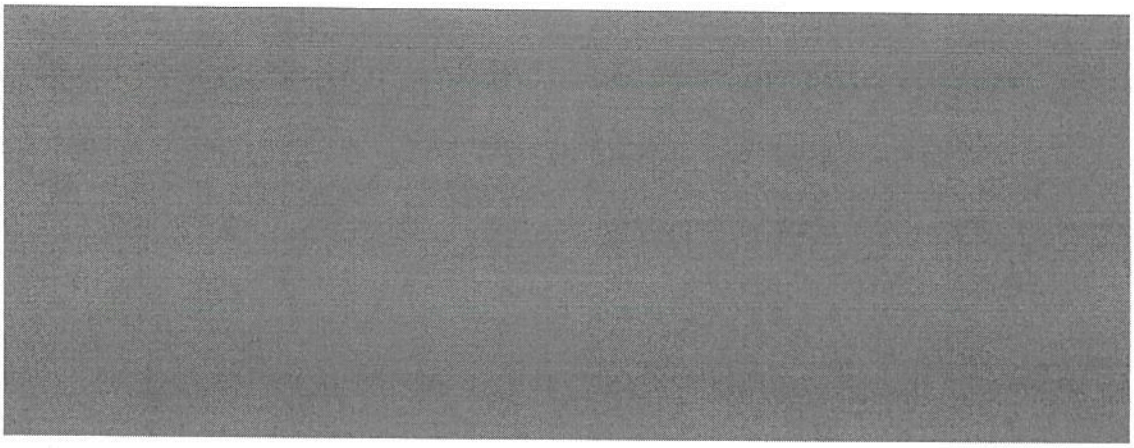

PAT - Phase 1

System Specification Document

This document's purpose is to outline the features of your programming project. It must contain a summary of the problem your software will attempt to address, a list of features (including the data that your program will store), a description of the user interface and help available, and a detailed description of the hardware and software requirements This section details WHAT your program will do. It does not say HOW.

October 2012

Contents

Summary

This section needs to contain a description of the problem area you are trying to address. This will be similarly worded to your feasibility study's "problem definition". It should encompass all aspects of the problem.

E.g. "My program aims to address the need for organisation X to perform task Y. When Person A needs to do Y then go to Person B who receives form 1A from them and records the following details..."

This section may also describe how your program will aid in solving this problem but this section should not be solely about how your program should work. It's main purpose is to give some background and justification as to why your program needs to be written.

This section should be at most two paragraphs.

Specifications of Program Function

This section contains a full and comprehensive list of the features and functions that your program will have. This section is best done in point/bullet format and it should include some detail of the functions.

Remember that simply saying "Add Customer" is not really sufficient. You should include a description such as "Add Customer – ability to add a customer to the database by supplying their name, contact details and payment method". This allows the reader of the document to see exactly what you intend to do for adding a customer.

This section needs to be an exhaustive step by step list of what the program needs to do. Each step will later be something that you need to code or create. Keep your sentences simple so that each step is something that is easy to code and achieve.

For example:

- *Display log on screen*
- *Input user name and password*
- *Check if password is valid*
- *If it is valid, open Game Screen*
- *If it is not valid display error message*
- *At Game Screen show show fighting characters*
- *Allow user to use keys to move character*
- *If key is up arrow move character up*

Specifications of Help

In this section you must detail what help will be provided for your program. A good guideline is to make sure that you cover internal and external help.

Internal Help: this is help that is coded into your program. It could be a help menu, tool-tips, tip-of-the-day, labels indicating format of dates or even useful error message and troubleshooting tips.

External Help: this consists of external documentation in the form of a printed or electronic (PDF, HTML) user manual which gives details on how to perform each of the functions.

Remember that you can also include information on context sensitive help (help that is specific to a particular section of your program) or details of a recorded video tutorial.

Specification of Data Storage

This section details what data your program will store. This is not a detailed section for your database design.

One way of writing this section would be to divide your program up into distinct parts and list the data that will be stored for that particular part. For example under the heading "Clients" you might have "First name, surname, date of birth, postal address, telephone number" etc. Remember that the more detail you give the more marks you will receive.

How can I give more detail? Consider indicating how your data will be stored exactly. An example of this is "Postal Address" will you store this as three separate lines "Address Line 1", "Address Line 2" and "Postal Code" or as one big line with a separate postal code? A similar concept applies to telephone numbers and area codes.

In addition to what you are going to store mention briefly whether you are considering using a database or text file. Remember that you need not describe in detail what your field types or names will be but rather just give an overview of perhaps what you will store in a database and what you will store in a text file (if you are using one)

Hardware and Software Specifications

This section may seem trivial but it is important. Most importantly is to be accurate and specific.

