Palendar

By
Cameron Stehlin, Evan Smith, Asem Alghamdi

Submitted to the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

Cameron Stehlin	4/13/20
Cameron Stehlin	Date
Evan Smith	4/13/20
Evan Smith	Date
Asem Alghamdi	4/13/20
Asem Alghamdi	Date
Abdou Fall	4/13/20
Abdou Fall, Faculty Advisor	Date

University of Cincinnati
College of Education, Criminal Justice, and
Human Services

April 2020

Table of Contents

Abstract 1
Introduction 2
Problem/Solution 3
Overview 4
Project Description / Goals 5
Objective/Deliverables 6
Methodology 7
User Profile 8
Use Case Diagram 11
Technical Elements 12
User Interface 14
Testing 20
Testing Schedule 24
Testing Report 25
Budget 30
Project Schedule 31
Problems Encountered 36
Future Recommendations 38
Conclusion 39
Bibliography 40
Appendix of Tools 41

Tables and Figures

Figure 1: User Profile 8-10

Figure 2: Use Case Diagram 11

Figure 3: Login Screenshot 14

Figure 4: Home Screenshot 15

Figure 5: Calendar Screenshot 16

Figure 6: Event Screenshot 17

Figure 7: Budget Screenshot 18

Figure 8: Friends Screenshot 19

Figure 9: Testing Schedule 24

Figure 10: UI Test Table 25

Figure 11: Mobile Test table 26

Figure 12: Fxn / Unit Test Table 27-29

Figure 13: UAT Table 29

Figure 14: Cost Chart 30

Figure 15: Work Breakdown Structure 31-34

Figure 16: Gannt Chart 35

Appendix of Tools 41

Abstract

According to a study published by Ohio State University news (Grabmeier, 2018), many college students have schedules that are all over the place. Seventy percent of college students are stressed about finances. A lot of them must budget their money as many are living on savings or working part-time. Many students also have a social life to balance as well. Planning to see others while working out a budget can be difficult. Palendar is a web-based application that allows users to create a monthly budget to help manage their money, as well as a social calendar to plan their time. There are social calendar and financial planner applications out there, but there aren't any that appeal directly to the younger demographic as well has having a budgeting tool built in. Having a web app that helps with both important areas of young people's lives allow Palendar to be a great app for many people to use.

Introduction

Our project, Palendar, came to fruition because of a problem that is personally involved in every one of our lives. That problem stems from having to budget as a college student, as well as manage a social life with all of the other stresses that come with being a student. This project was made through a collaboration of effort between Cameron Stehlin, Evan Smith and Asem Alghamdi. We do not have any plans to turn this project into a business asset and or an entrepreneurship opportunity. Our team has created a tool that all college students can take advantage of to keep their life under control.

Problem/Solution

Problem

Many college students have odd schedules due to work and class schedules. Most students have to budget their money as many are living on savings or working part-time. On top of this, many of them are proactive in their social lives. Seventy percent of college students are stressed about their finances. Planning to see others while working out a budget can be difficult too. Financial budgeting is often barely touched upon in high school and sometimes even college. There are social calendar and financial planner applications out there, but there aren't any that appeal directly to the college age demographic and that blend the two together. Our proposal is a web based application that takes both of those into account.

Solution

A web based application that allows users to create and manage a budget, plan events and share those events with friends. All of these items are tied to their budget and allows for a solution that has everything built together for a better user experience.

Overview

The remainder of this final report outlines in detail how the Palendar team completed the project and met out deliverables. The report includes the following sections:

- ➔ User Profile
- ➔ Use Case Diagram
- ➔ Objective / Deliverables
- ➔ Project Schedule
- ➔ Budget
- ➔ Technical Elements
- ➔ Testing
- ➔ Screenshots

Project Description / Goals

A web-based application that blends financial and social planning together. Our application is set apart from others as the combination of the two makes it easy to target college students. We allow users to create schedules, tasks, bill due dates, set a monthly budget, add and subtract from that monthly total budget, and share their schedules with their friends (without sharing the financial information of course). On top of this, we have some data visualization to allow the user to get a quick view of how they are doing, what they spent, and how much money they have leftover in the budget. The way that we have created the application allows for the user to get accustomed to the website quickly and have ease of use. Our project is set apart from the competitors because we have multiple tools built into our application. Most competitors that are in this space focus on one or the other while ours allows our users to get the most out of the project.

There were initial goals to port Palendar over to a fully operational application that would be available for iOS. Due to the timeframe of the project and the technical challenges that we faced, that was one of the initial goals that had to be abandoned in favor of delivering a complete web app.

Objective/Deliverables

1) Provide a solution that will allows users to utilize our application for monetary and social tracking

2) Design in a way that allows users to access their Palendar from either a computer or mobile device and allow for a seamless transition

3) Product must be running utilizing AWS

4) Allow users to compare their schedules through the calendar application. This will allow for easier scheduling where you need to invite your friends when they are not busy.

5) Allow events to be linked to the accounts budget. This allows a seamless transition that provides a good user experience. By attaching a cost to the event the total will automatically get taken out of their budget.

6) Provide data visualization to allow the users to easily look and see the amount that they are spending as well as what categories their spending is falling under

Methodology

Because of Palendar being a standard software development project with similar roles that would be used in a real life scenario, we decided to utilize the agile software development ideologies. We created a backlog of features and tasks that we needed to complete and assigned them to our group members based on their skill sets to utilize them the best we can. During the initialization phase of our project we organized basic requirements based on our group members knowledge as well as researching the best methods. We decided on the technologies that would be used and began development of the work items in the backlog. When the item was completed we marked it off of the backlog and implemented it into our master. From there we would conduct QA testing on work items that were turned in. When the entire backlog of items was completed we utilized all of our members to complete a variety of final testing. See **Figure 3** and **Figure 4** for a visual representation.

