Path Walking System (PWS) - CSCI 150 Fall 2020 Project

---

1. Introduction

1.1 Purpose

PWS is a web application that allows users to find the shortest path from a given source location, to any designated location at Fresno State. This wiki serves as a collection of requirements for the development of this software. The requirements given are to aid the development of the web application by providing a proper outline of priorities for the features.

1.2 Scope

PWS is a system created for Fresno State students. Students and visitors who are new to Fresno State campus typically have trouble traversing the campus. This application looks to provide a simple, yet effective web application that will help guide in finding walking paths to students and visitors to their requested destinations. Some additional notes will also be provided in the system to help the students and visitors choose the building entrance which is closest to their class or

room number. Students, faculty, staff members and any other user of the system will be able to navigate within the building itself to a specific room number for enhanced navigation with the help of floor plans associated with that building.

New users will have an option of creating an account and logging in or using the website as a guest. Each user will be able select their current location, or set a pin, as a starting point. Next, the user will select their destination. This will provide the user with the fastest route to their destination from the given starting point. The database will facilitate the user signup, login as well as saving the history of the searched locations or pinning their favorite locations to their account. The user needs to be connected to the database without any network issues for the login and search history to work.

1.3 Overview

This wiki will cover an in-depth description of the components of the web application. The wiki will be broken up into sections to clarify the design of the software. A description will be given about the layout of the user interface layer, business layer, and the database layer of the system. All the implemented features will then be introduced, where a description for each feature is given along with any dependencies associated with it. A schedule will be created along with a Use Case Diagram to help with the visual representation of the features.

2. Description

2.1 Product Components

The main components for the web application are distributed among these three layers of the system: user interface layer, business layer, and database layer

2.1.1 User-Interface

User interface layer provides the user an interactive experience. User interface provides the direct relationship between user and all the features of the PWS web application. HTML and CSS are powerful GUI development languages and therefore have been chosen to expedite the user interface of the application.

The user interface is incorporated with the sign up, login, forgot username & password, dropdown menu to provide the user-friendly interaction.

2.1.2 Business Layer/The server

Connects the user with the database for (signing up, logging in, user verification, providing user-specific history). This layer is responsible for all requests made by the user. This layer interacts with the database and provides essential responses to a user's request. For instance, when a user puts a request to login, then the business layer connects that login information request with the database, retrieves it and uses it for authentication of the user. Business layer is incorporated with Google maps API and interacts with the user interface and provides the requested path to the user.

In order to implement the essential tools that will be used are PHP, MySQL, Js, Google Maps API etc.

2.1.3 Database

The localhost database is used to store the user signup information such as username, passwords, verification details, history, and current location. Whenever a new user signs up, the data provided by the user is stored into the database which then is used for later user log-in. After

signing up, the user then can login using the information provided by the user at the time of signup. Provided input information in the login bar is verified by using MySQLi and php.
A query is done to make sure that the user logs in the system using the correct username and password, which already exists in the database. User's current position will also be stored into the database as well as with the search history of the user.
Example of the data stored in the database is given below:

8	A	A@gmail.com	$2y$10$IBDUNe3MVub2dEQxJXtb8.XMVtYIei7Lcb2C9Patpjy...
9	D	D@gmail.com	$2y$10$E73.U2Q992cJ01..chevOuduBQrJZFmiXcX1gSeBiU3...
10	Gabriel	GO@gmail.com	$2y$10$qWGIPg3MQ.vJHFZo./FRwediXRwupOgCjoz0zk6l3FB...

The saving search history feature worked separately but it was giving an error while collaborating with the web application. Below is the example of saving history into database(user-specific).

8	McKee-Fisk Room 113
8	McKee-Fisk Room 106

2.2 Users

The target users or audience for the PWS system are especially the Freshman students at Fresno State, who are not very familiar with the campus walking paths to a designated building. Although the system is designed user-friendly and therefore any type of user will be able to use the system very easily. Users will type in the address bar PWS and will be able to load into the home page, login/signup (if willing to do so) or else can use a guest account.

