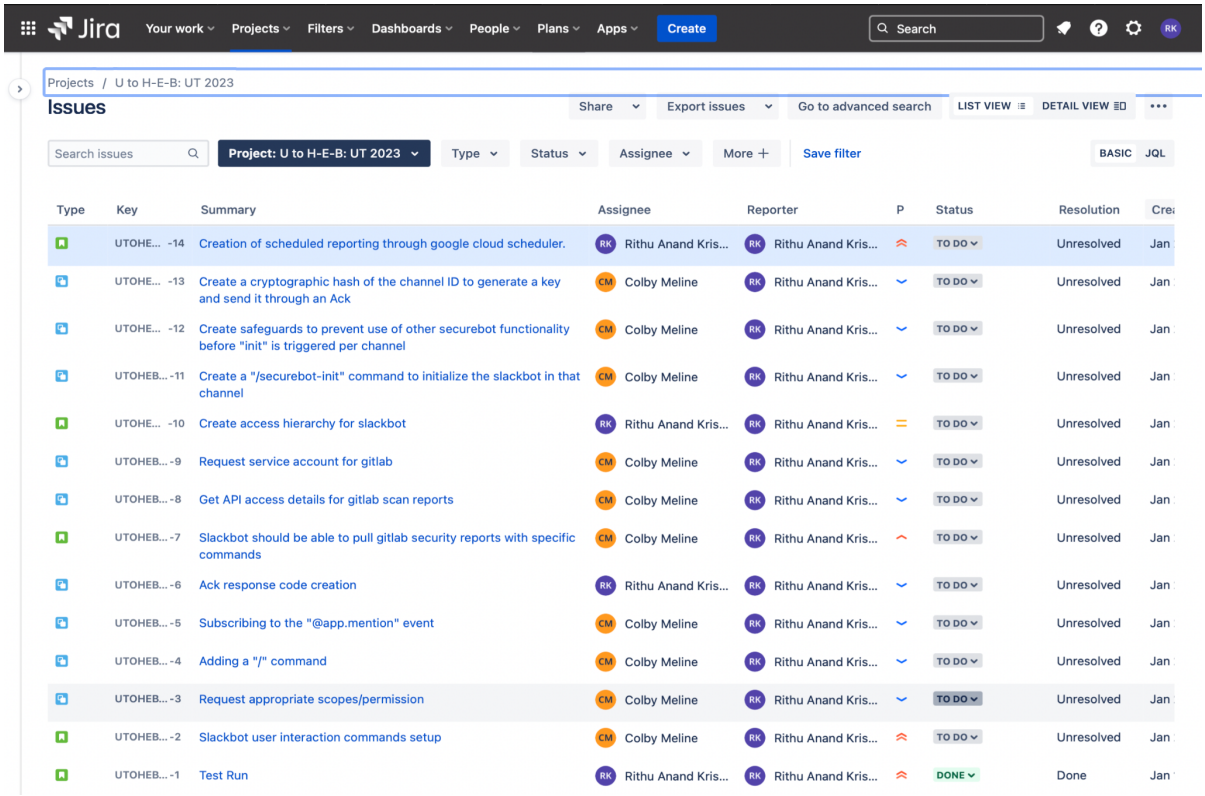
ACTIVITIES	TASKS	Completion Status (%)	week# 0 1/9-1/23			COLOR DEFINATION
Planning	1. HEB Business introduction	100%	Complete			COMPLETE
	2. HEB Organization Accesses (VPN)	100%	Complete			IN PROGRESS
	3. HEB Client KT meeting	50%	IN PROGRESS			PENDING START
	4. Project Pitch Report (Internal to MS-ITM) - I	100%	Complete			BEHIND SCHEDULE/RISK
	5. Project Requirements gathering	100%	Complete			