User Profile

The potential users for Palendar are mainly college students who do not have much familiarity with how to budget their money. The user can access Palendar from their laptop via web browser or from pulling up on their phone web browser. The user-interface will be simple and easy to use with well interacting pieces to provide a good user experience. See **Figure 1** below for more detail.

Project:

Social and Budget Planning Web App

Potential Users:

- ➔ College students and young adults who have a need for monetary planning help
- ➔ Heads of the family who like to plan out their social and budgeting ahead of time

Software, Interface, and Related Experience:

This web app is primarily targeted to young adults/college students who might not have a grasp of how to budget their money, as well as having a busy social schedule

This web app will be available to everyone, but it is more likely that older adults will have a better understanding of how to budget money and plan for their social life, so that demographic of users would be less than that of young adults.

No matter the age of the user and the experience in social and monetary planning, the user will find this an extremely useful tool for keeping track of things that might be forgotten.

Experience with Similar Applications:

There are other applications out there which help with budgeting, and other applications out there that help with social planning. We are putting both of those things together in trying to create a new app that can get the best of both worlds. Most young adults have probably used a calendar application for planning their social life/classes.

Older users might not have experience with similar applications because in the past there were not applications out there that provided help with budgeting. They will most likely have had experience with using a normal calendar to get their social life together.

Task Experience:

Technologically adept individuals may have experience performing tasks such as: network monitoring, asset management, and wireless configuration including port forwarding.

Frequency of Use:

Setting up this product will be easy; it will be making a simple account. Once an account is set up then the app will allow the user to input their budget and manage it. As well as getting their calendar set up as well. Once setup is complete it is up to the user to update and manage their budgets and schedules. Ideally the user will get on once per day to keep their account updated.

Key Project Design Requirements that the Profile Suggests:

- ➔ Easy learning curve and well-designed UI

➔ Make it easy to edit budget and add items

➔ Make it evident that there are two main functionalities of the web app

Figure 1: User Profile

Use Case Diagram

Below is our Use Case diagram which visually displays the user and how they will interact with Palendar, see **Figure 2**.

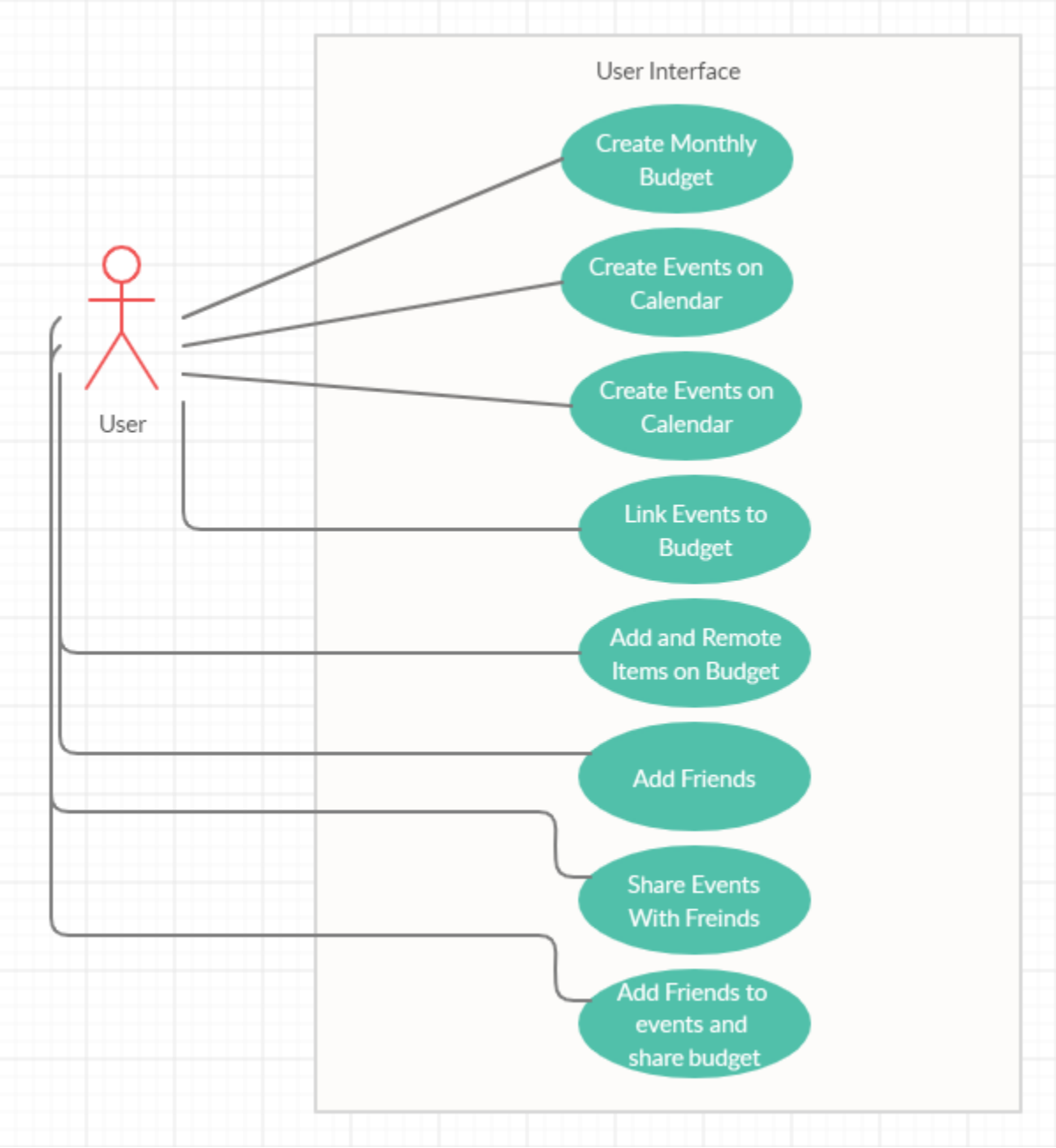

Figure 2: Use Case Diagram

Technical Elements

Network

The network consists of the virtual machines for servers that are being hosted. In this case, our front-end is hosted on S3 bucket and connected to EC2 instances which hold authentication applications and our back-end API. The Laravel application and .net application are hosted in two separated Elastic Beanstalk services which work on top of EC2 instances and they are connected to the S3 bucket using JS Vue framework inside Virtual Private Cloud (VPC). The Elasticsearch instance is hosted on a small server with only 4 gigabytes of ram and 2 cores. There is a CloudWatch logging service connected to each instance to log all traffic that is going in and out of EC2 instances.

