Joe Opseth
July 3, 2018

Project Portfolio

Exam Builder

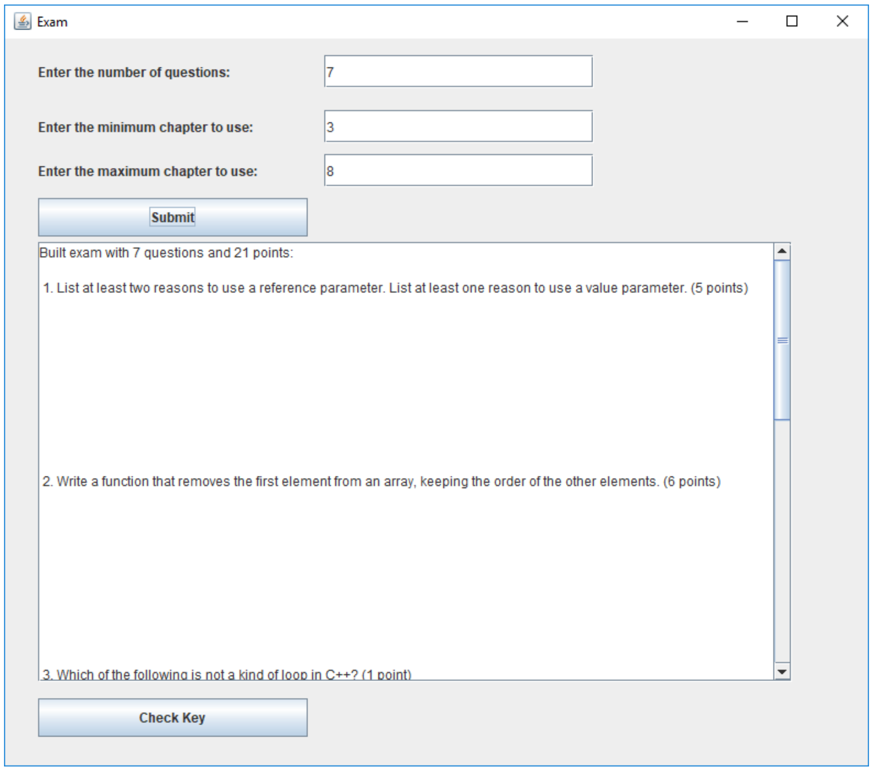
Object-Oriented Programming, University of Wisconsin-River Falls

A Java application that builds exams from a pool of questions stored in a text file. Includes a GUI and text-based interface. Questions may be multiple choice, short answer, or long answer, and the exam generated can be customized for the number of questions to use and the range of chapters to pull from.

Skills Used

– Java
– JUnit
– Java Collections
– Eclipse
– IntelliJ Idea

Github: `https://github.com/joe-op/course-project`



Enter the number of questions: 7
Enter the minimum chapter to use: 3
Enter the maximum chapter to use: 8

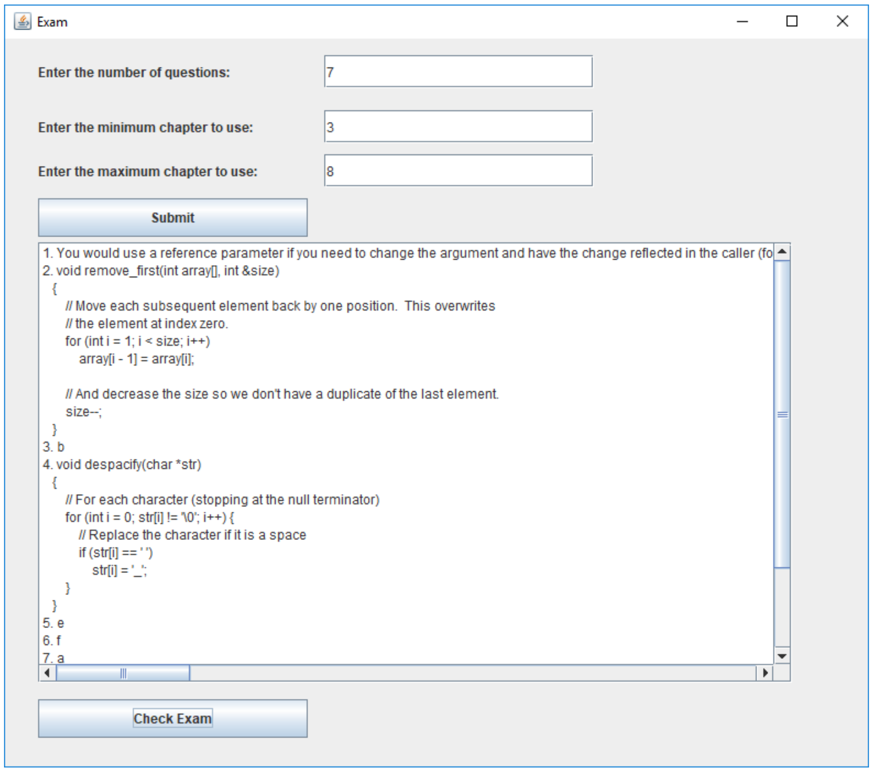
```
Built exam with 7 questions and 21 points:

1. List at least two reasons to use a reference parameter. List at least one reason to use a value parameter. (5 points)

2. Write a function that removes the first element from an array, keeping the order of the other elements. (6 points)

3. Which of the following is not a kind of loop in C++? (1 point)
```

