PEZANNE KHAMBATTA

www.github.com/pkhambat1 ◇ www.pezanne.com ◇ www.linkedin.com/in/pezanne
San Francisco, CA ◇ (+1)857-294-2105 ◇ pkhambatta@dons.usfca.edu

EDUCATION

University of San Francisco, CA *August 2019 - Present*
MS, Computer Science
Member of MAGICS - machine learning research lab for detecting bias in news

Rhode Island School of Design, Providence, RI *September 2013 - June 2017*
BFA, Industrial Design GPA: 3.6

SKILLS

Languages / Databases	Java, C#, Python, MySQL
Web Technologies	HTML, CSS, JavaScript, JQuery, AJAX, Bootstrap
Tools / Frameworks	ASP.Net, Flask
Other	Azure, Heroku, Razor Syntax, Regex

WORK EXPERIENCE

Polymerupdate, Mumbai, India January 2019 - August 2019
Dot Net Developer (Intern)

· Responsible for developing code for API and web applications in .NET MVC Framework, using C# for backend, HTML, CSS and JavaScript for frontend and MySQL for database operations

PROJECTS

ContentHub ASP.Net Core *2019*
Admin application that grants users (employers) different levels of access based on their role. The project uses a relational mapping of users to roles to permissions to rights, which allows for modular, real-time assignment and reassignment operations of authorizations.

pezanne.com and HTML Generator HTML/CSS/JavaScript, ASP.Net Core *2019*
pezanne.com - clean and trendy looking website designed to display a selection of favored industrial design projects, self-written Medium articles and an interactive section for curated music. HTML Generator - web app used to simplify content creation for pezanne.com by converting text and image input into raw HTML.

Hangman AI Bot Java *2019*
Bot that plays Hangman by generating a series of guesses based on the feedback it gets from the game. With the help of a 20,000 word dictionary, it uses Regex pattern matching to determine a list of possible words for each hidden word. It scores each letter (a-z) on a complex metric, and chooses the highest scoring letter. Scores an average of 0.9 incorrect guesses for a given list of 10 phrases.

Lexer, Parser and Bytecode Interpreter (BCI) Java *2019*
Mock compiler that takes (simplified) assignment statements as input and stores variables into "memory". Lexer tokenizes the string, Parser syntactically validates the list of tokens and the BCI generates "bytecode" which will eventually be read to perform all math and memory storage procedures. Project implements Shunting Yard algorithm to perform math calculations.