Location based multimedia application for mobile gadgets

By

Nacib Mohamed Ibraimo Dalsuco

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Technology(Hons)
(Information & Communication Technologies)

JULY 2007

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Location Based Multimedia Application for Mobile Gadgets

By

Nacib Mohamed Ibraimo Dalsuco

A project dissertation submitted to the
Information and Communication Technology Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
BACHELOR OF TECHNOLOGY (Hons)
(INFORMATION & COMMUNICATION TECHNOLOGY)

Approved by,

[illegible]

(Yew Kwang Hooi)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
July 2007

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

[illegible]

NACIB MOHAMED IBRAIMO DALSUCO

ABSTRACT

Nowadays tourists all around the world don't fully enjoy their vacations as they intended to. In order to assist tourist on their trip planning a location-based multimedia application is proposed. This application uses specific location points as the main criteria for retrieving and/or uploading information. It also allows the user to contribute to the application by uploading data. Therefore, tourist will be able to retrieve data related to where they are, as they travel. Also they will be able to contribute with data that they gather while traveling.

In this paper, the main parts that constitute the application are discussed, along with the methodologies taken to develop the application, as well as some of the tools that will be used to develop the application. Furthermore, using a Gantt chart, I give a brief preview of the tasks that are going to be completed throughout the development of the project.

ACKNOWLEDGEMENTS

I would like to deeply thank the various people who, during the several months in which this endeavor lasted, provided us with useful and helpful assistance. Without their care and consideration, this project wouldn't have been completed.

First, I would like to thank my supervisor Mr. Kwang Hooi, Yew for his support and dedication towards contributing with positive ideas which definitely contributed a lot to the completion of the project.

Second, I would like to thank my colleagues Sergio da Silva Jose Paua, and Helio Estevao Machabane for constantly giving me feedback on the project and providing some insightful ideas which also contributed to the completion of the project.

Last, but not least, I would like to thank UTP for giving me the opportunity and facilities to develop this project which has thought me a lot.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF EQUATIONS

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

Nowadays when people travel overseas they normally hire tour guides to assist them on their vacancy, and travel plans. Or, if they don't want to spend a lot of money on tour guides, since they are often expensive, they explore the country on their own. Either way, most of the times they do not enjoy the trip as they intended. This is often because, when you hire a tour guide, he/she has already a predefined tour, which may not be of full interest to a tourist, as tour guides tend to cover a broad area of interest as to cater to everybody's likes and dislikes. Furthermore, exploring a place that you have never been to by your own can be tiring and even dangerous.

Therefore, people need an application that can give them the ability to plan their trips before they travel. Such application would provide the users with videos, audio, images, and text about the tourist's possible places of interest. This way they would know where to go or not to in advance, thus giving a better and safer travel experience.

1.2 Objectives

The objectives of this project are to:

- create a multimedia application which is user friendly and interactive
- This application will be a location-based media application, which means that multimedia will be directly delivered to the user of a mobile device dependent upon their location.

- Locations will be the main criteria for the application to locate and/or upload data.
- In addition, the application will be an open application whereby users will have the ability to contribute to the application by uploading audiovisual content.

1.3 Scope of Study

In this project a functional prototype of the actual application will be developed for demonstration purposes. The application will be accessed either from a desktop computer and/or from a mobile gadget such as a notebook or a PDA. The prototype will be comprised of three main parts:

- The first part will be the backend of the application which will be mainly a database which will be used primarily to store predefined locations, audiovisual content, and user accounts
- The second part will be the user interface which will provide users all the tools they will need to fully utilize the application
- The third part and probably the most important part will be the algorithm which will be used to communicate between the database and the user interface.

For this project, UTP will be used as test location. That is, the prototype will be restricted to UTP's location. Also the information retrieved and/or uploaded by the application will be campus related. Furthermore, the application will use predefined location points which will be stored in the database, hence eliminating the dependency of special GPS devices, since this application will be mainly for mobile gadgets users.

From a user perspective view, the application should be user friendly, giving the user the ability to search and upload and/or search for location-based audiovisual content. In order for a user to upload information about a certain location he/she will need to create a user account, as to ensure security and reliability of the application. But, if a user wants to just search for information, he/she won't be requested to create a user account.

CHAPTER 2

LITERATURE REVIEW

2.1 Effective Location Based Services with Dynamic Data Management in Mobile Environments by Shiow-Yang Wu and Kun-Ta Wu

Shiow-Yang Wu and Kun-Ta Wu, from the Department of Computer Science and Information Engineering, National Dong Hwa University, Hwalien, Taiwan, Republic of China; on their article entitled "Effective Location Based Services with Dynamic Data Management in Mobile Environments", published online on 30 December 2005, acknowledged that the advances in portable devices and wireless communication technologies enabled a new form of services named location based services which delivered location dependent and context sensitive information to mobile users. In addition, they mentioned that a key characteristic of location based services is that the same service request may need to be answered with completely different results as the user changes his/her location or the targets move. Shiow-yang Wu and Kun-Ta Wu, based on their analysis, characterized the dynamic data management problem for location based services and devised a general service architecture flexible enough to provide dynamic access to both location dependent and location independent data.

Figure 1 depicts their proposed system architecture for location-based information services. This architecture was designed with several criteria in mind. First of all, the architecture reflects the current and foreseeable future status of the wireless networking technologies. Also this architecture was designed in order to make the system flexible enough to allow the installment of different data management strategies on different classes of applications.

The central server is used to model a service site for centralized data such as the New York Stock Exchange. In addition to a central information database, a push unit is included to facilitate server-initiated pushing strategies that proactively send selected data items toward the clients. Since the downlink bandwidth is usually much larger and cheaper than uplink connection, pushing techniques turn out to be efficient and valuable tools with little extra cost.

The local server is the data manager and wireless information server for a single cell. Each cell is assumed to have a unique local server which provides wireless access for all the clients in its cell and acts as a bridge between the central server and the client devices at the same time. It is the center for managing local information as well as the key player to provide location based services. All local servers are connected to the Internet via fixed network and therefore can send information to each others with almost negligible delay in comparison with wireless access. This is an important factor since neighboring local servers must work closely together to provide efficient location-based services.

The client is any end user device that is capable of wireless communication as well as user interface services. This is probably the branch of the wireless technologies that evolves with the fastest pace. The most distinctive feature of a client is that it moves. A client can change its position at will, in and out of a cell, from one cell to another, without the obligation to notify any server in advance. Since a client can issue a query at any time any where, this is especially challenging for information service providers. In this architectural design, a client always sends requests to the local server of the cell where the client resides. The target objects may be available right at the client cache,
at the local server of the same cell, from the local servers of other cells, or from central servers.

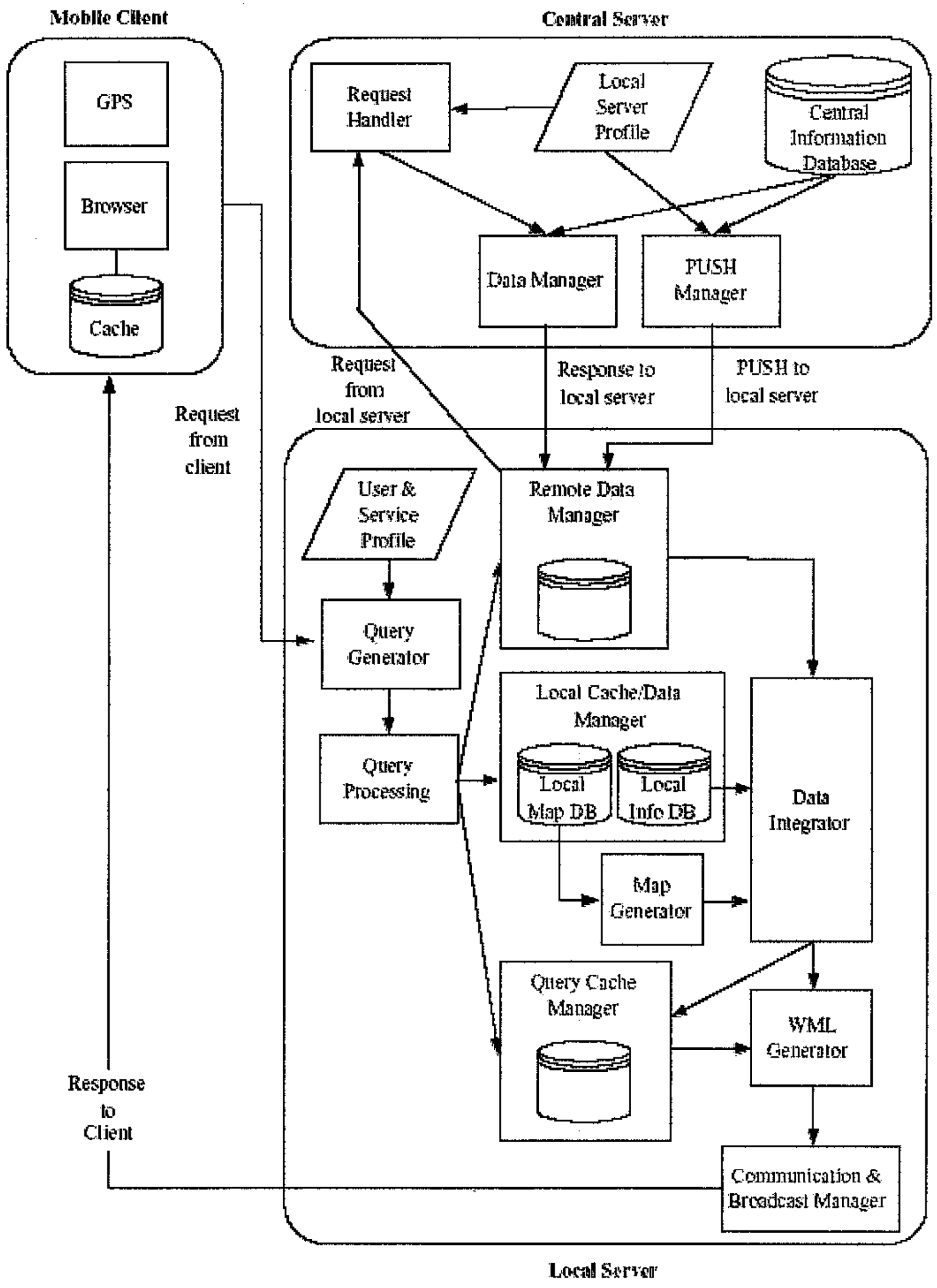


Figure 1 - System architecture for location based services.

2.2 Web Services

A web service (sometimes called an XML Web Service) is an application that enables distributed computing by allowing one machine to call methods on other machines via common data formats and protocols such as XML, and HTML. These method calls are implemented using the Simple Object Access Protocol (SOAP), an XML-based protocol describing hot to mark up requests and responses so that they can be transferred via protocols such as HTTP. Using SOAP, applications represent and transmit data in a standardized format – XML.

As larger numbers of people worldwide connect to the internet, the concept of applications that call methods across a network becomes more practical. Web services represent the next step in object-oriented programming: instead of developing software from small number of class libraries provided at one location, programmers can access countless libraries in multiple locations. By purchasing Web services that are relevant to their business, companies that create applications can spend less time coding and more time developing new products. In addition, e-business can employ Web services to provide their customers with an enhanced shopping experience.

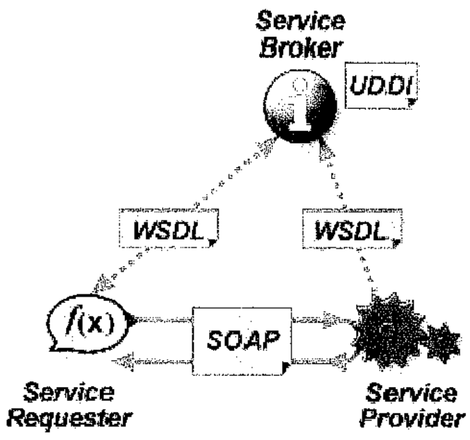


Figure 2 - Web Service

2.3 MySQL

MySQL is a multi-user, multithreaded (i.e. it allows multiple simultaneous connections) robust and scalable relational database management system (RDBMS).

MySQL has numerous important features such as:

1. Multithreading capabilities that enable the database to perform multiple tasks concurrently, allowing the server to process client requests efficiently.
2. Support for various programming languages (C, C++, Java, Perl, PHP, ColdFusion, Python, etc.).
3. Implementations of MySQL are available for Windows, Mac OS X, Linux and UNIX.
4. Full support of functions and operators within the SELECT and WHERE clauses of an SQL query that allows users to manipulate data.
5. The ability to access tables from different databases by using a single query, increasing the efficiency of retrieving accurate and necessary information.
6. the ability to handle large databases.

For these reasons, MySQL is becoming the database of choice for many businesses, universities and individuals. MySQL is an open-source software product, which means that it can be freely obtained and customized to fulfill corporate, educational or personal requirements.

2.4 PHP

PHP, or PHP: Hypertext Preprocessor, is quickly becoming one of the most popular server-side scripting languages for creating dynamic Web pages.

PHP is an open-source technology that is supported by a large community of users and developers. PHP is platform independent; implementations exist for all major UNIX, Linux and Windows operating systems. PHP also supports a large number of databases, including MySQL.

PHP code is embedded directly into XHTML documents. This allows the document author to write XHTML in clear, concise manner, without having to use multiple print statements, as is necessary with other CGI-based languages.

2.5 ASP.Net

ASP.NET is a web application framework marketed by Microsoft that programmers can use to build dynamic web sites, web applications and XML web services. It is part of Microsoft's .NET platform and is the successor to Microsoft's Active Server Pages (ASP) technology.

ASP.NET is built on the Common Language Runtime, meaning programmers can write ASP.NET code using any Microsoft .NET language.