GUI showing generated exam



Enter the number of questions: 7

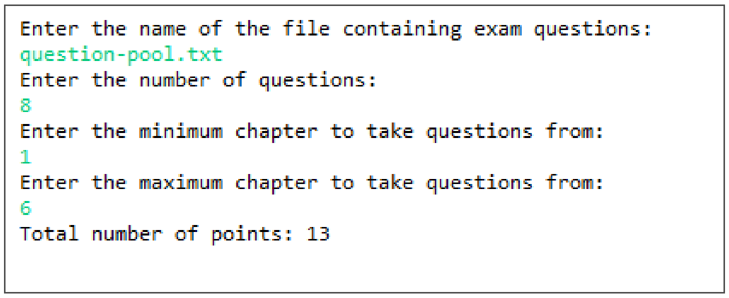
Enter the minimum chapter to use: 3

Enter the maximum chapter to use: 8

```
1. You would use a reference parameter if you need to change the argument and have the change reflected in the caller (fo
2. void remove_first(int array[], int &size)
  {
    // Move each subsequent element back by one position.  This overwrites
    // the element at index zero.
    for (int i = 1; i < size; i++)
      array[i - 1] = array[i];

    // And decrease the size so we don't have a duplicate of the last element.
    size--;
  }
3. b
4. void despacify(char *str)
  {
    // For each character (stopping at the null terminator)
    for (int i = 0; str[i] != '\0'; i++) {
      // Replace the character if it is a space
      if (str[i] == ' ')
        str[i] = '_';
    }
  }
5. e
6. f
7. a
```

GUI showing exam key



```
Enter the name of the file containing exam questions:
question-pool.txt
Enter the number of questions:
8
Enter the minimum chapter to take questions from:
1
Enter the maximum chapter to take questions from:
6
Total number of points: 13
```

Text interface for outputting exam to file

C++ Database Implementation

Data Structures and Algorithms, University of Wisconsin-River Falls

Collaborated on a program that implemented a simple database in C++ with insert, find, and delete features. The database used a binary search tree to find records and included a primary key and a secondary index.

Skills Used

- C++
- Algorithms
- Pointers
- Visual Studio

Github: `https://github.com/joe-op/237-PA4`

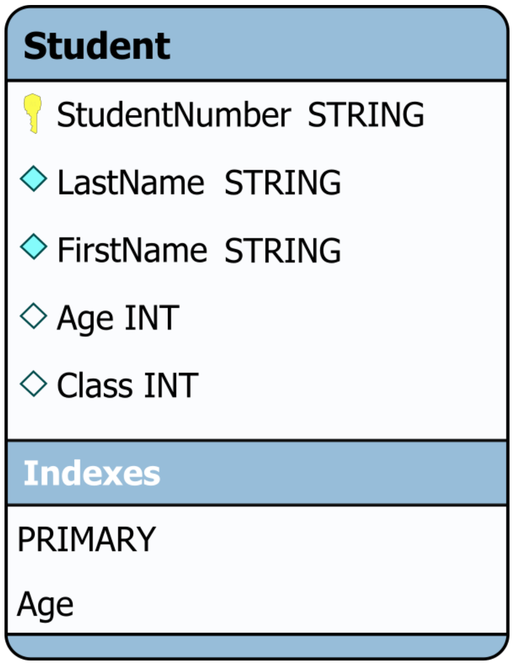
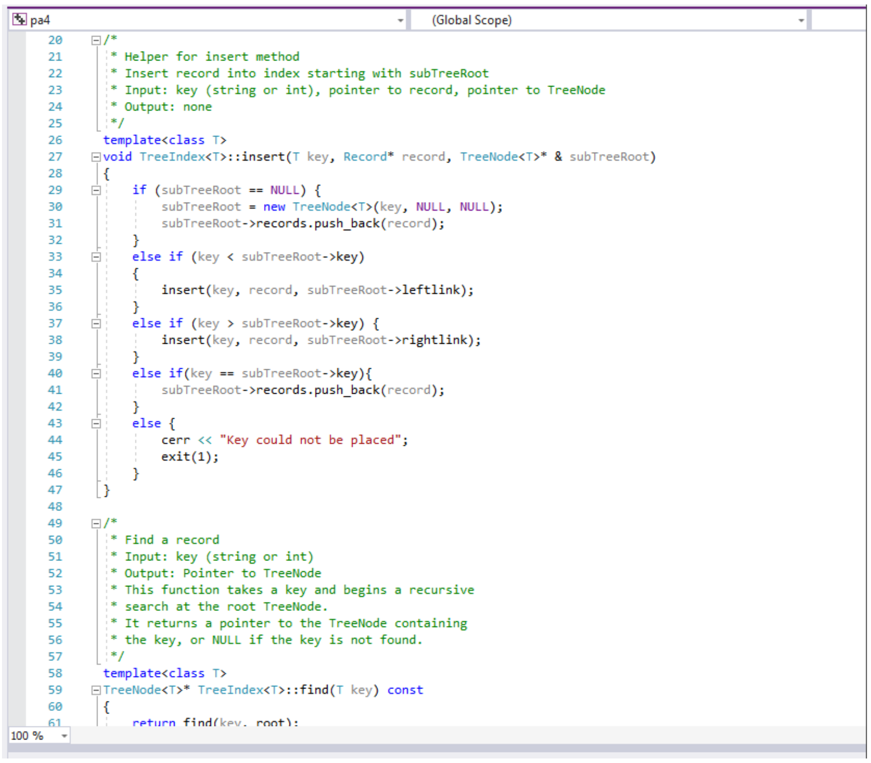