			SPIRINT 1 : 01/24/2023 - 02/7/2023		SPIRINT 2 : 2/7/2023-2/21/2023	
ACTIVITIES	TASKS	Completion Status (%)	week# 1 1/24-1/31	week# 2 2/1-2/7	week# 3 2/7-2/14	week# 4 2/15-2/21
DEVELOPMENT	UTOHEBUT-2	0%	PENDING START			
	UTOHEBUT-10	0%	PENDING START			
	UTOHEBUT-14	0%			PENDING START	
	UTOHEBUT-7	0%			PENDING START	
TESTING	UTOHEBUT-2	0%		PENDING START		
	UTOHEBUT-10	0%		PENDING START		
	UTOHEBUT-14	0%				PENDING START
	UTOHEBUT-7	0%				PENDING START

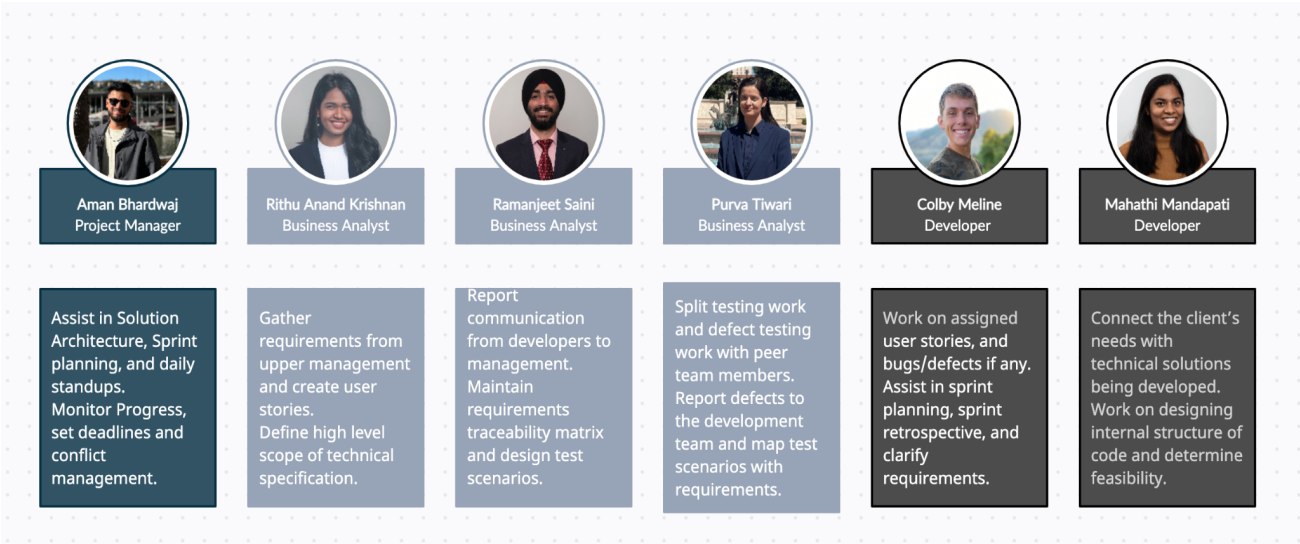
User stories and Sprint details

Jira Board Screenshot (Next Two Sprints)





McCombs representatives



S.No.	Name	Role	Responsibilities
1	Aman Bhardwaj	Project Manager	• Assist in solution architecture • Assist in sprint planning and daily standups • Assist in sprint retrospectives • Monitor progress and set deadlines • conflict management
2	Colby Meline	Software Developer	• Work on assigned user stories and bugs/defects if any. • Assist in sprint planning, sprint retrospective and clarify requirements.
3	Mahathi Mandapati	Software Developer	• Connect the client's needs with technical solutions being developed. • Work on designing internal structure of

			code and determine feasibility.
4	Purva Tiwari	Business Analyst	• Gather and define requirements and break requirements in Business Requirement (BR), Functional Requirement (FR) and Non-functional Requirements (NFR) and create user stories. • Map test scenarios with Business Requirement (BR). • Split testing work and defect testing work with peer team members.
5	Rithu Anand Krishnan	Business Analyst	• Gather requirements from upper management and create user stories. • Define high level scope of technical specification.
6	Ramanjeet Saini	Business Analyst	• Report communication from developers to management. • Maintain requirements for traceability matrix and design test scenarios.

