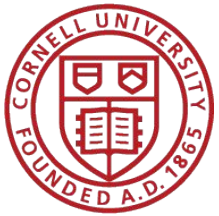


CHAPI

The ‘Conference Housing Availability Planning Instrument’ algorithm to automatically suggest housing locations for summer events.

At a High Level, What Does the CHAPI do?

Better assignments for Conference housing

- Planner preferences can be used to find the optimal housing solution

Automation

- Give Planners quick access to correct housing solutions

Housing data-structures

Campus Area

- List of **Buildings**

Building

- List of **Floors**
- Attributes
  - Air Conditioning
  - Elevators
  - etc.

Floor

- Floor Number
- List of **Suites**
- List of **Rooms**

Suite

- Suite Number
- List of **Rooms**

Room

- Room Base
- List of **Beds**
- Number of beds

Bed

- Bed Number
- Occupancy Graph (OG)

Useful Definitions

Constraint: physical building constraints, overlapping bookings

Strong Preference: what Planners prefer to happen with a given Event

Weak Preference: what Events prefer in their Buildings

Suggestion: one possible housing assignment for a set of Events

Step: forward progression to complete suggestion (i.e. assign an Event a Building/Floors/Rooms)

Preference Score: a heuristic calculation using a sum of preference weights

Housing Assignment Algorithm - Setup

1. Initialize all Campus Locations, Buildings, Floors, etc. with no occupancy
2. Import all facility holds, Staff rooms, other non-CES holds, etc.
3. Fill in hard-scheduled events (Reunions, Commencement, etc.)
4. Start algorithm

Housing Assignment Algorithm

1. Take a *step* (using *strong preferences* to guide) (within *constraints*)
2. Repeat until there are no *strong preferences* left
3. Take the rest of the *steps* necessary to complete a *suggestion*
4. Calculate *preference score* using *strong/weak preferences*
5. Repeat for all possible *suggestions* that are left
6. Pick the *suggestions* with the highest *preference score*

Databases

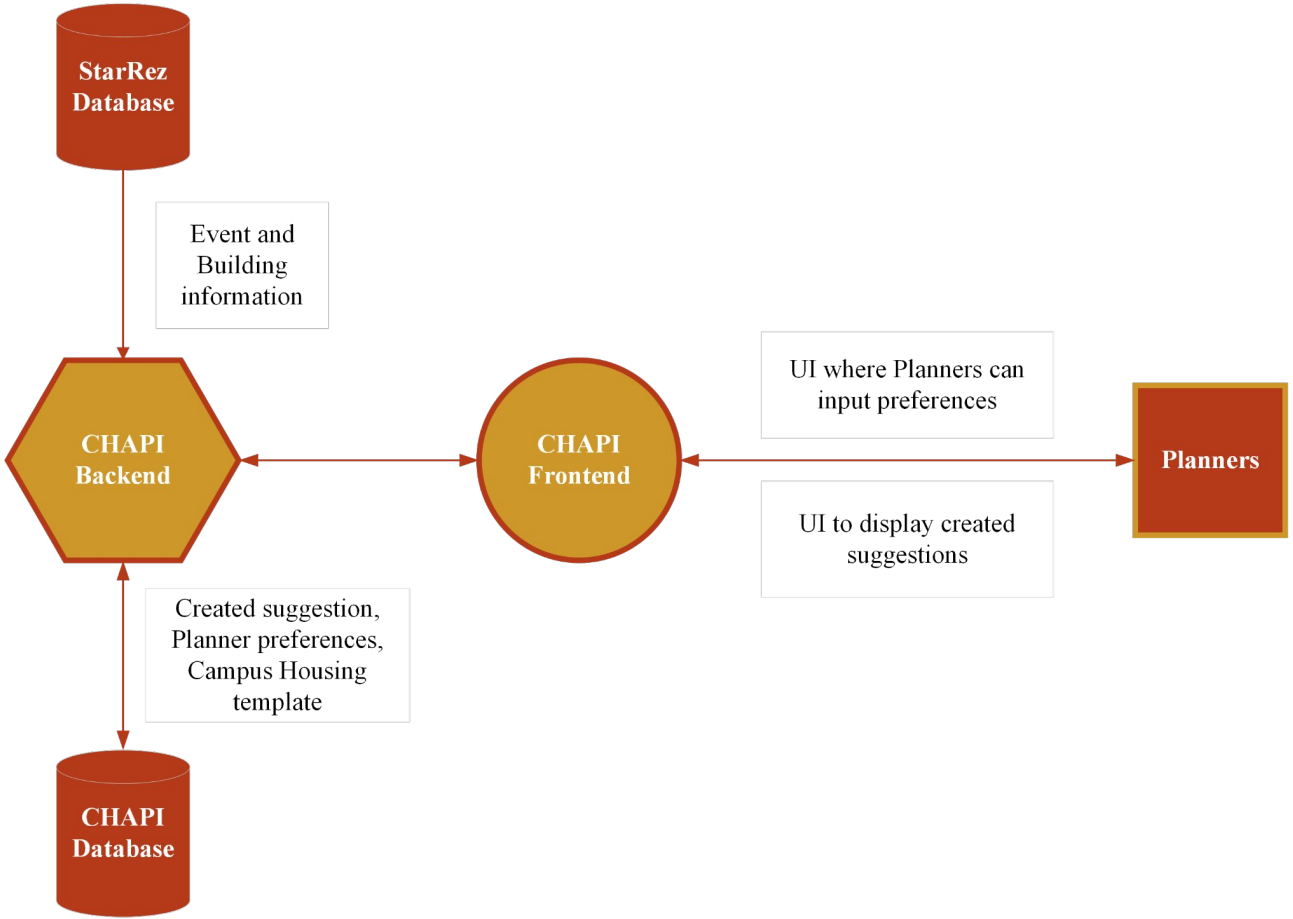
StarRez API

- Can pull event names, attendee estimates for the Summer Term

New CHAPI database

- Planner preferences for each event
- CHAPI housing suggestions
- Cornell University building's template

Workflow


StarRez Database
Event and Building information
CHAPI Backend
CHAPI Frontend
UI where Planners can input preferences
Planners
UI to display created suggestions
Created suggestion, Planner preferences, Campus Housing template
CHAPI Database

Code Base Maintenance

Maintenance Level: **Low**

- Once the algorithm is built, with input and output data structures defined, dynamic changes should be low
- Modularity will work if other applications need to be built on top
- Documentation and code comments are present

Pros

- Clearly doable, Max possesses a prototype
- Saves time and effort for Planners
- Could be a local on someone's computer or a small web-app
- Narrow problem

Cons

- Time and resource investment
- Implementation is not as simple as other projects (custom algorithm, custom preference calculation)

Progress

- Whiteboard of CHAPI
- Data structures for housing
- Pseudocode for algorithm

Roadblocks

- Need to start building CHAPI
- No real roadblocks yet

Technology Needs

In the simplest fashion, a virtual machine to run Docker containers to host frontend, backend, and database.

If more separation is wanted, AWS provides an array of products to:

- Host the database
- Host our backend API
- Create a web server for the frontend

Alternative Solutions

Locally run process

- Planner meet to collect all preferences, and then run algorithm

Do nothing

- Planners meet and use Max's iterative Excel solution

Microsoft Access (as a DBMS)

- Host data, use Microsoft tools to collect and display
- Still need algorithm code to create suggestions