

# SHANGWAY CALVIN HSU

☎ +1 408 368 9917  
✉ shangway.hsu@gmail.com  
🌐 <https://github.com/ShangwayHsu>  
🌐 <https://www.linkedin.com/in/shangwayhsu>

## EDUCATION

**University of California, San Diego**  
B.S. Computer Science, Minor - Cognitive Science

Expected Graduation: 2018  
Cumulative GPA: 3.74 - Major GPA: 3.89

### Related Coursework:

CSE 100: Advanced Data Structures and Object-Oriented Design	CSE 101: Design and Analysis of Algorithms
CSE 160: Introduction to Parallel Computing	CSE 110: Software Engineering
CSE 141: Introduction to Computer Architecture	CSE 132A: Database System Principles

## SKILLS

**Proficient in:** Python, Java, C++, REST, Swift/Objective-C, iOS/Xcode, Unix, Git

**Experience in:** C, HTML, CSS, JavaScript, SQL, Networking, Angular 2, Ionic 2, TypeScript, databases

## EXPERIENCE

### SSL – Space Systems Loral

Palo Alto, CA

#### Software Engineer Intern

Jun 2016 – Sept 2016

- Developed an internal RESTful Web Server using various Python frameworks such as Flask and Tornado with the goal of distributing data to other programmers in a standardized manner.
- Implemented REST styled realtime data streaming with WebSockets on both server and client sides.
- Created front-end web interfaces using Javascript/CSS/HTML to generate queries and plot historical and streaming data.
- Gained thorough understanding of networking through the development of RESTful APIs and socket connections.

### SLAC (Stanford Linear Accelerator Center) National Accelerator Laboratory

Menlo Park, CA

#### XFEL Software Intern

Jun 2015 – Aug 2015

- Implemented an optimization algorithm in C++ called Particle Swarm to find an optimal configuration for SLAC's LCLS (particle accelerator) in order to form a coherent electron beam.
- Used the Python MATLAB library for data visualization of the optimization results from the accelerator simulation.
- Made improvements to the algorithm responsible for the electron beam bandwidth calculation.

## PROJECTS

### Personal Website – <https://shangwayhsu.github.io>

Continuous

- Implemented personal website to showcase projects through the use of Bootstrap/HTML/CSS/JavaScript.
- Single-page website with scrolling animations and dynamic background.
- Responsive page with mobile support such as collapsible navigation bar to fit devices with smaller screen sizes.
- Email Contact form powered by Formspre.io.

### Free & For Sale 3.0 App – Web App Development

Oct 2016

- Developed using Angular 2 TypeScript framework paired with Ionic 2 and Firebase as database.
- Created majority of the front-end of the app such as the item-listing-details, inbox/messages, home-page item displays, etc.
- Implemented services that communicated with the database for ratings and messages service.
- Worked in a fast-paced 7 person team practicing agile software development.

### Instagram Replica App – iOS App Development

May 2016

- Developed using Parse as a backend cloud database in order to create custom data entries such as user logins/passwords, followers, posts, etc. Deployed using Heroku cloud platform.
- Extensive transfer of user data between application and the Parse server using Swift.
- Obtained a comprehensive grasp on application development in iOS/Swift/xCode.

### To-do List App - iOS App Development

Feb 2016

- Implemented To-do list in Swift and Xcode for iPhone as an entry point to iOS development.
- Extensive use of StoryBoard in Xcode for UI elements and navigation control. Including multiple views.
- Functionality includes adding new items with title and a short description, and deleting/editing existing fields.

### Autocomplete - C++

Jan 2016

- Used Multiway Trie to implement a dictionary capable of Autocomplete.
- Use of Priority Queue to store relations between nodes to decrease autocomplete time at the cost of space.
- Multiway Trie guarantees  $O(L)$  in `find()` and `autocomplete()`, where  $L$  is length of the longest word.

### Unix Argument Parser - Java

Apr 2015

- Implemented a general command-line argument parser library following the Unix single-character option conventions and also the GNU long-form option conventions by using Object Oriented Design.
- Created using data structures such as Hashmaps and Sets for maintaining an organized OOP structure.
- Uses the fluent-interface style described in <https://www.martinfowler.com/bliki/FluentInterface.html>.