Diagram of the Student table that was implemented using the database system



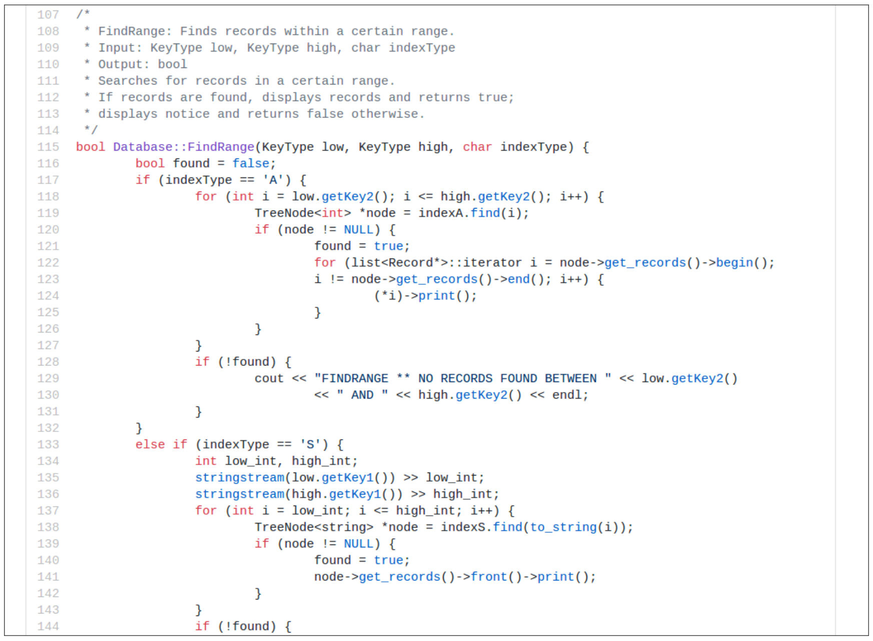
pa4 (Global Scope)

```cpp
/*
 * Helper for insert method
 * Insert record into index starting with subTreeRoot
 * Input: key (string or int), pointer to record, pointer to TreeNode
 * Output: none
 */
template<class T>
void TreeIndex<T>::insert(T key, Record* record, TreeNode<T>* & subTreeRoot)
{
    if (subTreeRoot == NULL) {
        subTreeRoot = new TreeNode<T>(key, NULL, NULL);
        subTreeRoot->records.push_back(record);
    }
    else if (key < subTreeRoot->key)
    {
        insert(key, record, subTreeRoot->leftlink);
    }
    else if (key > subTreeRoot->key) {
        insert(key, record, subTreeRoot->rightlink);
    }
    else if(key == subTreeRoot->key){
        subTreeRoot->records.push_back(record);
    }
    else {
        cerr << "Key could not be placed";
        exit(1);
    }
}

/*
 * Find a record
 * Input: key (string or int)
 * Output: Pointer to TreeNode
 * This function takes a key and begins a recursive
 * search at the root TreeNode.
 * It returns a pointer to the TreeNode containing
 * the key, or NULL if the key is not found.
 */
template<class T>
TreeNode<T>* TreeIndex<T>::find(T key) const
{
    return find(key, root);
```