Application

Our stack takes advantage of using various frameworks. Our front-end consists of JS Vue which is an open-source Model–view–view model JavaScript framework. We will also be utilizing the Bootstrap CSS library with our own tweak for Palendar branding. Our backend is utilizing the .NET framework which easily is incorporated into our front-end. Authentication is utilizing Laravel which is an open-source model of PHP framework.

Security and Databases

Users need to register and login in order to utilize our application. We are using Laravel framework to authenticate users who are registered in the database. All our applications and database are inside a Virtual private network which cannot be accessible via the internet. The only way to access the database is through an s3 bucket which holds our front-end application and is strictly ruled by Identity and Access Management (IAM) rules. We are two MySQL databases, the first is storing User information and passwords are salted hashed in the database and the second database is holding the back-end information.

User Interface

Login

Below is the login page where the user can enter their information or register an account, see **Figure 3**.

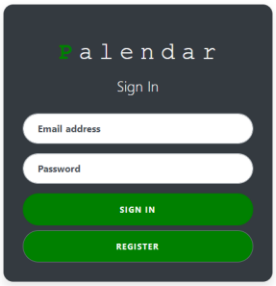

Figure 3: Login Page Screenshot

Home

Below is the home page which greets users when they first log into Palendar. Here they are provided an overview of all of their information, see **Figure 4**.

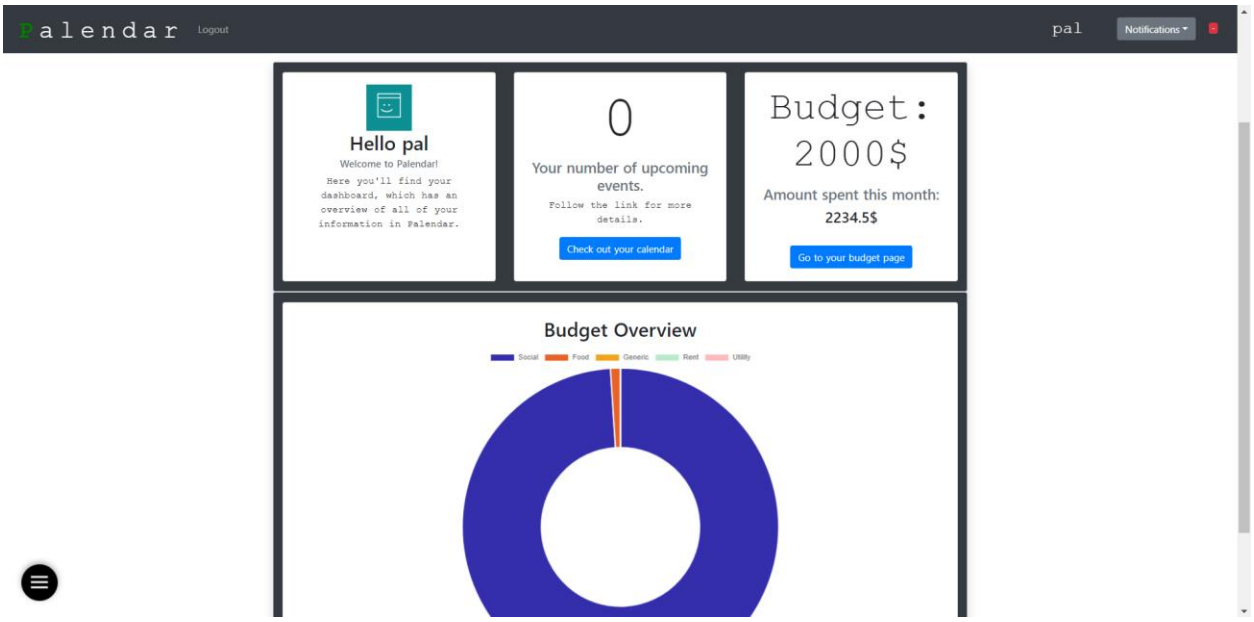

Figure 4: Home Page Screenshot

Calendar

Below is the calendar page where users can utilize the Palendar calendar feature and create events linked to their budget, see **Figure 5**.

