PaySmith

Product Documentation

Diego Cruz

Hung Nguyen

Michael Piccerillo

Bolang Yu

Table of Contents

Table of Contents **1**

Software Requirements Specification **2**

Abstract 2

Functional Requirements 2

High Level Requirements 2

Detailed Requirements 3

Traceability Matrix 4

Design 4

Testability 5

Organization and Implementation 7

User Stories 8

Table 1 - User Story 1: Crowdfund Management 8

Table 2 - User Story 2: Expense Management 8

Table 3 - User Story 3: Administrator Privileges for Devs 9

Nonfunctional Requirements 9

Security 9

Table 4 - Security Requirements 9

Performance 10

Model View Controller 11

Software Design **12**

Class Diagram 12

Data Flow Diagram 13

State Transition Diagram 14

Activity Diagram 15

Test Cases **16**

List of Libraries and Open Source Packages Used **19**

Installing Libraries and Open Source Packages **20**

XAMPP PHP Dev Environment 20

Setting up Paysmith after installation 23

Conclusion **28**

Software Requirements Specification

Abstract

PaySmith is meant to be the most convenient platform for users to manage their money. In practice, this product will allow users to track crowdfund money for experiences or causes that they align with, and will let users manage their financial health. Users can manage their financial health by allowing them to track their expense history, compare expenses between months or other defined periods of time, and view their transaction history.

Once the team implements how it wants PaySmith to function, the next focus will be addressing the speed and responsiveness of the project. The developer team intends for users to be able to login and see all of their information within a matter of seconds. Being able to fund money to crowdfunds will, hopefully, be just as fast as logging in.

Finally, the developer team wishes to make the user interface and design appealing to all visually, as well as intuitively. If time permits, the team would love to design this project in such a way that it's easy and appealing to use.

Functional Requirements

High Level Requirements

Upon loading the website, the user will first see a splash page which will prompt the user to either login or create a new account. Once that action is completed, the user will see a menu page where they will be given options to add expense, pitch in to a crowdfunded money pool, manage their expenses, or edit user information. Once a selection is made to what they would like to do, the user will then be redirected to a subpage or integrated sub window corresponding to the chosen action that they clicked on. The user can then proceed with what they would like to do.

Once the user is finished with what they were doing, or decides to go navigate away, they will be able to go back to the homepage by clicking on a home/dashboard icon to take them back or click on an area away from the sub window. The target audience includes those who want to better understand how they spend their money through the financial management page.

Detailed Requirements

(1) Registration/ Log in: There will be the option to either register as a new user or log in as a returning user. On the registration page, the user must enter their email address, last name, first name, email, and password. This information will be saved in the customer database. User inputs, such as email, will be checked in the database to avoid duplicated users. On the login page, the user will be prompted to enter their email and password, and that information would be checked in the database to see if it already exists. Web app administrators (admins) will be able to monitor and manage accounts from the customer and transaction database.

(2) Crowdfunding Money: The users and admins are able to create crowdfunds with information stored in the Crowdfund database. This information consists of the crowdfund name, and the total amount raised. The admins can edit crowdfunds in the database.

(3) Manage finances: users can see and manage an expendable budget for each certain period and the data will be stored in the transaction database. Additionally, expense history data will be stored in the transaction database, and is retrieved when users want to track and compare

monthly spending habits. Users and admins can monitor the user's transaction as well as crowdfund.

Traceability Matrix

User Story	Detailed Requirement	Design Requirement	Test Requirement	Organization/ Implementation Requirement
1 Crowdfund	2	2	Function Test 1, 2 ,3	3
2 Expense Management	4	2	Function Test 1, 2,	1, 2, 3
3 Admin Privileges	2,3	1	Function Test 1, 2 System Test 2, 3,	2, 3

Design

In terms of design, the development team will have two possible methods of approach depending on whether they consider the guided user interface(GUI)/guided user experience (GUX) to be a desirable objective or an optional objective.

(1) If the development team considers the design to be a desirable objective, then the GUI and the GUX will be made in a custom fashion that is unique to the web application. This means that the color scheme and the layout will be made from scratch using HTML 5. This also means that the connections between the interactive objects between the front end will have to be connected to the back end that involves databases. While making a custom front end to the web application will inevitably result in more lines of code and time/resources that could be spent elsewhere, it will have a more original look/feel which would appeal to the end user. If this

course of action were to be taken, the developers would focus on creating a simple, easy-to-follow GUI that will minimize the amount of tabs open at first glance. This would not be a reduction in functionality, because more tabs can be opened and the workflow can become as complicated as the user desires. Additionally, the main color theme of the web app would be dark colors composing the background with objects that contrast, but not too bright, to minimize eye strain.

(2) If the development team considers the design to be an optional objective, then the GUI and GUX could be shaped from an already existing open-source HTML 5 template for a website in which there already exists some documentation for connecting front-end interactive objects with the back end and the databases inside of it. This frees up time and resources for the developers to work on implementing more functionality with regard to the back end of the web application. For instance, more time and resources could be allocated to giving further development to the statistical analysis that could be given to the end user, based on trends of how certain fiscal data changes over time. The open-source templates that are available on sites like Github, and Themify, offer much documentation for their themes that allow for a good degree of flexibility when implementing the front end for creating a derivative of that work.

Testability

Process of testing:

1. Each function should be tested and debugged.
2. Run all files together and find files that have errors to debug.
3. After debug, check every file individually, then all files together again.
4. If there are defects, find the files that have errors and correct code as needed.
5. For the final step, run every file and check the website if there is no error.

Functional testing:

1) Main page
   - Goal: The page which gives the overall information about the website and the company.
   - Expectation: Users can easily see and access information about the website and the company.
2) Login page
   - Goal: The page in which users can log in or register.
   - Expectation: Users can register or log in without any error. Ex: duplicate user, null input.
3) Crowdfund page
   - Goal: The page where users can create, see and fund the crowdfunds.
   - Expectation: This page has to be user-friendly and bug-free.
4) Finance managing page:
   - Goal: the page where users can edit, remove transactions and compare spending between each month.
   - Expectation: All functions in this page must work with the database without any error.

Component testing:

All the webpages must connect to the database and the information on the webpages have to be retrieved from the database correctly.

Test metrics including success criteria; tools and methodology to be used:

- Success criteria requirement:

a. The software must satisfy all the quality/ functionality requirements.

b. The website's functions must be finished within the time-frame.

Method of testing:

Unit test: test program in isolation.

- Execute all the code in a unit at least once.
- Individual code units are tested by programmers as they are developed.

Feature testing: code units are integrated to create features.

- Test all aspects of features, such as interaction, usefulness.
- Programmers who contribute code units should be involved in this testing

System testing: code units are integrated to create a working version of a system.

- Check to make sure no unexpected interactions betweens the features in the system.
- Check if the system features work together effectively.
- Test to make sure that the system operates normally in different environments, such as Chrome, Firefox, Safari.
- Checking the responsiveness, reliability, and security in the system.

Release testing: The application is packed to release and the operation as expected.

- Check that the system is working properly before release.
- Check that no bugs exist in the system.

Organization and Implementation

1. The software utilized are mainly integrated development environments like Atom and PHPStorm, which are able to code languages like HTML, Java, PHP or JavaScript.
2. The database language used to store user information will be MySQL.

3. The publishing and synchronizing of codes will happen on the platform of Github, with some use of XAMPP.

User Stories

Table 1 - User Story 1: Crowdfund Management

Role	**Transaction**	**Reason**
User	Start a Crowdfund	To allow multiple users to pool money into a shared experience or some other purpose in mind.
User	View History of a Crowdfund	Allows users to see what contributions they have made to a crowdfund. This makes it easier to compensate someone for their contribution under certain uses of the crowdfund function.

Table 2 - User Story 2: Expense Management

Role	**Transaction**	**Reason**
User	Add a payment	Add to expense history.
User	Remove an expense	Users can remove false expense input.
User	See expense history	Compare expenses between months.
User	Remove an asset/debt balance	Able to remove falsely entered balances and clear balances that have been fully paid off.

Table 3 - User Story 3: Administrator Privileges for Devs

Role	Transaction	Reason
Admin	Add/Remove User	To manage the active users in the database and securely delete sensitive data should a user choose to delete their account from the web application.
Admin	Edit User Details	To update user information should the user choose to add/remove financial information linked to their account. This may include information like account/routing numbers, credit/debit card info, and asset values.

Nonfunctional Requirements

Security

With regard to security, there are a few functions deemed essential, a few deemed desirable, and one is designated as optional. Below is a table of which requirements are designated with their respective priority levels.