Risks and Mitigation Plans

Technology Risk

- Some of our team members need to gain hands-on experience in the google cloud platform and docker, which can cause delays in our deliverables.

  Mitigation Plan

  H-E-B will give a walkthrough around the GCP environment and small docker training so that we can easily understand with relation to our previous cloud experiences

  We will leverage documentation for the GCP environment and Slack integration from H-E-B, as they have maintained it as per their infrastructure regulations.

  Updates since fall

  There have been no updates for this risk from the previous report. We will follow our mitigation plan to handle this risk.

Operational Risk

- Since we are working on multiple platforms, there are chances that the integration does not work as expected. In the previous report, we mentioned we have to ensure that the bot should report only in the HEB slack environment and not anywhere else.

  Mitigation Plan

  We will have 2 levels of work review before it is implemented or pushed to production or UAT environments. Moreover, we will have our end-to-end testing within our team before moving it to the client's environment. Lastly, we will ensure everyone follows the guidelines H-E-B provided to work in their development environment.

Updates since fall

This is an ongoing risk, and we will try to overcome it with our defined mitigation plan. There is no update from the previous report.

Scope Creep Risk

- Since it is a short-duration project, there will be some hiccups, including changes in project deliverables in the middle of the sprint. This can make it challenging to adjust our project roadmap and fit within the stakeholder's expectations.

  Mitigation Plan

  We will plan our roadmap and sprint with some cushioning for developers to accommodate last-minute change requests if needed.

  Updates since fall

  This is a new risk we have defined while building the project roadmap. We will work according to our mitigation plan to handle it.

Communication Risk

- Team members are enrolled in different course electives, making it difficult to schedule meetings and work together outside class hours. Moreover, we need some help with scheduling stand-up meetings simultaneously, and troubleshooting might take longer in some cases.

  Mitigation Plan

  Blocked everyone’s calendar for daily standups with team members and weekly standups with stakeholders so that we could regularly discuss urgent issues, if any, without any delays.

Updates since fall

There have been no updates for this risk from the previous report. We will follow our mitigation plan to handle this risk.

Updates in risk management

Comparing our risk management plan from the previous report, we have addressed organizational risk as we have been provided with required points of contact supplemented with an escalation matrix.

However, we still face communication risks which will be prevented above with respective mitigation plans. Technology and operational risks are the same as previously mentioned in the management plan. These ongoing risks will be addressed with the project deliverables and sprint plan.

Security Risks

Since the nature of our capstone is related to assisting with HEB's code vulnerability scans, developing a tool that helps with this comes with inherent risks like unauthorized exposure. We have reviewed some of these risks with HEB.

Code Handoff

A potential risk is the process of code and knowledge handoff to HEB at the end of our capstone development cycle. Our team would need to ensure that all the work we do is always on HEB's systems and all the work done on our local system is securely transferred to HEB and disposed of on our machines.

Environment Secrets

A significant risk associated with the nature of this project is related to environment variables and secrets like authorization credentials. These secrets will be shared with us and reside on our machines to deliver our work. A possible way to mitigate this risk is to devise a plan to reset or deactivate these credentials at the end of our development cycle.

Vulnerability Scans in the wrong hands

The most considerable risk is associated with the handling of the HEB codebase vulnerability scans. These scans hold very sensitive information and need to be handled with extreme care because if these scans get in the hands of unauthorized users, they could compromise HEB's software and result in potential cyber security attacks.

Slack and External Tools

HEB uses a lot of external cloud-based tools like Slack and GitLab. Since these are external tools, storing and transporting sensitive vulnerability scan information on these platforms is a considerable potential risk of exposure to unauthorized access. There needs to be more care put into handling who gets to see what through a role-based access system. For example, in a given channel, only certain users should be able to see vulnerability scan information. We must make design decisions to avoid these exposures so only the right people can access these scan reports.