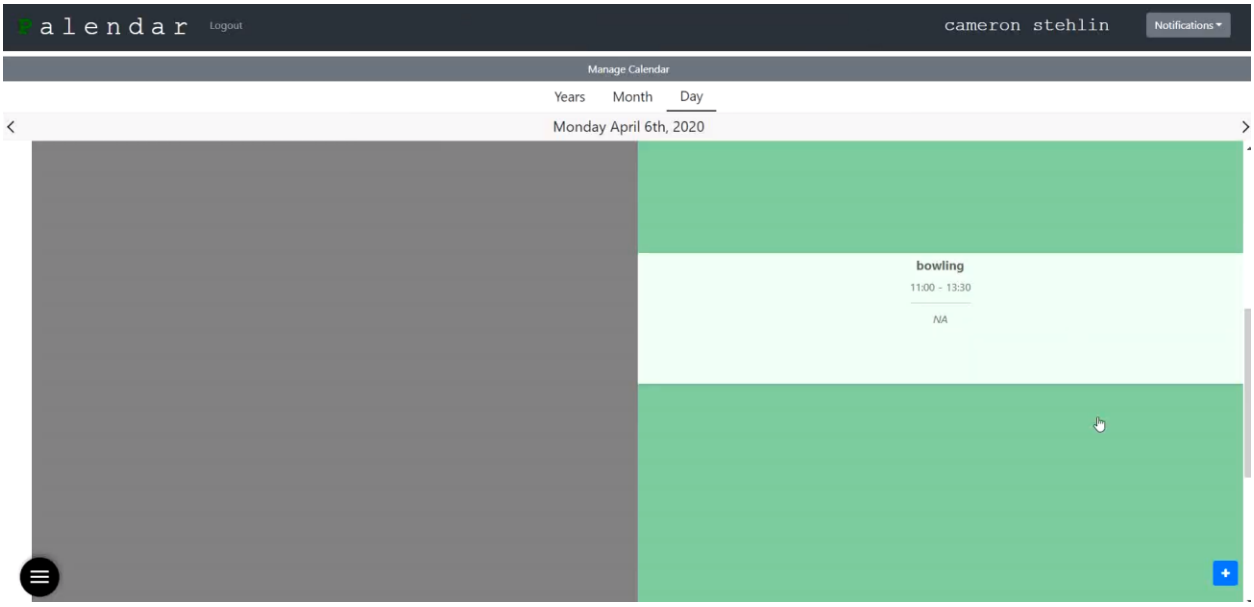

Figure 5: Calendar Page Screenshot

Event Creation

Below is the event creation screen which the user can input the information for their event, see **Figure 6**.

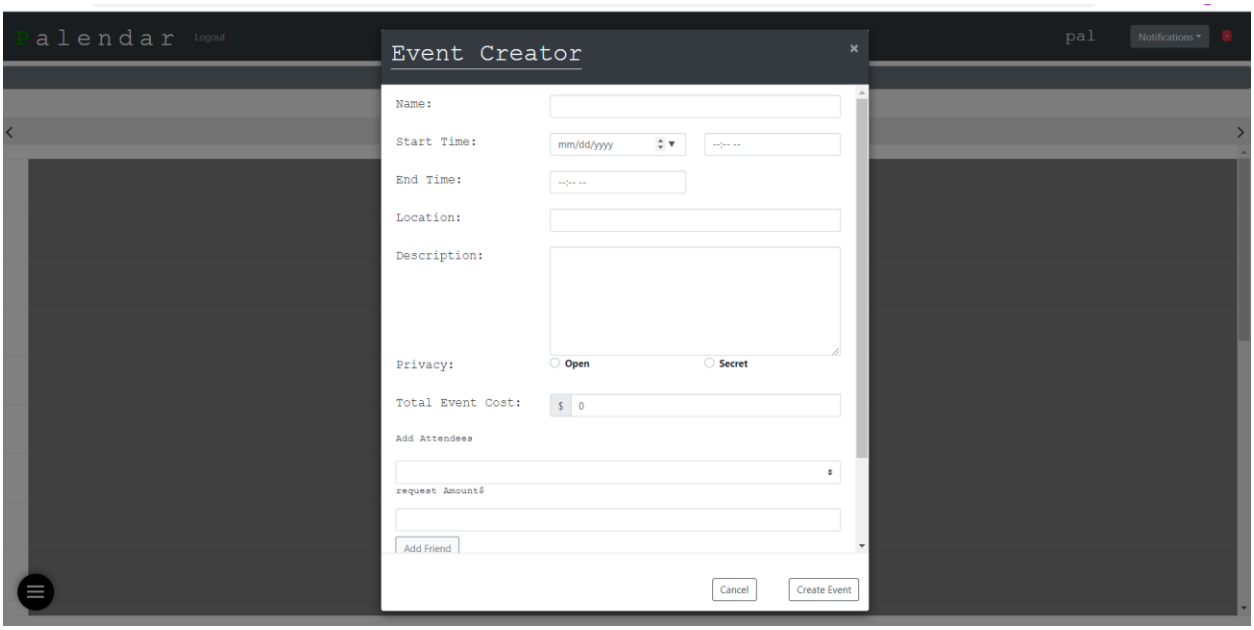

Figure 6: Event Creation Page Screenshot

Budget

Below is the budget page where the users can manage their budget, see **Figure 7**.

