TecInASec Software Requirements Specification

# Introduction

## Purpose

The purpose of the software requirements specification for TecInASec is to inform the intended audience about the plans that we had for the software and what we have done with those plans. The intended audience for the software TecInASec is users that are looking to buy a new computer, and wish to easily search for a name, brand or specify a price range to find the websites that have the best offers on the computers the user is searching for.

## Scope

TecInASec is a computer comparison price application used to search and compare a range of systems by parameters specified by the user. The information about the computers are stored in an SQL database on an online administration only accessible server open to the use of the application strictly for security reasons. Using an online server means the information that is stored can be collected from any device with an internet connection. The original concept was based on a python application with an add-on called Tkinter. Used for graphic user interface design the add-on allowed for code to be processed into user friendly visuals by applying order and function to the data collected. After some development with this original concept the team came to a conclusion that this would be used as research for the structure of a final product that will be applied to a web platform by using PHP, HTML and CSS. Queries to retrieve the information get processed by JSP and JSTL. This application of our program means we have creative control allowing the final product to be aesthetically pleasing and the functionality to be sound.

## Definitions, acronyms and abbreviations

**TecInASec**: The name defines the intention behind the software, the cheapest computers found for the user in just a couple of seconds.

**TIAS**: TecInASec, the software name.

**Basic** and **Advanced page**: The two main search pages in the application.

**Results page**: The main page for displaying results to the user.

**Tkinter**: The native GUI library for python we used for developing the GUI.

**Requests** and **Beautiful Soup**: Libraries used for the scraping process in the python code.

**Assets** and **Elements**: objects in the program that make up the look of the applications.

**CSS:** (Cascading Style Sheet) is the markup language that is used to format and style the TIAS website.

**MySQL:** The SQL database we use to hold all of the information scraped from the website we are scraping.

**PHP:** (**P**ersonal Home Page: **H**ypertext **P**reprocessor): Is the server-side programming language we are using to create dynamic content on our website, and providing communication between the MySQL database and the website.

**APPLICATION/PROGRAM:** When we write application or program, we are referring to the python code, php code and the html/css combined into our product: TecInASec.

**Web Server:** Currently a domain and webhotel hosted by Sean Traynor. The URL is [www.dbblts.com](http://www.dbblts.com). The server allows us to run serverside languages such as PHP, which will be used to provide communication with the database and perform calculations etc.

## References

## Overview

This software requirements specification is split into sections. At this point you would have already read the introduction to the document which is made up of a title (introduction) and sub sections labelled in ascending decimal value. This ascending decimal value format can be seen consistently throughout the document with appropriate headings. Section 2 will explain the overall description and will be followed by section 3 which talks about the specific requirements. The following subjects are appendixes and index.

# Overall Description

## 2.1 Product perspective

## 2.2 Product functions

List: [Write into sentences!]

* Search algorithm
* Displays computers based on query
* Adaptive user interface functions
  + Creates queries based on what fields have content

## 2.3 User characteristics

TIAS is made to be as simple as possible to use, there is not any extensive technical knowledge required, nor experience. The optimal user group would be users that have had some previous experience with using entry fields to fill in information, and users that understand that the software isn’t written by professionals, as there could occur errors if the user were to input something that we did not take into account whilst developing the software.

## 2.4 Constraints

## 2.5 Assumptions and dependencies

# Specific requirements

# Appendixes

# Index

An SRS outline

Introduction

Purpose: Provide a short statement of the purpose of the SRS and state the intended audience.

Scope: Identify the software by name. Explain what the software will do. Describe the application of the software.

Definitions, acronyms, and abbreviations: Give any definitions required to read the SRS and provide any acronyms and abbreviations used in the SRS.

References: Provide a list of references used in the SRS.

Overview: Provide an overview of the SRS and explain how it is organised.

Overall description

Product perspective: You should describe the overall system clearly stating if it is an independent system or part of a larger system. A block diagram of the product may be useful.

Product function: Provide a summary of the functions the software will perform. Organise the functions such that they are clear and understandable by the identified user(s).

User characteristics: Describe the characteristics of the users of the product indicating for example the required education level, experience, or technical knowledge. Be sure to include all the user types.

Constraints: List any constraints such as hardware limitations, safety and security.

Assumptions: List assumptions such as operating system requirements on the hardware to be used for the software product.

Apportioning of requirements: Provide detail regarding any delay of specific requirements to later versions of the software.

Specific requirements

External interface requirements: User interfaces, hardware interfaces, software interfaces, communications interfaces.

Functional: State all the fundamental actions of the software product being developed. Provide information on the user type to whom the functional requirement applies. For example "the customer" will be able to download the software from the app store, and "the administrator" will be able to create an account. List all the functional requirements providing sufficient detail to describe the function. Review the SRS document from the University of Gothenburg, provided in the chapter 3 initial reference material folder. Research SRS functional requirements.

Non-functional: These can be performance indicators such as the capacity, response time, or fault recovery time. They could be design constraints such as the hardware to be used or the memory requirements. They could be software system attributes such as the adaptability, availability, reliability, usability, or security.