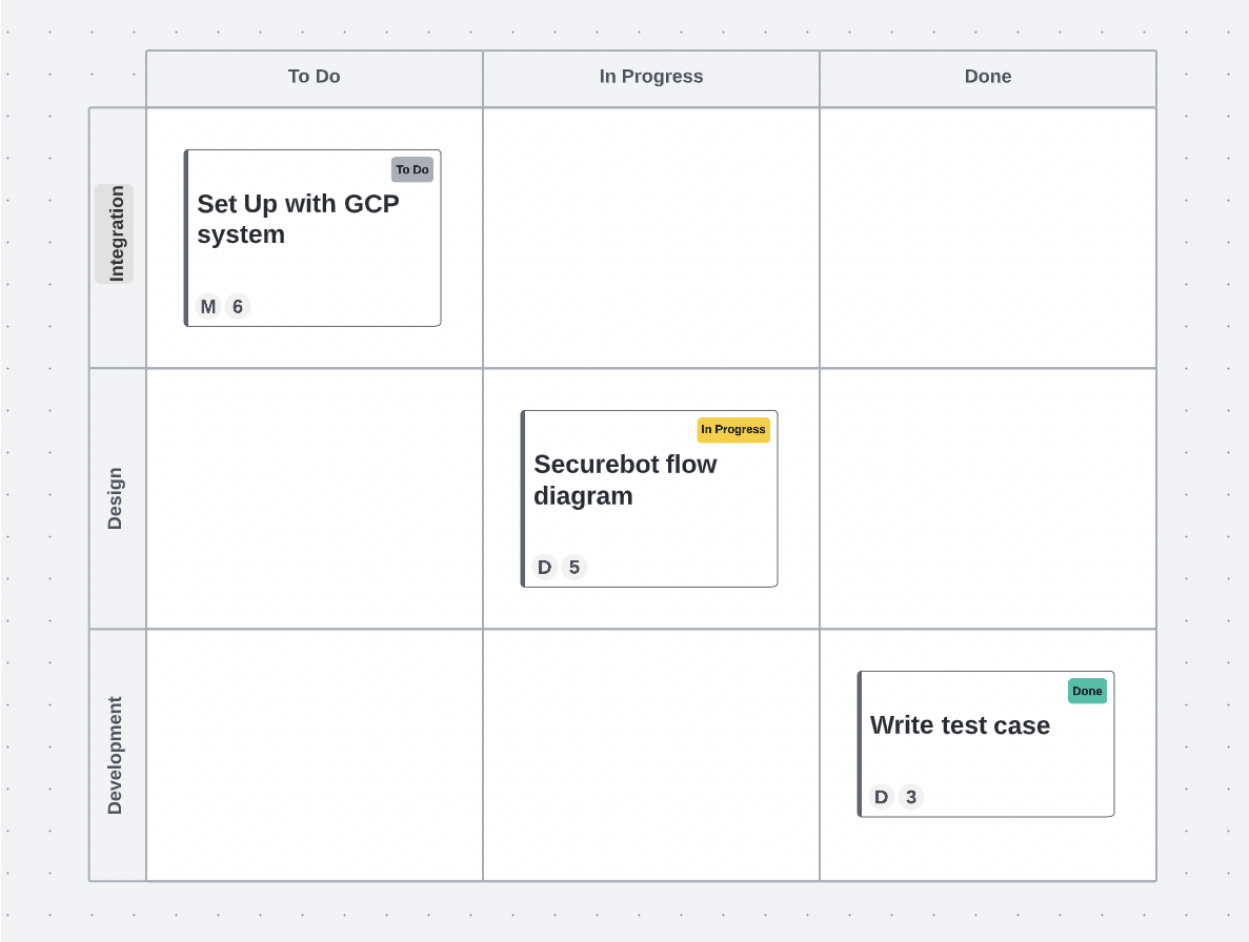
Service Accounts

Service account misuse is an inherent risk with all automated cross-platform functionalities. Since this slackbot will interact with GitLab and Slack, we need to ensure that GitLab service account credentials are secure and not exposed to anyone. We will also need to ensure that the Service Account can only access the information it is supposed to access and nothing else.