2.5.1 ASPX file format

ASPX is a text file format used to create Web form pages; in programming jargon, the ASPX file typically contains static HTML or XHTML markup, as well as markup defining Web Controls and Web User Controls where the developers place all the required static and dynamic content for the web page. Additionally, dynamic code which runs on the server can be placed in a page within a block <% -- dynamic code -- %>

which is similar to other web development technologies such as PHP, JSP, and ASP, but this practice is generally frowned upon by Microsoft except for the purposes of data binding since it requires more calls when rendering the page.

The method recommended by Microsoft for dealing with dynamic program code is to use the code-behind model, which places this code in a separate file or in a specially designated script tag. Code-behind files are typically named something to the effect of MyPage.aspx.cs or MyPage.aspx.vb based on the ASPX file name (this practice is automatic in Microsoft Visual Studio and other IDEs). When using this style of programming, the developer writes code to respond to different events, like the page being loaded, or a control being clicked, rather than a procedural walk through the document.

2.5.2 Rendering Technique

ASP.NET uses a *visited composites* rendering technique. During compilation the template (.aspx) file is compiled into initialization code which will build a control tree (the composite) representing the original (static) template. Literal text goes into instances of the Literal control class, server controls are represented by instances of a specific control class. The initialization code is combined with user-written code (usually by the assembly of multiple partial classes) and results in a class specific for the page. The page doubles as the root of the control tree.

Actual requests for the page are processed through a number of steps. First, during the initialization steps, an instance of the page class is created and the initialization code is executed. This produces the initial control tree which is now typically manipulated by the methods of the page in the following steps. As each node in the tree is a control represented as an instance of a class, the code may change the tree structure as well as manipulate the properties/methods of the individual nodes. Finally, during the rendering step a visitor is used to visit every node in the tree, asking each node to render itself using the methods of the visitor. The resulting HTML code is sent to the client.

After the request has been processed, the instance of the page class is discarded and with it the entire control tree.

2.5.3 Performance

ASP.NET aims for performance benefits over other script-based technologies (including ASP Classic) by compiling the server-side code to one or more DLL files on the web server. This compilation happens automatically the first time a page is requested (which means the developer need not perform a separate compilation step for pages). This feature provides the ease of development offered by scripting languages with the performance benefits of a compiled binary. However, the compilation might cause a noticeable delay to the web user when the newly-edited page is first requested from the web server.

The ASPX and other resource files are placed in a virtual host on an Internet Information Services (or other compatible ASP.NET servers; see Other Implementations, below). The first time a client requests a page, the .NET framework parses and compiles the file(s) into a .NET assembly and sends the response; subsequent requests are served from the dll files. By default ASP.NET will compile the entire site in batches of 1000 files upon first request. If the compilation delay is causing problems, the batch size or the compilation strategy may be tweaked.

Developers can also choose to pre-compile their code before deployment, eliminating the need for just-in-time compilation in a production environment.

2.6 Positioning Systems

2.6.1 GPS Location

The **Global Positioning System (GPS)** is currently the only fully functional Global Navigation Satellite System (GNSS). More than two dozen GPS satellites are in medium Earth orbit, transmitting signals allowing GPS receivers to determine the receiver's location, speed and direction.

Since the first experimental satellite was launched in 1978, GPS has become an indispensable aid to navigation around the world, and an important tool for map-making and land surveying. GPS also provides a precise time reference used in many applications including scientific study of earthquakes, and synchronization of telecommunications networks.

Developed by the United States Department of Defense, it is officially named **NAVSTAR GPS** (**NAV**igation **S**atellite **T**iming **A**nd **R**anging **G**lobal **P**ositioning **S**ystem). The satellite constellation is managed by the United States Air Force 50th Space Wing. The cost of maintaining the system is approximately US$750 million per year,[1] including the replacement of aging satellites, and research and development. Despite this fact, GPS is free for civilian use as a public good.

2.6.1.1 Simplified Method of Operation

A GPS receiver calculates its position by measuring the distance between itself and three or more GPS satellites. Measuring the time delay between transmission and reception of each GPS radio signal gives the distance to each satellite, since the signal travels at a known speed. The signals also carry information about the satellites' location. By determining the position of, and distance to, at least three satellites, the receiver can compute its position using trilateration.[2] Receivers typically do not have perfectly accurate clocks and therefore track one or more additional satellites to correct the receiver's clock error.

2.6.1.2 Drawbacks

To achieve the objective of providing the user's location GPS (Global Positioning System) looks like the obvious solution due to its accuracy. However, this solution has a number of restrictions such as the fact that indoor coverage it's not supported by the service and this service as an acquisition time in the region of 10-60 seconds. Also, much of the urban environments would restrict a mobile user's "view" of the satellites and it would be unreliable in such situations.

Also, the complexity and costs of implementation is an issue due to the fact that it would require mobile users to be equipped with GPS devices which are a bit expensive and not easily found in the market and by doing so this would severely increase the gadget's power consumption.

2.6.2 Cellular-network-based wide-area location systems

Several cellular-network-based wide-area location systems have been proposed in recent years. The technological methods of location determination involve measuring the signal strength, the angle of signal arrival, and/or the time difference of signal arrival. However, the accuracy of wide-area location systems is highly limited by the cell size. Moreover, the effectiveness of systems for an indoor environment is also limited by the multiple reflections suffered by the radio frequency (RF) signal.

2.6.3 Indoor location systems

For an indoor environment, several systems based on various technologies such as infrared (IR), ultrasound, video surveillance, and radio signal are emerging. Among these systems, radio-signal-based approaches—more specifically, the wireless local-area network (WLAN) (IEEE 802.11b, also named *Wi-Fi*) radio-signal-based positioning system—have drawn great attention in recent years. A WLAN-based positioning system has distinct advantages over all other systems.

First, it is an economical solution because the WLAN network usually exists already as part of the communications infrastructure. For a notebook computer, personal digital assistant (PDA), or other mobile devices equipped with WLAN capability, the positioning system can be implemented simply in software—generally in middleware or at the application level. This software-based location system significantly reduces cost with respect to dedicated architectures. Second, the WLAN-based positioning system covers a large area compared with other types of indoor positioning systems. The WLAN-based positioning system may work in a large building or even across many buildings. Third, it is a stable system owing to its robust RF signal propagation. Video- or IR-based location systems are subject to restrictions, such as line-of-sight limitations or poor performance with fluorescent lighting or in direct sunlight.

2.6.4 WLAN Positioning

In WLAN positioning we can find two main techniques:

- Empirical model
- Propagation Model

2.6.4.1 Empirical Model

Empirical model is based on storing pre-recorded measurements in a database. Before any location positioning can be performed, a radio map is created. This map stores the signal strengths at each particular location from all access points that can be read from that area into the database. When a device requests a location, it matches signals strengths from all the access points that it can read with the database.

In the Empirical model the following algorithms are used:

- K nearest neighbor
- Vitergi-like algorithm
- Naïve bayes

- Neural Networks
- Fuzzy logic
- Subspace Techniques
- Hidden Markov model based techniques

The Empirical model requires a large amount of manual effort before positioning can begin. Also, it cannot be improved beyond a certain point by increasing the granularity of the grid, because the variations in signal strengths become small. The time of the day when the RF map is created is of vital importance, since the radio wave properties in an indoor environment vary greatly depending on the number of people etc. in the building.

Even though, despite all the disadvantages mentioned above, the empirical model is more accurate than the propagation model.

2.6.5 Propagation Model

The propagation model is based on the fact that as a radio wave travels through environment it loses signal strength. This loss of signal strength by the radio wave is dependent on the environment, and is modeled by using given models such as the Hata-okumura model.

Using one specific model for signal strength loss modeling, the distance from a wireless device to an access point can be determined by the signal strength loss over space. Triangulation can be used to determine the location of the device by calculating the distance to three or more access points.

This positioning technique has a fairly good accuracy, given a reasonable proximity to the access points. This accuracy decreases as the distance between the device and the affiliated access point increases. One solution to this is to give the primary access point a higher weight in the triangulation algorithm. Also, accuracy can be improved by increasing the complexity of the model used.

2.6.6 Multilateration

Multilateration, also known as hyperbolic positioning, is the process of locating an object by accurately computing the time difference of arrival (TDOA) of a signal emitted from the object to three or more receivers. It also refers to the case of locating a receiver by measuring the TDOA of a signal transmitted from three or more synchronized transmitters.

Multilateration should not be confused with trilateration, which uses absolute measurements of time-of-arrival from three or more sites.

2.6.6.1 Principle

Multilateration is commonly used in civil and military surveillance applications to accurately locate an aircraft, vehicle or stationary emitter by measuring the time difference of arrival (TDOA) of a signal from the emitter at three or more receiver sites.

If a pulse is emitted from a platform, it will arrive at slightly different times at two spatially separated receiver sites, the TDOA being due to the different distances of each receiver from the platform. In fact, for given locations of the two receivers, a whole series of emitter locations would give the same measurement of TDOA. Given two receiver locations and a known TDOA, the locus of possible emitter locations is a one half of a two-sheeted hyperboloid.
In simple terms, with two receivers at known locations, an emitter can be located onto a hyperboloid. Note that the receivers do not need to know the absolute time at which the pulse was transmitted - only the time difference is needed.

Consider now a third receiver at a third location. This would provide a second TDOA measurement and hence locate the emitter on a second hyperboloid. The intersection of these two hyperboloids describes a curve on which the emitter lies.
If a fourth receiver is now introduced, a third TDOA measurement is available and the intersection of the resulting third hyperboloid with the curve already found with the

other three receivers defines a unique point in space. The emitter's location is therefore fully determined in 3D.

In practice, errors in the measurement of the time of arrival of pulses mean that enhanced accuracy can be obtained with more than four receivers. In general, N receivers provide N-1 hyperboloids. When there are N > 4 receivers, the N-1 hyperboloids should, assuming a perfect model and measurements, intersect on a single point. In reality, the surfaces rarely intersect, because of various errors. In this case, the location problem can be posed as an optimization problem and solved using, for example, a least squares method or an extended Kalman filter.

Additionally, the TDOA of multiple transmitted pulses from the emitter can be averaged to improve accuracy.

2.6.6.2 Reciprocal case: locating a receiver from multiple transmitter sites

Multilateration can also be used by a single receiver to locate itself, by measuring the TDOA of signals emitted from three or more synchronized transmitters at known locations. This can be used by navigation systems, an example being the British DECCA navigation system, developed during World War II, which used the phase-difference of two transmitters, rather than the TDOA of a pulse, to define the hyperboloids. This allowed the transmitters to broadcast a continuous wave signal. Phase-difference and time-difference can be considered the same for narrow-band transmitters.

2.6.6.3 Derivation

Consider an emitter at unknown location (x,y,z) which we wish to locate. Consider also a multilateration system comprising four receiver sites at known locations: a central site, C, a left site, L, a right site, R and a fourth site, Q.
The travel time (T) of pulses from the emitter at (x,y,z) to each of the receiver locations is simply the distance divided by the pulse propagation rate (c):

$$T_L = \frac{1}{c}\left(\sqrt{(x - x_L)^2 + (y - y_L)^2 + (z - z_L)^2}\right)$$

Equation 1 - Travel time for left site

$$T_R = \frac{1}{c}\left(\sqrt{(x - x_R)^2 + (y - y_R)^2 + (z - z_R)^2}\right)$$

Equation 2 - travel time for right site

$$T_Q = \frac{1}{c}\left(\sqrt{(x - x_Q)^2 + (y - y_Q)^2 + (z - z_Q)^2}\right)$$

Equation 3 - Travel time for fourth site

$$T_C = \frac{1}{c}\left(\sqrt{(x - x_C)^2 + (y - y_C)^2 + (z - z_C)^2}\right)$$

Equation 4 - Travel time for center site

If the site C is taken to be at the coordinate system origin,

$$T_C = \frac{1}{c}\left(\sqrt{x^2 + y^2 + z^2}\right)$$

Equation 5 - Travel time for center site

Then the time difference of arrival between pulses arriving directly at the central site and those coming via the side sites can be shown to be:

$$\tau_L = T_L - T_C = \frac{1}{c}\left(\sqrt{(x - x_L)^2 + (y - y_L)^2 + (z - z_L)^2} - \sqrt{x^2 + y^2 + z^2}\right)$$

Equation 6 - Travel time for left site when center site is at the origin

$$\tau_R = T_R - T_C = \frac{1}{c}\left(\sqrt{(x - x_R)^2 + (y - y_R)^2 + (z - z_R)^2} - \sqrt{x^2 + y^2 + z^2}\right)$$

Equation 7 - Travel time for right site when center site is at the origin

$$\tau_Q = T_Q - T_C = \frac{1}{c}\left(\sqrt{(x - x_Q)^2 + (y - y_Q)^2 + (z - z_Q)^2} - \sqrt{x^2 + y^2 + z^2}\right)$$

Equation 8 - Travel time for fourth site when center cite is at the origin

where $(x_L,\ y_L,\ z_L)$ is the location of the left receiver site, etc, and c is the speed of propagation of the pulse, often the speed of light. Each equation defines a separate hyperboloid.

The multilateration system must then solve for the unknown target location (x,y,z) in real time. All the other symbols are known.

Note that in civilian air traffic control multilateration systems, the unknown height, z, is often directly derived from the Mode C SSR transponder return. In this case, only three sites are required for a 3D solution.

2.6.6.4 Multilateration accuracy

Multilateration is, in general, far more accurate for locating an object than techniques such as triangulation (as it is easier to measure time accurately than it is to form a very narrow beam). The accuracy of multilateration is a function of several variables, including:

* The geometry of the receiver(s) and transmitter(s)
* The timing accuracy of the receiver system
* The accuracy of the synchronization of the transmitting sites or receiving sites. This can be degraded by unknown propagation effects.
* The bandwidth of the emitted pulse(s)
* Uncertainties in the locations of the receivers

CHAPTER 3
DISCUSSION

3.1 Development Model

For the purposes of planning, implementation, and testing, of this project a Spiral development model will be used. Such development model was chosen because this model explicitly recognizes risk. Risks result in project problems such as schedule overrun so risk minimization is a very important project management activity.