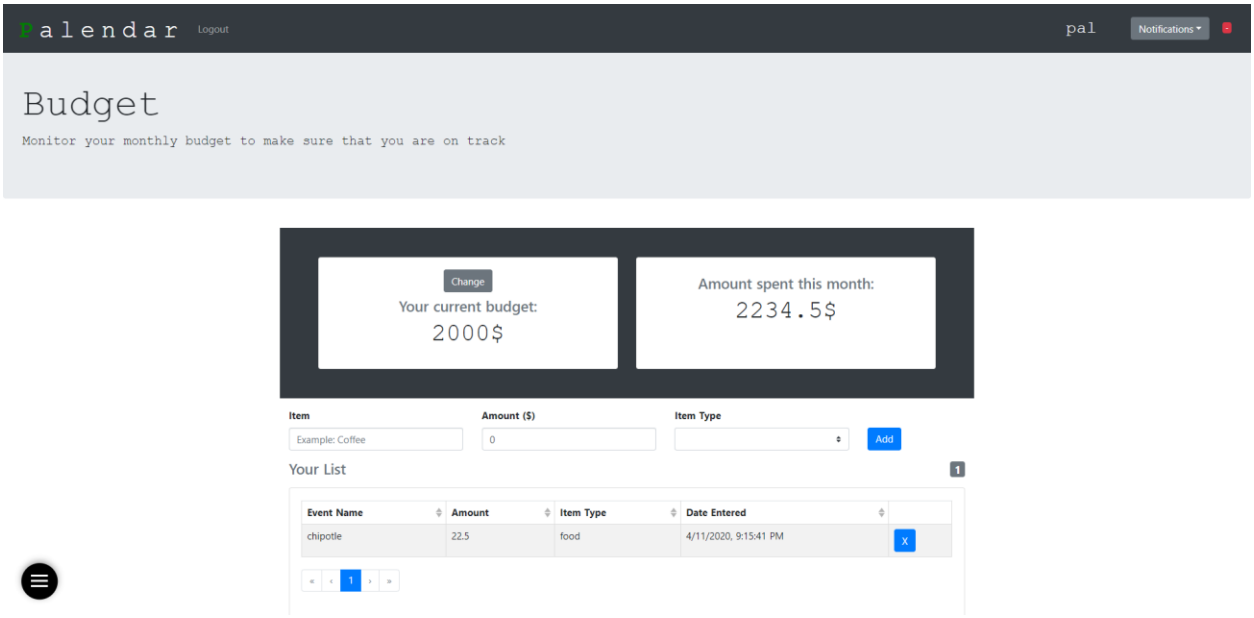

Figure 7: Budget Page Screenshot

Friends

Below is the friends page where users can manage their friends list, see **Figure 8**.

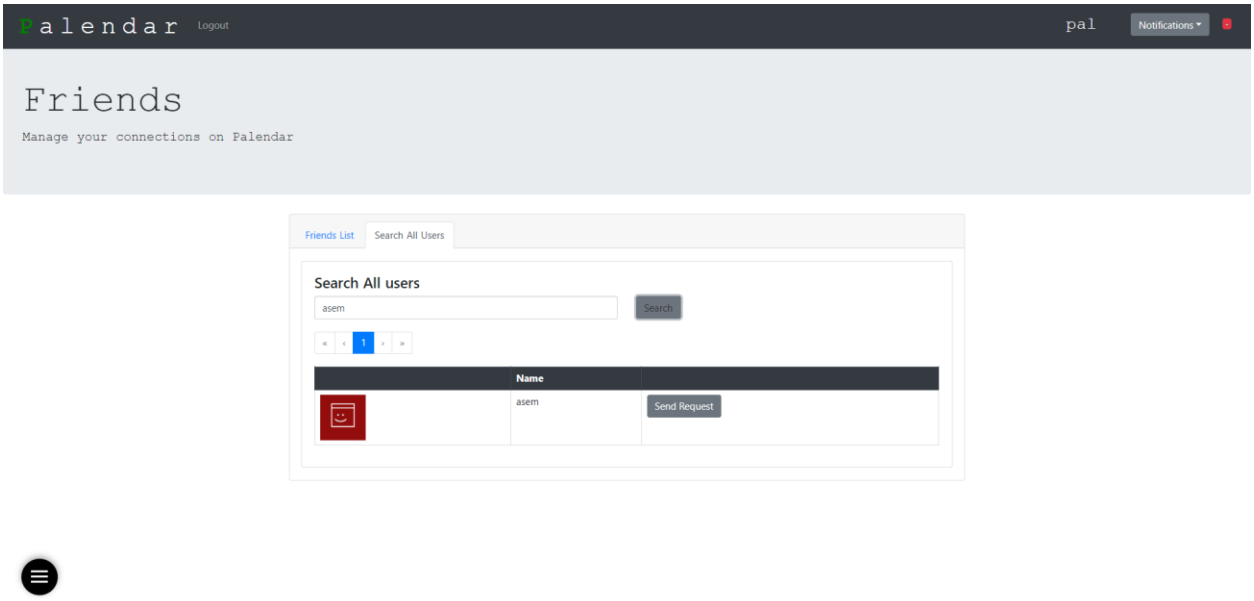

Figure 8: Friends Page Screenshot

Testing

Overview

This section will explain the testing for Palendar. This is a guide that the members of Palendar followed to insure the desired outcome was met. Every team member took part in testing and followed these standards.

Scope

The scope of the testing for Palendar was to ensure that the application's methods were working properly without any errors to make sure that the users information would not be misinterpreted. We also prioritized testing the mobile responsiveness of our site because we valued the ability for our users to have an easy transition to their phone if they wish. The tests that were made have been based on the requirements/deliverables of the project.

Objective

The objective of the testing we are undergoing is to verify that each feature that we have put into Palendar is working correctly. Every member will run different forms of testing.

Entry Criteria

- ➔ Complete Build
- ➔ Test environment is set up

Exit Criteria

➔ All tests have passed

➔ Any bugs found are fixed

Bug Fixes

If a bug is found in our code during any testing, the tester will stop and fix the bug to make sure the feature is working before returning to testing. The order of bug fixing will be decided upon severity analysis.

System Testing

Palendar will be tested end-to-end to ensure the complete and fully integrated application is working correctly. While testing, all interactable components will be clicked to ensure that even uncommonly used elements are working with the system properly. A different set of conditions in events and their users will be testing to make sure things are interacting properly. All of these tests will help Palendar become a better application.

Security Testing

➔ ✔ Laravel framework defends against SQL injection.

➔ ✔ Passwords are encrypted in the database.

➔ ✔ PHP scripts run successfully and they don’t have syntax errors.

➔ ✔ Users are only authorized to access their accounts.

➔ ✘ Users received verification emails for their account.

➔ ✔ The password should not appear while typing it.

➔ ✔ PHP script posted users successfully in the database.

➔ ✔ Unauthorized users can’t log in to the system

➔ ✔ The databases are secured and running successfully.

➔ ✔ Validation is applied when user register for their:

- Name
- Email
- Password
- Password Confirmation.

➔ ✔ APIs are inside the VPC and it is protected by Amazon IAM rules to don’t let unauthorized users access them.