Project Management Plan

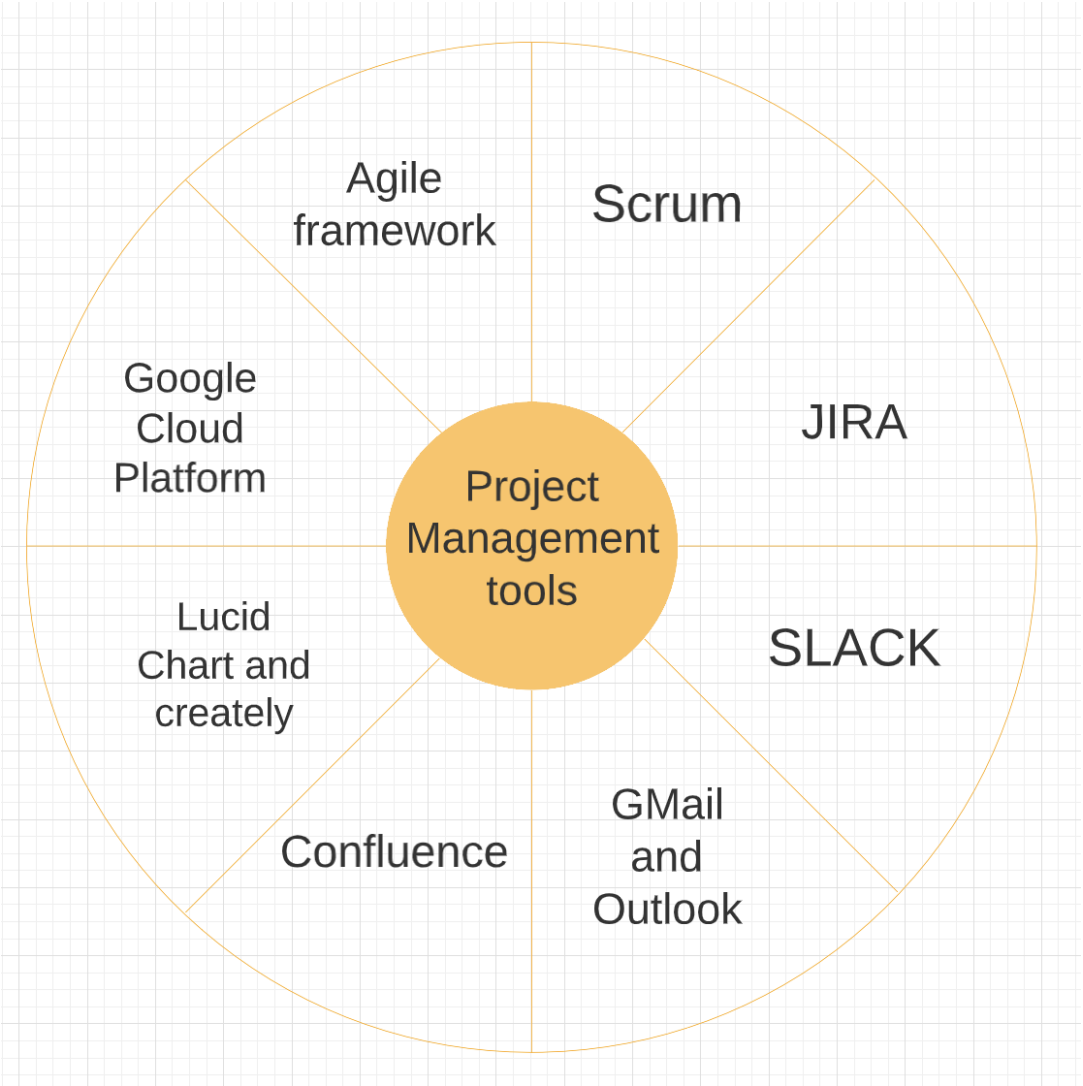
We have decided to follow **Scrum**, an AGILE framework, as our choice for project management structure. For a Capstone project of this nature, we have agreed upon using the same project management toolset HEB's internal teams use. Our team also brings in a diverse set of expertise from industry and academia, so we will fill in the gaps as needed with our choice of tools.