3. Requirements

3.1 User Interface

ID: R1

Homepage:

- Description: The homepage of the application. Other buttons will be mapped to this page.
- Rationale: In order for the user to have a start page.
- Dependency: None

ID: R2

Navigation page:

- Description: Page for searching and navigating to the buildings, with additional notes.
- Rationale: User interactive page for searching the path to specific building
- Dependency: R1

ID: R3

Sign up page:

- Description: User can input a new login and password which will be stored in the database as a new account.
- Rationale: For the user to have an option to create an account
- Dependency: R1 or R2

ID: R4

Login page:

- Description: User can input a username and password to login. The system will check if the inputted information is valid in the database.
- Rationale: In order for the user to login to their previously created account.
- Dependency: R1 or R2

ID: R5

Map:

- Description: The main section of the application. The user can zoom in and out, find locations, get current location, etc.
- Rationale: In order for the user to find the shortest route from two locations at Fresno State.
- Dependency: R2

ID: R6

Search bars:

- Description: User can input building names located at Fresno State. As you begin to input more characters, your search will become narrowed down.
- Rationale: In order for the user to find any building at Fresno State.
- Dependency: R2

ID: R7

Floorplans page:

- Description: After selecting a desired building from the search bar, the user will be able to view the inside of the building via the floor plans. The floors plans give details for each floor of a building, including classrooms, stairs, fire extinguishers, etc.
- Rationale: In order for the user to get a more detailed view of the desired location.
- Dependency: R2 or R6

ID: R8

About page:

- Description: Detailed information regarding the web application and its user guide.
- Rationale: For enhanced guidance of the application
- Dependency: R2

3.2 Business Layer Requirements

ID: R1

User SignUp:

- Description: User will provide the required information which then will be stored into the database using PHP and MySQLi.
- Rationale: In order for the user to have an option to create an account
- Dependency: None

ID: R2

User LogIn:

- Description: User will input the login information and this information will be verified with the associated stored information in the database. Once verified, the user will be prompted to the navigation bar, and upon false verification the user will have the option to look for *forgot username & password.*
- Rationale: In order for the user to login to their previously created account.
- Dependency: R1

ID: R3

Google Maps API:

- Description: This uses the Google Maps API to display a real time map of Fresno State.
- Rationale: In order for the user to see the map of Fresno state and other features will depend on this map.
- Dependency: None

ID: R4

Current Location:

- Description: The google maps API will get access to the user's current latitude and longitude and these points will be displayed in the user interface as the user's current location.
- Rationale: In order to enhance the overall navigation process for the user.
- Dependency: R3

ID: R5

Zoom In/Out:

- Description: The google maps API will provide the in-built user-interface for zooming in and out the map for a more enhanced view of navigation for the user.
- Rationale: In order to enhance the overall navigation process for the user.
- Dependency: R3

ID: R6

Full Screen:

- Description: The google maps API provides the functionality that allows the user to put the map on full screen mode, so that the user can see the directions clearly even on smaller screens.
- Rationale: In order to enhance the overall navigation process for the user.
- Dependency: R3

ID: R7

Street View:

- Description: The user will be able to see street view incorporated within the google maps API, for real world view of any given location.
- Rationale: In order to enhance the overall navigation process for the user.
- Dependency: R3

ID: R8

Search From:

- Description: The user will be able to choose their starting location by searching for a desired location at Fresno State, or their current location.
- Rationale: In order to allow the user to choose a starting point.
- Dependency: R3

ID: R9

Search To:

- Description: The user will be able to choose their ending location (end point) by searching for a desired location at Fresno State.
- Rationale: In order to allow the user to choose their ending point.
- Dependency: R3

ID: R10

Routes:

- Description: The user will be able to see multiple routes displayed on the map of the campus with the fastest route emphasized.
- Rationale: This will show the user how to reach their destination.
- Dependency: R3

ID: R11

Select route:

- Description: This will allow users to be able to select their preferred route to their destination. The selected route will be emphasized.
- Rationale: The user should have options on the route they want to take.
- Dependency: R3, R10

ID: R12

Time of Arrival:

- Description: The user will be able to see the estimated time of duration the route will be.
- Rationale: This will provide the user with important information regarding the time it will take the user to reach their destination on the route they decide to choose.
- Dependency: R3, R10

ID: R13

Floor Plan for respective building:

- Description: The user will be able to see the corresponding floor plan for the building they are going to.
- Rationale: This shall provide the user with an image for each corresponding building and floor.
- Dependency: R9

ID: R14

Floor Differentiation:

- Description: The user will be able to see the individual floors for each corresponding building.