Table 4 - Security Requirements

Essential	● Login Procedure ○ Store login details in database ● A security measure for concealing financially sensitive information ○ Data Encryption is possible,

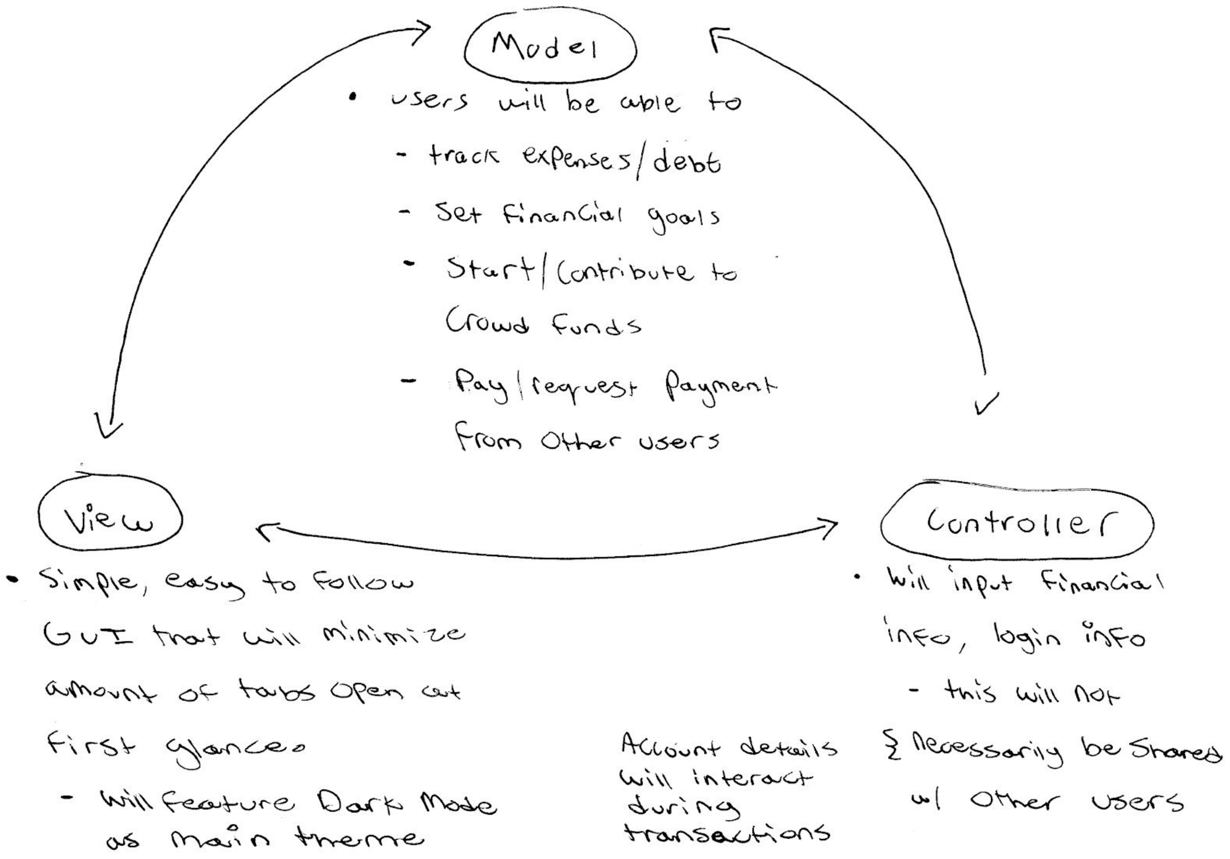
	but may be difficult to implement with current resources, desirable. ◦ Security work can also be offloaded onto server hosting services like Firebase.
Desirable	• Data Encryption for financially sensitive information ◦ Account/Routing Numbers ◦ Card Numbers/CVV sequence
Optional	• Two Factor Authentication for Login Procedure

Performance

With regard to the performance, there will inevitably be a degree of complexity that will impact both computational time and the amount of non-volatile memory used in the server. This is because the constant updating of information attached to a user would affect the performance of the system in a way that would impact loading times for the web app. Additionally, Adding on information would have to result in allocation of memory based on the needs of the user, which may upscale past the limitations of what the dev team has to work with with regards to server storage. The dev team will likely have to impose limitations on the amount of information that a user can input to attach to their profile for the sake of the beta being able to fully function by the end of the development period. With the involvement of basic statistics, the dev team will have to implement the back end in such a way that minimizes exponential complexities and instead maximize the amount of operations with a logarithmic complexity or some smaller complexity.

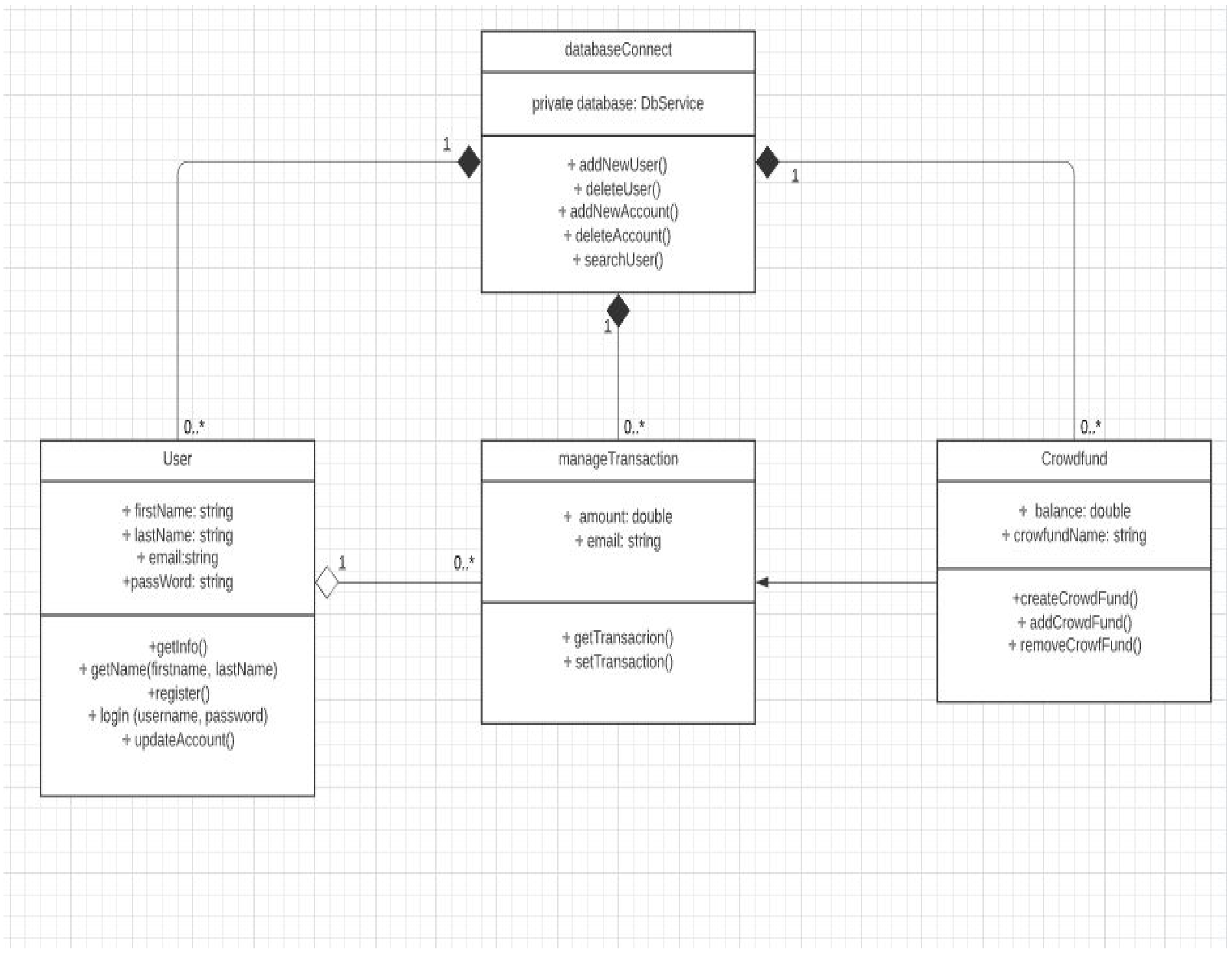
Another aspect of performance will be the general user experience with the web application. The main requirement, that will likely have to be achieved by a good execution of the design and testing requirements, involves ensuring that users cannot stress the system in such a way that it would overload the system and affect the user experience of others. This will likely have an impact on the amount of user customization that will be available for the user to tailor their experience with managing expenses, tracking fiscal trends, and making transactions with users or crowdfunds. If the developer team deems that user customization is not as important as stability, it will likely result in less customization (and the use of less statistics overall) but a more stable platform. This will minimize the growth of time and memory complexities to a manageable level for the server back end.