Here is an example as of how the agile board will look like:-



Tools of the Trade

We will use **Jira** to write, assign and track user stories to adhere to the Scrum framework. We will use Zoom for virtual meetings with HEB and among our team members. Since HEB uses **Slack** internally, we have decided to use Slack for instant messaging and daily stand-ups with our HEB counterparts. For the daily stand-ups, a Slack bot will be added to our team's dedicated slack channel in HEB's Slack workspace, including us and some client team members. This bot is dedicated to assisting in daily stand-ups. This bot will ask each member specific questions to post daily updates per HEB. For all emails, we will use **Gmail and Outlook**. We will use **LucidChart** for architecture designs and flow charts and **Confluence** for writing and posting documentation related to all deliverables.



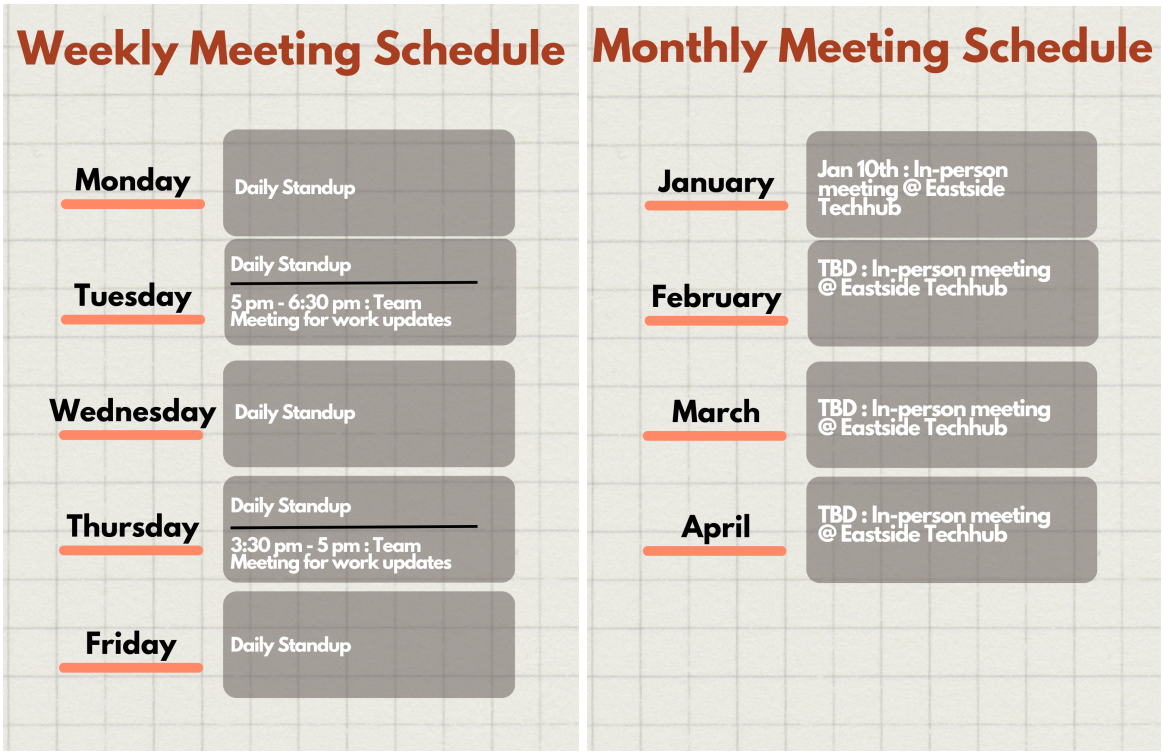
Collaboration Times, Meeting Expectations and Logistics

We will have daily standups with our HEB counterparts and meet twice weekly within our team. For daily standups, we will be assisted with a slack bot to post our daily updates within the dedicated slack channel in HEB's slack workspace. Suppose someone from HEB would be needed for any infrastructure issues, like access or architecture-related questions. In that case, we will request a Zoom or in-person meeting at the Eastside tech hub, if preferred.