Testing Procedures

➔ Test Palendar with realistic scenarios that users would often encounter

➔ Fix any problems that arise during the testing period

➔ Document tests and test results

Types of Tests performed

➔ User Interface Test

- Clicking buttons and entering input into fields to ensure that the interface is working as expected
- Will be completed during system testing when each page is loaded

➔ Mobile Test

- Tested on mobile to ensure that the user experience is up to Palendar standards

➔ Unit Testing

- Individual components of code will be testing using XUnit to ensure it is working as designed

➔ User Acceptance Test

- Ensure that our solution to the problem statement is meeting the standards of the users

Pass/Fail Conditions

If there is a test that has not met the exit criteria then one of the members of the Palendar team will be deployed to fix the issue.

Testing Schedule

Below are the types of testing that Palendar underwent as well as the dates which it took place, see **Figure 9**.

Test Name	Start Date	End Date
UI Testing	3/1/20	3/8/20
Mobile Testing	3/20/20	3/27/20
Unit Testing	3/24/20	3/31/20
UA Testing	4/1/20	4/5/20
Security Testing	4/1/20	4/5/20

Figure 9: Testing Schedule

Testing Report

UI Testing

➔ A bug will only be reported if the test has failed

➔ A test will fail only if the problem has not been fixed

See **Figure 10** below

Page Name	Browser	Expected	Actual	Pass / Fail	Bug
Home	Chrome	Works	Works	Pass	None
Budget	Chrome	Works	Works	Pass	None
Calendar	Chrome	Works	Works	Pass	None
Friends	Chrome	Works	Works	Pass	None
Home	Firefox	Works	Works	Pass	None
Budget	Firefox	Works	Works	Pass	None
Calendar	Firefox	Works	Works	Pass	None
Friends	Firefox	Works	Works	Pass	None
Home	Edge	Works	Works	Pass	None
Budget	Edge	Works	Works	Pass	None
Calendar	Edge	Works	Works	Pass	None
Friends	Edge	Works	Works	Pass	None

Figure 10: UI Test Table

Mobile Testing

➔ A bug will only be reported if the test has failed

➔ A test will fail only if the problem has not been fixed

See **Figure 11** below

Page Name	Device	Expected	Actual	Pass / Fail	Bug
Home	iPhone X	Works	Works	Pass	None
Budget	iPhone X	Works	Works	Pass	None
Calendar	iPhone X	Works	Works	Pass	None
Friends	iPhone X	Works	Works	Pass	None
Home	Galaxy S5	Works	Works	Pass	None
Budget	Galaxy S5	Works	Works	Pass	None
Calendar	Galaxy S5	Works	Works	Pass	None
Friends	Galaxy S5	Works	Works	Pass	None
Home	iPad Pro	Works	Works	Pass	None
Budget	iPad Pro	Works	Works	Pass	None
Calendar	iPad Pro	Works	Works	Pass	None
Friends	iPad Pro	Works	Works	Pass	None

Figure 11: Mobile Test Table

Unit / Functionality Testing

➔ A bug will only be reported if the test has failed

➔ A test will fail only if the problem has not been fixed

See **Figure 12** below

Test Case Name	Desired Result	Result	Bug
Was able to navigate all pages	Navigate all the pages without any errors	Pass	None
On start get event request notifications	Get list of event notifications that will appear in the global navigation	Pass	None
Delete event request	Deletes an event request from the request list	Pass	None
On start get friend request notifications	Get list of event request notifications that will appear in the global navigation	Pass	None
Get friends list	Get friends list in the event creator section	Pass	None
Delete friend request	Request is deleted	Pass	None
Delete friend	Friend gets	Pass	None

	deleted		
Create event	Event created and added to user's calendar	Pass	None
Get events in Calendar	On start of the calendar page will load all of the events into the calendar	Pass	None
Edit events	Users will be able to edit event	Pass	None
Delete events	The user will be able to delete events from their calendar	Pass	None
Validate that users can't put harmful data in the event creator/editor	User will not be able to put in too many characters or illegal ones like text in integer only input boxes	Pass	None
Send friend request	User will be able to send request to any other user	Pass	None
Accept friend request	User can accept friend request	Pass	None
Send event request	send event requests on creation	Pass	None
Set a budget	Set a budget in the budget page	Pass	None

Create a spending event	Create an event in the budget page	Pass	None
Delete a spending event	User will be able to delete any of their spending events	Pass	None
User receives notification email	An hour before the event starts the user will receive an event notification in their email of the event.	Pass	None

Figure 12: Functionality / Unit Test Table

User Acceptance Testing

Users were screened for their satisfaction, See **Figure 13** below.

User Name	OS	Satisfaction	Experience
Matt	iOS / Safari	Useful Tool	Pass
Marie	Windows	Easy to understand	Pass
Erin	Windows	Would help save money	Pass

Figure 13: User Acceptance Test Table

Budget

The budget for this project has a similar format to that which any other software development team would. Most of this cost is coming directly from having to hire and employ a development team, the actual cost of hosting and deploying the site is minor in comparison. In the below example we have assumed the hourly rate of 3 employees that would be working full time for a month, given that with the full time efforts of 3 employees the time from POC to Launch would be anywhere between 1-3 months. So we have calculated the costs of 1 month assuming that this is a company project which is being funded. See **Figure 14** below for more detail.

Cost Chart	
Hosting / Renting Costs	
AWS	$38.28 per month
Domain Name	$39.00 per year
Total Costs	$77.28 per month
Labor Costs	
Development - Labor	$120 per hour
Design - Labor	$41.00 per hour
Testing - Labor	$32.00 per hour
Total Costs	$223.00 per hour
Overall Total	$107,117.28

Figure 14: Cost Chart

Project Schedule

Below is our Work Breakdown Structure outlining the broken down development work and duration. See **Figure 15**.