Model View Controller

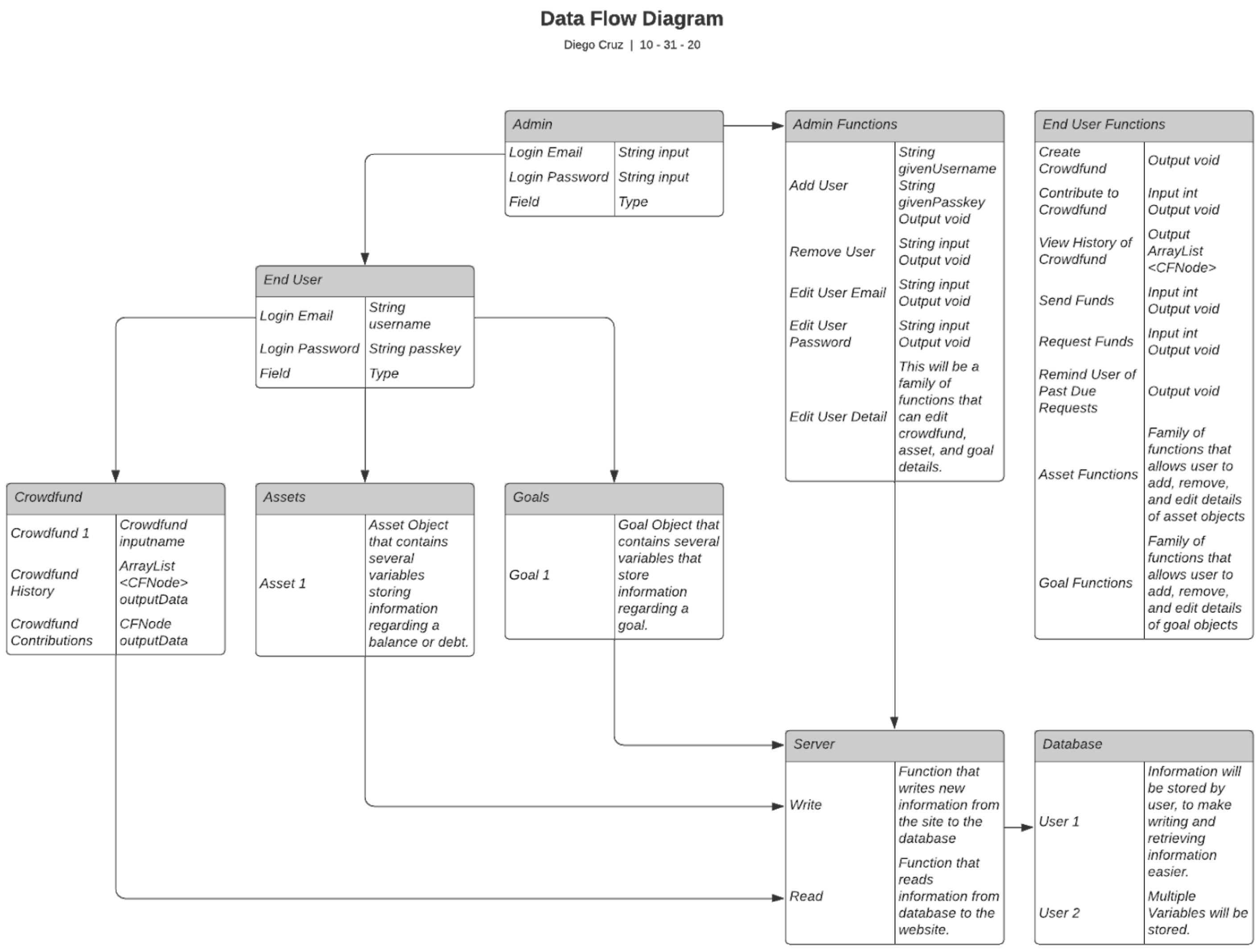

Model
• users will be able to
- track expenses/debt
- Set Financial goals
- Start/Contribute to Crowd Funds
- Pay/request Payment from Other users
View
• Simple, easy to follow GUI that will minimize amount of tabs open at first glance.
- Will feature Dark Mode as main theme
Controller
• Will input financial info, login info
- this will not necessarily be shared w/ Other users
Account details will interact during transactions