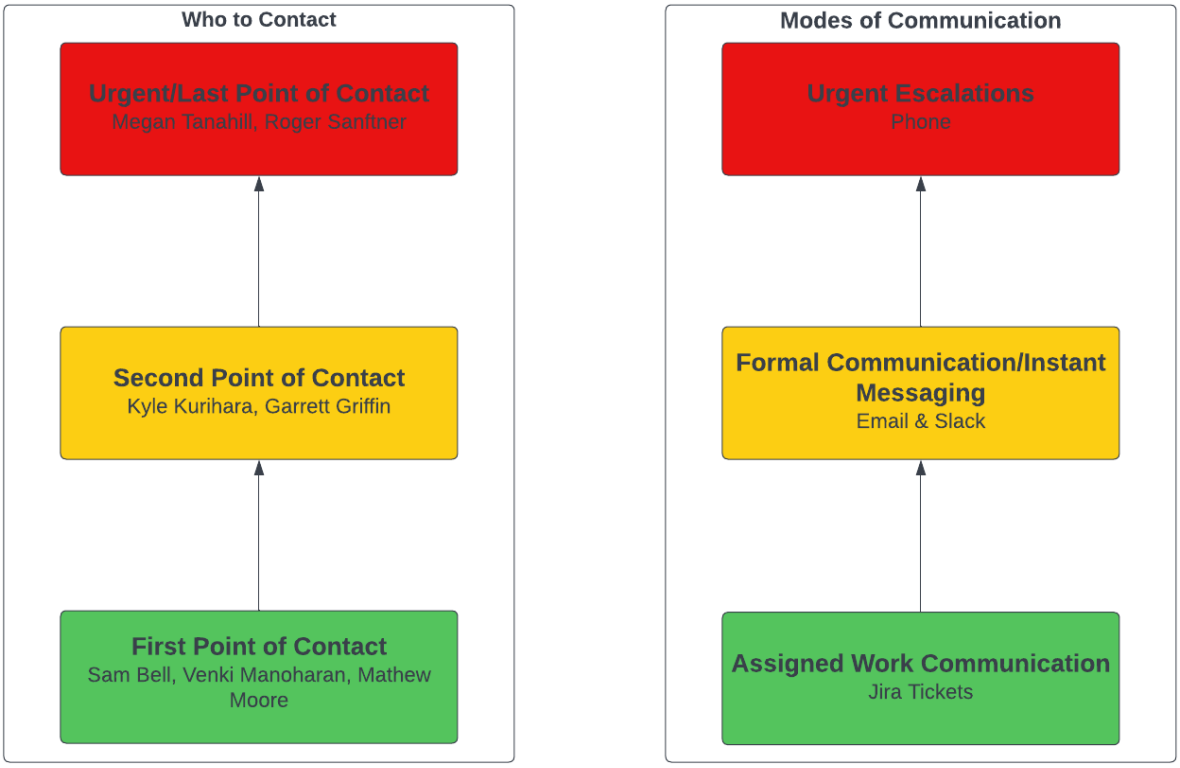
We plan to meet twice weekly during or after Capstone class times as preferred by the team (Tuesdays and Thursdays from 3:30 pm through 5 pm if we do not have a Capstone class meeting or 5 pm through 6:30 pm as preferred). We will hold these meetings virtually

or in person, depending on the team's preference. The meetings will be in place to discuss updates on team deliverables and team submissions. We will also discuss issues our team's developers might have in working on their deliverables from assigned user stories or any collaboration that any team member may need. We will also discuss any issues to bring up with HEB's team.