Task Name	Duration (Days)	Start Date	End Date
1.0 Project Management and Deliverables	252	8/26/2019	4/14/2020
1.1 Team Creation	7	8/26/2019	9/2/2019
1.2 Brainstorming	7	9/2/2019	9/9/2019
1.3 Contract	14	9/9/2019	9/23/2019
1.3.1 First Draft	13	9/10/2019	9/23/2019
1.3.2 Project Logo and Branding	7	9/23/2019	9/30/2019
1.4 Project Abstract	21	9/23/2019	10/13/2019
1.5 Contract Resubmission	14	9/23/2019	10/13/2019
1.6 User Profile	7	10/14/2019	10/21/2019
1.7 Use Case	7	10/14/2019	10/21/2019
1.8 Draft Report	14	10/22/2019	11/04/2019
1.9 Fall Final Report	22	11/10/2019	12/02/2019
1.10 Fall Presentation	14	11/29/2019	12/13/2019
2.0 Research	56	09/10/2019	10/21/2019
2.1 List of technology is being used	7	09/10/2019	09/17/2019
2.2 Research AWS	49	9/26/2019	10/13/2019

2.2.1 How Route 53 works	7	9/26/2019	10/02/2019
2.2.2 How Amazon CloudFront works	7	10/03/2019	10/09/2019
2.2.3 How Elastic Load Load Balancer Works	7	10/10/2019	10/16/2019
2.2.4 How Elastic Computing 2 works	7	10/17/2019	10/23/2019
2.2.5 How Amazon RDS works	7	10/24/2019	10/30/2019
2.2.6 How S3 Bucket works	7	10/31/2019	11/06/2019
2.2.7 How Identity and Access Management works	7	11/07/2019	10/13/2019
2.3 Create a diagram of the code	15	09/25/2019	10/09/2019
2.4 Create an initial prototype	12	10/10/2019	10/21/2019
3.0 System Design	17	09/25/2019	10/11/2019
3.1 Wireframe creation	11	09/25/2019	10/05/2019
3.2 User Interface Design	6	10/05/2019	10/11/2019
4.0 Environment	57	10/14/2019	12/02/2019
4.1 Set up AWS	57	10/14/2019	12/02/2019
4.1.1 Set up Route 53	7	10/14/2019	10/20/2019
4.1.2 Set up Amazon CloudFront	5	10/21/2019	10/26/2019
4.1.3 Set up Web Servers	5	10/27/2019	11/01/2019
4.1.4 Set up App	5	11/02/2019	11/07/2019

4.1.5 Set up Database	5	11/08/2019	11/12/2019
4.1.6 Set up the S3 bucket	5	11/13/2019	11/18/2019
4.2.1 Setup JS Vue	10	11/08/2019	11/18/2019
4.2.2 Set up C# web API	15	11/18/2019	12/02/2019
5.0 Development	70	01/13/2020	04/02/2020
5.1 Create a Budget page	4	01/13/2020	01/17/2020
5.2.1 Create Calendar	4	01/18/2020	01/22/2020
5.2.2 Add feature to add events and bills	4	01/23/2020	01/27/2020
5.3.1 Create notifications service	4	01/28/2020	02/31/2020
5.3.2 monthly update alert	4	02/01/2020	02/05/2020
5.3.3 Before alert	4	02/06/2020	02/10/2020
5.3.4 exceed monthly budget	4	02/11/2020	02/15/2020
5.4.1 Create friends list feature	4	02/16/2020	02/19/2020
5.4.2 add and delete friends feature	4	02/20/2020	02/25/2020
5.4.3 search friends	4	02/26/2020	02/29/2020
5.5 view friend’s calendar	4	03/01/2020	03/05/2020
5.6 mobile responsiveness	4	03/06/2020	03/10/2020
5.7.1 User profile	4	03/11/2020	03/15/2020
5.7.2 create acct	4	03/24/2020	03/28/2020

5.7.3 login	4	03/29/2020	04/02/2020
6.0 Testing	30	3/1/2020	4/1/2020
6.1 Set up Mock Users	14	3/1/2020	3/14/2020
6.2 Unit Testing	30	3/1/2020	3/30/2020
6.3 Mobile Test	30	3/1/2020	3/30/2020
6.4 User Acceptance Test	30	3/1/2020	3/30/2020

Figure 15: Work Breakdown Structure

Below is our Gantt Chart, seen in **Figure 16**, which outlines the timeline of Palendar and the tasks which were completed.