Software Design

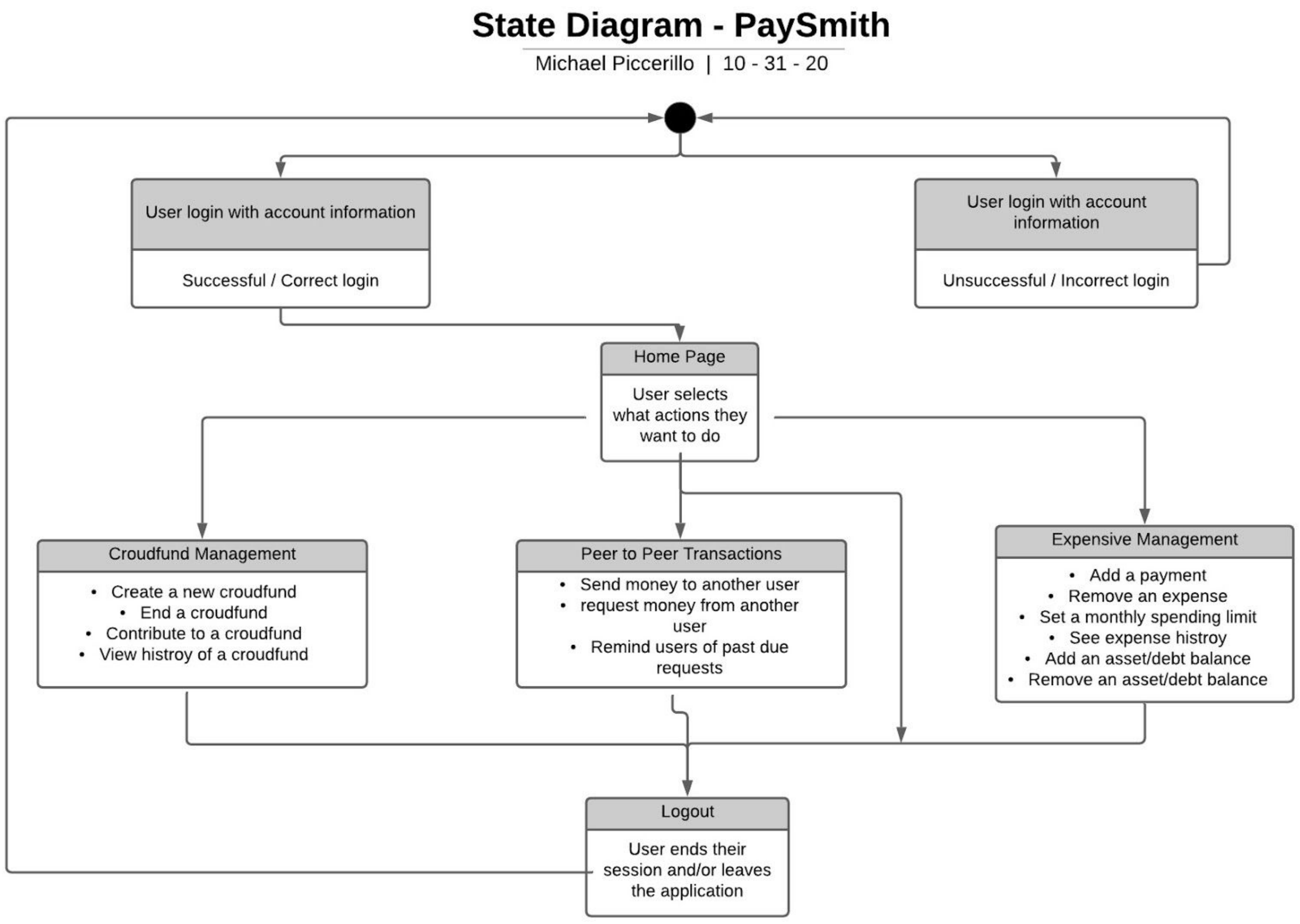
Class Diagram



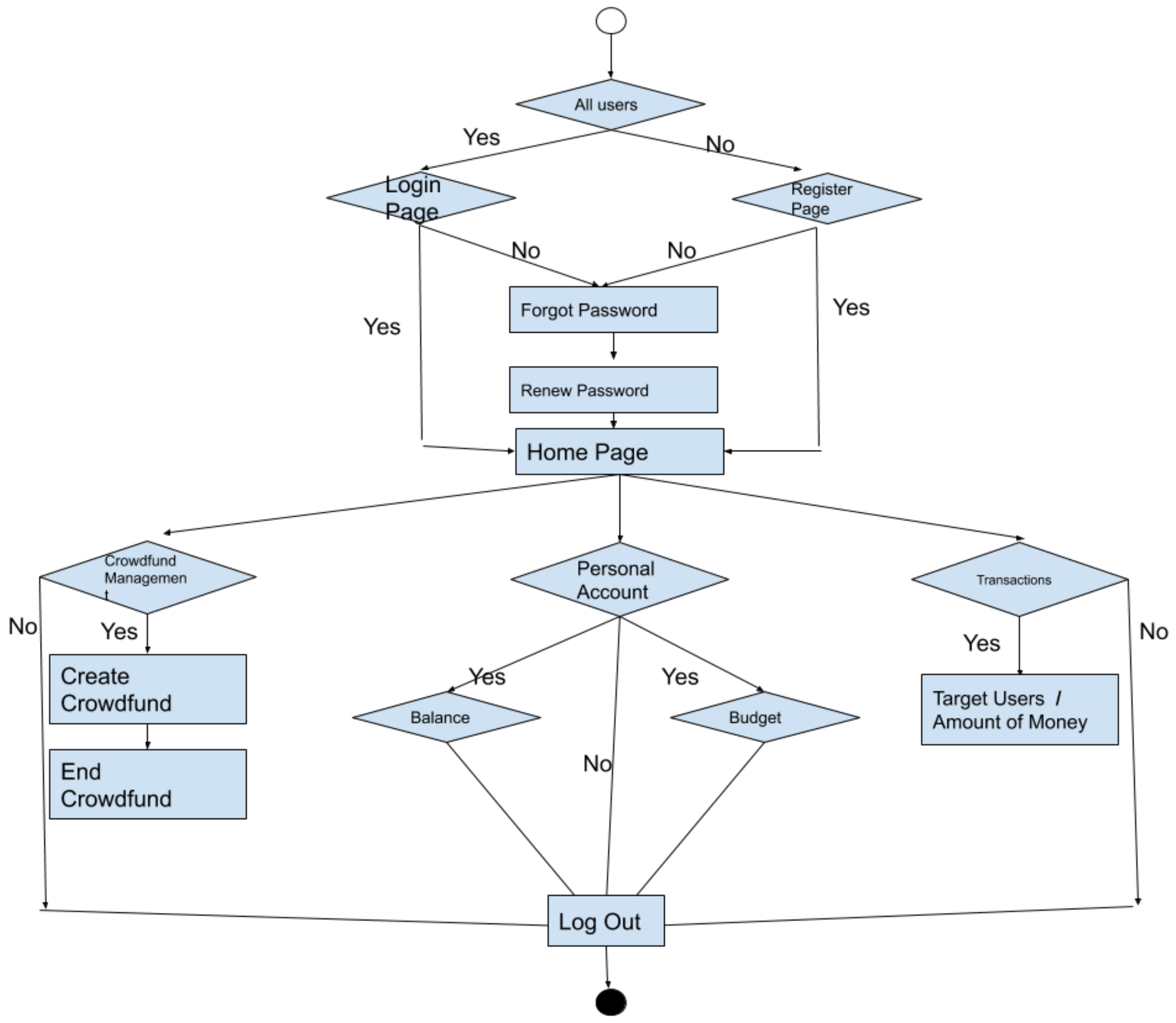
Data Flow Diagram


Data Flow Diagram
Diego Cruz | 10 - 31 - 20
Admin
Login Email | String input
Login Password | String input
Field | Type
Admin Functions
Add User | String givenUsername String givenPasskey Output void
Remove User | String input Output void
Edit User Email | String input Output void
Edit User Password | String input Output void
Edit User Detail | This will be a family of functions that can edit crowdfund, asset, and goal details.
End User Functions
Create Crowdfund | Output void
Contribute to Crowdfund | Input int Output void
View History of Crowdfund | Output ArrayList <CFNode>
Send Funds | Input int Output void
Request Funds | Input int Output void
Remind User of Past Due Requests | Output void
Asset Functions | Family of functions that allows user to add, remove, and edit details of asset objects
Goal Functions | Family of functions that allows user to add, remove, and edit details of goal objects
End User
Login Email | String username
Login Password | String passkey
Field | Type
Crowdfund
Crowdfund 1 | Crowdfund inputname
Crowdfund History | ArrayList <CFNode> outputData
Crowdfund Contributions | CFNode outputData
Assets
Asset 1 | Asset Object that contains several variables storing information regarding a balance or debt.
Goals
Goal 1 | Goal Object that contains several variables that store information regarding a goal.
Server
Write | Function that writes new information from the site to the database
Read | Function that reads information from database to the website.
Database
User 1 | Information will be stored by user, to make writing and retrieving information easier.
User 2 | Multiple Variables will be stored.

State Transition Diagram


State Diagram - PaySmith
Michael Piccerillo | 10 - 31 - 20
User login with account information
Successful / Correct login
User login with account information
Unsuccessful / Incorrect login
Home Page
User selects what actions they want to do
Croudfund Management
• Create a new croudfund
• End a croudfund
• Contribute to a croudfund
• View histroy of a croudfund
Peer to Peer Transactions
• Send money to another user
• request money from another user
• Remind users of past due requests
Expensive Management
• Add a payment
• Remove an expense
• Set a monthly spending limit
• See expense histroy
• Add an asset/debt balance
• Remove an asset/debt balance
Logout
User ends their session and/or leaves the application

Activity Diagram



Test Cases

Requirement	Test Case #	Input/Steps	Expected Output(s)
Register	1	1. At index page, Click on Sign up label on navigation bar 2. Enter first name 3. Enter last name 4. Enter "sjsu@sjsu.com" in the email field 5. Enter "123" in password field 6. Hit Sign up button	● "User registration successful!" pop-up message ● Menu page will open showcasing the default GUI
	2	1. Leave one or more input field empty 2. Hit Sign up button	● Error message " Please fill out this field" pop up
Log in	1	1. At the index page, click on the login button at the navigation bar. 2. Enter "sjsu@sjsu.com" in the username field 3. Enter "1234" in password field	● "Email or password is incorrect" pop-up message will occur.
	2	1. Leave one or more input field empty 2. Click on the login button	● Error message " Please fill out this field" pop up
	3	1. Enter "sjsu@sjsu.com" in the username field 2. Enter "123" in password field	● "Login successful" pop-up message ● Menu page automatically opens and the home page will display.
Start a crowdfund	1	1. On Menu page, click on Crowdfunds tag on	● A "Crowdfund has been added!" popup

		the dashboard column 2. Click on "Add Crowdfund" button 3. Enter day of crowdfund 4. Enter crowdfund's name 5. Enter amount 6. Click on "Add" button	message ● The total amount of crowdfund is shown on the menu page when we click on the dashboard tag. ● The history of crowdfund with amount, name and date will show on the screen
	2	1. Click on Crowdfunds tag on the dashboard column 2. Click on "Add Crowdfund" button 3. Leave one or more input field empty 4. Click on "Add" button	● "Please fill out this field!" popup message
Remove a crowdfund	1	1. On Menu page, click on Crowdfunds tag on the dashboard column 2. Click on "Edit Crowdfund(s)" button 3. On "Action" column, click on "Delete" button at the crowdfund row that you want to delete	● "Record successfully deleted!" popup message ● The crowdfund is no longer in the crowdfund history
Add a expense	1	1. On Menu page, click on "Transactions" tag on the dashboard column 2. Click on "Add Transaction" button 3. Enter day of transaction 4. Enter item's name 5. Enter amount 6. Click on "Add" button	● "Expense has been added!" popup message ● The total amount of expense is shown on the menu page when we click on the dashboard tag. ● The history of transaction with amount, name and date will show on the screen

	2	1. Click on "Transactions" tag on the dashboard column 2. Click on "Add Transaction" button 3. Leave one or more input field empty 4. Click on "Add" button	● "Please fill out this field!" popup message
Remove an expense	1	1. On Menu page, click on Crowdfunds tag on the dashboard column 2. Click on "Edit Transaction(s)" button 3. On "Action" column, click on "Delete" button at the transaction row that you want to delete	The expense is removed from the menu page and expense history
See expense history	1	1. On menu page, click on "Transaction report" tag 2. Choose your desire period time you want to see the report (day to day, month to month, year to year)	The system will show the report expend including, date, name and amount of the expend
Change profile information		1. Click on "Profile" tag on the dashboard 2. Change user information by clicking on the text fields including (firstname, last name, email) 3. Click on "Update" button	System show " User profile has been updated"
Log out		Click on the "Log out" tag on the dashboard	● System shows "logging out" message ● User is redirected to the index page

List of Libraries and Open Source Packages Used

Below are tables of the libraries used as well as the open-source packages used for PaySmith.

Name of Library	Retrieval Link	Function
Lucian Tartea Leno CSS template	https://www.free-css.com/free-css-templates/page252/leno No need to download, this will be included in the ZIP.	CSS template for the front end of the PaySmith site.
Daily Expense Tracker Template using MySQL, PHP, and CSS	https://phpgurukul.com/daily-expense-tracker-using-php-and-mysql/ No need to download, this will be included in the ZIP.	CSS template for the front end dashboard of the PaySmith site.

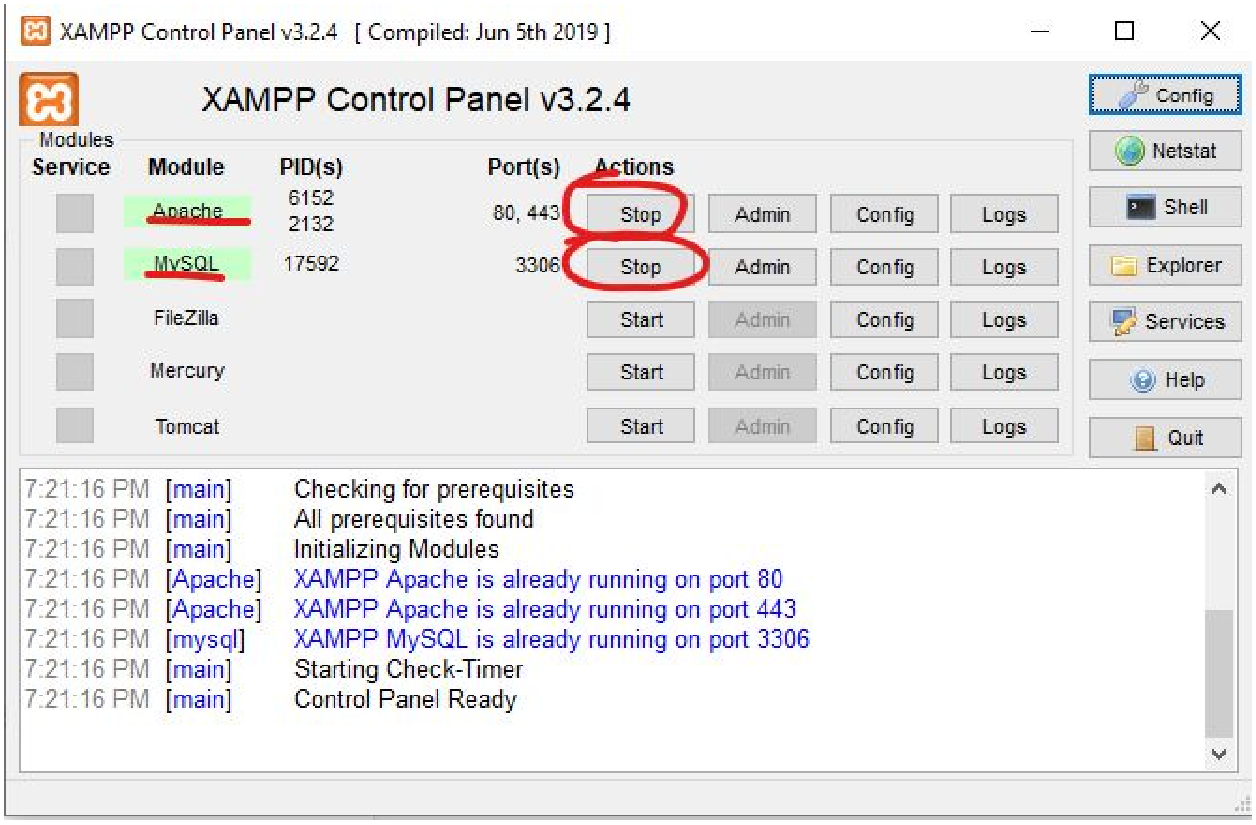
Name of Open Source Package	Retrieval Link	Function
Apache Friends XAMPP Version 7.4.12 Windows 10 Edition	https://www.apachefriends.org/index.html	PHP Dev Environment for hosting both a local database and the PaySmith site.