100 %

The database used a binary search tree to find records



```cpp
/*
 * FindRange: Finds records within a certain range.
 * Input: KeyType low, KeyType high, char indexType
 * Output: bool
 * Searches for records in a certain range.
 * If records are found, displays records and returns true;
 * displays notice and returns false otherwise.
 */
bool Database::FindRange(KeyType low, KeyType high, char indexType) {
        bool found = false;
        if (indexType == 'A') {
                for (int i = low.getKey2(); i <= high.getKey2(); i++) {
                        TreeNode<int> *node = indexA.find(i);
                        if (node != NULL) {
                                found = true;
                                for (list<Record*>::iterator i = node->get_records()->begin();
                                i != node->get_records()->end(); i++) {
                                        (*i)->print();
                                }
                        }
                }
                if (!found) {
                        cout << "FINDRANGE ** NO RECORDS FOUND BETWEEN " << low.getKey2()
                                << " AND " << high.getKey2() << endl;
                }
        }
        else if (indexType == 'S') {
                int low_int, high_int;
                stringstream(low.getKey1()) >> low_int;
                stringstream(high.getKey1()) >> high_int;
                for (int i = low_int; i <= high_int; i++) {
                        TreeNode<string> *node = indexS.find(to_string(i));
                        if (node != NULL) {
                                found = true;
                                node->get_records()->front()->print();
                        }
                }
                if (!found) {
```

Indices were made on StudentNumber and Age

Stock Management Web Application

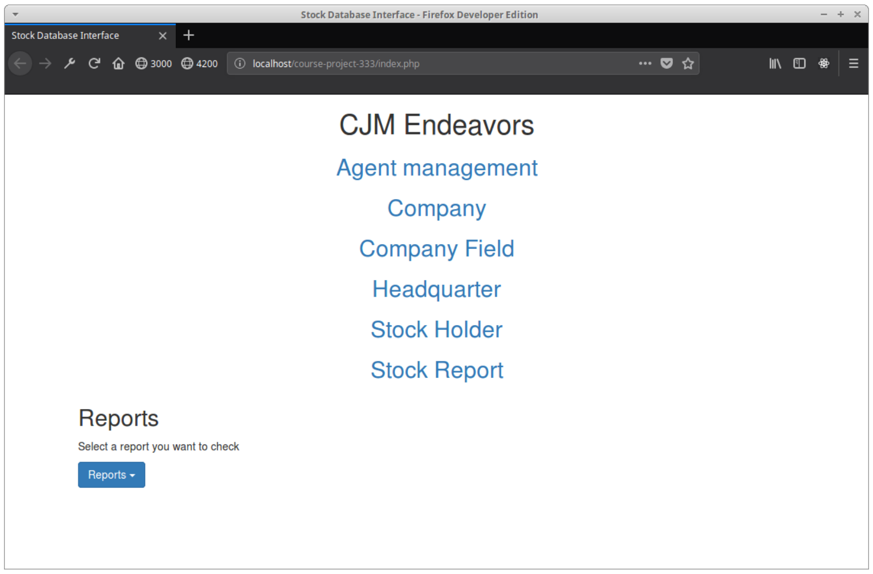
Database Management Systems, University of Wisconsin-River Falls

Collaborated on a web application for internal use of a stock management company. The application was written using PHP. We used SQL queries to retrieve information on agents, clients, companies, and stock reports, and we used the Ruby gem Faker to generate some of the data used to demonstrate the application. The application retrieved and displayed various reports and provided CRUD web forms for entities such as agents, companies, and headquarters.

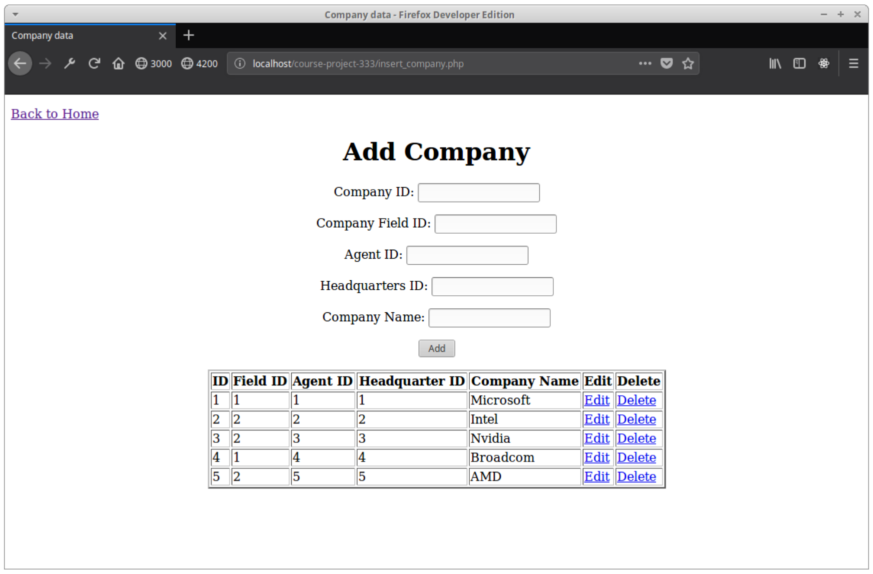
Skills Used

- MySQL
- PHP
- HTML
- CSS
- Apache
- MariaDB

Github: `https://github.com/joe-op/333-CourseProject`



The application provides an interface for managing the company's database. Also provided are various reports built from custom SQL queries