Hardware: Make sure your hardware specifications are the minimum hardware requirements. Also make sure they are realistic. Will the user realistically need 10GB free space? Is 600MB RAM a realistic amount of RAM? Consider units and modern computer specifications

Software: Aside from specifying the obvious operating system you need to make sure your other software requirements are realistic? Will the user really need JGrasp or just the Java Run Time libraries? If we compile our Delphi program will the user really need Delphi installed to run the program? Is Microsoft Access needed or just the ODBC driver? All of these details are important.

Example: A program to manage a school.

The four word documents that accompany your code are outlined in greater detail here with examples taken from a number of different sources.

Document 1 : Project Specification Document
Overview of your project. Summary of the problem your software will attempt to address, a list of features, the data your program will store and description of the hardware and software requirements. This section details **what** your program will do, not how it will do it.

1.1 Cover

- Performance Assessment Task
- Project Specification Document
- Your full name
- Your grade
- The year.

1.2 Table of contents

1.3 Summary
Summary of the problem your software will attempt to address, a list of features, the data your program will store and description of the hardware and software requirements. This section details **what** your program will do, not how it will do it.

1.4 Specifications of Program Function
Full and comprehensive list of the features and functions that your program will have; suggested format - bullets. Each bulleted point will turn into something that you need to create or code.

Some ideas for the school management program

- Opening screen – may have the date and who is the duty teacher
- Login screen – allows the administrator to log into the confidential information
- Display all the information of a particular student
- Add a new student into the database
- Delete a student from the database – also the mother from the mother table and the father from the father table according to the unique ID
- Update a student's details
- Search for a student using only part of their name
- Write a student's details to a text file that can be printed out.
- Search for students in a particular grade
- View the student's parents ie mother and or father
- View the student's register teacher
- Help. Displays information about the mail GUI which is read in from a text file

1.5 Specifications of the Graphic User Interface

Opening GUI

- Login – requires name and password – goes to Main GUI
- Exit button closes the program

Main GUI – some ideas that would be appropriate

- New student
  - Name, phone number, address, medical issues, grade, house, fathers name, mothers name, email, register teacher etc
  - Update button writes all data to the database
  - Clear button clears all fields
  - Back button goes back to the Main GUI
- Update student – includes information about parents
  - Update button writes to the database
  - Clear button clears all the fields
  - Back button goes back to the Main GUI
- Delete student
- View all students
- Search for student
  - Search by name
  - Search by grade
  - Search by ID
  - Back button goes back to Main GUI
- Search for parent
  - Search by name
  - Back button goes back to Main GUI
- Help button contains information on the functionality of all the buttons

1.6 Specification of Help

Program is user friendly and supplies error or success message on completion

- Login. If username and password is incorrect an error message is displayed.
- Help button on the Main GUI explaining the purposes of the button (taken from a text file)
- New Student. Error messages if any of the fields do not have data. Success notice will be displayed on successful addition

1.7 Specification of Data Storage Tables in the Access Database

Student – Table – some ideas

Field with example	Datatype
ID	AutoNumber
First Name. Eg "John"	String
Last Name. Eg "Edwards"	String
Address1. Eg "12 Ruby Rd	String
Address2 Eg "Sunningdale"	String
Address3 Eg "Randburg"	String
Country Eg "South Africa	String
Postal Code. Eg "1234"	String
Grade. Eg "12"	Integer
Mobile. Eg "0827867654"	String
Mother. Eg "Mary Edwards"	String
Father . Eg "Peter Jackson"	String
House. Eg "Saturn"	String
Email. Eg "john345@gmail.com"	String

Father – Table

etc

Mother – Table

etc

Register Teacher – Table

etc

Text File Storage

Help – Contains the text for the Help Screen

1.8) Hardware and Software Specifications

Hardware

Processor?
RAM needed?
Available space on HDD?

Software

Operating system
Java Runtime Environment?
Microsoft Access?
ODBC driver?

Document 2 : System Design Document
Details the actual design of your program i.e. GUI design, program flow, class design, database design, storage design. This will be the longest and the most detailed document. Many of the pages will contain screen shots, diagrams and images. This document details how your program is going to work.

2.1 Cover

- Performance Assessment Task
- System Design Document
- Your full name
- Your grade
- The year.

2.2 Table of contents

2.3 User Interface Design

A screen shot of every one of your GUIs and how they will work. Also how the user will interact with the various screens in your program.

- Data to be displayed/entered on the screen
- Security (which users can access the screen)
- Action Elements (what can be clicked and what does clicking on the GUI component actually do?)

Readable fonts, good layout, suitable colours, easy-to-use input boxes, visibility of features and functions are all important.

Turn overleaf for an example of what this page could look like . . .