Installing Libraries and Open Source Packages

XAMPP PHP Dev Environment

1. Download the XAMPP Control Panel ZIP executable, and follow the on-screen prompts to install the dev environment with default settings.

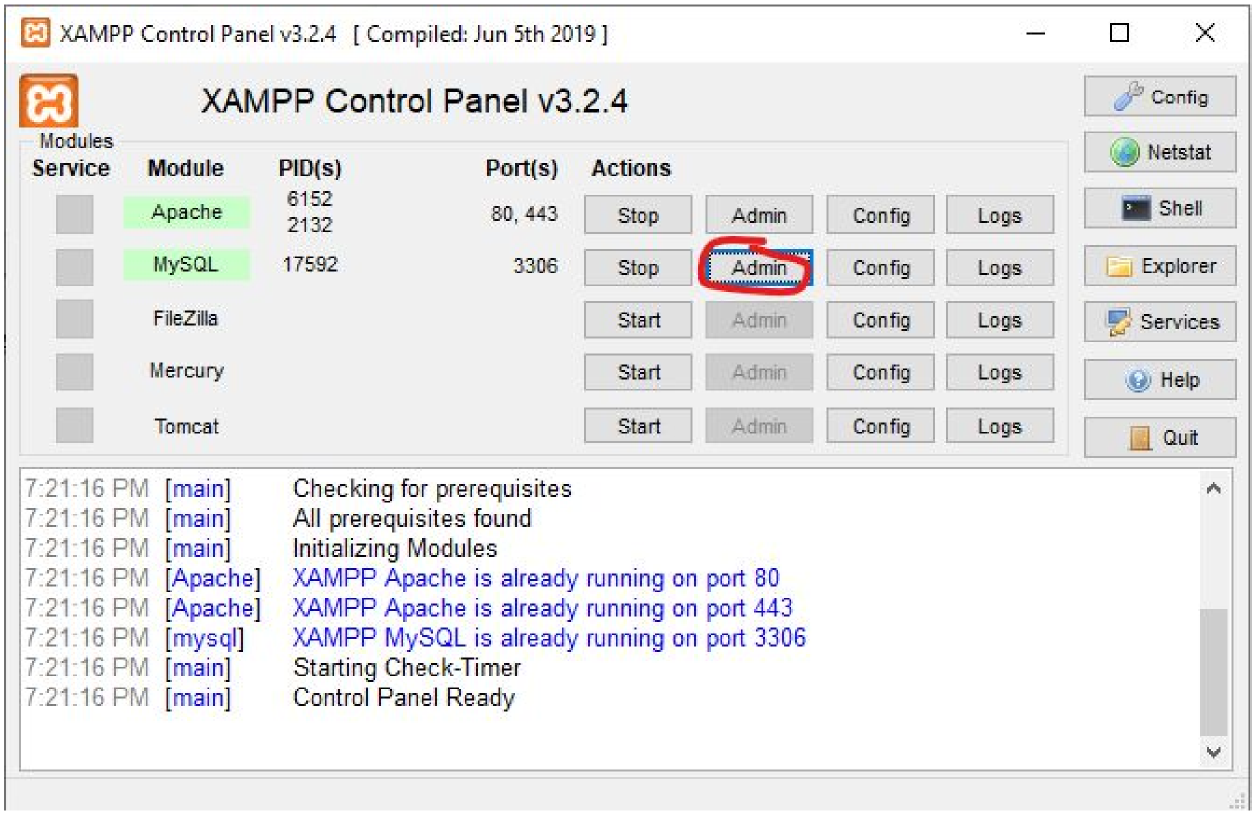
2. After installing XAMPP, run it and start both the Apache and MySQL modules until you get the following screen.



Module	PID(s)	Port(s)
Apache	6152 2132	80, 443
MySQL	17592	3306
FileZilla		
Mercury		
Tomcat		

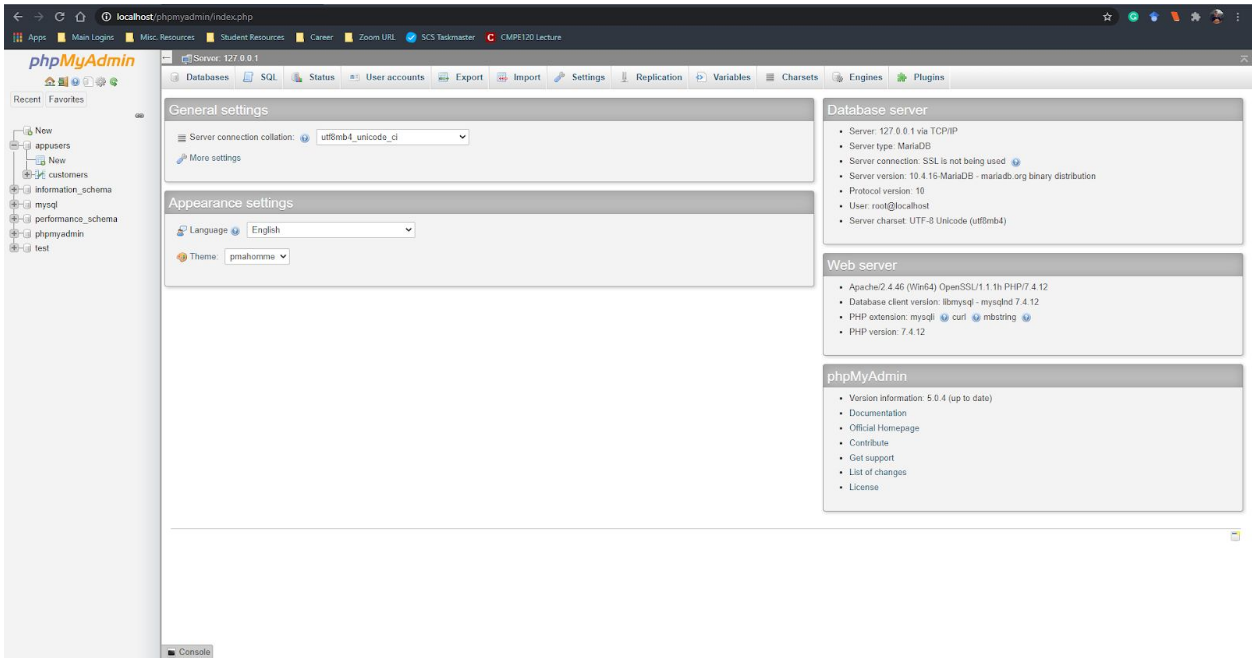
```
7:21:16 PM  [main]      Checking for prerequisites
7:21:16 PM  [main]      All prerequisites found
7:21:16 PM  [main]      Initializing Modules
7:21:16 PM  [Apache]    XAMPP Apache is already running on port 80
7:21:16 PM  [Apache]    XAMPP Apache is already running on port 443
7:21:16 PM  [mysql]     XAMPP MySQL is already running on port 3306
7:21:16 PM  [main]      Starting Check-Timer
7:21:16 PM  [main]      Control Panel Ready
```

3. After this has been set up, click on the admin button corresponding to the Apache module as shown below



```
7:21:16 PM  [main]     Checking for prerequisites
7:21:16 PM  [main]     All prerequisites found
7:21:16 PM  [main]     Initializing Modules
7:21:16 PM  [Apache]   XAMPP Apache is already running on port 80
7:21:16 PM  [Apache]   XAMPP Apache is already running on port 443
7:21:16 PM  [mysql]    XAMPP MySQL is already running on port 3306
7:21:16 PM  [main]     Starting Check-Timer
7:21:16 PM  [main]     Control Panel Ready
```

4. You will then be directed to the following screen.



General settings

Server connection collation: utf8mb4_unicode_ci

Appearance settings

Language: English

Theme: pmahomme

Database server

- Server: 127.0.0.1 via TCP/IP
- Server type: MariaDB
- Server connection: SSL is not being used
- Server version: 10.4.16-MariaDB - mariadb.org binary distribution
- Protocol version: 10
- User: root@localhost
- Server charset: UTF-8 Unicode (utf8mb4)

