NIKHIL PRASHAR
1221 East Genesee St, Syracuse, NY 13210
(914)733-8184 | www.nikhilprashar.us | nnprasha@syr.edu | www.github.com/nnprasha | https://www.linkedin.com/in/nikhil-prashar

OBJECTIVE: Actively looking for Full Time Jobs/Internships as a Software Engineer.

EDUCATION

Master of Science, Computer Engineering **GPA: 3.889/4.0** **May 2018 (Expected)**
Syracuse University, NY
Graduate Teaching Assistant at Syracuse University.

Bachelor of Technology, Information Technology **GPA: 3.78/4.0** **May 2016**
Manipal University, Manipal Institute of Technology, India

TECHNICAL SKILLS & INTERESTS

Languages and Frameworks: Java, C, C++, C#, SQL, HTML/CSS, Xaml, JavaScript, AngularJS, Angular 2, Xamirin, SpringMVC, Swift, Entity Framework, ASP .NET.
Software: Eclipse, Netbeans, XCode, Visual Studio, Git, MySQL, JIRA, Microsoft Visio, SQL Server Management Studio.

INTERNSHIPS

Software Engineering Intern **June 2017 - Present**
Slalom Consulting, Boston, MA

- Worked with a team using Agile Development cycle and developed a Web Application and a Mobile Application for Slalom Consultants focused on making entries and approvals for expenses (for reimbursements) quicker and easier. The Web Application was developed using **Angular 2**, **ASP .NET**, **Entity Framework**. The Mobile Application was developed using **Xamirin** and **Prism Frameworks**.

Graduate Analyst Intern.
Blackrock Services India Pvt. Ltd, Gurgaon, India **January 2016 - June 2016**

- In a team of three, we were assigned to work on a web application that would handle Financial Information Exchange (FIX) Message Configurations. The main idea was to automate the process of configuring and saving FIX Message configurations to the database, in a user-friendly way rather than manual database entries.
- For this we developed a web application using **SpringMVC** Framework and **AngularJS**. Back-end was purely based on **Java**. The application was thoroughly tested using **JUnit** and **DBUnit**.

ACADEMIC PROJECTS

Remote Code Publisher (C++, HTML, CSS, JS), Syracuse University **March 2017 – May 2017**
Project: Developed means to display source code files as web pages with embedded child links. Each link refers to a code file that the displayed code file depends on. Each file to be published is a C++ source file. Our publisher generated, for each of these, an HTML file, with most of the contents drawn from the code file. Implemented a local client, and communication channel that supports client access to the publisher from any internet enabled processor.

Type-based Dependency Analysis (C++), Syracuse University **February 2017 – March 2017**
Project: Developed and tested a type-based dependency analyzer that uses the analysis machinery based on extracting lexical content from source code files and building a Type Table for the same. In stage 1, lexical analysis was performed over a set of files in the repository and a Type Table (container class that stored type information) was constructed based on the types (classes, structs, enums, typedefs, and aliases) that were inferred from the analysis. In stage 2, dependency analysis was performed for each file in a specified file collection and logs all other files from the file collection on which they depend by making use of the Type Table constructed in stage 1.

Key/Value Database (C++), Syracuse University **January 2017 – February 2017**
Project: Developed a NoSQL In-memory database to handle massive data collection and analysis and using XML support to persist the database. A Template class was implemented that provided a Key/Value in-memory database with each value consisting of an item name, category name, text description, time-date recording the written time to the database and a list of child relationships with other values. Functions were included to support complex queries to the database (union/intersections of two or more keys).

Test Harness (C#, WCF, WPF), Syracuse University **August 2016 – December 2016**
Project: Developed an Automation Testing Tool based on a Client Server Architecture to automate integration testing. The automated test tool was made to run a specific set of tests on multiple packages (restricting individual tests to a separate child app domain). Each test execution runs a test driver on a small set of packages, recording pass status and logging execution details. Test requests were submitted to the Test Harness via a request message naming one or more test driver executions.