# Software Requirements Specification

for

# Autocomplete

Version 0.1

Prepared by
Palash Aggarwal
Srishtee Kriti
Satyam Gupta
Abhi Bothera

Team Autocomplete

October 22, 2018

# Table of Contents

# 1. Introduction

## 1.1 Purpose

To build a generic, multi language (natural or programming) autocomplete tool available for easy download and functional across operating systems. This SRS describes the Proof of Concept of this product, namely a CLI tool to showcase the capabilities of the autocomplete tool. In effect, this SRS details the data structure being employed, software dependencies, probabilistic model employed, and structure of data to be provided.

## 1.2 Document Conventions

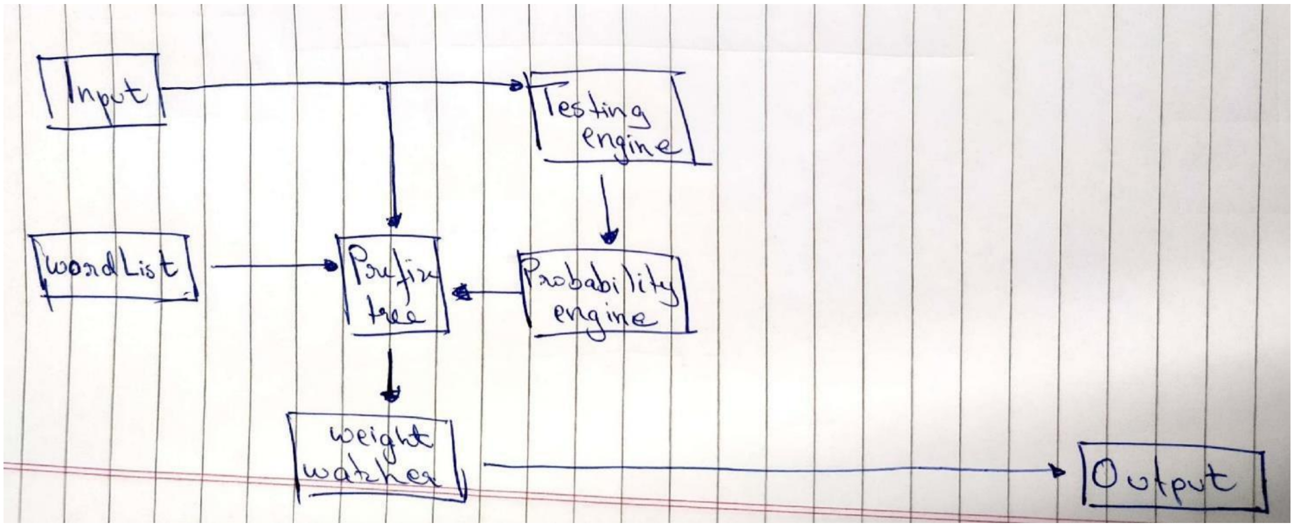
None

# 2. Description

## 2.1 Product Perspective

The project stemmed from a requirement of understanding and implementing tries and prefix trees. It has now evolved to being a cross platform service with users creating and sharing tries for autocomplete with others over a simple to use user interface in its entirety. This SRS covers the proof of concept for the mentioned product and only a simple implementation in one natural language (Hindi or English), a CLI tool for a demo of autocomplete, and if time permits, more features such as a python package to install autocomplete, user accounts, and a central repository of dictionaries and tries in multiple languages – natural or programming.

In its entirety, the *generic autocomplete* will be a self contained unit composed of various modules. Please see the diagram below for the major components of the entire project.



## 2.2 Product Functions

At the bare minimum, the user will have a CLI tool to play with the autocomplete where they type in sentences, while suggestions appear for the completion of the word, based on a probability of usage.

### 2.3 Operating Environment

Bash shell

Python 3.5+

MySql [for logins]

### 2.4 Assumptions and Dependencies

The only assumption is that the python packages incorporated will have backwards compatibility in their updates. This is a fair assumption as the norm in the developer community is to support a release for multiple years.

## 3. External Interface Requirements

### 3.1 User Interfaces

The project will consist of a UI on the command line terminal with the user typing in an editor and a bottom bar showing probabilistic suggested completions. The login of the user, if implemented, would be a CLI.

### 3.2 Hardware Interfaces

The requirement to run this bundle is the existence of an Operating System capable of supporting the Operating Environment, such as, macOS, Windows7, Windows10, Ubuntu 16.04, etc.

### 3.3 Software Interfaces

TBD – Project still in conception phase. However, the dependencies would not exceed beyond the scope of packages on PyPI and will be programmatically installed by our product.

### 3.4 Communications Interfaces

None

# 4. System Features

## 4.1 Probabilistic Prediction Model

### 4.1.1 Description and Priority

This is a high priority feature. Essentially a frequency based prediction model, this will provide better word prediction over longer periods of times as it learns from usage.

### 4.1.2 Stimulus/Response Sequences

As the user types in characters, their input goes into the testing engine which in turn feeds the probability engine, updating the weights as required. The response would be judged by change in suggestions for the user.

### 4.1.3 Functional Requirements

This feature’s only requirement is that the user has our product installed on their machine.