Company data - Firefox Developer Edition

localhost/course-project-333/insert_company.php

Back to Home

Add Company

Company ID:

Company Field ID:

Agent ID:

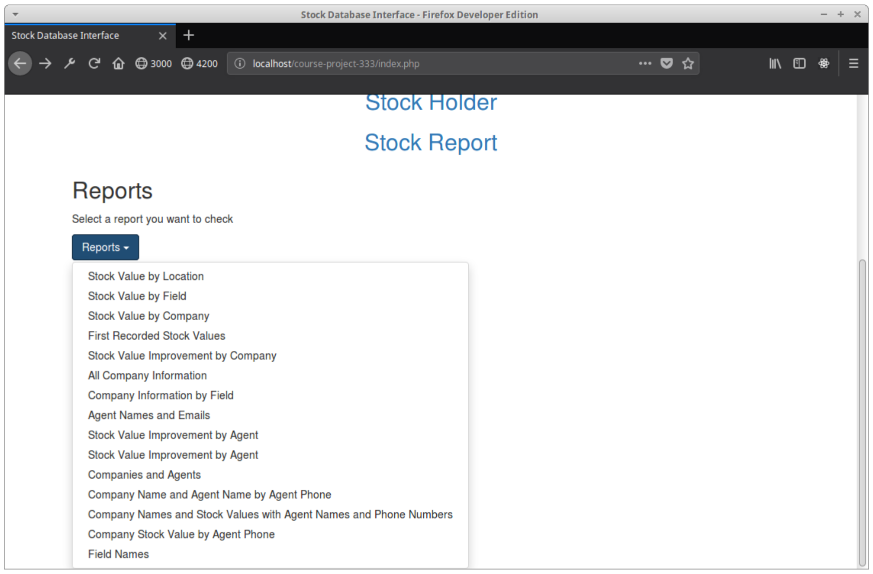
Headquarters ID:

Company Name:

Add

ID	Field ID	Agent ID	Headquarter ID	Company Name	Edit	Delete
1	1	1	1	Microsoft	Edit	Delete
2	2	2	2	Intel	Edit	Delete
3	2	3	3	Nvidia	Edit	Delete
4	1	4	4	Broadcom	Edit	Delete
5	2	5	5	AMD	Edit	Delete

Provides CRUD functionality for records


Stock Database Interface - Firefox Developer Edition

localhost/course-project-333/index.php

Stock Holder

Stock Report

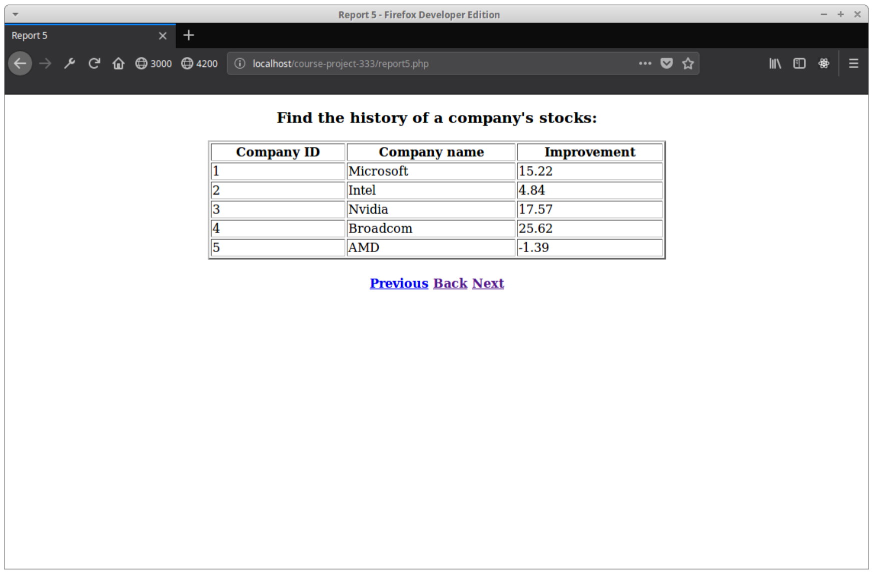
Reports

Select a report you want to check

Reports

- Stock Value by Location
- Stock Value by Field
- Stock Value by Company
- First Recorded Stock Values
- Stock Value Improvement by Company
- All Company Information
- Company Information by Field
- Agent Names and Emails
- Stock Value Improvement by Agent
- Stock Value Improvement by Agent
- Companies and Agents
- Company Name and Agent Name by Agent Phone
- Company Names and Stock Values with Agent Names and Phone Numbers
- Company Stock Value by Agent Phone
- Field Names