The spiral model of the software process was originally proposed by Barry Boehm (Boehm, 1988) as an iterative waterfall in which each iteration provides increasing capability. Rather, than representing the software process as a sequence of activities with some backtracking from one activity to another, the process is represented as a spiral. Each loop in the spiral represents a phase of the software process.

This project will have three main phases:

1. Phase 1: This phase will be more about understanding the requirements and scope of the project, and thoroughly identifying its objectives. The activities in this phase will be more inclined on researching the technologies and means to achieve the project's objectives and meet its requirements. As a result of this phase a simple prototype of the application will be devised.
2. Phase 2: In this phase the activities will be inclined more towards implementation and development, and at the same time briefly researching on how to do such tasks. This is more like a transition phase between phase 1 and phase 3. The basic idea is to efficiently apply and implement the findings on the

3. phase 1 while, at the same time, better understanding these findings. As a result of this phase a better and more functional prototype will be developed.

4. Phase 3: This phase will be mainly development and software testing and improvement. A final working prototype will be developed at the end of this phase.

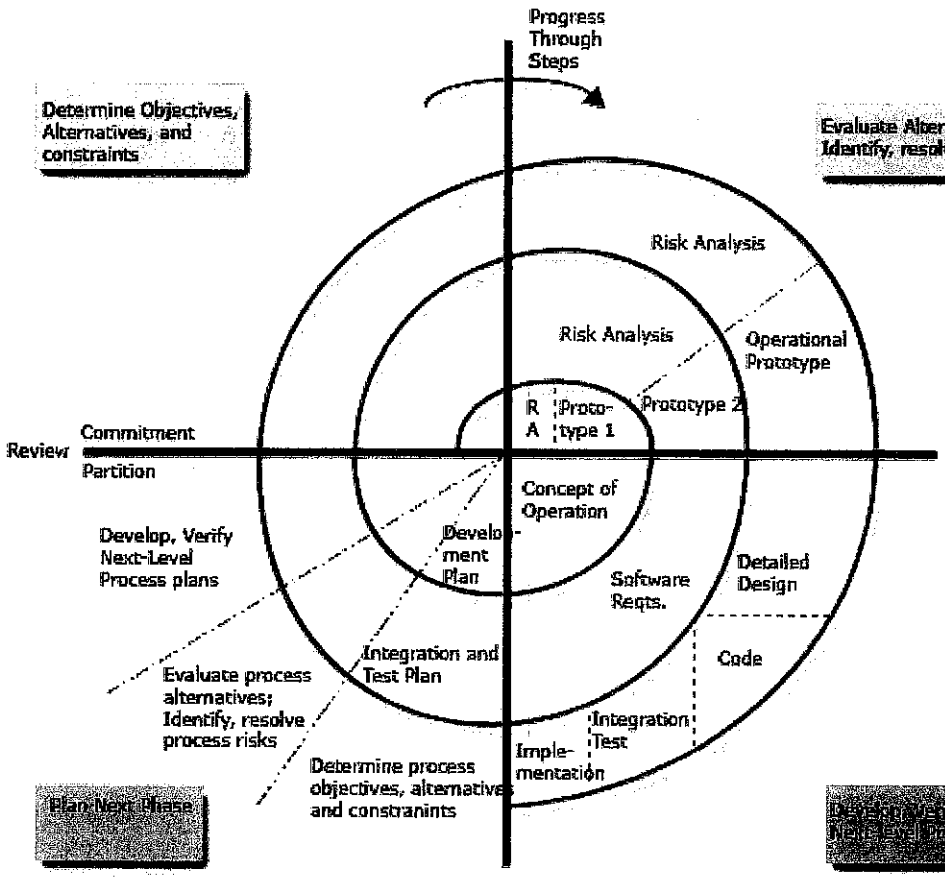

Progress Through Steps
Determine Objectives, Alternatives, and constraints
Evaluate Alternatives; Identify, resolve risks
Risk Analysis
Risk Analysis
Operational Prototype
Prototype 2
R A
Proto-type 1
Review
Commitment
Partition
Concept of Operation
Develop, Verify Next-Level Process plans
Develo-ment Plan
Software Reqts.
Detailed Design
Code
Integration and Test Plan
Evaluate process alternatives; Identify, resolve process risks
Integration Test
Imple-mentation
Determine process objectives, alternatives and constranints
Plan Next Phase
Develop, Verify Next-Level Product

Figure 3 - Project's Spiral Development Model

3.2 Project Architecture Overview

3.2.1 Overall architecture

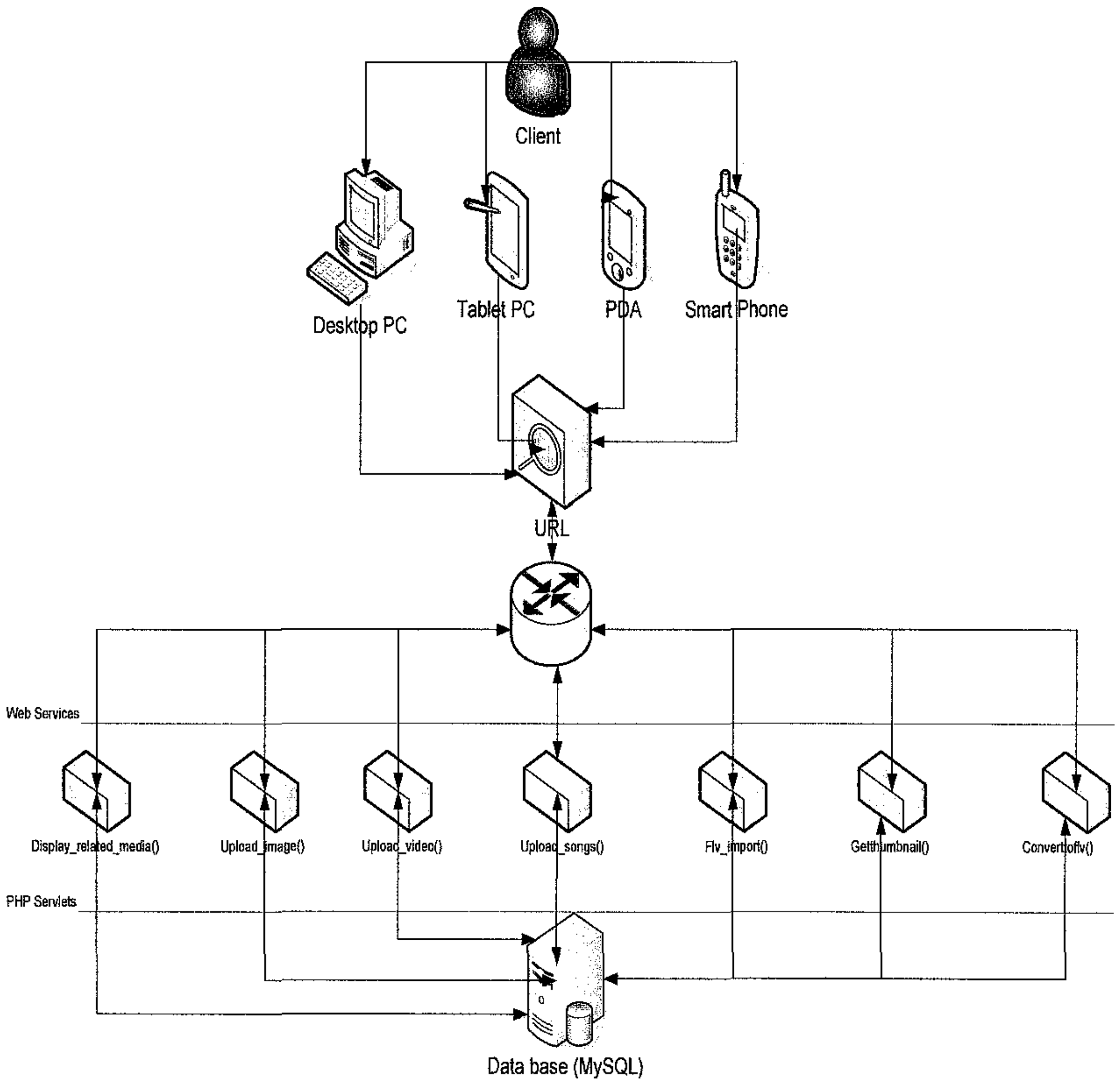


Figure 4 - Project Architecture Overview

3.2.1.1 Description

The basic idea of this architecture is that a client accesses the URL page either using a Desktop PC or a mobile gadget. In this URL there will be web services provided to the client that will give him options to access the main database. This web services are directly connected to the PHP servlets which in their turn connect to the MySQL database.

3.2.2 Web site map

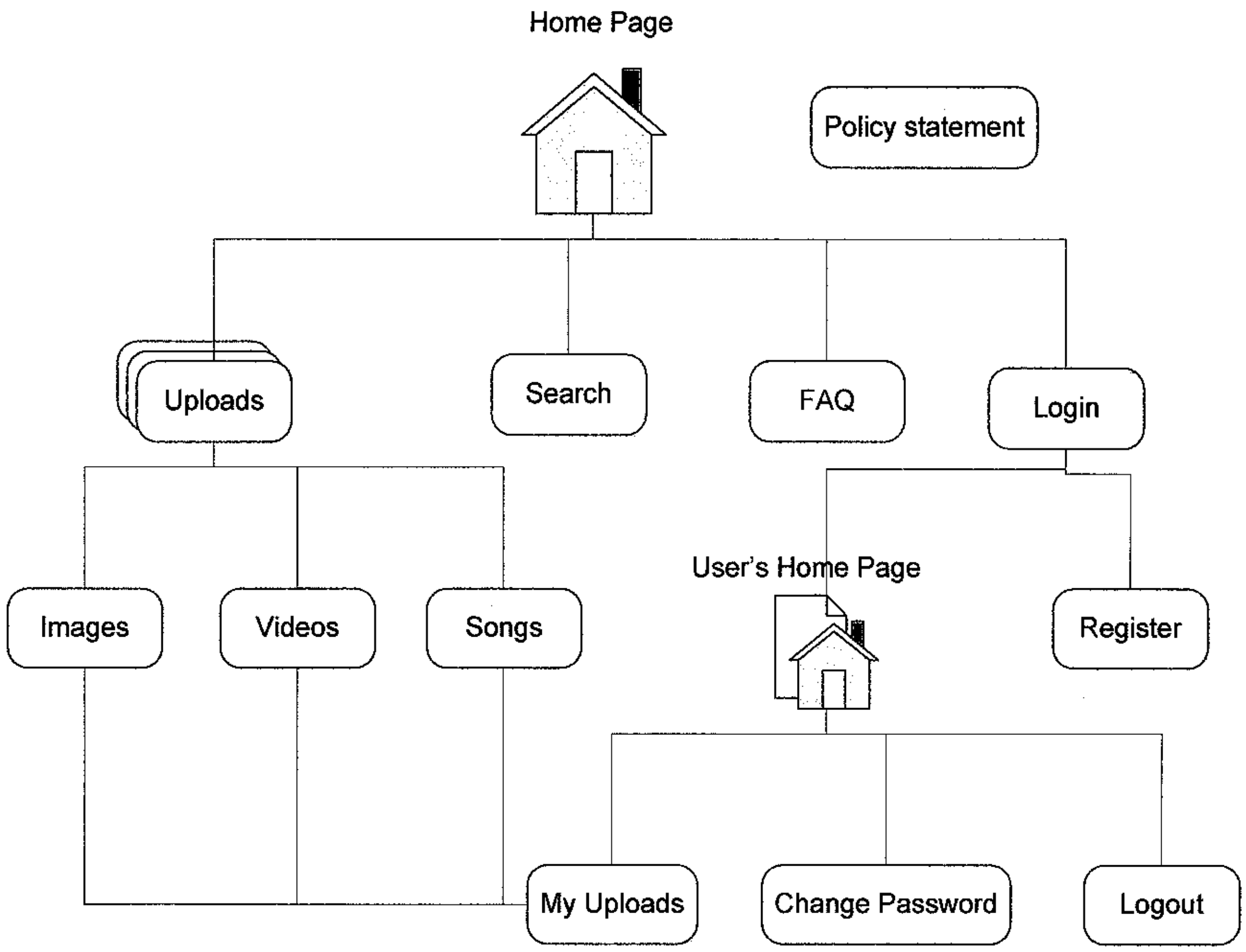


Figure 5 - Web site map

3.2.3 The Database

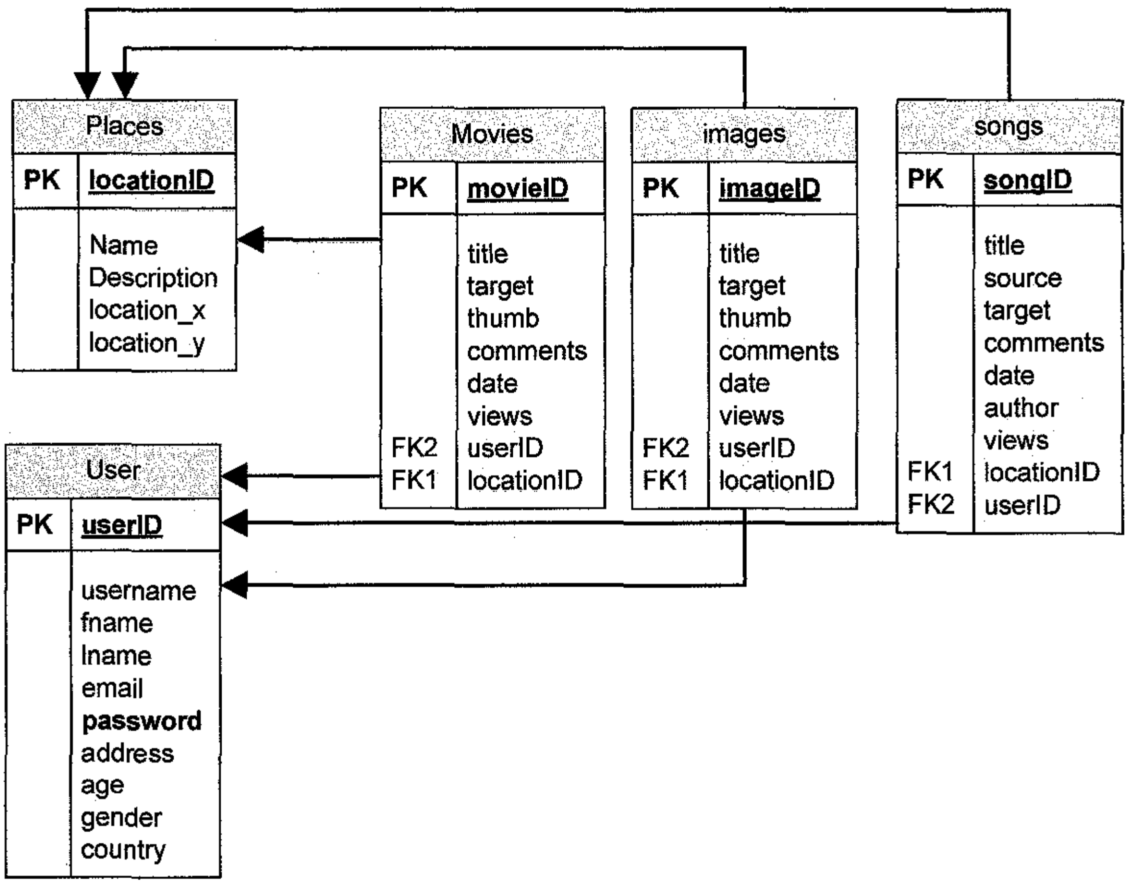


Figure 6 - Database Overview

In this project, tentatively, five main tables will be created:

1. Places Table: this table will store information about the place to which a particular GPS point points to.
2. User Table: this table will store information about the clients such a Name, address, etc.
3. Movies table: this table will store the information regarding the movies that user upload to the system.
4. Images table: this table will store the information regarding the images that users upload into the system
5. Songs table: this table will store the information regarding the audio files that users upload into the system

The database will be developed using MySQL, due to the features and benefits that this database system provides. To communicate with the database, PHP will be used. This combination (PHP and MySQL) is often called as the *Dynamic Duo.*

Files will be stored in a separate folder, other than being uploaded into the database directly. Performance wise, this option was found to be the best when dealing with file uploads, especially in such an application where video files, which tend to occupy large amounts of space, will be uploaded by the user. In the database, only the name of the files and the file path to the file will be stored. When a user requests for a specific file, the file path is fetched from the database, with php and presented to the user.

3.3 Functional Models

3.3.1 Use case Diagram

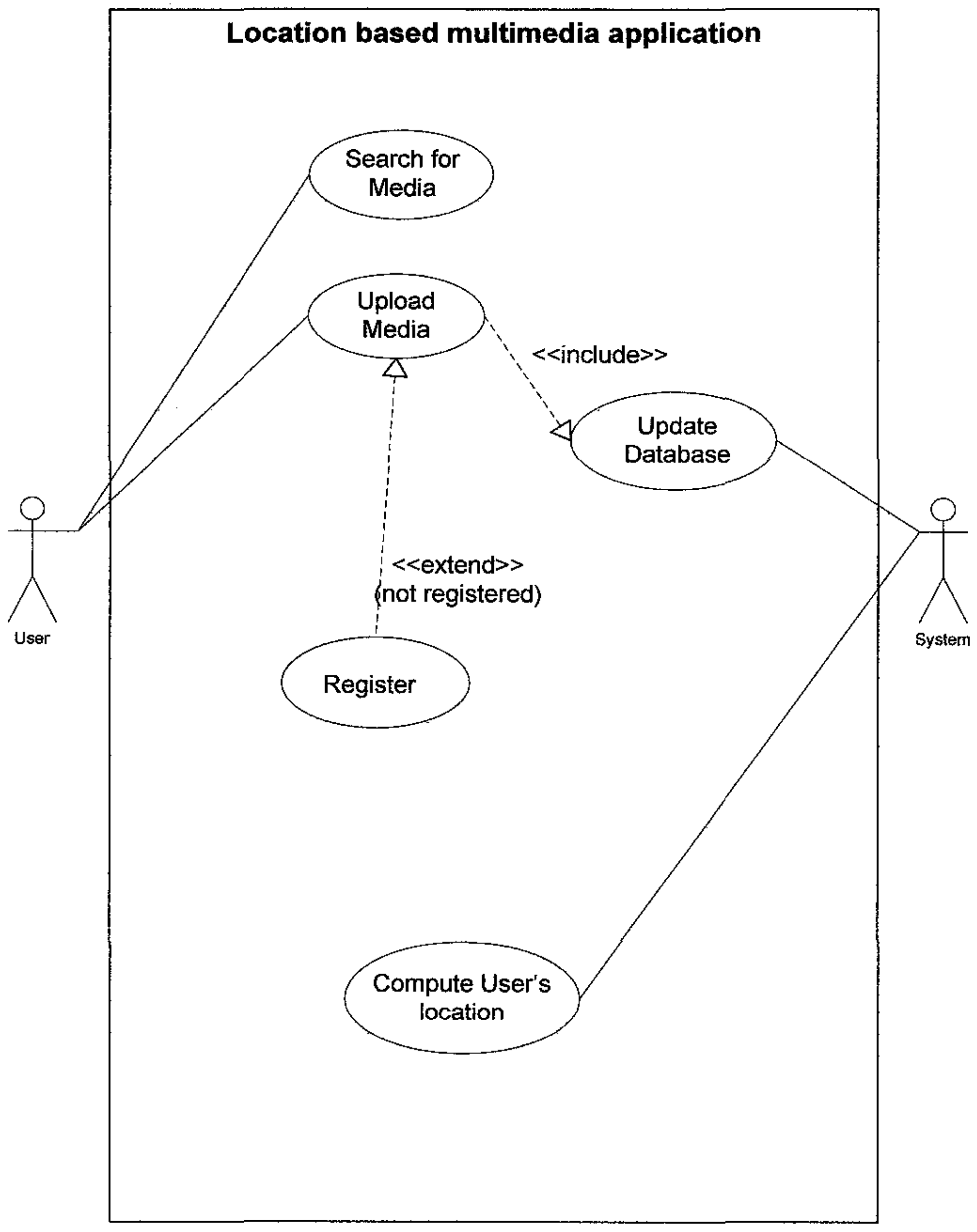


Figure 7 - Use case diagram

3.3.2 Use Case Descriptions

Search for Media

- **Use case name:** Search for media
- **Summary:** User searches for media in the application by giving a search criteria
- **Dependency:**
- **Actor:** User
- **Precondition:**
- **Description:**
    - User presses the search link in the system which will take him to the search page
    - When in the search page user enters search criteria and chooses what type of media is he looking for
    - System then displays the search results.
- **Alternatives:**
- ~~**Post condition:**~~

Upload Data

- **Use case name:** Upload Data
- **Summary:** User uploads data into the system
- **Dependency:**
- **Actor:** User
- **Precondition:** User must be registered and logged in
- **Description:**
    - User presses the upload link in the system
    - If the user is not registered, the system prompts the user to register. After registering the user must log into the system
    - The user then chooses the type of media that he would like to upload and also the location at which the media is bound to
    - After the upload the system updates the database
- **Alternatives:**

- **Post condition:** Database has been updated

Register

- **Use case name:** Register
- **Summary:** User registers to the system
- **Dependency:**
- **Actor:** User
- **Precondition:** User is not registered
- **Description:**
    - User presses the register link in the system
    - In the register page, the user enters he's details
    - After confirming his(her) details the user clicks the submit button to finalize the process
- **Alternatives:**
- **Post condition:** Database has been updated

Update Database

- **Use case name:** Update Database
- **Summary:** The system updates the database whenever there has been a change in the system data
- **Dependency:**
- **Actor:** System
- **Precondition:** A change in the system's data has been made
- **Description:**
    - The system establishes a connection with the database
    - After a successful connection has been made the system performs the requested operation on the database
    - After all the operations have been made, the connection with the database is closed
- **Alternatives:**
- **Post condition:**

Compute User's GPS location

- **Use case name:** Compute User's location
- **Summary:** The system, using the propagation method and triangulation, computes the user's GPS location
- **Dependency:**
- **Actor:** System
- **Precondition:** The user has accessed the system
- **Description:**
    - The system gets the list of all the access points accessible from the device
    - The system them accesses the database and get the location for each of the access points detected
    - The system then, pings the access points and determines their RTT
    - Once the RTT have been determined, multilateration is used to determine the user's location
    - After successfully calculating the user's location, the result is compared with the values in the database and then the corresponding location is displayed in the screen
- **Alternatives:**
- **Post condition:**

3.3.3 Activity Diagram

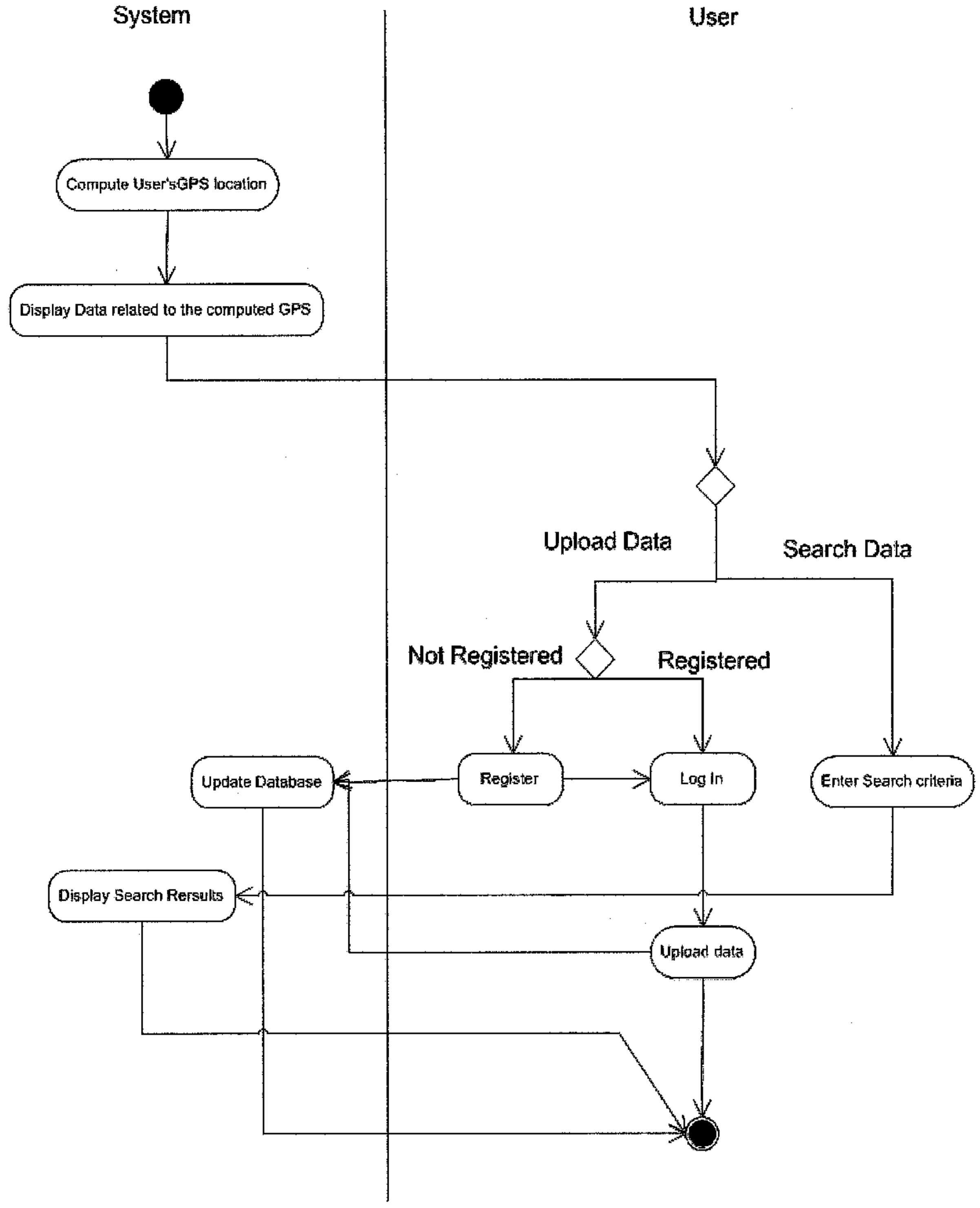


Figure 8 - Activity Diagram

3.4 Interface Design

When designing the interface for this project various aspects were taken into consideration, such as:

- **Consistency:** the components of the design where made in such a way that they behave consistently at all times for all screens. Also, consistent terminology between screens was used, as well as the icons and colors.
- **Simplicity:** complex tasks, such as file uploading, were broken down into simpler tasks by implementing it through separate steps. Also, to assist in the simplification of processes icons, words, and other elements were used.
- **Human Memory Limitations:** information was organized into small number of chunks. Also, cues/navigation aids were provided to assist the user on knowing where they are in the system. Reminders and warnings were provided appropriately. Furthermore, to reduce memory working loads, the length of sequences and quantity of information displayed was minimized.
- **Attention:** attention grabbing techniques were used cautiously (e.g. overusing of 'blinks' were avoided, as well as flashing messages and bold color).Not more than 4 different font sizes per screen were used. A maximum of 2 levels of intensity were used on a single screen.
- **Display issues:** display inertia was maintained, that is, the screen was made sure to change very little from one screen to another within a functional task. Screen complexity was organized. Concise, unambiguous wording was used for instructions and messages. A balanced screen layout was used, in such a way that information was distributed accordingly throughout the screen quadrant.