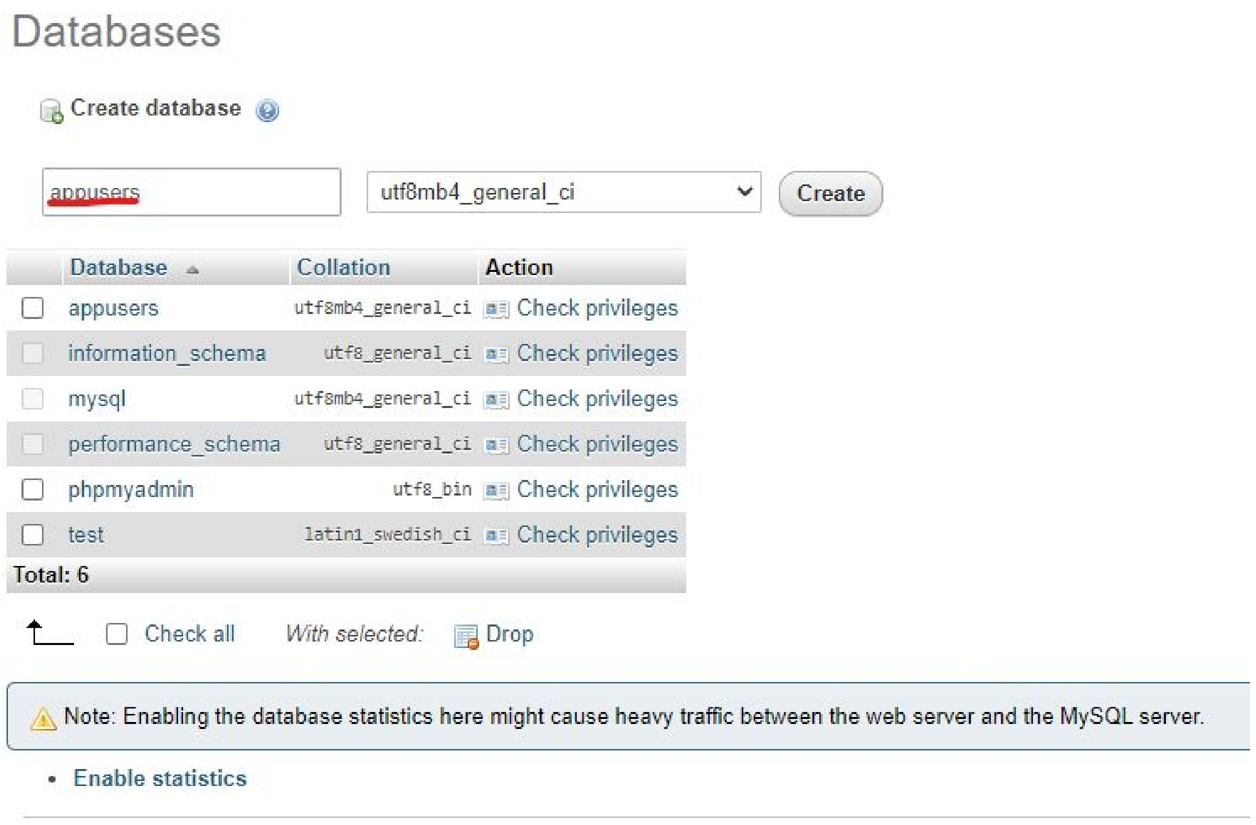
Web server

- Apache/2.4.46 (Win64) OpenSSL/1.1.1h PHP/7.4.12
- Database client version: libmysql - mysqlnd 7.4.12
- PHP extension: mysqli curl mbstring
- PHP version: 7.4.12

phpMyAdmin

- Version information: 5.0.4 (up to date)
- Documentation
- Official Homepage
- Contribute
- Get support
- List of changes
- License

5. Create a new database with the following parameters



Databases

Create database

appusers | utf8mb4_general_ci | Create

	Database	Collation	Action
☐	appusers	utf8mb4_general_ci	Check privileges
☐	information_schema	utf8_general_ci	Check privileges
☐	mysql	utf8mb4_general_ci	Check privileges
☐	performance_schema	utf8_general_ci	Check privileges
☐	phpmyadmin	utf8_bin	Check privileges
☐	test	latin1_swedish_ci	Check privileges
Total: 6			

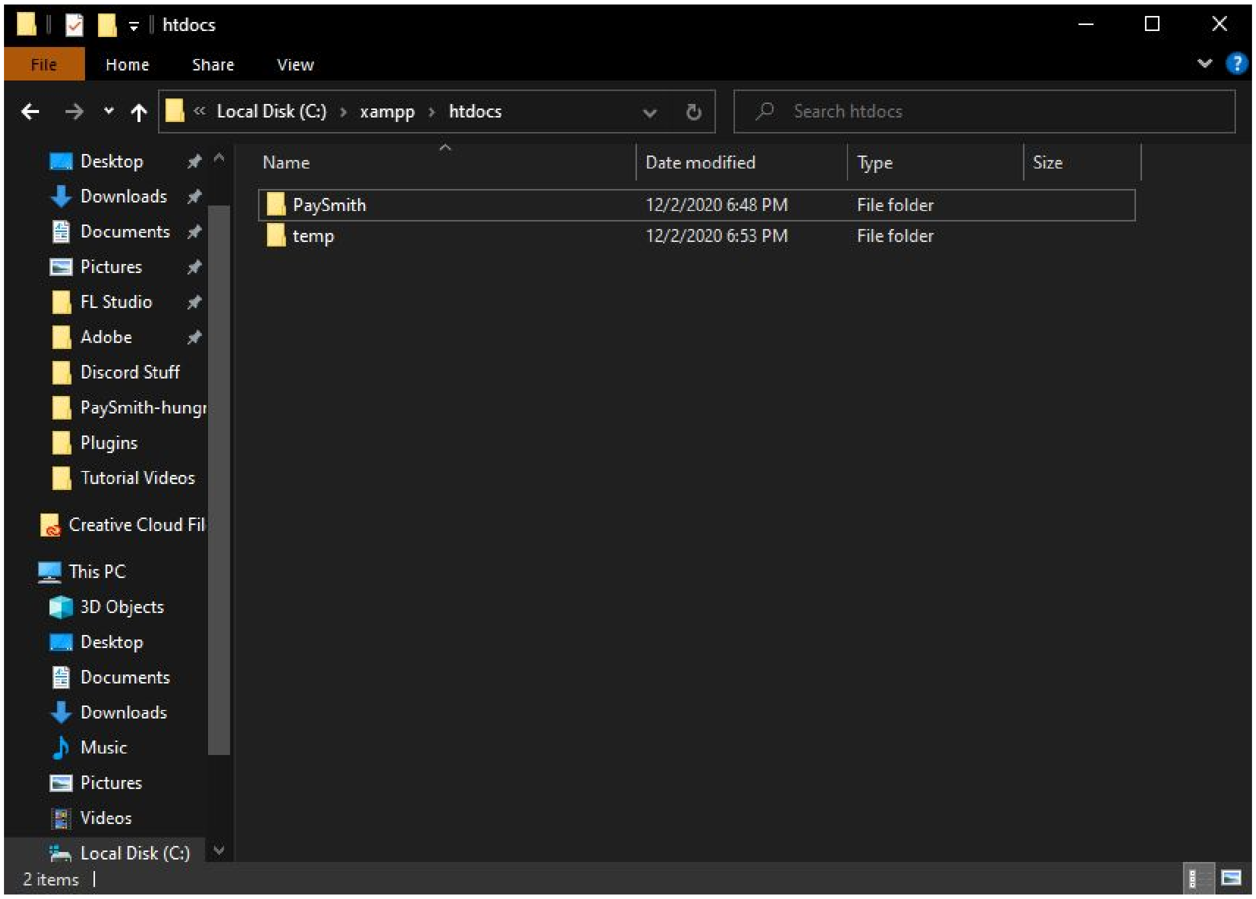
Check all   *With selected:*   Drop

Note: Enabling the database statistics here might cause heavy traffic between the web server and the MySQL server.

- Enable statistics

Setting up Paysmith after installation

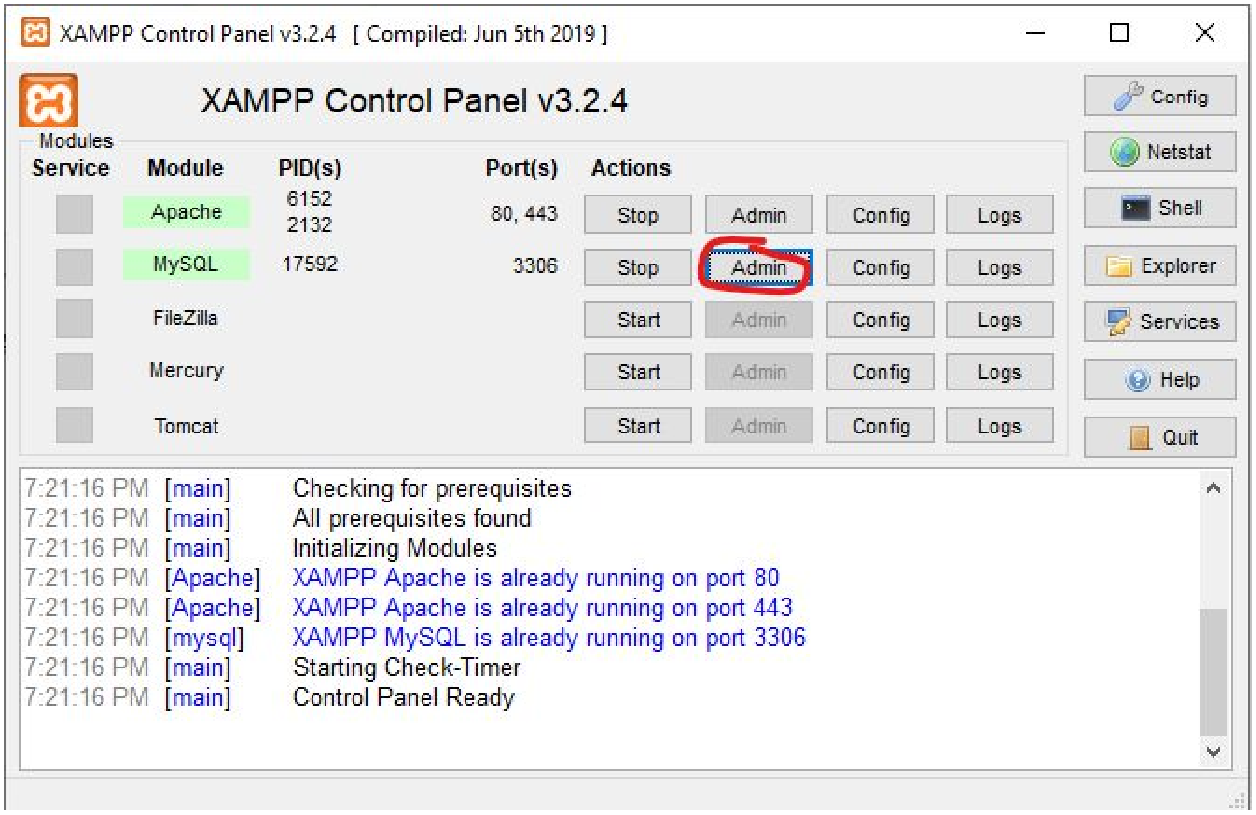
1. In the following filepath, move the contents of the ‘htdocs’ folder to a temp folder and create a PaySmith folder as shown below the filepath.
   a. C:\xampp\htdocs