Users can access various reports pulled from the database


Report 5 - Firefox Developer Edition

localhost/course-project-333/report5.php

Find the history of a company's stocks:

Company ID	Company name	Improvement
1	Microsoft	15.22
2	Intel	4.84
3	Nvidia	17.57
4	Broadcom	25.62
5	AMD	-1.39

Previous Back Next

A report showing the improvement in each company's stocks

Restaurant Website

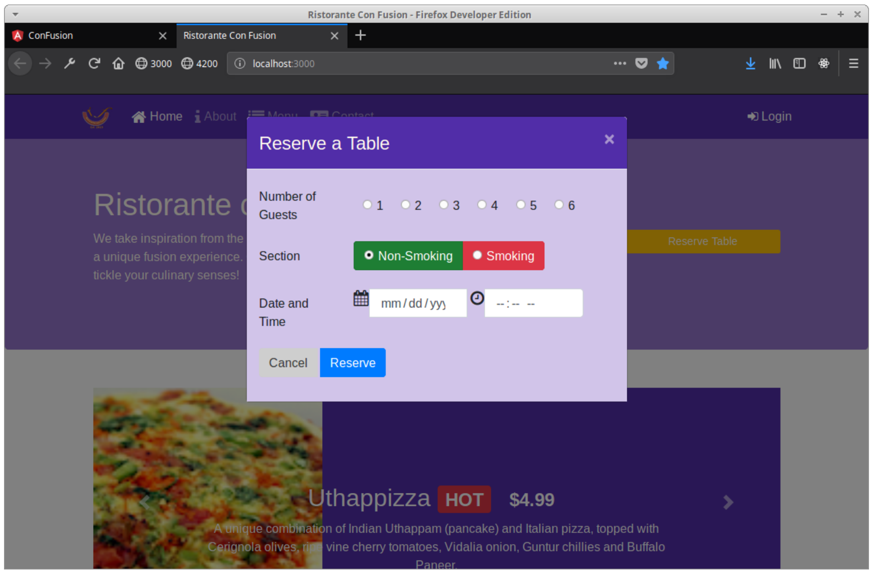
Full Stack Web and Multiplatform Mobile App Development
The Hong Kong University of Science and Technology through Coursera

A website for a restaurant, including the restaurant's menu, information on the corporate leaders, and contact information. In the specialization's first course – Bootstrap 4 – we built the user interface using Node.js with Bootstrap and FontAwesome modules. Features included modals for logging in and reserving a table, a responsive navigation bar that collapses into a dropdown menu for mobile devices, a carousel on the home page for featuring items, and stylized contact and social media links.

In the second course we are redesigning the site using Angular. The UI framework has been switched from Bootstrap to Material Design, and elements such as menu items and corporate leaders have been rewritten as TypeScript objects provided by services, allowing the HTML templates and the objects to be more easily updated and expanded.

Skills Used

- JavaScript & TypeScript
- Node.js & NPM
- Angular Framework
- Chrome & Firefox developer tools
- CSS preprocessors
- Gulp & Grunt taskrunners
- HTML5 & CSS3
- Bootstrap
- Angular Flex Layout
- Angular Material Design


Ristorante Con Fusion - Firefox Developer Edition
ConFusion
Ristorante Con Fusion
3000
4200
localhost:3000
Home
About
Menu
Contact
Login
Ristorante c
We take inspiration from the
a unique fusion experience.
tickle your culinary senses!
Reserve Table
Uthappizza HOT $4.99
A unique combination of Indian Uthappam (pancake) and Italian pizza, topped with Cerignola olives, ripe vine cherry tomatoes, Vidalia onion, Guntur chillies and Buffalo Paneer