3.4.1 Screen layout

The screen layout of the web site was based in a 1024 x 768 pixels screen resolution, which, due to technological advancements, tends to be the default screen size layout for web pages.

The screen is divided into three sections, title, navigation, and main section, respectively.

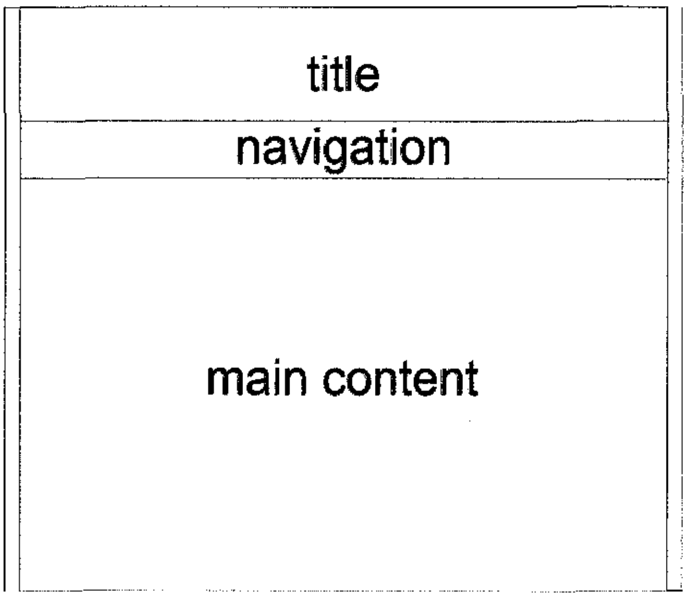


Figure 9 - Screen layout

3.4.1.1 Title section

In this section the title of the web page and the project's title are displayed. Also the slogan of the project is displayed.

Figure 10 - Title section

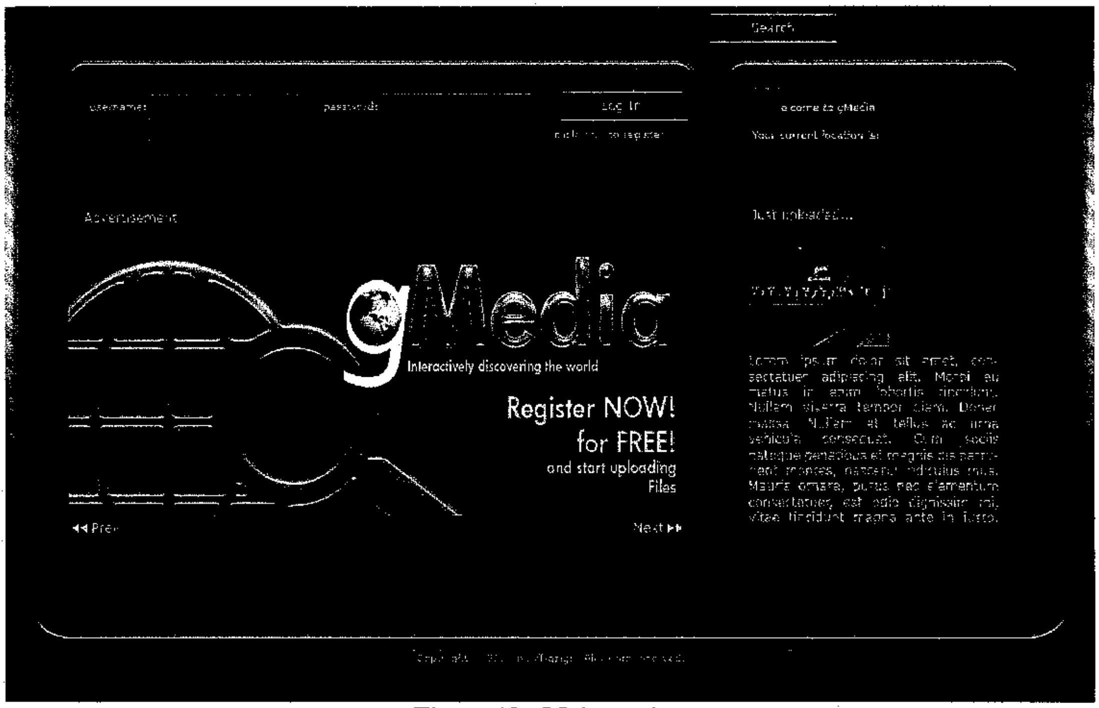
3.4.1.2 Navigation section

In the navigation section the user can find buttons which providade basic navigation through the web site.

Figure 11 - Navigation section

3.4.1.3 Main section

In the main section is where the actual content of the web site will be displayed, such as forms, search results, etc.



Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Morbi eu metus in enim lobortis tincidunt. Nullam viverra tempor diam. Donec massa. Nullam et tellus ac urna vehicula consequat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Mauris ornare, purus nec elementum consectetuer, est odio dignissim mi, vitae tincidunt magna ante in justo.

Figure 12 - Main section

3.5 Implementation

3.5.1 Positioning

For this project, the multilateration positioning technique will be used, whereby the position of the user will be determined by computing the time difference of arrival of a signal emitted from the object to three wireless routers.

The first step to achieve this will be to ping each of the wireless routers. By doing so, we will be able to determine the round trip time (RTT) for each of the wireless routers. The function to ping the routers will be written in ASP.NET framework. This function will export the results toa text file.

After "pinging" the routers and exporting the results to text file, we have to extract the average round trip time (RTT) from the text file. This can be easily achieved using java. Once the RTT as been extracted from the text file it's just a matter of applying the multilateration formula to compute the user's position.

After computing the user's location, this location is then compared with the locations stored in the database to determine where the user is in relation to the locations stored in the database (e.g. to find out whether the user is in the chancellor complex or in building 2).

Refer to appendix 1 for a detailed flowchart of the positioning process.

3.5.1.1 Test Plan

Fig. 7 depicts an experiment that will is to be conducted in one of the computer labs whereby the room will be divided into 16 areas of equal size, and in each of the area a unique identifier will be assigned. After assigning the unique identifier for each area, three wireless access point will be placed in the room and their location relatively to the room will be recorded. After setting up the environment a wireless gadget will be placed in one of the 16 areas of the room, and will be used to access the application and based on the multilateration positioning technique, the user's position in the room will be determined. The calculated position will be the compared to the position of the area at which the mobile gadget was placed, to check for reliability and accuracy of the system. In this experiment the upper left corner of the room will be used as origin point.

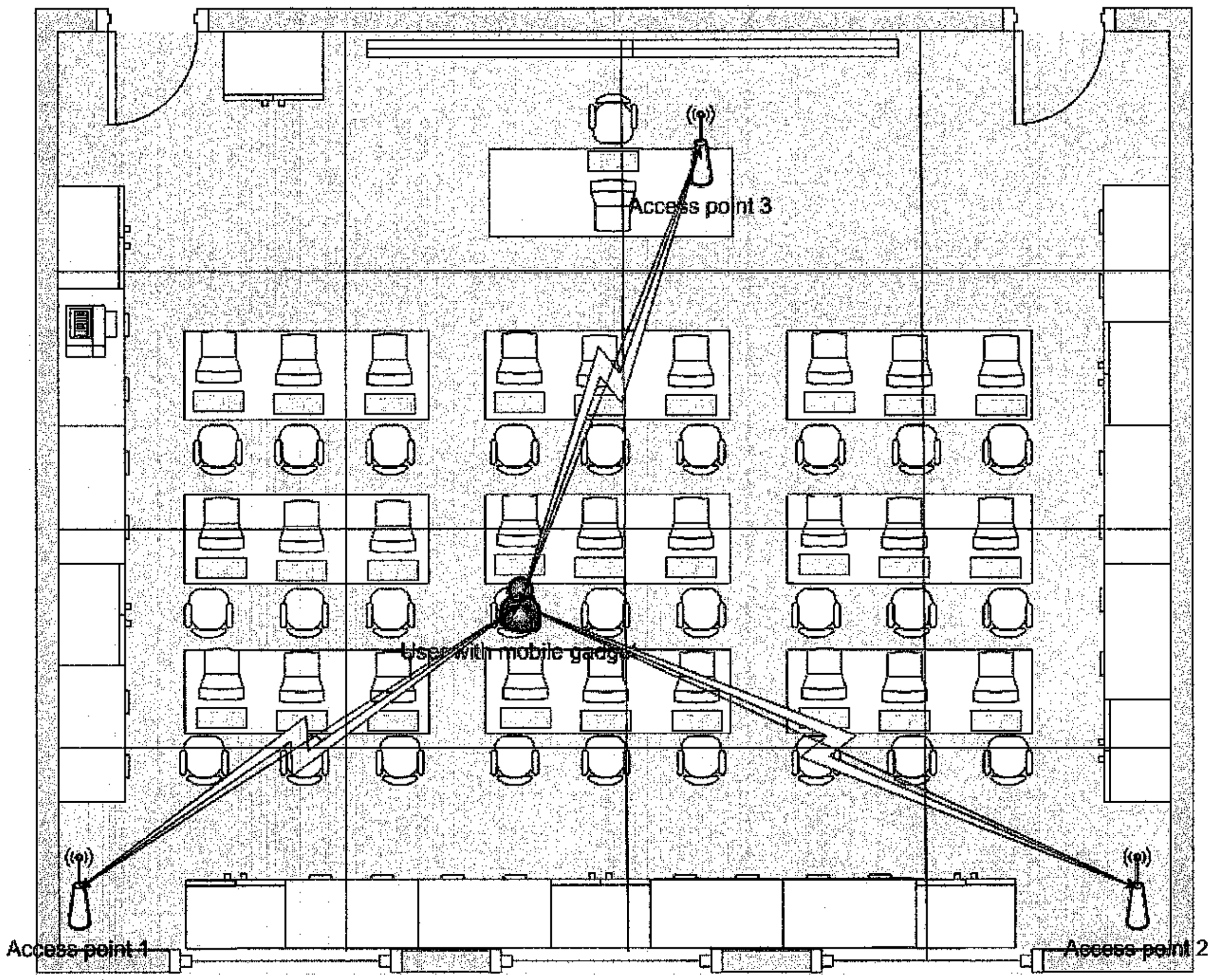


Figure 13 - Multilateration positioning Experiment

As you can see, the implementation of this project takes a hybrid solution. It uses the empirical model for WLAN positioning technique and mixes it with the multilateration technique

3.5.2 System security

When building an HTTP file upload system, one important thing that has to be considered is security, as improper design will make the system vulnerable to attacks. Therefore, various security mechanisms have been applied to the system so that it will become more reliable and secure. Such security mechanisms include the following:

- All information provided by the user is checked to ensure that it is safe
- File size limits are set so that users can't upload files that are too large or too small
- The application was made sure not to reveal too much information to the user when the error occurs. The information revealed can help malicious users find ways to attack the system
- Details of file uploads (such as time, the client's user name, and location) are logged down. Although logs only tell us what happened, they can help us check what types of attack have been made against the server and whether there were any successful attacks.
- Login and moderation is applied. This practice reduces deviant behaviors
- File types are restricted
- Uploaded files are renamed. In doing so the system checks for double-barreld extensions like "yourprofile.php.gif" and eliminates extensions that the system doesn't allow, or simply remove the file completely.

A developer implementing file upload functionality has to be careful not to expose the application to attack. In the worst case, a badly implemented files upload leads to remote code execution vulnerabilities.

CHAPTER 4

CONCLUSION AND RECOMMENDATIONS

4.1 Conclusion

This project faces many challenges which range from networking challenges to human-computer interaction and database development.

The current Technology environment provides tool to accomplish the project's objectives even though each of them has their own drawbacks and restrictions. Managing such drawbacks and restrictions it's the ultimate challenge. These tools also simplify the development of location-based applications.

In this paper the basic approaches to development of this project have been exposed. But, I do understand that there's still much information that needs to be gathered and analyzed in order to successfully accomplish the project's objectives.

With a successful development of this project tourists will definitely have a different experience in traveling.

4.2 Recommendations

Currently this project has various limitations, therefore, on the basis of such limitations, I recommend the following possible improvements to be implemented in this project:

1. Add the ability for the system to determine a wireless hub's IP address so that various wireless hub's could be utilized without the need to pre-record their IP addresses
2. Make the application cable of determining the user's location in a three dimensional perspective
3. Add the ability for user to rate the files, as well as locations so that they can share they opinions regarding the files.
4. Add the ability for user to comment on someone else's uploaded files. This will enhance the user's experience towards knowledge sharing.
5. Improve system performance when uploading files.
6. Improve the system security

By following the above recommendations, this project can further assist tourist and other kinds of user to discover the world how they want, and share they experience and knowledge on specific locations.