- Rationale: This will provide the user with an inside view of the building and the floors inside.
- Dependency: R9, R13

ID: R15

Note Section:

- Description: The user will be provided with notes about each floor.
- Rationale: This will provide the user with more information about the buildings and floors they choose.
- Dependency: R9, R13

3.3 Non-Functional Requirements

ID: R1

Hardware requirements:

Description: Any hardware device that supports the internet browser. Android/IOS/Blackberry mobile phones, windows /macos/linux/ubuntu/unix laptops or desktops.

ID: R2

Internet Connectivity:

Description: Requirement for an internet connection to use the web application.

ID: R3

Responsiveness:

Description: The responsive time between a given request by the user in the user interface and the provided response by the system should not be more than 5 seconds. (if provided sufficient internet connectivity)

ID: R4

Security:

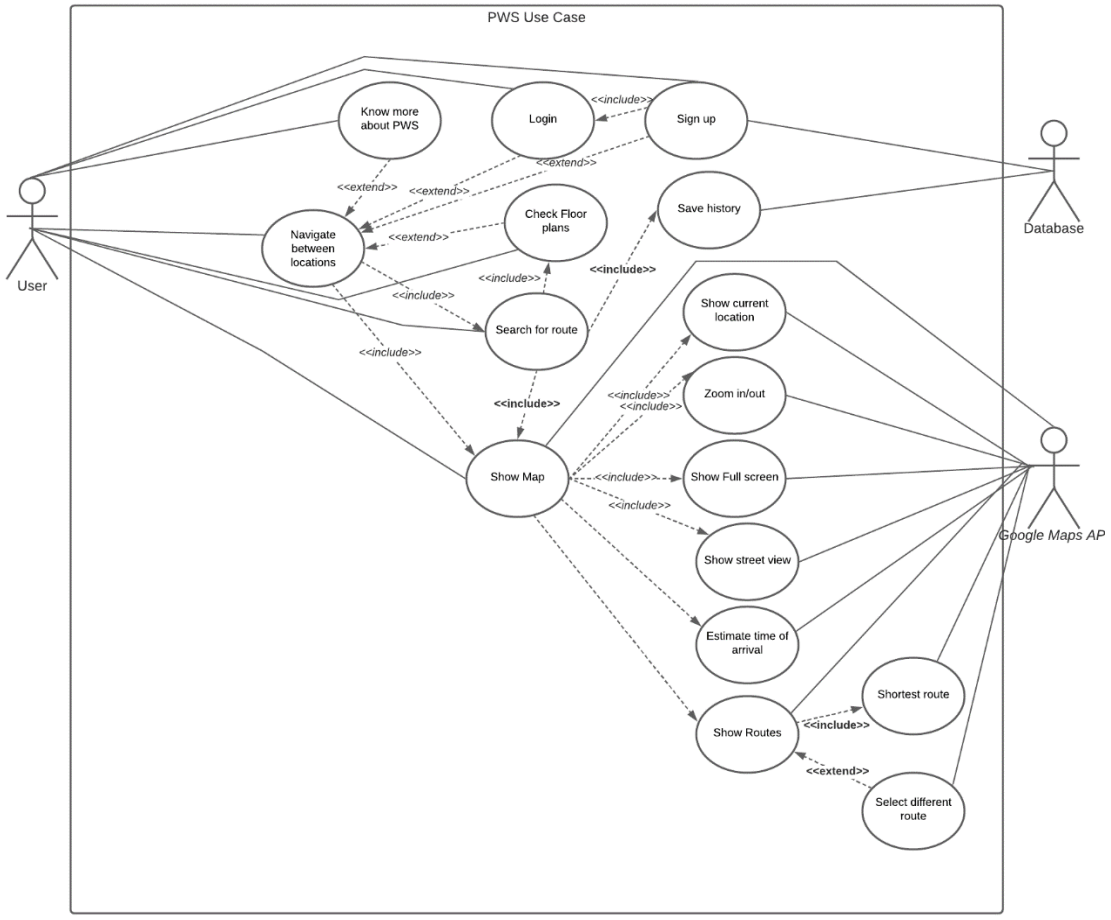
Description: Passwords of all the users are hashed during the time of Signup and are verified when the user logs in.