Reserve a Table

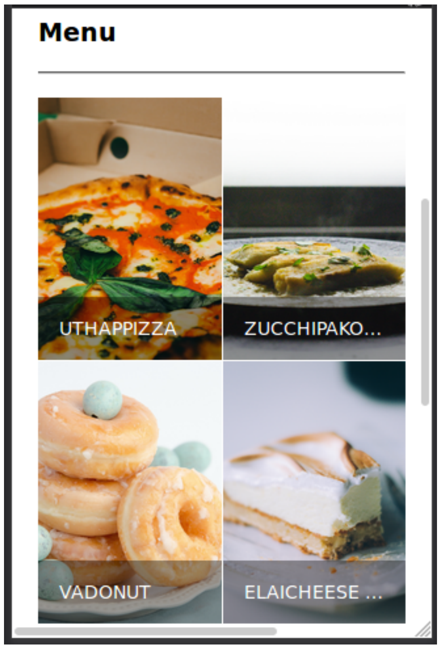
Number of Guests: 1 2 3 4 5 6

Section: [✓ Non-Smoking] Smoking

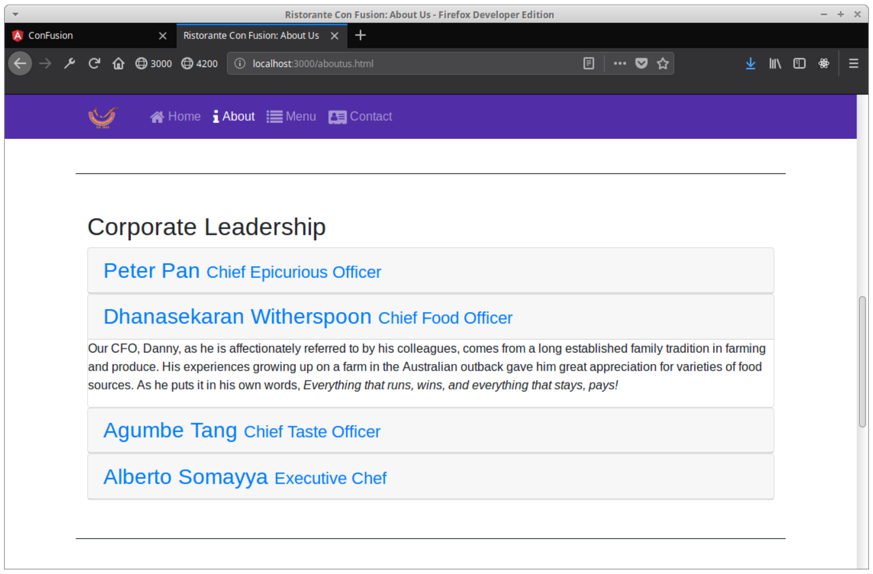
Date and Time: mm / dd / yyyy --:-- --

Cancel Reserve

A modal for making a reservation



The mobile view of the restaurant's menu


Ristorante Con Fusion: About Us - Firefox Developer Edition
ConFusion
Ristorante Con Fusion: About Us
localhost:3000/aboutus.html
Home
About
Menu
Contact

Corporate Leadership

Peter Pan Chief Epicurious Officer

Dhanasekaran Witherspoon Chief Food Officer

Our CFO, Danny, as he is affectionately referred to by his colleagues, comes from a long established family tradition in farming and produce. His experiences growing up on a farm in the Australian outback gave him great appreciation for varieties of food sources. As he puts it in his own words, *Everything that runs, wins, and everything that stays, pays!*