REFERENCES

1. 4 Guys from Rolla.com, How to ping using ASP. Retrieved 16 September 2007 from world wide web: http://www.4guysfromrolla.com/webtech/102998-1.shtml
2. 999 tutorials, Creating thumbnail with PHP. Retrieved 2 september, 2007 from world wide web: http://www.999tutorials.com/tutorial-create-thumbnail-with-php.html
3. Deitel ®, Internet World Wide Web How To Program, Third Edition
4. Fall, K., Varadhan, K., The *ns* Manual (formerly *ns* Notes and Documentation), May 2, 2007
5. Girardin, F., Building a mobile locative, and collaborative application, February, 2005
6. Lenihan, N., WLAN Positioning
7. Locher, T., Positioning with LAN, July 16, 2004
8. Maslakowski, Mark, and B. Tony. 2000. Sams teach yourself MySQL in 21 days. Sams Publishing
9. O'Reilly Network, Creating MyTube with FLEX and PHP. Retrieved 3 August, 2007 from the world wide web: http://www.onlamp.com/pub/a/php/2007/05/24/creating-mytube-with-flex-and-php.html?page=1
10. Pfeiffer, William S., Technical writing: a pratical approach. 5th ed. Prentice Hall
11. SCANIT The security company, B. Alla, Secure file upload in PHP web applications. Retrieved 26 July, 2007 from world wide web: http://www.scanit.be/uploads/php-file-upload.pdf
12. Tutorialized, PHP Sessions. Retrieved 25 August, from the world wide web: http://www.tutorialized.com/view/tutorial/PHP-SESSIONS/26973
13. Welling, Luke, and T. Laura. 2001. PHP and MySQL web development. Sams Publishing
14. Wikipedia, the free encyclopedia. GSM localization. Retrieved 15 August, 2007 from the world wide web: http://en.wikipedia.org/wiki/GSM_localization

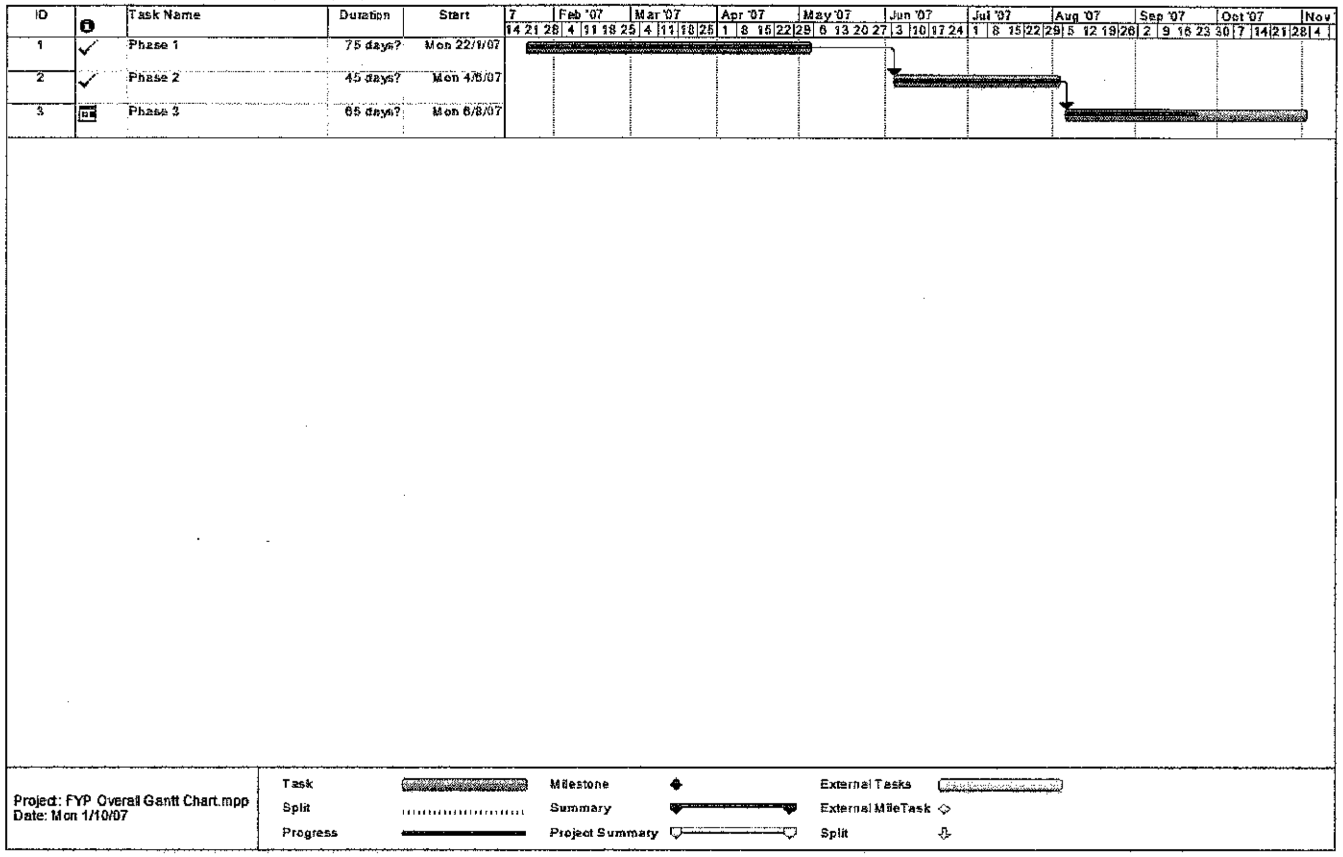
15. Wikipedia, the free encyclopedia. Multilateration. Retrieved 15 August, 2007 from the world wide web:
http://en.wikipedia.org/wiki/Time_difference_of_arrival

16. Wu, S., Wu, K., "Effective Location Based Services with Dynamic Data Management in Mobile Environments", December 30, 200

APPENDICES

(This page is intended to be empty. Please refer to the next page)

A-1 Overall Project Gantt Chart

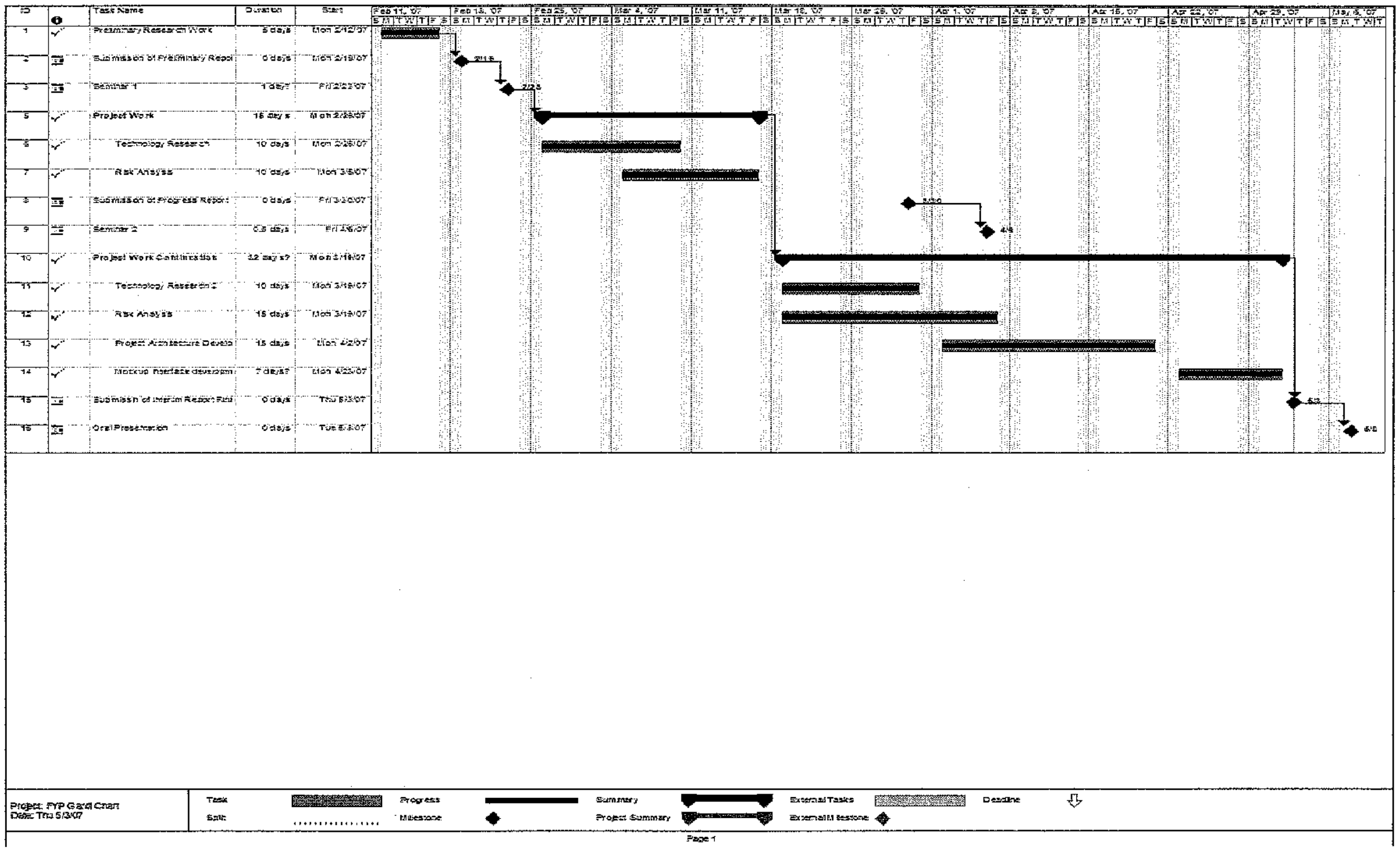


ID		Task Name	Duration	Start
1	✓	Phase 1	75 days?	Mon 22/1/07
2	✓	Phase 2	45 days?	Mon 4/6/07
3		Phase 3	65 days?	Mon 6/8/07

Project: FYP Overall Gantt Chart.mpp
Date: Mon 1/10/07

Figure 14 - Overall Project Gantt Chart

A-2 PHASE 1 GANTT CHART



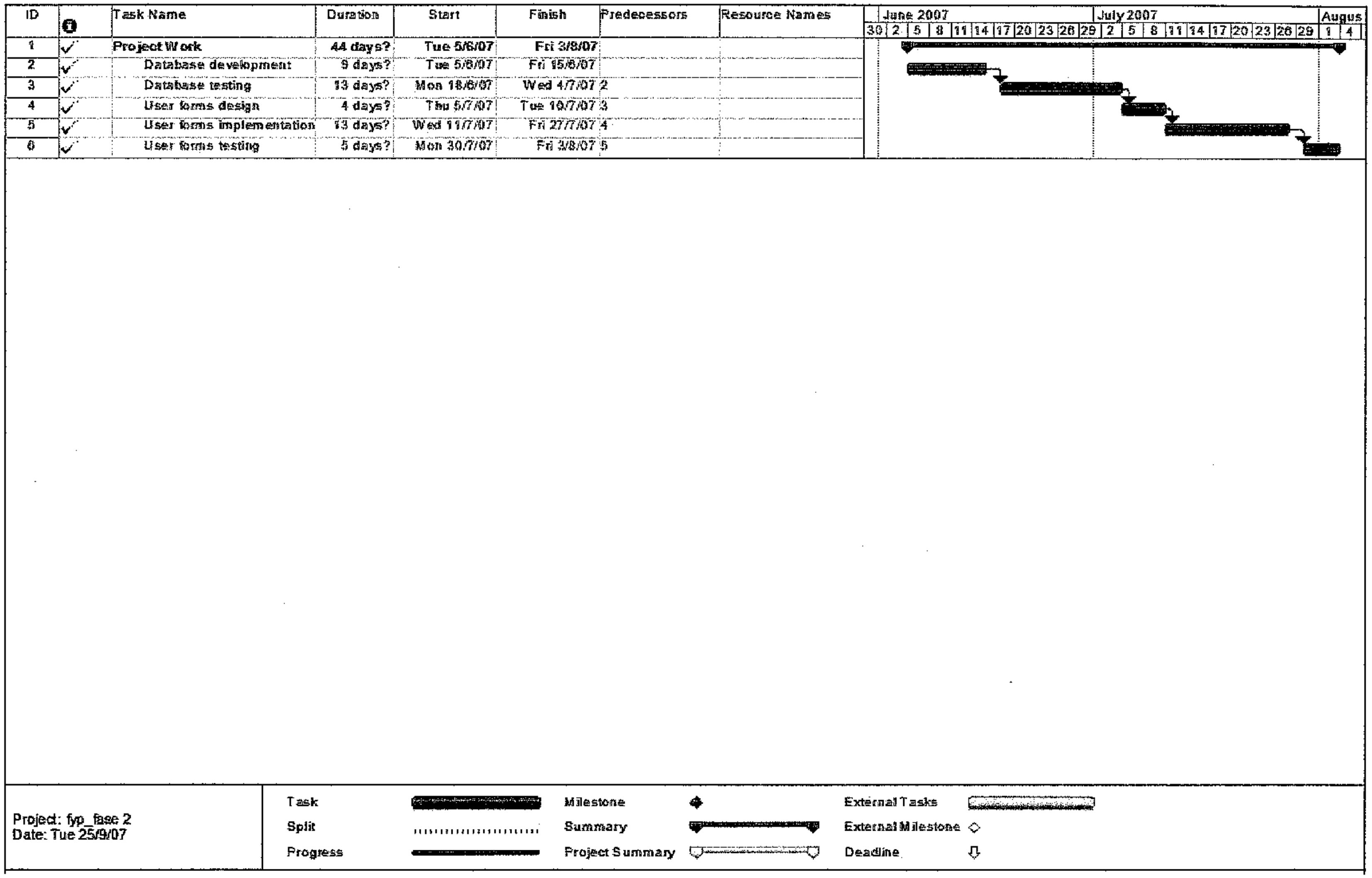
ID		Task Name	Duration	Start
1	✓	Preliminary Research Work	5 days	Mon 2/12/07
2	▤	Submission of Preliminary Repor	0 days	Mon 2/19/07
3	▤	Seminar 1	1 day?	Fri 2/23/07
5	✓	Project Work	16 days	Mon 2/26/07
6	✓	Technology Research	10 days	Mon 2/26/07
7	✓	Risk Analysis	10 days	Mon 3/5/07
8	▤	Submission of Progress Report	0 days	Fri 3/30/07
9	▤	Seminar 2	0.5 days	Fri 4/6/07
10	✓	Project Work Continuation	32 days?	Mon 3/19/07
11	✓	Technology Research 2	10 days	Mon 3/19/07
12	✓	Risk Analysis	15 days	Mon 3/19/07
13	✓	Project Architecture Develo	15 days	Mon 4/2/07
14	✓	Mockup Interface developm	7 days?	Mon 4/23/07
15	▤	Submission of Interim Report Fin	0 days	Thu 5/3/07
16	▤	Oral Presentation	0 days	Tue 5/8/07