4. Prioritization (Schedule)

Week	Back-end Goal	Front-end Goal
1	Collecting database (using PHP), Learning google maps API	Creating Home page for the user to interact with
2-3	Collecting Floor Plans database	Applying the map APIs, Layout for campus map
4-5	Database stores username and password	User login, Sign up option
6-7	Store search history in database, using Google Maps API	Search navigation bar with search history, zoom in/out
8-9	Store the different floors in Google Maps API	Showing the different floors within a building
10-11	Use Google Maps API to find the fastest route, ETA integration	Choice of route, finding the shorting path
12-13	Integration of unit code files into one system.	Testing the unit functionality an
13	Reviewing the code and refactoring the code to make any updates or changes.	Final updates and changes.

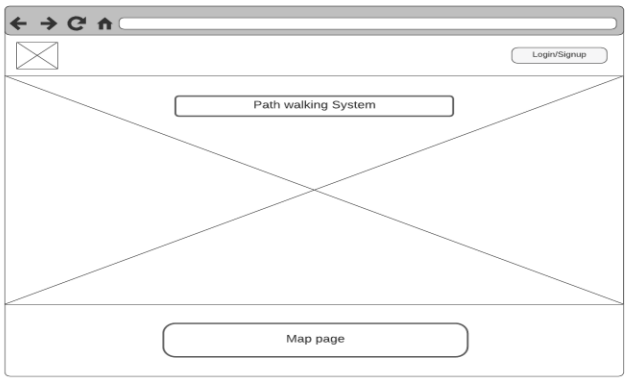
5. Diagrams

5.1 Use Case Diagram

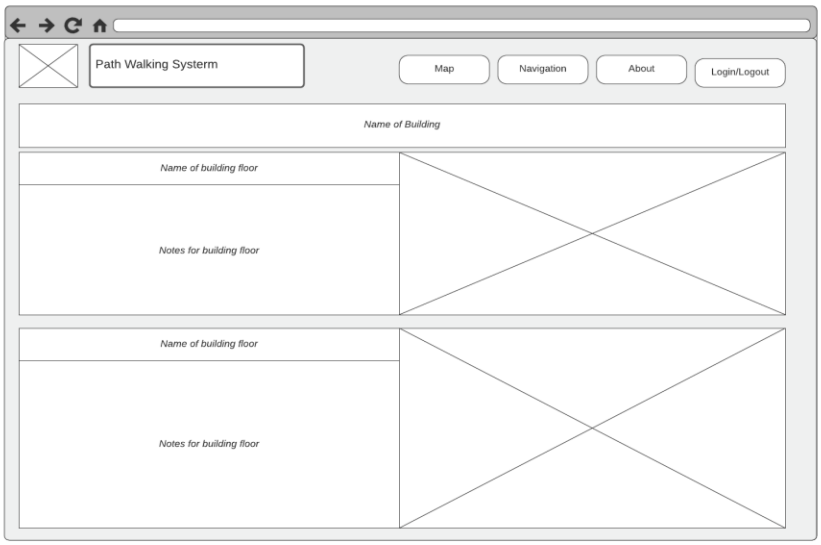


5.2 Wireframe diagram

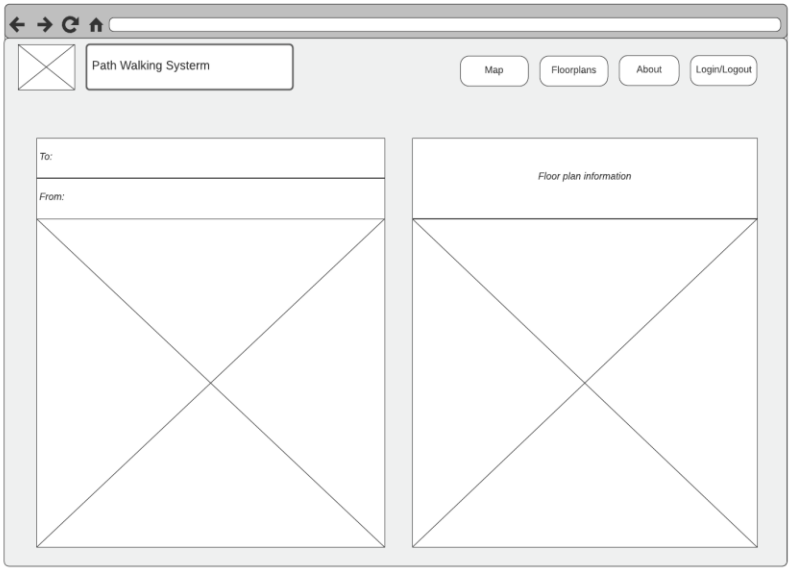
5.2.1 Home page



5.2.2 Floorplans page



5.2.3 Navigation page



5.2.4 About page

5.2.5 Login/Signup

5.3 Class Diagrams

Class Diagram: -

A. Noun Extraction

Step1: PWS system abstract: PWS is a system created for Fresno State students. Students and visitors who are new to Fresno State campus typically have trouble traversing the campus. This application looks to provide a simple, yet effective web application that will help guide in finding walking paths to students and visitors to their requested destinations. Some additional notes will also be provided in the system to help the students and visitors choose the building entrance which is closest to their class or room number. Students, faculty, staff members and any other user of the system will be able to navigate within the building itself to a