Name	Date modified	Type	Size
PaySmith	12/2/2020 6:48 PM	File folder	
temp	12/2/2020 6:53 PM	File folder	

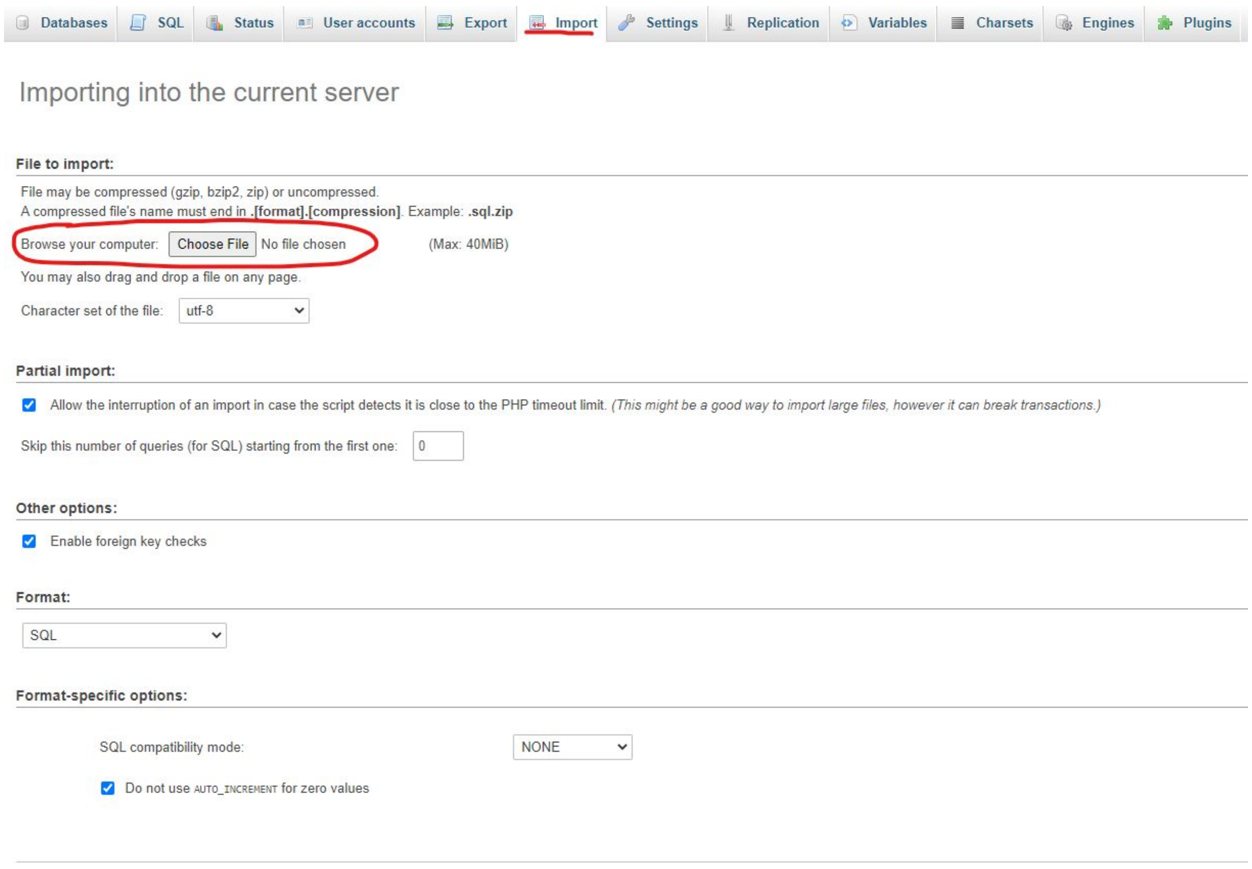
2. Import the contents of the ZIP into the following filepath:
   a. C:\xampp\htdocs\PaySmith

3. On the XAMPP Control Panel, open the mySQL Admin panel as shown below



```
7:21:16 PM  [main]     Checking for prerequisites
7:21:16 PM  [main]     All prerequisites found
7:21:16 PM  [main]     Initializing Modules
7:21:16 PM  [Apache]   XAMPP Apache is already running on port 80
7:21:16 PM  [Apache]   XAMPP Apache is already running on port 443
7:21:16 PM  [mysql]    XAMPP MySQL is already running on port 3306
7:21:16 PM  [main]     Starting Check-Timer
7:21:16 PM  [main]     Control Panel Ready
```

4. On the Control Panel,the import function on the screen, while having the ‘appusers’ database selected, and select the ‘appusers.sql’ file from the PaySmith folder set up earlier, as shown below.



Importing into the current server

File to import:

File may be compressed (gzip, bzip2, zip) or uncompressed.
A compressed file's name must end in **.[format].[compression]**. Example: **.sql.zip**

Browse your computer: Choose File No file chosen (Max: 40MiB)

You may also drag and drop a file on any page.

Character set of the file: utf-8

Partial import:

[✓] Allow the interruption of an import in case the script detects it is close to the PHP timeout limit. *(This might be a good way to import large files, however it can break transactions.)*

Skip this number of queries (for SQL) starting from the first one: 0

Other options:

[✓] Enable foreign key checks

Format:

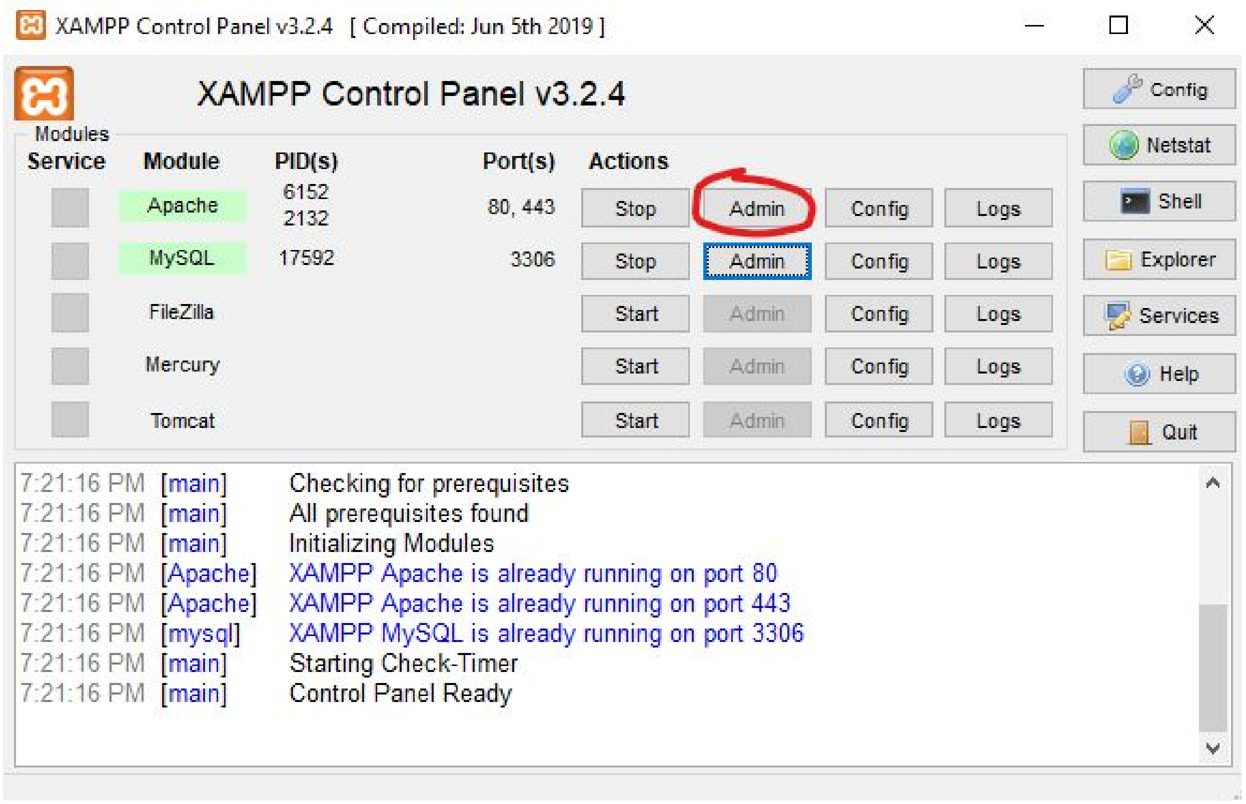
SQL

Format-specific options:

SQL compatibility mode: NONE

[✓] Do not use AUTO_INCREMENT for zero values

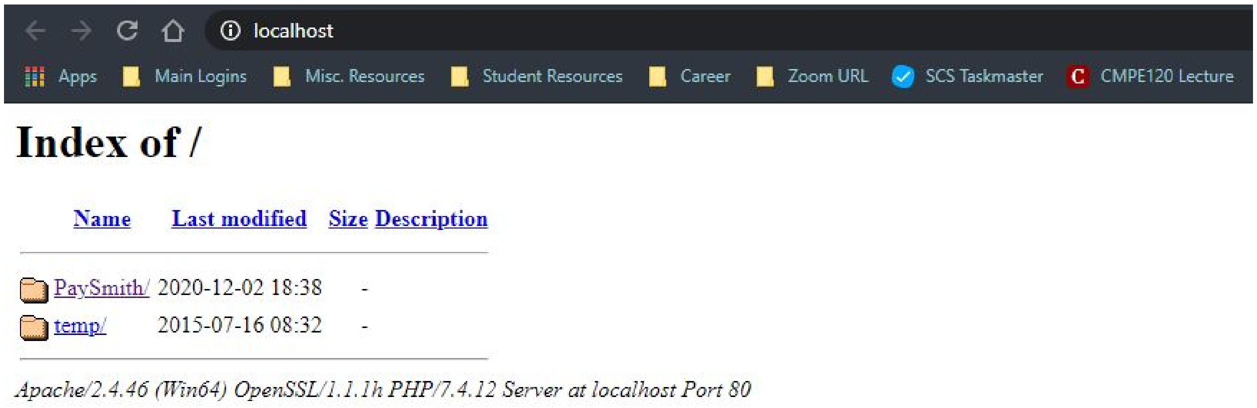
5. On the XAMPP control panel, open up the Admin panel corresponding to Apache.


XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]
XAMPP Control Panel v3.2.4
Config
Netstat
Shell
Explorer
Services
Help
Quit

Service	Module	PID(s)	Port(s)	Actions			
	Apache	6152 2132	80, 443	Stop	Admin	Config	Logs
	MySQL	17592	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

```
7:21:16 PM  [main]     Checking for prerequisites
7:21:16 PM  [main]     All prerequisites found
7:21:16 PM  [main]     Initializing Modules
7:21:16 PM  [Apache]   XAMPP Apache is already running on port 80
7:21:16 PM  [Apache]   XAMPP Apache is already running on port 443
7:21:16 PM  [mysql]    XAMPP MySQL is already running on port 3306
7:21:16 PM  [main]     Starting Check-Timer
7:21:16 PM  [main]     Control Panel Ready
```

6. You will then be directed to the following screen:

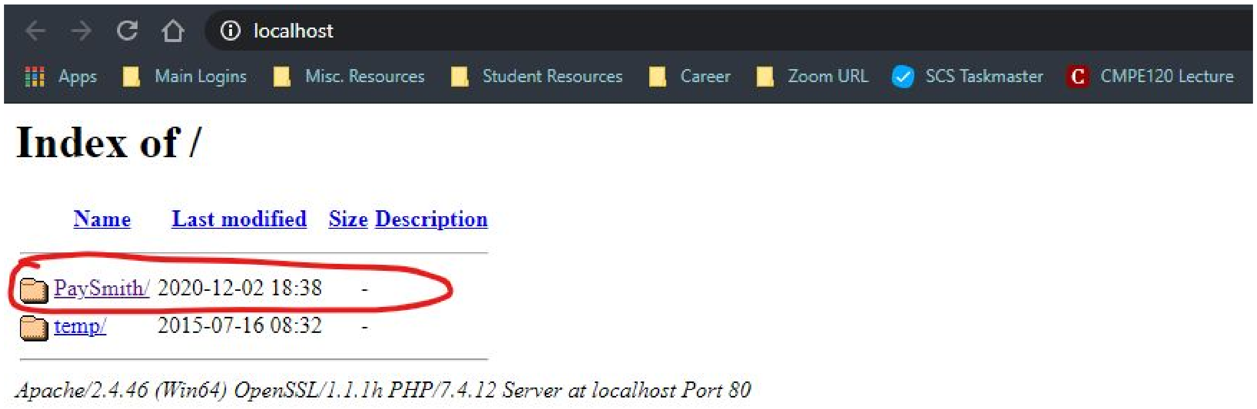


Index of /

Name	Last modified	Size	Description
PaySmith/	2020-12-02 18:38	-	
temp/	2015-07-16 08:32	-	

Apache/2.4.46 (Win64) OpenSSL/1.1.1h PHP/7.4.12 Server at localhost Port 80

7. Select the PaySmith Folder to reach the website as shown below:

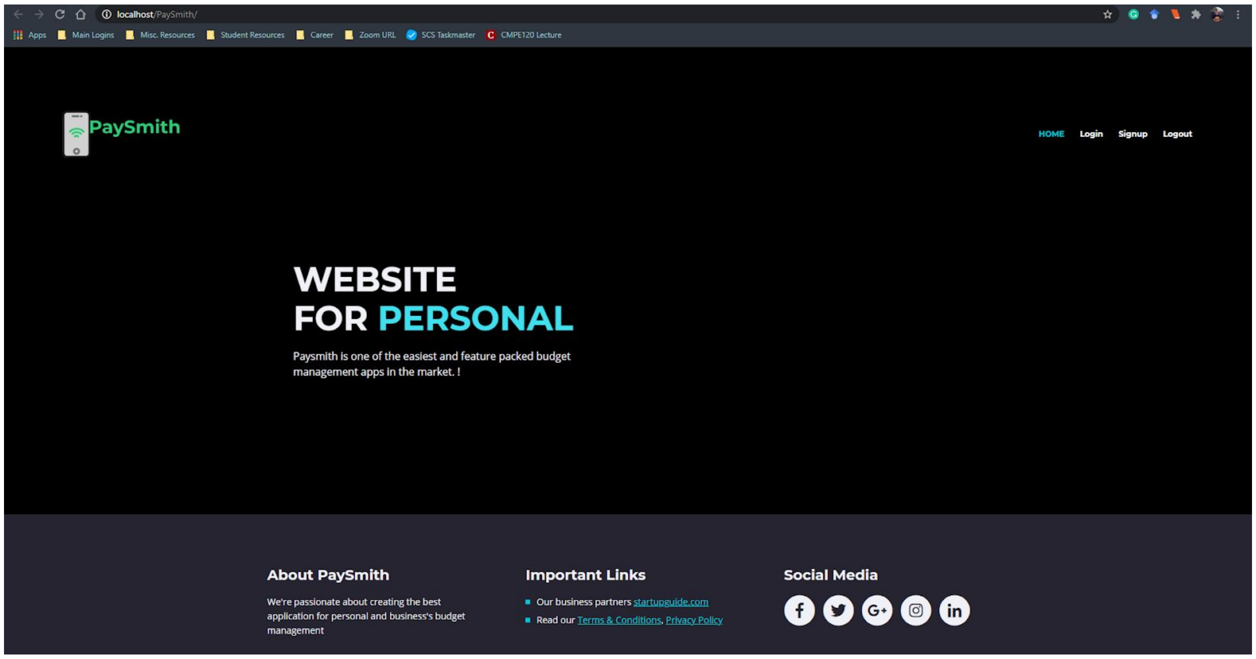


Index of /

Name	Last modified	Size	Description
PaySmith/	2020-12-02 18:38	-	
temp/	2015-07-16 08:32	-	

Apache/2.4.46 (Win64) OpenSSL/1.1.1h PHP/7.4.12 Server at localhost Port 80

8. Congratulations, you are running PaySmith on your machine!



WEBSITE
FOR PERSONAL

Paysmith is one of the easiest and feature packed budget management apps in the market. !

About PaySmith

We're passionate about creating the best application for personal and business's budget management

Important Links

- Our business partners startupguide.com
- Read our Terms & Conditions, Privacy Policy

Social Media

Conclusion

PaySmith set out to simplify an issue that a majority of people face. That issue? To have an easy and intuitive way to not only send money from friend to friend, but to track where your money is going in order to better maintain and save it. The biggest problem people seem to have is that they don't consciously know where every hard-earned dollar is going. With PaySmith, we set out to solve that problem and to not only inform, but educate those who have a hard time saving money. As college students, we know first hand exactly how hard it is to save money and to get by day-to-day. We also understand the importance of friendships, and that sometimes we pay for friends when they don't have the money available right away. Finally, we have the crowdfunding function. Here, we made a comprehensive way for groups, or individuals, to track a pool of money for trips, projects, support, etc; a simple way to gather money for a cause that anyone can contribute to. This would be an ideal way that student organizations on campus here at SJSU would use in order to raise funds.