Project: FYP Gantt Chart
Date: Thu 5/3/07

Page 1

Figure 15 - Phase 1 Gantt Char

A-3 PHASE 2 GANTT CHART



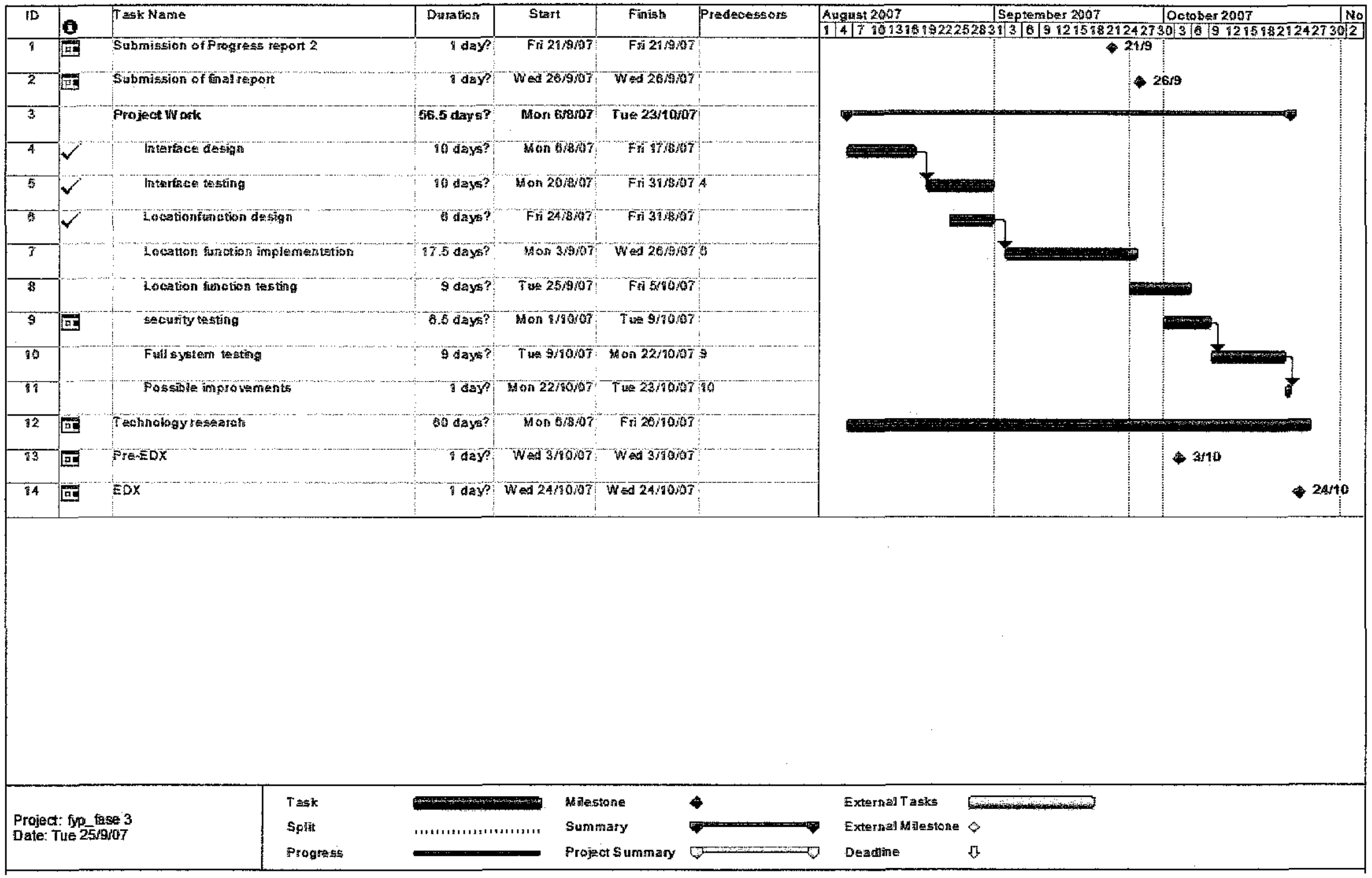
ID	❶	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1	✓	**Project Work**	**44 days?**	**Tue 5/6/07**	**Fri 3/8/07**		
2	✓	Database development	9 days?	Tue 5/6/07	Fri 15/6/07		
3	✓	Database testing	13 days?	Mon 18/6/07	Wed 4/7/07	2	
4	✓	User forms design	4 days?	Thu 5/7/07	Tue 10/7/07	3	
5	✓	User forms implementation	13 days?	Wed 11/7/07	Fri 27/7/07	4	
6	✓	User forms testing	5 days?	Mon 30/7/07	Fri 3/8/07	5	

Project: fyp_fase 2
Date: Tue 25/9/07

Task	▬	Milestone	◆	External Tasks	▭
Split	┆┆┆┆	Summary	▬	External Milestone	◇
Progress	━	Project Summary	▭	Deadline	⇩

Figure 16 - Phase 2 Gantt chart

A – 4 PHASE 3 GANTT CHART



ID		Task Name	Duration	Start	Finish	Predecessors
1		Submission of Progress report 2	1 day?	Fri 21/9/07	Fri 21/9/07	
2		Submission of final report	1 day?	Wed 26/9/07	Wed 26/9/07	
3		**Project Work**	**56.5 days?**	**Mon 6/8/07**	**Tue 23/10/07**	
4	✓	Interface design	10 days?	Mon 6/8/07	Fri 17/8/07	
5	✓	Interface testing	10 days?	Mon 20/8/07	Fri 31/8/07	4
6	✓	Locationfunction design	6 days?	Fri 24/8/07	Fri 31/8/07	
7		Location function implementation	17.5 days?	Mon 3/9/07	Wed 26/9/07	6
8		Location function testing	9 days?	Tue 25/9/07	Fri 5/10/07	
9		security testing	6.5 days?	Mon 1/10/07	Tue 9/10/07	
10		Full system testing	9 days?	Tue 9/10/07	Mon 22/10/07	9
11		Possible improvements	1 day?	Mon 22/10/07	Tue 23/10/07	10
12		Technology research	60 days?	Mon 6/8/07	Fri 26/10/07	
13		Pre-EDX	1 day?	Wed 3/10/07	Wed 3/10/07	
14		EDX	1 day?	Wed 24/10/07	Wed 24/10/07	

Project: fyp_fase 3
Date: Tue 25/9/07

Task		Milestone	◆	External Tasks	
Split	....	Summary		External Milestone	◇
Progress		Project Summary		Deadline	⇩

Figure 17 - Phase 3 Gantt chart

A – 5 Compute user's position flow chart

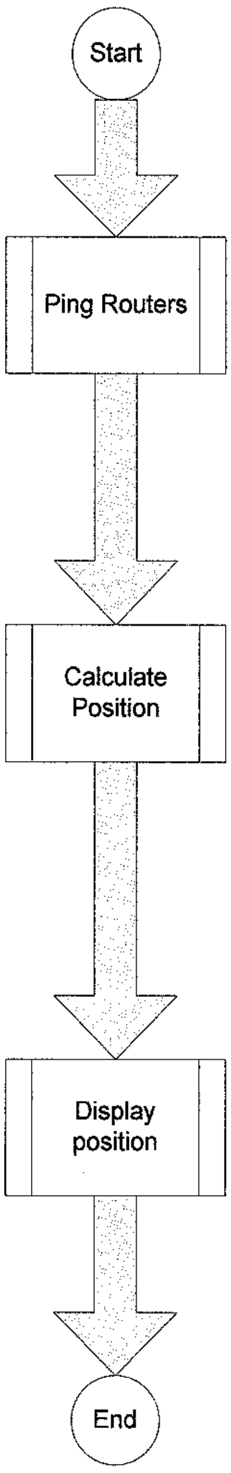


Figure 18 - User positioning flowchart

A – 6 Ping router flowchart

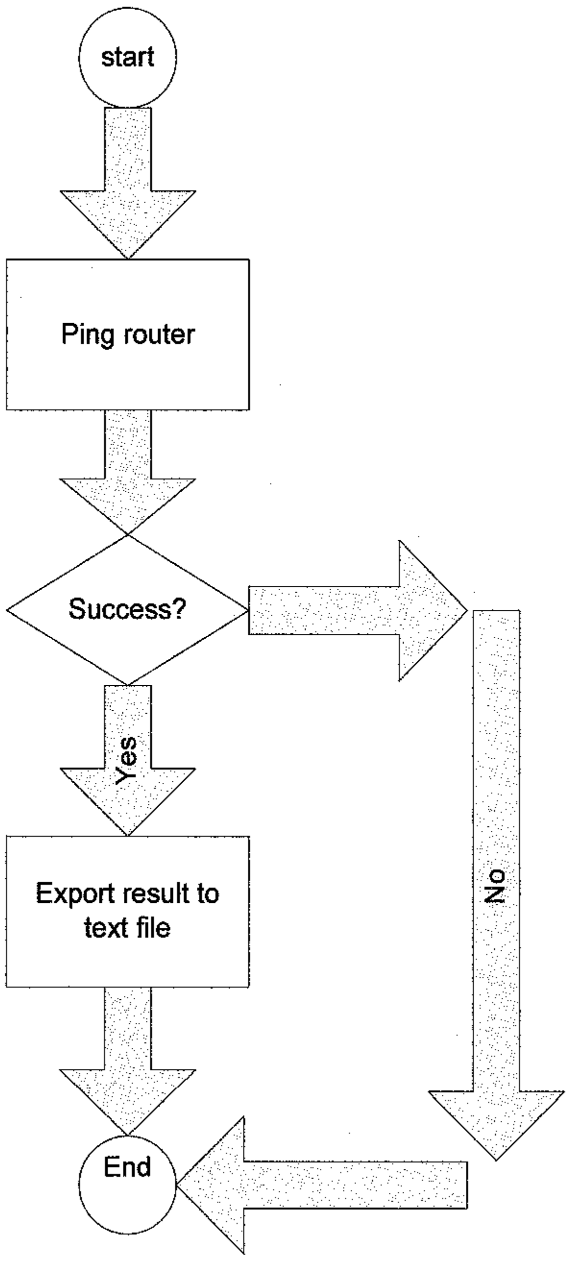


Figure 19 - Ping router flowchart

A – 7 Calculate position flowchart

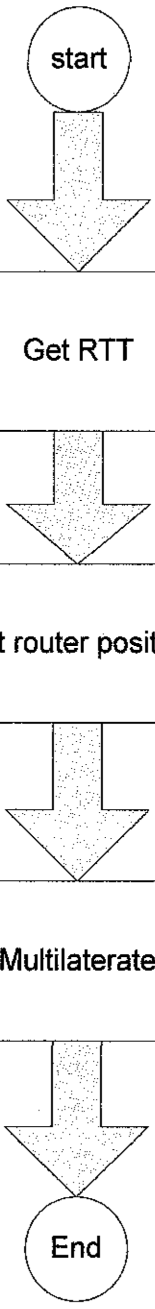


Figure 20 - calculate position flowchart

A – 8 Display position flowchart

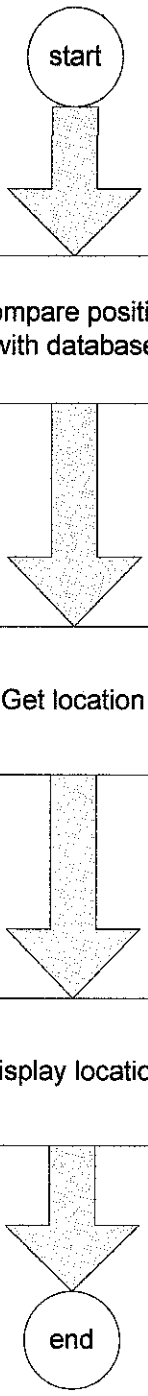


Figure 21 - Display position flowchart

A – 9 Router ping source code

```
<?php

// Function to check response time
function pingDomain($domain){

   $status  = 0;
         $average = 0;
         for($i = 0; $i < 4; $i++)
         {
                  $starttime = microtime(true);
                  $file     = fsockopen ($domain, 80, $errno, $errstr, 10);
                  $stoptime  = microtime(true);


                  if (!$file) $status = -1;  // Site is down
                  else {
                           fclose($file);
                           $status = ($stoptime - $starttime) * 1000;
                           $status = floor($status);
                           $time_arr[$i] = $status;
                  }
         }

         for($i = 0; $i < 4; $i++)
         {
                  $average += $time_arr[$i];
         }

         $average = $average / 4;

   return $average;
}
$my_domain = "www.utp.edu.my" //or any domain that you wish to ping;

$result = pingDomain($my_domain);

echo "Average RTT for ".$my_domain." = $result ms";
?>
```