We plan to participate in Slack bot-assisted daily standups in the dedicated slack channel created for us in HEB's slack workspace. We plan to meet with the client in person at least once every month at the East side tech hub location whenever HEB's team plans to be at the office. We also plan to schedule meetings with HEB on a need basis based upon requirements as discussed in our internal team meetings. These meetings will be virtual as HEB's team works from home. All instant/virtual communication will be done through Slack or email. We plan to use Slack and email for all virtual/instant communication. We expect to be in contact every two business days on average.



Escalation Plan and Procedure



Our Escalation Procedure works by having categories of Points of Contact. For each person in that category, we have three modes of communication defined by urgency. For an escalation, first, we assign Jira tickets to any relevant person in the First Point of Contact Category and set a deadline on the ticket based on how urgent the escalation is. This deadline is defined by the priority level in Jira as defined by HEB. If that still needs to resolve the escalation, we contact that person over email or Slack. If we do not get a response by the end of the next business day, we contact them via phone at the contact number listed in Slack profiles. Suppose we still cannot resolve the escalation. In that case, we move to the Second Point of Contact category and reach out to someone in that category. We assign a critical ticket and reach out to them over email/slack as needed or call them if it gets really urgent. Finally, suppose that also does not resolve the escalation. In that case, we reach out to the final points of contact and follow the same steps of modes of communication.

Budget

Financial budgeting should always be at the forefront of any project's plan. For our project's budget, we have estimated usage amounts based on three scenarios:

(i) Based on Minimum Viable Product

(ii) Based on product scaled to 3 to 5 teams

(iii) Based on product scaled to 10 teams for heavy usage.

We also calculated the costs of time investment by our HEB counterparts and us at McCombs. Since our solution is entirely cloud-based, we have negligible hardware costs. We are also bringing our own devices for all work related to this project.

The labor costs for the entire project total up to $30,680, including the labor costs of both Team HEB and Team McCombs. We also wanted to provide a range of dollar amounts in the budget to HEB, depending on the usage case. The monthly cost of usage for our MVP is around $61.24; for medium use, calculated by estimating scaling up to 3 to 5 teams is around $85.65 monthly; for heavy usage, which was calculated based on scaling up to 10 teams, is around $107.22 monthly.

Product and Labor Expenses Tabulated

PRODUCT EXPENSES				
ITEM NAME	ITEM DESCRIPTION	UNITS	$ / UNIT	TOTAL
Estimate of Usage of Minimal Viable Product				
PostgressSQL GCP	Database for storing scan report data based on 730 hours of usage	1	$ 61.24	$ 61.24
Cloud Scheduler	Free for single use	1	$ -	$ -
Cloud Run on GCP	Free for basic use	1	$ -	$ -
		PRODUCT EXPENSE TOTAL (Monthly)		$ 61.24
Estimate of usage scaled to 3 to 5 teams				
PostgressSQL GCP	Database for storing scan report data based on 730 hours of usage	1	$ 61.24	$ 61.24
Cloud Scheduler	Based on estimate for 50 schedules	50	$ 0.09	$ 4.70
Cloud Run on GCP	Based on estimate for 300 requests	300	$ 0.07	$ 19.71
		PRODUCT EXPENSE TOTAL (Monthly)		$ 85.65
Estimate of usage scaled to 10 teams (Heavy usage)				
PostgressSQL GCP	Database for storing scan report data based on 730 hours of usage	1	$ 61.24	$ 61.24
Cloud Scheduler	Based on estimate for 200 schedules	200	$ 0.10	$ 19.70
Cloud Run on GCP	Based on estimate for 900 requests	900	$ 0.03	$ 26.28
		PRODUCT EXPENSE TOTAL (Monthly)		$ 107.22

LABOR EXPENSES				
TASK NAME	TASK DESCRIPTION	HOURS	$ / HOUR	TOTAL
Stakeholder's Time Investment	3 Stakeholders @ 3 hour per week average	126	$ 63.50	$ 8,000.00
Team McCombs Time investment	6 members @ 9 hour per week average	756	$ 30.00	$ 22,680.00
		LABOR EXPENSE TOTAL (Entire Project)		$ 30,680.00