Agumbe Tang Chief Taste Officer

Alberto Somayya Executive Chef

A Bootstrap accordion showing information about the corporate leaders

Self-assembly Research

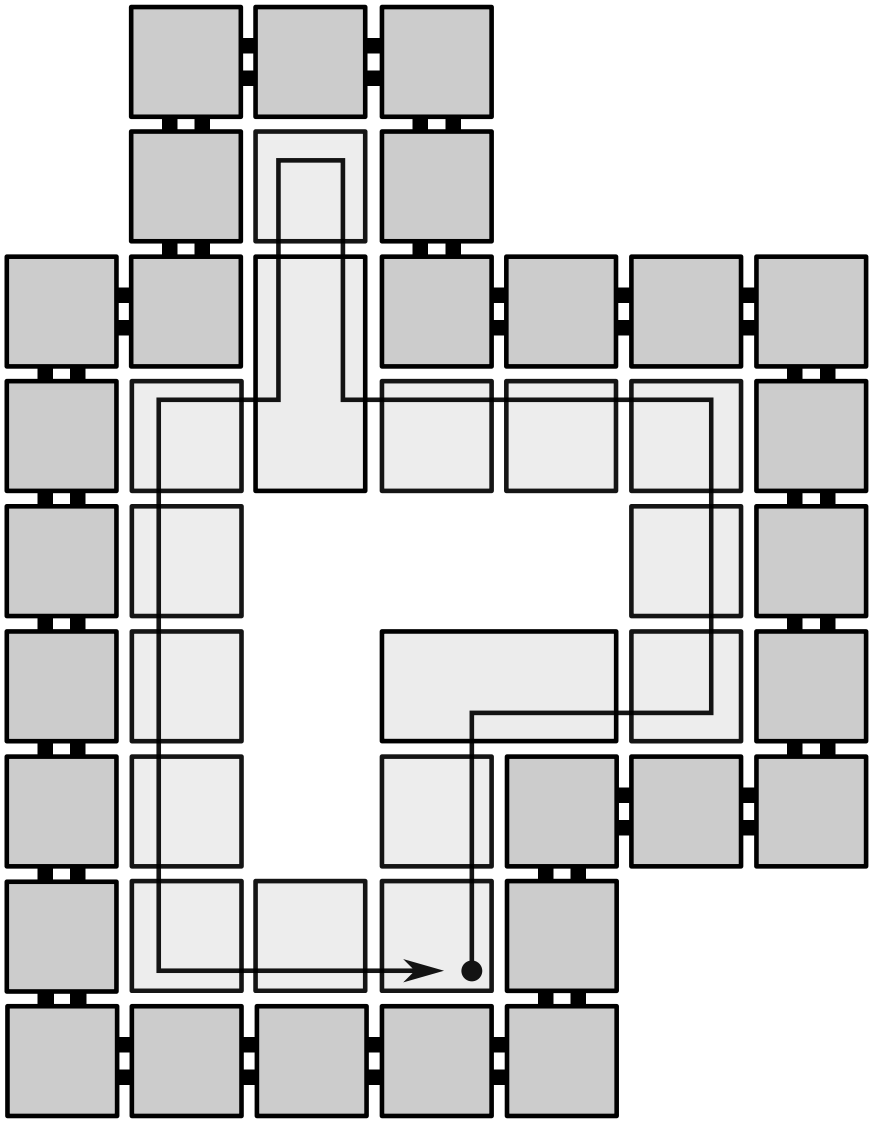
University of Wisconsin-River Falls

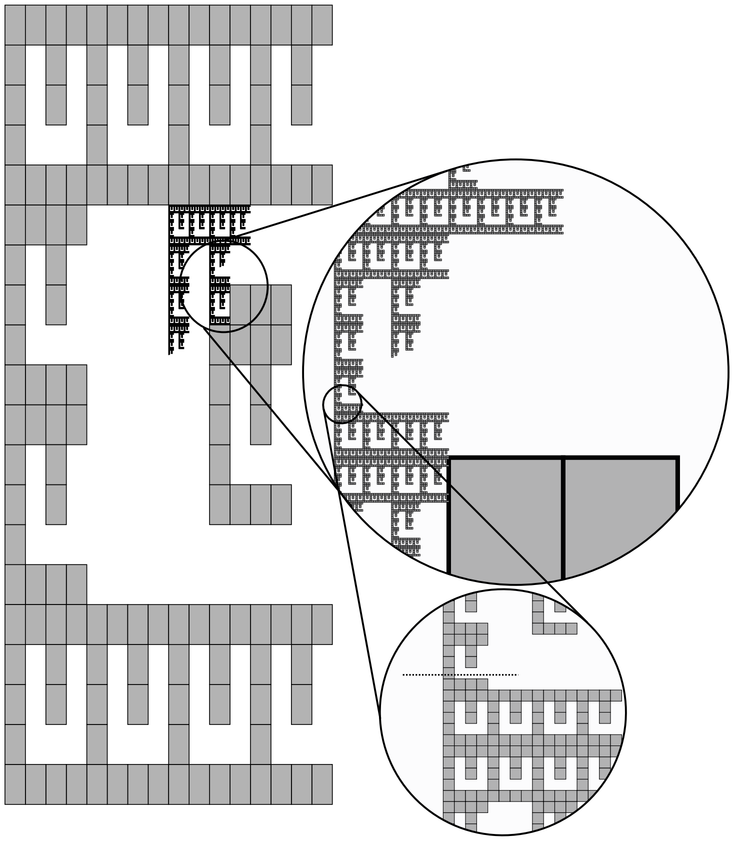
The field of self-assembly is motivated by the fact that DNA molecules can be designed in such a way that, when they are combined, they will autonomously form into “assemblies” whose structure is determined by the design of the molecules. Research in this field ranges from wet-lab work with DNA, to designing mathematical models based on the properties of DNA, to abstract mathematical research exploring the limits of these models. Our research in the last category concerned the possibility or impossibility of certain types of constructions in the 2HAM and STAM models of self-assembly. One result [1] shows that any fractal that belongs to a class of fractals we call “4-sided” can be strictly self-assembled in the 2HAM, while the same is not true for “3-sided” fractals (that is, there exists a 3-sided fractal that cannot be strictly self-assembled in the 2HAM).

A comprehensive introduction to the field of self-assembly can be found at the self-assembly wiki: `http://self-assembly.net`.

Skills Used

- Mathematical Modeling
- Mathematical Writing
- Figure design with Inkscape
- Typesetting with LaTeX


A figure showing one of the steps of shape replication in the STAM.


A figure used to show a 3-sided fractal that can grow incorrectly in the 2HAM, hence cannot be strictly self-assembled.

References

[1] Jacob Hendricks and Joseph Opseth. “Self-Assembly of 4-sided Fractals in the Two-handed Tile Assembly Model”. In: *Unconventional Computation & Natural Computation (UCNC) 2017, University of Arkansas, Fayetteville, Arkansas, USA*. URL: `https://arxiv.org/abs/1703.04774`. June 5-9, 2017.