A – 10 Function display_related_media() source code

```
        function display_related_media($location){
                extract ($_POST);

                //connect to database
                $conn = db_connect();
                if (!$conn){
                        return "Could not connect to database server - please try again
later.";
                }

                //get locationID
                $sql = "select locationID from tbl_place where name = '$location' limit 0,
1";
                $locations = mysql_query($sql);

                if($locations) {
                        $row = mysql_fetch_array($locations);
                        $locationID = $row['locationID'];
                        echo $locationID[0];
                }
                else{
                        echo "Could not retrieve locationID<br />";
                }

                //get images
                $query = "select tbl_images.title, tbl_images.thumb, tbl_place.name from
tbl_images, tbl_place where tbl_images.locationID = '$locationID' and
tbl_place.locationID = '$locationID'";
                $results = mysql_query($query);
                if(!$results){
                        echo "Unable to retrieve images";
                        exit;
                }

                //display images

                //get number of rows retrieved
                $num_rows = mysql_num_rows($results);

                echo "<p><h2> Images </h2><p>";

                if ($num_rows == 0){
                        echo "No images were found at this location";
                        do_html_url("upload_form_images.php", "image upload");
                }
```

```
		else {
			echo "<p> Number of uploaded images found at this location:
".$num_rows."</p>";

			for ($count = 0; $count < $num_rows; $count++){
				$row = mysql_fetch_array($results);
				//echo $row['thumb'];
				echo "<p><img src=\"".$row['thumb']."\"></p>";
				echo "<p><strong>".($count +1).". Title: </strong>";
				echo htmlspecialchars (stripslashes($row["title"]));
				echo "<br><strong>Location: </strong>";
				echo htmlspecialchars (stripslashes($row["name"]));
				echo "<br>";

			}
			//do_html_url("upload_form_images.php", "image upload");
		}


		//get videos
		$query2 = "select tbl_movies.title, tbl_movies.thumb, tbl_place.name
from tbl_movies, tbl_place where tbl_movies.locationID = '$locationID' and
tbl_place.locationID = '$locationID'";
		$results2 = mysql_query($query2);
		if(!$results2){
			echo "Unable to retrieve videos";
			exit;
		}
		//display videos

		//get number of rows retrieved
		$num_rows = mysql_num_rows($results2);

		echo "<p><h2> Videos </h2><p>";

		if ($num_rows == 0){
			echo "No videos were found at this location";
			do_html_url("upload_form_videos.php", "video upload");
		}
		else {
			echo "<p> Number of uploaded videos found at this location:
".$num_rows."</p>";

			for ($count = 0; $count < $num_rows; $count++){
				$row = mysql_fetch_array($results2);
				echo "<p><img src=\"".$row['thumb']."\"></p>";
				echo "<p><strong>".($count +1).". Title: </strong>";
```

```
                        echo htmlspecialchars (stripslashes($row["title"]));
                        echo "<br><strong>Location: </strong>";
                        echo htmlspecialchars (stripslashes($row["name"]));

                }
                //do_html_url("upload_form_videos.php", "video upload");
        }


        //get songs
        $query3 = "select tbl_songs.title, tbl_songs.author, tbl_place.name from
tbl_songs, tbl_place where tbl_songs.locationID = '$locationID' and tbl_place.locationID
= '$locationID'";
        $results3 = mysql_query($query3);
        if(!$results3){
                echo "Unable to retrieve songs";
                exit;
        }
        //display songs
        //get number of rows retrieved
        $num_rows = mysql_num_rows($results3);

        echo "<p><h2> Songs </h2><p>";

        if ($num_rows == 0){
                echo "No songs were found at this location";
                do_html_url("upload_form_songs.php", "audio file upload");
        }
        else {
                echo "<p> Nember of uploaded audio files found at this location:
".$num_rows."</p>";

                for ($count = 0; $count < $num_rows; $count++){
                        $row = mysql_fetch_array($results3);
                        //echo "<p><img src=\"".$row["thumb"]."\"></p>";
                        echo "<p><strong>".($count +1).". Title: </strong>";
                        echo htmlspecialchars (stripslashes($row["title"]));
                        echo "-";
                        echo htmlspecialchars (stripslashes($row["author"]));
                        echo "<br><strong>Location: </strong>";
                        echo htmlspecialchars (stripslashes($row["name"]));

                }
        }


    }
```

A – 11 Function uploadImage() source code

```
function uploadImage()
        {
                echo "uploadImage() function was called<br />";

                extract ($_POST);

                //connect to database
                $conn = db_connect();
                if (!$conn){
                        return "Could not connect to database server - please try again
later.";
                }

                //get locationID
                $sql = "select locationID from tbl_place where name = '$location' limit 0,
1";
                $locations = mysql_query($sql);

                if($locations) {
                        $row = mysql_fetch_array($locations);
                        $locationID = $row['locationID'];
                        echo $locationID[0];
                }
                else{
                        echo "Could not retrieve locationID<br />";
                }

                //get userID
                $sql2 = "select userID from tbl_user where username =
'".$_SESSION["username"]."' limit 0, 1";
                $result2 = mysql_query($sql2);

                if($result2) {
                        $row = mysql_fetch_array($result2);
                        $userID = $row['userID'];

                        echo $userID[0];
                }
                else {
                        echo "could not retrieve userID<br />";
                }

                //check that we have an image title
                if (!isset($title))
                {
                        do_html_header("Error:");
                        echo    "You must enter a title to your image - please go back "
                                ."and try again.";

                        do_html_footer();

                        exit;
                }

                //check that we have a file

                if((!empty($_FILES['uploaded_file'])) &&
($_FILES['uploaded_file']['error'] == 0)) {

                  //Check if the file is JPEG image and it's size is less than 2.0Mb
                  $filename = basename($_FILES['uploaded_file']['name']);
                  $ext = substr($filename, strrpos($filename, '.') + 1);

                  if (($ext == "jpg") || ($ext == "gif") || ($ext == "png") &&
($_FILES['uploaded_file']['type'] == "image/jpeg") || ($_FILES['uploaded_file']['type']
== "image/pjpeg")  ||  ($_FILES['uploaded_file']['type'] == "image/gif") ||
($_FILES['uploaded_file']['type'] == "image/png")&&
                        ($_FILES['uploaded_file']['size'] < 2000000)) {

                        //Determine the path to which we want to save this file and the
thumbnail
                          $filename2 = preg_replace( '/\..*$/', '', basename( $filename ) );
                          $tempname = "Media\\Images\\temp\\".$filename;
                          $thumbpath =
"Media/Images/thumbs/".$filename2."_".$title."_".$userID.".jpg";
                          $source =
"Media/Images/".$filename2."_".$title."_".$userID.".jpg";
                          $newname = "Media\\Images\\".$filename;
                          $newname2 =
"Media\\Images\\".$filename2."_".$title."_".$userID.".jpg";

                          //echo $tempname;
                          //Check if the file with the same name already exists on the
server

                          if (!file_exists($newname2)) {

                                //Attempt to move the uploaded file to it's new place
                                if
((move_uploaded_file($_FILES['uploaded_file']['tmp_name'],$tempname))) {

                                        //resize image
                                        if((resizeImage("Media\\images\\temp", $filename,
"Media\\images\\", $title, 600))){


                                                //set number of views to 0
                                                $views = 0;

                                                //insert values into the database
                                                $query = "insert into tbl_images
values(NULL, '$title', '$source', '$thumbpath', '$comments', NULL, '$views', '$userID',
'$locationID')";
                                                 $result = mysql_query($query);

                                                 if(!$result) {
                                                        echo "Error: could not upload
values to the database";
                                                 }
                                                 else {
                                                 //create image thumbnail
                                                        if
((createThumbnail("Media\\Images", $filename, "Media\\Images\\thumbs", $title,
150))){
                                                                echo "It's done! The file has
been saved as: ".$title.".jpg";
                                                }
                                                else {
                                                        echo "Error: A problem occurred
during image thumbnail creation";
                                                }
                                        }
                                }
                                        else {
                                                echo "Error: A problem occurred during
image resizing";
                                }
                                }
                            else {
                                 echo "Error: A problem occurred during file upload!";

                                }
                        } else {
                                echo "Error: File ".$_FILES["uploaded_file"]["name"]."
already exists";

                        }
                } else {
```

```
                                echo "Error: Only .jpg, .gif, and .png images under 2.0Mb are
accepted for upload";

                   }
                  }else {
                   echo "Error: No file uploaded";

                  }

        }
```

A – 12 Function uploadVideo() source code

```
function uploadVideo()
        {
                echo "uploadVideo() function was called<br />";

                extract ($_POST);

                //connect to database
                $conn = db_connect();
                if (!$conn){
                        return "Could not connect to database server - please try again
later.";
                }

                //get userID
                $sql2 = "select userID from tbl_user where username =
'".$_SESSION["username"]."' limit 0, 1";
                $result2 = mysql_query($sql2);

                if($result2) {
                        $row = mysql_fetch_array($result2);
                        $userID = $row['userID'];

                        echo $userID[0];
                }
                else {
                        echo "could not retrieve userID<br />";
                }

                //check that we have an video title
                if (!isset($title))
                {
                        do_html_header("Error:");
                        echo    "You must enter a title to your video - please go back "
                                ."and try again.";

                        do_html_footer();

                        exit;
                }

                //check that we have a file
                if((!empty($_FILES['uploaded_file'])) &&
($_FILES['uploaded_file']['error'] == 0)) {

                 //Check if the file is a video and its size is less than 20.0Mb
```

```
                    $filename = basename($_FILES['uploaded_file']['name']);
                    $ext = substr($filename, strrpos($filename, '.') + 1);

                    if ( ($ext == 'mpg') || ($ext == 'avi') || ($ext == 'mov') || ($ext == 'flv')
&& ($_FILES['uploaded_file']['size'] < 100000000)) {

                              $filename = str_replace(" ", "_", $filename);
                              $ufilename = str_replace(".".$ext, "", $filename);
                              $ufilename = str_replace(".", "", $ufilename);

                            //Determine the path to which we want to save this file and the
thumbnail
                              $flvname =
"Media\\Videos\\".$ufilename."_".$title."_".$userID.".flv";
                              //$thumb = "Media/Videos/thumbs/".preg_replace( '/\..*$/', '',
basename( $filename ) )."%d.jpg";
                              //$thumbpath = "Media\\Videos\\thumbs\\".preg_replace(
'/\..*$/', '', basename( $filename ) )."%d.jpg";
                              //$source = "Media/Videos/".preg_replace( '/\..*$/', '', basename(
$filename ) ).".flv";
                              $newname = "Media\\Videos\\".$filename;

                              //echo $tempname;
                              //Check if the file with the same name already exists on the
server
                              if (!file_exists($flvname)) {

                                      //copy files to the server
                               move_uploaded_file($_FILES['uploaded_file']['tmp_name'],
$newname);

                                      //Attempt to move the uploaded file to it's new place
                                      if ((flv_import($newname ,$ufilename, $title, $comments,
$location, $ext))) {
                                                      echo "It's done! The file has been saved as:
".$title;
                                              }
                                               else {
                                                echo "Error: A problem occurred during file type
convertion!";

                                              }

                                      } else {
                                      echo "Error: File ".$_FILES["uploaded_file"]["name"]."
already exists";
```

```
            }
        } else {
            echo "Error: Only videos under 40.0Mb are accepted for upload";

        }
       } else {
        echo "Error: No file uploaded";

       }
}
```

A – 13 function uploadSong() source code

```
	function uploadSong()
	{
		echo "uploadSong() function was called<br />";

		extract ($_POST);

		//connect to database
		$conn = db_connect();
		if (!$conn){
			return "Could not connect to database server - please try again
later.";
		}

		//get locationID
		$sql = "select locationID from tbl_place where name = '$location' limit 0,
1";
		$locations = mysql_query($sql);

		if($locations) {
			$row = mysql_fetch_array($locations);
			$locationID = $row['locationID'];
			echo $locationID[0];
		}
		else{
			echo "Could not retrieve locationID<br />";
		}

		//get userID
		$sql2 = "select userID from tbl_user where username =
'".$_SESSION["username"]."' limit 0, 1";
		$result2 = mysql_query($sql2);

		if($result2) {
			$row = mysql_fetch_array($result2);
			$userID = $row['userID'];

			echo $userID[0];
		}
		else {
			echo "could not retrieve userID<br />";
		}

		//check that we have an song title
		if (!isset($title))
		{
```

```
                    do_html_header("Error:");
                    echo    "You must enter a title to your song - please go back "
                            ."and try again.";

                    do_html_footer();

                    exit;
                }

                //check that we have a file
                if((!empty($_FILES['uploaded_file'])) &&
($_FILES['uploaded_file']['error'] == 0)) {

                 //Check if the file is a video and its size is less than 5.0Mb
                 $filename = basename($_FILES['uploaded_file']['name']);
                 $ext = substr($filename, strrpos($filename, '.') + 1);

                 if ( ($ext == 'mp3') || ($ext == 'wma') || ($ext == 'wav') &&
($_FILES['uploaded_file']['size'] < 5000000)) {

                    //Determine the path to which we want to save this file and the
thumbnail
                     $songname = "Media\\Songs\\".preg_replace( '/\..*$/', '',
basename( $filename ) )."_".$title."_".$userID.".".$ext;
                     //$thumb = "Media/Videos/thumbs/".preg_replace( '/\..*$/', '',
basename( $filename ) )."%d.jpg";
                     //$thumbpath = "Media\\Videos\\thumbs\\".preg_replace(
'/\..*$/', '', basename( $filename ) )."%d.jpg";
                     $source = "Media/Songs/".preg_replace( '/\..*$/', '', basename(
$filename ) )."_".$title."_".$userID.".".$ext;
                     //$newname = "Media\\Songs\\".$filename;

                     //echo $tempname;
                     //Check if the file with the same name already exists on the
server
                     if (!file_exists($songname)) {

                            //copy files to the server
                            $views = 0;

                            $query = "insert into tbl_songs values (NULL, '$title',
'$source', '$comments', NULL, '$views', '$userID', '$locationID', '$author')";

                            //echo $query;
                     $result = mysql_query($query);
                     if(!$result) {
                            echo "Unable to insert values to database<br>";
```

```
                              }
                              else {

                                        //Attempt to move the uploaded file to it's new place
                                        if
((move_uploaded_file($_FILES['uploaded_file']['tmp_name'], $songname))) {
                                                            echo "It's done! The file has been saved as:
".$author."-".$title;
                                                  }
                                                    else {
                                                     echo "Error: A problem occurred during file
uploading";

                                                  }
                                        }

                                         } else {
                                         echo "Error: File ".$_FILES["uploaded_file"]["name"]."
already exists";

                              }
                    }       else {
                              echo "Error: Only  audio (.mp3, .wma, and .wav) files under
4.0Mb are accepted for upload";

                    }
                   } else {
                    echo "Error: No file uploaded";

                   }
         }
```