specific room number for enhanced navigation with the help of floor plans associated with that building. New users will have an option of creating an account and logging in or using the website as a guest. Each user will be able select their current location, or set a pin, as a starting point. Next, the user will select their destination. This will provide the user with the fastest route to their destination from the given starting point. The database will facilitate the user signup, login as well as saving the history of the searched locations or pinning their favorite locations to their account. The user needs to be connected to the database without any network issues in order for the login and search history to work.

Step2: Noun Extraction: user, students, visitors, admin, destination, building, entrance, class, navigation, floor plans, account, login, signup, guest, current location, starting point, fastest route, database, searched locations, favorite locations.

Step3: candidate class: admin, user, guest, database, homepage, floorplan, building, navigation, search, history, route, google maps API

Step4: subclasses: zoom view

B. CRC Cards:

1. User class:
    a) Responsibilities:
        - Create user account/sign up
        - Sign in
        - Log out
    b) Collaboration
        - Database class
        - Homepage class
2. Admin Class:
    a) Responsibilities:
        - Sign up
        - Sign in
        - logout
        - Add/delete/update floorplans in database
    b) Collaboration:
        - Database class

3. Guest Class:
    a) Responsibilities:
        - None
    b) Collaboration:
        - Homepage class
4. Database class
    a) Responsibilities:
        - Store username and password

- Store floor plans
- Provide user confirmation upon sign in
- Store search history of the user

b) Collaboration:

- User class
- Admin class
- Search class
- Floorplan class

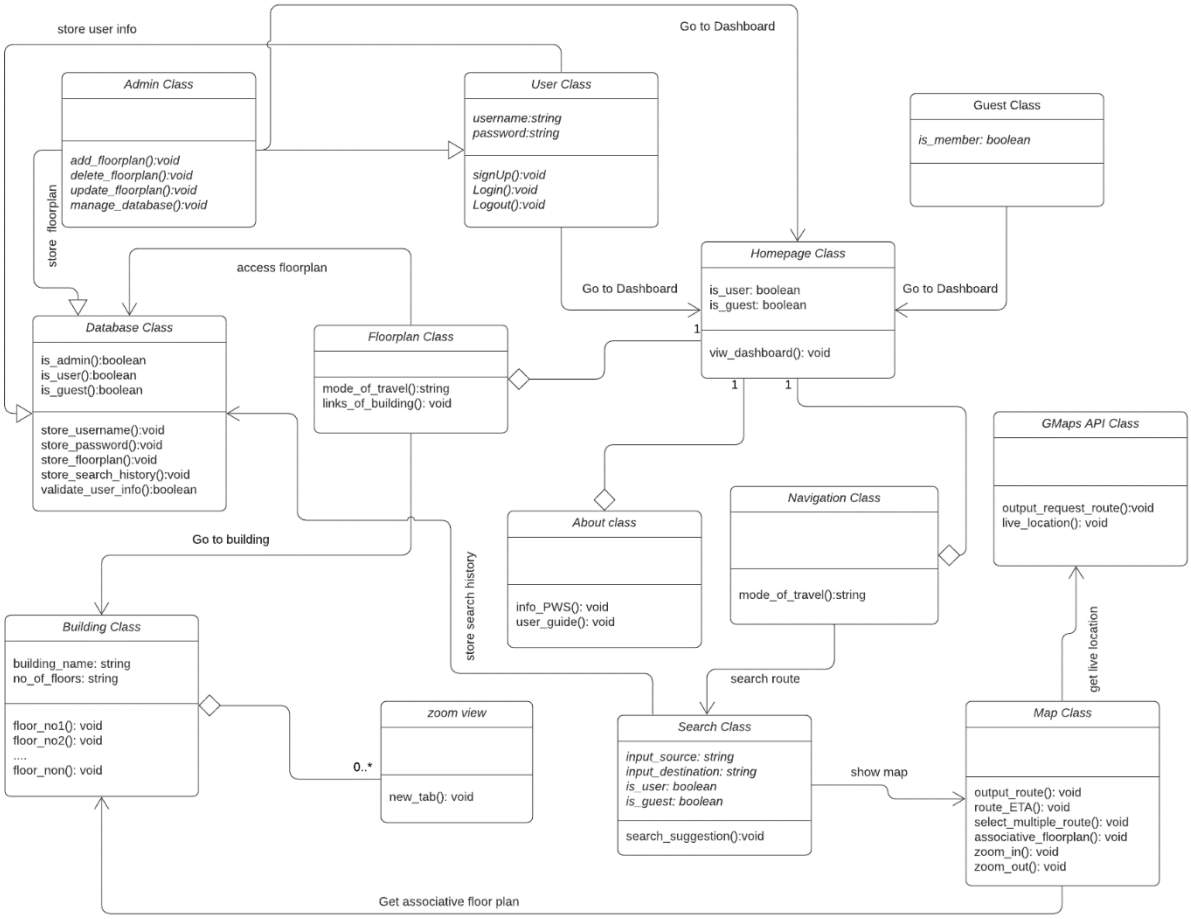
5. Homepage class:
    a) Responsibilities:
        - Provide dashboard
    b) Collaboration:
        - Floorplan class
        - Navigation class
6. Navigation class:
    a) Responsibilities:
        - Select mode of travel
    b) Collaboration:
        - Search class
7. Search Class:
    a) Responsibilities:
        - Take input as source
        - Take input as destination
        - Store searched history into the database
        - Suggest search history
    b) Collaboration:
        - Route class
8. Route Class:
    a) Responsibilities:
        - Show the requested route
        - Display multiple route
        - Provide selection between different routes
        - Show ETA (estimated time of arrival)

- Show associative floorplan of the destination

b) Collaboration:

- Floorplan class
- Google Maps API class

9. Floorplan Class:
    a) Responsibilities:
        - Access floorplans from the database
        - Provide links to the buildings floor plan
    b) Collaboration:
        - Building class
        - Database class
10. Building Class:
    a) Responsibilities:
        - Provide floor plans for all the floors
    b) Collaboration:
        - Zoom view
11. Zoom view:
    a) Responsibilities:
        - Provide enlarged view of specific floorplan
    b) Collaboration:
        - Building class
12. GMaps API Class:
    a) Responsibilities:
        - Provide all the requests for the routes
        - Provide user live location
    b) Collaboration:
        - Route class


store user info
Go to Dashboard
Admin Class
add_floorplan():void
delete_floorplan():void
update_floorplan():void
manage_database():void
User Class
username:string
password:string
signUp():void
Login():void
Logout():void
Guest Class
is_member: boolean
store floorplan
access floorplan
Homepage Class
is_user: boolean
is_guest: boolean
viw_dashboard(): void
Go to Dashboard
Go to Dashboard
Database Class
is_admin():boolean
is_user():boolean
is_guest():boolean
store_username():void
store_password():void
store_floorplan():void
store_search_history():void
validate_user_info():boolean
Floorplan Class
mode_of_travel():string
links_of_building(): void
1
1
1
GMaps API Class
output_request_route():void
live_location(): void
About class
info_PWS(): void
user_guide(): void
Navigation Class
mode_of_travel():string
Go to building
store search history
Building Class
building_name: string
no_of_floors: string
floor_no1(): void
floor_no2(): void
....
floor_non(): void
search route
get live location
zoom view
new_tab(): void
0..*
Search Class
input_source: string
input_destination: string
is_user: boolean
is_guest: boolean
search_suggestion():void
show map
Map Class
output_route(): void
route_ETA(): void
select_multiple_route(): void
associative_floorplan(): void
zoom_in(): void
zoom_out(): void
Get associative floor plan