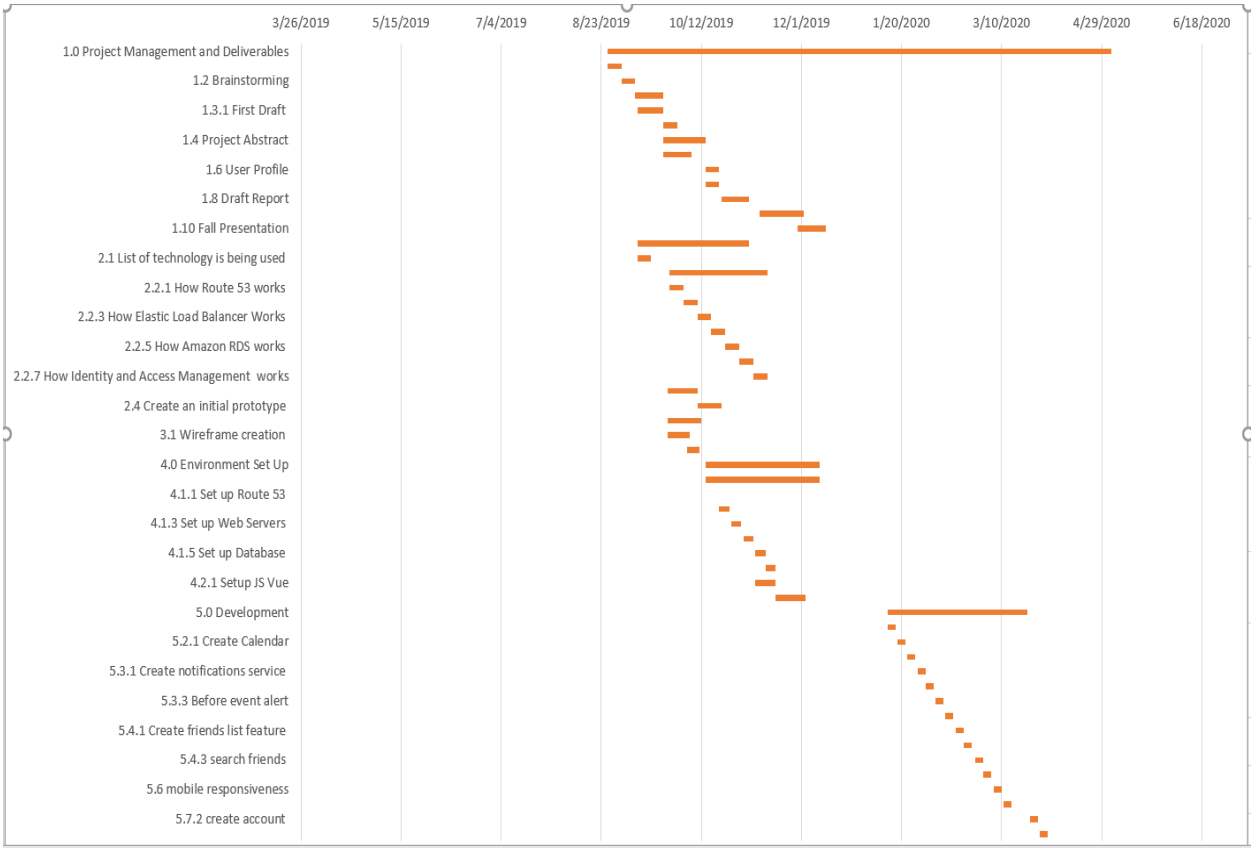

Figure 16: Gantt Chart

Problems Encountered

Issue 1: Integrating Chart.js into Palendar

A hurdle that happened in the front-end of this project occured when we tried to implement the doughnut chart seen on the home page by utilizing the chart.js library. This was caused by a conflict with the vue beautifier library called vuetify. Because of these conflicts we had to evaluate the benefit and losses of moving things over to allow us to use chart.js. Eventually we went with chart.js and removed vuetify from our project.

Issue 2: Authentication

We ran into an issue when integrating an authentication application with our front-end application. We developed php applications for authentication, but it could not be integrated with Vuejs because they are not compatible. So we did our research and found that Laravel Framework is compatible with Vuejs.

Issue 3: Deployment

We ran into a fairly big issue when we tried to deploy our application. Because we are using Vuejs, it is a server-client based application so it was hard to deploy it as a static application. We first started with Enom to deploy our application but it did not work because Enom did not give us access to the server, so we moved to host it locally on one of the group computers but another hurdle popped up which we could not assign a public IP address to the computer. So we decided to move back to AWS. When we deployed our application in AWS, we had to figure out the

networking part and where we should start to set up the environment. We also had trouble deploying our back-end API because it keeps returning “Error 500: internal server error” so we spent three days, we reached a point where we should move to a different solution than AWS but we tried one last solution which we thought would not work but it got the application working.

Future Recommendations

The Palendar team spent a lot of time familiarizing themselves with the tech stack that we decided to use, if done again it would be beneficial to mesh the strengths of every person in the project to create an environment where everyone can flourish. But this allowed us to learn a great deal while working, so there are benefits to both sides.

If there was more time for this project there are quite a few features that we had in mind that we did not have the time to develop. We would have liked to dip into application development that would go onto the App Store / Google Play Store so that our mobile users would not have to access the app through their web browser.

A final recommendation is to have better communication earlier on in the project. Our communication really picked up and worked well towards the end of the project period. If we had been able to do that during the entire working period a lot of things would have operated smoother.

Conclusion

Fall 2019

In the Fall Semester of 2019, the Palendar team completed a lot of research in which technology stack to use due to our varying degrees of experience, we eventually settled on JS Vue and .NET. A lot of issues arose on if the team wanted to utilize AWS or not, we decided to not go forward with it even though we came back full circle in the spring. Work items that were completed include a basic outline of our user interface, a good chunk of the back end allowing the calendar to work. Next semester we plan on finalizing the user interface as well as integrating the back end and implementing authorization.

Spring 2020

Over the two semesters that the Palendar team has worked on this project we have learned a lot about full stack development teamwork as well as more specific things about the technologies that we utilized to complete our project. We created an extremely useful tool that would allow for our audience to get started on saving money and being more aware of what they are spending and how they are spending their money and time. There are only a few direct competitors out there and Palendar is on par with them. We set some pretty ambitious goals with our application and are proud of the accomplishments we achieved during the course of our project.

Bibliography

Grabmeier, Jeff. “70 Percent of College Students Stressed about Finances.” *70 Percent of College Students Stressed about Finances*, The Ohio State University, 12 July 2018, news.osu.edu/70-percent-of-college-students-stressed-about-finances/.

“Chart.js.” *Chart.js | Open Source HTML5 Charts for Your Website*, www.chartjs.org/.

“Vue.js.” *Vue.js*, vuejs.org/.

“Where Developers Learn, Share, & Build Careers.” *Stack Overflow*, stackoverflow.com/.

Appendix of Tools

Vue JS - Model–View–Viewmodel JavaScript framework

.NET - Software framework

Bootstrap - CSS Library

Visual Studio - Integrated Development Environment by Microsoft

Visual Studio Code - Lightweight Source Code Editor used for Front-End

C# - Object Oriented Programming Language

Tech Expo Poster –